HP OpenVMS New Features and Release Notes for HP AlphaServer ES47, ES80, and GS1280 Series Systems

Order Number: AA-RTNZA-TE

November 2003

This manual provides new features and release notes specific to OpenVMS support of the AlphaServer ES47, ES80, and GS1280 Series systems.

Revision/Update Information: This is a new manual that was printed in late December 2002. This electronic version contains updates made between December 16, 2002 and November 4, 2003.

Software Version: HP OpenVMS Alpha Version 7.3–1

Hewlett-Packard Company
Palo Alto, California

ZK6665

The HP OpenVMS documentation set is available on CD-ROM.

This document was prepared using DECdocument, Version 3.3-1b.

Contents

2 USB Configuration Manager (UCM)

3 Managing Workloads With Partitions

4 Creating an OpenVMS Galaxy on AlphaServer ES47/ES80/GS1280 Systems

Index

Figures

Tables

Preface

This document describes OpenVMS support of the HP AlphaServer ES47, ES80, and GS1280 Series systems and provides release notes that are specific to running OpenVMS Alpha Version 7.3-1 on these systems.

Intended Audience

This document is for people who will manage the AlphaServer ES47, ES80, and GS1280 Series systems. It assumes a basic understanding of OpenVMS system management.

Related Documents

For additional information on the topics covered in this manual, refer to the following documents:

- *OpenVMS Alpha Version 7.3-1 Release Notes*
- *OpenVMS Cluster Software Software Product Description* (SPD 29.78.*xx*)
- *Guidelines for OpenVMS Cluster Configurations*
- *OpenVMS Cluster Systems*
- *HP OpenVMS Alpha Partitioning and Galaxy Guide*
- *HP OpenVMS System Manager's Manual*

For additional information about HP OpenVMS products and services, visit the following World Wide Web address:

`http://www.hp.com/go/openvms`

Reader's Comments

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

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

How To Order Additional Documentation

For information about how to order additional documentation, visit the following World Wide Web address:

`http://www.hp.com/go/openvms/doc/order`

Conventions

The following conventions are used in this manual:

Ctrl/*x*	A sequence such as Ctrl/*x* indicates that you must hold down the key labeled Ctrl while you press another key or a pointing device button.
PF1 *x*	A sequence such as PF1 *x* indicates that you must first press and release the key labeled PF1 and then press and release another key or a pointing device button.
Return	In examples, a key name enclosed in a box indicates that you press a key on the keyboard. (In text, a key name is not enclosed in a box.) In the HTML version of this document, this convention appears as brackets, rather than a box.
. . .	A horizontal ellipsis in examples indicates one of the following possibilities: • Additional optional arguments in a statement have been omitted. • The preceding item or items can be repeated one or more times. • Additional parameters, values, or other information can be entered.
. . .	A vertical ellipsis indicates the omission of items from a code example or command format; the items are omitted because they are not important to the topic being discussed.
( )	In command format descriptions, parentheses indicate that you must enclose the options in parentheses if you choose more than one.
[ ]	In command format descriptions, brackets indicate optional elements. You can choose one or more options, or no options. (Note, however, that brackets are not optional in the syntax of a directory name in an OpenVMS file specification or in the syntax of a substring specification in an assignment statement.)
[|]	In command format descriptions, vertical bars separating items inside brackets indicate that you choose one, none, or more than one of the options.
{ }	In command format descriptions, braces indicate a required choice of options; you must choose one of the options listed.
bold text	This text style represents the introduction of a new term or the name of an argument, an attribute, or a reason.
italic text	Italic text indicates important information, complete titles of manuals, or variables. Variables include information that varies in system output (Internal error *number*), in command lines (/PRODUCER=*name*), and in command parameters in text (where *dd* represents the predefined code for the device type).
UPPERCASE TEXT	Uppercase text indicates a command, the name of a routine, the name of a file, or the abbreviation for a system privilege.
`Monospace text`	Monospace type indicates code examples and interactive screen displays.

-	A hyphen at the end of a command format description, command line, or code line indicates that the command or statement continues on the following line.
numbers	All numbers in text are assumed to be decimal unless otherwise noted. Nondecimal radixes—binary, octal, or hexadecimal—are explicitly indicated.

1

New Features and Release Notes

HP OpenVMS supports the AlphaServer ES80 Series systems in addition to the AlphaServer ES47 and the GS1280 Series systems that were introduced in January 2003. In July 2003, support was introduced for AlphaServer GS1280 32-processor systems, followed by support for AlphaServer GS1280 64-processor systems in October 2003.

For detailed information about these systems, refer to the QuickSpecs for each system series at:

http://h18003.www1.hp.com/alphaserver/

This document describes OpenVMS support of these systems and provides OpenVMS release notes specific to them. Selected new features are discussed in the context of OpenVMS support. This document also provides the world wide web address of the patch kits for updating OpenVMS Alpha Version 7.3-1 on your new AlphaServer system.

This manual was printed in late December 2002 and is updated electronically, as additional information becomes available, as shown in Table 1–1.

Check the OpenVMS documentation web site for the latest version at:

http://h71000.www7.hp.com/doc/731FINAL/6665/6665PRO.HTML
http://h71000.www7.hp.com/doc/731FINAL/documentation/pdf/ovms_es47_gs1280_nf_rn.pdf

Table 1–1 Revision History

Date	Revision
Dec. 16, 2002	Original version sent to print.
Jan. 16, 2003	Updates to Section 1.10, Section 1.19, Section 1.23, Section 1.27, and Section 1.32, and four new release notes, Section 1.12, Section 1.17, Section 1.33, and Section 1.34.
Jan. 27, 2003	Updates to Section 1.17 to show when qualification is planned to be completed and to Section 1.28 and to Section 1.29 to indicate the rarity of these problems. Note that this version received limit distribution.
Feb. 4, 2003	Update to Section 1.17 to show that qualification has successfully completed and to Section 1.29 to provide more information about the conditions that existed when the problem occurred.

(continued on next page)

Table 1–1 (Cont.) Revision History

Date	Revision
Apr. 14, 2003	Introduction of AlphaServer ES80 Series systems, support for MEMORY CHANNEL II (Section 1.16), support for Capacity on Demand (CoD) (Section 1.18), additional support for the Smart Array 5300 Backplane RAID Controller (Section 1.31), patch kits for problems reported in earlier versions of this manual (Section 1.28, Section 1.32), and updates to Section 1.4 and Section 1.12.
May 9, 2003	Introduction of hard partition support (Chapter 3), 3D graphics support (Section 1.9), note stating the maximum number of USB controllers per I/O box and per system (Section 1.11), support for the PBXDD device (Section 1.12), update to Re-Creating the System Disk pertaining to MEMORY CHANNEL devices (Section 1.19), correction to CPUSPINWAIT problem (Section 1.29), and correction to environmental events logged with incorrect severity level (Section 1.33)
June 13, 2003	Addition to Galaxy and Hard Partition support section regarding subsystem building blocks (Section 1.13).
Oct. 6, 2003	Addition of Galaxy support, including a new chapter (Chapter 4), new release notes (Section 1.35 and Section 1.36) and updates to several other notes including Section 1.1, Section 1.8, Section 1.9, Section 1.10, Section 1.13, Section 1.19, and Section 1.30.
Nov. 4, 2003	Update of several web site addresses, description of AlphaServer GS1280 64P support (Section 1.7), the addition of a recently discovered Galaxy configuration problem (Section 1.13), and an update of the CoD note regarding a firmware requirement (Section 1.18).

