Tru64 UNIX

Hardware Management

Part Number: AA-RSFCA-TE

September 2002

Product Version: Tru64 UNIX Version 5.1B or higher

This manual is intended for experienced UNIX system administrators who manage hardware components and storage devices. This manual describes the tasks you must perform to maintain system hardware that is controlled by the HP Tru64 UNIX operating system running on an AlphaServer system. You use UNIX commands, shell scripts, and the SysMan Menu or SysMan Station user interfaces to perform such administrative tasks as relocating components, naming storage, and configuring the hardware environment.

Hewlett-Packard Company
Palo Alto, California

Contents

3 Using hwmgr to Manage Hardware

4 Dynamic Device Recognition (DDR)

5 Using Device Commands and Utilities

6 Processor-Specific Information

7 Managing Specific Device Types

8 Administering Legacy Hardware

Index

Tables

About This Manual

This is the first edition of the *Hardware Management* manual. In previous releases, the information in this manual was located in the *System Administration* manual.

This manual describes the tasks you perform to administer the hardware components (such as PCI bus option cards) and storage devices (such as disks and tapes) controlled by the HP Tru64 UNIX operating system running on an AlphaServer system.

Audience

This manual is for system administrators with the following qualifications and abilities:

- Know UNIX operating system concepts and the privileged commands and utilities
- Are trained in the operational aspects of UNIX system administration
- Are familiar with all the procedures necessary to maintain a UNIX system for high availability.

This manual is not intended as a training document for UNIX administrators.

Note

This manual describes only the management of hardware components on an AlphaServer processor and devices that are local to the processor, such as single-spindle disks. For important information on managing storage configuration, including the configuration and maintenance of storage arrays, see your hardware documentation. You use software applications, such as the StorageWorks Command Console (SWCC), in addition to the utilities provided by the operating system.

Updating from Previous Releases

If you are updating your system from an older version of the operating system, you must review all the changes implemented in the intervening releases. You can find this information in the HTML files provided on the Tru64 UNIX Documentation CD-ROM, especially the *New and Changed Features* manual. In addition, the following online resources are available:

- Documentation and Best Practices for all releases at the following Web site:

  http://www.tru64unix.compaq.com/docs/

- Technical updates for any information not included in the documentation provided with your media at the following Web site:

  http://www.tru64unix.compaq.com/docs/pub_page/update_list.html

Organization

This manual consists of the following chapters:

Chapter 1	Provides an overview of the hardware management model for Tru64 UNIX.
Chapter 2	Describes how you use the SysMan Menu and SysMan Station to perform common hardware management tasks.
Chapter 3	Describes generic procedures for administering classes of hardware such as disks or tapes by using the hwmgr command.
Chapter 4	Describes how you use dynamic device recognition (DDR) to manually add a device to the system.
Chapter 5	Describes the hardware management tools and utilities that you use to manage specific types of devices, or perform specific tasks such as partitioning (formatting) a disk.
Chapter 6	Provides information on features of the operating system that are processor-specific. For example, some processor models support soft partitioning.
Chapter 7	Describes how to manage specific classes of device and individual devices such as storage arrays, host bus adapters, and graphics controllers.
Chapter 8	Describes procedures for administering legacy hardware devices that are obsolete but still supported by Tru64 UNIX.

Related Documentation

The following documents are part of the standard Tru64 UNIX operating system documentation that is provided on the documentation CD-ROM with your licensed software kit. These documents provide important related information:

- *Release Notes* – Provide important information such as restrictions on using certain operating system features.

- *Installation Guide* and *Installation Guide — Advanced Topics* – Describe how to install your operating system. These manuals describe important hardware management tasks, such as initial storage device configuration, naming, and component assignment.
- *Managing Online Addition and Removal* – Describes management and configuration techniques used for Online Addition and Removal (OLAR) of system components using the System Management applications: SysMan Station, SysMan Menu, and the `hwmgr` command. Topics include component indictment, automatic deallocation, and related service tools.
- *Network Administration: Services* and *Network Administration: Connections* – Describe how to set up, configure, and troubleshoot your network. They contain information on configuring and troubleshooting network components, such as network interface cards (NICs) that are part of your AlphaServer system.
- *System Administration*, *AdvFS Administration*, and *Logical Storage Manager* – Provide information on administering the Advanced file system (AdvFS) and UNIX file system (UFS).

Icons on Tru64 UNIX Printed Manuals

The printed version of the Tru64 UNIX documentation uses letter icons on the spines of the manuals to help specific audiences quickly find the manuals that meet their needs. (You can order the printed documentation from HP.) The following list describes this convention:

G	Manuals for general users
S	Manuals for system and network administrators
P	Manuals for programmers
R	Manuals for reference page users

Some manuals in the documentation help meet the needs of several audiences. For example, the information in some system manuals is also used by programmers. Keep this in mind when searching for information on specific topics.

The *Documentation Overview* provides information on all of the manuals in the Tru64 UNIX documentation set.

Reader's Comments

HP welcomes any comments and suggestions you have on this and other Tru64 UNIX manuals.

You can send your comments in the following ways:

- Fax: 603-884-0120 Attn: UBPG Publications, ZKO3-3/Y32
- Internet electronic mail: `readers_comment@zk3.dec.com`

  A Reader's Comment form is located on your system in the following location:

  `/usr/doc/readers_comment.txt`

Please include the following information along with your comments:

- The full title of the manual and the order number. (The order number appears on the title page of printed and PDF versions of a manual.)
- The section numbers and page numbers of the information on which you are commenting.
- The version of Tru64 UNIX that you are using.
- If known, the type of processor that is running the Tru64 UNIX software.

The Tru64 UNIX Publications group cannot respond to system problems or technical support inquiries. Please address technical questions to your local system vendor or to the appropriate HP technical support office. Information provided with the software media explains how to send problem reports to HP.

Conventions

This manual uses the following conventions:

MB1, MB2, MB3	MB*N* refers to the mouse button that you must press to select an item or initiate an action.
%	
$	A percent sign represents the C shell system prompt. A dollar sign represents the system prompt for the Bourne, Korn, and POSIX shells.
#	A number sign represents the superuser prompt.
file	Italic (slanted) type indicates variable values, placeholders, and function argument names.
[|]	
{ | }	In syntax definitions, brackets indicate items that are optional and braces indicate items that are required. Vertical bars separating items inside brackets or braces indicate that you choose one item from among those listed.

. . .	In syntax definitions, a horizontal ellipsis indicates that the preceding item can be repeated one or more times.
⋮	A vertical ellipsis indicates that a portion of an example that would normally be present is not shown.
`cat`(1)	A cross-reference to a reference page includes the appropriate section number in parentheses. For example, `cat`(1) indicates that you can find information on the `cat` command in Section 1 of the reference pages.
Ctrl/*x*	This symbol indicates that you hold down the first named key while pressing the key or mouse button that follows the slash. In examples, this key combination is enclosed in a box (for example, Ctrl/C).
Return	In an example, a key name enclosed in a box indicates that you press that key.

1

Administering Hardware

This chapter provides an overview of the hardware management model and describes the resources that are available to you. It discusses the following topics:

- A conceptual overview of hardware management and how it relates to the organization of information in this manual (Section 1.1)
- Documentation resources that apply to hardware management, including reference pages for commands and utilities, and pointers to utilities that are associated with hardware administration (Section 1.2)
- A description of the system files and databases that contain information about system devices (Section 1.3)
- How the operating system uses the unique World-Wide Identifier (WWID) to manage devices (Section 1.4)
- A description of device special files, which are the logical representation of some types of device (Section 1.5)

The `hwmgr` command also enables you to perform service tasks such as hot-swapping CPUs. For information on this feature, see `hwmgr_ops`(8) and the *Managing Online Addition and Removal* manual.

1.1 Understanding Hardware

A hardware component is any discrete part of the system such as a CPU, a networking card, or a hard disk. The system is organized in a hierarchy with the central processing unit (CPU) at the top and peripheral components such as disks and tapes, at the bottom. This is sometimes also referred to as the system topology. The following components are typical of the device hierarchy of most computer systems, although it is not a definitive list:

- The CPU, which might be a single processor system, a multiprocessor system, or a set of processors joined into a cluster. The system is sometimes referred to as a host in the context of hardware management and has a designated host name and perhaps also a host address if the system is on a network. You often specify commands using the host name. The CPUs are the top of the system hardware hierarchy, and all other system components are organized under the CPUs.

Typical administrative tasks associated with the CPU are many, such as bringing CPUs online, starting and stopping them, or sharing CPU resources. See the *System Administration* manual and the *Managing Online Addition and Removal* manual for additional information.

- Buses – A system might have a number of main internal communication buses, which transfer data between components of the system. Adapters and controllers are physically plugged into buses and have both physical and logical addresses.

  Buses might have special software associated with the physical bus, but that software is usually managed within the context of the UNIX operating system. For example, when adding an option card such as a sound or network card to a PCI bus, you have to shut down the system, add the hardware, and reboot. Such components are usually automatically recognized and added to the system configuration on reboot, but you might need to run a firmware utility to install a driver for the device. Always consult your system documentation and the documentation that comes with the card for information on adding such components.

- Controllers and adapters – A system might have a number of controllers such as Small Computer System Interface (SCSI) controllers, which control one or more storage devices. There might be other controllers, such as the floppy disk interface (`fdi`) that support one kind of disk and usually have only one physical disk attached to the controller. A network adapter might be connected to a bus, but does not have any other components below it in the hierarchy other than the network cabling.

  Adapters occupy a physical slot on a bus, which gives them both a logical address and a physical location to administer. They might also provide slots for other components, which also have physical and logical addresses. (Controllers and adpaters are often referred to as HBAs, or host bus adapters.)

- Storage devices, such as SCSI disks or CD-ROM readers, are among the entities at the lowest level in the system hierarchy. They are typically attached to a controller or adapter, and often have both a physical location and a logical address to administer.

  Storage (and other) devices might be shared by components or members of a cluster. This means that a component might have different names and identifiers associated with it depending how you access the component. Understanding how to identify a component, and how that component appears to the rest of the hierarchy, is an important aspect of hardware management. You often need to know both logical and physical locations of components.

  When referring to SCSI devices in this manual, the SCSI disk is most frequently referenced as an example. It is often the target of hardware

management tasks and might appear to the system as a single device, or as a group or array. For example:

Single SCSI Disk or RZ devices

The operating system supports storage devices that conform to the SCSI interface technology. Not all SCSI devices closely conform to this standard and the system might not be automatically detect and add such devices. You might need to use `ddr_config` as described in Chapter 4 to add such devices

SCSI RAID or HSZ and HSV devices

The redundant array of inexpensive (or independent) disks (RAID) technology. These are storage boxes that contain several connected SCSI disks, appearing to the system as a single device. They might support features such as hot-swapping, failover, and redundancy, and be connected to the system by Fiber Channel controllers. Such storage arrays can be shared between many systems in a storage area network. An example is the HSV110-Enterprise Virtual Array HSV Controller

You use applications such as the StorageWorks Console (SWCC) to manage storage arrays and storage area networks. In such configurations, you can accomplish only a small proportion of your storage management tasks using features of the operating system, such as the `hwmgr` command. Consult your StorageWorks documentation for complete information on how you configure and manage storage arrays.

See RAID(7), SCSI(7), and rz(7) for more information on device characteristics. See tz(7) for more information on tape devices. See the *Technical Overview* manual and the Tru64 UNIX *Quickspecs* for the current supported standards for RAID and SCSI.

Hardware management involves understanding how all the components relate to each other, how they are logically and physically located in the system topology, and how the system software recognizes and communicates with components. To better understand the component hierarchy of a system, refer to the *System Administration* manual for an introduction to the SysMan Station. This is a graphical user interface that displays topological views of the system component hierarchy and allows you to manipulate such views.

The majority of hardware management tasks are automated. When you add a supported SCSI disk to a system and reboot the system, the disk is automatically detected and configured into the system. The operating system dynamically loads required drivers and creates the device special

files. You need only to partition the disk and create file systems on the partitions (described in the *System Administration* manual) before you use it to store data. However, you must periodically perform some hardware management tasks manually, such as when a disk crashes and you need to bring a replacement disk online at the same logical location. You might also need to manually add components to a running system or redirect I/O from one disk to another disk.

Many other hardware management tasks are part of regular system operations and maintenance, such as repartitioning a disk or adding an adapter to a bus. Often, such tasks are fully described in the hardware documentation that accompanies the component itself, but you often need to perform tasks such as checking the system for the optimum (or preferred) physical and logical locations for the new component.

Another important aspect of hardware management is preventative maintenance and monitoring. Use the following operating system features to maintain a healthy system environment:

- The Event Manager (EVM) – An event logging system that filters system events and then notifies you of selected events. It includes sophisticated features for warning you of problems by electronic mail or a pager. Refer to the *System Administration* manual for information on configuring EVM.
- The SysMan Station – A graphical user interface that enables you to view and monitor the entire system (or cluster) hardware and launch applications to perform administrative tasks on components. You can also launch these applications from the SysMan Menu, and some example applications are described later in this manual. See Section 2.1 for information on using the SysMan tasks, refer to the *System Administration* manual.
- The system census tool, `sys_check` – A command-line utility that provides you with data on your system's current configuration as an HTML document that you can read with a Web browser. You can use the data as a system baseline, perform tuning tasks, and check all log files. The Storage configuration section provides information on storage devices and file systems. Refer to the system customization information in the *System Administration* manual and to `sys_check`(8) for information on running this utility, and on configuring it to run regularly.
- HP Insight Management Agents for Tru64 UNIX – An enterprise-wide, Web-based management tool that enables you to view system and component status anywhere in your local area network. It includes launch points for the SysMan Station, the SysMan Menu, and the system census utility, `sys_check`. See `insight_manager`(5) for more information.

The organization of this manual reflects the hardware components and devices that you manage as follows:

- Generic hardware management tools – These tools enable you to perform operations on all components of a type, classes of component such as SCSI tapes, or individual components. The tools might in some cases operate on all systems in a cluster. An example of such a tool is the SysMan Station, which provides you with a graphical display of the entire component hierarchy for all members of a cluster.
- Software management – This involves the administration of the software that is associated with hardware components on the system, principally managing the device special files. These are the files associated with a hardware component that enable any application to access its device driver or pseudodriver.
- Targeted hardware management tools – These tools enable you to perform operations that are targeted to a specific component and perform a specific task. An example is the disk configuration command line interface, `disklabel`, and the analogous graphical user interface, Disk Configuration (`diskconfig`), which enable you to partition a disk by using the standard layouts or your own custom layouts.
- Device or system-specific tasks – These operations are targeted to a specific device or processor platform. An example is the configuration of logical partitions, which are supported only on certain classes of processor.

Another way to think of this is that with a generic tool you can perform a task on many components, while with a targeted tool you can perform a task on only a single component. Unless stated otherwise, most operations are specific to a single system or to a cluster. See the TruCluster Server documentation for additional information on managing cluster hardware.

1.2 Reference Information

The following sections contain reference information related to documentation, system files, related software tools. Some tools described here are obsolete and scheduled for removal in a future release. Consult the *Release Notes* for a list of operating system features that are scheduled for retirement and migrate to its replacement as soon as possible. Check your site-specific shell scripts for any calls that might invoke an obsolete command.

1.2.1 Documentation

The following documentation contains information about hardware management:

- Manuals (available online or hardcopy):
  - *Managing Online Addition and Removal* – Describes management and configuration techniques used for Online Addition and Removal (OLAR) of system components using the System Management applications: SysMan Station, SysMan Menu, and the `hwmgr` command. Topics including component indictment, automatic deallocation, and related service tools.
  - *Network Administration: Connections* and *Network Administration: Services* – Provide information on configuring or connecting network components.
  - Device Driver Documentation Kit – Contains related documents such as *Writing PCI Bus Device Drivers* and *Writing Device Drivers*.
  - *Logical Storage Manager* – Contains information on LSM concepts and commands. The Logical Storage Manager (LSM) consists of physical disk devices, logical entities, and the mappings that connect them.
- Reference pages:
  - `hwmgr`(8) – Summary information on the syntax and usage of the hardware manager command, `/sbin/hwmgr`.
  - `hwmgr_ops`(8) – System operation options for the `/sbin/hwmgr` command. Use these options to perform procedures such as CPU hot swap.
  - `hwmgr_show`(8) – Hardware information options for the `/sbin/hwmgr` command. Use these options to display information from the hardware databases.
  - `hwmgr_get`(8) – Component attribute information options for the `/sbin/hwmgr` command. Use these options to obtain and configure component attributes.
  - `hwmgr_view`(8) – Status information options for the `/sbin/hwmgr` command. Use these options to view component and system status.
  - `dsfmgr`(8) – Information on the command syntax for the device special file management command. Use this command to create device special files in the `/dev` directory. Refer also to Section 1.5.
  - `mknod`(8), `MAKEDEV`(8), `scu`(8), `ddr_config`(8), and `devswmgr`(8) – Miscellaneous commands and utilities that you might use while administering devices.

The command line and graphical user interfaces also provide extensive online help.

1.2.2 Web Resources

Many procedures described in this manual concern the administration of system hardware and peripherals such as storage devices. Consult the owner's manual for any hardware device, particularly if you need information on using hardware-specific application software.

The following Web sites provide information and resources such as driver updates:

- You can find software and drivers, including Alpha firmware downloads at the following Web site:

  http://www.compaq.com/support/files

- You can find general resources on AlphaServers at the following Web site:

  http://www.compaq.com/alphaserver/index.html

- The following Web site provides storage support, including firmware and drivers:

  http://www.compaq.com/storage/diskarrays-support.html
  You will find information about the following types of storage device:

  – Storage arrays (Fibre Channel and SCSI disk)

  – Storage interconnects (adapters, hubs and switches)

  – Storage adapters (host bus adapters, such as the KZPSA)

  You can find information about tape storage (such as the SDLT160/320) at the following URL:

  http://wwss1pro.compaq.com/support/TSSD2/default.asp

1.2.3 Software and Applications

Depending on your local system configuration, you might use one or more applications to manage system components. These applications often enable you to manage other devices connected to the system and its local components, such as Fibre Channel switches. The availability of a particular application depends on which storage components and which versions of Tru64 UNIX it supports. Some programs run only on a networked PC system and use a Web connection to access storage. The programs might also require a separate purchase under license, rather than bundled software provided with the hardware.

Often, a complex administrative procedure requires that you use one or more of these applications together with Tru64 UNIX commands and utilities such as the `hwmgr`. An example of such a complex procedure is configuring a redundant Fibre Channel storage array for multipath failover.

You can manage system components by using the following applications:

The system console

The system console is a command interface that is first visible when you power-on your AlphaServer. The system displays the console prompt (>>>) after the initial power-on tests complete successfully. This interface is the System Reference Manual (SRM) console, a term that is used synonymously with the phrase system firmware. (The firmware includes other components that are not part of the SRM.)

The console enables you to display information about system components. For example, the following command displays information about the hardware components in the system, such as the bus, target, and logical unit number (lun) address of a disk drive attached to a local SCSI bus:

```
>>>show device
```

Your AlphaServer owner's manual documents the SRM console commands. Minor variations in SRM command options exist between different AlphaServer systems. If you do not have the system's hardware manual, you can download a printable PDF version from the following Alpha Systems Technology Web site:

http://www.compaq.com/alphaserver/technology/index.html

Refer to the Tru64 UNIX *Release Notes* for important information on console-specific restrictions for your processor.

Array Controller Software and Hierarchical Storage Operating Firmware

The following software runs on certain models of storage array controllers, enabling you to perform administrative tasks:

- Array Controller Software (ACS)

  The ACS processes I/O requests from the host, performing the device-level operations required to satisfy requests. It provides you with utilities such as cloning and formatting. ACS operating firmware is stored in a PCMCIA program card that is shipped with the controller.

- Hierarchical Storage Operating Firmware (HSOF)

  The HSOF consists of functional code, diagnostics, utilities, and exercisers for certain HS-series array controllers. Its command-line interface enables you to access controller operations such as configuration and cloning. The HSOF is also stored in a PCMCIA program card that is shipped with the controller.

New PCMCIA program cards are provided for each controller model whenever there is an update to the ACS or HSOF operating firmware. You can purchase update cards separately for each release, or obtain

them automatically as part of an update service contract. Go to the following Web site for firmware update information:

http://www.compaq.com/products/storageworks/softwaredrivers/acs/index.html

Go to the following Web site to access ACS and HSOF online documentation:

http://www.compaq.com/products/storageworks/array-and-scsi-controllers/HSxuserdocs.html

StorageWorks Command Console (SWCC)

StorageWorks Command Console (SWCC) is a graphical storage configuration and monitoring software tool for arrays such as the EMA12000 and EMA16000 RAID Arrays. It reduces the task of storage management to simple point-and-click and enables you to configure and monitor storage graphically from a single management console. SWCC 2.3 provides an agent for Tru64 UNIX Version 4.0F and later releases. For information on the SWCC, go to the following Web site:

http://www.compaq.com/products/storageworks/swcc/index.html

Storage Area Networks (SAN)

HP supports SAN management through a growing number of software applications. Table 1–1 lists some of the applications available at the time of publication. Some applications run on a networked SAN appliance and might have dependencies on other software components. Other SAN tools provide only agents for specific versions of the operating system. This means that you manage your hardware from an interface running on an appliance (or a PC) that passes commands to an agent program running on Tru64 UNIX.

Table 1–1: SAN Software

Title	Description
SANworks Command Scripter	Provides you with command control of HSJ80, HSG60, HSG80, HSZ70 and HSZ80 Array Controllers. You can create, edit, and run script files that contain Command Line Interpreter (CLI) commands. This product provides both local and LAN connections and a browser-based interface for remote connections. This application works with the SWCC.

Table 1–1: SAN Software (cont.)

Title	Description
SANworks Data Replication Manager (DRM)	Provides a disaster-tolerance by providing hardware redundancy and data replication across multiple sites separated by some distance by replicating data at the target sites. The DRM sites are connected over some distance via fiber optic cable or asynchronous transfer mode (ATM). Data Replication Manager uses Fibre Channel gigabit switches to send the data between the sites.
SANworks Element Manager for StorageWorks HSG	Provides a Web-based graphical configuration and monitoring tool to centralize storage management on HSG controllers.
SANworks Enterprise Volume Manager	This Web-based application software enables you to manage controller-based clone and snapshot operations.
SANworks Network View	Provides a browser-based application to automatically map SAN topology, monitor its availability, and display a map of the storage environment.
SANworks Open SAN Manager	Provides centralized, appliance-based monitoring and management interface for the Open SAN. It enables you to organize, visualize, configure and monitor storage from a single navigation point on the SAN. It also provides a launch point for other SANworks applications and links to directly manage storage components on the SAN.

For SAN software and driver information, go to the following Web site:

`http://www.compaq.com/storage/sanworks-support.html`

Media Robot Utility (MRU)

An application that enables you to control robotic loaders such as the DLT and 4mm (DAT) or TKZ-series loaders.

1.2.4 Related Commands and Utilities

The following commands are also available to you for use in managing devices:

- The system exerciser utilities enable you to test devices for correct operation. See `diskx`(8), `tapex`(8), `cmx`(8), `fsx`(8), and `memx`(8).
- The `scu` command enables you to maintain and diagnose problems with SCSI peripherals and the CAM I/O subsystem. See `scu`(8) and the online help for the command.
- Use the `sysconfig` command to query or modify the kernel subsystem configuration. You use this command to add subsystems to your running kernel, reconfigure subsystems already in the kernel, ask for information about (query) subsystems in the kernel, and unconfigure and remove

subsystems from the kernel. You can use the `sysconfig` command to set some component attribute values. For information on using the `sysconfig` command, refer to the *System Administration* manual, which also documents the Kernel Tuner (`dxkerneltuner`). The Kernel Tuner is a graphical user interface that you can also use to modify attribute values.

- CDE Application Manager – SysMan Applications pop-up and System_Admin folders contain several hardware management tools, for example:
  - Configuration – Graphical user interfaces that you use to configure hardware such as ATM, disk devices, network devices, PPP (modem) devices, and LAT devices.
  - DailyAdmin – A graphical user interface for power management, which you use to set power attributes for certain devices.
  - SysMan Checklist, SysMan Menu, and SysMan Station – Interfaces that enable you to configure, monitor, and maintain system devices. You can invoke the SysMan Menu and SysMan Station from a variety of platforms, such as a personal computer or an X11–based environment. This enables you to perform remote monitoring and management of devices. See the *System Administration* manual for more information.

1.3 Identifying Hardware Management System Files

The following system files contain static or dynamic information that the system uses to configure the component into the kernel. Do not edit these files manually even if they are ASCII text files. Some files are context-dependent symbolic links (CDSLs), as described in the *System Administration* manual. If the links are accidentally broken, clustered systems cannot access the files until you verify and recreate the links. See `cdslinvchk`(8) and `mkcdsl`(8) for information on recreating CDSLs.

Caution

Although some hardware databases are text format, do not edit the databases. Any errors introduced into the databases might prevent your system from accessing devices, cause data corruption, or prevent your system from booting. Use only the appropriate commands and utilities to manage devices.

- `/dev` – A directory that contains device special files. Refer to Section 1.5 for more information.

- /etc/ddr_dbase – The device dynamic recognition (DDR) device information database. The content of this file is compiled into the binary file/etc/ddr.db, which the system uses to obtain device information.
- /etc/dec_devsw_db – A binary database owned by the kernel dev switch code. This database keeps track of the driver major numbers and driver switch entries.
- /etc/disktab – A file that specifies the disk geometry and partition layout tables. This file is useful for identifying disk device names and certain disk device attributes.
- /etc/dvrdevtab – A file that specifies the database name and the mapping of driver names to special file handlers.
- /etc/gen_databases – A text file that contains the information required to convert a database name to a database file location and a database handler.
- /etc/dec_hw_db – A binary database that contains hardware persistence information. Generally, this refers to hardware such as buses or controllers.
- /etc/dec_hwc_ldb – A binary database that contains information on hardware components that are local to a cluster member.
- /etc/dec_hwc_cdb – A binary database that contains information on hardware components that are shared by all members of a cluster. Hardware components with unique cluster names or mapped to dev_t are stored in this database.
- /etc/dec_scsi_db – A binary database owned by SCSI/CAM. It stores the worldwide identifier (WWID) of SCSI devices and enables CAM to track all SCSI devices that are known to the system.
- /etc/dec_unid_db – A binary database that stores the highest hardware identifier (HWID) assigned to a hardware component. The operating system uses this database to generate the next HWID that the system automatically assigns to a newly installed hardware component. The system never reuses an HWID. For example, assume you add a disk to a system and it is assigned an HWID of 124. Even if you remove that disk permanently from the system, the HWID 124 is never reassigned to its replacement disk or to any other device. The only way that you can reset the HWID numbering sequence is to perform a fresh installation of the operating system.

1.4 WWIDs and Shared Devices

SCSI device naming is based on the logical identifier (ID) of a device. This means that the device special filename has no correlation to the physical

location of a SCSI device. UNIX uses information from the device to create an identifier called a worldwide identifier, which is usually written as WWID.

Ideally, the WWID for a device is unique, enabling the identification of every SCSI device attached to the system. However, some legacy disks (and even some new disks available from third-party vendors) do not provide the information required to create a unique WWID for a specific device. For such devices, the operating system attempts to generate a WWID, and in the extreme case uses the device nexus (its SCSI bus/target/LUN) to create a WWID for the device.

Consequently, do not use devices that do not have a unique WWID on a shared bus. If a device that does not have a unique WWID is put on a shared bus, a different device special file is created for each different path to the device. This can lead to data corruption if the operating system uses two different device special files to access the same device at the same time. To determine if a device has a cluster-unique WWID, use the following command:

```
# /sbin/hwmgr show components
```

If a device has the `C` flag set in the `FLAGS` field, then it has a cluster-unique WWID and you can place it on a shared bus. Such devices are referred to as cluster-shareable because you can put them on a shared bus within a cluster.

Note

Exceptions to this rule are HSZ devices. Although an HSZ device might be marked as cluster shareable, some firmware revisions on the HSZ preclude multi-initiators from probing the device at the same time. See the owner's manual for the HSZ device and the Tru64 UNIX *Release Notes* for any current restrictions.

The following example displays all the hardware components that have cluster-unique WWIDs:

```
# /sbin/hwmgr show component -cshared
HWID: HOSTNAME FLAGS SERVICE COMPONENT NAME
---------------------------------------------
35:   pmoba    rcd-- iomap   SCSI-WWID:0410004c:"DEC  RZ28     ..."
36:   pmoba    -cd-- iomap   SCSI-WWID:04100024:"DEC  RZ25F    ..."
42:   pmoba    rcd-- iomap   SCSI-WWID:0410004c:"DEC  RZ26L    ..."
43:   pmoba    rcds- iomap   SCSI-WWID:0410003a:"DEC  RZ26L    ..."
48:   pmoba    rcd-- iomap   SCSI-WWID:0c000008:0000-00ff-fe00-0000
49:   pmoba    rcd-- iomap   SCSI-WWID:04100020:"DEC  RZ29B    ..."
50:   pmoba    rcd-- iomap   SCSI-WWID:04100026:"DEC  RZ26N    ..."
```

You might have a requirement to make a device available on a shared bus even though it does not have a unique WWID. Using such devices on a shared bus is not recommended, but there is a method that enables you to

create such as configuration. See Chapter 3 for a description of how you use the `hwmgr edit scsi` command option to create a unique WWID.

1.5 Device Naming and Device Special Files

Devices are made available to the rest of the system through device special files located in the `/dev` directory. A device special file enables an application (such as a database application) to access a device through its device driver, which is a kernel module that controls one or more hardware components of a particular type. For example, network controllers, graphics controllers, and disks (including CD-ROM devices).

The system uses device special files to access pseudodevice drivers that do not control a hardware component, for example, a pseudoterminal (`pty`) terminal driver, which simulates a terminal device. The `pty` terminal driver is a character driver typically employed by remote logins; it is described in Chapter 4. See the device driver documentation for detailed information on device drivers at the following URL:

`http://www.tru64unix.com/docs/`

Normally, device special file management is performed automatically by the system. For example, when you install a new version of the UNIX operating system, there is a point at which the system probes all buses and controllers and all the system devices are found. The system then builds databases that describe the devices and creates device special files that make devices available to users. The most common way that you use a device special file is to specify it as the location of a UFS file system in the system `/etc/fstab` file.

You need to perform manual operations on device special files only when there are problems with the system or when you need to support a device that the system cannot handle automatically. The following sections describe the way that devices and device special files are named and organized in Version 5.0 or higher.

The following considerations apply:

- The name of a device special file for a SCSI device has the format `/dev/disk/disk13a` for SCSI disks and `/dev/ntape/tape0_d0` for SCSI tapes. The name of a SCSI device special file in the format `/dev/rz10b` is a legacy device special file. The following sections differentiate between current and legacy device special files. You might also see these referred to as old (legacy) and new (current) device names in some scripts and commands. First time users of the operating system need not be concerned with legacy device special file names except where there is a need to use third-party drivers and devices that do not support

the current naming model. (The structure of a device special file is described in detail later in this section.)

- There is currently one device special file naming model for SCSI disk and tape devices and a different model for all other devices. The naming system for SCSI disk and tape devices will be extended to the other devices in future releases. This ensures that there is continued support for legacy devices and device names on nonclustered systems. Applications and commands either support all device names or display an error message informing you of the supported device name formats.

Legacy device names and device special files will be maintained for some time and their retirement schedule will be announced in a future release.

1.5.1 Related Documentation and Commands

The following documents contain information about device names:

- Manuals:
  - Information about context-dependent symbolic links (CDSLs). Some directories that contain device special files are CDSLs and you should be familiar with this concept before you proceed. See the *System Administration* manual for more information.
- Reference pages and commands:
  - See `dsfmgr`(8) for information on managing device special files. It replaces the `MAKEDEV` command. (See `MAKEDEV`(8).)
  - See `disklabel`(8) for information on maintaining disk pack labels.
  - See `diskconfig`(8) for instructions on invoking the Disk Configuration GUI, a disk management tool that provides additional features over `disklabel`. You can use it to partition disks and create file systems on the disks in a single operation. You can also launch the Disk Configuration interface from the CDE Application Manager - System_Admin folder. The Disk Configuration icon is located in the Configuration folder. Online help describes how to use this interface.

1.5.2 Device Special File Directories

To contain the device special files, a `/devices` directory exists under the root directory (`/`). This directory contains subdirectories that each contain device special files for a class of devices. A class of device corresponds to related types of devices, such as disks or nonrewind tapes. For example, the `/dev/disk` directory contains files for all supported disks, and the `/dev/ntape` directory contains device special files for nonrewind tape devices. In this release, only the subdirectories for certain classes are

created. For all operations you must specify paths by using the /dev directory and not the /devices directory.

Note

Some device special file directories are CDSLs, which enable devices to be available cluster wide when a system is part of a cluster. You should be familiar with the file system hierarchy described in the *System Administration* manual, in particular the implementation of CDSLs.

From the /dev directory, there are symbolic links to corresponding subdirectories to the /devices directory. For example:

```
lrwxrwxrwx 1 root system 25 Nov 11 13:02 ntape ->
../../../../devices/ntape

lrwxrwxrwx 1 root system 25 Nov 11 13:02 rdisk ->
../../../../devices/rdisk

lrwxrwxrwx 1 root system 24 Nov 11 13:02 tape ->
../../../../devices/tape
```

This structure enables certain devices to be host-specific when the system is a member of a cluster. It enables other devices to be shared between all members of a cluster. In addition, new classes of devices might be added by device driver developers and component vendors.

1.5.2.1 Pseudodevices and Non-storage Devices

To manage devices and their associated device special files, it is useful to know what is in the /dev directory on your system. In the /dev directory, you will find many devices listed, as shown in the following example output:

```
# ls -l
total 47
drwx------   2 root     system           512 Jun  5 15:55 .audit
-rwxr-xr-x   1 bin      bin            34777 Jun 19 00:02 MAKEDEV
-rw-r--r--   1 root     system          2238 Jun 28 16:42 MAKEDEV.log
-rwxr-xr-x   1 bin      bin             1418 Jun 19 00:02 SYSV_PTY
-rwxr-xr-x   1 bin      bin             1418 Aug  1  2001 SYSV_PTY.PreUPD
crw-------   1 root     system    47,      0 May 30 17:16 atm_cmm
crw-rw-rw-   1 root     system    87,      0 May 30 17:16 aud97
cr--------   1 root     system    17,      0 May 30 16:45 audit
srw-rw----   1 root     system             0 Jul 25 13:40 binlogdmb
crw-------   1 root     system    30,      0 May 30 16:45 cam
lrwxrwxrwx   1 root     system            27 May 30 17:16 changer -> \
../../../../devices/changer
crw--w--w-   1 root     daemon     0,      0 Jul 25 13:44 console
```

```
-rw-r--r--   1 root     system           12 Jul 25 13:39 console.gen
lrwxrwxrwx   1 root     system           25 May 30 17:16 cport -> \
../../../../devices/cport
lrwxrwxrwx   1 root     system           24 May 30 17:16 disk -> \
../../../../devices/disk
lrwxrwxrwx   1 root     system           25 May 30 17:16 dmapi -> \
../../../../devices/dmapi
crw-------   1 root     system    31,      0 Jul 26 13:43 kbinlog
crw-------   1 root     system     3,      1 May 30 16:45 kcon
crw-------   1 root     system    82,      0 May 30 17:16 kevm
crw-------   1 root     system    82,      2 May 30 17:16 kevm.pterm
crw-rw----   1 root     system    25,      1 May 30 16:45 keyboard0
crw-------   1 root     system     3,      0 May 30 16:45 klog
cr--r-----   1 root     mem        2,      1 May 30 16:45 kmem
drwxr-xr-x   2 root     system           512 May 12 18:14 lat
cr--r-----   1 root     mem       45,      0 May 30 16:45 lockdev
srw-rw-rw-   1 root     system           0 Jul 25 13:40 log
crw-rw-rw-   1 root     system    34,      0 May 30 17:16 lp0
cr--r-----   1 root     mem        2,      0 May 30 16:45 mem
crw-rw-rw-   1 root     system    88,      0 May 30 17:16 mmsess0
crw-rw----   1 root     system    23,      1 May 30 16:45 mouse0
crw-rw-rw-   1 root     system    52,      0 May 30 17:16 msb0
lrwxrwxrwx   1 root     system           15 May 30 17:16 none -> \
../devices/none
lrwxrwxrwx   1 root     system           25 May 30 17:16 ntape -> \
../../../../devices/ntape
crw-rw-rw-   1 root     system     2,      2 Jul 26 13:28 null
cr--r--r--   1 root     system    26,      0 May 30 16:45 pfcntr
crw-rw-rw-   1 root     system    32,     58 May 30 17:16 pipe
crw-rw-rw-   1 root     system    46,      0 May 30 16:45 poll
crw-------   1 root     system    37,      0 May 30 16:45 prf
srwxrwxrwx   1 root     system           0 Jul 25 13:43 printer
crw-rw-rw-   2 root     system    32,      7 May 12 18:12 ptm
crw-rw-rw-   1 root     system    32,     65 Jun 28 17:14 ptmx
crw-rw-rw-   2 root     system    32,      7 May 12 18:12 ptmx_bsd
drwxr-xr-x   2 root     system           512 May 12 18:12 pts
crw-rw-rw-   1 root     system     7,     31 May 30 16:45 ptyqf
crw-rw-rw-   1 root     system    89,      0 May 30 17:16 random
lrwxrwxrwx   1 root     system           25 May 30 17:16 rdisk -> \
../../../../devices/rdisk
drwxr-xr-x   2 root     system           512 May 30 17:16 sad
crw-------   1 root     system    80,1048575 May 30 17:16 scp_scsi
crw-rw-rw-   2 root     system    32,     49 May 12 18:12 snmpinfo
drwxr-xr-x   3 root     system           512 May 30 17:21 streams
crw-r--r--   1 root     system    15,      0 May 30 16:45 sysdev0
lrwxrwxrwx   1 root     system           24 May 30 17:16 tape -> \
../../../../devices/tape
crw-rw-rw-   1 root     system     1,      0 May 30 16:45 tty
crw-rw-rw-   1 root     system    35,      0 May 30 17:16 tty00
crw-rw-rw-   1 root     system    89,      1 Jul 25 13:39 urandom
```

```
crw-r-----   1 root     mem        2,      5 May 30 16:45 vmzcore
crw-rw----   1 root     system    33,      1 May 30 16:45 ws0
crw-rw-rw-   1 root     system    38,      0 May 30 16:45 zero
```

The preceding list is edited to remove duplicate device types. The directory usually contains many devices that are terminals (`ttys`), pseudoterminals (`ptys`), and disk storage devices (`dsk`). The number and varity of devices depend on the system's hardware and software configuration. Most of these devices are required by the operating system and applications, and do not relate to storage. You most frequently manage devices that involve data I/O, such as disks, tapes, and networking cards.

