OpenVMS Version 6.2 New Features Manual

Order Number: AA–QJEEA–TE

May 1995

This manual describes the new features of the OpenVMS VAX Version 6.2 and the OpenVMS Alpha Version 6.2 operating systems.

Revision/Update Information: This is a new manual.

Software Version: OpenVMS Alpha Version 6.2
OpenVMS VAX Version 6.2

Digital Equipment Corporation
Maynard, Massachusetts

May 1995

Digital Equipment Corporation makes no representations that the use of its products in the manner described in this publication will not infringe on existing or future patent rights, nor do the descriptions contained in this publication imply the granting of licenses to make, use, or sell equipment or software in accordance with the description.

Possession, use, or copying of the software described in this publication is authorized only pursuant to a valid written license from Digital or an authorized sublicensor.

Digital conducts its business in a manner that conserves the environment and protects the safety and health of its employees, customers, and the community.

© Digital Equipment Corporation 1995. All rights reserved.

The following are trademarks of Digital Equipment Corporation: AlphaServer, AlphaStation, AXP, Bookreader, CI, DECchip, DECmcc, DECmigrate, DECnet, DECnet/OSI, DECserver, DECwindows, Digital, HSC, HSJ, HSZ, InfoServer, LAT, MicroVAX, MSCP, OpenVMS, PATHWORKS, POLYCENTER, StorageWorks, ThinWire, TMSCP, TURBOchannel, ULTRIX, VAX, VAX DOCUMENT, VAXcluster, VMS, VMScluster, VT, and the DIGITAL logo.

The following are third-party trademarks:

Adaptec is a trademark of Adaptec, Inc.

Futurebus/Plus is a registered trademark of Force Computers GMBH, Germany.

IBM is a registered trademark of International Business Machines Corporation.

IEEE is a registered trademark and POSIX is a registered certification mark of the Institute of Electrical and Electronics Engineers.

Internet is a registered trademark of Internet, Inc.

Microsoft, MS, and MS–DOS are registered trademarks and Windows is a trademark of Microsoft Corporation.

NetView is a registered trademark of International Business Machines Corporation.

Proteon is a trademark of Proteon Associates, Inc.

UNIX is a registered trademark of Unix System Laboratories, Inc. a wholly-owned subsidiary of Novell, Inc.

All other trademarks and registered trademarks are the property of their respective holders.

ZK6442

This document is available on CD–ROM.

This document was prepared using VAX DOCUMENT Version 2.1.

Send Us Your Comments

We welcome your comments on this or any other OpenVMS manual. If you have suggestions for improving a particular section or find any errors, please indicate the title, order number, chapter, section, and page number (if available). We also welcome more general comments. Your input is valuable in improving future releases of our documentation.

You can send comments to us in the following ways:

- Internet electronic mail: `openvmsdoc@zko.mts.dec.com`
- Fax: `603-881-0120 Attn: OpenVMS Documentation, ZKO3-4/U08`
- Online form

  Print or edit the online form SYS$HELP:OPENVMSDOC_COMMENTS.TXT. Send the completed online form by electronic mail to our Internet address, or send the completed hardcopy form by fax or through the postal service.

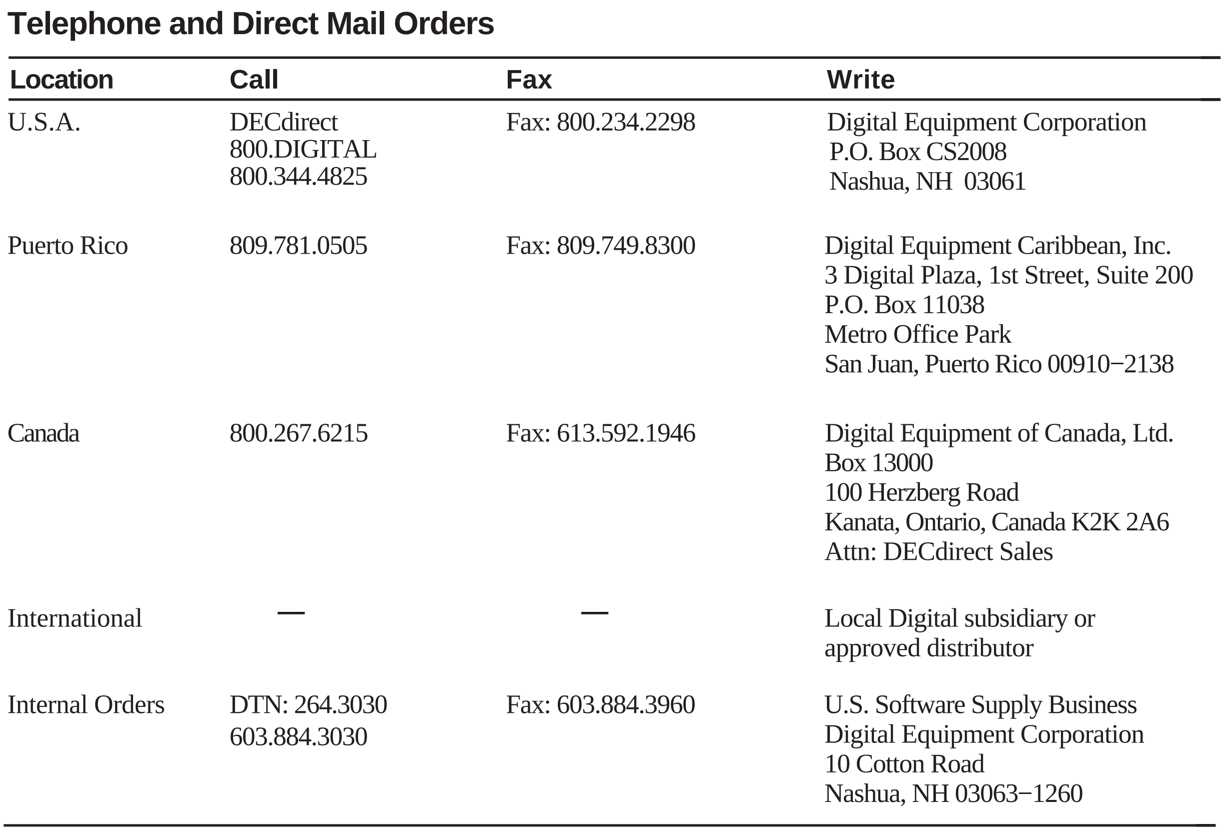
  Please send letters or the form to:

  ```
  Digital Equipment Corporation
  Information Design and Consulting
  OpenVMS Documentation
  110 Spit Brook Road, ZKO3-4/U08
  Nashua, NH  03062-2698
  USA
  ```

Thank you.

How To Order Additional Documentation

Use the following table to order additional documentation or information. If you need help deciding which documentation best meets your needs, call 800-DIGITAL (800-344-4825).



Telephone and Direct Mail Orders

Location	Call	Fax	Write
U.S.A.	DECdirect 800.DIGITAL 800.344.4825	Fax: 800.234.2298	Digital Equipment Corporation P.O. Box CS2008 Nashua, NH 03061
Puerto Rico	809.781.0505	Fax: 809.749.8300	Digital Equipment Caribbean, Inc. 3 Digital Plaza, 1st Street, Suite 200 P.O. Box 11038 Metro Office Park San Juan, Puerto Rico 00910−2138
Canada	800.267.6215	Fax: 613.592.1946	Digital Equipment of Canada, Ltd. Box 13000 100 Herzberg Road Kanata, Ontario, Canada K2K 2A6 Attn: DECdirect Sales
International	—	—	Local Digital subsidiary or approved distributor
Internal Orders	DTN: 264.3030 603.884.3030	Fax: 603.884.3960	U.S. Software Supply Business Digital Equipment Corporation 10 Cotton Road Nashua, NH 03063−1260

ZK−7654A−GE

Contents

4 ProgrammingFeatures

5 DEC C XPG4 Localization Utilities

6 Local Area Network (LAN) Management Enhancements

7 SCSI as a VMScluster Storage Interconnect

8 VMScluster Systems That Span Multiple Sites

A New OpenVMS System Messages

Index

Examples

Figures

Tables

Preface

Intended Audience

This manual is intended for general users, system managers, and programmers who use the OpenVMS operating system.

This document contains descriptions of the new features for Version 6.2 of the OpenVMS VAX and OpenVMS Alpha operating systems. For information about how some of the new features might affect your system, read the *OpenVMS Version 6.2 Release Notes* before you install, upgrade, or use Version 6.2.

Document Structure

This manual is organized as follows:

- Chapter 1 contains a summary of the new OpenVMS Version 6.2 software features.
- Chapter 2 describes new features of interest to general users of the OpenVMS VAX and OpenVMS Alpha Version 6.2 operating systems.
- Chapter 3 describes new features that are applicable to the tasks performed by system managers.
- Chapter 4 describes new features that support programming tasks.
- Chapter 5 describes the new DEC C XPG4 localization utilities.
- Chapter 6 describes local area network (LAN) management enhancements.
- Chapter 7 describes how VMScluster systems support the Small Computer Systems Interface (SCSI) as a storage interconnect.
- Chapter 8 discusses multiple-site VMScluster configurations.
- Appendix A describes new or changed messages from the Help Message database.

Related Documents

Refer to the following documents for detailed information about the software features described in this manual. For more information about how to order these documents, see the *Overview of OpenVMS Documentation* or contact your Digital representative.

- *DEC C Run-Time Library Reference Manual for OpenVMS Systems*
- *DECamds User's Guide*
- *DECnet for OpenVMS Networking Manual*
- *DECnet/OSI DECdns Management*
- *Guidelines for VMScluster Configurations*

- *OpenVMS Alpha Version 6.2 Upgrade and Installation Manual*
- *OpenVMS DCL Dictionary*
- *OpenVMS Guide to System Security*
- *OpenVMS I/O User's Reference Manual*
- *OpenVMS Linker Utility Manual*
- *OpenVMS Management Station Overview and Release Notes*
- *OpenVMS RTL Library (LIB$) Manual*
- *OpenVMS System Management Utilities Reference Manual*
- *OpenVMS System Manager's Manual*
- *OpenVMS System Messages: Companion Guide for Help Message Users*
- *OpenVMS System Services Reference Manual*
- *OpenVMS User's Manual*
- *OpenVMS VAX Version 6.2 Upgrade and Installation Manual*
- *OpenVMS Version 6.2 Release Notes*
- *TCP/IP Networking on OpenVMS Systems*
- *VMScluster Systems for OpenVMS*

Conventions

The name of the OpenVMS AXP operating system has been changed to OpenVMS Alpha. Any references to OpenVMS AXP or AXP are synonymous with OpenVMS Alpha or Alpha.

The following conventions are used to identify information specific to OpenVMS Alpha or to OpenVMS VAX:

Alpha The Alpha icon denotes the beginning of information specific to OpenVMS Alpha.

VAX The VAX icon denotes the beginning of information specific to OpenVMS VAX.

♦ The diamond symbol denotes the end of a section of information specific to OpenVMS Alpha or to OpenVMS VAX.

In this manual, every use of DECwindows and DECwindows Motif refers to DECwindows Motif for OpenVMS software.

The following conventions are also used in this manual:

Ctrl/*x* 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.

Return In examples, a key name enclosed in a box indicates that you press a key on the keyboard. (In text, a key name is not enclosed in a box.)

. . .

Horizontal ellipsis points in examples indicate one of the following possibilities:

- Additional optional arguments in a statement have been omitted.
- The preceding item or items can be repeated one or more times.
- Additional parameters, values, or other information can be entered.

.
.
.

Vertical ellipsis points indicate 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 command format descriptions, parentheses indicate that, if you choose more than one option, you must enclose the choices in parentheses.

[ ]

In command format descriptions, brackets indicate optional elements. You can choose one, none, or all of the options. (Brackets are not optional, however, in the syntax of a directory name in an OpenVMS file specification or in the syntax of a substring specification in an assignment statement.)

{ }

In command format descriptions, braces indicate a required choice of options; you must choose one of the options listed.

boldface text

Boldface text represents the introduction of a new term or the name of an argument, an attribute, or a reason.

Boldface text is also used to show user input in Bookreader versions of the manual.

italic text

Italic text indicates important information, complete titles of manuals, or variables. Variables include information that varies in system messages (Internal error *number*), in command lines (/PRODUCER=*name*), and in command parameters in text (where *device-name* contains up to five alphanumeric characters).

UPPERCASE TEXT

Uppercase text indicates a command, the name of a routine, the name of a file, or the abbreviation for a system privilege.

`struct`

Monospace type in text identifies the following C programming language elements: keywords, the names of independently compiled external functions and files, syntax summaries, and references to variables or identifiers introduced in an example.

-

A hyphen in code examples indicates that additional arguments to the request are provided on the line that follows.

numbers

All numbers in text are assumed to be decimal unless otherwise noted. Nondecimal radixes—binary, octal, or hexadecimal—are explicitly indicated.

1 Summary of Version 6.2 New Features

This chapter contains Table 1–1, which summarizes the new software features supported by OpenVMS VAX and OpenVMS Alpha Version 6.2.

Table 1–1 Summary of OpenVMS VAX and OpenVMS Alpha Version 6.2 Software Features

General User Features	
OpenVMS Freeware disc	Contains free software tools and utilities to aid software developers in creating applications, and managing and using OpenVMS systems.
Automatic foreign commands from DCL	It is now possible to tell DCL to search a specific set of directories for executable images or command procedures and have them automatically invoked without defining a DCL symbol.
DCL commands	Several DCL commands have been changed or enhanced.
Sort/Merge utility	The VALUE keyword is now available.
DCL commands for TCP/IP users	Several DCL commands have new qualifiers that make frequently used TCP/IP functions easier to access.
Specifying user addresses in Mail	Permits user addresses to be specified in the Internet format without requiring a transport name or quotes (unless the address contains reserved characters).
System Management Features	
OpenVMS Management Station	This powerful Microsoft Windows based management tool is now bundled with OpenVMS systems.
Volume Shadowing	New features include: • Support for geometry-based shadowing device recognition • Increased limits on numbers of disks allowed in shadow sets • StorageWorks RAID Array subsystem support
VMScluster systems	Includes support for SCSI interconnect, multiple-site clusters, and TMSCP served SCSI tapes.
Local area network (LAN) management enhancements	Two new LAN utilities perform system management tasks related to LAN operations.
Booting VMScluster satellite nodes	The new LANCP facility allows VMScluster systems to boot satellites without the use of DECnet and provides several features not available with DECnet MOP booting.
AUTOGEN	Changes to computation of SYSMWCNT parameter, suppression of informational messages, and overriding parameters related to DECnet.

(continued on next page)

Table 1–1 (Cont.) Summary of OpenVMS VAX and OpenVMS Alpha Version 6.2 Software Features

System Management Features	
System parameters	New parameters: • DR_UNIT_BASE • RMTDBG_SCRATCH_PAGES (Alpha systems only) • TMSCP_SERVE_ALL • VCC_PTES (VAX systems only) Changed parameters: • LGI_BRK_LIM, minimum value now 1 • SHADOWING, new values
Backup Manager	Provides a screen-oriented interface to the OpenVMS Backup utility (BACKUP).
OpenVMS BACKUP	The new /[NO]ALIAS qualifier allows users more control over how alias (synonym) file entries are processed by the OpenVMS Backup utility. BACKUP now supports up to 32 directory levels.
Full names	Naming conventions for DECnet node names correspond to the two types of DECnet functionality.
Two proxy authorization files	OpenVMS Version 6.2 offers two proxy authorization files, NET$PROXY.DAT (the primary proxy database) and NETPROXY.DAT (for use by DECnet for OpenVMS and for backward compatibility).
Security Server process	Creates and manages the system's intrusion database, and maintains the network proxy database file.
LAT	New features have been added to the SET HOST/LAT command, the LAT Control Program (LATCP) utility, and the programming environment.
LICENSE MODIFY command	The new /[NO]NO_SHARE qualifier lets you add the NO_SHARE option to a PAK registered in a license database (LDB).
Choosing languages and date/time formats	System managers can select the time and date formats for many SHOW commands from a predefined list or define new time and date formats.
OPCOM	Changes include enabling selected operator classes, overriding values for workstations, and new field names.
Invalid symbiont message codes	The invalid symbiont message errors logged to the operator console and operator log file now contain integer values that identify why the symbiont message was rejected.
Erasing old home blocks	The new /HOMEBLOCKS qualifier on the ANALYZE/DISK_STRUCTURE command enables you to erase old home blocks.
Determining volume rebuild status	You can now find out whether volumes that are mounted need rebuilding using the SHOW DEVICES/REBUILD_STATUS command.
SHUTDOWN.COM logicals	Two new logicals in the SHUTDOWN.COM command procedure allow system managers to determine the time when DECnet and queues are shut down.

(continued on next page)

Table 1–1 (Cont.) Summary of OpenVMS VAX and OpenVMS Alpha Version 6.2 Software Features

System Management Features	
Installing or upgrading from a running Alpha system	Beginning with OpenVMS Alpha Version 6.2, you can install or upgrade from a running OpenVMS Alpha system to a target system disk.
OpenVMS TCP/IP support	New OpenVMS manual describes TCP/IP networking support, including DCL TCP/IP commands.
Dump file off the system disk (VAX systems only)	On some VAX configurations, the system dump file can be placed on a device other than the system disk.
DECamds	Console now supported on OpenVMS Alpha systems. In addition, DECamds now has the ability to adjust VMScluster quorum values and supports some new customization options.
Inclusion of selected layered products on distribution kit	The OpenVMS CD–ROM distribution kit has been expanded to include many software products that previously required the purchase of additional CD–ROMs. Also, all OpenVMS online documentation will now be included on one OpenVMS documentation CD–ROM.
Programming Features	
DECnet/OSI full names	Several run-time routines and system services are provided to support DECnet/OSI full names.
LAN (Alpha systems only)	LAN programming features added to OpenVMS Alpha Version 6.2 include: • New devices and drivers • New LAN device names and devices • New controller characteristics • New procedures for configuring ISA devices
LAT	$QIO has a new LAT function modifier and a new LAT SENSEMODE argument.
System services	Intrusion database system services: • $DELETE_INTRUSION • $SCAN_INTRUSION • $SHOW_INTRUSION Proxy database system services: • $ADD_PROXY • $DELETE_PROXY • $DISPLAY_PROXY • $VERIFY_PROXY

(continued on next page)

Table 1–1 (Cont.) Summary of OpenVMS VAX and OpenVMS Alpha Version 6.2 Software Features

Programming Features	
	Cluster event notification system services: • $CLRCLUEVT • $SETCLUEVT • $TSTCLUEVT Persona system services: • $PERSONA_ASSUME • $PERSONA_CREATE • $PERSONA_DELETE
OpenVMS Alpha programming support for SCSI-2 devices	On Alpha systems, OpenVMS now supports the tagged command queuing architecture of the SCSI-2 standard.
C++ RTL object library	Object modules are now available in SYS$LIBRARY:STARLET.OLB.
Linker	The new linker option, RMS_RELATED_CONTEXT=, enables or disables RMS related name context processing within the linker option file.
DEC C XPG4 Localization Utilities	
Localization utilities	OpenVMS Version 6.2 provides XPG4-compliant utilities for managing localization data for international software applications or layered products.
Local Area Network (LAN) Management Enhancements	
Two new LAN utilities	You can perform the following system management tasks related to LAN operations: • Set LAN parameters to customize the LAN environment. • Display LAN settings and counters. • Provide Maintenance Operations Protocol (MOP) downline load support for booting satellites.
SCSI as a VMScluster Storage Interconnect	
Storage interconnect	VMScluster systems now support the Small Computer Systems Interface (SCSI) as a shared multihost storage interconnect.
VMScluster Systems That Span Multiple Sites	
Multiple-site VMScluster configurations	VMScluster systems can have member nodes in geographically separate areas, using the new DS3 and ATM communications service.

2
General User Features

This chapter provides the following new features information for all users of the OpenVMS operating system:

- OpenVMS Freeware for VAX and Alpha systems
- Automatic foreign commands from DCL (paths)
- DCL commands — changes and enhancements
- New keyword for Sort/Merge utility
- DCL commands for TCP/IP users
- Specifying user addresses in Mail

2.1 OpenVMS Freeware for VAX and Alpha Systems

OpenVMS Engineering is sponsoring an OpenVMS Freeware CD–ROM, which contains “public domain” and “freeware” software that runs on OpenVMS Version 6.0 or later. The purpose of the OpenVMS Freeware CD–ROM is to provide OpenVMS customers with easy access to public domain software and free internal Digital software and tools.

The OpenVMS Freeware disc contains free software tools and utilities to aid software developers in both creating applications and managing or using OpenVMS systems. Many of these tools are popular packages already in use, and others are internally developed Digital tools that our engineers are making available to OpenVMS customers. For example, this disc includes Bliss-32 for OpenVMS VAX, Bliss-32 for OpenVMS Alpha, and Bliss-64 for OpenVMS Alpha.

2.1.1 Features of the Freeware Disc

The Freeware disc includes the following features:

- An easy-to-use 4GL-based menu system for examining the contents of the CD–ROM
- Files–11 format to make it readable on OpenVMS systems
- Binaries, source code (except for the Bliss compilers), and documentation
- Some of the most popular free packages, plus new items never before made public

Disclaimer on Testing, Quality, and Licensing

The OpenVMS Freeware for VAX and Alpha Systems CD–ROM is provided “as is” without a warranty. Digital imposes no restrictions on its distribution nor on the redistribution of anything on it. Be aware, however, that some of the packages on the disc may carry restrictions on their use, imposed by the original authors. Therefore, you should carefully read the documentation accompanying the products.

2.1.2 Guidelines for Software

The guidelines for the software to be contained on the OpenVMS Freeware CD–ROM are:

- The software is unencumbered to the public. No license fees shall be requested of or required by the customer for its use.
- The software must run on OpenVMS Version 6.0 or later.
- Digital does not issue a warranty for this software.
- Digital does not provide services for this software, fix the software, or guarantee that it works correctly.
- Customers are free to adapt the source files to their needs.

2.1.3 Submission of Freeware

Instructions for submission of software are included on the disc.

2.2 Automatic Foreign Commands from DCL (Paths)

In previous versions of the OpenVMS operating system, DCL symbols needed to be defined to implement *user-defined* or *foreign* commands.

With this release of the OpenVMS operating system, it is now possible in many cases to tell DCL to search a specific set of directories for executable images or command procedures and have them automatically invoked without defining a DCL symbol.

For more information, see the *OpenVMS User's Manual*.

2.3 DCL Commands — Changes and Enhancements

Table 2–1 lists the new and updated DCL commands. Refer to the *OpenVMS DCL Dictionary* for details.

Note that BOOT and LOGIN have been added as keywords to the /BEFORE and /SINCE qualifiers for many commands to select files dated prior to or since the last system boot or the user's last login.

Table 2–1 DCL Dictionary Command Updates

Command	DCL Dictionary Modification
ANALYZE/IMAGE	Documentation updated
APPEND	/BEFORE (updated) /SINCE (updated)
ASSIGN	/USER_MODE (updated)

(continued on next page)

Table 2–1 (Cont.) DCL Dictionary Command Updates

Command	DCL Dictionary Modification
COPY	/BEFORE (updated) /SINCE (updated) /TRUNCATE (updated) /WRITE_CHECK (updated)
CREATE/NAME_TABLE	/USER_MODE (updated)
DEASSIGN	/USER_MODE (updated)
DEFINE	/USER_MODE (updated)
DELETE	/BEFORE (updated) /SINCE (updated)
DIFFERENCES	/CHANGE_BAR (updated)
DIRECTORY	/BEFORE (updated) /SINCE (updated) /TIME (added)
EXCHANGE/NETWORK	/BEFORE (updated) /SINCE (updated)
F$CONTEXT lexical	NEQ (updated)
F$GETQUI lexical	INTERVENING_BLOCKS (added) INTERVENING_JOBS (added)
F$GETSYI lexical	‡CONSOLE_VERSION (added) ‡PALCODE_VERSION (added) QUANTUM (added) USED_GBLPAGCNT (added) USED_GBLPAGMAX (added) CLUSTER_EVOTES (added) NODE_EVOTES (added) REAL_CPUTYPE (added)
INITIALIZE	/DENSITY (updated)
PRINT	/BEFORE (updated) /SINCE (updated)
PURGE	/BEFORE (updated) /SINCE (updated)
RECALL	Documentation (updated)
RENAME	/BEFORE (updated) /INHERIT_SECURITY (updated) /SINCE (updated)
RUN (process)	Documentation updated
SEARCH	/BEFORE (updated) /KEY (added) /MATCH=XOR (added) /MATCH=EQV (added) /SINCE (updated) /WARNINGS (added)
SET AUDIT	/ENABLE=(AUDIT=ILLFORMED) (added)
SET DIRECTORY	/BEFORE (updated) /SINCE (updated)

‡Alpha specific.

(continued on next page)

Table 2–1 (Cont.) DCL Dictionary Command Updates

Command	DCL Dictionary Modification
SET FILE	/BEFORE (updated) /SINCE (updated) Example (added)
SET HOST/DTE	Baud rates (updated)
SET HOST/LAT	/DIAL (added) /[NO]EIGHT_BIT (added) /SPEED (added)
SET MAGTAPE	Documentation updated
SET SECURITY	/BEFORE (updated) /SINCE (updated)
SET TERMINAL	/SECURE_SERVER (updated)
SET TIME	/CLUSTER (now supported)
SET VOLUME	Documentation updated
SHOW DEVICES	/REBUILD_STATUS (added)
SHOW DEVICES/SERVED	/EXACT (added) /HIGHLIGHT (added) /PAGE (added) /SEARCH (added)
SHOW PROCESS	/DUMP (added) /INTERVAL (updated)
SHOW SECURITY	/BEFORE (updated) /SINCE (updated)
SHOW SYSTEM	/[NO]BATCH (updated) /[NO]HEADING (added) /IDENT (added) /[NO]INTERACTIVE (added) /[NO]NETWORK (updated) /OWNER_UIC (added) /[NO]PROCESS (updated) /STATE (added) /[NO]SUBPROCESS (updated) Examples (added)
SHOW TERMINAL	/BRIEF (added) /FULL (added) Examples (added)
SHOW USERS	/[NO]BATCH (updated) /[NO]HEADING (added) /[NO]INTERACTIVE (updated) /[NO]NETWORK (updated) /[NO]SUBPROCESS (updated) Examples (added)
SPAWN	Documentation updated
STOP	Documentation updated
SUBMIT	/BEFORE (updated) /SINCE (updated)
TYPE	/BEFORE (updated) /SINCE (updated) /TAIL (updated)

2.4 Sort/Merge Utility (SORT/MERGE)

The VALUE keyword is now available for use with the /FIELD qualifier in Sort/Merge specification files. The VALUE keyword allows you to define a constant field and assign it a value of any valid Sort/Merge data-type. You can use this constant field in /KEY, /DATA, and /CONDITION statements.

The following are sample formats for the /FIELD qualifier:

```
/FIELD=(NAME=field-name,POSITION:n,SIZE:n[,DIGITS:n][,data-type])
```

```
/FIELD=(NAME=field-name,VALUE:n,SIZE:n[,DIGITS:n][,data-type])
```

If you specify VALUE:n, do not specify POSITION:n. The VALUE field is a constant and is not part of an input record.

For more information about the Sort/Merge utility, see the *OpenVMS User's Manual*.

2.5 DCL Commands for TCP/IP Users

Several DCL commands have new qualifiers that make frequently used TCP/IP functions easier to access and better integrated. Some of these commands are:

- COPY /FTP
- COPY /RCP
- DIRECTORY /FTP
- SET HOST /RLOGIN
- SET HOST /TELNET
- SET HOST /TN3270

Note that in order for the new commands to work, you must install a TCP/IP product that supports them. DEC TCP/IP Services for OpenVMS (UCX) Version 3.2 supports only the SET HOST /RLOGIN command; the other commands require Version 3.3.

For more information, see *TCP/IP Networking on OpenVMS Systems*.

2.6 Specifying User Addresses in Mail

When you use the Mail utility or callable Mail routines, you can now specify user addresses in the Internet format without specifying the transport or quotes (unless the address contains reserved characters).

For example, in previous versions of OpenVMS, SMTP%"user@node.org" was required to send mail to an Internet address. The string may now be specified as user@node.org.

- If the Internet transport on your machine is not SMTP, then the logical name MAIL$INTERNET_TRANSPORT may be defined to select an alternate transport.
- The user@node syntax can also be used to specify a DECnet phase IV node name or a DECnet/OSI alias. In this case, it is treated as node::user.

Note the following:

- This feature is available in character-cell mail and DECwindows mail.
- Although this change removes the requirement that you include the transport and quotes when specifying a user address, you must still include quotes if you choose to specify the transport.
- You can specify the user@node.org format only from a local system.

For more information about the Mail utility, see the *OpenVMS User's Manual*. For more information about callable Mail, see the *OpenVMS Utility Routines Manual*.

3

System Management Features

This chapter contains information about the following new features, changes, and enhancements for system managers:

- OpenVMS Management Station bundled with OpenVMS
- Volume Shadowing new features
- VMScluster systems new features
- Local area network (LAN) management enhancements
- Booting VMScluster satellite nodes using the new LANCP utility
- AUTOGEN changes
- New and changed system parameters
- Backup Manager
- New OpenVMS BACKUP features
- Full name support
- Two proxy authorization files, NETPROXY.DAT and NET$PROXY.DAT
- Security Server process
- New LAT system management features
- New /NO_SHARE qualifier for LICENSE MODIFY command
- Choosing languages and date/time formats
- Operator Communication Manager (OPCOM) changes
- Invalid symbiont message codes
- Erasing old home blocks
- Additional SHUTDOWN.COM logicals
- Installing or upgrading from a running Alpha system
- OpenVMS support for Transmission Control Protocol/Internet Protocol (TCP/IP) networking
- Dump file off the system disk (VAX systems only)
- DECamds
- Inclusion of selected layered products

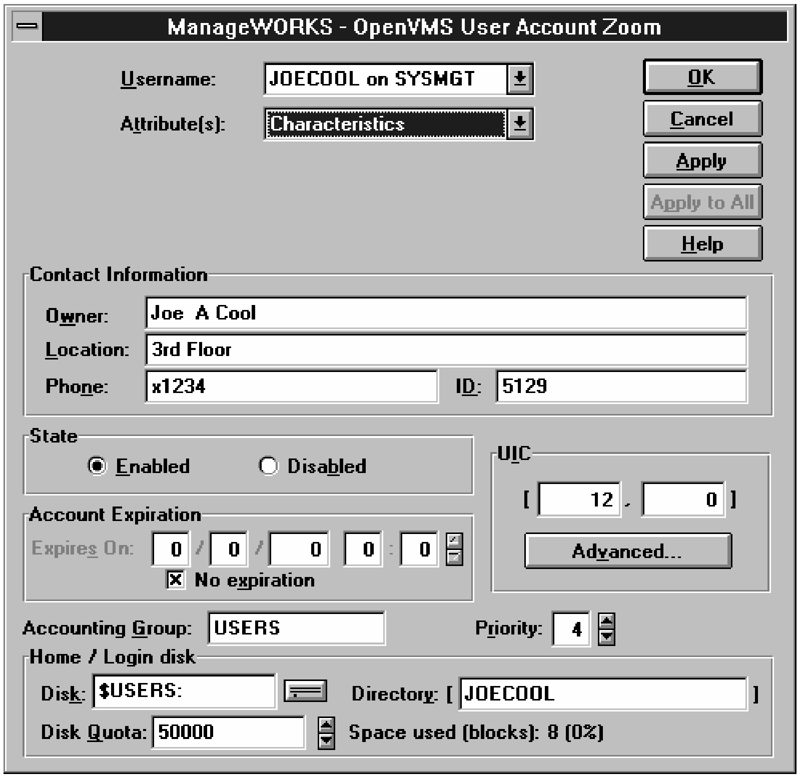
3.1 OpenVMS Management Station Bundled with OpenVMS

OpenVMS Management Station is a powerful Microsoft Windows based management tool for system managers and others who perform account management tasks on OpenVMS systems. OpenVMS Management Station provides a comprehensive user interface to OpenVMS account management across multiple systems.

OpenVMS Management Station coexists with all of the current OpenVMS system management utilities.

Figure 3–1 shows a sample OpenVMS Management Station screen.

Figure 3–1 Sample OpenVMS Management Station Screen



Username: JOECOOL on SYSMGT

Attribute(s): Characteristics

Owner: Joe A Cool

Location: 3rd Floor

Phone: x1234 ID: 5129

● Enabled ○ Disabled

[12 , 0]

Expires On: 0 / 0 / 0 0 : 0

☒ No expiration

Accounting Group: USERS Priority: 4

Disk: $USERS: Directory: [JOECOOL]

Disk Quota: 50000 Space used (blocks): 8 (0%)

OpenVMS Management Station addresses the problem of having to use multiple OpenVMS utilities to manage OpenVMS accounts. For example, creating an account traditionally consists of the following steps:

- Adding a UAF entry
- Granting rights identifiers
- Creating a directory
- Creating disk quotas
- Granting network proxies

This operation requires you to use DCL, the Authorize utility, and the DISKQUOTA component of the System Management utility. OpenVMS Management Station provides an easy-to-use interface to this process so that you do not need to use multiple utilities.

OpenVMS Management Station consists of two components. You install:

- The Microsoft Windows based client software on a PC to perform all management operations
- The server component on all of the OpenVMS systems you want to manage

You do not interact directly with the server component.

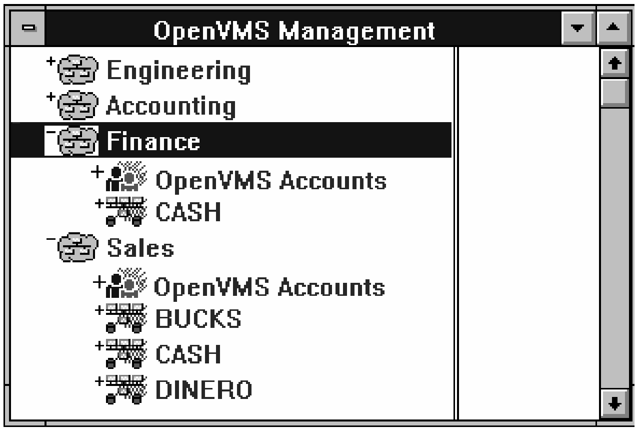
3.1.1 What Can You Do with OpenVMS Management Station?

OpenVMS Management Station allows you to organize the systems you need to manage in ways that are meaningful to you and your environment, and allows you to manage user accounts on those systems.

You can easily manage user accounts across multiple OpenVMS systems, depending on your needs. The systems might be some of the clusters in a network, all of the systems on a given floor of a building, a mix of clusters and nonclustered nodes, and so forth.

Figure 3–2 shows how a screen might look when you are managing user accounts.

Figure 3–2 Managing User Accounts with OpenVMS Management Station



You can use OpenVMS Management Station to manage OpenVMS user accounts in a convenient, easy manner. For example, when creating an account on multiple systems, OpenVMS Management Station can add a user authorization file (UAF) entry, grant rights identifiers, create an OpenVMS directory, set a disk quota, set up OpenVMS Mail characteristics, and so forth, for each instance of the account.

OpenVMS Management Station manages the following OpenVMS resources:

- SYSUAF.DAT user authorization file
- RIGHTSLIST.DAT user rights file
- Network proxy database
- Account login-directory trees
- User account disk quotas
- OpenVMS Mail VMSMAIL_PROFILE.DATA file

3.1.2 What Account Management Operations Are Supported?

OpenVMS Management Station supports the following account management operations:

- Creating user accounts
- Modifying user accounts (any aspect)
- Deleting user accounts
- Renaming user accounts
- Displaying user account attributes

3.1.3 Where Is OpenVMS Management Station Documented?

The features and functions of OpenVMS Management Station are completely described in extensive Microsoft Windows help files. The help files include step-by-step instructions and numerous examples.

The *OpenVMS Management Station Overview and Release Notes* document provides an overview of OpenVMS Management Station and describes how to get started using the software.

Information about installing OpenVMS Management Station on your Alpha or VAX computer and your PC is located in the following manuals:

- *OpenVMS Alpha Version 6.2 Upgrade and Installation Manual*
- *OpenVMS VAX Version 6.2 Upgrade and Installation Manual*

3.2 Volume Shadowing New Features

Volume Shadowing for OpenVMS contains the following new features:

- Support for geometry-based shadowing device recognition
- Increased limits on numbers of disks allowed in shadow sets
- StorageWorks RAID Array Subsystem support

In addition to these new features in Volume Shadowing, OpenVMS Alpha Version 6.2 and OpenVMS VAX Version 6.2 now support the HSJ40, HSJ30, HSD30, HSZ40, and HSD05 controllers. See the OpenVMS SPD for the complete list of controllers and firmware.

3.2.1 Support for Geometry-Based Shadowing Device Recognition

Formerly, Volume Shadowing checked device IDs and the maximum logical block numbers (LBNs) on devices. With this release, Volume Shadowing checks for geometries and maximum LBNs instead. This enables devices such as the RZ28 and the RZ28B to operate in the same shadow set. Even though their device IDs differ, their geometries and maximum LBNs will match when configured on like controllers (two HSJ controllers, for example).

3.2.2 Increased Limits on Numbers of Disks Allowed in Shadow Sets

Table 3–1 shows the new limits for numbers of disks allowed in shadow sets:

Table 3–1 Number of Shadow Sets Supported

Type of Shadow Set	Sets Supported
Single member	Unlimited sets
Multimember	Total of 400 disks in two- and three-member sets, or both

These limits apply per cluster. For example, 400 total disks could be configured into 200 two-member shadow sets or into 133 three-member shadow sets per cluster. If single, two- and three-member shadow sets are all present on a single cluster, then a maximum of 400 disks may be contained in the two- and three-member shadow sets.

3.2.2.1 Configuration Guidelines

Increasing the number of shadow sets supported on an OpenVMS cluster also increases the risk that the system overhead required to perform shadowing functionality will exceed customer expectations. Customers are urged to use care in configuring clusters with large numbers of shadow sets. The *Volume Shadowing for OpenVMS* manual is the best source for shadowing configuration information. The following are areas where users of the shadowing product should use caution:

- Shadow copy/merge performance

  The purpose of the shadowing product is to enhance storage availability. Copy and merge operations may have an impact on performance.

  Shadow copy/merge operations require that the entire disk be accessed. On very large devices that are not attached to controllers with shadowing assists, this can take a long time. A shadow copy on a one-gigabyte disk without copy assists will take about an hour. It will take longer if the copy operation competes with heavy user I/O, is served over a slow Ethernet link, or both. A large number of shadow sets performing copy/merge operations in parallel will compete for system resources. The system overhead involved in performing shadow copy and merge may limit the number of shadow sets mounted on a given system or cluster.

- Satellite-initiated copy/merge is not recommended

  Cluster satellite nodes should set the system parameter SHADOW_MAX_COPY to zero. This will prevent copy and merge operations from occurring over the Ethernet. This is especially important for slow Ethernet links or for high Ethernet traffic. Satellite nodes doing a large number of copies or merges will substantially degrade performance of the satellite node, the Ethernet link, or both. Shadow copies should occur on nodes with direct connections to shadow set members. This work is controlled by the system parameter SHADOW_MAX_COPY on each cluster node.

- Pool size

  Mounting large numbers of shadow sets will require additional nonpaged pool. For a larger number of shadow sets, the NPAGEDYN parameter may need to be increased. See the documentation on AUTOGEN.

- Mount /CLUSTER only when required

Mounting shadow sets only on cluster nodes that require them to be mounted will reduce cluster shadowing overhead, especially on very large clusters.

3.2.3 StorageWorks RAID Array Subsystem Support

The StorageWorks RAID Array 210 subsystem (KZESC-AA or KZESC-BA EISA Backplane RAID controllers) and the StorageWorks PCI Backplane RAID controller (KZPSC-AA or KZPSC-BA) have their own firmware implementations of RAID, levels 0, 1, and 5. You can use the firmware implementation of RAID level 1 (shadowing) to create shadow sets using the SCSI disks attached locally to a single RAID controller.

With OpenVMS Version 6.2, the SCSI disks connected to these controllers can also be included in shadow sets created using host-based Volume Shadowing for OpenVMS. In previous releases, these disks were not accessible by Volume Shadowing. With host-based Volume Shadowing, now you can create a RAID1 shadow set containing two like disks or two like RAID arrays, each of which is attached to a separate RAID controller located within a cluster. You also can create a RAID1 shadow set using one SCSI disk attached to the StorageWorks RAID controller and one SCSI disk attached elsewhere in the cluster (as long as the disk geometries and sizes match).

See the *StorageWorks RAID Array Subsystem User's Guide* for more information on using Volume Shadowing with these controllers.

3.3 VMScluster Systems New Features

In OpenVMS Version 6.2, VMScluster systems include the following new features:

- Support for SCSI interconnect in a VMScluster system
- Support for multiple-site VMScluster systems
- OpenVMS Cluster Client Software
- DECnet software no longer required for VMScluster software
- Support for TMSCP served SCSI tapes
- DECamds enhancements
- Enhanced support for HSJ, HSC, and HSD series controller failover
- Warranted and migration VMScluster system configurations

These features further enhance the performance, availability, and functionality of VMScluster systems. The following sections describe these features in more detail. In addition, Section 3.3.8 charts the most recent pairings information for supported VMScluster system configurations.

3.3.1 Support for SCSI Interconnect in a VMScluster System

You can use the SCSI interconnect as a storage interconnect for multiple Alpha systems in a VMScluster system. A SCSI interconnect, also referred to as a SCSI bus, is an industry-standard interconnect that supports SCSI storage devices, controllers, and support components.

VMScluster Software for OpenVMS Alpha Version 6.2 provides support for a limited set of multiple-host SCSI configurations. You can configure systems, SCSI adapters, controllers and disk devices on multiple-host SCSI buses using the components in Table 3–2.

Table 3–2 SCSI VMScluster System Hardware

Supported Component	Description
Alpha Systems:	AlphaServer 2100 AlphaServer 2000 AlphaServer 1000 AlphaServer 400 AlphaStation 2500 AlphaStation 400 AlphaStation 200
SCSI Adapters:	The embedded SCSI adapter available in all the supported Alpha systems. Optional add-on KZPAA adapters.
Controllers:	HSZ40 and all its supported disks. Note that DWZZA single-ended to fast-wide differential converters are required when configuring HSZ40 controllers.
Disks:	You can directly connect the following disks on a multihost SCSI bus: RZ28 RZ28B RZ26 RZ26L RZ29B

In a VMScluster system, you can configure:

- Up to two systems to a multiple-host SCSI bus
- A system to a maximum of two multiple-host SCSI buses
- Any number of single-host SCSI buses

Chapter 7 provides a complete discussion about SCSI configurations, setup, and management.

3.3.2 Support for Multiple-Site VMScluster Systems

Since OpenVMS Version 6.1, VMScluster systems have supported DS3 technology, also called T3, as a cluster interconnect when used in conjunction with FDDI and appropriate bridging technology. Using DS3 as an interconnect, nodes in a VMScluster can be located in multiple, geographically separate sites as far apart as 150 miles. With OpenVMS Version 6.2, this feature is enhanced to include ATM technology.

Chapter 8 provides a complete discussion of multiple-site clusters.

3.3.3 OpenVMS Cluster Client Software

OpenVMS Version 6.2 introduces a new license type for OpenVMS cluster software, called OpenVMS Cluster Client software. The new license provides a low-cost cluster client product for Alpha and VAX workstations.

The OpenVMS Cluster Client software provides fully functional OpenVMS Cluster software with two exceptions:

- Clients cannot provide VOTES to the VMScluster configurations
- Clients cannot MSCP serve disks or TMSCP serve tapes

OpenVMS Cluster Client software is available for VAX and Alpha systems. The software uses a License Management Facility (LMF) license name of VMSCLUSTER-CLIENT. OpenVMS Cluster Client software is included in the NAS 150 package, and can be ordered separately. Refer to the OpenVMS Cluster *Software Product Description* for ordering information.

3.3.4 DECnet Software No Longer Required for VMScluster Software

Prior to OpenVMS Version 6.2, DECnet Phase IV or DECnet/OSI software was a prerequisite for VMScluster software. With OpenVMS Version 6.2, there is no longer a requirement for DECnet software.

DECnet software is required in VMScluster environments under the following conditions:

- When applications perform node-to-node communication using DECnet mailboxes
- When the MONITOR/CLUSTER utility is used

Note that DECnet is not required for the correct operation of SYSMAN in a VMScluster configuration. Also, you can perform SET HOST operations without DECnet by using the SET HOST/LAT command.

3.3.5 Support for TMSCP Served SCSI Tapes

VMScluster Systems for OpenVMS has been enhanced to allow the TMSCP server to serve SCSI tapes. The TMSCP server makes locally connected tapes of the following types available across a cluster:

- TA series tapes for CI
- TF series tapes for DSSI
- TZ and TLZ tapes for SCSI

The TMSCP server is controlled by the TMSCP_LOAD and TMSCP_SERVE_ALL system parameters.

- The TMSCP_LOAD parameter controls whether the TMSCP server is loaded. By default, the value of the TMSCP_LOAD parameter is set to zero so that the TMSCP server is not loaded. Refer to *VMScluster Systems for OpenVMS* for information about setting this parameter.
- The TMSCP_SERVE_ALL system parameter is new with this release. This parameter specifies TMSCP tape-serving functions when the TMSCP server is loaded. If TMSCP_LOAD is set to zero, the TMSCP_SERVE_ALL parameter is ignored. Table 3–3 describes the TMSCP_SERVE_ALL parameter settings.

Table 3–3 The TMSCP_SERVE_ALL System Parameter

Value	Description
0	Serve no tapes. This is the default value.
1	Serve all available tapes.
2	Serve only locally connected tapes.

3.3.5.1 No TMSCP Server Support for SCSI Retension Command

The SCSI Retension command modifier is not supported by the TMSCP server. Retension operations should be performed from the node serving the tape.

3.3.6 DECamds Enhancements

DECamds includes enhancements to provide additional event filtering and the ability to perform real-time recomputing of VMScluster quorum. Additionally, the DECamds console has been ported from VAX so that you can run it on an Alpha system.

For more information, see Section 3.23.

3.3.7 Enhanced Support for HSJ, HSC, and HSD Series Controller Failover

In previous releases of VMScluster software, dual porting of disks between pairs of HSJ and HSC series controllers was supported when the controllers were attached to a common star coupler. Beginning with OpenVMS Version 6.2, you can connect dual-ported disks to pairs of HSJ and HSC series controllers that are attached to different star couplers. This feature enhances availability because failure of a CI adapter need not cause both controllers to become unreachable.

This feature is available for HSJ and HSC series CI controllers and for HSD30 DSSI controllers. It permits HSD30 controllers to be configured across different DSSI buses.

3.3.8 Warranted and Migration VMScluster System Configurations

OpenVMS Alpha Version 6.2 and OpenVMS VAX Version 6.2 provide two levels of support for mixed-version and mixed-architecture VMScluster systems. These two support types are warranted and migration.

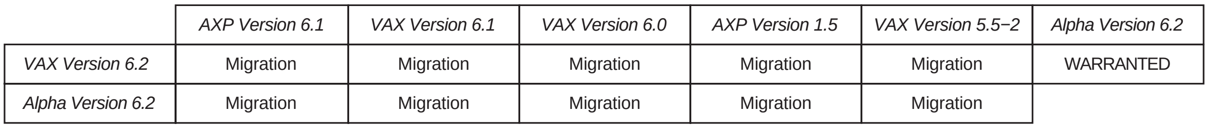
Warranted support means that Digital has fully qualified the two versions coexisting in a VMScluster system and will answer all problems identified by customers using these configurations.

Migration support is a superset of the rolling upgrade support provided in earlier releases of OpenVMS, and is available for mixes that are not warranted. Migration support means that Digital has qualified the versions for use together in configurations that are migrating in a staged fashion to a newer version of OpenVMS VAX or to OpenVMS Alpha. Problem reports submitted against these configurations will be answered by Digital. However, in exceptional cases, Digital may request that you move to a warranted configuration as part of fixing the problem.

Migration support will help customers move to warranted VMScluster version mixes with minimal impact on their cluster environments.

Figure 3–3 shows the level of support provided for all possible version pairings.

Figure 3–3 VMScluster Version Pairings



	AXP Version 6.1	*VAX Version 6.1*	*VAX Version 6.0*	*AXP Version 1.5*	*VAX Version 5.5−2*	*Alpha Version 6.2*
VAX Version 6.2	Migration	Migration	Migration	Migration	Migration	WARRANTED
Alpha Version 6.2	Migration	Migration	Migration	Migration	Migration	

ZK−7496A−GE

Note that Digital does not support the use of more than two versions in a VMScluster system at a time. In many cases, more than two versions will successfully operate but Digital cannot commit to resolving problems experienced with such configurations.

The next major release of the OpenVMS operating system will coexist in a mixed-version VMScluster system with OpenVMS Version 6.2 (both VAX and Alpha). This is consistent with the Digital policy of providing mixed-version coexistence across adjacent OpenVMS releases. However, due to the development of many significant new features within the OpenVMS operating system, it is expected that the next major release will not coexist with OpenVMS Version 6.1 and earlier releases.

3.4 Local Area Network (LAN) Management Enhancements

The local area network (LAN) software has been enhanced to include system management tools for LAN configurations. The enhancements include two LAN utilities that perform system management tasks.

Chapter 6 discusses these utilities.

3.5 Booting VMScluster Satellite Nodes Using the New LANCP Utility

LANCP (LAN Control Program) and LANACP (LAN Ancillary Control Process) are new images supplied with OpenVMS Version 6.2. These images provide a general purpose MOP booting service that can be used for VMScluster satellites. This new utility, along with the CLUSTER_CONFIG_LAN.COM procedure, allows VMScluster systems to boot satellites without the use of DECnet and provides several features not available with DECnet MOP booting.

By default in OpenVMS Version 6.2, DECnet is used to service satellite boot requests. However, you might want to use use LANCP/LANACP for satellite booting if you wish to operate a VMScluster without DECnet or for other reasons. This section provides instructions for using LANCP/LANACP for satellite booting.

For more information on the LANCP utility, see Chapter 6.

3.5.1 Overview

LANCP is the control program for LANACP, which runs as a detached process to service boot requests. LANCP and LANACP utilize the following data files:

- SYS$SYSTEM:LAN$DEVICE_DATABASE.DAT

  This file maintains information about devices on the local node. By default, the file is created in SYS$SPECIFIC:[SYSEXE], and the system looks for the file in that location. However, you can modify the file name or location for this file by redefining the systemwide logical name LAN$DEVICE_DATABASE.

- SYS$SYSTEM:LAN$NODE_DATABASE.DAT

  This file contains information about the nodes for which LANACP will supply boot service. This file should be shared among all nodes in the VMScluster, including both Alpha and VAX systems. By default, the file is created in SYS$COMMON:[SYSEXE], and the system looks for the file in that location. However, you can modify the file name or location for this file by redefining the systemwide logical name LAN$NODE_DATABASE.

3.5.2 Using LANACP to Boot Satellites

You can use the LANACP utility to service all types of MOP downline load requests. This includes VMScluster satellite nodes, terminal servers, LAN resident printers, and X-terminals.

Note

Digital recommends that customers currently using DECnet Phase IV or DECnet/OSI to perform MOP operations such as satellite loading, continue to do so. The new LANACP/LANCP feature is only necessary for configurations that are not using DECnet.

The next sections describe how to use LANACP for satellite booting for new installations and existing installations. Support for using LANCP and LANACP for satellite booting requires all VMScluster nodes to be running OpenVMS Version 6.2.

3.5.2.1 New Installations

For new OpenVMS installations to use LANCP/LANACP for satellite booting, follow these steps:

1. Add the startup command for LANACP

   You should start up LANACP as part of your system startup procedure. To do this, remove the comment from the line in SYS$MANAGER:SYSTARTUP_VMS.COM that runs the LAN$STARTUP command procedure. If your VMScluster system will have more than one system disk, see Section 3.5.1 for a description of logicals that can be defined for locating LANACP configuration files.

   ```
   $ @SYS$STARTUP:LAN$STARTUP
   ```

   You should now either reboot the system or invoke the above command procedure from the system manager's account to start LANACP.

2. Follow the steps in the *VMScluster Systems for OpenVMS* manual for configuring a VMScluster system and adding satellites. Use the command procedure CLUSTER_CONFIG_LAN.COM instead of CLUSTER_CONFIG.COM. If you invoke CLUSTER_CONFIG.COM, it will give you the option to switch to running CLUSTER_CONFIG_LAN.COM if the LANACP process has been started.

3.5.2.2 Existing Installations

For an existing installation to migrate to using LANCP/LANACP for satellite booting, several steps are required. These include configuration of LANACP and configuration of the node database with the nodes to be MOP booted by LANACP. This initial release of LANCP/LANACP is primarily targeted for new VMScluster installations. There does exist some support to assist in migrating from DECnet Phase IV and minimal support for migrating from DECnet/OSI to LANCP/LANACP.

Configuring LANCP/LANACP

To configure the data files SYS$SYSTEM:LAN$DEVICE_DATABASE.DAT and SYS$SYSTEM:LAN$NODE_DATABASE.DAT, follow these steps:

1. Define LANCP/LANACP database logicals

   Define systemwide logicals for LAN$DEVICE_DATABASE and LAN$NODE_DATABASE if you wish these files to be located in other than the default locations specified above. These definitions should be added to the system startup files.

2. Create the LAN$DEVICE_DATABASE

   Use LANCP to configure LAN devices for MOP booting. The permanent LAN$DEVICE_DATABASE is created when the first DEVICE command is issued. To create the database and get a list of available devices, enter:

   ```
   $ MCR LANCP
   LANCP> LIST DEVICE /MOPDLL
   %LANCP-I-FNFDEV, File not found, LAN$DEVICE_DATABASE
   %LANACP-I-CREATDEV, Created LAN$DEVICE_DATABASE file

   Device Listing, permanent database:
     --- MOP Downline Load Service Characteristics ---
   Device    State   Access Mode      Client           Data Size
   ------    -----   -----------      ------           ---------
   ESA0    Disabled NoExlusive  NoKnownClientsOnly     246 bytes
   FCA0    Disabled NoExlusive  NoKnownClientsOnly     246 bytes
   ```

3. Use LANCP to enable LAN device for MOP booting

   By default, the LAN devices have MOP booting capability disabled. Determine the LAN devices for which you want to enable MOP booting. Then use the DEFINE command in the LANCP utility to enable these devices to service MOP boot requests in the permanent database, as shown in the following example:

   ```
   LANCP> DEFINE DEVICE ESA0:/MOP=ENABLE
   ```

4. Configure the LAN$NODE_DATABASE

   If you are running DECnet Phase IV, you can use the command procedure SYS$EXAMPLES:LAN$POPULATE.COM. This procedure extracts all MOP booting information from a DECnet Phase IV NETNODE_REMOTE.DAT file and produces a site-specific command procedure called LAN$DEFINE.COM. You can then execute LAN$DEFINE.COM to populate LAN$NODE_DATABASE.

