

# HP RAID Software for OpenVMS

---

## Guide to Operations

This guide to operations provides user and management information for the HP RAID Software for OpenVMS layered software product.

**Software Version:**

Version 3.0

**Hewlett-Packard Company**  
**Palo Alto, California**

---

**January 2005**

Copyright 2005 Hewlett-Packard Development Company, L.P.

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

Intel and Itanium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Printed in the US

This document was prepared using VAX DOCUMENT, Version 2.1.

---

# Contents

<b>Preface</b> .....	xi
<b>1 What Is RAID?</b>	
1.1 RAID Technology Overview .....	1-1
1.2 HP Supported RAID Levels .....	1-3
1.3 Controller-Based Versus Host-Based RAID .....	1-4
1.3.1 Characteristics of Host-Based RAID Software Products .....	1-4
1.3.2 Characteristics of Controller-Based RAID Software Products .....	1-5
<b>2 Overview of RAID Functionality</b>	
2.1 What Is Supported .....	2-1
2.2 RAID Arrays .....	2-2
2.2.1 Member Structure .....	2-3
2.2.2 RAID Virtual Device Definition .....	2-3
2.2.3 Chunks .....	2-4
2.2.4 Parity Chunks .....	2-4
2.2.5 RAID0 User Data Mapping .....	2-4
2.2.6 RAID5 User Data Mapping .....	2-5
2.2.7 Size of the Array .....	2-5
2.2.8 Capacity of the Array .....	2-5
2.2.9 Array Capacity Calculation .....	2-6
2.3 The Member Volume File System Definition .....	2-7
2.4 Additional Control Information .....	2-7
2.5 RAID Spares and Sparesets <b>RAID5</b> .....	2-7
2.5.1 Definition of a Spare .....	2-7
2.5.2 Definition of a Spareset .....	2-7
2.5.3 Do I Need a Spareset? .....	2-8
2.5.4 Incorporating a Spare into a Spareset .....	2-8
2.5.5 Associating Sparesets with RAID Arrays .....	2-8
2.5.6 Limitations on Associating Sparesets with RAID Arrays .....	2-9
<b>3 Preparing to Use HP RAID Software for OpenVMS</b>	
3.1 RAID Array Planning Checklist .....	3-1
3.2 RAID Array Restrictions .....	3-2
3.3 User-Settable Attributes .....	3-2
3.3.1 Attributes That Affect Every Array on the CPU .....	3-2
3.3.2 Attributes That Affect Only Individual Arrays .....	3-3

## 4 Creating and Managing RAID Arrays

4.1	Creating RAID Arrays	4-1
4.1.1	Step 1: Defining a RAID Array	4-2
4.1.2	Step 2: Associating a RAID Array with a Virtual Device	4-3
4.1.3	Step 3: Placing a File System on the RAID Virtual Device	4-4
4.1.4	Step 4: Mounting an OpenVMS File System on the RAID Virtual Device	4-5
4.1.5	Example of RAID Array Creation	4-5
4.1.6	Creating a RAID Array on a VMScluster System	4-7
4.2	Undefining RAID Arrays	4-9
4.2.1	Disassociating the RAID Array from the Virtual Device	4-9
4.2.2	Reassociating the RAID Array with the Virtual Device	4-10
4.3	Changing the Characteristics of Your RAID Array	4-10
4.3.1	The RAID MODIFY Command	4-11
4.3.2	Removing a RAID Array Member [RAID5]	4-11
4.3.3	Replacing a Member in RAID Arrays [RAID5]	4-12
4.3.4	Learning about Your Array	4-12
4.4	Sparesets [RAID5]	4-12
4.4.1	Initializing Spares [RAID5]	4-13
4.4.2	Associating Spares with a Spareset [RAID5]	4-13
4.4.3	Disassociating Spares from a Spareset [RAID5]	4-13
4.4.4	Associating a Spareset with a RAID Array [RAID5]	4-14
4.4.5	Adding Spares to Sparesets [RAID5]	4-14
4.4.6	Removing Spares from Sparesets [RAID5]	4-14
4.4.7	RAID5 Spare Set Management [RAID5]	4-15
4.4.8	Learning about Your Spareset	4-15
4.5	Maintaining a RAID Array	4-15
4.5.1	Maintenance for RAID 5 Sets	4-15
4.6	Obtaining Information About RAID Arrays	4-16
4.6.1	Obtaining Information about RAID Arrays via Command Procedures	4-16
4.6.2	Obtaining Information about RAID Arrays Using the ANALYZE Command	4-18
4.6.2.1	Four RAID Created Files	4-18
4.6.2.2	Configuration File Report	4-18
4.6.2.3	Container File Report	4-19
4.7	Managing RAID Arrays Automatically	4-21

## 5 Partitioning

5.1	What Is Partitioning?	5-1
5.2	Why Use Partitioning?	5-1
5.3	Using Partitioning	5-1
5.3.1	Creating Partitions with RAID INITIALIZE	5-2
5.3.1.1	Partitioning by Default	5-2
5.3.1.2	Partitioning Using Specified Sizes	5-2
5.3.2	Binding	5-3
5.4	Partitioning Examples	5-3
5.4.1	Partitions in a RAID0 Array	5-3
5.4.2	Partitions in a RAID0+1 Array	5-4
5.4.3	Partitions with Truncation	5-5
5.4.4	Partitions with Specified Size Exceeding Disk Space	5-6
5.4.5	A BIND Command Without Enough Virtual Devices Specified	5-7
5.4.6	A BIND Command with All Partitions Specified	5-7

## 6 Using Shadowed Disks in RAID0 Arrays

6.1	Using Shadow Sets as Members of RAID0 Configurations	6-1
6.2	Why Use Volume Shadowing with RAID0 Arrays	6-2
6.3	Requirements for Using Volume Shadowing	6-3
6.4	Device Hierarchy in a RAID0 Array with Volume Shadowing	6-4
6.5	Related Documentation	6-4
6.6	Creating RAID0 Arrays with Shadow Sets	6-4
6.6.1	RAID Initializing a RAID0 Array Using Shadow Sets	6-5
6.6.2	Creating RAID0 Virtual Devices	6-5
6.6.3	How DSA Device Names Are Assigned	6-6
6.6.4	Initializing and Mounting the RAID0 Virtual Device	6-6
6.7	Adding Shadow Set Member Devices to a RAID0 Shadow Set	6-7
6.8	Removing Members from a Shadow Set	6-8
6.9	Rebinding with Shadowing	6-8
6.9.1	Converting a Nonshadowed RAID0 Array to a Shadowed RAID0 Array	6-8
6.10	Rebinding Without Shadowing	6-9
6.11	Backup of Arrays with Shadow Sets	6-10
6.12	RAID CLONE Command	6-10
6.12.1	Support for Shadow Minicopy	6-11
6.13	Using RAID0 and Shadowing in a VMSccluster	6-12
6.14	Performance Considerations	6-13
6.15	Error Handling	6-13

## 7 Internal Operation and Error Handling

7.1	RAID Array States	7-1
7.1.1	Normal State	7-2
7.1.2	Reduced State <b>RAID5</b>	7-2
7.1.3	Reconstructing State <b>RAID5</b>	7-2
7.2	RAID Array Status	7-3
7.2.1	Startup Status	7-3
7.2.2	RAID Array Inoperative Status	7-4
7.3	RAID Array Member States	7-4
7.4	RAID Virtual Unit Status	7-5
7.4.1	Accessible Status	7-5
7.4.2	Startup Status	7-5
7.5	RAID Shadow Set States <b>RAID0</b>	7-5
7.5.1	SteadyState State <b>RAID0</b>	7-5
7.5.2	ShadowMerging State <b>RAID0</b>	7-5
7.5.3	ShadowCopying State <b>RAID0</b>	7-6
7.5.4	Unknown State <b>RAID0</b>	7-6
7.6	HP RAID Error Recovery	7-6
7.6.1	Member Error Handling	7-6
7.6.2	Timeout Mechanism <b>RAID5</b>	7-7
7.6.3	Localized Error Handling	7-9
7.6.4	Handling Global Errors and Errors Accessing Control Data	7-10
7.6.5	Summary of Error Handling	7-10
7.7	VMSccluster State Transitions	7-11
7.8	Reconstruction Determinations <b>RAID5</b>	7-11

## 8 RAID0: Improving Performance

8.1	Benefits of RAID0 Technology . . . . .	8-1
8.2	I/O Workloads That Can Benefit from RAID0 Arrays . . . . .	8-2
8.3	Determining Whether RAID0 Arrays Will Improve Performance . . . . .	8-2
8.4	Configuring RAID0 Arrays for Data Transfer-Intensive I/O Workloads . . . . .	8-3
8.4.1	Designing RAID0 Arrays for High Data Transfer Rates . . . . .	8-4
8.4.2	Designing Applications for Maximum Data Transfer Rate . . . . .	8-5
8.5	RAID0 Arrays and I/O Load Balancing . . . . .	8-5
8.6	Configuring RAID0 Arrays for Request Rate-Intensive I/O Workloads . . . . .	8-6
8.7	Tools for Performance Analysis . . . . .	8-6
8.8	How RAID0 Plus Shadowing Helps Performance . . . . .	8-10
8.9	Guideline for RAID0 Arrays . . . . .	8-10
8.9.1	Number of Array Members . . . . .	8-10
8.9.2	Disk Capacity . . . . .	8-10
8.9.3	Chunk Size . . . . .	8-11
8.9.4	Queue Depth . . . . .	8-12
8.9.5	Disk MSCP-Serving . . . . .	8-12

## 9 RAID5: Improving Data Reliability, Availability, and Performance

9.1	Concerns for Storage Administrators . . . . .	9-1
9.2	Using RAID5 Array Size to Balance Cost, Availability, and Performance RAID5 . . . . .	9-3
9.2.1	Number of Disks Per RAID5 Array: Cost Versus Performance RAID5 . . . . .	9-5
9.3	Implications of Using Unlike Disks . . . . .	9-5
9.4	Data Reliability Versus Availability RAID5 . . . . .	9-6
9.4.1	Protecting Against Disk and Storage Element Failure RAID5 . . . . .	9-10
9.4.2	Protecting Against Controller Failure RAID5 . . . . .	9-11
9.4.3	Protecting Against Host Adapter Failure RAID5 . . . . .	9-13
9.5	Reconstruction, Data Reliability, and Performance RAID5 . . . . .	9-17
9.6	Configuring RAID5 Arrays and Sparesets RAID5 . . . . .	9-18

## 10 The OpenVMS Operating System Interface

10.1	Command Line Interface . . . . .	10-1
10.2	Using the OpenVMS BACKUP Command . . . . .	10-2
10.3	System Shutdown with the RAID SHUTDOWN Command . . . . .	10-2
10.4	Configuring the Software with the RAID SET Command . . . . .	10-2
10.5	How to Access OpenVMS Help for RAID . . . . .	10-2
10.6	OpenVMS Error Messages . . . . .	10-2
10.7	The Diagnostics File . . . . .	10-3
10.8	Other Files . . . . .	10-3
10.9	How to Access Your RAID Array Through the Virtual Device . . . . .	10-4

## 11 RAID Commands

11.1	Event_Procedure . . . . .	11-1
11.1.1	Event_types . . . . .	11-2
11.1.2	Logical_names . . . . .	11-3

11.1.3	Command Procedure Example	11-3
	RAID ADD/SHADOW_MEMBER [RAID0]	11-4
	RAID ADD/SPARESET [RAID5]	11-6
	RAID ANALYZE	11-7
	RAID ANALYZE/ERROR_LOG	11-9
	RAID BIND	11-11
	RAID BIND/SPARESET [RAID5]	11-15
	RAID CLONE [RAID0]	11-17
	RAID INITIALIZE	11-19
	RAID INITIALIZE/SPARE [RAID5]	11-22
	RAID MODIFY	11-24
	RAID REMOVE [RAID5]	11-26
	RAID REMOVE/SHADOW_MEMBER [RAID0]	11-27
	RAID REMOVE/SPARE [RAID5]	11-28
	RAID REPLACE [RAID5]	11-29
	RAID SET	11-30
	RAID SHOW	11-32
	RAID SHOW/SPARESET [RAID5]	11-42
	RAID SHUTDOWN	11-44
	RAID UNBIND	11-45
	RAID UNBIND/SPARESET [RAID5]	11-46

## A RAID OPCOM Information Messages

A.1	Message Format	A-1
A.2	RAID OPCOM Messages	A-2

## B RAID Command Messages

B.1	Message Format	B-1
B.2	RAID CLI and Server Messages	B-2

## C QIO Interface

C.1	Disk I/O Functions	C-1
-----	--------------------	-----

## D DCL Command files to Assist with RAID Management

D.1	Obtaining Information about RAID Arrays and Sparesets	D-1
D.1.1	RAID\$CONFIG.COM	D-1
D.1.2	RAID\$DISPLAY.COM	D-3

## Glossary

## Index

### Examples

4-1	Creating RAID Arrays .....	4-5
4-2	ANALYZE/REPAIR Example .....	4-16
4-3	ANALYZE/DISK/READ Example .....	4-16
4-4	Sample Configuration File Report .....	4-18
4-5	Sample Container File .....	4-19
4-6	Sample Event Notification procedure .....	4-22
5-1	Creating Partitions on a RAID0 Array .....	5-3
5-2	Creating Partitions on a RAID0 Array .....	5-4
5-3	Creating Partitions with Truncation .....	5-6
5-4	Partitions with Specified Size Exceeding Disk Space .....	5-6
5-5	A BIND Command Without Enough Virtual Devices Specified .....	5-7
5-6	A BIND Command with All Partitions Specified Correctly .....	5-7
8-1	Sample RAID SHOW/FULL Command Report .....	8-8
11-1	Command Procedure Example .....	11-3
11-2	RAID SHOW/BRIEF Command .....	11-33
11-3	RAID SHOW Command for RAID5 Array (Default Version) .....	11-34
11-4	RAID SHOW Command for RAID5 Array (During Reconstruct) .....	11-35
11-5	RAID SHOW Command for RAID0 Array .....	11-36
11-6	RAID SHOW /FULL Command for RAID0 Array with Shadowed Members .....	11-38
11-7	RAID SHOW Command for RAID0 Array (With Inoperative Status) .....	11-40
11-8	RAID SHOW /SPARESET Command .....	11-43

### Figures

1-1	Comparative Strengths and Weaknesses of RAID Levels .....	1-4
2-1	A RAID Array of Disks Appear as a Single Virtual Device .....	2-3
2-2	RAID0 Chunk Mapping .....	2-4
2-3	RAID5 Chunk Mapping .....	2-5
2-4	Capacity of a RAID Array .....	2-6
2-5	Different RAID Arrays May Share the Same Spareset .....	2-8
2-6	Spare Candidates .....	2-9
3-1	RAID Planning Checklist .....	3-5
3-2	RAID Membership Checklist .....	3-6
4-1	RAID BIND Command Is Entered at One Node .....	4-7
4-2	BIND Command Propagates to the Other Nodes .....	4-8
6-1	RAID0 Arrays and Shadow Sets .....	6-2
6-2	RAID0 Array with Two Members .....	6-3
6-3	RAID0 Array with a Different Number of Shadow Set Members .....	6-5
6-4	Sample RAID0 Array Using Shadow Sets .....	6-7
6-5	RAID0 Array with No Shadow Set Members .....	6-9
6-6	RAID0 Array With Two Shadow Sets .....	6-12



6-7	Two RAID0 Arrays . . . . .	6-13
8-1	Host Memory to Disk Drive Data Path . . . . .	8-4
8-2	Four-Member RAID0 Array with Shadow Sets . . . . .	8-11
9-1	User Requirements . . . . .	9-2
9-2	Data Reliability . . . . .	9-4
9-3	Generic Subsystem . . . . .	9-7
9-4	Sample HP Systems . . . . .	9-8
9-5	Simple RAID Arrays . . . . .	9-10
9-6	Duplicate Controller Configuration . . . . .	9-11
9-7	Multiple Controller Configuration . . . . .	9-12
9-8	Multiple Controllers with Multiple RAID Arrays . . . . .	9-13
9-9	Protection Against Host Adapter Failure . . . . .	9-14
9-10	Protection Against Host Adapter Failure . . . . .	9-15
9-11	VMScluster Technology . . . . .	9-16
9-12	Small System Model VMScluster System . . . . .	9-17

## Tables

1	Conventions Used in This Guide . . . . .	xii
1-1	Topics in This Chapter . . . . .	1-1
1-2	RAID Levels . . . . .	1-2
1-3	HP Supported RAID Levels . . . . .	1-3
2-1	Topics in This Chapter . . . . .	2-1
3-1	Topics in This Chapter . . . . .	3-1
3-2	Planning Checklist . . . . .	3-2
4-1	Topics in This Chapter . . . . .	4-1
4-2	Steps to Creating a RAID Array . . . . .	4-1
4-3	Example: RAID INITIALIZE/RAID_LEVEL=5 . . . . .	4-3
4-4	Example: RAID BIND Command . . . . .	4-4
4-5	Example: Placing a File System on the RAID Array . . . . .	4-5
4-6	Example: Mounting an OpenVMS File System on the RAID Array . . . . .	4-5
4-7	Steps to Creating a RAID Array on a VMScluster System . . . . .	4-9
4-8	Example: RAID UNBIND Command . . . . .	4-10
4-9	Example: RAID MODIFY Command . . . . .	4-11
4-10	Example: RAID REMOVE Command . . . . .	4-12
4-11	Example: RAID REPLACE Command . . . . .	4-12
4-12	Example: RAID INITIALIZE/SPARE Command . . . . .	4-13
4-13	Example: RAID BIND/SPARESET Command . . . . .	4-13
4-14	Example: Unbinding Sparesets Command . . . . .	4-13
4-15	Example: Associating a Spareset with a RAID Array . . . . .	4-14
4-16	Example: Adding Spares to Sparesets Command . . . . .	4-14
4-17	Example: Removing Spares from Sparesets Command . . . . .	4-14
5-1	Topics in This Chapter . . . . .	5-1
6-1	Topics in This Chapter . . . . .	6-1
6-2	Resulting Shadow Sets and Device Names . . . . .	6-6
7-1	Topics in This Chapter . . . . .	7-1
7-2	Localized Read and Write Errors . . . . .	7-9

7-3	Global Read and Write Errors and Errors Accessing Control Data . . .	7-10
8-1	Topics in This Chapter . . . . .	8-1
8-2	RAID0 Effectiveness . . . . .	8-3
8-3	Description of RAID SHOW Command Printout (Brief Version) . . . . .	8-10
9-1	Topics in This Chapter . . . . .	9-1
9-2	Performance, Cost, and Availability Considerations . . . . .	9-2
9-3	Read Performance . . . . .	9-5
9-4	Write Performance . . . . .	9-5
9-5	Subsystem Components . . . . .	9-7
9-6	Enhancing Data Reliability and Availability in Small and Medium System Models . . . . .	9-9
9-7	Enhancing Data Reliability and Availability in Multidrive-Based Subsystems . . . . .	9-9
10-1	Topics in This Chapter . . . . .	10-1
11-1	RAID Commands . . . . .	11-1
11-2	IO_TIMEOUT and TIMEOUT Interaction . . . . .	11-12
11-3	Resulting Shadow Sets . . . . .	11-14
11-4	Description of RAID SHOW Command Printout . . . . .	11-33
11-5	Description of RAID SHOW Command Printout . . . . .	11-35
11-6	Description of RAID SHOW Command Printout (Brief Version During Reconstruct) . . . . .	11-36
11-7	Description of RAID SHOW Command Printout (Brief Version) . . . . .	11-37
11-8	Description of RAID SHOW /FULL Command Printout . . . . .	11-39
11-9	Description of RAID SHOW Command Printout (With Inoperative Status) . . . . .	11-40
11-10	Description of RAID SHOW/SPARESET Command Printout . . . . .	11-43
A-1	Severity Levels . . . . .	A-1
D-1	Information Returned for Arrays . . . . .	D-1
D-2	RAID 0+1 Array Information Returned . . . . .	D-2
D-3	Information Returned for Saresets . . . . .	D-2

---

# Preface

## Purpose of This Guide

This guide to operations provides information on using and managing the HP RAID product for the OpenVMS operating system (versions listed in the release notes). This guide also contains instructions for using the DCL command interface, and a listing of system messages that are unique to HP RAID activities.

## Intended Audience

This guide to operations is designed for three major audiences:

- The system administrator who installs the HP RAID software, and configures and manages the RAID arrays.
- The user, who is either a programmer developing applications using HP RAID, or an operator running those applications.
- HP Services and customer support engineers.

## Related Documentation

The following related documents contain information you might find helpful.

- HP RAID Software for OpenVMS Guide to Operations
- HP RAID Software for OpenVMS Installation Guide
- HP OpenVMS System Manager's Manual: Essentials
- HP OpenVMS System Management Utilities Reference Manual: A-L  
BACKUP
- HP OpenVMS System Management Utilities Reference Manual: M-Z MOUNT  
and SYSGEN
- HP OpenVMS DCL Dictionary: A-M
- HP OpenVMS DCL Dictionary: N-Z
- HP OpenVMS I/O User's Reference Manual
- OpenVMS System Messages: Companion Guide for Help Message Users
- OpenVMS Cluster Systems

## Reader's Comments

HP welcomes your comments on this manual. Please send comments to either of the following addresses:

Internet	<b>openvmsdoc@hp.com</b>
Postal Mail	Hewlett-Packard Company OSSG Documentation Group, ZKO3-4/U08 110 Spit Brook Rd. Nashua, NH 03062-2698

## Conventions Used in this Guide

The conventions shown in Table 1 are used in this guide.

**Table 1 Conventions Used in This Guide**

Convention	Meaning
[parameter]	In command formats, brackets indicate optional parameters. When you enter the optional parameter, do not enter the brackets.
\$ SHOW TIME 18-MAY-2001 12:42:16	Command examples show the OpenVMS prompt character (\$). The command (what you enter) is in uppercase type.
<i>italic text</i>	In examples and messages, italic text represents input that is unique for each system. In other areas, italic text is used to introduce a new term.
<b>boldface text</b>	Boldface text is used to introduce a new term that is also found in the glossary.
<span style="border: 1px solid black; padding: 2px;">XXXXX</span>	A key name enclosed in a box indicates that you should press that key on the keyboard (for example <span style="border: 1px solid black; padding: 2px;">Return</span> or <span style="border: 1px solid black; padding: 2px;">Help</span> ).
data 1 . . . data n	In examples, a vertical ellipsis represents the omission of data that the system displays in response to a command or data that a user enters.
filespec[. . . ]	In command formats, a horizontal ellipsis indicates that the preceding item can be repeated one or more times.
OpenVMS VAX	The OpenVMS operating system for VAX hardware.
OpenVMS Alpha	The OpenVMS operating system for Alpha hardware.
OpenVMS, VMS	The terms OpenVMS and VMS refer to the OpenVMS operating system.
<span style="border: 1px solid black; padding: 2px;">RAID0</span>	The <span style="border: 1px solid black; padding: 2px;">RAID0</span> icon indicates that the following information applies to only RAID0 arrays. If no icon is used, assume that the information applies to both RAID0 and RAID5 arrays.
<span style="border: 1px solid black; padding: 2px;">RAID5</span>	The <span style="border: 1px solid black; padding: 2px;">RAID5</span> icon indicates that the following information applies to only RAID5 arrays. If no icon is used, assume that the information applies to both RAID0 and RAID5 arrays.

---

# What Is RAID?

This chapter provides a general overview of RAID technology and takes a look at the specific RAID products offered by Hewlett-Packard Company. Table 1-1 provides a list of topics in this chapter.

**Table 1-1 Topics in This Chapter**

Subject	Section
RAID Technology Overview	1.1
HP Supported RAID Levels	1.2
Controller-Based Versus Host-Based RAID	1.3

## 1.1 RAID Technology Overview

RAID, or Redundant Array of Independent<sup>1</sup> Disks, distributes data on multiple disks. Data requests are broken down and distributed to physical storage components that process these requests in parallel. With most RAID levels, shown in Table 1-2, redundant data is calculated and placed on another physical storage component or interleaved with the user data.

RAID technology has the following attributes:

- RAID views a number of disks as a logical unit called a RAID **virtual device**.
- RAID distributes user data across the physical disks.
- Most RAID levels provide redundant capacity so that data can be recovered even when a physical disk fails.

The manner in which the user data is distributed and how the redundant capacity is implemented distinguishes the different RAID levels.

---

<sup>1</sup> Sometimes referred to in the industry as *inexpensive*.

**Table 1–2 RAID Levels**

<b>RAID Level</b>	<b>Definition</b>
RAID 0	Also known as striping, RAID 0 is the only RAID level that does not provide redundancy. User data is distributed or "striped" across all the disks.
RAID 1	Also known as mirroring or shadowing, RAID 1 provides redundancy by duplicating the user data from the first disk onto one or more disks.
RAID 0+1	RAID 0+1 is a combination of mirroring and striping and provides the best combination of high performance and high reliability.
RAID 2	RAID 2 distributes data across a set of disks, using Hamming code to provide data integrity.
RAID 3	RAID 3 distributes data across a set of disks. RAID 3 creates the data parity on-the-fly and places it on a separate disk to provide redundancy. Every read or write operation accesses every disk.
RAID 4	RAID 4 distributes data across a set of disks. RAID 4 creates the data parity on-the-fly and places it on a single separate disk to provide redundancy. During reads and writes, only the data and parity disks are accessed rather than every disk.
RAID 5	RAID 5 is the same as RAID 4, except parity is distributed across the entire set of disks.
RAID 6	RAID 6 provides redundancy by calculating redundant information using two separate functions and placing the information in two separate locations for each original piece of data. Thus, any two disks may fail without loss of user data.

## 1.2 HP Supported RAID Levels

The RAID levels supported by HP provide trade-offs between performance, data availability, and cost. You may want to choose a RAID level based on which features are most important to your application.

Table 1–3 summarizes the characteristics of the different RAID levels. The performance, data availability, and relative cost factors provided in Table 1–3 were obtained by comparing disks that use the RAID technology against a collection of disks that do not use RAID technology (sometimes referred to in the industry as “just a bunch of disks,” or JBOD).

**Table 1–3 HP Supported RAID Levels**

Level	Relative Availability	Request Rate (Read/Write) §	Data Rate (Read/Write)§	Cost Factor†	Types of Applications
RAID 0 (HP RAID Software for OpenVMS)	Proportionate to number of disks; lower than one disk.	Large chunks‡: Excellent/excellent	Small chunks‡: Excellent/excellent	1.0	Batch data analysis
RAID 1 (Volume shadowing for OpenVMS)	Excellent	Excellent/fair	Good/good	2.0	System disks, critical files
RAID 0+1 (Combination of striping and shadowing)	Excellent	Large chunks: excellent/excellent	Small chunks: excellent/excellent	2.0	Any critical response-time and data availability application
RAID 5 (HP RAID Software for OpenVMS)	Excellent	Excellent/poor	Excellent/fair	1.25	High request rate read-intensive data lookup, any critical data availability application

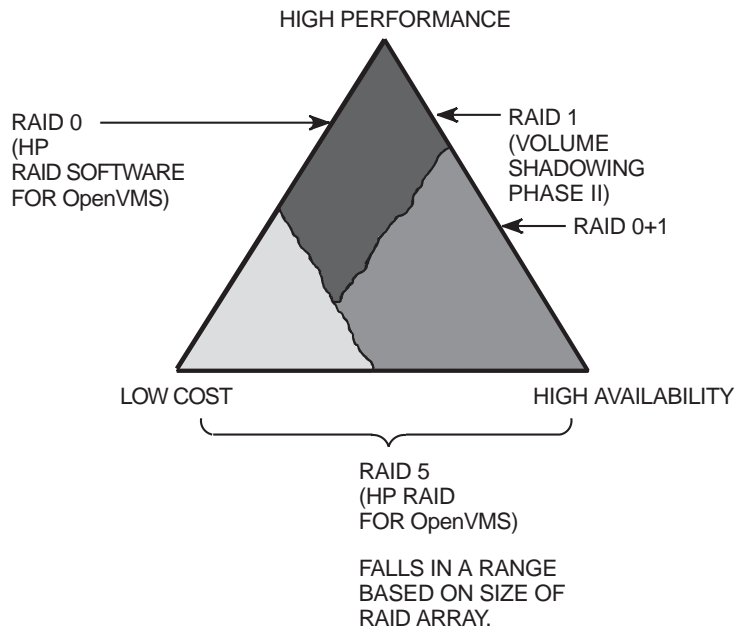
§Based on chunk size/request size ratio. Can optimize for request rate or data rate.

†Cost factor is an approximate multiplier of ordinary disk cost to achieve the specified level of RAID. This table assumes a 4+1 (five) member set of disks for RAID and a 4+4 member set of disks for shadowing.

‡A chunk is a group of consecutive virtual device logical blocks.

Figure 1–1 represents the balance between cost, performance, and availability of the various RAID levels. As you can see by looking at the triangle, the three aspects of the triangle can often be conflicting. Wanting higher performance, for example, will likely increase your costs. Wanting lower costs may mean giving up a certain amount of data availability. System administrators and others must choose which areas of the triangle are most important to them.

**Figure 1–1 Comparative Strengths and Weaknesses of RAID Levels**



CXO3713BRA

## 1.3 Controller-Based Versus Host-Based RAID

HP RAID Software for OpenVMS is a host-based product. Some RAID products on the market may be controller-based. The general characteristics of host-based and controller-based RAID products are summarized in the following sections.

### 1.3.1 Characteristics of Host-Based RAID Software Products

The characteristics of host-based RAID software products are as follows:

- Implemented in software on host CPUs
- Uses host CPU and memory resources
- May use multiple host adapters and multiple controllers for increased data rates and request rates
- Can be used with older, existing disks and controllers
- Implemented as a layered software product (costs money), but usually less expensive than new controller/disk hardware
- May not be able to boot the operating system from a RAID array
- Provides host-based RAID of shadow sets with higher availability than shadowing controller-based stripe sets.



### **1.3.2 Characteristics of Controller-Based RAID Software Products**

The characteristics of controller-based RAID software products are as follows:

- Implemented in firmware in the controller
- Takes no host CPU or memory resources
- Data rate is limited by the host-to-controller connection; request rate is usually limited by the controller CPU.
- Requires use of new controllers with RAID functionality (and new controllers may require new types of disks)
- Can boot operating system from a RAID array
- Can be used with operating systems which have no RAID driver



---

## Overview of RAID Functionality

This chapter provides an overview of HP RAID Software for OpenVMS. This software supports both RAID0 and RAID5 functionality. Table 2–1 lists the topics included in this chapter.

**Table 2–1 Topics in This Chapter**

Subject	Section
What Is Supported	2.1
RAID Arrays	2.2
The Member Volume File System Definition	2.3
Additional Control Information	2.4
RAID Spares and Sparesets	2.5

### 2.1 What Is Supported

HP RAID Software for OpenVMS conforms to OpenVMS software conventions for device drivers and command language interfaces.

#### Devices Supported

Dissimilar disk devices may be used as members of the same RAID array. When dissimilar disks are used, the capacity of the RAID array is a multiple of the size of the smallest disk.

**RAID0** Member devices of a RAID0 array may also be a shadow set supported by OpenVMS Volume Shadowing Phase II. See Chapter 6 for more information.

#### VMScluster Configuration Support

VMScluster configurations are fully supported. All RAID arrays may be accessed from any VMScluster node on which HP RAID Software for OpenVMS is running.

The HP RAID Software supports OpenVMS VMScluster configurations through a fully distributed model. This distributed model has the following constraints:

- The HP RAID Software must be installed, licensed, and loaded on all nodes that access RAID arrays. A RAID virtual device is only available to nodes that have the HP RAID Software installed (a RAID virtual device cannot be MSCP served).
- A RAID array bound on one node will be automatically bound on other nodes in the VMScluster configuration where the software is installed, licensed, and loaded.

- A RAID array unbound on one node is automatically unbound on all nodes in the VMScluster configuration where the HP RAID Software is installed, licensed, and loaded. An exception is the RAID SHUTDOWN command which will unbind all RAID arrays on only the local node in preparation for a complete system shutdown.
- HP RAID Software started on a new node causes all RAID arrays bound on other nodes to be bound on the new node.
- RAID array members may be located at any point in a VMScluster configuration (that is, they may be either local, MSCP served, or VMScluster accessible).

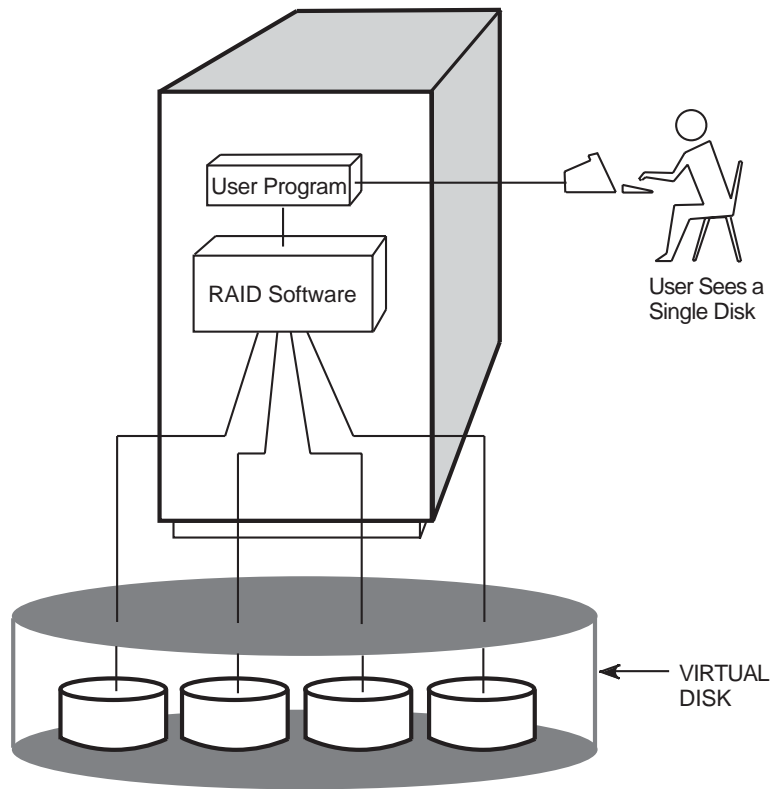
## 2.2 RAID Arrays

A **RAID array** is a collection of two or more disks called RAID array **members**. This set of disks is viewed by the user, the file system, and any software application as a single virtual device, as shown in Figure 2–1. Both RAID0 and RAID5 distribute data across the array members, but RAID5 also distributes parity information.

With RAID arrays, you can perform the following operations:

- Create or dissolve arrays from physical disks
- Remove or replace members
- Associate a spareset with an array (RAID5 only)

Figure 2–1 A RAID Array of Disks Appear as a Single Virtual Device



CXO-3700B-RA

### 2.2.1 Member Structure

Every array member disk device can itself be considered an array of logical blocks of 512 bytes. Each logical block has a **logical block number**, or **LBN**.

When a RAID array is created, the HP RAID Software divides the blocks of each member into four separate collections of blocks. These collections are as follow:

- The member volume file system
- The user data logical blocks
- The parity information (RAID5 only)
- Additional control information written to the media by the HP RAID Software

### 2.2.2 RAID Virtual Device Definition

The **RAID virtual device** is a logical unit through which RAID software accesses the array of physical disks in a way that is invisible to the user. The virtual device appears to the user as a physical disk that has an OpenVMS device name of the form **DPAnnnn**: (where *nnnn* is a number from 1 to 9999) to represent the virtual device.

### 2.2.3 Chunks

The blocks of a RAID virtual device are mapped onto the member disks in units called chunks. A **chunk** is a group of consecutive blocks placed on a single RAID array member. The blocks within a chunk on the virtual device map to consecutive logical blocks on a single RAID array member. Consecutive chunks are placed on the RAID array members in a round-robin fashion.

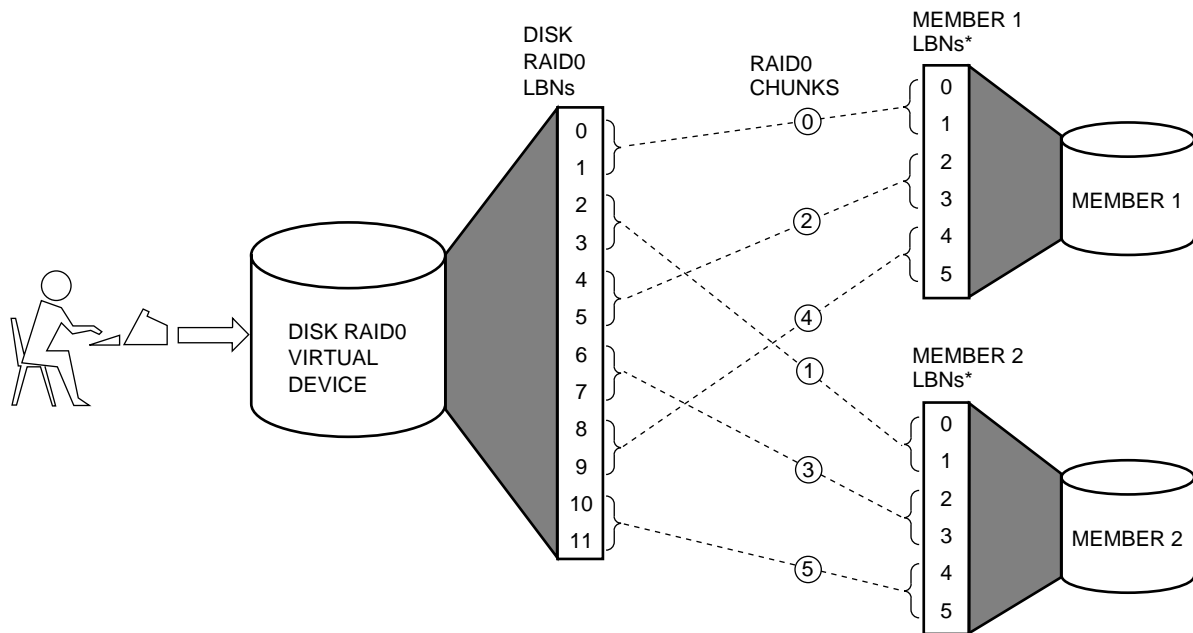
### 2.2.4 Parity Chunks

In a RAID5 array, **parity chunks** are used to regenerate missing information. A parity chunk is generated from a bit-by-bit **exclusive OR** of the contents of the corresponding chunks on all the other array members. By using the parity chunk information in combination with other bad block information derived from error correction at the end of each sector, it is possible for the contents of any chunk of data on any one of disk in the array to be regenerated from the contents of the corresponding chunks on the remaining disks in the array.

### 2.2.5 RAID0 User Data Mapping

RAID0, sometimes referred to as striping, works by breaking down application I/Os and distributing, or striping, them across a RAID array. The members then process the I/O requests concurrently. Figure 2–2 shows the mapping of chunks on a two-member disk array.

Figure 2–2 RAID0 Chunk Mapping



THESE LBNS DO NOT TAKE INTO ACCOUNT THE LOCATION OF THE MEMBER VOLUME FILE SYSTEM.

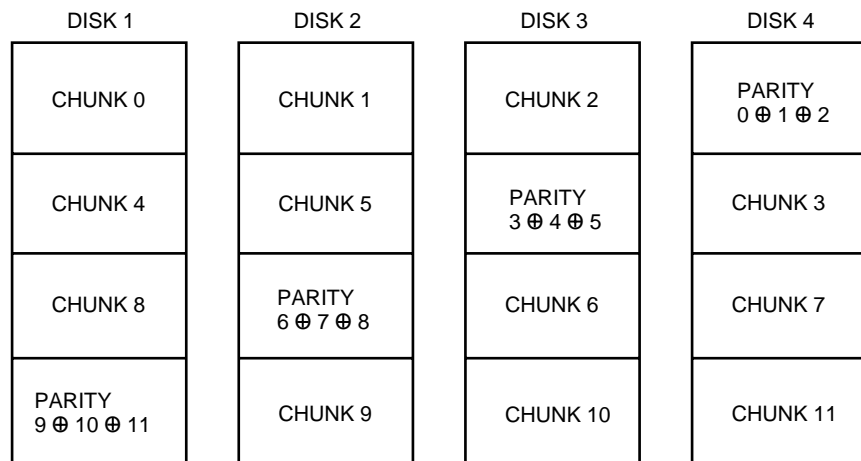
CXO-4064A-MC

## 2.2.6 RAID5 User Data Mapping

RAID5 stripes the user data across member disks just as RAID0 does. However, RAID5 also uses parity information for data checking and correction. This parity information is also striped across array members for data recovery purposes.

The RAID5 user data mapping algorithm is different than the one used for RAID0. The RAID5 mapping algorithm is chosen for maximum sequential data transfer speed. The distribution of data chunks and parity chunks in a RAID5 array is shown in Figure 2-3.

Figure 2-3 RAID5 Chunk Mapping



CXO-4037B-MC

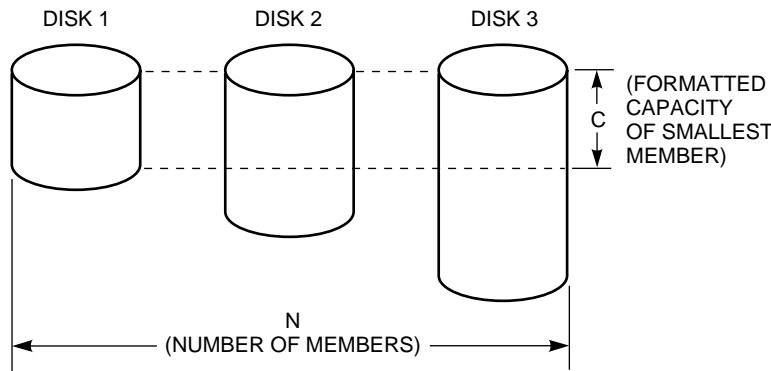
## 2.2.7 Size of the Array

The size of the RAID array is defined by the number of members. For a RAID5 array, the minimum size is three member devices. For a RAID0 array, the minimum size is one member. The maximum size is specified in the software release notes. Figure 2-1 is an example of a four-member array.

## 2.2.8 Capacity of the Array

The RAID **array capacity** is the number of logical blocks in the array that are available to the user. The array capacity is a multiple of the size of its smallest member. In Figure 2-4, “C” indicates the RAID array formatted capacity for “N” number of members. Note that the smallest member determines the RAID array capacity.

**Figure 2–4 Capacity of a RAID Array**



CXO-4036A-MC

## 2.2.9 Array Capacity Calculation

A RAID array consisting of, for example, six members appears to the OpenVMS operating system as a single array of logical blocks. In a RAID5 array, the parity information is approximately equal to one member and is distributed across the six members. Taking parity into account, the amount of data in a six member RAID5 array is approximately five times larger than the data capacity of the smallest member and is distributed across the six members of the RAID array. In a RAID0 array, because there is no parity, the data capacity is approximately six times larger than the data capacity of the smallest member.

The RAID array capacity depends upon the number and type of disk drives in the RAID array. The following formula may be used to determine your approximate total RAID array capacity. The example is calculated for a six member array:

$$\text{Capacity of RAID5 array} \cong (N - 1) \times C \times 0.99 \cong 5 \times 1,953,300 \times 0.99 \cong 9,766,500 \text{ blocks}$$

$$\text{Capacity of RAID0 array} \cong N \times C \times 0.99 \cong 6 \times 1,953,300 \times 0.99 \cong 11,602,602 \text{ blocks}$$

where:

*N* is the number of members (6 for this example)

*C* is the formatted capacity of smallest member in blocks

---

### Note

---

The previous example assumes a RAID array consisting of five RA92 disk drives and one RA72 disk drive. Because the RA72 disk drive has a smaller capacity than an RA92 disk drive, the capacity of the smaller drive is used in the calculation. The SHOW DEVICE/FULL command was used to find the device characteristics of the RA72 disk drive. It indicates that an RA72 disk drive has a total block size of 1,953,300.

The 0.99 factor used in the calculations is an approximation to account for the small amount of on-disk control information used by the HP RAID product.

---



## 2.3 The Member Volume File System Definition

The Member Volume File System is an array of user data LBNs that you can do I/O transfers to and mount and dismount.

### Contents

RAID array initialization automatically creates a minimal OpenVMS Files-11 On-Disk Structure Level 2 (ODS-2) file system that contains the required root directory files and reserved HP RAID files. This file system is created on each RAID array member.

### Modifying Files in the File System

Although the member volume file system can be mounted, its use is reserved for RAID commands.

---

#### CAUTION

---

Do not use utilities to modify or move files in the overhead file system. Specifically, use of any file-moving software on the overhead file system may result in corrupted data.

---

## 2.4 Additional Control Information

HP RAID Software writes additional information to the media. It describes the following:

- How many members exist in the RAID array and their names.
- The state of the RAID array

## 2.5 RAID Spares and Sparesets RAID5

This section describes spares and sparesets and their use in RAID arrays. The following topics are included in this section:

- Definition of a Spare
- Definition of a Spareset
- Do I Need a Spareset?
- Incorporating a Spare into a Spareset
- Associating Sparesets with RAID Arrays
- Limitations on Associating Sparesets with RAID Arrays

### 2.5.1 Definition of a Spare

A **spare** is a physical disk drive that has been initialized to serve as a replacement disk for RAID array members.

### 2.5.2 Definition of a Spareset

A **spareset** is a pool of spares that can be used as both an automatic replacement and a manual replacement for RAID array members that have been removed. A **populated spareset** is a spareset with a least one member.

### 2.5.3 Do I Need a Spareset?

Having a spareset that is associated with a RAID array ensures automatic replacement in the event of a failing or failed disk. The mean-time-to-repair (MTTR) is reduced and data availability and reliability are increased. When a disk fails, the software immediately incorporates a spareset member into the RAID array and reconstructs the information from the failed disk onto the spare. Because a single spareset may be associated with many RAID arrays, you need only one or two additional drives to ensure that your RAID array continues to operate with redundancy in the event of a disk failure.

Spares are tested at fixed intervals by the HP RAID Software to ensure that they are still functional. Spares are automatically removed from the spareset if they fail.

For more information on using sparesets, see Section 9.6.

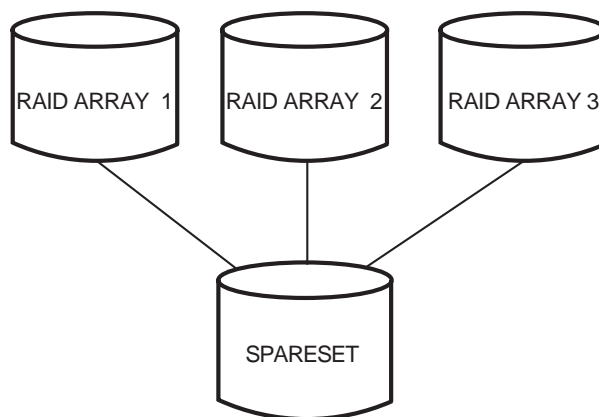
### 2.5.4 Incorporating a Spare into a Spareset

Like any physical disk, a spare has a characteristic size, which is a function of disk size and geometry. In order to incorporate a spare into a spareset, the characteristic size of the *spare* must be greater than, or equal to, the characteristic size of the *spareset*.

### 2.5.5 Associating Sparesets with RAID Arrays

Sparesets may be associated with one or more RAID arrays, as shown in Figure 2-5.

Figure 2-5 Different RAID Arrays May Share the Same Spareset



CXO-3744A-RA

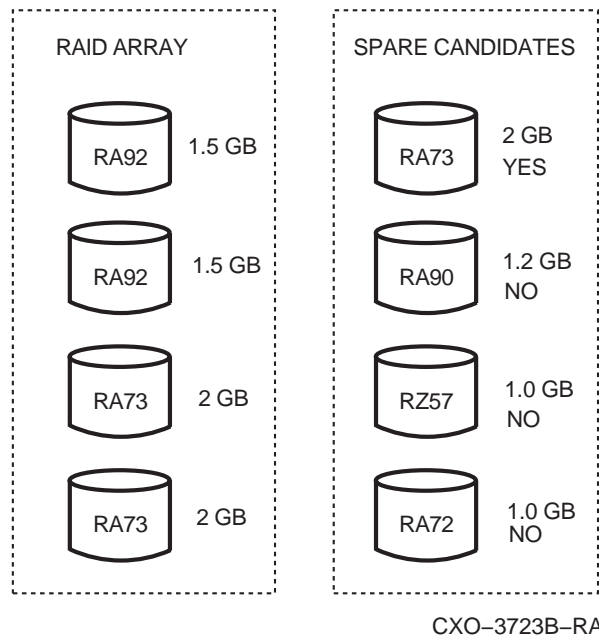
Different RAID arrays may share the same spareset, but each RAID array may only have one associated spareset.

## 2.5.6 Limitations on Associating Sparesets with RAID Arrays

The characteristic size of the spareset is determined at the time the spareset is created (bound) and is either what you specify or the characteristic size of the smallest spare. The characteristic size of the spareset must be greater than or equal to the characteristic size of the RAID array for an association between the two to work. See Chapter 3 for more information on characteristic size.