Use the `hwmgr show devices` command to list the devices and controllers that you manage most often. The output from this command excludes I/O devices that you cannot manage by using the `dsfmgr` or `hwmgr` commands. The output from the `hwmgr show devices` command also includes storage devices such as `dev/cport/scp2` which relate to devices such as SCSI cards, Fibre Channel switches, and storage array controllers. Entries for `dev/changer/mc*` are tape changer devices.

To obtain a list of such devices, use the following `hwmgr` command:

```
# hwmgr view dev
 HWID: Device Name          Mfg     Model           Location
 ------------------------------------------------------------------------
    6: /dev/dmapi/dmapi
    7: /dev/scp_scsi
    8: /dev/kevm
   53: /dev/cport/scp5              SWXCR            xcr0
   54: /dev/disk/dsk1197c           SWXCR            ctlr-0-unit-0
   69: /dev/disk/floppy53c          3.5in floppy     fdi0-unit-0
   46: /dev/disk/dsk1155c   DEC     HSG80            bus-4-targ-2-lun-17
   79: /dev/disk/dsk6c      DEC     HSZ22    (C) DEC bus-0-targ-0-lun-5
   81: /dev/random
   82: /dev/urandom
   84: /dev/ntape/tape11    COMPAQ  SDT-10000        bus-5-targ-0-lun-0
   90: /dev/disk/dsk1194c   COMPAQ  BD009122C6       bus-2-targ-0-lun-0
   91: /dev/disk/dsk1195c   COMPAQ  BD009122BA       bus-2-targ-1-lun-0
  871: /dev/disk/dsk1180c   DEC     HSG80            bus-4-targ-2-lun-13
  122: /dev/changer/mc1             TL800    (C) DEC bus-4-targ-0-lun-10
  127: /dev/cport/scp2              DATA ROUTER      bus-4-targ-0-lun-0
  128: /dev/cport/scp3              HSG80CCL         bus-4-targ-2-lun-0
  129: /dev/cport/scp4              HSV110 (C)COMPAQ bus-4-targ-4-lun-0
  926: /dev/ntape/tape0     COMPAQ  DLT8000          bus-4-targ-0-lun-8
  927: /dev/ntape/tape1     COMPAQ  DLT8000          bus-4-targ-0-lun-9
  942: /dev/disk/dsk1199c   COMPAQ  HSV110 (C)COMPAQ IDENTIFIER=1001
  687: /dev/disk/cdrom53c   COMPAQ  CDR-8435         bus-6-targ-0-lun-0
```

The preceding output is edited to remove many similar entries for disk storage devices. In this output, you can see that some kernel subsystem pseudodevices such as `/dev/kevm` (kernel event manager) are listed, but the numerous terminals and pseudoterminals are not listed. Information about kernel subsystems such as `/dev/kevm` is provided in the system attributes (`sys_attrs*`) reference pages. See `sys_attrs`(5). Other pseudodevices, such as `/dev/random` are described in reference pages. Use the `apropos`

command to locate the reference pages associated with a particular device as shown in the following examples:

```
# apropos random
.
.
random, urandom (4)     - Kernel random number source devices
# apropos disk
.
.
disk, dsk, cdrom, rz (7)  - SCSI disk interface
```

1.5.2.2 Legacy Device Special File Names

According to legacy device naming conventions, all device special files are stored in the `/dev` directory. The device special file names indicate the device type, its physical location, and other device attributes. Examples of the file name format for disk and tape device special file names that use the legacy conventions are `/dev/rz14f` for a SCSI disk and `/dev/rmt0a` for a tape device. The name contains the following information:

Path	Prefix	Type	Number (Instance)	Suffix
`/dev`	*r* (raw)	`rz` (disk)	`0`	`c` (partition)
`/dev`	(rewind)	`rmt` (tape)	`4`	`a` (density)
`/dev`	*n* (non-rewind)	`rmt` (tape)	`12`	`h` (density)

This information is interpreted as follows:

The path is the directory for device special files. All device special files are placed in the `/dev` directory.

The prefix differentiates one set of device special files for the same physical device from another set, as follows:

- `r` – Indicates a character (raw) disk device. Device special files for block devices have no prefix.
- `n` – Indicates a no-rewind-on-close tape device. Device special files for rewind-on-close tape devices have no prefix.

The type is the two or three-character driver name, such as `rz` for SCSI disk devices or `rmt` for tape devices.

The number is the unit number of the device, as follows:

- For SCSI disks, the unit number is calculated with the formula:

  ```
  unit = (bus * 8) + target
  ```

For HSZ40 and HSZ10 disk devices, a letter can precede the unit number to indicate the LUN, where a is LUN 0, b is LUN 1, and so on. You do not need to include the letter a for LUN 0 because it is the default.

- For tapes, the prefix is a sequential number from 0 through 7.

The suffix differentiates multiple device special files for the same physical device, as follows:

- Disks use the letters a through h to indicate partitions. In all, 16 files are created for each disk device: 8 for character device partitions a through h, 8 for block device partitions a through h.
- Tapes use suffixes to indicate tape densities. Up to eight files are created for each tape device: two for each density, using the suffixes defined in Table 1–2.

Table 1–2: Tape Device Suffix for Legacy Device Special Files

Suffix	Description
a	QIC-24 density for SCSI QIC devices.
l	The lowest density supported by the device, or QIC-120 density for SCSI QIC devices.
m	Medium density when a drive is triple density, or QIC-150 density for SCSI QIC devices.
h	The highest density supported by the device, or QIC-320 density for SCSI QIC devices.

Legacy device naming conventions are supported so that scripts continue to work as expected. However, features available with the current device naming convention might not work with the legacy naming convention. When Version 5.0 or higher is installed, none of the legacy device special files (such as rz13d) are created during the installation. If you determine that legacy device special file naming is required, you must create the legacy device names by using the appropriate commands described in dsfmgr(8). Some devices do not support legacy device special files.

1.5.2.3 Current Device Special File Names

Current device special files imply abstract device names and convey no information about the device architecture or logical path to the device. The new device naming convention consists of a descriptive name for the device and an instance number. These two elements form the basename of the device as shown in Table 1–3.

Table 1–3: Sample Current Device Special File Names

Location in /dev	Device Name	Instance	Basename
/disk	dsk	0	dsk0
/rdisk	dsk	0	dsk0
/disk	cdrom	1	cdrom1
/tape	tape	0	tape0

A combination of the device name, with a system-assigned instance number creates a basename such as dsk0.

The current device special files are named according to the basename of the devices, and include a suffix that conveys more information about the addressed device. This suffix differs depending on the type of device, as follows:

- Disks – These device file names consist of the basename and a suffix from a through z. For example, dsk0a. Disks use a through h to identify partitions. By default, CD-ROM and floppy disk devices use only the letters a and c. For example, cdrom1c and floppy0a.

  The same device names exist in the class directory /dev/rdisk for raw devices.

- Tapes – These device file names have the basename and a suffix comprised of the characters _d followed by an integer. For example tape0_d0. This suffix determines the density of the tape device, according to the entry for the device in the /etc/ddr.dbase file. For example:

Device	Density
tape0	Default density
tape0c	Default density with compression
tape0_d0	Density associated with entry 0 in /etc/ddr.dbase
tape0_d1	Density associated with entry 1 in /etc/ddr.dbase

Using the new device special file naming, there is a direct mapping from the legacy tape device name suffix to the current name suffix as follows:

Legacy Device Name Suffix	Current Suffix
l (low)	_d0
m (medium)	_d2
h (high)	_d1
a (alternate)	_d3

Two different sets of device names for tape devices conform to the current naming convention as follows:

`/dev/tape` – This directory contains device names for rewind tapes.

`/dev/ntape` – This directory contains device names for no rewind tapes.

To determine the correct device special file to use, look in the `/etc/ddr.dbase` file.

1.5.2.4 Converting Device Special File Names

If you have shell scripts that use commands that act on device special files, be aware that any command or utility supplied with the operating system operates on current and legacy file names in one of the following ways:

- The utility accepts both forms of device name.
- Only the current device names are supported by the utility. If you use legacy device names, you cannot use the command.
- Only the legacy device names are supported by the utility. If you use current device names, you cannot use the command.

No device can use both forms of device names simultaneously. Test your shell scripts for compliance with the device naming methods. Refer to the individual reference pages or the online help for a command.

If you want to update scripts, translating legacy names to the equivalent current name is a simple process. Table 1–4 lists some examples of legacy device names and corresponding current device names. There is no relationship between the instance numbers. A device associated with legacy device special file `/dev/rz10b` might be associated with `/dev/disk/dsk2b` under the current system.

Using these names as examples, you can translate device names that appear in your scripts. You can also use the `dsfmgr`(8) command to convert device names.

Table 1–4: Sample Device Name Translations

Legacy Device Special File Name	New Device Special File Name
`/dev/rmt0a`	`/dev/tape/tape0`
`/dev/rmt1h`	`/dev/tape/tape1_d1`
`/dev/nrmt0a`	`/dev/ntape/tape0_d0`
`/dev/nrmt3m`	`/dev/ntape/tape3_d2`

Table 1–4: Sample Device Name Translations (cont.)

Legacy Device Special File Name	New Device Special File Name
/dev/rz0a	/dev/disk/dsk0a
/dev/rz10g	/dev/disk/dsk10g
/dev/rrz0a	/dev/rdisk/dsk0a
/dev/rrz10b	/dev/rdisk/dsk10b

1.5.3 Managing Device Special Files

In most cases, the management of device special files is undertaken by the system itself. During the initial full installation of the operating system, the device special files are created for every SCSI disk and SCSI tape device found on the system. If you updated the operating system from a previous version by using the update installation procedure, both the current device special files and the legacy device files might exist. However, if you subsequently add new SCSI devices the `dsfmgr` command creates only the new device special files by default. When the system is rebooted, the `dsfmgr` command is called automatically during the boot sequence to create the new device special files for the device. The system also automatically creates the device special files that it requires for pseudodevices such as ptys (pseudoterminals).

When you add a SCSI disk or tape device to the system, the new device is found and recognized automatically, added to the hardware management databases, and its device special files created. On the first reboot after installation of the new device, the `dsfmgr` command is called automatically during the boot sequence to create the new device special files for that device.

To support applications that work only with legacy device names, you might need to manually create the legacy device special files, either for every existing device, or for recently added devices only. Some recent devices that support features such as Fibre Channel can use only the current special device file naming convention.

The following sections describe some typical uses of the `dsfmgr` command. See `dsfmgr`(8) for detailed information on the command syntax. The system script file `/sbin/dn_setup`, which runs at boot time to create device special files, provides an example of a script that uses `dsfmgr` command options.

1.5.3.1 Using dn_setup to Perform Generic Operations

The `/sbin/dn_setup` script runs automatically at system startup to create device special file names. The `/sbin/bcheckrc` utility also verifies device special files during system startup. If `/sbin/bcheckrc` discovers any problems, it displays the following message at the console:

```
bcheckrc: Device Naming failed initial check.
    Run dsfmgr -v, and if no errors are
    reported, exit or type CTRL-D to continue
    booting normally.
```

Normally, you do not need to use the dn_setup command. It is useful if you need to troubleshoot device name problems or restore a damaged special device file directory or database files. (See also Section 1.5.3.3.) If you frequently change your system configuration or install different versions of the operating system, you might see device-related error messages at the system console during system startup. These messages might indicate that the system is unable to assign device special file names. This problem can occur when the saved configuration does not map to the current configuration. Adding or removing devices between installations can also cause the problem.

The -sanity_check option alone is useful to administrators. Enter the following command to verify the device name database:

```
# /sbin/dn_setup -sanity_check
Passed.
```

If you see messages other than Passed, use the dsfmgr command to verify and fix the device name database. If the problem is serious and prevents a successful boot, you might be instructed to use other command options for debugging and problem solving under the guidance of your technical support office. See dn_setup(8).

1.5.3.2 Displaying Device Classes and Categories

Any individual type of device on the system is identified in the Category to Class-Directory, Prefix Database file, /etc/dccd.dat. You can display information in these databases by using the dsfmgr command. This information enables you to find out what devices are on a system, and obtain device identification attributes that you can use with other dsfmgr command options. For example, you can find a class of devices that have related physical characteristics, such as being disk devices. Each class of devices has its own directory in /dev such as /dev/ntape for nonrewind tape devices. Device classes are stored in the Device Class Directory Default Database file, /etc/dcdd.dat.

Use the following command to view the entries in the databases:

```
# /sbin/dsfmgr -s
dsfmgr: show all datum for system at /

Device Class Directory Default Database:
     # scope mode  name
    --  ---  ----  -----------
     1   l   0755  .
     2   c   0755  disk
```

```
     3   c   0755  rdisk
     4   c   0755  tape
     5   c   0755  ntape
     6   l   0755  none

Category to Class-Directory, Prefix Database:
 #   category       sub_category   type        directory  iw  t mode prefix
--   -------------- -------------- ----------  ---------  --  - ---- --------
 1   disk           cdrom          block       disk        1  b 0600 cdrom
 2   disk           cdrom          char        rdisk       1  c 0600 cdrom
 3   disk           floppy         block       disk        1  b 0600 floppy
 4   disk           floppy         char        rdisk       1  c 0600 floppy
 5   disk           floppy_fdi     block       disk        1  b 0666 floppy
 6   disk           floppy_fdi     char        rdisk       1  c 0666 floppy
 7   disk           generic        block       disk        1  b 0600 dsk
 8   disk           generic        char        rdisk       1  c 0600 dsk
 9   parallel_port  printer        *           .           1  c 0666 lp
10   pseudo         kevm           *           .           0  c 0600 kevm
11   tape           *              norewind    ntape       1  c 0666 tape
12   tape           *              rewind      tape        1  c 0666 tape
13   terminal       hardwired      *           .           2  c 0666 tty
14   *              *              *           none        1  c 0000 unknown

Device Directory Tree:
   12800    2 drwxr-xr-x  6 root system 2048 May 23 09:38 /dev/.
     166    1 drwxr-xr-x  2 root system  512 Apr 25 15:58 /dev/disk
    6624    1 drwxr-xr-x  2 root system  512 Apr 25 11:37 /dev/rdisk
     180    1 drw-r--r--  2 root system  512 Apr 25 11:39 /dev/tape
    6637    1 drw-r--r--  2 root system  512 Apr 25 11:39 /dev/ntape
     181    1 drwxr-xr-x  2 root system  512 May  8 16:48 /dev/none

Dev Nodes:
 13100  0 crw-------  1 root system 79,  0 May  8 16:47 /dev/kevm
 13101  0 crw-------  1 root system 79,  2 May  8 16:47 /dev/kevm.pterm
 13102  0 crw-r--r--  1 root system 35,  0 May  8 16:47 /dev/tty00
 13103  0 crw-r--r--  1 root system 35,  1 May  8 16:47 /dev/tty01
 13104  0 crw-r--r--  1 root system 34,  0 May  8 16:47 /dev/lp0
   169  0 brw-------  1 root system 19, 17 May  8 16:47 /dev/disk/dsk0a
  6627  0 crw-------  1 root system 19, 18 May  8 16:47 /dev/rdisk/dsk0a
   170  0 brw-------  1 root system 19, 19 May  8 16:47 /dev/disk/dsk0b
  6628  0 crw-------  1 root system 19, 20 May  8 16:47 /dev/rdisk/dsk0b
   171  0 brw-------  1 root system 19, 21 May  8 16:47 /dev/disk/dsk0c
    .
    .
    .
```

This display provides you with information that you can use with other `dsfmgr` commands. (See `dsfmgr`(8) for a complete description of the fields in the databases.) For example:

- `class` – The device class such as `disk` (a block device), `rdisk` (a character device), or `tape` (a rewind device). Use this information with the `dsfmgr -a` (add) or `dsfmgr -r` (remove) command options to add or remove classes.
- `category` – The primary description of a device. For example, SCSI disks, CD-ROM readers and floppy disk readers are all in the `disk` category. Use this information with the `dsfmgr -a` (add) or `dsfmgr -r` (remove) command options to add or remove categories.

1.5.3.3 Verifying and Fixing the Databases

Under unusual circumstances, the device databases might be corrupted or device special files might be accidentally removed from the system. You might see errors indicating that a device is no longer available, but the device itself does not appear to be faulty. If you suspect that there might be a problem with the device special files, you can check the databases by using the `dsfmgr -v` (verify) command option.

Caution

If you see error messages at system startup that indicate a device naming problem, use the verify command option to enable you to proceed with the boot. Check your system configuration before and after verifying the databases. The verification procedure fixes most errors and enables you to proceed. The verify option does not cure any underlying device or configuration problems.

Such problems are rare and usually arise when performing unusual operations such as switching between boot disks. Errors generally mean that the system is unable to recover and use a good copy of the previous configuration, and errors usually arise because the current system configuration no longer matches the database.

As for all potentially destructive system operations, ensure that you are able to restore the system to its identical previous configuration, and to restore the previous version of the operating system from your backup.

For example, if you attempt to configure the floppy disk device to use the `mtools` commands, and you find that you cannot access the device, use the following `dsfmgr` command to help diagnose the problem:

```
# /sbin/dsfmgr -v
dsfmgr: verify all datum for system at /

Device Class Directory Default Database:
    OK.

Device Category to Class Directory Database:
    OK.

Dev directory structure:
    OK.

Dev Nodes:
    ERROR: device node does not exist: /dev/disk/floppy0a
```

```
    ERROR: device node does not exist: /dev/disk/floppy0c
  Errors:   2

Total errors:   2
```

This output shows that the device special files for the floppy disk device are missing. To correct this problem, use the same command with the `-F` (fix) flag to correct the errors as follows:

```
# /sbin/dsfmgr -v -F

dsfmgr: verify all datum for system at /

Device Class Directory Default Database:
    OK.

Device Category to Class Directory Database:
    OK.

Dev directory structure:
    OK.

Dev Nodes:
    WARNING: device node does not exist: /dev/disk/floppy0a
    WARNING: device node does not exist: /dev/disk/floppy0c
    OK.

Total warnings:   2
```

In the preceding output, the `ERROR` changes to a `WARNING`, indicating that the device special files for the floppy disk are created automatically. If you repeat the `dsfmgr -v` command, no further errors are displayed.

1.5.3.4 Deleting Device Special Files

If a device is permanently removed from the system, you can remove its device special file to reassign the file to another type of device. Use the `dsfmgr -D` command option to remove device special files, as shown in the following example:

```
# ls /dev/disk
cdrom0a   dsk0a     dsk0c     dsk0e     dsk0g     floppy0a
cdrom0c   dsk0b     dsk0d     dsk0f     dsk0h     floppy0c

#  /sbin/dsfmgr -D /dev/disk/cdrom0*
 -cdrom0a -cdrom0a -cdrom0c -cdrom0c
#  ls /dev/disk
dsk0a     dsk0c     dsk0e     dsk0g     floppy0a
dsk0b     dsk0d     dsk0f     dsk0h     floppy0c
```

The output from the `ls` command shows that there are device special files for `cdrom0`. Running the `dsfmgr -D` command option on all `cdrom` devices,

as shown by the wildcard symbol (*), causes all device special files for that subcategory to be permanently deleted. The message that follows repeats the basename (`cdrom0`) twice, because it also deletes the device special files from the `/dev/rdisk` directory where the raw or character device special files are located.

If device special files are deleted in error, and no hardware changes are made, recreate the files as follows:

```
#  /sbin/dsfmgr -n cdrom0a

  +cdrom0a +cdrom0a
#  /sbin/dsfmgr -n cdrom0c

  +cdrom0c +cdrom0c
```

1.5.3.5 Moving and Exchanging Device Special File Names

You might want to move (reassign) the device special files between devices by using the `dsfmgr -m` (move) command option. You can also exchange the device special files of one device for those of another device by using the `-e` option. For example:

```
#  /sbin/dsfmgr -m dsk0 dsk10
#  /sbin/dsfmgr -e dsk1 15
```

The following procedure provides an example of how you use these command options when replacing a tape device. The example uses a desktop TLZ06 (DAT) SCSI drive as an emergency replacement for a failed internal tape drive. The procedure also applies to replacing an internal tape drive or adding and removing a drive mounted in a tape changer (jukebox).

This procedure applies to all configurations that have an available SCSI address and (if required) an appropriate free SCSI port. If you do not have an available SCSI bus, you might need to first remove the failed tape and then add the replacement. Adding a tape drive requires a quick reboot of the system. Advise users that the file system will be unavailable for whatever time it typically takes for your system to reboot.

The scenario described in the following procedure assumes that you want to preserve the identity of a device by transferring an existing device to the newly installed device instead of using the name that the system automatically assigns to the new device. For example, assume you create a backup script that uses tape device `tape0` of three possible devices attached to the system. The other tape devices are `tape1` and `tape2`. If the `tape0` fails and you replace it with a new device, the new device will be assigned the name `tape3`. Your backup script will then fail, because it addresses a missing device. You must either rewrite your scripts to address the new

device name, or rename the device so that it takes on the identity of the failed device.

Alternatively, consider the advantages of automatic device name creation, making your local scripts and utilities independent of device names. The script continues to function independent of any hardware configuration changes. The `dsfmgr` and `hwmgr` commands enable you to poll the system configuration and dynamically determine the characteristics and attributes of a device. This strategy is more efficient because it enables you to failover to a healthy device if your original target is unavailable.

Before you use the procedure you must do the following:

- Obtain a replacement tape drive and appropriate cabling. Consult the online specifications for your system to obtain information on supported options. The following URL provides information on supported options: *AlphaServer QuickSpecs*.

  Newer tape peripheral models might not be listed as a supported option for an older processor, however the installation documentation supplied with the device usually document any restrictions on its use.

  Note

  You must ensure that the tape drive is a supported model of the correct SCSI mode (such as SCSI 2, fast wide) and is compatible with your SCSI bus adapter. If in doubt, contact your technical support office.

- The owner's manual for any device that is to be removed or added to the system. The owner's manual contains important safety information that is not duplicated here. It might also contain important information that will prevent damage to your system. During the procedure, you might need to set the SCSI target on your tape drive as described in the owner's manual.
- Verify that your system's firmware is current. You can obtain information and download kits from the *Firmware Updates* Web page.
- Read `hwmgr`(8) for more information on using the `hwmgr` command.

Start the procedure as follows:

1. Use the `hwmgr` command to display information about all SCSI devices as follows:

```
# /sbin/hwmgr show scsi
       SCSI             DEVICE DEVICE  DRIVER NUM  DEVICE FIRST
HWID: DEVICEID HOSTNAME TYPE   SUBTYPE OWNER  PATH FILE   VALID PATH
------------------------------------------------------------------
 32:  0        f2394    disk   none    2      1    dsk0   [0/0/0]
```

```
 33:  1        f2394    disk   none    2      1    dsk1   [0/1/0]
 34:  2        f2394    cdrom  none    0      1    cdrom0 [0/4/0]
 35:  3        f2394    disk   none    0      1    dsk4   [0/2/0]
 41:  4        f2394    tape   none    0      1    tape0  [1/3/0]
```

Pipe the output to a file to create a record of the existing SCSI devices.

The output from the preceding command provides the following information, which is relevant to this procedure:

- The hardware identifier (HWID) is an integer assigned to the device by the system. In our example, HWID 41 (`tape0`) is the failed device.
- The SCSI DEVICEID (device identifier), is an integer. This is also called the `did`. You specify this number when you want to remove a device from the SCSI database.
- The DEVICE FILE is a string in the format *`device:instance`*, such as `tape1` for tape drive 1. This is an abbreviation of the location for the device special file `/dev/tape/tape1` in the case of a rewind device. Device special files for non-rewind devices are in the `/dev/ntape` directory.
- The FIRST VALID PATH is a string representing the SCSI bus, target, and logical unit number (LUN), which are three integers in the format *`N/N/N`*. This data represents the first valid data path to the device.

  If you are adding or replacing a tape drive, you might need to determine which SCSI bus target is free for the new device. Tape devices such as the TLZ06 often have a manual method for setting the SCSI target as described in the owner's manual. Make a note of any unused addresses. In the preceding example, bus 1 has only one device at address 3, which is the existing tape device (`1/3/0`). You can therefore set the SCSI target for your tape device as `1`, `2`, or `4-6`.

  Note

  If the output from the `hwmgr` command shows blank fields under the VALID PATH column, the absence of a bus, target, and LUN address indicates an I/O problem with the device. This might be the cause of the failure of the device. Troubleshooting information at the end of this procedure addresses this possibility.

  If one or more entries in the FIRST VALID PATH column is blank, make a note of the remaining (assigned) addresses. You can determine the missing addresses in Step 3.

2. Use the `sysman shutdown` or `shutdown` command to notify users and shut down the system. Specify that you want to stop the operating system by using the `-h` (halt) option . For example:

```
# shutdown -h +10
"System shutting down to replace backup device \
Back in 15 minutes"
```

3. At the console prompt, determine the current device address assignments by using the following command:

```
>>> show config
```

The output from this command includes a list of devices attached to each SCSI bus. For example, `MKB300.3.0.13.0` is the entry for a TLZ06 tape device attached at target `3` on bus `B`. The last three numbers are the bus, target, LUN address.

If you were unable to determine an available bus address in Step 1, you can determine it based on the output from the `show config` command.

4. Switch off power to the system using the front panel switch and wait for at least 20 seconds before removing cables.

5. Complete this step only if either of the following conditions apply. Otherwise, go to Step 6:

- You intend to permanently remove a tape drive from your system and you will not replace it with a new tape drive.

  (You can temporarily remove a tape drive from a system for use elsewhere. Its system record will persist, and you can replace the tape later and reassign its system record.)

- You need to remove an existing tape drive before installing a new one. This might be because the tape drive is installed in an internal bay, or because you need to reuse its logical bus address because of a lack of space.

If either condition applies, proceed as follows:

a. Switch off the power to the tape drive and physically remove it from the system as described in its owner's manual.

b. At the console prompt, reboot your system to single-user mode as follows:

```
>>> boot -flags s
```

c. Use the following command to delete the tape:

```
# /sbin/hwmgr delete scsi -did NN
```

The value specified for the `-did` option is the SCSI DEVICEID that you determined in Step 1.

d. Under rare circumstances, the deletion may not be complete. You should verify that the device is gone by reentering the `hwmgr show scsi` command as described in Step 1. You should also verify the content of the `/dev/tape` and `/dev/ntape` directories to ensure that the device special files are gone.

If any record of the device remains on the system, see the problem solving information later in this procedure.

6. Proceed with this step only if you intend to replace the tape device.

   Connect the new tape drive as described in the owner's manual. Switch on the power to the tape drive and to your system in the appropriate order.

   Reboot the system to single-user mode as follows:

   ```
   >>> boot -flags s
   ```

7. Watch the boot procedure for messages. During the reboot, the new tape drive is automatically detected and a message is displayed at the console that is similar to the following:

   ```
   dsfmgr:  Note creating device special files
   +tape16...
   ```

   This message confirms that the new tape device was discovered and the device special files are being created in the `/dev/tape` or `/dev/ntape` directory. If you do not see such a message, you must create the device special files manually. (However, you should first attempt to complete this procedure.)

8. When your system has rebooted to multiuser level, enter the following command to display information about the newly added tape drive:

   ```
   # /sbin/hwmgr show scsi
         SCSI              DEVICE DEVICE  DRIVER NUM  DEVICE FIRST
   HWID: DEVICEID HOSTNAME TYPE   SUBTYPE OWNER  PATH FILE   VALID PATH
   ---------------------------------------------------------------------
   48:  5        f2394     tape   none    0      1    tape16 [1/4/0]
   ```

   Comparing this command output to the output in Step 1, you can see that the new disk is now listed as `tape16`, HWID 48. Notice that the DEVICE FILE (device special file) instance for the new tape drive is `16` which considerably higher than that of the original tape drive (`0`).

9. When you replace a device, you might also want to take one of the following actions:

   a. Transfer the device name from an existing (perhaps failed) tape drive to a replacement drive. Use the `dsfmgr` command with the `-e` (exchange) option, specifying the DEVICE FILE as follows:

      ```
      # /sbin/dsfmgr -e tape16 tape0
      tape16<==>tape0       tape16_d0<==>tape0_d0 tape16_d1<==>tape0_d1
      tape16_d2<==>tape0_d2 tape16_d3<==>tape0_d3 tape16_d4<==>tape0_d4
      ```

```
tape16_d5<==>tape0_d5 tape16_d6<==>tape0_d6 tape16_d7<==>tape0_d7
tape16c<==>tape0c     tape16<==>tape0       tape16_d0<==>tape0_d0
tape16_d1<==>tape0_d1 tape16_d2<==>tape0_d2 tape16_d3<==>tape0_d3
tape16_d4<==>tape0_d4 tape16_d5<==>tape0_d5 tape16_d6<==>tape0_d6
tape16_d7<==>tape0_d7 tape16c<==>tape0c
```

The output contains bidirectional (<==>) arrows, indicating that the device special file names are exchanged. Verify the exchange as follows:

```
# /sbin/hwmgr show scsi
      SCSI              DEVICE DEVICE  DRIVER NUM  DEVICE FIRST
HWID: DEVICEID HOSTNAME TYPE   SUBTYPE OWNER  PATH FILE   VALID PATH
---------------------------------------------------------------------
48:   5        f2394    tape   none    0      1    tape0  [1/4/0]
```

The tape drive at SCSI address `1/4/0` (the newly added drive) is now `tape0`.

b. Change the device special file to a lower number or `0`. For example, the `dump` command defaults to `tape0_d0` (tape `0` with default density). If you are using a tape changer (jukebox) with multiple tape drives, you might need to renumber the devices so that your local scripts work as expected.

   Use the `dsfmgr` command with the `-m` (move) option, specifying the DEVICE FILE as follows:

```
# /sbin/dsfmgr -m tape16 tape1

tape16=>tape1       tape16_d0=>tape1_d0 tape16_d1=>tape1_d1
tape16_d2=>tape1_d2 tape16_d3=>tape1_d3 tape16_d4=>tape1_d4
tape16_d5=>tape1_d5 tape16_d6=>tape1_d6 tape16_d7=>tape1_d7
tape16c=>tape1c     tape16=>tape1       tape16_d0=>tape1_d0
tape16_d1=>tape1_d1 tape16_d2=>tape1_d2  ape16_d3=>tape1_d3
tape16_d4=>tape1_d4 tape16_d5=>tape1_d5 tape16_d6=>tape1_d6
tape16_d7=>tape1_d7 tape16c=>tape1c
```

The output contains one-way arrows to indicate that the device special file names are moved. Verify the exchange as follows:

```
# /sbin/hwmgr show scsi
      SCSI              DEVICE DEVICE  DRIVER NUM  DEVICE FIRST
HWID: DEVICEID HOSTNAME TYPE   SUBTYPE OWNER  PATH FILE   VALID PATH
---------------------------------------------------------------------
46:   4        f2394    tape   none    0      1    tape1  [0/5/0]
48:   5        f2394    tape   none    0      1    tape0  [0/6/0]
```

The tape drive at SCSI address `0/5/0` is now named `tape1`.

The checkpoints embedded in the procedure should ensure correct completion. Under rare circumstances, the addition of a new tape drive or

the removal of an existing tape drive might fail. You can resolve this problem by using the following troubleshooting procedure:

1. Keep a careful record of the options that you attempt in case you need to contact your field service representative.

   Avoid a trial-and-error approach by trying various combinations of `hwmgr` and `dsfmgr` commands. Most likely, the problem is that the device's database records are in an indeterminate state. Trying various command options might cause more problems.

2. Start the Event Manager (EVM) viewer and ensure that it is configured to display events of priority 300 and greater:

   ```
   # /usr/sbin/sysman event_viewer
   ```

   Use the event viewer to verify hardware events generated by the procedure or to trap errors. You must refresh the event viewer to view current events. The event viewer displays events confirming that the tape drive was recognized and assigned a new basename (`tape*`) and a hardware identifier (HWID). It may take some time for the system to post these events and you might need to refresh the event viewer to see the events.

3. Use the following command to scan the SCSI buses and assign identifiers to all new devices that are found:

   ```
   # /sbin/hwmgr scan scsi
   hwmgr: Scan request successfully initiated
   ```

   A hardware scan is asynchronous and, although the system prompt is returned almost immediately, it does not signify the end of a scan. A scan can take several minutes on a system with many SCSI devices. You must wait for the scan to complete. Because a scan is normally performed automatically on system startup, it does not display a message when it completes.

4. When the scan is complete, use the following command to verify whether the tape drive was detected:

   ```
   # /sbin/hwmgr show scsi
         SCSI              DEVICE DEVICE  DRIVER NUM  DEVICE FIRST
   HWID: DEVICEID HOSTNAME TYPE   SUBTYPE OWNER  PATH FILE   VALID PATH
   ------------------------------------------------------------------
   46:   4        f2394    tape   none    0      1    tape1  [0/5/0]
   48:   5        f2394    tape   none    0      1           [1/4/0]
   ```

5. If you notice a missing device special file in the `hwmgr` output in troubleshooting Step 4, verify whether or not the device special files were created for the newly added device. Verify the content of the `/dev/tape` or `/dev/ntape` directories as appropriate.

If there is no device special file, use the `dsfmgr` command (see `dsfmgr`(8)) with the `-K` option to create the device special files as follows:

```
# /sbin/dsfmgr -K
+tape16 +tape16_d0 +tape16_d1…
```

6. If the preceding steps are unsuccessful, the device database might be in an indeterminate state. Look for an entry for the tape drive in the `/etc/dfsc.dat` file by searching for the HWID or device special file name. For example, if the tape drive is listed as `tape0`, enter the following command:

   ```
   # grep "tape0" /etc/dfsc.dat
   ```

   The output from the preceding command is similar to the following:

   ```
   A: 0 130003e    48    9    6   c  ""   /dev/tape  tape 0
   ```

   In the preceding output, the fourth column contains the hardware identifier (HWID). In this case, it is 48.

7. Use the following command to confirm the status of the device, using the HWID 48 from the preceding step:

   ```
   # /sbin/hwmgr show component -id 48
   ```

   If there is output from the command, contact your technical support representative. Otherwise, proceed to the next step.