   For DECnet Phase IV sites, the LAN$POPULATE procedure scans all DECnet areas (1–63) by default. If you MOP boot systems from only a single or a few DECnet areas, you can cause the LAN$POPULATE procedure to operate on a single area at a time by providing the area number as the P1 parameter to the procedure, as shown in the following example (including log):

   ```
   $ @SYS$EXAMPLES:LAN$POPULATE 15

    LAN$POPULATE - V1.0

    Do you want help (Y/N) <N>:

    LAN$DEFINE.COM has been successfully created.

    To apply the node definitions to the LANCP permanent database,
    invoke the created LAN$DEFINE.COM command procedure.
   ```

```
        Digital recommends that you review LAN$DEFINE.COM and remove any
        obsolete entries prior to executing this command procedure.

 A total of 2 MOP definitions were entered into LAN$DEFINE.COM
```

The LAN$DEFINE procedure is created as part of the LAN$POPULATE procedure. In the following sequence, the LAN$DEFINE.COM procedure that was just created is typed on the screen and then executed:

```
$ TYPE LAN$DEFINE.COM

 $ !
 $ ! This file was generated by LAN$POPULATE.COM on 16-DEC-1994 09:20:31
 $ ! on node CLU21.
 $ !
 $ ! Only DECnet Area 15 was scanned.
 $ !
 $ MCR LANCP
 Define Node PORK    /Address=08-00-2B-39-82-85 /File=APB.EXE -
                  /Root=$21$DKA300:<SYS11.> /Boot_type=Alpha_Satellite
 Define Node JYPIG   /Address=08-00-2B-A2-1F-81 /File=APB.EXE -
                  /Root=$21$DKA300:<SYS10.> /Boot_type=Alpha_Satellite
 EXIT

$ @LAN$DEFINE

 %LANCP-I-FNFNOD, File not found, LAN$NODE_DATABASE
 -LANCP-I-CREATNOD, Created LAN$NODE_DATABASE file
 $
```

If you are currently running DECnet/OSI, information on the nodes that DECnet/OSI will MOP boot can be obtained with the following NCL command:

```
$ MCR NCL
NCL> SHOW MOP CLIENT * ALL
```

Nodes for MOP booting can then be added to LANCP with the following define command:

```
$ MCR LANCP
LANCP> DEFINE NODE node-name -
/ADDRESS=hw-address -
/BOOT_TYPE={VAX_SATELLITE,ALPHA_SATELLITE} -
/ROOT=disk-device:<root-name.>
```

Section 6.1.5 provides detailed information on the LANCP DEFINE command.

5. Start LANACP

   To start LANACP, execute the startup command procedure as follows:

```
$ @SYS$STARTUP:LAN$STARTUP
  %RUN-S-PROC_ID, identification of created process is 2920009B
  $
```

   You should start up LANACP for all boot nodes as part of your system startup procedure. To do this, include the following line in your site-specific startup file (SYS$MANAGER:SYSTARTUP_VMS.COM):

```
$ @SYS$STARTUP:LAN$STARTUP
```

   If you have defined logicals for either LAN$DEVICE_DATABASE or LAN$NODE_DATABASE, be sure that these are defined in your startup files prior to starting up LANACP.

6. Disable DECnet MOP booting

   If you use LANCP/LANACP for satellite booting, you may no longer need DECnet to handle MOP requests. If this is the case for your site, you can turn off this capability with the appropriate NCP command (DECnet Phase IV) or NCL commands (DECnet/OSI).

3.6 AUTOGEN Changes

This section describes the changes made to AUTOGEN and how these changes affect this release.

3.6.1 Changes to Computation of SYSMWCNT Parameter

The formula AUTOGEN uses for computing the SYSMWCNT parameter has been changed slightly. It will now typically calculate from 200 to 400 pages more than the previous formula. The number of pages might be higher on MicroVAX systems.

3.6.2 Preventing Autoconfigure

You can prevent AUTOGEN from performing a SYSGEN AUTOCONFIGURE by defining the AGEN$NO_AUTOCONFIGURE logical to be TRUE or 1. This makes it safer to run AUTOGEN on an active system because the AUTOCONFIGURE can interfere with active I/O on certain devices, and will not interfere with normal startup as incautious use of the logical STARTUP$AUTOCONFIGURE_ALL might. It has the disadvantage that the GETDATA phase will record only the hardware that is already present and will not look for more.

3.6.3 Suppression of Informational Messages

You can suppress the display of informational messages on your terminal by defining the AGEN$REPORT_NO_INFORMATIONALS logical to TRUE or 1. Informational messages are entered in SYS$SYSTEM:AGEN$PARAMS.REPORT regardless of the value of AGEN$REPORT_NO_INFORMATIONALS.

3.6.4 Preventing AUTOGEN from Running As a Spawned Subprocess

You can prevent AUTOGEN from running as a spawned subprocess by defining the AGEN$NO_SPAWN logical to TRUE or 1. AUTOGEN normally runs as a spawned subprocess for two reasons:

- AUTOGEN uses a large number of DCL symbols in its work, and when run in a process that already has defined a large number of symbols, it can run out of room in the symbol table. Running in a spawned subprocess makes this occurrence unlikely.
- When run in a spawned subprocess, AUTOGEN can define some additional symbols in order to improve its performance.

If you define AGEN$NO_SPAWN to force AUTOGEN to run without spawning and it runs out of room in its symbol table, you will receive the following error message:

```
%AUTOGEN-F-SYMOVF, no room for symbol definitions
%AUTOGEN-F-DEL_EXP, delete some symbols or expand CLISYMTBL
```

3.6.5 Overriding Parameters Related to DECnet

The MODPARAMS.DAT file can now include the pseudoparameter LOAD_DECNET_IMAGES. Setting LOAD_DECNET_IMAGES to be TRUE (or FALSE) overrides AUTOGEN's observations regarding the presence or absence of DECnet. This is expected to be useful for sites with no synchronous network hardware that want to run asynchronous DECnet.

3.7 New and Changed System Parameters

Table 3–4 describes the new and changed system parameters for OpenVMS Version 6.2.

Table 3–4 New and Changed System Parameters

Parameter	Description
DR_UNIT_BASE (New Special Parameter)	Specifies the base value from which unit numbers for DR devices (Digital StorageWorks RAID Array 200 Family logical RAID drives) are counted.
DUMPSTYLE† (Changed)	Specifies the method of writing system dumps.
LGI_BRK_LIM (Changed)	Specifies the number of failures that can occur at login time before the system will take action against a possible break-in.
RMTDBG_SCRATCH_PAGES‡ (New)	On Alpha systems, the number of pages of memory allocated for the remote debugger. See the *OpenVMS Alpha Device Support Manual* for more information.
SHADOWING (Changed)	Digital does not support Phase I shadowing for versions greater than OpenVMS Version 6.1. Use Phase II shadowing instead. Phase II shadowing provides shadowing of all disks located on a standalone system or an OpenVMS cluster system. SHADOWING has the following new values: 0 No shadowing is enabled; SHDRIVER is not loaded. (Default) 2 Phase II shadowing is enabled; SHDRIVER is loaded.
TMSCP_SERVE_ALL (New)	Specifies TMSCP tape-serving functions when TMSCP server is loaded. If TMSCP_LOAD is set to 0, the TMSCP_SERVE_ALL parameter is ignored. Refer to *VMScluster Systems for OpenVMS* for information about setting the TMSCP_SERVE_ALL parameter.
VCC_PTES† (New)	Used on VAX systems to specify the MAXIMUM size that the virtual I/O cache is permitted to expand dynamically. VCC_PTES is a static parameter and requires that the system reboot after being changed by the AUTOGEN mechanism.

†VAX only
‡Alpha only

Refer to the *OpenVMS System Management Utilities Reference Manual* for more information about these system parameters.

3.8 Backup Manager

Backup Manager provides a screen-oriented interface to the OpenVMS Backup utility (BACKUP) based on the OpenVMS Screen Management Run-Time Library (RTL) routines. It presents a subset of BACKUP's capabilities in an intuitive, task-oriented, self-documenting manner that does not require knowledge of BACKUP command syntax and qualifiers. Backup Manager is designed to help the user perform common system management tasks such as backing up files to a save set, restoring files from a save set, and listing the contents of a save set.

To start Backup Manager, issue the following command:

```
$ RUN SYS$SYSTEM:BACKUP$MANAGER
```

Documentation on features, concepts, and how to use the utility is available on line through each screen's help menu. Context-sensitive help on any field is provided by pressing the Help key.

3.9 BACKUP Changes

This section describes the new features added to the OpenVMS Backup utility for this release.

See the *OpenVMS System Manager's Manual* for information about how to use BACKUP.

3.9.1 New Qualifier, /[NO]ALIAS

The /[NO]ALIAS qualifier has been added to allow users more control over how alias (synonym) file entries are processed by the OpenVMS Backup utility (BACKUP). The default is /ALIAS.

See the *OpenVMS System Management Utilities Reference Manual* for more information about this qualifier.

3.9.2 Increased Number of Directory Levels Supported

BACKUP now supports saving individual files or groups of files located in disk directories greater than eight levels in depth, up to 32 levels of disk directories. Prior versions of BACKUP (and OpenVMS utilities in general) could not access, and thus could not save, files located in directories greater than eight levels in depth.

BACKUP also supports restoring these files selectively in a BACKUP restore operation. Note, however, that OpenVMS file system restrictions still apply when restoring these files. For nonimage and nonphysical restore operations, individual files or groups of files may be restored to a disk, but the destination directories must be no greater than eight levels deep.

This new feature is of most interest to PATHWORKS users.

3.10 Full Name Support

Information about full name support that was documented in OpenVMS Version 6.1 manuals as VAX specific also applies to Alpha Version 6.2 systems.

3.10.1 Use of Full Names with SYSMAN

SYSMAN now supports longer node names in an environment command. A valid node name can be up to 255 characters long.

Example

```
SYSMAN> SET ENVIRONMENT/NODE=VMS:.ZKO.VMSORG.SYSMAN.CLIENT1
```

In a local cluster, when full names are used instead of SCSNODE name in an environment command, DECnet will be used for communication, even though the systems are clustered. For example, if SYS1 and SYS2 are two systems in a cluster, the following command would result in usage of DECnet services for communication rather than SCS services. Due to this difference, the user will be prompted to enter a remote password.

The following command is issued from SYS1 to SYS2. SYS1 and SYS2 are assumed to be clustered and are running DECnet/OSI for OpenVMS.

```
SYSMAN> SET ENVIRONMENT/NODE=VMS:.ZKO.SYS2
Remote Password:
%SYSMAN-I-ENV, current command environment:
        Individual nodes: VMS:.ZKO.SYS2
        At least one node is not in local cluster
        Username SYSTEM  will be used on nonlocal nodes
```

For more information, see the *OpenVMS System Manager's Manual*.

3.11 Two Proxy Authorization Files, NETPROXY.DAT and NET$PROXY.DAT

Proxy authorization file information that was shown in OpenVMS Version 6.1 manuals as VAX specific also applies to Alpha Version 6.2 systems.

3.11.1 Creation of Proxy Authorization Files

OpenVMS Alpha Version 6.2 offers two proxy authorization files, NETPROXY.DAT and NET$PROXY.DAT, which are created with the following default protections:

```
NETPROXY.DAT    S:RWED, O:RWED, G:RWE, W
NET$PROXY.DAT   S:RWED, O, G, W
```

The primary proxy database that the system uses is the NET$PROXY.DAT file. NETPROXY.DAT is maintained for:

- Use by DECnet for OpenVMS
- Backward compatibility

3.11.2 Proxy Modifications

To properly maintain proxy database consistency across all members of a mixed-version VMScluster system, you must perform all proxy modifications from either of the following:

- A VAX Version 6.1 (or later) system
- An Alpha Version 6.2 system

This restriction ensures that the NET$PROXY.DAT database is updated with correct proxy information.

For more information about proxy accounts, see the *OpenVMS Guide to System Security*.

3.12 Security Server Process

The Security Server process, which is created as part of normal operating system startup, performs the following tasks:

- Creates and manages the system's intrusion database
- Maintains the network proxy database file (NET$PROXY.DAT)

The intrusion database is used by the system to keep track of failed login attempts. This information is scanned during process login to determine if the system should take restrictive measures to prevent access to the system by a suspected intruder. The contents of this database may be examined by the DCL SHOW INTRUSION command. Information can be deleted from the database using the DCL DELETE/INTRUSION command.

The network proxy database file (NET$PROXY.DAT) is used during network connection processing to determine if a specific remote user may access a local account without using a password. The information contained in this database is managed by the Authorize utility.

The following example shows the expanded expiration time field in the new SHOW INTRUSION output:

```
$ SHOW INTRUSION
Intrusion       Type       Count         Expiration         Source
   NETWORK      SUSPECT       1   21-OCT-1994 12:41:01.07  DEC:.ZKO.TIDY::SYSTEM
```

For more information, see the *OpenVMS Guide to System Security*.

3.13 New LAT System Management Features

The OpenVMS VAX Version 6.2 operating system includes new features that have been added to the SET HOST/LAT command, the LAT Control Program (LATCP) utility, and the programming environment.

Because these features were added to the OpenVMS AXP Version 6.1 operating system, see the following documents for more detailed information:

- *OpenVMS System Manager's Manual*, for LAT system management information
- *OpenVMS System Management Utilities Reference Manual*, for new LATCP commands and qualifiers
- *OpenVMS DCL Dictionary*, for information about the SET HOST/LAT command
- *OpenVMS I/O User's Reference Manual*, for information about new LAT programming features

3.14 New /NO_SHARE Qualifier for LICENSE MODIFY Command

This new qualifier lets you add the NO_SHARE option to a PAK registered in a license database (LDB). The NO_SHARE PAKs are assigned to a single node in a VMScluster system. A NO_SHARE PAK cannot be shared with other VMScluster nodes.

The use of this qualifier remedies problems that occasionally occur when you attempt to use the PAK of a software product for which you already have other PAKs in your LDB. The PAK does not combine with the other PAKs for the same software product, resulting in LICENSE-W-NOCOMB warnings. Often, the license is not loaded on the nodes you want it loaded on.

To remedy this problem, do the following:

1. Add the NO_SHARE option to the PAK or PAKs causing the NOCOMB warnings
2. Assign each PAK to a specific VMScluster node

For more information, see the *OpenVMS License Management Utility Manual*.

3.15 Choosing Languages and Date/Time Formats

Beginning with OpenVMS Version 6.2, system managers can select the time and date formats for many SHOW commands from a predefined list or define new time and date formats.

Note

The SHOW TIME command does not include this feature because the SHOW TIME command is processed completely by DCL, which does not have access to the LIB$ routines that are needed to format the output.

In addition, the SHOW commands for batch and print operations were modified to include, in the default timestamp, seconds as well as hours and minutes. These new features were not previously documented.

For example, rather than 15-JAN-1995 10:16:25.14, you can use a different format, such as the following:

```
$ SHOW USERS

      OpenVMS User Processes at JANUARY 15, 1995 10:16 AM
    Total number of users = 7,  number of processes = 11

 Username     Node     Interactive  Subprocess   Batch
 MCDERMOT    ARD26B            1
 PASTERNAK   ARD26B            -         2         1
   .
   .
   .
```

Users can override the system defaults set up by the system manager and select their own date and time formats.

System managers can also specify languages other than English. From the list that the system manager defines, users can later select a language that they want displayed.

For more information, see the *OpenVMS System Manager's Manual*.

3.16 Operator Communication Manager (OPCOM) Changes

The following Operator Communication Manager (OPCOM) changes appear in the OpenVMS Version 6.2 release:

- Selected operator classes are enabled, depending on whether or not the operator log file is open
- OPC$V_OPR_* replaces OPR$V_*
- OPC$OPA0_ENABLE is used to override values for workstations
- New field names are used

For more information, see the *OpenVMS System Manager's Manual*.

3.16.1 Enabling Selected Operator Classes

In previous versions of the OpenVMS operating system, after the initial creation of the OPERATOR.LOG file, any subsequent REPLY/LOG command created a new log file with all of the operator classes enabled.

In OpenVMS Version 6.2, this functionality has changed:

- If a log file is already open, the list of classes is preserved and enabled on the newly created log file.
- If a log file is not open, the value of the logical OPC$ENABLE_LOGFILE_CLASSES is used. If that logical does not exist, all classes are enabled on the new log file.

3.16.2 OPC$V_OPR_* Replaces OPR$V_*

The bit definitions OPR$V_*, which were loosely based on the operator classes, are now obsolete. (These definitions were never documented.) OPC$V_OPR_* replaces these definitions in OpenVMS Version 6.2.

3.16.3 Using OPC$OPA0_ENABLE to Override Values for Workstations

You can now use the OPC$OPA0_ENABLE logical to override the values for the symbols for a workstation in a cluster. In previous versions of the OpenVMS operating system, if a system was defined as a workstation in a cluster, the OPA0 device was set to NOBROADCAST.

In OpenVMS Version 6.2:

- For workstations in a cluster, you can define the OPC$OPA0_ENABLE logical as TRUE to override NOBROADCAST and thus set the OPA0 device to BROADCAST.
- For all systems that are not workstations in clusters, you can define the logical OPC$OPA0_ENABLE as FALSE to set the OPA0 device to NOBROADCAST.

3.16.4 Using New Field Names

In OpenVMS Version 6.2, the following fields have new names (although you can still use the old names to refer to the fields):

- Change OPC$B_MS_TARGET to OPC$Z_MS_TARGET_CLASSES

  This is a three-byte field that stores the bit mask of target operator classes.
- Change OPC$B_MS_ENAB to OPC$Z_MS_ENAB_TERMINALS

  This is a value that indicates whether the given classes should be disabled (zero) or enabled (nonzero).

3.17 Invalid Symbiont Message Codes

The invalid symbiont message errors logged to the operator console and operator log file now contain integer values that identify why the symbiont message was rejected. There are two types of symbiont message errors logged:

- Informational invalid symbiont messages
- Messages that cause the queue manager to abort the symbiont

3.17.1 Informational Invalid Symbiont Messages

The informational invalid symbiont messages that are logged to the operator console are listed in Table 3–5. When the queue manager logs these messages, the contents of the message causing the error are ignored. The queue manager continues to accept messages from the symbiont.

In the following example, the queue manager signals that the symbiont sent a STOP_STREAM request at an inappropriate time for queue TEST_QUEUE:

```
%%%%%%%%%%%  OPCOM  ss-MMM-ddd hh:mm:ss.hh  %%%%%%%%%%%
Message from user QUEUE_MANAGE
%QMAN-I-INVSMBMSG2, invalid data 10 in message from symbiont on queue TEST_QUEUE
```

Table 3–5 Informational Invalid Symbiont Message Type Definitions

Value	Description
00	Error processing symbiont message item list.
01	Job in symbiont message is not valid for queue, or job is not active. The job does exist in the queue database.
02	Unused.
03	Job state is starting but symbiont message received was not START_TASK.
04	Symbiont device status requesting that tasks be paused when queue state is not pausing, busy, or available.
05	Symbiont device status requesting that queue be stalled when queue is not busy or available.
06	Symbiont PAUSE_TASK message received when queue state is not pausing.
07	Symbiont RESUME_TASK message received when queue state is not resuming.
08	Symbiont START_STREAM message received when queue state is not starting.
09	Unused.
10	Symbiont stopped the stream when the queue state is not stopping.

3.17.2 Messages That Cause the Queue Manager to Abort the Symbiont

Invalid symbiont messages that cause the queue manager to request the symbiont to abort are listed in Table 3–6. The queue manager returns a JBC$_INVMSG status in the iosb of the $SNDJBC SYMBIONT_SERVICE request that generated the error. The PRTSMB or LATSYM process will exit, producing a dump file. For example, if a start queue requiring that a new symbiont process be created is immediately followed by a stop queue for the same queue, an invalid message type number 53 might be logged, as in the following example:

```
%%%%%%%%%%%  OPCOM  ss-MMM-ddd hh:mm:ss.hh  %%%%%%%%%%%
Message from user QUEUE_MANAGE
%QMAN-I-INVSMBMSG2, invalid data 53 in message from symbiont id 4
```

Table 3–6 Invalid Symbiont Message Definitions

Value	Description
51	Symbiont message SMBMSG$K_STRUCTURE_LEVEL field is invalid or message length less than SMBMSG$S_REQUEST_HEADER
52	Invalid symbiont id number in message header

(continued on next page)

Table 3–6 (Cont.) Invalid Symbiont Message Definitions

Value	Description
53	All queues were stopped before message received; the symbiont is expected to terminate
54	The current incarnation of the queue manager did not create the symbiont process sending the message (might be seen during failovers)
55	The symbiont sent a duplicate START_SYMBIONT message to the queue manager
56	The maximum streams in the START_SYMBIONT message is out of range
57	The stream number in the message is greater than maximum allowable streams, or the PID of the symbiont sending the message is unknown to the queue manager

3.18 Erasing Old Home Blocks

When you initialize a volume, the initialize operation may not erase old home blocks. These are home blocks that were created by previous initialize operations.

If a volume that has old home blocks is damaged, you may not be able to recover the volume.

You can now erase old home blocks manually, using the new /HOMEBLOCKS qualifier on the ANALYZE/DISK_STRUCTURE command as follows:

```
$ ANALYZE/DISK_STRUCTURE/REPAIR/HOMEBLOCKS
```

Note that this operation can take up to 30 minutes to complete.

For further information, see the *OpenVMS System Manager's Manual* and the *OpenVMS System Management Utilities Reference Manual*.

3.19 Additional SHUTDOWN.COM Logicals

Two new logicals in the SHUTDOWN.COM command procedure allow system managers to determine the time when DECnet and queues are shut down:

Logical Name	Description
SHUTDOWN$DECNET_MINUTES	Defines the number of minutes remaining before shutdown when DECnet is shut down; must be defined with the /SYSTEM qualifier. The default is 6 minutes.
SHUTDOWN$QUEUE_MINUTES	Defines the number of minutes remaining before shutdown when queues are shut down; must be defined with the /SYSTEM qualifier. The default is 1 minute.

For more information about SHUTDOWN.COM, see the *OpenVMS System Manager's Manual*.

3.20 Installing or Upgrading from a Running Alpha System

Alpha

Beginning with OpenVMS Alpha Version 6.2, you can install or upgrade from a running OpenVMS Alpha system to a target system disk. To do this, run the same installation procedure that you use to install or upgrade OpenVMS Alpha from the distribution compact disc.

This new feature provides functionality that is similar to that of the command procedure VMSKITBLD.COM, which is no longer supported on Alpha systems.

For more information, see the *OpenVMS Alpha Version 6.2 Upgrade and Installation Manual*. ♦

3.21 OpenVMS Support for TCP/IP Networking

For this release, the OpenVMS operating system supports Transmission Control Protocol/Internet Protocol (TCP/IP) parameters and qualifiers for the DCL commands SET HOST, COPY, and DIRECTORY. These commands invoke the TCP/IP layered software products that access remote terminals, files, and directories.

For more information, see the *TCP/IP Networking on OpenVMS Systems* manual.

3.22 Dump File Off the System Disk

VAX

To enable OpenVMS VAX customers to place the system dump file on a device other than the system disk, a new capability has been added for a restricted set of configurations.

The current mechanism of using the system disk as a dump device will not be otherwise affected by this new capability.

For more information, see the *OpenVMS System Manager's Manual*. ♦

3.23 DECamds

The Digital Availability Manager for Distributed Systems (DECamds) is a real-time monitoring, diagnostic, and correction tool that assists system managers to improve OpenVMS system and VMScluster availability. DECamds is also helpful to system programmer/analysts to target a specific node or process for detailed analysis, and system operators and service technicians to help resolve hardware and software problems.

DECamds simultaneously collects and analyzes system data and process data from multiple nodes and displays the output on a DECwindows Motif display. Based on the collected data, DECamds analyzes, detects, and proposes actions to correct resource and denial issues in real time.

New features for OpenVMS Version 6.2 include:

- Console application support for OpenVMS Alpha systems
- Ability to modify cluster quorum
- Enhanced customization options for Locks, Events, and Disk Volumes
- Ability to search for specific processes by name

For more information, see the *DECamds User's Guide* and the *DECamds Release Notes*.

3.24 Inclusion of Selected Layered Products

OpenVMS is introducing additional software to the OpenVMS CD–ROM distribution kit with the release of OpenVMS Alpha and OpenVMS VAX Version 6.2. The kit has been expanded to include software products that previously required the purchase of additional CD–ROMs. This new packaging strategy simplifies customer purchases and installation processes. In addition, all OpenVMS VAX and OpenVMS Alpha online documentation will now be included on one OpenVMS documentation CD–ROM.

The following products have been added to the OpenVMS Version 6.2 CD–ROM distribution:

	VAX Version	Alpha Version
POSIX	2.0	2.0
DECwindows Motif	1.2-3	1.2-3
Softwindows	N/A	1.0
SoftPC	4.0	N/A
TCP/IP Services	3.3	3.3
DECmigrate	1.1A	1.1A
DCE Runtime Services	1.3	1.3
DCE Security Server	1.3	1.3
DCE Cell Directory Server	1.3	1.3
DCE Application Developer's Kit	1.3	1.3

These layered software products will continue to be included on the quarterly VAX CONDIST distribution and the Alpha Layered Product Software Library, except for the DCE products, which are available as a separate kit. Customers should continue to purchase these quarterly distributions to have access to the most current software versions. The OpenVMS CD–ROM distribution will contain the layered product version that is available at the time the OpenVMS operating system is released.

The rights to use POSIX and DCE Runtime Services are granted with the OpenVMS VAX and OpenVMS Alpha operating system licenses. All other layered products continue to require a license to run the software.

This new packaging will require changes to the contents of the OpenVMS CD–ROM distribution kits. The new media contents of these kits are as follows:

- OpenVMS Alpha V6.2 CD–ROM H-kit (QA-MT1AA-H8):
  - OpenVMS Alpha Version 6.2 Binary CD–ROM
  - OpenVMS VAX and Alpha Version 6.2 Online Documentation CD–ROM
  - OpenVMS Freeware CD–ROM for VAX & Alpha Systems
  - Alpha Firmware CD–ROM
- OpenVMS VAX V6.2 CD–ROM H-kit (QA-XULAA-H8):
  - OpenVMS VAX Version 6.2 Binary CD–ROM
  - OpenVMS VAX and Alpha Version 6.2 Online Documentation CD–ROM
  - OpenVMS Freeware CD–ROM for VAX & Alpha Systems
- OpenVMS Combined (VAX & Alpha) V6.2 CD–ROM H-kit (QA-MT3AA-H8):
  - OpenVMS Alpha Version 6.2 Binary CD–ROM
  - OpenVMS VAX Version 6.2 Binary CD–ROM
  - OpenVMS VAX and Alpha Version 6.2 Online Documentation CD–ROM
  - OpenVMS Freeware CD–ROM for VAX & Alpha Systems
  - Alpha Firmware CD–ROM

The OpenVMS Freeware CD–ROM has also been added to all OpenVMS VAX V6.2 and OpenVMS Alpha V6.2 distribution kits. This CD–ROM contains free software tools and utilities to aid software developers in both creating applications and managing or using OpenVMS systems.

See Section 2.1 for more information about the OpenVMS Freeware CD–ROM.

See the *OpenVMS Version 6.2 CD–ROM User's Guide* for product directory and documentation information.

4
ProgrammingFeatures

This chapter describes new features relating to application and system programming on this version of the OpenVMS operating system. These features include:

- DECnet/OSI full name support
- LAN programming features (Alpha only)
- LAT programming features
- System services changes and enhancements
- OpenVMS Alpha programming support for SCSI-2 devices
- C++ RTL object library
- New linker option RMS_RELATED_CONTEXT=

4.1 DECnet/OSI Full Name Support

Alpha

DECnet/OSI full names are supported in this version of OpenVMS Alpha. Full names are hierarchically structured node names that can be a maximum of 255 bytes long. Several run-time routines and several system services are provided to support this function. They are documented, respectively, in the *OpenVMS RTL Library (LIB$) Manual* and in the *OpenVMS System Services Reference Manual*.

For each of the full name routines and system services, this documentation currently states that they are available for VAX only. This is no longer true. The documentation will be updated to reflect the availability of full name routines on both OpenVMS Alpha and OpenVMS VAX. ♦

4.1.1 DECnet/OSI Full Name Support Routines

The DECnet/OSI full name support routines that are currently available in LIBRTL on both OpenVMS VAX and on OpenVMS Alpha are:

- LIB$BUILD_NODESPEC
- LIB$COMPARE_NODENAME
- LIB$COMPRESS_NODENAME
- LIB$EXPAND_NODENAME
- LIB$FIT_NODENAME
- LIB$GET_FULLNAME_OFFSET
- LIB$GET_HOSTNAME
- LIB$TRIM_FULLNAME

For more information, see the documentation on each of the routines.

4.2 LAN Programming Features

Alpha

The following sections describe new LAN programming features for OpenVMS Alpha Version 6.2.

4.2.1 New Devices and Drivers

Table 4–1 lists new devices and drivers supported by OpenVMS Version 6.2.

Table 4–1 Supported Communication Devices

Device	Driver
Proteon PROnet-4/16 ISA NIC (DW110)	IRDRIVER
Digital FDDI PCI-bus (DEFPA)	FWDRIVER
Digital Quad Ethernet PCI-bus (DE436)	EWDRIVER
Digital Ethernet PCI-bus (DE434)	EWDRIVER
Digital Ethernet ISA-bus (DE200)	ERDRIVER
Digital Ethernet ISA-bus (DE201)	ERDRIVER
Digital Ethernet ISA-bus (DE202)	ERDRIVER

For more information, see the *OpenVMS I/O User's Reference Manual*.

4.2.2 New LAN Device Names and Devices

Table 4–2 lists the new device names and devices supported by OpenVMS Version 6.2.

Table 4–2 LAN Device Names and Devices

Device Name	Device	Medium
IRc0:	DW110	Token Ring
FWc0:	DEFPA	FDDI
EWc0:	DE436	Ethernet
EWc0:	DE434	Ethernet
ERc0:	DE200	Ethernet
ERc0:	DE201	Ethernet
ERc0:	DE202	Ethernet

For more information, see the *OpenVMS I/O User's Reference Manual*.

4.2.3 New Controller Characteristics

Table 4–3 lists the new Ethernet controller and its device characteristics when using the Get Device/Volume Information (SYS$GETDVI) system service with the DVI$_DEVTYPE and DVI$_DEVCLASS items.

Table 4–3 Ethernet Controller Device Characteristics

Ethernet Controller	Device Type	Device Class Name
DW110	DT$_IR_DW300	DC$_SCOM
DEFPA	DT$_FW_DEFPA	DC$_SCOM
DE200	DT$_ER_DE422	DC$_SCOM
DE201	DT$_ER_DE422	DC$_SCOM
DE202	DT$_ER_DE422	DC$_SCOM
DE436	DT$_EW_DE435	DC$_SCOM
DE434	DT$_EW_DE435	DC$_SCOM

For more information, see the *OpenVMS I/O User's Reference Manual*.

4.2.4 New Procedures for Configuring ISA Devices

There are two ways to configure an ISA LAN device on OpenVMS. They are as follows:

- Create an entry in the SYS$MANAGER:ISA_CONFIG.DAT file.
- Use the isacfg command at the console prompt (>>>). This requires the use of SYSMAN IO CONNECT.

If you choose to create an entry in the SYS$MANAGER:ISA_CONFIG.DAT file, see SYS$MANAGER:ISA_CONFIG.TEMPLATE for an example. To use the isacfg command at your console prompt, see the hardware documentation associated with your system for more information.

Regardless of which configuration method you choose, you should become familiar with the concepts listed in Table 4–4. See your LAN hardware documentation for configuring information on how to set the jumper settings for those features listed in Table 4–4.

Table 4–4 ISA Configuring Concepts

Concept	Explanation
ISA slot number	Also called node. The ISA slot number that the device is in. The *OpenVMS Alpha Device Support Manual* describes how to number ISA slots.
IRQ	Interrupt request line. Used to establish the interrupt level. Boards support IRQs from 1 to 15. Since OpenVMS does not support shared IRQs, every ISA device must have its own IRQ value reserved for it by the console ISACFG utility.
Port address	I/O base control and status register (CSR) address. Boards usually have I/O addresses associated with their CSR locations. These locations must not be in use by any other device on the system.
Memory address	Also called shared memory. This address range is used to share memory resident on the adapter card between the host CPU and the LAN device on the adapter card. If the device uses shared memory, the I/O addresses for accessing this memory must be reserved. These locations must not be used by any other device on the system.
DMA channel	If the device performs DMA (either slave or bus mastering), a DMA channel is required. Channels are numbered 1 to 7, but channel 4 is always reserved for the system. See your adapter documentation for DMA channels supported by the device.

4.2.4.1 OpenVMS LAN Devices Requiring Configuration

The DE202 Ethernet and DW110 Token Ring devices require configuring. The following information and examples show how this is done.

4.2.4.1.1 DE202 Ethernet

The DE202 is a shared memory Ethernet device. To configure the jumper settings on the board, see the adapter documentation. Example 4–1 and Example 4–2 show how to configure the OpenVMS software to use the DE202. The examples illustrate a configuration of:

- Slot 1
- IO Base at %x300
- IRQ 5
- Shared memory at %xd0000 with length of %x10000

Example 4–1 Using the isacfg Command at the Console Prompt

```
>>> isacfg -slot 1 -etyp 1 -ena 1 -irq0 5 -iobase0 %x300
          -membase0 %xd0000 -memlen0 %x10000 -handle "DE20" -mk
```

The -mk command makes an isacfg entry for an ISA device at slot 1. It is a single-port type of device (-etyp 1). The -handle parameter tells the operating system what type of device it is.

Example 4–2 Using the SYS$MANAGER:ISA_CONFIG.DAT Entry

```
[ERA0]
NAME=ER
DRIVER=SYS$ERDRIVER
NODE=1                       ; plugged into ISA Option slot 1
IRQ=5
PORT=(300:f)                 ; 15 bytes starting at 300
MEM= (D0000:10000)           ; 64 Kbytes starting at D0000
```

4.2.4.1.2 DW110 Token Ring

The DW110 is a bus mastering DMA device on the ISA bus. In addition to setting up the ISA I/O parameters, you may configure ring speed (4 or 16 Mb) and media (UTP or STP). By using LANCP, you can also configure ring speed and media during system startup. Example 4–3 and Example 4–4 show how to configure the OpenVMS software to use the DW110. The examples illustrate a configuration of:

- Slot 4
- IRQ 10
- DMA channel 7
- Base %x4e20
- Shielded twisted pair (STP)
- Ring speed of 16

Example 4–3 Using the isacfg Command at the Console Prompt

```
>>> isacfg -slot 4 -etyp 1 -ena 1 -irq0 %xa -dmachan0 7
          -iobase0 %x4e20 -handle "DW11,STP,16" -mk
```

The -mk command makes an isacfg entry for an ISA device at slot 4. It is a single-port type of device (-etyp 1). The -handle parameter tells the operating system that this is a DW110, that STP media is to be used, and that the ring speed is 16.

Example 4–4 Using the SYS$MANAGER:ISA_CONFIG.DAT Entry

```
[IRA0]
NODE=4
NAME=IR
DRIVER=SYS$IRDRIVER
IRQ=A
DMA=7
PORT=(4E20:20)
USER_PARAM="STP,16"
```

For more information, see the *OpenVMS Version 6.2 Release Notes*, Appendix A. ♦

4.3 LAT Programming Features

Several LAT programming features are new for OpenVMS VAX Version 6.2, but they were introduced with OpenVMS AXP Version 6.1. They include:

- LAT $QIO function modifier IO$M_LT_QUE_CHG_NOTIF
- LAT SENSEMODE and SETMODE argument LAT$C_ENT_QUEUE_ENTRY

- LAT SENSEMODE item codes for port node counters, node counters, protocol errors, service counters subblocks, port entities, and queue entries
- LAT SETMODE item codes for node entities, service type, and port type

For more information, see the *OpenVMS I/O User's Reference Manual*.

4.4 System Services Changes and Enhancements

This section contains information about new features for system services.

4.4.1 Intrusion Database System Services

Table 4–5 describes the new OpenVMS Alpha Version 6.2 system services for accessing the intrusion database:

Table 4–5 Intrusion Database System Services

System Service	Function
$DELETE_INTRUSION	Searches for and deletes all records in the intrusion database matching the caller's specifications
$SCAN_INTRUSION	Scans the intrusion database for suspects or intruders during a login attempt, audits login failures and updates records, or adds new records to the intrusion database
$SHOW_INTRUSION	Searches for and returns information about records in the intrusion database matching the caller's specifications

For details about these services, see the OpenVMS Version 6.1 *OpenVMS System Services Reference Manual*. (These system services are already available on OpenVMS VAX.)

4.4.2 Proxy Database System Services

Table 4–6 describes the new OpenVMS Alpha Version 6.2 system services for accessing the proxy database:

Table 4–6 Proxy Database System Services

System Service	Function
$ADD_PROXY	Adds a new proxy to or modifies an existing proxy in the proxy database
$DELETE_PROXY	Deletes an existing proxy, or removes the default user or a local user from an existing proxy in the proxy database
$DISPLAY_PROXY	Returns information about one or more existing proxies
$VERIFY_PROXY	Verifies that a proxy exists and returns a valid local user for the caller to use to create a local login

For details about these services, see the OpenVMS Version 6.1 *OpenVMS System Services Reference Manual*. (These system services are already available on OpenVMS VAX.)

4.4.3 Cluster Event Notification System Services

Table 4–7 describes the new OpenVMS VAX Version 6.2 system services for detection of cluster membership changes:

Table 4–7 Cluster Event Notification System Services

System Service	Function
$CLRCLUEVT	Removes one or more notification requests previously established by a call to $SETCLUEVT
$SETCLUEVT	Establishes a request for notification when a VMScluster configuration event occurs
$TSTCLUEVT	Simulates the occurrence of a cluster configuration event to test the functionality of the notification AST

For details about these services, see the OpenVMS Version 6.1 *OpenVMS System Services Reference Manual*. (These system services are already available on OpenVMS Alpha.)

4.4.4 Persona System Services

In the past, a privileged server had to explicitly verify that a client's access to an OpenVMS object was allowed by the site-defined security policy. The server process used privileges to access the OpenVMS objects, then used $CHCKPRO or $CHECK_ACCESS to determine a client's access rights to an object. The following three problems resulted from this approach:

- It was difficult to duplicate the security checks performed by the base system components.
- Superfluous security auditing messages resulted from the server's privileged access to the host system files.
- The performance of the server was adversely affected by having to perform secondary protection checking.

The persona system services allow programmers to eliminate the extraneous audits and the cost of secondary protection checks, and to determine access with the normal OpenVMS access tests.

The three new services are $PERSONA_ASSUME, $PERSONA_CREATE, and $PERSONA_DELETE.

4.4.4.1 $PERSONA_ASSUME

The $PERSONA_ASSUME system service modifies the context of the current process to match the context of a given persona. This service allows an OpenVMS process to assume the identity of another user or to discard a persona to return the process to its original state.

Format

SYS$PERSONA_ASSUME persona, [flags]

Arguments

persona

OpenVMS usage:	integer
Type:	longword (unsigned)
Access:	read
Mechanism:	by reference

Address of a longword in which the persona handle is expected.

If the value of the context passed is 1, then the current persona is discarded and the state of the calling process is returned to the state that existed prior to the first call to $PERSONA_CREATE.

flags

OpenVMS usage:	mask_longword
Type:	longword (unsigned)
Access:	read only
Mechanism:	by value

Flag mask specifying which persona services options are to be employed when the persona is assumed. This argument is ignored when a persona is being discarded.

The following table describes each flag:

Flag	Description
IMP$M_ASSUME_SECURITY	Assume access rights, UIC, authorized privileges, user name, and security audit flag
IMP$M_ASSUME_ACCOUNT	Assume OpenVMS account
IMP$M_ASSUME_JOB_WIDE	Assume the new persona, even in a multiprocess job

Description

When assuming a persona using the IMP$M_ASSUME_SECURITY option, any previously enabled image privileges will be disabled. The caller's process will have only the privileges of the impersonated user enabled. These privileges are enabled in the Current, Process, and Authorized privilege masks.

When using IMP$M_ASSUME_SECURITY, access to the job logical name table may no longer be possible since the table is protected by the UIC of the user on whose behalf the current process was created. Also, a new access to the process' controlling terminal may fail, and the process may be in a different default resource domain for locking.

Any persona is automatically discarded and deleted upon image exit. Hence, it is not possible to permanently change the persona of a process using $PERSONA_ASSUME.

The arguments are read in caller's mode, so an invalid argument may cause an access violation to be signaled.

Required Access or Privileges

None.

Required Quota

None.

Related Services

$PERSONA_CREATE, $PERSONA_DELETE

Condition Values Returned

SS$_NORMAL	The service completed successfully.
IMP$_NOCHJIB	The Job Information Block cannot be modified.
IMP$_PERSONANONGRATA	Invalid persona argument.

4.4.4.2 $PERSONA_CREATE

The $PERSONA_CREATE system service creates a persona that may be assumed using the $PERSONA_ASSUME service.

Format

SYS$PERSONA_CREATE persona, usrnam, flags

Arguments

persona

OpenVMS usage:	integer
Type:	longword (unsigned)
Access:	write
Mechanism:	by reference

Address of a longword into which the persona handle is written.

usrnam

OpenVMS usage:	char_string
Type:	character coded text string
Access:	read only
Mechanism:	by descriptor (fixed-length descriptor)

Name of the user to be impersonated.

The **usrnam** argument is the address of a descriptor pointing to a character string containing the user name. The string may contain a maximum of 12 alphanumeric characters.

flags

OpenVMS usage:	mask_longword
Type:	longword (unsigned)
Access:	read only
Mechanism:	by value

Flag mask specifying which persona services options are to be employed when the persona is created.

The following table describes each flag:

Flag	Description
IMP$M_ASSUME_DEFPRIV	Create a persona with only default privileges
IMP$M_ASSUME_DEFCLASS	Create a persona with default classification

Description

On calling the $PERSONA_CREATE system service, the required information concerning the OpenVMS user specified by the **usrnam** argument is read from the user authorization file and rights database, and is stored in system memory. A handle to refer to the created persona is returned in the **persona** argument.

It is not possible to create a persona for a user name that has been disabled.

No changes are made to the caller’s process as a result of calling $PERSONA_CREATE.

Some of the $PERSONA_CREATE service executes in the caller's access mode (assumed to be user mode). Improper use of the **usrnam** argument may cause an access violation to be signaled.

Required Access or Privileges

All calls to $PERSONA_CREATE require DETACH privilege and access to the system authorization database.

Required Quota

None.

Related Services

$PERSONA_ASSUME, $PERSONA_DELETE

Condition Values Returned

SS$_NORMAL	The service completed successfully.
SS$_ACCVIO	The **persona** argument cannot be written by the caller.
SS$_NODETACH	Operation requires DETACH privilege.
SS$_INSFMEM	Insufficient memory.
IMP$_USERDISABLED	User name disabled.

Any condition value returned by $LKWSET, $GETUAI, or $FIND_HELD may also be returned.

4.4.4.3 $PERSONA_DELETE

The $PERSONA_DELETE system service deletes a persona created using the $PERSONA_CREATE service.

Format

SYS$PERSONA_DELETE persona

Arguments

persona

OpenVMS usage:	integer
Type:	longword (unsigned)
Access:	read
Mechanism:	by reference

Address of a longword in which the persona handle is expected.

Description

The $PERSONA_DELETE service frees the resources used by the persona. No changes to the caller's process are made as a result of calling $PERSONA_DELETE.

Required Access or Privileges

None.

Required Quota

None.

Related Services

$PERSONA_ASSUME, $PERSONA_CREATE

Condition Values Returned

SS$_NORMAL	The service completed successfully.
SS$_INUSE	Persona is in use; it cannot be deleted.

4.5 OpenVMS Alpha Support for SCSI-2 Devices

Alpha

On Alpha systems, OpenVMS now supports the **tagged command queuing** functionality of the SCSI-2 standard. Tagged command queuing allows a SCSI class driver to pass multiple queued I/O requests directly to a port driver without waiting for any one I/O request to complete. The port driver then sends each I/O request directly to the target SCSI device, where the device queues the request on its internal command device queues.

Tagged command queuing functionality increases overall disk I/O performance. Because a device knows which I/O requests are in progress, it can optimize the order in which it handles the requests based on its own hardware capabilities (dynamic positioning and latency schedules, and spindle configuration).

The OpenVMS Alpha implementation includes tagged command queuing support only on DISK class devices on the SCSI port drivers shown in Table 4–8.

Table 4–8 OpenVMS Alpha Tagged Command Queuing Support

Port Driver	Adapter	Adapter Classification
PKCDRIVER	NCR 53C94	DEC 3000 Alpha family supporting TURBOchannel
PKTDRIVER	NCR 53C710	DEC 4000 Alpha family supporting Futurebus/Plus
PKEDRIVER	NCR 53C810	Digital Alpha family supporting PCI
PKSDRIVER	KZTSA	SIMport TURBOchannel add-on adapter
	KZPSA	SIMport PCI add-on adapter
PKQDRIVER	KZPBA, P1SE, P2SE	Qlogic PCI SCSI add-on adapter

OpenVMS Alpha continues to support nonqueuing operations for SCSI magnetic tapes, class drivers, and the following SCSI port drivers:

- PKJDRIVER — Adaptec 1742A (EISA)
- PKZDRIVER — KZMSA (XMI)

♦

4.6 C++ RTL Object Library

With OpenVMS Version 6.2, the C++ RTL object modules are available in SYS$LIBRARY:STARLET.OLB. Most C++ programs can include these object modules directly by linking /NOSYSSHR in their link commands. C++ programs that include the Task class must explicitly include the CMA shared library when they link /NOSYSSHR, as in the following example:

```
$ LINK/NOSYSSHR foo,SYS$INPUT:/OPT
SYS$SHARE:CMA$LIB_SHR/SHARE
^Z
```

For more information, see the *DEC C++ Class Library Reference Manual*.

4.7 New Linker Option RMS_RELATED_CONTEXT=

This option enables or disables RMS related name context processing within the linker option file. This is also known as file specification stickiness. The default is to have RMS related name context processing enabled. This default applies at the start of each options file regardless of the setting in a previous options file. The related name context itself (the collection of data structures RMS maintains to supply the most recently specified fields) does not carry over from one linker options file to the next (that is, specified fields in the previous options file are not used to fill in absent fields for file specifications in the current options file).

The format is as follows:

```
RMS_RELATED_CONTEXT=YES/NO
```

Option Values:

- YES

  Enables RMS related name context processing starting with the context previously saved by a RMS_RELATED_CONTEXT=NO command. If RMS related name context processing is already enabled, this option has no effect.

- NO

  Disables RMS related name context processing. When RMS related name context processing is disabled, the current context is saved for a possible future RMS_RELATED_CONTEXT=YES option. If RMS related name context processing is already disabled, specifying RMS_RELATED_CONTEXT=NO has no effect.

When RMS related name processing is enabled (by default at the beginning of each options file), file specifications that do not have all fields of the file specification present will have the missing fields replaced with the corresponding fields most recently specified in earlier file specifications. When disabled, fields in the file specification that are absent are not replaced with corresponding fields of previous file specifications. When RMS related name processing is disabled, the current related name context is saved. When RMS related name processing is enabled via this option, the saved related name context is restored.

For more information about linker options, see the *OpenVMS Linker Utility Manual*.

5
DEC C XPG4 Localization Utilities

OpenVMS Version 6.2 provides XPG4-compliant utilities for managing localization data for international software applications or layered products. Localization data is defined separately from the application and is bound to it only at run time.

Note

> The addition of the DEC C XPG4 code to OpenVMS Version 6.2 means that developers who link their code on OpenVMS VAX Version 6.2 will not be able to run it on OpenVMS VAX Version 6.1.

The following localization utilities are described in this chapter:

- GENCAT
- ICONV COMPILE
- ICONV CONVERT
- LOCALE COMPILE
- LOCALE LOAD
- LOCALE UNLOAD
- LOCALE SHOW CHARACTER_DEFINITIONS
- LOCALE SHOW CURRENT
- LOCALE SHOW PUBLIC
- LOCALE SHOW VALUE

These utilities are provided only on CD–ROM.

Because these utilities support the XPG4 model of internationalization, they are only useful for localizing applications written to that model. See the user documentation for each application or layered product to see if it supports XPG4 internationalization.

This chapter also describes the locale file format and the character set description (charmap) file.

5.1 GENCAT

The GENCAT command merges one or more message text source files into a message catalog file.

5.1.1 Format

GENCAT msgfile[,...] catfile

5.1.1.1 Parameters

msgfile

The file specification of a message text source file. The default file type for a message text source file is .MSGX.

catfile

The file specification of the message catalog file that is created. If *catfile* already exists, a new version is created that includes the messages in the existing catalog. The file type for a message catalog file is .CAT.

5.1.1.2 Qualifiers

None.

5.1.2 Description

The GENCAT command creates new message catalogs from one or more input source files and an existing catalog file (if there is one). A message catalog is a binary file containing the messages for an application. This includes all messages that the application issues, such as error messages, screen displays, and prompts. Applications retrieve messages from a message catalog using the `catopen`, `catgets`, and `catclose` C run-time library routines. See the *DEC C Run-Time Library Reference Manual for OpenVMS Systems* for details of these routines.

A message text source file is a text file that you create to hold messages printed by your program. You can use any text editor to enter messages into the text source file. Messages can be grouped into sets, usually to represent functional subsets of your program. Each message has a numeric identifier, which must be unique within its set. The message text source file can also contain commands recognized by GENCAT for manipulating sets and individual messages.

If a message catalog with the name *catfile* exists, GENCAT creates a new version of the file that includes the contents of the older version and then modifies it. If the catalog does not exist, GENCAT creates the catalog with the name *catfile*.

You can specify any number of message text source files. The GENCAT command processes multiple source files one after the other in the sequence that you specify them. Each successive source file modifies the catalog.

The *catfile* can contain the following commands:

> *message_number text*
>
> Inserts *text* as a message with the identifier *message_number*. Follow these guidelines:
>
> - Numbers must be ascending within each set. You can skip a number, but you cannot go back to add a missing number or replace an existing number during a GENCAT session.
> - If the message text is empty and a space or tab field separator is present, an empty string is stored in the message catalog.
> - If a message source line has a message number but neither a field separator nor message text, the existing message with that number (if any) is deleted from the catalog.

$delset *set_number*

Deletes the set of messages indicated by *set_number*.

$quote *character*

Sets the quote character to *character*. See the Examples section for more information.

$set *set_number*

Indicates that all messages entered after this command are placed in the set indicated by *set_number*. You can change the set by entering another $set command. However, set numbers must be entered in ascending order; you cannot go back to a lower numbered set during the GENCAT session. If the command is not used, the default set number is 1.

Each initial keyword or number must be followed by white space. The GENCAT utility ignores any line that begins with a space, a tab, or a dollar sign ($) character followed by a space, a tab, or a newline character. Therefore, you can use these sequences to start comments in your *catfile*. Blank lines are also ignored. Finally, you can place comments on the same line after the $delset, $quote, or $set commands because GENCAT ignores anything that follows these commands.

A line beginning with a digit marks a message to be included in the catalog. You can specify any amount of white space between the message ID number and the message text; however, when the message text is not delimited by quotation marks, one space or tab character is recommended. When message text is not in quotation marks, GENCAT treats additional white space as part of the message. When message text is enclosed in quotation marks, GENCAT ignores all spaces or tabs between the message ID and the first quotation character.

Escape sequences like those recognized by the C language can be used in text. The escape character (\), a backslash, can be used to insert special characters in the message text. See Table 5–1.

Table 5–1 Special Characters

Escape Sequence	Character
\ n	New Line
\ t	Horizontal Tab
\ v	Vertical Tab
\ b	Backspace
\ r	Carriage Return
\ f	Form Feed
\ \	Backslash Character (\ ). Use to continue message text on the following line.

(continued on next page)

Table 5–1 (Cont.) Special Characters

Escape Sequence	Character
\ *ddd*	The single-byte character associated with the octal value *ddd*. You can specify one, two, or three octal digits. However, you must include leading zeros if the characters following the octal digits are also valid octal digits; for example, the octal value for $ (dollar sign) is 44. To insert $5.00 into a message, use \ 0445.00, not \ 445.00; otherwise the 5 is parsed as part of the octal value.

Notes

GENCAT conforms to X/Open specifications. In an X/Open conforming application, the set numbers must be integers in the range of 1 to NL_SETMAX, inclusive; message numbers must be integers in the range of 1 to NL_MSGMAX, inclusive. NL_SETMAX and NL_MSGMAX are defined in the `<limits.h>` header file that comes with DEC C and DEC C++. For OpenVMS Version 6.2, each of these limits is 65535.

The value of LC_CTYPE from the LOCALE SHOW CURRENT command determines the interpretation of message text in the message source files *msgfile*....

5.1.3 Errors

When GENCAT reports an error, no action is taken on any commands and an existing catalog is left unchanged.

5.1.4 Examples

1.

```
$set 10 Communication Error Messages
```

This example uses the $set command in a source file to assign a set number to a group of messages.

The message set number is 10. All messages after the $set command and up to the next $set command are assigned a message set number of 10. (Set numbers must be assigned in ascending order but they need not be contiguous.) You can include a comment in the $set command.

2.

```
$delset 10 Communication Error Messages
```

This example uses the $delset command to remove from a catalog all messages belonging to the specified message set (10, in this case).

The $delset command must be placed in the proper set number order with respect to any $set commands in the same source file. You can include a comment in the $delset command.

3.

```
12 "file removed"
```

This example shows how to enter the message text and assign a message ID number to it. In this case, a message ID of 12 is assigned to the text that follows it.

You must leave at least one space or tab character between the message ID number and the message text but you can include more spaces or tabs if you prefer. If you do include more spaces or tabs, they are ignored when the message text is in quotation marks and they are considered part of the text when the message text is not in quotation marks.

Message numbers must be in ascending order within a single message set but they need not be contiguous.

All text following the message number and up to the end of the line is included as message text. If you place the escape character (\), a backslash, as the last character on the line, the message text continues on the following line. Consider the following example:

```
This is the text associated with \
message number 5.
```

The two lines in the example define the following single-line message:

```
This is the text associated with message number 5.
```

4.

```
$quote "   Use a double quote to delimit message text
$set 10            Message Facility - Quote command messages
1 "Use the $quote command to define a character \
\n for delimiting message text" \n
2 "You can include the \"quote\" character in a message \n \
by placing a \\ (backslash) in front of it" \n
3 You can include the "quote" character in a message \n \
by having another character as the first nonspace \
\n character after the message ID number \n
$quote
4 You can disable the quote mechanism by \n \
using the $quote command without \n a character \
after it \n
```

This example shows the effect of a quote character.