Figure 2-6 shows an example of acceptable and unacceptable spares for a RAID array made up of RA92 and RA73 disk drives. The RAID array on the left in Figure 2-6 has been initialized with a characteristic size equal to the characteristic size of an RA92 disk drive. Of the disks on the right of the figure, only the RA73 disk drive is equal to or larger than the disks on the left. Thus, it is the only spare candidate that can be used by the RAID array.

Figure 2-6 Spare Candidates





---

## Preparing to Use HP RAID Software for OpenVMS

After you have installed HP RAID Software for OpenVMS, creating a RAID array is the first step in using the software. This chapter discusses decisions you need to make before you can create a RAID array. Options for each choice are provided in Figure 3-1 (included at the end of this chapter). The default values that will be assigned if you do not enter your own choice are also listed in Figure 3-1 (included at the end of this chapter). Other defaults are discussed in the software release notes. Figure 3-2 supplies a sample RAID array member list form for you to plan your member device names.

Table 3-1 provides a list of topics in this chapter.

**Table 3-1 Topics in This Chapter**

Subject	Section
RAID Array Planning Checklist	3.1
RAID Array Restrictions	3.2
User-Settable Attributes	3.3

### 3.1 RAID Array Planning Checklist

Figure 3-1 is a checklist of decisions to make before you create a RAID array.

To prepare for the creation of your RAID array, follow the steps shown in Table 3-2.

**Table 3–2 Planning Checklist**

Step	Activity
1	Review the user-settable attributes in Section 3.3 so you know what decisions you need to make.
2	Review the planning checklist in Figure 3–1.
3	Look at the default values or list of choices for each attribute as shown in Figure 3–1.
4	Enter your choice in the column labeled <i>Your Choice</i> .
5	Go on to Chapter 4 to actually create your RAID array.

## 3.2 RAID Array Restrictions

A RAID array may not be used as either of the following:

- A bootable system disk
- A VMScluster quorum disk

In each case, low-level operating system software performs these functions. This software is unaware of HP RAID Software for OpenVMS.

## 3.3 User-Settable Attributes

User-settable attributes come in two categories:

- Those that affect every array on the CPU
- Those that only affect individual arrays.

When creating your RAID array, you may choose from among the attributes presented in the following sections or allow the default value to be used.

### 3.3.1 Attributes That Affect Every Array on the CPU

#### **/PAGESMAX=pages**

This is the maximum number of pages of memory taken from the OpenVMS free-list that may be utilized by the HP RAID Software for OpenVMS on the issuing node at any given time. At node startup, the default value is /NOPAGESMAX. NOPAGESMAX allows the HP RAID Software to use available resources to the extent needed. HP recommends the default value for most situations.

#### **/POOLMAX=bytes**

This is the maximum number of bytes of the non-paged pool utilized by the HP RAID Software for OpenVMS on the issuing node at any given time. NOPOOLMAX allows the HP RAID Software to use available resources to the extent needed. At node startup, the default value is /NOPOOLMAX. HP recommends the default value for most situations.

### **/NORECONSTRUCT** RAID5

This qualifier is issued on a per-node basis and only applies to the node on which it is issued. The /NORECONSTRUCT qualifier will not allow the node on which it was issued to initiate any new reconstructs. In addition, reconstructs in progress on this node will stop and continue on another available node. The default at node startup is /RECONSTRUCT.

## **3.3.2 Attributes That Affect Only Individual Arrays**

### **/ASSOCIATED\_SPARESET** RAID5

Associating a populated spareset ensures that any failed disks in the RAID array will be replaced automatically by the HP RAID Software. If you want to have a spareset associated with your RAID array, you can indicate this at the time you bind your RAID array. You can also add an associated spareset after binding the RAID array by using the RAID MODIFY command. For information on sparesets, refer to Section 2.5.

### **/CHARACTERISTIC\_SIZE** RAID5

This number is a function of the size and geometry of the smallest disk in the RAID array. Normally, the characteristic size is calculated for you. If you plan to use replacement disks or to have an associated spareset that contains smaller disks than those in your RAID array, your characteristic size will need to match the smallest replacement disk or smallest spare in the spareset. The spareset characteristic size cannot be less than the RAID array characteristic size. The characteristic size may vary from 64 blocks to the total available blocks on the smallest array member.

You can learn what your member characteristic size is by initializing the device as a spare, and binding it into a spareset using the following RAID commands:

```
$ RAID INITIALIZE/SPARE DUA1
$ RAID BIND/SPARESET TEST_SPARE DUA1
$ RAID SHOW/SPARESET TEST_SPARE
$ RAID UNBIND/SPARESET TEST_SPARE
```

When you issue these commands, the software calculates the characteristic size for you and displays it when you use the RAID SHOW/SPARESET command.

Changing the characteristic size requires re-creating the RAID array.

### **/CHUNK\_SIZE**

This is the number of logical blocks in each chunk. Chunk size will affect the I/O performance of the RAID array. See Chapter 9 for additional information.

The HP RAID Software provides a default chunk size if you do not specify the /CHUNK\_SIZE qualifier when you use the RAID INITIALIZE command. This default value works well for most applications and is found in the software release notes.

Occasionally, users with atypical I/O characteristics or other concerns may want to specify the chunk size to improve performance. The chunk size range is from 1 to 65535.

Changing the chunk size requires re-creating the RAID array.

### **/FILES-11**

Obsolete Qualifier.

### **/NOIO\_TIMEOUT** RAID5

If the `/NOIO_TIMEOUT` qualifier is specified, the RAID driver avoids the time overhead associated with providing data availability in the event that a member disk drive does not respond in a timely fashion. The overhead avoided involves extra data copying that consumes CPU cycles and memory. Turning off `IO_TIMEOUT` results in better performance, but may result in delayed application execution whenever an application I/O request being processed by RAID software results in an I/O request to a member device and that member device does not promptly complete the request. This situation is uncommon, but should it occur, the delay could be measured in hours. Users may utilize this qualifier to trade reduced I/O request overhead for reduced data availability. The `RAID SHOW /FULL` command will display the state of this setting.

The default for this qualifier is to perform the buffer copies. Also, the `/NOTIMEOUT` qualifier overrides the `IO_TIMEOUT` qualifier. In other words, if you turn off member timeouts, buffer copies are also effectively disabled as a result regardless of the `/NOIO_TIMEOUT` specified (or defaulted). A message will be displayed when this override occurs.

### **Number of Members**

This is the number of disks in your RAID array. You must have at least one member in a RAID0 array, and at least three members in a RAID5 array. The maximum number of RAID array members allowed is specified in the release notes. Changing the number of members requires re-creating the RAID array. There is no default value.

### **RAID Array Membership**

You need to know which devices you will use as members of your RAID array.

### **RAID Array Name**

This is the unique identifier you assign to your RAID array. The name may be a 1- to 32-character string containing alphanumeric characters, the dollar sign (\$), and underscore (\_). This name is used in many DCL commands to identify different RAID arrays. There is no default value.

### **/RAID\_LEVEL**

This qualifier is required when a RAID array is initialized to indicate whether to initialize the array as a RAID0 or RAID5 array.

### **/TIMEOUT**

The timeout qualifier is used with the `RAID INITIALIZE` command and can be modified with the `RAID BIND` or `RAID MODIFY` commands.

The timeout qualifier specifies the number of seconds that HP RAID Software will wait for a member I/O to complete before declaring the device unavailable and using RAID algorithms to complete the I/O. It is preserved across binding and unbinding of the RAID array. The timeout may range from 1 to 99999 seconds.

For more information on the timeout qualifier, see Section 7.6.2.

### **Virtual Device Name**

The virtual device name specifies the DPA unit that will provide access to the RAID array. This must be in the form `DPA $nnnn$` , where  $nnnn$  is any number from 1 to 9999. This may not be the same as the virtual device name of another currently bound RAID array. There is no default value.



**Note**

DPA0 is not a valid drive device name. DPA0 is reserved for the internal use of the RAID software.

**Figure 3–1 RAID Planning Checklist**

RAID ARRAY ATTRIBUTES		
DESCRIPTION	OPTION OR DEFAULT VALUE	YOUR CHOICE
RAID ARRAY NAME	1- to 32-character string	
NUMBER OF MEMBERS	Minimum of three Maximum as specified in Release Notes	
CHUNK SIZE	Default recommended See Release Notes for Value	
ASSOCIATED SPARESET? (Y/N)	If yes, a 1- to 32- character string	
CHARACTERISTIC SIZE	Default: characteristic size of smallest RAID ARRAY member	
MEMBER TIMEOUT	Default recommended See Release Notes for Value	
VIRTUAL DEVICE NAME	DPA $n$ $n$ $n$ $n$ : where $n$ $n$ $n$ $n$ = a number between 1 and 9999	
IO TIMEOUT	Default: / IO_TIMEOUT	
RAID LEVEL	Choose 0 or 5	

CXO-4065A-MC

**Figure 3–2 RAID Membership Checklist**

MEMBER	YOUR CHOICE; RAID ARRAY MEMBER DEVICE NAME
MEMBER #1	
MEMBER #2	
MEMBER #3	
MEMBER #4	
MEMBER #5	
MEMBER #6	
MEMBER #7	
MEMBER #8	

CXO-4066A-MC

## Creating and Managing RAID Arrays

This chapter contains a series of procedures on how to define, dissolve, alter, and manage your RAID array and its associated virtual device. It also discusses the use of spares and sparesets. The information in this chapter applies to both RAID0 and RAID5 unless indicated otherwise. Table 4–1 provides a list of topics in this chapter.

**Table 4–1 Topics in This Chapter**

Subject	Section
Creating RAID Arrays	4.1
Undefining RAID Arrays	4.2
Changing the Characteristics of Your RAID Array	4.3
Sparesets	4.4
Maintaining a RAID Array	4.5
Obtaining Information About RAID Arrays	4.6

### 4.1 Creating RAID Arrays

Once you have decided what attributes you want your RAID array to have, creating a RAID array is the next step in using HP RAID Software for OpenVMS. This section provides the commands necessary to create a RAID array. Perform the steps shown in Table 4–2 when creating your RAID array. These steps are detailed in sections following the table.

**Table 4–2 Steps to Creating a RAID Array**

Step	Do This:	By Using the:
<b>Unique RAID Commands</b>		
1	Define a RAID array	RAID INITIALIZE command.
2	Associate the RAID array with a virtual device	RAID BIND command.
<b>Common OpenVMS Commands</b>		
3	Place a Files–11 ODS–2 file system on the virtual device if you are using the Files–11 file system	INITIALIZE command.

(continued on next page)

**Table 4–2 (Cont.) Steps to Creating a RAID Array**

Common OpenVMS Commands		
4	Mount the virtual device	MOUNT command.

All of the steps in Table 4–2 are done through the DIGITAL Command Language (DCL). After Steps 1 and 2, you will have a RAID array accessible through a virtual device. After Steps 3 and 4, the virtual device will be available for file access.

**Note**

Once initialized, only Steps 2 and 4 are necessary for regaining access to a RAID array when your RAID array becomes unbound (for example, at a reboot).

### 4.1.1 Step 1: Defining a RAID Array

Defining a RAID array using the RAID INITIALIZE command prepares the RAID array members.

**CAUTION**

You must do a backup<sup>1</sup> of the proposed member disks prior to entering the RAID INITIALIZE command. Using the RAID INITIALIZE command will destroy all data on the RAID5 array and make the user data indeterminate on a RAID0 array.

#### Time Required for Initialization

**RAID5** RAID5 array initialization using **Small Computer System Interface (SCSI)** devices as array members can take a significant amount of time to complete. The RAID INITIALIZE command initiates an OpenVMS INITIALIZE /ERASE operation on each SCSI device. Depending on the SCSI device type, adapter, and operating characteristics, the OpenVMS INITIALIZE operation can take from 20 to 90 minutes per disk to complete. If the SCSI devices are on the same bus, these operations cannot be processed in parallel. As a result, initialization of a SCSI RAID array can take several hours to complete, depending on the number of RAID array members to be initialized.

**RAID5** MSCP devices (RA, RF, and RD) are also initialized by the RAID INITIALIZE command in the same manner. However, because I/O to these device types can usually be done in parallel and the aggregate bus bandwidth is greater, the duration of a RAID INITIALIZE command on MSCP devices is usually much less than on SCSI devices.

**RAID0** RAID0 initialization does not erase the disks, so it will complete very quickly. However, the state of the user data previously on the disk will be indeterminate.

<sup>1</sup> See the *OpenVMS Backup Manual* for more information on performing backup operations.

### Example

Table 4–3 provides an example of the RAID INITIALIZE command.

**Table 4–3 Example: RAID INITIALIZE/RAID\_LEVEL=5**

Command	Result
RAID INITIALIZE/RAID_LEVEL=5 - ENGINEERING_PAYROLL - DUA1:,DUA2:,DUA3:	Prepares the RAID array members DUA1, DUA2, and DUA3; associates the name ENGINEERING_PAYROLL with the RAID array; writes data describing the RAID array onto each member.

### Result

The RAID INITIALIZE command produces the following results:

- Creates the member volume file system and necessary HP RAID files.
- Defines the mapping between the RAID array and the member logical block numbers, and records this information on each member.

#### Note

RAID array names are not case sensitive. Regardless of how you enter the name of your RAID array, the HP RAID software will convert the name to all uppercase letters. Thus, entering the name Engineering\_Payroll will create a name viewed by the software as ENGINEERING\_PAYROLL.

## 4.1.2 Step 2: Associating a RAID Array with a Virtual Device

The next step in creating a RAID array is associating the array with a virtual device, which is called binding. You do this with the RAID BIND command.

### The DPA<sub>nnnn</sub>: Unit

When you enter a RAID BIND command, you specify the DPA unit that will provide access to the RAID array. This DPA unit provides a user-specifiable virtual device name to the virtual device, which is needed for OpenVMS commands.

### Changes in RAID Array Membership

**RAID5,RAID0** If you do not specify all the members of a RAID5 or RAID0 array with the RAID BIND command, the HP RAID software will try to find the remaining members.

**RAID5** If the software can find all but one of the RAID array members, it will bind the set as a **reduced** RAID array, unless you use the /NOREMOVAL\_ALLOWED qualifier with the RAID BIND command. For information on reduced RAID arrays, see Sections 4.3.2 and 7.1.2. For example, a RAID array with three members may be bound with only two members if one member cannot be found.

### Example

Table 4–4 provides an example of the RAID BIND command.

**Table 4–4 Example: RAID BIND Command**

Command	Result
RAID BIND ENGINEERING_PAYROLL - DUA1:,DUA2:,DUA3: DPA1:	Binds the RAID array ENGINEERING_ PAYROLL consisting of members DUA1, DUA2, and DUA3 to a virtual device with the name DPA1:.

### Result

The RAID BIND command produces the following results:

- Forms a consistent RAID array using the information written on each member by the RAID INITIALIZE command
- Creates a virtual device that provides access to the RAID array
- Creates a *DPA<sub>nnnn</sub>*: virtual device name with which to access the virtual device
- For VMScluster systems, propagates the bind to all nodes running the HP RAID Software for OpenVMS software (see Section 4.1.6.)

### 4.1.3 Step 3: Placing a File System on the RAID Virtual Device

After entering the RAID INITIALIZE and RAID BIND commands, you have an accessible but uninitialized virtual disk. You must use the OpenVMS INITIALIZE command to initialize the virtual disk to use it as a Files–11 device. Refer to the *OpenVMS DCL Dictionary* for more information on this command.

Table 4–5 provides an example of the INITIALIZE command.

**Table 4–5 Example: Placing a File System on the RAID Array**

Command	Result
INITIALIZE DPA1: - PAYROLL93	Places a Files–11 file system on the RAID virtual device. The identifier, PAYROLL93, is the volume label for the virtual volume.

#### 4.1.4 Step 4: Mounting an OpenVMS File System on the RAID Virtual Device

To make the RAID virtual device available for use, you must use the OpenVMS MOUNT command. All the mount qualifiers are valid for RAID virtual devices.

Table 4–6 provides an example of the MOUNT command.

**Table 4–6 Example: Mounting an OpenVMS File System on the RAID Array**

Command	Result
MOUNT/SYSTEM DPA1: - PAYROLL93	Makes the virtual device DPA1: available for use.§

§Refer to the *HP OpenVMS System Management Utilities Reference Manual* for more information on the MOUNT command.

#### 4.1.5 Example of RAID Array Creation

Example 4–1 shows the interactive series of commands used to create two RAID arrays from initialization through mounting.

##### Example 4–1 Creating RAID Arrays

```
$ !+
$ ! Create a four member RAID5 array called PAYROLL.
$ !-
$ RAID INITIALIZE/RAID_LEVEL=5 PAYROLL $14$DIA3:,$3$DUA41:,$3$DUA54:,$14$DKA100:
  INIT will destroy existing data, do you want to continue [N]? Y
%RAID-I-PRTCREATED, Array partition 1 created: 1793992 blocks
$ !+
$ ! Create a three member RAID0 array called ACCOUNTS.
$ !-
$ RAID INITIALIZE/RAID_LEVEL=0 ACCOUNTS $14$DKA200:,$14$DKA300:,$14$DKA400:
  INIT will destroy existing data, do you want to continue [N]? Y
%RAID-I-PRTCREATED, Array partition 1 created: 6140152 blocks
$ !+
$ ! The member volumes have now been initialized and the
$ ! member data files have been created.
$ !-
$ !+
$ ! Bind the RAID arrays. Assign a DPAnnnn
$ ! number with which to access the virtual devices.
$ !-
```

(continued on next page)

### Example 4–1 (Cont.) Creating RAID Arrays

```
$ RAID BIND PAYROLL $14$DIA3:,$3$DUA41:,$3$DUA54:,$14$DKA100: DPA9998:
%RAID-I-ISBOUND, unit _$14$DIA3: is bound as a member of RAID array PAYROLL
%RAID-I-ISBOUND, unit _$3$DUA41: is bound as a member of RAID array PAYROLL
%RAID-I-ISBOUND, unit _$3$DUA54: is bound as a member of RAID array PAYROLL
%RAID-I-ISBOUND, unit _$14$DKA100: is bound as a member of RAID array PAYROLL
%RAID-I-VUCREATE, virtual unit DPA9998: was created for partition 1 on PAYROLL
$ !
$ RAID BIND ACCOUNTS $14$DKA200:,$14$DKA300:,$14$DKA400: DPA9999:
%RAID-I-ISBOUND, unit _$14$DKA200: is bound as a member of RAID array ACCOUNTS
%RAID-I-ISBOUND, unit _$14$DKA300: is bound as a member of RAID array ACCOUNTS
%RAID-I-ISBOUND, unit _$14$DKA400: is bound as a member of RAID array ACCOUNTS
%RAID-I-VUCREATE, virtual unit DPA9999: was created for partition 1 on ACCOUNTS
$ !+
$ ! The virtual devices have now been created and may be accessed
$ ! like any other device. For example:
$ !-
$ INIT DPA9998: PAYROLL
$ INIT DPA9999: ACCOUNTS
$ MOUNT DPA9998: PAYROLL
  %MOUNT-I-MOUNTED, PAYROLL      mounted on _DPA9998:
$ MOUNT DPA9999: ACCOUNTS
  %MOUNT-I-MOUNTED, ACCOUNTS    mounted on _DPA9999:
$ !
$ DIR DPA9998:[000000]

Directory DPA9998:[000000]

000000.DIR;1          1  25-MAR-2001  14:40:12.26
BACKUP.SYS;1        0  25-MAR-2001  14:40:12.26
BADBLK.SYS;1        0  25-MAR-2001  14:40:12.26
BADLOG.SYS;1        0  25-MAR-2001  14:40:12.26
BITMAP.SYS;1       224 25-MAR-2001  14:40:12.26
CONTIN.SYS;1        0  25-MAR-2001  14:40:12.26
CORIMG.SYS;1        0  25-MAR-2001  14:40:12.26
INDEXF.SYS;1       114 25-MAR-2001  14:40:12.26
VOLSET.SYS;1        0  25-MAR-2001  14:40:12.26

Total of 9 files, 339 blocks.
$ !-
$ DIR DPA9999:[000000]

Directory DPA9999:[000000]

000000.DIR;1          1  25-MAR-2001  14:52:06.35
BACKUP.SYS;1        0  25-MAR-2001  14:52:06.35
BADBLK.SYS;1        0  25-MAR-2001  14:52:06.35
BADLOG.SYS;1        0  25-MAR-2001  14:52:06.35
BITMAP.SYS;1        3  25-MAR-2001  14:52:06.35
CONTIN.SYS;1        0  25-MAR-2001  14:52:06.35
CORIMG.SYS;1        0  25-MAR-2001  14:52:06.35
INDEXF.SYS;1       14  25-MAR-2001  14:52:06.35
VOLSET.SYS;1        0  25-MAR-2001  14:52:06.35

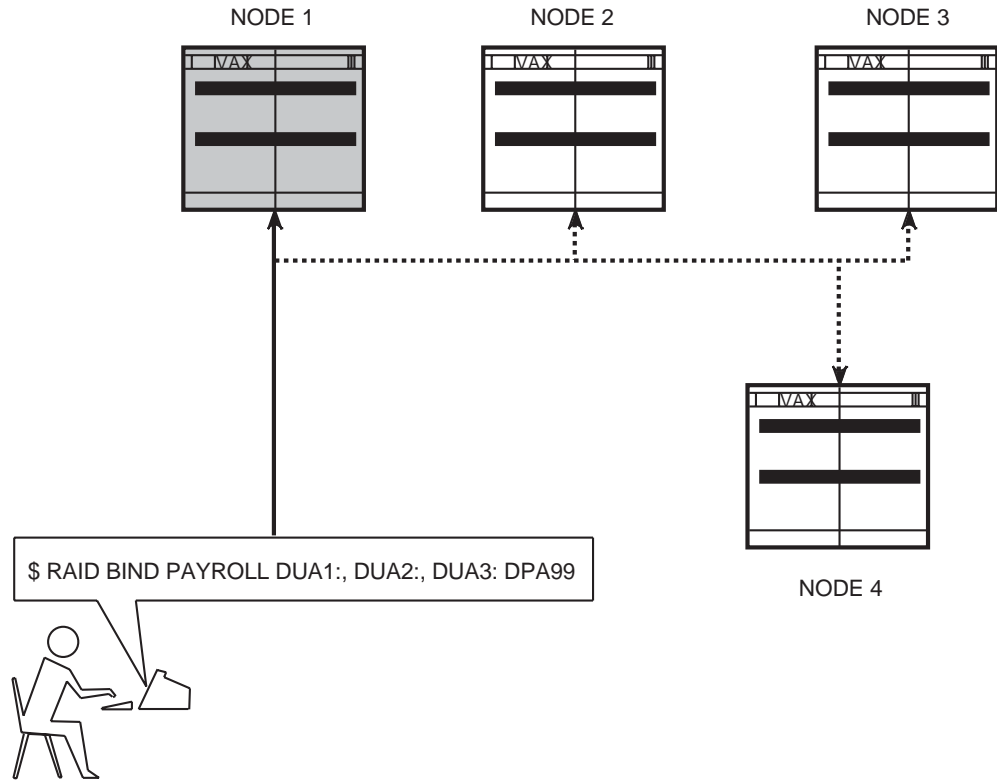
Total of 9 files, 18 blocks.
```



### 4.1.6 Creating a RAID Array on a VMSccluster System

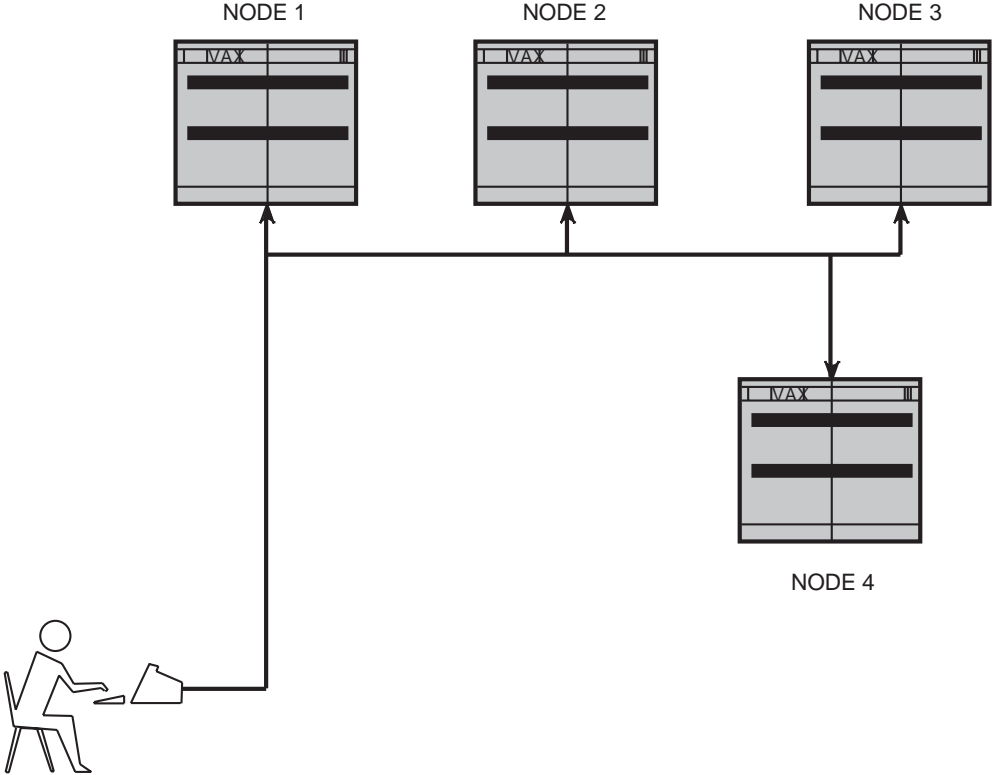
You need only enter the RAID BIND command on one node of a VMSccluster system for each array. The BIND command will then propagate to the other nodes on the cluster as shown in Figures 4-1 and 4-2.

Figure 4-1 RAID BIND Command Is Entered at One Node



CXO-3725B-RA

Figure 4-2 BIND Command Propagates to the Other Nodes



CXO-3726A-RA

Use the procedure in Table 4–7 to create a RAID array on a VMSccluster system.

**Table 4–7 Steps to Creating a RAID Array on a VMSccluster System**

Step	Issuing Node	Command	Result
1	Any cluster node	RAID INITIALIZE/RAID_LEVEL=5 PAYROLL - DUA1:, DUA2:, DUA3:	Initializes RAID array members DUA1, DUA2, and DUA3; deletes all previous data on the RAID array members.
2	Any cluster node	RAID BIND PAYROLL - DUA1:,DUA2:,DUA3: DPA99:	Associates RAID array PAYROLL with a virtual device on all nodes in the cluster. The virtual device can be accessed through DPA99:.
3	Any cluster node	INITIALIZE DPA99: - PAYROLL93	Initializes a Files–11 file system on the RAID virtual device DPA99: and with the volume label PAYROLL93.
4	Any cluster node	MOUNT/SYSTEM DPA99: - PAYROLL93	Makes the Files–11 file system of the virtual device available on the issuing node. Using the /CLUSTER qualifier will make the file system available on all nodes.

Steps 1 and 3 are normally entered once in the life of a RAID array. Steps 2 and 4 are normally placed in the SYSSMANAGER:RAIDSSYSTARTUP.COM file for each VMSccluster node that will access the RAID virtual device.

## 4.2 undefining RAID Arrays

This section provides information on how to disassociate and reassociate your RAID array and the virtual device.

### 4.2.1 Disassociating the RAID Array from the Virtual Device

When you no longer need access to a RAID array, you can use the RAID UNBIND command. The RAID UNBIND command disassociates the virtual device from the RAID array. As with any physical device, you must dismount the RAID array virtual device before you can unbind the RAID array.

Table 4–8 provides an example of the RAID UNBIND command.

**Table 4–8 Example: RAID UNBIND Command**

Command	Result
DISMOUNT DPA99:	Dismounts the virtual device DPA99:. Does not disassociate the RAID array from the virtual device.
RAID UNBIND PAYROLL	Places the virtual device associated with the RAID array named PAYROLL off line.

---

**Note**

---

The UNBIND command requires the RAID array name and not the virtual device name.

---

The RAID UNBIND command does not affect the data stored on the members, but dismounts the member.

The RAID virtual device does not disappear; it is taken off line and is unusable until a RAID BIND command again associates it with a RAID array.

If there has been an automatic replacement of a RAID array member since the last use of the RAID BIND command, the software will use membership recorded on current members to determine true membership at the next use of the RAID BIND command.

#### **4.2.2 Reassociating the RAID Array with the Virtual Device**

Rebinding the unbound RAID array using the RAID BIND command causes the RAID array and RAID array user data to become available again. All data and characteristics except spareset associations and the virtual device name are preserved across the UNBIND/BIND operations. The virtual device name of the rebound RAID array need not be the same as the original name.

#### **4.3 Changing the Characteristics of Your RAID Array**

Any changes other than those using the RAID MODIFY, RAID REMOVE, RAID REPLACE, or RAID BIND commands require reinitializing the RAID array with the new parameters.

### 4.3.1 The RAID MODIFY Command

The RAID MODIFY command allows you to modify the configuration or characteristics of your RAID array or spareset. The features you can alter with RAID MODIFY are as follow:

- The RAID array name
- The value of the /TIMEOUT qualifier [RAID5]
- The association of RAID arrays with sparesets [RAID5]

Table 4–9 provides an example of using the RAID MODIFY command to change the name of a RAID array.

**Table 4–9 Example: RAID MODIFY Command**

Command	Result
RAID MODIFY PAYROLL/ARRAY_NAME=DB_RAID	Specifies the new name DB_RAID for the RAID array previously named PAYROLL.

For more information on the RAID MODIFY command, see the RAID Commands Chapter.

### 4.3.2 Removing a RAID Array Member [RAID5]

You may want to remove a RAID array member for any of the following reasons:

- You want to perform preventative drive maintenance that requires taking the drive off line without unbinding.
- You see that a member is about to fail.
- You see the error rate going up drastically.

Removing a RAID array member may be initiated by one of the following:

- The RAID REMOVE command
- The HP RAID software error handling routines

For more information on removal of individual shadow disks in RAID0 arrays, refer to Chapter 6.

#### Reduced RAID Array

When you remove a RAID array member, the array enters a reduced state and is no longer redundant. When a member is removed from a RAID array, no further I/O requests will be issued to that member. Read requests for the now missing data must be accomplished through **regeneration**. Data that normally would have been written to the missing member will be used to recalculate parity to keep the RAID array consistent. Once a member is removed, it is no longer recognized by the HP RAID software and will not be included in future bind operations.

### Use of Spareset

Removing a member from a RAID array that has an associated populated spareset causes the RAID array to enter a **reconstructing mode**. The HP RAID software will use a spareset member to replace the removed RAID array member. You may avoid this automatic replacement by disassociating the spareset from the RAID array prior to using the RAID REMOVE command. To disassociate a spareset from the RAID array, use the RAID MODIFY command. Unbinding the spareset or RAID array also disassociates the spareset.

Removing a second member from a RAID array is not allowed. Table 4–10 provides an example of the RAID REMOVE command.

**Table 4–10 Example: RAID REMOVE Command**

Command	Result
RAID REMOVE PAYROLL DUA1:	Removes DUA1 from RAID array PAYROLL.

### 4.3.3 Replacing a Member in RAID Arrays RAID5

If there is a populated spareset associated with a RAID array, when a member is removed, replacing the member is done automatically.

If you do not have an associated populated spareset, you may replace the missing member manually with the RAID REPLACE command. Data and parity for the new member is reconstructed from the data and parity on the rest of the RAID array.

Table 4–11 provides an example of the RAID REPLACE command.

**Table 4–11 Example: RAID REPLACE Command**

Command	Result
RAID REPLACE PAYROLL SPARE1	Replaces the missing member of RAID array PAYROLL with a spare from spareset SPARE1.

For information on replacing or adding disks to a shadow set that is a member of a RAID0 array, see Chapter 6.

### 4.3.4 Learning about Your Array

Use the RAID SHOW command to obtain a list of RAID arrays, characteristics, membership, and other information. Refer to the RAID SHOW command examples in the RAID Commands Chapter.

## 4.4 Sparesets RAID5

As with RAID5 arrays, you can perform many operations on RAID5 spares and RAID5 sparesets, as discussed in the following sections.

#### 4.4.1 Initializing Spares RAID5

Like RAID array members, spares must be RAID initialized.

---

#### CAUTION

---

Using the RAID INITIALIZE/SPARE command overwrites most of the data on the disk. If you need this data, you must do a backup prior to entering the RAID INITIALIZE/SPARE command.

---

Table 4–12 provides an example of the RAID INITIALIZE/SPARE command.

**Table 4–12 Example: RAID INITIALIZE/SPARE Command**

Command	Result
RAID INITIALIZE/SPARE DUA10:, - DUA11:, DUA12:, DUA13:, DUA14:	Initializes the spares DUA10 through DUA14 so that they may be included in a spareset and used for replacement. Note that DUA10 through DUA14 are not ordered pairs. That is, they can be used and bound separately so that they can be included in separate sparesets even though they were initialized simultaneously.

#### 4.4.2 Associating Spares with a Spareset RAID5

A spareset is a collection of zero or more spares. A spareset with at least one spare is a **populated spareset**.

Once initialized, spares may be bound into a spareset with the RAID BIND/SPARESET command to make them available for automatic replacement in RAID arrays.

Table 4–13 provides an example of the RAID BIND/SPARESET command.

**Table 4–13 Example: RAID BIND/SPARESET Command**

Command	Result
RAID BIND/SPARESET SPARE1 - DUA10:,DUA11:	Defines two spares, DUA10 and DUA11, as members of a spareset called SPARE1.

#### 4.4.3 Disassociating Spares from a Spareset RAID5

The RAID UNBIND/SPARESET command dissolves the spareset, disassociating it from any RAID arrays and dismounting its spares.

Table 4–14 provides an example of the RAID UNBIND/SPARESET command.

**Table 4–14 Example: Unbinding Sparesets Command**

Command	Result
RAID UNBIND/SPARESET SPARE1	Dissolves the spareset SPARE1 and dismounts its spares.

#### 4.4.4 Associating a Spareset with a RAID Array RAID5

A spareset is associated with a RAID array by using the RAID BIND command with the /ASSOCIATED\_SPARESET qualifier. Different RAID arrays may share the same spareset, but each RAID array can have only one associated spareset. A spareset may also be associated with a RAID array using the RAID MODIFY command.

Table 4–15 provides an example of associating a spareset using the /ASSOCIATED\_SPARESET qualifier.

**Table 4–15 Example: Associating a Spareset with a RAID Array**

Command	Result
RAID INIT/RAID_LEVEL=5 RAID_A \$3SDUA20:, \$3SDUA21:, - \$3SDUA22:, \$3SDUA23: RAID BIND RAID_A - /ASSOCIATED_SPARESET=SPARE1 - \$3SDUA20:,\$3SDUA21:,\$3SDUA22:,\$3SDUA23: - DPA1334:	Makes accessible a RAID array called RAID_A and associates it with a spareset called SPARE1.
or	
RAID MODIFY/ASSOCIATED_SPARESET=SPARE1 - RAID_B	Associates spareset SPARE1 to an existing (bound) RAID array called RAID_B.

#### 4.4.5 Adding Spares to Sparesets RAID5

Use the RAID ADD/SPARESET command to add a spare to an existing spareset. The spare must not be mounted and must have been initialized by the RAID INITIALIZE/SPARE command. Spares with a characteristic size smaller than the spareset characteristic size cannot be added to the spareset. See Section 3.3 for information on characteristic size.

Table 4–16 provides an example of the RAID ADD/SPARESET command.

**Table 4–16 Example: Adding Spares to Sparesets Command**

Command	Result
RAID ADD/SPARESET SPARE1 DUA14:	Adds the spare DUA14 to the existing spareset called SPARE1

#### 4.4.6 Removing Spares from Sparesets RAID5

Use the RAID REMOVE/SPARE command to remove a spare from a spareset. Table 4–17 provides an example of the RAID REMOVE/SPARE command.

**Table 4–17 Example: Removing Spares from Sparesets Command**

Command	Result
RAID REMOVE/SPARE SPARE1 DUA14:	Removes the spare DUA14 from the spareset called SPARE1



#### 4.4.7 RAID5 Spare Set Management RAID5

OpenVMS Version 6.2 changes the OpenVMS device initialization algorithms. In some cases trying to re-add a removed disk into the same array on OpenVMS versions prior to Version 6.2 fails with:

```
%RAID-F-REPLACEFAIL, replacement with spare ... into array ... failed
-RAID-F-CHRSZTOOSMALL, unit ... size is too small for array ...
```

This happens only if the RAID5 array was initialized under OpenVMS Version 6.2 and the spare disk has been initialized under an OpenVMS version prior to Version 6.2.

Initialize the spare disk under OpenVMS Version 6.2 and add it back into the array.

#### 4.4.8 Learning about Your Spareset

Use the RAID SHOW/SPARESET command to learn the characteristics, membership, and other information about your spareset. Refer to the RAID SHOW/SPARESET command in the RAID Commands Chapter.

### 4.5 Maintaining a RAID Array

As with any OpenVMS disk, it is occasionally necessary to verify, back up, and defragment a RAID virtual device. Utilities designed to verify, back up, and defragment a disk may be used on the RAID virtual disk as on a physical disk.

---

#### CAUTION

---

It is important to understand that these utilities should **not** be used on the individual members of the RAID array, but only on the virtual disk. **Using such utilities on the individual RAID array members will result in lost or corrupted data.**

---

#### 4.5.1 Maintenance for RAID 5 Sets

RAID 5 technology needs to maintain extra or redundant data on the member disks to recover from loss of data blocks. This extra data is stored in so called Parity Blocks. These parity blocks are maintained during a write operation and are used during read if the array has lost data blocks, e.g. missing member device.

During a write operation, parity blocks are marked invalid until the user data has been written and the parity blocks have been updated. Any failure during a write operation (e.g. system crash) could leave a parity block invalid. Read requests are served directly from the member containing the user data. If the member is missing (array in reduced state) then the parity blocks are used to recover the user data. If the parity block is invalid then a parity error will be reported to the user program. It is possible, even though it is rare, that such areas remain undetected for quite some time until the array becomes reduced.

A RAID 5 array with invalid parity blocks is called "not fully redundant" because not all user data could be recovered when the array is in a reduced state (member missing).

The RAID ANALYZE/REPAIR allows a user with OPER privilege to verify the redundancy state of an array and, where redundancy has been lost, to recover it. In some cases redundancy cannot be recovered and data is permanently lost. The ANALYZE/REPAIR command lists these areas by virtual unit and logical block number.

#### Example 4–2 ANALYZE/REPAIR Example

```
$ RAID ANALYZE/ARRAY/REPAIR SUPPORT_DB
%RAID-E-ANREPPAR, parity error on device DPA1341: logical block number 370080
%RAID-E-ANREPPAR, parity error on device DPA1341: logical block number 370081
%RAID-E-ANREPERR, check repair report
$
```

The repair operation recovered full redundancy but data of 2 blocks has been lost. To identify files affected by these errors use DCL command ANALYZE/DISK/READ.

#### Example 4–3 ANALYZE/DISK/READ Example

```
$ ANALYZE/DISK/READ DPA1341:
Analyze/Disk_Structure for _DPA1341: started on ...

%ANALDISK-W-READFILE, file (13,1,1) [RAID_DB]SUPPORT_DB.DAT;1 error reading VBN 1
-SYSTEM-F-PARITY, parity error
%ANALDISK-W-READFILE, file (13,1,1) [RAID_DB]SUPPORT_DB.DAT;1 error reading VBN 2
-SYSTEM-F-PARITY, parity error
```

Files reported with parity errors need to be replaced or restored. Writing new data onto blocks with parity errors writes new data and parity blocks and clears the error if successful.

The repair operation can be performed while the array is in use but causes extra I/O load and should therefore be avoided during peak usage hours.

## 4.6 Obtaining Information About RAID Arrays

There are a couple ways of obtaining information about RAID Arrays. One is by using command procedures. The second is by using the ANALYZE command.

### 4.6.1 Obtaining Information about RAID Arrays via Command Procedures

Because OpenVMS software does not provide sufficient information about RAID arrays using the \$GETDVI system service or the DCL lexical function F\$GETDVI, a command procedure is included with the RAID software to aid in obtaining information about RAID arrays.

#### RAID\$CONFIG.COM

The command procedure SYS\$EXAMPLES:RAID\$CONFIG.COM may be used to obtain information about the existence, membership, and status of the RAID arrays and sparesets in existence on the system.

It works by parsing the output from RAID SHOW commands to pick up critical items of data (supplementing this with information from the DCL lexical function F\$GETDVI about shadow set membership for RAID 0+1 arrays) and then placing the information into either DCL global symbols or logical names, from which another command procedure or program may obtain the information and either display it or take appropriate action based on the information so gleaned.

By default, the results are returned in DCL global symbols. If the P1 parameter passed to RAID\$CONFIG is "LOGICAL\_NAMES", the results are instead returned using logical names. By default, logical names are defined in the process logical name table, but a P2 parameter of either SYSTEM or GROUP may be specified to place the resulting logical names in the system or group logical name tables.

The information returned for arrays is:

RAID\$CONFIG_ARRAY_COUNT	Number of RAID arrays
RAID\$CONFIG_ARRAY_n_ID	RAID array IDs
RAID\$CONFIG_ARRAY_n_RAID_LEVEL	RAID array RAID Level (0, 5, 0+1)
RAID\$CONFIG_ARRAY_n_STATE	RAID array state (normal, reconstr., etc.)
RAID\$CONFIG_ARRAY_n_VIRTUAL_DEVICE_LIST	DPAn: device name(s) for partition(s)
RAID\$CONFIG_ARRAY_n_MEMBER_COUNT	RAID array size (# of members)
RAID\$CONFIG_ARRAY_n_MEMBER_m_DEVICE_NAME	Member device names
RAID\$CONFIG_ARRAY_n_MEMBER_m_STATE	Member state (normal, missing...)

In addition, for RAID 0+1 arrays, the following information is returned:

RAID\$CONFIG_ARRAY_n_MEMBER_m_SHADOW_MEMBER_COUNT	(Shadow set depth)
RAID\$CONFIG_ARRAY_n_MEMBER_m_SHADOW_MEMBER_s_DEVICE_NAME	(Device names)
RAID\$CONFIG_ARRAY_n_MEMBER_m_SHADOW_MEMBER_s_STATE	State of each of the following: "ShadowMember", "FullCopying", or "MergeCopying"

The information returned for sparesets is:

RAID\$CONFIG_SPARESET_COUNT	(Number of sparesets)
RAID\$CONFIG_SPARESET_n_ID	(Spareset IDs)
RAID\$CONFIG_SPARESET_n_CHARACTERISTIC_SIZE	(Spareset charact. size)
RAID\$CONFIG_SPARESET_n_MEMBER_COUNT	(Spareset # of members)
RAID\$CONFIG_SPARESET_n_MEMBER_m_DEVICE_NAME	(Member device names)
RAID\$CONFIG_SPARESET_n_MEMBER_m_STATE	(Member state)

When the output of RAID\$CONFIG is directed into logical names, it is possible to set up one process running RAID\$CONFIG and defining logical names which are accessible by other processes. So that processes using the information can tell when it has been updated, there is also a timestamp logical name defined if the results are being placed into logical names:

## 4.6.2 Obtaining Information about RAID Arrays Using the ANALYZE Command

The RAID ANALYZE command allows you to create a report of on-disk metadata used by RAID. The command can be used on active RAID sets using the /ARRAY qualifier. The command can also be used on former members which are currently not in use by using the /UNITS qualifier.

The main purpose of the ANALYZE report is to provide technical support with additional data to investigate problems with your RAID sets. Some information can also be of interest to system managers.

### 4.6.2.1 Four RAID Created Files

RAID creates 4 files on a RAID set member:

- Configuration File: RAID\$CONFIGURATION\_MANAGEMENT.SYS, the configuration file containing the hints list of possible members
- Container File: RAID\$BC1.SYS, the container file with the user data plus overhead data
- Test Area: RAID\$FAULT\_MANAGEMENT.SYS, used as I/O test area for spare sets
- Unused Space: RAID\$RESERVED.SYS, occupies the unused space on this member

The RAID ANALYZE command produces a formatted report of the configuration and the container file. These files contain sections called Prolog, Header and Prefix which are overhead data and do not contain useful information for RAID users and are therefore not covered here.

### 4.6.2.2 Configuration File Report

Example 4–4 shows a sample report from a RAIDset Configuration File.

#### Example 4–4 Sample Configuration File Report

```
Processing $1$DUA130:[000000]RAID$CONFIGURATION_MANAGEMENT.SYS
```

```
Last known configuration: 1
```

```
Usage:          RAID set member
RAIDset Name:   SUPPORT_DB
RAIDset UID:    7C98E6F3 11D0801D 00083B87 DFC7392B
Member #0:     normal disk device
Member #1:     normal disk device
Member #2:     normal disk device
Member #3:     normal disk device
Member devices: $1$DUA130
                $10$DKD500
                $1$DUA151
                $1$DUA149
```

Number	Title	Description
1	Last known configuration	<p>Reports how this device is used (RAID or spare set member), the name of the set it is/was part of and the characteristic of all members (normal or shadowed member).</p> <p>Lists all the physical devices which have been last known. This list serves as a hint to the BIND command to search for possible members of this set.</p>

#### 4.6.2.3 Container File Report

Example 4–5 shows the contents of a container file report.

##### Example 4–5 Sample Container File

```
Processing $1$DUA130:[000000]RAID$BC1.SYS
```

```
Base Container: 1
```

```
Base Data Loc: 0
Read Errors: 0
Read Window: 0
Write Errors: 0
Write Window: 0
Timeout: 120000 in 10ms
```

```
Forced Error Descriptor: 2
```

```
FE Cont. Base: 0
FE Blks Size: 477
Location[0]: 1951583
Location[1]: 1951106
FE flags 0 through 1951105 were CLEAR
```

```
RAID Container Member: 3
```

```
Rotation: RAID 5 Left Symetric
Members: 4
Dist. Member: 0
State: Normal
Generation: 23
Char. Size: 600000
Chunk Size: 32
Chunks/MDU: 64
PMD Segm. Size 1
Flags/PMD Seg.: 512 (2 bits each)
MDUs: 292
MDU #0 Loc.: 0
Unused: 0
Recon. Pri: medium
Members Base Container UIDs:
```

```
Member # 0: DDF57350 11D0801B 00083987 DFC7392B
Member # 1: 80210BF7 11D0801D 00083B87 DFC7392B
Member # 2: 92C2FC11 11D0801D 00083B87 DFC7392B
Member # 3: B6DCB0CD 11D08023 00083B87 DFC7392B
```

```
Flags in PMD Segments: 4
```

```
PMD flags 0 through 11046 were CLEAR
PMD flags 11047 through 11056 were SET
PMD flags 11057 through 149503 were CLEAR
%RAID-W-ANPMD, invalid parity blocks detected
```

(continued on next page)

## Example 4–5 (Cont.) Sample Container File

```
Segmentation: 5
  Segment 0:
    Location: 0
    Size:      1794042
SEGMENT 0 information follows:
Device Presentation:
  Base location: 0
  Geometry:
    Track size: 255
    Cyl size  : 255
    Cylinders : 28
```

### 1 Base Container

Contains the timeout value for this array.

### 2 Forced Error Descriptor (Used for RAID 5 only)

Describes size and location of the forced error (FE) flags. The FE flags protect each block in the container file. A set flag indicates that the specified block contains undefined data and a READ operation will return a parity error. A successful WRITE operation clears the flag.

In case of set FE flags use DCL command ANALYZE/DISK/READ on the virtual units (DPAnn) of this array to find all files with parity errors.

### 3 RAID Container Member

Rotation	RAID level either RAID 0 or RAID 5
Members	total members in a set, for a RAID 0 it includes a permanently missing unused member #0
Dist. Member	index of missing/removed/replaced member
State	either NORMAL, REDUCED or RECONSTRUCT
Generation	age number of this member
Char. Size	useable size of container
Chunk Size	chunk size
Chunks/MDU	number of chunks per metadata unit (MDU)
PMD Segm. Size	disk block size of parity metadata in a MDU
Flags/PMD Seg.	number of 2-bit flags in a PMD segment
MDUs	total number of MDUs in container
MDU #0 Loc.	location of the beginning of the first MDU on the underlying container
Recon. Pri	reconstruct priority (always medium)
Members Base Container UIDs	lists the unique IDs of all member's container files in order

#### **4 Flags in PMD Segments (Used for RAID 5 only)**

Parity metadata (PMD) flags are used to qualify the contents of a parity block. Parity blocks contain the extra or redundant data on a RAID 5 array to recover user data in failure situations. A parity block contains an exclusive OR of all user data blocks at the same location in all container files.

Before a write operation starts the PMD flag for this area is set. When the write completes successfully, the flag is cleared. It is normal to see PMD flags set while the array is in use. If a flag remains set and the array becomes reduced, the user data cannot be reconstructed and a parity error will be returned.

If there are PMD flags which seems to be set permanently use RAID ANALYZE/ARRAY/REPAIR to recover full redundancy, where possible.

#### **5 Segmentation**

Lists the size and disk geometry of each segment or partition on this array. This section is only reported when in NORMAL state.