1.1 OpenVMS Alpha Version 7.3-1 Plus Upgrade Kit

OpenVMS Alpha Version 7.3-1 is the minimum version of the OpenVMS operating system that can run on AlphaServer ES47, ES80, and GS1280 Series systems. When first released in January, 2003, an upgrade kit named VMS731_EV7-V0100 was required. This software was factory installed.

In mid-June 2003, an OpenVMS Version 7.3-1 update kit named VMS731_UPDATE-V0100 became available. A prerequisite to installing this kit is the VMS731_PCSI-V0100 kit. The VMS731_UPDATE-V0100 kit contains all the images (or their successors) contained in the VMS731_EV7-V0100 kit plus most of the newer patch kits listed in this document. HP recommends that you install the VMS731_PCSI-V0100 kit and the VMS731_UPDATE-V0100 kits.

Note

The VMS731_PCSI-V0100 kit is a prerequisite for many of the newer patch kits for individual software components, such as the VMS731_CDRECORD-V0100 kit, and is incompatible with the VMS731_EV7-V0100 kit. If you choose to install any of the newer patch kits that require the VMS731_PCSI-V0100 kit, you must install either the VMS731_EV7-V0200 kit or the VMS731_UPDATE-V0100 kit, which includes the VMS731_EV7-V0200 kit.

HP also recommends that you monitor the web site listed in the following note to identify and install all the most up-to-date OpenVMS Alpha Version 7.3-2 patch kits.

If you need to reinstall OpenVMS Alpha Version 7.3-1 on your system, you *must* also apply the VMS731_EV7-V0100 update kit or its successor. As already noted, HP recommends that you install the VMS731_PCSI-V0100 kit and the VMS731_UPDATE-V0100 kits instead of the VMS731_EV7-V0100 kit. (Section 1.4 lists the patch kits included in the VMS731_EV7-V0100 update kit. For a list of the contents of the VMS731_UPDATE-V0100 kit, see its README file.)

After you reinstall the OpenVMS Alpha Version 7.3-1 operating system, the VMS731_PCSI-V0100 kit (log out and log back in after installing the PCSI kit), and the VMS731_UPDATE-V0100 kit, you must reboot your system for the changes to take effect. Consult the cover letter of each kit for information about the kit contents and for installation instructions.

Note

You can obtain the VMS731_PCSI-V0100 kit and the VMS731_UPDATE-V0100 kit and all OpenVMS Version 7.3-1 patch kits from the HP IT Resource Center's Patch Database at:

`http://www1.itrc.hp.com/service/patch/mainPage.do`

Note that the web site address is case sensitive. Note, too, that this site requires that you register. Registration is unrestricted, and you can access the patches as soon as you register.

1.2 OpenVMS Software Product Description Updates

The following OpenVMS Software Product Descriptions (SPDs) have been updated to include information about the new AlphaServer ES47 and GS1280 Series systems:

- OpenVMS Operating System for Alpha and VAX, Version 7.3 and Alpha Version 7.3-1
- OpenVMS Cluster Software

You can view and download these updated SPDs from the following web site:

`http://h18000.www1.hp.com/info/spd/`

1.3 OpenVMS Alpha Version 7.3-1 Documentation

The following OpenVMS Alpha Version 7.3-1 manuals are included on your system and on your OpenVMS Alpha Version 7.3-1 Operating System CD-ROM:

- *OpenVMS Alpha Version 7.3-1 Release Notes*
- *OpenVMS Alpha Version 7.3-1 New Features and Documentation Overview*
- *OpenVMS Alpha Version 7.3-1 Upgrade and Installation Manual*

The full OpenVMS Alpha Version 7.3-1 documentation set is available on the following web site:

`http://www.hp.com/go/openvms/doc/`

You can order printed documentation sets and individual manuals, as described in the *OpenVMS Alpha Version 7.3-1 New Features and Documentation Overview*.

1.4 VMS731_EV7-V0100 Update Kit Contents

The VMS731_EV7-V0100 update kit contains the following patch kits in their entirety:

- VMS731_FIBRE_SCSI-V0100
- VMS731_DCL-V0200
- VMS731_RMS-V0200

In addition to these kits, the VMS731_EV7-V0100 update kit contains additional images for the following facilities: USB, SYS, DW_DRIVER, BOOTDRIVER and LAN, CPU270F, CLUE$SDA, DRIVER (DQDRIVER), and MANAGE. For a complete list of all the kits and images contained within the VMS731_EV7-V0100 kit, refer to the cover letter for that kit.

Note

A problem was discovered in the VMS731_SYS-V0200 patch kit, parts of which are included in the VMS731_EV7-V0100 update kit. HP recommends that you install the VMS731_SYS-V0300 patch kit, which fixes the problem. For more information, see Section 1.32.

1.5 Minimum Firmware Version

The ES47, ES80, and GS1280 systems ship with factory-installed firmware. The firmware for these systems is updated periodically and will eventually be included on the Alpha firmware CD.

HP recommends that you check the firmware web site on a regular basis and update the firmware on your system when newer versions become available. Check the Download Firmware Images section of the firmware web site at:

`http://ftp.digital.com/pub/Digital/Alpha/firmware/`

Note that this web site address is case sensitive.

1.6 Maximum Throughput on GS1280 Series Systems

The AlphaServer ES47, ES80, and GS1280 Series systems are high-performance server platforms that incorporate state-of-the-art server design. The server design is based on a mesh architecture and advanced memory technology that provide significant advances in memory bandwidth and a low-latency I/O subsystem.

Detailed and thorough testing of selected real-world customer applications on a 16-CPU AlphaServer GS1280 Series system has shown consistent and significant gains in maximum high-end throughput over similarly configured AlphaServer GS160 systems. The gains in throughput have ranged from 1.5 to more than 3.5 times higher throughput.

The GS1280 produced superior results in every case with most tests showing gains of 2 times or more. These total gains achieved are a composite of better baselevel performance (for example, at the 4-CPU level) and consistently better SMP scaling. For example, one real-world test showed a 2 times gain at 4 CPUS and a 3.5 times gain at 16 CPUs.

A key factor in all of these results is the greatly improved memory access afforded by the GS1280's mesh architecture. Testing has shown that this faster memory access benefits both base level performance and SMP scaling.

The performance of your new GS1280 system as compared with your existing AlphaServer system will vary depending on many factors. These include your existing hardware configuration, the current versions of software, your application and current work load, as well as other factors.

HP and OpenVMS provide a full range of tools, services, and benchmark centers to help you explore your detailed performance requirements and needs. For more information, contact your HP account representive.