The $quote command defines the double quote (") as the quote character. The quote character must be the first nonspace character after the message number. Any text following the next occurrence of the quote character is ignored.

This example also shows two ways to include the quote character in the message text:

- Place a \ in front of the quote character.
- Use another character as the first nonspace character after the message number. This disables the quote character for that message only.

This example also shows the following:

- A \ is still required to split a quoted message across lines.
- To display a \ in a message, you must place another \ in front of it.
- You can format your message with a new-line character by using \n.
- If you use the $quote command with no character argument, you disable the quote mechanism.

5.2 ICONV COMPILE

The ICONV COMPILE command creates a conversion table file from a conversion source file. The conversion table file is used by the ICONV CONVERT command to convert characters in a file from one codeset to another.

5.2.1 Format

ICONV COMPILE sourcefile tablefile

5.2.1.1 Parameters

sourcefile

The file specification of the conversion source file. The default file type is .ISRC. The file naming convention Digital uses for conversion source files is:

```
fromcodeset_tocodeset.isrc
```

tablefile

The file specification of the conversion table file to be created. The default file type is .ICONV. The file naming convention for conversion table files is:

```
fromcodeset_tocodeset.iconv
```

You must follow this convention for naming conversion table files for the ICONV CONVERT command to recognize them.

Public conversion table files are in the directory defined by the logical name SYS$I18N_ICONV. Put new conversion table files in the same directory if you want to make them available systemwide.

5.2.1.2 Qualifier

/LISTING[=listfile]

Directs ICONV COMPILE to produce a listing file, which contains the source file listing and any error messages generated during compilation. If the file name is omitted from the qualifier, the default listing file name is *sourcefile*.LIS.

5.2.2 Description

The ICONV commands support any 1- to 4-byte codesets that are state independent. They do not support state-dependent codesets.

Note

There is an implementation restriction in the *tocodeset* encodings in this implementation. The characters in *tocodeset* must not use 0XFF in the fourth byte.

The conversion source file contains the character conversion rules for a specific conversion.

The format of a codeset conversion source file is defined as follows:

```
<fromcodeset_mb_cur_max>    value
<fromcodeset_mb_cur_min>    value
<tocodeset_mb_cur_max>      value
<tocodeset_mb_cur_min>      value
<fallback_code>             value
<escape_char>               value
<comment_char>              value
<fromcodeset_range>         value...value;value...value;...;value...value
ICONV_TABLE
fromvalue                         tovalue
fromvalue                         tovalue
   .                                 .
   .                                 .
   .                                 .
fromvalue                         tovalue
END ICONV_TABLE
```

where the <...> symbols and their associated values are codeset declarations, and the *fromvalue*/*tovalue* pairs are character conversion rules.

Codeset Declarations

The codeset declarations must precede the character conversion rules. Each declaration consists of a symbol, starting in column 1 and including the surrounding brackets, followed by one or more blanks (tabs or spaces), followed by the value to be assigned to the symbol. See Table 5–2.

Table 5–2 Codeset Declarations

Symbol	Value
<fromcodeset_mb_cur_max>	The maximum number of bytes in a character in the fromcodeset. This value defaults to 1.
<fromcodeset_mb_cur_min>	The minimum number of bytes in a character in the fromcodeset. This value must be less than or equal to fromcodeset_mb_cur_max. If this value is not specified, it defaults to the value of fromcodeset_mb_cur_max.
<tocodeset_mb_cur_max>	The maximum number of bytes in a character in the tocodeset. This value defaults to 1.
<tocodeset_mb_cur_min>	The minimum number of bytes in a character in the tocodeset. This value must be less than or equal to tocodeset_mb_cur_max. If this value is not specified, it defaults to the value of tocodeset_mb_cur_max.

(continued on next page)

Table 5–2 (Cont.) Codeset Declarations

Symbol	Value
<fallback_code>	The *tovalue*s for the *fromvalue*s that appear in the <fromcodeset_range> but are not specified between ICONV_TABLE and END ICONV_TABLE. Specify one of three kinds of values: • SAME — specifies that the *tovalue*s are the same as the *fromvalue*s. • ERROR — specifies that the conversion from the *fromvalue* to a *tovalue* is not supported. ICONV CONVERT issues a warning and ignores the rest of the record read. The DEC C run-time library routine iconv returns to the caller with an "illegal character" error. • User-defined *tovalue* — the *fromvalue*s are converted to the specified user-defined *tovalue*. The user-defined *tovalue* can represent a multibyte character with the restriction that 0xff cannot be used as the value in the fourth byte. The settings for user-defined *tovalue*s for <fallback_code> are the same as the settings for character conversion rule values. You can use octal, decimal, or hexadecimal digits. If the <fallback_code> is not specified, it defaults to SAME.
<escape_char>	The escape character used to indicate that subsequent characters are interpreted in a special way. The escape character defaults to backslash (\ ).
<comment_char>	The character that, when placed in column 1 of a line, indicates that the line will be ignored. The default comment character is the number sign (#).
<fromokcodeset_range>	The fromcodeset encoding ranges. Specify this declaration if the fromcodeset is a multibyte codeset. If the fromcodeset is omitted, it defaults to a single-byte codeset and the table created by ICONV COMPILE will support only single-byte fromcodeset conversions.

When specifying codeset encoding ranges for the fromcodeset, every zone of characters must be specified. If any zones of characters are missing from the <fromcodeset_range> specification, the codeset conversion might be incorrect. It is very important to specify the codeset encoding ranges correctly for the fromcodesets supported by the rest of the DEC C run-time library (RTL). If this is not done, the codeset support for iconv and the rest of the DEC C RTL will not be consistent.

For example, the fromcodeset ranges for EUCJP are specified as:

```
<fromcodeset_range>  \x0...\x7f;\x8e\xa1...\x8e\xfe;
                     \xa1\xa1...\xfe\xfe;\x8f\xa1\xa1...\x8f\xfe\xfe
```

The settings for <fromcodeset_range> values are the same as the settings for character conversion rule values. You can use octal, decimal or hexadecimal digits.

Character Conversion Rules

The character conversion rules are all the lines between the string ICONV_TABLE starting in column 1 and END ICONV_TABLE starting in column 1.

Character conversion rules must begin in column 1.

Empty lines and lines containing a comment_char in the first column are ignored. Comments are optional.

Character conversion rules can have one of two forms:

```
fromvalue                   tovalue

fromvalue...fromvalue       tovalue
```

Place one or more blanks (tabs or spaces) between *fromvalue* and *tovalue*.

Use the first format to define a single-character conversion rule. For example:

```
\d32       \d101
\d37       \d106
```

Use the second format to define a range of character conversion rules. In this format, the ending *fromvalue* must be equal to or greater than the starting *fromvalue*. The subsequent *fromvalue*s defined by the range are converted to *tovalue*s in increasing order.

For example, consider the following line:

```
\d223\d32...\d223\d35       \d129\d254
```

This line is interpreted as:

```
\d223\d32        \d129\d254
\d223\d33        \d129\d255
\d223\d34        \d130\d0
\d223\d35        \d130\d1
```

For settings of *fromvalue* and *tovalue*:

- A decimal constant is defined as one, two, or three decimal digits preceded by the escape character and lowercase d. For example: \ d42.
- An octal constant is defined as one, two, or three octal digits preceded by the escape character. For example: \ 141.
- A hexadecimal constant is defined as one or two hexadecimal digits preceded by the escape character and a lowercase x. For example: \ x6a.

Each constant represents a single-byte value. You can represent multibyte values by concatenating two or more decimal, octal, or hexadecimal constants.

Note

When constants are concatenated for multibyte values, they must have the same radix (decimal, octal, or hexadecimal). Only characters in the Portable Character Set can be used to construct conversion source files.

Also see the ICONV CONVERT command.

5.2.3 Errors

If an error is encountered during processing, ICONV COMPILE does not generate an output *tablefile*. If a warning is encountered, a valid table file is created. However, because a warning can indicate a user error, you should check the returned warning messages.

Some ICONV COMPILE error messages and their descriptions follow.

```
%ICONV-E-INVFCSRNG, syntax error in <fromcodeset_range> definition
```

This error occurs when the definition of the <fromcodeset_range> symbol does not conform to the required syntax. The <fromcodeset_range> symbol defines encoding ranges and is required for multibyte codesets.

```
%ICONV-E-INVSYNTAX, invalid file syntax
```

This error occurs when a line in the source does not conform to the required syntax.

```
%ICONV-E-BADTABLE, bad table caused by invalid value for <fromcodeset_range>
definition
```

This error occurs when an invalid value is specified for the codeset encoding ranges. The encoding ranges are defined by the <fromcodeset_range> symbol.

5.2.4 Example

```
$ ICONV COMPILE/LISTING EUCTW_DECHANYU.ISRC EUCTW_DECHANYU.ICONV
```

This example shows how to create a conversion table file to convert the EUCTW codeset to the DECHANYU codeset. The listing file, EUCTW_DECHANYU.LIS, contains a listing of the source file and any error messages generated by the compiler.

5.3 ICONV CONVERT

The ICONV CONVERT command converts characters in a file from one codeset to another codeset. The converted characters are written to an output file.

5.3.1 Format

ICONV CONVERT infile outfile

5.3.1.1 Parameters

infile

The file specification of the file that contains the characters to be converted. The /FROMCODE qualifier specifies the codeset of the characters in this file.

outfile

The file specification of the file created by ICONV CONVERT. The /TOCODE qualifier specifies the codeset of the characters in this file.

5.3.1.2 Qualifiers

/FROMCODE=fromcodeset

A required qualifier that specifies the codeset of the characters in the input file *infile*.

/TOCODE=tocodeset

A required qualifier that specifies the codeset of the characters in the output file *outfile*.

5.3.2 Description

The ICONV CONVERT command converts the characters in *infile* from the codeset identified by the /FROMCODE qualifier to the codeset identified by the /TOCODE qualifier. The converted file is written to *outfile*.

The conversion is done in one of two ways:

- Using a conversion table file to look up the converted characters. This is the default method. The conversion table file naming convention is:

  ```
  fromcodeset_tocodeset.iconv
  ```

  ICONV CONVERT searches your current directory for this file. If it cannot find the file, it searches the system directory defined by the logical name SYS$I18N_ICONV. You can create a conversion table using the ICONV COMPILE command.

  Note that if you add conversion tables to your system, they must use the same file naming convention. Otherwise, the ICONV CONVERT will not recognize them.

- Using an algorithm. This is implemented as a conversion function that is built into the DEC C run-time library. If there is an algorithm available for the specific conversion, ICONV CONVERT uses it instead of the conversion table. You cannot use this method to add codeset converters to your system.

The ICONV commands support any 1- to 4-byte codesets that are state independent. They do not support state-dependent codesets.

5.3.3 Example

```
$ ICONV CONVERT/FROMCODE=EUCTW/TOCODE=DECHANYU FROMFILE.DAT TOFILE.DAT
```

This example shows a conversion from EUCTW characters to DECHANYU characters. The EUCTW characters in the file FROMFILE.DAT are converted to the corresponding DECHANYU characters. The converted characters are stored in the file TOFILE.DAT.

5.4 LOCALE COMPILE

The LOCALE COMPILE command converts a locale source file into a binary locale file. The binary locale file is used by those utilities and C routines that are dependent on the setting of the international environment logical names.

5.4.1 Format

LOCALE COMPILE sourcefile

5.4.1.1 Parameter

sourcefile

The file specification of the locale source file. This file defines each category of the locale. The default file type for the source file is .LSRC. For the definition of the locale source file format, see Section 5.11.

5.4.1.2 Qualifiers

/CHARACTER_DEFINITIONS=filename
/NOCHARACTER_DEFINITIONS (default)

Specifies a character-set description file (charmap) for the locale. This file maps characters to their actual character encodings. If a charmap is not specified, no symbolic names (other than collating symbols defined in a collating symbol keyword) are allowed in the locale source file. For definition of the charmap file format, see Section 5.12. The default file type for a charmap is .CMAP.

/DISPLAY[=[NO]HOLE]

Used with certain Chinese locales and terminals to specify that 4-byte characters occupy four printing positions (columns) on the terminal display. The default value (/DISPLAY=NOHOLE) specifies that 4-byte characters occupy two printing positions.

/IGNORE=WARNINGS
/NOIGNORE (default)

Generates an output file even if LOCALE COMPILE issues warning messages. Use the /IGNORE keyword cautiously because the warnings could indicate user errors that you might want to correct before using the resulting locale file.

/LISTING [=filename] (batch default)
/NOLISTING (interactive default)

Specifies the name of the listing file. The /SHOW qualifier controls the information included in the listing file. If the file name is omitted, the default is *sourcefile*.LIS.

/OUTPUT=[filename]
/NOOUTPUT

Specifies the name of the output file. If the /OUTPUT qualifier is omitted, the default output file name is *sourcefile*.LOCALE. Public locales are stored in the directory defined by the logical name SYS$I18N_LOCALE. If the output file is in any other location, the locale is private.

If /NOOUTPUT is specified, the compiler does not create an output file, even if the compilation is successful.

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

Use /SHOW together with /LIST to control the information included in the listing file. You can specify the following keywords:

Keyword	Description
ALL	Include all information.
BRIEF	Include a summary of the symbol table.
[NO]CHARACTER_ DEFINITIONS	Include or omit the charmap file.
NONE	Do not print any information. If NONE is specified, the listing file only contains the error messages generated.
[NO]SOURCE	Include or omit a listing of the source file.
[NO]STATISTICS	Include or omit compiler performance information.

Keyword	Description
[NO]SYMBOLS	Include or omit a listing of the charmap symbol table.
[NO]TERMINAL	Display compiler messages at the terminal.

The default is /SHOW=(SOURCE,TERMINAL).

5.4.2 Description

Use the LOCALE COMPILE command to add new locales to your system in addition to those supplied by Digital. To compile a locale, LOCALE COMPILE requires two files:

- A charmap file that defines the character set for the locale. If you do not specify a charmap file, symbolic names cannot be specified in the locale source file. If this happens, LOCALE COMPILE issues an error or warning message, depending on the category processed, and no output file is produced. (Also see the /IGNORE qualifier.)
- A locale source file. This file describes one or more of the locale categories: LC_CTYPE, LC_COLLATE, LC_MESSAGES, LC_MONETARY, LC_NUMERIC, and LC_TIME.

5.4.3 Errors

Some LOCALE COMPILE error messages and their descriptions follow.

```
%LOCALE-E-CASEALRDY, case conversion already exists for 'character'
```

Where ‘character‘ is a character from the codeset. This error can occur when the locale compiler is processing the LC_CTYPE category. It indicates that more than one case conversion is specified for ‘character‘.

```
%LOCALE-E-PREOFCMAP, premature end of file in charmap file
```

This error occurs if there is no END CHARMAP statement in the charmap file.

```
%LOCALE-E-PREEOFSRC, premature end of file in source file
```

This error occurs if there is an error with the END statements in the locale source file.

```
%LOCALE-F-NOADDSYM, failed to add symbol to symbol table
```

This error can occur when there is insufficient memory to finish the compilation. Check the amount of memory available to your process.

```
%LOCALE-F-NOINITSYM, failed to initialize symbol table
```

This error may be caused if there is insufficient memory to finish the compilation. Check the amount of memory available to your process.

5.4.4 Example

```
$ LOCALE COMPILE EN_GB_ISO8859-1/CHARACTER_DEFINITIONS=ISO8859-1 -
/LIST/SHOW=(CHARACTER_DEFINITIONS,SYMBOLS,STATISTICS)
```

This example shows how to generate a locale file named EN_GB_ISO8859-1.LOCALE from the source file EN_GB_ISO8859-1.LSRC, using the charmap file ISO8859-1.CMAP. To use this locale file, copy it to the SYS$I18N_LOCALE directory and set the LANG logical to "EN_GB.ISO8859-1". The listing file contains a listing of the charmap file, the symbol table, performance information, and any error messages generated by the compiler.

5.5 LOCALE LOAD

This command loads the specified locale name into the system's memory as shared, read-only global data.

5.5.1 Format

LOCALE LOAD name

5.5.1.1 Parameter

name

A character string that identifies the locale to be loaded. This can be one of the following:

- The name of the public locale

  Specifies the public locale. The format of the name is:

  ```
  language_country.codeset[@modifier]
  ```

  LOCALE LOAD searches for the public locale binary file in the location defined by the logical name SYS$I18N_LOCALE. The file type defaults to .LOCALE. The period (.) and at-sign (@) characters in the name specified are replaced by underscore (_) characters.

  For example, if the name specified is "zh_CN.dechanzi@radical", LOCALE LOAD searches for the following binary locale file:

  SYS$I18N_LOCALE:ZH_CN_DECHANZI_RADICAL.LOCALE

- A file specification

  Specifies the binary locale file. This can be any valid file specification. If either the device or directory is not specified, LOCALE LOAD first applies the current caller's device and directory as defaults. If the file is not found, the device and directory defined by the SYS$I18N_LOCALE logical name are used as defaults. The file type defaults to .LOCALE.

  Wildcards are not allowed. The binary locale file cannot reside on a remote node.

5.5.1.2 Qualifiers

None.

5.5.2 Description

This command loads the specified locale name into the system's memory as several shared, read-only global sections. All processes that access the loaded locale then use this one copy of the locale, thereby reducing overall demand on system memory.

LOCALE LOAD is a privileged OpenVMS command, typically issued by the system manager. The following privileges are required:

- SYSGBL
- PRMGBL

5.6 LOCALE UNLOAD

This command unloads the specified locale name from the system's memory.

5.6.1 Format

LOCALE UNLOAD name

5.6.1.1 Parameter

name

A character string that identifies the locale to be unloaded. See the LOCALE LOAD command for acceptable formats for this parameter.

5.6.1.2 Qualifiers

None.

5.6.2 Description

This command unloads the specified locale name from the system's memory. If a process is accessing the locale when the UNLOAD command is entered, the global sections are deleted after the process deaccesses the locale.

LOCALE UNLOAD is a privileged OpenVMS command, typically issued by the system manager. The following privileges are required:

- SYSGBL
- PRMGBL

Note

Only locale files loaded by the LOCALE LOAD command can be unloaded.

5.7 LOCALE SHOW CHARACTER_DEFINITIONS

This command lists character set description files (charmaps).

5.7.1 Format

LOCALE SHOW CHARACTER_DEFINITIONS

5.7.1.1 Parameters

None.

5.7.1.2 Qualifiers

None.

5.7.2 Description

This command lists all the character set description files (charmaps) in the public directory defined by the logical name SYS$I18N_LOCALE. A charmap defines the symbolic names and values of characters in a coded character set. Charmaps are used by the LOCALE COMPILE command when compiling a locale. A charmap file has the file type .CMAP.

5.7.3 Example

```
$ LOCALE SHOW CHARACTER_DEFINITIONS
[SYS$I18N.LOCALES.SYSTEM]DECHANYU
[SYS$I18N.LOCALES.SYSTEM]DECHANZI
[SYS$I18N.LOCALES.SYSTEM]DECKANJI
[SYS$I18N.LOCALES.SYSTEM]DECKOREAN
[SYS$I18N.LOCALES.SYSTEM]EUCJP
[SYS$I18N.LOCALES.SYSTEM]EUCTW
[SYS$I18N.LOCALES.SYSTEM]ISO8859-1
[SYS$I18N.LOCALES.SYSTEM]ISO8859-2
[SYS$I18N.LOCALES.SYSTEM]ISO8859-3
[SYS$I18N.LOCALES.SYSTEM]ISO8859-4
[SYS$I18N.LOCALES.SYSTEM]ISO8859-5
[SYS$I18N.LOCALES.SYSTEM]ISO8859-7
[SYS$I18N.LOCALES.SYSTEM]ISO8859-8
[SYS$I18N.LOCALES.SYSTEM]ISO8859-9
[SYS$I18N.LOCALES.SYSTEM]MITACTELEX
[SYS$I18N.LOCALES.SYSTEM]SDECKANJI
[SYS$I18N.LOCALES.SYSTEM]SJIS
```

This example shows a system with several charmap files in the SYS$I18N_LOCALE directory.

5.8 LOCALE SHOW CURRENT

This command displays a summary of the current international environment as defined by several international environment logical names.

5.8.1 Format

LOCALE SHOW [CURRENT]

5.8.1.1 Parameters

None.

5.8.1.2 Qualifiers

None.

5.8.2 Description

The LOCALE SHOW CURRENT command lists the settings for each locale category and the values of the environment variables LC_ALL and LANG. The CURRENT keyword is the default and is, therefore, optional. The logical name that defines a category has the same name as the category. For example, the LC_MESSAGES logical name defines the setting for the LC_MESSAGES category. The locale categories are:

Category	Description
LC_COLLATE	Information about collating sequences.
LC_CTYPE	Character classification information.
LC_MESSAGES	Information about the language of program messages and the format of yes/no prompts.
LC_MONETARY	Monetary formatting information.
LC_NUMERIC	Information about formatting numbers.
LC_TIME	Time and date information.

Each locale category is defined by scanning the following logical names in the order shown, until a logical name is found. If the logical name found does not represent a valid locale file, then LOCALE SHOW displays the string "C" for all the categories.

1. LC_ALL
2. Logical names corresponding to the categories specified in the table (for example, if LC_NUMERIC is specified as a valid locale category, the LOCALE SHOW CURRENT command displays the name of the category and the locale name it defines).
3. LANG
4. SYS$LC_ALL
5. The system default for the locale categories as specified by the SYS$* logical names. For example, the default for the category LC_NUMERIC is defined by the SYS$LC_NUMERIC logical name.
6. SYS$LANG

The system manager can choose to define SYS$* logicals in the site-specific system startup files to set the default locale. If no definition is provided, programs operate using the built-in "C" locale, in which case the LOCALE SHOW CURRENT command displays the string "C" for the current locale categories.

5.8.3 Example

```
$ DEFINE LC_COLLATE EN_US.ISO8859-1 ! NOTE: the collate category in unquoted
$ DEFINE LANG EN_GB_ISO8859-1
$ DEFINE LC_MESSAGES PRIVATE$DISK:[APPL.LOCALES]SPECIAL.LOCALE
$ LOCALE SHOW CURRENT
LANG="EN_GB_ISO8859-1"
LC_CTYPE="EN_GB_ISO8859-1"
LC_COLLATE=EN_US_ISO8859-1
LC_TIME="EN_GB_ISO8859-1"
LC_NUMERIC="EN_GB_ISO8859-1"
LC_MONETARY="EN_GB_ISO8859-1"
LC_MESSAGES=PRIVATE$DISK:[APPL.LOCALES]SPECIAL.LOCALE;1
LC_ALL=
```

This example shows a process where all locale categories except LC_COLLATE and LC_MESSAGES have defaulted to the same locale, EN_GB.ISO8859-1. A setting enclosed in double quotes indicates that the setting is implied by the setting of one of the following logical names: LANG, LC_ALL, SYS$LC_ALL, or SYS$LANG. A setting not enclosed by double quotes indicates that the logical name for that category defines the international environment. This example also shows that if a locale category is specified by a complete file specification, then the complete file specification is displayed.

5.8.4 Errors

If any logical names that define the environment are improperly defined, no warning message is issued. However, the actual international environment is listed exactly as it would be seen by an application that uses the DEC C run-time library routine `setlocale` (for instance, if in the previous example the SPECIAL.LOCALE file does not exist, then the display for the LC_MESSAGES category would show LC_MESSAGES="C").

5.9 LOCALE SHOW PUBLIC

This command lists all the public locales on the system.

5.9.1 Format

LOCALE SHOW PUBLIC

5.9.1.1 Parameters

None.

5.9.1.2 Qualifiers

None.

5.9.2 Description

This command lists all the public locales on the system. The set of public locales contains all the locales that reside in the directory defined by the logical name SYS$I18N_LOCALE as well as the system's built-in locales supplied with the DEC C run-time library.

5.9.3 Example

```
$ LOCALE SHOW PUBLIC

C (Built-in)
POSIX (Built-in)
[SYS$I18N.LOCALES.SYSTEM]EN_GB_ISO8859_1
[SYS$I18N.LOCALES.SYSTEM]EN_US_ISO8859_1
[SYS$I18N.LOCALES.SYSTEM]FR_CA_ISO8859_1
[SYS$I18N.LOCALES.SYSTEM]GRBAGE_LOCALE (bad file header checksum)
[SYS$I18N.LOCALES.SYSTEM]JA_JP_DECKANJI (Permanently Loaded)
```

This example shows a system with three locale files in the SYS$I18N_LOCALE directory. The C and POSIX locales are built in with the system and, therefore, cannot be found in SYS$I18N_LOCALE directory.

This example also shows the effect of having a bad file or a nonlocale file in the public directory and the effect of having a locale file loaded into the system's memory by the LOCALE LOAD command.

5.10 LOCALE SHOW VALUE

This command displays the value of one or more keywords from the current international environment.

5.10.1 Format

LOCALE SHOW VALUE name[,...]

5.10.1.1 Parameter

name[,...]

The name of a keyword or category. If you specify a keyword, the value of that keyword in the current locale is displayed. If you specify a category, the values of the keywords in that category are displayed. For integer keywords that have no value assigned, the value CHAR_MAX (127) is displayed. When a keyword value includes semicolons, double quotes, backslashes, or control characters, they are preceded by an escape character (usually a backslash).

Table 5–3 lists the categories and keywords you can specify for *name*.

Table 5–3 Locale Categories and Keywords

Category	Keyword	Keyword Description
LC_CTYPE		Character classification names
LC_TIME	DAY	Full weekday names
	ABDAY	Abbreviated weekday names
	MON	Full month names
	ABMON	Abbreviated month names
	D_T_FMT	Date and time format
	D_FMT	Date format
	T_FMT	Time format
	T_FMT_AMPM	Time format in the 12-hour clock
	AM_PM	Defines how the ante meridiem (a.m.) and post meridiem (p.m.) strings are represented
	ERA	Defines how years are counted and displayed for eras in a locale
	ERA_D_FMT	Era date format
	ERA_D_T_FMT	Era date and time format
	ERA_T_FMT	Era time format
	ALT_DIGITS	String defining alternative symbols for digits
LC_NUMERIC	DECIMAL_POINT	Character used as a decimal delimiter
	THOUSANDS_SEP	Character used to group digits to the left of the decimal delimiter
	GROUPING	Defines how characters to the left of the decimal delimiter are grouped
LC_MONETARY	INT_CURR_SYMBOL	Character string representing the international currency symbol
	CURRENCY_SYMBOL	String used as the local currency symbol
	MON_DECIMAL_POINT	Character used as a decimal delimiter when formatting monetary quantities
	MON_THOUSANDS_SEP	Character used as a separator for groups of digits to the left of the decimal delimiter
	POSITIVE_SIGN	String used to represent positive monetary quantities

(continued on next page)

Table 5–3 (Cont.) Locale Categories and Keywords

Category	Keyword	Keyword Description
	NEGATIVE_SIGN	String used to represent negative monetary quantities
	INT_FRAC_DIGITS	Number of digits displayed to the right of the decimal delimiter when formatting monetary quantities using the international currency symbol
	FRAC_DIGITS	Number of digits displayed to the right of the decimal delimiter when formatting monetary quantities using the local currency symbol
	P_CS_PRECEDES	For positive monetary values, this is set to 1 if the local currency symbol precedes the number and 0 if the symbol follows the number
	N_CS_PRECEDES	For negative monetary values, this is set to 1 if the local currency symbol precedes the number and 0 if the symbol follows the number
	P_SEP_BY_SPACE	For positive monetary values, this is set to 0 if there is no space between the currency symbol and the value, 1 if there is a space, and 2 if there is a space between the symbol and the sign string
	N_SEP_BY_SPACE	For negative monetary values, this is set to 0 if there is no space between the currency symbol and the value, 1 if there is a space, and 2 if there is a space between the symbol and the sign string
	P_SIGN_POSN	Integer used to indicate where the POSITIVE_SIGN string should be placed
	N_SIGN_POSN	Integer used to indicate where the NEGATIVE_SIGN string should be placed
	MON_GROUPING	Defines how digits are grouped when formatting monetary values
LC_MESSAGES	YESSTR	String representing YES in the current locale
	NOSTR	String representing NO in the current locale
	YESEXPR	Expression representing an affirmative response in the current locale
	NOEXPR	Expression representing a negative response in the current locale

Note

When an environment variable that affects the setting of the current locale points to an invalid locale, then the "C" locale is set.

Other valid keywords that are not displayed by default as part of any category include:

- CHARMAP — displays the file specification of the charmap used when the locale was created.
- CODE_SET_NAME — defines the name of the coded character set for which the charmap file is defined.

- MB_CUR_MAX — defines the maximum number of bytes in a multibyte character.
- MB_CUR_MIN — defines the minimum number of bytes in a character in the coded character set.

5.10.1.2 Qualifiers

/CATEGORY

Displays the category name before each keyword. If /CATEGORY is not specified, the category name is not displayed.

/KEYWORD

Displays the keyword name before the value of a keyword. If /KEYWORD is not specified, the value of the keyword is displayed, but not its name.

5.10.2 Errors

```
%LOCALE-E-NOKEYFND, no keyword keyword-name found
```

The *keyword-name* is not a valid keyword. Specify only the keywords listed in Table 5–3.

5.10.3 Description

This command displays the value of one or more keywords from the current international environment.

5.10.4 Examples

1.

```
$ LOCALE SHOW VALUE NOEXPR
"^[nN][[:alpha:]]*"
```

Issuing LOCALE SHOW VALUE without qualifiers displays the value of the NOEXPR string.

2.

```
$ LOCALE SHOW VALUE/CATEGORY NOEXPR
LC_MESSAGES
"^[nN][[:alpha:]]*"
```

Specifying /CATEGORY displays the category name (LC_MESSAGES) before the value of the NOEXPR string.

3.

```
$ LOCALE SHOW VALUE/KEYWORD NOEXPR
noexpr= "^[nN][[:alpha:]]*"
```

Specifying /KEYWORD displays the keyword name before its value.

4.

```
$ LOCALE SHOW VALUE/KEYWORD/CATEGORY NOEXPR
LC_MESSAGES
noexpr= "^[nN][[:alpha:]]*"
```

Specifying /KEYWORD and /CATEGORY displays the category and keyword name before the keyword value.

5.11 Locale File Format

A locale definition source file contains one or more categories that describe a locale. You can convert a locale definition source file into a locale by using the LOCALE COMPILE command. Locales can be modified only by editing a locale definition source file and then using the LOCALE COMPILE command again on the new source file. Each locale source file section defines a category of locale data. A source file cannot contain more than one section for the same category.

5.11.1 Locale Categories

The following standard locale categories are supported:

- LC_COLLATE — Defines character or string collation information
- LC_CTYPE — Defines character classification, case conversion, and other character attributes
- LC_MESSAGES — Defines the format for affirmative and negative responses
- LC_MONETARY — Defines rules and symbols for formatting monetary numeric information
- LC_NUMERIC — Defines a list of rules and symbols for formatting nonmonetary numeric information
- LC_TIME — Defines a list of rules and symbols for formatting time and date information

5.11.1.1 Overriding Defaults

You can include optional declarations at the beginning of your locale source file to override the default comment and escape characters used in locale category definitions:

- Escape character

  The escape character is used in decimal or hexadecimal constants when they are specified in the locale file. The default escape character is the backslash (\). To define another escape character, include a line with the following format:

  ```
  escape_char  <char_symbol>
  ```

- Comment character

  The comment character is the first character of any comment entries in the locale file. The default comment character is the number sign (#). To define another comment character, use the following format:

  ```
  comment_char  <char_symbol>
  ```

In the preceding formats, <*char_symbol*> is the character's symbolic name as defined in the charmap file used to build the locale's codeset. One or more blank characters (spaces or tabs) must separate `escape_char` or `comment_char` from <*char_symbol*>.

5.11.1.2 Category Source Definitions

Each category source definition consists of the following:

- The category header (*category_name*)
- The associated keyword or value pairs that comprise the category body
- The category trailer (END *category_name*)

For example:

```
LC_CTYPE
<source for LC_CTYPE category>
END LC_CTYPE
```

The source for all of the categories is specified using keywords, strings, character literals, and character symbols. Each keyword identifies either a definition or a rule. The remainder of the statement containing the keyword contains the operands to the keyword. Operands are separated from the keyword by one or more blank characters (spaces or tabs). A statement may be continued on the next line by placing a backslash (\) as the last character before the new-line character that terminates the line. Lines containing the comment character (#) in the first column are treated as comment lines.

A symbolic name begins with the left angle-bracket character (<) and ends with the right angle-bracket character (>). The characters between the < and the > can be any characters from the Portable Character Set, except for the control and space characters. For example, <A-diaeresis> could be a symbolic name for a character. Any symbolic name referenced in the locale source file must be defined via the Portable Character Set or in the character set description (charmap) file for that locale.

A character literal is the character itself, or a decimal, hexadecimal, or octal constant. A decimal constant contains two or three decimal digits and has the following form, where *n* is any decimal digit:

\d*nn* or \d*nnn*

A hexadecimal constant contains two hexadecimal digits and has the following form, where *n* is any hexadecimal digit:

\x*nn*

An octal constant contains two or three octal digits and has the following form, where *n* is any octal digit:

nn or *nnn*

The explicit definition of each category in a locale definition source file is not required. When a category is undefined in a locale definition source file, the LOCALE COMPILE command will not store any data value for this category in the resulting locale file.

5.11.2 The LC_COLLATE Category

The LC_COLLATE category defines the relative order between collation items. This category begins with the LC_COLLATE header and ends with the END LC_COLLATE trailer.

A collation item is the unit of comparison for collation. A collation item may be a character or a sequence of characters. Every collation item in the locale has a set of weights, which determine if the collation item collates before, equal to, or after the other collation items in the locale. Each collation item is assigned collation weights by the LOCALE COMPILE command when the locale definition source file is compiled. These collation weights are then used by applications programs that compare strings.

String comparison is performed by comparing the collation weights of each character in the string until either a difference is found or the strings are determined to be equal. This comparison may be performed several times if the locale defines multiple collation orders. For example, in the French locale, the strings are compared using a primary set of collation weights. If they are equal on the basis of this comparison, they are compared again using a secondary set of collation weights. A collation item has a set of collation weights associated with it that is equal to the number of collation sort rules defined for the locale.

Every character defined in the charmap file (or every character in the Portable Character Set if no charmap file is specified) is itself a collation item. Additional collation items can be defined using the `collating-element` statement (see the description that follows).

Table 5–4 lists the statement keywords recognized in the LC_COLLATE category.

Table 5–4 LC_COLLATE Category Keywords

Keyword	Description
`copy`	Specifies the name of an existing locale to be used as the definition of this category. If you specify a copy statement, you need not specify any other keywords in this category.
`collating-element`	Specifies multicharacter collation items.
`collating-symbol`	Specifies collation symbols for use in collation sequence statements.
`order_start`	Specifies collation order statements that assign collation weights to collation items.

The `collating-element`, `collating-symbol`, and `order_start` statements are further described in the following sections.

5.11.2.1 The collating-element Statement

The `collating-element` statement specifies multicharacter collation items.

Syntax:

```
collating-element <character_symbol> from <string>
```

The *character_symbol* argument defines a collation item that is a string of one or more characters as a single collation item. The *character_symbol* cannot duplicate any symbolic name in the current charmap file or any other symbolic name defined in this collation definition.

The *string* argument specifies a string of two or more characters that define the *character_symbol* argument. The following are examples of the syntax for the `collating-element` statement:

```
collating-element <ch> from "<c><h>"
collating-element <e-acute> from "<acute><e>"
collating-element <11> from "<1><1>"
```

A *character_symbol* argument defined by the `collating-element` statement is recognized only within the LC_COLLATE category.

5.11.2.2 The collating-symbol Statement

The `collating-symbol` statement specifies collation symbols for use in collation sequence statements.

Syntax:

```
collating-symbol <collating_symbol>
```

The `collating_symbol` argument cannot duplicate any symbolic name in the current charmap file or any other symbolic name defined in this collation definition. The following are examples of `collating-symbol` statements:

```
collating-symbol <UPPER_CASE>
collating-symbol <HIGH>
```

An argument defined by the `collating_symbol` statement is recognized only within the LC_COLLATE category.

5.11.2.3 The order_start Statement

The `order_start` statement is followed by one or more collation order statements that assign collation weights to collation items and the `order_end` keyword. The `order_start` statement is a required statement.

Syntax:

```
order_start sort_rules;sort_rules;...;sort_rules
collation_order_statements
order_end
```

Sort Rules

The *sort_rules* directives have the following syntax:

```
keyword, keyword,...,keyword
```

where *keyword* is FORWARD, BACKWARD, or POSITION.

The *sort_rules* directives are optional. If specified, they define the rules to apply during string comparison. The number of specified *sort_rules* directives defines the number of weights each collation item is assigned (that is, the directives define the number of collation orders in the locale). If no *sort_rules* directives are specified, one forward directive is assumed and comparisons are made on a character basis rather than a string basis.

If *sort_rules* directives are present, the first one applies when comparing strings that use the primary weight, the second when comparing strings that use the secondary weight, and so on. Each set of *sort_rules* directives is separated by a semicolon (;). A *sort_rules* directive consists of one or more keywords separated by commas. The following keywords are supported:

FORWARD — Specifies that collation weight comparisons proceed from the beginning of a string to the end of the string.

BACKWARD — Specifies that collation weight comparisons proceed from the end of a string to the beginning of the string.

POSITION — Specifies that collation weight comparisons consider the relative position of nonignored elements in the string (that is, if strings compare as equal, the element with the shortest distance from the starting point of the comparison collates first).

The forward and backward keywords are mutually exclusive.

Here is an example of a *sort_rules* directive:

```
order_start         forward;backward
```

Collation Order Statements

The following syntax rules apply to the collation order statements:

- Each collation order statement consists of a <*character_symbol*> specification followed by white space and a set of collation orders.
- Characters in the character set can be explicitly specified in the collation order statements or implicitly specified using the ellipsis symbol (...).
- A collation order statement that begins with the UNDEFINED special symbol specifies any characters that are in the character set but not explicitly or implicitly specified by other collation order statements.

The optional operands for each collation item are used to define the primary, secondary, or subsequent weights for the collation item. The special symbol IGNORE is used to indicate a collation item that is to be ignored when strings are compared.

An ellipsis keyword appearing in place of a collating_element_list indicates the weights are to be assigned, for the characters in the identified range, in numerically increasing order from the weight for the character symbol on the left side of the preceding statement.

The use of the ellipsis keyword results in a locale that may collate differently when compiled with different character set description (charmap) source files.

The UNDEFINED special symbol includes all coded character set values not specified explicitly or with an ellipsis symbol. These characters are inserted in the character collation order at the point indicated by the UNDEFINED special symbol and are all assigned the same weight. If no UNDEFINED special symbol exists and the collation order does not specify all collation items from the coded character set, a warning is issued and all undefined characters are placed at the end of the character collation order.

Example

The following is an example of a collation order statement section in the LC_COLLATE locale definition source file category:

```
order_start     forward;backward
UNDEFINED       IGNORE;IGNORE
<LOW>
<space>         <LOW>;<space>
...              <LOW>;...
<a>             <a>;<a>
<a-acute>       <a>;<a-acute>
<a-grave>       <a>;<a-grave>
<A>             <a>;<A>
<A-acute>       <a>;<A-acute>
<A-grave>       <a>;<A-grave>
```

```
<ch>            <ch>;<ch>
<Ch>            <ch>;<Ch>
<s>             <s>;<s>
<ss>            <s><s>;<s><s>
<eszet>         <s><s>;<eszet><eszet>
...             <HIGH>;...
<HIGH>
order_end
```

This example is interpreted as follows:

- The UNDEFINED special symbol indicates that all characters not specified in the definition (either explicitly or by the ellipsis symbol) are ignored for collation purposes.
- All collation items between <space> and <a> have the same primary equivalence class and individual secondary weights based on their coded character-set values.
- All versions of the letter a (uppercase and lowercase, and with or without diacriticals) belong to the same primary collation class.
- The <c><h> multicharacter collation item is represented by the <ch> collating symbol and belongs to the same primary equivalence class as the <C><h> multicharacter collation item.
- The <eszet> character is collated as an <s><s> string (that is, one <eszet> character is expanded to two characters before comparing).

5.11.3 The LC_CTYPE Category

The LC_CTYPE category defines character classification, case conversion, and other character attributes. This category begins with the LC_CTYPE header and ends with the END LC_CTYPE trailer.

All operands for LC_CTYPE category statements are defined as lists of characters. Each list consists of one or more characters or symbolic character names separated by semicolons. An ellipsis (...) can represent a series of characters; for example, <a>;...;<z> represents the characters in the range a through z.

Table 5–5 lists the statement keywords recognized in the LC_CTYPE category. In the keyword descriptions, the phrase "automatically included" means that an error does not occur if the referenced characters are included or omitted; the characters are provided if they are missing, and are accepted if they are present.

Table 5–5 LC_CTYPE Category Keywords

Keyword	Description
copy	Specifies the name of an existing locale to be used as the definition for this category. If you specify a copy statement, you cannot specify any other keyword.
upper	Defines uppercase letter characters. Do not specify any character defined by the cntrl, digit, punct, or space keyword. The uppercase letters A through Z are automatically included in this set.

(continued on next page)

Table 5–5 (Cont.) LC_CTYPE Category Keywords

Keyword	Description
`lower`	Defines lowercase letter characters. Do not specify any character defined by the `cntrl`, `digit`, `punct`, or `space` keyword. The lowercase letters a through z are automatically included in this set.
`alpha`	Defines all letter characters. Do not specify any character defined by the `cntrl`, `digit`, `punct`, or `space` keyword. Characters defined by the `upper` and `lower` keywords are automatically included in this character class.
`digit`	Defines numeric digit characters. Only the digits 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9 can be specified. The digits 0 through 9 are automatically included in this set.
`space`	Defines white-space characters. Do not specify any character defined by the `upper`, `lower`, `alpha`, `digit`, `graph`, or `xdigit` keyword. The space, form-feed, new-line, carriage-return, tab, and vertical tab characters are automatically included in this set.
`cntrl`	Defines control characters. Do not specify any character defined by the `upper`, `lower`, `alpha`, `digit`, `punct`, `graph`, `print`, or `xdigit` keyword.
`punct`	Defines punctuation characters. Do not specify the space character or any character defined by the `upper`, `lower`, `alpha`, `digit`, `cntrl`, or `xdigit` keywords.
`graph`	Defines printable characters, excluding the space character. Do not specify any character defined by the `cntrl` keyword. The characters defined by the `upper`, `lower`, `alpha`, `digit`, `xdigit`, and `punct` keywords are automatically included in this character class.
`print`	Defines printable characters, including the space character. Do not specify any character defined by the `cntrl` keyword. The space character and characters defined by the `upper`, `lower`, `alpha`, `digit`, `xdigit`, and `punct` keywords are automatically included in this character class.
`xdigit`	Defines hexadecimal digit characters. Only the digits 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9 can be specified. Any character, however, can be specified for the hexadecimal values for 10 to 15. These alternate hexadecimal digits are not used by standard conversion routines when converting digit strings from hexadecimal to numeric quantities. The numbers 0 through 9 and the letters A through F and a through f are automatically included in this set.
`blank`	Defines blank characters. The space and horizontal tab characters are included in this character class. Any characters defined by this statement are automatically included in the `space` class.

(continued on next page)

Table 5–5 (Cont.) LC_CTYPE Category Keywords

Keyword	Description
toupper	Defines the mapping of lowercase characters to uppercase characters. Operands for this keyword consist of character pairs separated by commas. Each character pair is enclosed in parentheses ( ) and separated from the next pair by a semicolon (;). The first character in each pair is considered a lowercase character; the second character is considered an uppercase character. Only characters defined by the lower and upper keywords can be specified. If toupper is not specified, a through z is mapped to A through Z by default.
tolower	Defines the mapping of uppercase characters to lowercase characters. Operands for this keyword consist of character pairs separated by commas. Each character pair is enclosed in parentheses ( ) and separated from the next pair by a semicolon (;). The first character in each pair is considered an uppercase character; the second character is considered a lowercase character. Only characters defined by the lower and upper keywords can be specified. If tolower is not specified, the mapping defaults to the reverse mapping of the toupper keyword, if specified. If the toupper and tolower keywords are both omitted, the mapping for each defaults to that of the C locale.

Additional keywords can be provided to define new character classifications. For example:

```
charclass vowel
vowel        <a>;<e>;<i>;<o>;<u>;<y>
```

The LC_CTYPE category does not support multicharacter elements (for example, the German Eszet character is traditionally classified as a lowercase letter). In proper capitalization of German text, the Eszet character is replaced by the two characters SS; there is no corresponding uppercase letter. This kind of conversion is outside the scope of the toupper and tolower keywords.

The following is an example of a possible LC_CTYPE category listed in a locale definition source file:

```
LC_CTYPE
#"alpha" is by default "upper" and "lower"
#"alnum" is by definition "alpha" and "digit"
#"print" is by default "alnum", "punct" and the space character
#"graph" is by default "alnum" and "punct"
#"tolower" is by default the reverse mapping of "toupper"
#
upper   <A>;<B>;<C>;<D>;<E>;<F>;<G>;<H>;<I>;<J>;<K>;<L>;<M>;\
        <N>;<O>;<P>;<Q>;<R>;<S>;<T>;<U>;<V>;<W>;<X>;<Y>;<Z>
#
lower   <a>;<b>;<c>;<d>;<e>;<f>;<g>;<h>;<i>;<j>;<k>;<l>;<m>;\
        <n>;<o>;<p>;<q>;<r>;<s>;<t>;<u>;<v>;<w>;<x>;<y>;<z>
#
digit   <zero>;<one>;<two>;<three>;<four>;<five>;<six>;\
        <seven>;<eight>;<nine>
#
space   <tab>;<newline>;<vertical-tab>;<form-feed>;\
        <carriage-return>;<space>
#
cntrl   <alert>;<backspace>;<tab>;<newline>;<vertical-tab>;\
        <form-feed>;<carriage-return>;<NUL>;<SOH>;<STX>;\
        <ETX>;<EOT>;<ENQ>;<ACK>;<SO>;<SI>;<DLE>;<DC1>;<DC2>;\
        <DC3>;<DC4>;<NAK>;<SYN>;<ETB>;<CAN>;<EM>;<SUB>;\
        <ESC>;<IS4>;<IS3>;<IS2>;<IS1>;<DEL>
#
punct   <exclamation-mark>;<quotation-mark>;<number-sign>;\
        <dollar-sign>;<percent-sign>;<ampersand>;<asterisk>;\
        <apostrophe>;<left-parenthesis>;<right-parenthesis>;\
        <plus-sign>;<comma>;<hyphen>;<period>;<slash>;\
        <colon>;<semicolon>;<less-than-sign>;<equals-sign>;\
        <greater-than-sign>;<question-mark>;<commercial-at>;\
        <left-square-bracket>;<backslash>;<circumflex>;\
        <right-square-bracket>;<underline>;<grave-accent>;\
        <left-curly-bracket>;<vertical-line>;<tilde>;\
        <right-curly-bracket>
#
xdigit  <zero>;<one>;<two>;<three>;<four>;<five>;<six>;\
        <seven>;<eight>;<nine>;<A>;<B>;<C>;<D>;<E>;<F>;\
        <a>;<b>;<c>;<d>;<e>;<f>
#
blank   <space>;<tab>
#
toupper (<a>,<A>);(<b>,<B>);(<c>,<C>);(<d>,<D>);(<e>,<E>);\
        (<f>,<F>);(<g>,<G>);(<h>,<H>);(<i>,<I>);(<j>,<J>);\
        (<k>,<K>);(<l>,<L>);(<m>,<M>);(<n>,<N>);(<o>,<O>);\
        (<p>,<P>);(<q>,<Q>);(<r>,<R>);(<s>,<S>);(<t>,<T>);\
        (<u>,<U>);(<v>,<V>);(<w>,<W>);(<x>,<X>);(<y>,<Y>);\
        (<z>,<Z>)
#
END LC_CTYPE
```

5.11.4 The LC_MESSAGES Category

The LC_MESSAGES category defines the format for affirmative and negative system responses. This category begins with the LC_MESSAGES header and ends with the END LC_MESSAGES trailer.

All operands for the LC_MESSAGES category are defined as strings or extended regular expressions bounded by double quotation marks ("). These operands are separated from the keyword they define by one or more blank characters (spaces or tabs). Two adjacent double quotation marks ("") indicate an undefined value.

Table 5–6 lists the statement keywords recognized in the LC_MESSAGES category.

Table 5–6 LC_MESSAGES Category Keywords

Keyword	Description
copy	Specifies the name of an existing locale to be used as the definition of this category. If you specify a copy statement, you cannot specify any other keyword.
yesexpr	Specifies an extended regular expression that describes the acceptable affirmative response to a question expecting an affirmative or negative response.
noexpr	Specifies an extended regular expression that describes the acceptable negative response to a question expecting an affirmative or negative response.
yesstr	Specifies the locale's equivalent of an acceptable affirmative response. This string is accessible to applications through the nl_langinfo subroutine as nl_langinfo (YESSTR). Note that yesstr is likely to be withdrawn from the XPG4 standard; yesexpr is the recommended alternative.
nostr	Specifies the locale's equivalent of an acceptable negative response. This string is accessible to applications through the nl_langinfo subroutine as nl_langinfo (NOSTR). Note that nostr is likely to be withdrawn from the XPG4 standard; noexpr is the recommended alternative.

The following is an example of a possible LC_MESSAGES category listed in a locale definition source file:

```
LC_MESSAGES
#
yesexpr "<circumflex><left-square-bracket><y><Y>\
<right-square-bracket>"
noexpr  "<circumflex><left-square-bracket><n><N>\
<right-square-bracket>"
yesstr  "<y><e><s>"
nostr   "<n><o>"
#
END LC_MESSAGES
```

5.11.5 The LC_MONETARY Category

The LC_MONETARY category defines rules and symbols for formatting monetary numeric information. This category begins with the LC_MONETARY header and ends with an END LC_MONETARY trailer.

5.11.5.1 LC_MONETARY Keywords

All operands for the LC_MONETARY category keywords are defined as string or integer values. String values are bounded by double quotation marks ("). All values are separated from the keyword they define by one or more blank characters (spaces or tabs). Two adjacent double quotation marks ("") indicate an undefined string value. A negative one (–1) indicates an undefined integer value.

Table 5–7 lists the statement keywords recognized in the LC_MONETARY category.

Table 5–7 LC_MONETARY Category Keywords

Keyword	Description
copy	Specifies the name of an existing locale to be used as the definition of this category. If you specify a copy statement, you cannot specify any other keyword.
int_curr_symbol	Specifies the string used for the international currency symbol. The operand for this keyword is a 4-character string†. The first three characters contain the alphabetic international currency symbol. The fourth character defines a character separator for insertion between the international currency symbol and a monetary quantity.
currency_symbol	Specifies the string used for the local currency symbol.
mon_decimal_point	Specifies the the decimal delimiter string used for formatting monetary quantities.
mon_thousands_sep	Specifies the character separator used for grouping digits to the left of the decimal delimiter in formatted monetary quantities.
mon_grouping	Specifies a string that defines the size of each group of digits in formatted monetary quantities. The operand for this keyword consists of a sequence of integers separated by semicolons. Each integer specifies the number of digits in a group. The first integer defines the size of the group immediately to the left of the decimal delimiter. Subsequent integers define succeeding groups to the left of the previous group. If the last integer is not –1, it is used to group any remaining digits. If the last integer is –1, no further grouping is performed. A sample interpretation of the mon_grouping statement follows. Assuming a value of 123456789 to be formatted and a mon_thousands_sep operand of ’ (single quotation mark), the following results occur: mon_grouping Formatted Value 3;-1 123456’789 3 123’456’789 3;2;-1 1234’56’789 3;2 12’34’56’789
positive_sign	Specifies the string used to indicate a nonnegative formatted monetary quantity.
negative_sign	Specifies the string used to indicate a negative formatted monetary quantity.
int_frac_digits	Specifies an integer value representing the number of fractional digits (those after the decimal delimiter) to be displayed in a formatted monetary quantity using the int_curr_symbol value.

†The current implementation of DEC C RTL allows more than four characters to be specified. However, the user should not rely on this fact and use it exactly as specified. The 4-character limit will be implemented in a future version of DEC C RTL.

(continued on next page)

Table 5–7 (Cont.) LC_MONETARY Category Keywords

Keyword	Description
`frac_digits`	Specifies an integer value representing the number of fractional digits (those after the decimal delimiter) to be displayed in a formatted monetary quantity using the `currency_symbol` value.
`p_cs_precedes`	Specifies an integer value indicating whether the `int_curr_symbol` or `currency_symbol` string precedes or follows the value for a nonnegative-formatted monetary quantity. The following integer values are recognized: 0 The currency symbol follows the monetary quantity. 1 The currency symbol precedes the monetary quantity.
`p_sep_by_space`	Specifies an integer value indicating whether the `int_curr_symbol` or `currency_symbol` string is separated by a space from a nonnegative-formatted monetary quantity. The following integer values are recognized: 0 No space separates the currency symbol from the monetary quantity. 1 A space separates the currency symbol from the monetary quantity. 2 A space separates the currency symbol and the `positive_sign` string, if adjacent.
`n_cs_precedes`	Specifies an integer value indicating whether the `int_curr_symbol` or `currency_symbol` string precedes or follows the value for a negative-formatted monetary quantity. The following integer values are recognized: 0 The currency symbol follows the monetary quantity. 1 The currency symbol precedes the monetary quantity.
`n_sep_by_space`	Specifies an integer value indicating whether the `int_curr_symbol` or `currency_symbol` string is separated by a space from a negative-formatted monetary quantity. The following integer values are recognized: 0 No space separates the currency symbol from the monetary quantity. 1 A space separates the currency symbol from the monetary quantity. 2 A space separates the currency symbol and the `negative_sign` string, if adjacent.