### **4.7 Managing RAID Arrays Automatically**

Some systems require an automated system management setup. RAID 5 arrays can be set up to have member devices automatically replaced from a set of spares. Over time a spare set can become empty, or a RAID 0+1 set can lose a shadow set member. This typically requires some human intervention. For such cases, a system manager can define command procedures which are executed when such events happen.

When disk devices are being removed from RAID arrays or spare sets become empty, these are critical events that could be handled by automated command procedures. A command procedure could automatically replace the failing disk from a pool of available disk devices, or send a mail warning message or even execute a program to send a message to a pager.

The event notification system is controlled by logical names in the system logical name table. For each event, a logical name can be defined referencing a command procedure. If the logical name is not defined or points to a non-existing command procedure file no notification will be done.

Missing parts of the file specification for the event command procedure will be filled in from "SYSS\$MANAGER:.COM". If a command procedure file can be found, it is submitted to the SYSS\$BATCH queue, using the SYSTEM account. The default queue and account name can be overridden by logical names in the system logical name table.

The logical names will be translated with every new event thus allowing you to turn this feature off or on any time. Please refer to the RAID Commands Chapter for more details. Example 4-6 shows a sample event notification procedure.

### Example 4-6 Sample Event Notification procedure

```
$ DEFINE/SYSTEM RAID$NOTIFY_PROCEDURE_RECONINC RAID_NOTIFY.COM
$ DEFINE/SYSTEM RAID$NOTIFY_PROCEDURE_RECONCOMP RAID_NOTIFY.COM
$ DEFINE/SYSTEM RAID$NOTIFY_PROCEDURE_SPAREREMOV RAID_NOTIFY.COM
$ DEFINE/SYSTEM RAID$NOTIFY_PROCEDURE_SPARESETEEMPTY RAID_NOTIFY.COM
$ DEFINE/SYSTEM RAID$NOTIFY_PROCEDURE_SHADOWREMOVE RAID_NOTIFY.COM
$ DEFINE/SYSTEM RAID$NOTIFY_PROCEDURE_REMOVE RAID_NOTIFY.COM
$ DEFINE/SYSTEM RAID$NOTIFY_PROCEDURE_DUMP RAID_NOTIFY.COM
$ DEFINE/SYSTEM RAID$NOTIFY_PROCEDURE_EXLICENSE RAID_NOTIFY.COM
$ DEFINE/SYSTEM RAID$NOTIFY_PROCEDURE_NEWNODECOMP RAID_BIND.COM
$ DEFINE/SYSTEM RAID$NOTIFY_USERNAME_OPERATOR
$ DEFINE/SYSTEM RAID$NOTIFY_QUEUE SYSMGMT

$!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
$! RAID_NOTIFY.COM
$!
$! This is a general command procedure to be activated through RAID
$! events.
$!
$! Inputs: P1 = event type
$! P2 = array name or node name
$! P3 = empty or device name depending on event type
$!
$! Outputs: Events RECONINC, RECONCOMP, SPAREREMOV, REMOVE,
$! DUMP and EXLICENSE cause mail to be sent to SYSTEM
$!
$! Events SHADOWREMOVE and SPARESETEEMPTY replace the removed
$! shadow set member or spareset member with a spare
$! device of same type.
$!
$ SET NOON
$ SET VERIFY
$ Spares = "$1$DUA500/$1$DUA501/$1$DUA502/$1$DUA503/$1$DUA504/$1$DUA505/" + -
$ "$1$DUA130/$1$DUA131/$1$DUA132/$1$DUA133/$1$DUA134/$1$DUA135"
$ GOTO Label_'P1' ! switch to event type
$ EXIT
$!
$!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
$ Label_RECONINC:
$ Label_RECONCOMP:
$ Label_SPAREREMOV:
$ Label_REMOVE:
$ Label_DUMP:
$ Label_EXLICENSE:
$!
$! for these events just send a mail message
$!
$ MAIL/SUBJECT="RAID reported event ''P1': ''P2' ''P3' ''P4'" -
$ NLA0: SYSTEM
$ EXIT
$!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
$ Label_SHADOWREMOVE:
$!
$! SHADOWREMOVE is the only one event which might be called more
$! then once for the same event. To make sure that we have not
$! already added a member we first check how many member there are
$! in the shadowset currently.
$!
$ NextMember = F$GETDVI(P3, "SHDW_NEXT_MBR_NAME") ! get first member
$ IF NextMember .NES. "" THEN -
$ NextMember = F$GETDVI(NextMember, "SHDW_NEXT_MBR_NAME")
```

(continued on next page)



#### Example 4-6 (Cont.) Sample Event Notification procedure

```
$ IF NextMember .NES. "" THEN EXIT ! more than one member - nothing to do
$!
$! at this point the shadowset has less than 2 members
$!
$ GOSUB GetSpare
$!
$! add new spare to shadowset
$!
$ MOUNT/SYSTEM/NOASSIST 'P3'/SHADOW='SpareDev' 'F$GETDVI(P3, "VOLNAM")
$ IF $SEVERITY
$ THEN
$   MAIL NLA0: SYSTEM -
$     /SUBJECT="'SpareDev' added to shadowset ''P3' in array ''P2'"
$ ELSE
$   MAIL NLA0: SYSTEM -
$     /SUBJECT="Failed to add ''SpareDev' to shadowset ''P3' in array ''P2'"
$ ENDIF
$ EXIT
$!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
$ Label_SPARESETEEMPTY:
$!
$ GOSUB GetSpare
$!
$! initialize new spare
$!
$ RAID INITIALIZE/SPARE 'SpareDev'/NOCONFIRM
$ IF .NOT. $SEVERITY
$ THEN
$   MAIL NLA0: SYSTEM -
$     /SUBJECT="Failed to initialize ''SpareDev' for spareset ''P2'"
$   EXIT
$ ENDIF
$!
$! add spare to spareset
$!
$ RAID ADD/SPARE 'P2' 'SpareDev'
$ IF $SEVERITY
$ THEN
$   MAIL NLA0: SYSTEM -
$     /SUBJECT="'SpareDev' added to empty spareset ''P2'"
$ ELSE
$   MAIL NLA0: SYSTEM -
$     /SUBJECT="Failed to add ''SpareDev' to empty spareset ''P2'"
$ ENDIF
$ EXIT
$!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
$!
$! Subroutine GetSpare
$!
$! Input: P3 = device name of shadowset or device name of removed spare
$!
$! Output: SpareDev = device name of available spare device which
$!   matches the device type of the input device
$!   If no spare can be found the command procedure exits.
$!
$ GetSpare:
$!
$ N = 0
$ SpareDev = ""
$!
$ NextN:
```

(continued on next page)

### Example 4-6 (Cont.) Sample Event Notification procedure

```
$!  
$ NextSpare = F$ELEMENT(N, "/", Spares)  
$ N = N + 1  
$ IF NextSpare .EQS. "/" ! end of list?  
$ THEN  
$   MAIL NLAO: SYSTEM -  
$     /SUBJECT="No spare found to replace ''P3' in ''P2'"  
$   EXIT  
$ ENDIF  
$!  
$! Make sure device exists, is available, not allocated and not mounted.  
$!  
$ IF .NOT. F$GETDVI(NextSpare, "EXISTS") THEN GOTO NextN  
$ IF .NOT. F$GETDVI(NextSpare, "AVL") THEN GOTO NextN  
$ ALLOCATE 'NextSpare'  
$ IF .NOT. $SEVERITY THEN GOTO NextN  
$ DEALLOCATE 'NextSpare'  
$ IF F$GETDVI(NextSpare, "MNT") THEN GOTO NextN  
$!  
$! Make sure we replace with same device type.  
$!  
$ IF 'F$GETDVI(NextSpare, "DEVTYPE)"'.NE. -  
$   'F$GETDVI(P3, "DEVTYPE)'  
$   THEN GOTO NextN  
$ SpareDev = NextSpare  
$ RETURN  
  
$!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!  
$! RAID_BIND.COM  
$!  
$! This command procedure is to be activated through RAID event  
$! NEWNODECOMP and can serve as the only method of binding arrays.  
$! At this point the newly started RAID server has bound all arrays  
$! which were bound elsewhere in a cluster. Only arrays which are  
$! unknown yet need to be bound.  
$!  
$! With this procedure in place RAID$SYSTARTUP.COM is not needed.  
$!  
$! Inputs: P1 = NEWNODECOMP  
$! P2 = nodename  
$!  
$! Outputs: Binds all arrays which have not been bound yet.  
$!  
$ SET NOON  
$ SET VERIFY  
$!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!  
$!  
$! The following lists describe the arrays to be bound with all  
$! their qualifiers and parameters.  
$!  
$! List of arrays to be bound:  
$!  
$ RAIDArrays = "ARRAY1/ARRAY2/ARRAY3/SPARE1"  
$!  
$! List of qualifiers for each array:  
$!  
$ Qualifiers = "/SHADOW/USE=(DSA101,DSA102)::/SPARE"  
$!  
$! List of devices per array:  
$!  
$ Devices = "$1$DUA101,$1$DUA102/" + -  
$           "$1$DUA201,$1$DUA202,$1$DUA203/" + -
```

(continued on next page)

#### Example 4-6 (Cont.) Sample Event Notification procedure

```
"$1$DUA301,$1$DUA302,$1$DUA303/" + -
"$1$DUA911,$1$DUA912"
$!
$! List of virtual units per array:
$!
$ Units          = "DPA101/DPA201,DPA202/DPA301,DPA302,DPA303/"
$!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
$!
$ N = 0
$!
$ NextN:
$!
$ ArrayN = F$ELEMENT(N, "/", RAIDArrays)
$ IF ArrayN .EQS. "/" THEN EXIT          ! All done!
$ RAID SHOW/OUTPUT=NLA0: 'ArrayN'       ! Does server know about array?
$ IF .NOT. $SEVERITY THEN GOTO BindN    ! No! Go and bind it.
$ N = N + 1
$ GOTO NextN
$!
$ BindN:
$!
$ RAID BIND 'F$ELEMENT(N, ":", Qualifiers)' 'ArrayN' -
           'F$ELEMENT(N, "/", Devices)' 'F$ELEMENT(N, "/", Units)'
$ IF .NOT. $SEVERITY THEN -
$   MAIL NLA0: SYSTEM -
           /SUBJECT="Array ''ArrayN' failed to bind on node ''P2'"
$ N = N + 1
$ GOTO NextN
```



This chapter defines what RAID partitioning is and explains why it is used. It also explains how to use this partitioning feature of HP RAID Software for OpenVMS and gives examples. Table 5–1 list the topics covered in this chapter.

**Table 5–1 Topics in This Chapter**

<b>Subject</b>	<b>Section</b>
What Is Partitioning?	Section 5.1
Why Use Partitioning?	Section 5.2
Using Partitioning	Section 5.3
Partitioning Examples	Section 5.4

## 5.1 What Is Partitioning?

Partitioning is defined as the act or process of dividing something into parts. In the HP RAID Software for OpenVMS, partitioning is the act or process of dividing a RAID array into one or more RAID virtual devices.

## 5.2 Why Use Partitioning?

There are several situations in which partitioning is especially useful:

- A single RAID virtual device, which maps onto the RAID array, may present a device larger than what the operating system supports. Therefore, without partitioning, some of the array would be inaccessible. Access to the complete RAID array can be accomplished by partitioning the RAID array into multiple RAID virtual devices, each of a size that the operating system supports.
- A set of RAID virtual devices which map to a single RAID array may make the system manager's job easier. For example, the system manager can more easily backup multiple, smaller RAID virtual devices than a single, larger RAID virtual device.
- A single storage device may have more space than the operating system supports. Creating a single member RAID0 array with multiple RAID partitions each of a size supported by the operating system provides access to all the available space on the device.

## 5.3 Using Partitioning

The RAID software allows you to specify how many partitions are to be used in a given RAID array and how large each partition should be. One or more partitions will be created on a single-member RAID0 array or on multiple-member RAID0 or RAID5 arrays. The user must specify unique virtual device names for each configured partition on the array.

### 5.3.1 Creating Partitions with RAID INITIALIZE

RAID INITIALIZE will always create at least one partition in an array. The /SIZE qualifier to the RAID INITIALIZE command give the user the control mechanism for establishing the number and size of partitions across a RAID array. Refer to Chapter 11.

- By default, at least one partition will be created. However, size limits supported by the Files-11 system may restrict the partition size.
- To specify the size of a partition, or to create multiple partitions, use the /SIZE qualifier to the RAID INITIALIZE command.

The two mechanisms used to partition RAID arrays are discussed in more detail in the following sections.

#### 5.3.1.1 Partitioning by Default

At least one partition will be created. In this case the RAID software partitions the RAID array according to:

- The space available on the RAID array, and
- The size limitations imposed by the Files-11 file system.

#### 5.3.1.2 Partitioning Using Specified Sizes

Partition sizes are specified with the /SIZE qualifier on the RAID INITIALIZE command. Use the qualifier to specify the number of disk blocks for each partition desired on the RAID array. The RAID software then allocates disk space to partitions sequentially based on their location in the RAID INITIALIZE command string.

You may specify different sizes for different partitions. However, the actual size of any partition created by the RAID software will be no greater than the maximum size supported by the operating system.

The RAID software adjusts specified partition sizes as necessary in a manner similar to that described in the previous section. For example:

- If disk space is still available after the RAID software has created all the partitions specified with the /SIZE qualifier, the remaining space will be partitioned into one or more additional partitions according to the space available.
- If during the execution of a RAID INITIALIZE command the RAID software encounters a partition requiring more disk space than remains on the RAID array, that partition will be truncated to contain only the remaining space. Other subsequent partitions specified in the RAID INITIALIZE command string will be ignored. The output display from the RAID INITIALIZE command displays the partition sizes actually created.

---

#### Note

---

Once the partition has been created, the sizes of the partitions cannot be changed without executing the RAID INITIALIZE command again. The RAID INITIALIZE command causes all user data to be lost.

---

## 5.3.2 Binding

The RAID BIND command requires a user to specify a unique RAID virtual device name that is unique for each partition created during the execution of the RAID INITIALIZE command.

The RAID command will fail if there is not a one-for-one correspondence between the number of partitions displayed during the RAID initialization process and the virtual device names specified in the RAID BIND command. A message describing the reason for the failure and the number of required virtual device names will be displayed.

---

### Note

---

The RAID UNBIND command unbinds all the virtual devices associated with the RAID array. It is not possible to unbind a selected list of RAID virtual devices associated with the same RAID array.

---

## 5.4 Partitioning Examples

This section provides examples of RAID INITIALIZE, RAID BIND, and RAID SHOW commands used to create and show partitions on different types of RAID arrays.

### 5.4.1 Partitions in a RAID0 Array

Example 5–1 shows how to initialize and bind a RAID0 array with four partitions of 3,000,000 blocks each. These devices are initialized into the array "MY\_ARRAY," and the command line specifies the size of four partitions. In this case, however, the total number of disk blocks specified with the /SIZE qualifier is fewer than the total number of disk blocks available in the array. The RAID software creates an additional partition using the remaining space, and the size of the five partitions is displayed during the initialization process.

#### Example 5–1 Creating Partitions on a RAID0 Array

```
$ RAID INITIALIZE MY_ARRAY $1$DKA100:,$1$DKB0:,$1$DKB200: /RAID_LEVEL=0 -  
/SIZE=(3000000,3000000,3000000,3000000)  
INIT will destroy existing data, do you want to continue [N]? y  
  
%RAID-I-PRTCREATED, Array partition 1 created: 3000000 blocks  
%RAID-I-PRTCREATED, Array partition 2 created: 3000000 blocks  
%RAID-I-PRTCREATED, Array partition 3 created: 3000000 blocks  
%RAID-I-PRTCREATED, Array partition 4 created: 3000000 blocks  
%RAID-I-PRTCREATED, Array partition 5 created: 309104 blocks  
  
$ RAID BIND MY_ARRAY $1$DKA100:,$1$DKB0,$1$DKB200 DPA11,DPA22,DPA33,DPA44,DPA55  
%RAID-I-ISBOUND, unit _$1$DKA100: is bound as a member of RAID array MY_ARRAY  
%RAID-I-ISBOUND, unit _$1$DKB0: is bound as a member of RAID array MY_ARRAY  
%RAID-I-ISBOUND, unit _$1$DKB200: is bound as a member of RAID array MY_ARRAY  
%RAID-I-VUCREATE, virtual unit DPA0011 was created for partition 1 on MY_ARRAY  
%RAID-I-VUCREATE, virtual unit DPA0022 was created for partition 2 on MY_ARRAY  
%RAID-I-VUCREATE, virtual unit DPA0033 was created for partition 3 on MY_ARRAY  
%RAID-I-VUCREATE, virtual unit DPA0044 was created for partition 4 on MY_ARRAY  
%RAID-I-VUCREATE, virtual unit DPA0055 was created for partition 5 on MY_ARRAY  
  
$ RAID SHOW/FULL MY_ARRAY
```

(continued on next page)

### Example 5-1 (Cont.) Creating Partitions on a RAID0 Array

HP RAID Software V3.0 Display Time: 1-NOV-2004 12:53:36.13  
© 2004 Hewlett-Packard Development Company, L.P. All Rights Reserved.

RAID Array Parameters:

```
Current RAID Array ID:  MY_ARRAY
Permanent RAID Array ID: MY_ARRAY
Date Created:           1-NOV-2004 11:10:27.89
Last Bind:              1-NOV-2004 11:11:41.00
RAID Level:             0
Current State:          NORMAL

Characteristic Size:    4106822          Total Capacity: 12309104
Member Count:           3                Chunk Size:      120
Pool in Use:            6912             Max Pool Value:  0
Pages in Use:           0                Max Page Value:  0
```

RAID Array Configuration:

Member Index	Name	State
0	_\$1\$DKA100:	NORMAL
1	_\$1\$DKB0:	NORMAL
2	_\$1\$DKB200:	NORMAL

RAID Array Operations:

Member Index	Name	Reads	Writes	Errors
0	_\$1\$DKA100:	1096	6023	0
1	_\$1\$DKB0:	1173	8648	0
2	_\$1\$DKB200:	1132	6065	0
Member Total:		3401	20736	0

Virtual Unit	Size	Status	Reads	Writes	Errors
DPA0011:	3000000	ACCESS	1532	14403	0
DPA0022:	3000000	ACCESS	472	1344	0
DPA0033:	3000000	ACCESS	381	1651	0
DPA0044:	3000000	ACCESS	344	723	0
DPA0055:	309104	ACCESS	43	61	0

## 5.4.2 Partitions in a RAID0+1 Array

Example 5-2 shows how to initialize and bind a RAID0+1 array with four partitions of 3,000,000 blocks each. Three devices are initialized into the array "MY\_ARRAY," and the command line specifies the size of four partitions. In this case, however, the total number of disk blocks specified with the /SIZE qualifier is fewer than the total number of disk blocks available in the array. The RAID software creates an additional partition using the remaining space, and the size of the five partitions is displayed during the initialization process.

### Example 5-2 Creating Partitions on a RAID0 Array

(continued on next page)



### Example 5–2 (Cont.) Creating Partitions on a RAID0 Array

```
$ RAID INITIALIZE MY_ARRAY $1$DKA100:,$1$DKB0:,$1$DKB200: /RAID_LEVEL=0 -
                               /SIZE=(3000000,3000000,3000000,3000000)
INIT will destroy existing data, do you want to continue [N]? y

%RAID-I-PRTCREATED, Array partition 1 created: 3000000 blocks
%RAID-I-PRTCREATED, Array partition 2 created: 3000000 blocks
%RAID-I-PRTCREATED, Array partition 3 created: 3000000 blocks
%RAID-I-PRTCREATED, Array partition 4 created: 3000000 blocks
%RAID-I-PRTCREATED, Array partition 5 created: 309104 blocks

$ RAID BIND MY_ARRAY $1$DKA100:,$1$DKB0,$1$DKB200 -
DPA11,DPA22,DPA33,DPA44,DPA55/SHADOW/USE_SHADOW_DEVICES=(DSA1,DSA2,DSA3)
%RAID-I-ISBOUND, unit DSA1: is bound as a member of RAID array MY_ARRAY
%RAID-I-INUSE, unit _$1$DKA100: is a shadow set member of _DSA1:
%RAID-I-ISBOUND, unit _DSA2: is bound as a member of RAID array MY_ARRAY
%RAID-I-INUSE, unit _$1$DKB0: is a shadow set member of _DSA2:
%RAID-I-ISBOUND, unit _DSA3: is bound as a member of RAID array MY_ARRAY
%RAID-I-INUSE, unit _$1$DKB200: is a shadow set member of _DSA3:
%RAID-I-VUCREATE, virtual unit DPA0011 was created for partition 1 on MY_ARRAY
%RAID-I-VUCREATE, virtual unit DPA0022 was created for partition 2 on MY_ARRAY
%RAID-I-VUCREATE, virtual unit DPA0033 was created for partition 3 on MY_ARRAY
%RAID-I-VUCREATE, virtual unit DPA0044 was created for partition 4 on MY_ARRAY
%RAID-I-VUCREATE, virtual unit DPA0055 was created for partition 5 on MY_ARRAY

$ RAID SHOW MY_ARRAY

HP RAID Software V3.0 Display Time: 1-NOV-2004 13:04:08.55
© 2004 Hewlett-Packard Development Company, L.P. All Rights Reserved.
```

#### RAID Array Parameters:

```
Current RAID Array ID: MY_ARRAY
Permanent RAID Array ID: MY_ARRAY
RAID Level: 0+1
Current State: NORMAL
```

#### RAID Array Configuration:

Member Index	Name	State	ShadowSet Members	ShadowSet State
0	_DSA1:	NORMAL	1	SteadyState
1	_DSA2:	NORMAL	1	SteadyState
2	_DSA3:	NORMAL	1	SteadyState

Virtual Unit	Size	Status	Reads	Writes	Errors
DPA0011:	3000000	ACCESS	0	0	0
DPA0022:	3000000	ACCESS	0	0	0
DPA0033:	3000000	ACCESS	0	0	0
DPA0044:	3000000	ACCESS	0	0	0
DPA0055:	309104	ACCESS	0	0	0

### 5.4.3 Partitions with Truncation

Example 5–3 shows an example of a single disk being used to create a RAID0 array. In this example, the user has specified a size of 2,000,000 blocks for each of three partitions. The total number of disk blocks available for use on the RAID array is fewer than the total number of disk blocks specified in the RAID INITIALIZE command string. The RAID software therefore creates the first two partitions using the size specified by the user, and then truncates the third partition so it fits within the remaining disk space.

### Example 5–3 Creating Partitions with Truncation

```
$ RAID INITIALIZE MY_ARRAY $1$DKA100:/RAID=0/SIZE=(2000000,2000000,2000000)
INIT will destroy existing data, do you want to continue [N]? y

%RAID-I-PRTCREATED, Array partition 1 created: 2000000 blocks
%RAID-I-PRTCREATED, Array partition 2 created: 2000000 blocks
%RAID-I-PRTCREATED, Array partition 3 created: 103028 blocks

$ RAID BIND MY_ARRAY $1$DKA100: DPA1,DPA2,DPA3
%RAID-I-ISBOUND, unit _$1$DKA100: is bound as a member of RAID array MY_ARRAY
%RAID-I-VUCREATE, virtual unit DPA0001 was created for partition 1 on MY_ARRAY
%RAID-I-VUCREATE, virtual unit DPA0002 was created for partition 2 on MY_ARRAY
%RAID-I-VUCREATE, virtual unit DPA0003 was created for partition 3 on MY_ARRAY

$ RAID SHOW MY_ARRAY

HP RAID Software V3.0 Display Time: 1-NOV-2004 13:24:29.97
© 2004 Hewlett-Packard Development Company, L.P. All Rights Reserved.
```

RAID Array Parameters:

```
Current RAID Array ID: MY_ARRAY
Permanent RAID Array ID: MY_ARRAY
RAID Level: 0
Current State: NORMAL
```

RAID Array Configuration:

Member					
Index	Name	State			
-----	-----	-----	-----	-----	-----
0	_\$1\$DKA100:	NORMAL			

Virtual Unit	Size	Status	Reads	Writes	Errors
-----	-----	-----	-----	-----	-----
DPA0001:	2000000	ACCESS	252	14283	0
DPA0002:	2000000	ACCESS	353	1077	0
DPA0003:	103028	ACCESS	20	39	0

### 5.4.4 Partitions with Specified Size Exceeding Disk Space

Example 5–4 shows an example of a RAID5 array initialized and bound with partitions of different sizes. Only the first two partitions match the size specified. The third partition consists of the remaining space. The last partition specified is ignored in this example.

#### Example 5–4 Partitions with Specified Size Exceeding Disk Space

```
$ RAID INITIALIZE MY_ARRAY $1$DKA100:,$1$DKB0:,$1$DKB200:/RAID=5 -
/SIZE=(3000000,4000000,6000000,3000000)
INIT will destroy existing data, do you want to continue [N]? y

%RAID-I-PRTCREATED, Array partition 1 created: 3000000 blocks
%RAID-I-PRTCREATED, Array partition 2 created: 4000000 blocks
%RAID-I-PRTCREATED, Array partition 3 created: 1207988 blocks
```

## 5.4.5 A BIND Command Without Enough Virtual Devices Specified

Assuming that three partitions were initialized on array "MY\_ARRAY," Example 5-5 shows the error message received when the BIND command does not specify enough virtual devices.

### Example 5-5 A BIND Command Without Enough Virtual Devices Specified

```
$ RAID BIND MY_ARRAY $1$DKA100: DPA1
%RAID-F-CMDFAIL, RAID command failed
-RAID-F-VUMISMATCH, incorrect number of virtual devices on the command line,
must specify 3 virtual devices, 1 virtual device specified
```

## 5.4.6 A BIND Command with All Partitions Specified

Example 5-6 shows an example of the same BIND command when all the partitions are specified correctly. This is followed by a RAID SHOW command.

### Example 5-6 A BIND Command with All Partitions Specified Correctly

```
$ RAID BIND MY_ARRAY $1$DKA100: DPA1,DPA2,DPA3
%RAID-I-ISBOUND, unit _$1$DKA100: is bound as a member of RAID array MY_ARRAY
%RAID-I-ISBOUND, unit _$1$DKB0: is bound as a member of RAID array MY_ARRAY
%RAID-I-ISBOUND, unit _$1$DKB200: is bound as a member of RAID array MY_ARRAY
%RAID-I-VUCREATE, virtual unit DPA0001 was created for partition 1 on MY_ARRAY
%RAID-I-VUCREATE, virtual unit DPA0002 was created for partition 2 on MY_ARRAY
%RAID-I-VUCREATE, virtual unit DPA0003 was created for partition 3 on MY_ARRAY
```

```
$ RAID SHOW MY_ARRAY
```

```
HP RAID Software V3.0 Display Time: 1-NOV-2004 14:16:05.24
```

```
© 2004 Hewlett-Packard Development Company, L.P. All Rights Reserved.
```

```
RAID Array Parameters:
```

```
Current RAID Array ID: MY_ARRAY
Permanent RAID Array ID: MY_ARRAY
RAID Level: 5
Current State: RECONSTRUCTING
                Operation 15.50% complete
                In progress on node NODEA
Associated Spareset: (none)
```

```
RAID Array Configuration:
```

Member						
Index	Name	State				
-----	-----	-----				
0	_\$1\$DKA100:	NORMAL				
1	_\$1\$DKB0:	RECONSTRUCTING				
2	_\$1\$DKB200:	NORMAL				

Virtual Unit	Size	Status	Recovered Operations	Reads	Writes	Errors
-----	-----	-----	-----	-----	-----	-----
DPA0001:	3000000	ACCESS	1322	372	6897	0
DPA0002:	4000000	ACCESS	153	387	825	0
DPA0003:	1207988	ACCESS	0	143	162	0



---

## Using Shadowed Disks in RAID0 Arrays

This chapter describes how HP RAID Software for OpenVMS is used to create RAID0 arrays using shadow sets. Table 6–1 lists the topics covered in this chapter.

**Table 6–1 Topics in This Chapter**

Subject	Section
Using Shadow Sets as Members of RAID0 Configurations	6.1
Why Use Volume Shadowing with RAID0 Arrays	6.2
Requirements for Using Volume Shadowing	6.3
Device Hierarchy in a RAID0 Array with Volume Shadowing	6.4
Related Documentation	6.5
Creating RAID0 Arrays with Shadow Sets	6.6
Adding Shadow Set Member Devices to a RAID0 Shadow Set	6.7
Removing Members from a Shadow Set	6.8
Rebinding with Shadowing	6.9
Rebinding Without Shadowing	6.10
Backup of Arrays with Shadow Sets	6.11
RAID CLONE Command	6.12
Using RAID0 and Shadowing in a VMScCluster	6.13
Performance Considerations	6.14
Error Handling	6.15

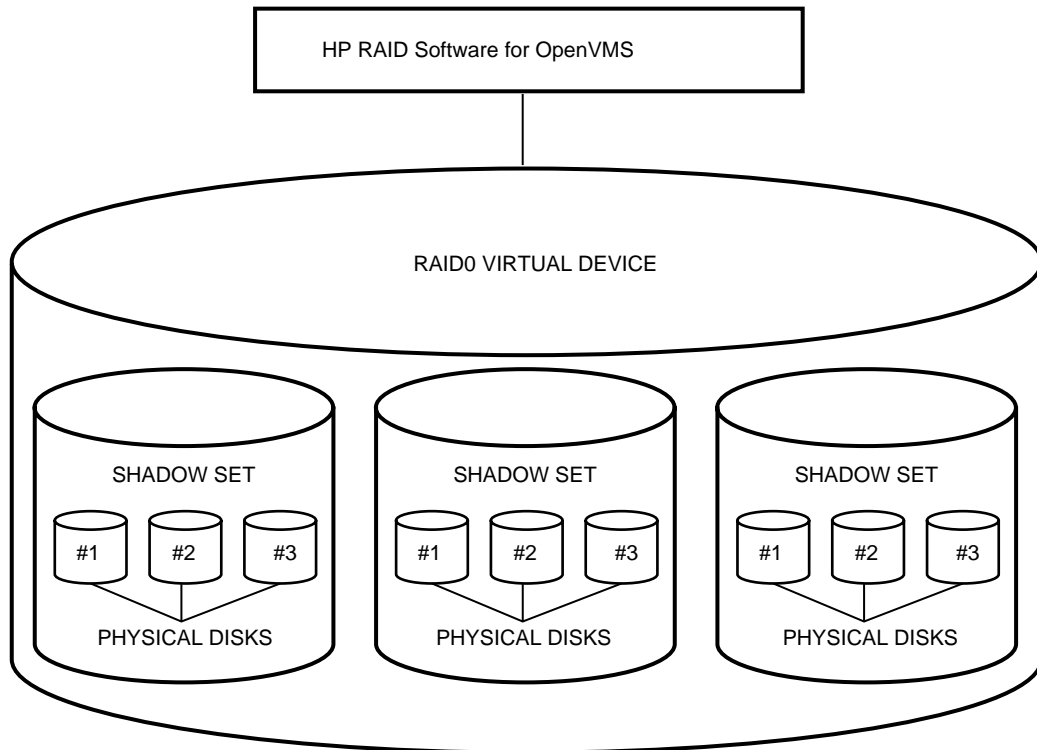
### 6.1 Using Shadow Sets as Members of RAID0 Configurations

A RAID0 Virtual Device may have shadow set virtual units as RAID0 array members. Volume Shadowing for OpenVMS Phase II logically associates physical devices and represents them to HP RAID software as a single virtual unit.

Compatible disk volumes may be mounted into a shadow set as shown in Figure 6–1. Each disk in the shadow set is known as a shadow set member, and the shadow set itself is the virtual unit.

In a RAID0 array, either all array members are shadow sets or no array members are shadow sets. The limits on the number of shadow sets in a RAID0 array are found in the release notes for HP RAID Software for OpenVMS. Figure 6–2 shows an example of a RAID0 array composed of two members, each of which is a shadow set virtual unit containing RA72 disk drives.

Figure 6–1 RAID0 Arrays and Shadow Sets



CXO-4054A-MC

## 6.2 Why Use Volume Shadowing with RAID0 Arrays

An OpenVMS HP RAID array may consist of shadow sets for a number of reasons:

### Increased availability

RAID0 arrays spread data across all the members and contain no parity for data redundancy. The loss of any member means that user data will be lost. However, Volume Shadowing adds the redundancy RAID0 functionality lacks so using Volume Shadowing in conjunction with RAID0 will greatly increase data availability.

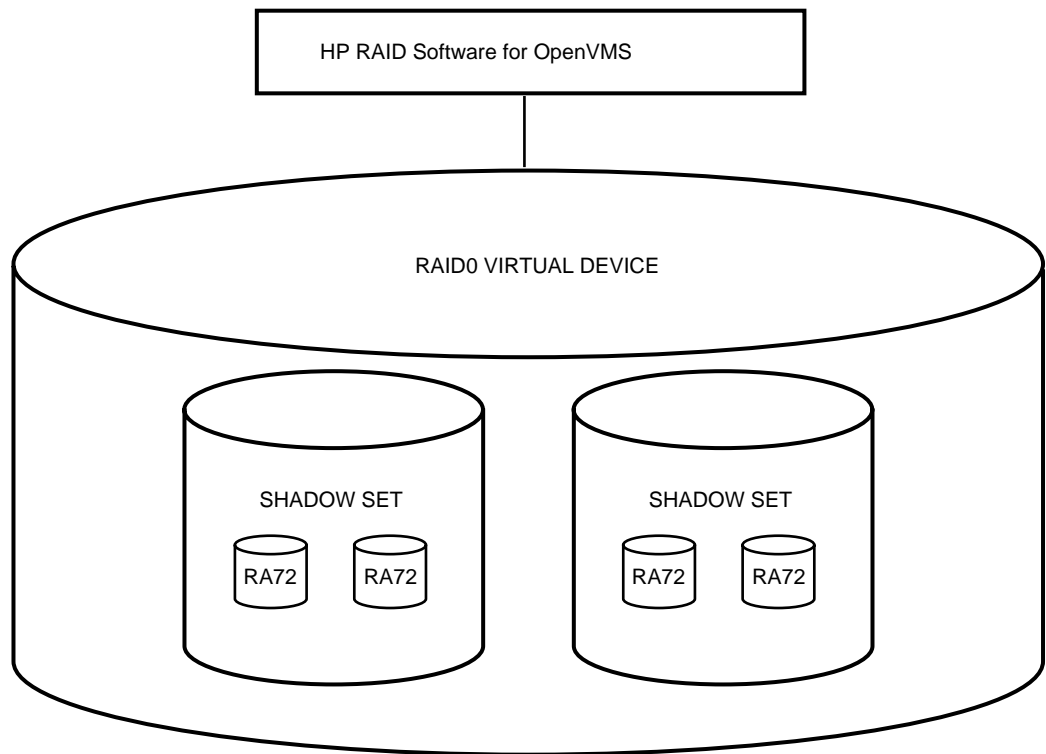
### No compromise of performance with increased availability

RAID0 arrays, RAID5 arrays, and shadow sets all provide higher read performance than the equivalent individual disks. The read and write performance of a RAID0 array is higher, but the data availability is lower.

The write performance of a shadow set is only slightly less than an individual disk. For applications which need both data protection and high write performance, it may be preferable to use a RAID0 array of shadow sets.

RAID5 arrays provide protection against the loss of any single member disk, but write performance is less compared with individual disks.

Figure 6–2 RAID0 Array with Two Members



CXO-4067A-MC

#### Near-online backup

Because Volume Shadowing software keeps identical copies of data on multiple disks, backing up data may be less disruptive to system operations.

A special RAID command can be used to separate RAID array shadow sets to facilitate backup operations. See Section 6.11.

#### Disaster tolerance

It may be desirable to keep copies of user data at two separate locations to provide protection against the destruction of an entire site. In a VMScluster configuration with CPU nodes and disks at two sites, Volume Shadowing software can be used to keep identical copies of data logically located on a single RAID array member.

### 6.3 Requirements for Using Volume Shadowing

Shadow sets created by OpenVMS Volume Shadowing Phase II may be used as members of RAID0 arrays with the following requirements:

- OpenVMS Volume Shadowing Phase I shadow sets are not supported. RAID0 arrays must consist of Phase II shadow sets.
- A separate license is required for using OpenVMS Volume Shadowing. The HP RAID Software for OpenVMS license does not include a license for OpenVMS Volume Shadowing.

- The disks within a shadow set must appear to Volume Shadowing as having compatible device type and geometry (compatibility is defined in the Release Notes for the version of Volume Shadowing you are running). However, different shadow sets within a RAID0 array may contain different device types. See Figure 6–3.
- Any physical disk supported both by Volume Shadowing and by the versions of OpenVMS supported by HP RAID Software for OpenVMS may be used.

## 6.4 Device Hierarchy in a RAID0 Array with Volume Shadowing

The user of a RAID0 array performs I/O to the RAID0 virtual device. In Figure 6–3, the RAID0 virtual device is named DPA137. DPA is the device-type, 137 is the unit number.

The RAID0 virtual device is mapped onto three RAID0 array members. The HP RAID software performs I/O to the RAID0 array members. In Figure 6–3, these three array members are known as shadow set **virtual units** DSA6000, DSA6001, and DSA6002. DSA is the device-type, and 6000 is the (virtual) unit number.

Other software in the OpenVMS operating system directs RAID0 array I/O to the appropriate physical devices. In Figure 6–3, I/O directed to the RAID0 array member DSA6000 is directed by OpenVMS software to the physical devices DUA1 and DUA2. These physical devices are the shadow set members of the virtual unit, and this virtual unit is presented to the HP RAID software as DSA6000.

---

### Note

---

RAID0 array members do not have to be shadow sets. But in a RAID0 array, either all members are shadow sets or no members are shadow sets. A RAID0 array may not be a member of a shadow set.

---

## 6.5 Related Documentation

For further information related to the use of shadow sets, refer to the *Volume Shadowing for OpenVMS* Manual.

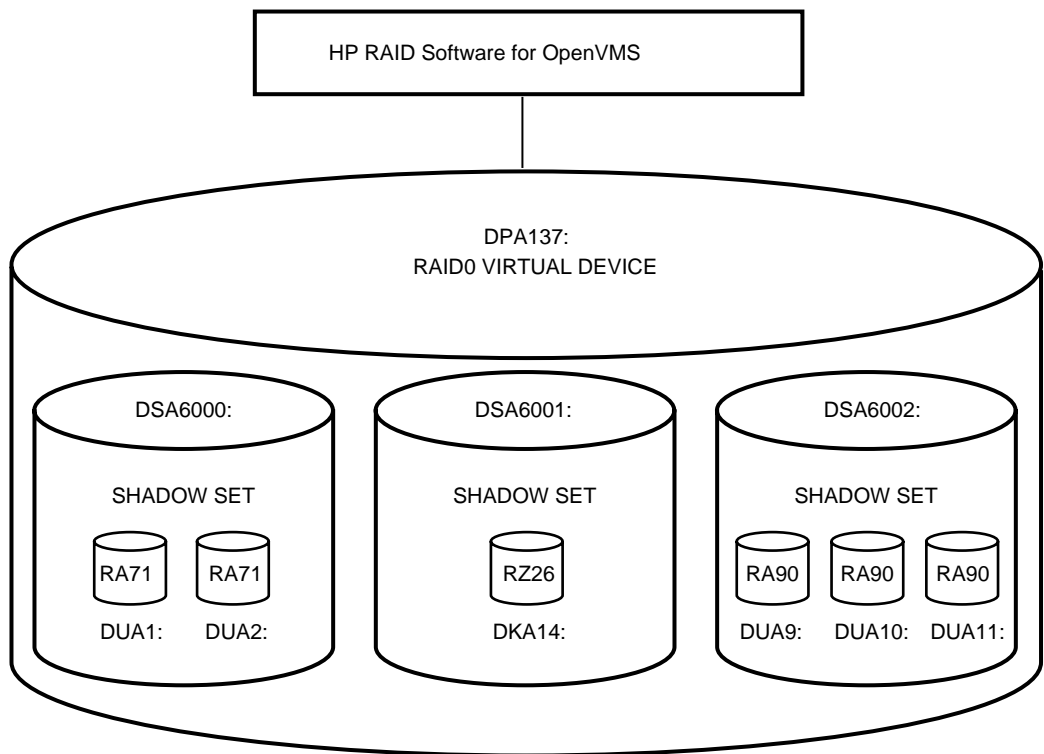
## 6.6 Creating RAID0 Arrays with Shadow Sets

This section describes the activities involved in creating RAID0 arrays with shadow sets. The topics discussed are:

- RAID initializing a RAID0 array using shadow sets
- Creating RAID0 virtual devices
- Assigning DSA device names
- Initializing and mounting the RAID0 virtual device



Figure 6–3 RAID0 Array with a Different Number of Shadow Set Members



CXO-4069A-MC

### 6.6.1 RAID Initializing a RAID0 Array Using Shadow Sets

To create a RAID0 array of shadow sets, use the RAID INITIALIZE command, and specify only one disk in each potential shadow set. Once a RAID0 array has been initialized, the number of array members cannot be changed without a RAID reinitialization of the array. Once the array is bound, additional shadow set members may be added to each shadow set. The following command sequence will initialize the RAID0 array shown in Figure 6–4.

```
$ RAID INITIALIZE/RAID_LEVEL=0 PAYROLL DUA1:,DKA14:,DUA10:
```

### 6.6.2 Creating RAID0 Virtual Devices

After the RAID0 array has been initialized it may be bound, the RAID0 virtual device may be created, and the RAID0 array may be associated with the RAID0 virtual device. This sequence of events is accomplished by one RAID BIND command, as follows.

```
$ RAID BIND/SHADOW PAYROLL DUA1:,DKA14:,DUA10: DPA137:
```

The RIND BIND creates single member shadow sets, using the specified devices, and associates the RAID0 virtual device DPA137: to the RAID0 array PAYROLL. Table 6–2 lists the physical disks, device names, and their associated shadow set virtual units as shown in Figure 6–4.

**Table 6–2 Resulting Shadow Sets and Device Names**

Shadow Set Virtual Units	Physical Device Names
DSA6000:	DUA1
DSA6001:	DKA14
DSA6002:	DUA10

### 6.6.3 How DSA Device Names Are Assigned

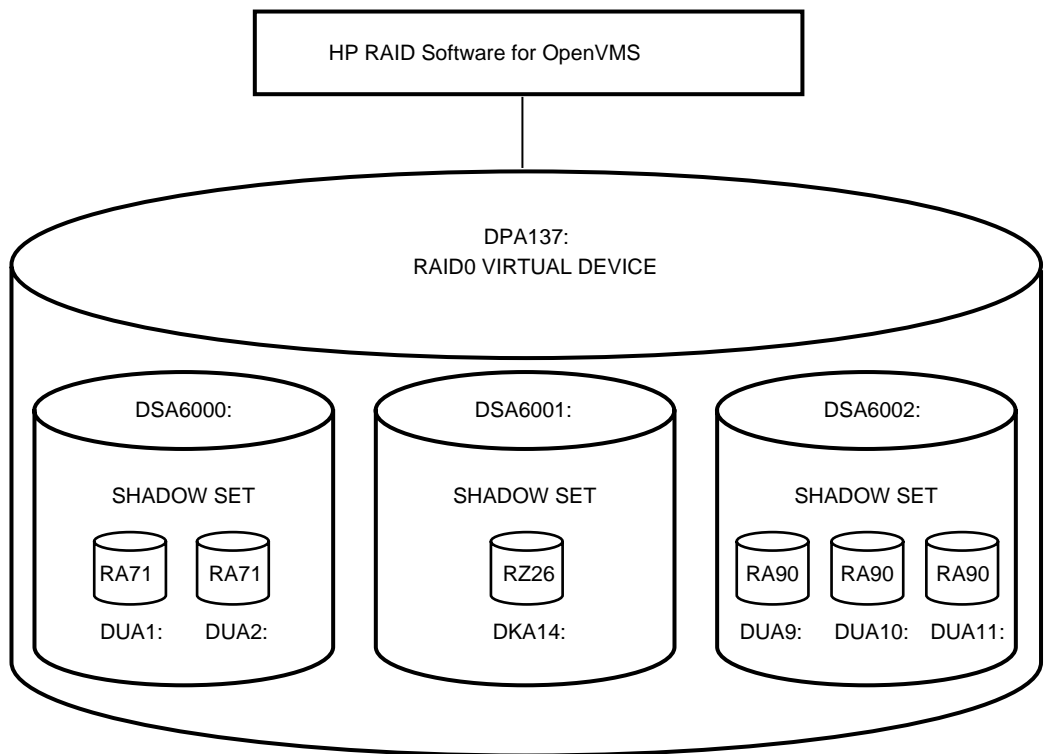
By default, HP RAID Software for OpenVMS will assign DSA device numbers starting at DSA6000. To override the default DSA device number assignment, use the /USE\_SHADOW\_DEVICES qualifier. (See the BIND command in Chapter 11.)

### 6.6.4 Initializing and Mounting the RAID0 Virtual Device

After a RAID0 array has been bound and a RAID0 virtual device has been associated with it, the virtual device may be used like any other disk device. It may be initialized and mounted as a Files–11 or as a foreign disk.

```
$ INITIALIZE DPA137 TEST
$ MOUNT DPA137 TEST
```

Figure 6–4 Sample RAID0 Array Using Shadow Sets



CXO-4069A-MC

## 6.7 Adding Shadow Set Member Devices to a RAID0 Shadow Set

Anytime after the RAID0 array has been bound, additional shadow members may be added to each shadow set with the RAID ADD command. This initiates a full shadow copy with the specified unit as the target.

The following example shows how devices may be added to shadow sets to create the RAID0 array shown in Figure 6–4.

```
$ RAID ADD/SHADOW_MEMBER/DEVICE=DSA6000: PAYROLL DUA2:
$ RAID ADD/SHADOW_MEMBER/DEVICE=DSA6002: PAYROLL DUA9:
$ RAID ADD/SHADOW_MEMBER/DEVICE=DSA6002: PAYROLL DUA11:
```

RAID0 members are automatically mounted by HP RAID Software for OpenVMS during a RAID BIND command and kept in a mounted state until the array is unbound. If additional disks are to be added to shadow sets of the array, **HP recommends that the RAID ADD/SHADOW\_MEMBER command be used instead of the MOUNT/SHADOW command.** This will immediately inform HP RAID Software for OpenVMS of the user's intent so that the new configuration may be re-created at the next RAID BIND.

**If the MOUNT command must be used, the /CLUSTER qualifier must be specified.**

As an alternative to the /DEVICE qualifier, the /INDEX qualifier may be used. The index positions may be obtained from the RAID SHOW command. See Chapter 11.

## 6.8 Removing Members from a Shadow Set

Shadow members may be removed from a RAID0 shadow set as long as one member remains in the shadow set. That is, if a shadow set has three members, two may be removed. If it has only one member, that member cannot be removed.

To remove a member from a RAID0 shadow set, use the RAID REMOVE /SHADOW\_MEMBER command. As an example, the following command removes the DUA11 device from shadow set DSA6002.

```
$ RAID REMOVE/SHADOW_MEMBER PAYROLL DUA11:
```

This results in DSA6002 being left with two shadow members, DUA9 and DUA10.

After a device has been mounted into a shadow set, whether with MOUNT /SHADOW or RAID ADD/SHADOW\_MEMBER, it may be removed using the RAID REMOVE/SHADOW\_MEMBER or the DCL command DISMOUNT. **HP recommends that the RAID REMOVE/SHADOW\_MEMBER be used to remove shadow set members from RAID0 arrays rather than the DISMOUNT command.**

## 6.9 Rebinding with Shadowing

HP RAID Software for OpenVMS records shadow set configurations in the on-disk information. This information is updated whenever a shadow set membership change occurs. When an array is rebound, it will be re-created with the same configuration, using the on-disk information.

If you want to rebind a RAID array containing shadow sets, either of the following commands may be used.

```
$ RAID BIND PAYROLL DUA1:,DKA14:,DUA10: DPA137:
```

```
$ RAID BIND/SHADOW PAYROLL DUA1:,DKA14:,DUA10: DPA137:
```

It is not necessary to include the /SHADOW qualifier because the BIND command reads the on-disk information that contains the last state of the members. The BIND command remounts the shadow sets using the on-disk information in the given devices and associates the RAID0 virtual device DPA137: to the RAID0 array PAYROLL. All shadow sets are re-created as they existed when the RAID0 array was unbound. That is, all shadow set members are mounted into the shadow sets, not just the three devices specified on the RAID BIND command line.