8. If you still cannot access the device, reboot your system to reset the device database. Contact your technical support representative if you are unable to reboot the system.

1.5.3.6 Renumbering Device Special Files

When you reconfigure components on your system (such as by deleting devices), the instance number assigned to device special files is incremented. In time, you might find that the device special files appear to be randomly numbered. Also, if you back up the system by using a clone-copy-delete procedure, the instance value increases at each backup and device names can become large and difficult to manage. Because clone backup programs must also determine the new device instances at each backup, the changing device special file names will increase the time required for a backup.

Using the `dsfmgr -vI` command option minimizes (or resets) the instance number of each device to the lowest possible value. If your system has a fixed configuration except for the backup procedure, using this option ensures that each new set of cloned backup devices always has the same new names, simplifying your backup procedure. The following procedure demonstrates how to use the `dsfmgr -vI` command to reset the device instances for all

devices to the lowest possible value. The following procedure demonstrates how the `dsfmgr -vI` command works:

1. The following (truncated) output shows how a tape's device special file is initially numbered on a newly installed system:

   ```
   # /sbin/hwmgr view device
   61: /dev/ntape/tape0    COMPAQ  SDT-10000
   bus-5-targ-0-lun-0
   ```

2. To simulate the effect of many changes to the system's configuration, the following command renumbers the tape's device special file to 40:

   ```
   # /sbin/dsfmgr -m tape0 tape40
   tape0=>tape40  tape0_d0=>tape40_d0  tape0_d1=>tape40_d1
   tape0_d2=>tape40_d2  tape0_d3=>tape40_d3
   tape0_d4=>tape40_d4.......
   ```

3. The following (truncated) output shows how a tape's device special file is now increased to `tape40`:

   ```
   # /sbin/hwmgr view device
   .61: /dev/ntape/tape40    COMPAQ   SDT-10000
   bus-5-targ-0-lun-0
   ```

4. To simulate the effect of removing devices from the system, the following command renumbers the tape's device special file to 10:

   ```
   # /sbin/dsfmgr -m tape0 tape40
   tape40>tape10.
   ```

   This means that there exist a number of unused device special file instances in the range11 through 39. However, the system is currently unaware that these instances are free, and it will continue to number newly installed devices at instance 41.

5. To cause the system to allocate the lowest available instance, enter the following command:

   ```
   # /sbin/dsfmgr -vI
   dsfmgr: verify all datum for system (version) at /

   Default File Tree:
       OK.

   Device Class Directory Default Database:
       OK.

   Device Category to Class Directory Database:
       OK.

   Dev directory structure:
       OK.
   ```

```
Device Status Files:
    OK.

Dev Nodes:
    OK.

Minimize next instance values:
      tape     41  =>   11
```

The message “Minimize next instance values:” shows that the lowest available instance is now 11.

6. To test that the operation is successful, the following commands delete and rescan the tape device to cause the system to allocate a new device special file at instance 11:

```
# /sbin/hwmgr delete scsi -did 3
hwmgr: The delete operation was successful
# /sbin/hwmgr scan scsi
hwmgr: Scan request successfully initiated
# /sbin/hwmgr show scsi
84:  3        argo1      tape      none    0      1
tape11 [5/0/0]
```

The tape’s device special file is now numbered 11, indicating that the system used the lowest available instance.

To use this option on your system, you need only amend your backup script, or create a new script that runs the `dsfmgr -vI` command option at the appropriate time. Alternatively, run the command after performing any hardware configuration changes.

2

Using the SysMan Menu and SysMan Station

This chapter describes the available SysMan Menu tasks that can assist you in administering the system hardware. The utilities work on single systems and on clustered systems. You can run the SysMan Menu remotely from different operating environments, or invoke it as a Web application.

This chapter also describes how you use the SysMan Station to monitor hardware status and to launch hardware management utilities. The SysMan Station is an X-compliant graphical user interface (GUI) that runs under the Common Desktop Environment (CDE) or other X-windows user environment.

This chapter contains the following topics:

- How to use the following SysMan Menu hardware tasks:
  - How to view the hardware hierarchy (Section 2.1.1)
  - How to view the cluster (Section 2.1.2)
  - How to view device information (Section 2.1.3)
  - How to view central processing unit (CPU) information (Section 2.1.4)
- How to use the SysMan Station to find and monitor components and devices (Section 2.2)

The information and examples in this chapter describe how to invoke and use the SysMan Menu and SysMan Station from the Tru64 UNIX command line or from CDE. See the *System Administration* manual for information on using alternative operating environments such as the Web.

Most hardware management operations require root user privileges but you can assign such privileges to nonroot users by using the SysMan division of privileges (DOP) feature. See `dop`(8) for more information.

The SysMan Menu also provides the following serviceability tasks:

- Manage CPUs
- Online Addition and Replacement (OLAR) policy information

For information on these feature, see `hwmgr_ops`(8) and the *Managing Online Addition and Removal* manual.

2.1 Using the SysMan Menu Hardware Tasks

The SysMan Menu tasks provide you with a subset of the many more hardware management options that are available from the command line when you use the `hwmgr` command. A more detailed discussion of the `hwmgr` command and its options is located in Chapter 3. See `hwmgr`(8) for a complete listing of the command syntax and options. Selecting the help option in one of the SysMan Menu hardware tasks invokes the appropriate reference pages.

When you invoke the SysMan Menu as described in the *System Administration* manual, hardware management options are available under the Hardware branch of the menu. Doubleclick on this branch to expand it and display the following tasks:

- View hardware hierarchy
- View cluster
- View device information
- View central processing unit (CPU) information

These tasks launch basic hardware management tasks that are described in the following sections. See the *Managing Online Addition and Removal* manual for information on online addition and removal (OLAR).

The following option buttons (or choices, in a terminal) are available in all the tasks:

- Rerun – Runs the command again, updating the information in the display.
- Stop – Stops the command. Use the Rerun option to update the information or choose OK to exit.
- OK – Ends the task and closes the window.
- Help – Displays the reference page.

2.1.1 Viewing the Hardware Hierarchy

The View Hardware Hierarchy task invokes the`/sbin/hwmgr view hierarchy` command. The following example shows output from a single-CPU system that is not part of a cluster:

```
                View hardware hierarchy

HWID:  hardware component hierarchy
-------------------------------------------------
   1:  platform AlphaServer 800 5/500
   2:    cpu CPU0
   4:    bus pci0
   5:      connection pci0slot5
```

```
13:         scsi_adapter isp0
14:           scsi_bus scsi0
30:             disk bus-0-targ-0-LUN-0 dsk0
31:             disk bus-0-targ-4-LUN-0 cdrom0
 7:       connection pci0slot6
15:         graphics_controller trio0
 9:       connection pci0slot7
16:         bus eisa0
17:           connection eisa0slot9
18:             serial_port tty00
19:           connection eisa0slot10
display truncated
```

Use this task to display the hardware hierarchy for the entire system or cluster. The hierarchy shows every bus, controller, and other components on the system from the CPUs down to the individual peripheral components such as disks and tapes. On a system or cluster that has many devices, the output is lengthy and you might need to scroll the display to see components at the beginning of the output.

The output is useful because it provides you with component information that you can specify with `hwmgr` command options to perform hardware management operations such as viewing more component detail and adding or deleting devices. You can use the following items shown in the hierarchy as command input:

- HWID – The hardware identifier (or `id`), which is an integer that is unique to each individual entry in the hierarchy.
- The component name, such as `pci` for the Peripheral Component Interconnect (PCI) bus.
- The component basename, which is a mnemonic followed by an integer that identifies the component such as `cdrom0`, which relates to the device special file for the component (`/dev/disk/cdrom0`). More information on device special file names is located in Section 1.5.
- The physical location attribute specifies the address or path to a device, such as `bus-0-targ-0-LUN-0`, sometimes written as `0/0/0`, which provides the following information:
  - `scsi-0` is the bus and provides number of the bus to which the component is attached.
  - `targ-0` is the target number for this component on the bus, in this case the first target on bus 0.
  - `LUN-0` is the logical unit number (LUN), in this case the first logical unit number at target 0 on bus 0.
- The hardware category of a device, such as a `bus` or `ide_controller`.

- Connections to slots, which show the slot number for a device, such as `pci0slot5` and `eisa0slot9`.
- Bus, controller, and component relationships, such as the following sample output showing two disk devices on controller `scsi_adapter isp0`, which is on the bus `scsi_bus scsi0`:

```
13:         scsi_adapter isp0
14:           scsi_bus scsi0
30:             disk bus-0-targ-0-LUN-0 dsk0
31:             disk bus-0-targ-4-LUN-0 cdrom0
```

Because the same component might be shared (for example, on a shared bus), it might appear in the hierarchy more than once and has a unique identifier each time it appears. An example of shared devices is provided in Section 3.4.7.

You can use the information from the `view hierarchy` command output in other `hwmgr` commands when you want to focus an operation on a specific hardware component, as shown in the following command, which gets the value of a component attribute named `device_starvation_time` for the component with the HWID (`id`) of 30. Component 30 is the SCSI disk at bus 0, target 0 and LUN 0 in the example hierarchy:

```
# /sbin/hwmgr get attribute -id 30 \
-a device_starvation_time
30:
  device_starvation_time = 25 (settable)
```

The output shows that the value of the `device_starvation_time` attribute is 25. The label `(settable)` indicates that this is a configurable attribute that you can set by using the following command option:

```
# /sbin/hwmgr set attribute -id 35 \
-a device_starvation_time=30
```

Understand the impact of the changes before modifying the value of any component attribute. See the documentation provided with a device.

2.1.2 Viewing the Cluster

Selecting the View Cluster task invokes the command `/sbin/hwmgr view cluster`, directing the output to the SysMan Menu window (or screen, if a terminal) as follows:

```
                                View cluster

Starting /sbin/hwmgr view cluster ...

/sbin/hwmgr view cluster run at Fri May 21 13:42:37 EDT 1999
```

```
Member ID          State           Member HostName
---------          -----           ---------------
     1              UP                    rene (localhost)
    31              UP                    witt
    10              UP                    rogr
```

If you attempt to run this command on a system that is not a member of a cluster, the following message is displayed:

```
hwmgr: This system is not a member of a cluster.
```

You can specify the `Member ID` and the `HostName` in some `hwmgr` commands when you want to focus an operation on a specific member of a cluster, as shown in the following example:

```
# /sbin/hwmgr scan scsi -member witt
```

2.1.3 Viewing Device Information

Selecting the View Device Information task invokes the command `/sbin/hwmgr view devices`, directing the output to the SysMan Menu window (or screen, if a terminal).

Use this option to display the component information for the entire system or cluster. The output shows every component and pseudodevice (such as the `/dev/kevm` pseudodevice) that is connected to system. The following example shows the output from a small single-CPU system that is not part of a cluster:

```
                          View device information

Starting /sbin/hwmgr view devices ...

/sbin/hwmgr view devices run at Fri May 21 14:20:08 EDT 1999

HWID:  Device Special File  Mfg Model           Location
                      Name
------------------------------------------------------------------
 3:            /dev/kevm
28:   /dev/disk/floppy0c        3.5in floppy    fdi0-unit-0
30:      /dev/disk/dsk0c   DEC  RZ1DF-CB(C)DEC  bus-0-targ-0-LUN-0
31:    /dev/disk/cdrom0c   DEC  RRD47  (C)DEC   bus-0-targ-4-LUN-0
```

For the purpose of this command, a component is any entity in the hierarchy that has the attribute `dev_base_name` and has an associated device special file (DSF). The output from this command provides the following information that you can use with the `hwmgr` command to perform hardware management operations on the device:

- The hardware identifier (HWID or `id`), an integer that is unique to each individual entry in the hierarchy,

- The DSF Name, such as `/dev/disk/cdrom0c`. In the case of disk devices, this is the name of the device special file associated with the `c` partition that maps to the entire capacity of the disk. For a tape, it shows the device special file name that maps to the default density for the device. See Section 1.5 for a description of these names.
- The model, which specifies a manufacturer model number or a generic description such as `3.5in floppy`.
- The physical location of a device, such as the SCSI `bus-0-targ-0-LUN-0`, sometimes written as 0/0/0, which specifies the following:
  - `bus-0` – The number of the bus to which the component is attached. In this case, it is SCSI bus 0.
  - `targ-0` – The target number for this component on the bus. In this case, it is the first target on the bus.
  - `LUN-0` – The logical unit number. In this case it is the first lun on the bus.

  The previous output also shows a floppy disk attached to the floppy disk interface, `fdi` as device 0, unit 0.

You can specify this information to certain `hwmgr` commands to perform hardware management operations on a particular device. The following example of disk location specifies a device special file for a disk, causing the light (LED) on that disk to flash for 30 seconds:

```
# /sbin/hwmgr locate -id 60 -time 90
hwmgr: Locate request successfully initiated
```

The preceding command works only for some SCSI devices, and might not work for disks that are part of a managed array, such as an HSV110. However, storage arrays usually detect a failed disk and signal its failure by flashing an amber or red disk error light.

2.1.4 Viewing CPU Information

Selecting the View Central Processing Unit (CPU) Information task invokes the command `/usr /sbin/psrinfo -v`, directing the output to the SysMan Menu window (or screen, if a terminal). Use this option to display the CPU status information, as shown in the following sample output for a single-processor system.

The output from this task describes the processor and its status:

```
                                /usr/sbin/psrinfo
Starting /usr/sbin/psrinfo -v ...

/usr/sbin/psrinfo -v run at Fri May 21 14:22:05 EDT 1999

Status of processor 0 as of: 05/21/99 14:22:05
```

```
Processor has been on-line since 05/15/1999 14:42:28
 The alpha EV5.6 (21164A) processor operates at 500 MHz,
  and has an alpha internal floating point processor.
```

2.2 Using the SysMan Station

The SysMan Station is a graphical user interface that runs under various windowing environments or from a Web browser. See the *System Administration* manual and the online help for information on launching and using the SysMan Station. Features of the SysMan Station that assist you in hardware management are as follows:

Monitoring systems and devices

The SysMan Station provides a live view of system and component status. You can customize views to focus on parts of a system or cluster that are of most interest to you. You are notified when a hardware problem occurs on the system by color changes to icons displayed by the GUI. System views are hierarchical, showing the complete system topology from CPUs down to discrete components such as tapes. You can observe the layout of buses, controllers, and adapters and see their logical addresses. You can see what components are attached to each bus or controller, and their slot numbers. Such information is useful for running `hwmgr` commands from the command prompt.

Viewing device properties (or attributes)

You can select a component and view detailed attributes of that device. For example, if you select a SCSI disk and press the right mouse button, a menu of options is displayed. You can choose to view the component properties for the selected disk. If you opt to do this, an extensive table of component properties is displayed. This action is the same as using the `hwmgr` command, as shown in the following (truncated) sample output:

```
# hwmgr get attr -id 30
30:
  name = SCSI-WWID:0c000008:0060-9487-2a12-4ed2
  category = disk
  sub_category = generic
  architecture = SCSI
  phys_location = bus-0-targ-0-LUN-0
  dev_base_name = dsk0
  access = 7
  capacity = 17773524
  block_size = 512
  open_part_mask = 59
  disk_part_minor_mask = 4294967232
  disk_arch_minor_mask = 4290774015
```

display truncated

Launching hardware management tasks

When you select a device, you can also choose to launch a command and perform configuration or daily administrative operations on the selected device. For example, if you select a network adapter, you can configure its settings or perform related tasks such as configure the domain name server (DNS). You can launch the Event Viewer to see if any system events (such as errors) pertaining to this component are posted.

See the *System Administration* manual for more information on remote management options.

3

Using hwmgr to Manage Hardware

The principal command that you use to manage hardware is the `hwmgr` command line interface (CLI). Other interfaces, such as the SysMan tasks provide a limited subset of the features provided by `hwmgr`. For example, you can use `hwmgr` to set an attribute for all components of a particular type (such as SCSI disks) on all SCSI adapters in all members of a cluster.

Most hardware management is performed automatically by the system and you need only intervene under certain circumstances, such as replacing a failed component so that the replacement component takes on the identity of the failed component. This chapters discusses the following topics:

- Understanding the hardware management model (Section 3.1)
- Understanding the principal user options available for the `hwmgr` command (Section 3.2)
- Configuring the `hwmgr` command environment (Section 3.3)
- Performing the following administrative tasks using the `hwmgr` command (Section 3.4):
  - Finding the location of SCSI devices (Section 3.4.1)
  - Viewing the hardware hierarchy or system topology (Section 3.4.2)
  - Viewing the component categories (Section 3.4.3)
  - Viewing component attributes (Section 3.4.4)
  - Setting the modifiable component attributes (Section 3.4.5)
  - Viewing the cluster hardware status (Section 3.4.6)
  - Viewing component status (Section 3.4.7)
  - Viewing hardware management transactions (Section 3.4.8)
  - Defining a custom World-Wide Identifier (Section 3.4.9)
  - Deleting SCSI devices (Section 3.4.10)
  - Reconfiguring disks under RAID arrays (Section 3.4.11)
  - Replacing failed SCSI disks (Section 3.4.12)
  - Replacing failed SCSI tape drives. You can also use this procedure to replace disks if the `redirect` option fails (Section 3.4.13)

- Replacing a failed cluster boot disk (Section 3.4.14)
- Viewing the database for the name subsystem (Section 3.4.15)
- Deleting and removing a component from the name persistence database (Section 3.4.16)
- Using the `refresh` option to remove stale paths and optimizes databases (Section 3.4.17)
- Renaming components (Section 3.4.18)
- Moving a components within the system (Section 3.4.19)

3.1 Understanding the Hardware Management Model

Within the operating system kernel, hardware data is organized as a hardware set managed by the kernel set manager. Application requests are passed by library routines to kernel code, or remote code. The latter deals with requests to and from other systems. The hardware component module (HWC) resides in the kernel, and contains all the registration routines to create and maintain hardware components in the hardware set. It also contains the device nodes for device special file management, which is performed by using the `dsfmgr` command.

The hardware set consists of data structures that describe all of the hardware components that are part of the system. A hardware component becomes part of the hardware set when registered by its driver. Many components support attributes that describe their function and content or control how they operate. Each attribute is assigned a value. You can read, and sometimes manipulate, these attribute values by using the `hwmgr` command.

The system hardware is organized into three parts, identified as subsystems by the `hwmgr` command. The subsystems are identified as component, SCSI, and name. The subsystems are related to the system hardware databases as follows:

- The component subsystem references all hardware components specified in the (binary) `/etc/dec_hwc_ldb` and `/etc/dec_hwc_cdb` databases. This includes most components on a system.
- The name subsystem references all hardware components in the binary `/etc/dec_hw_db` database, often referred to as the hardware topology. The database contains hardware persistence information, maintained by the kernel driver framework and includes data for buses, controllers, and components.
- The SCSI subsystem references all SCSI devices in the binary `/etc/dec_scsi_db` database. The SCSI database contains entries for all devices managed by the SCSI/CAM architecture.

The general features of hwmgr are as follows:

- It provides a wide range of hardware management functions under a single command.
- It enables you to manage (to a small extent) hardware components that are currently not connected to your system but were seen on a previous boot.
- It enables you to manage hardware components that are connected to multiple systems in a cluster.
- It enables you to propagate a management request to multiple members of a cluster.

3.2 Understanding hwmgr Command Options

The hwmgr command works with the kernel hardware management module, providing you with the ability to manage hardware components. Examples of a hardware component are storage peripherals, such as a disk or tape, or a system component such as a CPU or a bus. Use the hwmgr command to manage hardware components on either a single system or on a cluster.

Operational commands are characterized by a subsystem identifier after the command name. The subsystems are: component, scsi, and name.

Some hwmgr operation commands are available for more than one subsystem. You should use the subsystem most closely associated with the type of operation you want to perform, depending on the parameter information that you obtained using the view and show command options.

Some commands require you to specify a subsystem name. However, if you specify the identity of a hardware component, then you do not need to specify a subsystem name. The hwmgr command is able to determine the correct subsystem on which to operate, based on the component identifier.

The command options are organized by task application. The command options, the subsystems on which they operate, and the nature of the operation are listed in the following table:

Option	Subsystem	Operation
add	name	Database management
delete	component, name, and scsi	Database management
edit	name, scsi	Database management
locate	component	Hardware configuration

Option	Subsystem	Operation
offline	component, name	Online Addition and Removal
online	component, name	Online Addition and Removal
power	component, name	Online Addition and Removal
redirect	scsi	Hardware configuration
refresh	component, scsi	Database management
reload	name	Driver configuration
remove	name	Database management
scan	component, name, and scsi	Hardware configuration
status	component	Hardware configuration
unconfigure	component, name	Hardware configuration
unindict	component	Online Addition and Removal
unload	name	Driver configuration

3.3 Configuring the hwmgr Environment

The hwmgr command provides environment settings that you can use to control the amount of information displayed. Use the following command to display the default environment settings:

```
# /sbin/hwmgr view env

  HWMGR_DATA_FILE = "/etc/hwmgr/hwmgr.dat"
  HWMGR_DEBUG = FALSE
  HWMGR_HEXINTS = FALSE
  HWMGR_NOWRAP = FALSE
  HWMGR_VERBOSE = FALSE
```

You can set the value of environment variables in your login script, or at the command line as shown in the following example:

```
# HWMGR_VERBOSE=TRUE
# export HWMGR_VERBOSE
```

You usually need to define only the value of the HWMGR_HEXINTS HWMGR_NOWRAP, and the HWMGR_VERBOSE environment variables as follows:

- If the HWMGR_HEXINTS environment variable is defined as TRUE, any numerical data output from the hwmgr command is displayed in hexadecimal numbers.

- If the HWMGR_NOWRAP environment variable is defined as TRUE, the output from the hwmgr command is truncated at 80 characters. In some cases it is difficult to read the output from hwmgr command options because it wraps. Setting the value of the HWMGR_NOWRAP environment variable to TRUE makes the output more legible at the console. A horizontal ellipsis marks truncated lines.
- If the HWMGR_VERBOSE environment variable is defined as TRUE, the output from the hwmgr command contains more detailed information. The default setting of the hwmgr command is to hide any errors that are not critical. To view more verbose information, you can also append the verbose switch to any of the hwmgr command options.

  For example, if you query an attribute that does not exist for all hardware components, by default the hwmgr command displays only the output from hardware components that support the attribute, as shown in the following example:

```
# /sbin/hwmgr get attribute -a type
  6:
   type = local
  7:
   type = local
  9:
   type = MOUSE
```

  Not all hardware components on the system support the type attribute, If the HWMGR_VERBOSE environment variable is not defined as TRUE, the errors generated by the preceding command are suppressed. To see the errors, use the -verbose switch with the command line as follows:

```
# /sbin/hwmgr get attribute -a type -verbose
  1: Attribute "type" not defined.
  2: Attribute "type" not defined.
  4: Attribute "type" not defined.
  5: Attribute "type" not defined.
  6:
   current type = local
  7:
   current type = local
  8: Attribute "type" not defined.
  9:
   current type = MOUSE
  10: Attribute "type" not defined.
  11: Attribute "type" not defined.
            .
            .
  (long display, output truncated)
```

  You can use the verbose switch with all hwmgr commands, although it does not always produce additional output.

3.4 Using hwmgr to Manage Hardware

The following sections contain examples of tasks that you might need to perform by using the `hwmgr` command. Some of these examples might not be useful for managing a small server with few components attached to the CPU. However, when you are managing a large installation with many networked systems or clusters with hundreds of components, they become very useful. Using the `hwmgr` command enables you to connect to an unfamiliar system, obtain information about its component hierarchy, and then perform administrative tasks without any previous knowledge about how the system is configured and without consulting system logs or files to find components.

3.4.1 Locating SCSI Hardware

The `locate` option, which currently works only for some SCSI devices, enables you to identify a device. You might use this command when you are trying to physically locate a SCSI disk. The following command flashes the light on a SCSI disk for one minute:

```
# /sbin/hwmgr locate -id 42 -time 60
```

You can then look the disk bays for the component that is flashing its light. You cannot use this option to locate some SCSI devices such as CD-ROM readers and disks that are part of an array (such as an HSV110). However, disks that are part of an array are detected on failure and an event is posted in the Event Manager log. The array controller identifies the failed disk by flashing its read or amber error light. You do not need to manually search for the failed device.

3.4.2 Viewing the System Hierarchy

Use the `view` command to view the hierarchy of hardware within a system. This command enables you to find what adapters are controlling devices, and discover where adapters are installed on buses. The following example shows typical output on a small system that is not part of a cluster:

```
# /sbin/hwmgr view hierarchy
HWID:   hardware hierarchy
-------------------------------------------------
   1:   platform AlphaServer 800 5/500
   2:     cpu CPU0
   6:     bus pci0
   7:       connection pci0slot5
  15:         scsi_adapter isp0
  16:           scsi_bus scsi0
  32:             disk bus-0-targ-0-lun-0 dsk0
  33:             disk bus-0-targ-4-lun-0 cdrom0
  34:             disk bus-0-targ-5-lun-0 dsk1
```

```
  35:             disk bus-0-targ-6-lun-0 dsk2
  36:             disk bus-0-targ-8-lun-0 dsk3
   9:       connection pci0slot6
  17:         graphics_controller s3trio0
output truncated
```

Some components might appear as multiple entries in the hierarchy. For example, if a disk is on a SCSI bus that is shared between two adapters, the hierarchy shows two entries for the same device. You can obtain similar views of the system hardware hierarchy by using the SysMan Station GUI.

3.4.3 Viewing Component Categories

To perform hardware management options on all components of the same category, or to select a particular component in a category, you might need to know what categories of components are available. The hardware manager `get category` command fetches all the possible values for hardware categories.

This command is useful when you use it in conjunction with the `get attributes` and `set attributes` options, which enable you to display and configure the attributes (or properties) of a particular component. When you know the hardware categories you can limit your attribute queries to a specific type of hardware, as follows:

```
# /sbin/hwmgr get category
  Hardware Categories
  -------------------
  category = undefined
  category = platform
  category = cpu
  category = pseudo
  category = bus
  category = connection
  category = serial_port
  category = keyboard
  category = pointer
  category = scsi_adapter
  category = scsi_bus
  category = network
  category = graphics_controller
  category = disk
  category = tape
```

Knowing the categories, you can focus your attribute query by specifying a category as follows:

```
# /sbin/hwmgr get attribute -category platform
  1:
   name = AlphaServer 800 5/500
   category = platform
```

This output informs you that the system platform has a hardware ID of 1, and that the platform name is AlphaServer 800 5/500. See also the `get attribute` and `set attribute` command options.

3.4.4 Obtaining Component Attributes

Attributes are characteristics of the component that might be read-only information, such as the model number of the component, or you might be able to set a value to control some aspect of the behavior of the component, such as the speed at which it operates. The `get attribute` command option fetches and displays attributes for a component. The hardware manager command is specific to managing hardware and fetches only attributes from the hardware set. All hardware components are identified by a unique hardware identifier, otherwise known as the hardware ID or HWID.

The following command fetches all attributes for all hardware components on the local system and directs the output to a file that you can search for information:

```
# /sbin/hwmgr get attribute > sysattr.txt
```

However, if you know which component category you want to query, as described in Section 3.4.3, you can focus your query on that particular category.

Querying a hardware component category for its attributes can provide useful information. For example, you might not be sure if the network is working for some reason. You might not even know what type of network adapters are installed in a system or how they are configured. Use the `get attribute` option to determine the status of network adapters as shown in the following example:

```
# /sbin/hwmgr get attribute -category network
  203:
   name = ln0
   category = network
   sub_category = Ethernet
   model = DE422
   hardware_rev =
   firmware_rev =
   MAC_address = 08-00-2B-3E-08-09
   MTU_size = 1500
   media_speed = 10
   media_selection = Selected by Jumpers/Switches
   media_type =
   loopback_mode = 0
   promiscuous_mode = 0
   full_duplex = 0
```

```
multicast_address_list = CF-00-00-00-00-00 \
 01-00-5E-00-00-01
interface_number = 1
```

This output provides you with the following information:

- The number 203 is the HWID for this Ethernet adapter.
- The fields and values listed below the HWID are the attribute names and their current values. Some values might be blank if they are not initialized by the driver. Using this information, you know that the system has a model DE422 Ethernet adapter that has a component name of `ln0`.
- You can then check the status of this network adapter by using the `ifconfig` command, as follows:

```
# ifconfig ln0
ln0: flags=c62 inet XX.XXX.XXX.XX netmask ffffff00 \
broadcast XX.XXX.XX.XXX ipmtu 1500
```

In some cases, you can change the value of a component attribute to modify component information or change its behavior on the system. Setting attributes is described in Section 3.4.5. To find which attributes are settable, you can use the `get` option to fetch all attributes and use the `grep` command to search for the `(settable)` keyword as follows:

```
# /sbin/hwmgr get attribute | grep settable

   device_starvation_time = 25 (settable)
   device_starvation_time = 0 (settable)
   device_starvation_time = 25 (settable)
   device_starvation_time = 25 (settable)
   device_starvation_time = 25 (settable)
   device_starvation_time = 25 (settable)
```

The output shows that there is one settable attribute on the system, `device_starvation_time`. Having found this, you can now obtain a list of components that support this attribute as follows:

```
# /sbin/hwmgr get attribute -a device_starvation_time
  23:
   device_starvation_time = 25 (settable)
  24:
   device_starvation_time = 0 (settable)
  25:
   device_starvation_time = 25 (settable)
  31:
   device_starvation_time = 25 (settable)
  34:
   device_starvation_time = 25 (settable)
  35:
   device_starvation_time = 25 (settable)
```

The output from this command displays the HWID of the components that support the device_starvation_time attribute. Reading the HWID in the hierarchy output, you can determine that this attribute is supported by SCSI disks.

To determine the link speed of a Fibre Channel adapter, you can query its link speed, as follows:

```
# /sbin/hwmgr get attribute -a link_speed
656:
  link_speed = 1Ghz
```

If more than one adapter is connected, the preceding command displays the attribute value for all the adapters. The link_speed attribute supports the following values:

- A speed value, which can be 1GHz or 2GHz.
- The string N/A (not applicable), which indicates that the adapter firmware does not support the link_speed attribute.

See also the set attribute and get category options.

3.4.5 Setting Component Attributes

The set attribute command option allows you to set (or configure) the value of settable attributes. You cannot set all component attributes. When you use the get attribute command option, the output flags any configurable attributes by labeling them as (settable) next to the attribute value. A method of finding such attributes is described in Section 3.4.4.

As demonstrated in Section 3.4.4, the value of device_starvation_time is an example of a settable attribute supported by SCSI disks. This attribute controls the amount of time that must elapse before the disk driver determines that a component is unreachable due to SCSI bus starvation (no data transmitted). If the device_starvation_time expires before the driver is able to determine that the component is still there, the driver posts an error event to the binary error log.

Using the following commands, you can change the value of the device_starvation_time attribute for the component with the HWID of 24, and then verify the new value:

```
# /sbin/hwmgr set attribute -id 24 \
-a device_starvation_time=60
# /sbin/hwmgr get attribute -id 24 \
-a device_starvation_time
  24:
   device_starvation_time = 60 (settable)
```

This action does not change the saved value for this attribute. All attributes have three possible values, a current value, a saved value and a default

value. The default value is a constant and you cannot modify it. If you never set a value of an attribute, the default value applies. When you set the saved value, it persists across boots. You can think of it as a permanent override of the default.

When you set the current value, it does not persist across reboots. You can think of it as a temporary value for the attribute. When a system is rebooted, the value of the attribute reverts to the saved value (if there is a saved value). If there is no saved value, the attribute value reverts to the default value. Setting an attribute value always changes the current value of the attribute. The following examples show how you get and set the saved value of an attribute:

```
# /sbin/hwmgr get attribute  saved -id 24 \
-a device_starvation_time
  24:
   saved device_starvation_time = 0 (settable)

# /sbin/hwmgr get attribute saved -id 24 \
-a device_starvation_time=60
    saved device_starvation_time = 60 (settable)
# /sbin/hwmgr get attribute saved -id 24 \
-a device_starvation_time
  24:
   saved device_starvation_time = 60 (settable)
```

See also the get attribute and get category command options.

3.4.6 Viewing the Cluster

If you are working on a cluster, you often need to focus hardware management commands at a particular host on the cluster. The view cluster command option enables you to obtain details of the hosts in a cluster. The following sample output shows a typical cluster:

```
# /sbin/hwmgr view cluster
  Member ID     State   Member HostName
  ---------     -----   ---------------
    1           UP      ernie.zok.paq.com (localhost)
    2           UP      bert.zok.paq.com
    3           DOWN    bigbird.zok.paq.com
```

You can also use this option to verify that the hwmgr command is aware of all cluster members and their current status.

The preceding example indicates a three member cluster with one member (bigbird) currently down. The (localhost) marker indicates that hwmgr is currently running on cluster member ernie. Any hwmgr commands that you enter by using the -cluster option are sent to members bert and ernie, but not to bigbird because that system is unavailable. Additionally,

any hwmgr commands that you issue with the -member bigbird option fail because the cluster member state for that host is DOWN.

The view cluster command option works only if the system is a member of a cluster. If you attempt to run it on a single system, an error message is displayed. See also the clu_get_info command, and the TruCluster Server documentation for more information on clustered systems.

3.4.7 Viewing Devices

You can use the hwmgr command to display all components that have a device special file name, such as /dev/disk/dsk34 by using the view devices option. The hardware manager considers any hardware component that has the attribute dev_base_name to be an accessible device. (See Section 3.4.4 for information on obtaining the attributes of a device.)

The view devices option enables you to determine what components are currently registered with hardware management on a system, provides information that enables you to access these components through their device special file. For example, if you load a CD-ROM into a reader, use this output to determine whether you mount the CD-ROM reader as /dev/disk/cdrom0. The view devices option is also useful to find the HWIDs for any registered devices. When you know the HWID for a device, you can use other hwmgr command options to query attributes on the device, or perform other operations on the device.

Typical output from this command is shown in the following example:

```
# /sbin/hwmgr view dev


  HWID:             DSF Name Mfg      Model       Location

------------------------------------------------------------------------
     3:            /dev/kevm
    22:      /dev/disk/dsk0c DEC      RZ26        bus-0-targ-3-LUN-0
    23:    /dev/disk/cdrom0c DEC      RRD42       bus-0-targ-4-LUN-0
    24:      /dev/disk/dsk1c DEC      RZ26L       bus-1-targ-2-LUN-0
    25:      /dev/disk/dsk2c DEC      RZ26L       bus-1-targ-4-LUN-0
    29:     /dev/ntape/tape0 DEC      TLZ06       bus-1-targ-6-LUN-0
    35:     /dev/disk/dsk8c  COMPAQ   RZ1CF-CF    bus-2-targ-12-LUN-0
```

The output shows all hardware components that have the dev_base_name attribute on the local system. The hardware manager attempts to resolve the dev_base_name to the full path location to the device special file, such as /dev/ntape/tape0. It always uses the path to the device special file with the c partition. The c partition represents the entire capacity of the device, except in the case of tapes. See Chapter 1 for information on device special file names and functions.

If you are working on a cluster, you can view all components registered with hardware management across the entire cluster with the `-cluster` option, as follows:

```
# /sbin/hwmgr view devices -cluster


  HWID:             DSF Name    Model       Location        Hostname

------------------------------------------------------------------
   20:   /dev/disk/floppy0c    3.5in          fdi0-unit-0   tril7e
   34:    /dev/disk/cdrom0c    RRD46   bus-0-targ-5-LUN-0   tril7e
   35:      /dev/disk/dsk0c    HSG80   bus-4-targ-1-LUN-1   tril7d
   35:      /dev/disk/dsk0c    HSG80   bus-6-targ-1-LUN-1   tril7e
   36:      /dev/disk/dsk1c    RZ26N   bus-1-targ-0-LUN-0   tril7e
   37:      /dev/disk/dsk2c    RZ26N   bus-1-targ-1-LUN-0   tril7e
   38:      /dev/disk/dsk3c    RZ26N   bus-1-targ-2-LUN-0   tril7e
   39:      /dev/disk/dsk4c    RZ26N   bus-1-targ-3-LUN-0   tril7e
   40:      /dev/disk/dsk5c    RZ26N   bus-1-targ-4-LUN-0   tril7e
   41:      /dev/disk/dsk6c    RZ26N   bus-1-targ-5-LUN-0   tril7e
   42:      /dev/disk/dsk7c    RZ26N   bus-1-targ-6-LUN-0   tril7e
   43:      /dev/disk/dsk8c    HSZ40   bus-3-targ-2-LUN-0   tril7d
   43:      /dev/disk/dsk8c    HSZ40   bus-3-targ-2-LUN-0   tril7e
   44:      /dev/disk/dsk9c    HSZ40   bus-3-targ-2-LUN-1   tril7d
   44:      /dev/disk/dsk9c    HSZ40   bus-3-targ-2-LUN-1   tril7e
   45:     /dev/disk/dsk10c    HSZ40   bus-3-targ-2-LUN-2   tril7d
   45:     /dev/disk/dsk10c    HSZ40   bus-3-targ-2-LUN-2   tril7e
```

Some devices, such as the disk with the HWID of `45:`, appear more than once in this display. These are components that are on a shared bus between two cluster members. The hardware manager displays the component entry as seen from each cluster member.