(continued on next page)

Table 5–7 (Cont.) LC_MONETARY Category Keywords

Keyword	Description
p_sign_posn	Specifies an integer value indicating the positioning of the positive_sign string for a nonnegative-formatted monetary quantity. The following integer values are recognized: 0 A left_parenthesis and right_parenthesis symbol enclose both the monetary quantity and the int_curr_symbol or currency_symbol string. 1 The positive_sign string precedes the quantity and the int_curr_symbol or currency_symbol string. 2 The positive_sign string follows the quantity and the int_curr_symbol or currency_symbol string. 3 The positive_sign string immediately precedes the int_curr_symbol or currency_symbol string. 4 The positive_sign string immediately follows the int_curr_symbol or currency_symbol string.
n_sign_posn	Specifies an integer value indicating the positioning of the negative_sign string for a negative-formatted monetary quantity. The following integer values are recognized: 0 A left_parenthesis and right_parenthesis symbol enclose both the monetary quantity and the int_curr_symbol or currency_symbol string. 1 The negative_sign string precedes the quantity and the int_curr_symbol or currency_symbol string. 2 The negative_sign string follows the quantity and the int_curr_symbol or currency_symbol string. 3 The negative_sign string immediately precedes the int_curr_symbol or currency_symbol string. 4 The negative_sign string immediately follows the int_curr_symbol or currency_symbol string.

5.11.5.2 Monetary Format Variations

You can produce a unique customized monetary format by changing the value of a single statement. Table 5–8 shows the results of using all combinations of defined values for the p_cs_precedes, p_sep_by_space, and p_sign_posn statements.

Table 5–8 Monetary Format Variations

	p_sep_by_space =	2	1	0
p_cs_precedes = 1	p_sign_posn = 0	($1.25)	($ 1.25)	($1.25)
	p_sign_posn = 1	+ $1.25	+$ 1.25	+$1.25
	p_sign_posn = 2	$1.25 +	$ 1.25+	$1.25+
	p_sign_posn = 3	+ $1.25	+$ 1.25	+$1.25
	p_sign_posn = 4	$ +1.25	$+ 1.25	$+1.25

(continued on next page)

Table 5–8 (Cont.) Monetary Format Variations

	p_sep_by_space =	2	1	0
p_cs_precedes = 0	p_sign_posn = 0	(1.25 $)	(1.25 $)	(1.25$)
	p_sign_posn = 1	+1.25 $	+1.25 $	+1.25$
	p_sign_posn = 2	1.25$ +	1.25 $+	1.25$+
	p_sign_posn = 3	1.25+ $	1.25 +$	1.25+$
	p_sign_posn = 4	1.25$ +	1.25 $+	1.25$+

The following is a sample LC_MONETARY category specified in a locale definition source file:

```
LC_MONETARY
#
int_curr_symbol        "<U><S><D><space>"
currency_symbol        "<dollar-sign>"
mon_decimal_point      "<period>"
mon_thousands_sep      "<comma>"
mon_grouping           3
positive_sign          "<plus-sign>"
negative_sign          "<hyphen>"
int_frac_digits        2
frac_digits            2
p_cs_precedes          1
p_sep_by_space         2
n_cs_precedes          1
n_sep_by_space         2
p_sign_posn            3
n_sign_posn            3
#
END LC_MONETARY
```

5.11.6 The LC_NUMERIC Category

The LC_NUMERIC category defines rules and symbols for formatting nonmonetary numeric information. This category begins with the LC_NUMERIC header and ends with the END LC_NUMERIC trailer.

All operands for the LC_NUMERIC category keywords are defined as string or integer values. String values are bounded by double quotation marks ("). All values are separated from the keyword they define by one or more blank characters (spaces or tabs). Two adjacent double quotation characters ("") indicate an undefined string value. A negative one (–1) indicates an undefined integer value.

Table 5–9 lists the statement keywords recognized in the LC_NUMERIC category.

Table 5–9 LC_NUMERIC Category Keywords

Keyword	Description
`copy`	Specifies the name of an existing locale to be used as the definition of this category. If you specify a `copy` statement, you cannot specify any other keyword.
`decimal_point`	Specifies the decimal delimiter string used to format nonmonetary numeric quantities. This keyword cannot be omitted and cannot be set to the undefined string value.
`thousands_sep`	Specifies the string separator used for grouping digits to the left of the decimal delimiter in formatted nonmonetary numeric quantities.

`grouping`	Defines the size of each group of digits in formatted monetary quantities.
	The operand for the grouping keyword consists of a sequence of integers separated by semicolons. Each integer specifies the number of digits in a group. The first integer defines the size of the group immediately to the left of the decimal delimiter. Subsequent integers define succeeding groups to the left of the previous group. Grouping is performed for each integer specified for the `grouping` keyword. If the last integer is not –1, it is used repeatedly to group any remaining digits. If the last integer is –1, no more grouping is performed.
	A sample interpretation of the `grouping` statement follows. Assuming a value of 123456789 to be formatted and a `thousands_sep` operand of ’ (single quotation mark), the following results occur:
	`grouping` — `Formatted Value`
	`3;-1` — `123456’789`
	`3` — `123’456’789`
	`3;2;-1` — `1234’56’789`
	`3;2` — `12’34’56’789`

The following is a sample LC_NUMERIC category specified in a locale definition source file:

```
LC_NUMERIC
#
decimal_point   "<period>"
thousands_sep   "<comma>"
grouping        <3>
#
END LC_NUMERIC
```

5.11.7 The LC_TIME Category

The LC_TIME category defines rules and symbols for formatting time and date information. This category begins with the LC_TIME category header and ends with the END LC_TIME trailer.

All operands for the LC_TIME category keywords are defined as string or integer values. String values are bounded by double quotation marks ("). All values are separated from the keyword they define by one or more blank characters (spaces or tabs). Two adjacent double quotation characters ("") indicate an undefined string value. Field descriptors, described later in this section, are used by commands and subroutines that query the LC_TIME category to represent elements of time and date formats.

5.11.7.1 Keywords

Table 5–10 lists the statement keywords recognized in the LC_TIME category.

Table 5–10 LC_TIME Category Keywords

Keyword	Description
`copy`	Specifies the name of an existing locale to be used as the definition of this category. If you specify a `copy` statement, you cannot specify any other keyword.
`abday`	Defines the abbreviated weekday names corresponding to the %a field descriptor. Recognized values consist of seven strings separated by semicolons. The first string corresponds to the abbreviated name for the first day of the week (Sun), the second to the abbreviated name for the second day of the week, and so on.
`day`	Defines the full spelling of the weekday names corresponding to the %A field descriptor. Recognized values consist of seven strings separated by semicolons. The first string corresponds to the full spelling of the name of the first day of the week (Sunday), the second to the name of the second day of the week, and so on.
`abmon`	Defines the abbreviated month names corresponding to the %b field descriptor. Recognized values consist of 12 strings separated by semicolons. The first string corresponds to the abbreviated name for the first month of the year (Jan), the second to the abbreviated name for the second month of the year, and so on.
`mon`	Defines the full spelling of the month names corresponding to the %B field descriptor. Recognized values consist of 12 strings separated by semicolons. The first string corresponds to the full spelling of the name for the first month of the year (January), the second to the full spelling of the name for the second month of the year, and so on.
`d_t_fmt`	Defines the string used for the standard date-and-time format corresponding to the %c field descriptor. The string can contain any combination of characters and field descriptors.
`d_fmt`	Defines the string used for the standard date format corresponding to the %x field descriptor. The string can contain any combination of characters and field descriptors.
`t_fmt`	Defines the string used for the standard time format corresponding to the %X field descriptor. The string can contain any combination of characters and field descriptors.
`am_pm`	Defines the strings used to represent a.m. (before noon) and p.m. (afternoon) corresponding to the %p field descriptor. Recognized values consist of two strings separated by semicolons. The first string corresponds to the a.m. designation, the second string corresponds to the p.m. designation.

(continued on next page)

Table 5–10 (Cont.) LC_TIME Category Keywords

Keyword	Description
t_fmt_ampm	Defines the string used for the standard 12-hour time format that includes an am_pm value (%p field descriptor). This statement corresponds to the %r field descriptor. The string can contain any combination of characters and field descriptors. If the string is empty, the 12-hour format is not supported by the locale.

era	Defines how the years are counted and displayed for each era in a locale, corresponding to the %E field descriptor modifier.
	For each era, there must be one string in the following format:
	direction:*offset*:*start_date*:*end_date*:*name*:*format*
	The variables for the era string format are defined as follows:
	• *direction* — Specifies a minus (-) or a plus (+) character.
	The minus character (-) indicates that years count in the negative direction when moving from the start date to the end date. The plus character (+) indicates that years count in the positive direction when moving from the start date to the end date.
	• *offset* — Specifies a number representing the first year of the era corresponding to the %Ey field descriptor.
	• *start_date* — Specifies the starting date of the era in yyyy/mm/dd format, where yyyy, mm, and dd are the year, month, and day, respectively, on the Gregorian calendar.
	Years prior to the year A.D. 1 are represented as negative numbers. For example, an era beginning March 5 in the year 100 B.C. would be represented as -100/03/05.
	• *end_date* — Specifies the ending date of the era in the same form used for the start_date variable or one of the two special values -* or +*.
	A -* value indicates that the ending date of the era extends backward to the beginning of time. A +* value indicates that the ending date of the era extends forward to the end of time. Therefore, the ending date can be chronologically before or after the starting date of the era. For example, the strings for the Christian eras A.D. and B.C. would be entered, respectively, in the following way:
	`+:0:0000/01/01:+*:AD:%Ey %EC`
	`+:1:-0001/12/31:-*:BC:%Ey %EC`
	• *name* — Specifies a string representing the name of the era that is substituted for the %EC field descriptor.
	• *format* — Specifies a strftime, strptime, and wcsftime format string to use when formatting the %EY field descriptor.
	This string can contain any strftime, strptime, and wcsftime format control characters (except %EY) and locale-dependent multibyte characters.
	An era value consists of one string (enclosed in quotes) for each era. If more than one era is specified, each era string is separated by a semicolon (;).

(continued on next page)

Table 5–10 (Cont.) LC_TIME Category Keywords

Keyword	Description
era_d_fmt	Defines the string used to represent the date in alternate-era format corresponding to the %Ex field descriptor. The string can contain any combination of characters and field descriptors.
era_t_fmt	Defines the locale's alternative time format as represented by the %EX field descriptor for strftime, strptime, and wcsftime.
era_d_t_fmt	Defines the locale's alternative date-and-time format as represented by the %Ec field descriptor for strftime, strptime, and wcsftime.
alt_digits	Defines alternate strings for digits corresponding to the %O field descriptor. Recognized values consist of a group of strings separated by semicolons. The first string represents the alternate string for 0 (zero), the second string represents the alternate string for 1, and so on. You can specify a maximum of 100 alternate strings.

5.11.7.2 Field Descriptors

The LC_TIME locale definition source file uses field descriptors to represent elements of time and date formats. You can combine these field descriptors to create other field descriptors or to create time and date format strings. When used in format strings that contain field descriptors and other characters, field descriptors are replaced by their current values. All other characters are copied without change. Table 5–11 lists the field descriptors used by commands and subroutines that query the LC_TIME category for time formatting.

Table 5–11 LC_TIME Locale Field Descriptors

Field Descriptor	Meaning
%a	Represents the abbreviated weekday name (for example, Sun) defined by the abday statement.
%A	Represents the full weekday name (for example, Sunday) defined by the day statement.
%b	Represents the abbreviated month name (for example, Jan) defined by the abmon statement.
%B	Represents the full month name (for example, January) defined by the month statement.
%c	Represents the date-and-time format defined by the d_t_fmt statement.
%C	Represents the century as a decimal number (00 to 99).
%d	Represents the day of the month as a decimal number (01 to 31).
%D	Represents the date in %m/%d/%y format (for example, 01/31/91).
%e	Represents the day of the month as a decimal number (1 to 31). If the day of the month is not a 2-digit number, the leading digit is filled with a space character.
%Ec	Specifies the alternate date-and-time representation for the locale.
%EC	Specifies the name of the base year (period) in the locale's alternate representation.

(continued on next page)

Table 5–11 (Cont.) LC_TIME Locale Field Descriptors

Field Descriptor	Meaning
%Ex	Specifies the alternate date representation for the locale.
%Ey	Specifies the offset from %EC (year only) in the locale's alternate representation.
%EY	Specifies the full alternate year representation.
%h	Represents the abbreviated month name (for example, Jan) defined by the abmon statement. This field descriptor is a synonym for the %b field descriptor.
%H	Represents the 24-hour clock hour as a decimal number (00 to 23).
%I	Represents the 12-hour clock hour as a decimal number (01 to 12).
%j	Represents the day of the year as a decimal number (001 to 366).
%m	Represents the month of the year as a decimal number (01 to 12).
%M	Represents the minutes of the hour as a decimal number (00 to 59).
%n	Specifies a new-line character.
%Od	Specifies the day of the month by using the locale's alternate numeric symbols.
%Oe	Specifies the day of the month by using the locale's alternate numeric symbols.
%OH	Specifies the hour (24-hour clock) by using the locale's alternate numeric symbols.
%OI	Specifies the hour (12-hour clock) by using the locale's alternate numeric symbols.
%Om	Specifies the month by using the locale's alternate numeric symbols.
%OM	Specifies the minutes by using the locale's alternate numeric symbols.
%OS	Specifies the seconds by using the locale's alternate numeric symbols.
%OU	Specifies the week number of the year (with Sunday as the first day of the week) by using the locale's alternate numeric symbols.
%Ow	Specifies the weekday as a number in the locale's alternate representation (Sunday = 0).
%OW	Specifies the week number of the year (with Monday as the first day of the week) by using the locale's alternate numeric symbols.
%Oy	Specifies the year (offset from %C) using the locale's alternate numeric symbols.
%p	Represents the a.m. or p.m. string defined by the am_pm statement.
%r	Represents the 12-hour clock time with a.m./p.m. notation as defined by the t_fmt_ampm statement.
%S	Represents the seconds of the minute as a decimal number (00 to 59).
%t	Specifies a tab character.
%T	Represents 24-hour clock time in the format %H:%M:%S (for example, 16:55:15).
%U	Represents the week of the year as a decimal number (00 to 53). Sunday, or its equivalent as defined by the day statement, is considered the first day of the week for calculating the value of this field descriptor.

(continued on next page)

Table 5–11 (Cont.) LC_TIME Locale Field Descriptors

Field Descriptor	Meaning
%w	Represents the day of the week as a decimal number (0 to 6). Sunday, or its equivalent as defined by the `day` statement, is considered to be 0 (zero) for calculating the value of this field descriptor.
%W	Represents the week of the year as a decimal number (00 to 53). Monday, or its equivalent as defined by the `day` statement, is considered the first day of the week for calculating the value of this field descriptor.
%x	Represents the date format defined by the `d_fmt` statement.
%X	Represents the time format defined by the `t_fmt` statement.
%y	Represents the year of the century (00 to 99).
%Y	Represents the year as a decimal number (for example, 1989).
%%	Specifies a % (percent sign) character.

5.11.7.3 Sample Locale Definition

The following is a sample LC_TIME category specified in a locale definition source file:

```
LC_TIME
#
#Abbreviated weekday names (%a)
abday   "<S><u><n>";"<M><o><n>";"<T><u><e>";"<W><e><d>";\
        "<T><h><u>";"<F><r><i>";"<S><a><t>"

#Full weekday names (%A)
day     "<S><u><n><d><a><y>";"<M><o><n><d><a><y>";\
        "<T><u><e><s><d><a><y>";"<W><e><d><n><e><s><d><a><y>";\
        <T><h><u><r><s><d><a><y>";"<F><r><i><d><a><y>";\
        <S><a><t><u><r><d><a><y>"

#Abbreviated month names (%b)
abmon   "<J><a><n>";"<F><e><b>";"<M><a><r>";"<A><p><r>";\
        "<M><a><y>";"<J><u><n>";"<J><u><l>";"<A><u><g>";\
        <S><e><p>";"<O><c><t>";"<N><o><v>";"<D><e><c>"

#Full month names (%B)
mon     "<J><a><n><u><a><r><y>";"<F><e><b><r><u><a><r><y>";\
        "<M><a><r><c><h>";"<A><p><r><i><l>";"<M><a><y>";\
        <J><u><n><e>";"<J><u><l><y>";"<A><u><g><u><s><t>";\
        "<S><e><p><t><e><m><b><e><r>";"<O><c><t><o><b><e><r>";\
        <N><o><v><e><m><b><e><r>";"<D><e><c><e><m><b><e><r>"

#Date-and-time format (%c)
#Note that for improved readability, this section uses actual
#characters, rather than symbolic names, and is inconsistent with
#the other sections in this example.  This is bad form.
#In practice, symbolic names should be used.
d_t_fmt         "%a %b %d %H:%M:%S %Y"
#
#Date format (%x)
d_fmt           "%m/%d/%y"
#
#Time format (%X)
t_fmt           "%H:%M:%S"
#
#Equivalent of AM/PM (%p)
am_pm           "<A><M>";"<P><M>"
```

```
#
#12-hour time format (%r)
#Note that for improved readability, this section uses actual
#characters, rather than symbolic names, and is inconsistent with
#the other sections in this example.  This is bad form.
#In practice, symbolic names should be used.
t_fmt_ampm      "%I:%M:%S %p"
#
era             "+:0:0000/01/01:+*:AD:%Ey %EC";\
"+:1:-0001/12/31:-*:BC:%Ey %EC"

era_d_fmt     ""
alt_digits    "<0><t><h>";"<1><s><t>";"<2><n><d>";"<3><r><d>";\
              "<4><t><h>";"<5><t><h>";"<6><t><h>";"<7><t><h>";\
              "<8><t><h>";"<9><t><h>";"<1><0><t><h>"
#
END LC_TIME
```

5.12 Character Set Description (Charmap) File

This section describes the character set description file, or charmap file. The charmap file defines character symbols as character encodings and is the source file for a coded character set, or codeset.

5.12.1 Portable Character Set

All supported codesets have the Portable Character Set (PCS) as a proper subset. The PCS consists of the following character symbols (listed by their standardized symbolic names) and their hexadecimal encodings. See Table 5–12.

Table 5–12 Portable Character Set

Symbol Name	Hexadecimal Encoding
<NUL>	\ x00
<alert>	\ x07
<backspace>	\ x08
<tab>	\ x09
<newline>	\ x0A
<vertical-tab>	\ x0B
<form-feed>	\ x0C
<carriage-return>	\ x0D
<space>	\ x20
<exclamation-mark>	\ x21
<quotation-mark>	\ x22
<number-sign>	\ x23
<dollar-sign>	\ x24
<percent>	\ x25
<ampersand>	\ x26
<apostrophe>	\ x27
<left-parenthesis>	\ x28

(continued on next page)

Table 5–12 (Cont.) Portable Character Set

Symbol Name	Hexadecimal Encoding
<right-parenthesis>	\ x29
<asterisk>	\ x2A
<plus-sign>	\ x2B
<comma>	\ x2C
<hyphen>	\ x2D
<period>	\ x2E
<slash>	\ x2F
<zero>	\ x30
<one>	\ x31
<two>	\ x32
<three>	\ x33
<four>	\ x34
<five>	\ x35
<six>	\ x36
<seven>	\ x37
<eight>	\ x38
<nine>	\ x39
<colon>	\ x3A
<semi-colon>	\ x3B
<less-than>	\ x3C
<equal-sign>	\ x3D
<greater-than>	\ x3E
<question-mark>	\ x3F
<commercial-at>	\ x40
<A>	\ x41
<B>	\ x42
<C>	\ x43
<D>	\ x44
<E>	\ x45
<F>	\ x46
<G>	\ x47
<H>	\ x48
<I>	\ x49
<J>	\ x4A
<K>	\ x4B
<L>	\ x4C
<M>	\ x4D
<N>	\ x4E

(continued on next page)

Table 5–12 (Cont.) Portable Character Set

Symbol Name	Hexadecimal Encoding
<O>	\ x4F
<P>	\ x50
<Q>	\ x51
<R>	\ x52
<S>	\ x53
<T>	\ x54
<U>	\ x55
<V>	\ x56
<W>	\ x57
<X>	\ x58
<Y>	\ x59
<Z>	\ x5A
<left-bracket>	\ x5B
<backslash>	\ x5C
<right-bracket>	\ x5D
<circumflex>	\ x5E
<underscore>	\ x5F
<grave-accent>	\ x60
<a>	\ x61
<b>	\ x62
<c>	\ x63
<d>	\ x64
<e>	\ x65
<f>	\ x66
<g>	\ x67
<h>	\ x68
<i>	\ x69
<j>	\ x6A
<k>	\ x6B
<l>	\ x6C
<m>	\ x6D
<n>	\ x6E
<o>	\ x6F
<p>	\ x70
<q>	\ x71
<r>	\ x72
<s>	\ x73
<t>	\ x74

(continued on next page)

Table 5–12 (Cont.) Portable Character Set

Symbol Name	Hexadecimal Encoding
<u>	\ x75
<v>	\ x76
<w>	\ x77
<x>	\ x78
<y>	\ x79
<z>	\ x7A
<left-brace>	\ x7B
<vertical-line>	\ x7C
<right-brace>	\ x7D
<tilde>	\ x7E

5.12.2 Components of a Charmap File

A charmap file has the following components:

- An optional special symbolic name declarations section

  Each declaration in this section consists of a special symbolic name, followed by one or more space or tab characters, and a value. The following list describes the special symbolic names that you can include in the declarations section:

  `<code_set_name>`

  Specifies the name of the codeset for which the charmap file is defined. This value determines the value returned by the `nl_langinfo (CODESET)` subroutine. If <code_set_name> is not declared, the name for the Portable Character Set is used.

  `<mb_cur_max>`

  Specifies the maximum number of bytes in a character for the codeset. Valid values are 1 to 4. The default value is 1.

  `<mb_cur_min>`

  Specifies the minimum number of bytes in a character for the codeset. Since all supported codesets have the Portable Character Set as a proper subset, this value must be 1.

  `<escape_char>`

  Specifies the escape character that indicates encodings in hexadecimal or octal notation. The default value is a backslash (\ ).

  `<comment_char>`

  Specifies the character used to indicate a comment within a charmap file. The default value is the number sign (#).

- The CHARMAP section header

  This header marks the beginning of the section that associates character symbols with encodings.

- Mapping statements for characters in the codeset

  Each statement specifies a symbolic name for a character and the associated encoding for that character. A mapping statement has the following format:

  ```
  <char_symbol> encoding
  ```

  A symbolic name begins with the left angle-bracket (<) character and ends with the right angle-bracket (>) character. For *char_symbol* (the name between < and >), you can use any characters from the Portable Character Set, except for control and space characters. You can use a > in *char_symbol*; if you do, precede all > characters except the last one with the escape character (as specified by the `<escape_char>` special symbolic name).

  An encoding is specified as one or more character constants, with the maximum number of character constants specified by the `<mb_cur_max>` special symbolic name. The encoding may be specified as decimal, octal, or hexadecimal constants with the following formats:

  - Decimal constant: \ d*nn* or \ d*nnn*, where *n* is any decimal digit
  - Octal constant: \ *nn* or \ *nnn*, where *n* is any octal digit
  - Hexadecimal constant: \ x*nn*, where *n* is any hexadecimal digit

  The following are sample character symbol definitions:

  ```
  <A>        \d65        #decimal constant
  <B>        \x42        #hexadecimal constant
  <j10101>   \x81\xA1    #multiple hexadecimal constants
  ```

  You can also define a range of symbolic names and corresponding encoded values, where the nonnumeric prefix for each symbolic name is common, and the numeric portion of the second symbolic name is equal to or greater than the numeric portion of the first symbolic name. In this format, a symbolic name value consists of zero or more nonnumeric characters followed by an integer of one or more decimal digits. This format defines a series of symbolic names. For example, the string <j0101>...<j0104> is interpreted as the symbolic names <j0101>, <j0102>, <j0103>, and <j0104>, in that order.

  In statements defining ranges of symbolic names, the specified encoded value is the value for the first symbolic name in the range. Subsequent symbolic names have encoded values in increasing order. Consider the following sample statement:

  ```
  <j0101>...<j0104>        \d129\d254
  ```

  This sample statement is interpreted as follows:

  ```
  <j0101> \d129\d254
  <j0102> \d129\d255
  <j0103> \d130\d0
  <j0104> \d130\d1
  ```

  You cannot assign multiple encodings to one symbolic name, but you can create multiple names for one encoded value because some characters have several common names. For example, the . character is called a period in some parts of the world, and a full stop in others. You can specify both names in the charmap. For example:

  ```
  <period>        \x2e
  <full-stop>     \x2e
  ```

Any comments must begin with the character specified by the <comment_char> special symbolic name. When an entire line is a comment, you must specify the <comment_char> in the first column of the line.

- The END CHARMAP section trailer

  This trailer indicates the end of character map statements.

The following is a portion of a sample charmap file:

```
CHARMAP
<code_set_name>         "ISO8859-1"
<mb_cur_max>            1
<mb_cur_min>            1
<escape_char>           \
<comment_char>          #

<NUL>                   \x00
<SOH>                   \x01
<STX>                   \x02
<ETX>                   \x03
<EOT>                   \x04
<ENQ>                   \x05
<ACK>                   \x06
<alert>                 \x07
<backspace>             \x08
<tab>                   \x09
<newline>               \x0a
<vertical-tab>          \x0b
<form-feed>             \x0c
<carriage-return>       \x0d
END CHARMAP
```

6 Local Area Network (LAN) Management Enhancements

The local area network (LAN) software has been enhanced to include system management tools for LAN configurations. The enhancements include two LAN utilities that work in conjunction with the OpenVMS LAN driver system software. The LAN utilities perform system management tasks related to LAN operations and provide other benefits.

The LAN system management enhancements:

- Allow you to set LAN parameters to customize your LAN environment.
- Display LAN settings and counters.
- Provide Maintenance Operations Protocol (MOP) downline load support for devices such as terminal servers, x-terminals, and LAN-based printers, and for booting satellites in a VMScluster environment. This enhancement provides an alternative to the traditional method of using either DECnet for OpenVMS or DECnet/OSI software.

Table 6–1 describes the LAN utilities and the functionality supported on systems running OpenVMS Alpha and OpenVMS VAX.

Table 6–1 LAN System Management Enhancements

Utility	Description	OpenVMS Support
LAN Auxiliary Control Program (LANACP)	Runs as a server process whose primary function is to provide MOP downline load service.	The LANACP utility provides identical functionality on VAX and Alpha systems running OpenVMS Version 6.2.

(continued on next page)

Table 6–1 (Cont.) LAN System Management Enhancements

<table>
<tr><th>Utility</th><th>Description</th><th>OpenVMS Support</th></tr>
<tr>
<td>LAN Control Program (LANCP)</td>
<td>Allows you to control LAN software parameters and obtain information from the LAN software. You can use the LANCP utility to:
• Obtain LAN device counters, revision, and configuration information
• Change the operational parameters of LAN devices on the system
• Maintain the LAN device database and the LAN node database
• Update the firmware on LAN devices
• Control the LANACP LAN Server process (including MOP downline load server related functions)
• Initiate MOP console carrier and MOP trigger boot operations</td>
<td>OpenVMS Alpha Version 6.1 contains the initial implementation of LANCP, which does not include MOP-related functions.
OpenVMS Version 6.2 (VAX and Alpha) adds MOP-related functions and extends this capability to VAX systems. The following table shows how the LAN utility functions are supported on VAX and Alpha systems:
<table>
<tr><th>Function</th><th>OpenVMS Alpha V6.2</th><th>OpenVMS VAX V6.2</th></tr>
<tr><td>Update firmware?</td><td>Yes</td><td>No</td></tr>
<tr><td>Change operational parameters of LAN devices?</td><td>Yes</td><td>No</td></tr>
<tr><td>Display LAN device information?</td><td>Yes</td><td>Limited</td></tr>
</table>
</td>
</tr>
</table>

This chapter describes how to invoke and use the LANCP and LANACP utilities.

6.1 The LANCP Utility

The LANCP utility implements commands to set and show LAN parameters. Section 6.1.1 describes how to invoke the LANCP utility. Table 6–2 describes LAN functions and provides section references to the LANCP commands that help you perform these functions.

Table 6–2 Functions of the LANCP Utility

Category	Function	Reference
LAN device management	Changes operational parameters, and displays counters and status information, and performs firmware updates of LAN devices.	Section 6.1.3
LAN device database management	Enables or disables MOP downline load service, and displays MOP counters information for LAN devices in the LAN permanent and volatile device databases.	Section 6.1.4
LAN node database management	Changes node data and displays MOP counters information in the LAN permanent and volatile node databases.	Section 6.1.5
LANACP MOP downline load service management	Changes operational parameters, and displays and clears counters and status information.	Section 6.1.6

(continued on next page)

Table 6–2 (Cont.) Functions of the LANCP Utility

Category	Function	Reference
LANCP MOP console carrier	Initiates console carrier connections.	Section 6.1.7
LANCP MOP trigger boot	Sends trigger boot requests to other nodes.	Section 6.1.8

6.1.1 Invoking and Exiting the LANCP Utility

Table 6–3 describes the ways you can invoke the LANCP utility (SYS$SYSTEM:LANCP.EXE).

Table 6–3 Invoking the LANCP Utility

Command	Example
Use the RUN command	At the DCL command prompt, enter: `$ RUN SYS$SYSTEM:LANCP` The LANCP utility responds by displaying the LANCP prompt at which you can enter LANCP commands.
Define LANCP as a foreign command	Either at the DCL prompt or in a startup or login command file, enter: `$ LANCP :== $SYS$SYSTEM:LANCP` Then, you can enter the command LANCP at the DCL prompt to invoke the utility and enter LANCP commands. When you enter the LANCP command: • Without specifying any command qualifiers, the LANCP utility displays the LANCP prompt at which you can enter commands. • With command qualifiers, the LANCP utility terminates after it executes the command and the DCL command prompt is displayed.
Use the MCR command	At the DCL command prompt, enter: `$ MCR LANCP` When you enter the MCR LANCP command: • Without specifying any command qualifiers, the LANCP utility displays the LANCP prompt at which you can enter commands. • With command qualifiers, the LANCP utility terminates after it executes the command and the DCL command prompt is displayed.

For information about the LANCP utility, enter the HELP command at the LANCP prompt.

To exit from the LANCP utility, enter the EXIT command at the LANCP prompt or press Ctrl/Z.

6.1.2 LAN Databases

The LANCP and LANACP utilities manipulate two databases:

- The *device database*, one for each node in the cluster, contains information about the LAN devices on the system.
- The *node database*, usually one for the entire cluster, contains information about all the nodes on the LAN for which LANCP/LANACP will provide MOP load service.

These two databases are stored on disk as the **permanent databases**. When the LANACP is started (usually during system startup) the contents of these databases are loaded into memory as the **volatile databases**. Maintaining a version of the databases in memory improves LANACP performance. The volatile databases are reloaded from the permanent databases every time the LANACP process is started, and by explicit operator command.

The permanent (disk-resident) versions of the databases are manipulated by LANCP DEFINE and PURGE commands.

- DEFINE commands are used to enter information into the databases.
- PURGE commands are used to remove information.

The volatile (memory-resident) versions of the databases are manipulated by LANCP SET and CLEAR commands. They provide the same functions as the permanent database DEFINE and PURGE commands, but only affect the volatile databases. Because the volatile databases are reloaded from the permanent databases each time the LANACP is started, the effect of SET and CLEAR commands is lost at each startup.

Use DEFINE and PURGE commands to modify the permanent databases for settings that you wish to keep permanently. Use SET and CLEAR commands to modify the volatile databases for settings that you wish to keep only for the current session. LANCP commands are provided to update the volatile databases with changes made to the permanent databases, and vice versa.

6.1.3 LAN Device Management

LAN device management consists of setting device parameters and displaying device characteristics. You can use the LANCP utility to set parameters for the types of LAN devices shown in Table 6–4.

Table 6–4 LAN Devices

LAN	Device Examples	Description
Ethernet	DE425, DE434, DE435, DE436, DE500, DECchip 21040	Allow the selection of media type (type of cable connected) and the speed of connection (Ethernet or FastEthernet). Allow full-duplex operation (point-to-point operation between similar devices or between the device and a switch).
FDDI	DEFTA, DEFPA, DEFAA, DEFEA, DEMFA	Allow full-duplex operation.
Token Ring	DETRA, DW300, DW110	Allow the setting of Token Ring parameters and the definition of source routing and functional address mapping.

(continued on next page)

Table 6–4 (Cont.) LAN Devices

LAN	Device Examples	Description
All	Any	Allow the setting of generic parameters such as the number of receive buffers.

6.1.3.1 Setting Device Parameters

All LAN devices are characterized by a collection of parameters. The parameters define the operational characteristics of a LAN device on the medium to which the device is connected.

At the LANCP> prompt, enter the SET DEVICE command to set LAN device parameters. The LANCP utility issues this command directly to the specified device (without interaction with the LANACP server process).

The syntax for the SET DEVICE command is:

SET DEVICE *device-name/qualifiers*

In this command, you specify:

- *Device-name*

  Supply the LAN controller device name. For example, you can specify a DEMNA controller as either EXA, EXA0, or EXA0:. To select all LAN devices, omit the device name and include the /ALL qualifier.

- *Qualifiers*

  See Table 6–5 for a description of the LANCP SET DEVICE command qualifiers.

Table 6–5 LANCP SET DEVICE Command Qualifiers

Qualifier	Description
/AGING_TIMER=*value*	Sets the amount of time in seconds to age source routing cache entries before marking them stale. This timer expires when no traffic is sent to or received from the remote node in this amount of time. **Default value:** 60 seconds **Note:** Increase this value when idle connections bounce between the stale and known states. Setting this value too low may cause unnecessary explorer traffic to traverse the LAN. **Devices:** Token Ring devices
/ALL	Sets data for all LAN devices. If a device name is specified, all matching LAN devices are selected (for example, E to select all Ethernet devices, F for FDDI devices, I for Token Ring devices, and EW for Ethernet PCI Tulip devices).
/CACHE_ENTRIES=*value*	Sets the number of entries to reserve for caching source routing address entries. **Default value:** 200 entries **Note:** If your system directly communicates to a large number of systems, you may want to increase this number. **Devices:** Token Ring devices
/CONTENDER	Specifies that the device is to participate in the Monitor Contention process when it joins the ring. **Default value:** /NOCONTENDER **Note:** The default setting directs the device not to challenge the current ring server. **Devices:** Token Ring devices
/DISCOVERY_TIMER=*value*	Sets the number of seconds to wait for a reply from a remote node when performing the source routing route discovery process. **Default value:** 2 seconds **Note:** If you have nodes that respond slowly on your extended LAN, you may need to increase this number to reduce the amount of explorer traffic that traverses your LAN. **Devices:** Token Ring devices
/EARLY	Enables Early Token Release on the device. The negated form of the qualifier, /NOEARLY, disables Early Token Release. **Default value:** /EARLY **Devices:** Token Ring devices operating on 16-megabit rings
/FULL_DUPLEX	Enables usage of full-duplex operation on a device. You may also need to specifically set up the device or network connection for full-duplex operation to get full-duplex operation. The negated form of the qualifier, /NOFULL_DUPLEX, disallows full-duplex operation regardless of the hardware configuration. **Default value:** /NOFULL_DUPLEX **Devices:** DEMFA, DEFAA, DEFEA, DEFPA, DEFTA, DE425, DE434, DE435, DE436, DECchip 21040
/MAP=(MULTICAST_ADDRESS=*address*, FUNCTIONAL_ADDRESS=*address*)	

(continued on next page)

Table 6–5 (Cont.) LANCP SET DEVICE Command Qualifiers

Qualifier	Description
	Defines or deletes a functional address mapping entry. Token Ring devices do not support IEEE 802 standard globally defined group addresses. They do support functional addresses. A functional address is a locally administered group address that has 31 possible values. Each functional address sets one bit in the third through sixth bytes of the address, and bytes 1 and 2 are 03-00 (C0:00 in bit reversed format). This command maps a standard multicast address to a functional address. The negated form of the qualifier, /NOMAP=(MULTICAST_ADDRESS=*address*), clears the mapping established for the specified address. Specify the functional address as follows: • The MULTICAST_ADDRESS qualifier requires a standard 6-byte multicast address. • The FUNCTIONAL_ADDRESS qualifier requires only the last 4 bytes of the functional address (the preceding 03-00 bytes are automatically prefixed). • The variable *address*, given as hexadecimal byte characters separated by hyphens, specifies the canonical form of the address. Use a colon as the separator character to indicate the bit-reversed form of the address. For example, to map the multicast address CB-00-01-02-03-04 to the functional address 03-00-00-80-00-00 on the Token Ring device IRA0, enter the following command: `SET DEVICE IRA0/MAP=(MULTI=CB-00-01-02-03-04,FUNCT=00:01:00:00)` **Default value:** See Table 6–6 for the default address mapping or issue the command SHOW DEVICE/MAP *device-name*. **Devices:** Token Ring devices
/MAX_BUFFERS=*value*	Sets the maximum number of receive buffers to be allocated and used by the LAN driver for the LAN device. The value must be within the range 4 to 32. The value cannot be less than the minimum number of receive buffers. **Default value:** 32 **Devices:** Any LAN device
/MEDIA=*value*	To set the /MEDIA qualifier: • For Token Ring devices, this qualifier selects the type of cable media that is being used to connect the adapter to the Token Ring Media Access Unit (MAU) for devices that do not automatically detect cable types. Acceptable values for this qualifier are either UTP (unshielded twisted pair) or STP (shielded twisted pair). **Default value:** STP • For Ethernet devices DE425, DE434, DE435, DE436, DE500, and DECchip 21040, this qualifier selects the cable connection. Normally, the selection is made during device initialization using a limited autosensing algorithm that selects twisted pair, but fails over to AUI if twisted pair does not appear to be functional. Thereafter, a cabling change would require a reboot of the system to take effect. This qualifier allows you to change the selection without rebooting. Acceptable values are AUI (10Base2, 10Base5), TWISTEDPAIR (10BaseT), and AUTOSENSE (perform the limited autosense algorithm again). **Default value:** AUTOSENSE **Note:** Some devices, such as the DE435, require a jumper change on the Ethernet card to switch between 10Base2 and 10Base5 (ThinWire and thickwire). Other devices, such as the DE434, DE436 and DE500, only have twisted-pair connections. **Devices:** Any LAN device that has software-settable media selection

(continued on next page)

Table 6–5 (Cont.) LANCP SET DEVICE Command Qualifiers

Qualifier	Description
/MIN_BUFFERS=*value*	This qualifier sets the minimum number of receive buffers to be allocated and used by the LAN driver for the LAN device. The value must be within the range 4 to 32. The value can not exceed the maximum number of receive buffers. **Default value:** 32 **Devices:** Any LAN device
/SOURCE_ROUTING	Enables source routing on the Token Ring device. If you have only one ring in your LAN or you use transparent bridging, use the /NOSOURCE_ROUTING qualifier to turn off source routing. **Default value:** /SOURCE_ROUTING **Devices:** Token Ring devices
/SPEED=*value*	Sets the speed of the LAN. For Token Ring, valid values are either 4 or 16, indicating 4 megabits per second or 16 megabits per second. For Ethernet, valid values are either 10 or 100, which selects the 10 megabits per second Ethernet port or the 100 megabits per second FastEthernet port. **Default value:** The default value for Token Ring is 16 unless the LAN adapter supports a nonvolatile mechanism for setting this parameter (as does the DEC Token Ring Controller 700). The default value for Ethernet is to sense automatically which type of port is connected and select the appropriate speed. **Devices:** Token Ring devices and the DE500 Ethernet/FastEthernet device
/SR_ENTRY=(LAN_ADDRESS=*address*, RI=*routing-information*)	
	Statically defines a specific source routed route for a specific node. This caching will remain valid while used or until the aging timer expires. The negated form of the qualifier, /NOSR_ENTRY=(LAN_ADDRESS=*address*), clears the previously defined static source routed route. The *address* is a standard 6-byte LAN address given as hexadecimal byte characters separated by hyphens, which specifies the canonical form of the address. Using a colon as the separator character indicates the bit-reversed form of the address. The *routing-information* is the source routing field, specified as a series of two-byte hexadecimal characters (each byte separated by a hyphen). The field consists of a two-byte routing control field followed by up to 14 two-byte segment identifiers, each containing the ring number and the bridge number used in the hop. **Default value:** No routes specified **Note:** Use this only as a last resort when isolating communication failures on extended LAN topologies. **Devices:** Token Ring devices

Table 6–6 lists the default address mapping for Token Ring devices.

Table 6–6 Default Functional Address Mapping for Token Ring Devices

Multicast Address	Functional Address	Description
09-00-2B-00-00-04	03-00-00-00-02-00	ISO ALL ES
09-00-2B-00-00-05	03-00-00-00-01-00	ISO ALL IS
CF-00-00-00-00-00	03-00-00-08-00-00	Loopback Assistant
AB-00-00-01-00-00	03-00-02-00-00-00	DNA MOP Dump/Load
AB-00-00-02-00-00	03-00-04-00-00-00	DNA MOP Remote Console
AB-00-00-03-00-00	03-00-08-00-00-00	DNA L1 Routers

(continued on next page)

Table 6–6 (Cont.) Default Functional Address Mapping for Token Ring Devices

Multicast Address	Functional Address	Description
09-00-2B-02-00-00	03-00-08-00-00-00	DNA L2 Routers
09-00-2B-02-01-0A	03-00-08-00-00-00	DNA Phase IV Primary Router
AB-00-00-04-00-00	03-00-10-00-00-00	DNA Endnodes
09-00-2B-02-01-0B	03-00-10-00-00-00	DNA Phase IV Prime Unknown Destination
09-00-2B-00-00-07	03-00-20-00-00-00	PCSA NETBIOS Emulation
09-00-2B-00-00-0F	03-00-40-00-00-00	LAT Service Advertisement
09-00-2B-02-01-04	03-00-80-00-00-00	LAT Service Solicit
09-00-2B-02-01-07	03-00-00-02-00-00	LAT Xwindown Service Solicit
09-00-2B-04-00-00	03-00-00-04-00-00	LAST
09-00-2B-02-01-00	03-00-00-00-08-00	DNA Name Service Advertisement
09-00-2B-02-01-01	03-00-00-00-10-00	DNA Name Service Solicit
09-00-2B-02-01-02	03-00-00-00-20-00	DNA Time Service
03-00-00-00-00-01	03-00-00-00-00-01	NETBUI Emulation
03-00-02-00-00-00	03-00-02-00-00-00	RIPL

The following shows some SET DEVICE command examples:

SET DEVICE Examples

1.
```
LANCP> SET DEVICE/CONTENDER/MEDIA=UTP/NOEARLY/SOURCE ICA0
```

   This command enables monitor contention, UTP cable media, and source routing, and disables early token release for Token Ring device ICA0.

2.
```
LANCP> SET DEVICE/MEDIA=TWIST EWB0
```

   This command sets the media type to twisted pair for the second Tulip Ethernet device.

3.
```
LANCP> SET DEVICE/ALL/MIN_BUFFERS=12
```

   This command sets the number of receive buffers for all LAN devices to be no less than 12.

6.1.3.2 Displaying Device Parameters

The SHOW DEVICE command displays LAN device parameters. At the LANCP> prompt, enter the command using the following syntax:

SHOW DEVICE *device-name[/qualifiers*]

In this command, you specify:

- *Device-name*

  Supply the LAN controller device name. For example, you can specify a DEMNA controller as either EXA, EXA0, or EXA0:. This refers to the LAN template device, which is maintained for most of the device parameters and counters. Also, the device name can refer to a device unit representing an actual user or protocol. For example, the cluster protocol can be started on a device as EWA1. You can specify device units to view unit-specific parameter information.

  If no device name is given, all devices are displayed.

If a device name is specified, all matching LAN devices are displayed (for example, E to select all Ethernet devices, F for FDDI devices, I for Token Ring devices, and EW for Ethernet PCI Tulip devices).

- *Qualifiers*

  See Table 6–7 for a description of the SHOW DEVICE command qualifiers.

Note

If you do not specify a qualifier, the utility displays the matching devices without additional information.

Table 6–7 LANCP SHOW DEVICE Command Qualifiers

Qualifier	Description
/COUNTERS	Displays device counters. **Devices:** All LAN devices, template devices only
/MAP	Displays the current configuration of the functional address mapping table. **Devices:** Token Ring devices, template devices only
/PARAMETERS	Displays status and related information about the device. **Devices:** All LAN devices
/REVISION	Displays the current firmware revision of the adapter, if available or applicable. Not all LAN devices return revision information. **Devices:** All LAN devices, template devices only
/SR_ENTRY	Displays the contents of the current source routing cache table. This data is displayed for template devices only, regardless of the device specification. **Devices:** Token Ring devices, template devices only

The following examples show how to use the SHOW DEVICE command:

SHOW DEVICE Examples

1. `LANCP> SHOW DEVICE/COUNTERS EXA0`

```
Device Counters EXA0:
             Value  Counter
             -----  -------
            259225  Seconds since last zeroed
           5890496  Data blocks received
           4801439  Multicast blocks received
            131074  Receive failure
         764348985  Bytes received
         543019961  Multicast bytes received
                 3  Data overrun
           1533610  Data blocks sent
            115568  Multicast packets transmitted
            122578  Blocks sent, multiple collisions
             86000  Blocks sent, single collision
            189039  Blocks sent, initially deferred
         198120720  Bytes sent
          13232578  Multicast bytes transmitted
           7274529  Send failure
                 0  Collision detect check failure
                 0  Unrecognized frame destination
                 0  System buffer unavailable
                 0  User buffer unavailable
```

This command displays counters for Ethernet device EXA0.

2. `LANCP> SHOW DEVICE/MAP ICA0`

```
Multicast to Functional Address Mapping ICA0:
   Multicast address   Functional Address   Bit-Reversed
   -----------------   ------------------   ------------
   09-00-2B-00-00-04   03-00-00-00-02-00    C0:00:00:00:40:00
   09-00-2B-00-00-05   03-00-00-00-01-00    C0:00:00:00:80:00
   CF-00-00-00-00-00   03-00-00-08-00-00    C0:00:00:10:00:00
   AB-00-00-01-00-00   03-00-02-00-00-00    C0:00:40:00:00:00
   AB-00-00-02-00-00   03-00-04-00-00-00    C0:00:20:00:00:00
   AB-00-00-03-00-00   03-00-08-00-00-00    C0:00:10:00:00:00
   09-00-2B-02-00-00   03-00-08-00-00-00    C0:00:10:00:00:00
   09-00-2B-02-01-0A   03-00-08-00-00-00    C0:00:10:00:00:00
   AB-00-00-04-00-00   03-00-10-00-00-00    C0:00:08:00:00:00
   09-00-2B-02-01-0B   03-00-10-00-00-00    C0:00:08:00:00:00
   09-00-2B-00-00-07   03-00-20-00-00-00    C0:00:04:00:00:00
   09-00-2B-00-00-0F   03-00-40-00-00-00    C0:00:02:00:00:00
   09-00-2B-02-01-04   03-00-80-00-00-00    C0:00:01:00:00:00
   09-00-2B-02-01-07   03-00-00-02-00-00    C0:00:00:40:00:00
   09-00-2B-04-00-00   03-00-00-04-00-00    C0:00:00:20:00:00
   09-00-2B-02-01-00   03-00-00-00-08-00    C0:00:00:00:10:00
   09-00-2B-02-01-01   03-00-00-00-10-00    C0:00:00:00:08:00
   09-00-2B-02-01-02   03-00-00-00-20-00    C0:00:00:00:04:00
   03-00-00-00-00-01   03-00-00-00-00-01    C0:00:00:00:00:80
   03-00-02-00-00-00   03-00-02-00-00-00    C0:00:40:00:00:00
```

This command displays mapping information for Token Ring device ICA0.

3.
```
LANCP> SHOW DEVICE/PARAM IRA0

Device Parameters IRA0:
             Value  Parameter
             -----  ---------
            Normal  Controller mode
          External  Internal loopback mode
 00-00-93-58-5D-32  Hardware LAN address
        Token Ring  Communication medium
           Enabled  Functional address mode
                No  Full duplex enable
                No  Full duplex operational
                16  Line speed (megabits/second)
           16 Mbps  Ring speed
               STP  Line media
           Enabled  Early token release
          Disabled  Monitor contender
               200  SR cache entries
                 2  SR discovery timer
                60  SR Aging Timer
           Enabled  Source routing
                 3  Authorized access priority
 AA-00-04-00-92-FF  Upstream neighbor
                 0  Ring number
```

   This command displays status and parameter information for Token Ring device IRA0.

4.
```
LANCP> SHOW DEVICE/REVISION FXA0
Device revision FXA0:  05140823
```

   This command displays revision information for FDDI device FXA0.

5.
```
LANCP> SHOW DEVICE/SR_ENTRY ICA0

Source Routing Cache Table ICA0:
      LAN address      State    XmtTmo   RcvTmo  StaleTmo DiscvTmo
   -----------------   -----   -------- -------- -------- --------
   AA-00-04-00-92-FF   LOCAL   00000028 00000028 00000245 00000000
```

   This command displays source routing entry information for Token Ring device ICA0.

6.1.3.3 Displaying Device Configuration

The SHOW CONFIGURATION command displays the LAN devices on the system. At the LANCP> prompt, enter the command using the following syntax:

SHOW CONFIGURATION

The following example shows the output from a SHOW CONFIGURATION command that was entered on a node that has three LAN devices: two DE435s and a DETRA.

```
LANCP> SHOW CONFIGURATION
LAN Configuration:
   Device   Medium      Default LAN Address   Version
   ------   ------      -------------------   -------
    EWA0    CSMA/CD      08-00-2B-E4-00-BF    02000023
    EWB0    CSMA/CD      00-00-C0-92-A4-0D    02000023
    IRA0    Token Ring   00-00-93-58-5D-32    20000223
```

The version is the device-specific representation of the actual version. In this example, for two devices on the PCI bus, the actual version is in the low byte (2.3 for the DE435 adapters). A device that does not have a readable version is shown as version zero.

Consult your device-specific documentation to correlate the version returned with a particular hardware or firmware implementation of the device.

6.1.3.4 Firmware Updates

LAN devices contain firmware images in EEPROM or FLASH ROM that you can update using the LANCP utility. You can update devices such as the DEMNA, DEMFA, DEFAA, DEFTA, DEFEA, and DEFPA.

Note

You can also use methods other than the LANCP utility to update firmware. For example, you can use the LFU update utility on DEC 7000 and DEC 10000 systems to update DEMNA and DEMFA devices.

At the LANCP> prompt, enter the UPDATE DEVICE command using the following syntax:

UPDATE DEVICE *device-name*/FILE=*filename*[/RESET]

In this command, you can specify:

- *Device-name*

  Supply the LAN controller device name. For example, you can specify a DEMNA controller as either EXA, EXA0, or EXA0:.

- FILE=*filename*

  Provide the file specification of the file to be loaded into the device. The file consists of a 2 block file header followed by the binary firmware image.

- /RESET

  This is an optional qualifier. The default setting, /RESET, indicates that the device will begin using the new image when the firmware update completes. Use the /NORESET qualifier to prevent a device reset.

For example, the following command updates FDDI device FAA0 with the firmware image FBUS_MAIN.SYS located on DKA0:[FW]. The device begins using the new image after the firmware update has completed and a device reset has been done.

```
LANCP> UPDATE DEVICE FAA0/FILE=DKA0:[FW]FBUS_MAIN.SYS
```

6.1.4 LANACP Device Database Management

The LAN volatile and permanent device databases each contain a single entry for each LAN device that exists on the system. Each entry in the LAN volatile device database contains device information and MOP downline load counters information. Each entry in the LAN permanent device database contains device information that is used to populate the volatile database when the LANACP LAN Server process is started.

Typically, each database contains the same devices. However, the permanent database may contain entries for devices that have not yet been configured or installed in the system. The LANACP LAN Server process maintains the volatile device database. The LANCP utility maintains the permanent device database. You can manipulate either database using the LANCP utility commands depending on your user privileges, as follows:

- Privileged users can add or delete device entries from each database, enable or disable MOP downline load service, and clear MOP downline load counters information for LAN devices
- Unprivileged users can view the MOP downline load status and counters information

The following sections describe how to enter and remove devices from the LAN permanent and volatile device databases, and how to enable and disable MOP downline load service.

6.1.4.1 Entering Devices into the LAN Device Databases

Use the following command syntax to enter a device into the LAN permanent device database or to change device data for an entry that already exists in the database:

DEFINE DEVICE *device-name[/qualifiers*]

Use the following command syntax to enter a device into the LAN volatile device database or to change device data for an entry that already exists in the database:

SET DEVICE *device-name[/qualifiers*]

In these commands, you specify:

- *Device-name*

  Supply the LAN controller device name. For example, you can specify a DEMNA controller as either EXA, EXA0, or EXA0:. To select all LAN devices, omit the device name and include the /ALL qualifier.
- *Qualifiers*

  See Table 6–8 for a description of the LANCP command qualifiers.

Table 6–8 LANCP DEVICE Command Qualifiers

Qualifier	Description
/ALL	Defines data for all LAN devices in the LAN permanent or volatile device database. If a device name is specified, all matching LAN devices are selected (for example, E to select all Ethernet devices, F for FDDI devices, I for Token Ring devices, and EW for Ethernet PCI Tulip devices).
/MOPDLL=(*enable-option, exclusive-option, size-option, knownclientsonly-option*)	Provides the MOP downline load service settings for the device. Note that defaults apply to creation of an entry in the device database. If an existing entry is being modified, fields not specified are not changed. In this qualifier, you can specify: • *Enable-option* keyword Specify ENABLE or DISABLE to indicate that MOP downline load service should be enabled or disabled for the device. • *Exclusive-option* keyword Specify EXCLUSIVE to indicate that no other provider of MOP downline load service is allowed on the specified LAN device at the same time as LANACP. Specify NOEXCLUSIVE to indicate that LANACP MOP downline load service can coexist with other implementations (in particular, the DECnet Phase IV implementation that operates the MOP protocol in shared mode). • *Knownclientsonly-option* keyword Specify KNOWNCLIENTSONLY to indicate that MOP downline load requests should be serviced only for clients defined in the LANACP volatile node database. When NOKNOWNCLIENTSONLY is selected, LANACP searches the LAN$DLL directory for any images requested by clients that are not defined in the LANACP volatile node database. • *Size-option* Use SIZE=*value* to specify the size in bytes of the file data portion of each downline load message. The permitted range is 246 to 1482 bytes. Use a larger size for better load performance and less server overhead. Note that some clients may not support the larger size. You can override the size on a per-node basis. See Section 6.1.5.1 for details. The recommended size is the largest size that results in successful loads of all clients. The 1482 value is derived from the maximum packet size for CSMA/CD (Ethernet) of 1518 bytes less the 802e header and CRC is 1492 bytes, less 10 bytes of MOP protocol overhead. This leaves 1482 bytes as the maximum length of the file data portion of a downline load data message. **Default values:** /MOPDLL=(DISABLE, NOEXCLUSIVE, SIZE=246, NOKNOWNCLIENTSONLY)
/PERMANENT_DATABASE (SET command only)	Updates the device entries in the LAN volatile device database with any data currently set in the permanent database. This allows you to update the volatile database after changing data in the permanent database, rather than repeating the commands for each updated entry to apply the changes to the volatile database.
/UPDATE	Adds existing LAN devices to the LAN permanent or volatile device database that are not currently in the database.
/VOLATILE_DATABASE (DEFINE command only)	Updates the device entries in the LAN permanent device database with any data currently set in the volatile database. This allows you to update the permanent database after changing data in the volatile database, rather than repeating the commands for each updated entry to apply the changes to the permanent database.