### 6.9.1 Converting a Nonshadowed RAID0 Array to a Shadowed RAID0 Array

The /SHADOW qualifier is required if you are rebinding a RAID array that was not previously shadowed. The command line is as follows:

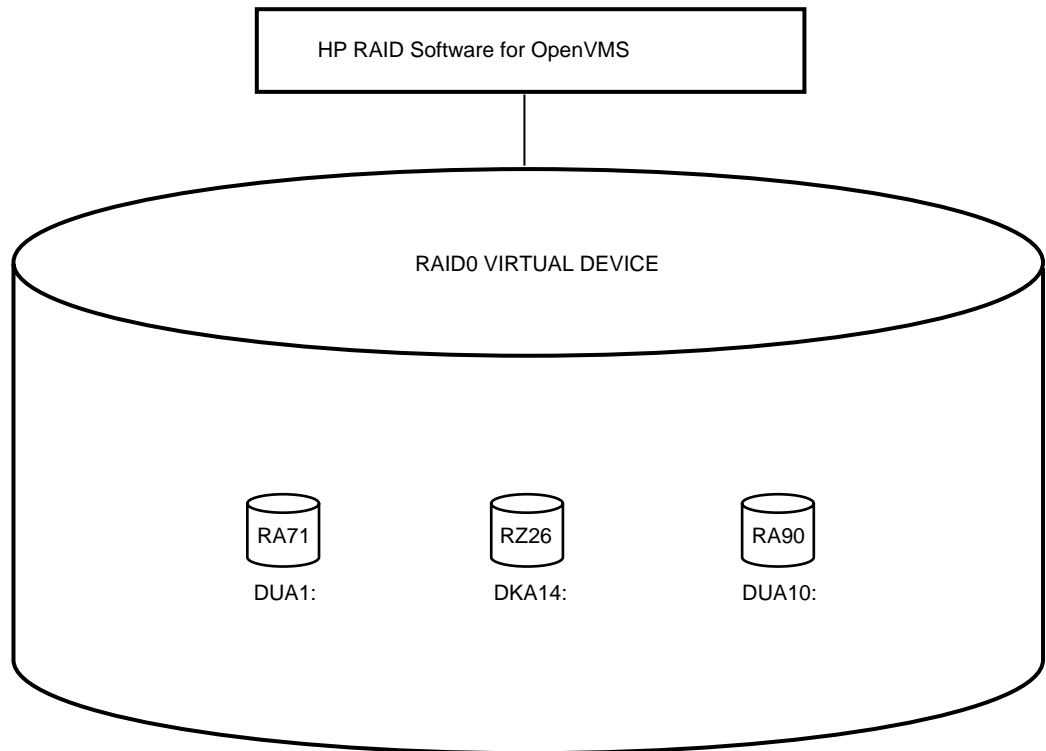
```
$ RAID BIND/SHADOW PAYROLL DUA1:,DKA14:,DUA10: DPA137:
```

This command creates single member shadow sets using the given devices and associates the RAID0 virtual device DPA137: to the RAID0 array PAYROLL. Note that the RAID INITIALIZE command is not needed. After the RAID BIND /SHADOW command is executed, the RAID array members are single member shadow sets.

## 6.10 Rebinding Without Shadowing

Use the RAID BIND/NOSHADOW command if you want to reconfigure RAID0 arrays originally composed of shadow sets so that the new array does not have shadow sets. As an example, suppose the RAID0 array shown in Figure 6-4 is being changed to conform to Figure 6-5.

Figure 6-5 RAID0 Array with No Shadow Set Members



CXO-4070A-MC

Use the following RAID BIND command to create this configuration. No shadow sets will be created.

```
$ RAID BIND/NOSHADOW PAYROLL DUA1:,DKA14:,DUA10: DPA137:
```

Note that a RAID INITIALIZE command is not needed. However, after the RAID BIND/NOSHADOW has been executed, the former shadow set members which are not part of the resulting RAID0 array (in this example, DUA2:, DUA9: and DUA11: are no longer in array PAYROLL), cannot be bound into this or another RAID0 array in this VMScluster system unless they are reinitialized with RAID INITIALIZE or they are added to an array composed of shadow sets with RAID ADD/SHADOW\_MEMBER.

Alternatively, if you wish to create two RAID0 arrays from one original shadowed array, use the RAID CLONE command as described in Section 6.12.

## 6.11 Backup of Arrays with Shadow Sets

RAID virtual devices may be backed up with OpenVMS BACKUP. All the files on the RAID virtual device must be closed prior to using OpenVMS BACKUP. One way to assure this is to dismount the DPA virtual device, back up the RAID array, and remount the DPA virtual device. However, this will make the DPA device unavailable for the duration of the backup operation.

## 6.12 RAID CLONE Command

An alternative method for backing up RAID0 arrays whose members are multi-member shadow sets is to use the RAID CLONE command. This command removes one member from each shadow set and configures the removed members into a new RAID array, which you may then BIND and use for OpenVMS BACKUP. The original RAID array is available throughout this procedure and does not need to be unbound. However, to assure consistency of the user's data (such as to prevent the "snapshot" of the array from containing only a part of a multi-I/O data update), HP recommends that the DPA virtual device be dismounted before entering a RAID CLONE command.

The qualifier /OVERRIDE=CHECK must be used with the RAID CLONE command if the associated DPA device will remain mounted during the execution of this command. If the DPA device is mounted and this qualifier is not used, the RAID CLONE command will fail. If the qualifier is used, the RAID CLONE command will be allowed to proceed, but an informational message is displayed to inform you that the new RAID array may not contain application-consistent data.

The following sample procedure demonstrates how to prepare to use the BACKUP Utility on a RAID0 array. The RAID CLONE command is used on the RAID0 array shown in Figure 6-6 to create the configuration shown in Figure 6-7.

1. Bring the application to a quiescent point. If possible, dismount the RAID virtual device using the DISMOUNT command, as follows:

```
$ DISMOUNT DPA9:
```

2. Use the RAID CLONE command to "clone" the array. This creates two separate arrays with identical user data, with the clone array using one shadow member from each shadow set.

```
-----  
$ RAID CLONE PAYROLL BACKUP_PAYROLL  
%RAID-I-CLONEMEMBER, device_$_3$DUA143: is now a member of cloned  
raid array BACKUP_PAYROLL  
%RAID-I-CLONEMEMBER, device_$_3$DUA145: is now a member of cloned  
raid array BACKUP_PAYROLL  
$ SHOW LOGICAL RAID$CLONE_MEMBERS  
  "RAID$CLONE_MEMBERS" = "_$_3$DUA143:" (LNM$PROCESS_TABLE)  
    = "_$_3$DUA145:"
```

The members of the original array PAYROLL continue to be shadow sets, but each shadow set now contains only a single disk drive. The new array BACKUP\_PAYROLL has two unshadowed members, but must be bound with RAID BIND before it can be used, as shown below.

3. Resume the application after remounting the RAID0 virtual device.

```
$ MOUNT/SYSTEM DPA9: PAYROLL
```

4. Bind the cloned RAID array using the BIND command.

```
$ RAID BIND BACKUP_PAYROLL 'F$LOGICAL("RAID$CLONE_MEMBERS")' DPA25:
%RAID-I-ISBOUND, unit _$3$DUA143: is bound as a member of RAID
array BACKUP_PAYROLL
%RAID-I-ISBOUND, unit _$3$DUA145: is bound as a member of RAID
array BACKUP_PAYROLL
%RAID-I-VUCREATE, virtual unit DPA25: was created for partition 1
on BACKUP_PAYROLL
```

Before the cloned array can be used, it must be bound and a DPA device created. In this example the translation of logical name "RAID\$CLONE\_MEMBERS" is used to specify the first disk device of the cloned array. But you could also use the output from the RAID CLONE command to specify one or more disk devices that are separated from the original RAID array.

5. Perform a backup on the cloned copy of the RAID array.

```
$ MOUNT DPA25: PAYROLL
$ BACKUP/IMAGE DPA25: MUA0:PAYROLL.BCK
$ DISMOUNT DPA25:
```

The virtual device for the cloned array must be mounted privately.

6. After the BACKUP operation is complete, you may add the disks back into the original array using the RAID ADD/SHADOW\_MEMBER command. This initiates a full shadow copy with the specified unit as the target.

```
$ RAID UNBIND BACKUP_PAYROLL
$ RAID ADD/SHADOW_MEMBER/DEVICE=DSA6000 $3$DUA30:
$ RAID ADD/SHADOW_MEMBER/DEVICE=DSA6001 $14$DKA66:
```

### 6.12.1 Support for Shadow Minicopy

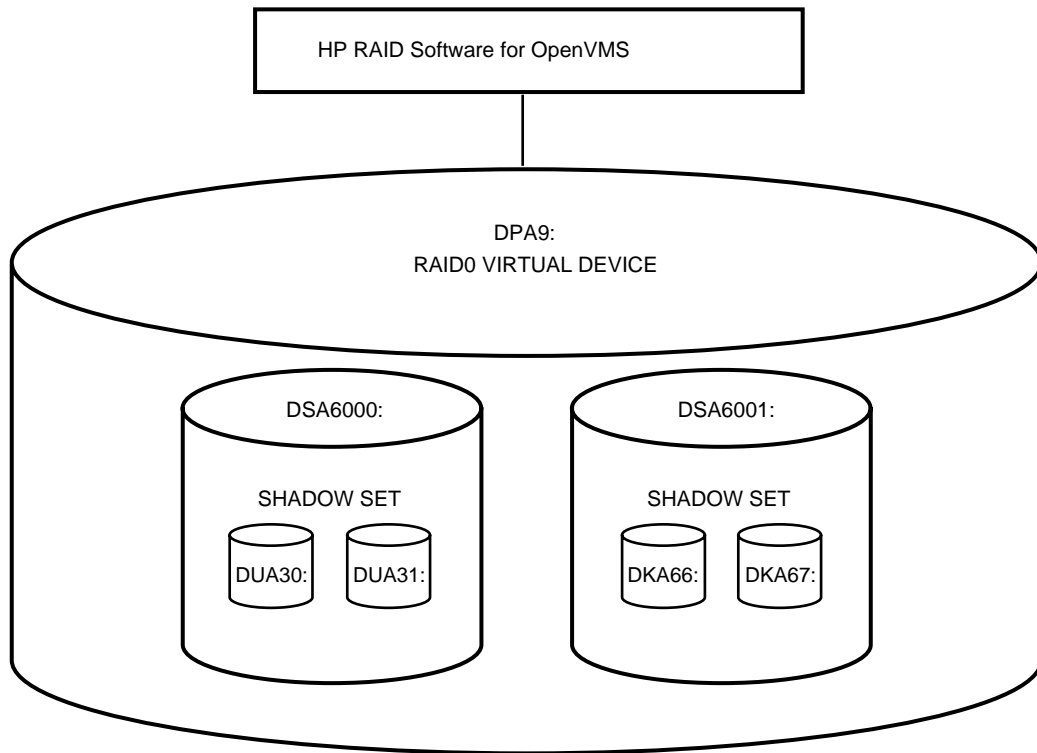
Newer versions of Volume Shadowing for OpenVMS support so called minicopies. When adding a shadow set member back into the set only the modified pages will be copied from the shadow master to the new shadow member thus reducing copy time. For this to happen the new shadow member has to be used in read-only mode while outside of the shadow set.

RAID Software supports this shadowing feature by making a copy of the shadow context when executing the "RAID CLONE/POLICY=MINICOPY" command. When binding the clone members the virtual RAID disk units are created write locked by default. When adding the physical devices back to the original RAID array the "RAID ADD/SHADOWING" command restores the shadow context. Now the shadowing software can add the former shadow set member with a minicopy operation instead of a full copy.

If a cloned array is bound using the "RAID BIND/WRITE" command the saved shadow context will be erased and the virtual units will be created writeable. When adding the physical devices back to the original RAID array the "RAID ADD/SHADOWING" command initiates a full shadow copy.

See OpenVMS Release Notes under Volume Shadowing for more information.

Figure 6–6 RAID0 Array With Two Shadow Sets



CXO-4071A-MC

### 6.13 Using RAID0 and Shadowing in a VMSccluster

When shadow set virtual units are created by HP RAID Software for OpenVMS as a result of a RAID BIND/SHADOW command, the virtual units are automatically mounted on every node of the VMSccluster system that is running HP RAID Software for OpenVMS using the MOUNT/SYSTEM command.

---

**Note**

Volume Shadowing Phase II must be enabled on all nodes in the cluster if you want to build a RAID0 array with shadow set virtual units.

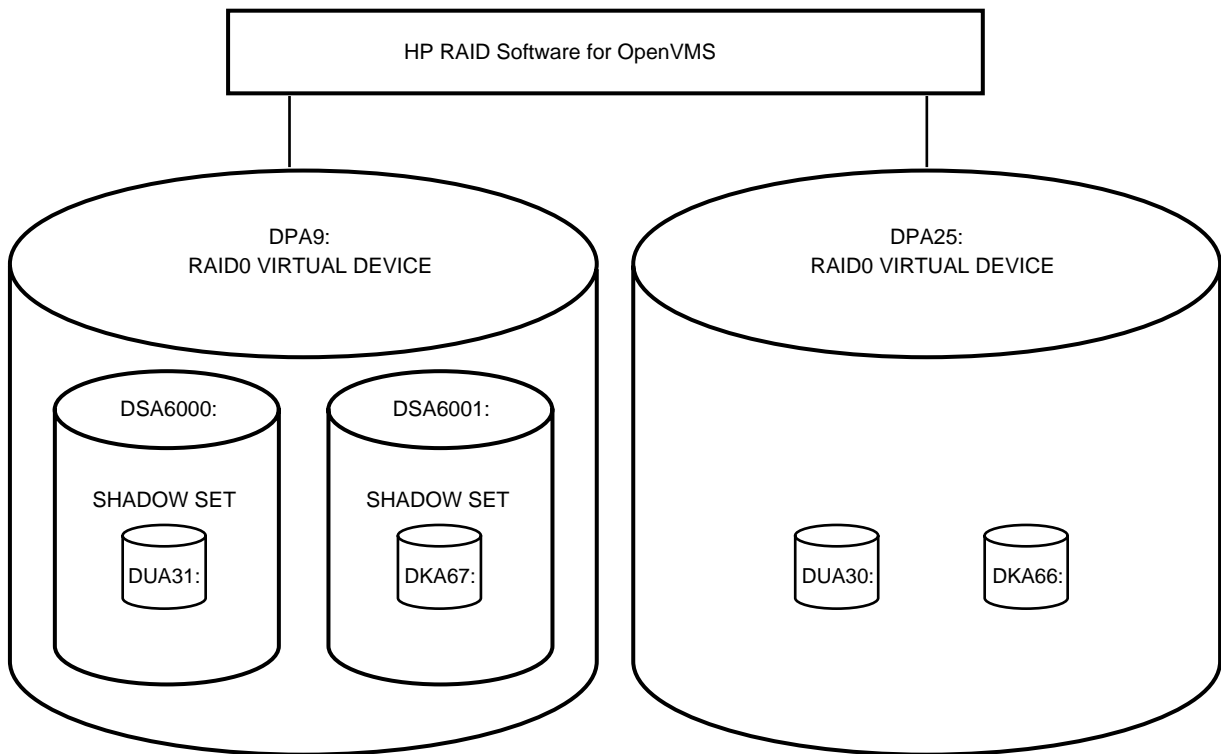
---

Shadow set virtual units that are members of RAID0 arrays may not be MSCP served to other nodes in the VMSccluster system. The physical devices comprising the shadow set virtual units may be MSCP served.

The RAID BIND/SHADOW, the RAID ADD/SHADOW\_MEMBER, the RAID REMOVE/SHADOW\_MEMBER, and the RAID CLONE commands automatically take effect on all other nodes that are running HP RAID Software for OpenVMS.



Figure 6–7 Two RAID0 Arrays



CXO-4072A-MC

## 6.14 Performance Considerations

RAID0 arrays with shadow sets as members provide a high performance and high reliability RAID solution. Optimal performance requires careful choice of hardware adapters, buses, controllers and physical disks. For additional information on performance considerations see Chapter 9.

## 6.15 Error Handling

Without shadowing, when any member of a RAID0 array becomes Inaccessible, the entire RAID0 virtual device becomes Inoperative and user data cannot be accessed. With shadowing, for a RAID member to become Inaccessible, all the shadow set members of that set have to be Inaccessible, which is unlikely to occur.

Errors that are detected by HP RAID Software for OpenVMS when accessing the RAID0 array or any of its shadowed or non-shadowed members are logged in the system error log. RAID array membership changes are also logged, for example, as a result of RAID ADD/ SHADOW\_MEMBER and RAID REMOVE/SHADOW\_MEMBER. The RAID ANALYZE/ERROR\_LOG command may be used to format and display these error logs. Because HP RAID Software for OpenVMS is layered on top of OpenVMS Volume Shadowing, errors detected by Volume Shadowing that are passed to HP RAID Software for OpenVMS may be logged by both the Volume Shadowing software and HP RAID Software for OpenVMS.

For more information on error handling by HP RAID Software for OpenVMS, see Chapter 7. For information on the RAID ANALYZE/ERRORLOG command, see Chapter 11.

---

## Internal Operation and Error Handling

This chapter provides information on the following:

- The various states of a RAID0 and RAID5 arrays
- How the HP RAID Software for OpenVMS handles error recovery
- A discussion of VMScluster system state transitions

Whenever there is no `RAID0` or `RAID5` icon on a papagraph, assume that the information applies to both RAID0 and RAID5 arrays. Table 7–1 provides a list of topics in this chapter.

**Table 7–1 Topics in This Chapter**

Subject	Section
RAID Array States	7.1
RAID Array Status	7.2
RAID Array Member States	7.3
RAID Virtual Unit Status	7.4
RAID Shadow Set States	7.5
HP RAID Error Recovery	7.6
VMScluster State Transitions	7.7

### 7.1 RAID Array States

RAID5 arrays can be in one of three operational states and RAID0 arrays in one of these. These states are as follow:

- Normal state
- Reduced state `RAID5`
- Reconstructing state `RAID5`

The RAID SHOW command will display these states. While in Normal state, Reduced state, or Reconstructing state, the array can have an Inoperative status. See Section 7.2.2 for further information on the Inoperative status.

These states are discussed in the following sections.

### 7.1.1 Normal State

In the Normal state, all members of the RAID array are present.

#### Read Operations

**RAID5** Read operations touch data blocks, not parity blocks. There is a performance increase relative to disks not using the RAID5 technology. This performance increase is due to load balancing that results from data being distributed across the members.

**RAID0** The read operation for RAID0 arrays is similar that for RAID5 arrays except that RAID0 arrays have no parity blocks.

#### Write Operations

**RAID5** Write operations of RAID5 arrays require an update of both data and parity blocks. The parity block updates decrease write performance.

**RAID0** Write operations for RAID0 arrays only require the update of the data blocks.

### 7.1.2 Reduced State **RAID5**

A RAID array is in a Reduced state when one of its members is missing. RAID arrays become reduced due to one of the following events:

- Automatic removal of a member by the HP RAID software. This is usually due to a global error. For information on global errors, see Section 7.6.4.
- When a RAID REMOVE command is entered to manually remove a member. Note that you cannot remove a member from a RAID array that is already reduced.

#### Read Operations **RAID5**

With read operations that would normally access the missing member, data is regenerated from the remaining RAID array members. There is a performance impact relative to disks not using RAID5 technology due to additional member read operations that take place during the data regeneration.

#### Write Operations **RAID5**

With write operations, data is written to the remaining RAID array members when possible, and parity is updated to be consistent with the new data. Performance is similar to a RAID array in the Normal state.

### 7.1.3 Reconstructing State **RAID5**

A reduced RAID array enters the Reconstructing state when a spare has been added to the RAID array to replace a missing member. A missing member is replaced by a spare in either of the following cases:

- Automatically by the software when a spareset is associated with the RAID array.
- When a RAID REPLACE command is entered.

HP RAID software regenerates all data and parity onto the replacement member. This activity is executed by a single node in a VMSccluster system chosen by the HP RAID software. If that node fails or the reconstruction activity becomes blocked on that node (due to loss of the access path to a member of the RAID array), the reconstruction will be continued on another licensed node in the VMSccluster. Each node in a VMSccluster system can perform reconstruct activity

on one RAID array at a time. Data is available to your application throughout the member reconstruction process.

---

**Note**

---

Because reconstruction uses system resources, you may want to prevent some nodes in the cluster from performing reconstruction by using the RAID SET/NORECONSTRUCT command and qualifier.

---

### Read Operations RAID5

If data has already been reconstructed by the HP RAID software, read operations are handled in the same manner as reads to a RAID array in the Normal state. If data has not yet been reconstructed by the software, data is regenerated for the user and is also written to the replacement member. Read performance is similar to disks not using the RAID5 technology.

### Write Operations RAID5

With write operations, data is written to the replacement member and parity is updated. There is a performance impact relative to disks not using the RAID 5 technology due to parity updates.

## 7.2 RAID Array Status

RAID arrays have two kinds of status conditions, as follows:

- Startup status
- Inoperative status

These conditions are discussed in the following sections.

### 7.2.1 Startup Status

The STARTUP status occurs when a new node enters a VMScluster system and RAID\$STARTUP has been executed. When the new node receives information from an existing node in the cluster about a currently bound RAID array, it tries to bind the new array. The RAID array that the new node is attempting to bind will be in the Startup status for that new node until the RAID array is bound. The array will continue to be in the Normal (or Reduced or Reconstructing) state for other nodes in the VMScluster system just as it was prior to the new node entering the cluster. While in the Startup status, data on the RAID array is not available to that node.

If the new node is unable to mount one or more member devices in the RAID array, one of the following will occur:

- RAID5 If the new node cannot mount one member, and the RAID array is normal (not reduced or reconstructing), the HP RAID software will reduce the RAID array. The array will now be reduced for all nodes.
- RAID5 If the new node cannot mount one member, and that member is reconstructing (the array is in the Reconstructing state), then the HP RAID software will reduce the RAID array. The array will now be reduced for all nodes.

- **[RAID5]** If the new node cannot mount two or more members, or if the RAID array is already reduced, the RAID array will stay in the Startup status for the new node. The array will continue to operate in the Normal (or Reduced) state for all other nodes on the VMScluster system. The new node will continue to retry the bind periodically until successful.
- **[RAID0]** If a RAID0 array cannot mount all the members, it will stay in the STARTUP status.

While a RAID array is in a STARTUP status, you may only enter RAID SHOW and RAID UNBIND commands to the RAID array *from the node which shows the RAID array to be in the STARTUP status*. Other nodes in the VMScluster system can continue to access the RAID array normally.

## 7.2.2 RAID Array Inoperative Status

**[RAID0]** A RAID0 array will enter the Inoperative status if one or more members are unavailable.

**[RAID5]** The array will enter the Inoperative status if two or more members of an array are unavailable. With less than  $n-1$  members, your data would not be available anyway. It does not help to reduce the array at this point because you must wait for enough members to continue operation.

**[RAID5]** There are cases where it is likely you will lose availability of more than one member temporarily, such as during an **HSC** controller crash. In such an event, members are expected to return when the HSC controller reboots. When enough members return and the total reaches  $n-1$ , array operation can continue. At this time the array becomes operative and the user data is again available.

When transitioning from an Inoperative to Operative status, a delay is introduced by the software. This is to accommodate recoveries from a controller crash and reboot where individual members become available in a staggered fashion. Under these cases, it would not be the appropriate action to reduce or declare the array Inoperative as soon as  $n-1$  members are available. In other words, where a member is truly lost and never becomes available after a delay, the software will automatically make the determination to reduce the array, or make the array Inoperative, if a RAID0 array.

This Inoperative status is displayed by the RAID SHOW command. The Inoperative status will appear in addition to the appropriate RAID array states for as long as the array is Inoperative.

## 7.3 RAID Array Member States

Like RAID arrays, RAID array *members* can be in one of four states as a part of normal operation. These states are as follow:

- **Normal state**—The drive is functioning normally.
- **Missing state** **[RAID5]**—The drive is missing from the RAID array. Its status is displayed when a RAID SHOW command is entered because you might not have been aware which member was removed or even that a member was removed.
- **Reconstructing state** **[RAID5]**—The drive is a replacement drive for a missing member and is in the process of being reconstructed.

- **Mounting state**—When a RAID array is in a Startup status, any members that have not yet been mounted are in the Mounting state. The Mounting state also applies to spares. Spares get mounted automatically on a new node when the node enters the cluster. Until spares are mounted, they are in the Mounting state.

## 7.4 RAID Virtual Unit Status

The RAID Virtual Unit can be in one of the following two status conditions:

- Accessible status
- Inaccessible status
- Startup status

These two conditions are discussed in the following sections.

### 7.4.1 Accessible Status

When a virtual unit is in an Accessible status condition, it indicates that normal operation can occur on the virtual unit on a per node basis.

### 7.4.2 Startup Status

In the event of problems with creating the RAID virtual device, the RAID SHOW command will report the RAID virtual device with a status of STARTUP. The status will transition to ACCESS/INACCESS when the error condition is resolved.

## 7.5 RAID Shadow Set States RAID0

The RAID SHOW command provides information about shadow sets in RAID arrays. This information is based on the state of the shadow set as maintained by Volume Shadowing for OpenVMS software. Refer to the *Volume Shadowing for OpenVMS* manual.

RAID shadow sets can be in one of the four following operational states:

- SteadyState state
- ShadowMerging state
- ShadowCopying state
- Unknown state

These four states are discussed in the following sections.

### 7.5.1 SteadyState State RAID0

If the shadow set is shown as **SteadyState**, all members of the shadow set are fully operational and all data is redundant across members of the shadow set. No copy or merge operations are in progress.

### 7.5.2 ShadowMerging State RAID0

If the shadow set is shown as ShadowMerging, all members of the shadow set are fully operational and data is being made redundant across members of the shadow set. Special algorithms for read and write operations are used by Volume Shadowing for OpenVMS to ensure data integrity for user operations. Merge operations occur either when a node with the shadow set mounted crashes or when a shadow set virtual unit is improperly dismounted and then remounted.

### 7.5.3 ShadowCopying State RAID0

If the shadow set is shown as ShadowCopying, at least one member of the shadow set is not fully operational yet. At least one member of the shadow set is the target for a full copy operation. This copy operation is intended to bring the copy target to full parity with the other operational members in the shadow set.

---

**Note**

---

A shadow set may be in both the copying and merging states. If a shadow set is in both of these states, the RAID SHOW command will display the state as ShadowCopying. This indicates that a full copy operation must complete before any pending merge operation begins.

---

To determine which member of the shadow set is the full copy target, use the DCL SHOW DEVICE DSAxxxx command. Refer to the *Volume Shadowing for OpenVMS* manual.

### 7.5.4 Unknown State RAID0

If the shadow set is shown as Unknown, at least one member of the shadow set is not yet mounted on this node and its state cannot be determined from the Volume Shadowing for OpenVMS software.

## 7.6 HP RAID Error Recovery

RAID5 HP RAID software detects and recovers from member device errors and access path errors while maintaining continuous access to user data and protecting the integrity of that data.

HP RAID software learns about member errors in one of two ways:

- An error is reported by the member's device driver to the HP RAID software during an access of data on that member.
- The HP RAID software timed out an access to an array member. For more information on the timeout mechanism, see Section 7.6.2.

### 7.6.1 Member Error Handling

Errors reported to the HP RAID software from the member's device driver during accesses of data on that member are classified as one of the following:

- Potentially recoverable errors
- Localized errors
- Global errors

#### **Potentially Recoverable Errors**

These are errors for which the HP RAID software may invoke OpenVMS mount verification processing on the member and retry the failed member I/O operation in an attempt to recover from the failure. Examples of this kind of error are member device offline errors or write lock errors.

#### **Localized Errors**

These errors suggest that there is an unreadable block on the member device. An example of a localized error is a parity error.



### Global Errors

These errors indicate that the member device is unavailable due to an unrecoverable problem with the entire device or the access path to that device. Examples of this kind of error are as follows:

- A member device offline error that underwent OpenVMS mount verification processing and retries and was not recovered
- A controller error

### 7.6.2 Timeout Mechanism RAID5

The RAID software times I/O accesses to member devices. This timed interval includes any OpenVMS mount verification processing and retries that are done during processing of a potentially recoverable error that occurs during a member I/O access.

If a member I/O (and any associated OpenVMS mount verification processing or retries) does not complete within the timeout interval, the HP RAID software will handle this member I/O timeout as a global error. This may cause removal of the member and recovery of the I/O operation, as described in Section 7.6.4.

---

#### Note

---

OpenVMS mount verification processing may continue on a member device even after the HP RAID software has timed out a member I/O operation and begun global error processing for that I/O.

---

You may set the timeout interval used by the HP RAID software by using the timeout qualifier, /TIMEOUT, with the RAID INITIALIZE, RAID BIND, and RAID MODIFY commands.

The use of a timeout mechanism by the HP RAID software provides you with a trade-off between data availability and data reliability (redundancy) when a problem occurs accessing a member device.

Following are three timeout mechanisms you can use.

#### Low Timeout Value RAID5

A low timeout value provides higher data availability. If the HP RAID software encounters a problem trying to perform an I/O operation to a member device, a low timeout value causes the software to treat the member I/O as having failed in a much shorter time period, thus improving the response time of the RAID array. The drawback of a low timeout value is that a member access problem will cause the software to remove a member from a RAID array in cases where the member may still recover in time. The cost is lost redundancy and the time and resources to do a reconstruct.

#### High Timeout Value RAID5

A high timeout value provides better data reliability by attempting to preserve the membership, and thus the redundancy, of a RAID array in the event of a member access problem. If the HP RAID software encounters a problem trying to perform an I/O operation to a member device, a high timeout value causes the software to give the device and/or its access path more time to recover at the expense of a slower I/O response time from the RAID virtual device. The cost is longer I/O response time in the face of temporary member failures.

**NOIO\_TIMEOUT** RAID5

If the `/NOIO_TIMEOUT` is specified, the RAID driver avoids the time overhead associated with error handling where a member's drive does not respond in a timely fashion. The overhead avoided involves buffer copies that consume CPU cycles and memory. Turning off `IO_TIMEOUT` results in better performance, but could result in a much longer delays (it could be hours) if a member device has a certain class of problems. You would use this qualifier if you want to increase performance and are willing to sacrifice some availability in abnormal circumstances. The `RAID SHOW/FULL` command will display the state of this setting.

The default for this qualifier is to perform the buffer copies. Also, the `/NOTIMEOUT` qualifier overrides the `IO_TIMEOUT` qualifier. In other words, if you turn off member timeouts, buffer copies are also effectively disabled as a result regardless of the `/NOIO_TIMEOUT` specified (or defaulted). A message is displayed when this override occurs.

### 7.6.3 Localized Error Handling

Table 7–2 shows how the HP RAID software handles localized read and write errors.

**Table 7–2 Localized Read and Write Errors**

<b>Error: Member Device Read Failure</b>		
<b>If ...</b>	<b>Then ...</b>	<b>The Application Sees ...</b>
The RAID5 array state is normal,	The member data is regenerated. Regenerated data is written back to the member.	Success.
The RAID5 array state is reduced, or the RAID0 array state is normal,	Because the RAID array has no redundancy, data regeneration is not possible.	Failure.
The RAID5 array state is reconstructing,	If redundancy has been restored by reconstruct processing, data will be regenerated and written back to the member. If redundancy has not yet been restored, data regeneration is not possible.	Success if redundancy exists. Failure if redundancy does not exist.
<b>Error: Member Device Write Failure</b>		
<b>If ...</b>	<b>Then ...</b>	<b>The Application Sees ...</b>
The RAID5 array state is normal,	The block is marked invalid. New data is readable via regeneration.	Success.
The RAID5 array state is reduced, or the RAID0 array state is normal,	For RAID5 the block is marked invalid.	Failure.
The RAID5 array state is reconstructing,	The block is marked invalid. If redundancy has been restored by reconstruct processing, new data is readable via regeneration.	Success if block is readable through regeneration. Failure if block cannot be regenerated.

## 7.6.4 Handling Global Errors and Errors Accessing Control Data

Table 7–3 shows how the HP RAID software handles global read and write errors and errors accessing HP RAID control data.

**Table 7–3 Global Read and Write Errors and Errors Accessing Control Data**

<b>Error: Member Device Read Failure</b>		
<b>If ...</b>	<b>Then ...</b>	<b>The Application Sees ...</b>
The RAID5 array state is normal,	The member is removed. Data is regenerated.	Success.
The RAID5 array state is reduced, or the RAID0 array state is normal,	The RAID array becomes Inoperative until the member with an error becomes accessible.	Delay while OpenVMS mount verification processing is in progress. Mount verification will result in success or failure.
The RAID5 array state is reconstructing,	If the problem is on the reconstructing member, then the result will be the same as if the RAID array state were normal. If the problem is on a member other than the reconstructing member, the result will be the same as if the RAID array were in a reduced state.	Success or failure depending on whether or not recovery is treated as if the array state is normal or reduced.
<b>Error: Member Device Write Failure</b>		
<b>If ...</b>	<b>Then ...</b>	<b>The Application Sees ...</b>
The RAID5 array state is normal,	The member is removed. New data is readable via regeneration.	Success.
The RAID5 array state is reduced, or the RAID0 array state is normal,	RAID array becomes Inoperative until the member with the error becomes accessible.	Delay while OpenVMS mount verification processing is in progress. Mount verification will result in success or failure.
The RAID5 array state is reconstructing,	If the problem is on the reconstructing member, then the result will be the same as if the RAID array state were normal. If the problem is on a member other than the reconstructing member, the result will be the same as if the RAID array were in a reduced state.	Success or failure depending on whether or not recovery is treated as if the array state is normal or reduced.

## 7.6.5 Summary of Error Handling

HP RAID software recovers from member I/O errors whenever possible.

When the RAID array state is normal and either a global error occurs during access of data or parity on a member device or any type of error occurs accessing the RAID5 control structures on a member device, that member will be removed from the RAID array.

When the RAID array state is reduced and either a global error occurs during access of data or parity on a member device or any type of error occurs accessing the RAID5 control structures on a member device, that RAID array will be placed in an Inoperative state.

## 7.7 VMSccluster State Transitions

### Node Joining a VMSccluster System

When a node joins a cluster and the HP RAID software is started on this node, the software goes through a process to make all RAID arrays currently bound in the cluster accessible to this new node. A RAID array will enter a startup state until all members of that array become visible to that node and become consistent with the state of the array on other nodes.

### Node Leaving a VMSccluster System

When a node leaves a cluster, access to all RAID arrays from other cluster nodes continues.

**RAID5** Reconstruction activity on a node departing the cluster will be continued on another node in the cluster that is not already processing reconstruct on a different RAID array. If all nodes busy are processing reconstruct operations on other RAID arrays, the HP RAID software will wait to continue reconstruction until a node is free.

**RAID5** When the reconstruct activity is occurring, a RAID SHOW command will show a certain percentage completed. If the node doing the reconstruct leaves the cluster, then the reconstruct will continue on another node.

## 7.8 Reconstruction Determinations **RAID5**

The HP RAID software will recover from CPU failures during reconstruct activities by moving the reconstruct activity from the failing node to another node. The reconstruct activity will also dynamically move to another node when the RAID SET/NORECONSTRUCT command is issued on the reconstruction node.

The reconstruct state appears in the RAID SHOW command along with “% complete” information. When no node is performing the reconstruction, the “% complete” field will not appear. The absence of the “% complete” field may occur for the following reasons:

- The reconstruction has just completed and the array has not transitioned to the normal state.
- No nodes are available to do the reconstruct at this time due to either reconstructs are disabled or all enabled nodes already have reconstructs in progress.
- The reconstruct is in the process of moving to another node.

As the reconstruct continues on the other node, you will notice a temporary drop (0%) in the percentage completed and then a jump back to the expected percentage within minutes. The 0% indication is normal while the new node determines where to continue the reconstruct activity.

---

## RAID0: Improving Performance

This chapter describes how to improve performance for I/O intensive workloads by using the RAID0 technology features of HP RAID Software for OpenVMS.

This chapter also discusses some of the principal design choices and tradeoffs to consider when attempting to improve the performance of a RAID0 array.

Table 8–1 lists the topics covered in this chapter.

**Table 8–1 Topics in This Chapter**

Subject	Section
Benefits of RAID0 Technology	8.1
I/O Workloads That Can Benefit from RAID0 Arrays	8.2
Determining Whether RAID0 Arrays Will Improve Performance	8.3
Configuring RAID0 Arrays for Data Transfer-Intensive I/O Workloads	8.4
RAID0 Arrays and I/O Load Balancing	8.5
Configuring RAID0 Arrays for Request Rate-Intensive I/O Workloads	8.6
Tools for Performance Analysis	8.7
How RAID0 Plus Shadowing Helps Performance	8.8
Guideline for RAID0 Arrays	8.9

### 8.1 Benefits of RAID0 Technology

The purpose of RAID0 technology is to provide I/O intensive workloads with greater I/O performance from a given configuration of disks and I/O hardware than would normally be achieved by structuring and using the disks as individual OpenVMS volumes. Greater I/O performance may be delivered as one of the following:

- A higher I/O throughput (more I/O requests serviced per unit time) due to probabilistic load balancing
- A higher I/O bandwidth due to concurrent transfer of data to or from more than one disk to satisfy a single request

## 8.2 I/O Workloads That Can Benefit from RAID0 Arrays

An I/O intensive application is one whose performance is primarily determined by the rate at which I/O requests can be satisfied. RAID0 arrays improve I/O performance for most I/O intensive applications. I/O intensive workloads fall into two types:

- I/O intensive workloads with a higher data transfer rate than is available from a single disk. This is called a data transfer-intensive workload. See Section 8.4 for more information.
- I/O intensive workloads with a higher I/O request rate than is available from the single available disk. This may arise either because I/O demands are greater than can be satisfied by a single disk or because multiple disks are used in an unbalanced way. See Sections 8.5 and 8.6 for more information.

While individual processes may not exhibit the previously-mentioned characteristics, the aggregate I/O workload presented to the I/O subsystem by an intensive application most often does. Therefore, almost any intensive aggregate I/O workload does show some performance benefit with RAID0 arrays when compared to the same hardware configuration used as individual disks.

## 8.3 Determining Whether RAID0 Arrays Will Improve Performance

To determine whether using a RAID0 array is likely to have a positive effect on an I/O subsystem's performance, the storage administrator must know three things about the system's I/O workload:

- Does the workload consist of synchronous or asynchronous requests?
- Are the requests that comprise the workload sequential or random?
- Does the workload consist of large or small requests?

These terms are defined in the following sections. In practice, few I/O workloads consist exclusively of one kind of request. When assessing these three characteristics, the storage administrator should look for dominant characteristics that apply to most of a workload's requests.

### Synchronous and Asynchronous I/O Requests

A stream of I/O requests is **synchronous** if the I/O requests are issued one at a time with each one completed before the next one is issued. In contrast, a stream of I/O requests is **asynchronous** if multiple I/O requests are outstanding concurrently.

### Sequential and Random I/O Requests

A stream of I/O requests is **sequential** if consecutive I/O requests are for consecutive block address ranges on a disk. A stream of I/O requests are **random** if they do not have this property.

### Large and Small I/O Request Size

For our purposes, **small I/O requests** specify the transfer of 8k bytes or fewer. **Large I/O requests** specify a transfer size larger than 8K bytes.

Using these definitions, use Table 8-2 to decide whether the use of a RAID0 array is likely to improve performance.



**Table 8–2 RAID0 Effectiveness**

Synchronous or Asynchronous	Sequential or Random	Average I/O Request Size	Probable Effectiveness of a RAID0 Array
Synchronous	Sequential	Small	Low
Synchronous	Sequential	Large	Medium
Synchronous	Random	Small	Low
Synchronous	Random	Large	Medium
Asynchronous	Sequential	Small	Low
Asynchronous	Sequential	Large	High
Asynchronous	Random	Small	High
Asynchronous	Random	Large	High

**Sources of Information about I/O Workloads**

Application designers and developers are generally a good source of information about an application's I/O workload characteristics. Information obtained from them can be verified using one of HP's performance monitoring tools. Monitoring I/O activity also provides information about the system's overall I/O workload (including physical I/O requests made from sources other than the application's own processes). See Section 8.7 for more information on determining workload characteristics.

**Where RAID0 Arrays Don't Work Well**

According to *The RAIDBook*,<sup>1</sup> RAID0 arrays are not suitable for use in the following applications: “

- Applications which make sequential requests for small amounts of data  
These applications use most of their I/O time waiting for disks to spin, whether or not they use striped arrays as storage media. The I/O performance of such applications can often be improved by software tuning, (by raising the blocking factor on the application's heavily-used files) rather than requiring application changes.
- Applications which make synchronous random requests for small amounts of data.  
If these applications cannot be modified to make asynchronous requests, or to use a data management tool that will make asynchronous requests for them, the only way to improve their performance is to move their data to an I/O device that provides higher single-stream performance, such as a RAM disk or electronic storage device. ”

**8.4 Configuring RAID0 Arrays for Data Transfer-Intensive I/O Workloads**

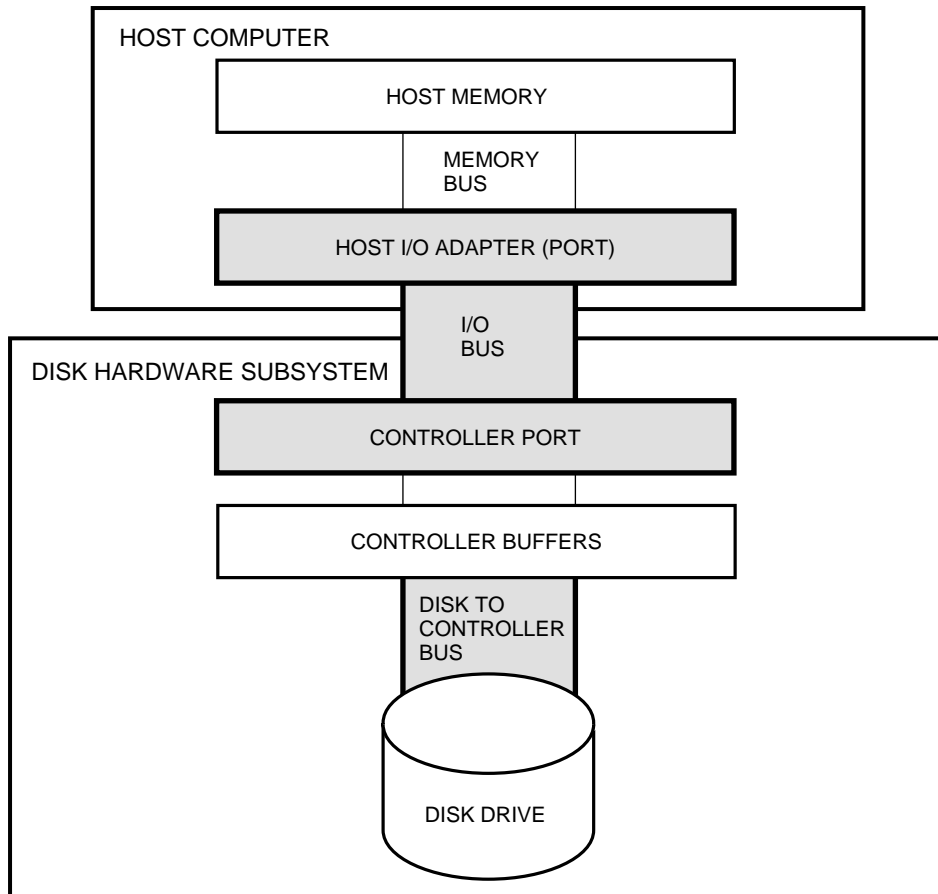
When configuring RAID0 arrays for data transfer intensive I/O workloads, the following conditions must be met:

- A RAID0 array must have sufficient disks so that the sum of their data transfer capacities equals or exceeds the required data transfer rate.

<sup>1</sup> *The RAIDBook* Published by The RAID Advisory Board, Lino Lakes, MN. First Edition June 9, 1993

- Adequate data transfer resources must exist along the entire path between each disk drive and host memory. Figure 8–1 shows this data transfer path; the components that can affect the data transfer rate are shaded.

Figure 8–1 Host Memory to Disk Drive Data Path



CXO-4140A-MC

### 8.4.1 Designing RAID0 Arrays for High Data Transfer Rates

The following procedure can be used to design a RAID0 array optimized for high data rates:

1. Determine the data transfer rate required by the application.
2. To determine the number of disks required, divide the spiral data transfer rate for the type of disk to be used into the required transfer rate and round up. The **spiral data transfer rate** is the rate at which a disk drive can continuously read or write data.

Refer to the specific product specifications to find the spiral data transfer rate for a given disk drive.

3. Configure disk-to-controller buses so that the sum of the spiral data transfer rates of array members attached to a single bus does not exceed the data transfer capacity of the bus.
4. For multidisk I/O subsystems, configure the array so the total data transfer capacity of all member disks attached to a single controller does not exceed the controller's data transfer capacity.
5. Configure I/O buses and host adapters so that the total data transfer rate of the array members attached to any single I/O bus or host adapter port does not exceed the I/O bus's or port's data transfer capacity.

#### 8.4.2 Designing Applications for Maximum Data Transfer Rate

Once a RAID0 configuration is capable of delivering the required data rate, there are three application-related design possibilities that will help get the most from RAID0:

- Use large application I/O requests to minimize the time spent processing requests as compared to the time transferring data.
- Use RMS multibuffering and/or multiblocking to ensure a steady stream of I/O requests at the array member level.
- Perform I/O sequentially, if possible, to minimize the time spent seeking.

### 8.5 RAID0 Arrays and I/O Load Balancing

Hot, or overutilized, disks are common, especially in multiuser and server systems. A disk is overutilized if there is a significant probability that an arriving I/O request must wait, or be queued, because the disk is in use executing a previous request.

A disk becomes overutilized because many concurrent requests are made for data stored on it. If a system with overutilized disks also has underutilized disks, it may be possible to organize the overutilized and underutilized disks into a RAID0 array, thereby balancing their respective workloads more evenly by reducing the probability that a request will arrive at a busy disk. This is called probabilistic load balancing.

It is possible to load balance a set of disks without RAID0 arrays. Either the OpenVMS Monitor utility or other performance analysis tools may be used to determine which of a system's disks are overutilized. Once a storage administrator knows which resources are overutilized, frequently accessed data can be moved to the underutilized disks.

Using RAID0 technology to balance the load offers two advantages over manually moving data from disk to disk:

- RAID0 technology provides automatic load balancing without continuous effort by the storage administrator. Load balancing is labor intensive, requiring both analysis by the storage administrator and modification of applications (such as, batch command file modifications to reflect changes in data file location). Moreover, load balancing must be repeated periodically because system I/O workloads can change with time. Because RAID0 technology balances I/O load by spreading accesses to all data files evenly across multiple disk drives, the I/O workload of a RAID0 array remains balanced no matter which data file is heavily utilized.

- Load balancing by RAID0 technology occurs in real-time. When a storage administrator moves data files to balance loads, past system behavior is being used to predict future behavior. While this may be accurate for stable workloads, response to workload changes occurs after the fact, and it may not be possible to respond to rapidly changing workloads at all. RAID0 technology automatically spreads I/O requests across disks as they occur.

## 8.6 Configuring RAID0 Arrays for Request Rate-Intensive I/O Workloads

To satisfy a given I/O request rate requirement, adequate request processing capacity must exist along the entire path between host memory and the array disks. I/O request processing occurs at the host I/O adapter, the controller (if one is present), and the disks.

To take full advantage of the request processing potential of a RAID0 array, an I/O workload must consist of asynchronous I/O requests so that more than one I/O request may be simultaneously outstanding. Otherwise, the RAID0 array will be unable to dispatch requests to members so that they may execute concurrently.

### Designing RAID0 Arrays for High I/O Request Rates

The general procedure for designing a RAID0 array for high request rate is as follows:

1. Determine the required request rate (I/O requests per second) and the characteristics of the I/O request workload.
2. Determine the type of disk drive that will comprise the RAID0 array and the request processing capacity under the given workload. Configure sufficient disks so the sum of their I/O request processing capacities equals or exceeds the required I/O request rate.
3. Configure sufficient controllers and/or adapters so that each is capable of satisfying the request rates that will be demanded by the member disks attached to them.
4. Tune any disk subsystem participating in the array (that is, enable the disk hardware subsystem or data manager cache, if they are available).
5. Follow the guidelines for determining chunk size in Section 8.9.3.

## 8.7 Tools for Performance Analysis

Following are three tools that can be used by the storage administrator to analyze I/O performance. Each of these tools provides some unique information about I/O performance that can help the storage administrator analyze I/O performance.

- **OpenVMS Monitor**

The OpenVMS Monitor utility samples and provides real-time displays of various aspects of system activity and resource utilization. It collects performance data in disk files for after-the-fact summarization and analysis.

The OpenVMS Monitor utility is part of the OpenVMS operating system and is available to every OpenVMS storage administrator. This product is described in the *HP OpenVMS System Management Utilities Reference Manual*.

- **The RAID SHOW/FULL Command**

The HP RAID Software for OpenVMS RAID SHOW/FULL command displays the characteristics and configuration members of the RAID array. This display includes information on your current chunk size and a histogram of array I/O request sizes. The histogram will help to determine the average I/O request size and whether there may be multiple I/O request sizes dominant.

A sample printout of a RAID SHOW/FULL command for a RAID0 array with shadowed members is shown in Example 8-1. Useful performance information for the following SHOW/FULL command report is described in Table 8-3.