See also the following `hwmgr` command options: `show scsi`, `show components`, and `get attributes`.

3.4.8 Viewing Transactions

Hardware management operations are transactions that must be synchronized across a cluster. The `view transaction` command option displays the state of any hardware management transactions that have occurred since you booted the system. Use this option to find failed hardware management transactions.

If you do not specify the `-cluster` or `-member` option, the command displays status on transactions that are processed or initiated by the local host (the system on which the command is entered). The `view transaction` command option is primarily for debugging problems with hardware management in a cluster, and you are likely to use this command infrequently. The command has the following typical output:

```
# /sbin/hwmgr view transactions
   hardware management transaction status
  -------------------------------------------------
  there is no active transaction on this system
```

```
  the last transaction initiated from this system was:
   transaction = modify cluster database
   proposal    = 3834
   sequence    = 0
   status      = 0
  the last transaction processed by this system was:
   transaction = modify cluster database
   proposal    = 3834
   sequence    = 0
   status      = 0

proposal                      last status  success  fail
---------------------------  -----------  -------  ----
             Modify CDB/ 3838  0            3        0
               Read CDB/ 3834  0            3        0
           No operation/ 3835  0            1        0
            Change name/ 3836  0            0        0
            Change name/ 3837  0            0        0
              Locate HW/ 3832  0            0        0
                Scan HW/ 3801  0            0        0
  Unconfig HW - confirm/ 3933  0            0        0
   Unconfig HW - commit/ 3934  0            0        0
    Delete HW - confirm/ 3925  0            0        0
     Delete HW - commit/ 3926  0            0        0
  Redirect HW - confirm/ 3928  0            0        0
  Redirect HW - commit1/ 3929  0            0        0
  Redirect HW - commit2/ 3930  0            0        0
        Refresh - lock/ 3937  0            0        0
```

This output indicates that the last transaction was a modification of the cluster database.

3.4.9 Creating a User-Defined SCSI Device Name

Most components have an identification attribute that is a unique to the device. You can read it as the `serial_number` or name attribute of a SCSI device. For example, the following `hwmgr` command returns both these attributes for the component with a HWID of 30, a SCSI disk:

```
# /sbin/hwmgr get attribute -id 30 -a serial_number -a name
30:
  serial_number = SCSI-WWID:0c000008:0060-9487-2a12-4ed2
  name = SCSI-WWID:0c000008:0060-9487-2a12-4ed2
```

This string is known as a worldwide identifier (WWID) because it is unique for each component on the system.

Some components do not provide a unique identifier. The operating system creates such a number for the component by using valid path `bus/target/LUN` data that describes the physical location of the device. Because systems can share devices, each system that has access to the

component sees a different path and creates its own unique WWID for that device. Concurrent I/O access to such shared devices might occur, possibly resulting in data corruption. To find such devices, use the following command:

```
# /sbin/hwmgr show component -cshared

 HWID:  HOSTNAME   FLAGS SERVICE COMPONENT NAME
---------------------------------------------
   40:  joey       -cd-- iomap   SCSI-WWID:04100026:"DEC \
 RZ28M    (C) DEC00S846590H7CCX"
   41:  joey       -cd-- iomap   SCSI-WWID:04100026:"DEC \
 RZ28L-AS (C) DECJEE019480P2VSN"
   42:  joey       -cd-- iomap   SCSI-WWID:0410003a:"DEC \
 RZ28     (C) DECPCB=ZG34142470  ; HDA=000034579643"
   44:  joey       rcd-- iomap   SCSI-WWID:04100026:"DEC \
 RZ28M    (C) DEC00S735340H6VSR"
.
.
.
```

Some devices, such as the TL895 model media changer, do not support `INQUIRY` pages 0x80 or 0x83 and are unable to provide the system with a unique WWID. To support features such as path failover or installation into a cluster on a shared bus, you must manually add such devices to the system. This is the recommended method to add only media changers to a shared bus. Other types of devices such as disks, CD-ROM readers, tape drives, or RAID controllers provide a unique string (such as a serial number), from which the system can create a unique WWID. You can use such a component on a shared bus because its WWID is always the same and the operating system always recognizes it as the same device.

You can use the `hwmgr` command to create a user-defined unique name that in turn enables you to create a WWID known to all systems that are sharing the device. Because the component has a common WWID, it has one set of device special file names, preventing the risk of concurrent I/O.

The process for creating a user-defined name is as follows:

- Choose the name that you want to assign. Make this name unique within the scope of all systems that have access to the device. Although the name need not be as long and complex as the WWIDs shown in the preceding example, make it sufficiently long to provide the information that you need to recognize the renamed component and differentiate it from others.
- Decide what component uses this name. When renamed, the component is seen as the same component on all systems. You must update the systems so that the component is seen.

- Each system that shares the component creates a new WWID by using the string and uses this new WWID for all subsequent registrations with the system. Internally, the component is still tracked by its default WWID (if one existed). However, all external representations display the new WWID based on the user-defined name. On a cluster you must run the `edit scsi` command option on every cluster member that has access to the device.

Caution

You must update all clustered systems that have access to the device.

The following example shows how you assign a user-defined name. Although the `edit scsi` command option is recommended only for devices that do not have a unique WWID, the example uses disks for the sake of simplicity.

```
# /sbin/hwmgr show scsi

      SCSI           DEVICE DEVICE  DRIVER NUM  DEVICE FIRST
HWID: DEVICEID HOST  TYPE   SUBTYPE OWNER  PATH FILE   VALID
      ID       NAME                              PATH
 -------------------------------------------------------------
  22: 0       ftwod  disk   none    0      1   dsk0   [0/3/0]
  23: 1       ftwod  cdrom  none    0      1   cdrom0 [0/4/0]
  24: 2       ftwod  disk   none    0      1   dsk1   [1/2/0]
  25: 3       ftwod  disk   none    2      1   dsk2   [2/4/0]
```

This command displays which SCSI devices are on the system. On this system the administrator knows that there is a shared bus and that hardware components 24 and 25 are actually the same device. The WWID constructed for this component is constructed by using the bus/target/LUN address information. Because the bus/target/LUN addresses are different, the component is seen as two separate devices. This can cause data corruption problems because the operating system might use two different sets of device special files to access the disk (`dev/disk/dsk1` and `/dev/disk/dsk2`).

The following command shows how you can rename the device, and how it appears after it is renamed:

```
# /sbin/hwmgr edit scsi -did 2 \
-uwwid "this is a test"
    hwmgr: Operation completed successfully.

# /sbin/hwmgr show scsi -did 2 -full

       SCSI              DEVICE  DEVICE  DRIVER NUM  DEVICE FIRST
  HWID: DEVICEID HOSTNAME  TYPE    SUBTYPE OWNER  PATH FILE   VALID PATH
  -----------------------------------------------------------------------
```

```
  24:  2        ftwod     disk    none    0      1    dsk1   [1/2/0]

     WWID:0910003c:"DEC    (C) DECZG41400123ZG41800340:d01t00002l00000"
     WWID:ff10000e:"this is a test"

     BUS   TARGET  LUN   PATH STATE
     ------------------------------
     1     2       0     valid
```

You repeat the operation on the other component path and the same name is given to the component at address 2/4/0. After you do this, hardware management uses your user-defined name to track the component and to recognize the alternate paths to the same device:

```
# /sbin/hwmgr edit scsi -did 3 -uwwid "this is a test"
    hwmgr: Operation completed successfully.

# /sbin/hwmgr show scsi -did 3 -full

       SCSI              DEVICE  DEVICE  DRIVER NUM  DEVICE FIRST
 HWID: DEVICEID HOSTNAME  TYPE    SUBTYPE OWNER  PATH FILE   VALID PATH
 ---------------------------------------------------------------------
  25:  3        ftwod     disk    none    0      1    dsk1   [2/4/0]

     WWID:0910003c:"DEC    (C) DECZG41400123ZG41800340:d02t00004l00000"
     WWID:ff10000e:"this is a test"

     BUS   TARGET  LUN   PATH STATE
     ------------------------------
     2     4       0     valid
```

Both of these devices now use device special file name (`/dev/disk/dsk1`). There is no longer any risk of data corruption resulting from two sets of device special files accessing the same disk.

3.4.10 Deleting a SCSI Device

Under some circumstances, you might want to remove a SCSI device from a system, such as when it is logging errors and you must replace it. Use the `delete scsi` command option to remove a SCSI component from all hardware management databases clusterwide. This option unregisters the component from the kernel, removes all persistent database entries for the device, and removes all device special files. When you delete a SCSI component, it is no longer accessible and its device special files are removed from the appropriate `/dev` subdirectory. You cannot delete a SCSI component that is currently open. You must terminate all I/O connections to the device (such as mounts).

You might need to delete a SCSI component if you are removing it from your system and you do not want information about the component remaining on the system. You might also want to delete a SCSI component because of operating system problems, rather than hardware problems. For example, if the component operates correctly but you cannot access it through the device

special file for some reason. In this case you can delete the component and use the `scan scsi` command option to find and register it.

To replace the SCSI component (or bring the old component back) you can use the `scan scsi` command option to find the component again. However, when you delete a component and then perform a `scan` operation to bring the component back on line, it does not always have the same device special file. To replace a component as an exact replica of the original, you must perform the additional operations described in Section 3.4.12. A `scan` operation might not find the component if it is not actively responding during the bus scan.

This option accepts the SCSI device identifier `-did`, which is not equivalent to the HWID. The following examples show how you examine the SCSI database and then delete a SCSI device:

```
# /sbin/hwmgr show scsi

      SCSI           DEVICE DEVICE  DRIVER NUM  DEVICE FIRST
HWID: DEVICEID HOST- TYPE   SUBTYPE OWNER  PATH FILE   VALID
               NAME                                    PATH
-------------------------------------------------------------
23:   0       bert   disk   none    2      1   dsk0   [0/3/0]
24:   1       bert   cdrom  none    0      1   cdrom0 [0/4/0]
25:   2       bert   disk   none    0      1   dsk1   [1/2/0]
30:   4       bert   tape   none    0      1   tape2  [1/6/0]
31:   3       bert   disk   none    0      1   dsk4   [1/4/0]
34:   5       bert   disk   none    0      1   dsk7   [2/5/0]
35:   6       bert   disk   none    0      1   dsk8
```

In this example, the `DRIVER OWNER` field is not zero for component ID 23, indicating that the device is currently open by a driver. Any number other than zero in the `DRIVER OWNER` field means that a driver has opened the component for use. Therefore, you cannot delete SCSI component 23 because it is currently in use.

However, component ID 35 is not open by a driver, and it currently has no valid paths shown in the `FIRST VALID PATH` field. The component is not currently accessible and you can delete it safely. When you delete the device, you also delete the `/dev/disk/dsk8*` and `/dev/rdisk/dsk8*` device special files.

To delete the SCSI device, specify the SCSI DEVICEID value with the `delete` option, and then review the SCSI database as follows:

```
# /sbin/hwmgr delete scsi -did 6
   hwmgr: The delete operation was successful.
# /sbin/hwmgr show scsi

      SCSI            DEVICE  DEVICE  DRIVER NUM  DEVICE FIRST
HWID: DEVICE HOSTNAME TYPE    SUBTYPE OWNER  PATH FILE   VALID
      ID                                                 PATH
```

```
  ------------------------------------------------------------
23:   0      bert     disk    none    2      1   dsk0   [0/3/0]
24:   1      bert     cdrom   none    0      1   cdrom0 [0/4/0]
25:   2      bert     disk    none    0      1   dsk1   [1/2/0]
30:   4      bert     tape    none    0      1   tape2  [1/6/0]
31:   3      bert     disk    none    0      1   dsk4   [1/4/0]
34:   5      bert     disk    none    0      1   dsk7   [2/5/0]
```

The component /dev/disk/dsk8 is successfully deleted.

3.4.11 Reconfiguring Disks Under RAID Arrays

When you reconfigure RAID arrays the new block zero might be the same block as the previous block zero. This can lead to problems caused by applications that see the disklabel as valid even though it might extend beyond the end of the disk. After a scan, the system recognizes the new unit(s) as dsk*NN*. Before using the disk, run the following command to zero any inappropriate label:

```
# /sbin/disklabel -z dskNN
```

Run this command when you construct a new unit on a RAID array or when you move one or more disks comprising a unit on a raid array to connect them directly to a host bus adapter.

Next, run the disklabel command to create a new default label (or apply a preconfigured label from a proto file) as follows:

```
# /sbin/disklabel -rwn dskNN
# /sbin/disklabel -Rr dskNN PROTOFILE
```

3.4.12 Replacing a Failed SCSI Disk

When a SCSI disk fails, you might want to replace it in such a way that the replacement disk takes on hardware characteristics of the failed device, such as ownership of the same device special files. The redirect command option enables you to assign such characteristics. For example, if you have an HSZ (RAID) cabinet and a disk fails, you can hot-swap the failed disk and then use the redirect command option to bring the new disk on line as a replacement for the failed disk.

Do not use this procedure alone if a failed disk is managed by an application such as AdvFS or LSM. Before you can swap managed disks, you must put the disk management application into an appropriate state or remove the disk from the management application. See the appropriate documentation, such as the *Logical Storage Manager* and *AdvFS Administration* manuals.

Note

The replacement disk must be of the same type for the `redirect` operation to work.

The following example shows how you use the `redirect` option:

```
# /sbin/hwmgr show scsi
      SCSI          DEVICE DEVICE DRIVER NUM  DEVICE  FIRST
HWID: DEVICE- HOST- TYPE   SUB-   OWNER  PATH FILE    VALID
      ID      NAME         TYPE                       PATH
  ------------------------------------------------------
 23:   0     fwod  disk   none   2      1    dsk0   [0/3/0]
 24:   1     fwod  cdrom  none   0      1    cdrom0 [0/4/0]
 25:   2     fwod  disk   none   0      1    dsk1   [1/2/0]
 30:   4     fwod  tape   none   0      1    tape2  [1/6/0]
 31:   3     fwod  disk   none   0      1    dsk4
 37:   5     fwod  disk   none   0      1    dsk10  [2/5/0]
```

This output shows a failed SCSI disk of HWID 31. The component has no valid paths. To replace this failed disk with a new disk that has device special file name `/dev/disk/dsk4`, and the same `dev_t` information, use the following procedure:

1. Install the component as described in the hardware manual.
2. Use the following command to find the new device:

   ```
   # /sbin/hwmgr scan scsi
   ```

   This command probes the SCSI subsystem for new devices and registers those devices. You can then repeat the `show scsi` command and obtain the SCSI device id (`did`) of the replacement device.
3. Use the following command to reassign the component characteristics from the failed disk to the replacement disk. This example assumes that the SCSI device id (`did`) assigned to the new disk is 36:

   ```
   # /sbin/hwmgr redirect scsi -src 3 -dest 36
   ```

Note

If the `redirect` option fails to work for a disk, use the alternate procedure described in Section 3.4.13.

3.4.13 Replacing a Failed SCSI Tape Drive (or Hard Disk)

When a SCSI tape fails, you might want to replace it in such a way that the replacement drive takes on hardware characteristics of the failed device, such as ownership of the same device special files. The `redirect` command

option described in Section 3.4.12 might not work for certain SCSI tape drives or certain disks. Use the following procedure to ensure that the operation completes successfully. The following prerequisites and options apply to this procedure:

- The replacement tape drive must be of the same model.
- If a SCSI tape drive fails while operating, might not be possible for the device driver to close the device. Reboot the system to reset the device before using the procedure.
- If you must shut down your system to add or replace the device (such as an internal tape drive), see the alternate procedure.

1. Verify the failed component by using the following command:

```
# /sbin/hwmgr show scsi
      SCSI          DEVICE DEVICE DRIVER NUM  DEVICE  FIRST
HWID: DEVICE- HOST- TYPE   SUB-   OWNER  PATH FILE    VALID
      ID      NAME         TYPE                      PATH
  ------------------------------------------------------------
 31:  5       rocym  tape  none    2     1   tape0  [     ]
```

   The preceding output shows a failed SCSI disk tape drive with HWID 31. The drive has no valid paths. To replace this failed tape drive with a new tape drive that has device special file name `/dev/tape/tape2`, and the same `dev_t` information, use the following procedure:

2. Install the component as described in the hardware manual.
3. Scan for the new device by using the following command:

```
# /sbin/hwmgr scan scsi | grep tape
```

4. Use the following command to find the SCSI database entry for both the failed and the replacement device:

```
# /sbin/hwmgr show scsi | grep tape
```

   You will see a new entry for the replacement device and an incomplete entry for the original (failed) device, as shown in the following sample output:

```
     SCSI             DEVICE DEVICE  DRIVER NUM  DEVICE FIRST
HWID: DEVICEID HOSTNAME TYPE   SUBTYPE OWNER  PATH FILE   VALID
                                                        PATH
------------------------------------------------------------
31:   5       rocym     tape    none    2     1   tape0  [     ]
35:   7       rocym     tape    none    0     1   tape5  [0/7/0]
```

5. Use the `hwmgr` command to delete the database entries for the failed device, specifying its hardware identifier (HWID) as follows:

```
# /sbin/hwmgr delete component -id 31
```

6. Rename the replacement device and transfer the device special files of the failed device by using the following command:

```
# /sbin/dsfmgr -m tape5 tape0
tape5=>tape0  tape5_d0=>tape0_d0  tape5_d1=>tape0_d1
tape5_d2=>tape0_d2  tape5_d3=>tape0_d3  tape5_d4=>tape0_d4
tape5_d5=>tape0c
```

If you are replacing a device that you cannot hot-swap, you must shut down the system. In such cases, modify the first two steps of the procedure as follows:

1. Insert the replacement device and boot the system. A SCSI scan runs automatically at boot time, and you will see console messages indicating that the replacement device was found and registered.
2. When the system is at single-user mode, verify that the replacement device was found by using the following command:

   ```
   # /sbin/hwmgr show scsi | grep device_type
   ```

   Replace the *device_type* variable with the type, of device such as `disk` or `tape`.

Proceed with Step 3 of the original procedure.

3.4.14 Using hwmgr to Replace a Cluster Member's Boot Disk

On a single system, the `hwmgr` command provides a `redirect` option which you use as part of the procedure to replace a failing disk. When you replace the failed disk, you use the `redirect` option to direct I/O from the failed component to the replacement device. This option redirects device special file names, cluster dev_t values, local dev_t values, logical ID, and HWID.

Only unique device identifiers (did) are accepted by the `redirect` option. In a cluster, device identifiers are not guaranteed to be unique and the command might fail as shown in the following example:

```
# /sbin/hwmgr redirect scsi -src source_did -dest target_did
# "Error (95) Cannot start operation."
```

For the redirect operation to succeed, both or neither of the hardware identifiers must exist on each member of the cluster. Use the following procedure to ensure that the `redirect` operation works:

1. Verify whether the source and destination component exist. Use the following command on each member of the cluster:

   ```
   # /sbin/hwmgr show scsi -did device_identifier
         SCSI           DEVICE DEVICE  DRIVER NUM  DEVICE FIRST
   HWID: DEVICEID HOST  TYPE   SUBTYPE OWNER  PATH FILE   VALID PATH
   32:   DID      rymoc disk   none     2     1    dsk1   [0/1/0]
   ```

2. Follow this step only if the source component exists on other cluster members but the destination component does not.

Configure the destination component on those cluster members as follows:

```
# /sbin/hwmgr scan scsi
```

Note

The bus scan is an asynchronous operation. The system prompt returns immediately but that does not mean that the scan is complete. On systems with many devices, the scan can take several minutes to complete.

3. Follow this step only if the destination component exists on other members of the system but the source component does not.

   Delete the destination component from those cluster members as follows:

   ```
   # /sbin/hwmgr delete scsi did
   ```

4. You can now use the `redirect` option to direct I/O to the replacement drive.

3.4.15 Viewing the Persistence Database for the name Subsystem

The name persistence database stores information about the hardware topology of the system. This data is maintained by the kernel and includes data for controllers and buses. Use the `show name` command option to display persistence data that you can manipulate by using other `hwmgr` commands.

The following example shows typical output from the `show name` command option on a small system:

```
# /sbin/hwmgr show name -member ychain

 HWID:  NAME    HOSTNAME   PERSIST TYPE    PERSIST AT
-----------------------------------------------------
   13:  isp0    ychain     BUS             pci0 slot 5
    4:  pci0    ychain     BUS             nexus
   14:  scsi0   ychain     CONTROLLER      isp0 slot 0
   29:  tu0     ychain     CONTROLLER      pci0 slot 11
```

The following information is provided by the output:

- `HWID:` – The unique hardware identifier for this component. You can also determine this by using the `view hierarchy` command option.
- `NAME` – The component name and the instance number, such as `pci0`, for personal computer interconnect (PCI) bus 0. Each additional PCI bus has a different instance number.

- HOSTNAME – The host on which the command is run. When working in a cluster you can specify the cluster name on which the command is to operate.
- PERSIST TYPE – The type of hardware component, which is usually a bus or a controller.
- PERSIST AT – The logical address of the component, which might map to a physical location in the hardware. For example, the SCSI controller scsi0 persists at slot 0 of the bus isp0.

3.4.16 Deleting and Removing a Component from the name Persistence Database

One of the options for manipulating the name subsystem is to remove components from the persistence database. The hwmgr command offers two methods of removal:

- remove – Use this option to take an entry out of the persistence database. This does not affect the running system but at the next reboot the component is no longer seen.
- delete – Use this option to take an entry out of the persistence database and delete it from the running system. This command unregisters and unconfigures the component, removing it from all hardware management databases.

The following example shows typical output from the show name command option on a small system. You specify the variable *name*, which is the component name shown in the output from the show name command option described in Section 3.4.15:

```
# /sbin/hwmgr show name
 HWID:  NAME    HOSTNAME  PERSIST TYPE    PERSIST AT

-----------------------------------------------------
   33:  aha0    fegin     BUS             eisa0 slot 7
   31:  ln0     fegin     CONTROLLER      eisa0 slot 5
    8:  pci0    fegin     BUS             ibus0 slot 0
   34:  scsi1   fegin     CONTROLLER      aha0 slot 0
   17:  scsi0   fegin     CONTROLLER      psiop0 slot 0
   15:  tu0     fegin     CONTROLLER      pci0 slot 0
```

Two SCSI adapters are shown in the preceding output. If scsi0 is the target of a remove operation, then scsi1 does not become scsi0. The location of the adapter persists at aha0 slot 0 and the name scsi1 is saved across boots.

To remove scsi0 and rename scsi1, use the following commands:

```
# /sbin/hwmgr remove name -entry scsi0
# /sbin/hwmgr edit name -entry scsi1 -parent_num 0
```

3.4.17 Optimizing the Hardware Databases

The number of stale paths might impact the system boot time, not because it takes any longer to read the database, but because the SCSI subsystem attempts to probe each path even if it is stale. Such stale paths can occur if you make many changes to the system's configuration, such as by moving storage to different adapters or if you remove or replace adapters. However, if you have inexplicably large numbers of stale paths on your system, it might indicate a configuration problem and you should consult your technical support representative before using the `refresh` option.

To remove stale paths, use the following option:

```
# /sbin/hwmgr refresh scsi
```

The preceding command deletes the stale paths to SCSI devices, except for any stale path that the system sees as the first path to the device.

The `refresh component` option is not strictly necessary and generally has no impact on boot but makes the output from commands easier to interpret. If you make significant hardware configuration changes, particularly when you remove and replace components, there will be many irrelevant entries in the command output. These unused entries are only visible when you display the component database by using the following command:

```
# /sbin/hwmgr show component
```

Use the following command to remove database entries for components that will never be returned to the system:

```
# /sbin/hwmgr refresh component
```

3.4.18 Renaming Components

Component names are based on the driver interfaces used, and on the instance of the device. For example, a component named `tu0` is a NIC that supports the Tulip Ethernet interface. (See `tu`(7)). A model DE500 NIC appears as a component named `ee`*N* in the name database (See `ee`(7).)

Based on its physical location, a component is assigned a persistence type and persistence location. This information is stored in the name database. The persistence type can be a network controller or a peripheral component interconnect (PCI) bus providing a series of slots. The persistence location of a component can be the logical address of a particular slot on a bus, such as slot 2 on PCI bus 1 or a main bus location (nexus).

The following procedure shows how you can assign your preferred names to components. You might want to do this to keep component names consistent across different systems so that scripts which address a specific component are easier to maintain.

Note

HP recommends that you let your system dynamically assign component names whenever possible. There are alternate procedures that enable you to preserve system customizations when updating your environment, For example, the supported method of cloning systems is the installation cloning utility. See the *Installation Guide — Advanced Topics* manual for your version of the operating system. Because component names are dynamic, consider making your local scripts and programs independent of component names. To aid you in this, the `hwmgr` and `dsfmgr` commands enable you to easily determine component names and component availability. Scripts and programs are much more dependable and portable if they do not address static names.

3.4.18.1 Identifying the Components

Use the following `hwmgr` command to view the content of the name database:

```
# /sbin/hwmgr show name
 HWID:  NAME    HOSTNAME    PERSIST TYPE    PERSIST AT
-------------------------------------------------------
   41:  ata1    rocym       BUS             pci0 slot 205
   39:  ata0    rocym       BUS             pci0 slot 105
   56:  ee2     rocym       CONTROLLER      pci3 slot 5
   19:  ee1     rocym       CONTROLLER      pci2 slot 5
   18:  ee0     rocym       CONTROLLER      pci2 slot 4
   54:  itpsa1  rocym       BUS             pci3 slot 4
```

The preceding truncated display shows three NICs in the system named `ee0`, `ee1`, and `ee2`. The display also provides the physical location of the network components, such as PCI bus 2, slot 5, for component `ee1`.

To change the names of the components to `ee3`, `ee4` and `ee5`, use the procedure in Section 3.4.18.2.

3.4.18.2 Renaming the Components

Rename the components by using the following procedure:

1. Use the following command to remove the hardware persistence entries from the name database:

   ```
   # /sbin/hwmgr remove name -entry ee0
   # /sbin/hwmgr remove name -entry ee1
   # /sbin/hwmgr remove name -entry ee2
   ```

   Repeat the command for each component that you want to rename. This action does not affect any hardware component that is currently using the removed name. It only affects the persistence of the name across reboots.

   Note

   Using the `remove` option instead of the `delete` option preserves any attribute settings that exist for the components. For example, you can define your own user name for component `ee0` by specifying a value for its `user_name` attribute. Your user-specified name (and any other customized settings) are not preserved if you use the `delete` option.

2. Shut down and reboot the system by using the following command:

   ```
   # shutdown -r now
   ```

   As the system reboots, the persistent names are recreated. The new names are assigned in the order in which the hardware is probed. For example, `ee0` is assigned to the first NIC discovered during hardware probe, `ee1` is assigned to the next, and so on until all the components are discovered.

3. Delete each component's previous name from the name database by using the following commands:

   ```
   # /sbin/hwmgr delete name -entry ee0
   # /sbin/hwmgr delete name -entry ee1
   # /sbin/hwmgr delete name -entry ee2
   ```

3.4.18.3 Verifying the Renaming Procedure

Verify the renaming procedure as follows:

1. Use the following command to show which components map to a specific PCI slot. In this case, the command searches for `ee*` components:

```
# /sbin/hwmgr show name | grep ee
.
.
.
56:  ee5     rocym           CONTROLLER      pci3 slot 5
19:  ee4     rocym           CONTROLLER      pci2 slot 5
18:  ee3     rocym           CONTROLLER      pci2 slot 4
.
.
.
```

2. Use the following command to ensure that all the incorrect names were permanently removed from the name databases:

```
# /sbin/hwmgr show components | grep ee
.
.
.
18:  rocym      r---- none    ee3
19:  rocym      r---- none    ee4
56:  rocym      r---- none    ee5
.
.
.
```

If the renaming process appears to be unsuccessful, use the troubleshooting suggestions described in the following table:

Problem	Possible Solutions
A component is not visible when you use the `hwmgr` command.	Shut down the system and use the console commands to ensure that the component is visible as part of the configuration. If the component is not visible, see the hardware documentation for component test and verification procedures.
The component is visible, but is apparently generating errors.	If the component is visible to the console, reboot the system and watch the boot messages for any component-specific errors. Such messages might be logged to the log files in the `/var/adm` directory.
The system reboots correctly, but there are post-boot component errors.	Use the Event Viewer (EVM) to view binary events (`binlogd`) by using the following command: # **sysman event_viewer** See the *System Administration* manual for your version of the operating system for more information about the event viewer and associated diagnostic tools.

Problem	Possible Solutions
After a successful reboot and renaming procedure, one or more components still has the incorrect name.	After you have ensured that the component is connected to the system and is visible to the Hardware Manager, repeat the renaming procedure.
You cannot rename the component.	Under unusual circumstances, the content of the hardware databases can become corrupt. Do not attempt to edit or refresh these databases. Contact your technical support service for assistance or perform a full installation and reapply your customizations.

3.4.19 Relocating (Moving) a Component

In Tru64 UNIX, component names are designed to be unique. Some hardware components derive their unique names based on their physical location. In particular, most buses and controllers derive their system-wide unique name from their physical location in the system. If you move such a component from one physical location in the system to another, it might appear as a new device, with a new name, when you subsequently reboot the system.

A script or program that address a specific component (such as a network card) might not find that component if its name changes. If you need to move a component within a system, you might want to preserve its original component name. The procedure described in this section shows how to preserve a component name, using a memory channel card as an example of the moved component.

3.4.19.1 Finding the Component Name

Many component names are based on the driver interfaces used by the device, and the instance of the device. For example, a component named `mchan0` is a Memory Channel card, instance `0`. Based on the physical location of the device, components are assigned a persistence type and persistence location. This information is stored in the name database.

Use the following `hwmgr` command to view the content of the name database:

```
# /sbin/hwmgr show component
HWID:  NAME    HOSTNAME  PERSIST TYPE    PERSIST AT
  62:  ata0    host12    BUS             pci0 slot 15
  45:  emx2    host12    BUS             pci0 slot 1
 228:  mchan0  host12    CONTROLLER      pci1 slot 9
.
.
```

.

A component might be installed in a bus, such as a peripheral component interconnect (PCI) bus that provides a series of slots. In the preceding output, the PERSIST AT field is comprised of:

- *parent_nam* – The logical name of the parent component, such as the PCI bus pci0
- *parent_num* – The physical address at the parent location, such as slot 15 on PCI bus 1

To display the topology of subcomponents under a particular component, use the hwmgr view hierarchy command, specifying a component by its hardware identifier, as follows:

```
# /sbin/hwmgr view hierarchy -id 610
HWID:   hardware hierarchy
-------------------------------------------------------
 610:   bus pci2
 611:     connection pci2slot0
 617:       scsi_adapter itpsa0
 618:         scsi_bus scsi0
 674:           disk bus-0-targ-5-lun-0 cdrom51
 613:     connection pci2slot1
 619:       scsi_adapter itpsa1
 620:         scsi_bus scsi1
 208:           disk bus-1-targ-0-lun-0 dsk1188
 209:           disk bus-1-targ-1-lun-0 dsk1189
 210:           disk bus-1-targ-2-lun-0 dsk1190
 615:     connection pci2slot2
 621:       network tu0
```

3.4.19.2 Relocating the Component

Relocate the component as follows:

1. Use the following command to find the current bus and slot location of the component that you intend to relocate. This example specifies a memory channel card (mchan):

   ```
   # /sbin/hwmgr show name | mchan
   228:  mchan0  host12            CONTROLLER      pci1 slot 9
   ```

2. Modify the hardware name database, specifying the new location for the component as follows:

   ```
   # /sbin/hwmgr edit name -entry mchan0 parent_num 5 slot 8
   ```

   This command relocates the component mchan0 from its present location on PCI bus 1 (pci1), slot 9 to a new location at PCI bus 5 (pci5), slot 8.

Note

If you cannot obtain the name and location pairing that you want, you might need to delete existing names in the database by using the `hwmgr remove name` command or `hwmgr delete entry` command. However, take care not to remove required entries for existing (unmoved) components.

3. Shut down the system as follows:

```
# /usr/sbin/shutdown now
```

4. Physically relocate the component to its new slot.
5. Reboot the system to single-user mode as follows:

```
>>>boot -flags s
```

6. Verify that the component is found at the correct location:

```
# /sbin/hwmgr show name | mchan
228:  mchan0  host12             CONTROLLER      pci5 slot 8
```

7. Boot the system to multiuser mode as follows:

```
# Ctrl/D
228:  mchan0  host12             CONTROLLER      pci5 slot 8
```

Problem	Possible Solutions
A component is not visible when you use the `hwmgr` command. Most likely, this problem is caused by a failure to properly install the component in its new location.	Shut down the system and use the console (>>>) commands to ensure that the component is visible as part of the configuration. If it is visible, repeat the relocation process. If the component is not visible, see its hardware documentation for component test and verification procedures.
The component is visible, but is apparently generating errors.	If the component is visible to the console, reboot the system and watch the boot messages for any component-specific errors. Such messages might be logged to the log files in the `/var/adm` directory.

Problem	Possible Solutions
The system reboots correctly, but there are post-boot component errors.	Use the Event Viewer (EVM) to view binary events (`binlogd`) by using the following command: # **`sysman event_viewer`** See the *System Administration* manual for your version of the operating system for more information about the event viewer and associated diagnostic tools.
After a successful reboot and relocation procedure, one or more devices still has the incorrect name.	Verify that you inserted the component in the correct new location. After you have ensured that the component is connected to the system and is visible to the Hardware Manager, repeat the renaming procedure.

4

Dynamic Device Recognition (DDR)

Most device management is automatic. A device added to a system is recognized, mapped, and added to the device databases as described in Chapter 3. However, you might sometimes need to add devices that the system cannot detect and add to the system automatically. These devices might be old, or new prototypes, or they might not adhere closely to supported standards such as SCSI. In these cases, you must manually configure the device and its drivers in the kernel, by using the `ddr_config` command described in this chapter.

This chapter discusses the following topics:

- An overview of the kernel reconfiguration process (Section 4.1)
- Using the dynamic method to reconfigure the kernel (Section 4.2)
- Manually changing the DDR database (Section 4.3)
- Converting customized Common Access Method (CAM) Data (Section 4.4)
- How to create pseudoterminals (ptys), which are terminal pseudodevices that enable remote logins (Section 4.5)

4.1 Understanding Kernel Reconfiguration

You use two processes to reconfigure and rebuild a kernel:

- The dynamic method uses the `ddr` command to rebuild the kernel and implement the disk configuration changes without shutting down the operating system.
- The static method requires that you use the `MAKEDEV` and `config` commands. You must also shut down the system and restart it to rebuild the kernel and implement the changes.

  Use the `MAKEDEV` command or the `mknod` command to create device special files instead of using the `dsfmgr` command. The `kmknod` command creates device special files for third-party kernel layered products. See `MAKEDEV`(8), `mknod`(8), and `kmknod`(8) for more information.

  For loadable drivers, the `sysconfig` command creates the device special files by using the information specified in the driver's stanza entry in the `/etc/sysconfigtab` database file.

4.2 Using the Dynamic Method

The following sections explain how to use the `ddr_config` command to manage the Dynamic Device Recognition (DDR) database for your system. These sections introduce DDR, then describe how you use the `ddr_config` command to:

- Add SCSI devices to the DDR database
- Convert a customized `cam_data.c` file

4.2.1 Understanding Dynamic Device Recognition

DDR is a framework for describing the operating parameters and characteristics of SCSI devices to the SCSI CAM I/O subsystem. You can use DDR to add new and changed SCSI devices to your system without rebooting the operating system. You do not disrupt user services and processes, as happens with static methods of device recognition.

DDR is preferred over the static method for recognizing SCSI devices. The current, static method is to edit the `/sys/data/cam_data.c` data file and include custom SCSI device information, reconfigure the kernel, and shut down and reboot the operating system.