1.7 GS1280 Model 64 Support

OpenVMS supports the AlphaServer GS1280 Model 64, a system with 64 processors. OpenVMS maximum processor support is 32 processors (32P) per partition. To achieve 64 processors, 2, 32P OpenVMS partitions are required. For more information about the GS1280 Model 64 and its configuration requirements, refer to its QuickSpecs at the following location:

`http://h18000.www1.hp.com/products/quickspecs/Division/10410.html`

1.8 Hardware Support

This section describes OpenVMS support of selected new and existing hardware. For a complete list of all hardware supported on your system, see the documentation that accompanies your system. You can also check the QuickSpecs for your system on the AlphaServers web page at:

`http://h18002.www1.hp.com/alphaserver/`

Among the many options available on the AlphaServer ES47, ES80, and GS1280 Series systems are the following:

- DEGXA, a Gigabit Ethernet PCI adapter that can be used as a boot device.
- LP9002, an existing 2-Gb PCI-FC host bus adapter.
- LP9802, a new, 2-Gb PCI-FC host bus adapter

  To support this new host bus adapter, the latest Fibre Channel/SCSI patch kit is required (dec-axpvms-vms731_fibre_scsi-v300–4 or higher). Boot support is *not* available for this adapter on OpenVMS Alpha Version 7.3-1.

  Support is *not* planned for the LP8000, an earlier PCI-FC host bus adapter.
- USB keyboards and mouse.

1.9 Graphics Support

For the initial releases of the AlphaServer ES47, ES80, and GS1280 systems, 2D graphics support was available. Since then, support for the ATI Radeon 7500 Graphics Card, which provides both 2D and 3D support, has been completed. To enable support of this card on OpenVMS Alpha Version 7.3-1, the following patch kit is required:

`VMS731_GRAPHICS-V0100 (or higher)`

You can obtain this kit from the HP IT Resource Center's Patch Database at:

http://www1.itrc.hp.com/service/patch/mainPage.do

Note that the web site address is case sensitive.

This kit contains the 2D and 3D graphics software to support the ATI RADEON 7500 AGP and PCI graphics cards on OpenVMS Alpha Version 7.3-1. The part numbers are as follows:

- AGP card: 3X-PBXGG-AB
- PCI card: 3X-PBXGG-AA
- Open3D license: QL-0ADA9-AA

Note

This kit ships with full support for the AGP card. The PCI card is also supported; it requires the VMS731_GRAPHICS-V0200 (or higher) kit.

For updates on support for the AGP and PCI cards, refer to the QuickSpecs webpage at:

http://h18000.www1.hp.com/products/quickspecs/Division/10410.html

1.10 Patch Kit for DEGXA Support

This release note has been updated since December 16, 2002.

The DEGXA Gigabit Ethernet PCI adapter is supported on OpenVMS Alpha Version 7.3-1 by means of the following patch kit:

VMS731_LAN-V0100 (or higher)

You can obtain this kit from the HP IT Resource Center's Patch Database at:

http://www1.itrc.hp.com/service/patch/mainPage.do

Note that the web site address is case sensitive.

The VMS731_LAN-V0100 kit (or higher version number) is included in the factory-installed software for the AlphaServer ES47, ES80, and GS1280 Series systems.

Note

The latest version of this kit is VMS_LAN-V0600, which requires the VMS731_PCSI-V0100 kit as a prerequisite.

1.11 Universal Serial Bus (USB) Support

The AlphaServer ES47, ES80, and GS1280 Series systems provide USB support, which replaces the traditional device interface support available on previous Alpha systems and on VAX systems. In other words, the AlphaServer ES47, ES80, and GS1280 systems have no general purpose serial ports, no parallel ports for printers, no mouse ports, and no keyboard ports. USB support is enabled by default on OpenVMS.

Note

OpenVMS supports a maximum of three USB controllers per I/O box with a maximum of 26 controllers for eight I/O boxes.

OpenVMS provides USB support for the following devices:

- Keyboard

  A keyboard with the traditional OpenVMS keyboard layout is supported.

- Mouse

Refer to the QuickSpecs for your system for the part numbers for supported devices.

OpenVMS provides a new utility, the USB Configuration Manager (UCM) utility, for managing USB devices. For currently supported USB devices, its principal use will be for viewing the event log with the UCM utility command SHOW EVENTS. For more information about UCM, see Chapter 2.

1.12 Patch Kits Needed for X.25 and WAN Devices

This release note has been updated since December 16, 2002.

Customers using the PBXDD device need a patch kit to enable this device to work on the AlphaServer GS1280 Series systems. Beginning in May, 2003, the kit that provides this support is available. It is named X25ALP X25_V16ECO1 X.25 V1.6 for OpenVMS Alpha and contains two separate kits, one for X.25 and one for WANDD, whose names follow:

- DEC-AXPVMS-X25-V0106-1-1.PCSI
- DEC-AXPVMS-WANDD-V0106-1-1.PCSI

You can access the X25_V16ECO1 kit if you are a contract customer and already have an account from the following web site:

`http://ftp.support.compaq.com/cgi-bin/entitlement.cgi/patches/entitled/`

If you are not already a contract customer or if you do not have an account at the patch site, you can create an account at this site or contact a customer representative to enable you to obtain the kit.

1.13 Galaxy and Hard Partition Support

Galaxy is now supported on AlphaServer ES47, ES80, and GS1280 Series systems.

Galaxy support requires the installation of two patch kits and a firmware upgrade, as follows:

- Partitioning kit name: `DEC-AXPVMS-VMS731_PARTITIONING-V0200--4.PCSI (or higher)`
- Threads kit name: `DEC-AXPVMS-VMS731_PTHREAD-V0200--4.PCSI`
- Firmware version: 6.6

The following restrictions and problem exist:

- The OpenVMS Graphical Configuration Manager (GCM) is not supported for Alphaserver ES47/ES80/GS1280 Galaxy configurations at this time. However the Graphical Configuration Utility (GCU) is supported. The goal is to remove this restriction in the future.
- The Smart Array 5300 (KZPDC) Backplane Raid Controller is currently supported only as a data device in ES47/ES80/GS1280 Galaxy configurations. Boot and crash dump capability are not supported at this time on these controllers. The goal is to provide support with corrected firmware or corrected OpenVMS software.
- Since initially releasing Galaxy support, a problem has been identified with the SMCI (Shared Memory Cluster Interconnect) pseudo-device. If instances are booted together, the devices operate normally. However, a reboot of one instance causes its device to go offline. A fix for this problem is being developed and will be provided in the following kit:

  `DEC-AXPVMS-VMS731_PARTITIONING-V0300--4.PCSI`

  This kit will be available soon from the HP IT Resource Center's Patch Database at:

  `http://www1.itrc.hp.com/service/patch/mainPage.do`

For information about configuring a Galaxy on an AlphaServer ES47, ES80, and GS1280 Series system, see Chapter 4.

Hard partition support, which requires a firmware update and a patch kit, has been qualified and is now available on the AlphaServer ES47, ES80, and GS1280 Series systems. Chapter 3 provides more information about the firmware and patch kit requirements and describes how to configure hard partitions on these systems.

Note

The former limitation of hard partitions on system building block boundaries, only, has been removed. Hard partitions on subsystem building block boundaries are now supported, as described in Chapter 3. Please note the constraints on hard partitions in subsystem building blocks as described in Chapter 3.

1.14 RAD Support

OpenVMS support for resource affinity domains (RADs), also known as NUMA support or awareness, is not provided in the initial release of the AlphaServer ES47, ES80, and GS1280 Series systems. RAD support may be considered for a future release of OpenVMS.