### Example 8-1 Sample RAID SHOW/FULL Command Report

```
$ RAID SHOW MY_ARRAY/FULL
```

```
HP RAID Software V3.0 Display Time: 1-NOV-2004 12:53:36.13
© 2004 Hewlett-Packard Development Company, L.P. All Rights Reserved.
```

RAID Array Parameters:

```
Current RAID Array ID: MY_ARRAY
Permanent RAID Array ID: MY_ARRAY
Date Created: 1-NOV-2004 11:10:27.89
Last Bind: 1-NOV-2004 11:11:41.00
RAID Level: 0
Current State: NORMAL

Characteristic Size: 4106822 Total Capacity: 12309104
Member Count: 3 Chunk Size: 120
Pool in Use: 6912 Max Pool Value: 0
Pages in Use: 0 Max Page Value: 0
```

RAID Array Configuration:

Member Index	Name	State
0	_\$1\$DKA100:	NORMAL
1	_\$1\$DKB0:	NORMAL
2	_\$1\$DKB200:	NORMAL

RAID Array Operations:

Member Index	Name	Reads	Writes	Errors
0	_\$1\$DKA100:	1096	6023	0
1	_\$1\$DKB0:	1173	8648	0
2	_\$1\$DKB200:	1132	6065	0
Member Total:		3401	20736	0

Virtual Unit	Size	Status	Reads	Writes	Errors
DPA0011:	3000000	ACCESS	1532	14403	0
DPA0022:	3000000	ACCESS	472	1344	0
DPA0033:	3000000	ACCESS	381	1651	0
DPA0044:	3000000	ACCESS	344	723	0
DPA0055:	309104	ACCESS	43	61	0

2 Histogram of I/O sizes, Virtual Unit DPA0011:

Blocks/IO	Reads	Writes	Total
1	382	3096	3478
2	0	886	886
4	1	1773	1774
8	0	3544	3544
16	1	2702	2703
32	1	2402	2403
64	1147	0	1147
128	0	0	0
256	0	0	0
512	0	0	0
more than 512	0	0	0

Histogram of I/O sizes, Virtual Unit DPA0022:

(continued on next page)

**Example 8-1 (Cont.) Sample RAID SHOW/FULL Command Report**

Blocks/IO	Reads	Writes	Total
-----	-----	-----	-----
1	472	768	1240
2	0	23	23
4	0	21	21
8	0	19	19
16	0	12	12
32	0	6	6
64	0	495	495
128	0	0	0
256	0	0	0
512	0	0	0
more than 512	0	0	0

Histogram of I/O sizes, Virtual Unit DPA0033:

Blocks/IO	Reads	Writes	Total
-----	-----	-----	-----
1	381	410	791
2	0	0	0
4	0	1	1
8	0	0	0
16	0	2	2
32	0	1	1
64	0	1237	1237
128	0	0	0
256	0	0	0
512	0	0	0
more than 512	0	0	0

Histogram of I/O sizes, Virtual Unit DPA0044:

Blocks/IO	Reads	Writes	Total
-----	-----	-----	-----
1	344	723	1067
2	0	0	0
4	0	0	0
8	0	0	0
16	0	0	0
32	0	0	0
64	0	0	0
128	0	0	0
256	0	0	0
512	0	0	0
more than 512	0	0	0

Histogram of I/O sizes, Virtual Unit DPA0055:

Blocks/IO	Reads	Writes	Total
-----	-----	-----	-----
1	43	61	104
2	0	0	0
4	0	0	0
8	0	0	0
16	0	0	0
32	0	0	0
64	0	0	0
128	0	0	0
256	0	0	0
512	0	0	0
more than 512	0	0	0

**Table 8–3 Description of RAID SHOW Command Printout (Brief Version)**

Number	Description
1	The chunk size is found in this field. To find some rules for setting chunk size, see Section 8.9.3
2	This field contains the histogram of the number of reads and writes for each range of block sizes. The histogram can be used to calculate the average I/O transfer size or to determine if multiple I/O transfer sizes are dominant in the workload.

## 8.8 How RAID0 Plus Shadowing Helps Performance

Although RAID0 arrays help improve I/O performance when compared to the same hardware configured as individual disks, the reliability of the RAID0 array is less because a single disk failure will cause the entire array to become unusable. RAID0 may be combined with Volume Shadowing to improve the array's reliability.

Figure 8–2 shows a RAID0 array whose four members are two-member shadow sets. This configuration provides the benefits of increased I/O performance obtained from RAID0 arrays with the benefits of data reliability that come from the use of shadow sets. In this configuration, each member of the RAID0 array is shadowed and the RAID0 array is protected in the event that one disk fails or a disk controller fails.

In addition, shadowing improves I/O performance if the I/O workload has a high proportion of reads because if two read requests to the same shadow member set are outstanding, they can be serviced concurrently, one by each shadowed disk. For read-intensive workloads, this may result in a performance improvement of 50%-70%.

## 8.9 Guideline for RAID0 Arrays

Use the general guidelines in the following sections to improve RAID0 performance.

### 8.9.1 Number of Array Members

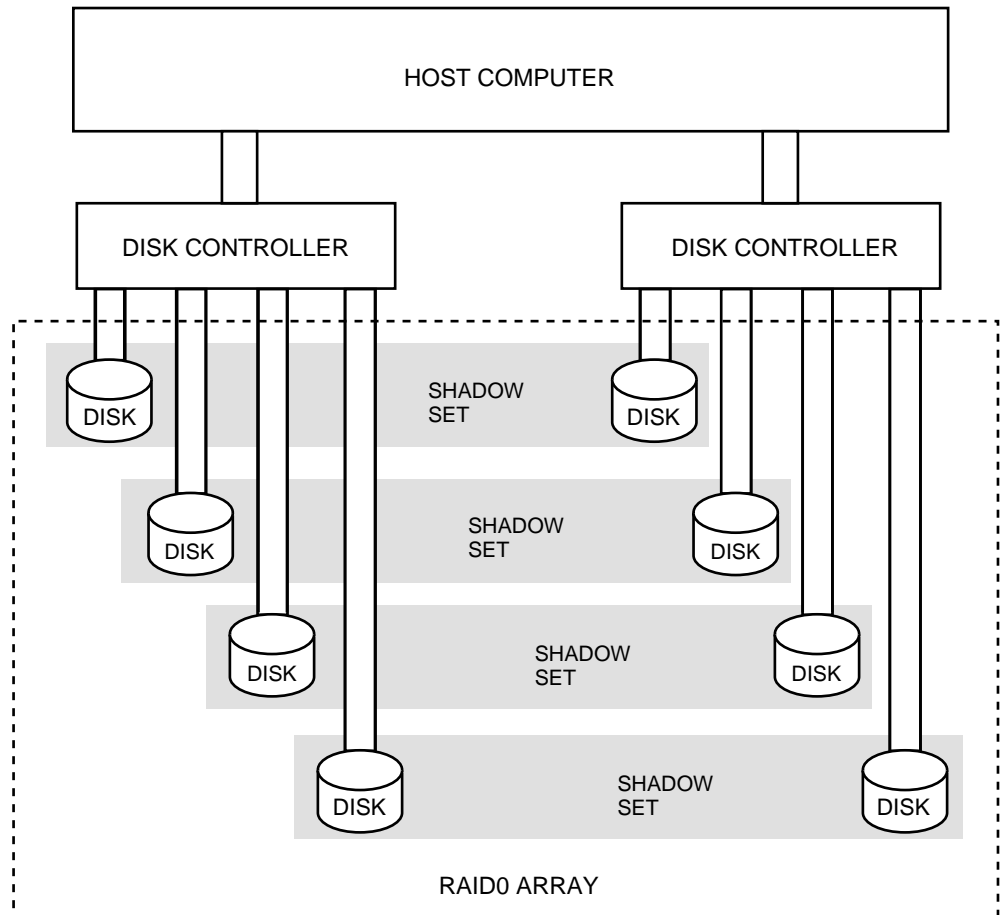
In general, the more members there are in an array, the more actuators are available to access the data, resulting in increased performance. However, more members also means an increased chance that one actuator will fail.

### 8.9.2 Disk Capacity

In general, using more smaller-capacity disks rather than fewer larger-capacity disks can result in increased I/O performance predominantly in request processing.



Figure 8–2 Four-Member RAID0 Array with Shadow Sets



CXO-4139A-MC

### 8.9.3 Chunk Size

The present default chunk size used by HP RAID Software for OpenVMS should be adequate for most typical OpenVMS applications.

HP recommends that the default chunk size be used unless one of the following conditions exist:

- The workload characteristics are well understood.
- The workloads are expected to be stable over time.
- There is reason to believe that modifying the chunk size will significantly improve performance.

If you must change your chunk size, the choice of chunk size should be determined if the following objectives are desired:

- High data transfer rate (for sequentially accessed files)
- High I/O request rate (usually for randomly accessed files)

The optimal chunk size for high data transfer rates is the average I/O transfer size divided by the number of disk members in the RAID0 array. To achieve load balancing or high request rates, choose a chunk size equal to approximately 10 to 15 times the average I/O transfer size.

#### **8.9.4 Queue Depth**

In general, the larger the queue depth, the more opportunity there is for the controller optimization code to rearrange requests for highest total request rate. The tradeoff is that larger queue depths also result in longer I/O response times for user I/Os.

Moving data from a single disk to RAID0 arrays normally results in lower queue depths and thus faster response times.

#### **8.9.5 Disk MSCP-Serving**

In general, a direct connection to a disk will provide higher performance than an MSCP-served connection to the same disk.

---

## RAID5: Improving Data Reliability, Availability, and Performance

This chapter provides information on how to improve or enhance performance, cost, and availability of RAID5 arrays with the HP RAID Software for OpenVMS. Table 9–1 provides a list of topics in this chapter.

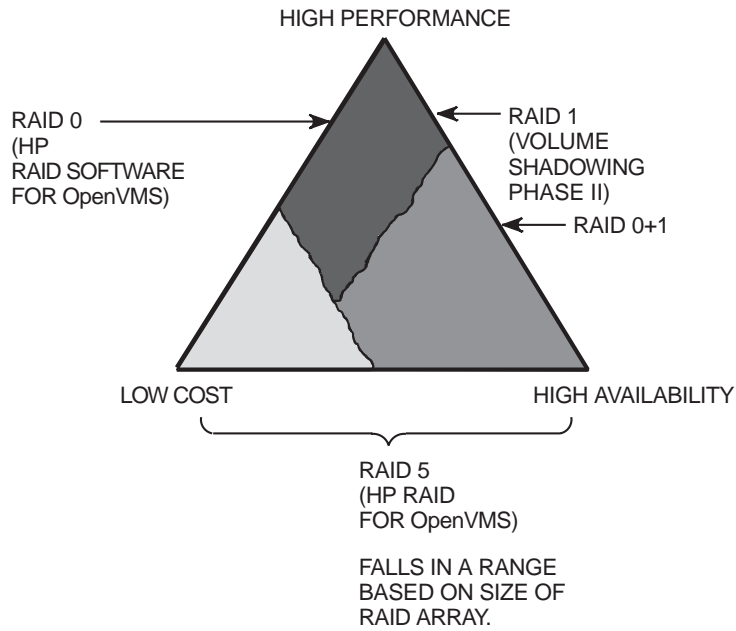
**Table 9–1 Topics in This Chapter**

<b>Subject</b>	<b>Section</b>
Concerns for Storage Administrators	9.1
Using RAID5 Array Size to Balance Cost, Availability, and Performance	9.2
Implications of Using Unlike Disks	9.3
Data Reliability Versus Availability	9.4
Reconstruction, Data Reliability, and Performance	9.5
Configuring RAID5 Arrays and Sparesets	9.6

### 9.1 Concerns for Storage Administrators

The triangle in Figure 9–1 represents the range of mass storage cost, performance, and availability options available to you as a storage administrator. This chapter shows you how to combine HP RAID with features from other parts of HP's product line for storage solutions with differing levels of each of these properties.

**Figure 9–1 User Requirements**



CXO3713BRA

Table 9–2 lists the storage solutions you might want to consider.

**Table 9–2 Performance, Cost, and Availability Considerations**

When evaluating ...	Then look at ...	In Section ...
Cost	Number of disks and arrays	9.2
	Suggested hardware configurations	9.4
Availability	Number of disks and arrays	9.2
	Suggested hardware configurations	9.4
	Using sparesets	9.6
	Using the timeout qualifier	9.5
Performance	Number of disks and arrays	9.2
	Using like or unlike disks	9.3
	Using the timeout qualifier	9.5

## 9.2 Using RAID5 Array Size to Balance Cost, Availability, and Performance RAID5

HP RAID Software for OpenVMS supports multiple-member arrays and multiple arrays per VMScluster system, as specified in the release notes. This gives you a wide latitude in configuring arrays to balance performance and availability requirements against cost constraints.

### **Cost**

An array with more disks provides less incremental cost for redundancy than does an array with fewer disks. An array with five identical member disks provides the same usable capacity as four independent disks at an additional 25 percent cost (one extra disk and port for every four user disks). An array with eight member disks provides the same usable capacity as seven independent disks at an additional 15 percent cost.

### **Availability**

An array with fewer disks provides greater protection against the impact of a failure than an array with more disks. If you require six disks of usable capacity, you can implement this with two four-disk arrays (eight physical disks) or one seven-disk array. If you implement with two five-disk arrays, then each array can suffer a failure without affecting your data reliability. With the single array, any two disk failures make all the data inaccessible.

### **Performance**

An array with fewer disks provides greater load balancing *per gigabyte* than an array with more disks. Moreover, if you configure with more arrays rather than fewer, your system can execute more write requests concurrently. On the other hand, larger arrays provide greater load balancing *across all your data* than do smaller arrays.

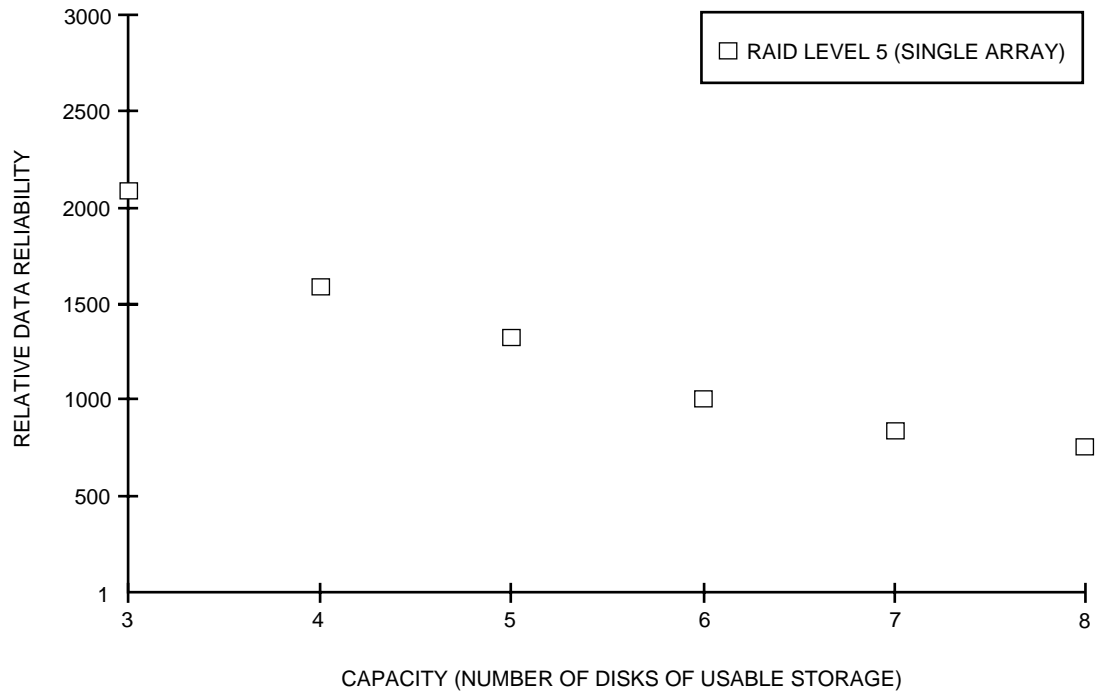
An array with three members provides the storage capacity of two independent disks. When servicing read requests, however, accesses are distributed across all three physical disks. That is, there are 50 percent more actuators and data channels available to execute requests than with independent disks.

On the other hand, multiple arrays are subject to the same kind of I/O load imbalances that characterize independent disks. Some files are just accessed more heavily than others (hot files). With a single large array, the load to these files is spread evenly across all of the array members. With multiple smaller arrays, only the disks in the array on which the hot file is stored serve requests to it.

### **Reliability**

Figure 9-2 shows the data reliability provided by HP RAID arrays relative to that of independent disks.

Figure 9–2 Data Reliability



CXO-3793B-MC

The horizontal axis in Figure 9–2 shows the **mean time to data loss (MTDL)** value normalized to one for different numbers of independent disks. The points on the vertical axis show the relative increase in data reliability when these independent disks are used in RAID5 arrays.

The curve shows that as the number of disks in the RAID array increases from the three to eight, the MTDL diminishes by a factor of slightly over two. Even the largest HP RAID array is some 900 times as reliable as an equal number of independent disks. That is, if the mean time between failure (MTBF) of a single disk is 150,000 hours (about 17 years), then the MTDL of eight identical independent disks is about two years. A RAID5 array of equivalent capacity made up of an equal number of disks, however, has an MTDL of 892 times that, or about 1900 years. Unless your population of disks is very large, data reliability should be a secondary consideration in determining array size.

One consideration is the time required to reconstruct the contents of a disk after a failure. This grows with the number of member disks in the array.

In determining **mean time to replacement (MTTR)**, it is necessary to include not only the amount of time required to make the physical substitution of a functional disk for the failed one, but also the reconstruction time because the array is not brought back to its full level of data protection until the reconstruction is complete.

### 9.2.1 Number of Disks Per RAID5 Array: Cost Versus Performance RAID5

In trying to find a compromise between having high performance and trying to keep costs down, you must look at reads and writes separately to decide which kind of array configuration will work best for you.

#### Reads

Consider a choice between a single 8-member array or two 5-member arrays. Table 9–3 summarizes the plusses and minuses of these two configurations for read operations.

**Table 9–3 Read Performance**

More Arrays	Fewer Arrays
Plus: more spindles for load balancing	Plus: fewer user visible disks
Minus: more user visible disks means more chances for hot spots	Minus: fewer spindles for load balancing

If you have a reasonably well-balanced load on your user disks, then you will benefit from having more spindles and thus using the more expensive scenario of a larger number of smaller arrays. If your I/O environment has one or two hot disks, then you are better off spending less money and using a smaller number of larger arrays.

#### Writes

Table 9–4 summarizes the plus and minus of a single 8-member array versus two 5-member arrays for write operations.

**Table 9–4 Write Performance**

More Arrays	Fewer Arrays
Plus: Can do more writes concurrently	Minus: Can do fewer writes concurrently

If your applications require numerous writes, it is better for performance to use the more expensive scenario of a larger number of smaller arrays because, during parts of a write operation, access to surrounding data in the array is serialized (that is, no other writes can occur to that area of the array). Having more arrays will keep more data space available during write operations. Thus you will improve your performance at the cost of three additional disks.

## 9.3 Implications of Using Unlike Disks

HP RAID for OpenVMS allows combining different types of disks within a single RAID array. Member devices of a RAID array can be any drive supported by the OpenVMS class drivers, DUDRIVER and DKDRIVER, or a nonshadowed drive supported by DSDRIVER. Supported drives thus include RA, RD, RF, RZ series, and any other drives supported by these class drivers.

Be aware that when different sized physical disks are used in a RAID array, the amount of usable data space available to you is affected. The usable data space on each disk is the number of available logical block numbers (LBNs), less some overhead, in the *smallest* member of the set. Any leftover space on the larger disks of the RAID array is not available to you.

## 9.4 Data Reliability Versus Availability RAID5

HP RAID software greatly increases the data reliability of data stored on a collection of physical disks. Data reliability is expressed in terms of MTDL due to the failure of a disk. Data reliability should be distinguished from enhanced availability, which is the ability to provide applications with continued access to data in the event of a component failure. Because any block of user data stored on a HP RAID array can be regenerated from remaining blocks in the event of a disk failure, the data reliability of a RAID array is very high. Furthermore, for disk failures, the HP RAID software provides enhanced availability of data to applications in the event of failure of either of the following:

- A member disk
- The access path to a single member disk

The software does not provide continued availability of data if the failure is more general (that is, failure of a controller which provides a path to multiple disks, a host I/O bus, or a host itself). HP RAID software does, however, interact with VMScluster system capabilities and other mass storage and I/O capabilities provided by HP products to produce high availability of data when those components are present in the system.

With software-based RAID, the number and locations of single points of failure are dependent on the hardware configuration chosen by the user. The tradeoff is between cost, reliability, and availability.

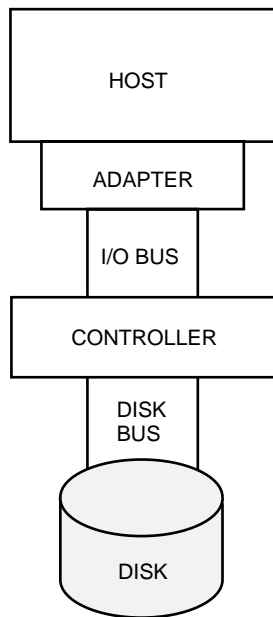
To understand how this interaction takes place, we must analyze a disk subsystem in terms of the components that can fail. Table 9–5 lists these components. Figure 9–3 shows the path between an application and its data in terms of components that can fail.



**Table 9–5 Subsystem Components**

Component	Description
Host	The center of control for the disk subsystem. This is where the application executes.
Adapter	This component moves data between host memory and the system's I/O bus. It provides data conversion between the two bus formats and may provide some short-term immediate buffering.
I/O bus	The I/O bus moves data between a host I/O adapter and one or more peripheral controllers.
Controller	Translates data from the I/O bus into a format understandable by the disk drive logic and vice versa.
Disk bus	The component that moves data between the disk controller and the disk.
Disk	A long-term, nonvolatile data storage device.

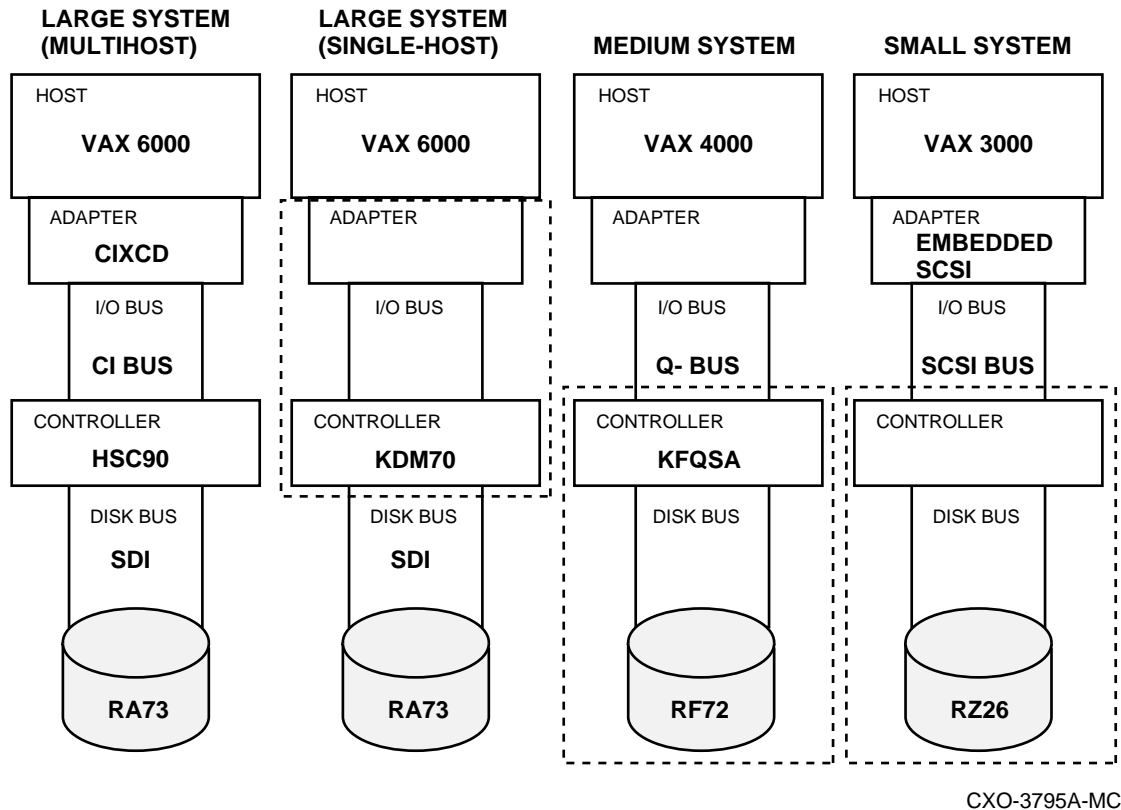
**Figure 9–3 Generic Subsystem**



CXO-3794A-MC

The basic components, listed in Table 9–5, exist in all of HP's systems. Technological and business considerations, however, have led to different packaging styles which provide the capabilities in different components. Figure 9–4 shows examples of the basic components as they are implemented in four types of HP systems.

Figure 9–4 Sample HP Systems



In the multihost large system shown in Figure 9–4, all of the functional components are implemented as discretely identifiable parts.

In the single-host large system, the adapter, I/O bus, and controller functions have been combined in the KDM70 single-host storage controller.

In the medium and small systems, the disk control function, the disk bus, and the disk itself have been combined into an **integrated storage element (ISE)**. This is significant from an availability standpoint because when any of the three functional components in either of these storage element fails, the entire element has failed and must be replaced as a unit. The difference between the medium and small models is in the multihost capability of the DSSI bus, which will appear later in this discussion.

The usual reason for building systems according to the large system model in Figure 9–4 is the increased functionality and connectivity to disks that can be provided by storage controllers such as the HSC controller family.

To determine how to provide protection against failure of any of these functional components, it is necessary to examine how to protect against each possible failure. Table 9–6 describes how to protect against failure for the small and medium system models.

**Table 9–6 Enhancing Data Reliability and Availability in Small and Medium System Models**

<b>Functional Component</b>	<b>To Minimize Exposure to Failure:</b>	<b>See Section:</b>
Storage element	Combine multiple storage elements into a RAID array using HP RAID software.	9.4.1
I/O bus	Attach each RAID array member to a separate I/O bus.	9.4.1
Host adapter	Replicate host adapter or use VMScluster technology to provide multiple hosts with access to the same data.	9.4.3
Host	Use VMScluster technology to provide multiple hosts with access to the same data.	9.4.3

Table 9–7 describes how to protect against failure for the large system model shown in Figure 9–4.

**Table 9–7 Enhancing Data Reliability and Availability in Multidrive-Based Subsystems**

<b>Functional Component</b>	<b>To Minimize Exposure to Failure:</b>	<b>See Section:</b>
Disk	Combine multiple storage elements into a RAID array using HP RAID software.	9.4.1
Controller	Use dual access disks, or attach each disk in a given array to a separate controller.	9.4.2
I/O bus	Attach each RAID array member to a separate I/O bus or use an inherently redundant I/O bus (such as, CI).	9.4.1
Host adapter	Replicate host adapter or use VMScluster technology to provide multiple hosts with access to the same data.	9.4.3
Host	Use VMScluster technology to provide multiple hosts with access to the same data.	9.4.3

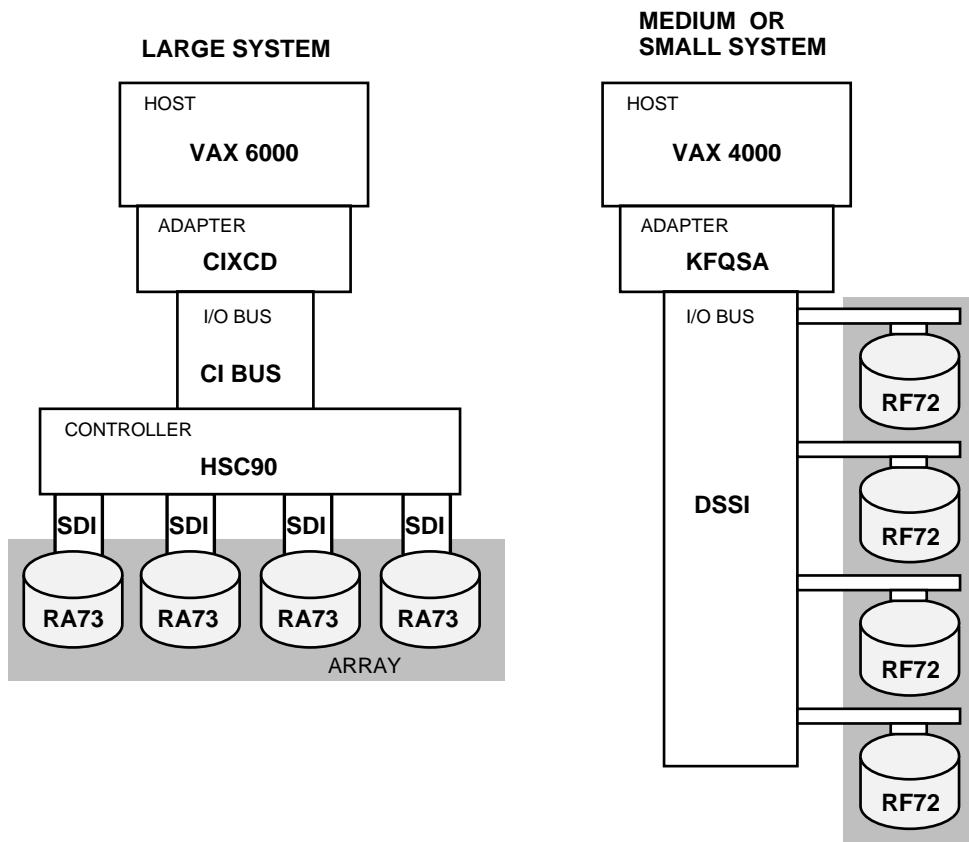
To simplify the analysis, host and host adapter have been considered as a unit because the failure of either will generally block application access to data. The effects of the application have deliberately been omitted from these charts because, in general, application failures leave data in a correct state from the disk

subsystem's standpoint, but in an incorrect state from the application's standpoint (that is, transactions may be half recorded). For this reason, HP RAID software (and RAID technology in general) should not be regarded as a substitute for a storage management program that includes regular backups.

### 9.4.1 Protecting Against Disk and Storage Element Failure RAID5

Protection against disk and storage element failure is the essence of RAID technology. Figure 9-5 shows a simple RAID array in both the large system, and the small and medium system models.

Figure 9-5 Simple RAID Arrays



CXO-3796A-MC

Either of the arrays shown in Figure 9-5 will survive the failure of any single disk, providing both data protection and continued availability. The large system array will also survive the failure of an SDI bus because each disk in this model has an independent SDI bus connecting the disk to its controller. (Not shown are the HSC channels. SDI buses may be connected to a common channel, or may be connected to separate channels for an added measure of failure protection).

In the medium or small system model, a failure of the DSSI or SCSI bus will interrupt access to data, although the data itself will be intact and may be accessed when the failure is repaired.

In the large system model, a failure of the controller will interrupt access to data. The CI bus, however, is inherently redundant, so two failures are required to interrupt data access.

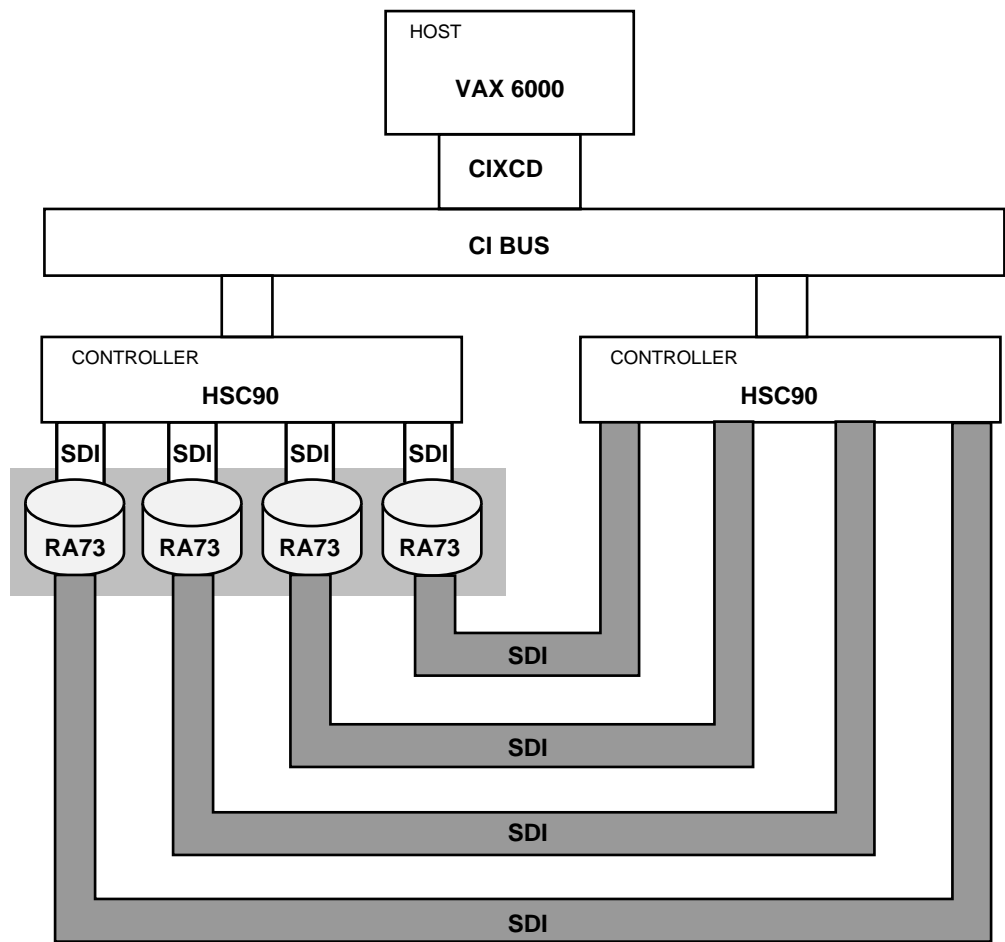
In both cases, adapter failures will interrupt access to data.

Just configuring arrays from multiple disks, therefore, generally affords a high level of data reliability, but protects against only a fraction of the other failures that can occur in the I/O path.

#### 9.4.2 Protecting Against Controller Failure RAID5

Protection against controller failure is provided in multidisk subsystems (the large system models) by replicating the controller. Figure 9-6 shows this for the HSC-family controller variation of the large system model.

Figure 9-6 Duplicate Controller Configuration



CXO-3797A-MC

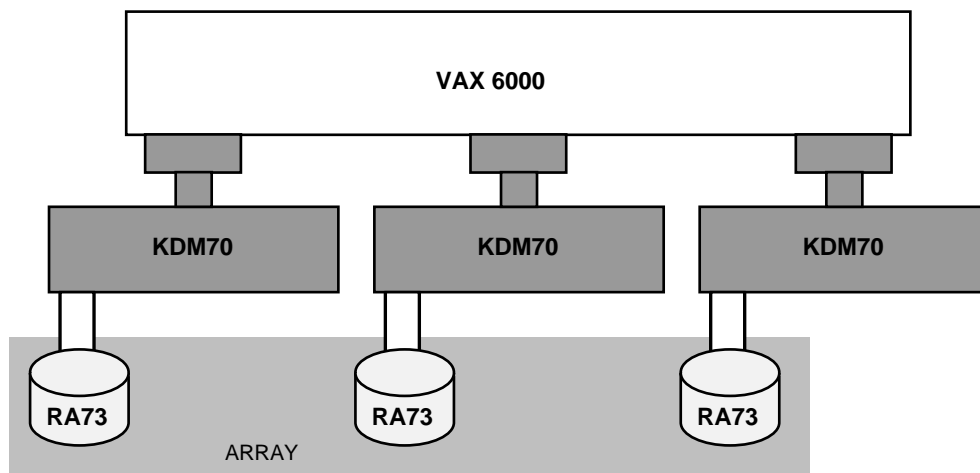
The example shown in Figure 9-6 makes use of the dual access feature of disks that attach to the CI bus to connect the disks to two controllers. In the event of failure of a controller, the disks can be switched to the other controller and access to data can continue. A system configured in this way provides protection against failures up to but not including the host adapter.

This configuration relies on a feature of the disks (SDI dual access), and also on the ability of the OpenVMS operating system and the HSC-family controllers to recognize the identity of the disks as they switch from a failed controller to a secondary one. While a similar configuration could be constructed using KDM70 controllers, the ability to recognize failed over disks is not present in that controller and so that configuration is not supported by the OpenVMS Operating System.

In the medium and small system models where the entire storage element is regarded as a single entity from a failure standpoint and a storage element has only a single bus connection, it is not relevant to discuss protection against controller failure.

An alternate, although somewhat expensive, way to protect against controller and bus failure in single-host large systems is to replicate the controller and rely on the properties of RAID5 technology for failure protection, as shown in Figure 9-7.

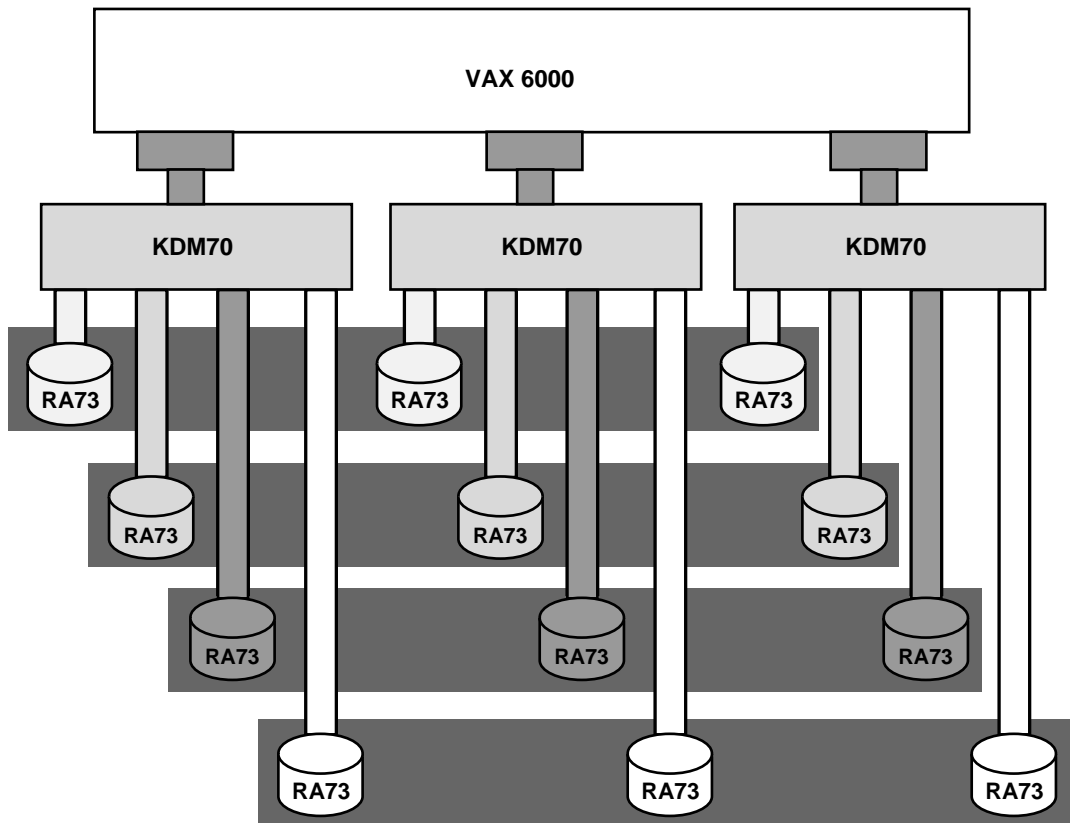
**Figure 9-7 Multiple Controller Configuration**



CXO-3798A-MC

In Figure 9-7, each disk has been given an independent path directly to host memory. Any part of the path can fail. RAID 5 technology will both protect against data loss and provide continued access to data (enhanced availability) by regenerating data requested from the failed disk using the surviving members. A configuration like this becomes more cost-effective if each controller supports members of multiple arrays, as shown in Figure 9-8.

Figure 9–8 Multiple Controllers with Multiple RAID Arrays



CXO-3799A-MC

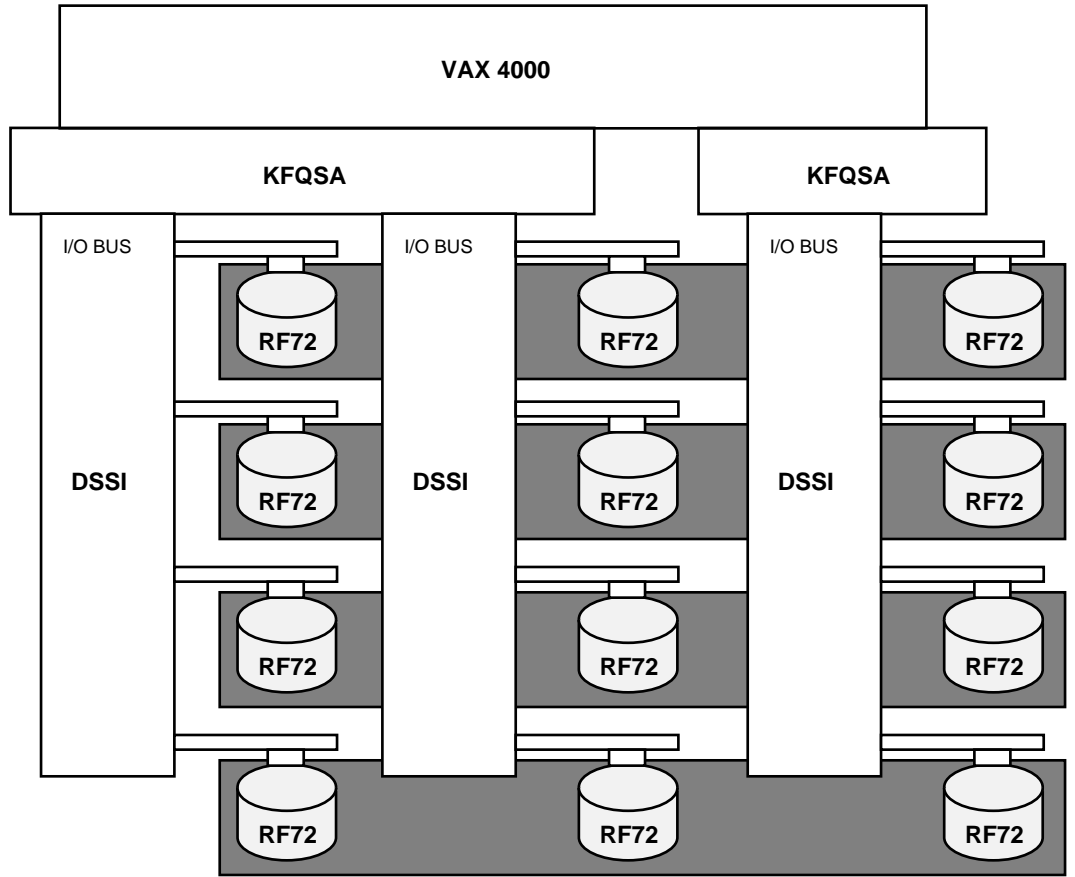
In the configuration shown in Figure 9–8, the failure of a controller leaves all arrays intact and functional. The failure of a drive while the system is in this reduced state, however, interrupts access to the array of which that drive is a part.

#### 9.4.3 Protecting Against Host Adapter Failure RAID5

Some instances of both the large and medium system models have separate components as host adapters. DSSI host adapters exist in both forms. VAX 6000 systems, for example, may be configured with the KFMSA storage adapter, which provides two DSSI buses. Other systems, such as the smaller VAX 4000 systems, have integrated DSSI bus adapters which are properly regarded as part of the host from an availability standpoint. Figure 9–9 shows protection against host adapter failure using a large system model example.

Figure 9–10 shows the same level of protection in the multihost large system model (although only a single host is shown).

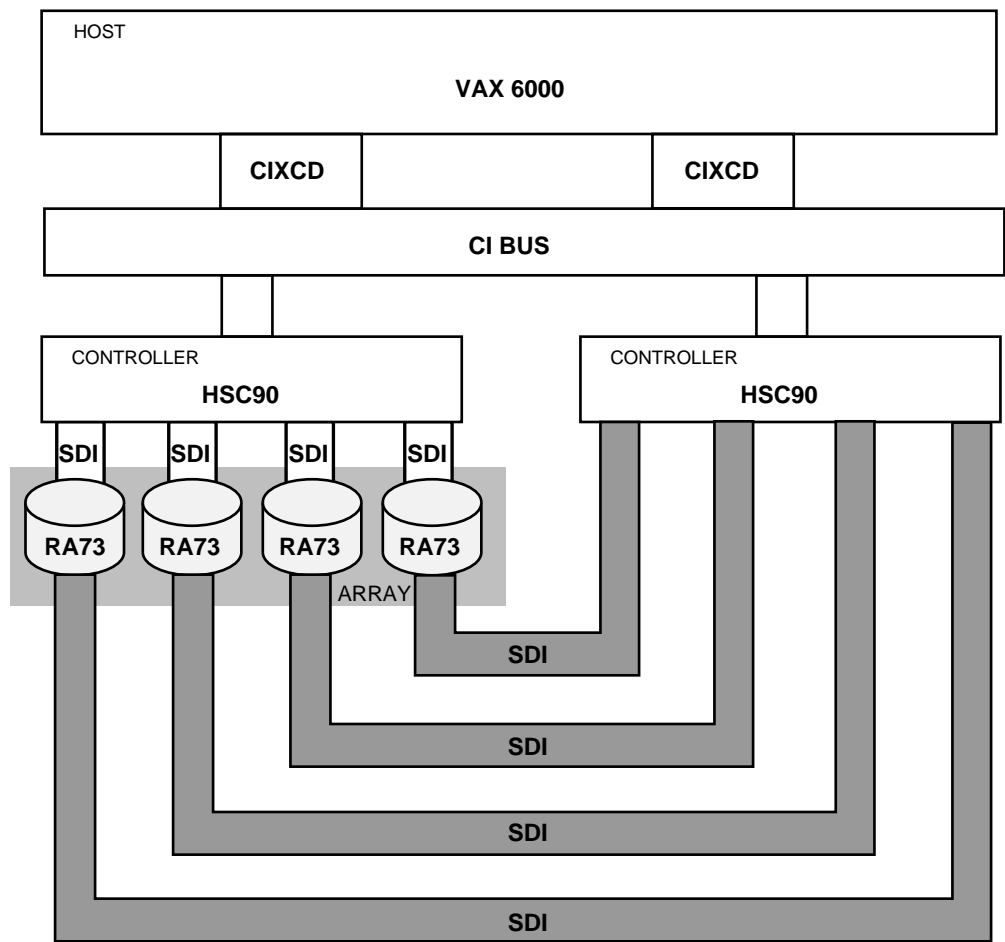
Figure 9-9 Protection Against Host Adapter Failure



CXO-3800A-MC



Figure 9–10 Protection Against Host Adapter Failure

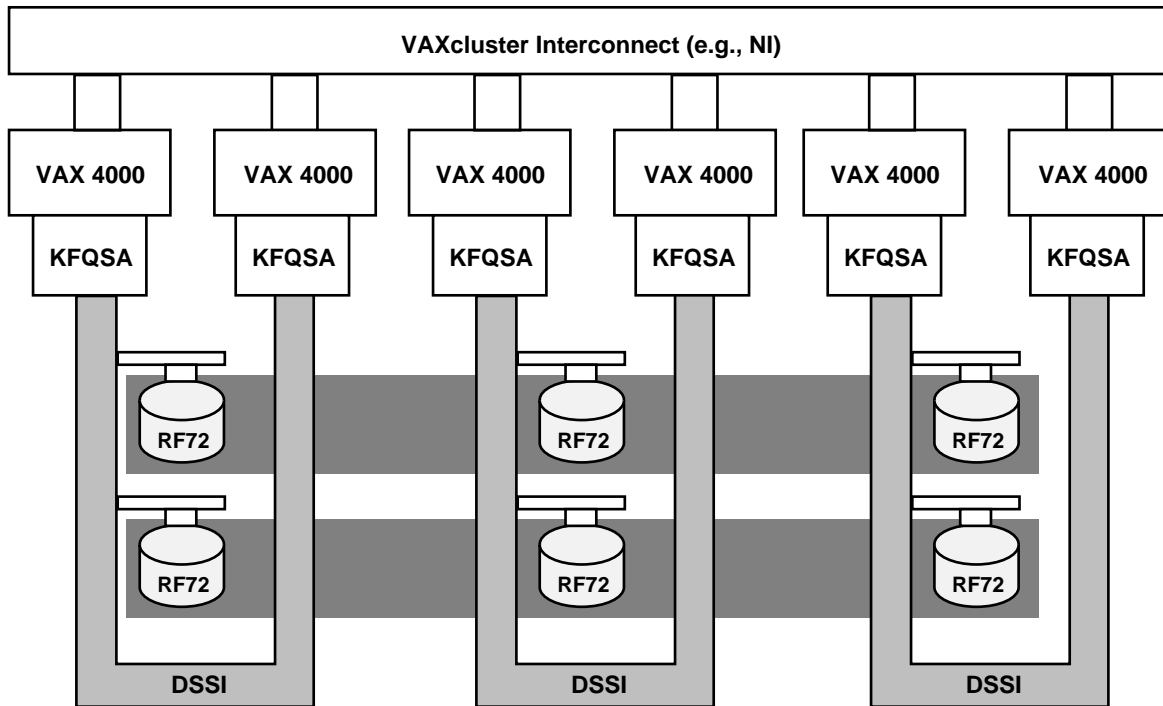


CXO-3801A-MC

In Figure 9–10, the host adapter to the CI bus has been duplicated, providing protection against failure of that component. As mentioned earlier, the CI bus itself is inherently redundant (each bus consists of two completely independent paths), so the system shown in Figure 9–10 has only the host as a single point of failure.