Note

Support for the static method of recognizing SCSI devices will be retired in a future release.

You can use both methods on the same system, with the restriction that the devices described by each method are exclusive to that method (nothing is doubly-defined).

The information DDR provides about SCSI devices is needed by SCSI drivers. You can supply this information by using DDR when you add new SCSI devices to the system, or you can use the `/sys/data/cam_data.c` data file and static configuration methods. The information provided by DDR and the `cam_data.c` file have the same objectives. When compared to the static method of providing SCSI device information, DDR minimizes the amount of information that is supplied by the device driver or subsystem to the operating system and maximizes the amount of information that is supplied by the device itself or by defaults specified in the DDR databases.

4.2.1.1 Conforming to Standards

Devices you add to the system should minimally conform to the SCSI-2 standard, as specified in *SCSI-2, Small Computer System Interface-2 (X3.131-1994)*. If your devices do not comply with the standard, or if

they require exceptions from the standard, you store information about these differences in the DDR database. If the devices comply with the SCSI-2 standard, you do not need to modify the database. The system will automatically recognize such devices and you can configure them by using the `hwmgr` command.

4.2.1.2 Understanding DDR Messages

The following list indicates the most common DDR message categories and the action, if any, that you should take:

- Console messages are displayed during the boot sequence. Frequently, these messages indicate that the kernel cannot read the DDR database. This error occurs when the system's firmware is not at the proper revision level. Upgrade to the correct revision level of the firmware.
- Console messages warn about corrupted entries in the database. Recompile and regenerate the database.
- Runtime messages generally indicate syntax errors that are produced by the `ddr_config` compiler. The compiler runs when you use the –`c` option to the `ddr_config` command and does not produce an output database until all syntax errors are corrected.

Use the –`h` option to the `ddr_config` command to display help on command options.

4.2.2 Sample Database Entries

This section provides two examples of typical entries in the `/etc/ddr.dbase file` that support third-party products. This information will be supplied to you by the vendor of the supported device. Consult the vendors documentation for important information about setting the correct parameters for your installed version of HP Tru64 UNIX.

Caution

Do not use the sample entries included in this section. Contact the vendor of a device to obtain the latest configuration settings. HP does not provide configuration information for third-party devices other than for those listed in the system default `/etc/ddr.dbase` file.

4.2.2.1 Sample DDR Entry for a StorageTek 9840

The following sample DDR entry supports the StorageTek 9840 magnetic tape device. For information about the parameters, mode select, density, and

attribute fields, see `ddr.dbase`(4). The hardware documentation supplied with the device must provide documentation about the attribute settings.

```
# Please note that this entry is not supported by HP
# It is supplied by StorageTek as a means of enabling
# their tape drive to work in HP Tru64 UNIX.
# If you experience problems when using this drive
# contact StorageTek Technical Support.

SCSIDEVICE
  #
  # STK 9840
  #
  Type = tape
  Name = "STK" "9840"
  #
 PARAMETERS:
  TypeSubClass   = 3480
  MaxTransferSize  = 0x40000     #256k
  SyncTransfers  = enabled
  WideTransfers  = enabled
  Disconnects   = enabled
  CmdReordering  = disabled
  TaggedQueuing  = disabled
  TagQueueDepth  = 0
  WCE_Capable   = false
  PwrMgmt_Capable  = false
  LongTimeoutRetry  = disabled
  ReadyTimeSeconds  = 240
  DisperseQueue  = false
  CMD_PreventAllow = supported
  CMD_ExtReserveRelease = supported

 DENSITY:
 #
 # /dev/tape/tapeX_d0, _d4
 #
 DensityNumber = 0,4
 DensityCode = density_code_42
 CompressionCode = 0
 Buffered = 0x1
 #
 DENSITY:
 #
 # /dev/tape/tapeX_d1, _d5
 #
 DensityNumber = 1,5
 DensityCode = density_code_42
 CompressionCode = 1
 Buffered = 0x1
 #
 #
 # /dev/tape/tapeX_d2, _d6
 #
 DENSITY:
 DensityNumber = 2,6
 DensityCode = density_code_43
 CompressionCode = 0
 Buffered = 0x1
 #
 #
 # /dev/tape/tapeX_d3, _d7
 #
 DENSITY:
```

```
DensityNumber = 3,7
DensityCode = density_code_43
CompressionCode = 1
Buffered = 0x1


        Buffered = 0x1

    DENSITY:
        #
        DensityNumber = 1
        DensityCode = default
        CompressionCode = 0x1
        Buffered = 0x1
```

4.2.2.2 Sample DDR Entries for EMC Symmetrix

The following sample DDR entry supports the EMC Symmetrix disk storage array and Fibre Channel devices. For a description of the parameter, mode select, density, and attribute fields, see `ddr.dbase`(4).

The hardware documentation supplied with any third-party device must provide documentation about the attribute settings. In this example, the `ATTRIBUTE` settings include comments defining potential settings for the value of the `ubyte[0]` attribute. Consult the device's hardware documentation or contact the vendor for information about the settings.

```
SCSIDEVICE
#
# SAMPLE Entry for Symmetrix SCSI devices (DO NOT USE, CONSULT EMC)
#
Type = disk
Name = "EMC" "SYMMETRIX"
PARAMETERS:
TypeSubClass = hard_disk, raid
BlockSize = 512
BadBlockRecovery = disabled
DynamicGeometry = true
LongTimeoutRetry = enabled
DisperseQueue = false
TagQueueDepth = 20
ReadyTimeSeconds = 45
InquiryLength = 160
RequestSenseLength = 160
PwrMgmt_Capable = false
ATTRIBUTE:
# ubyte[0] = 8 Disable AWRE/ARRE only, PR enabled
# ubyte[0] = 25 Disable PR & AWRE/ARRE, Enable I/O Barrier Patch resets
AttributeName = "DSBLflags"
Length = 4
ubyte[0] = 8

SCSIDEVICE
#
# Entry for Symmetrix Fibre Channel devices
#
Type = disk
Stype = 2
Name = "EMC" "SYMMETRIX"
PARAMETERS:
TypeSubClass = hard_disk, raid
```

```
BlockSize = 512
BadBlockRecovery = disabled
DynamicGeometry = true
LongTimeoutRetry = enabled
DisperseQueue = false
TagQueueDepth = 20
ReadyTimeSeconds = 45
InquiryLength = 160
RequestSenseLength = 160
PwrMgmt_Capable = false
ATTRIBUTE:
AttributeName = "DSBLflags"
Length = 4
ubyte[0] = 8
```

4.3 Manually Changing the DDR Database

When you make a change to the operating parameters or characteristics of a SCSI device, you must describe the changes in the `/etc/ddr.dbase` file. You must compile the changes by using the `ddr_config -c` command.

Two common reasons for changes are:

- Your device deviates from the SCSI standard or reports something different from the SCSI standard.
- You want to optimize device defaults, most commonly the `TagQueueDepth` parameter, which specifies the maximum number of active tagged requests the device supports.

You use the `ddr_config -c` command to compile the `/etc/ddr.dbase` file and produce a binary database file, `/etc/ddr.db`. When the kernel is notified that the file's state has changed, it loads the new `/etc/ddr.dbase` file. In this way, the SCSI CAM I/O subsystem is dynamically updated with the changes that you made in the `/etc/ddr.dbase` file and the contents of the on-disk database are synchronized with the contents of the in-memory database.

Use the following procedure to compile the `/etc/ddr.dbase` database:

1. Log in as root or become the superuser.
2. Enter the `ddr_config -c` command, for example:

   ```
   # /sbin/ddr_config -c
   ```

There is no message confirming successful completion. When the prompt is displayed, the compilation is complete. Any syntax errors are displayed as standard output and no output file is compiled.

4.4 Converting Customized cam_data.c Information

You use the following procedure to transfer customized information about your SCSI devices from the `/sys/data/cam_data.c` file to the

/etc/ddr.dbase text database. In this example, *MACHINE* is the name of your machine's system configuration file.

1. Log in as root or become the superuser.
2. To produce a summary of the additions and modifications that you should make to your /etc/ddr.dbase file, enter the ddr_config -x command. For example:

   ```
   # /sbin/ddr_config -x MACHINE > output.file
   ```

   This command uses as input the system configuration file. (You specify the configuration file when you build your running kernel.) The procedure runs in multiuser mode and requires no input after it starts. Redirect output to a file to save the summary information. Compile errors are reported to standard error and the command terminates when the error is reported. Warnings are reported to standard error and do not terminate the command.
3. Edit the characteristics that are listed on the output file into the /etc/ddr.dbase file, following the syntax requirements of that file. Instructions for editing the /etc/ddr.dbase database are found in ddr.dbase(4).
4. Enter the ddr_config -c command to compile the changes.

See Section 4.3 for more information.

You can add pseudodevices, disks, and tapes statically (without using DDR) by using the methods described in the following sections.

4.5 Adding Pseudoterminals and Devices Without Using DDR

System V Release 4 (SVR4) pseudoterminals (ptys) are implemented by default and are defined as follows:

/dev/pts/N

The variable *N* is a number from 0-9999.

This implementation allows for more scalability than the BSD ptys (tty[a-zA-Z][0-9a-zA-Z]). The base system commands and utilities support both SVR4 and BSD ptys. To revert back to the original default behavior, create the BSD ptys by using the MAKEDEV command. See also SYSV_PTY(8), pty(7), and MAKEDEV(8).

4.5.1 Adding Pseudoterminals

Pseudoterminals enable users to use the network to access a system. A pseudoterminal is a pair of character devices that emulate a hardware

terminal connection to the system. Instead of hardware, however, there is a master device and a slave device. Pseudoterminals, unlike terminals, have no corresponding physical terminal port on the system. Remote login sessions, window-based software, and shells use pseudoterminals for access to a system. By default, SVR4 device special files such as `/dev/pts/`*n* are created. You must use `/dev/MAKEDEV` to create BSD pseudoterminals such as `/dev/ttyp/`*n*. Two implementations of pseudoterminals are offered: BSD STREAMS and BSD `clist`.

For some installations, the default number of `pty` devices is adequate. However, as your user community grows, and each user wants to run multiple sessions of one or more timesharing machines in your environment, the machines might run out of available `pty` lines. The following command enables you to review the current value:

```
# sysconfig -q pts
pts:
nptys = 255
```

You can dynamically change the value with the `sysconfig` command, although this change is not preserved across reboots:

```
# sysconfig -r pts nptys=400
```

To modify the value and preserve it across reboots, use the following procedure:

1. Log in as superuser (root).
2. Add or edit the pseudodevice entry in the system configuration file `/etc/sysconfigtab`. By default, the kernel supports 255 pseudoterminals. If you add more pseudoterminals to your system, you must edit the system configuration file entry and increment the number 255 by the number of pseudoterminals you want to add. The following example shows how to add 400 pseudoterminals:

   ```
   pts:
   nptys=400
   ```

   The pseudodevice entry for `clist`-based pseudoterminals is as follows:

   ```
   pseudo-device pty 655
   ```

   For more information on the configuration file and its pseudodevice keywords, refer to the configuration information in the *System Administration* manual.
3. For `clist`-based pseudoterminals, you also need to rebuild and boot the new kernel. Use the information on rebuilding and booting the new kernel in the *System Administration* manual.

   When the system is first installed, the configuration file contains a pseudodevice entry with the default number of 255 pseudoterminals. If

for some reason the number is deleted and not replaced with another number, the system defaults to supporting the minimum value of 80 pseudoterminals. The maximum value is 131072.

If you want to create BSD terminals, use the /dev/MAKEDEV command as follows:

1. Log in as root and change to the /dev directory.
2. Create the device special files by using the MAKEDEV command, as follows:

   ./MAKEDEV pty_#

   The number sign (#) represents the set of pseudoterminals (0 to 101) you want to create. The first 51 sets (0 to 50) create 16 pseudoterminals for each set. The last 51 sets (51 to 101) create 46 pseudoterminals for each set. See MAKEDEV(8) for instructions on making a large number of pseudoterminals. (See the Tru64 UNIX *QuickSpecs* for the maximum number of supported pseudoterminals.)

   Note

   By default, the installation software creates device special files for the first two sets of pseudoterminals, pty0 and pty1. The pty0 pseudoterminals have corresponding device special files named /dev/ttyp0 through /dev/ttypf. The pty1 pseudoterminals have corresponding device special files named /dev/ttyq0 through /dev/ttyqf.

   If you add pseudoterminals to your system, the pty# variable must be higher than pty1 because the installation software sets pty0 and pty1. For example, to create device special files for a third set of pseudoterminals, enter:

   ```
   # ./MAKEDEV pty2
   ```

   The MAKEDEV command lists the device special files it has created. For example:

   ```
   MAKEDEV: special file(s) for pty2:
   ptyr0 ttyr0 ptyr1 ttyr1 ptyr2 ttyr2 ptyr3 ttyr3 ptyr4 ttyr4
   ptyr5 ttyr5 ptyr6 ttyr6 ptyr7 ttyr7 ptyr8 ttyr8 ptyr9 ttyr9
   ptyra ttyra ptyrb ttyrb ptyrc ttyrc ptyrd ttyrd ptyre ttyre
   ptyrf ttyrf
   ```

3. To remove BSD ptys, use the /dev/SYSV_PTY command.
4. If you want to allow root logins on all pseudoterminals, make sure an entry for ptys is present in the /etc/securettys file. If you do not want to allow root logins on pseudoterminals, delete the entry for ptys

from the /etc/securettys file. For example, to add the entries for the new tty lines and to allow root login on all pseudoterminals, enter the following lines in the /etc/securettys file:

```
/dev/tty08     # direct tty
/dev/tty09     # direct tty
/dev/tty10     # direct tty
/dev/tty11     # direct tty
ptys
```

See securettys(4) for more information.

4.5.2 Adding Other Devices

When you add new SCSI devices to your system, the system automatically detects and configures them. It runs the appropriate hwmgr and dsfmgr commands to register the devices, assign identifiers, and create device special files. However, you might need to manually create device names for other devices by using the MAKEDEV command. You might also need to recreate device special files that you incorrectly deleted from the system.

For new devices, you must physically connect the devices and then make the devices known to the system. There are two methods, one for static drivers and another for loadable drivers. Before adding a device, read the owner's manual that came with your system processor and any documentation that came with the device itself. You might also require a disk containing the driver software.

Chapter 8 provides an outline example of adding a PCMCIA modem to a system, and shows you how to create the device special files.

You do not need to use the MAKEDEV command if you want only to create legacy rz or tz device special files in /dev such as /dev/rz5. The dsfmgr command provides a method of creating these device names. To add a device for a loadable driver, see the device driver documentation.

To add a device for a static driver, see Section 4.5.1.

Next, make the device special files for the device, by following these steps:

1. Change to the /dev directory.
2. Create the device special files by using the MAKEDEV command as follows:

   ```
   # ./MAKEDEV deviceN
   ```

   The *device* variable is the device mnemonic for the drive you are adding. The variable (*N*) is the number of the device. For example, to create the device special files for two PCMCIA modem cards, use the following command:

```
# ./MAKEDEV ace2 ace3

MAKEDEV: special file(s) for ace2:
tty02
MAKEDEV: special file(s) for ace3:
tty03
```

The generated special files should look like this:

```
crw-rw-rw-   1 root     system    35,  2 Oct 27 14:02 tty02
crw-rw-rw-   1 root     system    35,  3 Oct 27 14:02 tty03
```

3. Stop system activity by using the `shutdown` command and then turn off the processor. See the *System Administration* manual and `shutdown`(8) for more information.
4. Power up the machine. To ensure that all the devices are seen by the system, power up the peripherals before powering up the system box.
5. Boot the system with the new kernel. See the *System Administration* manual for information on booting your processor.

5

Using Device Commands and Utilities

The preceding chapters described generic hardware management tools that you use to manage many aspects of all devices, such as the `hwmgr` command described in Chapter 3. The following sections describe hardware management tools that are targeted at a particular kind of device and perform a specific task:

- Finding device utilities (Section 5.1)
- Using SCSI utilities (Section 5.2)
- Partitioning (formatting) disks by using the `diskconfig` GUI (Section 5.3)
- Partitioning (formatting) disks by using the `disklabel` command (Section 5.4)
- Copying disks (Section 5.5)
- Monitoring disks (Section 5.6)

5.1 Finding Device Utilities

Many of the device utilities are documented elsewhere in this manual or at other locations in the documentation set. For example, utilities that enable you to configure network devices are documented in detail in the *Network Administration: Services* manual. Table 5–1 provides references to utilities, including those listed in this chapter. Other utilities are documented in reference pages. Table 5–2 provides references to utilities documented in the reference pages and also provides pointers to reference data such as the Section 7 interface reference pages.

Table 5–1: Device Utilities Documented in the Manuals

Device	Task	Location
Processor	Starting or Stopping	*System Administration*
	Sharing Resources	*System Configuration and Tuning*
	Monitoring	*System Administration*
	Power Management	*System Administration*, `dxpower`

Table 5–1: Device Utilities Documented in the Manuals (cont.)

Device	Task	Location
	Testing Memory	*System Administration*
	Error and Event Handling	*System Administration*
SCSI buses	Advanced Configuration and Management	Section 5.2.1, `scu`
Disks	Partitioning	`diskconfig`, `disklabel`
	Copying	Section 5.5, `dd`
	Monitoring Usage	Section 5.6, `df` and `du`
	Power Management	*System Administration*
	File Systems Status	*System Administration*
	Testing and Exercising	*System Administration*
Tapes (and disks)	Archiving	*System Administration*
	Testing and Exercising	*System Administration*
Clock	Setting	*System Administration*
Modem	Configuring	*System Administration*

Table 5–2: Device Utilities Documented in the Reference Pages

Device	Task	Location
Devices (General)	Configuring	`hwmgr`(8), `devswmgr`(8), `dsfmgr`(8)
	Device Special Files	`kmknod`(8) , `mknod`(8), `MAKEDEV`(8), `dsfmgr`(8)
	Interfaces	`atapi_ide`(7), `devio`(7), `emx`(7)
Processor	Starting and Stopping	`halt`(8), `psradm`(8), `reboot`(2).
	Allocating CPU Resources	`class_scheduling`(4), `processor_sets`(4), `runon`(1)
	Monitoring	`dxsysinfo`(8), `psrinfo`(1).
SCSI buses	Managing	`sys_attrs_cam`(5), `ddr.dbase`(4), `ddr_config`(8)

Table 5–2: Device Utilities Documented in the Reference Pages (cont.)

Device	Task	Location
Disks	Partitioning	diskconfig(8), disklabel(4), disklabel(8), disktab(4)
	Monitoring	dxsysinfo(8) , diskusg(8), acctdisk(8), df(1), du(1), quota(1)
	Testing and Maintenance	diskx(8), zeero(8)
	Interfaces	ra(7), radisk(8), ri(7), rz(7)
	Swap Space	swapon(8)
Tapes (and Disks)	Archiving	bttape(8), dxarchiver(8), rmt(8)
	Testing and Maintenance	tapex(8)
	Interfaces	tz(7), mtio(7), tms(7)
Floppy	Tools	dxmtools(1), mtools(1)
	Testing and Maintenance	fddisk(8)
	Interfaces	fd(7)
Terminals, ports	Interfaces	ports(7)
Modem	Configuring	chat(8)
	Interfaces	modem(7)
Keyboard, mouse	Interfaces	dc(7), scc(7)

See the *System Administration* manual for a list of the utilities provided by the SysMan graphical user interface.

5.2 SCSI and Device Driver Utilities

The following sections describe utilities that you use to manage SCSI devices and device drivers.

5.2.1 Using the SCSI Configuration Utility, scu

The SCSI/CAM Utility Program, scu, provides commands for advanced maintenance and diagnostics of SCSI peripheral devices and the CAM I/O subsystem. For most daily operations, you use the hwmgr command. The scu program has an extensive help feature that describes its options and conventions. See scu(8) for detailed information on using this command.

You can use scu to:

- Format disks
- Reassign a defective disk block
- Reserve and release a device
- Display and set device and program parameters
- Enable and disable a device

Note

For Digital Storage Architecture (DSA) disks, use the radisk program. See radisk(8) for information.

Examples of scu usage are:

```
# scu
scu> set nexus bus 0 target 0 LUN 0
Device: RZ1CB-CA, Bus: 0, Target: 0, LUN: 0, Type: Direct Access
scu> show capacity

Disk Capacity Information:

                  Maximum Capacity: 8380080 (4091.836 megabytes)
                      Block Length: 512
scu> show scsi status 0
SCSI Status = 0 = SCSI_STAT_GOOD = Command successfully completed
```

5.2.2 Using the Device Switch Manager, devswmgr

The devswmgr command enables you to manage the device switch table by displaying information about the device drivers in the table. You can also use the command to release device switch table entries. Typically, you release the entries for a driver after you have unloaded the driver and do not plan to reload it later. Releasing the entries frees them for use by other device drivers.

Examples of devswmgr usage for device data are:

```
# devswmgr -display
device switch database read from primary file
  device switch table has 200 entries
# devswmgr -getnum

Device switch reservation list
                         (*=entry in use)
  driver name            instance   major
----------------------   --------   -----
                    pfm          1      71*
```

```
            fdi          2      58*
            xcr          2      57
           kevm          1      56*
       cam_disk          2      55*
            emx          1      54
          TMSCP          2      53
           MSCP          2      52
            xcr          1      44
            LSM          4      43
            LSM          3      42
            LSM          2      41*
            LSM          1      40*
            ace          1      35*
parallel_port            1      34*
       cam_uagt          1      30
           MSCP          1      28
          TMSCP          1      27
            scc          1      24
         presto          1      22
        cluster          2      21*
        cluster          1      19*
            fdi          1      14*
       cam_tape          1       9
       cam_disk          1       8*
            pty          2       7
            pty          1       6
            tty          1       1
        console          1       0
```

5.3 Partitioning Disks Using diskconfig

The Disk Configuration graphical user interface (`diskconfig`) enables you to perform the following tasks:

- Display attribute information for existing disks
- Modify disk configuration attributes
- Administer disk partitions
- Create AdvFS and UFS file systems on a disk partition
- Administer disk aliases

See `diskconfig`(8) for information on invoking the Disk Configuration GUI (`diskconfig`). An online help volume describes how you use the graphical interface. See `disklabel`(8) for information on command options.

The Disk Configuration GUI provides a graphical interface to several disk maintenance tasks that you can perform manually, by using the following commands:

- `disklabel` – Use this command to install, examine, or modify the label on a disk drive or pack. The disk label contains information about the disk, such as type, physical parameters, and partitioning. See also the `/etc/disktab` file, described in `disklabel`(4).
- `newfs` – This command creates a new UFS file system on the specified device. You cannot use the `newfs` command to create Advanced File System (AdvFS) domains. Instead, use the `mkfdmn` command, as described in `mkfdmn`(8).
- `mkfdmn` and `mkfset` – Use these commands to create Advanced File System (AdvFS) domains and filesets.

Invoke the Disk Configuration interface in either of the following ways:

- At the system prompt, enter **`diskconfig`**.
- From the CDE Front Panel, SysMan Applications pop-up menu, choose Configuration. Then select the Disk icon from the SysMan Configuration folder.

Caution

Disk Configuration displays appropriate warnings when you attempt to change partition sizes. Plan the changes in advance to ensure that you do not overwrite any required data. Back up any data partitions before attempting this task.

The system displays a window titled Disk Configuration on *hostname*. This main window for the DiskConfig utility lists the following information for each disk:

- The disk basename, such as `dsk10`. See Section 1.5 for information on disk names.
- The device model, such as `RZ1CB-CA`.
- The physical location of the device, specifying Bus, Target and LUN (logical unit number). See Chapter 3 for information on the device location.

When you select a device by doubleclicking on the list item (or click on the configure button when a disk is highlighted), the two windows are displayed at the same time. The windows are displayed overlapped. Move the cursor to the appropriate window to configure partitions or edit the disk's partition table as follows:

Disk Configuration: Configure Partitions: *`device name device type`*

This window provides the following information and options:

- A graphical representation of the disk partitions, in a horizontal bar-chart format. The currently-highlighted partition is a different color, and the details of that partition are displayed in the Selected Partition box. You can use the bar chart handles (or flags) to change the partition sizes. Position the cursor as follows:
  - On the center handle to change both adjacent partitions
  - On the top flag to move up the start of the right hand partition
  - On the bottom flag to move down the end of the left-hand partition

  Press MB1 and drag the mouse to move the handles.
- A pull-down menu that enables you to choose how the utility displays disk size information. You can choose megabytes, bytes, or blocks.
- A statistics box, which displays disk information such as the device name, the total size of the disk, and usage information. This box enables you to assign or edit the disk label, and create an alias name for the device.
- The Selected Partition box, which displays dynamic sizes for the selected partition. These sizes are updated as you change the partitions by using the bar chart. You can also type the partition sizes directly into these windows to override the current settings. This box also enables you to select the file system for the partition and, if using AdvFS, the domain name and fileset name.
- The Disk Attributes... option.

  This button displays some of the physical attributes of the device.
- The Partition Table... option, which displays the window titled Disk Configuration: Partition Table.

Disk Configuration: Partition Table: *device name device type*

This window displays a bar chart of the current partitions in use, their sizes, and the file system in use. You can choose from the following:

- The partitions that are currently displayed
- The default table for the disk device
- Reset the partitions to their original settings when this session started (the on-disk partition table).

If you make errors on a manual partition change, you can use this window to reset the partition table.

Refer to the online help for more information on these windows.

After making partition adjustments, use the SysMan Menu options to mount any newly created file systems as follows:

1. Invoke the SysMan Menu.
2. Expand the Storage options, and select Basic File System Utilities - Mount File Systems.
3. In the Mount Operation window, select the option to mount a specific file system and click on the Next button.
4. In the Name and Mount Point window:
   a. Type a mount point, such as `/usr/newusers`
   b. Type the partition name, such as `/dev/disk/dsk0g` or a domain name, such as `newusr_domain#usr`.
5. Click on the OK button to implement the partition adjustment.

Your new file system is now accessible.

5.4 Manually Partitioning Disks

This section provides the information you need to change the partition scheme of your disks. In general, you allocate disk space during the initial installation or when adding disks to your configuration. Usually, you do not have to alter partitions; however, there are cases when it is necessary to change the partitions on your disks to accommodate changes and to improve system performance.

The disk label provides detailed information about the geometry of the disk and the partitions into which the disk is divided. You can change the label with the `disklabel` command. You must be the root user to use the `disklabel` command.

There are two copies of a disk label, one located on the disk and one located in system memory. Because it is faster to access system memory than to perform I/O, when the system boots, it copies the disk label into memory. Use the `disklabel –r` command to directly access the label on the disk instead of going through the in-memory label.

Caution

Before you change disk partitions, back up all the file systems if there is any data on the disk. Changing a partition overwrites the data on the old file system, destroying the data.

When changing partitions, remember that:

- You cannot change the offset, which is the beginning sector, or shrink any partition on a mounted file system or on a file system that has an open file descriptor.
- If you need a single partition on the entire disk, use partition `c`.
- Unless it is mounted, you must specify the raw device for partition `a`, which begins at the start of the disk (sector 0), when you change the label. If partition `a` is mounted, you must then use partition `c` to change the label. Partition `c` must also begin at sector 0.

Caution

If partition `a` is mounted and you attempt to edit the disk label using device partition `a`, you cannot change the label. Furthermore, you do not receive any error messages indicating that the label is not written.

Before changing the size of a disk partition, review the current partition setup by viewing the disk label. The `disklabel` command allows you to view the partition sizes. The bottom, top, and size of the partitions are in 512-byte sectors.

To review the current disk partition setup, use the following `disklabel` command:

`/sbin/disklabel -r` *device*

Specify the device with its directory name (`/dev`) followed by the raw device name, drive number, and partition `a` or `c`. You can also specify the disk unit and number, such as `dsk1`.

An example of using the `disklabel` command to view a disk label follows:

```
# disklabel –r /dev/rdisk/dsk3a
type: SCSI
disk: rz26
label:
flags:
bytes/sector: 512
sectors/track: 57
tracks/cylinder: 14
sectors/cylinder: 798
cylinders: 2570
rpm: 3600
interleave: 1
trackskew: 0
cylinderskew: 0
headswitch: 0           # milliseconds
track-to-track seek: 0  # milliseconds
```

```
drivedata: 0

8 partitions:
#       size offset   fstype [fsize bsize cpg]
 a:  131072       0   4.2BSD   1024  8192  16  # (Cyl.    0 - 164*)
 b:  262144  131072   unused   1024  8192      # (Cyl.  164*- 492*)
 c: 2050860       0   unused   1024  8192      # (Cyl.    0 - 2569)
 d:  552548  393216   unused   1024  8192      # (Cyl.  492*- 1185*)
 e:  552548  945764   unused   1024  8192      # (Cyl. 1185*- 1877*)
 f:  552548 1498312   unused   1024  8192      # (Cyl. 1877*- 2569*)
 g:  819200  393216   unused   1024  8192      # (Cyl.  492*- 1519*)
 h:  838444 1212416   4.2BSD   1024  8192 16   # (Cyl. 1519*- 2569*)
```

Take care when you change partitions because you can overwrite data on the file systems or make the system inefficient. If the partition label becomes corrupted while you are changing the partition sizes, you can return to the default partition label by using the `disklabel` command with the `–w` option, as follows:

```
# disklabel -r -w /dev/rdisk/dsk1a rz26
```

The `disklabel` command allows you to change the partition label of an individual disk without rebuilding the kernel and rebooting the system. Use the following procedure:

1. Display disk space information about the file systems by using the `df` command.
2. View the `/etc/fstab` file to determine if any file systems are designated as swap space.
3. Examine the disk's label by using the `disklabel` command with the `–r` option. (See `rz`(7), `ra`(7), and `disktab`(4) for information on the default disk partitions.)
4. Back up the file systems.
5. Unmount the file systems on the disk whose label you want to change.
6. Calculate the new partition parameters. You can increase or decrease the size of a partition. You can also cause partitions to overlap.
7. Edit the disk label by using the `disklabel` command with the `–e` option to change the partition parameters, as follows:

   ```
   # /sbin/disklabel -e disk
   ```

   An editor, either the `vi` editor or that specified by the EDITOR environment variable, is invoked so you can edit the disk label, which is in the format displayed with the `disklabel –r` command.

   The `–r` option writes the label directly to the disk and updates the system's in-memory copy, if possible. The *disk* parameter specifies the unmounted disk (for example, `dsk0` or `/dev/rdisk/dsk0a`).

   After you quit the editor and save the changes, the following prompt is displayed:

```
write new label? [?]:
```

Enter y to write the new label or n to discard the changes.

8. Use the disklabel command with the –r option to view the new disk label.

5.4.1 Finding Overlapping Partitions

Commands to mount or create file systems, add a new swap device, and add disks to the Logical Storage Manager first check whether the disk partition specified in the command already contains valid data, and whether it overlaps with a partition that is already marked for use. The fstype field of the disk label enables you to determine when a partition or an overlapping partition is in use.

If the partition is not in use, the command continues to execute. In addition to mounting or creating file systems, commands such as mount, newfs, fsck, voldisk, mkfdmn, rmfdmn, and swapon also modify the disk label, so that the fstype field specifies partition usage. For example, when you add a disk partition to an AdvFS domain, the fstype field is set to AdvFS.

If the partition is not available, these commands return an error message and prompt you to continue or abort the operation, as shown in the following example:

```
# newfs /dev/disk/dsk8c
WARNING: disklabel reports that basename,partition currently
is being used as "4.2BSD" data. Do you want to
continue with the operation and possibly destroy
existing data? (y/n) [n]
```

Applications, as well as operating system commands, can modify the fstype of the disk label, to indicate that a partition is in use. See check_usage(3) and set_usage(3) for more information.

5.5 Copying Disks

You can use the dd command to copy a complete disk or a disk partition; that is, you can produce a physical copy of the data on the disk or disk partition.

Note

Because the dd command is not meant for copying multiple files, copy a disk or a partition only to a disk that you are using as a data disk, or to a disk that does not contain a file system. Use the dump and restore commands, as described in the *System Administration* manual, to copy disks or partitions that contain a UFS file system. Use the vdump and vrestore commands, as

described in the *AdvFS Administration* manual, to copy disks or partitions that contain an AdvFS fileset.

UNIX protects the first block of a disk with a valid disk label because this is where the disk label is stored. As a result, if you copy a partition to a partition on a target disk that contains a valid disk label, you must decide whether you want to keep the existing disk label on that target disk.

If you want to maintain the disk label on the target disk, use the `dd` command with the `skip` and `seek` options to move past the protected disk label area on the target disk. The target disk must be the same size as or larger than the original disk.

To determine whether the target disk has a label, use the following `disklabel` command:

```
# /sbin/disklabel -r target_disk
```

You must specify the target device directory name (`/dev`) followed by the raw device name, drive number, and partition `c`. If the disk does not contain a label, the following message is displayed:

```
Bad pack magic number (label is damaged, or pack is unlabeled)
```

The following example shows a disk that already contains a label:

```
# disklabel -r /dev/rdisk/dsk1c
type: SCSI
disk: rz26
label:
flags:
bytes/sector: 512
sectors/track: 57
tracks/cylinder: 14
sectors/cylinder: 798
cylinders: 2570
rpm: 3600
interleave: 1
trackskew: 0
cylinderskew: 0
headswitch: 0           # milliseconds
track-to-track seek: 0  # milliseconds
drivedata: 0

8 partitions:
#      size  offset  fstype [fsize bsize  cpg]
 a:  131072       0  unused 1024 8192 # (Cyl.    0 - 164*)
 b:  262144  131072  unused 1024 8192 # (Cyl.  164*- 492*)
 c: 2050860       0  unused 1024 8192 # (Cyl.    0 - 2569)
 d:  552548  393216  unused 1024 8192 # (Cyl.  492*- 1185*)
 e:  552548  945764  unused 1024 8192 # (Cyl. 1185*- 1877*)
 f:  552548 1498312  unused 1024 8192 # (Cyl. 1877*- 2569*)
 g:  819200  393216  unused 1024 8192 # (Cyl.  492*- 1519*)
 h:  838444 1212416  unused 1024 8192 # (Cyl. 1519*- 2569*)
```

If the target disk already contains a label and you do not want to keep the label, you must clear the label by using the `disklabel –z` command. For example:

```
# disklabel -z /dev/rdisk/dsk1c
```

To copy the original disk to the target disk and keep the target disk label, use the `dd` command, specifying the device directory name `(/dev)` followed by the raw device name, drive number, and the original and target disk partitions. For example:

```
# dd if=/dev/rdisk/dsk0c of=/dev/rdisk/dsk1c \
skip=16 seek=16 bs=512k
```

5.6 Monitoring Disk Use

To ensure an adequate amount of free disk space, regularly monitor the disk use of your configured file systems. You can do this in any of the following ways:

- Check available free space by using the `df` command (Section 5.6.1)
- Check disk use by using the `du` command or the `quot` command (Section 5.6.2)
- Verify disk quotas (if imposed) by using the `quota` command (Section 5.6.3)

You can use the `quota` command only if you are the root user.

5.6.1 Checking Available Free Space

To ensure sufficient space for your configured file systems, use the `df` command regularly to determine the amount of free disk space in all of the mounted file systems. The `df` command displays statistics about the amount of free disk space on a specified file system or on a file system that contains a specified file.

With no arguments or options, the `df` command displays the following:

- The amount of free disk space on all of the mounted file systems.
- Each file system's configured size, in 512-byte blocks. If you specify the `–k` option, size information is provided in kilobyte blocks.
- The total amount of space, the amount in use, the amount available (free), the percentage in use, and the directory on which the file system is mounted.
- For AdvFS file domains, the `df` command displays disk space usage information for each fileset.

If you specify a device that has no file systems mounted on it, df displays the information for the root file system. You can also specify a file path name to display the amount of available disk space on the file system that contains the file. See df(1) for more information.

Note

You cannot use the df command with the block or character special device name to find free space on an unmounted file system. Instead, use the dumpfs command.

The following example displays disk space information about all the mounted file systems:

```
# /sbin/df
Filesystem         512-blks   used  avail capacity Mounted on
/dev/disk/dsk2a      30686  21438   6178    77%  /
/dev/disk/dsk0g     549328 378778 115616    76%  /usr
/dev/disk/dsk2g     101372   5376  85858     5%  /var
/dev/disk/dsk3c     394796     12 355304     0%  /usr/users
/usr/share/mn@tsts  557614 449234  52620    89%  /usr/share/mn
domain#usr          838432 680320  158112   81%  /usr
```

Note

The newfs command reserves a percentage of the file system disk space for allocation and block layout. This can cause the df command to report that a file system is using more than 100 percent of its capacity. You can change this percentage by using the tunefs command with the –minfree flag.