1.15 Patch Kits in OpenVMS Cluster Systems

No additional patch kits are required for AlphaServer ES47, ES80, and GS1280 Series systems to be members of an OpenVMS Cluster system. However, if an AlphaServer ES47, ES80, or GS1280 Series system becomes a member of a mixed-version OpenVMS Cluster, you must use the kits listed in Table 4-1, Remedial Kits Required for Cluster Compatibility, of the *OpenVMS Alpha Version 7.3-1 Release Notes*.

1.16 Cluster Interconnects Supported

The following support for OpenVMS Cluster system interconnects is available:

- LAN (Ethernet, FDDI, and ATM)
- CI
- Fibre Channel as a shared-storage cluster interconnect

  A second interconnect is required for cluster communications.
- SCSI as a shared-storage cluster interconnect, using the KZPBA-CC adapter

  A second interconnect is required for cluster communications.

MEMORY CHANNEL II support has been qualified using MEMORY CHANNEL adapters Revision D02.

No support is planned for the proprietary DSSI interconnect.

1.17 Gigabit Ethernet Qualified as Cluster Interconnect

The qualification of the DEGXA Gigabit Ethernet PCI adapter to provide cluster interconnect support for AlphaServer ES47, ES80, and GS1280 Series systems has successfully completed. No additional support is required beyond the patch kit named in Section 1.10.

1.18 Capacity on Demand (CoD) Support

Capacity on Demand (CoD) software has been qualified on OpenVMS on the AlphaServer ES47, ES80, and GS1280 Series systems. CoD allows you to configure your system with two types of CPUs:

- Active CPUs that are purchased initially with the system
- CoD CPUs that are physically available and can be activated as needed without rebooting the system

When you install the CoD software, your system is configured with the CPUs that were purchased initially with the system as well as with the additional CoD CPUs that are reserved for future demand. You can activate the CoD CPUs when you need them without rebooting the system.

To use the CoD software, your console firmware must be at V6.5-8 (or higher) (the same firmware version that is required for 64P support).

For more information about CoD, contact your HP or HP partner representative.

1.19 Re-creating the System Disk

Note

HP advises you to back up your system disk when you first receive your system.

Backing up your system disk is especially important if you depend on a graphics console to interact with the system, because you cannot turn on DECwindows on an ES47, ES80, or a GS1280 system unless the VMS731_EV7-V0100 kit is installed. To re-create your system disk you need OpenVMS Alpha Version 7.3-1 plus the VMS731_EV7-V0100 kit. To obtain the kit, you need web access.

Note

Re-creating the system disk is not supported with a MEMORY CHANNEL device installed. If you attempt to do that, the operation will fail with an invalid exception (INVEXCEPTN) bugcheck.

If your ES47, ES80, or GS1280 system is in a cluster with other systems that have access to the World Wide Web, then you can download the VMS731_EV7-V0100 kit to another cluster member and copy it to your new system. Refer to Section 1.1 for the web site location.

If you need to re-create your system disk from the OpenVMS Version 7.3-1 CD-ROM, you may need to alter some system parameters to allow the system to boot. This is generally necessary if you have several network devices. Network devices consume considerable amounts of nonpaged pool, and the default amount needs to be increased.

OpenVMS recommends increasing nonpaged pool by the increments *per adapter* shown in Table 1–2:

Table 1–2 Network Devices and Additional Nonpaged Pool

Network Device Type	Amount to Add
Ethernet (ei)	300 KB (307200)
Gigabit (ew or eg)	600 KB (614400)
FDDI (fw)	1000 KB (1024000)

After you have calculated how much additional pool is needed for all network device adapters, add that amount to the NPAGEDYN and NPAGEVIR system parameters. This should allow you to boot, and subsequent AUTOGEN calculations should set the nonpaged pool size correctly.

GS1280 systems with large amounts of memory may experience difficulty recreating the system disk because they may run out of S0 virtual address space. To work around this problem, you can remove graphics devices or lower the value of the MAXPROCESSCNT and WSMAX parameters. This problem is partially addressed in the required patch kits. As there is no way to fix this problem on the shipping CD-ROM, the goal is to provide a solution in a future release.

At this time, HP recommends that you *not* re-create the system disk using the graphics device as a console device. Sometimes the installation process completes normally, but occasionally the process hangs. The problem is under investigation. The goal is to provide a remedy in the future.

1.20 License Requirement Change

The AlphaServer ES47, ES80, and GS1280 Series systems require a minimum of two OpenVMS software licenses: one license for base support and one license for dual SMP support for the first two processors. This is a change from the previous way of licensing OpenVMS AlphaServer SMP systems. The dual SMP licenses for OpenVMS are included with the CPU modules when you purchase an OpenVMS system or when you purchase additional CPU modules for an OpenVMS system.

1.21 Changes to System Serial Number Command

On earlier OpenVMS AlphaServer systems, the SRM command SET SYS_SERIAL_NUM *nnnnn* allowed you to specify a serial number up to 16 characters long. Now, the maximum-length serial number that you can specify is 12 characters.

On OpenVMS AlphaServer ES47, ES80, and GS1280 Series systems, and the systems to follow in this series, the system serial number is set at the Management Backplane Module (MBM) console, as shown in the following example:

```
MBM> SHOW SYS_SERIAL_NUM
SYS_SERIAL_NUM is not set!
MBM> SET SYS_SERIAL_NUM 123456789012
Updating SYS_SERIAL_NUM in FRU EEPROMs....
MBM> SHOW SYS_SERIAL_NUM
SYS_SERIAL_NUM          123456789012
MBM>
```

1.22 Installing Licenses

To ensure that the common license database can share license units among hard and soft partitions (when they are supported), perform the following steps:

1. Calculate required units.
   - Load the base OpenVMS license.
   - Load the SMP licenses.
   - Use the following command to verify that you have the correct number of license units:

     ```
     $ SHOW LICENSE /UNIT_REQUIREMENTS /CLUSTER
     ```

     Note

     The base OpenVMS license does not allow an interactive login. An SMP license is required for an interactive login.

2. Add your licenses to the common license database. For example:

   ```
   $ LICENSE REGISTER OPENVMS-ALPHA /ISSUER=DEC -
   _$ /AUTHORIZATION=USA123456 -
   _$ /PRODUCER=DEC -
   _$ /UNITS=1050 -
   _$ /AVAILABLITY=H -
   _$ /OPTIONS=(NO_SHARE) -
   _$ /CHECKSUM=2-BGON-IAMA-GNOL-AIKO
   ```

   Note that you cannot use the /INCLUDE qualifier with the LICENSE REGISTER command to override the NO_SHARE attribute of the license.

3. Modify the license to override the NO_SHARE attribute of the PAKs with the command LICENSE MODIFY /INCLUDE=(*node-name-list*). For example:

   ```
   $ LICENSE MODIFY  OPENVMS-ALPHA -
   _$ /AUTHORIZATION=USA123456 -
   _$ /INCLUDE=(NODEA, NODEB, NODEC)
   ```

4. To make OpenVMS Alpha license units available to the instance of OpenVMS running in each partition, you must ensure that the MBM environment variable SYS_SERIAL_NUM is the same in each partition. To do so, perform the following steps:

   a. From the MBM console, use the SHOW SYS_SERIAL_NUM command to display the system serial number. For example:

   ```
   MBM>>>SHOW SYS_SERIAL_NUM
   sys_serial_num G2A105
   ```

   b. If the value of SYS_SERIAL_NUM is blank, use the SET SYS_SERIAL_ NUM command at the MBM console to set the system serial number. For example:

   ```
   MBM>>>SET SYS_SERIAL_NUM G2A105
   ```

   The system serial number is propagated to every partition, hard or soft.