Finally, VMSccluster technology, in the form of served disks, can be combined with HP RAID to provide a further level of protection against loss of data availability. Figure 9–11 shows this using both the medium and small system models.

Figure 9–11 VMScluster Technology

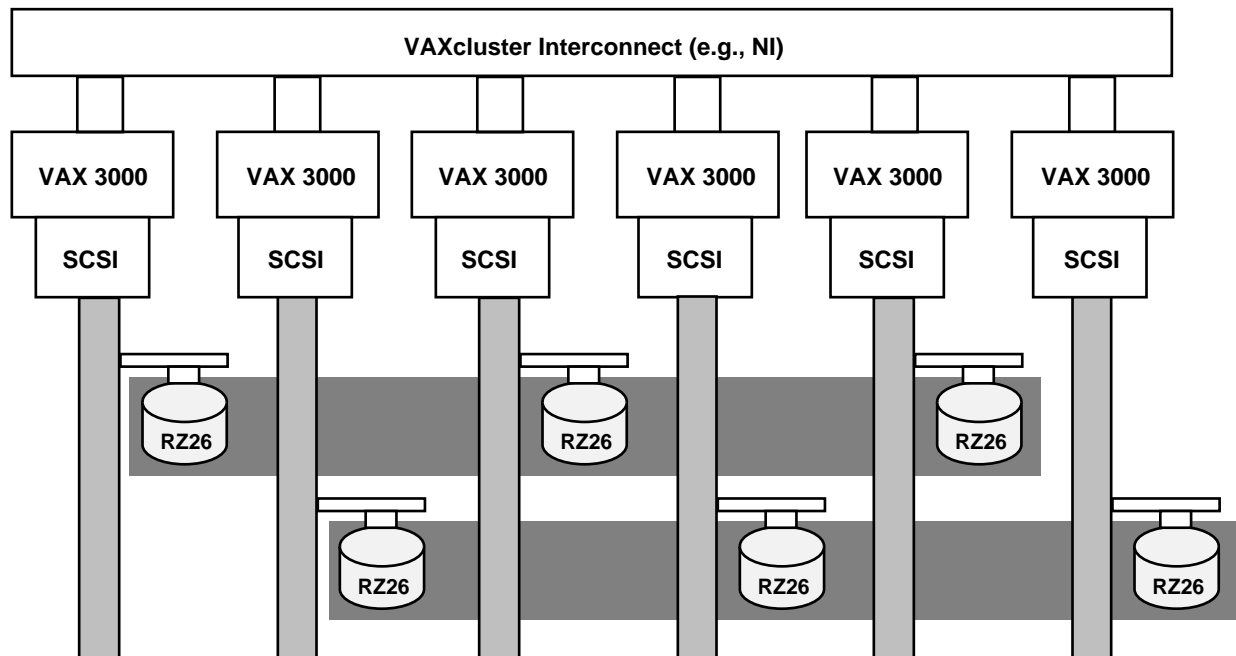


CXO-3802A-MC

In the example system shown in Figure 9–11, dual-host DSSI connections provide pairs of hosts with direct access to pairs of disks. MSCP service provides access to the disks on the other pairs of hosts. Any component of this system, including any host, can fail, but data access to the surviving hosts will not be interrupted. Perhaps the closest thing to a disruptive failure is the failure of a DSSI bus, which might incapacitate a pair of disks, but even then the hosts local to the failed DSSI bus still have access to their data, which is regenerated from the surviving disks.

Similar functionality is available to small system model VMScluster systems, as shown in Figure 9–12.

Figure 9–12 Small System Model VMScluster System



CXO-3803A-MC

In the example shown in Figure 9–12, both of the arrays are visible to all six hosts, at least partially through MSCP service. Again, any single host/host adapter combination, SCSI bus, or disk can fail without loss of access to data. The VMScluster interconnect can be replicated for higher availability.

## 9.5 Reconstruction, Data Reliability, and Performance RAID5

Mean time to repair (MTTR) is a key element of the data reliability of RAID5 technology. A change in MTTR from 24 hours to 48 hours, for example, can approximately halve the mean time to data loss (MTDL) of an 8-member array. MTTR is a measurement of the time from when the disk is removed from the array until it is returned to the array, its contents synchronized with the other array members, and full redundancy restored. In terms of HP RAID, reconstruction of the failed member's data onto the replacement member must be complete.

If maximum data reliability is your goal, it is usually advantageous to maintain a spareset and allow HP RAID's automatic member removal and replacement mechanisms to operate, as these tend to minimize MTTR.

The /TIMEOUT qualifier used with the RAID INITIALIZE command is also key to maximizing data reliability. A lower /TIMEOUT value will tend to cause earlier removal of a failing member from an array; this will probably tend to maximize data reliability. This comes at the expense of application I/O performance, however. When a member is removed from an array and replaced, reconstruction of the removed member's data onto the replacement member must occur. Reconstruction is a demanding process, both in terms of I/O capacity (primarily data transfer bandwidth) and in terms of CPU capacity (every stripe of

data in the array must have new parity or contents computed for the replacement member).

Because reconstruction can have a dramatic effect on the I/O resources available to applications, it may be desirable under some circumstances to delay reconstruction until, for example, a time-critical application is finished executing. The system manager does this by breaking the association between the array with the failing or failed member and its spareset, or by removing all members from the failed member's spareset temporarily. This forces the array to run in reduced mode, which has a much lesser effect on performance. When the critical job is finished, the spareset can be repopulated or reassociated, and HP RAID Software for OpenVMS will effect the replacement.

The system manager who adopts this approach must be aware that this action is creating a data reliability risk. The system manager is trading performance (the importance of getting the critical application finished) against data reliability (protection against the failure of a second disk), which will render the array's data irretrievable.

Because reconstruction uses system resources, you may want to prevent some nodes in the cluster from performing reconstruction by using the RAID SET /NORECONSTRUCT command and qualifier.

## 9.6 Configuring RAID5 Arrays and Sparesets RAID5

In general, the optimal way to configure RAID5 arrays is to provide as independent a path as possible between host memory, and each array member and the spares in its spareset. This tends to maximize the number of failures that can be tolerated, and, sometimes equally important, reduces the requirement to reconfigure an array after the spareset has been brought into service. With a sufficiently flexible configuration, a spare drive should be able to serve as a permanent array member, with the physical replacement for the originally failed disk becoming a spare.

---

## The OpenVMS Operating System Interface

This chapter provides information on how HP RAID Software for OpenVMS communicates with the OpenVMS operating system. Table 10–1 provides a list of topics covered in this chapter.

**Table 10–1 Topics in This Chapter**

Subject	Section
Command Line Interface	10.1
Using the OpenVMS BACKUP Command	10.2
System Shutdown with the RAID SHUTDOWN Command	10.3
Configuring the Software with the RAID SET Command	10.4
How to Access OpenVMS Help for RAID	10.5
OpenVMS Error Messages	10.6
The Diagnostics File	10.7
Other Files	10.8

### 10.1 Command Line Interface

A DIGITAL Command language (DCL) command line interface enables the operator to create and manage RAID arrays and to create the virtual devices that represent the RAID arrays. The DCL command line interface also allows statistics and configuration information associated with the RAID arrays and virtual devices to be displayed. See Chapter 11 for descriptions of the RAID commands used from the command line interface.

---

#### CAUTION

---

Because data on individual members is always contiguous with HP RAID, you **should not attempt to defragment the individual RAID array members**. Defragmenting individual members will not improve performance and **will result in corrupted data**.

---



---

#### Note

---

Defragmenting utilities can be run on the RAID virtual device.

---

## 10.2 Using the OpenVMS BACKUP Command

RAID virtual devices may be backed up by using the OpenVMS BACKUP command.

Because the RAID mapping scheme spreads data and files over an entire set, a single disk's data will not be coherent when viewed individually. Subsequent restores of data backed up from the RAID array members will compromise the data of the entire RAID array. Therefore, backing up and restoring individual members are not recommended.

OpenVMS standalone backup may not be used to back up or restore RAID virtual devices.

HP RAID is not a substitute for doing backups of the virtual device. While RAID5 provides protection against loss of a disk, it provides no protection against accidental deletion or modification of files or application programming errors.

## 10.3 System Shutdown with the RAID SHUTDOWN Command

HP recommends that the RAID SHUTDOWN command be inserted into the system shutdown file (SYSSMANAGER:SYSHUTDWN.COM) to ensure that underlying RAID members are dismounted correctly. To ensure that the RAID SHUTDOWN command completes successfully on a node, all RAID virtual devices must be dismounted.

The RAID SHUTDOWN command will perform the equivalent actions of the RAID UNBIND command on all RAID arrays, except that the unbinds will only be done on the node which issued the RAID SHUTDOWN command. RAID arrays, as seen by other nodes in the cluster will not change. The RAID SHUTDOWN command will also stop the RAID\$SERVER process. As the RAID\$SERVER process is not restartable without a node reboot, HP suggests that the RAID SHUTDOWN only be done when a node is being shutdown.

## 10.4 Configuring the Software with the RAID SET Command

The RAID SET command manages the software by setting limits on system resources to be used by the HP RAID software. For further information on this command, see Chapter 11.

## 10.5 How to Access OpenVMS Help for RAID

Online help for the HP RAID commands is available to you. You can access this file by entering the DCL command HELP RAID at the OpenVMS system prompt:

```
$ HELP RAID
```

## 10.6 OpenVMS Error Messages

HP RAID Software for OpenVMS provides error logging to the error log file, event notification to the OPCOM facility, and error messages for users entering DCL commands. Appendices Appendix A and Appendix B provide a list of these messages and the user action required. Event notification to the OPCOM facility includes removals, replaces, and reconstructs.

---

**Note**

---

You must use the RAID ANALYZE command in order to display the error log messages in the system error log file. A description of RAID ANALYZE command is found in Chapter 11.

---

## 10.7 The Diagnostics File

The RAID\$DIAGNOSTICS\_*nodename*.LOG file exists for analyzing various kinds of problems you may experience with your RAID array. Events and errors are recorded in this file, which is found in SYSS\$MANAGER. The file will be re-created automatically when deleted. The file can be relocated by defining the RAID\$DIAGNOSTICS\_*nodename* logical.

---

**Note**

---

To recover disk space, you should occasionally delete this file if it gets too large.

---

If you are having problems with your RAID array, save the diagnostics file for access by a support engineer. The first few lines of the diagnostic file will contain the log of the first command from the Installation Verification Procedure (IVP) if it is run.

## 10.8 Other Files

Others files you may see as a result of running the HP RAID software include RAID\$SERVER\_MAIN.LOG and RAID\$SERVER\_MAIN.DMP.

### The RAID\$SERVER\_MAIN.LOG File

The RAID\$SERVER\_MAIN.LOG file contains errors that would normally go to SYS\$ERROR from the RAID\$SERVER process. If this file gets created, it can be found in the SYS\$SPECIFIC:[SYSMGR] area of every node that is running the RAID\$SERVER process.

---

**Note**

---

This file should be purged periodically to clean up old versions from previous RAID\$SERVER process startups.

---

The type of information found in this file includes the following:

- Indications of what errors occurred if there were any problems in starting the RAID\$SERVER as a detached process.
- Any error messages generated if SYSS\$MANAGER:SYLOGIN.COM was not set up correctly to handle detached processes. Because the RAID\$SERVER process starts through LOGINOUT.EXE as a detached process, it will execute the file SYSS\$MANAGER:SYLOGIN.COM, if it exists.

### The RAID\$SERVER\_MAIN.DMP File

This file will appear only in the case of server-related problems. Under this condition, the RAID\$SERVER\_MAIN.DMP file can be found in the SYSSSPECIFIC:[SYSMGR] area of the node which had the server-related problems. Save this file for access by a support engineer.

## 10.9 How to Access Your RAID Array Through the Virtual Device

A program or programmer accesses a RAID array through a virtual device called *DPA<sub>nnnn</sub>*. The binding between DPA units and RAID arrays is dynamic, which means binding does not occur unless explicitly requested, and it may be dissolved and re-created at will using the RAID BIND and RAID UNBIND commands.

A *DPA<sub>nnnn</sub>* virtual device is created and associated with a RAID array by the user at the time of the RAID BIND command. The bind operation ensures that a RAID array is internally valid.



Each *DPA<sub>nnnn</sub>* unit is a virtual device that provides access to a RAID array. A RAID array must be bound to a DPA unit to create the association.

Once a RAID array is bound, the programming interface to a RAID virtual device is exactly like that of any other OpenVMS disk. The RAID virtual device responds to queue input/outputs (QIOs), record management system (RMS) calls, system utilities, and applications software as though the members of the RAID array which the virtual device represents were one physical device. A RAID virtual device, just like a physical disk, may be accessed using the MOUNT or MOUNT/FOREIGN commands, and need not contain an OpenVMS file system.














---

## RAID Commands

This chapter describes the commands for StorageWorks RAID Software for OpenVMS. The commands are listed in alphabetical order. Table 11–1 presents a list of the RAID commands. Icon  applies only to RAID5 arrays. Icon  applies only to RAID0 arrays. When no RAID icon is present, assume that the information applies to both RAID0 and RAID5.

**Table 11–1 RAID Commands**

Command	Software Product
RAID ADD/SHADOW_MEMBER	
RAID ADD/SPARESET	
RAID ANALYZE	Applies to both RAID0 and RAID5
RAID ANALYZE/ERROR_LOG	Applies to both RAID0 and RAID5
RAID BIND	Applies to both RAID0 and RAID5
RAID BIND/SPARESET	
RAID CLONE	
RAID INITIALIZE	Applies to both RAID0 and RAID5
RAID INITIALIZE/SPARE	
RAID MODIFY	Applies to both RAID0 and RAID5
RAID REMOVE	
RAID REMOVE/SHADOW_MEMBER	
RAID REMOVE/SPARE	
RAID REPLACE	
RAID SET	Applies to both RAID0 and RAID5
RAID SHOW	Applies to both RAID0 and RAID5
RAID SHOW/SPARESET	
RAID SHUTDOWN	Applies to both RAID0 and RAID5
RAID UNBIND	Applies to both RAID0 and RAID5
RAID UNBIND/SPARESET	

### 11.1 Event\_Procedure

Certain RAID events can be used to start a command procedure in a batch job. The command procedure receives additional information through its parameters P1..P8 depending on the event type. Logical names are used to control the use of this feature.

### 11.1.1 Event\_types

In the following list of event types, Pn refers to the parameter passed to the command procedure.

- DUMP - The RAID\$SERVER process has exited with a process dump.  
Parameter: P1="DUMP"
- EXLICENSE - RAID software license has expired.  
Parameter: P1="EXLICENSE"
- NEWNODECOMP - A newly started RAID\$SERVER process has completed all implicit binds.  
Parameters: P1="NEWNODECOMP"  
P2="<node\_name>"
- RECONCOMP - A reconstruct on a RAID 5 member device has completed successfully.  
Parameters: P1="RECONCOMP"  
P2="<array-ID>"  
P3="<device\_name>"
- RECONINC - A reconstruct on a RAID 5 member device has completed but some areas could not be reconstructed successfully.  
  
Use RAID ANALYZE/ARRAY/REPAIR to find out which DPAnn devices are affected. Use ANALYZE/DISK/READ to find out which files have no redundancy.  
  
Parameters: P1="RECONINC"  
P2="<array-ID>"  
P3="<device\_name>"
- REMOVE - A member of a RAID 5 array has been removed.  
Parameters: P1="REMOVE" or "REPLACE"  
P2="<spare\_set-ID>"  
P3="<device\_name>"
- SHADOWREMOVE - A member has been removed from a shadow set of a RAID 0+1 array. This event will be triggered on each node in a cluster with RAID 0+1 arrays.  
Parameters: P1="SHADOWADD" or "SHADOWREMOVE"  
P2="<spare\_set-ID>"  
P3="<shadow\_unit>"
- SPAREREMOV - A spare has been removed from a spare set.  
Parameters: P1="SPAREREMOV"  
P2="<spare\_set-ID>"  
P3="<device\_name>"
- SPARESETEMPY - A spare has been removed from a spare set and the spare set is now empty.  
Parameters: P1="SPARESETEMPY"  
P2="<spare\_set-ID>"  
P3="<device\_name>"

## 11.1.2 Logical\_names

The following logical names can be defined in the SYSTEM logical name table to direct the execution of the event batch jobs. These logical names are translated with every new event.

```
RAID$NOTIFY_QUEUE - Defines the name of the batch queue for the
                    event batch jobs. Default is SYS$BATCH.

RAID$NOTIFY_USERNAME
                    - Account name to use for submitting the event
                      batch jobs. Default is SYSTEM.

RAID$NOTIFY_PROCEDURE <event_type>
                    - Filename of the event command procedure to be
                      run for this event type. Missing parts of the
                      filename are filled in from the default file
                      specification SYS$MANAGER:.COM.

                    No batch is started if the command procedure
                      file does not exist.
```

## 11.1.3 Command Procedure Example

Example 11-1 illustrates a sample command procedure for triggering off events.

### Example 11-1 Command Procedure Example

Setup of logical names:

```
$ DEFINE/SYSTEM RAID$NOTIFY_QUEUE  SYSMGR_QUEUE
$ DEFINE/SYSTEM RAID_USERNAME      OPERATOR
$ DEFINE/SYSTEM RAID$NOTIFY_PROCEDURE_SPARESETEEMPTY -
COMMON_DISK:[SYSMGR]NOTIFY.COM

NOTIFY.COM:
$ MAIL -
  /SUBJECT="Spare set 'P2' empty. Last spare removed was 'P3'." -
  NLA0: OPERATOR
$ EXIT
```

## RAID ADD/SHADOW\_MEMBER RAID0

---

## RAID ADD/SHADOW\_MEMBER RAID0

This command adds a device to the shadow set and takes effect on all nodes in a VMScLuster configuration.

To use this command you must have OPER and VOLPRO privileges.

### Format

```
RAID ADD/SHADOW_MEMBER array-id unit-id
```

### Parameters

#### **array-id**

The *array-id* is the identifier for the bound RAID array.

#### **unit-id**

Specifies the *unit-id* that will be added to the specified shadow set. The disk must not be mounted or allocated.

### Description

The RAID ADD/SHADOW\_MEMBER command adds the specified unit to the shadow set that is specified by the /INDEX or /DEVICE qualifier. If possible, a full shadow copy will be started on the new shadow member. Refer to the OpenVMS Volume Shadowing documentation for more information on when shadow copies are handled.

### Qualifiers

#### **/DEVICE=DSA $nnnn$ :**

This qualifier will cause the ADD command to mount the unit-id into the shadow set specified by device DSA $nnnn$ ;, where  $nnnn$  is from 1 to 9999.

#### **/INDEX= $n$**

The qualifier will cause the ADD command to mount the unit-id into the shadow set specified by index position  $n$ . The list of index positions can be obtained by doing a RAID SHOW of the RAID array.

#### **/POLICY=[NO]VERIFY\_LABEL**

#### **/NOPOLICY(default)**

Require that any member that is going to be added to the shadow set must have a volume label of "SCRATCH\_DISK".

See HELP MOUNT/POLICY for availability and more information.

When the /SHADOW\_MEMBER qualifier is specified, then either the /INDEX or the /DEVICE must also be specified, but not both.

### Examples

1. \$ RAID ADD/SHADOW\_MEMBER PAYROLL \$3\$DUA220:/DEVICE=DSA99:

This command adds device \$3\$DUA220 to the shadow set DSA99, which is a member of the array PAYROLL.

2. \$ RAID ADD/SHADOW\_MEMBER PAYROLL \$3\$DUA220:/INDEX=2

This command adds device \$3\$DUA220 to the shadow set at index position 2 of array PAYROLL.

Suppose that a RAID array has the following configuration as displayed by the RAID SHOW/FULL command:

Member Index	Name	State	ShadowSet Members	ShadowSet State
0	_DSA6000: _ \$1\$DKA200:	NORMAL	1	SteadyState
1	_DSA6001: _ \$1\$DKA300:	NORMAL	1	SteadyState

The following RAID ADD command will add device DKA17: to shadow set DSA6000: and initiate a full copy using DKA17: as the target and \$1\$DKA200: as the master.

\$ RAID ADD/SHADOW\_MEMBER/INDEX=0 PAYROLL DKA17:

## RAID ADD/SPARESET RAID5

---

## RAID ADD/SPARESET RAID5

Adds spares to a spareset. This command takes effect on all nodes in a VMScluster configuration.

To use this command you must have OPER and VOLPRO privileges.

### Format

```
RAID ADD/SPARESET set-id unit-list
```

### Parameters

#### **set-id**

The *set-id* is an identifier for the bound spareset. The list of spares originally bound in the spareset will remain the same.

#### **unit-list**

This parameter lists the intentional device names of the spares, separated by commas, that will be added to the spareset. The disks must not be mounted and must have been previously initialized via the RAID INITIALIZE/SPARE command.

### Description

The RAID ADD/SPARESET command adds the units identified by *unit-list* to the specified spareset.

Units smaller than the characteristic size value defined when the spareset was bound will not be allowed to join the spareset.

The RAID ADD/SPARESET command is allowed only for sparesets that have been bound.

### Examples

1. 

```
$ RAID ADD/SPARESET SPARE1 DUA1:,DUA2:
```

This command adds spares DUA1 and DUA2 as additional members to spareset SPARE1.

---

## RAID ANALYZE

Provides a means of analyzing the on-disk metadata of RAID arrays or disk volumes. The command can also be used to recover full redundancy for RAID 5 arrays.

You must have OPER privilege to use this command.

### Format

RAID ANALYZE/ARRAY array-id

RAID ANALYZE/UNITS unit-list

### Parameters

#### **array-id**

The array-id is the identifier for the bound RAID array.

#### **unit-list**

This parameter lists the device names, separated by commas, that will be used in the report. The disks can be members of a bound array, a spare set member or any other disk device. These devices must be physical devices or logical names that translate to physical devices.

### Description

The ANALYZE/ARRAY or /UNITS command generates a formatted report of on-disk metadata. The metadata is used to describe and control the layout of user data on the RAID array. It can be used to identify an unused RAID array member (e.g. array name, member devices) or create a report in case of RAID problems (e.g. fail to bind an array) for further analysis. It can be run while the array is in use and it causes very little I/O overhead.

The ANALYZE/ARRAY/REPAIR command performs a reconstruct on all RAID 5 array members to recover full redundancy. It can be run while the array is in use but causes extra I/O load and should therefore be avoided during peak usage hours. The repair operation will list all areas which could not be recovered. In such a case, use the DCL command ANALYZE/DISK/READ to list the file with parity errors. These files need to be restored to remove the parity errors.

### Qualifiers

#### **/ARRAY**

Produces a formatted report of on-disk meta data structures of RAID set members devices. The array has to be bound.

#### **/OUTPUT=filename**

Directs the output of the RAID ANALYZE command to the specified file. If this qualifier is not specified, the output is directed to SYSS\$OUTPUT.

#### **/REPAIR** RAID5

Analyzes a bound RAID 5 array for full redundancy. Full redundancy will be established where necessary. Remaining non-redundant areas will be reported. The array has to be in either NORMAL or RECONSTRUCT mode. This command has no meaning for RAID 0 arrays.

## RAID ANALYZE

### **/UNITS**

Produces a formatted report of on-disk meta data structures of RAID set members devices. The specified disk devices can be either members of a bound array, a spare set or any other disk device.

### **Examples**

1. See Section 4.6.2.



---

## RAID ANALYZE/ERROR\_LOG

Provides a means of analyzing the error log entries that the HP RAID Software enters into the system error log file (SYSSERRORLOG:ERRLOG.SYS). Entries are not logged against sparesets in the error log. Operations, such as sparesets, are arranged by the RAID\$SERVER process and recorded by OPCOM.

### Format

RAID ANALYZE/ERROR\_LOG filename

### Parameters

**filename**

Specifies one or more file names that contain binary information to be interpreted for the error log report.

### Description

The ANALYZE /ERROR\_LOG subcommand provides a means of formatting the OpenVMS error log entries that the HP RAID Software generates.

OpenVMS ANALYZE/ERROR\_LOG command cannot format HP RAID Software error log messages. To enable you to display meaningful messages from the error log, the RAID ANALYZE/ERROR\_LOG command has been provided in this version of the software. Future versions of OpenVMS may provide the ability to display RAID software error log information using the OpenVMS ANALYZE /ERROR\_LOG command.

### Qualifiers

**/BEFORE=vms-date-time**

This causes the selection of records that were entered before the date specified.

**/SELECT=select\_items**

This causes the selection of records based on characteristics of their entry. The following list of select\_items may be used:

- **DEVICE:** Selects entries that relate to the RAID virtual device and errors that would be returned to the application.
- **MEMBER:** Selects entries that relate to the members of the RAID virtual device. Errors on a member device may or may not be returned to you.
- **EVENT=event\_items:** Selects entries that relate to the events on a RAID virtual device. The following list of event\_items may be used:
  - **ALL:** Selects all event types. This is the default.
  - **BIND:** Selects only BIND events.
  - **UNBIND:** Selects only UNBIND events.
  - **NORMAL** [RAID5]: Selects events related to a RAID array transitioning from a reconstruction state to a normal state.
  - **REDUCED** [RAID5]: Selects events related to a RAID array transitioning from a normal or reconstructing RAID array to a reduced RAID array.

## RAID ANALYZE/ERROR\_LOG

- **RECONSTRUCT** `[RAID5]`: Selects events related to a RAID array transitioning to a reconstructing RAID array.
- **REMOVE** `[RAID5]`: Selects events related to member remove requests.

### **/OUTPUT=filename**

Specifies the destination of the formatted output information. The default is SYS\$OUTPUT.

### **/SINCE=vms-date-time**

This causes the selection of records that were entered since the date specified.

## Examples

1. \$ RAID ANALYZE /ERROR\_LOG SYS\$ERRORLOG:ERRLOG.SYS

This command displays events relating to all RAID virtual devices that can be found in this SYS\$ERRORLOG.SYS file.

---

## RAID BIND

Associates a RAID array with a RAID virtual device. This command takes effect on all nodes in a VMScluster configuration.

To use this command you must have OPER and VOLPRO privileges.

### Format

```
RAID BIND array-id unit-list virtual-device-list
```

### Parameters

#### **array-id**

The *array-id* is the identifier for the bound RAID array.

#### **unit-list**

This parameter lists the device names, separated by commas, that will be added to the possible members of the array. The disks must not be mounted and must have been previously initialized via the RAID INITIALIZE command. These devices must be physical devices or logical names which translate to physical devices.

#### **virtual-device-list**

With multiple partitions possible, you must specify a name for each virtual unit that is to be bound to a virtual device. This parameter specifies the RAID virtual-device-list that will provide access to the RAID array. This list must be in the form of *DPAnnnn*, *DPAnnnn*, ..., where *nnnn* is any number from 1 to 9999. This may not be the same as the virtual device of another currently bound RAID array.

### Description

The RAID BIND command associates a RAID array with a virtual device. This action occurs on all nodes of the cluster that are running the HP RAID Software. Before entering the RAID BIND command, the RAID array members must have been initialized via the RAID INITIALIZE command.

A search algorithm helps ensure that a bind operation succeeds, even when the command line specifies disk devices that have been removed from the RAID array and replaced by other disks.

Given at least one member of a RAID array, the RAID BIND command attempts to find the current members. If all members are found and accessible, then the RAID array will be bound. If you specify units that are not members of the array, they will be ignored.

**RAID5** If all members, except one, are found and accessible, the RAID BIND command attempts to bind the RAID5 array by removing the missing member as long as the /NOREMOVAL\_ALLOWED qualifier is not specified.

**RAID0** All members must be found and accessible for a RAID0 array to be bound.

The RAID BIND command mounts each member on each node running the HP RAID Software and confirms that the members form a valid set.

## RAID BIND

Note that the RAID BIND command does not mount the RAID array virtual device(s). The OpenVMS INITIALIZE or MOUNT commands, or both, may be applied to a virtual device once a bind operation succeeds. However, neither the INITIALIZE or MOUNT commands are required.

After a RAID bind operation, the OpenVMS software views the resultant RAID virtual device(s) as another drive to which all valid OpenVMS disk management and I/O commands apply.

### Qualifiers

#### **/ALLOW\_LARGE**

Obsolete qualifier.

#### **/ASSOCIATED\_SPARESET=set-id** RAID5

#### **/NOASSOCIATED\_SPARESET (default)**

Associates a spareset with a RAID array to allow automatic replacement of missing RAID array members.

The spareset must already be bound. The spareset must have a characteristic size that is greater than or equal to the RAID array characteristic size. If it is not, the association to the spareset will not be made, but the bind will continue. A warning will be displayed.

Only one spareset may be associated with a RAID array.

#### **/IO\_TIMEOUT (Default)** RAID5

#### **/NOIO\_TIMEOUT**

If the /NOIO\_TIMEOUT is specified, the RAID driver avoids the CPU time overhead associated with error handling where a member's drive does not respond in a timely fashion. These qualifiers may be used with the /TIMEOUT and /NOTIMEOUT qualifiers to produce the results shown in Table 11–2.

**Table 11–2 IO\_TIMEOUT and TIMEOUT Interaction**

<b>TIMEOUT Qualifiers</b>	<b>/NOIO_TIMEOUT (No buffer copies)</b>	<b>/IO_TIMEOUT (default) (buffer copies)</b>
<b>/NOTIMEOUT</b>	Indefinitely wait for member I/Os to complete. No error recovery necessary so IO_TIMEOUT is disabled.	Indefinitely wait for member I/Os to complete. No error recovery is necessary so /IO_TIMEOUT is disabled. The /NOTIMEOUT overrides the /IO_TIMEOUT qualifier.
<b>/TIMEOUT (default)</b>	Member I/Os get timed out but error handling will not take place. There is no overhead for member error handling.	Member I/Os get timed out and there will be overhead associated with handling these errors.

#### **/OVERRIDE**

Specifies that the *array-id* name specified in the command line does not need to match the *array-id* name that is found in the on-disk information. If they do not match, the *array-id* from the command line will be used, overriding the

on-disk information. The on-disk information will not be modified. Use the RAID MODIFY command to change the name permanently after the bind.

**/REMOVAL\_ALLOWED (Default)** RAID5

**/NOREMOVAL\_ALLOWED**

The **/REMOVAL\_ALLOWED** qualifier specifies that the bind should succeed even if one member is unavailable. In this case, the unavailable member will be removed from the RAID array membership. HP RAID Software will then attempt to automatically replace the missing member if there is a populated spareset associated with the RAID array.

The **/NOREMOVAL\_ALLOWED** qualifier specifies that the RAID BIND command will fail if it is unable to bind the RAID array with its entire membership.

If the RAID array is in a reconstructing state, only the reconstructing member can be removed from the array.

Note that this qualifier is ignored for RAID arrays from which a member has already been removed (that is, a RAID array in a reduced state).

**/SHADOW** RAID0

**/NOSHADOW**

This qualifier specifies the policy for how the underlying members of the array will be mounted, either as shadow sets, or as individual members. If the **/NOSHADOW** qualifier is specified, and the array index positions have multiple members, then the array members will be mounted as single member nonshadowed disks, and it is undetermined which of the previously shadow members will actually be mounted.

If this qualifier is not specified, the array members will be mounted in the same manner as the last bind. If this is the first bind and the qualifier is not specified, then the members will be mounted with no shadowing.

**/TIMEOUT=n seconds (1 to 99999)** RAID5

**/NOTIMEOUT**

The **/TIMEOUT** qualifier specifies the number of seconds that HP RAID Software will wait for a member I/O to complete before declaring the device unavailable and using RAID algorithms to complete the I/O. If you do not specify a value, the default value (as specified in the release notes) will be used. The timeout value is persistent. That is, it retains the original value specified across binds and unbinds until a new value is specified.

Using the **/NOTIMEOUT** qualifier means that the HP RAID Software will never time out a member I/O.

Not specifying the **/TIMEOUT** qualifier means that the last value specified will continue to be used. If a value was never specified using the RAID INITIALIZE command, the default value (specified in the release notes) will be used. See Section 7.6.2 for more information on the HP RAID timeout mechanism.

**/USE\_SHADOW\_DEVICES=(dsa-list)** RAID0

**/NOUSE\_SHADOW\_DEVICES**

The **dsa-list** is a preferred list of DSA device names, by index position, that is used by the BIND command to mount shadow sets. The index position in the **dsa-list** begins with zero.

## RAID BIND

If there are not enough devices on the dsa-list for mounting RAID array member, the RAID BIND command will assign free DSA devices starting at 6000. If a DSA device in the list is in use, the RAID BIND command will assign a replacement DSA number and it will be the first free DSA device number starting with 6000.

To use this qualifier, the /SHADOW qualifier must be used.

**/WRITE(default for non-cloned array)**

**/NOWRITE(default for cloned array)**

The /NOWRITE qualifier allows to bind a RAID array with its virtual unit(s) write locked.

The /WRITE qualifier allows to override the write locked state of a cloned array. Once a cloned array has been bound /WRITE the array members can only be added back to the original array with a full shadow copy. See RAID CLONE /POLICY for more information.

## Examples

1. `$ RAID BIND PAYROLL DUA1:,DUA2:,DUA3: DPA137:`

This command associates the RAID virtual device DPA137: with the RAID array PAYROLL.

2. `$ RAID BIND PAYROLL DUA1:,DUA2:,DUA3:/SHADOW-  
/USE_SHADOW_DEVICES=(DSA3,DSA4,DSA5) DPA137:`

This command associates RAID virtual device DPA137 with RAID array PAYROLL and creates the volume shadowing virtual devices DSA3, DSA4, and DSA5.

3. `$ RAID BIND/SHADOW/USE_SHADOW_DEVICES=(DSA100:,"",DSA300:) -  
PAYROLL DUA1:,DKA14:,DUA10: DPA137:`

The above RAID BIND command will create three shadow sets (DSA100:, DSA6000:, and DSA300:) each with a single shadow member (DUA1:, DKA14: and DUA10:, respectively). Note DSA6000 was used because the second index position was left blank using double quotes.

4. `$ RAID BIND/SHADOW/USE_SHADOW_DEVICES=(DSA100:,DSA200:,DSA300:) -  
PAYROLL DUA1:,DKA14:,DUA10: DPA137:`

In this example, if DSA100: and DSA300: already exist in the VMScluster and therefore cannot be used, DSA6000: and DSA6001: will be used in their place (or the next sequential DSA device name that is available starting with DSA6000:). Table 11-3 lists the resulting shadow sets.

**Table 11-3 Resulting Shadow Sets**

Shadow Set Virtual Units	Physical Device names
DSA6000:	DUA1
DSA200:	DKA14
DSA6001:	DUA10

---

## RAID BIND/SPARESET RAID5

Creates a spareset containing the specified spares. This command takes effect on all nodes in a VMSccluster configuration.

To use this command you must have OPER and VOLPRO privileges.

### Format

```
RAID BIND/SPARESET set-id [unit-list]
```

### Parameters

#### **set-id**

The *set-id* is an identifier for the spareset.

This command cannot be used on a spareset that is already bound. The list of spares originally bound in the spareset will remain the same.

#### **unit-list**

This parameter lists the intentional device names of the spares, separated by commas, that will be members of the spareset. The spares must not be mounted, bound, or allocated. These devices must be physical devices or logical names which translate to physical devices and have already been initialized by the RAID INITIALIZE/SPARE command.

The *unit-list* parameter is optional as sparesets may be empty. The /CHARACTERISTIC\_SIZE qualifier is required if no valid spares are specified.

### Description

The RAID BIND/SPARESET command creates a spareset and associates the specified units with the spareset. These spares then form a pool from which HP RAID Software may select units to replace removed RAID array members. Before entering the RAID BIND/SPARESET command, the spares must have been initialized via the RAID INITIALIZE/SPARE command.

The RAID BIND/SPARESET command mounts each unit on each VMSccluster node running the product and confirms that the specified volumes have been initialized as spares. Units that are not spares are omitted from the spareset and a warning is displayed. The devices that are spares are included in the spareset.

Spares may not be bound into more than one spareset at the same time.

If the spareset already exists from a previous RAID BIND/SPARESET command, this command fails.

## RAID BIND/SPARESET RAID5

### Qualifiers

#### **/CHARACTERISTIC\_SIZE=blocks**

Specifies the characteristic size of a spareset. Spares with a characteristic size smaller than the value specified are not allowed to join the spareset. For a RAID array to use the spareset, the RAID array characteristic size must be less than or equal to the spareset characteristic size. If not specified via this qualifier, the characteristic size defaults to the characteristic size of the smallest spare in the set. This qualifier is required if no spares are specified. See Chapter 3 for more information on characteristic size.

### Examples

1. `$ RAID BIND/SPARESET ACCOUNTS DUA1:,DUA2:`

This command creates the spareset ACCOUNTS, which contains the spares DUA1 and DUA2.



---

## RAID CLONE RAID0

The RAID CLONE command breaks out shadow set members of a RAID array so they can be bound as a different RAID array.

To use this command, you must have OPER and VOLPRO privileges.

### Format

```
RAID CLONE array-id new-array-id
```

### Parameters

**array-id**

The *array-id* is an identifier for the bound RAID array.

**new-array-id**

The *new-array-id* is the new identifier for the RAID array that will be made up of the shadow set member broken out of the existing bound RAID array.

### Description

The RAID CLONE command breaks out shadow set members of a RAID array so they can be bound into a different RAID array with duplicate array data. This command enables RAID arrays to be manipulated so that one member of each shadow set is made available for binding into another RAID array. Once bound, this duplicate RAID array can then be used to make a backup of the array.

The RAID CLONE command will only work on a RAID array whose membership consists solely of shadow sets. Each shadow set must have at least two shadow set members. The RAID CLONE command also checks that at least two of the shadow set members are in steady-state, that is, they are not copy targets or merge members.

The RAID CLONE command will create the process logical name "RAID\$CLONE\_MEMBERS". Its equivalence is a search list containing all the disk device names of the cloned array.

### Qualifiers

**/OVERRIDE=CHECK**

This will cause the RAID CLONE command to perform the command even if the RAID virtual device(s) associated with the array-id is mounted.

**/POLICY=[NO]MINICOPY[=OPTIONAL]****/NOPOLICY(default)**

This will create a cloned array which preserves the shadow context of the physical devices. When binding the cloned array the virtual units are write locked. In this state the members of the cloned array can be added back to the original array with RAID ADD/SHADOW\_MEMBER by performing a shadow minicopy.

See RAID BIND RAID\_ARRAY/WRITE for overriding this state.

See MOUNT/POLICY for more information on shadow minicopy. Not available on OpenVMS VAX.

## RAID CLONE RAID0

### **/USE\_DEVICES=(unit-list)**

The unit-list is a list of device names, separated by commas, that will be used as the members of the cloned array.

## Examples

1. 

```
$ RAID CLONE PAYROLL BACKUP_PAYROLL
%RAID-I-CLONEMEMBER, device _$3$DUA143: is now a member of cloned
raid array BACKUP_PAYROLL
%RAID-I-CLONEMEMBER, device _$3$DUA145: is now a member of cloned
raid array BACKUP_PAYROLL
$ RAID BIND BACKUP_PAYROLL 'F$LOGICAL("RAID$CLONE_MEMBERS") DPA25:
```

This command sequence creates an identical copy of the PAYROLL RAID 0 array and names it BACKUP\_PAYROLL. The logical name created from the RAID CLONE command is used to specify the disk device to bind the new array. Device name DPA25 can then be used to access the cloned array.

2. 

```
$ RAID CLONE PAYROLL BACKUP_PAYROLL
%RAID-I-CLONEMEMBER, device _$3$DUA143: is now a member of cloned
raid array BACKUP_PAYROLL
%RAID-I-CLONEMEMBER, device _$3$DUA145: is now a member of cloned
raid array BACKUP_PAYROLL
$ RAID BIND BACKUP_PAYROLL $3$DUA143:,$3$DUA145: DPA25:
```

This command sequence creates an identical copy of the PAYROLL RAID 0 array and names it BACKUP\_PAYROLL. The output from the RAID CLONE command is used to specify the list of device names for the cloned array. Device name DPA25 can then be used to access the cloned array.

3. 

```
$ RAID CLONE ARRAY1 ARRAY2 /USE_DEVICES=(DKD300, DKD400)
%RAID-I-CLONEMEMBER, device _$1$DKD300: is now a member of cloned
raid array ARRAY2
%RAID-I-CLONEMEMBER, device _$1$DKD400: is now a member of cloned
raid array ARRAY2
$ RAID BIND ARRAY2 DKD300, DKD400 DPA25:
```

This command sequence creates an identical copy of the ARRAY1 RAID 0 array and names it ARRAY2. The devices specified in the unit-list are used to create the cloned array. Device name DPA25 can then be used to access the cloned array.

---

## RAID INITIALIZE

Creates a RAID array and defines its membership and permanent characteristics. To use this command, you must have OPER and VOLPRO privileges.

### Format

RAID INITIALIZE array-id unit-list

### Parameters

#### array-id

The *array-id* is the identifier for the RAID array. The *array-id* is specified as a 1- to 32-character string containing only alphanumeric characters, including the dollar sign (\$) and underscore (\_). The identifier is used by other commands to identify the RAID array.

#### unit-list

This parameter lists the device names of the disks, separated by commas, that will be members of the array. The disks must not be mounted, bound, or allocated. The number of devices in this list determine the number of members in the RAID array. For RAID5, you need a minimum of three members and a maximum as specified in the release notes. For RAID0, you need a minimum of two members and maximum as specified in the release notes. These devices must be physical devices or logical names that translate to physical devices.

### Description

The RAID INITIALIZE command creates a RAID array using the members specified. It records the membership and permanent characteristics of the RAID array.

Member devices of a RAID array can be any drive supported by the OpenVMS class drivers, DUDRIVER and DKDRIVER, or a non-shadowed drive supported by DSDRIVER. Supported drives thus include RA, RD, RF, RZ, and any other drives supported by these class drivers.

---

#### CAUTION

---

Any previous data on a member volume is lost when the RAID INITIALIZE command is entered.

---

By default, the RAID INITIALIZE command asks for confirmation before initializing the members. You may override this prompt with the /NOCONFIRM qualifier.

RAID INITIALIZE mounts each member and writes the RAID additional control information to the media. RAID5 initializations will also initialize the user data blocks to all zeroes. The RAID INITIALIZE command then dismounts the members. The RAID virtual device is not created until the RAID array is bound using the RAID BIND command.

## RAID INITIALIZE

The RAID INITIALIZE command creates volume labels for the members based upon the *array-id*. The software generates the first 12 characters of the volume label (the first 10 characters of the RAID array name, and 2 hex characters that uniquely identify the particular member), but you may change the *volume-id* to suit your needs.

HP RAID Software will occasionally change volume labels to avoid conflict with previously mounted disks. The system administrator may also change the volume labels.

For RAID5, the RAID INITIALIZE command can take a significant time to complete, especially when you are using SCSI devices in your RAID arrays.

### Qualifiers

#### **/CHARACTERISTIC\_SIZE=blocks** RAID5

Specifies the characteristic size of the RAID array. If not specified via this qualifier, the characteristic size defaults to the characteristic size of the smallest device in the unit-list. Refer to the software release notes for a valid range and the default value.

#### **/CHUNK\_SIZE=n**

Specifies the organization of RAID array data on the RAID array member. Logical block numbers are assigned to members in chunks of size *n*. That is, the first *n* blocks are contained on one member unit, the next *n* on the next member, and so on. If not specified, chunk size defaults to the value specified in the release notes. Refer to the software release notes for a valid range. Refer to Section 3.3 for further information on chunk size.

#### **/CONFIRM (Default)**

#### **/NOCONFIRM**

If the /CONFIRM qualifier is specified, the RAID INITIALIZE command prompts for confirmation that the correct units were selected. The RAID INITIALIZE command requires an affirmative response prior to writing on the member volumes. Any other response ends the operation abnormally.

---

#### **CAUTION**

---

If the /NOCONFIRM qualifier is specified, the RAID INITIALIZE command will not prompt for confirmation and will write over data on the member volumes.

---

#### **/FILES\_11**

Obsolete qualifier.

#### **/RAID\_LEVEL=0** RAID0

#### **/RAID\_LEVEL=5** RAID5

This qualifier is required when a RAID array is initialized to indicate whether to initialize the array as a RAID0 or RAID5 array.

#### **/SIZE=lbn-size-list**

This qualifier specifies a list of sizes in blocks for the virtual devices. Each list element contains a number to specify the number of blocks to allocate for each virtual device created. Refer to Chapter 5 for more information.

**/TIMEOUT=n seconds (1 to 99999)** RAID5  
**/NOTIMEOUT**

The **/TIMEOUT** qualifier specifies the number of seconds that HP RAID Software will wait for a member I/O to complete before declaring the device unavailable and establish the policy for later use in the bound array. If you do not specify the **TIMEOUT** qualifier, the default value (as specified in the release notes) will be used. The timeout value is persistent. That is, it retains the original value specified across bind and unbind operations until a new value is specified.

Using the **/NOTIMEOUT** qualifier means that the HP RAID Software will never time out a member I/O. See Section 7.6.2 for more information on the HP RAID timeout mechanism.

## Examples

1. `$ RAID INITIALIZE/RAID_LEVEL=0 PAYROLL DUA1:,DUA2:,DUA3:`

This command creates the RAID0 array PAYROLL made up of members DUA1, DUA2, and DUA3.

2. `$ RAID INITIALIZE/RAID_LEVEL=5 PAYROLL DUA1:,DUA2:,DUA3:`

This command creates the RAID5 array PAYROLL made up of members DUA1, DUA2, and DUA3.

## RAID INITIALIZE/SPARE RAID5

---

## RAID INITIALIZE/SPARE RAID5

Initializes disks as spares so that they may be included in a spareset or used as replacement disks in a RAID array.

To use this command, you must have OPER and VOLPRO privileges.

### Format

RAID INITIALIZE/SPARE unit-list

### Parameters

#### **unit-list**

This parameter lists the intentional device names of the disks that will be initialized as spares. The disks must not be mounted, bound, or allocated. These devices must be physical devices or logical names that translate to physical devices.

### Description

The RAID INITIALIZE/SPARE command initializes spares that may be used to replace members of a RAID array.

---

#### **CAUTION**

---

All data on a device is lost when the RAID INITIALIZE/SPARE command is entered.

---

By default, the RAID INITIALIZE/SPARE command will ask for confirmation before initializing the members.

A spare can be any drive supported by the OpenVMS class drivers, DUDRIVER and DKDRIVER, or a nonshadowed drive supported by DSDRIVER. Supported drives thus include RA, RD, RF, RZ, and any other drives supported by these class drivers.

The RAID INITIALIZE/SPARE command mounts each unit and writes the RAID control information on each disk. The RAID INITIALIZE/SPARE command does not leave the members mounted.

### Qualifiers

#### **/CONFIRM (Default)**

#### **/NOCONFIRM**

If the /CONFIRM qualifier is specified, the RAID INITIALIZE/SPARE command prompts for confirmation that the correct units were selected. The RAID INITIALIZE/SPARE command requires an affirmative response prior to writing on the spare volumes. Any other response ends the operation abnormally.

If the /NOCONFIRM qualifier is specified, the RAID INITIALIZE/SPARE command will not prompt for confirmation, regardless of what is on the spare volumes.

## Examples

1. `$ RAID INITIALIZE/SPARE DUA1:, DUA2:, DUA3:`

Disk DUA1, DUA2, and DUA3 are initialized so that they may be used as spares.

## RAID MODIFY

---

### RAID MODIFY

Modifies the characteristics of a RAID array. This command takes effect on all nodes in a VMScLuster configuration.

To use this command, you must have OPER privileges.

#### Format

```
RAID MODIFY array-id
```

#### Parameters

**array-id**

The *array-id* is the identifier for the bound RAID array.

#### Description

The RAID MODIFY command changes the characteristics of the RAID array. All modifications except spareset associations are permanent (that is, they remain in effect across bind and unbind operations).

#### RESTRICTIONS

The RAID MODIFY command will not work on unbound RAID arrays. A qualifier is mandatory with this command.

#### Qualifiers

**/ARRAY\_NAME=new-array-id**

Specifies a new name for the RAID array. The *new-array-id* is specified as a 1- to 32-character string containing only alphanumeric characters, including the dollar sign (\$) and underscore (\_). This identifier is used by other commands to identify the RAID array. The new name must now be used in any subsequent RAID commands.

**/ASSOCIATED\_SPARESET=set-id** RAID5

**/NOASSOCIATED\_SPARESET**

Associates the specified spareset with the specified RAID array. Any previous association is overwritten.