5.6.2 Checking Disk Use

If you determine that a file system has insufficient space available, you might want to find out who is using the space. You can do this with the du command or the quot command.

The du command returns disk space allocation by directory. With this information you can decide who is using the most space and who should free up disk space.

The du command displays the number of blocks contained in all directories (listed recursively) within each specified directory, file name, or (if none are specified) the current working directory. The block count includes the indirect blocks of each file in 1-kilobyte units, independent of the cluster size in use by the system

If you do not specify any options, an entry is generated for each directory. See du(1) for more information on command options.

The following example displays a summary of blocks that all main subdirectories in the /usr/users directory use:

```
# /usr/bin/du –s /usr/users/*
440     /usr/users/barnam
43      /usr/users/broland
747     /usr/users/frome
6804    /usr/users/morse
11183   /usr/users/rubin
2274    /usr/users/somer
```

From this information, you can determine that user Rubin is using the most disk space.

The following example displays the space that each file and subdirectory in the /usr/users/rubin/online directory uses:

```
# /usr/bin/du -a /usr/users/rubin/online
1 /usr/users/rubin/online/inof/license
2 /usr/users/rubin/online/inof
7 /usr/users/rubin/online/TOC_ft1
16 /usr/users/rubin/online/build
  .
  .
  .
251 /usr/users/rubin/online
```

Note

As an alternative to the du command, you can use the ls –s command to obtain the size and usage of files. Do not use the ls –l command to obtain usage information; ls –l displays file sizes only.

5.6.3 Verifying Disk Quotas

You can use the quot command to list the number of blocks in the named file system currently owned by each user. You must be root user to use the quot command.

The following example displays the number of blocks that each user owns and the number of files owned by each user in the /dev/disk/dsk0h file system:

```
# /usr/sbin/quot -f /dev/disk/dsk0h
```

Note

You must specify the character device special file to return the information, because when the device is mounted the block special device file is busy.

See `quot`(8) for more information.

6

Processor-Specific Information

This chapter provides information on operating system features that are designed to support specific models and classes of AlphaServer processor. It also describes configuration restrictions that are permanent, and specific to certain models of processor. The following information is provided:

- Processor-specific information for older platforms (Section 6.1)
- Processor-specific information for the AlphaServer TS202c (Section 6.2)
- Configuring logical partitions on an AlphaServer GS140 (Section 6.3)
- AlphaServer 1000 and 1000A configuration information (Section 6.4)
- AlphaServer GS-series configuration information (Section 6.5)
- Personal Workstation 433au, 500au, and 600au systems (Section 6.6)

6.1 Legacy Processors

The following notes apply to older systems:

Alpha VME Single-board Computers

For information about configuring the operating system on Alpha VME single-board computers (SBCs) and PCI/ISA EBMnn modular SBCs, see the *System Configuration Supplement: OEM Platforms* manual. (The PCI/ISA modular systems and components product family was formerly known as DIGITAL Modular Computing Components, or DMCC.)

Support for the VME bus will be retired in a future release of the operating system. This includes retirement of systems and options that use this bus technology.

EISA Configuration Utility

For Tru64 UNIX and its software supplements, the supported version of the EISA Configuration Utility (ECU) is Version 1.10 or higher. If your system is configured with an EISA bus, update the ECU to this supported version.

6.2 AlphaServer TS202c

The AlphaServer TS202c system is a dual-processor system that you can configure with up to 16 GB of memory. The system contains two cPCI slots, but is configured without disk storage devices. The system is intended to run a network-bootable standalone system (SAS) kernel.

This section contains information that is specific to the AlphaServer TS202c system for Tru64 UNIX Version 5.1B. This information was first published in May 2001 as the *Release Notes and Installation Instructions for AlphaServer TS202c Systems*. In this release, the installation information is integrated into the Tru64 UNIX *Installation Guide*.

The following information is provided:

- Specific features of the operating system that you can configure on the AlphaServer TS202c (Section 6.2.1)
- Current restrictions on using certain operating system features (Section 6.2.2)
- Using the `mksas` command to create network-bootable kernels for diskless systems (Section 6.2.3)
- Configuring exempt memory regions (Section 6.2.4)
- Configuring the AlphaServer TS202c (Section 6.2.5)

6.2.1 Operating System Features

The following features were first added to the Tru64 UNIX Version 5.1 base operating system to support the release of the AlphaServer TS202c:

- Network-bootable standalone kernel that runs on an AlphaServer TS202c system in a diskless environment. See Section 6.2.3 and `mksas`(8).
- Exempted memory regions. See Section 6.2.4 and the `dump_exmem*` attributes documented in `sys_attrs_generic`(5).
- Saving system core dump files to memory and to writing them to a remote host. See `savecore`(8) and `vmzcore`(7).
- Specifying the size and the range of a memory segment that is allocated to a shared object when it is loaded into memory by the run-time loader. See `loader`(5).
- New hardware error codes that support new operating system features and new hardware features. See `binlogd`(8) for information on displaying and interpreting these error codes. The new codes are:
  - 113 – `Double error halt`.

    The system has detected a memory error. The system has detected a second fatal error during the processing of a fatal error. An error

detected by the memory troller is a correctable error, a noncorrectable error, or a double-bit error.

– 115 – `(Un)correctable environmental error`.

  The environmental monitoring daemon has detected an error.

– 120 – `Reporting of correctable errors is disabled because the single-bit error reporting threshold has been reached`.

6.2.2 Restrictions on Operating System Features

The following usage restrictions apply only to Tru64 UNIX Version 5.1B:

Firmware Requirement

The AlphaServer TS202c system firmware revision level must be minimally at Version X6.0–0 to run a network booted kernel.

RIS Installation

This release of the operating system does not support installation from a RIS server. This software can only be installed from the distribution CD–ROM.

Incorrectly Formatted rc.config File

The Event Manager (EVM) can fail if the `rc.config` file contains blank lines. The failures that may occur include the EVM daemon core dumping or the system displaying the following errors during the boot process:

```
evmwatch: Failed to create EVM listening connection
evmwatch: Error: Connection error
Failed to create EVM listening connection
evmwatch: Error: Connection error
S97evm: Communication with syslogd is not functioning
evmpost: Failed to create EVM posting connection
evmpost: Error: Connection error
evmpost: Failed to create EVM posting connection
evmpost: Error: Connection error
evmpost: Failed to create EVM posting connection
evmpost: Error: Connection error
SysMan authentication server started
SysMan Station Server (smsd) started
The system is ready.
```

Environmental Monitor Displays Incorrect Error Messages

When you change the ENVMON_HIGH_THRESH attribute value using the `/usr/sbin/envconfig` command line utility, the following error message is displayed:

```
env_high_temp_thresh: attribute does not allow this operation
```

Despite the message, the attribute value is changed and the daemon is restarted.

To view the correct attribute values, use the following command:

```
# /usr/sbin/envconfig -q
```

Incorrect Entries in the Binary Error Log

The system incorrectly logs Correctable Environmental Machine Check errors (Error 686) in the binary error log as Uncorrectable Environmental Machine Check errors (Error 682). This problem is specific to the AlphaServer TS202c systems and will be corrected in a future release of the operating system.

Binary Error Log Event May be Incorrectly Reported as Double Error Halt

A binary error log (binlog) event with type 113 is reported as a Double Error Halt event when reported by the Event Manager (EVM), but is reported as a Console Data Log event by the analysis utility. EVM might report this event by mailing it to the `root` user and by displaying it on the system console.

The event is actually a Console Data Log event. This event type is posted when any of several different errors occur, including double error halts, uncorrectable environmental errors, and platform-specific system faults. Refer to the event's translation data for information about its cause.

hwautoconfig Does Not Create Device Special Files for Subsystems

When the system boots, the `hwautoconfig` utility configures subsystems but does not create device special files for the subsystems. To create the device special files for a subsystem, unconfigure the subsystem by using the `sysconfig -u` command, and then reconfigure the subsystem using the `sysconfig -c` command.

Alternately, you can work around this problem by adding the subsystem names to the environment variable SUBSYSTEM_LIST in the `autosysconfig` file before running the `hwautoconfig` utility.

Idle System Might Panic When kmem_debug_leak Flag Is Set

An idle AlphaServer TS202c system might panic if `kmem_debug` is enabled and the `kmem_debug_leak` flag is set. To avoid this problem, do not set the `kmem_debug_leak` flag.

Incorrect Code Being Entered in the SEL Log

When 670 and OS Panic events are logged on an AlphaServer TS202c system, the SEL event information is not logged correctly. These events are logged with byte 3 set to `DF`, instead of `C0` for 660/670 entries and `C1` for OS Panics.

The following events overlap because of the incorrect logging:

670 events codes

x40 PAL detected Bugcheck Error
x43 EV68 detected Dcache Tag Parity Error
x45 EV68 detected Duplicate Dcache Tag Parity Error
x46 EV68 detected Bcache Tag Parity Error
x4A EV68 detected Double Bit ECC Memory Fill Error
x5C EV68 detected Second Dcache Store Data ECC Error

OS panic codes

x40 ku_recvfrom - not SO_NAME
x45 m_copym offset
x46 m_copym sanity
x43 nfs3_bio: write count
x4A nfs_dgreceive 3
x5C rtinithead

These events are logged correctly in the binary error log.

6.2.3 Using the mksas Command

The following usage notes apply only to the `mksas` command when used on the AlphaServer TS202c system.

6.2.3.1 Configuration File Requirement

You use the `mksas` utility to generate the bootable image and in-memory file system. This image provides you with the ability to boot over network and to run diskless. Using `mksas` requires a configuration file for the target system. For example, if the target host is TS2ONE, a configuration file for TS2ONE must be located on the host system, TS2TWO for the host system to generate the bootable image.

6.2.3.2 Restriction on Using the su Command

When you run a kernel that you created by using the `mksas` utility, you cannot use the `su` command to substitute a user ID to any user ID other than the root user. (This problem is caused by an incorrect `umask` setting in the `mksas` utility.)

6.2.3.3 Default Configuration File Does Not Include ftp

To configure the system so that it writes core dump files to a remote system, you must edit the default configuration file and include the `ftp` command as an option. For more information on editing the default configuration file, see Section 6.2.3 and `mksas`(8).

6.2.3.4 The mksas Command Generates Warning Messages

If the configuration file that you specify with the `mksas` command contains duplicate entries, error messages similar to the following are displayed:

```
Entry no: 160  -> /usr/lib/sabt/etc/mksas.inittab /etc/inittab
WARNING Entry no: 160. Duplicated entry. The file
/etc/inittab in the second field
is already given earlier.
Entry no: 161  -> /etc/rc.config /etc/rc.config
WARNING Entry no: 161. Duplicated entry. The file
/etc/rc.config in the second field
is already given earlier.
Entry no: 162  -> /etc/rc.config.common /etc
WARNING Entry no: 162. Duplicated entry.
The file/etc/rc.config.common in the second field
is already given earlier.
Entry no: 163  -> /etc/hosts /etc/
WARNING Entry no: 163. Duplicated entry.
The file /etc/hosts in the second field
is already given earlier.
```

You can ignore these messages. The kernel will build without errors.

6.2.3.5 Example addfile Available

An example `/usr/lib/sabt/etc/addlist_file` is supplied with the system. You use the `addlist_file` to include additional files in the miniroot file system. The example is located in the `/usr/lib/sabt/etc` directory and is named `mksas.inv`.

6.2.3.6 The -t Option Deletes the Workspace Directory When mksas Completes

You use the `-t`*directory_name* option with the `mksas` command to specify a temporary work that is used during the kernel build. When the `mksas` utility completes, it deletes both the temporary work space and the directory

containing the work space. If you use the `-t` option, do not specify as the *directory_name* any location containing files that you do not want to delete.

6.2.4 Allocating Exempt Memory

Exempt memory is memory managed by UNIX, but not included in testing and core dumps. You can allocate an exempt region of memory by using `contig_malloc()`. This can be done in a pseudodevice driver in the `postconfig_callback` routine.

The process of allocating exempt memory is similar to a normal contiguous memory allocation (`malloc`), with the following exceptions:

- The `type` field is `M_EXEMPT`.

  You can track the memory usage by type by using the `vmstat -M` command. Find the section of display output that is formatted as shown in the following example:

```
Memory usage by Type and Number of bytes being used

ADVFS       = 7088016  KERN        = 1317456  SONAME      = 11264
AIO         = 8192     KERNEL TBL  = 456912   STREAMS     = 1856
ANON        = 111200   KEVM        = 65536    STRHEAD     = 2560
.
.
.
```

- The `addrlimit` field is overloaded to allow you to specify a starting address. If you specify an address that is not available, the call returns a failure. (The alignment parameter is useless in this case.)

A call will appear similar to the following example:

```
addr = contig_malloc(size,  alignment , start_addr, M_EXEMPT, flags);
                   |      |           |
                   |      |           Base address to allocate from or
                   |      |           zero if no preference.
                   |      Allocation alignment if no start_addr or zero to
                   |      default to base page size.
                   Allocation size in bytes, must be specified.
```

You can also configure exempt memory by defining the `cma_dd` subsystem, using a `sysconfigtab` entry similar to the following:

```
CMA_option = size - 0x40000, alignment - 0, Addrlimit - 0, Type - 150, Flag - 1
```

The arguments are the same as in the previous call. The value 150 corresponds to the contents of `malloc.h`.

The returned `addr` has the low bit set if there is a bad page in the allocated region.

6.2.5 Configuring the AlphaServer TS202c

This section provides the following configuration instructions for the AlphaServer TS202c:

- How to create a network-bootable kernel (Section 6.2.5.1)
- How to configure the server and clients (Section 6.2.5.2)

6.2.5.1 Creating a Network-Bootable Kernel

You use the `mksas` utility to create the network-bootable (SAS) kernel. By default, the SAS kernel is created with the minimum number of files and directories required to boot the system to single-user mode, is contained in a memory file system, and is intended to run as an in-memory file system. You can also build a disk-bootable version of the SAS kernel for testing and debugging. The `mksas` utility uses the kernel configuration file of the system that the kernel will run on to build the kernel; it then adds additional files.

The following steps provide an example of how to build the SAS kernel and boot it across the network. Debugging steps are also provided.

1. You configure a host system so that it is running a version of the Tru64 UNIX operating system that supports the `mksas` utility. The host system must also contain a copy of the AlphaServer TS202c kernel configuration file.
2. You create the network-bootable SAS kernel and miniroot file system on the host system. For debugging purposes, you also create a disk-bootable copy of the same SAS kernel and miniroot file system.
3. You configure the host system to run BOOTP and `tftp` and identify client systems that will boot the SAS kernel across the network.
4. On the client system, you issue a `boot` command specifying the network interconnect as the boot device, for example, **`>>>boot eia0`**.

To add additional features and files to the miniroot file system, you use the `-a` option and specify an *addfile_list*, or list of additional files, with the `mksas` command. See `mksas`(8) for a complete list of the options available with this command. When you add files to the default configuration, you must include any files that are dependencies. For example, if you intend to write crash-related files to a remote host using the `savecore` command, you must include `/usr/sbin/ftp` and `/usr/sbin/ftpd` in your configuration.

While you add files, you can determine the size of the file system using the `-C` option of the `mksas` command. This option determines the size of the miniroot file system, but does not include the size of the kernel. Normally, a kernel is approximately 13 MB. You must add this amount to the size that

is returned by the -C option to determine the total size of the file system. AlphaServer systems support booting an image up to 92 MB.

6.2.5.2 Configuring the Server and Clients

After you have created the network-bootable SAS kernel, you must configure a server system that the SAS kernel can be booted from and configure client systems to boot and run the SAS kernel. The server system requires BOOTP support, and the tftp program, including RFCs 1782 and 1783.

To configure the server, perform the following steps:

1. Edit the /etc/inetd.conf file so that joind and tftp service boot requests from the client system, and identify the directory that the SAS kernel image will be booted from. To do this, uncomment the following lines and append the correct directory to the end of the tftp line:

   ```
   #tftp  dgram  udp  wait  root  /usr/sbin/tftpd  tftpd /tmp
   #bootps  dgram  udp  wait  root  /usr/sbin/joind  joind
   ```

   Remove the reference to the /tmp directory and replace it with the directory in which you will store the SAS kernel.

2. Use the rcmgr command to set the run-time configuration variables for joind:

   ```
   # rcmgr set JOIND yes

   # rcmgr set JOIND_FLAGS ""
   ```

3. Create or edit the /etc/bootptab file. This file is a text file containing information that the server needs to boot a remote client. Use the xjoin GUI to edit or create the /etc/bootptab file. See xjoin(8) and bootptab(4) for more information.

   Add the following lines to this file:

   ```
   .ris.dec:hn:vm=rfc1048:sm=255.255.255.0:
   .ris0.alpha:tc=.ris.dec:bf=/usr/mksas.kernel:
   anchor:tc=.ris0.alpha:ht=ethernet:gw=16.142.160.1:ha=00508B6B32CC:
   ip=16.142.160.55:
   ```

   The .ris.dec entry defines characteristics common to all clients. The fields specify the following:

   - hn: – Informs the boot server to send the name of the client system to the client when it makes a boot request.
   - vm: – Defines vendor-specific information.

   The .ris0.alpha entry defines characteristics common to all clients using the bootp server. The fields specify the following:

- `tc:` Lets you follow pointers back to common entries. For example, the `tc` entry for `.ris0.alpha` points to the `.ris.dec` entry. The `.ris.dec` entry contains the common hardware type (`ht`) and vendor specific (`vm`) information. The `.ris0.alpha` entry itself contains common information about the boot file location.
- `bf:` Specifies the name of the boot file.

The host name entry in this example defines characteristics for a specific client. The fields specify the following:

- `tc:` Points to `ris0.alpha`, which contains its boot file information. The `ris0.alpha` entry in turn points back to `ris.dec`, which contains relevant hardware type and vendor specific information.
- `ht:` Defines the client's hardware type as `ethernet`, `fddi`, or `ieee802` (for Token Ring).
- `ha:` Specifies the client's network hardware address.
- `gw:` Defines the gateway.

4. Start the `joind` server process using the following command:

```
/sbin/init.d/dhcp start
```

To boot the client, you use the following command:

```
>>> boot eia0
```

The device name `eia0` is the name of the network interconnect. This device name may be different on some AlphaServer systems. You can use the following console command to determine the name:

```
>>> show dev
```

If you are booting a SAS kernel that is larger than 64 MB, you need to set the `eia0_TFTP_blocksize` console variable to 1450:

```
>>> set eia0_TFTP_blocksize 1450
```

You can write crash dump files to exempt memory or to a remote host. Exempt memory dumps are performed by setting the `dump_to_memory` system attribute and then by identifying the portion of exempt memory or all of the memory that the kernel dump subsystem will use in the event of a memory core dump. The system attribute `dump_exmem_addr` identifies the start of the dumpable region (as either a virtual or physical address). The `dump_exmem_size` system attribute specifies the number of bytes available. By default, the content of exempt memory is not included in the dump. You can change this by setting the system attribute `dump_exmem_include`. See `sys_attrs_generic`(5) for more information on these system attributes.

To write crash-related files to a remote host, you use the `-r` option of the `savecore` command. When using this option, you specify a host, a username,

a password, and the file that is to be written to the remote host. You can use a configuration file to specify the `ftp` information and the name of the file that will be written. Using a configuration file allows you to keep the account name and password private. See `savecore`(8) for more information.

6.3 AlphaServer GS140 Logical Partitions

A single AlphaServer GS140 system can be divided into a maximum of three logical partitions. Each partition is allocated its own dedicated set of hardware resources. A partition is viewed by the operating system and applications software as a single AlphaServer GS140 system.

Logical partitions employ a share nothing model. That is, all hardware resources (processors, memory, and I/O) allocated to a partition are isolated to that partition. Only the instance of an operating system that is running on a partition can access that partition's hardware resources.

You can use logical partitions to reduce floor space requirements, power consumption, or improve heat dissipation (by reducing computer room cooling requirements). For example, two departments in an enterprise with different computing requirements might run different applications and require different configuration and tuning of the operating system. Logical partitioning allows you to configure a single AlphaServer GS140 computer to meet the computing needs of both departments.

6.3.1 Hardware Requirements

The hardware requirements for a partition are:

- An AlphaServer GS140 with a minimum of six center plane slots

  Only the AlphaServer GS140 6–525 is supported. See the *Systems and Options Catalog* for information on newly supported systems. The logical partitions feature is supported on the AlphaServer GS140 system. An AlphaServer 8400 (upgraded to a GS140 by replacing the processor modules) is also supported.

- A console device

  This console device can be a character cell video terminal or serial line connection to another system or terminal concentrator. Supported graphics devices can be used by the operating system's windowing software, but not as the console device.

  The restriction of a graphics device to the windowing software (which cannot be the console device) applies only to secondary partitions. A supported graphics device can be the console for the primary partition (partition 0). To use a graphics console, set the value of the `console` environment variable to `BOTH` before initializing partitions. For example:

```
P00>>> set console BOTH
```

The AlphaServer GS140 includes one console serial port. This port becomes the console for the first partition (partition 0). Each additional partition requires the installation of a KFE72 option. This option includes two serial ports (port 0 is the console port). See the hardware documentation for the KFE72 option information and installation instructions.

- One dual processor CPU module
- One I/O Port (IOP) module

  The minimum requirement for a partition is one IOP module. A partition might include a second IOP module. The maximum number of IOP modules for the entire system (the sum of all partitions) is three.
- XMI hardware might be used with logical partitions. However, XMI controllers and devices must be configured into partition zero (0). This is a console firmware restriction.
- One memory module

  The minimum memory size supported for a partition is 512 MB. However, applications running in a partition might require more than the 512 MB minimum memory.
- A software load source device (CD-ROM drive or network adapter)
- A minimum AlphaServer GS140 console firmware revision level of Version 5.4–19

  When installing and configuring logical partitions on a system, see the *Release Notes* for the operating system release that you are installing, and update the firmware revision if required. See the *Installation Guide* for information on updating the firmware.

The remainder of this section describes the tasks you perform to configure partitions, and provides information about managing a partitioned AlphaServer GS140 system. The topics covered describe the following activities:

- Preparing to install and operate a partitioned system
- Verifying system hardware is properly configured for partitions
- Verifying the revision level of your system's console firmware and upgrade the firmware if necessary
- Configuring partitions for your system by creating the logical partitioning console firmware `environment variables` (EVs)
- Initializing partitions and bootstrap secondary partitions to console mode (the `P##>>>` prompt)

- Installing UNIX and applications software to each partition
- Operating and managing a partitioned system

6.3.2 Preparing to Install and Operate Logical Partitions

Read the hardware documentation supplied with your system to become familiar with the operation of your system. Of particular interest for partitioning are the operation of the system's OFF/SECURE/ENABLE/RESET switch and several console commands (such as `boot`, `create`, `init`, `set`, and `show`).

Before setting up your partitions, make sure the system hardware is fully installed and passes all self-test diagnostics.

Note

Before installing the operating system software to any partition, read all of the partitioning procedure. There are certain aspects of managing a partitioned system you must be aware of prior to making the system operational. Precautions must be taken to prevent actions by the console on a partition from interfering with operation of another partition.

The next section describes logical partitioning terms used throughout the rest of this document. After reviewing these terms, proceed to Section 6.3.3.

6.3.2.1 Definition of Commonly Used Terms

Familiarize yourself with the following terms before configuring your partitions.

logical partition

A logical grouping of hardware resources (CPU, I/O, MEMORY, and console) within a single system for exclusive use by an instance of the operating system. A single physical system might have several logical partitions, each running a separate instance of the operating system.

primary partition

Partition number zero. The partition with the active console terminal if partitioning is disabled (that is, all hardware resources are in one partition).

secondary partition

Partition with a number greater than zero. One of the partitions that displays the console prompt after the `lpinit` command is executed on the primary partition's console.

primary console

The console terminal connected to the primary partition. The only active console terminal if partitions are disabled.

secondary console

The console terminal connected to a secondary partition. Active only if partitions are enabled.

power OFF/ENABLE switch

The four-position switch located on the AlphaServer GS140 control panel. The four positions perform the following functions:

- OFF – System power (all partitions) is off.
- SECURE – Power is applied to the system (all partitions). The primary console's `ctrl/p halt` function is disabled.
- ENABLE – Power is applied to the system (all partitions). The primary console's `ctrl/p halt` function is enabled.
- RESET – This is a momentary position. Moving the switch to RESET and then releasing it causes a complete initialization of the system. All secondary partitions are immediately terminated. The primary partition displays the normal power-on self-test messages and enters console mode.

console prompt

The prompt displayed on the console terminal of a partition to indicate the console firmware is ready to accept commands:

```
P##>>>
```

Where ## is the processor number on which the console firmware is currently executing. This is normally the primary processor of the current partition as shown in the following examples:

- For partition 0 with CPU 0:

  ```
  P00>>>
  ```

- For partition 1 with CPU 4:

  ```
  P14>>>
  ```

`ctrl/p halt`

Holding down the control key and typing the letter `p` causes the primary processor for partition 0 to halt and enter console mode (`P00>>>` prompt). This is possible only on the primary console. The `halt` operation can be disabled by setting the power switch to the `SECURE` position. The halt operation is ignored on secondary partitions.

`P##>>>stop N`

Typing `stop` *N* at the console prompt (P##>>>) causes processor *N* to halt and enter console mode. Issuing this command on the primary console

terminal can stop any processor in any partition. For example, if the primary processor for partition 1 is processor 4, the following command causes processor 4 to enter console mode:

```
P00>>>stop 4
```

P##>>>continue *N*

If processor *N* entered console mode as the result of a `ctrl/p halt` or `stop` *N* command, typing `continue` *N* at the P##>>> prompt causes the processor to resume program execution. For example:

```
P##>>>continue 4
```

If you halt a single processor you can omit the processor number (*N*).

P##>>>init

Typing `init` at the console (P##>>>) prompt of any partition causes a complete reinitialization of the entire system. All active partitions are immediately terminated and the system is reset (as if the power switch is momentarily moved to the RESET position). If partitions are enabled, the console requests verification of the `init` command by displaying the following prompt:

```
Do you really want to reset ALL partitions? (Y/<N>)
```

Type Y to complete the `init` command or N to cancel it.

6.3.3 Logical Partitions Configuration and Installation Tasks

Each of the following sections describes a task you perform to partition your AlphaServer GS140 system. Each task is performed in the order presented, although some tasks might be skipped in certain cases.

If you have read this section previously, and require only a summary of the normal sequence of startup commands, they are:

```
P00>>> set lp_count  n
            (Set the count of n logical partitions)

P00>>> init
            (Initialize the primary partition)

P00>>> lpinit
            (Start the secondary partitions)

P00>>> boot
            (Boot the primary partition)

P##>>> boot
            (boot the secondary partitions)
```

Improper operation results if the `lpinit` command is omitted. The console firmware prevents this by automatically executing the `lpinit` command if the `lp_count` is nonzero and a boot command is issued on the primary partition's console terminal.

On startup, each secondary partition displays configuration information. It is possible for this message to be proceeded by a series of Y characters as described in Section 6.3.3.8. This is not an error and can be ignored.

6.3.3.1 Verifying Your System's Hardware Configuration

You need to verify that your hardware is properly configured for logical partitioning. You also need to record certain information about your hardware configuration for later use (when you configure partitions). Follow these steps to verify your hardware configuration:

1. Power on your system by setting the power `OFF/ENABLE` switch to the `ENABLE` position.

   Note

   A newly installed system (with factory installed software) or an existing system with the `auto_action` console EV set to BOOT or RESTART, automatically boots the operating system disk after the hardware's self-test is completed. In this case, you need to interrupt the automatic boot by typing `ctrl/c` at the console terminal. If you cannot interrupt the automatic boot, allow the operating system boot completely, then shut it down (do not type `ctrl/p` to halt the automatic boot). See the *Installation Guide* for information on factory installed software before you attempt to set up logical partitions.

   The factory installed software disk might be used as the system disk for one of the partitions. See Section 6.3.6 for information on installing the operating system.

2. After a short delay (about 15 seconds) configuration information similar to the following example is displayed on the primary console screen:

```
F   E   D   C   B   A   9   8   7   6   5   4   3   2   1   0   NODE #
                        A   A   M   .   M   P   P   P   P   TYP
                        o   o   +   .   +  ++  ++  ++  ++   ST1
                        .   .   .   .   .  EE  EE  EE  EB   BPD
                        o   o   +   .   +  ++  ++  ++  ++   ST2
                        .   .   .   .   .  EE  EE  EE  EB   BPD
                        +   +   +   .   +  ++  ++  ++  ++   ST3
                        .   .   .   .   .  EE  EE  EE  EB   BPD

                            .   +   +   +   .   +   +   +   C0 PCI +
```

```
    .   .   .   .   .   .   +   .   .   +   .   .   +   +       C1 XMI +

.   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   C4
                +   .   +   +   .   .   .   .   +   .   .   +   C5 PCI +
.   .   .   .   .   .   .   .   .   .   .   .   .   .   .   .   C6
                +   .   +   +   .   +   +   +   .   .   .   +   C7 PCI +
                            .   .   .   +   .   .   .   .           EISA +

                            .   .  A1   .  A0   .   .   .   .   ILV
                            .   . 1GB   . 1GB   .   .   .   .   2GB
Compaq AlphaServer GS140 8-6/525, Console V5.4 15-MAR-99 10:07:33
SROM V1.1, OpenVMS PALcode V1.48-3, Tru64 UNIX PALcode V1.45-3
System Serial =  , OS = UNIX, 12:58:49  March 15, 1999
Configuring I/O adapters...
isp0, slot 0, bus 0, hose0
isp1, slot 1, bus 0, hose0
tulip0, slot 2, bus 0, hose0
isp2, slot 4, bus 0, hose0
isp3, slot 5, bus 0, hose0
tulip1, slot 6, bus 0, hose0
demna0, slot 1, bus 0, xmi0
kzmsa0, slot 2, bus 0, xmi0
kzmsa2, slot 5, bus 0, xmi0
kzpsa0, slot 3, bus 0, hose5
tulip2, slot 8, bus 0, hose5
tulip3, slot 9, bus 0, hose5
pfi0, slot 11, bus 0, hose5
tulip4, slot 12, bus 0, hose7
floppy0, slot 0, bus 1, hose7
kzpsa1, slot 4, bus 0, hose7
tulip5, slot 4, bus 2, hose7
tulip6, slot 5, bus 2, hose7
tulip7, slot 6, bus 2, hose7
tulip8, slot 7, bus 2, hose7
pfi1, slot 6, bus 0, hose7
pfi2, slot 8, bus 0, hose7
kzpsa2, slot 9, bus 0, hose7
P00>>>
```

3. The line ending with NODE # indicates the slot number (referred to later in the configuration process). Your system provides up to nine slots, each of which is labeled with its slot number. The next line (ending with TYP) indicates the type of module in each slot. Record the type of module in each slot:

```
P = CPU (dual processor CPU module)
M = MEM (memory module)
A = IOP (IO port module)

  8   7   6   5   4   3   2   1   0
+---+---+---+---+---+---+---+---+---+
|   |   |   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+
```

4. Divide your system into logical partitions by assigning slots (and therefore modules) to each partition. Each partition must be assigned

at least one dual CPU module, one MEM module, and one IOP module. With a total of nine slots, the AlphaServer GS140 can be configured for a maximum of three partitions.

Note

Each CPU module includes two processors, both of which must be assigned to the same partition.

5. If your system meets the minimum requirements, proceed to the next section. Otherwise, you need to take corrective action (such as installing additional hardware), then proceed to the next section.

6.3.3.2 Verifying the Firmware Revision Level

Logical partitions require console firmware support. See the *Release Notes* for changes to the minimum revision. To verify that your system's firmware includes support for logical partitions, use the following command at the primary console to display the firmware revision level:

```
P00>>>show version
```

The console displays a message similar to the following:

```
version  V5.4,  15-MAR-1999 10:07:33
```

Verify the revision of your firmware is Version 5.4 or later. If you need to upgrade your system's firmware, see the firmware upgrade instructions in the hardware documentation. The firmware CD-ROM is shipped with the software kit, or you can download the firmware from the World Wide Web or using ftp. The information on finding and updating the firmware is in the *Installation Guide*.

6.3.3.3 Configuring Logical Partitions

You configure and enable (or disable) logical partitions using a set of console environment variables (EVs). Two console EVs take the form of hexadecimal numbers, which are bit masks in which a bit position in the mask corresponds to a module or processor number. Hardware configuration rules require modules to be installed in specific slot numbers, based on the module type, according to the following criteria:

- IO port (IOP) modules are installed in slots 8, 7, and 6 in descending order with a maximum of three IOP modules allowed.
- CPU (dual processor) modules are installed in slots 0 through N in ascending order (N depends on the number of CPU modules installed). The value of N is limited by the number of IOP and MEM modules.

- MEM (memory) modules are installed in any available slot between the highest numbered CPU module and the lowest numbered IOP module.

Set the processor mask variable (lp_cpu_mask) by shifting the number 3 by two times the slot number of the CPU module. Possible CPU masks for each slot are:

```
Processors 00 and 01  (slot 0): 3 << (2 * 0) = 003
Processors 02 and 03  (slot 1): 3 << (2 * 1) = 00c
Processors 04 and 05  (slot 2): 3 << (2 * 2) = 030
Processors 06 and 07  (slot 4): 3 << (2 * 4) = 0c0
Processors 08 and 09  (slot 5): 3 << (2 * 5) = 300
Processors 10 and 11  (slot 6): 3 << (2 * 6) = c00
```

Calculate the value of the lp_cpu_mask variable by combining (using a logical OR operation) the masks for individual CPU module slots. For example, to assign the four processors on the CPU modules in slot 0 and 1 to partition 0, you assign the lp_cpu_mask0 variable a value of 00f.

Set the I/O port mask variable (lp_io_mask) by left shifting the number 1 by the slot number of the IOP module. Potential IOP masks for each slot are:

```
IO Port module in slot 8: 1 << 8 = 100
IO Port module in slot 7: 1 << 7 = 080
IO Port module in slot 6: 1 << 6 = 040
```

If a partition consists of two IOP modules, create the value of the lp_io_mask variable by combining (using a logical OR operation) the masks for individual IOP module slots. For example, if you assign IOP modules in slots 7 and 8 to partition 1, the value of the lp_io_mask1 variable is 180.

When assigning IOP modules to secondary partitions, it is important to remember that one of the IOPs assigned to the partition must be connected to a DWLPB option with a KFE72 option installed. The KFE72 option provides the console serial port for secondary partitions.

6.3.3.4 Determining and Setting Environment Variables

To create the console environment variables for your logical partitions, first determine the number of partitions and which slots (that is, CPU, MEM, and IOP modules) are assigned to each partition (using the module types and slot numbers you recorded previously). Then, you can create the console EVs.

A summary of console EVs and values follows:

Console EV	Value
lp_count	Number of partitions
lp_cpu_mask*N*	CPU assignment mask for partition *N*

Console EV	Value
lp_io_mask*N*	IOP module assignment mask for partition *N*
lp_mem_mode	Memory isolation mode

The following table shows a sample configuration of two partitions based on the configuration information in Section 6.3.3.3, with the following modules:

- 4 CPU modules (in slots 0 through 3)
- 2 MEM modules (in slots 4 and 6)
- 2 IOP modules (in slots 7 and 8)

Partition	Modules
Partition 0	CPU modules in slots 0 and 1 (CPU 0-3, mask = 00F)
	IOP module in slot 8 (I/O Port, mask = 100)
	MEM module in slot 6 (2GB memory)
Partition 1	CPU modules in slots 2 and 3 (CPU 4-7, mask = 0F0)
	IOP module in slot 7 (I/O Port, mask = 080)
	MEM module in slot 4 (1GB memory)

There is no console EV mask for memory. The console firmware assigns memory modules to partitions. The firmware attempts to balance the amount of memory assigned to each partition.

To create or change the EVs, execute the following commands at the console prompt. The values used are for the two-partition example described at the start of this section. The actual values you enter depend on your hardware configuration and your partition layout.

The value of the lp_count EV is zero (which changes later).

The following command displays the console EVs if you have created them. No output appears if the console EVs do not exist.

```
P00>>>show lp*
```

If the console EVs do not exist (were not previously created) use the following commands to create the EVs.