The following examples show how to use the DEFINE DEVICE and SET DEVICE commands:

DEFINE DEVICE and SET DEVICE Examples

1.
```
LANCP> DEFINE DEVICE EXA0/MOPDLL=(ENABLE,EXCLUSIVE)
```

   This command defines LAN device EXA0 to enable LANACP MOP downline load service in exclusive mode. The settings of the KNOWNCLIENTSONLY and SIZE characteristics are not changed. If the device entry does not currently exist in the LAN permanent device database, these settings will be set to the defaults.

2.
```
LANCP> DEFINE DEVICE/ALL/MOPDLL=NOEXCLUSIVE
```

   This command sets all LAN devices defined in the LAN permanent device database to nonexclusive mode for LANACP MOP downline load service.

3.
```
LANCP> SET DEVICE EXA0/MOPDLL=(ENABLE,NOEXCLUSIVE)
LANCP> SET DEVICE FXA0/MOPDLL=(ENABLE,EXCL,KNOWN)
```

   These commands enable LANACP MOP downline load service for:

   - LAN device EXA0 in nonexclusive mode
   - LAN device FXB0 in exclusive mode for only known clients

6.1.4.2 Displaying Devices in the LAN Device Databases

Use the following command syntax to display device information in the LAN permanent device database:

LIST DEVICE *device-name*[/MOPDLL]

Use the following command syntax to display device information in the LAN volatile device database:

SHOW DEVICE *device-name*[/MOPDLL]

In these commands, you specify:

- *Device-name*

  Supply the LAN controller device name. For example, you can specify a DEMNA controller as either EXA, EXA0, or EXA0:.

  If no device name is given, all devices are displayed.

  If a device name is specified, all matching LAN devices are displayed (for example, E to select all Ethernet devices, F for FDDI devices, I for Token Ring devices, and EW for Ethernet PCI Tulip devices).

- /MOPDLL

  Display MOP downline load information.

Note

If you do not specify a qualifier, the utility displays the matching devices without additional information.

6.1.4.3 Deleting Devices from the LAN Device Databases

Use the following command syntax to delete a device from the LAN permanent device database:

PURGE DEVICE *device-name*[/ALL]

Use the following command syntax to delete a device from the LAN volatile device database:

CLEAR DEVICE *device-name*[/ALL]

In these commands, you specify:

- *Device-name*

  Supply the LAN controller device name. For example, you can specify a DEMNA controller as either EXA, EXA0, or EXA0:. To select all LAN devices, omit the device name and include the /ALL qualifier.

- /ALL

  Deletes all LAN devices in the LAN permanent device database. If a device name is specified, all matching LAN devices are selected (for example, E to select all Ethernet devices, F for FDDI devices, I for Token Ring devices, and EW for Ethernet PCI Tulip devices).

The following examples show how to use the PURGE DEVICE and CLEAR DEVICE commands:

PURGE DEVICE and CLEAR DEVICE Examples

1.
```
LANCP> PURGE DEVICE/ALL
```

   This command deletes all devices from the LAN permanent device database.

2.
```
LANCP> CLEAR DEVICE EXA0
```

   This command deletes device EXA0 from the LAN volatile device database.

6.1.4.4 Enabling MOP Downline Load Service

Use the following command syntax to enable MOP downline load service:

SET DEVICE *device-name*/MOPDLL=ENABLE

In this command, use the *device-name* parameter to supply the LAN controller device name.

See Section 6.1.4.1 for a complete description of this command.

6.1.4.5 Disabling MOP Downline Load Service

Use the following command syntax to disable MOP downline load service:

SET DEVICE *device-name*/MOPDLL=DISABLE

In this command, use the *device-name* parameter to supply the LAN controller device name.

See Section 6.1.4.1 for a complete description of this command.

6.1.5 LANACP Node Database Management

The LAN volatile and permanent node databases each contain a single entry for each defined LAN node. Each entry in the LAN volatile node database contains node information and MOP downline load counters information. Each entry in the LAN permanent node database contains node information that is used to populate the volatile database when the LANACP LAN Server process is started.

Typically, each database contains the same nodes. The LANACP LAN Server process maintains the volatile node database. The LANCP utility maintains the permanent node database. You can manipulate either database using the LANCP utility commands depending on your user privileges, as follows:

- Privileged users can add or delete node entries from each database, and clear MOP downline load counters information for LAN nodes
- Unprivileged users can view the node information and MOP downline load status and counters information

The following sections describe how to enter and remove nodes from the LAN permanent and volatile node databases.

6.1.5.1 Entering Nodes into the LAN Node Databases

Use the following command syntax to define node information in the LAN permanent node database:

DEFINE NODE *node-name[/qualifiers*]

Use the following command syntax to define node information in the LAN volatile node database:

SET NODE *node-name[/qualifiers*]

In these commands, you specify:

- *Node-name*

  Specify the node name, using up to 63 characters, associated with the node address. You can include the name of the LAN device in the node name to distinguish between multiple LAN devices on the same system. For example, use GALAXY_EXA, GALAXY_EXB, and GALAXY_FXA to distinguish between the three LAN devices on node GALAXY.

  If a node name is specified with the /ALL qualifier, all matching nodes are selected. For example, A/ALL selects all nodes beginning with A.

- *Qualifiers*

  See Table 6–9 for a description of the LANCP command qualifiers.

Table 6–9 LANCP NODE Command Qualifiers

Qualifier	Description
/ADDRESS=*node-address*	Specifies a unique LAN address associated with the node name. Specify the address as 6 bytes in hexadecimal notation, separated by hyphens. The address does not have to be unique (as might be the case when the address is not known, so a nonexistent address is specified). The default is 00-00-00-00-00-00. If the qualifier is not present, the setting will not be changed. You can specify the /NOADDRESS qualifier to clear the field.
/ALL	Defines data for all nodes in the LAN permanent or volatile node database. If a node name is specified, all matching nodes are selected. For example, A/ALL selects all nodes beginning with A.
/BOOT_TYPE=VAX_SATELLITE | ALPHA_SATELLITE | OTHER	Indicates the type of processing required for downline load requests. The downline load can be for a VAX satellite cluster boot (VAX_SATELLITE), an Alpha satellite cluster boot (ALPHA_SATELLITE), or the specified image (OTHER). The distinction is necessary because cluster satellite loads require that additional cluster-related data be appended to the load image indicated by the /FILE qualifier. The default value is OTHER. If the qualifier is not present, the setting will not be changed. Use the /NOBOOT_TYPE to clear the field.
/FILE=*file-specification*	Supplies the file name you want provided when the downline load request does not include a requested file name. The file specification is limited to 127 characters. If the /FILE qualifier is not present, the setting will not be changed. Use the /NOFILE qualifier to clear the field.
/ROOT=*directory-specification*	Supplies the directory specification to be associated with the file name. For cluster satellite service, the /ROOT qualifier specifies the satellite root directory. For noncluster service, this qualifier specifies the location of the file. If the file specification or the file name given in the boot request includes the directory name, this qualifier is ignored. The directory specification is limited to 127 characters. If you do not specify a /ROOT qualifier, the setting remains unchanged. Use the /NOROOT qualifier to clear a field.
/SIZE=*value*	Overrides the default load data size specified for the device. This is the size in bytes of the file data portion of each downline load message. The permitted range is 246 to 1482 bytes. Use a larger size for better load performance and less server overhead. See Section 6.1.4.1 for more details. If you do not specify a /SIZE qualifier, the setting remains unchanged. Use the /NOSIZE qualifier to clear the setting.
/V3	Forces the server to respond to only MOP Version 3 boot requests from this node. This qualifer may be used to resolve a load problem where the node does not implement MOP Version 4 correctly and cannot boot. If you do not specify a /V3 qualifier, the setting remains unchanged. Use the /NOV3 qualifier to clear the setting.
/VOLATILE_DATABASE (DEFINE command only)	Updates the node entries in the LAN permanent node database with any data currently set in the volatile database. This allows you to update the permanent database after changing data in the volatile database, rather than repeating the commands for each updated entry to apply the changes to the permanent database.
/PERMANENT_DATABASE (SET command only)	Updates the node entries in the LAN volatile node database with any data currently set in the permanent database. This allows you to update the volatile database after changing data in the permanent database, rather than repeating the commands for each updated entry to apply the changes to the volatile database.

The following examples show how to use the DEFINE NODE and SET NODE commands:

DEFINE NODE and SET NODE Examples

1.
```
DEFINE NODE GALAXY/ADDRESS=08-00-2B-11-22-33 -
                  /FILE=NISCS_LOAD.EXE -
                  /ROOT=$64$DIA14:<SYS10.> -
                  /BOOT_TYPE=VAX_SATELLITE
```

   This command sets up node GALAXY in the LAN permanent node database for booting as a VAX satellite into a VMScluster system.

   The NISCS_LOAD.EXE file is actually located on $64$DIA14:<SYS10.SYSCOMMON.SYSLIB>. The <SYSCOMMON.SYSLIB> is supplied by the LANACP LAN Server process and is not included in the root definition.

2.
```
DEFINE NODE ZAPNOT/ADDRESS=08-00-2B-11-22-33 -
                  /FILE=APB.EXE -
                  /ROOT=$64$DIA14:<SYS10.> -
                  /BOOT_TYPE=ALPHA_SATELLITE
```

   This command sets up node ZAPNOT for booting as an Alpha satellite into a VMScluster system.

   The APB.EXE file is actually located on $64$DIA14:<SYS10.SYSCOMMON.SYSEXE>. Note that <SYSCOMMON.SYSEXE> is supplied by the LANACP LAN Server process and is not included in the root definition.

3.
```
SET NODE CALPAL/ADDRESS=08-00-2B-11-22-33 -
              /FILE=APB_061.EXE
```

   This command sets up node CALPAL for booting an InfoServer image. It defines the file that should be loaded when a load request without a file name is received from node CALPAL.

   Because the file does not include a directory specification, the logical name LAN$DLL defines where to locate the file. You could give directory specification by using the file name or by using the /ROOT qualifier.

   Note that specifying the file name explicitly in the boot command overrides the file name specified in the node database entry.

6.1.5.2 Displaying Devices in the LAN Node Databases

Use the following command syntax to display node information in the LAN permanent node database:

LIST NODE *node-name*[/ALL]

Use the following command syntax to display node information in the LAN volatile node database:

SHOW NODE *node-name*[/ALL]

In these commands, you specify:

- *Node-name*

  Supply the node name. If no node name is given, all nodes are displayed.

- /ALL

  Displays data for all nodes in the LAN permanent or volatile node database. If a node name is specified, all matching nodes are selected. For example, A/ALL selects all nodes beginning with A.

6.1.5.3 Deleting Nodes from the LAN Node Databases

Use the following command syntax to delete a node from the LAN permanent node database:

PURGE NODE *node-name*[/ALL]

Use the following command syntax to delete a node from the LAN volatile node database:

CLEAR NODE *node-name*[/ALL]

In this command, you specify:

- *Node-name*

  Supply the node name.

- /ALL

  Deletes all LAN nodes in the LAN permanent or volatile node database. If a node name is specified, all matching nodes are selected. For example, A/ALL deletes all nodes whose name begins with A.

6.1.6 LANACP MOP Downline Load Service Management

The LANACP LAN Server process maintains the LAN volatile node and device databases. The LANCP utility provides commands that:

- Display MOP downline load status and counters information
- Clear counters information
- Enable or disable OPCOM messages and packet tracing.

Counters and status information is maintained for each node and device. Counters information includes transmitted and received byte and packet counts, transmit errors, logical errors such as protocol violations and timeouts, and number of load requests. Status includes the time of the last load and the status of the last load.

6.1.6.1 Displaying the Status and Counters Data

Use the following command syntax to display MOP downline load status:

SHOW MOPDLL

The following display shows counters information for a particular node.

```
LAN MOP DLL Status:
   EXA enabled in exclusive mode for known nodes only, data size 1482 bytes
   FXA disabled

       #Loads   Packets        Bytes     Last load time       Last loaded
       ------   -------        -----   --------------------  -----------------
   EXA      5      1675      4400620  23-SEP-1994 10:27.51  GALAXY
   FXA      0         0            0
```

On this node, there are two LAN devices, EXA (DEMNA) and FXA (DEMFA). MOP downline load service is enabled on EXA in exclusive mode.

Requests are answered only for nodes that are defined in the LANACP node database. The image data size in the load messages is 1482 bytes. There have been five downline loads, the last one occured on node GALAXY at 10:27. Finally, there are no recorded downline loads for FXA, which is currently disabled for downline load service.

Use the following command syntax to display recent downline load activity that has been logged in the LAN$ACP.LOG file:

```
SHOW LOG
```

6.1.6.2 Clearing the Counters Data

Use the following command syntax to clear MOP downline load counters for all nodes and devices:

```
CLEAR MOPDLL
```

6.1.6.3 Displaying the Status and Counters Data for Individual Nodes

Use the following command syntax to display MOP downline load information for nodes in the LAN permanent node database:

LIST NODE *node-name*[/ALL][/OUTPUT=*commandfilename*]

Use the following command syntax to display MOP downline load status and counters information for nodes in the LAN volatile node database:

SHOW NODE *node-name*[/ALL][/OUTPUT=*commandfilename*][/TOTAL]

In these commands, you can specify:

- *Node-name*

  Specify the node name. The name can include up to 63 characters associated with the node address. If a node is not specified, the /ALL qualifier is assumed. If you use the /ALL qualifier, the node name is ignored and all nodes are displayed.

- /ALL

  Display information for all nodes in the database.

- /OUTPUT=*commandfilename*

  Indicate that the output should be directed to the specified file in the form of a list of DEFINE NODE or SET NODE commands. The resulting command file can be used to create the LAN node databases.

- /TOTAL (SHOW NODE command only)

  Display counter totals only.

The following shows sample output from a node on which there are three nodes defined (GALAXY, ZAPNOT, and CALPAL). CALPAL has received two load requests:

- The first request is the multicast request that CALPAL volunteered to accept
- The second request is the load request sent directly to CALPAL for the actual load data. The elapsed time from the second load request to completion of the load was 6.65 seconds.

```
Node Listing:

GALAXY (08-00-2B-2C-51-28):
  MOP DLL:  Load file:   APB.EXE
            Load root:   $64$DIA24:<SYS11.>
            Boot type:   Alpha satellite

ZAPNOT (08-00-2B-18-7E-33):
  MOP DLL:  Load file:   NISCS_LOAD.EXE
            Load root:   LAVC$SYSDEVICE:<SYS10.>
            Boot type:   VAX satellite

CALPAL (08-00-2B-08-9F-4C):
  MOP DLL:  Load file:   READ_ADDR.SYS
            Last file:   LAN$DLL:APB_X5WN.SYS
            Boot type:   Other
            2 loads requested, 1 volunteered
            1 succeeded, 0 failed
            Last request was for a system image, in MOP V4 format
            Last load initiated 30-OCT-1994 09:11:17 on EXA0 for 00:00:06.65
            527665 bytes, 4161 packets, 0 transmit failures

Unnamed (00-00-00-00-00-00):

Totals:
  Requests received    2
  Requests volunteered 1
  Successful loads     1
  Failed loads         0
  Packets sent         2080
  Packets received     2081
  Bytes sent           523481
  Bytes received       4184
  Last load            CALPAL at 30-OCT-1994 09:11:17.29
```

6.1.6.4 OPCOM Messages

By default, OPCOM messages are enabled. Messages are generated by the LANACP LAN Server process when a device status changes, load requests are received, and loads complete. These messages are displayed on the operator's console and included in the log file written by LANACP, SYS$MANAGER:LAN$ACP.LOG.

Use the following command to enable OPCOM messages:

```
SET ACP/OPCOM
```

Use the following command to disable OPCOM messages:

```
SET ACP/NOOPCOM
```

6.1.6.5 Load Trace Facility

If the error data produced by the LANACP LAN Server process for a load request is not sufficient to help you determine why the load is failing, you can direct the server process to record trace data. The data consists of transmit and receive packet information for every transmit and receive done by the server, and written to a log file for each load attempt. The name of the log file is SYS$MANAGER:LAN$*nodename*.LOG. You can record either all packet data or only the first 32 bytes of each packet.

The following list describes the typical load sequence:

1. Receive a Program Request message on the Load Assistance Multicast Address from the requesting node, code 8.
2. Transmit an Assistance Volunteer message to the requesting node, code 3.

3. Receive a Program Request message on your node address from the requesting node, code 8.
4. Transmit a Memory Load message to the requesting node with sequence number zero, code 2.
5. Receive a Request Memory Load message requesting the next sequence number (modulo 256), code 10 (decimal).
6. Repeat steps 4 and 5 until there is no more data to send.
7. Transmit a Memory or Parameter Load with Transfer Address message, code 0 or 20 (decimal).
8. Receive a final Request Memory Load message requesting the next sequence number (modulo 256) indicating that the last message has been received, code 10 (decimal).

For cluster satellite loads, the last Memory Load message contains cluster parameters. This message and the final Load with Transfer Address messages are displayed in full even if only partial trace echo has been enabled.

Note that packet tracing significantly slows downline load operations for all requests and may result in very large LAN$ACP.LOG files if packet tracing is inadvertently left enabled.

Use the following command to enable partial tracing of packet data:

```
SET ACP/ECHO
```

Use the following command to enable full tracing of packet data:

```
SET ACP/ECHO/FULL
```

Use the following command to disable tracing of packet data:

```
SET ACP/NOECHO
```

6.1.7 LANCP MOP Console Carrier

Console carrier provides a mechanism to connect to a LAN device, such as a terminal server, that implements a management interface using the MOP console carrier protocol. The LANCP utility provides this function in the form of a CONNECT NODE command.

The command syntax is:

CONNECT NODE *node-specification/qualifiers*

In this command, you specify:

- *Node-specification*

  Supply either the node name or the node address of the target node. If you supply the node name, the node address is obtained via lookup of the node name in the LAN volatile node database. If you supply the node address, the corresponding node need not be defined in the LAN volatile node database.

- *Qualifiers*

  See Table 6–10 for a description of the LANCP command qualifiers.

Table 6–10 LANCP CONNECT NODE Command Qualifiers

Qualifier	Description
/DEVICE=*device-name*	Specifies the LAN controller device name to be used for the connection. For example, a DEMNA controller, EXA, is specified as EXA, EXA0 or EXA0:.
/DISCONNECT= *disconnect-character*	Specifies the disconnect character that LANCP will recognize to terminate the connection to the remote node. The default disconnect character is D, which indicates that a Ctrl/D will terminate the connection. Any ASCII character is allowed from @ through Z, except C, M, Q, S, Y, and the left bracket ([).
/PASSWORD=16hexdigits	Specifies the password to be used when the connection is initiated, in hexadecimal. For example, /PASSWORD=0123456789ABCDEF. The default password is zero. You can omit leading zeros.
/V3 or /V4	Allows override of the MOP format. Normally, LANCP determines the format by sending a MOP Request ID message in MOP Version 4 format to the remote node, waiting for a response. If no response is received, LANCP sends a message in MOP Version 3 format and waits for a response. The LANCP utility repeats this process several times until a response is received. You can specify the format: • To allow connection to nodes that do not support Request ID messages • As a means of getting around implementation problems with one of the formats

The following examples show how to use the CONNECT NODE command:

CONNECT NODE Examples

1.
```
CONNECT NODE GALAXY/DEVICE=EWA0
```

   This command attempts a console-carrier connection to node GALAXY using the Ethernet device EWA0.

2.
```
CONNECT NODE 08-00-2B-11-22-33/DEVICE=EWA0/PASSWORD=0123456789ABCDEF
```

   This command attempts a console-carrier connection to the given node address using the Ethernet device EWA0, with a password.

6.1.8 LANCP MOP Trigger Boot

Some systems recognize and respond to MOP remote boot requests. These systems typically require a password or other mechanism to prevent unwanted boot requests from triggering a reboot of the system. The LANCP utility provides this function in the form of the TRIGGER NODE command.

Use the following command syntax:

TRIGGER NODE *node-specification/qualifiers*

In this command, you can specify:

- *Node-specification*

  Supply either the node name or the node address of the target node. If you supply the node name, the node address is obtained via lookup of the node name in the LAN volatile node database. If you supply the node address, the corresponding node need not be defined in the LAN volatile node database.

- *Qualifiers*

  See Table 6–11 for a description of the LANCP command qualifiers.

Table 6–11 LANCP TRIGGER NODE Command Qualifiers

Qualifier	Description
/DEVICE=*device-name*	Specifies the LAN controller device name to be used for sending the boot messages. For example, a DEMNA controller, EXA, is specified as EXA, EXA0 or EXA0:.
/PASSWORD=16hexdigits	Specifies the password to be used when the connection is initiated, in hexadecimal (for example, /PASSWORD=0123456789ABCDEF). The default password is zero. You can omit leading zeros.

Rather than specify the format to send MOP Version 3 or 4, the LANCP utility sends one message in each format to the target node.

The following examples show how to use the TRIGGER NODE command:

TRIGGER NODE Examples

1. `TRIGGER NODE GALAXY/DEVICE=EWA0`

   This command sends MOP trigger boot messages to node GALAXY using Ethernet device EWA0.

2. `TRIGGER NODE 08-00-2B-11-22-33/DEVICE=EWA0/PASSWORD=0123456789ABCDEF`

   This command sends MOP trigger boot messages to the given node address using the Ethernet device EWA0, with indicated password.

6.1.9 LANCP Miscellaneous Functions

Use the SPAWN command to create a subprocess of the current process. The SPAWN command copies the context of the subprocess from the current process. The syntax for the SPAWN command is as follows:

```
SPAWN [optional command line]
```

You can set up the LANCP utility to execute commands from a command file. The LANCP utility recognizes the command file as the file name preceded by an at sign (@). The default file name extension is .COM.

6.2 The LANACP LAN Server Process

The LANACP Server process provides the following services:

- Maintenance of the LAN volatile node database
- Maintenance of the LAN volatile device database
- MOP downline load

Table 6–12 describes the logical name and a collection of files associated with the LANACP LAN Server process.

Table 6–12 The LANACP Utility

Component	Description
LAN$ACP system logical name	This logical name defines the name of the LANACP Server process log file, containing entries describing changes to the LAN permanent device and node databases, and load request and load status information. By default, this is defined as SYS$SPECIFIC:[SYSMGR]LAN$ACP.LOG.
LAN$DEVICE_DATABASE system logical name	This logical name defines the name of the LAN permanent device database. By default, this is defined as SYS$SPECIFIC:[SYSEXE]LAN$DEVICE_DATABASE.DAT.
LAN$DLL system logical name	This logical name defines the location of downline load files, where the location of the file is not provided in the load request or explicitly defined in the LAN volatile node database. By default, this will be defined as MOM$SYSTEM. When DECnet is used for MOP downline loading, it is defined as SYS$SYSROOT:[MOM$SYSTEM].
LAN$NODE_DATABASE system logical name	This logical name defines the name of the LAN permanent node database. By default, this is defined as SYS$COMMON:[SYSEXE]LAN$NODE_DATABASE.DAT.
SYS$MANAGER:SYSTARTUP_VMS.COM	This file contains an entry that may be used to start LANACP automatically at system startup.
SYS$STARTUP:LAN$STARTUP.COM	This file starts the LANACP server process.
SYS$SYSTEM:LANACP.EXE	This is the LANACP utility program.

6.2.1 Running the LANACP Process

To start the LANACP LAN Server process, type @SYS$STARTUP:LAN$STARTUP at the DCL prompt, or include this line in the SYS$MANAGER:SYSTARTUP_VMS.COM command file to start LANACP automatically at system startup.

The following shows the command line as it appears in SYS$MANAGER:SYSTARTUP_VMS.COM:

```
$!
$! To start the LANACP LAN server application, remove the comment delimiter
$! from the command line below.
$!
$! @SYS$STARTUP:LAN$STARTUP
$!
```

6.2.2 Stopping the LANACP LAN Server Process

To stop the LANACP LAN Server process, enter the SET ACP/STOP command at the LANCP utility prompt.

6.3 LAN MOP Downline Load Services

The collection of utilities and startup command files for LANCP and LANACP provide the necessary functionality for MOP downline load service. These utilities and files load cluster satellites, terminal servers, and systems requiring downline load of special images, such as console update images or system software update images (for Infoserver load).

This environment provides functionality that is similar to that provided by DECnet. The result is that a system manager can choose which functionality to use, DECnet MOP or LAN MOP. For VMScluster systems, LAN MOP permits the operation of a VMScluster without the presence of DECnet.

6.3.1 Coexistence with DECnet MOP

LAN MOP can coexist with DECnet MOP in the following ways:

- Running on different systems

  For example, DECnet MOP service is enabled on some of the systems on the LAN and LAN MOP is enabled on other systems.

- Running on different LAN devices on the same system

  For example, DECnet MP service is enabled on a subset of the available LAN devices on the system and LAN MOP is enabled on the remainder.

- Running on the same LAN device on the same system but targeting a different set of nodes for service

  For example, both DECnet MOP and LAN MOP are enabled but LAN MOP has limited the nodes to which it will respond. This allows DECnet MOP to respond to the remaining nodes.

6.3.2 Migrating from DECnet MOP to LAN MOP

To migrate to LAN MOP, follow these steps:

1. Decide which nodes are to provide MOP downline load service. These may be the same nodes that currently have service enabled for DECnet.
2. Populate the LAN permanent device database by typing MCR LANCP DEFINE DEVICE/UPDATE at the DCL prompt.
3. Populate the LAN permanent node database by entering a node definition for each of the cluster satellite nodes and any other nodes that are similarly defined in the DECnet node database. You can enter this data manually or execute the command procedure SYS$EXAMPLES:LAN$POPULATE.COM, following the directions and help provided.
4. Disable service on each of the DECnet circuits where it is currently enabled in the volatile database.
5. Enable service on each LAN device in the LAN permanent device database that you would like to use by typing MCR LANCP DEFINE DEVICE *device-name*/MOPDLL=ENABLE at the DCL prompt for each device.
6. If high performance is required, select a data size of 1482 bytes and only reduce this if some load requests now fail. Alternatively, set up one system to load those clients that require a small data size and set up a different system to load the other clients.
7. Start the LANACP server process by typing @SYS$STARTUP:LAN$STARTUP at the DCL prompt.

To migrate permanently, follow these steps:

1. Disable service on each of the DECnet circuits in the permanent database.
2. Edit SYS$MANAGER:SYSTARTUP_VMS.COM to start LANACP at system startup.

To migrate back to DECnet MOP, follow these steps:

1. Stop the LANACP server process by issuing the SET ACP/STOP command to the LANCP utility.
2. Reenable service on each of the DECnet circuits in the permanent and volatile databases.
3. Edit SYS$MANAGER:SYSTARTUP_VMS.COM to disable startup of LANACP at system startup.

6.3.3 Using CLUSTER_CONFIG_LAN.COM and LAN MOP

A new cluster management command procedure has been provided to facilitate the use of LANCP for LAN MOP booting of satellites. Called CLUSTER_CONFIG_LAN.COM, it resides in SYS$MANAGER and is a direct parallel to CLUSTER_CONFIG.COM, which is used by cluster managers to configure and reconfigure a VMScluster system. The two procedures perform the same functions, except CLUSTER_CONFIG.COM uses DECnet MOP for downline load, whereas CLUSTER_CONFIG_LAN.COM uses LAN MOP. Therefore, when you add a new node, CLUSTER_CONFIG_LAN.COM does not ask for the node's DECnet node name and address. Instead, it queries for an SCS node name and an SCS node id number.

For your convenience, you can still run CLUSTER_CONFIG.COM. When you execute CLUSTER_CONFIG.COM, it checks whether LANACP for MOP booting is running. It also checks to see if DECnet is running. If LANACP is running and DECnet is not, then CLUSTER_CONFIG.COM dispatches to CLUSTER_CONFIG_LAN.COM. If CLUSTER_CONFIG.COM discovers that both LANACP and DECnet are running, it asks the user whether LAN MOP booting is being used, and whether it should call CLUSTER_CONFIG_LAN.COM for the user.

6.3.3.1 Sample Satellite Load

The following shows how to issue commands to the LANCP utility to enable MOP downline load service and to define node ZAPNOT:

```
set acp/opcom
enable mop/dev=eza0
set node ZAPNOT/addr=08-00-2B-33-FB-F2/file=APB.EXE-
             /root=$64$DIA24:<SYS11.>/boot=Alpha
```

The following shows the OPCOM messages displayed when you start up the LANACP LAN Server process:

```
%%%%%%%%%%%  OPCOM  30-OCT-1994 06:47:35.18  %%%%%%%%%%%
Message from user SYSTEM on GALAXY
LANACP MOP Downline Load Service
Found LAN device EZA0, hardware address 08-00-2B-30-8D-1C

%%%%%%%%%%%  OPCOM  30-OCT-1994 06:47:35.25  %%%%%%%%%%%
Message from user SYSTEM on GALAXY
LANACP MOP Downline Load Service
Found LAN device EZB0, hardware address 08-00-2B-30-8D-1D

%%%%%%%%%%%  OPCOM  30-OCT-1994 06:47:54.80  %%%%%%%%%%%
Message from user SYSTEM on GALAXY
LANACP MOP V3 Downline Load Service
Volunteered to load request on EZA0 from ZAPNOT
Requested file:  $64$DIA24:<SYS11.>[SYSCOMMON.SYSEXE]APB.EXE
```

```
%%%%%%%%%%%  OPCOM  30-OCT-1994 06:48:02.38  %%%%%%%%%%%
Message from user SYSTEM on GALAXY
LANACP MOP V3 Downline Load Service
Load succeeded for ZAPNOT on EZA0
System image, $64$DIA24:<SYS11.>[SYSCOMMON.SYSEXE]APB.EXE (Alpha image)
```

The following display shows the contents of the LAN$ACP.LOG file:

```
30-OCT-1994 06:47:35.02  Found LAN device EZA0, hardware address 08-00-2B-30-8D-1C
30-OCT-1994 06:47:35.18  Found LAN device EZB0, hardware address 08-00-2B-30-8D-1D
30-OCT-1994 06:47:35.25  LANACP initialization complete
30-OCT-1994 06:47:45.39  Enabled LAN device EZA0 for MOP downline load service in exclusive mode
30-OCT-1994 06:47:54.70  Volunteered to load request on EZA0 from ZAPNOT
    Requested file:  $64$DIA24:<SYS11.>[SYSCOMMON.SYSEXE]APB.EXE
30-OCT-1994 06:48:02.23  Load succeeded for ZAPNOT on EZA0
    MOP V3 format, System image, $64$DIA24:<SYS11.>[SYSCOMMON.SYSEXE]APB.EXE
    Packets:  2063 sent, 2063 received
    Bytes:    519416 sent, 4126 received, 507038 loaded
    Elapsed time:  00:00:07.42, 68276 bytes/second
```

6.3.3.2 Cross-Architecture Booting

The LAN enhancements permit cross-architecture booting in a VMScluster system. VAX boot nodes can provide boot service to Alpha satellites and Alpha boot nodes can provide boot service to VAX satellites. Note that each architecture must include a system disk that is used for installations and upgrades.

7

SCSI as a VMScluster Storage Interconnect

One of the benefits of VMScluster systems is that multiple computers can simultaneously access storage devices connected to a VMScluster storage interconnect. Together, these systems provide high performance and highly available access to storage.

This chapter describes how VMScluster systems support the Small Computer Systems Interface (SCSI) as a storage interconnect. Multiple Alpha computers, also referred to as hosts or nodes, can simultaneously access SCSI disks over a SCSI interconnect. A SCSI interconnect, also called a SCSI bus, is an industry-standard interconnect that supports one or more computers, peripheral devices, and interconnecting components.

The discussions in this chapter assume that you already understand the concept of sharing storage resources in a VMScluster environment. VMScluster concepts and configuration requirements are described in the following VMScluster documentation:

- *Guidelines for VMScluster Configurations*
- *VMScluster Systems for OpenVMS*
- VMScluster Systems for OpenVMS Alpha *Software Product Description* (SPD 42.18.*xx*)

This chapter includes two primary parts:

- Section 7.1 through Section 7.6.6 describe the fundamental procedures and concepts that you would need to plan and implement a SCSI VMScluster system.
- Section 7.7 and its subsections provide additional technical detail and concepts; these sections can be seen as containing supplementary information about SCSI VMScluster systems that would typically be located in an appendix.

7.1 Conventions Used in This Chapter

Certain conventions are used throughout this chapter to identify the ANSI Standard and for elements in figures.

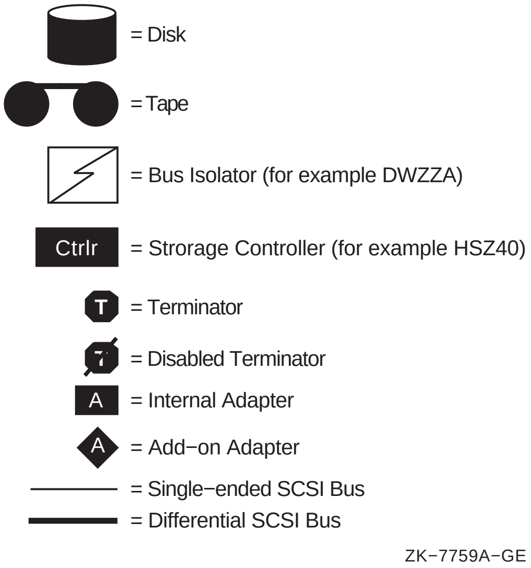
7.1.1 SCSI ANSI Standard

VMScluster systems configured with the SCSI interconnect must use standard SCSI–2 components. The SCSI–2 components supported must be compliant with the architecture defined in the *American National Standards Institute (ANSI) Standard SCSI–2*. This standard defines extensions to the SCSI–1 standard. For ease of discussion, this chapter uses the term SCSI or SCSI–2 to refer to the SCSI–2 implementation as specified in the ANSI Standard SCSI–2 document X3T9.2, Rev. 10L.

7.1.2 Symbols Used in Figures

Figure 7–1 is a key to the symbols used in figures throughout this chapter.

Figure 7–1 Conventions: Key to Symbols Used in Figures



7.2 Accessing SCSI Storage

In VMScluster configurations, multiple VAX and Alpha hosts can directly access SCSI devices in any of the following ways:

- CI interconnect with HSJ or HSC controllers
- Digital Storage Systems Interconnect (DSSI) with HSD controller
- SCSI adapters directly connected to VAX or Alpha systems

You can also access SCSI devices indirectly using the OpenVMS MSCP server.

The following sections describe single-host and multiple-host access to SCSI storage devices.

7.2.1 Single-Host SCSI Access in VMScluster Systems

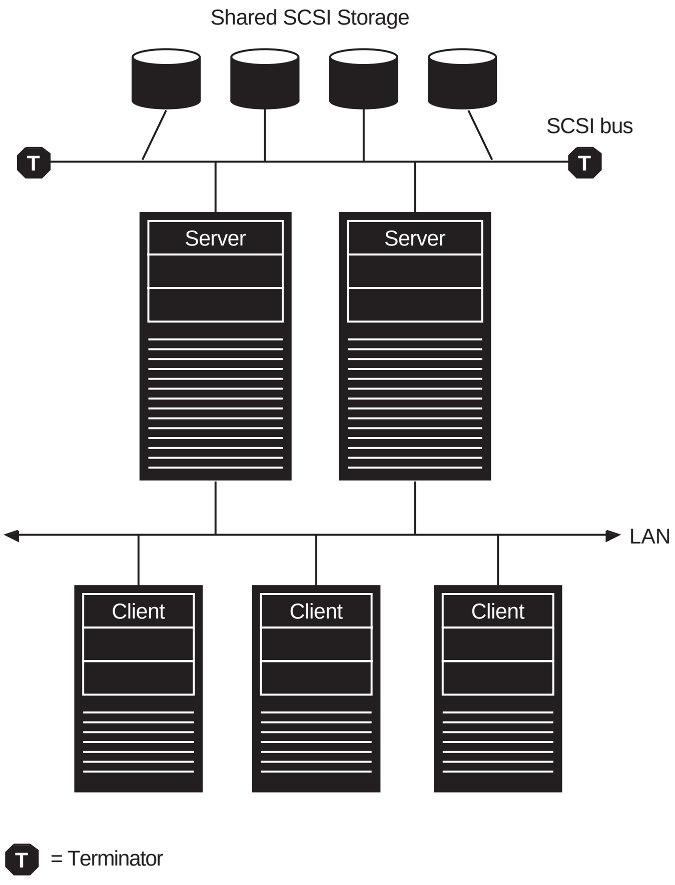
Prior to OpenVMS Version 6.2, VMScluster systems provided support for SCSI storage devices connected to a single host using an embedded SCSI adapter, an optional external SCSI adapter, or a special-purpose RAID (redundant array of independent disks) controller. Only one host could be connected to a SCSI bus.

7.2.2 Multiple-Host SCSI Access in VMScluster Systems

With OpenVMS Alpha Version 6.2, multiple Alpha hosts in a VMScluster system can be connected to a single SCSI bus to share access to SCSI storage devices directly. This capability allows you to build highly available servers using shared access to SCSI storage.

Figure 7–2 shows a VMScluster configuration that uses a SCSI interconnect for shared access to SCSI devices. Note that another interconnect (for example, a local area network [LAN]) is required for host-to-host VMScluster (System Communications Architecture [SCA]) communications.

Figure 7–2 Highly Available Servers for Shared SCSI Access



ZK−7479A−GE

You can build a two-node VMScluster system using the shared SCSI bus as the storage interconnect, or you can include shared SCSI buses within a larger VMScluster configuration. A quorum disk can be used on the SCSI bus to improve the availability of two-node configurations. Host-based RAID (including host-based shadowing) and the MSCP server are supported for shared SCSI storage devices.

7.3 Configuration Requirements and Hardware Support

This section lists the configuration requirements and supported hardware for SCSI VMScluster systems for OpenVMS Version 6.2.

7.3.1 Configuration Requirements

Table 7–1 shows the requirements and capabilities of the basic software and hardware components you can configure in a SCSI VMScluster system.

Table 7–1 Requirements for SCSI VMScluster Configurations

Requirement	Description
Software	All Alpha hosts sharing access to storage on a SCSI interconnect must be running: • OpenVMS Alpha Version 6.2 • VMScluster Software for OpenVMS Alpha Version 6.2
Hardware	Table 7–2 lists the supported hardware components for SCSI VMScluster systems. See also Section 7.7.7 for information about other hardware devices that might be used in a SCSI VMScluster configuration.
SCSI tape, floppies and CD–ROM drives	You cannot configure SCSI tape drives, floppy drives, or CD–ROM drives on multiple-host SCSI interconnects. If your configuration requires SCSI tape, floppy, or CD–ROM drives, configure them on single-host SCSI interconnects. Note that SCSI tape, floppy, or CD–ROM drives may be MSCP or TMSCP served to other hosts in the VMScluster configuration.
Maximum hosts on a SCSI bus	You can connect up to two hosts on a multiple-host SCSI bus. You can configure any mix of the hosts listed in Table 7–2 on the same shared SCSI interconnect.
Maximum SCSI buses per host	You can connect each host to a maximum of two multiple-host SCSI buses. The number of nonshared (single-host) SCSI buses that can be configured is limited only by the number of available slots on the host bus.
Host-to-host communication	All members of the cluster must be connected by an interconnect that can be used for host-to-host (SCA) communication; for example, DSSI, Ethernet, or FDDI.
Host-based RAID (including host-based shadowing)	Supported in SCSI VMScluster configurations.
SCSI device naming	The name of each SCSI device must be unique throughout the VMScluster system. When configuring devices on systems that include a multiple-host SCSI bus, adhere to the following requirements: • A host can have, at most, one controller attached to a particular SCSI interconnect. • All host controllers attached to a given SCSI interconnect must have the same OpenVMS device name (for example, PKA0). • Each system attached to a SCSI interconnect must have the same nonzero allocation class. Each disk device name (for example, DKB100), whether or not on a shared SCSI bus, must be unique within an allocation class. Refer to Section 7.6.2 for more information.

7.3.2 Hardware Support

Table 7–2 shows the supported hardware components for SCSI VMScluster systems; it also lists the minimum required revision for these hardware components (that is, for any component, you must use either the version listed in Table 7–2 or a subsequent version).

The SCSI interconnect configuration and all devices on the SCSI interconnect must meet the requirements defined in the *ANSI Standard SCSI–2* document and the requirements described in this chapter.

See also Section 7.7.7 for information about other hardware devices that might be used in a SCSI VMScluster configuration.

Table 7–2 Supported Hardware for SCSI VMScluster Systems

Component	Supported Item	Minimum Version or H/W Revision	How to Find Your Version
Hosts	AlphaServer 400 4/xxx	See footnote[1]	Console SHOW VERSION command
Hosts	AlphaServer 1000 4/xxx	See footnote[1]	Console SHOW VERSION command
	AlphaServer 2000 4/xxx	See footnote[1]	
	AlphaServer 2100 4/xxx	See footnote[1]	
	AlphaStation 200 4/xxx	See footnote[1]	
	AlphaStation 250 4/xxx	See footnote[1]	
	AlphaStation 400 4/xxx	See footnote[1]	
Disks	RZ26	392A	Console SHOW DEVICE command
	RZ26L	442D	
	RZ28	442D	
	RZ28B	0006	
	RZ29B	0006	
Controller	HSZ40	2.5	Console SHOW DEVICE command
Bus Isolators	DWZZA–AA	E01	Examine product sticker
	DWZZA–VA	F01	
Adapters[2]	Integral system adapter	N/A	
	KZPAA (PCI to SCSI)		

[1] The minimum revision of this component for SCSI VMScluster configurations is the version included in the Version 3.2 Firmware Kit on the May, 1995 CD–ROM. The revision number is listed in the *Firmware Release Notes Overview* that accompanies that Firmware Kit.

[2] You can configure other types of SCSI adapters in a system for single-host access to local storage.

7.4 SCSI Interconnect Concepts

The SCSI standard defines a set of rules governing the interactions between initiators (typically, host systems) and SCSI targets (typically, peripheral devices). This standard allows the host to communicate with SCSI devices (such as disk drives, tape drives, printers, and optical media devices) without having to manage the device-specific characteristics.

The following sections describe the SCSI standard and the default modes of operation. The discussions also describe some optional mechanisms you can implement to enhance the default SCSI capabilities in areas such as capacity, performance, availability, and distance.

7.4.1 Number of Devices

The SCSI bus is an I/O interconnect that can support up to eight devices. The devices can include host adapters, peripheral controllers, and discrete peripheral devices such as disk or tape drives. The devices are addressed by a unique ID number from 0 through 7. You assign the device IDs by entering console commands, or by setting jumpers or switches.

To increase the number of devices on the SCSI interconnect, some devices implement a second level of device addressing using logical unit numbers (LUNs). For each device ID, up to eight LUNs (0–7) can be used to address a single SCSI device as multiple units. The maximum number of LUNs per device ID is eight.

Note

When connecting devices to a SCSI interconnect, each device on the interconnect must have a unique device ID. You may need to change a device's default device ID to make it unique. For information about setting a device's ID, refer to the owner's guide for the device.

7.4.2 Performance

The default mode of operation for all SCSI devices is 8-bit asynchronous mode. This mode, sometimes referred to as narrow mode, transfers 8 bits of data from one device to another. Each data transfer is acknowledged by the device receiving the data. Because the performance of the default mode is limited, the SCSI standard defines optional mechanisms to enhance performance. The following list describes two optional methods for achieving higher performance:

- Increase the amount of data that is transferred in parallel on the interconnect. The 16-bit and 32-bit wide options allow a doubling or quadrupling of the data rate, respectively. Because the 32-bit option is seldom implemented, this chapter discusses only 16-bit operation and refers to it using the term **wide**.
- Use synchronous data transfer. In synchronous mode, multiple data transfers can occur in succession, followed by an acknowledgment from the device receiving the data. The standard defines a Slow Mode (also called Standard Mode) and a Fast Mode for synchronous data transfers:
  - In Standard Mode, the interconnect achieves up to 5 million transfers per second.
  - In Fast Mode, the interconnect achieves up to 10 million transfers per second.

Because all communications on a SCSI interconnect occur between two devices at a time, each pair of devices must negotiate to determine which of the optional features they will use. Most, if not all, SCSI devices implement one or more of these options.

Table 7–3 shows data rates when using 8- and 16-bit transfers with Standard and Fast synchronous modes.

Table 7–3 Maximum Data Transfer Rates in Megabytes per Second

Mode	Narrow (8-Bit)	Wide (16-Bit)
Standard	5	10
Fast	10	20

7.4.3 Distance

The maximum length of the SCSI interconnect is determined by the signaling method used in the configuration and, in some cases, by the data transfer rate. There are two types of electrical signaling for SCSI interconnects:

- Single-ended signaling

  The single-ended method is the most common and the least expensive. It can operate in either Standard or Fast Mode. The mode used determines the length of the interconnect, as follows:

  - When standard transfers are in use, the interconnect can be up to 6 meters in length[1]
  - When fast transfers are in use, the interconnect can be up to 3 meters in length

- Differential signaling

  This method provides higher signal integrity, thereby allowing a SCSI bus to span distances of up to 25 meters. Differential signaling allows both standard and fast data transfers regardless of the length of the SCSI bus.

When considering cable distance issues, be sure to include both internal and external cabling in your calculations. Table 7–5 lists the internal cable lengths for various configurations.

Table 7–4 summarizes how the type of signaling method affects SCSI interconnect distances.

Table 7–4 Maximum SCSI Interconnect Distances

Signaling Technique	Rate of Data Transfer	Maximum Cable Length
Single ended	Standard	6 meters†
Single ended	Fast	3 meters
Differential	Standard or Fast	25 meters

†The SCSI standard specifies a maximum length of 6 meters for this type of interconnect. However, Digital recommends that, where possible, you limit the cable length to 4 meters to ensure the highest level of data integrity.

A **DWZZA converter** is a single-ended to differential converter that you can use to connect single-ended and differential SCSI interconnect segments. The differential segments are useful for the following:

- Overcoming the distance limitations of the single-ended interconnect
- Allowing communication between single-ended and differential devices

[1] See the note following Table 7–4 for information about the maximum length recommended by Digital.

Because the DWZZA is strictly a signal converter, you do not need to assign a SCSI device ID to it. You can configure a maximum of two DWZZA converters in the path between any two hosts.

7.4.4 Cabling and Termination

Each single-ended and differential SCSI interconnect must have two terminators, one at each end. The specified maximum interconnect lengths are measured from terminator to terminator.

The interconnect terminators are powered from the SCSI interconnect line called TERMPWR. Each Digital host adapter and enclosure supplies the TERMPWR interconnect line, so that as long as one host or enclosure is powered on, the interconnect remains terminated.

Devices attach to the interconnect by short cables (or etch), called stubs. Stubs must be short in order to maintain the signal integrity of the interconnect. The maximum stub lengths allowed are determined by the type of signaling used by the interconnect, as follows:

- For single-ended interconnects, the maximum stub length is .1 meters
- For differential interconnects, the maximum stub length is .2 meters

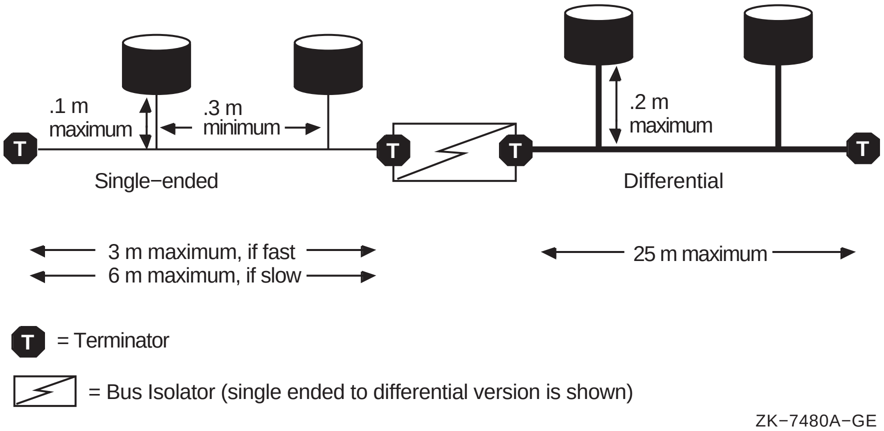
Additionally, the minimum distance between stubs on a single-ended interconnect is .3 meters. Refer to Figure 7–3 for an example of this configuration.

Note

Terminate single-ended and differential buses individually, even when using DWZZA converters.

When you are extending the SCSI bus beyond an existing terminator, it is necessary to disable or remove that terminator.

Figure 7–3 Maximum Stub Lengths



When using the host system's internal SCSI adapter, you must configure the system at the end of the single-ended SCSI segment. This is because the internal SCSI cable lengths exceed the allowable SCSI stub length. However, host systems are not required to be configured at the ends of the bus segment when an add-on SCSI adapter is used.

See Table 7–5 for information about internal SCSI cable lengths.

7.5 SCSI VMScluster Hardware Configurations

The hardware configuration that you choose depends on a combination of factors:

- Your computing needs—for example, continuous availability, or the ability to disconnect or remove a system from your SCSI VMScluster system
- Your environment—for example, the physical attributes of your computing facility
- Your resources—for example, your capital equipment or the available PCI slots

You can connect up to two hosts on a shared SCSI interconnect. Each host can be connected to one or two shared SCSI interconnects. The number of nonshared SCSI buses that can be configured is limited only by the number of available slots on the host bus.

The following sections provide guidelines for building SCSI configurations and describe potential configurations that might be suitable for various sites.

7.5.1 Systems Using Add-On SCSI Adapters

Shared SCSI bus configurations may use optional add-on KZPAA adapters. These adapters are generally easier to configure than internal adapters because they do not consume any SCSI cable length. Additionally, when you configure systems using KZPAA adapters for the shared SCSI bus, the internal adapter is available for connecting devices that cannot be shared (for example, SCSI tape, floppy, and CD–ROM drives).

When using KZPAA adapters, storage is configured using BA350, BA353, or HSZ40 StorageWorks enclosures. These enclosures are suitable for all data disks, and for shared VMScluster system and quorum disks. By using StorageWorks enclosures, it is possible to shut down individual systems without losing access to the disks.

The following sections describe some SCSI VMScluster configurations that take advantage of add-on adapters.

7.5.1.1 Building a Basic System Using Add-On SCSI Adapters

Figure 7–4 shows a logical representation of a basic configuration using SCSI adapters and a StorageWorks enclosure. This configuration has the advantage of being relatively simple, while still allowing the use of tapes, floppies, CD–ROMs, and disks with nonshared files (for example, page files and swap files) on internal buses. Figure 7–5 shows this type of configuration using AlphaServer 1000 systems and a BA350 enclosure.

The BA350 enclosure uses 0.9 meters of SCSI cabling, and this configuration typically uses two 1-meter SCSI cables. (A BA353 enclosure also uses 0.9 meters, with the same total cable length.) The resulting total cable length of 2.9 meters allows Fast SCSI Mode operation.

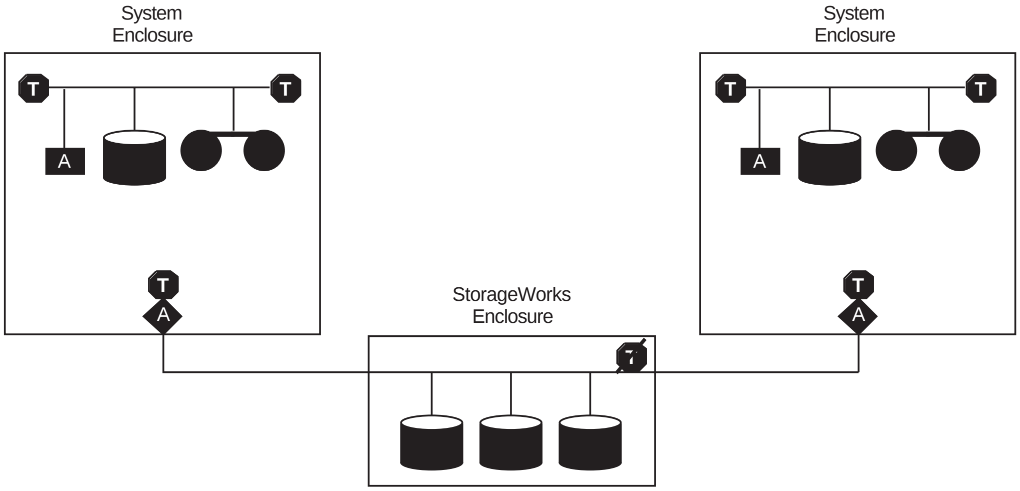
Although the shared BA350 storage enclosure is theoretically a single point of failure, this basic system is a very reliable SCSI VMScluster configuration. When the quorum disk is located in the BA350, you can shut down either of the AlphaStation systems independently while retaining access to the VMScluster system. However, you cannot physically remove the AlphaStation system because that would leave an unterminated SCSI bus.

If you need the ability to remove a system while your VMScluster system remains operational, build your system using DWZZA converters, as described in Section 7.5.1.2. If you need continuous access to data if a SCSI interconnect fails, you should do both of the following:

- Add a redundant SCSI interconnect with another BA350 shelf.
- Shadow the data.

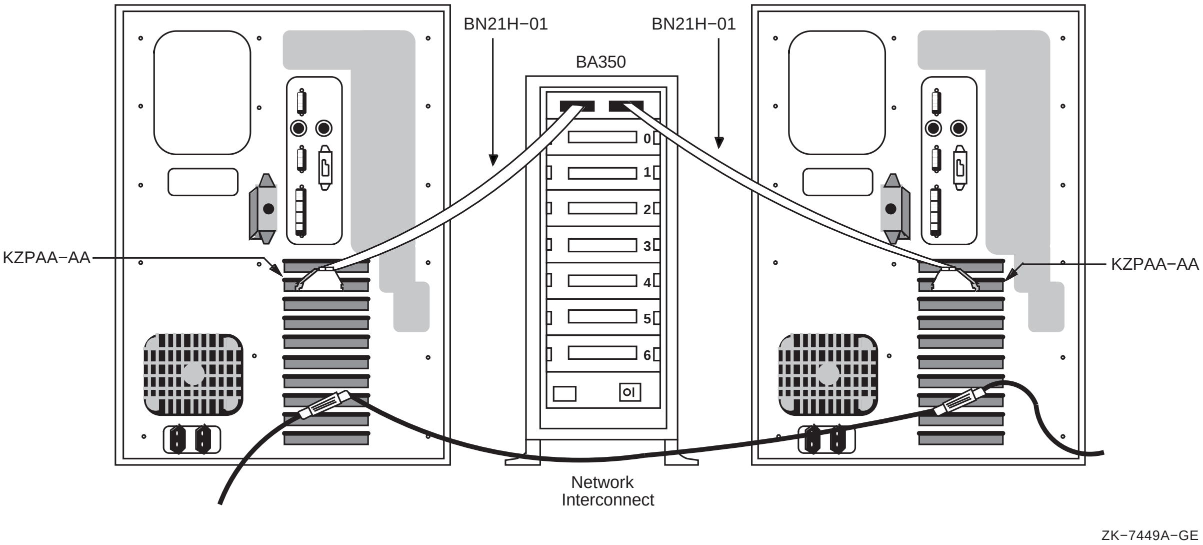
In Figure 7–4 and the other logical configuration diagrams in this chapter, the required network interconnect is not shown. (See Figure 7–1 for the key to the symbols used in the following figures.)