The specified spareset must have a characteristic size that is greater than or equal to the RAID array characteristic size. If not, the association to the spareset will not be made.

Specifying the /NOASSOCIATED\_SPARESET qualifier causes a previous association to disassociate.

Not specifying either the /ASSOCIATED\_SPARESET or /NOASSOCIATED\_SPARESET qualifier will leave the associations specified at RAID BIND.

**/TIMEOUT=n seconds (1 to 99999)** RAID5

**/NOTIMEOUT**

The /TIMEOUT qualifier specifies the number of seconds that the HP RAID software will wait for a member I/O to complete before declaring the device unavailable and using RAID algorithms to complete the I/O. The timeout value



is persistent. That is, it retains the original value specified across binds and unbinds until a new value is specified.

Using the /NOTIMEOUT qualifier means that the HP RAID software will never time out a member I/O.

---

**Note**

---

NOTIMEOUT will automatically override the IO\_TIMEOUT. The IO\_TIMEOUT will be disabled.

---

Not specifying the /TIMEOUT qualifier means that the last value specified will continue to be used. If a value was never specified, the default value (specified in the release notes) will be used. See Section 7.6.2 for more information on the HP RAID timeout mechanism.

### Examples

1. `$ RAID MODIFY PAYROLL/ASSOCIATED_SPARESET=SPARE1`

This command allows certain RAID array attributes to be changed. In this example, an association between RAID array PAYROLL and spareset SPARE1 is made.

2. `$ RAID MODIFY PAYROLL/ARRAY_NAME=PAY93`

This command changes the RAID array name from PAYROLL to PAY93.

## RAID REMOVE RAID5

---

### RAID REMOVE RAID5

Removes a member from a RAID array. This command takes effect on all nodes in a VMScluster configuration.

To use this command, you must have OPER privileges.

#### Format

```
RAID REMOVE array-id unit-id
```

#### Parameters

##### **array-id**

The *array-id* is the identifier for the bound RAID array.

##### **unit-id**

Specifies the device name of the member to be removed from the RAID array.

#### Description

The RAID REMOVE command removes the specified unit from the RAID5 array. RAID5 arrays missing one member continue to provide access to user data. There may be a brief delay before the removed device is dismounted.

If the RAID5 array is in the process of reconstructing, this command stops the reconstruct if the *unit-id* specified is the member being reconstructed. Removing the reconstructing member causes the RAID5 array to go into a reduced state. When the RAID5 array is in the process of reconstructing, you cannot remove any member other than the one being reconstructed.

Removing a member from a RAID5 array that has an associated spareset causes an automatic replacement of the removed RAID array member if the spareset is not empty. HP RAID Software for OpenVMS starts reconstruction of the missing member onto the replaced member.

#### Examples

1. \$ RAID REMOVE PAYROLL DUA1:

This command removes a single member, DUA1, from RAID array PAYROLL.

---

## RAID REMOVE/SHADOW\_MEMBER RAID0

Removes a member from a RAID shadow set. This command takes effect on all nodes in a VMSccluster configuration.

To use this command, you must have OPER privileges.

### Format

```
RAID REMOVE/SHADOW_MEMBER array-id unit-id
```

### Parameters

**array-id**

The *array-id* is the identifier for the bound RAID array.

**unit-id**

Specifies the device name of the member to be removed from the RAID shadow set.

### Description

The RAID REMOVE/SHADOW\_MEMBER command on a RAID0 array removes a shadow set member from the shadow set, provided there are at least two devices in the shadow set.

### Examples

1. `$ RAID REMOVE/SHADOW_MEMBER PAYROLL $3$DUA1:`

This command example removes device \$3\$DUA1, which must be a member of a shadow set with two or more members.

## RAID REMOVE/SPARE RAID5

---

## RAID REMOVE/SPARE RAID5

Removes a spare from a spareset. This command takes effect on all nodes in a VMScluster configuration.

To use this command, you must have OPER privileges.

### Format

```
RAID REMOVE/SPARE set-id unit-id
```

### Parameters

#### **set-id**

The *set-id* is an identifier for the bound spareset. This identifier is used by other commands to identify sparesets.

The remainder of spares currently bound in the spareset will remain unaffected.

#### **unit-id**

Specifies the device name of the spare to be removed from the spareset.

### Description

The RAID REMOVE/SPARE command removes the specified unit from the spareset. The specified unit is still a valid spare and may be used in RAID BIND /SPARESET, RAID ADD/SPARESET and RAID REPLACE/DEVICE commands.

### Examples

1. `$ RAID REMOVE/SPARE SPARE1 DUA1:`

This command removes spare DUA1 from spareset SPARE1.

---

## RAID REPLACE RAID5

Replaces the missing member of a RAID array. This command takes effect on all nodes in a VMSccluster configuration.

To use this command, you must have OPER privileges.

### Format

```
RAID REPLACE array-id replacement-id
```

### Parameters

**array-id**

The *array-id* is the identifier for the bound RAID array.

**replacement-id**

Specifies the *set-id* of the spareset from which a spare will be obtained. If the */DEVICE* qualifier is specified, the *replacement-id* is the device name of the spare to be used.

### Description

The RAID REPLACE command uses the spare specified or obtains a spare from the specified spareset and uses it to replace the missing member of a RAID array. The new member's data will be reconstructed from the rest of the RAID array.

### Qualifiers

**/DEVICE**

Indicates *replacement-id* is a spare rather than a spareset. This device must have been initialized as a spare using the RAID INITIALIZE/SPARE command. The device must not be a member of a spareset and must not be mounted or allocated.

### Examples

1. \$ RAID REPLACE PAYROLL SPARE1

This command obtains a spare from SPARE1 and replaces the missing member of RAID array PAYROLL.

2. \$ RAID REPLACE PAYROLL DUA32:/DEVICE

This command replaces a missing member of RAID array PAYROLL with the spare DUA32, which must not be in a spareset.

## RAID SET

---

### RAID SET

Sets limits on system resources to be used by HP RAID. This command applies only to the issuing node.

To use this command, you must have OPER privileges.

#### Format

RAID SET

#### Description

HP RAID Software for OpenVMS will be given the limits to system resources that will be used by the software. HP recommends this default value for most situations.

#### Qualifiers

**/PAGESMAX=pages** RAID5  
**/NOPAGESMAX**

The /PAGESMAX qualifier specifies the maximum number of pages of memory taken from the OpenVMS free-list that may be utilized by the HP RAID Software for OpenVMS on the issuing node at any given time.

The /NOPAGESMAX qualifier allows the HP RAID Software to use available resources to the extent needed. When the RAID\$SERVER process is started, the default is /NOPAGESMAX. HP recommends /NOPAGESMAX as the default.

**/POOLMAX=bytes**  
**/NOPOOLMAX**

The /POOLMAX qualifier specifies the maximum number of bytes of the non-paged pool utilized by the HP RAID Software for OpenVMS on the issuing node at any given time.

The /NOPOOLMAX qualifier allows the HP RAID Software to use available resources to the extent needed. When the RAID\$SERVER process is started, the default is /NOPOOLMAX. HP recommends /NOPOOLMAX as the default.

**/RECONSTRUCT** RAID5  
**/NORECONSTRUCT**

This qualifier is issued on a per-node basis and only applies to the node on which it is issued. The /NORECONSTRUCT qualifier will not allow the node on which it was issued to initiate any new reconstructs. In addition, reconstructs in progress on this node will stop and continue on another available node. The default is reconstruct.

When the RAID\$SERVER process is started, the default is /RECONSTRUCT. The RAID SHOW/FULL command will display the state of this setting.

## Examples

1. `$ RAID SET/NORECONSTRUCT`

This command disallows reconstructs on the node where the command is issued.

2. `$ RAID SET/PAGESMAX=1000`

This command sets limits on the number of pages of memory taken from the OpenVMS free list by the HP RAID Software for OpenVMS.

## RAID SHOW

---

### RAID SHOW

Reports the configuration, characteristics, and performance data of a RAID array. RAID array operations shown in the output of the RAID SHOW command are for the issuing node only. No privileges are required to use this command.

#### Format

```
RAID SHOW [array-id]
```

#### Parameters

##### **array-id**

The *array-id* is the identifier for the bound RAID array. If this parameter is omitted, information about all bound RAID arrays will be displayed.

#### Description

The RAID SHOW command lists the characteristics, configuration, and performance data of the specified RAID array. This includes a list of all the members, the associated spareset, and performance statistics.

#### Qualifiers

##### **/BRIEF**

Displays a quick overview of RAID arrays. The state field indicates whether or not an array is in a NORMAL state or needs further checking. The state SHADOWING indicates that a RAID 0+1 array has active shadow management operation (MERGE or COPY).

Member devices in parentheses need further checking.

The brief report is directly obtained from the driver source and therefore faster than a regular SHOW or SHOW/FULL.

##### **/FULL**

Displays the basic subset of information, plus additional information on the RAID virtual device and RAID array.

##### **/OUTPUT=file-spec**

Directs the output of the RAID SHOW command to the specified file. If this qualifier is not specified, the output is directed to SYSS\$OUTPUT.

##### **/RAID\_LEVEL=0**

##### **/RAID\_LEVEL=5**

This qualifier will list either RAID 0 Arrays or RAID 5 Arrays.

A sample printout of a RAID SHOW command for a RAID5 array is shown in RAID SHOW Command for RAID5 Array. RAID SHOW command for RAID5 Array is described in detail in Description of RAID SHOW Command Printout.



A sample printout of the RAID SHOW command using the BRIEF qualifier is shown in Example 11-2. The details for the callouts are shown in Table 11-4.

#### Example 11-2 RAID SHOW/BRIEF Command

```

1 $ RAID SHOW/BRIEF
   RAID Array ID   State      Members
2 PAYROLL_1       RECONSTRUCT $1$DKD300,$1$DKD00,($1$DKD500)
3 ALLIN1USER      NORMAL     $3$DUA76,$3$DUA71
4 DB1             SHADOWING  DSA6001,$1$DUA11,(DSA6002,$1$DUA21,$1$DUA22)

```

**Table 11-4 Description of RAID SHOW Command Printout**

Number	Description
1	The command line displays information about the array.
2	The array is reconstructing member device \$1\$DKD500
3	The array is in normal state.
4	The array has shadow management operation going on for DSA6002 with shadow set members \$1\$DUA21 and \$1\$DUA22. A DCL "SHOW DEVICE DSA6002" can be used to find out more detail.

A sample printout of the RAID SHOW command using the default is shown in Example 11-3. The details for the callouts are shown in Table 11-5.

# RAID SHOW

## Example 11-3 RAID SHOW Command for RAID5 Array (Default Version)

1 \$ RAID SHOW MYARRAY\_5

HP RAID Software V3.0 Display Time: 13-NOV-2004 14:10:25.87.  
© 2004 Hewlett-Packard Development Company, L.P. All Rights Reserved.

2 RAID Array Parameters:

Current RAID Array ID: MYARRAY\_5  
Permanent RAID Array ID: MYARRAY\_5  
RAID Level: 5  
Current State: NORMAL  
Associated Spareset: (none)

3 RAID Array Configuration:

Member		
Index	Name	State
0	_\$1\$DKB100:	NORMAL
1	_\$1\$DKC100:	NORMAL
2	_\$1\$DKC200:	NORMAL

Virtual Unit	Size	Status	Recovered Operations	Reads	Writes	Errors
DPA0123:	17992	INACCESS	0	0	0	0

**Table 11–5 Description of RAID SHOW Command Printout**

Number	Description
1	The command line displays information about the array.
2	This field presents the current and permanent names of the RAID array, RAID level, current state of the array, and whether or not there is an associated spareset. If there is no associated spareset, this field indicates "none".
3	This field presents the device names of the members of the RAID array and their state.
4	This field presents the name of the virtual unit, its size, its status, the number of reads and writes, and the number of errors.

This is a per-node status and does not apply to the RAID array itself, nor to the individual members, but only to the virtual unit.

A sample printout of a RAID SHOW command for a RAID5 array during the reconstruct state is shown in Example 11–4. Example 11–4 is described in detail in Table 11–6.

**Example 11–4 RAID SHOW Command for RAID5 Array (During Reconstruct)**

```

1 $ RAID SHOW MYARRAY_5
HP RAID Software V3.0      Display Time:  4-NOV-2004 15:40:43.46
© 2004 Hewlett-Packard Development Company, L.P. All Rights Reserved.

2 RAID Array Parameters:
    Current RAID Array ID:  MYARRAY_5
    Permanent RAID Array ID: MYARRAY_5
    RAID Level:             5
    Current State:          RECONSTRUCTING
                           Operation 7.89% complete
                           In progress on node NODEA
    Associated Spareset:    MYSPARESET

3 RAID Array Configuration:
    Member
    Index  Name                State
    -----
    0      _$3$DUA216:          RECONSTRUCTING
    1      _$3$DUA213:          NORMAL
    2      _$3$DUA214:          NORMAL

4 Virtual
    Unit   Size   Status   Recovered   Reads   Writes   Errors
    -----
    DPA0002: 1778392 ACCESS      0          226      853      0

```

## RAID SHOW

**Table 11–6 Description of RAID SHOW Command Printout (Brief Version During Reconstruct)**

Number	Description
1	The command line displays information about the array.
2	This field presents the current and permanent names of the RAID array. This is the spareset associated with the array. If there is no associated spareset, this field indicates "none". During the reconstructing state additional information indicating percent completion and the node on which reconstruction is occurring is displayed. If there are no nodes to reconstruct, then this information is not displayed.
3	This field presents the device names of the members of the RAID array and their state.
4	This field presents the name of the virtual unit, its size, how many recoveries occurred, the number of reads and writes, the number of errors, and if the virtual unit is accessible.

A sample printout of a RAID SHOW command for a RAID0 array is shown in Example 11–5. Example 11–5 is described in detail in Table 11–7.

### Example 11–5 RAID SHOW Command for RAID0 Array

```
1 $ RAID SHOW MYARRAY_0
HP RAID Software V3.0 Display Time: 13-NOV-2004 14:09:24.65
© 2004 Hewlett-Packard Development Company, L.P. All Rights Reserved.

2 RAID Array Parameters:
    Current RAID Array ID: MYARRAY_0
    Permanent RAID Array ID: MYARRAY_0
    RAID Level: 0
    Current State: NORMAL

3 RAID Array Configuration:
    Member
    Index Name State
    -----
    0 _$1$DKA200: NORMAL
    1 _$1$DKA300: NORMAL

4 Virtual
  Unit Size Status Reads Writes Errors
  -----
DPA0099: 471592 ACCESS 0 0 0
```

Table 11–7 Description of RAID SHOW Command Printout (Brief Version)

Number	Description
1	The command line displays information about the array.
2	This field presents the current and permanent names of the RAID array, RAID level, and current state of the array.
3	This field presents the device names of the members of the RAID array and their state.
4	This field presents the name of the virtual unit, its size, its status, the number of reads and writes, and the number of errors.  This is a per node status and does not apply to the RAID array itself, nor to the individual members, but only to the virtual unit.

A sample printout of a RAID SHOW (full) command for a RAID0 array with shadowed members is shown in Example 11–6. The details are described in detail in Table 11–8.

# RAID SHOW

## Example 11-6 RAID SHOW /FULL Command for RAID0 Array with Shadowed Members

1 \$ RAID SHOW MYARRAY\_0 /FULL

HP RAID Software V3.0 Display Time: 13-NOV-2004 14:09:24.65  
© 2004 Hewlett-Packard Development Company, L.P. All Rights Reserved.

2 RAID Array Parameters:

Current RAID Array ID: MYARRAY\_0  
Permanent RAID Array ID: MYARRAY\_0  
Date Created: 13-NOV-2004 14:04:25.14  
Last Bind: 13-NOV-2004 14:07:13.13  
RAID Level: 0+1  
Current State: NORMAL

3 Characteristic Size: 236904 Total Capacity: 471592  
Member Count: 2 Chunk Size: 120  
Pool in Use: 4736 Max Pool Value: 0  
Pages in Use: 0 Max Page Value: 0

4 RAID Array Configuration:

Member Index	Name	State	ShadowSet Members	ShadowSet State
0	_DSA6000: _ \$1\$DKB0: _ \$1\$DKA200:	NORMAL	2	ShadowCopying
1	_DSA6001: _ \$1\$DKA300:	NORMAL	1	SteadyState

5 RAID Array Operations:

Member Index	Name	Reads	Writes	Errors
0	_DSA6000:	6	2	0
1	_DSA6001:	10	2	0
Member Total:		16	4	0

6 Virtual Unit Size Status Reads Writes Errors

Virtual Unit	Size	Status	Reads	Writes	Errors
DPA0099:	471592	ACCESS	0	0	0

7 Histogram of I/O sizes, Virtual Unit DPA0099:

Blocks/IO	Reads	Writes	Total
1	0	0	0
2	0	0	0
4	0	0	0
8	0	0	0
16	0	0	0
32	0	0	0
64	0	0	0
128	0	0	0
256	0	0	0
512	0	0	0
more than 512	0	0	0

Table 11–8 Description of RAID SHOW /FULL Command Printout

Number	Description
1	The command line displays information about the array.
2	This field presents the current and permanent names of the RAID array, the current state of the RAID array, the date when the RAID array was created, and the date of the last BIND command for this array. In addition, the RAID level is also displayed. In this example, 0+1 RAID LEVEL is displayed, indicating a RAID0 array with shadowing. Note that there is only one state valid for a RAID0 or RAID0+1 array (NORMAL), but several statuses that are valid for all RAID arrays (STARTUP, INOPERATIVE, nothing). These statuses are described in Section 7.1.
3	This field presents the RAID array characteristic size, total capacity of the RAID array, the number of RAID array members, the chunk size, the timeout value, the amount of pool and pages in use, and the MAXPOOL and MAXPAGE values. The amount of pool and pages, and the MAXPOOL and MAXPAGE values are for the issuing node only.
4	This field presents the shadow set virtual units, the device names of each shadow set member, the state of the RAID array member, the number of members in each shadow set, and the state of each shadow set (SteadyState, ShadowCopying, ShadowMerging). The Index position of each RAID array member is also displayed, starting with Index 0.
5	This field presents the number of read and writes processed by each member of the RAID array, the number of read/write errors for each member, and the total number of reads, writes, and errors for each member. These numbers are for the issuing node only and are reset upon a BIND command.
6	This field presents the name of the virtual unit, its size, the number of reads and writes, and the number of errors. In addition, a virtual unit status field is displayed to indicate whether the virtual unit is accessible from the issuing node. Number of reads, writes, errors, and the status are for the issuing node only, and are reset upon a BIND command. Refer to Example 11–3 of the RAID SHOW command for more details on the status field.
7	A histogram of I/O sizes for the virtual unit

## RAID SHOW

A sample printout of a RAID SHOW command for a RAID0 array in the inoperative state is shown in Example 11-7. Example 11-7 is described in detail in Table 11-9.

### Example 11-7 RAID SHOW Command for RAID0 Array (With Inoperative Status)

```
1 $ RAID SHOW MYARRAY_SHAD
HP RAID Software V3.0 Display Time: 18-NOV-2004 20:56:05.33
© 2004 Hewlett-Packard Development Company, L.P. All Rights Reserved.
2 RAID Array Parameters:
    Current RAID Array ID:    MYARRAY_SHAD
    Permanent RAID Array ID: MYARRAY_SHAD
    RAID Level:              0+1
    Current State/Status:    NORMAL / INOPERATIVE
3 RAID Array Configuration:
    Member
    Index  Name          State      ShadowSet  ShadowSet
    -----  -----  -----  -----  -----
    0      _DSA300:    NORMAL    2         SteadyState
    1      _DSA301:    NORMAL    2         SteadyState
    2      _DSA302:    NORMAL    2         SteadyState
4 Virtual
    Unit      Size      Status      Reads      Writes      Errors
    -----  -----  -----  -----  -----  -----
    DPA0030:  2488312  ACCESS     0          0          0
```

**Table 11-9 Description of RAID SHOW Command Printout (With Inoperative Status)**

Number	Description
1	The command line displays information about the array.
2	This field presents the current and permanent names of the RAID array, the current state and status of the RAID array, the date when the RAID array was created, and the date of the last BIND command for this array. In addition, the RAID level is also displayed. In this example 0+1 RAID LEVEL is displayed, indicating a RAID0 array with shadowing. Note that there is only one state valid for a RAID0 or RAID0+1 array (NORMAL) but several statuses that are valid for all RAID arrays (STARTUP, INOPERATIVE, nothing.)

(continued on next page)



**Table 11–9 (Cont.) Description of RAID SHOW Command Printout (With Inoperative Status)**

Number	Description
3	This field presents the shadow set virtual units, and the state of each shadow set (SteadyState, ShadowCopying, ShadowMerging). The Index position of each RAID array member is also displayed, starting with Index 0.
4	This field presents the name of the virtual unit, its size, the number of reads and writes, and the number of errors. In addition, a virtual unit status field is displayed to indicate whether the virtual unit is accessible from the issuing node. Number of reads, writes, errors, and the status are for the issuing node only, and are reset upon a BIND command.

## RAID SHOW/SPARESET RAID5

---

## RAID SHOW/SPARESET RAID5

Reports the configuration and characteristics of a spareset. Using this command requires no privileges.

### Format

```
RAID SHOW/SPARESET [set-id]
```

### Parameters

**set-id**

The *set-id* is an identifier for the bound spareset. If this parameter is omitted, the characteristics of all bound sparesets are displayed.

### Description

The RAID SHOW/SPARESET command lists the characteristics of the specified spareset. This includes a list of all the member spares.

### Qualifiers

**/OUTPUT=file-spec**

Directs the output of the RAID SHOW/SPARESET command to the specified file. If this qualifier is not specified, the output is directed to SYSSOUTPUT.

This is a sample printout of a RAID SHOW /SPARESET command that shows the configuration data of spareset MYSPARESET. The following example is described in detail in Table 11–10.

#### Example 11–8 RAID SHOW /SPARESET Command

```

1 $ RAID SHOW/SPARESET MYSPARESET
2 HP RAID Software V3.0    Display Time:  4-NOV-2004 14:58:15.33
  © 2004 Hewlett-Packard Development Company, L.P. All Rights Reserved.
3           Spareset ID:           MYSPARESET
           Last Bind Date:        4-NOV-2004 14:54:29.19
           Characteristic Size:    889736
4   Spareset Member Information:
      Member                        Characteristic Size      State
      -----                        -
      _$3$DUA216:                   889736                 BOUND
      _$3$DUA215:                   889736                 BOUND
5                                     Associated RAID Arrays
                                     -----
                                     MYARRAY

```

**Table 11–10 Description of RAID SHOW/SPARESET Command Printout**

Number	Description
1	The command line displays information about the array.
2	The software ID and version number of the software installed on your system.
3	The name of the spareset, the date it was last bound, and the characteristic size of the spareset.
4	The device names of the spares in the spareset, their characteristic sizes, and their status.
5	Any RAID arrays associated with the spareset.

## RAID SHUTDOWN

---

### RAID SHUTDOWN

Informs the HP RAID software that the system is about to shut down. This command should be placed in your system shutdown file, SYSSMANAGER:SYSHUTDOWN.COM. This command operates only on the issuing node.

To use this command, you must have OPER privileges.

#### Format

```
RAID SHUTDOWN
```

#### Restrictions

The RAID SHUTDOWN command should only be issued immediately before a system shutdown. It is not reversible.

#### Description

This command informs the RAID subsystem that the node is about to shutdown. The RAID software then prepares its arrays and the members of those arrays for the shutdown.

This command will unbind all arrays on the issuing node providing the RAID virtual devices are dismounted.

#### Examples

1. \$ RAID SHUTDOWN

This command allows the RAID subsystem to prepare for a node shutdown.

---

## RAID UNBIND

Sets a RAID virtual device off line and dissolves the association between the RAID array and its virtual device(s).

To use this command, you must have OPER privileges.

### Format

```
RAID UNBIND array-id
```

### Restrictions

The RAID virtual device must be dismounted before the RAID array can be unbound.

### Parameters

**array-id**

The *array-id* is the identifier for the bound RAID array.

### Description

The RAID UNBIND command sets the specified RAID virtual device off line, dismounts the members, and dissolves any shadow sets for RAID arrays. When executed on one node in a VMSccluster configuration, the command is propagated to all nodes of the cluster. For RAID5 arrays, associations with any RAID5 sparesets are dissolved.

### Examples

1. \$ RAID UNBIND PAYROLL

This command frees the members of RAID array PAYROLL and dissolves the association between the RAID array and its virtual device. The RAID UNBIND command requires the RAID array name and not the virtual device name.

## RAID UNBIND/SPARESET RAID5

---

### RAID UNBIND/SPARESET RAID5

This command dissolves associations of the spareset with all RAID arrays.

To use this command, you must have OPER privileges.

#### Format

```
RAID UNBIND/SPARESET set-id
```

#### Parameters

**set-id**

The *set-id* is an identifier for the spareset. This identifier is used by other commands to identify sparesets.

#### Description

RAID UNBIND/SPARESET dissolves the association of a spareset with its spares and associations of that spareset with all RAID arrays. Data on any spares is unchanged. The devices are still spares. The spare devices are dismounted. This command takes effect on all nodes in a VMSccluster configuration.

#### Examples

1. `$ RAID UNBIND/SPARESET SPARE1`

The RAID UNBIND/SPARE command dissolves the association of spareset SPARE1 with its spares and with all RAID arrays.

---

## RAID OPCOM Information Messages

This appendix contains the OPCOM messages that are issued by the HP RAID software. Refer to the *OpenVMS System Messages: Companion Guide for Help Message utility* for explanations of messages issued by the OpenVMS operating system.

### A.1 Message Format

The messages documented in this appendix are displayed in the following format:

```
%RAID-s-mnemonic, text  
-RAID-s-mnemonic, text
```

where:

% is the prefix to all primary messages.

– is the prefix to any continuation messages.

RAID is the source designation of the message.

*s* is the severity level of the message.

*mnemonic* is an abbreviation of the message text.

*text* is the expanded text of the message.

#### Severity Level

The severity level of the OPCOM error messages is one of the following:

**Table A–1 Severity Levels**

Code	Meaning
S	Success—Successful completion of the request
I	Informational—May or may not require user action
W	Warning—Request may not have completed and may require user action
E	Error—System encountered an error which may be recoverable
F	Fatal—System encountered a fatal error and cannot continue processing this request

## A.2 RAID OPCOM Messages

The following message descriptions are alphabetized by the mnemonic portion of the message. The message prefix, source designation, and severity code are not shown.

DUMP, the server process has dumped,

**Fatal:** The RAID\$SERVER process has terminated. In most cases process dump SYSSMANAGER:RAID\$SERVER\_MAIN.DMP is created. Please keep a copy of this file for further analysis.

NEWNODECOMP, all implicit binds completed on node *node-id*,

**Informational:** A RAID\$SERVER has started and completed binds for all existing arrays.

EXLICENSE, number of bound RAID array members exceeds active license limits,

**Warning:** There are not enough license units to bind this array.

RECONCOMP, reconstruction complete on RAID array *array-id*

**Informational:** Reconstruction has completed on the specified RAID array. The array is now fully redundant.

RECONINC, reconstruction incomplete on RAID array *array-id*,

**Informational:** Reconstruction has completed on the specified RAID array. The array is not fully redundant. Use DCL command "RAID ANALYZE /ARRAY/REPAIR *array-id*" to find out more details.

REMOVE, device *unit-id* has been removed from RAID array *array-id*

**Informational:** A RAID array member has been removed. The array is no longer redundant.

REPLACE, missing drive in RAID array *array-id* replaced by *unit-id*

**Informational:** A missing member in the specified RAID array has been replaced. It should begin reconstructing soon.

SPAREREMOV, spare *unit-id* was removed from spareset *set-id*

**Informational:** A spare has been removed from the indicated spareset because of a RAID REMOVE/SPARE command or something wrong with the spare. Refer to the diagnostic log for the cause.

SPARESETEMPY, spare *unit-id* was removed from spareset *set-id*, spareset *set-id* now empty

**Informational:** A spare has been removed from the indicated spareset because of a RAID REMOVE/SPARE command or something wrong with the spare. Refer to the diagnostic log for the cause. The indicated spareset no longer has spares. You should resupply the spareset with spares.



---

## RAID Command Messages

This appendix contains the informational messages that are issued by and are unique to RAID commands. Some messages may appear at different times with different severity levels. Not all variations of the severity level are documented in this appendix.

Refer to the *OpenVMS System Messages: Companion Guide for Help Message utility* for explanations of messages issued by the OpenVMS operating system.

### B.1 Message Format

The messages documented in this appendix are displayed in the following format:

```
%RAID-s-mnemonic, text
-RAID-s-mnemonic, text
```

where:

% is the prefix to all primary messages.

– is the prefix to any continuation messages.

RAID is the source designation of the message.

s is the severity level of the message.

*mnemonic* is an abbreviation of the message text.

*text* is the expanded text of the message.

#### Severity Level

The severity level of the RAID command messages is one of the following:

Code	Meaning
I	Informational—May or may not require user action
W	Warning—Request may not have completed and may require user action
E	Error—System encountered an error which may be recoverable
F	Fatal—System encountered a fatal error and cannot continue processing this request

## B.2 RAID CLI and Server Messages

The following message descriptions are alphabetized by the mnemonic portion of the message. The message prefix, source designation, and severity code are not shown.

ADDFAILED, failed to add device *unit-id* to spareset *set-id*

**Informational:** During an ADD/SPARESET command, the server failed to add a specified device to the spareset. Additional status will be provided indicating the reason for failure.

**User Action:** Refer to additional status information to determine why the device cannot be added.

ALLUNITSREQ, all specified units must be valid for this command

**Fatal:** Certain commands require all command line units to be valid and accessible. At least one of the specified devices was not valid.

**User Action:** Correct the command line unit list.

ALREADYBOUND, RAID array is already bound

**Informational:** A RAID array with the same name specified in a RAID BIND command has already been bound. You cannot rebind a previously bound array without first unbinding it.

**User Action:** Unbind the array, or bind the new array with the /OVERRIDE qualifier.

AMBIGRA, unit-list contains members of more than one RAID array

**Fatal:** Units listed on command line were members of different RAID arrays. Each array had the same name as that specified by the user.

**User Action:** Determine the correct membership of the array.

ANERR, check analyze report,

**Warning:** The analyze command has found one or more inconsistencies.

**User Action:** Refer to chapter Creating and Managing RAID Arrays.

ANFE, software forced error flags detected,

**Warning:** The analyze command has found forced error flags set.

**User Action:** Use DCL command "ANALYZE/DISK/READ *unit-id*" on all virtual units of this array to find all files affected by this error. Files reported with parity error need to be replaced.

ANNOREPAIR, array not RAID 5, not in NORMAL or RECONSTRUCT state,

**Informational:** The /REPAIR qualifier is only valid for RAID 5 arrays in either NORMAL or RECONSTRUCT state.

**User Action:** For a RAID 5 array use the RAID REPLACE command to start an array reconstruction and repeat the command when the array has changed state to NORMAL.

ANPMD, invalid parity blocks detected,

**Warning:** The analyze command has found non-redundant areas.

**User Action:** Use RAID ANALYZE/ARRAY/REPAIR to recover full redundancy. Refer to chapter Creating and Managing RAID Arrays for more details.

ANREPERR, check repair report,

**Error:** The repair could not recover full redundancy for this array.

**User Action:** Use DCL command "ANALYZE/DISK/READ *unit-id*" on all virtual units of this array to find all files affected by this error. Files reported with parity error need to be replaced.

ANREPPAR, parity error on device virtual-unit-id logical block number *block-number*,

**Error:** A repair could not recover full redundancy for the logical blocks listed on this virtual unit.

**User Action:** Use DCL command "ANALYZE/DISK/READ *unit-id*" on all virtual units of this array to find all files affected by this error. Files reported with parity error need to be replaced.

BADCHARSIZE, unit *unit-id* is too small for specified characteristic size

**Fatal:** The specified device does not have enough LBNs, or the specified characteristic size is too large to allow it to be included in the RAID array or spareset specified.

**User Action:** Use a larger device or change the characteristic size.

BADSPARE, unit-id failed testing and was removed from spareset *spare-id*,

**Warning:** The specified spare failed the periodic I/O test and was removed from the spare set.

**User Action:** None.

BODYVERMIS, metadata descriptor body version rev-num is out of rev,

**Error:** An inconsistency was detected in RAID's on-disk structure.

**User Action:** Re-INITIALIZE the array.

BUG, a software bug was detected at:

*\_line line-number, file filename*

**Fatal:** The message signaled for conditions that the software is not prepared to handle. This message will usually have accompanying messages that provide more detailed information.

**User Action:** See additional messages for detail. If these do not clarify what is wrong, contact HP customer support and provide them with all the accompanying information.

CANTALLOC, device *unit-id* could not be allocated

**Fatal:** This message is returned by the RAID INITIALIZE command when one of the specified devices cannot be allocated. This usually indicates that another process has the device allocated.

**User Action:** Determine why the device cannot be allocated and free it before reissuing the RAID INITIALIZE command. SHOW DEVICE/FULL should reveal this information.

CANTASSIGN, Unable to assign a channel to *unit-id*

**Fatal:** The specified device could not be accessed. A typical reason for the HP RAID software being unable to access a device is due to the device being used by another user.

**User Action:** Make sure the device is accessible by the HP RAID software.

CANTREDUCE, BIND failed: one member had an error and removal inhibited by /NOREMOVAL\_ALLOWED

**Fatal:** A bind failed due to one member having an error and the /NOREMOVAL\_ALLOWED qualifier was specified.

**User Action:** Repair the unavailable device or bind RAID array without the /NOREMOVAL\_ALLOWED qualifier.

CHRSZTOOSMALL, unit *unit-id* size is too small for array *array-id*

**Fatal:** The characteristic size of a replacement member must be equal to or greater than the characteristic size of the RAID array.

**User Action:** Use a spare with a larger characteristic size.

CHUNKRNG, CHUNK\_SIZE is out of range *minimum ... maximum*

**Fatal:** The Parameter is out of range bounds. The specified value is too small or too large.

**User Action:** Specify allowable value or the default.

CLONEMEMBER, device *device-id* is now a member of cloned raid array *array-id*

**Informational:** This messages indicates that the clone operation was successful and informs the user of the membership of the cloned array.

**User Action:** None. This message informs the user of the membership and the name of the cloned array. It is iterated for each member removed from the target and placed into the cloned array.

CMDFAIL, RAID command failed

**Fatal:** This is a primary message that will be followed by additional messages

**User Action:** Correct the command line based on additional messages.

CMDLANBUG, illegal combination of command elements - check documentation *command-element*.

**Fatal:** This message indicates that one of several illegal combinations of command elements are present in the DCL command line entered. It indicates an internal problem with the RAID software.

CONDONNODE, node *nodename* detected the condition

**Informational:** This message is seen only in conjunction with other messages that describe the problem. This message identifies the node in the cluster that detected the error. This message typically accompanies messages resulting from a command that is distributed cluster wide.

**User Action:** User action depends on the previous messages.

CONFIGFILEACCESS, Unable to access configuration file on any member,

**Fatal:** A BIND command failed because not all on-disk metadata could be updated.

**User Action:** Retry command. If it still fails verify with "SHOW DEVICE unit-id" that all member devices are on-line and ready for mounting on all nodes running the RAID software.

CSTOOSMALL, CHARACTERISTIC\_SIZE is less than minimum allowed  
(*minimum size*)

**Fatal:** The characteristic size specified is too small.

**Informational:** The remaining good devices in spare list will be initialized.

**User Action:** Change the specified characteristic size or use the default.

DEVACCESS, cannot access device *unit-id*

**Informational or Fatal:** The specified device cannot be accessed.

**User Action:** Resolve device conflict.

DEVINUSE, device *unit-id* is already in use

**Fatal:** The operation being performed found the device already mounted.

**User Action:** Make sure all devices involved in this operation are not mounted and retry the operation.

DEVNOTRESP, device *unit-id* has not responded within the timeout period,

**Informational:** A RAID set member device has failed to complete a metadata I/O within 2 seconds.

**User Action:** Use DCL command "SHOW DEVICE/FULL" and ANALYZE /ERROR\_LOG to find a reason for the timeout.

DIAGNOSTIC, Received *date* from *source* on node *nodename*

**Fatal:** This message provides information that may be useful in tracking RAID subsystem errors.

**User Action:** Contact HP customer support with all accompanying information.

DIFFNAME, this RAID array is already bound with a different name

**Error:** The RAID array specified in a BIND command is already bound with a different name. The unit-list specified in the BIND command is the same as the unit-list of an already bound RAID array. This is possible if two commands that are binding the same array are issued at exactly the same time.

**User Action:** To have the RAID array bound with a new name, use the RAID MODIFY command with the /ARRAY\_NAME qualifier.

DISMOUTIMOUT, timed out waiting for dismount of device *unit-id*,

**Warning:** The specified device failed to dismount within 60 seconds.

**User Action:** Use DCL command "SHOW DEVICE/FULL" and ANALYZE /ERROR\_LOG to find a reason for the timeout.

DPFXVERMIS, metadata descriptor prefix version *rev-num* is out of rev,

**Error:** An inconsistency was detected in RAID's on-disk structure.

**User Action:** Re-INITIALIZE the array.

DRIVER, Error resulting from RAID Driver QIO was detected

**Fatal:** This message indicates that the RAID device driver encountered a fatal error.

**User Action:** This message is signaled in conjunction with other messages, so the user action will depend on what the other messages indicate.

- DRVNOTLOAD, The RAID driver is not loaded.  
Refer to the Operations Guide for information on loading driver  
**Fatal:** The RAID device driver (DPDRIVER) has not been loaded on the system.  
**User Action:** Use the start procedure, SYSSSTARTUP:RAID\$STARTUP to start up the product.
- DSCCONTOVERLAP, metadata descriptor overlaps with container data,  
**Error:** An inconsistency was detected in RAID's on-disk structure.  
**User Action:** Re-INITIALIZE the array.
- DSCCOPIES, metadata descriptor copies num exceeds maximum of 4,  
**Error:** An inconsistency was detected in RAID's on-disk structure.  
**User Action:** Re-INITIALIZE the array.
- DSCOVERLAP, metadata descriptors overlap each other,  
**Error:** An inconsistency was detected in RAID's on-disk structure.  
**User Action:** Re-INITIALIZE the array.
- DUMP, the server process has dumped,  
**Fatal:** The RAID\$SERVER process has terminated. In most cases process dump SYSSMANAGER:RAID\$SERVER\_MAIN.DMP is created. Please keep a copy of this file for further analysis.  
**User Action:** Restart the RAID\$SERVER process using "@SYSSSTARTUP:RAID\$STARTUP SERVER".
- DUPRAIDMEM, duplicate RAID array member *unit-id* was found  
**Fatal:** A valid RAID array was determined from the units listed on the command line. However, at least two of the members listed had identical metadata identifiers. It is not possible to tell which of the members to use.  
**User Action:** Determine which of the units listed in the command line is a duplicate and exclude it. If the same error occurs again, either the other duplicate is the correct member, or more than one member is duplicated. Initialize the duplicate using the OpenVMS INITIALIZE command, take the duplicate off line, or mount the duplicate and try again.
- DUPUNIT, duplicate unit *unit-id* on command line  
**Fatal:** The same device has been specified twice.  
**User Action:** Specify the device only once.
- DUPVUNAME, duplicate virtual device name *device-id*  
**Fatal:** The same virtual unit was specified more than once on the command line.  
**User Action:** The message indicates the virtual device name which is duplicated. Specify a different virtual device name for the second (and subsequent) repetitions of that name.

EXLICENSE, number of bound RAID array members exceeds active license limits,

**Warning:** There are not enough license units to bind this array.

**User Action:** Obtain a SW-RAID5-STG-USER license with more units or UNBIND a bound array.

FAILEDUNITS, one or more units could not be bound into the spareset

**Informational:** One or more of the specified units could not be bound into the spareset.

**User Action:** See additional messages to determine user action.

FAILINIT, OpenVMS INIT failed for device *unit-id*

**Fatal:** Initialization of a specific RAID array member failed.

**User Action:** Dependent upon specific messages accompanying this error.

FAILMOUNT, device *unit-id* could not be mounted

**Fatal:** The HP RAID software was unable to mount the specified device.

**User Action:** Resolve device conflict.

FAILRELABEL, failed to relabel device *devicename* with new volume label *volume-label*

**Error:** This message indicates that the attempt to change the volume label of a device by the RAID software has failed.

**User Action:** Select a different spare device for the replacement.

FAILSHOWRA, no RAID array named *array-id* is bound

**Fatal:** When the SHOW command was entered, the information for the spareset was not available. This could happen during the brief time when the spareset was unbound from another CLI while the SHOW is occurring.

**User Action:** Enter the command again.

FAILSHOWSS, failed to obtain information for Spareset *set-id*

**Fatal:** When the SHOW command was entered, the information for the spareset was not available. This could happen during the brief time when the spareset was unbound from another CLI while the SHOW is occurring.

**User Action:** Reenter with the correct spareset *id*.

FETAGCONTOVERLAP, FE metadata tag data overlaps with container data,

**Error:** An inconsistency was detected in RAID's on-disk structure.

**User Action:** Re-INITIALIZE the array.

FETAGDSCOVERLAP, FE metadata descriptor overlaps with tag data,

**Error:** An inconsistency was detected in RAID's on-disk structure.

**User Action:** Re-INITIALIZE the array.

FETAGTAGOVERLAP, FE metadata tag data overlaps with another copy of tag data,

**Error:** An inconsistency was detected in RAID's on-disk structure.

**User Action:** Re-INITIALIZE the array.

HDRDSCMIS, metadata header and descriptors overlap each other,

**Error:** An inconsistency was detected in RAID's on-disk structure.

**User Action:** Re-INITIALIZE the array.

HWTYPTOOLONG, metadata hardware type string is too long /string/,

**Error:** An inconsistency was detected in RAID's on-disk structure.

**User Action:** Re-INITIALIZE the array.

INCOMPATIBLE, incompatible on-disk structures, re-init the array

**Fatal:** A RAID BIND operation detected on-disk structures that were created by an incompatible version of the software. The array will not work with this newer version of the RAID software.

**User Action:** Bind the array on a system with the older software and back up your data from the virtual device. Use the RAID INITIALIZE command with the new software to re-create the array in the new format. Bind the array and restore your data to the virtual device.

INCOMP SRV, RAID Server's version is incompatible with other RAID Servers in the cluster

**Informational:** RAID command cannot be performed because not all RAID servers in the cluster support this command.

**Fatal:** RAID Server's version too old to join other servers in the cluster.

**User Action:** Install same version of RAID Software on all nodes in the cluster.

INCOMPVER, RAID driver's version *version* is incompatible with the RAID server's version *version*

**Fatal:** This message is generated if either of the following conditions exists:

- A new server was started with an outdated driver on the system.
- A new driver was loaded and old server bind was started.

**User Action:**

- If the problem is an old driver, then reboot your system and load a driver with the same version as the server.
- If the problem is an old server, install the server version that matches the driver.

INCSHAMEM, Incompatible shadow set member

**Fatal:** The member being added to the shadow set is incompatible with the members already present in the shadow set. Compatible members of a shadow set have like values of device type, maximum block value, and the added member must have a non-zero allocation class.

**User Action:** Review the characteristics of the shadow device and verify these with the member attempting to be added.

INITCONFIRM, INIT will destroy existing data, do you want to continue [N]?

**Informational:** This message appears when the RAID INITIALIZE command is used with the /CONFIRM qualifier.

**User Action:** You must enter Y to confirm or N to cancel the initialization.



INITERR, an unrecoverable drive error occurred initializing *unit-id*  
**Fatal:** An I/O error occurred during initialization of an array member.  
**User Action:** Resolve device conflict.

INSFSSMEMBERS, Insufficient shadow set members to perform remove  
**Fatal:** There are too few master/source members of the shadow set to remove this particular member. There must be at least one surviving master member of a shadow set remaining.  
**User Action:** Verify the shadow set membership with the SHOW DEVICE DCL command.

INSTALLCLI, Insufficient privileges to execute command.  
 Ensure that the RAID\$CLI\_MAIN.EXE image is installed correctly.  
**Informational:** This message indicates that the RAID command failed because the process did not have enough privileges to execute the function. Since the RAID\$CLI\_MAIN image is installed with privileges by the RAID startup procedure, if this command fails, one likely reason is that the image is not properly installed.  
**User Action:** Use the startup procedure, SYSS\$STARTUP:RAID\$STARTUP to correctly start up the product.

INUSE, unit *unit-id* is a shadow set member of *array array-id*  
**Informational:** This message indicates that a particular device is used in an array.  
**User Action:** This is an informational message telling the user what the RAID subsystem did. There is no user action required.

INVALCMD, command is invalid for array in the STARTUP state  
**Error:** A node joining the cluster did not completely bind pre-existing arrays.  
**User Action:** Enter this command after the BIND or UNBIND is complete.

INVALIDSTRUCT, Invalid on-disk structures found on device *unit-id*  
**Fatal:** Invalid on-disk structures were detected in RAID array or spare.  
**User Action:** Bind the array with the /REMOVAL\_ALLOWED qualifier so that the array goes into a reduced mode. The error is fatal because the /NOREMOVAL qualifier was specified. Alternately, reinitialize the RAID array and restore from backup if more than one member has this error.

INVARRAYID, invalid array-id specified - check spelling */array-id*  
**Fatal:** This message indicates that an invalid string was specified for the array-id.  
**User Action:** Reissue the command with a valid array id of the form DPAn:.

INVDSNAME, invalid shadowset device name *device-name*  
**Fatal:** This message indicates that the string specified for a shadow set virtual device is not valid.  
**User Action:** Reissue the command with a valid OpenVMS Volume Shadowing virtual device name of the form DSAn:.

INVINDEXPOS, Invalid index position specified: *number*

**Fatal:** The index position given to the command exceeds the number of devices that are present in the array.

**User Action:** Review the index numbers from the RAID SHOW command and reissue the command.

INVMEMBER, *unit-id* is not a valid RAID array member

**Informational:** This member is not a properly initialized RAID array member.

**User Action:** Specify appropriate RAID array member.

INVOPRAID0, The requested operation is invalid for a RAID 0 array

**Fatal:** Some operations do not make sense for stripeset because they are non-redundant. Some examples of these operations are removals and replacements, and using associated sparesets. This message results from attempting one of these invalid operations.

**User Action:** Do not attempt invalid operations on stripesets.

INVPRTSIZ, invalid partition size specified: *bogus partition size*, partition skipped

**Warning:** This is signalled when the size specified for a partition is outside the valid range. That partition will not be created, but the command will continue, creating the other partitions, if possible.

**User Action:** There is no user action other than to respecify the command with a partition size > 0, if they don't like what they got.

INVREMOVESTATE, state of array *array-id* prohibits member removal

**Fatal:** The RAID array is either already missing a member or the RAID array is reconstructing. You have attempted to remove a member from the RAID array which cannot have a member removed for one of three reasons:

- The RAID array is in the process of reconstructing.  
If the RAID array is reconstructing, you can only remove the reconstructing member.
- The RAID array is reduced.  
If the RAID array is reduced, it cannot have another member removed.
- The array is a RAID0 array

**User Action:** Allow the reconstructing array to complete or perform a replace operation if the RAID array is reduced.

INVREPDEV, an invalid replacement device was specified

**Fatal:** An invalid replacement device was specified on the command line during a REPLACE/DEVICE operation. The /DEVICE qualifier requires a valid device be supplied.

**User Action:** Check the spelling of the device. This is a primary message that may be preceded by additional messages.

INVARRAYID, invalid *array-id* specified - check spelling

**Fatal:** The user specified an invalid *array-id*.

**User Action:** Check spelling of *array-id*. The attempted RAID array name is returned as part of error message. Verify that the RAID array name is valid.

INVSPARE, spare *unit-id* is unusable

**Error:** The indicated device cannot be incorporated into the RAID array during a replace operation. This is a primary message that will be followed by additional messages.

**User Action:** Dependent upon specific messages accompanying this error.

INVSTATUS, invalid cli\$present status for *item = status*

**Informational:** An invalid status was returned by the OpenVMS routine CLISPRESNT. The status provides additional information.

**User Action:** The user action depends on the accompanying messages.

INVUNSNUM, invalid unsigned numeric value *value* for *parameter*

**Fatal:** The user specified a signed number where an unsigned number was expected.

**User Action:** Correct the command line and reenter the command.

INVVUINDEX, invalid virtual unit index specified by CLI to SERVER

**Fatal:** This message indicates an internal RAID subsystem error.

**User Action:** The RAID software has encountered an internal error. Call HP customer support.

INVVUNAME, invalid virtual device name

**Fatal:** A virtual device name specified on the BIND command line was invalid. The device must be in the form *DPA<sub>nnnn</sub>*, where *nnnn* is a number between 1 and 9999.

**User Action:** Check the spelling of the device name and reissue the command with the correct name.

IOTIMEOVER, qualifier /NOTIMEOUT overrides /IO\_TIMEOUT qualifier

**Warning:** When you specify the /NOTIMEOUT qualifier, the /IO\_TIMEOUT qualifier is also set. Both of these qualifiers affect performance and are dependent on each other.