   Note

   You must create a nonzero value of up to 12 characters. (Earlier systems supported 16-character values.) Ensure that the system serial number that you create is not used on any other AlphaServer system on this OpenVMS Cluster.

5. To correctly update the OpenVMS Cluster license database, HP recommends that you completely shut down and reboot all OpenVMS Cluster common nodes. A rolling upgrade does not correctly update the common license database.

For partitionable systems that share NO_SHARE licenses across partitions, you might see the following error messages when the system boots:

```
%LICENSE-E-NOAUTH, DEC OPENVMS-ALPHA use is not authorized on this node
-LICENSE-F-EXCEEDED, attempted usage exceeds active license limits
-LICENSE-I-SYSMGR, please see your system manager
Startup processing continuing...
```

You can safely ignore these messages. The messages are displayed when someone has logged into a system that is actively sharing the OPENVMS-ALPHA PAK. This will be fixed in a future release.

For more information about installing and managing licenses, refer to the *OpenVMS License Management Utility Manual*.

1.23 STOP/CPU and Shutdown Behavior on AlphaServer ES47, ES80, and GS1280 Series Systems

This release note has been updated since December 16, 2002, to describe the potential shutdown behavior.

Because of hardware restrictions, any CPU on an ES47, ES80, or GS1280 Series system with an attached I/O drawer cannot be stopped by means of the DCL command STOP/CPU. In contrast, CPUs on these systems without an attached I/O drawer can be stopped with this command.

In addition, when the shutdown procedure is invoked on an ES47, ES80, or GS1280 Series system with an attached I/O drawer, an error message such as the following might be displayed:

```
%SYSTEM-W-WRONGSTATE, CPU 5 is in the wrong state for the requested operation
```

You can ignore such messages. The shutdown will complete correctly.

1.24 Setting Time at MBM Required

You must set the correct time and date on the MBM of the AlphaServer ES47, ES80, and GS1280 Series systems. If you do not, OpenVMS might display an incorrect time and date.

1.25 TCP/IP Services NTP Problem and Resolution

A problem has been identified in TCP/IP Services with NTP (Network Time Protocol) when running on an ES47, ES80, or GS1280 system. NTP cannot adjust the system clock. Consequently, NTP cannot provide accurate timekeeping. The following error message appears in the NTP log file:

```
%SYSTEM-F-BADLOGIC, internal logic error detected
VMS timekeeping is not working as expected - can't proceed
```

To correct this problem, install the latest TCP/IP ECO kit, V5.3 ECO2.

You can obtain this kit from the HP IT Resource Center's Patch Database at:

```
http://www1.itrc.hp.com/service/patch/mainPage.do
```

Note that the web site address is case sensitive.

1.26 DECnet DTS/DTR Large Transfer Restriction

During testing, a problem was detected while attempting DECnet DTS/DTR transfers greater than 4092 bytes. Transfers of this size fail, with the following error message:

```
%NONAME-F-NOMSG, Message number 00000004
```

This restriction is not specific to AlphaServer ES47, ES80, and GS1280 systems. Support for transfers greater than 4092 bytes is planned for a future release.

1.27 ERLBUFFERPAGES Must Be Increased for GS1280 Series Systems

Some AlphaServer systems capture error state when the operating system is unable to do so. When such data exists, the operating system logs the data to the error log during the next boot.

The AlphaServer GS1280 Series systems capture a large amount of data — too large for the default error log buffer to hold. HP recommends that you set the ERLBUFFERPAGES system parameter on these systems to its maximum of 128. The setting of 128 makes each error log buffer 64K bytes. You might want to reduce the number of buffers by decreasing the ERRORLOGBUFFERS system parameter.

After resetting the size of the error log buffer, run AUTOGEN. If AUTOGEN is unable to extend the SYS$ERRLOG.DMP file, you can do that with the following command, using the size recommended by AUTOGEN. This size can also be found in the AGEN$PARAMS.REPORT file.

```
$ RUN SYS$SYSTEM:SYSGEN
SYSGEN> CREATE SYS$SYSTEM:SYS$ERRLOG.DMP/SIZE=xx
```

1.28 Fibre Channel Problem Fixed by Patch Kit

During extensive testing, a *single* failure with the following footprint was produced. A fix is available in patch kit VMS731_FIBRE_SCSI-V0200. You can obtain this kit from the HP IT Resource Center's Patch Database at:

http://www1.itrc.hp.com/service/patch/mainPage.do

Note that the web site address is case sensitive.

HP recommends that you install this patch kit (or later version) at your earliest convenience.

```
Crashdump Summary Information:
------------------------------
Crash Time:         3-DEC-2002 09:52:10.89
Bugcheck Type:      INVEXCEPTN, Exception while above ASTDEL
Node:               M01P0   (Standalone)
CPU Type:           hp AlphaServer ES47 7/1000
VMS Version:        V7.3-1
Current Process:    NULL
Current Image:      <not available>
Failing PC:         FFFFFFFF.80519CEC    SYS$FGEDRIVER+15CEC
Failing PS:         18000000.00000804
Module:             SYS$FGEDRIVER    (Link Date/Time: 13-NOV-2002   10:01:39.26)
Offset:             00015CEC
```

1.29 CPUSPINWAIT Problem Fixed by Graphics Patch Kit

The problem described in this release note is fixed by the following graphics patch kit:VMS731_GRAPHICS-V0100 (or higher).

You can obtain this kit from the HP IT Resource Center's Patch Database at:

http://www1.itrc.hp.com/service/patch/mainPage.do

Note that the web site address is case sensitive.

Extensive testing of GS1280 and ES47 Series systems produced a *single* CPUSPINWAIT failure during boot. This occurred during DECwindows startup when a USB keyboard was plugged in.

1.30 CD-R Benign Device Error Problem Fixed

The problem described in this release note is fixed by the patch kit DEC-AXPVMS-VMS731_CDRECORD-V0100–4.PCSI.

You can obtain this kit from the HP IT Resource Center's Patch Database at:

http://www1.itrc.hp.com/service/patch/mainPage.do

Note that the web site address is case sensitive.

Note that the VMS731_PCSI-V100 kit is a prerequisite for installing this kit.

Problem Description

A device error for the CD-R drive on the ES47, ES80, and GS1280 Series systems might be logged during a reboot of the system. This error is benign, and the device should work properly. Keeping valid media in the drive might prevent the error from occurring.

Furthermore, during a CD record session, you may see the following message:

```
error 0. write_g1: scsi sendcmd: retryable error
```

The operation should complete successfully.

1.31 Smart Array 5300 (KZPDC) Backplane RAID Controller Support

The Smart Array 5300 (KZPDC) backplane RAID controller on the AlphaServer ES47, ES80, and GS1280 Series systems, formerly supported only as a data device, can now be used as a system disk or as a target for dump off system disk (DOSD), provided that you:

- Use only one Smart Array 5300 (KZPDC) backplane RAID controller per system, regardless of the size of the system.
- Install the latest Fibre Channel patch kit.
- Install a newer version of the console firmware.