Figure 7–4 Conceptual View: Basic SCSI System



ZK−7501A−GE

Figure 7–5 Sample Configuration: Basic SCSI System Using AlphaServer 1000, KZPAA Adapter, and BA350 Enclosure



7.5.1.2 Building a System That Allows a Server to Be Removed (Using DWZZA Converters)

The capability of removing an individual system from your SCSI VMScluster configuration (for maintenance or repair) while the other systems in the cluster remain active gives you an especially high level of availability. To have this capability, use a configuration that includes a DWZZA converter (a SCSI bus isolator). DWZZA converters provide additional SCSI bus length capabilities because the DWZZA allows you to connect a single-ended device to a bus that uses differential signaling. As described in Section 7.4.3, SCSI bus configurations that use differential signaling may span distances up to 25 meters, whereas single-ended configurations can span only 3 meters when Fast Mode data transfer is used.

DWZZA converters are available as standalone, desktop components or as StorageWorks compatible building blocks. DWZZA converters can be used with the internal SCSI adapter or the optional KZPAA adapters.

Figure 7–6 shows a logical view of a configuration that uses internal SCSI adapters and a pair of bus isolators, and Figure 7–7 shows a physical view of the same configuration using two AlphaServer 1000 systems. In configurations such as those shown in Figure 7–6 and Figure 7–7, you can also remove either of the AlphaServer enclosures because the SCSI bus remains terminated.

In each of these figures, note that a single-ended SCSI bus is used to connect a DWZZA to the AlphaServer systems, and another single-ended bus is used to connect the second DWZZA to the disks. The two DWZZAs are connected to each other by a differential bus. The differential signaling is necessary because the cabling between the DWZZA and the AlphaServer systems consumes virtually all of the 3 meters of single-ended cabling that is allowed for Fast Mode data transfer.

Note

See Figure 7–1 for the key to the symbols used in the following figures.

Figure 7–6 Conceptual View: SCSI System with Bus Isolator (DWZZA Converter)

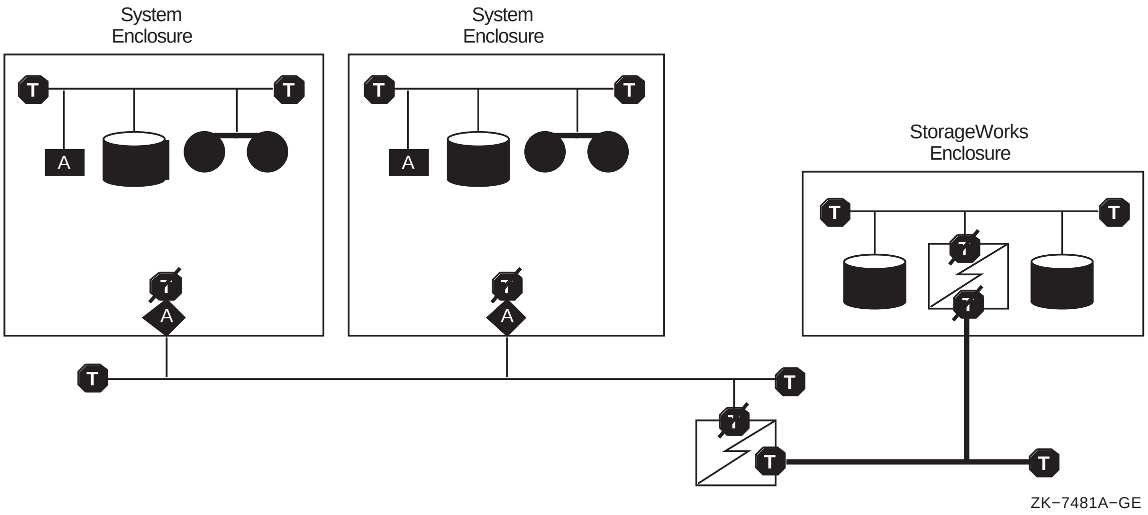
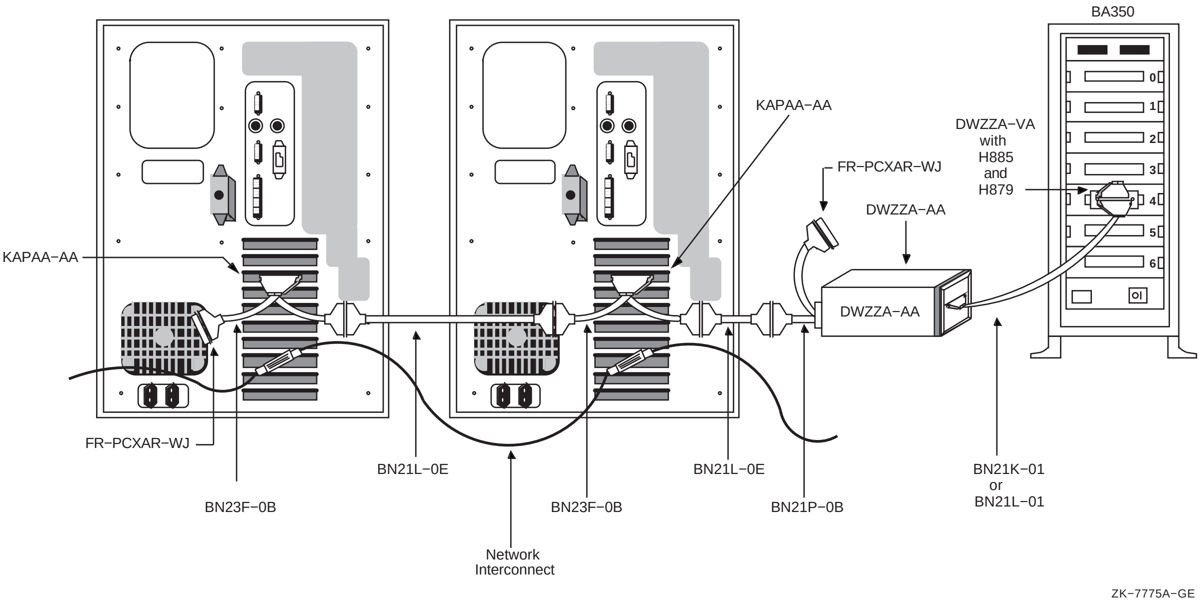


Figure 7–7 Sample Configuration: SCSI System with DWZZA Converter, AlphaServer 1000 Systems, and BA350 Enclosure



7.5.1.3 Building a System That Allows Additional Features and Performance Using an HSZ40 Controller

The HSZ40 is a high-performance differential SCSI controller that can be connected to a differential SCSI bus, and supports up to 42 SCSI devices. An HSZ40 may be configured on a shared SCSI bus that includes DWZZA single-ended to differential converters. Disk devices configured on HSZ40 controllers can be combined into RAID sets to further enhance performance and provide high availability.

Figure 7–8 shows a logical view of a configuration that uses differential SCSI controllers, and Figure 7–9 shows a physical example of the same configuration using an HSZ40 in an SW300 enclosure. Note that the DWZZA is not needed in the StorageWorks enclosure in this type of configuration because the HSZ40 is compatible with differential signaling.

Figure 7–8 Conceptual View: System Using Differential Controllers

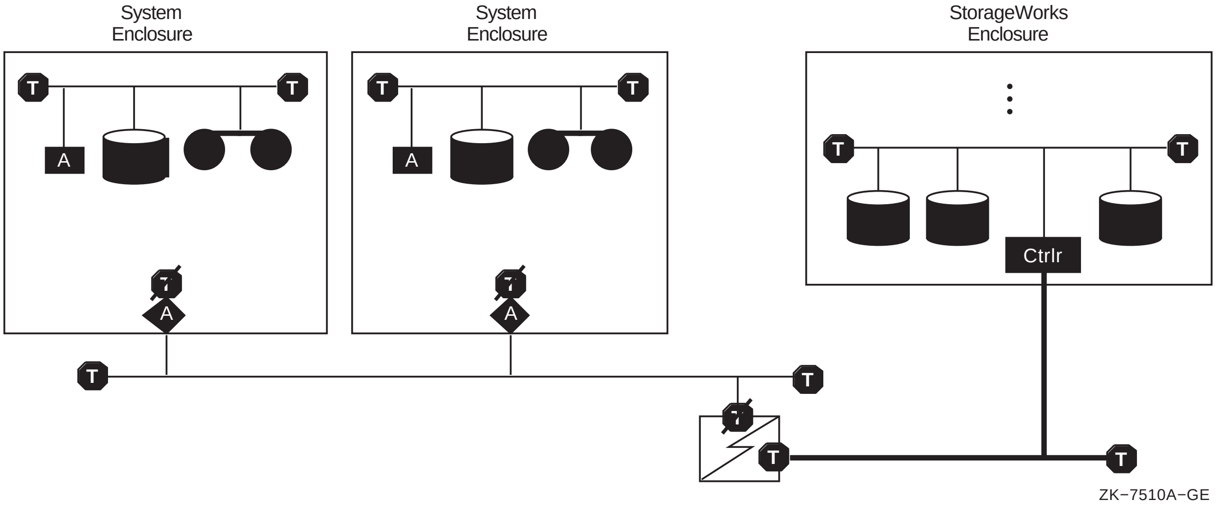
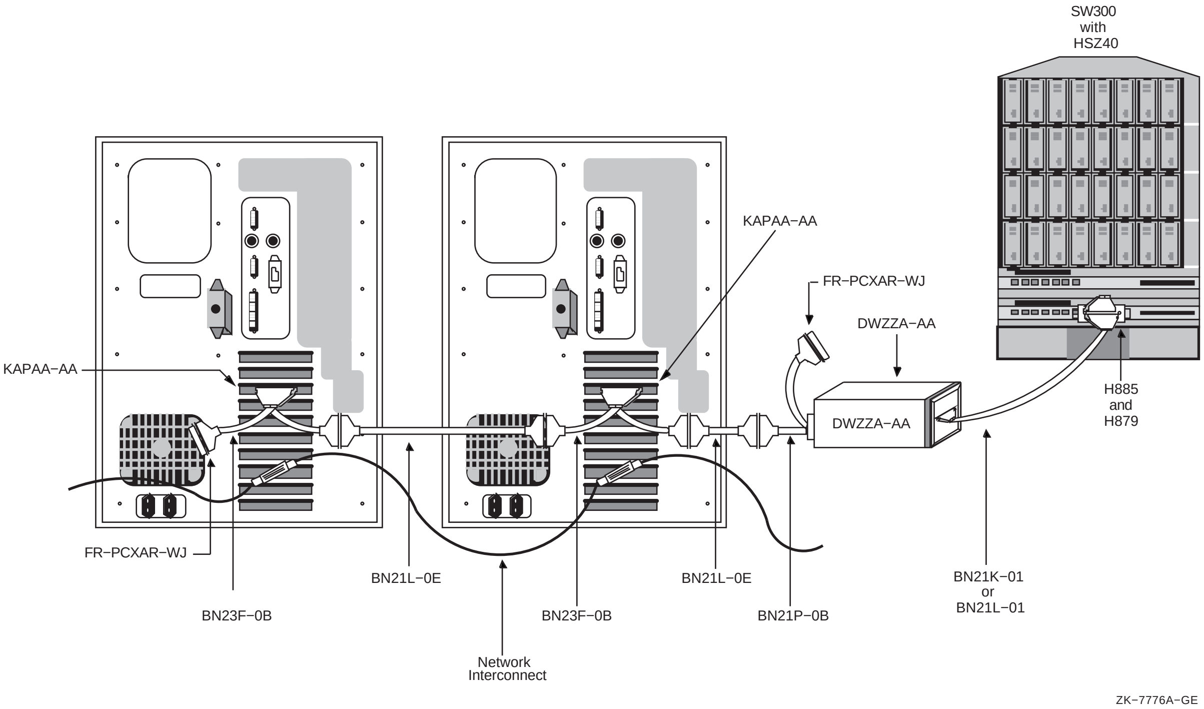


Figure 7–9 Sample Configuration: System Using HSZ40 Controller in an SW300 Enclosure



7.5.1.4 Building a System with More Enclosures or Greater Separation

If you need additional enclosures, or if the needs of your site require a greater physical separation between systems, you can use a configuration in which DWZZAs are placed between systems with single-ended signaling and a differential-cabled SCSI bus.

Figure 7–10 shows a logical view of a configuration that uses additional DWZZAs to increase the potential physical separation (or to allow for additional enclosures), and Figure 7–11 shows a sample representation of this configuration.

Figure 7–10 Conceptual View: Using DWZZAs to Allow for Increased Separation or More Enclosures

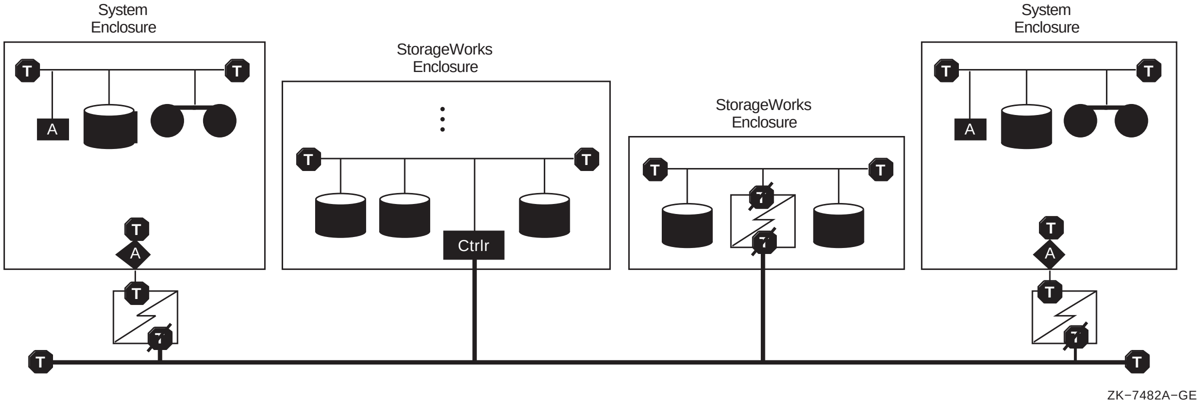
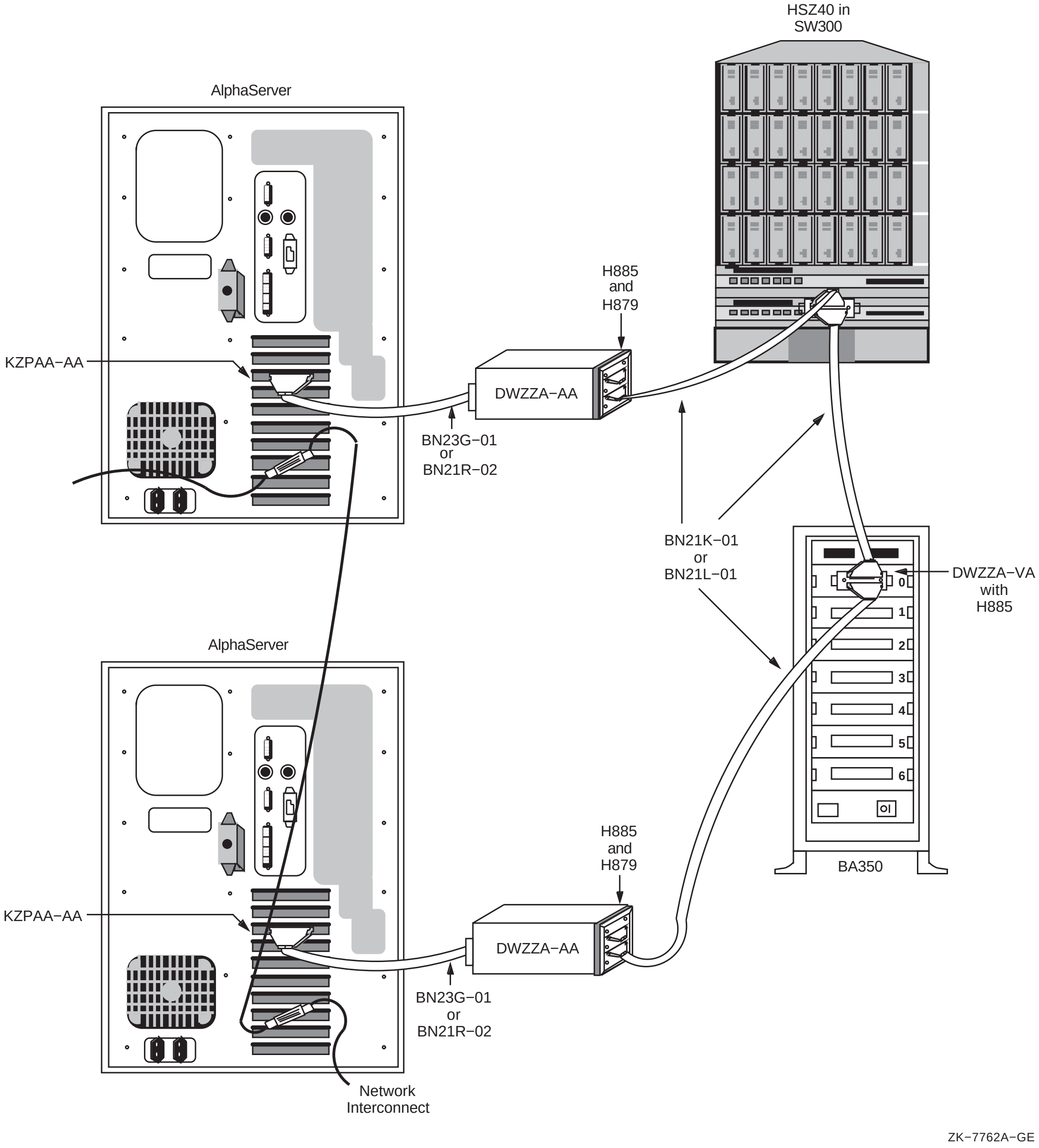


Figure 7–11 Sample Configuration: Using DWZZAs to Allow for Increased Separation or More Enclosures

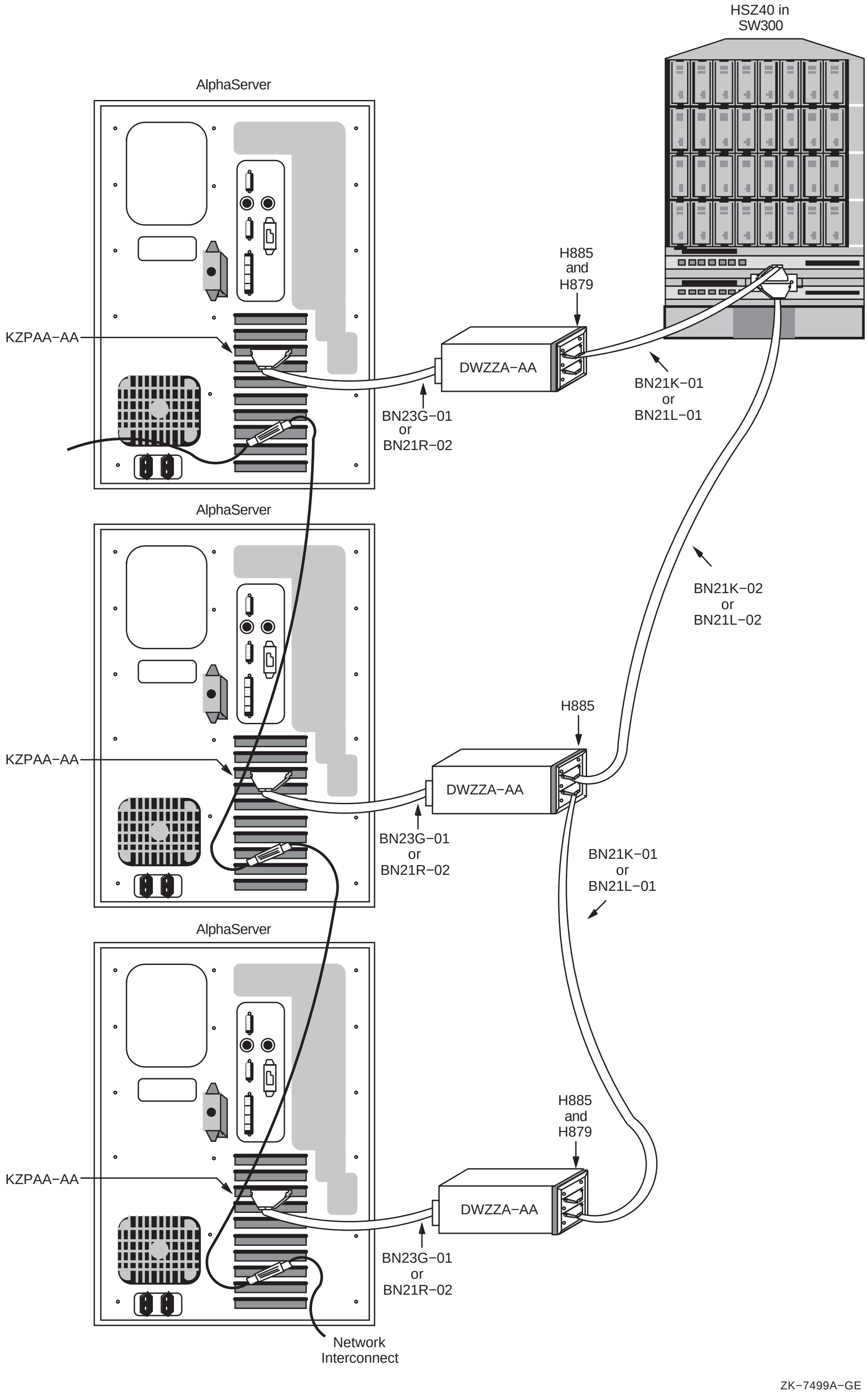


7.5.1.5 Planning for the Future

In OpenVMS Version 6.2, you can connect up to two hosts on a multiple-host SCSI bus. In the future, Digital expects to provide the capability for connecting up to three hosts on the same multiple-host SCSI bus.

Although this capability is not currently supported, Figure 7–12 is included to show how a three-host SCSI VMScluster system might be configured.

Figure 7–12 Sample Configuration: Anticipated Future Support for Three Hosts on a SCSI Bus



7.5.2 Building a System Using Internal SCSI Adapters

You can build a multiple-host SCSI VMScluster configuration with two systems using internal adapters that are joined by a single SCSI cable. This type of configuration is relatively inexpensive, and it provides some of the benefits of multiple-host SCSI VMScluster systems that use external adapters (for example, fully shared disks and twice the serving performance of a single system). This system configuration can also be expanded to provide improved performance, availability, and scaling.

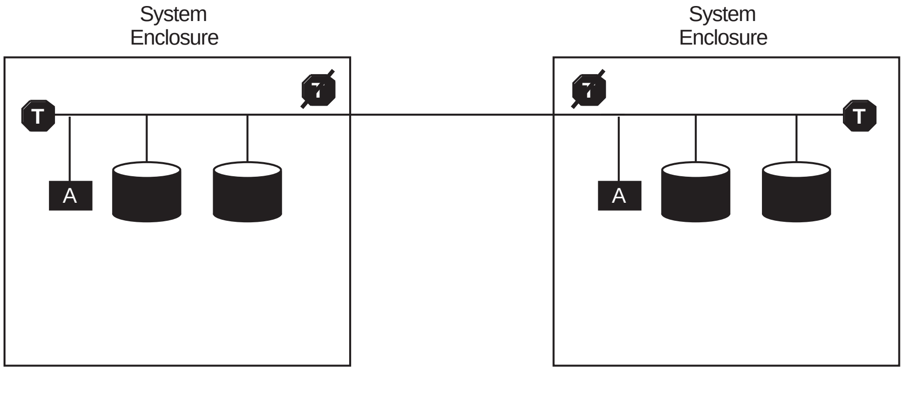
However, a multiple-host SCSI VMScluster system that uses only internal SCSI adapters has the following limitations:

- On most systems, the internal cabling lengths exceeds 3 meters and therefore precludes the use of Fast Mode data transfer (see Table 7–5).
- You cannot remove either of the AlphaServer systems for maintenance or repair while the remaining cluster systems stay active (because the bus would be unterminated).
- If these are the only members of the cluster, then quorum is lost when one (or either, depending on how the votes are allocated) of the enclosures goes down.
- You cannot use tape or CD–ROM drives.

Some of the limitations associated with the internal adapter can be removed by using DWZZAs, additional SCSI adapters, and additional storage enclosures.

Figure 7–13 shows a conceptual view of a SCSI system using internal adapters, and Figure 7–14 shows a sample configuration of such a system.

Figure 7–13 Conceptual View: SCSI VMScluster System Using Internal Adapters



ZK−7760A−GE

Figure 7–14 Sample Configuration: SCSI VMScluster System with AlphaStation 200 Systems Using Internal Adapters

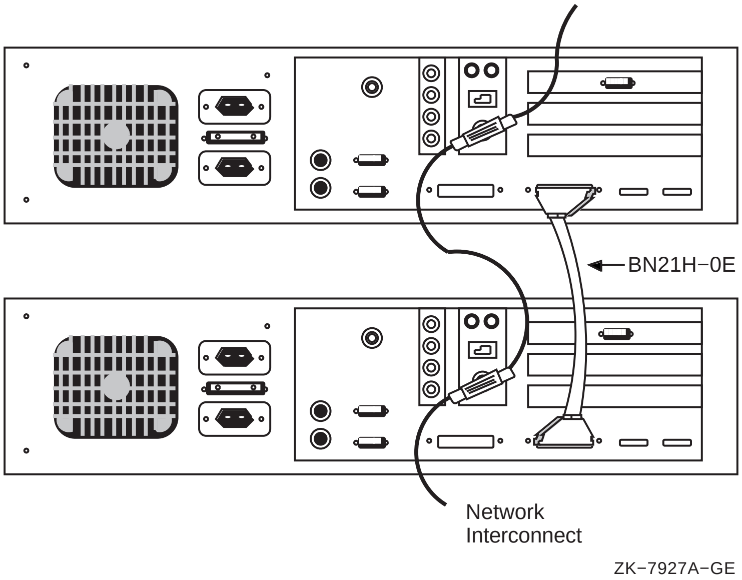


Table 7–5 Internal SCSI Cable Lengths

System Type	Internal Cable Length
AlphaServer 1000 rackmount	1.6 meters
AlphaServer 1000 pedestal with an internal StorageWorks shelf in a dual-bus configuration[1]	2.0 meters
AlphaServer 2000 pedestal with the internal StorageWorks shelf that is not connected	1.7 meters
AlphaServer 2100 rackmount	2.0 meters
AlphaServer 2100 pedestal with the internal StorageWorks shelf that is not connected	1.6 meters
AlphaStation 200	1.2 meters
AlphaStation 400	1.4 meters

[1] See your hardware manual for an explanation of single-bus and dual-bus configurations, and how to switch from one to the other.

7.6 Installation

This section describes the steps required to set up and install the hardware in a SCSI VMScluster system. The assumption in this section is that a new VMScluster system, based on a shared SCSI bus, is being created. If, on the other hand, you are adding a shared SCSI bus to an existing VMScluster configuration, then you should integrate the procedures in this section with those described in *VMScluster Systems for OpenVMS* to formulate your overall installation plan.

Table 7–6 lists the steps required to set up and install the hardware in a SCSI VMScluster system.

Table 7–6 Steps for Installing a SCSI VMScluster System

	Description	Reference
1	Ensure proper grounding between enclosures.	Section 7.6.1 and Section 7.7.8
2	Configure SCSI host IDs.	Section 7.6.2
3	Power up the system and verify devices.	Section 7.6.3
4	Set SCSI console parameters.	Section 7.6.4
5	Install the OpenVMS operating system.	Section 7.6.5
6	Configure additional systems.	Section 7.6.6

7.6.1 Step 1: Meet SCSI Grounding Requirements

You must ensure that your electrical power distribution systems meet local requirements (for example, electrical codes) prior to installing your VMScluster system. If your configuration consists of two or more enclosures connected by a common SCSI interconnect, you must also ensure that the enclosures are properly grounded. Proper grounding is important for safety reasons and to ensure the proper functioning of the SCSI interconnect.

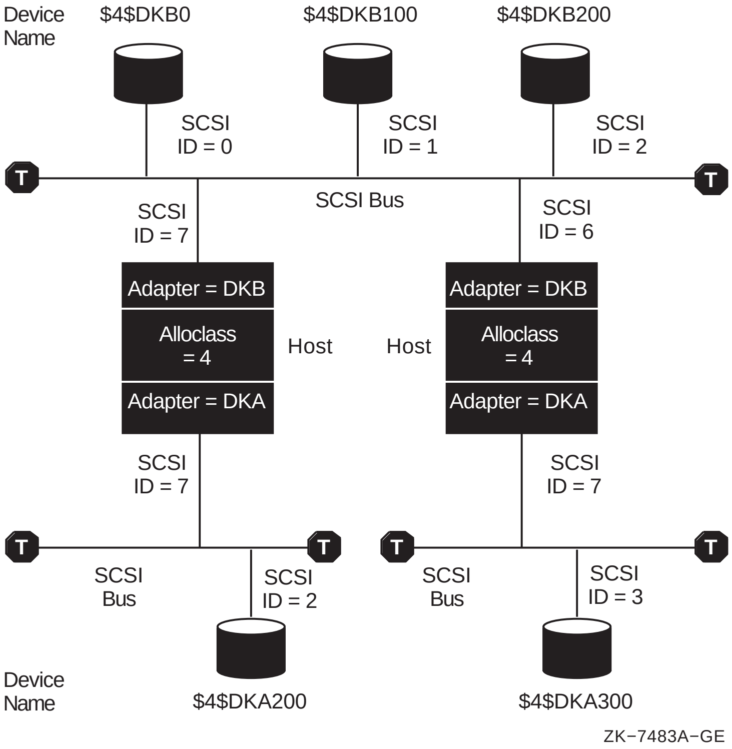
Electrical work should be done by a qualified professional. Section 7.7.8 includes details of the grounding requirements for SCSI systems.

7.6.2 Step 2: Configure SCSI Node IDs

This section describes how to configure SCSI node and device IDs. SCSI IDs must be assigned separately for multiple-host SCSI buses and single-host SCSI buses.

Figure 7–15 shows two hosts; each one is configured with a single-host SCSI bus and shares a multiple-host SCSI bus.

Figure 7–15 Setting Allocation Classes for SCSI Access



The following sections describe how IDs are assigned in this type of multiple-host SCSI configuration. For more information about this topic, see *VMScluster Systems for OpenVMS*.

7.6.2.1 Configuring Device IDs on Multiple-Host SCSI Buses

When configuring multiple-host SCSI buses, adhere to the following rules:

- Set each host adapter on the multiple-host bus to a different ID. Start by assigning ID 7, then ID 6, and so on, using decreasing ID numbers.

  If a host has two multiple-host SCSI buses, allocate an ID to each SCSI adapter separately. There is no requirement that you set the adapters to the same ID, although using the same ID may simplify configuration management. (Section 7.6.4 describes how to set host IDs for the internal adapter using SCSI console parameters.)

- When assigning IDs to devices and storage controllers connected to multiple-host SCSI buses, Digital recommends starting at ID 0 (zero), assigning the highest ID numbers to the disks that require the fastest I/O response time.

- Devices connected to a multiple-host SCSI bus must have the same name as viewed from each host. To achieve this, ensure that:

  – All hosts connected to a multiple-host SCSI bus are set to the same allocation class

  – All host adapters connected to a multiple-host SCSI bus have the same controller letter

7.6.2.2 Configuring Device IDs on Single-Host SCSI Buses

The following discussion applies to hosts that include both a single-host SCSI bus and a multiple-host SCSI bus.

In multiple-host SCSI configurations, device names generated by OpenVMS use the format $*allocation_class*$DKA300. You set the allocation class using the ALLOCLASS system parameter. OpenVMS generates the controller letter (for example, A, B, C, and so forth) at boot time by allocating a letter to each controller. The unit number (for example, 0, 100, 200, 300, and so forth) is derived from the SCSI device ID.

When configuring devices on single-host SCSI buses that are part of a multiple-host SCSI configuration, take care to ensure that the disks connected to the single-host SCSI buses have unique device names. Do this by assigning different IDs to devices connected to single-host SCSI buses with the same controller letter on systems that use the same allocation class. Note that the device names must be different, even though the bus is not shared.

For example, in Figure 7–15, the two disks at the bottom of the picture are located on SCSI bus A of two systems that use the same allocation class. Therefore, they have been allocated different device IDs (in this case 2 and 3).

For a given allocation class, SCSI device type, and controller letter (in this example, “$4$DKA”), there can be up to 8 devices in the cluster, one for each SCSI bus ID. To use all 8 IDs, it is necessary to configure a disk on one SCSI bus at the same ID as a processor on another bus. (See Section 7.7.5 for a discussion of the possible performance impact this can have.)

SCSI bus IDs can be effectively “doubled up” by configuring different SCSI device types at the same SCSI ID on different SCSI buses. For example, device types “DK” and “MK” could produce “$4$DKA100” and “$4$MKA100”.

7.6.3 Step 3: Power Up and Verify SCSI Devices

After connecting the SCSI cables, power up the system. Enter a console SHOW DEVICE command to verify that all devices are visible on the SCSI interconnect.

If there is a SCSI ID conflict, the display may omit devices that are present, or it may include nonexistent devices. If the display is incorrect, then check the SCSI ID jumpers on devices, the automatic ID assignments provided by the StorageWorks shelves, and the console settings for host adapter and HSZ40 controller IDs. If changes are made, type INIT, then SHOW DEVICE again. If problems persist, check the SCSI cable lengths and termination.

The following is a sample output from a console SHOW DEVICE command. This system has one host SCSI adapter on a private SCSI bus (pka0), and two additional SCSI adapters (pkb0 and pkc0), each on separate, shared SCSI buses.

```
>>>SHOW DEVICE
dka0.0.0.6.0               DKA0 1                        RZ26L  442D
dka400.4.0.6.0             DKA400                        RRD43  2893
dkb100.1.0.11.0            DKB100                         RZ26  392A
dkb200.2.0.11.0            DKB200                        RZ26L  442D
dkc400.4.0.12.0            DKC400                        HSZ40   V25
dkc401.4.0.12.0            DKC401 2                      HSZ40   V25
dkc500.5.0.12.0            DKC500                        HSZ40   V25
dkc501.5.0.12.0            DKC501                        HSZ40   V25
dkc506.5.0.12.0            DKC506                        HSZ40   V25
dva0.0.0.0.1               DVA0
jkb700.7.0.11.0            JKB700 3                     OpenVMS  V62
jkc700.7.0.12.0            JKC700                       OpenVMS  V62
mka300.3.0.6.0             MKA300 4                      TLZ06  0389
era0.0.0.2.1               ERA0                   08-00-2B-3F-3A-B9
pka0.7.0.6.0               PKA0 5                      SCSI Bus ID 7
pkb0.6.0.11.0              PKB0                        SCSI Bus ID 6
pkc0.6.0.12.0              PKC0                        SCSI Bus ID 6
```

The following list describes the device names in the preceding example:

1 DK devices represent SCSI disks. Disks connected to the SCSI bus controlled by adapter PKA are given device names starting with the letters DKA. Disks on additional buses are named according to the host adapter name in a similar manner (DKB devices on adapter PKB, and so on).

  The next character in the device name (0 in this case) represents the device's SCSI ID. Make sure that the SCSI ID for each device is unique for the SCSI bus to which it is connected.

2 The last digit in the DK device name (1 in this case) represents the LUN number. The HSZ40 virtual DK device in this example is at SCSI ID 4, LUN 1. Note that some systems do not display devices that have nonzero LUNs.

3 JK devices represent nondisk or nontape devices on the SCSI interconnect. In this example, JK devices represent other processors on the SCSI interconnect that are running the OpenVMS operating system. If the other system is not running, these JK devices do not appear in the display. In this example, the other processor's adapters are at SCSI ID 7.

4 MK devices represent SCSI tapes. The third character in this device's name is A, indicating that it is attached to adapter pka0, the private SCSI bus.

5 PK devices represent the local SCSI adapters. The information in the rightmost column indicates this adapter's SCSI ID. Make sure this is different from the IDs used by other devices and host adapters on its bus.

  The third character in the device name (in this example, a) is assigned by the system so that each adapter has a unique name on that system. The fourth character is always zero.

Note

Make sure that all host adapters attached to a SCSI interconnect have the same name; that is, the third character in the device names must be the same. OpenVMS configures the internal SCSI adapter as DKA, the first KZPAA adapter as DKB, and so on. For example, to ensure that device names are consistent on a shared SCSI bus, do not connect the internal adapter in one system and a KZPAA adapter in a second system to the same bus. For more information, see Section 7.7.4.2.1.

7.6.4 Step 4: Show and Set SCSI Console Parameters

When creating a SCSI VMScluster system, you need to verify the settings of the console environment parameters shown in Table 7–7 and, if necessary, reset their values according to your configuration requirements.

Table 7–7 provides a brief description of SCSI console parameters. Refer to your system-specific documentation for complete information about setting these and other system parameters.

> **Note**
>
> If you need to modify any parameters, first change the parameter (using the appropriate console SET command), and then enter a console INIT command or press the Reset button to make the change effective.

Table 7–7 SCSI Environment Parameters

Parameter	Description
bootdef_dev *device_name*	Specifies the default boot device to the system.
boot_osflags *root_number,bootflag*	The boot_osflags variable contains information that is used by the operating system to determine optional aspects of a system bootstrap (for example, conversational bootstrap).
pk*0_disconnect	Allows the target to disconnect from the SCSI bus while the target acts on a command. When this parameter is set to 1, the target is allowed to disconnect from the SCSI bus while processing a command. When the parameter is set to 0, the target retains control of the SCSI bus while acting on a command.
pk*0_fast	Enables SCSI adapters to perform in Fast SCSI Mode. When this parameter is set to 1, the default speed is set to Fast Mode; when the parameter is 0, the default speed is Standard Mode.
pk*0_host_id	Sets the SCSI device ID of host adapters to a value between 0 and 7.
scsi_poll	Enables console polling on all SCSI interconnects when the system is halted.
control_scsi_term	Enables and disables the terminator on the integral SCSI interconnect at the system bulkhead (for some systems).

Before setting boot parameters, display the current settings of these parameters, as shown in the following examples:

Examples

1. `>>>SHOW *BOOT*`

```
boot_osflags            10,0
boot_reset              OFF
bootdef_dev             dka200.2.0.6.0
>>>
```

The first number in the boot_osflags parameter specifies the system root. (In this example, the first number is 10.) The boot_reset parameter controls the boot process. The default boot device is the device from which the OpenVMS operating system is loaded. Refer to the documentation for your specific system for additional booting information.

Note that you can identify multiple boot devices to the system. By doing so, you cause the system to search for a bootable device from the list of devices that you specify. The system then automatically boots from the first device on which it finds bootable system software. In addition, you can override the default boot device by specifying an alternative device name on the boot command line.

Typically, the default boot flags suit your environment. You can override the default boot flags by specifying boot flags dynamically on the boot command line with the -flags option.

2.
   ```
   >>>SHOW *PK*
   ```

   ```
   pka0_disconnect         1
   pka0_fast               1
   pka0_host_id            7
   ```

   The pk*0_disconnect parameter determines whether or not a target is allowed to disconnect from the SCSI bus while it acts on a command. On a multiple-host SCSI bus, the pk*0_disconnect parameter *must* be set to 1 so that disconnects can occur.

   The pk*0_fast parameter controls whether Fast SCSI devices on a SCSI controller perform in Standard or Fast Mode. When the parameter is set to 0, the default speed is set to Standard Mode; when the pk*0_fast parameter is set to 1, the default speed is set to Fast SCSI Mode. In this example, devices on SCSI controller pka0 are set to Fast SCSI Mode. This means that both Standard and Fast SCSI devices connected to this controller will automatically perform at the appropriate speed for the device (that is, in either Fast or Standard Mode).

   The pk*0_host_id parameter assigns a bus node ID for the specified host adapter. In this example, pka0 is assigned a SCSI device ID of 7.

3.
   ```
   >>>SHOW *POLL*
   scsi_poll               ON
   ```

   Enables or disables polling of SCSI devices while in console mode.

   Set polling ON or OFF depending on the needs and environment of your site. When polling is enabled, the output of the SHOW DEVICE is always up-to-date. However, because polling can consume SCSI bus bandwidth (proportional to number of unused SCSI IDs), you might want to disable polling if one system on a multiple-host SCSI bus will be in console mode for an extended period of time.

   Polling *must* be disabled during any hot plugging operations. For information on hot plugging in a SCSI VMScluster environment, see Section 7.7.6.

4.
   ```
   >>>SHOW *TERM*
   control_scsi_term       external
   ```

   Used on some systems (such as the AlphaStation 400) to enable or disable the SCSI terminator next to the external connector. Set the control_scsi_term parameter to external if a cable is attached to the bulkhead. Otherwise, set the parameter to internal.

7.6.5 Step 5: Install the OpenVMS Operating System

Refer to the OpenVMS Alpha or VAX upgrade and installation manual for information about installing the OpenVMS operating system. Perform the installation once for each system disk in the VMScluster system. In most configurations, there is a single system disk. Therefore, you need to perform this step once, using any system.

During the installation, when you are asked if the system is to be a cluster member, answer Yes. Then, complete the installation according to the guidelines provided in *VMScluster Systems for OpenVMS*.

7.6.6 Step 6: Configure Additional Systems

Use the CLUSTER_CONFIG command procedure to configure additional systems. Execute this procedure once for the second host that you have configured on the SCSI bus. (See Section 7.7.1 for more information.)

7.7 Supplementary Information

The following sections provide supplementary technical detail and concepts about SCSI VMScluster systems.

7.7.1 Running the CLUSTER_CONFIG Command Procedure

You execute the CLUSTER_CONFIG.COM command procedure to set up and configure nodes in your VMScluster system.† Typically, the first computer is set up as a VMScluster system during the initial OpenVMS installation procedure (see Section 7.6.5). The CLUSTER_CONFIG procedure is then used to configure additional nodes. However, if you originally installed OpenVMS without enabling clustering, the first time you run CLUSTER_CONFIG, the procedure converts the standalone system to a cluster system.

To configure additional nodes in a SCSI VMScluster, execute CLUSTER_CONFIG.COM for each additional node. Table 7–8 describes the steps to configure additional SCSI nodes.

Table 7–8 Steps for Installing Additional Nodes

	Procedure
1	From the first node, run the CLUSTER_CONFIG.COM procedure and select the default option [1] for ADD.
2	Answer Yes when CLUSTER_CONFIG.COM asks if you want to proceed.
3	Supply the DECnet name and address of the node that you are adding to the existing single-node cluster.
4	Confirm that this will be a node with a shared SCSI interconnect.
5	Answer No when the procedure asks if this node will be a satellite.
6	Configure the node to be a disk server if it will serve disks to other cluster members.
7	Place the new node's system root on the default device offered.

(continued on next page)

† In OpenVMS Version 6.2, sites that choose to boot their VMScluster systems using the LANCP utility rather than DECnet use the CLUSTER_CONFIG_LAN.COM procedure. See Section 3.5 for information about the use of this alternative procedure.

Table 7–8 (Cont.) Steps for Installing Additional Nodes

	Procedure
8	Select a system root for the new node. The first node uses SYS0. Take the default (SYS10 for the first additional node), or choose your own root numbering scheme. You can choose from SYS1 to SYS*n*, where *n* is hexadecimal FFFF.
9	Select the default disk allocation class so that the new node in the cluster will use the same ALLOCLASS as the first node.
10	Confirm whether or not there is a quorum disk.
11	Answer the questions about the sizes of the page file and swap file.
12	When CLUSTER_CONFIG.COM completes, boot the new node from the new system root. For example, for SYSFF on disk DKA200, enter the following command: `BOOT -FL FF,0 DKA200` On the BOOT command, you can use the following flags: • -FL indicates boot flags • FF is the new system root • 0 means there are no special boot requirements, such as conversational boot

Example 7–1 shows how to run the CLUSTER_CONFIG.COM procedure to set up an additional node in a SCSI VMScluster.

Example 7–1 Adding a Node to a SCSI VMScluster

```
$ @SYS$MANAGER:CLUSTER_CONFIG

           Cluster Configuration Procedure

    Use CLUSTER_CONFIG.COM to set up or change a VMScluster configuration.
    To ensure that you have the required privileges, invoke this procedure
    from the system manager's account.

    Enter ? for help at any prompt.

            1. ADD a node to a cluster.
            2. REMOVE a node from the cluster.
            3. CHANGE a cluster member's characteristics.
            4. CREATE a duplicate system disk for CLU21.
            5. EXIT from this procedure.

    Enter choice [1]:

    The ADD function adds a new node to a cluster.

    If the node being added is a voting member, EXPECTED_VOTES in
    every cluster member's MODPARAMS.DAT must be adjusted, and the
    cluster must be rebooted.

    WARNING - If this cluster is running with multiple system disks and
              if common system files will be used, please, do not
              proceed unless you have defined appropriate logical
              names for cluster common files in SYLOGICALS.COM.
              For instructions, refer to the VMScluster Systems for
              OpenVMS manual.

              Do you want to continue [N]? y
```

(continued on next page)

Example 7–1 (Cont.) Adding a Node to a SCSI VMScluster

```
    If the new node is a satellite, the network databases on CLU21 are
    updated. The network databases on all other cluster members must be
    updated.

    For instructions, refer to the VMScluster Systems for OpenVMS manual.

What is the node’s DECnet node name? SATURN
What is the node’s DECnet node address? 7.77
Is SATURN to be a clustered node with a shared SCSI bus (Y/N)? y
Will SATURN be a satellite [Y]? N
Will SATURN be a boot server [Y]?

    This procedure will now ask you for the device name of SATURN’s system root.
    The default device name (DISK$BIG_X5T5:) is the logical volume name of
    SYS$SYSDEVICE:.

What is the device name for SATURN’s system root [DISK$BIG_X5T5:]?
What is the name of SATURN’s system root [SYS10]? SYS2
    Creating directory tree SYS2 ...
    System root SYS2 created

    NOTE:
        All nodes on the same SCSI bus must be members of the same cluster
        and must all have the same non-zero disk allocation class or each
        will have a different name for the same disk and data corruption
        will result.

Enter a value for SATURN’s ALLOCLASS parameter [7]:
Does this cluster contain a quorum disk [N]?
Updating network database...
Size of pagefile for SATURN [10000 blocks]?
   .
   .
   .
```

7.7.2 Errors Reports and OPCOM Messages in Multiple-Host SCSI Environments

Certain common operations, such as booting or shutting down a host on a multiple-host SCSI bus, can cause other hosts on the SCSI bus to experience errors. In addition, certain errors that are unusual in a single-host SCSI configuration may occur more frequently on a multiple-host SCSI bus.

These errors are transient errors that OpenVMS detects, reports, and recovers from without losing data or affecting applications that are running. This section describes the conditions that generate these errors and the messages that are displayed on the operator console and entered into the error log.

7.7.2.1 SCSI Bus Resets

When a host connected to a SCSI bus first starts, either by being turned on or by rebooting, it does not know the state of the SCSI bus and the devices on it. The ANSI SCSI–2 Standard provides a method called BUS RESET to force the bus and its devices into a known state. A host typically asserts a RESET signal one or more times on each of its SCSI buses when it first starts up and when it shuts down. While this is a normal action on the part of the host asserting RESET, other hosts consider this RESET signal an error, because RESET requires that the hosts abort and restart all I/O operations that are in progress.

A host may also reset the bus in the midst of normal operation if it detects a problem that it cannot correct in any other way. These kinds of resets are uncommon but occur most frequently when something on the bus is disturbed. For example, an attempt to hot plug a SCSI device while the device is still active (see Section 7.7.6) or halting one of the hosts with CTRL/P can cause a condition that forces one or more hosts to issue a bus reset.

7.7.2.2 SCSI Timeouts

When a host exchanges data with a device on the SCSI bus, there are several different points where the host must wait for the device or the SCSI adapter to react. In an OpenVMS system, the host is allowed to do other work while it is waiting, but a timer is started to make sure that it does not wait too long. If the timer expires without a response from the SCSI device or adapter, this is called a timeout.

There are three kinds of timeouts:

- Disconnect timeout—The device accepted a command from the host and disconnected from the bus while it processed the command but never reconnected to the bus to finish the transaction. This error happens most frequently when the bus is very busy. See Section 7.7.5 for more information. The disconnect timeout period varies with the device, but for most disks, it is about 20 seconds.
- Selection timeout—The host tried to send a command to a device on the SCSI bus, but the device did not respond. This condition might happen if the device did not exist or if it were removed from the bus or powered down, for example. (This failure is not more likely with a multi-initiator system; it is mentioned here for completeness.) The selection timeout period is about 0.25 seconds.
- Interrupt timeout—The host expected the adapter to respond for any other reason, but it did not respond. This error is usually an indication of a busy SCSI bus. It is more common if you have initiator unit numbers set low (0 or 1) rather than high (6 or 7). The interrupt timeout period is about 4 seconds.

Timeout errors are not inevitable on SCSI VMScluster systems. However, they are more frequent on SCSI buses with heavy traffic and those with two initiators. They do not necessarily indicate a hardware or software problem. If they are logged frequently, you should consider ways to reduce the load on the SCSI bus (for example, adding an additional bus).

7.7.2.3 Mount Verify

Mount verify is a condition declared by a host about a device. The host declares this condition in response to a number of possible transient errors, including bus resets and timeouts. When a device is in the mount verify state, the host suspends normal I/O to it until the host can determine that the correct device is there, and that the device is accessible. Mount verify processing then retries outstanding I/Os in a way that ensures that the correct data is written or read. Application programs are unaware that a mount verify condition has occurred as long as the mount verify completes.

If the host cannot access the correct device within a certain amount of time, it declares a mount verify timeout, and application programs are notified that the device is unavailable. Manual intervention is required to restore a device to service after the host has declared a mount verify timeout. A mount verify timeout usually means that the error is not transient. The system manager can choose the timeout period for mount verify; the default is one hour.

7.7.2.4 Shadow Volume Processing

Shadow volume processing is a process similar to mount verify, but it is for shadow set members. An error on one member of a shadow set places the set into the volume processing state, which blocks I/O while OpenVMS attempts to regain access to the member. If access is regained before shadow volume processing times out, then the outstanding I/Os are reissued and the shadow set returns to normal operation. If a timeout occurs, then the failed member is removed from the set. The system manager can select one timeout value for the system disk shadow set, and one for application shadow sets. The default value for both timeouts is 20 seconds.

Note

The SCSI disconnect timeout and the default shadow volume processing timeout are the same. If the SCSI bus is heavily utilized so that disconnect timeouts may occur, it may be desirable to increase the value of the shadow volume processing timeout. (A recommended value is 60 seconds.) This may prevent shadow set members from being expelled when they experience disconnect timeout errors.

7.7.2.5 Expected OPCOM Messages in Multiple-Host SCSI

When a bus reset occurs, an OPCOM message is displayed as each mounted disk enters and exits mount verification or shadow volume processing.

When an I/O to a drive experiences a timeout error, an OPCOM message is displayed as that drive enters and exits mount verification or shadow volume processing.

If a quorum disk on the shared SCSI bus experiences either of these errors, then additional OPCOM messages may appear, indicating that the connection to the quorum disk has been lost and regained.

7.7.2.6 Error-Log Basics

In the OpenVMS system, the Error Log utility allows device drivers to save information about unusual conditions that they encounter. In the past, most of these unusual conditions have happened as a result of errors such as hardware failures, software failures, or transient conditions (like loose cables, for example).

If you type the DCL command SHOW ERROR, the system displays a summary of the errors that have been logged since the last time the system booted. For example:

```
$ SHOW ERROR

Device                            Error Count
SALT$PKB0:                                 6
$1$DKB500:                                10
PEA0:                                      1
SALT$PKA0:                                 9
$1$DKA0:                                   0
```

In this case, 6 errors have been logged against host SALT’s SCSI port B (PKB0), 10 have been logged against disk $1$DKB500, and so forth.

To see the details of these errors, you can use the command ANALYZE/ERROR /SINCE=DD-MMM-YYYY:HH:MM:SS at the DCL prompt. The output from this command displays a list of error-log entries with information similar to the following:

```
******************************* ENTRY    2337. *******************************
 ERROR SEQUENCE 6.                               LOGGED ON:  CPU_TYPE 00000002
 DATE/TIME 29-MAY-1995 16:31:19.79                            SYS_TYPE 0000000D

<identification information>

       ERROR TYPE            03
                                       COMMAND TRANSMISSION FAILURE
       SCSI ID               01
                                       SCSI ID = 1.
       SCSI LUN              00
                                       SCSI LUN = 0.
       SCSI SUBLUN           00
                                       SCSI SUBLUN = 0.
       PORT STATUS     00000E32
                                       %SYSTEM-E-RETRY, RETRY OPERATION

<additional information>
```

For this discussion, the key elements are the “ERROR TYPE” and, in some instances, the “PORT STATUS” fields. In this example, the ERROR TYPE is “03, COMMAND TRANSMISSION FAILURE”, and the PORT STATUS is “00000E32, SYSTEM-E-RETRY”.

7.7.2.7 Error-Log Entries in Multiple-Host SCSI

The error-log entries listed in this section are likely to be logged in a multiple-host SCSI configuration, and you usually do not need to be concerned about them. You should, however, examine any error-log entries for messages other than those listed in this section.

- ERROR TYPE 0007, BUS RESET DETECTED

  Occurs when the other system asserts the SCSI bus reset signal. This happens when a:

  - System’s power-up self-test runs
  - Console INIT command is executed
  - EISA Configuration utility (ECU) is run
  - Console BOOT command is executed (several resets occur in this case)
  - System shutdown completes

  This error causes all mounted disks to enter mount verification.