**User Action:** None. If this is not the behavior you want, do not use the /NOTIMEOUT qualifier with the /IO\_TIMEOUT qualifier.

ISBOUND, unit *unit-id* is bound as a member of RAID array *array-id*

**Informational:** This message indicates that a particular unit has been bound into a specific array.

**User Action:** No user action is required. This is an informational message to let the user know what the software did.

IS\_INFO, Strip num Vert. offset num Height num Index num Array *array-id*  
Count num,

**Informational:** A reconstruct operation in a certain area on a RAID 5 arrays has completed. The additional message line reports the status of this operation.

**User Action:** Run "RAID ANALYZE/ARRAY/REPAIR *array-id*" to obtain more information.

LICENSENAME, License name: *name*

**Fatal:** This is a secondary information message to the NOLICENSE message to provide the name of the missing license.

**User Action:** No user action is required. This is an informational message that provides information for another message that tells them what is wrong.

LOCALSWMIS, Software version mismatch on local node  
reinstall your kit

**Warning:** Some of the images used in the RAID subsystem are not the same software version. This can be eliminated by reinstalling the software.

**User Action:** Reinstall the software.

MAXRABOUND, maximum number of RAID arrays exceeded

**Error:** The maximum configuration of the RAID arrays is exceeded.

**User Action:** Unbind a currently bound array to provide space in the configuration, then reissue the command.

MAXRNG, value of *actual value* exceeds maximum of *maximum value*

**Informational:** The parameter specified out of range bounds.

**User Action:** Reenter with the correct value.

MAXSSBOUND, maximum number of sparesets exceeded

**Error:** During a BIND/SPARESET operation, the number of sparesets already bound is equal to or greater than the supported number that is specified in the release notes.

**User Action:** Unbind one or more sparesets and reenter the BIND /SPARESET command.

MEMALLOC, memory allocation failure

**Fatal:** The RAID software was not able to allocate enough memory to perform the current command.

**User Action:** Check accompanying messages for more information, then verify the process and system memory quotas.

MEMBERMISSING, a RAID array member is missing whose probable volume label was *volume-label*

**Informational:** During a RAID BIND operation, a member of the RAID array was determined to be missing.

**User Action:** The missing member can be found by examining the volume labels of cluster devices. For a reduced RAID array, these messages are normal if the missing member is the reduced state's missing member.

---

**Note**

---

It is possible that members of different RAID arrays will have identical names because unbound RAID arrays are not required to have unique names.

---

MEMOFDIFRA, device *unit-id* and *unit-id* in *array-id* are members of different RAID arrays.

**Fatal:** You specified at least two units on the command line that are in different RAID arrays and also specified the /OVERRIDE qualifier. Specifying the /OVERRIDE qualifier makes it impossible to determine which of multiple valid RAID arrays is intended.

**User Action:** Because the membership of the intended array may have changed, check the units listed for correctness. Avoid using the /OVERRIDE qualifier when you are not sure of the actual membership of the RAID array.

MINDEVICE, smallest device is too small to be a RAID array member

**Fatal:** The device characteristic size is smaller than the RAID array characteristic size.

**User Action:** Reinitialize with a smaller characteristic size or use a larger device.

MINRNG, value of *actual value* is less than minimum of *minimum value*

**Informational:** This message is signaled with other messages to provide added information.

**User Action:** No user action is required. This message provides additional information to other messages.

MSGEXP, *message-id* msg expected, received msg code = *message-id*

**Informational:** The RAID software received an internal message which it was not expecting.

**User Action:** Call HP customer support with the information from the message.

NAMECONFLICT, new array name conflicts with currently bound array

**Fatal:** The RAID array name modification failed because the specified name is already in use. Array names must be unique among bound arrays within a cluster

**User Action:** Choose a different name or unbind the array with the conflicting name.

NOCHARSIZE, characteristic size must be specified to create empty spareset

**Error:** Creating an empty spareset requires specifying the characteristic size.

**User Action:** Reenter the command with the characteristic size.

NOINITFILE, can't initialize RAID files on unit *unit-id*

**Fatal:** Device errors on the specified device prevented initialization of the RAID array.

**User Action:** Supply another device or repair the device specified in the error message.

NOLICENSE, none of the following licenses are active for this software product

**Fatal:** There is no LMF license for the RAID product installed on your system. Additional messages provide the LMF license names.

**User Action:** Register and load the product license indicated in the accompanying message.

NONDISK, device *unit-id* is not a disk device

**Fatal:** The specified device is not a disk.

**User Action:** Enter a valid disk.

NOOPCOM, date-time Unable to write to OPCOM,

**Informational:** Failed to send a message to OPCOM.

**User Action:** Use the additional status message to resolve the problem.

NOOPERPRV, operation requires OPER privilege

**Fatal:** You do not have the required privileges to use this command.

**User Action:** Refer to the RAID Commands Chapter and see your system manager to enable these privileges.

NORAIDARRAYS, there are no RAID arrays bound

**Informational:** There are no RAID arrays currently bound.

**User Action:** None.

NORESTART, server cannot be restarted

**Fatal:** The server cannot be restarted. The server is being started for a second time. This is not allowed without rebooting. The reason is that the driver is not reloadable.

**User Action:** Reboot.

NOSERVER, failed to communicate with server process RAID\$SERVER

**Fatal:** The RAID\$SERVER process is not present or is not responding.

**User Action:** Reboot the node to reload the HP RAID software. Otherwise, your RAID arrays are still available for access, but you cannot execute any RAID commands. The diagnostic log may indicate why the server is not running.

NOSPARESETS, there are no sparesets bound

**Informational:** There are no sparesets currently bound. You cannot get information using the RAID SHOW/SPARESETS command without bound sparesets.

**User Action:** None.

NOSRVRSP, RAID Server did not respond within timeout period

**Fatal:** The RAID server did not respond to the command it was sent within the timeout period.

**User Action:** The command will be performed later and there is no user action required. If the server is hung or not present on the system, the server needs to be restarted.

NOSSREPLACE, REPLACE from spareset *set-id* failed

**Fatal:** This message is the first part of a two-part message. It indicates that a REPLACE command from a spareset failed. The second part of the message indicates why.

**User Action:** Correct the problem indicated by the additional status message and try again.

NOSYSNAMPRV, operation requires SYSNAM privilege

**Fatal:** You do not have the required privileges to use this command.

**User Action:** Refer to the RAID Commands Chapter and see your system manager to enable these privileges.

NOTASSOCIATED, RAID array *array-id* not associated with a spareset

**Informational:** The RAID array you have specified is not associated with a spareset. The /NOASSOCIATED\_SPARESET qualifier was unnecessary.

**User Action:** None.

NOTAVAIL, device *unit-id* is not available

**Fatal:** The device specified is already allocated or not available.

**User Action:** Make the device available or choose another device.

NOTMODIFIED, no modifications were made

**Informational:** While executing a BIND or MODIFY command to request modifications, none were made.

**User Action:** Examine the previous message and correct the problem. This is a primary message that may be followed by additional messages.

NOTSHADOWED, RAID array is not using shadowed devices

**Fatal:** An attempt was made to ADD a member to a shadow set on a RAID array that is not using volume shadowing.

**User Action:** UNBIND and then re-BIND the array with the /SHADOW qualifier.

NOTSPARE, device *unit-id* is not a spare

**Error:** The device specified is not a spare, could not be accessed, or had errors.

**User Action:** Resolve device conflict.

NOTUSED, unit *unit-id* is not used as a member of the RAID array *array-id*

**Informational:** This message indicates that a specified unit will not be used in the RAID array due to being inaccessible or invalid as a member of the specified array.

**User Action:** If the RAID BIND command cannot determine the true membership from information on valid specified units, then determine the true membership and reenter the command.

NOUSABLESPARE, no usable spares were found

**Fatal:** During binding or adding a spareset, or during a RAID INITIALIZE /SPARE operation, no usable spares were found. All specified units are unusable as spares for one reason or another.

**User Action:** Refer to the preceding messages to determine what caused the specified units to be rejected as usable spares. Correct the problem with the specified units or select other units.

NOVOLPROPRIV, this operation requires VOLPRO privilege

**Fatal:** You do not have the required privileges to use this command.

**User Action:** Refer to RAID Commands Chapter and see your system manager to enable these privileges.

OPENERR, error opening filespec,

**Error:** The specified file could not be opened.

**User Action:** Retry command. If it still fails try DCL command "DUMP filespec" to obtain more hints on why the fail cannot be opened.

OPENOUT, could not open file *file name*

**Fatal and Error:** The OpenVMS software could not open the file you specified when using the /OUTPUT qualifier.

**User Action:** Check user access to specified files.

PAGESMAXRNG, PAGESMAX parameter is out of range

**Fatal:** The SET command parameter /PAGESMAX is out of range

**User Action:** Additional messages determine the range that was exceeded.

PARAMTOOLONG, value of *parameter* is too long

**Fatal:** The value you have entered for a parameter exceeds the allowable number of characters.

**User Action:** Check the parameter restrictions and reenter the command.

PARTIALMOD, modification was made but may not be permanent

**Informational:** The MODIFY command did not permanently update the RAID array. This change is in effect for the current bind, but may be lost on subsequent BINDS. This may happen if some members can be updated but others cannot be.

**User Action:** Reenter the MODIFY command. If the same error occurs, prepare the RAID array for possible removal of the failing member. Automatic removal is likely when the virtual device is accessed.

PKTEXP, *packet-type* pkt expected, received pkt code = *packet-type*

**Informational:** This message provides additional information for the PROTOCOL message.

**User Action:** No user action is required. This is an informational message that provides information for other messages.

POOLMAXRNG, POOLMAX parameter is out of range

**Fatal:** The SET command parameter /POOLMAX is out of range.

**User Action:** Additional messages determine the range that was exceeded.

PRTCREATED, Array partition *n1* created: *n2* blocks

**Informational:** An array partition (numbered *n1*) was created on the RAID array. The size of this partition was *n2* blocks.

**User Action:** None. The message informs the user of the exact size of the array partitions which the RAID INITIALIZE command creates.

PROTOCOL, subsystem protocol error detected

**Error:** An error was detected in the handshake protocol between the RAID subsystem components.

**User Action:** Contact HP customer support with all available information.



PRTRNG, specified number of partitions is out of range

**Fatal:** More array partitions were created than the maximum configured limit.

**User Action:** Repeat the command specifying less partitions. A subsequent message will inform the user how many partitions they created and how many are allowed by the software.

PRTSUPRNG, number of partitions is outside of supported range

**Informational:** More array partitions were created than the maximum supported limit.

**User Action:** None. The message is to inform the user that they have exceeded the supported configuration limit of the product. It may still work correctly, but is not supported by HP.

QIOCANCELED, service-name service timed out and issued SYSSCANCEL, context=hex-num,

**Informational:** A system service call timed out and had been cancelled.

**User Action:** None.

QIOCANCELEDONE, timed\_qio\_and\_cancel service finish following cancel, context=hex-num,

**Informational:** A server I/O to the on-disk metadata timed out and had been cancelled.

**User Action:** None.

RAIDSUBSYS, failed to access RAID subsystem

**Error:** This message indicates that an unexpected failure occurred communicating with the RAID driver. Additional information may be presented.

**User Action:** Check the accompanying messages for more information.

RANOTBOUND, specified RAID array is not bound

**Fatal:** During a REMOVE operation, an unbound RAID array was specified. BIND attempted to search for members of the specified RAID array based on the unit-list provided in the command line. No valid RAID arrays were found, or none matching the name given were found during the search.

**User Action:** Because the RAID array membership may have changed without you being aware of it, check the units listed in the command line for correctness. Also check the error log. There may be additional messages preceding this one indicating a problem with one or more of the units specified.

RANOTFOUND, specified RAID array was not found

**Fatal:** The devices have not been initialized by the RAID INITIALIZE command. The array name you have entered does not match any currently bound RAID arrays. You may have entered a logical name rather than the *array-id*.

**User Action:** Respecify using the current *array-id*, not the permanent *array-id* or logical name. Or, if the devices have not been initialized by RAID, then issue RAID INITIALIZE, or if you don't know the *array-id*, use the /OVERRIDE to change the *array-id*.

RANOTREDUCED, state of RAID array *array-id* is not REDUCED

**Fatal:** A REPLACE operation can only be done on a RAID array which is in a reduced state (that is, the array is already missing a member).

**User Action:** None.

RAUNBOUND, Array *array-id* has been unbound from virtual unit *DPAnnn*:

**Informational:** The array specified has just been unbound.

**User Action:** None.

RECONCILE\_INFO, Error with RECONCILE FE,

**Informational:** An error occurred during reconciliation of on disk metadata.

**User Action:** Use the additional status message to find out more about the problem.

RECONNOTDIST, a member other than the RECONSTRUCTING member is missing

**Fatal:** The missing member is not the reconstructing member of a RAID array. It is acceptable for a member to be missing during a RAID BIND operation, but it must be the member that is reconstructing.

**User Action:** Ensure that all members of the reconstructing RAID array are available, with the possible exception of the reconstructing member.

REDUCED, RAID array was bound but is reduced

**Informational:** The RAID array was bound, but will operate in the reduced state.

**User Action:** Replace the missing member to restore redundancy.

REDUCEDNOTDIST, an additional member of a REDUCED RAID array is missing

**Fatal:** An additional member of an already reduced RAID array could not be found.

**User Action:** Ensure that all remaining members of the reduced RAID array are available.

REINIT, device *unit-id* will have to be reinitialized

**Informational:** The device indicated has been partially used for a replacement operation and is not usable as a spare without reinitialization.

**User Action:** Use the RAID INITIALIZE command to initialize the disks to form a new array. This will destroy all existing data on the array.

REPLACE, missing drive in RAID array *array-id* replaced by *unit-id*

**Informational:** The REPLACE operation was successful.

**User Action:** None.

REPLACEFAIL, Replacement with spare *unit-id* into array *array-id* failed

**Fatal:** This is a primary message that will be accompanied by additional messages.

**User Action:** Dependent upon specific messages accompanying this error message.

REPLFAIL, Replace failed for RAID array *array-id*,

**Informational:** A REPLACE operation failed.

**User Action:** Use the additional status message to find out more about the problem.

RSPWRFAIL, failed writing to response mailbox,

**Informational:** The RAID\$SERVER process could not write to the CLI communication mailbox.

**User Action:** If the user interrupts a command before it has completed this message can appear. No action necessary.

SAMENAME, a different RAID array is already bound with the same name

**Error:** The name of the RAID array specified in a BIND command is the same name as another currently bound RAID array.

**User Action:** Enter the RAID SHOW command to display the names of all currently bound RAID arrays. Reenter the BIND command with the /OVERRIDE qualifier to create a unique RAID array name, or unbind the conflicting array.

SHADNOTENABLED, shadowing is not enabled on the node,

**Error:** The RAID BIND command requested the RAID array to be bound with shadowset members, and shadowing is not enabled on the specified node.

**User Action:** Check that the node with the error has SHADOWING sysgen parameter is set up for OpenVMS Host-Based Volume Shadowing, and that the SHADOWING license is enabled and that there are enough licenses to mount the required number of shadowsets.

SHADOWADD, unit *unit-id* added to shadow set *shadow-set*

**Informational:** This message indicates that the RAID software has added the specified unit to the specified shadow set. This message is also displayed by the OpenVMS Volume Shadowing software via OPCOM.

**User Action:** None required.

SHADOWREMOVE, unit *unit-id* removed from shadow set *shadow-set*

**Informational:** This message indicates that the RAID software has added the specified unit to the specified shadow set. This message is also displayed by the OpenVMS Volume Shadowing software via OPCOM.

SHADRAID5, Binding a RAID5 array with shadow set members is not supported,

**Fatal:** Shadow sets are not supported as RAID 5 members.

**User Action:** Repeat the BIND command using non-shadowed devices.

**User Action:** None required.

SPARENOTFOUND, the specified spare was not found in the spareset

**Error:** During a REMOVE/SPARE operation, the member specified was not a member of the specified spareset.

**User Action:** The RAID SHOW/SPARESET command will display the membership of all or specified sparesets. Determine the correct membership and reenter the command if necessary.

SSBOUND, spareset *set-id* is already bound

**Error:** The spareset name is already in use.

**User Action:** Use a unique spareset name or use the ADD/SPARESET command.

SSFULL, spareset is full, could not add new member

**Error:** During a BIND/SPARE or ADD/SPARE operation, enough members were specified that it would cause the spareset to contain more than the maximum number of supported spareset members.

**User Action:** Remove some members with the RAID REMOVE/SPARESET command before adding more spares to that spareset.

SSNOTBOUND, spareset *set-id* is not bound

**Error:** A command was entered specifying a spareset that is not bound.

**User Action:** Check the spelling of the spareset and try again.

SUCCESS, operation was successful

**Success:** The operation succeeded.

**User Action:** None.

SWTYPTOOLONG, metadata software type string is too long */string/*,

**Error:** An inconsistency was detected in RAID's on-disk structure.

**User Action:** Re-INITIALIZE the array.

TIMEOUTRNG, timeout is out of range

**Fatal:** The */TIMEOUT* is out of range. This is a primary message that may be followed by additional messages.

**User Action:** Use a valid range.

TOOFEWMEMBERS, not enough valid members to bind the RAID array

**Fatal:** One or more of the following conditions exists:

- The device(s) could not be found or was inaccessible.
- The device(s) was not a valid member of the specified RAID array.
- The device(s) was removed.
- The device(s) failed one of the above tests when the */NOREMOVAL\_ALLOWED* qualifier was specified.
- Shadowing server is not running on all RAID\$SERVER nodes and a BIND/SHADOW was attempted.

**User Action:** Ensure all members of the RAID array are available. If a reduced RAID array is acceptable, you do not need to use the */NOREMOVAL\_ALLOWED* qualifier.

TOOMANYSHAUNITS, too many shadow units specified

**Fatal:** More OpenVMS Volume Shadowing virtual devices were specified in the *BIND/USE\_SHADOW\_DEVICES* qualifier than the RAID software supports.

**User Action:** Reissue the command with less OpenVMS Volume Shadowing virtual device names in the */USE\_SHADOW\_DEVICES* qualifier.

TRYAGAIN, further processing of your command is not possible try your command again later,

**Error:** The command collided with another command for the same RAID array.

**User Action:** Retry the command later.

UNBINDFAIL, UNBIND failed

**Fatal:** The requested unbind operation did not complete.

**User Action:** Check accompanying messages and reissue the command.

UNEXPSTCHG, Unexpected state change request: from-state to to-state,

**Informational:** The current from-state of the array precludes a change to to-state.

**User action:** None.

UNEXPVMS, received an unexpected status from OpenVMS

*\_line line-number, file filename*

*\_system service service name*

**Fatal:** This message indicates a failure from a OpenVMS routine. Additional messages will provide more information.

**User Action:** Other messages with this one will determine the user action.

UNITNOTINRA, Unit *unit-id* is not a member of RAID array *array-id*

**Fatal and Error:** The unit specified for removal must be a member of the RAID array.

**User Action:** Specify a member of the RAID array.

UNITSRNG, specified number of units is out of range

**Fatal:** You have specified too few or too many units. This is a primary message that may be followed by additional messages.

**User Action:** Reenter the list of units with the appropriate members.

UNITSSBOUND, unit *unit-id* was successfully bound into spareset *set-id*

**Informational:** A specific spare was successfully bound into a spareset. This message is generated for each spare that is successfully bound.

**User Action:** None.

UNITSSUPRNG, number of members is outside of supported range

**Informational:** The use of RAID arrays with this many members is not supported by HP. This is a primary message that may be followed by additional messages.

**User Action:** HP does not recommend having RAID arrays outside the supported range.

UNSUPDRIVE, disk drive/driver *unit-id* is not supported

**Informational:** The driver supporting the specified device is not one supported by the HP RAID software.

**User Action:** Use a supported device type.

UNSUPPORTED, unsupported operation or function

**Fatal:** The command and its qualifer(s) are not supported on this platform.

**User Action:** Verify which platform supports this function.

VIRDEVACCESS, Unable to access virtual device,

**Fatal:** RAID Software could not access device DPA0.

**User Action:** Restart RAID\$SERVER process. If problem persists reboot the system and startup the RAID Software.

VIRTDEVMOUNTED, RAID Array virtual device is mounted

**Fatal:** The user attempted to unbind an array with a virtual device that is still mounted. Arrays may not be unbound if a virtual device on that array is still mounted.

**User Action:** Dismount the virtual device before unbinding the array.

VMSDISMOU, device unit-id could not be dismounted,

**Informational or Error:** The specified device failed to dismount.

**User Action:** Use DCL command DISMOUNT to dismount the device.

VUINUSE, virtual device name *virtual-unit* already in use

**Fatal:** This message indicates that the RAID virtual device name specified in the RAID BIND command is already in use.

**User Action:** Reissue the command with a different RAID virtual device or unbind the array using the RAID virtual device name before issuing the command.

VUMISMATCH, incorrect number of virtual devices on the command line, must specify req-num virtual device, cmdline-num virtual device specified,

**Fatal:** The array has req-num partitions defined but only cmdline-num virtual devices have been specified on the command line.

**User Action:** Specify req-num number of virtual devices in the command line and retry the command.

VUMOUNTED, virtual device *virtual-device* is currently mounted,

**Fatal:** Unable to UNBIND the specified RAID array because DPA devices are still mounted. A list of all DPA virtual devices that are still mounted will be displayed to the user.

**User Action:** Dismount the DPA virtual device(s) specified, and then re-issue the RAID UNBIND command.

VURNG, specified number of virtual units is out of range,

**Fatal:** The number of virtual devices specified in the command line is beyond the allowed maximum.

**User Action:** Retry the command with fewer virtual devices. Refer to the Release Notes for the current maximum.

VUSUPRNG, number of virtual units is outside of supported range,

**Informational:** The number of virtual devices specified in the command line is beyond the supported maximum.

**User Action:** None. If problems arise with this array you have to re-initialize the array with fewer units. Refer to the Release Notes for the current supported maximum.

WRONGRSID, array-id cmdline-id does not match permanent array-id *perm-id*,

**Fatal:** The array id specified in the command line does not match the permanent array id found.

**User Action:** Use RAID ANALYZE/UNITS to find out the array id for your disk devices.





This appendix lists the queue inputs/outputs (QIOs) function codes supported by HP RAID Software for OpenVMS.

## C.1 Disk I/O Functions

Although most applications use high-level language read and write commands, the result of every record and file I/O request is ultimately one or more disk SYSSQIO system service calls, or simply QIOs. Each disk QIO specifies that a group of consecutive blocks be written to, read from, or erased from a disk. The SYSSQIO system service has arguments that, for reads and writes, specify the following:

- The size of the transfer
- The disk address of the start of the transfer
- The virtual address of the memory buffer

The following QIOs are supported by the HP RAID Software for OpenVMS. For information on how to use the QIO interfaces with a standard disk driver, see the *OpenVMS I/O User Reference Manual, Part I*.

IOS\_ACCESS  
IOS\_ACPCONTROL  
IOS\_AVAILABLE  
IOS\_CREATE  
IOS\_DEACCESS  
IOS\_DELETE  
IOS\_MODIFY  
IOS\_MOUNT  
IOS\_NOP  
IOS\_PACKACK  
IOS\_READLBLK  
IOS\_READPBLK  
IOS\_READVBLK  
IOS\_SENSECHAR  
IOS\_SENSEMODE  
IOS\_SETCHAR  
IOS\_SETMODE  
IOS\_WRITECHECK  
IOS\_WRITELBLK  
IOS\_WRITEPBLK  
IOS\_WRITEVBLK  
IOS\_UNLOAD



---

## DCL Command files to Assist with RAID Management

### D.1 Obtaining Information about RAID Arrays and Sparesets

Because OpenVMS itself does not provide sufficient information about RAID arrays using the \$GETDVI system service or the DCL lexical function F\$GETDVI, a command procedure is included with the RAID software to aid in obtaining information about RAID arrays.

Another command procedure is also included as an example of how this information may be utilized to track the status of RAID arrays and sparesets.

#### D.1.1 RAID\$CONFIG.COM

The command procedure SYS\$EXAMPLES:RAID\$CONFIG.COM may be used to obtain information about the existence, membership, and status of the RAID arrays and sparesets in existence on the system.

It works by parsing the output from RAID SHOW commands to pick up critical items of data (supplementing this with information from the DCL lexical function F\$GETDVI about shadow set membership for RAID 0+1 arrays) and then placing the information into either DCL global symbols or logical names, from which another command procedure or program may obtain the information and either display it or take appropriate action based on the information so gleaned.

By default, the results are returned in DCL global symbols. If the P1 parameter passed to RAID\$CONFIG is "LOGICAL\_NAMES", the results are instead returned using logical names. By default, logical names are defined in the process logical name table, but a P2 parameter of either SYSTEM or GROUP may be specified to place the resulting logical names in the system or group logical name tables.

Table D-1 shows the information returned for arrays.

**Table D-1 Information Returned for Arrays**

Global Symbol Name	Description
RAID\$CONFIG_ARRAY_COUNT	Number of RAID arrays
RAID\$CONFIG_ARRAY_n_ID	RAID array IDs
RAID\$CONFIG_ARRAY_n_RAID_LEVEL	RAID array RAID Level (0, 5, 0+1)

(continued on next page)

**Table D-1 (Cont.) Information Returned for Arrays**

Global Symbol Name	Description
RAID\$CONFIG_ARRAY_n_STATE	RAID array state (normal, reconstr., etc.)
RAID\$CONFIG_ARRAY_n_VIRTUAL_DEVICE_LIST	DPAn: device name(s) for partition(s)
RAID\$CONFIG_ARRAY_n_MEMBER_COUNT	RAID array size (# of members)
RAID\$CONFIG_ARRAY_n_MEMBER_m_DEVICE_NAME	Member device names
RAID\$CONFIG_ARRAY_n_MEMBER_m_STATE	Member state (normal, missing...)

In addition, Table D-2 shows the information returned for RAID 0+1 arrays.

**Table D-2 RAID 0+1 Array Information Returned**

Global Symbol Name	Description
RAID\$CONFIG_ARRAY_n_MEMBER_m_SHADOW_MEMBER_COUNT	Shadow set depth
RAID\$CONFIG_ARRAY_n_MEMBER_m_SHADOW_MEMBER_s_DEVICE_NAME	Device names
RAID\$CONFIG_ARRAY_n_MEMBER_m_SHADOW_MEMBER_s_STATE	State of each: "ShadowMember", "FullCopying", or "MergeCopying"

Table D-3 shows the information returned for sparesets.

**Table D-3 Information Returned for Sparesets**

Global Symbols Name	Description
RAID\$CONFIG_SPARESET_COUNT	Number of sparesets
RAID\$CONFIG_SPARESET_n_ID	Spareset IDs
RAID\$CONFIG_SPARESET_n_CHARACTERISTIC_SIZE	Spareset charact. size
RAID\$CONFIG_SPARESET_n_MEMBER_COUNT	Spareset # of members
RAID\$CONFIG_SPARESET_n_MEMBER_m_DEVICE_NAME	Member device names
RAID\$CONFIG_SPARESET_n_MEMBER_m_STATE	Member state, bound, etc.

When the output of RAID\$CONFIG is directed into logical names, it is possible to set up one process running RAID\$CONFIG and defining logical names which are accessible by other processes. So that processes using the information can tell when it has been updated, there is also a timestamp logical name defined if the results are being placed into logical names:

```
RAID$CONFIG_TIMESTAMP           Time of data-gathering
```

## D.1.2 RAID\$DISPLAY.COM

The procedure `SYS$EXAMPLES:RAID$DISPLAY.COM` is provided as an example of how one might use the RAID array and spareset information gathered by `RAID$CONFIG.COM` to display the status of RAID arrays. It draws pictures on a color VT terminal (or a DECterm display on a color workstation monitor) showing the state of RAID arrays and sparesets, using color to indicate conditions which should be of concern or which indicate danger of data loss.

In general, red indicates danger of data loss:

- Reduced RAID-5 array
- RAID 0+1 array, where a member has no redundancy to protect it

Yellow indicates a condition of concern, but which is expected to clear itself up in time:

- RAID-5 array under reconstruction
- RAID 0+1 array where a member has no redundancy, but where a shadow copy is already in progress to resolve that

Green means the array is in a stable, normal state.

Because of internal limits within DCL itself that constrain the size of the contents of DCL symbols, coupled with the length of escape sequences required to produce the color display, this example procedure is limited in the size of RAID arrays and sparesets which it can display:

- at most 5 RAID array members for:
  - RAID5 arrays
  - RAID0 arrays
  - sparesets
- at most 3 RAID array members for:
  - RAID 0+1 arrays

If a P1 parameter is included, only those RAID arrays and sparesets whose IDs are included in the P1 parameter list will be displayed.

If a P2 parameter of "CONTINUOUS" is specified, this will cause the procedure to automatically update and refresh the display periodically. To make such a continuous display operate more efficiently, by avoiding disk I/O operations:

1. Put the `RAID$CONFIG.COM` and `RAID$DISPLAY.COM` procedures onto a DECram RAMdisk
2. Also point the `SYSSCRATCH` logical name to the RAMdisk so that the temporary file used to contain output from the `RAID SHOW` command need not actually land on magnetic disk
3. 'INSTALL' the image called by DCL for the `SET MESSAGE` command so that the image need not be re-opened and the image header located again for each loop through the procedure

**Here are some example setup procedures to achieve this:**

```
SETUP.COM (done once, called from SYSTARTUP_V*.COM):
$! Set up our RAMdisk
$ @RAMDISK !Create a DECram disk
$ define/system rambo mda1:
$ create/dir rambo:[display]
$ copy sys$examples:raid$config.com,raid$display.com -
    rambo:[display]
$! Keep SET MESSAGE from causing a system disk access
$! every time we do it
$ install replace sys$message:SYSMGTMSG.EXE/open/head/share
$!
$ exit

RAMDISK.COM:
$ IF F$GETSYI("CPU") .LT. 128
$ THEN !VAX
$ mcr sysgen connect mda1/driver=mddriver/noadapter
$ ELSE !Alpha
$ mcr sysman io connect mda1/driver=sys$mddriver/noadapter
$ ENDIF
$ initialize/size=2000 mda1: rambo
$ mount/system mda1: rambo

LOGIN_SETUP.COM (done before DISPLAY.COM is called):
$ !Put scratch files into RAMdisk
$ define sys$scratch rambo:[display]
```

---

# Glossary

**array capacity**

The RAID array capacity is the number of logical blocks in the array that are available to the user.

**availability**

A measure of the level of assurance of continued access to data.

**bind**

When you bind a RAID array, the RAID array members collectively form a virtual device so that the separate members appear to the user as a single device.

**chunk**

A group of consecutive virtual device logical blocks placed on a single RAID array member. Chunk size may be specified by the user when the RAID array is created.

**chunk size**

The number of logical blocks in each chunk (see chunk).

**data reliability**

A measure of the existence of valid data, measured in mean time to data loss (MTDL). That is, the average time until loss of data.

**DCL**

See DIGITAL Command Language.

**DIGITAL Command Language**

A standard command interface used to enter the RAID and OpenVMS commands.

**HP Storage Architecture (CSA)**

Specifications from HP governing a way to design and interface with mass storage products. CSA defines the functions to be performed by host computers, controllers, and drives, and specifies how they interact to manage mass storage.

**DPA $nnnn$ : unit**

Each DPA $nnnn$ : unit represents a single RAID virtual unit and is used by a program or programmer to reference RAID array data.

**CSA**

See HP Storage Architecture.

**CSSI: HP Storage Systems Interconnect**

A data bus that uses the System Communication Architecture (SCA) protocols for direct host-to-storage communications.

**exclusive OR**

The Boolean operator that gives a true value if an odd number of its  $n$  operands are true. Abbreviated XOR.

**empty spareset**

A spareset with no spares

**Files-11 On-Disk Structure Level 2 (ODS-2)**

The disk file structure supported by OpenVMS. The Files-11 data structure prepares a volume to receive and store data in a way recognized by the OpenVMS operating system.

**gigabyte**

Specifically, 1,073,741,824 or  $2^{30}$  bytes. Abbreviated GB.

**I/O request rate**

The number of individual I/O requests that can be processed in a unit time, usually measured in I/O requests per second.

**HSC (Hierarchical Storage Controller)**

A self-contained, intelligent mass storage controller that communicates with multiple host processors over a CI bus.

**integrated storage element (ISE)**

A generation of storage devices in which the controller and the device are in the same box.

**JBOD**

Just a bunch of disks. A term referring to a collection of disks not using RAID technology.

**LBN**

See logical block number.

**License Management Facility**

Software in the OpenVMS operating system that helps a system administrator conform to licensing agreements encoded in Product Authorization Keys (PAKs).

**LMF**

See License Management Facility.

**load balancing**

Dividing the I/O requests among disks so that no one disk is over-utilized.

**logical block number**

A volume-relative address of a data block on a mass storage device. Usually contains 512 bytes of user data.



**member**

One of the disk devices that make up a RAID array.

**MSCP (Mass Storage Control Protocol)**

The software protocol used to communicate I/O commands between a host processor and CSA-compliant devices on the system.

**MTBF**

Mean Time Between Failures.

**MTDL**

Mean Time to Data Loss.

**MTTR**

Mean Time to Repair.

**normal mode**

When a RAID array is operating in normal mode, it means that all of its members are present, and the array has full redundancy.

**PAK**

See Product Authorization Key.

**parity chunks**

Parity chunks are a fundamental principle used on RAID levels 3,4, and 5. A parity chunk is generated from a bit-by-bit Exclusive OR of the contents of the corresponding chunks on all the other array members.

**partitioning**

In HP RAID Software for OpenVMS, partitioning is the act of dividing up a RAID array into one or more RAID virtual devices.

**physical device**

A peripheral storage device that is visible to the user programs through standard drivers.

**populated spareset**

A spareset with at least one spare.

**Product Authorization Key**

A license, issued to a customer, that contains codes that authorize the use of a software product. The HP RAID Software for OpenVMS will not operate without a valid PAK registered with LMF. PAKs may have expiration dates or other restrictions. See *HP OpenVMS License Management Utility Manual*.

**QIO functions**

Queue input/output. The OpenVMS system service that prepares an I/O request for processing by the device driver. A QIO is synonymous with an I/O request in OpenVMS software.

**RAID**

Redundant Array of Independent Disks.

**RAID 0**

Also known as striping, RAID 0 is the only RAID level that does not have redundancy. User data is distributed across a set of disks. See disk striping.

**RAID 0+1**

RAID 0+1 is a combination of mirroring and striping that provides high performance and high reliability.

**RAID 1**

Also known as shadowing or mirroring, RAID 1 provides redundancy by having the user data on the first disk replicated on two or more disks.

**RAID 2**

RAID 2 distributes data across a set of disks. Uses Hamming code to provide data integrity.

**RAID 3**

RAID 3 distributes data across a set of disks. Creates the data parity on-the-fly and places it on a separate disk to provide redundancy. Every read or write operation accesses every disk.

**RAID 4**

RAID 4 distributes data across a set of disks. Creates the data parity on-the-fly and places it on a single separate disk to provide redundancy. During reads and writes, only the data and parity disks are accessed, rather than every disk.

**RAID 5**

RAID 5 is the same as RAID 4, except parity is distributed across the entire set of disks.

**RAID 6**

RAID 6 provides redundancy by calculating redundant information using two separate functions and placing the information in two separate locations for each original piece of data. Thus, any two disks may fail without loss of user data.

**RAID array**

A set of physical devices organized according to RAID 5 technology.

**RAID virtual device**

An OpenVMS device that provides an interface to the RAID array. The virtual device logically represents a collection of disks to the user as a single disk.

**reconstructing mode**

When a RAID array is operating in reconstructing mode, it means that one of its members has failed, been replaced, and the data from that member is being re-created on the replacement device.

**reconstruction**

The process of incorporating a spare device as a member of a RAID array and regenerating data from the failed device onto the replacement device.

**reduced mode**

When a RAID array is operating in reduced mode, it means that one of its members is missing, and the array is operating without redundancy.

**redundancy**

With RAID 5, the process of maintaining the parity blocks and the data blocks provides redundant information, allowing the regeneration of data.

**regeneration**

The online process of re-creating data from a failed or failing drive as part of the original read. This regenerated data is re-created from working members and is identical to the data from the failed member.

**SCSI**

Small Computer System Interface.

**SDI**

HP's Standard Disk Interconnect.

**spare**

A disk initialized by the HP RAID software so that it may be included in a spareset or used as a replacement for a removed member of a RAID array.

**spareset**

A set of zero or more spares that can be substituted for individual members of a RAID array.

**striping**

A technique that improves performance by distributing I/O requests over multiple disk drives and concurrently processing multiple I/O requests. Also referred to as RAID 0.

**unbind**

The process of dissolving the association between a virtual device and a RAID array. The process of dissolving the association between spares and a spareset.

**unit**

A disk drive.

**VMScluster system**

A loosely coupled, highly integrated, distributed computing environment supported by the OpenVMS operating system. VMScluster system configurations support ethernet, CI, DSSI, and FDDI buses for mixed-interconnect.

**volume**

A disk-class medium.

**XOR**

See exclusive Or.



## A

---

Accessible status, 7-5

ADD/SPARESET

See RAID ADD/SPARESET

Array

capacity calculation, 2-6

size, 2-5

Asynchronous I/O requests

definition of, 8-2

## B

---

Backup

of arrays with shadow sets, 6-10

OpenVMS, 10-2

Benefits of RAID0 technology, 8-1

BIND

See RAID BIND

BIND/SPARESET

See RAID BIND/SPARESET

## C

---

Capacity

calculation, 2-6

of array, 2-5

Checklist, 3-1

Chunks

definition, 2-4

parity, 2-4

CLI

See Command line

Cluster

See VMScluster

Command line

errors, B-1

interface, 10-1

command procedure, 11-1

Commands

RAID ADD/SHADOW\_MEMBER, 11-4

RAID ADD/SPARESET, 11-6

RAID ANALYZE/ERROR\_LOG, 11-9

RAID BIND, 11-11

RAID BIND/SPARESET, 11-15

RAID CLONE, 11-17

Commands (cont'd)

RAID INITIALIZE, 11-19

RAID INITIALIZE/SPARE, 11-22

RAID MODIFY, 11-24

RAID REMOVE, 11-26

RAID REMOVE/SHADOW\_MEMBER, 11-27

RAID REMOVE/SPARE, 11-28

RAID REPLACE, 11-29

RAID SET, 11-30

RAID SHOW, 11-32

RAID SHOW/SPARESET, 11-42

RAID SHUTDOWN, 11-44

RAID UNBIND, 11-45

RAID UNBIND/SPARESET, 11-46

Control information, 2-7

## D

---

DCL

See Command line

Defragmenting, 4-15

Devices, using dissimilar, 9-5

Devices supported, 2-1

Disks, using dissimilar, 9-5

DPA units, 10-4

DSA device names

assignment of, 6-6

## E

---

Error

member, 7-6

member device read failure, 7-9, 7-10

member device write failure, 7-9, 7-10

Error handling

global errors, 7-10

in RAID0 arrays, 6-13

localized, 7-9

of members, 7-6

summary, 7-10

Error messages

See OpenVMS error messages

OPCOM, A-1

RAID command messages, B-1

Error recovery, 7-6

event procedure, 11-1  
event types, 11-2  
  DUMP, 11-2  
  EXLICENSE, 11-2  
  NEWNODECOMP, 11-2  
  RECONCOMP, 11-2  
  RECONINC, 11-2  
  REMOVE, 11-2  
  SHADOWREMOVE, 11-2  
  SPAREREMOV, 11-2  
  SPARESETEEMPTY, 11-2

---

## F

File system  
  contents of, 2-7  
  mounting, 4-5  
  placing on RAID array, 4-4  
  use of, 2-7

---

## H

Help utility  
  See OpenVMS help utility

---

## I

I/O requests  
  large or small size, 8-2  
  sequential versus random, 8-2  
  synchronous versus asynchronous, 8-2  
Improving performance, 8-10  
INITIALIZE  
  See RAID INITIALIZE  
INITIALIZE/SPARE  
  See RAID INITIALIZE/SPARE  
Inoperative status, 7-4

---

## J

Just a bunch of disks (JBOD), 1-3

---

## L

Large I/O requests  
  definition of, 8-2  
Logical block  
  with RAID 5, 2-4  
  with unlike disks, 9-5  
Logical name, 10-4  
logical names  
  RAID\$NOTIFY\_PROCEDURE, 11-3  
  RAID\$NOTIFY\_QUEUE, 11-3  
  RAID\$NOTIFY\_USERNAME, 11-3

---

## M

Member structure, 2-3  
Member volume file system  
  See File system  
Mirroring, 1-2  
Missing state, 7-4  
MODIFY  
  See RAID MODIFY  
MOUNT  
  See OpenVMS MOUNT  
Mounting state, 7-4  
  spares, 7-4

---

## N

NOIO\_TIMEOUT  
  description, 7-8  
Normal state, 7-2, 7-4  
  See RAID array states

---

## O

OpenVMS  
  backup, 10-2  
  error messages, 10-2  
  help utility, 10-2  
  MOUNT, 4-5  
OpenVMS Help utility, 10-2

---

## P

Parity  
  chunks, 2-4  
Partitioning  
  bind example, 5-7  
  binding, 5-3  
  by default, 5-2  
  creating, 5-2  
  definition, 5-1  
  examples, 5-3, 5-4  
  exceeding disk space, 5-6  
  using, 5-1  
  why use?, 5-1  
  with specified sizes, 5-2  
  with truncation, 5-5  
Planning checklist, 3-5, 3-6  
Populated spareset  
  definition of, 2-7

---

## Q

QIO Interface, C-1

## R

---

### RAID

- arrays, 2-2
  - attributes, 1-1
  - command messages, B-1
  - controller-based versus host-based, 1-4
  - HP supported level, 1-3
  - member structure, 2-3
  - shutdown, 10-2
  - spares, 2-7
  - sparesets, 2-7
  - technology overview, 1-1
  - trade-offs, 1-3
  - what is supported?, 2-1
- RAID\$DIAGNOSTICS.LOG, 10-3
- RAID\$SERVER\_MAIN.DMP, 10-3
- RAID\$SERVER\_MAIN.LOG, 10-3
- ### RAID0 array
- BACKUP with shadow sets, 6-10
  - benefits, 8-1
  - chunk size, 8-11
  - converting from nonshadowed to a shadowed array, 6-8
  - creating with shadow sets, 6-4
  - design applications, 8-5
  - designing for high I/O request rates, 8-6
  - device hierarchy, 6-4
  - disk MSCP-service, 8-12
  - disk size, 8-10
  - error handling, 6-13
  - for data transfer-intensive I/O workloads, 8-3
  - for high data transfer rates, 8-4
  - for I/O load balancing, 8-5
  - for request rate-intensive, 8-6
  - I/O workloads, 8-2
  - improve performance?, 8-2
  - initialize using shadow sets, 6-5
  - number of members, 8-10
  - OpenVMS Monitor, 8-6
  - performance, 6-13
  - performance analysis tools, 8-6
  - plus shadowing to improve performance, 8-10
  - queue depth, 8-12
  - RAID SHOW/FULL command, 8-7
  - rebinding without shadowing, 6-9
  - rebinding with shadowing, 6-8
  - related documentation, 6-4
  - shadowing in a VMScluster, 6-12
  - shadow sets, 6-1
  - user data mapping, 2-4
  - volume shadowing, 6-2
  - where RAID0 don't work well, 8-3
  - with shadow sets, 6-4
- ### RAID0 shadow set
- adding members, 6-7
  - removing members, 6-8
- ### RAID0 virtual device
- creating, 6-5
  - initializing, 6-6
- ### RAID5 array
- configuring sparesets, 9-18
  - cost, availability, and performance, 9-3
  - number of disks, 9-5
  - protecting against controller failure, 9-11
  - protecting against disk failure, 9-10
  - protecting against host adapter failure, 9-13
  - reconstruction, data reliability, and performance, 9-17
  - reliability versus availability, 9-6
  - user data mapping, 2-5
- RAID ADD/SHADOW\_MEMBER, 11-4
- RAID ADD/SPARESET, 4-14, 11-6
- RAID ANALYZE/ERROR\_ILOG, 11-9
- ### RAID array, 2-2
- access through virtual devices, 10-4
  - capacity, 2-5
  - changing characteristics, 4-10
  - command procedures, 4-16
  - creating, 4-1
    - example, 4-5
  - defining, 4-2
  - identifier for, 11-11
  - maintenance, 4-15
  - modifying, 4-11
  - name, 4-3
  - planning checklist, 3-1, 3-5, 3-6
  - recreating, 4-10
  - reduced, 4-11
  - removing a member, 4-11
  - replacing a member, 4-12
  - restrictions, 3-2
  - states, 7-1
  - undefining, 4-9
  - user-settable attributes, 3-2
  - using unlike devices, 9-5
  - virtual device
    - associating with, 4-3
- RAID array member states, 7-4
- RAID array states, 7-1
- Normal state, 7-1, 7-2
  - Reconstructing state, 7-1, 7-2
  - Reduced state, 7-1, 7-2
- RAID array status, 7-3
- Inoperative status, 7-4
  - Startup status, 7-3
- RAID BIND, 4-3, 11-11
- RAID BIND/SPARESET, 4-13, 11-15
- RAID CLONE, 11-17
- RAID CLONE command, 6-10
- RAID command messages
- CLI, B-1
- RAID INITIALIZE, 4-2, 11-19
- time required, 4-2

- RAID INITIALIZE/SPARE, 4-13, 11-22
- RAID levels, 1-3
  - RAID 0, 1-2
  - RAID 0+1, 1-2
  - RAID 1, 1-2
  - RAID 2, 1-2
  - RAID 3, 1-2
  - RAID 4, 1-2
  - RAID 5, 1-2
  - RAID 6, 1-2
  - supported by HP, 1-3
- RAID MODIFY, 4-11, 11-24
- RAID REMOVE, 4-12, 11-26
- RAID REMOVE/SHADOW\_MEMBER, 11-27
- RAID REMOVE/SPARE, 4-14, 11-28
- RAID REPLACE, 11-29
- RAID SET, 11-30
- RAID SET command, 10-2
- RAID shadow set states, 7-5
  - ShadowCopying state, 7-6
  - ShadowMerging state, 7-5
  - SteadyState, 7-5
  - Unknown state, 7-6
- RAID SHOW, 4-12, 11-32
- RAID SHOW/SPARESET, 4-15, 11-42
- RAID SHUTDOWN, 11-44
- RAID SHUTDOWN command, 10-2
- RAID UNBIND, 4-9, 11-45
- RAID UNBIND/SPARESET, 4-13, 11-46
- RAID virtual unit status, 7-5
  - Accessible, 7-5
  - Startup status, 7-5
- Random I/O requests
  - definition of, 8-2
- Reconstructing state, 7-2, 7-4
  - See RAID array states
- Reconstruction, 4-12
  - determinations, 7-11
- Reduced state, 7-2
  - See RAID array states
- Redundancy, 7-9, 7-10
- Redundant Array of Independent Disks
  - See RAID
- REMOVE
  - See RAID REMOVE
- REMOVE/SPARE
  - See RAID REMOVE/SPARE

## S

---

- Sequential I/O requests
  - definition of, 8-2
- Server
  - problems with, 10-3
- ShadowCopying state, 7-6

- Shadowing, 1-2
- ShadowMerging state, 7-5
- Shadow sets
  - as RAID array members, 2-1
  - with RAID0 arrays, 6-1
- SHOW
  - See RAID SHOW
- SHOW/SPARESET
  - See RAID SHOW/SPARESET
- Size of the array, 2-5
- Small I/O requests
  - definition of, 8-2
- Spares, 2-7
  - definition of, 2-7
  - incorporation into spareset, 2-8
  - mounting state, 7-4
- Spareset, 2-7, 4-12
  - adding to, 4-14
  - associating, 4-13
  - associating with a RAID array, 4-14
  - associating with RAID arrays, 2-8
  - binding, 4-13
  - definition of, 2-7
  - disassociating, 4-13
  - initializing, 4-13
  - limitations, 2-9
  - necessity of, 2-8
  - removing members from, 4-14
  - replacing members, 4-15
  - unbinding, 4-13
- Spiral data transfer rate
  - definition of, 8-4
- Startup status, 7-3, 7-5
- States
  - of RAID array members, 7-4
  - of RAID arrays, 7-1
- Status
  - of RAID arrays, 7-3
  - of RAID array virtual units, 7-5
- SteadyState, 7-5
- Storage administrator
  - concerns for, 9-1
- Stripeset
  - as RAID array members, 2-1
- Striping, 1-2
- Synchronous I/O requests
  - definition of, 8-2
- System shutdown, 10-2

## T

---

- Timeout mechanism, 7-7
  - high timeout value, 7-7
  - low timeout value, 7-7



## U

---

UNBIND

See RAID UNBIND

UNBIND/SPARESET

See RAID UNBIND/SPARESET

Unknown state, 7-6

User-settable attributes, 3-2

## V

---

Virtual device

accessing, 10-4

definition of, 2-3

VMScluster

creating a RAID array on, 4-7

environment, 2-1

VMScluster state transitions, 7-11

VMS MOUNT

See OpenVMS MOUNT

Volume shadowing

device hierarchy, 6-4

requirements, 6-3

why use?, 6-2