There is a 10 second delay after you issue each command and that the console echoes the value of each EV after you create it.

```
P00>>>create -nv lp_count 0
P00>>>create -nv lp_cpu_mask0 f
P00>>>create -nv lp_cpu_mask1 f0
```

```
P00>>>create -nv lp_io_mask0 100
P00>>>create -nv lp_io_mask1 80
P00>>>create -nv lp_mem_mode isolate
```

If the console EVs already exist (previously created), use these commands to set their values:

```
P00>>>set lp_count 0
P00>>>set lp_cpu_mask0 f
P00>>>set lp_cpu_mask1 f0
P00>>>set lp_io_mask0 100
P00>>>set lp_io_mask1 80
P00>>>set lp_mem_mode isolate
```

Use the information in the following two sections to display (and if necessary correct) the console EV settings.

6.3.3.5 Displaying Console Environment Variables

Display the value of a console EV on the console of any partition by using the `show` command. For example, to display the value of `lp_count` enter the following:

```
P00>>>show lp_count
```

To display all the partitioning EVs, enter the following:

```
P00>>>show lp*
```

If the console EVs are correct, ignore the next section and proceed to Section 6.3.3.7. Otherwise, continue with Section 6.3.3.6 and make any necessary corrections.

6.3.3.6 Correcting Console Environment Variables

Note

You must set console EVs with `lp_` prepended to the EV name by using only the console of the primary partition (partition 0). You must not change the value of these variables on any secondary partition.

Use the `set` command to change the value of any or all the console EVs. For example, to change all the EVs, enter the following:

```
P00>>>set lp_count 0
P00>>>set lp_cpu_mask0 f
P00>>>set lp_cpu_mask1 f0
P00>>>set lp_io_mask0 100
P00>>>set lp_io_mask1 80
P00>>>set lp_mem_mode isolate
```

6.3.3.7 Disabling Automatic Boot Reset

The *Installation Guide* recommends setting the `boot_reset` console environment variable to ON. This setting is not compatible with logical partitions for which the `boot_reset` console EV must be set to `OFF`. This is required so booting a partition does not interfere with the operation of other (previously booted) partitions. If the `boot_reset` console EV is set to `ON`, then a system-wide reset happens after you execute the boot command (`P00>>>boot`). This reset immediately terminates the operation of all partitions.

Execute the following command to disable the `boot_reset` console EV:

```
P00>>>set boot_reset off
```

6.3.3.8 Setting Memory Interleave Mode

Set the value of the `interleave` console EV to `none`:

```
P00>>>set interleave none
P00>>>init
```

When setting the interleave mode to none, the console might echo a series of Y characters to the console display screen. (There might be several lines of Y characters.) Ignore this output.

6.3.3.9 Setting the Operating System Type to UNIX

Set the value of the `os_type` console EV to UNIX:

```
P00>>>set os_type UNIX
```

6.3.3.10 Setting the auto_action Console Environment Variable

To halt the processor after a POWER-ON or RESET (using the RESET switch), use the following command:

```
P00>>>set auto_action halt
```

To automatically boot the operating system after a POWER-ON or RESET, use the following command:

```
P00>>>set auto_action boot
```

6.3.4 Initializing Partitions

Before installing Tru64 UNIX to partitions, you must initialize the partitions. This operation assigns hardware resources (`CPU`, `IOP`, and

MEM modules) to each partition and spawns a console for each secondary partition. Use the following procedure:

1. Set the value of the lp_count EV to the number of partitions. For example, to enable two partitions:

   ```
   P00>>>set lp_count 2
   ```

2. Initialize partition 0:

   ```
   P00>>>init
   ```

   Configuration information (as previously described) is displayed on the primary console screen, followed by the console prompt; P00>>>.

3. Initialize all secondary partitions.

   ```
   P00>>>lpinit
   ```

A series of partition configuration messages is displayed on the primary console, including the starting address of physical memory for each partition. Record these addresses so you can determine whether a kernel rebuild is needed in the event of a memory configuration change.

The following example shows a typical configuration display:

```
Partition 0: Primary CPU = 0
Partition 1: Primary CPU = 4
Partition 0: Memory Base = 000000000   Size = 080000000
Partition 1: Memory Base = 080000000   Size = 040000000
No Shared Memory
LP Configuration Tree = 128000
starting cpu 4 in partition 1 at address 040010001
starting cpu 5 in partition 1 at address 040010001
starting cpu 6 in partition 1 at address 040010001
starting cpu 7 in partition 1 at address 040010001
```

For each secondary partition configured, information is displayed on the secondary console screens, followed by a console prompt such as P04>>>. There is a 20–second delay after you enter the lpinit command before the secondary consoles display their configuration information.

6.3.5 Correcting Interleave Mode Errors

If the interleave EV is incorrectly set, the console displays the following error message:

```
Insufficient memory interleave sets to partition system.
Issue command "set interleave none" then reset system.
```

To recover from this error, enter the following commands:

```
P00>>>set interleave none
```

```
P00>>>set lp_count 0

P00>>>init
```

Then repeat the steps in this section.

6.3.6 Installing the Operating System

After the partitions are configured and initialized, you can install the operating system to each partition. Install the operating system by following the instructions in the *Installation Guide*.

AlphaServer GS140 systems ship with Tru64 UNIX preinstalled on one of the disks. You can use this disk as the root disk for one of the partitions (usually partition 0). To use the preinstalled disk, boot it and follow the instructions for completing the installation. By default, the `bootdef_dev` console EV is set to automatically boot the preinstalled disk. If it is not, use the `bootdef_dev` value you recorded in Section 6.3.3.1.

Note

Depending on how you assigned IOP modules, the name of the factory installed software (FIS) disk might change and might not be assigned to partition 0. You can use the following command in each partition to locate the disk:

```
P##>>> show device
```

The operating system can also be installed from a CD-ROM or over the network from a Remote Installation Server (RIS). Configuring a CD-ROM drive on all partitions might not always be practical and a RIS server might not be available. An alternative, (assuming a local network is available) is to install the operating system to one partition from a CD-ROM, then configure that partition as a RIS server for the other partitions. Refer to the *Sharing Software on a Local Area Network* manual for instructions on setting up a Remote Installation Server.

6.3.7 Managing a Partitioned System

The operating system running in each partition can be managed as if it were running on a system that is not partitioned. However, there are some AlphaServer GS140-specific operational characteristics that you must be aware of and take into account when managing a partitioned system. These topics are documented in the following sections.

6.3.7.1 Operational Characteristics

During the course of normal partitioned system operations you might need to repeat some of the configuration and initialization tasks. Some of these tasks require special precautions to prevent interference between partitions. The following sections describe these tasks.

6.3.7.1.1 Console init command (P##>>>init)

Typing the `init` command at the console prompt in any partition reinitializes the entire system. This immediately terminates the operating system on all partitions. Therefore, do not execute the `init` command unless you need to reinitialize the entire system.

When you execute the `init` command, the console prompts you to confirm that you actually want to reset all partitions. Answer `no` to abort the `init` command or `yes` to continue with the `init` command.

6.3.7.1.2 Shutting Down or Rebooting the Operating System

To shut down the operating system running in a partition and return to console mode (P##>>> prompt), use the `shutdown` command. For example:

```
# /usr/sbin/shutdown -h +5 "Shutting down the OS"
```

The `shutdown` command can also shut down and reboot the operating system. For example:

```
# /usr/sbin/shutdown -r +5 "Rebooting the OS"
```

6.3.7.2 Recovering an Interrupted Operating System Boot

An incomplete or interrupted operating system boot might leave the console boot drivers in an inconsistent state. In this case, the console displays the following message:

```
Inconsistent boot driver state.
System is configured with multiple partitions.
A complete INIT must be performed before rebooting.
```

Use the following procedure to recover from this condition:

1. Shut down the operating system in all running partitions.
2. Execute the following commands on the primary console:

   ```
   P00>>>set lp_count 0
   P00>>>init
   P00>>>set lp_count N
   ```

   (where `N` is the number of partitions)

```
P00>>>init
P00>>>lpinit
```

3. Boot the operating system in each partition. For example:

```
P00>>>boot
P04>>>boot
```

6.3.7.3 Halting Processors

Under normal operating conditions, it is not necessary to manually halt processors. The processor halts and enters console mode after you shut down the operating system. You must manually halt the processor if the operating system hangs for some reason (for example, while debugging a loadable device driver).

Note

In the unlikely event that the processor cannot be halted the system must be reset by momentarily setting the four way OFF/ENABLE switch to the RESET position, then releasing it.

The following procedures work only if the Power OFF/ENABLE switch is in the ENABLE position.

Primary Partition

Pressing Ctrl/p on the primary console terminal forces the primary processor to enter console mode and display the P##>>> prompt. You can use the `stop` *N* command (where *N* is a processor number) to stop secondary processors (though this is not normally necessary). See Section 6.3.2.1 for definitions of the console prompt and the `stop` command.

Secondary Partitions

Secondary partitions do not halt in response to a Ctrl/p command on the secondary console terminal. To force a secondary partition to enter console mode, use the following proedure:

1. Shut down the operating system on the primary partition:

```
# /usr/sbin/shutdown -h +5 "Shutting down the OS"
```

2. Stop the primary processor of the secondary partition.

```
P00>>>stop N
```

Where *N* is the CPU number of the primary processor of the secondary partitions (normally the lowest numbered CPU assigned to the secondary partition). For example:

```
P00>>>stop 4
```

6.3.7.4 Power OFF/ENABLE Switch Position

During normal system operation, the Power OFF/ENABLE switch is set to the SECURE position. This prevents you from accidentally halting the processor with Ctrl/p.

6.3.7.5 Reconfiguring Partitions by Changing Console EVs

The console EVs that control logical partitions (names begin with `lp_`) must not be changed on any secondary partition. You can change these console EVs only by shutting down all partitions and setting new values on the primary partition's console terminal.

After you have determined the layout of the new partition, follow these steps to reconfigure your partitions:

1. Shut down the operating system in each partition:

   ```
   # /usr/sbin/shutdown -h +5 "Shutting down to
   reconfigure partitions"
   ```

2. Disable partitions and reset the system:

   ```
   P00>>>set lp_count 0
   P00>>>init
   ```

3. Use the console `set` command to change the value of any or all of the console EVs. For the two-partition example discussed in Section 6.3.3.4, use the following commands:

   ```
   P00>>>set lp_count 2
   P00>>>set lp_cpu_mask0 f
   P00>>>set lp_cpu_mask1 f0
   P00>>>set lp_io_mask0 100
   P00>>>set lp_io_mask1 80
   P00>>>set lp_mem_mode isolate
   ```

4. Initialize the primary partition:

   ```
   P00>>>init
   ```

5. Initialize all secondary partitions:

   ```
   P00>>>lpinit
   ```

6. Boot the operating system in each partition using commands similar to the following:

   ```
   P00>>>boot
   P04>>>boot
   ```

6.3.7.6 Checking Other Console EVs Before Booting

Before booting the operating system in each partition, use the console `show` command to verify the correct state of the console EVs:

```
P0##>>>show boot_reset
```

The `boot_reset` EV must be `off`.

```
P0##>>>show interleave
```

The `interleave` EV must be `none`.

```
P0##>>>show auto_action
```

The `auto_action` EV can be set to `HALT` or `BOOT`.

```
P0##>>>show os_type
```

The `os_type` EV is set to UNIX.

6.3.7.7 Logical Partitioning Informational Messages at Boot Time

If you configure and enable logical partitions, the operating system displays informational messages for each partition. These messages appear on the console terminal during the early stages of the bootstrap process. The following example shows typical messages for a two partition system:

```
Partition 0
-----------
LP_INFO: 2 partition(s) established via lp_count
LP_INFO: primary processor for partition 0 is CPU 0
LP_INFO: partition 0 CPU allocation mask = 0xf
LP_INFO: partition 0 IOP allocation mask = 0x100
LP_INFO: Memory partitioning mode set to isolate
LP_INFO: partition 0 memory starting address = 0x0

Partition 1
-----------
LP_INFO: 2 partition(s) established via lp_count
LP_INFO: primary processor for partition 1 is CPU 4
LP_INFO: partition 1 CPU allocation mask = 0xf0
LP_INFO: partition 1 IOP allocation mask = 0x80
LP_INFO: Memory partitioning mode set to isolate
LP_INFO: partition 1 memory starting address = 0x80000000
```

These messages provide the following information:

- Number of active partitions
- Number of the primary processor for the current partition
- Which processors are allocated to the current partition
- Which I/O port modules are allocated to the current partition
- Memory partitioning mode (which is always set to `isolate`)
- Starting address of memory for the current partition

6.3.8 Hardware Management and Maintenance

For the AlphaServer GS140, partitions share a common physical enclosure and hardware (such as power supplies, system bus, and control panel power switch). You cannot perform the following hardware management and maintenance tasks on individual partitions. You must disable partitions and reset the system to a unpartitioned state.

Tasks that require a complete system reinitialization are:

- Performing corrective or preventive maintenance on system hardware.
- Installing AlphaServer GS140 firmware upgrades, including I/O controller firmware upgrades.
- Adding or removing system hardware components (CPUs, memory, IOPs, PCI buses, I/O controllers, and I/O devices [except for hot swappable disks]).
- Changing any partition's hardware resource assignments by modifying any console EV with `lp_` prepended to its name.
- Running the ECU EISA Configuration Utility (ECU) or the Raid Configuration Utility (RCU) from the floppy disk drive.

6.3.8.1 Obtaining Technical Support

If you need to escalate a problem to your technical support organization, it is important that you tell the customer services representative that the system is partitioned (particularly if the service operation uses remote diagnosis). When you place the service call, state that your system is using logical partitions.

The logical partitioning software provides two methods for the customer services representative to determine whether or not a system is partitioned. The `LP_INFO` messages printed during operating system startup are also entered into the binary error log as part of the Startup ASCII Message. You can run the `sizer -P` command on any instance of the operating system to display the partitioning status of the system:

```
# sizer -P
   Host hostname is instance 1 of 2 partitions.
   Physical memory starts at address 0x80000000.
   Memory mode is isolate.
   Processors assigned to instance 1: 4  5  6  7
   IO Port (s) assigned to instance 1: slot 7
```

If the system is not partitioned, the following message is displayed, where *hostname* is the name of the system:

```
Host hostname is not partitioned.
```

6.3.8.2 Performing Hardware Management and Maintenance Tasks

Before performing any management or maintenance tasks, you must terminate operation of all partitions and return the system to an unpartitioned state. Use the following steps to shut down partitions:

1. Shut down the operating system in each partition:

   ```
   # /usr/sbin/shutdown -h +5 "Shutting down for maintenance"
   ```

2. Disable partitions by executing the following command at the primary console terminal:

   ```
   P00>>>set lp_count 0
   ```

3. Set the auto_action console EV for the primary partition to HALT:

   ```
   P00>>>set auto_action halt
   ```

   You might need to reset the auto_action EV in step 1 of the next procedure, initializing and rebooting the partitions.

4. Reinitialize the system by typing this command on the primary console terminal.

   ```
   P00>>>init
   ```

When the system returns to the P00>>> prompt you can perform system management and maintenance tasks. After completing system management and maintenance tasks, use the following procedure to reinitialize and reboot your partitions:

1. Verify the console EVs are set to the correct values:

   ```
   P00>>>show lp*
   P00>>>show boot_reset
   P00>>>show interleave
   P00>>>show auto_action
   ```

   The boot_reset EV is set to off, the interleave EV is set to none, and the auto_action EV is set to either HALT or BOOT.

2. Set the lp_count EV to the correct number of partitions. For example:

   ```
   P00>>>set lp_count 2
   ```

3. Initialize the primary partition:

   ```
   P00>>>>init
   ```

4. Initialize all secondary partitions:

   ```
   P00>>>lpinit
   ```

5. Boot the operating system on each partition. If you changed the system's hardware configuration or reassigned any hardware resources to a different partition, a kernel rebuild might be required. Use the

procedure in Section 6.3.9 to determine if you need to rebuild the kernel for any partition.

If you do not require a kernel rebuild, boot the operating system:

```
P##>>>boot
```

Where ## is the CPU number of the partition's primary processor.

6.3.9 Hardware Changes Requiring a UNIX Kernel Rebuild

If you change your system's hardware configuration you might need to rebuild the kernel. The following table defines the hardware configuration changes that require a rebuilt kernel:

Change	Requirements
Processors – adding, removing, or reassigning CPU modules.	Changing the `lp_cpu_mask#` EV for any partition does not require a kernel rebuild. You must assign to the same partition both processors on a dual CPU module.
I/O Processors – adding, removing, or reassigning IOP modules.	Rebuild the kernel if you added or removed an IOP module. You need only rebuild the kernel for the changed partition). Moving an IOP module across partitions requires a kernel rebuild on both partitions. The `lp_io_mask#` EV assigns IOP modules.
	Adding or removing I/O buses and I/O controllers requires a kernel rebuild for the affected partition.
Memory Modules – changing the memory module configuration.	For the primary partition (partition 0), changes to the memory module configuration do not require a kernel rebuild.
	The kernel for any secondary partition must be built to run at a specific memory address (that is, the physical memory starting address for the partition). Certain types of memory reconfiguration change this address and require a kernel rebuild. A partition's memory starting address changes if the memory size for any lower numbered partition increases or decreases.
	For example, if you replaced a 2GB memory module in partition zero with a 4GB module, the memory starting address of partition one increases by 2GB. In this example you must rebuild the kernel.

Rebuild the kernel if a secondary partition's kernel fails to boot after a memory module configuration change.

The memory starting address for each partition is displayed at the primary console after each iteration of the `P00>>>lpinit` command.

6.3.9.1 How to Rebuild the UNIX Kernel for a Partition

The following steps describe how you rebuild the kernel, which is a special case of the typical kernel build instructions documented in the *System Administration* manual. This procedure assumes that you initialized partitions as described in Section 6.3.4 and the partition requiring a kernel rebuild is halted at the `P##>>>` console prompt. Refer to the kernel configuration information in the *System Administration* manual for information on:

- Kernel booting and the single-user mode prompt
- Saving and copying kernels

1. Boot the generic kernel to single-user mode:

   `P##>>>`**`boot -fl s -fi genvmunix`**

2. Check and mount file systems:

   `#` **`/sbin/bcheckrc`**

   Refer to the *System Administration* manual for more information on mounting file systems.

3. Set the host name (system name) for this partition:

   `#` **`hostname`** *`NAME`*

4. Rebuild the kernel:

   `#` **`doconfig`**

   Note

   You must not use `doconfig` with the `-c` option to rebuild the kernel.

5. Save the current kernel:

   `#` **`cp /vmunix /vmunix.save`**

6. Install the new kernel, where *SYSNAME* is the local host name:

   `#` **`cp /sys/`*`SYSNAME`*`/vmunix /vmunix`**

7. Unmount the file systems:

```
# umount -a
```

8. Halt the operating system:

```
# sync
# sync
# halt
```

9. Boot the new kernel:

```
P##>>>boot
```

6.3.10 Handling Nonrecoverable Hardware Error Machine Checks

There are two main classes of hardware errors:

- Recoverable errors are corrected by the hardware and reported to the operating system. The operating system logs recoverable errors in the binary error log and continues normal system operation. Nonrecoverable hardware errors require immediate termination of normal system operation and some form of corrective action (such as a system reset).
- Nonrecoverable hardware errors are reported to the operating system as a machine check. The operating system crashes with a panic message, such as the following:

  ```
  panic (cpu 0): tlaser: \
  MACHINE CHECK Non-recoverable hardware error
  ```

  The system then writes out a crash dump, and reboots or halts (depending on the setting of the `auto_action` console EV, which can be BOOT or HALT).

Some hardware errors require a complete system reset before the operating system can be rebooted.

For system-wide hardware faults, the operating system forces a system reset after writing the crash dump. After the reset is completed, if `auto_action` is set to BOOT, the console firmware automatically reinitializes all partitions. Boot the operating system in each partition, using the following commands:

```
P00>>>boot
P##>>>boot
```

Otherwise, the system halts and enters console mode (`P00>>>` prompt). If this occurs, enter the following commands to restart partitions and reboot the operating system (where *N* is the number of partitions):

```
P00>>>set lp_count N
P00>>>init
P00>>>lpinit
P00>>>boot
```

For each secondary partition, enter the boot command:

```
P##>>>boot
```

For local hardware faults (contained within a partition), the operating system running in the affected partition unconditionally halts after writing the crash dump. This allows other partitions to continue operating until a shut down can be scheduled. Restarting the affected partition requires a complete system reset, using the following procedure:

1. Shut down the operating system in each running partition:

   ```
   # /usr/sbin/shutdown -h +5 "Shutting down for error recovery"
   ```

2. At the primary console terminal, enter the following commands:

   ```
   P00>>>set lp_count 0
   P00>>>init
   ```

3. The console displays the following prompt:

   ```
   Do you really want to reset ALL partitions? (Y/<N>)
   ```

   Type Y to perform the reset. After the reset is complete, and if `auto_action` is set to BOOT, the console firmware automatically reinitializes all partitions.

4. Boot the operating system in each partition, using the following commands:

   ```
   P00>>>boot
   P##>>>boot
   ```

   Otherwise, enter the following commands (where *N* is the number of partitions):

   ```
   P00>>>set lp_count N
   P00>>>init
   P00>>>lpinit
   P00>>>boot
   ```

   For each secondary partition enter the following:

   ```
   P##>>>boot
   ```

If these recovery procedures fail to restore full system operation for all partitions, reset the system manually by momentarily moving the OFF/ENABLE switch to the RESET position, then releasing it. Repeat the recovery procedure after the reset completes. If the failure persists, contact your technical support organization.

6.3.11 Logical Partitioning Error Messages

If an error condition occurs (such as an invalid partition configuration) the partition's console terminal displays an error message. After displaying the error message, the primary processor for the current partition halts and returns to the console prompt. To recover from any of these errors, correct the logical partitioning console EVs and reboot the partition.

The following error messages might be displayed:

```
LP_ERROR: invalid partition count (lp_count = #, max nodes
= #)
```

The `lp_count` console EV is set incorrectly. The value is less than zero or exceeds the maximum number of partitions supported for the AlphaServer GS140.

```
LP_ERROR: no CPUs for partition (check lp_cpu_mask)
```

The value of `lp_cpu_mask#` (# represents the current partition number) is set incorrectly. This partition has no allocated processors.

```
LP_ERROR: no IOP for partition (check lp_io_mask)
```

The value of `lp_io_mask#` (# represents the current partition number) is set incorrectly. This partition has no allocated I/O Port modules.

```
LP_ERROR: lp_count > 1, but partitions not initialized
Please execute 'lpinit' command at >>> prompt
```

The message indicates that partitions were configured, but not initialized.

```
LP_ERROR: must set lp_mem_mode [share or isolate]
```

The `lp_mem_mode` console EV is not set or set incorrectly. For logical partitions, `lp_mem_mode` must be set to `isolate`.

```
Bootstrap address collision, image loading aborted
```

The kernel's link address does not match the memory starting address of the partition. Refer to Section 6.3.9 for instructions on how to recover from this error.

6.3.12 Understanding Console Firmware Error or Informational Messages

The console firmware implements several safety checks during certain events (such as system reset and partition startup). These checks help

prevent cross-partition interference. The partition's console displays one of the following messages if an anomaly is detected:

```
Do you really want to reset ALL partitions?  (Y/<N>)
```

This message is displayed after a system reset is requested, either by the operation issuing the `init` command or as a result of booting with the `boot_reset` console EV set to ON. This message warns you that continuing with the operation will terminate all partitions and reset he system. If a reset is necessary, shut down the operating system in all operational partitions before proceeding with the reset.

```
Auto-Starting secondary partitions...
```

This message indicates the console firmware is initializing logical partitions (by running the `lpinit` command automatically). An auto-starting event occurs after a system reset (or power on). The console firmware boots the operating system in all partitions provided that:

- The `auto_action` console EV is set to BOOT
- You initiate the reset by using the RESET switch on power-on, and not by using the `init` command

```
Insufficient memory interleave sets to partition system.
Issue command "set interleave none" then reset system.
```

This message indicates that the interleave console EV is incorrectly set. Change the setting to `none`.

```
Insufficient memory modules to partition system.
```

Each partition requires a dedicated memory module. Reduce the number of partitions or install a memory module for each partition.

This message indicates that the `lp_count` console EV might not be set correctly. For example, you have two partitions, but `lp_count` is set to four. In this case, set `lp_count` to match the actual number of partitions.

```
Inconsistent boot driver state.
System is configured with multiple partitions.
A complete INIT must be performed before rebooting.
```

An incomplete or interrupted operating system boot caused the console boot drivers to enter an inconsistent state. Refer to Section 6.3.7.2 for instructions on recovering from this state.

```
Do you want to attempt to boot secondary partitions anyway?
(Y/<N>).
```

This message indicates that the console detected an inconsistency in your partitions set up (probably due to incorrect setting of `lp_` console EVs). Unless you are certain it is safe to proceed, answer no (`N`) to this question and correct the inconsistency.

```
TIOP # not configured in any partition.
Non-existent TIOP # configured in a partition.
```

These messages (together or separately) indicate incorrect setting of the `lp_io_mask#` console EV. The mask might be set to zero or to the wrong IOP module slot number. Correct the setting and retry the `lpinit` command.

```
Secondary partitions have already been started.
```

This message most likely indicates you issued a second `lpinit` command after starting partitions. Before booting the operating system, check the values of the `lp_` console EVs.

```
CPU # not configured in any partition.
No valid primary processor specified for partition #.
```

In this message, the CPU number (#) might be a single CPU or a list of CPUs.

These messages (together or separately) indicate incorrect setting of the `lp_cpu_mask#` console EV. The mask might be set to zero or to incorrect CPU numbers. Correct the setting and retry the `lpinit` command.

6.4 AlphaServer 1000 and 1000A Configuration Information

The following configuration restrictions are specific to AlphaServer 1000 and 1000A systems.

6.4.1 EISA Configuration Utility Version 1.10

This note applies to users of the embedded Cirrus VGA graphics controller.

The default setting for the VGA graphics controller when running the EISA Configuration Utility (ECU) Version 1.10 is `Disabled`. For previous versions, the default is `Enabled`.

When you run the ECU Version 1.10 for the first time on a system that was previously configured with an earlier version of the ECU, the setting for the embedded VGA graphics controller is automatically set to `Disabled`. To change the default value, run the ECU, select Step 3: View and edit details,

and set the VGA graphic controller to `Enabled` before exiting. If you do not set the VGA graphic controller to `Enabled` prior to booting the operating system, your X server will not start and your system will have generic console support when you boot the operating system.

6.4.2 Graphics Resolution

The default graphics resolution for AlphaServer 1000A systems that contain built-in Cirrus video with 1 MB of video RAM is 1024x768. If the optional 512 KB of video RAM is not present, the operating system supports resolutions of 640x480 (by default) or 800x600 only.

The default graphics resolution for AlphaServer 1000 systems that contain built-in Cirrus video with 512 KB of video RAM is 640x480. This configuration also supports 800x600 resolution.

To use 800x600 resolution, edit the following line in the `/usr/lib/X11/xdm/Xservers` file:

```
:0 local /usr/bin/X11/X
```

Change the line to:

```
:0 local /usr/bin/X11/X "-screen0 800x600"
```

To use 800x600 resolution for the CDE Session Manager, edit the following line in the `/usr/dt/config/Xservers` and `Xservers.conf` files:

```
:0 Local local@console /usr/bin/X11/X :0
```

Change the line to:

```
:0 Local local@console /usr/bin/X11/X :0 -screen0 800
```

Before editing these files for XDM or CDE, be sure that your system's monitor supports 800x600 resolution.

6.5 AlphaServer GS-series Configuration Information

The following configuration restrictions are specific to AlphaServer GS systems.

6.5.1 Possible OLAR Errors on Primary CPU

Starting with Version 5.1A of the operating system, the primary CPU is capable of being taken off line and removed on AlphaServer GS80, GS160, and GS320 systems. When you take the primary CPU off line, another CPU is automatically delegated the role of primary CPU. Due to intermittent problems with this operation, do not take the primary CPU off line at this time. The primary CPU is normally CPU0. To verify which CPU is currently the primary, use the `pset_info` command:

```
# pset_info
number of processor sets on system = 1

pset_id  # cpus   # pids   # threads  load_av    created
  0         4        89       453       0.13     07/12/2001 17:25:28

total number of processors on system = 4

cpu #    running  primary_cpu  pset_id  assigned_to_pset
 0          1         1           0     07/12/2001 17:25:28
 1          1         0           0     07/12/2001 17:25:28
 2          1         0           0     07/12/2001 17:25:28
 3          1         0           0     07/12/2001 17:25:28
```

This output indicates that CPU0 is the primary CPU.

When you attempt to take the primary CPU off line, the operation will likely succeed, and a new primary CPU (for example CPU1) is automatically selected. However, when subsequently attempting to bring the previously assigned primary CPU back on line, you may encounter the following error:

```
Processor X failed to start
 console callback PARTITION, POWER-HW timed_out.
 wf_hal_pwr_ctl: call to prom_power failed with status [ffffffe6]
```

To clear this condition, you must initialize the system using the SRM console. This action requires shutting down the operating system. This problem will be fixed in a future release of the console firmware.

6.5.2 Do Not Repetitively Power Cycle CPUs

The use of OLAR management commands in continuous test loops can degrade the reliability of the CPU. These commands remove the DC power source from the CPU module. We recommend that you do not power cycle CPUs repetitively with shell scripts.

The CPU module's DC-to-DC converter is specified to have a maximum of 1000 power cycles. Do not exceed his number of power cycles for a CPU.

6.5.3 Hot Add Restriction

This release of the operating system supports GS80, GS160, and GS320 CPU hot additions with the following restriction: A Quad Building Block (QBB) booted without memory and without at least one CPU cannot have CPUs hot added to that QBB. Doing so will result in a system panic. If the target QBB is booted with memory and at least one CPU, additional CPUs can be hot added as desired. (This restriction will be lifted in a future kernel update.)

6.6 Personal Workstation 433au, 500au, and 600au Systems

The following configuration restrictions are specific to Personal Workstation class systems.

6.6.1 64-Bit PCI Option Cards

The 64-bit PCI slots, slots 4 and 5, are intended only for those cards listed in the *Systems and Options Catalog* as supported for slots 4 and 5. The console prevents system operation and displays the following error if an unsupported card is present in one of these slots (*n*):

```
Illegal device detected on primary bus in physical slot n
Power down the system and remove the unsupported
device from slot n
```

6.6.2 Incorrect Default Keyboard Mappings

If you use a PCXLA-NA keyboard on a Personal Workstation 433au, 500au, or 600au class system, the keys will not map properly unless you reconfigure the keyboard driver to use the correct keymaps.

You can do this by executing the following command:

```
# sysconfig -r gpc_input kbd_scancode=2
```

If you prefer, you can use the `sysconfigdb` command to add the following entry to the `/etc/sysconfigtab` file:

```
gpc_input:
kbd_scancode = 2
```

If you use the `sysconfig` command to reconfigure the driver, you must execute the command each time you reboot the system. Using the `sysconfigdb` utility to make the change preserves the information across reboots, and no other user intervention is required.

7

Managing Specific Device Types

Most information that you use to install and configure a device is provided in the owner's manual that is shipped with the device itself. The owner's manual contains installation instructions for the device and specifies any special configuration requirements or usage restrictions for releases of the Tru64 UNIX operating system. This information is often supplemented in the *Release Notes* manual for a given release of the operating system.

The *Release Notes* describe any known problems or temporary restrictions on using a device. This chapter provides information on configuration options, permanent configuration requirements, or usage restrictions and discusses the following topics:

- Managing storage (Section 7.1)
- Managing host bus adapters (Section 7.2)
- Managing graphics adapters (Section 7.3)

Note

This chapter describes how to configure certain devices and operating system options. Some of these devices are platform-dependent. That is, the device is only supported on certain models of processor. Similarly, some operating system features are also restricted to certain models of processor. Other operating system options depend on your configuration. Consult your system's hardware documentation for more information.

Follow the instructions in the *Installation Guide* and those provided by your hardware and firmware documentation when you add new options or change your system hardware. However, if the new option is supported only in the newest version of the operating system, you must perform the upgrade in the following sequence:

1. Update your operating system software.
2. Upgrade your firmware.
3. Upgrade your hardware or install the new option.
4. Follow the instructions in the Tru64 UNIX *Installation Guide* to rebuild your system kernel.

7.1 Storage Devices

This section provides specific information about configuring and managing storage devices and storage arrays under Tru64 UNIX. The following topics are included:

- The parallel scanning feature, which enables systems that have large storage configurations to boot faster (Section 7.1.1)
- Console-level multipath support restrictions (Section 7.1.2)
- Fibre Channel system configurations that properly identify the boot and swap devices required to obtain crash dumps (Section 7.1.3)
- Ensuring that disk labels are usable after a disk is moved out of a RAID array (Section 7.1.4)
- Replacing failed HSZ40 and HSZ50 controllers (Section 7.1.5)
- Replacing a failed HSZ70 controller (Section 7.1.6)
- Clearing persistent reservations on disks behind HSZ80 and HSG80 controllers (Section 7.1.7)
- Restrictions on using the CD burner in place of the floppy disk drive (Section 7.1.8)
- Configuring the floppy disk drive for use with UFS or DOS-formatted floppies (Section 7.1.9)

Notes

You must use the StorageWorks Command Console (SWCC) Version 2.3 or higher to manage HSZ or HSG controllers. Versions prior to Version 2.3 are not supported for use on Tru64 UNIX Version 5.0, or higher.

Some HSZ-series controller hardware is retired in Tru64 UNIX Version 5.1B. See the *Release Notes* for information on retiring hardware options and see the online

7.1.1 Parallel Scanning

Parallel scanning is a new feature in Tru64 UNIX Version 5.1B that is designed to shorten the system's boot time. Parallel scanning is disabled in the default system configuration. To enable this feature you must set the value of a configuration variable, as described later in this section.

When you enable parallel scanning of SCSI and Fibre Channel buses, the system initiates the scan on all buses simultaneously (rather than sequentially). This feature reduces the time required to find devices. On

systems with moderate to large numbers of storage devices, the reduction of time required to boot can be significant.

However, parallel scanning affects the assignment of device names to newly added devices. Because the scan probes all buses simultaneously, the names assigned to new devices appear to be in random order. The system derives these device names from the numerical order in which it discovers devices during the scan.

For example, a storage array might contain disks that are named `/dev/disk/dsk8` through `/dev/disk/dsk32`. When you add a new device, its device name might be named `/dev/disk/dsk159`, depending on when it is discovered by the scan. This is only a problem if you want to name the devices in your storage configuration in a contiguous sequence.

Consequently, if you require a specific device name sequence you must turn off parallel scanning before the system scans any newly added devices. You can turn parallel scanning on or off by using several methods. Depending on how it is set, the setting remains in force until you reset it or until you reboot the system. Use the following procedure to temporarily turn parallel scanning on or off:

1. Shut down the system.

   Refer to the *System Administration* manual and `shutdown`(8) for information on your shutdown options.

2. Install the new host bus adapter and the new devices.

3. Reboot the system interactively by using the following console command:

```
P0>>> boot -flag ai
UNIX boot - Friday December 15, 2000

Enter: <kernel_name> [option_1 ... option_n]
or: ls [name][help] or: quit to return to console
Press Return to boot vmunix
```

   The system displays the interactive prompt (#).

4. Disable parallel scanning (for this boot only) by using the following command:

```
# vmunix io:parallel_edt_scan=0
```

5. When the system is up and running you can turn off parallel scanning by using the following command:

```
# sysconfig -r io_parallel_edt_scan=0
```

When the system is up, and if you want certain bus events (such as bus resets) to be processed faster, turn on parallel probing by using the following command:

```
# # sysconfig -r io parallel_edt_scan=1
```

You can turn parallel probing on permanently (the setting is retained across a system boot) by using the `sysconfigdb` command or the `dxkerneltuner` utility. Use the following procedure:

1. As root user, create the following `io` subsystem stanza file:

   ```
   io:
   parallel_edt_scan=1
   ```

   Name this file `parallel.stanza`. If an `io` stanza already exists, add the preceding line to the existing file, instead of creating a new file.

2. Determine whether any entries exist for the `io` subsystem by using the following command:

   ```
   # sysconfigdb -l io
   ```

   The results of this step determine which command option (`-a` or `-u`) to use in Step 3.

3. Configure parallel probing by using one of the following command variants:

   - To add the entry to the target file, use the following command:

     ```
     # sysconfigdb -a -f parallel.stanza io
     ```

   - To replace an existing `io` subsystem entry with the subsystem entry specified in the stanza, use the following command.