- ERROR TYPE 05, EXTENDED SENSE DATA RECEIVED

  When a SCSI bus is reset, an initiator must get “sense data” from each device. When the initiator gets this data, an “EXTENDED SENSE DATA RECEIVED” error is logged. This is expected behavior.

- ERROR TYPE 03, COMMAND TRANSMISSION FAILURE
  PORT STATUS E32, SYSTEM-E-RETRY

  Occasionally, one host may send a command to a disk while the disk is exchanging error information with the other host. Many disks respond with a SCSI “BUSY” code. The OpenVMS system responds to a SCSI BUSY code by logging this error and retrying the operation. You are most likely to see this error when the bus has been reset recently. This error does not always happen near resets, but when it does, the error is expected and unavoidable.

- ERROR TYPE 204, TIMEOUT

  An interrupt timeout has occurred (see Section 7.7.2.2). The disk is put into mount verify when this error occurs.

- ERROR TYPE 104, TIMEOUT

  A selection timeout has occurred (see Section 7.7.2.2). The disk is put into mount verify when this error occurs.

7.7.3 Restrictions and Known Problems

The current release of VMScluster software has the following restrictions when multiple hosts are configured on the same SCSI bus:

1. A node's access to a disk will not fail over from a direct SCSI path to an MSCP served path. This is not expected to be a significant limitation since most of the failures that cause a SCSI disk to become inaccessible to one node on the SCSI bus affect all the nodes on the SCSI bus. Thus, when a failure occurs, the served path to the disk tends to fail at the same time that the direct path fails.

   Conversely, a node's access to a disk will not fail over from an MSCP served path to a direct SCSI path. Normally, this type of failover is not a consideration because when OpenVMS discovers both a direct and a served path, it chooses the direct path permanently. It is necessary, however, to avoid situations in which the MSCP served path becomes available first, and is selected by OpenVMS before the direct path becomes available. You must avoid this by observing the following rules:

   - A node that has a direct path to a SCSI system disk must boot the disk directly from the SCSI port, not over the LAN.

   - If a node is running the MSCP server, then a SCSI disk must not be added to the multiple-host SCSI bus after the node boots. This is necessary to prevent the second node on the SCSI bus from seeing the served path to the new disk and configuring it, thereby precluding the second node from configuring its direct path.

2. The SYS$DEVICE_SCAN system service (and the F$DEVICE lexical function that calls it) can be executed repeatedly to obtain a list of devices on the system. If this is done on a system with a multiple-host SCSI bus, and the other system is running the MSCP server, then each device on the multiple-host SCSI bus is reported twice by SYS$DEVICE_SCAN (or F$DEVICE).

   The reason for this is that each device on the multiple-host SCSI bus has two UCBs, one for the direct SCSI path, and one for the MSCP served path. (The MSCP served path is not used.)

   Programs that use SYS$DEVICE_SCAN or F$DEVICE to search the IO database may need to be modified to check for, and ignore, duplicate device names.

3. If a system on a multiple-host SCSI bus boots a disk that is served to it over the LAN, and the other system on the SCSI bus is running the MSCP server, then each device on the multiple-host SCSI bus is reported twice by the DCL SHOW DEVICE command. This not known to result in any other adverse effects.

4. Abruptly halting a system on a multiple-host SCSI bus (by typing CTRL/P on the console, for example) may leave the SCSI adapter in a state that can interfere with the operation of the other host on the bus. It is recommended that you either initialize, boot, or continue an abruptly halted system as soon as possible after it has been halted.
5. All I/O to a disk drive must be stopped while its microcode is updated. This typically requires more precautions in a multiple-host environment than are needed in a single-host environment. Refer to Section 7.7.6.3 for the necessary procedures.
6. The EISA Configuration Utility (ECU) causes a large number of SCSI bus resets. These resets cause the other system on the SCSI bus to pause while its I/O subsystem recovers. It is suggested (though not required) that both systems on a shared SCSI bus be shut down when the ECU is run.

The current release of VMScluster software also places one new restriction on the SCSI quorum disk, whether the disk is located on a single-host SCSI bus or a multiple-host SCSI bus: the SCSI quorum disk must support Tagged Command Queueing. This is required because of the special handling that quorum I/O receives in the OpenVMS SCSI drivers.

This restriction is not expected to be significant because all disks on a multiple-host SCSI bus must support Tagged Command Queueing (see Section 7.7.7), and because quorum disks are normally not used on single-host buses.

7.7.4 Troubleshooting

The following sections describe troubleshooting tips for solving common problems in a VMScluster system using a SCSI interconnect.

7.7.4.1 Termination Problems

Verify that two terminators are on every SCSI interconnect (one at each end of the interconnect). The BA350 enclosure, the DWZZA, and the KZPAA adapter have internal terminators that are not visible externally. (See Section 7.4.4.)

7.7.4.2 Booting or Mounting Failures Caused by Incorrect Configurations

OpenVMS automatically detects configuration errors described in this section and prevents the possibility of data loss that could result from such configuration errors, either by bugchecking or by refusing to mount a disk.

7.7.4.2.1 Bugchecks During the Bootstrap Process There are three types of configuration error can can cause a bugcheck (the bugcheck code is: “VAXCLUSTER, Error detected by VMScluster software” during booting). These errors are described in this section.

When OpenVMS boots, it determines which devices are present on the SCSI bus by sending an inquiry command to every SCSI ID. When a device receives the inquiry, it indicates its presence by returning data that indicates whether it is a disk, tape, or processor.

Some processor devices (host adapters) answer the inquiry without assistance from the operating system; others require that the operating system be running. The adapters supported in VMScluster systems require the operating system to be running. These adapters, with the aid of OpenVMS, pass information in their response to the inquiry that allows the recipient to detect the following configuration errors:

- Different controller device names on the same SCSI bus

The OpenVMS device name of each adapter on the SCSI bus must be identical (all named pkc0, for example), or the VMScluster software cannot coordinate the host's accesses to storage (see Section 7.6.2 and Section 7.6.3).

OpenVMS can check this automatically because it sends the controller letter in the inquiry response. A booting system receives this response, and it compares the remote controller letter with the local controller letter. If a mismatch is detected, then an OPCOM message is printed, and the system stops with an VAXCLUSTER bugcheck to prevent the possibility of data loss. See the description of the NOMATCH error in either the Help Message utility or in Appendix A. (To use the Help Message utility for NOMATCH, enter HELP/MESSAGE NOMATCH at the DCL prompt.)

- Different or zero allocation class values

  Each host on the SCSI bus must have the same nonzero disk allocation class value, or the VMScluster software can not coordinate the host's accesses to storage (see Section 7.6.3 and Section 7.6.2). The disk allocation class value is controlled by the ALLOCLASS SYSGEN parameter.

  OpenVMS is able to automatically check this because it sends the ALLOCLASS value in the inquiry response. A booting system receives this response, and compares the remote ALLOCLASS value with the local ALLOCLASS value. If a mismatch or a zero value is detected, then an OPCOM message is printed, and the system stops with an VAXCLUSTER bugcheck, in order to prevent the possibility of data loss. See the description of the ALLODIFF and ALLOZERO errors in either the Help Message utility or in Appendix A.

- Unsupported processors

  Finally, there may be processors on the SCSI bus that are not running OpenVMS or that do not return the controller name or allocation class information needed to validate the configuration. If a booting system receives an inquiry response and the response does not contain the special OpenVMS configuration information, then an OPCOM message is printed and an VAXCLUSTER bugcheck occurs. See the description of the CPUNOTSUPP error in either the Help Message utility or in Appendix A.

  (If your system requires the presence of a non-VMScluster processor device on a SCSI bus, refer to the CPUNOTSUPP message description in either the Help Message utility or in Appendix A for instructions on the use of a special SYSGEN parameter for this purpose.)

Hint

The OPCOM error code that is printed for each of the failures described above is preserved in R8 of the VAXCLUSTER bugcheck. You can examine register R8 to quickly determine the cause of the error. For example:

```
$ ANAL/CRASH SYSDUMP.DMP
SDA> examine @r8;50
2D462D47 49464E4F 43415453 250A0D4B  K..%STACONFIG-F-     00020430
4C412065 6854202C 46464944 4F4C4C41  ALLODIFF, The AL     00020440
6574656D 61726170 20535341 4C434F4C  LOCLASS paramete     00020450
20656874 20726F66 2065756C 61762072  r value for the      00020460
00000000 6E6F2072 6F737365 636F7270  processor on....     00020470
```

7.7.4.2.2 Mount Failures There are two types of configuration error can can cause a disk to fail to mount. These are described in this section.

First, when a system boots from a disk on the shared SCSI bus, it may fail to mount the system disk. This happens if there is another system on the SCSI bus that is already booted, and the other system is using a different device name for the system disk. (Two systems will disagree about the name of a device on the shared bus if their controller names or allocation classes are misconfigured, as described in the previous section.) If the system does not execute one of the bugchecks described in the previous section first, then the following error message is displayed on the console:

```
%SYSINIT-E- error when mounting system device, retrying..., status = 007280B4
```

The decoded representation of this status is:

```
VOLALRMNT,  another volume of same label already mounted
```

This error indicates that the system disk is already mounted in what appears to be another drive in the VMScluster system, so it is not mounted again. Solve this problem by checking the controller letters and allocation class values for each node on the shared SCSI bus.

Second, SCSI disks on a shared SCSI bus will fail to mount on both systems unless the disk supports Tagged Command Queueing (TCQ). This is because TCQ provides a command ordering guarantee that is required during VMScluster state transitions.

OpenVMS determines that another processor is present on the SCSI bus during autoconfiguration, using the mechanism described in Section 7.7.4.2.1. The existence of another host on a SCSI bus is recorded and preserved until the system reboots.

This information is used whenever an attempt is made to mount a non-TCQ device. If the device is on a multiple-host bus, the mount attempt fails and returns the following message:

```
%MOUNT-F-DRVERR, fatal drive error.
```

If the drive is intended to be mounted by multiple hosts on the same SCSI bus, then it must be replaced with one that supports TCQ.

Note that the first processor to boot on a multiple-host SCSI bus does not receive an inquiry response from the other hosts because the other hosts are not yet running OpenVMS. Thus, the first system to boot is unaware that the bus has multiple hosts, and it allows non-TCQ drives to be mounted. The other hosts on the SCSI bus detect the first host, however, and they are prevented from mounting the device. If two processors boot simultaneously, it is possible that they will detect each other, in which case neither is allowed to mount non-TCQ drives on the shared bus.

7.7.4.3 Grounding

Having excessive ground offset voltages or exceeding the maximum SCSI interconnect length can cause system failures or degradation in performance. See Section 7.7.8 for more information about SCSI grounding requirements.

7.7.4.4 Interconnect Lengths

Adequate signal integrity depends on strict adherence to SCSI bus lengths. Failure to follow the bus length recommendations can result in problems (for example, intermittent errors) that are difficult to diagnose. See Section 7.4.3 for information on SCSI bus lengths.

7.7.5 SCSI Arbitration Considerations

Only one initiator (typically, a host system) or target (typically, a peripheral device) can control the SCSI bus at any one time. In a computing environment where multiple targets frequently contend for access to the SCSI bus, you could experience throughput issues for some of these targets. This section discusses control of the SCSI bus, how that control can affect your computing environment, and what you can do to achieve the most desirable results.

Control of the SCSI bus changes continually. When an initiator gives a command (such as READ) to a SCSI target, the target typically disconnects from the SCSI bus while it acts on the command, allowing other targets or initiators to use the bus. When the target is ready to respond to the command, it must regain control of the SCSI bus. Similarly, when an initiator wishes to send a command to a target, it must gain control of the SCSI bus.

If multiple targets and initiators want control of the bus simultaneously, bus ownership is determined by a process called arbitration, defined by the SCSI Standard. The default arbitration rule is simple: control of the bus is given to the requesting initiator or target that has the highest unit number.

The following sections discuss some of the implications of arbitration and how you can respond to arbitration situations that affect your environment.

7.7.5.1 Arbitration Issues in Multiple-Disk Environments

When the bus is not very busy, and bus contention is uncommon, the simple arbitration scheme is adequate to perform I/O requests for all devices on the system. However, as initiators make more and more frequent I/O requests, contention for the bus becomes more and more common. Consequently, targets with lower ID numbers begin to perform poorly because they are frequently blocked from completing their I/O requests by other users of the bus (in particular, targets with the highest ID numbers). If the bus is sufficiently busy, low-numbered targets may never complete their requests. This situation is most likely to occur on systems with more than one initiator because more commands can be outstanding at the same time.

The OpenVMS system attempts to prevent low-numbered targets from being completely blocked by monitoring the amount of time an I/O request takes. If the request is not completed within a certain period, the OpenVMS system stops sending new requests until the tardy I/Os complete. While this algorithm does not ensure that all targets get equal access to the bus, it does prevent low-numbered targets from being totally blocked.

7.7.5.2 Solutions for Resolving Arbitration Problems

If you find that some of your disks are not being serviced quickly enough during periods of heavy I/O, try some or all of the following, as appropriate for your site:

- Assign the highest ID numbers to those disks that require the fastest response time.
- Spread disks across more SCSI buses.

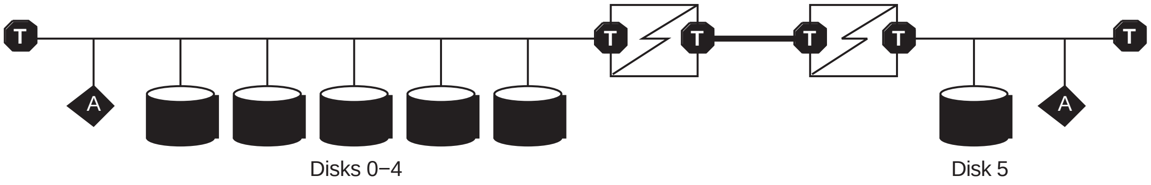
- Keep disks that need to be accessed only by a single host (for example, page and swap disks) on a nonshared SCSI bus.

Another method that might provide for more equal servicing of lower and higher ID disks is to set the host IDs to the lowest numbers (0 and 1) rather than the highest. When you use this method, the host cannot gain control of the bus to send new commands as long as any disk, including those with the lowest IDS, need the bus. Although this option is available to improve fairness under some circumstances, Digital considers this configuration to be less desirable in most instances, for the following reasons:

- It can result in lower total throughput.
- It can result in timeout conditions if a command cannot be sent within a few seconds.
- It can cause physical configuration difficulties. For example, StorageWorks shelves such as the BA350 have no slot to hold a disk with ID 7, but they do have a slot for a disk with ID 0. If you change the host to ID 0, you must remove a disk from slot 0 in the BA350, but you cannot move the disk to ID 7. If you have two hosts with IDs 0 and 1, you cannot use slot 0 or 1 in the BA350. (Note, however, that you *can* have a disk with ID 7 in a BA353.)

7.7.5.3 Arbitration and Bus Isolators

Any active device, such as a DWZZA, that connects bus segments introduces small delays as signals pass through the device from one segment to another. Under some circumstances, these delays can be another cause of unfair arbitration. For example, consider the following configuration, which could result in disk servicing problems (starvation) under heavy work loads:



ZK−7913A−GE

Although disk 5 has the highest ID number, there are some circumstances under which disk 5 has the lowest access to the bus. This can occur after one of the lower numbered disks has gained control of the bus and then completed the operation for which control of the bus was needed. At this point, disk 5 does not recognize that the bus is free and might wait before trying to arbitrate for control of the bus. As a result, one of the lower numbered disks, having become aware of the free bus and then submitting a request for the bus, will gain control of the bus.

If you see this type of problem, the following suggestions can help you reduce its severity:

- Try to place all disks on the same bus segment.
- If placing all disks on the same bus segment is not possible (for example, if you have both some RZ28 disks by themselves and an HSZ40), try to use a configuration that has only one isolator between any pair of disks.
- If your configuration requires two isolators between a pair of disks (for example, to meet distance requirements), try to balance the number of disks on each bus segment.

- Follow the suggestions in Section 7.7.5.2 to reduce the total traffic on the logical bus.

7.7.6 Removal and Insertion of SCSI Devices While the VMScluster System is Operating

With proper procedures, certain SCSI devices can be removed from or inserted onto an active SCSI bus without disrupting the on-going operation of the bus. This capability is referred to as **hot plugging**. Hot plugging can allow a suitably configured VMScluster system to continue to run while a failed component is replaced. Without hot plugging, it is necessary to make the SCSI bus inactive and remove power from all the devices on the SCSI bus before any device is removed from it or inserted onto it.

In a SCSI VMScluster system, hot plugging requires that all devices on the bus have certain electrical characteristics and be configured appropriately on the SCSI bus. Successful hot plugging also depends on strict adherence to the procedures described in this section. These procedures ensure that the hot-plugged device is inactive and that active bus signals are not disturbed.

Hot Plugging for SCSI Buses Behind a Storage Controller

This section describes hot-plugging procedures for devices that are on the same SCSI bus as the host that is running OpenVMS. The procedures are different for SCSI buses that are behind a storage controller, such as the HSZ40. Refer to the storage controller documentation for the procedures to hot plug devices that they control.

7.7.6.1 Terminology for Describing Hot Plugging

The terms shown in bold in this section are used in the discussion of hot plugging rules and procedures.

- A SCSI bus **segment** consists of two terminators, the electrical path forming continuity between them, and possibly, some attached stubs. Bus segments may be connected together by bus isolators (for example, DWZZA) to form a **logical SCSI bus** or just **SCSI bus**.
- There are two types of connections on a segment: **bussing connections**, which break the path between two terminators, and **stubbing connections**, which disconnect all or part of a stub.
- A device is **active** on the SCSI bus when it is asserting one or more of the bus signals. A device is **inactive** when it is not asserting any bus signals.

  The segment attached to a bus isolator is inactive when all devices on that segment, except possibly the bus isolator, are inactive.
- A port on a bus isolator has **proper termination** when it is attached to a segment that is terminated at both ends and has TERMPWR in compliance with SCSI–2 requirements.

7.7.6.2 Rules for Hot Plugging

The following rules must be followed when planning for and performing hot plugging:

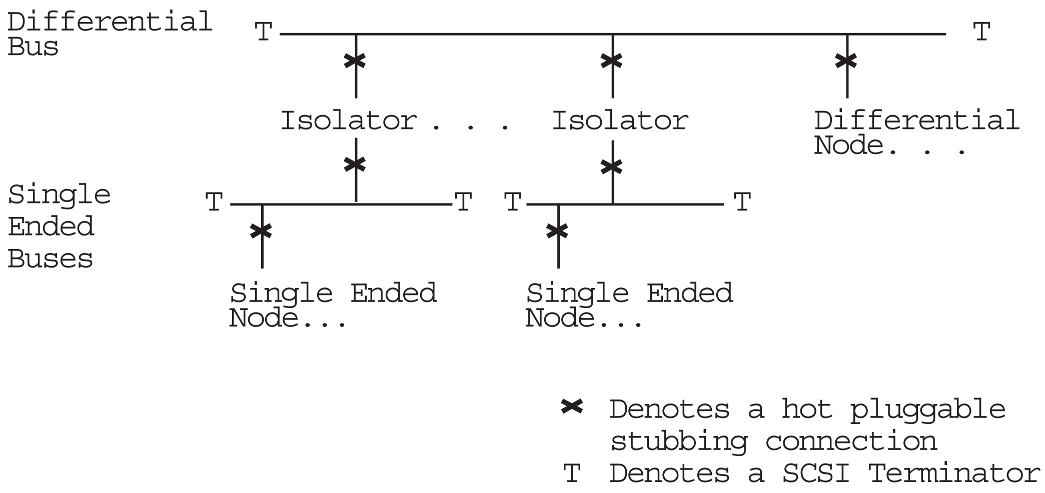
- The device to be hot plugged, and all other devices on the same segment, shall meet the electrical requirements described in Annex A, Section A.4, of the SCSI-3 Parallel Interface (SPI) Standard, working draft X3T10/855D.[1] The SPI document places requirements on the receivers and terminators on the segment where the hot plugging is being performed, and on the transceivers, TERMPWR, termination, and power/ground/signal sequencing, of the device that is being hot plugged.

  All the devices in Table 7–2 meet these requirements, except the DWZZA. The DWZZA's transceivers do not meet the requirements for glitch-free power on/off. The rules and procedures in this section have been adjusted to compensate for this fact.

- Hot plugging shall occur only at a stubbing connection.

  This implies that a hot-plugged device shall make only one connection to the SCSI bus, the device shall not provide termination for the SCSI bus, and the device's connection shall not exceed the maximum stub length, as shown in Figure 7–3. An example of a SCSI bus topology showing the valid hot plugging connections is illustrated in Figure 7–16.

Figure 7–16 SCSI Bus Topology



- Precautions shall be used to ensure that electrostatic discharge (ESD) does not damage devices or disrupt active signals on the SCSI bus. These precautions shall be taken during the process of disconnecting and connecting, as well as during the time that SCSI bus conductors are exposed.

- Precautions shall be used to ensure that ground offset voltages do not pose a safety hazard and will not interfere with SCSI bus signaling, especially in single-ended configurations. The procedures for measuring and eliminating ground offset voltages are described in Section 7.7.8.

[1] Reference to this draft standard is necessary because the SCSI–2 Standard does not adequately specify the requirements for hot plugging.

- The device that is hot plugged shall be inactive during the disconnection and connection operations. Otherwise, the SCSI bus may become hung.[1]

  Note

  Ideally, a device will also be inactive whenever its power is removed, for the same reason.

  The procedures for ensuring that a device is inactive are described in Section 7.7.6.3.

- A quorum disk shall not be hot plugged. This is because there is no mechanism for stopping the I/O to a quorum disk and because the replacement disk will not contain the correct quorum file.

  The VMScluster system must be reconfigured to remove a device as a quorum disk before that device is removed from the bus. The procedure for accomplishing this is described in *VMScluster Systems for OpenVMS* (Section 8.3.3 in the OpenVMS Version 6.1 edition).

  An alternate method for increasing the availability of the quorum disk is to use an HSZ40 mirror set as the quorum disk. This would allow a failed member to be replaced while maintaining the quorum disk functionality.

- Disks shall be logically dismounted before removing or replacing them in a hot-plug operation. This is required to ensure that the disk is inactive and to ensure the integrity of the file system.

- The DWZZA shall be powered up when it is inserted onto an active SCSI bus. The DWZZA should remain powered up at all times while it is attached to the active SCSI bus. This is because the DWZZA can disrupt the operation of the attached segments when it is powering up or down.

- The segment attached to a bus isolator shall be maintained in the inactive state whenever the other port on the bus isolator is improperly terminated. This is required because an improperly terminated bus isolator port may pass erroneous signals to the other port.

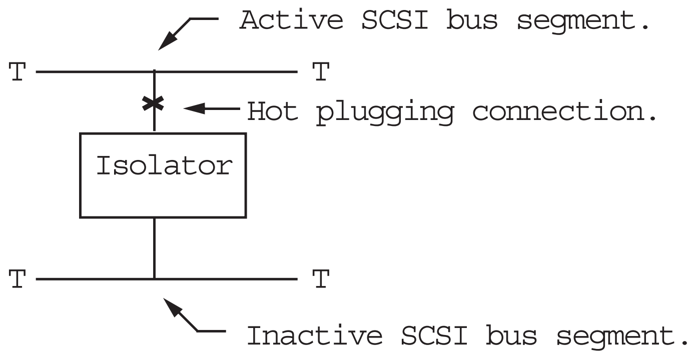
  Thus, for a particular hot-plugging operation, one of the segments attached to a bus isolator shall be designated as the (potentially) active segment, and the other shall be maintained in the inactive state, as illustrated in Figure 7–17. The procedures for ensuring that a segment is inactive are described in Section 7.7.6.3.

  Note that although a bus isolator may have more than one stubbing connection and thus be capable of hot plugging on each of them, only one segment can be the active segment for any particular hot-plugging operation.

- Precautions shall be taken to ensure that the only electrical conductor that contacts a connector pin is its mate. These precautions must be taken during the process of disconnecting and connecting as well as during the time the connector is disconnected.

[1] OpenVMS will eventually detect a hung bus and reset it, but this may be temporarily disruptive to VMScluster operations.

Figure 7–17 Hot Plugging a Bus Isolator



- Devices shall be replaced with devices of the same type (that is, if any member in the VMScluster system configures a SCSI ID as a "DK" or "MK" device, then that SCSI ID shall contain only "DK" or "MK" devices, respectively, for as long as that VMScluster member is running).

  Different implementations of the same device type may be substituted (for example, an RZ26L may be replaced with an RZ28B). Note that the system will not recognize the change in device type until an attempt is made to mount the new device. Also, note that host-based shadowing continues to require that all members of a shadow set be the same device type.

- SCSI IDs that are empty when a system boots shall remain empty as long as that system is running. This rule only applies if there are multiple processors on the SCSI bus and the MSCP server is loaded on any of them. (The MSCP server is loaded when the system parameter MSCP_LOAD equals 1).

  This is required to ensure that nodes on the SCSI bus use their direct path to the disk, rather than the served path. When the new device is configured on a system (using SYSMAN IO commands), that system serves it to the second system on the shared SCSI bus. The second system automatically configures the new device via the MSCP served path. Once this occurs, the second system will be unable to use its direct SCSI path to the new device because failover from an MSCP served path to a direct SCSI path is not implemented.

7.7.6.3 Procedures for Ensuring That a Device or Segment Is Inactive

Use the following procedures to ensure that a device or a segment is inactive:

- To ensure that a disk is inactive:
  1. Dismount the disk on all members of the VMScluster system.
  2. Ensure that any I/O that can occur to a dismounted disk is stopped, for example:
     - Disable the disk as a quorum disk.
     - Allocate the disk (using the DCL ALLOCATE command) to block further mount or initialization attempts.
     - Disable console polling by all halted hosts on the logical SCSI bus (by setting the console variable SCSI_POLL to OFF, and entering the INIT command).
     - Ensure that no host on the logical SCSI bus is executing power-up or initialization self-tests, booting, or configuring the SCSI bus (using SYSMAN IO commands).

- To ensure that an HSZ40 controller is inactive:
  1. Dismount all of the HSZ40 virtual disks on all members of the VMScluster system.
  2. Shut down the controller, following the procedures in the *HS Family of Array Controllers User's Guide*.
  3. Power down the HSZ40, if desired.
- To ensure that a host adapter is inactive:
  1. Halt the system.
  2. Power down the system or set the console variable SCSI_POLL to OFF and then enter the INIT command on the halted system. This ensures that the system will not poll or respond to polls.
- To ensure that a segment is inactive, follow the procedure described above for every device on the segment.

7.7.6.4 Procedure for Hot Plugging StorageWorks SBB Disks

To remove an SBB disk from an active SCSI bus:

1. Use an ESD grounding strap that is attached either to a grounding stud or to unpainted metal on one of the cabinets in the system. Refer to the system installation procedures for guidance.
2. Follow the procedure in Section 7.7.6.3 to make the disk inactive.
3. Squeeze the clips on the side of the SBB, and slide the disk out of the StorageWorks shelf.

To insert an SBB disk onto an active SCSI bus:

1. Use an ESD grounding strap that is attached either to a grounding stud or to unpainted metal on one of the cabinets in the system. Refer to the system installation procedures for guidance.
2. Ensure that the SCSI ID associated with the device (either by jumpers or by the slot in the StorageWorks shelf) conforms to the following:
   - The SCSI ID is unique for the logical SCSI bus
   - The SCSI ID is already configured as a "DK" device on all of the following:
     - Any member of the VMScluster system that already has that ID configured
     - Any OpenVMS processor on the same SCSI bus that is running the MSCP server
3. Slide the SBB into the StorageWorks shelf.
4. Configure the disk on VMScluster members, if required, using SYSMAN IO commands.

7.7.6.5 Procedure for Hot Plugging HSZ40s

To remove an HSZ40 controller from an active SCSI bus:

1. Use an ESD grounding strap that is attached either to a grounding stud or to unpainted metal on one of the cabinets in the system. Refer to the system installation procedures for guidance.
2. Follow the procedure in Section 7.7.6.3 to make the HSZ40 inactive.
3. The HSZ40 can be powered down, but it must remain plugged in to the power distribution system to maintain grounding.
4. Unscrew and remove the differential tri-link from the HSZ40.
5. Protect all exposed connector pins from ESD and from contacting any electrical conductor while they are disconnected.

To insert an HSZ40 controller onto an active SCSI bus:

1. Use an ESD grounding strap that is attached either to a grounding stud or to unpainted metal on one of the cabinets in the system. Refer to the system installation procedures for guidance. Also, ensure that the ground offset voltages between the HSZ40 and all components that will be attached to it are within the limits specified in Section 7.7.8.
2. Protect all exposed connector pins from ESD and from contacting any electrical conductor while they are disconnected.
3. Power up the HSZ40 and ensure that the disk units associated with the HSZ40 conform to the following:
   - The disk units are unique for the logical SCSI bus
   - The disk units are already configured as “DK” devices on the following:
     - Any member of the VMScluster system that already has that ID configured
     - Any OpenVMS processor on the same SCSI bus that is running the MSCP server
4. Ensure that the HSZ40 will make a legal stubbing connection to the active segment. (The connection is legal when the tri-connector is attached directly to the HSZ40 controller module, with no intervening cable.)
5. Attach the differential tri-link to the HSZ40, using care to ensure that it is properly aligned. Tighten the screws.
6. Configure the HSZ40 virtual disks on VMScluster members, as required, using SYSMAN IO commands.

7.7.6.6 Procedure for Hot Plugging Host Adapters

To remove a host adapter from an active SCSI bus:

1. Use an ESD grounding strap that is attached either to a grounding stud or to unpainted metal on one of the cabinets in the system. Refer to the system installation procedures for guidance.
2. Verify that the connection to be broken is a stubbing connection. If not, then hot plugging must not be performed.
3. Follow the procedure in Section 7.7.6.3 to make the host adapter inactive.

4. The system can be powered down, but it must remain plugged in to the power distribution system to maintain grounding.
5. Remove the “Y” cable from the host adapter’s single-ended connector.
6. Protect all exposed connector pins from ESD and from contacting any electrical conductor while they are disconnected.
7. Do *not* unplug the adapter from the host’s internal bus while the host remains powered up.

   At this point, the adapter has disconnected from the SCSI bus. To remove the adapter from the host, power down the host first, and then remove the adapter from the host’s internal bus.

To insert a host adapter onto an active SCSI bus:

1. Use an ESD grounding strap that is attached either to a grounding stud or to unpainted metal on one of the cabinets in the system. Refer to the system installation procedures for guidance. Also, ensure that the ground offset voltages between the host and all components that will be attached to it are within the limits specified in Section 7.7.8.
2. Protect all exposed connector pins from ESD and from contacting any electrical conductor while they are disconnected.
3. Ensure that the host adapter will make a legal stubbing connection to the active segment (the stub length must be within allowed limits, and the host adapter must not provide termination to the active segment).
4. Plug the adapter into the host (if it is unplugged).
5. Plug the system into the power distribution system to ensure proper grounding. Power up if desired.
6. Attach the “Y” cable to the host adapter, ensuring that it is properly aligned.

7.7.6.7 Procedure for Hot Plugging DWZZAs

Use the following procedure to remove a DWZZA from an active SCSI bus:

1. Use an ESD grounding strap that is attached either to a grounding stud or to unpainted metal on one of the cabinets in the system. Refer to the system installation procedures for guidance.
2. Verify that the connection to be broken is a stubbing connection. If not, then hot plugging must not be performed.
3. Do not power down the DWZZA. This can disrupt the operation of the attached SCSI bus segments.
4. Determine which SCSI bus segment will remain active after the disconnection. Follow the procedure in Section 7.7.6.3 to make the other segment inactive.

   When the DWZZA is removed from the active segment, the inactive segment must remain inactive until the DWZZA is also removed from the inactive segment, or proper termination is restored to the DWZZA port that was disconnected from the active segment.
5. The next step depends on the type of DWZZA and the segment that is being hot plugged, as follows:

DWZZA Type	Condition	Action
DWZZA-VA	Single-ended segment will remain active	Squeeze the clips on the side of the SBB, and slide the DWZZA-VA out of the StorageWorks shelf.
DWZZA-VA	Differential segment will remain active	Unscrew and remove the differential tri-link from the DWZZA-VA.
DWZZA-AA	Single-ended segment will remain active	Remove the "Y" cable from the DWZZA-AA's single-ended connector.
DWZZA-AA	Differential segment will remain active	Unscrew and remove the differential tri-link from the DWZZA-AA.

6. Protect all exposed connector pins from ESD and from contacting any electrical conductor while they are disconnected.

To insert a DWZZA onto an active SCSI bus:

1. Use an ESD grounding strap that is attached either to a grounding stud or to unpainted metal on one of the cabinets in the system. Refer to the system installation procedures for guidance. Also, ensure that the ground offset voltages between the DWZZA-AA and all components that will be attached to it are within the limits specified in Section 7.7.8.
2. Protect all exposed connector pins from ESD and from contacting any electrical conductor while they are disconnected.
3. Ensure that the DWZZA will make a legal stubbing connection to the active segment (the stub length must be within allowed limits, and the DWZZA must not provide termination to the active segment).
4. The DWZZA must be powered up. The SCSI segment that is being added must be attached and properly terminated. All devices on this segment must be inactive.
5. The next step depends on the type of DWZZA, and which segment is being hot plugged, as follows:

DWZZA Type	Condition	Action
DWZZA-VA	Single-ended segment is being hot plugged	Slide the DWZZA-VA into the StorageWorks shelf.
DWZZA-VA	Differential segment is being hot plugged	Attach the differential tri-link to the DWZZA-VA, using care to ensure that it is properly aligned. Tighten the screws.
DWZZA-AA	Single-ended segment is being hot plugged	Attach the "Y" cable to the DWZZA-AA, using care to ensure that it is properly aligned.
DWZZA-AA	Differential segment is being hot plugged	Attach the differential tri-link to the DWZZA-VA, using care to ensure that it is properly aligned. Tighten the screws.

6. If the newly attached segment has storage devices on it, then configure them on VMScluster members, if required, using SYSMAN IO commands.

7.7.7 OpenVMS Requirements for Devices Used on Multiple-Host SCSI VMScluster Systems

At this time, the only devices approved for use on multiple-host SCSI VMScluster systems are those listed in Table 7–2. While not specifically approved for use, other disk devices might be used in a multiple-host VMScluster system when they conform to the following requirements:

- Support for concurrent multi-initiator I/O.
- Proper management for the following states or conditions on a per-initiator basis:
  - Synchronous negotiated state and speed
  - Width negotiated state
  - Contingent Allegiance and Unit Attention conditions
- Tagged Command Queueing. This is needed to provide an ordering guarantee used in VMScluster systems to ensure that I/O has been flushed. The drive must implement queuing that complies with Section 7.8.2 of the SCSI–2 Standard, which says (in part):

  > “...All commands received with a simple queue tag message prior to a command received with an ordered queue tag message, *regardless of initiator*, shall be executed before that command with the ordered queue tag message.” (Emphasis added.)

- Support for command disconnect.
- A reselection timeout procedure compliant with Option b of Section 6.1.4.2 of the SCSI–2 Standard. Furthermore, the device shall implement a reselection retry algorithm that limits the amount of bus-time spent attempting to reselect a nonresponsive initiator.
- Automatic read reallocation enabled (ARRE) and automatic write reallocation enabled (AWRE), (that is, drive-based bad block revectoring), to prevent multiple hosts from unnecessarily revectoring the same block. To avoid data corruption, it is essential that the drive comply with Section 9.3.3.6 of the SCSI–2 Standard, which says (in part):

  > “...The automatic reallocation shall then be performed only if the target *successfully recovers the data*.” (Emphasis added.)

- Storage devices should not supply TERMPWR. If they do, then it is necessary to apply configuration rules to ensure that there are no more than four sources of TERMPWR on a segment.

Finally, if the device or any other device on the same segment will be hot plugged, then the device must meet the electrical requirements described in Section 7.7.6.2.

7.7.8 Grounding Requirements

This section describes the grounding requirements for electrical systems in a SCSI VMScluster system.

Improper grounding can result in voltage differentials, called ground offset voltages, between the enclosures in the configuration. Even small ground offset voltages across the SCSI interconnect (as shown in Step 3 in Table 7–9) can disrupt the configuration, and the user may experience performance degradation or data corruption.

Table 7–9 describes important considerations to ensure proper grounding.

Table 7–9 Steps for Ensuring Proper Grounding

Description

1 Ensure that site power distribution meets all local electrical codes.

2 Inspect the entire site power distribution system to ensure that:

- All outlets have power ground connections
- A grounding prong is present on all computer equipment power cables
- Power outlet neutral connections are not actual ground connections
- All grounds for the power outlets are connected to the same power distribution panel
- All devices that are connected to the same circuit breaker as the computer equipment are UL or IEC approved

3 If you have difficulty verifying these conditions, you can use a hand-held multimeter to measure the ground offset voltage between any two cabinets. To measure the voltage, connect the multimeter leads to unpainted metal on each enclosure. Then, determine whether the voltage exceeds the following allowable ground offset limits:

SCSI Signaling Method	Maximum Allowable Offset
Single-ended	50 millivolts
Differential	800 millivolts

The multimeter method provides data for only the moment it is measured. The ground offset values may change over time as additional devices are activated or plugged into the same power source. To ensure that the ground offsets remain within acceptable limits over time, Digital recommends that you have a power survey performed by a qualified electrician.

4 If you are uncertain about the grounding situation or if the measured offset exceeds the allowed limit, Digital recommends that a qualified electrician correct the problem. It may be necessary to install grounding cables between enclosures to reduce the measured offset.

5 If an unacceptable offset voltage was measured and a ground cable was installed, then measure the voltage again to ensure it is less than the allowed limits. If not, an electrician must determine the source of the ground offset voltage and reduce or eliminate it.

8

VMScluster Systems That Span Multiple Sites

This chapter discusses multiple-site VMScluster configurations, with an emphasis on the new wide area network ATM and DS3 communications services. It provides configuration guidelines and system management suggestions for VMScluster systems in which multiple nodes are located at sites separated by relatively long distances.

The information in this chapter supersedes the *Multiple-Site VMScluster Systems Addendum* manual, and it supplements multiple-site VMScluster information in the following VMScluster manuals:

- *VMScluster Systems for OpenVMS*
- *Guidelines for VMScluster Configurations*

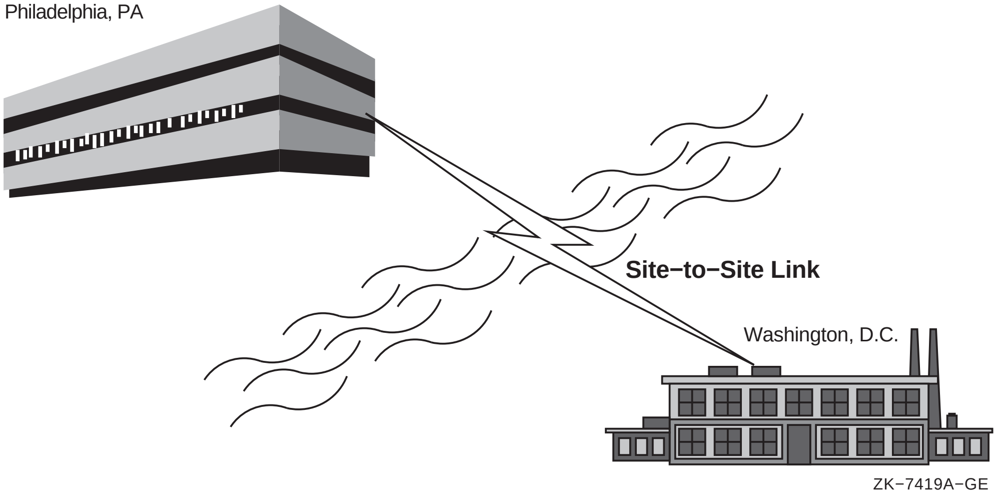
The sections that follow describe multiple-site VMScluster configurations and some of the benefits you can derive from them.

8.1 What Is a Multiple-Site VMScluster System?

A **multiple-site VMScluster system** is a VMScluster system in which the member nodes are located in geographically separate sites. When an organization has geographically disperse sites, a multiple-site VMScluster system allows the organization to realize the benefits of VMScluster systems (for example, sharing data among sites while managing data center operations at a single, centralized location).

Figure 8–1 illustrates the concept of a multiple-site VMScluster system for a company with a manufacturing site located in Washington, D.C. and corporate headquarters in Philadelphia. This configuration spans a geographical distance of approximately 130 miles (210 km).

Figure 8–1 Site-to-Site Link Between Philadelphia and Washington



The Fiber Distributed Data Interface (FDDI) has been in general use since VMS Version 5.4–3 to carry out cluster communications over distances of approximately 25 miles (approximately 40 km).[1]

8.1.1 ATM, DS3, and FDDI Intersite Links

The following link technologies between sites are approved for OpenVMS VAX and OpenVMS AXP systems:

- Asynchronous Transfer Mode (ATM)
- DS3
- FDDI

High-performance local area network (LAN) technology combined with the ATM, DS3, and FDDI interconnects allows you to utilize wide area network (WAN) communication services in your VMScluster configuration. VMScluster systems configured with the GIGAswitch crossbar switch and ATM, DS3, or FDDI interconnects approve the use of nodes located miles apart.[2] Section 8.3 describes VMScluster systems and the WAN communications services in more detail.

Note

To gain the benefits of disaster tolerance across a multiple-site VMScluster, use the Business Recovery Server combined with Volume Shadowing for OpenVMS.

Consult your Digital Services Group or see the Software Product Descriptions (SPDs) for complete and up-to-date details about these products.

[1] The cable route distance between sites.

[2] The actual distance between any two sites is determined by the physical intersite cable-route distance, and not the straight-line distance between the sites.

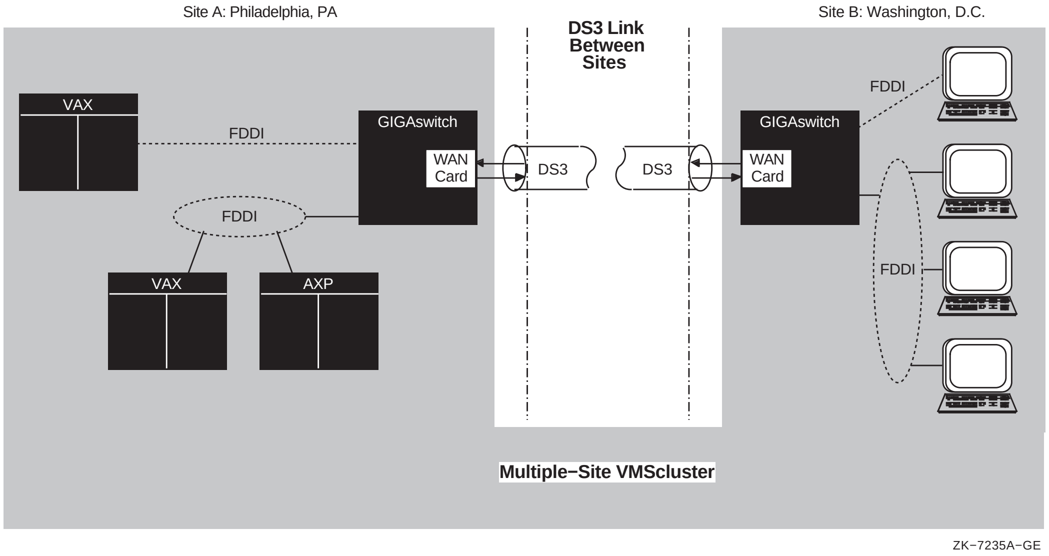
8.1.2 Benefits of Multiple-Site VMScluster Systems

The benefits you can realize with a multiple-site VMScluster system include:

Benefit	Description
Remote satellites and nodes	A few systems can be remotely located at a secondary site and can benefit from centralized system management and other resources at the primary site, as shown in Figure 8–2. For example, a main office data center could be linked to a warehouse or a small manufacturing site that has a few local nodes with directly attached site-specific devices. Alternatively, some engineering workstations could be installed in an office park across the city from the primary business site.
Data center management consolidation	A single management team can manage nodes located in data centers at multiple sites.
Physical resource sharing	Multiple sites can readily share devices such as high-capacity computers, tape libraries, disk archives, or phototypesetters.
Remote archiving	Backups can be made to archival media at any site in the cluster. A common example would be to use disk or tape at a single site to back up the data for all sites in the multiple-site VMScluster. Backups of data from remote sites can be made transparently (that is, without any intervention required at the remote site).
Increased availability	In general, a multiple-site VMScluster system provides all of the availability advantages of a LAN VMScluster. (See *VMScluster Systems for OpenVMS* for information about LANs.) Additionally, by connecting multiple, geographically separate sites, multiple-site VMScluster configurations can increase the availability of a system or elements of a system in a variety of ways: • Logical volume/data availability—Volume shadowing or redundant arrays of independent disks (RAID) can be used to create logical volumes with members at both sites. If one of the sites becomes unavailable, data can remain available at the other site. • Site failover—By adjusting the VOTES system parameter, you can select a preferred site to continue automatically if the other site fails or if communications with the other site are lost. • Disaster tolerance—When combined with the software, services, and management procedures provided by the Business Recovery Server and Volume Shadowing for OpenVMS products, you can achieve a high level of disaster tolerance. The Software Product Descriptions (SPDs) for these products provide further information.

Figure 8–2 shows a VMScluster system with satellites accessible from a remote site.

Figure 8–2 Multiple-Site VMScluster Configuration with Remote Satellites



8.1.3 General Configuration Guidelines

The same configuration rules that apply to VMScluster systems on a LAN also apply to a multiple-site VMScluster configuration that includes ATM, DS3, or FDDI intersite interconnects. General LAN configuration rules are stated in the following documents:

- VAXcluster Software for OpenVMS VAX *Software Product Description* (SPD 29.78.*xx*)
- VMScluster Software for OpenVMS AXP *Software Product Description* (SPD 42.18.*xx*)
- *Guidelines for VMScluster Configurations*

Some configuration guidelines are unique to multi-site VMSclusters, and these guidelines are described in Section 8.3.4.

8.2 Using FDDI to Configure Multiple-Site VMScluster Systems

Since VMS Version 5.4–3, FDDI has been the most commonly used means to connect two distant VMScluster sites. Using high-speed FDDI fiber-optic cables, you can connect sites with an intersite cable-route distance of up to 25 miles (40 km).

You can connect sites using these FDDI methods:

- To obtain maximum performance, use a full-duplex FDDI link at 100 Mb/s both ways between GIGAswitch/FDDI bridges at each site for maximum intersite bandwidth.
- To obtain maximum availability, use a dual FDDI ring at 100 Mb/s between dual attachment station (DAS) ports of wiring concentrators or GIGAswitch /FDDI bridges.

- For maximum performance and availability, use two disjoint FDDI LANs, each with dedicated host adapters and full-duplex FDDI intersite links connected to GIGAswitch/FDDI bridges at each site.

Refer to the *GIGAswitch/FDDI ATM Linecard Reference Manual* for configuration information. Additional VMScluster configuration guidelines and system management information can be found in *Guidelines for VMScluster Configurations* and *VMScluster Systems for OpenVMS*. See the *Overview of OpenVMS Documentation* for information about ordering the current version of these manuals.

The inherent flexibility of VMScluster systems and improved VMScluster LAN protocols also allow you to connect multiple VMScluster sites using the ATM and/or DS3 communications services.

8.3 Using WAN Services to Configure Multiple-Site VMScluster Systems

This section provides an overview of the ATM and DS3 wide area network (WAN) services, describes how you can bridge an FDDI interconnect to the ATM and/or DS3 communications services, and provides guidelines for using these services to configure multiple-site VMScluster systems.

The ATM and DS3 services provide long-distance, point-to-point communications that you can configure into your VMScluster system to gain WAN connectivity. The ATM and DS3 services are available from most common telephone service carriers and other sources.

Note

DS3 is not available in Europe and some other locations. Also, ATM is a new and evolving standard, and ATM services might not be available in all localities.

ATM and DS3 services are approved for use with the following OpenVMS versions:

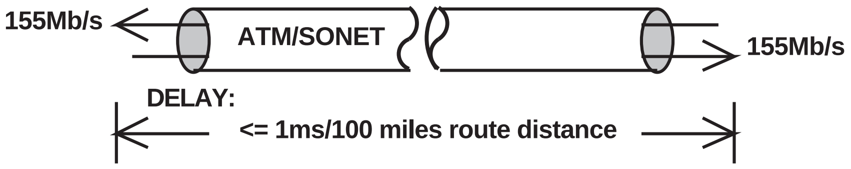
Service	Approved Versions of OpenVMS
ATM	OpenVMS Version 6.2 or later
DS3	OpenVMS Version 6.1 or later

The following sections describe the ATM and DS3 communications services and how to configure these services in multiple-site VMScluster systems.

8.3.1 The ATM Communications Service

The ATM communications service that uses the SONET physical layer (ATM/SONET) provides full-duplex communications (that is, the bit rate is available simultaneously in both directions as shown in Figure 8–3). ATM /SONET is compatible with multiple standard bit rates. The SONET OC-3 service at 155 Mb/s full-duplex rate is the best match to FDDI's 100 Mb/s bit rate. ATM/SONET OC-3 is a standard service available in most parts of the world. In Europe, ATM/SONET is a high performance alternative to the older E3 standard.

Figure 8–3 ATM/SONET OC-3 Service



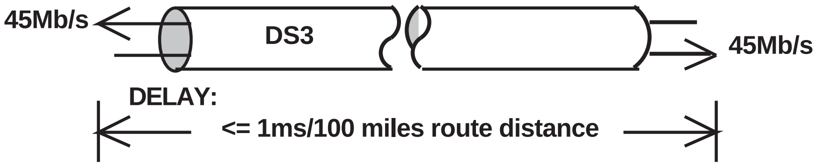
ZK−7614A−GE

To transmit data, ATM frames (packets) are broken into **cells** for transmission by the ATM service. Each cell has 53 bytes, of which 5 bytes are reserved for header information and 48 bytes are available for data. At the destination of the transmission, the cells are reassembled into ATM frames. The use of cells permits ATM suppliers to multiplex and demultiplex multiple data streams efficiently at differing bit rates. This conversion of frames into cells and vice versa is transparent to higher layers.

8.3.2 The DS3 Communications Service

The DS3 communications service provides full-duplex communications as shown in Figure 8–4. DS3 (also known as T3) provides the T3 standard bit rate of 45 Mb/s. T3 is the standard service available in North America and many other parts of the world.

Figure 8–4 DS3 Service

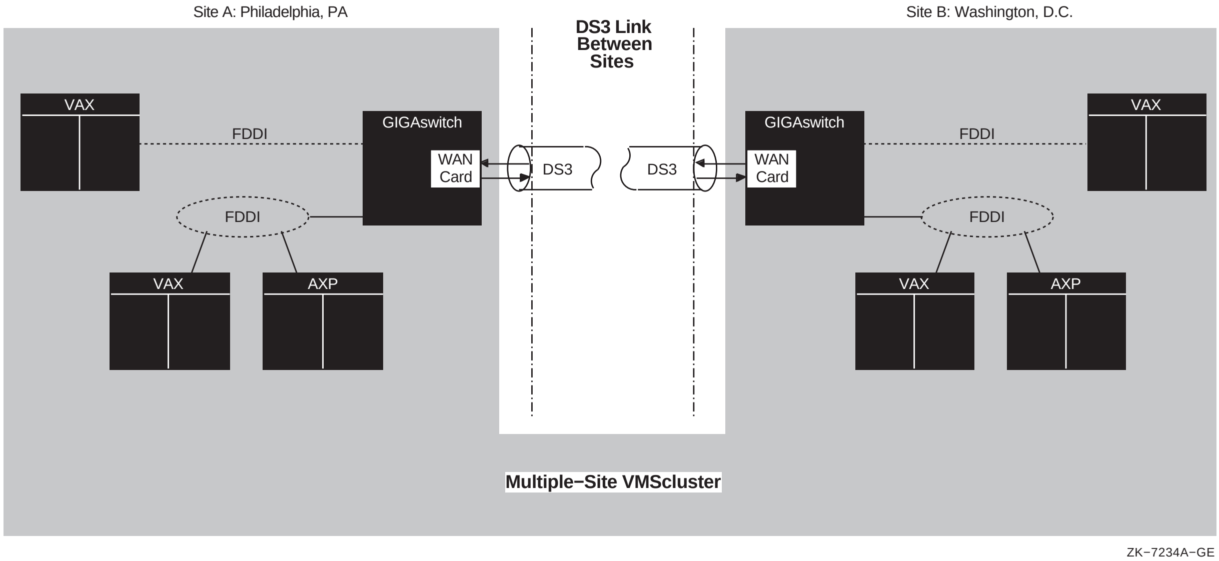


ZK−7615A−GE

8.3.3 FDDI-to-WAN Bridges

You can use FDDI-to-WAN (for example, FDDI-to-ATM and/or FDDI-to-DS3) bridges to configure a VMScluster with nodes in geographically separate sites, such as the one shown in Figure 8–5. In this figure, the VMScluster nodes at each site communicate as though the two sites are connected by FDDI. The FDDI-to-WAN bridges make the existence of ATM and DS3 transparent to the VMScluster software.

Figure 8–5 Multiple-Site VMScluster Configuration Connected by DS3



In Figure 8–5, the FDDI-to-DS3 bridges and DS3 operate as follows:

1. The local FDDI-to-DS3 bridge receives FDDI packets addressed to nodes at the other site
2. The bridge converts the FDDI packets into DS3 packets and sends the packets to the other site via the DS3 link
3. The receiving FDDI-to-DS3 bridge converts the DS3 packets into FDDI packets and transmits them on an FDDI ring at that site

Digital recommends using the GIGAswitch/FDDI system to construct FDDI-to-WAN bridges. Digital used the GIGAswitch/FDDI, combined with the DEFGT WAN T3/SONET option card, during qualification testing of the ATM and DS3 communications services in multiple-site VMScluster systems.

8.3.4 Guidelines for Configuring ATM and DS3 in a VMScluster System

When configuring a multiple-site VMScluster configuration, you must ensure that the intersite link's delay, bandwidth, availability, and bit error rate characteristics meet application needs. This section describes the requirements and provides recommendations for meeting those requirements.

8.3.4.1 Requirements

To be a configuration approved by Digital, a multiple-site VMScluster must comply with the following rules:

Maximum intersite link route distance

The total intersite link cable route distance between members of a multiple-site VMScluster cannot exceed 150 miles (242 km). You can obtain exact distance measurements from your ATM or DS3 supplier.

This distance restriction may be exceeded by Business Recovery Server configurations that meet Business Recovery Server configuration rules.