You can obtain the Fibre Channel patch kit, VMS731_FIBRE_SCSI-V0200 (or higher) from the HP IT Resource Center's Patch Database at:

http://www1.itrc.hp.com/service/patch/mainPage.do

Note that the web site address is case sensitive.

The required version of the console firmware is V6.14-12 (or higher), which you can obtain from the following web site:

http://ftp.digital.com/pub/Digital/Alpha/firmware/readme.html

Note that the address for the firmware web site is case sensitive.

See the HP Alpha Systems Storage web site for Smart Array controllers for up-to-date restrictions and release notes.

http://h18002.www1.hp.com/alphaserver/products/storage/sa5300a/

1.32 VMS731_SYS-V0200 Kit Problem Fixed

Note

This note is included because some images from the VMS731_SYS-V0200 patch kit are included in the VMS731_EV7-V0100 Update Kit, and you might encounter this problem. The problem has been fixed in the VMS731_SYS-V0300 patch kit, which you can obtain from the HP IT Resource Center's Patch Database at:

http://www1.itrc.hp.com/service/patch/mainPage.do

Note that the web site address is case sensitive.

HP recommends that you install the VMS731_SYS-V0300 patch kit.

Problem Description

After installing the DEC-AXPVMS-VMS731_SYS-V0200–4.PSCI patch kit, you may experience process hangs when using SMTP mail.

When sending a mail message via SMTP, the mail is delivered, but the process handling the SMTP mail delivery hangs when creating the mail notification. The mail delivery notification is never sent, and any subsequent SMTP mail accumulates unsent in the queue.

The process hangs occur because BRKTHR tries to allocate memory from P1 space. If it is unsuccessful, it tries to allocate from P0 space, resulting in a hang.

Workaround

Increasing the size of CTLPAGES provides more P1 space and helps alleviate the process hangs.

If you have experienced these hangs and have undelivered SMTP mail messages in the queue, executing the following command procedure, in batch, on each node with a problem, will clear the mail messages. To use this command file, replace *node-name* with the name of the system. Note that the lines enclosed by brackets ([ ]) are optional. They are provided if you want to list multiple nodes.

```
$ SET NOON
$ TCPIP STOP MAIL
$ stop/que/reset TCPIP$SMTP_node-name
[$ stop/que/reset TCPIP$SMTP_node-name]
$ tcpip anal mail/repair
$ start/que TCPIP$SMTP_node-name
[$ start/que TCPIP$SMTP_node-name]
$ TCPIP START MAIL
$ sho que TCPIP$SMTP_node-name/full/all
[$ sho que TCPIP$SMTP_node-name/full/all]
$ exit
```

1.33 Environmental Events Logged with Incorrect Severity Fixed

The problem described in this release note is fixed by the patch kit VMS731_CPU270F-V0100.

You can obtain this kit from the HP IT Resource Center's Patch Database at:

http://www1.itrc.hp.com/service/patch/mainPage.do

Note that the web site address is case sensitive.

Environmental events on the AlphaServer ES47, ES80, and GS1280 Series systems are logged as Fatal Environmental Events (682) when they should be logged as Environmental Warning Events (686).

1.34 Machine Checks with Smart Array 5300 Adapters

Systems with Smart Array 5300 (KZPDC) adapters have been known to suffer a 660 machine check during a system shutdown, resulting in the console display of the machine check data. These machine checks appear to be benign. The crash dump is written successfully and can be analyzed. The goal is to provide a solution to this problem in the future.

1.35 Problem with DCL and Some GETSYI Item Codes

The output for some GETSYI item codes on an AlphaServer ES47/ES80/GS1280 Series system may exceed the DCL maximum string length, resulting in a TKNOVF warning from DCL. Item codes that exhibit this behavior include GLX_MBR_NAME and GLX_MBR_MEMBER.

This problem will be addressed in a future release of OpenVMS.

1.36 Problems with large DECram Devices and Galaxy Shared Memory

A problem was noticed during testing with large DECram devices and Galaxy shared memory.

The system crashed with the following footprint:

```
Crashdump Summary Information:

------------------------------

Crash Time: 19-MAY-2003 10:42:32.76

Bugcheck Type: INCONMMGST, Inconsistent memory management state

Node: QTV18 (Cluster)

CPU Type: hp AlphaServer GS1280 7/1150

VMS Version: V7.3-1

Current Process: STARTUP

Current Image: $1$DGA6005:[SYS2.SYSCOMMON.][SYS$STARTUP]MDRECOVER.EXE;1

Failing PC: FFFFFFFF.805EE314 GLX$MMAP_ALLOC_PAGES_C+006C4

Failing PS: 30000000.00000803

Module: SYS$GALAXY (Link Date/Time: 18-JUL-2002 19:55:04.43)

Offset: 00014314
```

This problem will be addressed in a future release of OpenVMS. In the meantime, to work around this problem, increase the size of shared memory.

2

USB Configuration Manager (UCM)

The USB (Universal Serial Bus) Configuration Manager (UCM) utility allows you to connect a computer to a variety of devices using a single four-wire cable.

Note

The devices named in this chapter are for purposes of example. OpenVMS support for USB has been coded according to the USB standards; therefore, USB devices may work with our implementation. However, only those devices qualified by HP for a specific platform are supported by OpenVMS. Refer to the hardware QuickSpecs for supported USB devices.

The USB Configuration Manager (UCM) is an OpenVMS utility that does the following:

- Records events such as plugging or unplugging devices and errors that occur on a USB bus. This is the USB event-logging function of UCM.
- Maps physical devices to persistent device names (based on either serial number or bus location).
- Manages additions, deletions, and modifications to devices configured on the system.

The following sections describe USB and UCM concepts.