     ```
     # sysconfigdb -u -f parallel.stanza io
     ```

7.1.2 Console-Level Multipath Support Restrictions

The console firmware on the AlphaServer 1000, AlphaServer 1000A, and AlphaServer 2x00 systems does not support selecting multiple boot or dump devices for storage units located behind HSZ70, HSZ80, or HSG80 RAID Array controllers that are enabled for multiple-bus failover mode.

The console must have a visible path to the storage unit that it is booting from or to which it is dumping. If a controller, in multiple-bus failover mode, fails over to the other controller, all devices served by the failed controller are now visible on alternate paths. Therefore, before booting the system, reset the `bootdef_dev` console environment variable to a path that is visible to the boot device.

After the operating system has been booted, multipath support is fully functional.

7.1.3 Fibre Channel Dump Configuration

If your system configuration includes Fibre Channel devices, its console ports must be configured with the unique world-wide identifier (WWID) of the boot and swap disks defined for the operating system. This configuration is required to enable crash dumps.

The system requirements and configuration settings are defined in the following procedure:

1. The system must have the swap space configured to minimum of 1.25GB to twice the size of its physical memory. For example, a minimum of 5GB of on-disk swap space for 4GB of physical memory. Refer to the *System Administration* manual for information on how to add swap partitions and configure swap space.
2. All the boot and swap devices must be configured to use one of the four console ports. Ensure that the console WWID number for each boot or swap device is identical to the WWID number displayed by using the following procedure:

   a. Identify the console port (N*n*) and WWID*n* mapping by using the `consvar` command as follows:

      ```
      # consvar -g N1
      # consvar -g wwid0
      ```

      Repeat the command for each of the four ports and WWID values.

   b. For each Fibre Channel boot and swap device, find the device name (`dev/disk/dsk`*N*). You can do this by examining the `/etc/fstab` file, and by using the `showfdmn` for AdvFS root domains, and the `swapon -s` command for swap devices.

      Alternatively, use the SysMan Station as described in the *System Administration* Manual.

   c. For each Fibre Channel boot and swap device, find the device's hardware identifier (HWID) by using the device name that you obtained in the preceding step. For example:

      ```
      # /sbin/hwmgr view devices | grep devname
      ```

   d. Obtain the WWID of each Fibre Channel boot and swap device by using the following command, specifying its HWID:

      ```
      # /sbin/hwmgr show scsi -id HWID -full
      ```

   e. For each Fibre Channel boot and swap device, verify that the WWIDs displayed by the `hwid` command match the WWIDs that are mapped to the console ports.

3. If the device WWID and the console port WWID values do not match, use the following procedure:

   a. Shut down the system by using the following command:

      ```
      # /sbin/shutdown +5 "system going down for
      console reconfiguration"
      ```

   b. Use the `wwidmgr` utility as described in the *Wwidmgr Users Manual* to match the console WWID settings to the device WWIDs. The manual is located in the `../documentation` directory on the system's Firmware CD-ROM.

7.1.4 Moving Disks Out of an Array

A disk's label might be unusable if a new LSM unit is constructed on a RAID array or if you move one or more disks from a RAID array to a direct connection under a host bus adapter. This might cause the disk's new block zero to be the same block as the old block zero for the prior unit. The system might see this incomplete label as valid, even though it might extend beyond the end of the disk. Use the following procedure to correct the disk label:

1. Use the following command to zero the disk's label:

   ```
   # disklabel -z dskNN
   ```

2. Use either of the following commands to create a new default label, or to apply a preconfigured label from a proto file:

   ```
   # disklabel -rwn dskNN
   # # disklabel -Rr dskNN PROTOFILE
   ```

7.1.5 Replacing Failed HSZ40 and HSZ50 Controllers

This section describes how you replace a failed dual redundant controller for the following dual unit models:

- HSZ40 and HSZ50, which can be in either single or transparent failover mode. They cannot be in multibus mode.
- HSZ70, which can be in single, transparent, or multibus failover mode. Configurations that are not dual redundant use the same procedure as described for single-spindle disks.

If a controller fails, use the following procedure to replace the failed component and restart your configuration.

Caution

Several cautions are specified in the documentation for the controller's HSOF commands. To avoid the risk of data loss,

familiarize yourself with the following CLI commands before proceeding:

- `SET FAILOVER COPY=configuration-source`
- `SET NOFAILOVER`
- `SET MULTIBUS_FAILOVER COPY=configuration-source`

The Tru64 UNIX operating system must not be at the console prompt (`P0>>>`). If it is, boot the system to single-user mode before applying the following procedure:

1. Using the HSOF CLI on the good (functioning) controller, enter the following command to stop the failed controller:

   ```
   HSZnn> set nofailover
   ```

2. Verify that the failed controller has stopped by ensuring that the green LED status light is no longer blinking.
3. Physically remove the failed controller as described in the hardware documentation.
4. At the UNIX command prompt, enter the following command:

   ```
   # /sbin/hwmgr scan scsi -bus N
   ```

   The `-scan` option is asynchronous. When you issue this command, the prompt returns immediately, although the scan can still be working in the kernel, particularly if many devices are connected to the system. To test for discovery of the failed device, use the Event Manager (EVM) `evmwatch` command to monitor for hardware events as follows:

   ```
   # evmwatch -A -f '[name sys.unix.hw.*]'
   ```

   The presence of tape devices increases the delay to complete the scan. Use the `-bus` qualifier to specify the bus you want to scan. Determine the correct bus number by examining the location field of the output from the following command:

   ```
   # hwmgr view devices
   ```

   The display shows the HSZ units going from dual to single configuration.
5. Install a replacement controller as described in the hardware documentation.
6. Start the replacement controller as described in the hardware documentation.
7. Repeat the `hwmgr -scan scsi` command described in Step 4.

8. At the good controller, enter either of the following HSOF CLI commands to return the controllers to dual redundant mode in the appropriate failover mode:

```
> set failover copy=this
> set multibus_failover copy=this
```

Messages are displayed indicating that the HSZ units are going from single to dual configuration, confirming that the operation was successful.

7.1.6 Replacing a Failed HSZ70 Controller

Some installations require a very strict recover or fail procedure, causing a HSZ70 controller to failover in a very short time when the system cannot recover the I/O path. However, this short failover time can cause problems if you want to change a running configuration, such as replace a failed controller. To avoid the recover or fail restriction you must temporarily set a system configuration parameter, causing the system to use a longer recovery period.

The following restrictions apply:

- Transparent failover configurations cannot be put into a mode of operation that causes them to recover or fail. Recover or fail mode applies only to multibus configurations. The HSZ70 controller supports such multibus configurations. The HSZ80 controller supports multibus configurations but is not affected by short recovery times.
- You can configure HSZ40 and HSZ50 controllers only in either single or transparent failover mode. HSZ40 and HSZ50 controllers are therefore unaffected by short recovery times. Use the published procedures for managing your controller configuration, such as that documented in Section 7.1.5.

The following procedure describes how to reset the recovery period during a controller swap operation for a HSZ70 in multibus failover mode. The HSZ70 controller replacement procedure may create phantom entries for devices if the replacement is performed during times of heavy system I/O to the controller pair involved in the replacement. To avoid this problem, reduce or eliminate I/O to the controller pair before replacing a controller. (See the HSZ70's hardware documentation concerning the port quiesce buttons on the controller.)

Caution

Several data loss warnings are specified in the documentation for the controller's command line interface (HSOF). To avoid the risk of data loss, familiarize yourself with the following HSOF commands before proceeding:

- SET FAILOVER COPY=*configuration-source*
- SET NOFAILOVER
- SET MULTIBUS_FAILOVER COPY=*configuration-source*

---

1. At the Tru64 UNIX system prompt, enter the following command to switch to alternate recovery timing during the subsequent steps:

   ```
   # /sbin/sysconfig -r cam_disk rec_use_alt_params=1
   ```

2. Set up an Event Manager (EVM) background task to monitor for hardware events by using the following command:

   ```
   # evmwatch -A -f '[name sys.unix.hw.*]'
   6771
   ```

   In later steps, this task will display hardware events resulting from `hwmgr` commands. Make a record of the process ID assigned to the task, which is `6771` in the preceding example.

3. Using the HSOF command line interface on the good (functioning) controller, enter the following command to stop the failed controller:

   ```
   HSZ70> set nofailover
   ```

   When its green LED status light is no longer blinking, the failed controller has stopped and you can proceed with the next step.

4. Remove the failed controller as described in its hardware documentation.

5. At the Tru64 UNIX system prompt, enter the following command:

   ```
   # /sbin/hwmgr scan scsi
   ```

   Messages displayed at the console will show the HSZ70 controller pair going from dual to single configuration.

   The `scan` option is asynchronous. When you issue this command, the prompt returns immediately, although the scan can still be working in the kernel, particularly if many devices are connected to the system. Your background EVM task displays any messages that relate to the registration of the replacement controller.

   To make the scan complete as quickly as possible, use `hwmgr` commands to determine the system bus at which the controller persists, and specify a scan of that bus alone. Determine the correct bus number by examining the location field of the output from the following command:

   ```
   # /sbin/hwmgr view devices
   ```

6. Install and start the replacement HSZ70 controller as described in its hardware documentation.

7. At the Tru64 UNIX system prompt, repeat the `scan` command as follows:

   ```
   # /sbin/hwmgr scan scsi
   ```

8. At the good controller (not the controller that you just replaced), enter the following HSOF command:

   ```
   hsz70> set multibus_failover copy=this
   ```

   This command returns the controller pair to dual redundant mode in the appropriate failover mode.

9. At the Tru64 UNIX system prompt, enter the following command to switch to default recovery timing:

   ```
   # /sbin/sysconfig -r cam_disk rec_use_alt_params=0
   ```

When the HSZ70 controller pair goes from single to dual configuration, messages displayed at the console verify that the operation was successful.

7.1.7 Clearing Persistent Reservations on Disks Behind HSZ80 and HSG80 Controllers

TruCluster Server cluster members use persistent reservations to coordinate access to shared storage. These persistent reservations control access to devices and logical volumes, and are used to erect a barrier against any system that is not a member of the current cluster. The base operating system does not know how to manage these persistent reservations. Attempts by the base operating system to access a disk that has a persistent reservation will return I/O error messages. All I/O attempts will fail.

If you boot (or reinstall) the base operating system on a system that attempts to access disks that were previously used by a cluster behind an HSZ80 or HSG80 controller, the base operating system might not be able to see the disks. This is most likely because the disk participated previously in a cluster, and one of the cluster members set a reservation on the disk to prohibit any other hosts (or noncluster members) on this shared bus from accessing the disk. The problem occurs because the reservation was not cleared when the cluster member or the cluster was shut down.

To remove the reservations, use the `/usr/sbin/cleanPR clean` command. The `cleanPR` script finds and clears all persistent reservations from the attached HSZ80 and HSG80 devices.

Caution

Do not run the `cleanPR` script on a system that is in a cluster. You do not want to remove reservations that the running cluster

is using to actively control disk access. Doing so can result in data corruption.

Consider the following two scenarios:

- In the first scenario, you can boot a disk containing the base operating system and run the `cleanPR` command.

  If you encounter persistent reservations when attempting to access storage that was previously used in a cluster, boot the operating system and enter the `/usr/sbin/cleanPR clean` command to clear the reservations.

- In the second scenario, you are trying to install the base operating system on a disk that is behind an HSG80 or HSZ80 controller, has a persistent reservation on it, and does not have an installed version of the base operating system.

  This scenario is more complex because you do not have a running version of the operating system from which you can run the `cleanPR` script. However, the `cleanPR` script is on the Tru64 UNIX Operating System CD–ROM. Therefore, you can start the installation from CD–ROM, get to the single-user shell, and enter the `/usr/sbin/cleanPR clean` command to clear the reservations.

7.1.8 Compact Disk (CD-R/W) Burner Option

Some systems are shipped with a CD-R/W burner as a console storage device in place of both the CD-ROM reader and the 1.44MB floppy disk device. Unlike the floppy disk device, the CD-R/W device is not available for console operations, such as saving copies of your system configuration at the console. However, in multiuser mode you can burn files onto recordable CD-R and CD-RW media.

To burn a CD-R/W, see `cdrecord`(1) and `mkisofs`(8). A description of the procedure is provided in the online Best Practice *Recording a Data CD-ROM* at the following web location:
`http://www.tru64unix.compaq.com/docs/best_practices`

The following constraints apply to systems equipped with the CD-R/W device instead of a floppy disk device:

- Any procedures that require the use of the floppy drive do not apply to systems equipped only with a CD-R/W device.
- You cannot use the `mtools` command options and the CDE `dxmtools` utility to read and write DOS-formatted floppies.
- See `mount`(8) for command options that apply when you mount CD-ROMs.

7.1.9 Floppy Disk Configuration

The following procedure describes how to configure a floppy disk for use with Tru64 UNIX:

1. Use the following commands to determine the type of floppy drive and its device special files:

```
# hwmgr view devices | grep floppy
69: /dev/disk/floppy0c           3.5in floppy     fdi0-unit-0
```

   The preceding command output indicates that the single floppy drive is an fdi-attached device rather than a SCSI device. Its device special files are named `floppy0c`. However, when formatting the drive, you use its `a` partition (`floppy0a`).

2. Insert a blank diskette into the drive and enter the following command to format the diskette:

```
# /sbin/fddisk -fmt -f /dev/rdisk/floppy0a
Disk type: 3.50 inch, HD  (1.44MB)
Number of sectors per track: 18
Number of surfaces:   2
Number of cylinders: 80
Sector size:  512
interleave factor:  2:4
Formatting disk...
Percentage complete: Format complete, checking...
Quick check of disk passes OK.
```

3. Write a disk label to the floppy:

```
# disklabel -rw /dev/rdisk/floppy0a
/dev/rdisk/floppy0a:
2880 sectors in 80 cylinders of 2 tracks, 18 sectors
1.4MB in 5 cyl groups (16 c/g, 0.28MB/g, 64 i/g)
super-block backups (for fsck -b #) at:
32, 640, 1184, 1792, 2336,
```

4. Create a UFS file system on the floppy:

```
# newfs  /dev/rdisk/floppy0a
```

5. Mount the floppy for use:

```
# mount /dev/disk/floppy0a /mnt
```

   The preceding command assumes that the `/mnt` directory exists and is currently unused. If a mount directory is unavailable, create one by using the following command:

```
# mkdir /floppy_disk
```

See fddisk(8) for more information.

You can also access DOS-formatted floppies by using the mtools commands provided the appropriate software subsets are installed and the programs exist in the /usr/ucb directory. See the *Installation Guide* for information about the optional software subsets.

Before using the floppy with the mtools commands, you must create a target device as follows:

1. Your /dev/disk directory must contain the device special files for two floppy disk partitions named /dev/disk/floppy*N*a, and /dev/disk/floppy*N*c. If these files do not exist, create them by using the dsfmgr -n command
2. Link the floppy's c partition to the mtools target file as follows:

   ```
   # ln -s /dev/disk/floppy0c /dev/disk/floppy
   ```

3. To test the configuration of the diskette drive, insert a DOS-formatted disk and enter the following command:

   ```
   # /usr/ucb/mtools/mdir
   Volume in drive A is "volume_name."
        Directory for A:/

        file type size date time
        file type size date time
   ```

You can use the commands located in the /usr/ucb/mtools directory to manage floppy diskettes in MS-DOS format. See mtools(1) for information about configuring SCSI-attached floppy drives.

7.2 Managing Host Bus Adapters (HBAs)

This section provides information on managing specific host bus adapters (HBAs), such as SCSI adapters. It describes how you can manage and obtain information on HBAs. It also includes information on configuring or operating specific models of HBA and documents any configuration restrictions.

- KZPSA on AlphaServer 1000A and 2100A systems (Section 7.2.1)
- KZPBA (Qlogic ISP1040B) CAM errors (Section 7.2.2)

7.2.1 KZPSA Firmware on AlphaServer 1000A and 2100A Systems

The KZPSA is a PCI to Single Channel, FWD, SCSI-2 Adapter. On AlphaServer 1000A and 2100A class systems, updating the firmware on a KZPSA SCSI adapter is not supported when the adapter is installed behind

the PCI-to-PCI bridge. See your hardware installation manual for further information.

7.2.2 KZPBA (Qlogic ISP1040B) Configuration

On systems with a Qlogic ISP1040B option, CAM errors similar to the following might occur when you boot the system:

```
pci2000 at pci0 slot 8
isp0 at pci2000 slot 0
isp0: QLOGIC ISP1020A
cam_logger: CAM_ERROR packet
cam_logger: bus 0
isp_probe
NVRAM parameters invalid, using driver Fast10 defaults
```

To correct the error, you must use the `eeromcfg` utility to program the NVRAM with the proper set of parameters. The `eeromcfg` utility is provided in the */mnt-pnt/*`utility` directory of the *Alpha Systems Firmware Update* CD–ROM. Consult the `readme.txt` file in that directory for information about how to use the utility.

Note

The KZPBA SCSI controller has two variants. KZPBA-CA/CX is a Qlogic ISP1020/1040 Single Ended PCI to UltraSCSI single channel controller. KZPBA-CB/CY is a Qlogic ISP1020/1040 Differential PCI to UltraSCSI controller. Both controllers are identified as Qlogic ISP1020/1040 controllers at the SRM console.

To identify which controller is in the system, look for a symbol near the external connector 68 pin (HD). The marking on the left hand side of the external cable connector `-< >>` , indicates that the controller is an ISP1020/1040 Differential controller. The Single Ended PCI controller is marked `-< >` .

7.3 Graphics Adapters

This section describes how to obtain information about the graphic adapters installed on your system. It also describes how to configure certain models of graphic adapter.

Note

Ensure that your video monitor supports the settings before you make any of the optional changes referenced in this section. For monitors that are listed as supported in the Tru64 UNIX

QuickSpecs, consult the section of your owner's manual that is titled *Video Resolutions*.

7.3.1 Related Documentation and Resources

See the following documents for more information on configuring graphics adapters:

- *X Window System Administrator's Guide* – Provides general information about configuring your X display.
- `Xdec`(1X) – Defines the options available for the X Window System server and its associated system files.
- `xdpyinfo`(1X) – Displays information about the Xserver.
- `xgc`(1X) – Demonstrates the system's graphics capability program.
- `comet`(7) – Describes the device driver for the PowerStorm 4D10T, Elsa GLoria, IntraServer Combo adapters.
- `glXIntro`(3) – Provides an introduction to the OpenGL application programming interface.
- `sys_attrs_s3trio`(5) and `sys_attrs_vga`(5) – Provide information about the kernel's graphics adapter subsystems.

Best Practice documents on the topic of graphics are available online at the following Web location:

`http://www.tru64unix.compaq.com/docs/best_practices`.

Current titles are as follows:

- **`Adjusting Your Screen Settings`**
- **`Configuring Multiple Monitors on a Single System`**

7.3.2 Obtaining Information About Graphics Controllers

To determine the current model of graphics controller that is installed and in use, enter the following command:

```
# /usr/sbin/sizer -gt
COMET
```

You can also use the `hwmgr` command to display information about graphics controllers, as shown in the following example:

```
# /sbin/hwmgr get attribute -category graphics_controller
57:
  name = comet0
  category = graphics_controller
```

```
sub_category = 3D
model = COMET
power_mgmt_capable = 1
video_memory_size =  4 MB
num_planes = 8
num_colors = 256
X_resolution = 1024
Y_resolution = 768
registration_time = Thu Jul 25 13:12:24 2002
user_name = (null) (settable)
location = (null) (settable)
software_module = (null)
state = available
state_previous = unknown
state_change_time = none
event_count = 0
last_event_time = none
access_state = online
access_state_change_time = none
capabilities = 0
indicted = 0
indicted_probability = (null)
indicted_urgency = (null)
disabled = 0
```

In the preceding example, the first command determines the name of the graphics adapter. The second command displays the attributes (properties) associated with the adapter. Using the adapter's name (`comet0`) and its hardware identifier (HWID) of 57, you can obtain other information about the graphics adapter, such as where it is located in the hardware hierarchy:

```
# /sbin/hwmgr view hierarchy
52:   connection pci3slot6
57:     graphics_controller comet0
```

The preceding display is truncated for clarity. You can also use the SysMan Station to determine the location of a graphics adapter and display its attributes. See Chapter 2.

7.3.3 3Dlabs OXYGEN VX1 PCI/AGP Graphics Controller

The 3Dlabs OXYGEN VX1 PCI Graphics Controller (system option SN-PBXGF-AB) is a 2D controller based on the Graphics Chip 3Dlabs GLINT P3 graphics chip. It supports dual display and 24 color planes at a maximum resolution of 1920 x 1200 pixels (16-bit) at 70Hz frequency or 1280 x 1024 pixels (24 bit) at 85 Hz frequency. A single 15-pin VGA connector is provided on the card.

The 3DLabs OXYGEN VX1 AGP graphics controller accelerator module is a single expansion-slot, 32-bit AGP bus graphics option that provides 2D graphics acceleration for supported systems. It is based on 3DLabs' Permedia P4 graphics chip.

Installation and configuration instructions for supported releases of Tru64 UNIX are provided in the document titled *3Dlabs OXYGEN VX1 PCI Graphics Controller Installation Guide*. You can download this manual from the following location:
http://www.compaq.com/alphaserver/download/ek-vx1gc-ig.pdf

The following operating restrictions for Tru64 UNIX are specified in full in the hardware documentation:

- The minimum Tru64 UNIX firmware version for the PCI graphics controller single head support is V5.8 for a single head configuration and V5.9 for multihead configurations.
- The minimum Tru64 UNIX firmware version for the AGP graphics controller single head support is V6.0.
- Multiple colormaps are not supported. Applications should not install or deinstall colormaps themselves.
- The 3DLabs OXYGEN VX1 controller supports only one visual type (8-bit PseudoColor) at one time.
- By default, backing store and save unders are enabled.

7.3.4 Radeon 7500 PCI/AGP Graphics Controller

The ATI Radeon 7500 AGP & PCI (System options 3X-PBXGG-AB & 3X-PBXGG-AA, respectively) are 3D/2D graphics controllers based on ATI's RV200 Graphics Chip. The AGP variant is a single slot, 32 bit card that is compliant with the AGP Specification 2.0 at a 2X/4X speed. The PCI variant is a single PCI slot, 32 bit card that is compliant with the PCI 2.2 & hot-plug 1.1 specification at a 66MHZ bus speed.

The Radeon 7500 cards provide 64 MB of on-board DDR video memory, dual independent display controllers, and support for Digital Flat Panel Monitors (DVI-I). The primary video output port with connector for standard CRT monitors is capable of resolutions up to 2048 x 1536 pixels, 24 bits per pixel and 75 HZ refresh rate. The secondary video output port is capable of supporting either a Digital Flat Panel Monitor up to 1280 x1024 resolution, 24 bits per pixel and 75HZ refresh rate or a CRT monitor with resolutions upto 2048 X 1536, 24 bits per pixel, 75HZ.

7.3.4.1 Identifying the Radeon 7500

During system boot, the Radeon 7500 messages similar to the following are displayed at the console, depending on how many controllers are installed on the system:

```
radeon0 at pci0 slot 13
AGP 0.99 on Titan @ 0x00000000 16777216MB
radeon_initialize: Initialized radeon 1.0.0 20010105
```

When the system is at single or multiuser mode, use the following `hwmgr` commands to determine the hardware identifier (used with the `-id` option) and display the controller's attributes (properties):

```
# /sbin/hwmgr view hierarchy | grep graphics_controller
 57:             graphics_controller radeon0
# /sbin/hwmgr get attributes -id 57
   name = radeon0
   category = graphics_controller
   sub_category = 2D
   model = radeon
   power_mgmt_capable = 1
   video_memory_size =  0 MB
   num_planes = 8
   num_colors = 256
   X_resolution = 1280
   Y_resolution = 1024
   registration_time = Fri Jul 19 09:00:33 2002
   user_name = (null) (settable)
   location = (null) (settable)
   software_module = (null)
   state = available
   state_previous = unknown
   state_change_time = none
   event_count = 0
   last_event_time = none
   access_state = online
   access_state_change_time = none
   capabilities = 0
   indicted = 0
   indicted_probability = (null)
   indicted_urgency = (null)
   disabled = 0
```

The preceding output is an example. The actual output from the command might not equate to the controller's current settings, which are determined by the operating system. Refer to Section 7.3.4.4 and Section 7.3.4.5 for configuration information. You can determine the graphics resolution of the controller on a running system by using the following command:

```
# /usr/sbin/sizer -gr
1024x768
```

7.3.4.2 Restrictions on Use

The following restrictions apply to the current release of Tru64 UNIX:

- Colormap Information

  The Radeon 7500 graphics controller supports only one installed colormap at one time. Multiple colormaps are not supported. You must keep this in mind if you switch to the 8-bit color depth, where the default visual mode is PseudoColor. Any attempt to use more than one PseudoColor colormap at a time will cause colormap flashing (also known as technicolor).

  Applications must not install or deinstall colormaps themselves. Only the window manager must perform these operations. However, your application is responsible for providing the window manager with information about which colormaps to install or deinstall. You provide this information by using the `Xlib` function called `XSetWMColormapWindows()`. This function sets the `WM_COLORMAP_WINDOWS` property for a given window. See `XSetWMColormapWindows`(3X11) for the syntax.

- Backing Store and Save Under Information

  Backing store and save unders are disabled. Applications must not assume that these features are available and must be capable of handling exposure events. For example, see `DefaultColormapOfScreen`(3X11) (`DoesBackingStore`) .

Refer to the hardware documentation for the Radeon 7500 for more information.

7.3.4.3 Radeon 7500 Video Modes

lists the video modes (resolution) supported by the Radeon 7500 graphics controller (or adapter).

Table 7–1: Radeon 7500 Video Modes (Resolution)

Resolution (pixels)	Color Depth (bits per pixel)	Refresh Rates (Hz)
640x480	24, 16, 8	60, 72, 75, 85
800x600	24, 16, 8	60, 72, 75, 85
1024x768	24, 16, 8	60, 70, 75, 85
1152x864	24, 16, 8	60

Table 7–1: Radeon 7500 Video Modes (Resolution) (cont.)

Resolution (pixels)	Color Depth (bits per pixel)	Refresh Rates (Hz)
1280x1024	24, 16, 8	60, 75, 85
1600x1200	24, 16, 8	60, 65, 75, 85
1920x1440	24, 16, 8	60, 75
2048x1536	24, 16, 8	60, 65, 70, 75

The Radeon 7500 graphics adapter supports only one color depth at a time. The default visual mode is as follows:

- Color Depth (bits per pixel) – 24-bit TrueColor
- Resolution (pixels) – 1024x768
- Refresh rate – 70 Hz

7.3.4.4 Configuring the Xserver

Video modes are specified in the `/usr/var/X11/Xserver.conf` file, which is the Xserver configuration file. See `Xdec`(1X) for more information. To modify the video mode, change the following parameters as necessary:

- `-screen` – Specifies the screen resolution.
- `-depth` – Specifies the color depth.
- `-vclass` – Specifies the default visual.
- `-vsync` – Specifies the refresh rate.

The following procedure describes how to change your graphics mode:

1. Log in as root user on a remote terminal.

   Make any changes by using a remote terminal login or the console character cell terminal. If you use the root CDE login, you might experience problems when the video resolution changes.

2. Preserve the original version of the `/usr/var/X11/Xserver.conf` file so that you can restore it if there are problems with the new version. For example:

   ```
   # cp /usr/var/X11/Xserver.conf /usr/var/X11/Xserver.confORIG
   ```

3. Edit the original `/usr/var/X11/Xserver.conf` file with your required changes. At the end of the file you will see the following `args` location where you insert user command options:

   ```
   ! you specify command line arguments here
   args <
           -pn
   ```

>

For example, to change the default video mode to a resolution of 1280x1024, 8 bits per pixel, PseudoColor, at 75 Hz, modify the `args` section of the `Xserver.conf` file as follows:

```
args <
     -pn -screen 1280x1024 -depth 8 -vclass PseudoColor -vsync 75
```

Save the `Xserver.conf` file.

4. Restart the `Xserver` as follows:

```
# /usr/sbin/init.d/xlogin stop
# /usr/sbin/init.d/xlogin start
```

7.3.4.5 Configuring the CDE Xserver

For the CDE user environment, you can change the default video mode by modifying the `/var/dt/Xservers` file as follows. This procedure works only for CDE and not for other X-compliant user environments.

1. Log in as root user on a remote terminal.
2. Verify that the `/etc/dt/config` directory exists. If it does not exist, create it as follows:

```
# mkdir /etc/dt/config
# chmod 755 /etc/dt/config
```

3. Copy the default `Xservers` file as follows:

```
# cp /usr/dt/config/Xservers etc/dt/config/Xservers
```

At the end of the file you will see the following location where you insert your user command options:

```
   :0   Local local@console /usr/bin/X11/X :0
```

4. Edit the `/etc/dt/config/Xservers` file to add the appropriate options. For example, change the default to 1280x1024, 8 bits per pixel, PseudoColor, at 75 Hz, modify the Xservers file as follows:

```
:0 Local local@console /usr/bin/X11/X :0 -screen 1280x1024 -depth 8
-vclass PseudoColor -vsync 75
```

There should be no line breaks in this line. The preceding example is shown on two lines for clarity.

5. Restart the `Xserver` as follows:

```
# /usr/sbin/init.d/xlogin stop
# /usr/sbin/init.d/xlogin start
```

8

Administering Legacy Hardware

This chapter describes the procedures for adding and configuring certain legacy hardware devices or options as follows:

- PCMCIA cards (Section 8.1)
- CalComp graphics tablet (Section 8.2)
- Support of the Computer Interconnect (CI) bus and Hierarchical Storage Controllers (HSC) (Section 8.3)

8.1 PCMCIA Support

Certain processors are able to support PCMCIA (PC cards) as stated in the owner's manual and the *QuickSpecs* of a given release of the operating system. Only a small number of cards are qualified, but if the card adheres closely to standards it might work.

The steps involved in configuring a PCMCIA card are as follows:

1. Verify that your hardware and operating system support PCMCIA (PC cards). Consult the information from the adapter vendor and card vendor for any additional configuration steps that are necessary and contact the vendor if you are uncertain.
2. Determine the bus type, which can be ISA or EISA. This step determines the method of console configuration you are using.
3. Install the adapter and configure it using the appropriate console commands.
4. Configure a custom kernel and create the device special files. This step might be unnecessary if an adapter is installed and a card is inserted during initial installation and configuration of the operating system. In this case, the operating system detects the card and creates the kernel configuration entries and device special files. See the *System Administration* manual for information on kernel configuration and Section 1.5 for information on device special files.
5. Update the `/etc/remote` file.
6. Insert and eject the card as required.

See pcmcia(7) for specific information on configuring cards and on any current restrictions in a given release. See modem(7) and the *Network Administration: Connections* manual for information on modem connections.

8.2 CalComp Graphics Tablet

Certain processors are able to support the CalComp DrawingBoard III Tablet as stated in the Owner's Manual and the *QuickSpecs* of a given release of the Tru64 UNIX operating system. Other input devices supported by the Xinput extension to the Xserver might work using a similar configuration. When the software for the tablet is installed on your system, you can configure it to emulate a system mouse.

The steps involved in configuring a tablet are as follows:

1. Ensure that the /usr/var/X11/Xserver.conf file contains a line similar to the following:

   ```
   input <
   <_dec_xi_db3  lib_dec_xi_db3.so  XiDb3Init /dev/tty00:1:12:12:16:\
   1:8:1000:1:1 >
   >
   ```

   The tty that is specified is the serial port (COMM) where the tablet is connected to your system.

2. Specify settings for the tablet in the last line of this file by using the following syntax:

   ```
   device:mode:tabletWidth:tabletHeight:numbtns:corePointer:mouseScale:\
   resolution:Xincrement:Yincrement
   ```

   See calcomp(7) for an explanation of the data fields.

3. Connect the tablet to your system and turn it on.

4. Enter the following command to restart the Xserver so that the Xinput extension can recognize the tablet:

   ```
   # /usr/sbin/shutdown -r +5 \
   "Turning on support for the Calcomp Drawingboard III tablet"
   ```

   When the restart completes, the tablet is configured into the Xserver and ready to use.

See calcomp(7) for more information and for restrictions on use.

8.3 Support of the CI and HSC

The Computer Interconnect (CI) bus is a high-speed, dual-path bus that connects processors and Hierarchical Storage Controllers (HSCs) in a computer room environment. An HSC is an I/O subsystem that is a self-contained, intelligent mass storage controller that provides access to disks and tapes from multiple host nodes attached to the CI bus.

Note

The Tru64 UNIX implementation has the following limitations:

- You can attach a maximum of four HSCs to a CI bus.
- You can attach a single CI bus to a host.
- Under no circumstances can a Tru64 UNIX node participate as an OpenVMS cluster member. A configuration that includes an OpenVMS system and a Tru64 UNIX system residing on the same CI bus is not supported.

Tru64 UNIX supports the System Communication Architecture (SCA) for CI port adapters and HSCs. SCA implements port and class driver support, and standardizes the ways in which TMSCP (tms) and MSCP (ra) devices are handled. SCA separates features into different architectural layers, thus minimizing the effect that software changes to one layer have on other layers.

8.3.1 Hardware Setup, Restrictions, and Revision Levels

For information on physical components and setup, refer to the HSC hardware documentation and the hardware documentation for your processor and supported devices. Only processors with CI adapters can support HSC configurations.

When setting up the HSC controller hardware, attach a terminal to the HSC in order to use commands to get or set HSC parameters, to monitor connections between remote systems, and to identify the disk or tape status.

The maximum number of hosts on a CI bus is 16. The host number for any host on the CI bus must be between 0 and 15.

Note

Two parameters of particular importance are the system ID and the system name. Do not duplicate any system identification or names of nodes on the star coupler.

8.3.2 Software Installation and Restrictions

The installation software assists you in identifying and configuring the components of your system. Familiarize yourself with the basic installation instructions for your processor before starting the actual installation.

During installation of the Tru64 UNIX software, each accessible MSCP (`ra`) disk device must be uniquely identified by its unit plug number as follows:

- The unit plug number must be between 0 and 254, inclusive.
- Each unit plug number must be unique. Two different disks cannot have the same unit plug number even if the disks are on separate controllers. For example, if the unit plug number for a disk on controller A is 5 and the unit plug number for a separate disk on controller B is also 5, you must change one of the numbers.
- You can connect a disk with a unique unit plug number to two different controllers (dual or porting). Refer to `ra`(7) for information on how to specify the device entry in the system configuration file.

8.3.3 Configuration File Entries

The installation software ensures that your HSC components are configured into the kernel and are included in the `/usr/sys/conf/`*`NAME`* system configuration file, where *`NAME`* specifies your system name in uppercase letters.

The *System Administration* manual provides information on the following entries that correspond to a CI or HSC configuration:

- Description of the `scs_sysid` parameter
- CI adapter specifications
- Controller and device specifications

8.3.4 Booting an HSC Controller or an HSC Disk

The Tru64 UNIX software supports booting an HSC disk on the DEC 7000 and DEC 10000 processors. If an HSC controller fails, any disks connected to that HSC controller are inaccessible. Attempts to access those disks will cause the accessing system to hang until the HSC reboots completely. Refer to your processor hardware documentation for explicit instructions on booting an HSC disk.

8.3.5 Sharing Disk and Tape Units Among Several Hosts

Although an HSC can be shared among several hosts, there is no software interlocking mechanism to prevent concurrent write operations to the same partition by multiple Tru64 UNIX systems. The following restrictions must be observed:

- Only multiple readers can share a disk unit; writable file systems cannot be shared.
- If a disk will be shared, it should be hardware write protected.
- Each host must mount the file systems to be accessed with the read-only (`-r`) option to the `mount` command.

- Only a single host can mount a disk that contains writable file systems.
- Use the Network File System (NFS) if multiple writers need to share partitions.

You should coordinate disk unit ownership among the hosts on the CI bus; for example, assign a range of disk unit numbers to each host. The HSC controller can also be directed to limit disk access to an exclusive host system. This limitation protects the disk from accidental access by another host on the CI bus. For more information, see `radisk`(8), in particular the –`e` and –`n` options.

Tape drives that are attached to an HSC controller can be shared. This feature is recommended and provides greater use of tape drives. Be aware that the access mechanism provides serial sharing of the drives, not simultaneous access.

Index

D

E

F

G

H

I

J

K

L

M

N

S

T

U

V

W

X