Maximum intersite link utilization	Average intersite link utilization in either direction must be less than 80% of the link’s bandwidth in that direction for any 10-second interval. Exceeding this utilization is likely to result in intolerable queuing delays or packet loss.
Intersite link specifications	The intersite link must meet the VMScluster requirements specified in Table 8–3.
VMScluster LAN configuration rules	Apply the configuration rules for VMScluster systems on a LAN to a configuration. Documents describing configuration rules are referenced in Section 8.1.3.

8.3.4.2 Recommendations

When configuring the DS3 interconnect, apply the configuration guidelines for VMScluster systems interconnected by LANs that are stated in the cluster Software Product Descriptions (SPDs) and in the *Guidelines for VMScluster Configurations* manual. VMScluster members at each site can include any mix of satellites, systems, and other interconnects such as CI and DSSI.

This section provides additional recommendations for configuring a multiple-site VMScluster system.

DS3 link capacity/protocols

The GIGAswitch with the WAN T3/SONET option card provides a full-duplex 155 Mb/s ATM/SONET link. The entire bandwidth of the link is dedicated to the WAN option card. However, The GIGAswitch/FDDI’s internal design is based upon full-duplex extensions to FDDI. Thus, the GIGAswitch/FDDI’s design limits the ATM/SONET link’s capacity to 100 Mb/s in each direction.

The GIGAswitch with the WAN T3/SONET option card provides several protocol options that can be used over a DS3 link. Use the DS3 link in clear channel mode, which dedicates its entire bandwidth to the WAN option card. The DS3 link capacity varies with the protocol option selected. Protocol options are described in Table 8–1:

Table 8–1 DS3 Protocol Options

Protocol Option	Link Capacity
ATM[1] AAL–5[2] mode with PLCP[3] disabled.	39 Mb/s
ATM AAL–5 mode with PLCP enabled.	33 Mb/s
HDLC[4] mode (not currently available).	43 Mb/s

[1] Asynchronous Transfer Mode

[2] ATM Adaptation Layer

[3] Physical Layer Convergence Protocol

[4] High-Speed Datalink Control

For maximum link capacity, Digital recommends configuring the WAN T3/SONET option card to use ATM AAL–5 mode with PLCP disabled.

Intersite bandwidth

The intersite bandwidth can limit application locking and I/O performance (including volume shadowing or RAID set copy times) and the performance of the lock manager.

To promote reasonable response time, Digital recommends that average traffic in either direction over an intersite link not exceed 60% of the link's bandwidth in that direction for any 10-second interval. Otherwise, queuing delays within the FDDI-to-WAN bridges can adversely affect application performance.

Remember to account for both VMScluster communications (such as locking and I/O) and network communications (such as TCP/IP, LAT, and DECnet) when calculating link utilization.

Intersite delay

An intersite link introduces a one-way delay of up to 1 ms per 100 miles of intersite cable route distance plus the delays through the FDDI-to-WAN bridges at each end. Digital recommends that you consider the effects of intersite delays on application response time and throughput.

For example, intersite link one-way path delays have the following components:

- Cable route one-way delays of 1 ms/100 miles (0.01 ms/mile) for both ATM and DS3
- FDDI-to-WAN bridge delays (approximately 0.5 ms per bridge, and 2 bridges per one-way trip)

Calculate the delays for a round trip as follows:

$$WAN\ ROUND\ TRIP\ DELAY =$$

$$2 \times (\ N\ miles\ \times\ 0.01\ ms\ per\ mile\ +\ 2\ \times\ 0.5\ ms\ per\ FDDI-WAN\ bridge\ )$$

An I/O write operation that is MSCP served requires a minimum of two round-trip packet exchanges:

$$WAN\ I/O\ Write\ Delay = 2 \times WAN\ Round\ Trip\ Delay$$

Thus, an I/O write over a 100-mile WAN link takes at least 8 ms longer than the same I/O write over a short, local FDDI.

Similarly, a lock operation typically requires a round trip exchange of packets:

$$WAN\ Lock\ Operation\ Delay = WAN\ Round\ Trip\ Delay$$

An I/O operation with N locks to synchronize it incurs the following delay due to WAN:

$$WAN\ Locked\ IO\ Operation\ Delay =$$

$$(N\ \times\ WAN\ Lock\ Operation\ Delay)\ +\ WAN\ I/O\ Delay$$

Bit error ratio

The bit error ratio (BER) parameter is an important measure of the frequency that bit errors are likely to occur on the intersite link. You should consider the effects of bit errors on application throughput and responsiveness when configuring a multiple-site VMScluster. Intersite link bit errors can result in packets being lost and retransmitted with consequent delays in application I/O response time (see Section 8.3.6). You can expect application delays ranging from a few hundred milliseconds to a few seconds each time a bit error causes a packet to be lost.

Intersite link availability

Interruptions of intersite link service can result in the resources at one or more sites becoming unavailable until connectivity is restored (see Section 8.3.5).

System disks

Sites with nodes contributing quorum votes should have a local system disk or disks for those nodes.

System management

A large, multiple-site VMScluster requires a system management staff trained to support an environment that consists of a large number of diverse systems that are used by many people performing varied tasks.

Microwave DS3 links

You can provide portions of a DS3 link with microwave radio equipment. The specifications in Section 8.3.6 apply to any DS3 link. The BER and availability of microwave radio portions of a DS3 link are affected by local weather and the length of the microwave portion of the link. Consider working with a microwave consultant who is familiar with your local environment if you plan to use microwaves as portions of a DS3 link.

8.3.5 Availability Considerations

If the FDDI-to-WAN bridges and the link that connects multiple sites become temporarily unavailable, the following events could occur:

- Intersite link failures can result in the resources at one or more sites becoming unavailable until intersite connectivity is restored.
- Intersite link bit errors (and ATM cell losses) and unavailability can affect:
  - System responsiveness
  - System throughput (or bandwidth)
  - Virtual circuit (VC) closure rate
  - VMScluster transition and site failover time

Many communication service carriers offer availability-enhancing options, including path diversity and protective switching, that can significantly increase the intersite link's availability.

8.3.6 Specifications

This section describes the requirements for successful communications and performance with the WAN communications services.

To assist you in communicating your requirements to a WAN service supplier, this section uses WAN specification terminology and definitions commonly used by telecommunications service providers. These requirements and goals are derived from a combination of Bellcore Communications Research specifications and a Digital analysis of error effects on VMSclusters.

Table 8–2 describes terminology that will help you understand the Bellcore and VMScluster requirements and goals used in Table 8–3.

Use the Bellcore and VMScluster requirements for ATM/SONET - OC3 and DS3 service error performance (quality) specified in Table 8–3 to help you assess the impact of the service supplier's service quality, availability, down time, and service-interruption frequency goals on the system.

Note

To ensure that the VMScluster system meets your application response-time requirements, you might need to establish WAN requirements that exceed the Bellcore and VMScluster requirements and goals stated in Table 8–3.

Table 8–2 Bellcore and VMScluster Requirements and Goals Terminology

<table>
<tr><th>Specification</th><th>Requirements</th><th>Goals</th></tr>
<tr><td>Bellcore Communications Research</td><td>Bellcore specifications are the recommended “generic error performance requirements and objectives” documented in the Bellcore Technical Reference TR–TSY–000499 <i>TSGR: Common Requirements</i>. These specifications are adopted by WAN suppliers as their service guarantees. The FCC has also adopted them for tariffed services between common carriers. However, some suppliers will contract to provide higher service-quality guarantees at customer request.
Other countries have equivalents to the Bellcore specifications and parameters.</td><td>These are the recommended minimum values. Bellcore calls these goals their “objectives” in the <i>TSGR: Common Requirements</i> document.</td></tr>
<tr><td>VMScluster</td><td>In order for Digital to approve a configuration, parameters must meet or exceed the values shown in the VMScluster Requirement column in Table 8–3.
<table><tr><th>IF...</th><th>THEN...</th></tr><tr><td>These values are not met</td><td>VMScluster performance will probably be unsatisfactory because of interconnect errors/error recovery delays, and VC closures that may produce VMScluster state transitions and/or site failover.</td></tr><tr><td>These values are met or exceeded</td><td>Interconnect bit error–related recovery delays will not significantly degrade average VMScluster throughput. VMScluster response time should be generally satisfactory.
Note that if the requirements are only being met, there may be several application pauses <b>per hour</b>. [1]</td></tr></table></td><td>For optimal VMScluster operation, all parameters should meet or exceed the VMScluster Goal values in Table 8–3.
Note that if these values are met or exceeded, then interconnect bit errors and bit error recovery delays should not significantly degrade average VMScluster throughput.
VMScluster response time should be generally satisfactory, although there may be brief application pauses a few times <b>per day</b>.[2]</td></tr>
</table>

[1] Pauses are due to a virtual circuit retransmit timeout resulting from a lost packet on one or more NISCA transport virtual circuits. Each pause might last from a few hundred milliseconds to a few seconds.

[2] Application pauses may occur every hour or so (similar to what is described under VMScluster Requirement in Table 8–3) because of packet loss caused by bit error.

Table 8–3 VMScluster DS3 and SONET OC3 Error Performance Requirements

Parameter	Bellcore Requirement	Bellcore Goal	VMScluster Requirement[1]	VMScluster Goal[1]	Units
Errored seconds (% ES)	<1.0%	<0.4%	<1.0%	<0.028%	% ES/24 hr
	The ES parameter can also be expressed as a count of errored seconds, as follows:				
	<864	<345	<864	<24	ES per 24-hr period
Burst errored seconds (BES)[2]	≤4	–	≤4	Bellcore Goal	BES/day
Bit error ratio (BER)[3]	1×10^{-9}	2×10^{-10}	1×10^{-9}	6×10^{-12}	Errored bits/bit
DS3 channel unavailability	None	≤97 @ 250 miles, linearly decreasing to 24 @ ≤25 miles	None	Bellcore Goal	Min/yr
SONET channel unavailability	None	≤105 @ 250 miles, linearly decreasing to 21 @ ≤50 miles	None	Bellcore Goal	Min/yr
Channel unavailable event[4]	None	None	None	1 to 2	Events/year

[1] Application requirements might need to be more rigorous than those shown in the VMScluster Requirements column.

[2] Averaged over many days.

[3] Does not include any burst errored seconds occurring in the measurement period.

[4] The average number of channel down-time periods occurring during a year. This parameter is useful for specifying how often a channel might become unavailable.

Table Key

- Availability—The long-term fraction or percentage of time that a transmission channel performs as intended. Availability is frequently expressed in terms of unavailability or down time.
- BER (bit error ratio)—"The BER is the ratio of the number of bits in error to the total number of bits transmitted during a measurement period, excluding all burst errored seconds (defined below) in the measurement period. During a burst errored second, neither the number of bit errors nor the number of bits is counted."
- BES (burst errored second)—"A burst errored second is any errored second containing at least 100 errors."
- Channel—The term for a link that is used in the Bellcore *TSGR: Common Requirements* document for a SONET or DS3 link.
- Down time—The long-term average amount of time (for example, minutes) that a transmission channel is not available during a specified period of time (for example, 1 year).

  "...unavailability or downtime of a channel begins when the first of 10 [or more] consecutive Severely Errored Seconds (SESs) occurs, and ends when the first of 10 consecutive non-SESs occurs."

  The unavailable time is counted from the first SES in the 10–SES sequence.

  "The time for the end of unavailable time is counted from the first fault-free second in the [non-SES] sequence."
- ES (errored second)—"An errored second is any one-second interval containing at least one error."
- SES (severely errored second)—"...an SES is a second in which the BER is greater than 10^{-3}."

8.4 Managing VMScluster Systems Across Multiple Sites

In general, you manage a multiple-site VMScluster using the same tools and techniques that you would use for any VMScluster interconnected by a LAN. The following sections describe additional considerations and recommend system management tools and techniques.

The following table lists system management considerations specific to multiple-site VMScluster systems:

Problem	Possible Solution
Multiple-site configurations present an increased probability of the following failure modes: • VMScluster quorum loss resulting from site-to-site communication link failure • Site loss resulting from power failure or other breakdown can affect all systems at that site	Assign votes so that one preferred site has sufficient votes to maintain quorum and to continue operation if the site-to-site communication link fails or if the other site is unavailable. Select the site with the most critical applications as the primary site. Sites with a few noncritical systems or satellites probably should not have sufficient votes to continue.
Users expect that the local resources will either continue to be available or will rapidly become available after such a failure. This might not always be the case.	Consider some of the following options for setting user expectations: • Set management and user expectations regarding the likely effects of failures, and consider training remote users in the procedures to be followed at a remote site when the system becomes unresponsive because of quorum loss or other problems. • Develop management policies and procedures for what actions will be taken to identify and handle these failure modes. These procedures may include manually adjusting quorum to allow a site to continue.

8.4.1 Methods and Tools

You can use the following system management methods and tools to manage both remote and local nodes:

- There are two options for remote-site console access when you use an intersite link through a DECserver in reverse LAT mode.
  - Use the following tools to connect remote consoles:
    - SET HOST/LAT command
    - POLYCENTER Console Manager
    - VMScluster Console System (VCS)
    - Business Recovery Server Operations Management Station (includes VCS)
  - Use a modem to dial up the remote system consoles.
- An alternative to remote-site console access is to have a system manager at each site.
- To enable device and processor control commands to take effect across all nodes in a VMScluster system, use the System Management utility (SYSMAN) that is supplied with the OpenVMS operating system.

8.4.2 Shadowing Data

Volume Shadowing for OpenVMS allows you to shadow data volumes across multiple sites. System disks can be members of a volume shadowing or RAID set within a site; however, use caution when configuring system disk shadow set members in multiple sites. This is because it may be necessary to boot from a remote system disk shadow set member after a failure. If your system does not support FDDI booting, it will not be possible to do this.

See the Software Product Descriptions (SPDs) for complete and up-to-date details about Volume Shadowing for OpenVMS and StorageWorks RAID for OpenVMS.

8.4.3 Monitoring Performance

Monitor performance for multiple-site VMScluster systems as follows:

- Monitor the virtual circuit (VC) packet-loss count and round-trip time values using the System Dump Analyzer (SDA). The procedures for doing this are documented in *VMScluster Systems for OpenVMS*.
- Monitor the intersite link bit error ratio (BER) and packet loss using network management tools. You can use tools such as POLYCENTER NetView or DECmcc to access the GIGAswitch and WAN T3/SONET option card's management information and to set alarm thresholds. See the GIGAswitch, WAN T3/SONET card, POLYCENTER, and DECmcc documentation, as appropriate.

A
New OpenVMS System Messages

This release includes new or changed messages for the following OpenVMS facilities:

- ANALDISK, Analyze/Disk_Structure Utility
- BACKUP, Backup Utility
- DUMP, DUMP Command
- IMGACT, Image Activator
- LAT, LAT Facility
- LIB, Library Facility
- LICENSE, License Management Utility
- LOADER, Executive Image Loader
- MONITOR, Monitor Utility
- MOUNT, Mount Utility
- RMS, OpenVMS Record Management Services
- SECSRV, Security Server
- SET, SET Command and SET Utility
- STACONFIG, Standalone Configure Process
- SYSBOOT, System Bootstrap Facility
- SYSMAN, System Management Utility
- SYSTEM, System Services

This appendix alphabetically lists and describes messages that have been added or changed for this release. You can access online descriptions of these and all other OpenVMS system messages by using the online Help Message utility. For information about the HELP/MESSAGE command and qualifiers, see DCL help (type HELP HELP/MESSAGE at the DCL prompt) or refer to *OpenVMS System Messages: Companion Guide for Help Message Users*.

ACA_ACTIVE, SCSI ACA operations are active

Facility: SYSTEM, System Services

Explanation: You attempted to execute a SCSI I/O operation while the SCSI device was busy performing a recovery operation.

User Action: Retry the operation.

AECREATED, created alias file entry ′dev:[directory]entry.ALIAS′

Facility: BACKUP, Backup Utility

Explanation: The specified alias file entry was created during a backup restore operation.

User Action: None.

ALFNOMATCH, no records matched search criteria

Facility: SMI, System Management Integrator/Server

Explanation: No records in the ALF database match the search criteria of the command.

User Action: None.

ALFWILCRDREQ, more than one record might match - Wildcard or unit number of device required

Facility: SYSMAN, System Management Utility

Explanation: While removing records from SYSALF.DAT, the SYSMAN ALF component has encountered more than one record that matches the input from a terminal device.

User Action: Supply a wildcard or unit number with the device input.

ALIASQUAL, saveset was created /NOALIAS, restore /ALIAS qualifier will be ignored

Facility: BACKUP, Backup Utility

Explanation: The backup restore operation could not be performed as specified because the save set was created ignoring alias entries. Therefore, there are no separate files in the save set to restore in place of the alias directory entries. The restore operation was performed by processing the alias file entries as directory entries instead of as separate file entries.

User Action: Examine the save set used in the backup restore operation to determine if it is the correct save set. If not, restore the correct image and incremental save sets in the recommended order. If the save set is the correct one, no additional action is required.

ALLODIFF, The ALLOCLASS parameter value for the processor on
SCSI bus ′device-name′, ID ′slot-number′, is different from the value on this system.
This condition creates multiple names for the same device, which can lead to data corruption.

Facility: STACONFIG, Standalone Configure Process

Explanation: The value of the SYSGEN disk allocation class parameter differs between the local node and the specified remote node.

User Action: Set the ALLOCLASS parameter to the same nonzero value on both nodes.

ALLOZERO, This system and/or the processor on SCSI bus ′device-name′, ID ′slot-number′, has a zero ALLOCLASS value. This condition creates multiple names for the same device, which can lead to data corruption.

Facility: STACONFIG, Standalone Configure Process

Explanation: The local node, remote node, or both nodes have the SYSGEN disk allocation class parameter set to zero.

User Action: Set the ALLOCLASS parameter to the same nonzero value on both nodes.

ARESTERR, error restoring alias file entry ′dev:[directory]entry.alias′, the primary file entry was ′dev:[prim_dir]primary.file′

Facility: BACKUP, Backup Utility

Explanation: An error occurred when the Backup utility tried to restore an alias file entry. The alias file entry was not restored. Note that in most cases the alias file entry is eliminated from the directory.

User Action: Examine the primary file, the directory, and the alias entry directory to determine the cause of the error. Then, based on the data in this error message and any secondary error status, correct the problem and create the alias file entry using the DCL command SET FILE/ENTER.

As a general practice, Digital recommends that you execute the DCL command ANALYZE/DISK after Backup restore operations of all save sets have been completed and any subsequent error corrections have been made, for example, using SET FILE/ENTER commands.

ASSIGNFAILED, security server failed to assign a channel to a client reply mailbox

Facility: SECSRV, Security Server

Explanation: The security server's call to the $ASSIGN system service failed to assign a channel to the requestor's mailbox.

User Action: Contact your system manager.

AUDITFAILED, security server failed to audit an event because of the following error:

Facility: SECSRV, Security Server

Explanation: The security server could not perform an audit because of the error reported in the accompanying message.

User Action: Take action based on the accompanying message.

BAD_DIREFBLK, EFBLK indicates zero length directory file

Facility: ANALDISK, Analyze/Disk_Structure Utility

Explanation: The Analyze/Disk_Structure utility detected a directory file of zero length.

User Action: Do a SET FILE/NODIRECTORY on the bad directory file; then delete the directory file. Last, run ANALYZE/DISK/REPAIR to rename all the files from the bad directory into the SYSLOST.DIR;1 directory.

BAD_GSD, an inconsistency was detected while traversing the GST

Facility: LOADER, Executive Image Loader

Explanation: An executive image was loaded containing a global symbol that is not vectored through either the SYS$BASE_IMAGE or the SYS$PUBLIC_VECTORS image.

User Action: Use either /NOSYSSHR or /NOSYSLIB to correct the link procedure to exclude any shareable image other than SYS$BASE_IMAGE or SYS$PUBLIC_VECTORS.

BAD_NAMEORDER, filename ordering incorrect in VBN ′virtual-block-number′ of directory ′file-id′
Filenames are ′file-id′ and ′file-id′

Facility: ANALDISK, Analyze/Disk_Structure Utility

Explanation: The Analyze/Disk_Structure utility detected two files that are out of alphabetical order.

User Action: Do a SET FILE/NODIRECTORY on the directory file in which the files are located; then delete the directory file. Last, run ANALYZE /DISK/REPAIR to rename all the files from the bad directory into the SYSLOST.DIR;1 directory.

BAD_VERSORDER, version ordering incorrect in VBN ′virtual-block-number′ of directory ′file-id′
Filename is ′file-id′
Versions are ′ version-number′ and ′version-number′

Facility: ANALDISK, Analyze/Disk_Structure Utility

Explanation: The Analyze/Disk_Structure utility detected two versions of a file that are out of numerical order.

User Action: Do a SET FILE/NODIRECTORY on the directory file in which the files are located; then delete the directory file. Last, run ANALYZE /DISK/REPAIR to rename all the files from the bad directory into the SYSLOST.DIR;1 directory.

BADIMGOFF, image offset not within any image section

Facility: LOADER, Executive Image Loader

Explanation: During an image load request, a relocation or fixup operation was attempted on an image offset that has no resultant address within the image.

User Action: Disable executive slicing by specifying the LDR$V_NO_SLICE option or by using the LOAD_SYS_IMAGES system parameter. Correct the bad references in the source code.

BADINITD_MFD, Root directory 000000.DIR;1 file header incorrectly initialised, RVN ′relative-volume-number′

Facility: ANALDISK, Analyze/Disk_Structure Utility

Explanation: The Analyze/Disk_Structure utility detected an incorrectly initialized master file directory (MFD), 000000.DIR;1.

User Action: None, if you specified the /REPAIR qualifier. If you omitted /REPAIR, reenter the command with the qualifier.

BADJOBTYPE, an invalid job type was used to audit a login failure or breakin attempt

Facility: SECSRV, Security Server

Explanation: A request to audit a login failure in the security server failed because the type of job specified in the request is invalid.

User Action: Specify the job type correctly and retry the call to the intrusion system services.

BADLOCALUSERLEN, local user name length is out of range

Facility: SECSRV, Security Server

Explanation: The proxy record you want to add contains a local user name that is too long.

User Action: Specify a user name that is valid for your environment. The proxy services accept 32 characters but currently OpenVMS allows only 12.

BADNODENAMELEN, remote node name length is out of range

Facility: SECSRV, Security Server

Explanation: The proxy record you want to add contains a remote node name that is too long.

User Action: Specify a remote node name that is valid for your environment. The proxy services allow 1024 characters but currently OpenVMS allows only 256.

BADREFCNT, faulty reference count maintenance

Facility: LOADER, Executive Image Loader

Explanation: A call was made to the SYS$REMOVE_REF system service and the outstanding reference count for the image is negative.

User Action: Check all references to the routines in the executive image to ensure that each SYS$MAKE_REF has only one associated SYS$REMOVE_REF.

BADREMUSERLEN, remote user name length is out of range

Facility: SECSRV, Security Server

Explanation: The proxy record you want to add contains a remote user name that is too long.

User Action: Specify a remote user name of 32 or fewer characters.

BADSECSYS, failed to create or access SECURITY.SYS

Facility: MOUNT, Mount Utility

Explanation: The Mount utility was unable to create and write the SECURITY.SYS file on a volume that was initialized prior to OpenVMS Version 6.0, usually because the device is full.

User Action: Try any of the following actions:

- Mount the device using the /NOWRITE qualifier.
- Mount the device with /OVERRIDE=SECURITY if the security policy of the system permits this. Once the disk is mounted, free up approximately 10 blocks on the disk, and then remount the disk without using the /OVERRIDE=SECURITY qualifier.

- Mount the device on a version of OpenVMS prior to Version 6.0 and free up about 10 blocks on the disk. Then remount the disk on the later version of the system.

BADWIDTH, illegal width value 'n'

Facility: DUMP, DUMP Command

Explanation: The specified width value is illegal.

User Action: Specify a legal width value of either 80 or 132 to format the output.

CANTASSIGN, Error assigning a channel to the PK device.

Facility: STACONFIG, Standalone Configure Process

Explanation: The $ASSIGN system service did not complete successfully.

User Action: Contact a Digital support representative.

CANTRUNACP, attempt to run LATACP incorrectly

Facility: LAT, LAT Facility

Explanation: An attempt was made to run the LATACP process interactively. This action is prohibited because it can cause LATACP to start improperly.

User Action: Use the LAT$STARTUP command procedure to automatically start both LAT and LATACP.

CIADBEMPTY, no intruders or suspects currently exist

Facility: SECSRV, Security Server

Explanation: A request was made to show the contents of the intrusion database, and the database is empty.

User Action: None.

CIASHUTDOWN, breakin detection and evasion processing is shutting down

Facility: SECSRV, Security Server

Explanation: The break-in detection and evasion processing component of the security server is shutting down upon request.

User Action: None required. If you wish to restart the security server, issue the DCL command SET SERVER SECURITY/START.

CIASTARTINGUP, breakin detection and evasion processing now starting up

Facility: SECSRV, Security Server

Explanation: The break-in detection and evasion processing component of the security server is starting up, as requested.

User Action: None.

CIATERMINATED, an error caused breakin detection and evasion processing to terminate

Facility: SECSRV, Security Server

Explanation: An error occurred in the break-in detection and evasion processing component of the security server.

User Action: Contact a Digital support representative.

CONNDELETEONLY, connection can only be deleted

Facility: LAT, LAT Facility

Explanation: You tried to change a virtual circuit characteristic. This information cannot be changed.

User Action: Do not attempt to change virtual circuit characteristics.

CONNECTQUEUED, queued at service position ′position′

Facility: LAT, LAT Facility

Explanation: A LAT connection request issued to a service with no available resources has been placed in the remote node's service connection queue. When resources become available, the connection request will be completed.

User Action: Wait until the connection request is serviced. To abort the queued connection, press Ctrl/Y.

CONSTERROR, security server experienced a CONSTRAINT_ERROR exception

Facility: SECSRV, Security Server

Explanation: An error occurred in the security server.

User Action: Contact a Digital support representative.

CONVERT, converting proxy database to new format

Facility: SECSRV, Security Server

Explanation: The NETPROXY.DAT file is being converted to the new format; the new file name is NET$PROXY.DAT.

User Action: None.

CONVERT_SUCCESS, conversion of proxy database to new format was successful

Facility: SECSRV, Security Server

Explanation: The old proxy database, NETPROXY.DAT, was successfully converted to the new format database named NET$PROXY.DAT.

User Action: None.

COULDNTRESTART, security server cannot restart because of the following error:

Facility: SECSRV, Security Server

Explanation: The security server was directed to restart itself but the error reported in the accompanying message prevented this action.

User Action: Take action based on the accompanying message.

COULDNTSTART, security server cannot start functioning properly

Facility: SECSRV, Security Server

Explanation: The security server failed to start for the reason described in an accompanying message.

User Action: Take action based on the accompanying message.

CPUNOTSUPP, The processor device on SCSI bus 'device-name', ID 'slot-number', is not supported in a multihost SCSI VMScluster environment.
The processor vendor ID is 'vendor-id' and the product ID is 'product-id'.

Facility: STACONFIG, Standalone Configure Process

Explanation: An unsupported processor device was discovered on the bus.

User Action: If you are operating in a multihost SCSI VMScluster environment, remove the specified unsupported device.

If you are not operating in a multihost SCSI VMScluster environment, specify the processor device using the SYSGEN parameter SCSICLUSTER_P*n*=*abcd*, where:

n = 1, 2, 3, or 4
ab = The first two letters of the processor's vendor ID
cd = The first two letters of the processor's product ID

CREATEPROXYDB, attempting to create proxy database

Facility: SECSRV, Security Server

Explanation: As requested, the security server is attempting to create a new proxy database.

User Action: None.

CRELNM_FAILED, failed to create logical name 'name' in table 'table-name'

Facility: MOUNT, Mount Utility

Explanation: The mount operation completed successfully, but the Mount utility could not create the requested logical name. An accompanying message explains the reason for the failure.

User Action: None required. If you wish, you can manually assign the logical name after the reason for the failure has been resolved.

CREMBXFAILED, security server failed to create input mailbox

Facility: SECSRV, Security Server

Explanation: The security server could not create a mailbox.

User Action: Contact a Digital support representative.

DASSGNFAILED, security server could not deassign a channel to a client reply mailbox

Facility: SECSRV, Security Server

Explanation: The security server could not deassign the channel assigned to the reply mailbox.

User Action: Contact a Digital support representative.

DBALREADYEXISTS, proxy database already exists

Facility: SECSRV, Security Server

Explanation: The security server cannot create a new proxy database because a proxy database already exists.

User Action: Delete or rename the current proxy database before attempting to create a new one.

DELENTRY, deleted queue entry 'id'

Facility: LAT, LAT Facility

Explanation: The specified queue entry ID was deleted.

User Action: None.

DELETEDCONN, deleted connection 'connect-id'

Facility: LAT, LAT Facility

Explanation: A delete circuit operation completed successfully.

User Action: None.

DENIGNORED, /DENSITY qualifier not appropriate to this device; ignored

Facility: MOUNT, Mount Utility

Explanation: The /DENSITY qualifier was supplied on the command line for a device that does not support various densities. The qualifier is ignored.

User Action: Omit the /DENSITY qualifier to avoid getting this system message. Use the /MEDIA_FORMAT=[NO]COMPACTION qualifier if the device supports compaction.

DEVNAMLNG, device or port or node::username length restricted to 63 characters

Facility: SYSMAN, System Management Utility

Explanation: The device name parameter specified in the SYSMAN ALF ADD command exceeds 63 characters. The 63-character restriction is determined by the field size in SYSALF.DAT.

User Action: Specify a device name that is no more than 63 characters.

DRVERR, fatal drive error

Facility: MOUNT, Mount Utility

Explanation: In a SCSI VMScluster environment, a request has been made to mount a device on a shared SCSI bus and the device does not support SCSI-2 tagged command queuing (TCQ).

User Action: If you want to mount a device on a shared SCSI bus, the device must support TCQ.

DUPLICATEUSER, username already exists in the proxy record

Facility: SECSRV, Security Server

Explanation: A request was made to add a local user name to a proxy record that already contains that local user name.

User Action: Use the SHOW/PROXY command in the Authorize utility to see what proxy records exist for a specific combination of remote node name and user name.

DZRO_ISD, image contains demand zero sections

Facility: LOADER, Executive Image Loader

Explanation: A load request was made for an executive image that illegally contains demand zero sections.

User Action: Correct the link procedure used to build the executive image; specify the appropriate PSECT_ATTR and COLLECT statements to eliminate the demand zero sections.

ENTRYDELONLY, queue entry can only be deleted

Facility: LAT, LAT Facility

Explanation: An attempt was made to use a SETMODE $QIO operation on a queue entry. You can only delete queue entries from the system or use SENSEMODE $QIO operations on them.

User Action: Do not use SETMODE $QIO operations on queue entries.

EXDEPTH, exceeded allowed depth

Facility: SYSTEM, System Services

Explanation: Either a programming error has occurred or the resource name tree does not have enough depth. This error can be caused by any of the following conditions:

- The operation has exceeded the maximum number (64K) of child locks that can be associated with one parent lock.
- The operation has exceeded the maximum number of locks you can have on one resource name (64K).
- The operation has exceeded the maximum number of parent generations for a lock. The maximum is fixed at 127 for Alpha systems. The maximum for VAX systems is computed using the values for the INTSTKPAGES and DLCKEXTRASTK system generation parameters, as follows:

  ```
  (INTSTKPAGES*512 - DLCKEXTRASTK) / 32 = maximum
  ```

  The maximum VAX value can range from 4 to 255.

User Action: Reprogram the operation so that it does not exceed the maximums. If you need to increase the number of parent generations on VAX systems, you can increase the value of the INTSTKPAGES parameter.

FILFAIMAT, file failed to match selection criteria

Facility: LIB, Library Facility

Explanation: No files meet the search criteria for a PRINT or SUBMIT command.

User Action: None.

FREESPADRIFT, free block count of 'n' is incorrect (RVN 'n'); the correct value is 'n'

Facility: ANALDISK, Analyze/Disk_Structure Utility

Explanation: The free block count for this disk is incorrect.

User Action: None. The Analyze/Disk_Structure utility automatically corrects this error if you used the /REPAIR qualifier and you have the CMKRNL privilege.

INCOMPDRVACP, incompatible LATACP and LTDRIVER - LATACP terminating

Facility: LAT, LAT Facility

Explanation: An attempt to start LAT failed because of a mismatch between the LAT driver and LATACP.

User Action: Locate the mismatched image and remove it from the system.

INCONOWNER, inconsistent /OWNER_UIC option. Verify volume owner

Facility: MOUNT, Mount Utility

Explanation: The UIC specified in the /OWNER_UIC qualifier is not consistent with the protection that this device has been mounted with on other nodes in the cluster.

User Action: Verify that the /OWNER_UIC option is specified consistently on all nodes.

INCONPROT, inconsistent /PROTECTION option. Verify volume protection

Facility: MOUNT, Mount Utility

Explanation: The protection specified in the /PROTECTION qualifier is not consistent with the protection that the device has been mounted with on other nodes in the cluster.

User Action: Verify that the /PROTECTION option is specified consistently on all nodes.

INSUFINFO, not enough information to produce a breakin record

Facility: SECSRV, Security Server

Explanation: A request to produce a break-in record did not contain the required information.

User Action: Include at least one of the following in a call to request a break-in record: source terminal, source user, failed user, failed password, parent user, or source node.

INTRUDER, matching intruder found

Facility: SECSRV, Security Server

Explanation: A record matching the request was found in the intrusion database.

User Action: None. The search was successful.

INVALIDDELETE, you cannot delete the only user in a record; you must delete the entire record

Facility: SECSRV, Security Server

Explanation: A request was made to delete an explicit local user from a proxy record; however, that is the only local user in the record.

User Action: Remove the explicitly named local user from your request to delete the entire record or add another local user to this proxy record and perform the delete request again.

INVALIDTERMNAME, received invalid terminal name for intruder/suspect

Facility: SECSRV, Security Server

Explanation: An invalid terminal name was specified in a request to generate a break-in record.

User Action: Specify a valid terminal name and retry the request.

LOGRDERR, error reading LBN 'n' to LBN 'n', RVN 'n'

Facility: ANALDISK, Analyze/Disk_Structure Utility

Explanation: The Analyze/Disk_Structure utility detected an error while trying to read from disk.

User Action: Take action based on the accompanying message.

LOGWRTERR, error writing to LBN 'n', RVN 'n'

Facility: ANALDISK, Analyze/Disk_Structure Utility

Explanation: The Analyze/Disk_Structure utility detected an error while trying to write to the disk.

User Action: Take action based on the accompanying message.

LRJINCOMPVER, incompatible LAT version with remote node

Facility: LAT, LAT Facility

Explanation: An attempt was made to run LATACP with an incompatible LTDRIVER.

User Action: Check whether an incorrect LATACP was placed in SYS$SYSTEM or an incorrect LTDRIVER was placed in SYS$LOADABLE_IMAGES. Remove the incorrect file from the system.

LRJIVMSG, invalid message or slot received

Facility: LAT, LAT Facility

Explanation: An invalid LAT slot or LAT message was received from the remote node. This message indicates that a protocol violation occurred during a LAT connection.

User Action: File a problem report against the remote LAT implementation that sent the invalid LAT slot or LAT message.

LRJREMDISABLED, remote node is disabled

Facility: LAT, LAT Facility

Explanation: A LAT $QIO connection request failed because the target node has disabled LAT connections.

User Action: Enable connections on the remote node and retry the connection.

MARKUNL, exec image is marked for unload

Facility: LOADER, Executive Image Loader

Explanation: A call was made to the LDR$UNLOAD_IMAGE routine to unload a removable executive image that already has an outstanding unload request against it.

User Action: None.

MULTIPLE_ISDS, more than one image section of a given type

Facility: LOADER, Executive Image Loader

Explanation: A load request was made for an image that was not linked correctly because it contains more than one each of the following types of sections:

fixup
initialization
nonpaged code
nonpaged data
paged code

paged data

User Action: Correct the link procedure used to build the executive image; specify the appropriate PSECT_ATTR and COLLECT statements to ensure that each executive image contains only one each of the types of information listed above.

NO_PAGED_ISDS, cannot load paged ISDs in SYSBOOT

Facility: LOADER, Executive Image Loader

Explanation: SYSBOOT failed to load the executive image because it contains either paged code or paged data sections.

User Action: Correct the link procedure used to build the executive image; specify the appropriate PSECT_ATTR and COLLECT statements to eliminate the paged image sections.

NO_SUCH_IMAGE, the requested image cannot be located

Facility: LOADER, Executive Image Loader

Explanation: A load request was made for an executive image that was linked against a shareable image that is not loaded. The only legal shareable images for executives are SYS$BASE_IMAGE and SYS$PUBLIC_VECTORS.

User Action: Use /NOSYSSHR or /NOSYSLIB to correct the link procedure to exclude any shareable images other than SYS$BASE_IMAGE or SYS$PUBLIC_VECTORS.

NOACTLINKS, no active LAT links to service request

Facility: LAT, LAT Facility

Explanation: A LAT connection to the local node was attempted from the local node. The operation failed because LAT cannot communicate with the datalink layer.

User Action: Ask your system manager to properly create LAT$LINK and make it available.

NOCMKRNL, correcting free block count requires CMKRNL privilege

Facility: ANALDISK, Analyze/Disk_Structure Utility

Explanation: To correct the free block count, you must have the CMKRNL privilege.

User Action: Enable the CMKRNL privilege, then enter the ANALYZE/DISK_STRUCTURE/REPAIR command again.

NODISCON, disconnect character disabled

Facility: LAT, LAT Facility

Explanation: Use of the disconnect character has been disabled in response to a SET HOST/LAT/NODISCONNECT command.

User Action: None. This is a Success message.

NOLOGIO, erasing old home block requires LOG_IO privilege

Facility: ANALDISK, Analyze/Disk_Structure Utility

Explanation: To erase an old home block, you must have the LOG_IO privilege.

User Action: Enable the LOG_IO privilege, then enter the ANALYZE/DISK_STRUCTURE/REPAIR command again.

NOMATCH, The processor on SCSI bus ′device-name′, ID ′slot-number′, is attached using controller ′device-name′.
This condition creates multiple names for the same device, which can lead to data corruption.

Facility: STACONFIG, Standalone Configure Process

Explanation: The local and remote nodes are attached to the SCSI bus through unmatched ports.

User Action: Connect the nodes to the SCSI bus through matching ports on each system.

NOMOREENTRIES, no more entries in queue

Facility: LAT, LAT Facility

Explanation: There are no more entries in the LAT connection queue.

User Action: None.

NOPROXYDB, cannot find proxy database file NET$PROXY.DAT

Facility: SECSRV, Security Server

Explanation: The security server cannot find the proxy database file NET$PROXY.DAT.

User Action: If a proxy database exists, make sure that it is in SYS$SYSTEM or that the logical name NET$PROXY points to it.

If a proxy database does not exist, create one by using the DCL command RUN SYS$SYSTEM:CONVERT_PROXY to convert the old NETPROXY.DAT file or by issuing the CREATE/PROXY command in the Authorize utility to create a new NET$PROXY.DAT file.

NORDPROXYREC, proxy record is internally inconsistent; cannot read it

Facility: SECSRV, Security Server

Explanation: A proxy record was read from the proxy database, but the record is not internally consistent.

User Action: Examine the NETPROXY.DAT database from which the new format NET$PROXY.DAT proxy database was created. Delete erroneous records and reconvert the proxy database using the DCL command RUN SYS$SYSTEM:CONVERT_PROXY. If the conversion process does not create a NET$PROXY.DAT file, contact a Digital support representative.

NOSCANNEDINTRUDER, no matching intruder or suspect found

Facility: SECSRV, Security Server

Explanation: No record in the intrusion database matches the $SCAN_INTRUSION request; the specified user is not a suspect or intruder.

User Action: None.

NOSHARE, NO_SHARE option may not be removed from ′product′

Facility: LICENSE, License Management Utility

Explanation: The NO_SHARE option cannot be removed from a PAK that was originally issued with the NO_SHARE option. The NO_SHARE option can be removed with a LICENSE MODIFY/NONO_SHARE command only if the NO_SHARE option was added with a LICENSE MODIFY/NO_SHARE command.

User Action: If you require a PAK without the NO_SHARE option for a particular product, contact the software vendor to see if such a PAK is available.

NOSUCHINTRUDER, no intruder or suspect matches your specification

Facility: SECSRV, Security Server

Explanation: The specified intruder or suspect has no entry in the intrusion database.

User Action: None.

NOSUCHNODE, node 'node-name' not known

Facility: LAT, LAT Facility

Explanation: A LAT connection was attempted to an unknown target node.

User Action: Retry the connection until the specified node is known or check whether some condition on the specified node prohibits its network visibility. Possibly, the local node's service group codes do not intersect with the group codes of the specified remote node.

NOSUCHPROXY, no proxy record matches your specification

Facility: SECSRV, Security Server

Explanation: The requested proxy record does not exist.

User Action: Specify an existing proxy record or add a new proxy record.

NOSUCHUSER, no user matches your specification

Facility: SECSRV, Security Server

Explanation: The user name you specified does not exist in the specified proxy record.

User Action: Specify a user name that exists in the proxy record or add a new user name for the proxy record.

NOT_UNL, image is not unloadable or not loaded

Facility: LOADER, Executive Image Loader

Explanation: A call was made to LDR$UNLOAD_IMAGE to unload an executive image that is not loaded or that was not loaded with the LDR$V_UNL flag bit set.

User Action: If you want to be able to unload an executive image, be sure to set the LDR$V_UNL bit in the flags passed to the LDR$LOAD_IMAGE routine when the image is loaded. The LDR$V_UNL bit cannot be specified for OpenVMS-supplied executive images or for images loaded using VMS$SYSTEM_IMAGES.DATA.

NOTALLSET, authorization file NOT modified on some or all members; check before rebooting any member.

Facility: SYSMAN, System Management Utility

Explanation: A CONFIGURATION SET CLUSTER operation to set either the cluster group number or the cluster password failed on one or more nodes of the cluster. This message appears only when the environment is set to CLUSTER and not all the members are updated.

User Action: Ensure that all members of the cluster are updated with the new password or group number before rebooting any members.

NOTIMEOUT, no timeout period set

Facility: SYSMAN, System Management Utility

Explanation: Either SYSMAN received a timeout value of 0-00:00:00.00 for the SET TIMEOUT command (meaning that no timeout value is set for cluster communication) or a SYSMAN SHOW TIMEOUT command was issued when there is no timeout in effect.

User Action: None.

NOTNATIVE, image is not an OpenVMS AXP image

Facility: IMGACT, Image Activator

Explanation: The image is not an OpenVMS Alpha image. It might be an OpenVMS VAX image.

User Action: Use the DCL command ANALYZE/IMAGE to get more information about the image.

NOTVAXIMG, image is not an OpenVMS VAX image

Facility: IMGACT, Image Activator

Explanation: The image is not an OpenVMS VAX image. It might be an OpenVMS Alpha image.

User Action: Use the DCL command ANALYZE/IMAGE to get more information about the image.

NXR, nonexistent record

Facility: RMS, OpenVMS Record Management Services

Explanation: A secondary index data record (SIDR) points to a nonexistent primary data record. This condition (commonly referred to as a dangling SIDR) can occur when a power failure, system crash, or STOP/ID occurs while a process has modified buckets whose write-back to disk has been deferred by the deferred-write option.

This condition can be detected when a $GET or $FIND operation performs an exact-match keyed search using a secondary key. An RNF is reported in the status (STS) of the RAB and NXR is returned in the associated status value (STV) of the RAB.

User Action: None. If the associated file was opened with write access, RMS transparently cleans up the SIDR element before returning the NXR status value.

OBSPASSALL, terminal set to PASSALL mode even though /PASSALL qualifier is obsolete

Facility: SET, SET Command and SET Utility

Explanation: You specified the obsolete /PASSALL qualifier with the SET TERMINAL command.

User Action: Use the /PASSTHRU qualifier to perform this function.

OLDHM, old home block found at LBN 'n', RVN 'n'

Facility: ANALDISK, Analyze/Disk_Structure Utility

Explanation: The Analyze/Disk_Structure utility found a home block that was created by a previous initialize operation.

User Action: None. The Analyze/Disk_Structure utility automatically erases the old home block if you used the /REPAIR qualifier and you have the LOG_IO privilege.

OUTCOMTERMINATED, security server's outgoing message mechanism failed and is exiting

Facility: SECSRV, Security Server

Explanation: An error occurred in the security server's communication system and the server is shutting down.

User Action: Issue the DCL command SET SERVER SECURITY/START to restart the security server. If this condition persists, contact a Digital support representative.

PAGED_GST_TOBIG, paged global symbol fixup data exceeds table size

Facility: LOADER, Executive Image Loader

Explanation: An executive image has more global symbols in the fixup data than can fit in the loader's internal tables.

User Action: Contact a Digital support representative.

PROXYACTIVE, proxy processing is active; you must shut down proxy processing to perform this action

Facility: SECSRV, Security Server

Explanation: A request was made to start proxy processing or to create a proxy database but proxy processing is already active.

User Action: You must shut down proxy processing before performing either of these actions.

PROXYMODIFIED, existing proxy entry modified

Facility: SECSRV, Security Server

Explanation: The request to modify the proxy entry succeeded.

User Action: None.

PROXYNOTACTIVE, proxy processing is not currently active; please try your request later

Facility: SECSRV, Security Server

Explanation: Proxy processing is not active because either the security server was not started or proxy processing halted for some reason.

User Action: Check whether a new format NET$PROXY.DAT proxy database exists and use the DCL command SET SERVER SECURITY/START to start the security server if it has not been started.

If a proxy database does not exist, create one by using the DCL command RUN SYS$SYSTEM:CONVERT_PROXY to convert the old NETPROXY.DAT file or by issuing the CREATE/PROXY command in the Authorize utility to create a new NET$PROXY.DAT file.

PROXYNOTOPEN, cannot open proxy database

Facility: SECSRV, Security Server

Explanation: The proxy database NET$PROXY.DAT exists, but it cannot be opened for the reason stated in an accompanying message.

User Action: Take action based on the accompanying message.

PROXYSHUTDOWN, proxy processing is shutting down

Facility: SECSRV, Security Server

Explanation: As requested, proxy processing is shutting down.

User Action: None.

PROXYSTARTINGUP, proxy processing now starting up

Facility: SECSRV, Security Server

Explanation: As requested, proxy processing is starting up.

User Action: None.

PROXYTERMINATED, an error caused proxy processing to terminate

Facility: SECSRV, Security Server

Explanation: An error occurred in the proxy processing component of the security server.

User Action: Contact a Digital support representative.

PSB_FIXUPS, image contains LPPSB fixups, link NATIVE_ONLY

Facility: LOADER, Executive Image Loader

Explanation: A load request was made for an executive image that contains LPPSB fixup data because it was linked /NONATIVE_ONLY. Executive images must be linked /NATIVE_ONLY.

User Action: Relink the image using /NATIVE_ONLY.

QIOCALLFAIL, QIO service call failure.
Unable to obtain inquiry data from one or more SCSI initiators.

Facility: STACONFIG, Standalone Configure Process

Explanation: The $QIOW system service did not complete successfully.

User Action: Contact a Digital support representative.

QIOFAILED, security server QIO on client mailbox failed

Facility: SECSRV, Security Server

Explanation: An error has occurred in the security server.

User Action: Contact a Digital support representative.

QIOOPFAIL, QIO service operation failure.
Unable to obtain inquiry data from one or more SCSI initiators.

Facility: STACONFIG, Standalone Configure Process

Explanation: The operation requested by the $QIOW system service did not complete successfully.

User Action: Contact a Digital support representative.

RIGHTSLIM, Rights limit exceeded

Facility: SMI, System Management Integrator/Server

Explanation: A SYSMAN user with more than 125 rights identifiers attempted to issue a SYSMAN command to a remote node. This rights limitation includes a minimum of three identifiers that are granted at login when the process rights list is created: a UIC identifier, a system identifier, and at least one environmental identifier.

User Action: Run SYSMAN from another account with 125 or fewer rights identifiers or reduce the number of identifiers on the current account to fall within the required range.

RUNNING, security server is already running; it will not be restarted

Facility: SECSRV, Security Server

Explanation: An attempt was made to start a security server but another is already running.

User Action: Shut down the current server before starting another.

SAMEDEVICE, you cannot write output to the disk you are repairing

Facility: ANALDISK, Analyze/Disk_Structure Utility

Explanation: You used the /LIST, /OUTPUT, or /USAGE qualifier to write output to the disk you are repairing.

User Action: Enter the command again, making sure that none of the files specified by the /LIST, /OUTPUT, or /USAGE qualifier are on the disk you are repairing.

SERVERNOTACTIVE, security server is not active

Facility: SECSRV, Security Server

Explanation: A request was made of the security server but the server is not currently running.

User Action: Start the security server and retry the request.

SERVERRESTART, security server restarting

Facility: SECSRV, Security Server

Explanation: As requested, the security server is restarting.

User Action: None.

SERVERSHUTDOWN, security server shutting down

Facility: SECSRV, Security Server

Explanation: As requested, the security server is shutting down.

User Action: None required. If you wish to start a security server, use the DCL command SET SERVER SECURITY/START.

SERVERSTARTINGUP, security server starting up

Facility: SECSRV, Security Server

Explanation: As requested, the security server is starting up.

User Action: None.

SERVERTERMINATED, an error caused the security server to terminate

Facility: SECSRV, Security Server

Explanation: An error caused the security server to perform cleanup and terminate break-in detection, evasion processing, and proxy processing.

User Action: Use the DCL command SET SERVER SECURITY/START to restart the security server. If this condition persists, contact a Digital support representative.

SHADOWLBS, clearing low-order bit in SHADOWING; function no longer supported

Facility: SYSBOOT, System Bootstrap Facility

Explanation: The low-order bit in the SYSGEN parameter SHADOWING was set at boot time. This bit has been cleared and the system was booted from the physical device named to VMB register 3 (R3). Unless you have migrated to phase II volume shadowing, no shadow sets will be created.

User Action: None.

SPF_TOBIG, second pass fixup data exceeds table size

Facility: LOADER, Executive Image Loader

Explanation: The loader's internal tables cannot accommodate all of the executive image fixups that must be postponed to later in the bootstrap operation.

User Action: Contact a Digital support representative.

SRVREPLYTIMEOUT, timed out waiting for reply from security server

Facility: SECSRV, Security Server

Explanation: The security server did not reply to a request within the timeout period. Most likely, the security server encountered an error while processing the request and had to restart itself.

User Action: Check the operator log and correct any error condition that caused the security server to terminate. If this problem persists, contact a Digital support representative.

SUSPECT, matching suspect found

Facility: SECSRV, Security Server

Explanation: A record matching the request was found in the intrusion database.

User Action: None. The search was successful.

TAPEALLODIFFF, The TAPE_ALLOCLASS parameter value for the processor on SCSI bus 'device-name', ID 'slot-number', is different from the value on this system.
This condition creates multiple names for the same device, which can lead to data corruption.

Facility: STACONFIG, Standalone Configure Process

Explanation: The value of the SYSGEN tape allocation class parameter differs between the local node and the specified remote node.

User Action: Set the TAPE_ALLOCLASS parameter to the same nonzero value on both nodes.

TAPEALLOZERF, This system and/or the processor on SCSI bus ′device-name′, ID ′slot-number′, has a zero TAPE_ALLOCLASS value. This condition creates multiple names for the same device, which can lead to data corruption.

Facility: STACONFIG, Standalone Configure Process

Explanation: The local node, remote node, or both nodes have the SYSGEN tape allocation class parameter set to zero.

User Action: Set the TAPE_ALLOCLASS parameter to the same nonzero value for both nodes.

TASKERROR, security server experienced a TASKING_ERROR exception

Facility: SECSRV, Security Server

Explanation: An error occurred in the security server.

User Action: Contact a Digital support representative.

TOOMANYELTS, an attempt was made to ′action′ ′nnnn′ ′class-name′ elements. The maximum allowed is ′nnnn′.

Facility: MONITOR, Monitor Utility

Explanation: MONITOR attempted to display or record a number of elements that exceeds the maximum allowed for the class of data specified by *class-name*.

When this message is seen with the DISK class, the elements being displayed or recorded are individual disks. MONITOR will attempt to process all the mounted disks seen by the system. If that number exceeds MONITOR's internal limits, the MONITOR request terminates with this message.

User Action: If your MONITOR request selected the /RECORD option, you have reached the record limit of 909. If you reissue the request without the /RECORD qualifier, and select display and/or summary output, you can monitor as many as 1817 disks without recording.

Because MONITOR collects information about mounted disks only, you can reduce the number of disks monitored by dismounting unused disks.

TOOMANYUSERS, proxy already has the maximum number of associated users

Facility: SECSRV, Security Server

Explanation: A request was made to add a local user entry to a proxy record that already has the maximum number of local users.

User Action: Delete a local user entry to make room for the new one.

UNDEFENTRY, invalid queue entry request ID

Facility: LAT, LAT Facility

Explanation: The queue entry you specified does not exist.

User Action: Use the LATCP SHOW QUEUE_ENTRY command to list all entries in the local queue. Retry the command, using a valid queue entry.

UNL_PEN, exec image unload is pending

Facility: LOADER, Executive Image Loader

Explanation: A call was made to LDR$UNLOAD_IMAGE to unload an executive image that is in use. The image is marked to be unloaded later.

User Action: None.

VALUOVRFLOW, value too large for 'system-parameter'; field width is only 'n' bytes

Facility: SYSMAN, System Management Utility

Explanation: A system parameter is set to a value that exceeds its actual field width. With SYSGEN, you can set a parameter value to more than its actual field width without getting an error message but SYSGEN silently truncates the value to fit the parameter's field width. In contrast, SYSMAN gives a warning and does not change the value of the parameter even if PARAMETER DISABLE CHECK is in effect.

User Action: Set the parameter to a value within the acceptable range.

VEC_TOBIG, symbol vector reset data exceeds table size

Facility: LOADER, Executive Image Loader

Explanation: An attempt to load an executive image failed because the image's symbol vector updates for SYS$BASE_IMAGE and SYS$PUBLIC_VECTORS exceed the size of the loader's internal tables.

User Action: Contact a Digital support representative.

VERIFY_CONVERSION, verifying that proxy database conversion is correct

Facility: SECSRV, Security Server

Explanation: The contents of the new proxy database, NET$PROXY.DAT, are being compared to the contents of the old proxy database, NETPROXY.DAT.

User Action: None.

ZEROALLOCLS, unit has zero allocation class

Facility: SYSTEM, System Services

Explanation: The physical unit cannot be added to the shadow set because the allocation class is zero.

User Action: Make sure all the disks being used to make the shadow set are configured with an allocation class. Then reenter the command to form or add to the shadow set. See the *VMScluster Systems for OpenVMS* manual for more information about allocation classes.

Index