2.1 USB and UCM Concepts

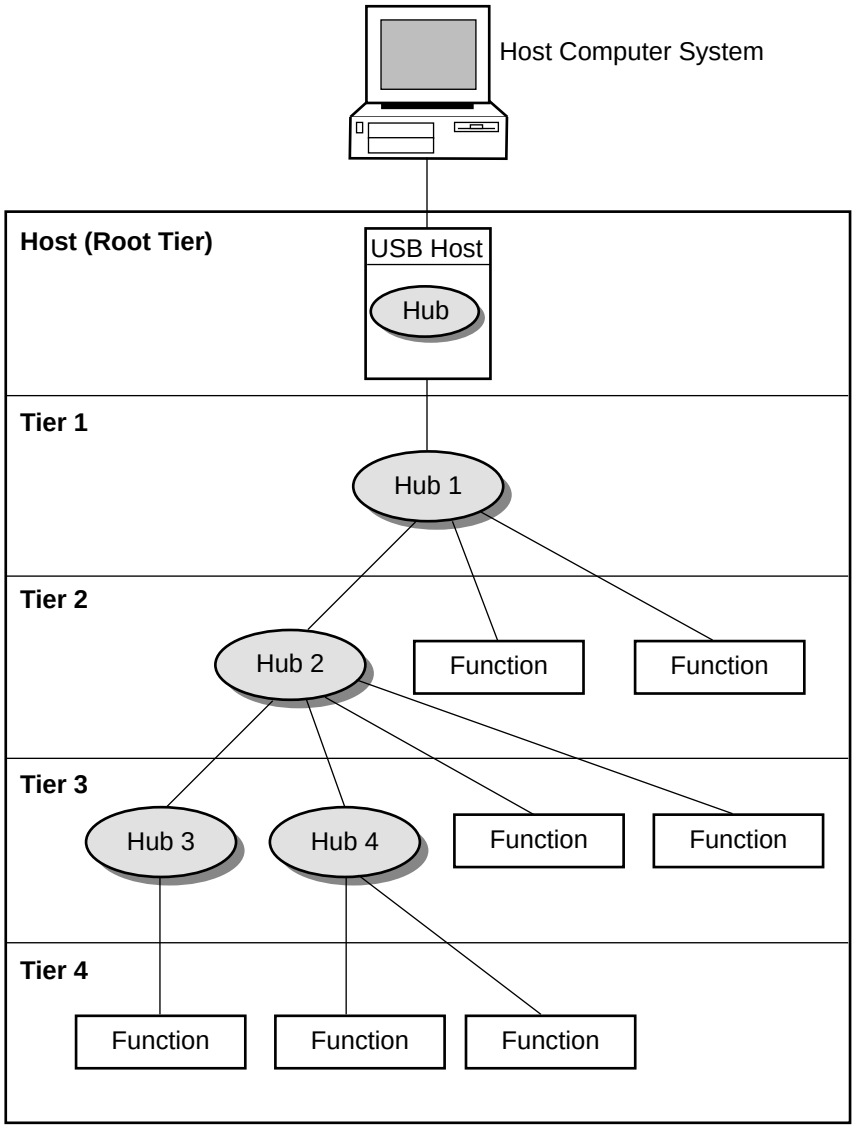
The following sections introduce and explain USB and UCM.

2.1.1 Introduction to USB

Universal Serial Bus (USB) is a communications architecture that enables a computer to interconnect a variety of devices using a single four-wire cable. The purpose of USB is to provide a user-friendly way to connect low and medium-speed devices to host computers.

The USB connects USB devices to the USB host, which, in turn, connects with a host computer system. Each USB has only one host, labeled USB Host in Figure 2–1. (A host, however, can have multiple USBs.)

Figure 2–1 USB Configuration



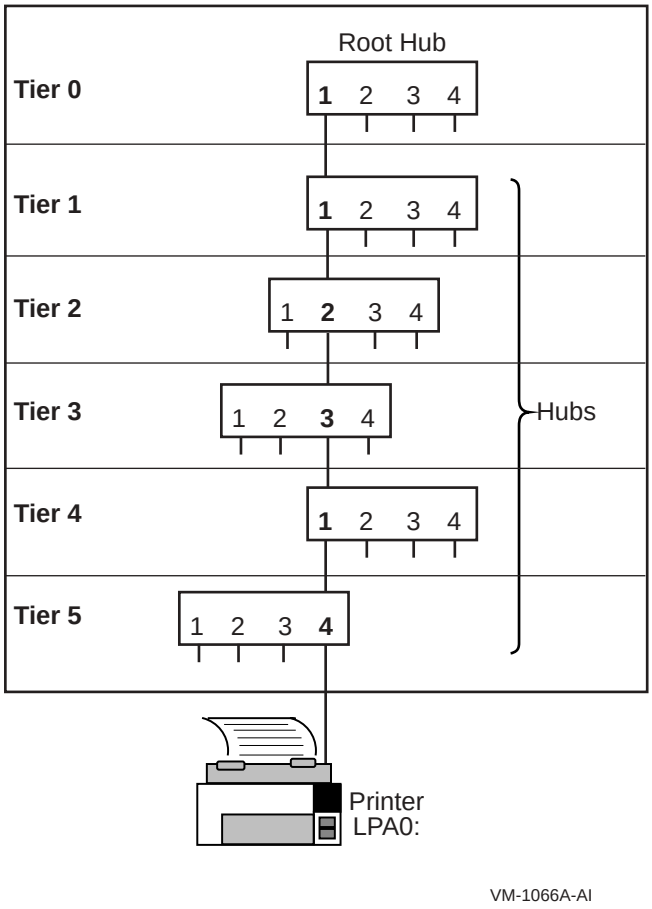
The USB host is integrated with a root hub, which provides one or more attachment points for devices. The USB physical interconnections from each hub form a "star," with a hub at the center of each star.

Point-to-point wire connections link the USB host to a hub or a function, or a hub to another hub or function. Hubs and functions are USB devices that do the following:

- **Hubs** provide additional attachment points to the USB.
- **Functions** provide capabilities to the system, such as a mouse or a keyboard.

Figure 2–2 shows that up to six hubs can be chained to create a tiered configuration. (The device shown here is for purposes of example. Refer to the hardware QuickSpecs for supported USB devices.) The path of a device is determined by its location in the structure; for example, the path to the printer LPA0: in Figure 2–2 is 1.1.2.3.1.4. (Note, however, that the numbers printed on the physical hub might not match the numbers that UCM displays.)

Figure 2–2 Hub Tiers



The OpenVMS device names of USB devices follow:

Device Name	Description
KBD*n*[1]	Keyboard
MOU*n*[1]	Mouse
TXA*n*[1]	Modem
LPA*n*[1]	Printer driver
HID0	Special-case driver that users cannot access
UCM0	Hub driver (one per system)

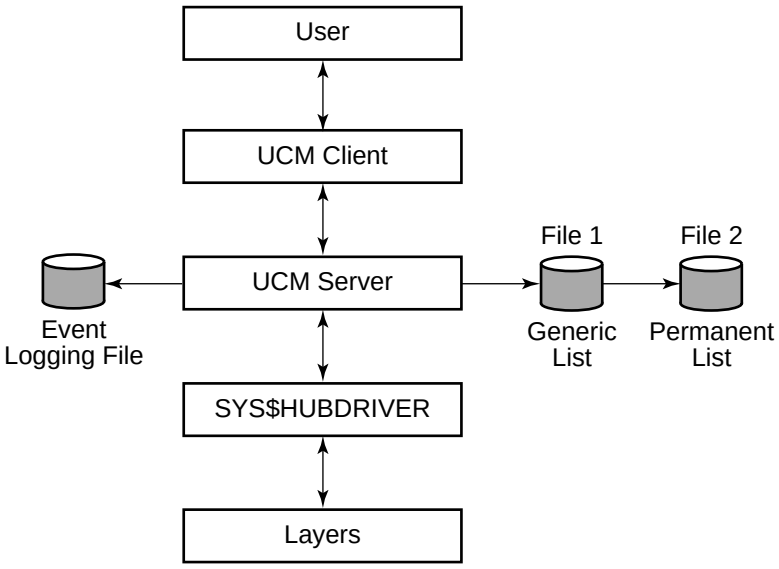
[1]The value of *n* can be between 0 and 9999.

The UCM works with the hub driver to configure USB-supported devices.

2.1.2 UCM Concepts and Operation

The UCM is made up of client and server layers. The user interacts with the client layer, and the client layer interacts with the server layer. It is the server layer that interacts with the USB. Figure 2–3 shows the interaction of these layers.

Figure 2–3 UCM Architecture



As the figure indicates, the UCM server maintains the event-logging file and the generic and permanent list files. These files are passed to the UCM client, which can display the files to the user. (The types of lists that the UCM server uses are explained in Section 2.1.2.1.) The UCM server is in contact with the UCM driver, SYS$HUBDRIVER, which maintains connections with other layers of the architecture.

2.1.2.1 Types of UCM Lists

The UCM server has the following three types of lists:

- Generic list

  This list contains descriptions of devices that UCM supports. The generic list is part of the file-based device configuration information that is maintained on the system. (Refer to Chapter 7 of the *HP OpenVMS System Manager's Manual*.)

  The installation process creates this list. A device that has no matching entry in the generic list is an unknown device type and cannot be configured.

- Tentative list

  This is a list of devices that UCM will configure if you tell it to. The tentative list, which is in memory, disappears when the server is restarted or when the system reboots.

- Permanent list

This list contains devices that UCM configures if the device is connected to the bus. The permanent list supplies a persistent name for a USB device – that is, the name is maintained across reboots and server restarts.

Persistent names work somewhat differently on devices that have or do not have serial numbers:

- On a device with a serial number, a persistent name always works.
- On a device without a serial number, a name is persistent only if you attach the device in the same place on the hub.

A running system has only one permanent list, which UCM reads from SYS$SYSTEM:USB$UCM_DEVICES.DAT. For most customers, this is the minimal file that OpenVMS provides in SYS$COMMON:[SYSEXE].

Caution

Never delete USB$UCM_DEVICES.DAT. Deleting this file might result in the inability to use your USB attached devices.

2.1.2.2 How UCM Configures Devices

At system startup time, the following steps occur:

1. OpenVMS starts the UCM server, which does the following:
   a. Reads the generic list of supported devices.
   b. Reads the permanent list for descriptions of devices.
2. UCM initializes an empty tentative list.
3. UCM turns on the USB bus; a device on the bus announces itself.
4. UCM checks for device data on the permanent list. If device data is on the permanent list, UCM loads it and makes it available.

   If device data is not on the list, UCM performs the next step.
5. UCM checks for data on the generic list. If device data is on this list, UCM uses it to make an entry in the tentative list.

   The data stays on the tentative list until the user either adds the device to or deletes the device from the permanent list.

Notes

- Unplugging a device does not delete a tentative item.
- Each replugging of a device creates a new tentative item.
- A device added to the permanent list following UCM startup is not configured (that is, it is not available for use) until the device is subsequently disconnected and reconnected (unplugged and replugged).
- Any modification to the permanent list creates a new version of the file.

6. Steps 3 through 5 repeat until all devices on all buses are processed.

7. UCM waits for a user request or until a device is plugged in or unplugged.

Log Files

UCM uses the following log file to record disconnections, connections, and errors:

```
SYS$MANAGER:USB$UCM_EVENTS.LOG
```

You do not need special permission to access the event log. However, you do need OPER privilege to use the UCM command SET LOG/NEW command to create a new log file. (Section 2.4 contains a table listing UCM commands and the privileges required to issue each command.)

2.2 Using UCM to Manage Devices and View Events

You can use UCM commands to select the devices you want to configure and to view USB events such as connections, disconnections, and errors. The following sections explain how to configure USB devices and how to view USB device information.

2.2.1 Configuring Devices

Before the system can configure a USB device, the device must have a corresponding entry on the permanent list. Although a few entries are included as part of USB installation, these entries provide only minimal support for a mouse and a keyboard. For most devices, you need to take certain steps to add an entry to the permanent list.

When you connect a USB device of a known type that has no entry on the permanent list, UCM uses information in the read-only generic list to create an entry in the tentative list. You must approve the entry before UCM creates an entry in the permanent list.

The following section explains how to create an entry in the permanent list.

Creating an Entry in the Permanent List

To set up a device to be configured, simply add the device as an entry in the permanent list. Once you do this, UCM recognizes the device each time you connect it.

In the following example, you connect a printer to the USB. The printer is a known device type; in other words, it has an entry in the generic list. However, it does not yet have an entry in the permanent list.

Follow these steps to configure the device:

1. Physically connect the printer.
2. Enter the UCM command to enter the UCM environment and to display a message about configured and unconfigured devices on your system:

   ```
   $ UCM
   UCM>
   ```

3. To display more information about the unconfigured device, enter the following command:

   ```
   $ UCM
   Universal Serial Bus Configuration Manager, Version V1.0
   ```

```
UCM> SHOW DEVICE /UNCONFIGURED
DEVICE
DEVICE_TYPE                     TENTATIVE
DEVICE_NAME_ROOT                LP
UNIT_NUMBER                     0
BUS                             1
PATH                            1.0.0.0.0.0
END_DEVICE

UCM>
```

Note that the display on your screen might be somewhat different from the one you see here.

4. Next, you must approve the entry by entering the ADD command. For example:

```
UCM> ADD DEVICE LPA0

UCM> EXIT
```

This command places the device information in the permanent list.

5. The last step before using the printer is to unplug the printer and reconnect it. This makes the device available for use. (If a device has no serial number, you must either plug it into the same port, or use the MODIFY command to indicate its new location.)

When you reconnect the printer, its serial number and vendor ID identify it as LPA0. UCM configures the device and makes it available for use.

Note that this step is not necessary if the UCM server is restarted or if the system is rebooted.

2.2.2 Viewing Events

The UCM event logger records events such as device connections and disconnections and certain types of errors. To see this information, you use the UCM utility SHOW EVENTS command. You can also use qualifiers to limit the display of various types of events.

Events stored in the event log include the following:

- A device was configured or deconfigured.
- A known device was connected but not configured.
- An unknown device was connected.
- Text messages were sent by USB drivers.

The following sections explain how to display information about unknown devices and configuration failures.

2.2.2.1 Information About Unknown Devices

The UCM records unknown device connections in its event log. You can view this information by adding the /TYPE=UNKNOWN qualifier to the SHOW EVENT command.

The information in the following example includes the vendor ID, the product ID, and other optional device-supplied information. If an unknown device is connected to the USB, you might want to view only events showing the activity of unknown devices for today; for example:

```
UCM> SHOW EVENTS /TYPE=UNKNOWN /SINCE=TODAY

Date        Time        Type         Priority Component
-------------------------------------------------------------------------------
22-AUG-2002 13:04:23.26 UNKNOWN      NORMAL   UCM UNKNOWN DEVICE
     Message: VENDOR_ID = 1118.PRODUCT_ID = 8.RELEASE_NUMBER = 256.BUS_NUMBER
     = 1.PATH = 1.0.0.0.0.0.DEVICE_CLASS = 0.DEVICE_SUB_CLASS =
     0.DEVICE_PROTOCOL = 0.NUMBER_OF_INTERFACES = 1.NUMBER_OF_CONFIGURATIONS
     = 1.MANUFACTURER_STRING = Microsoft.PRODUCT_STRING =
     Microsoft SideWinder Precision Pro (USB).CONFIGURATION_NUMBER = 0.

UCM>
```

Note that the display on your screen might be somewhat different from the one you see here.

2.2.2.2 Information about Configuration Failures

When UCM does not configure a device—because UCM cannot find an entry in the permanent list or because of a driver error—it stores this information in the event log. You can view such information using the SHOW EVENTS command and a qualifier that limits the display. For example:

```
UCM> SHOW EVENTS /SINCE=YESTERDAY

Date        Time        Type         Priority Component
-------------------------------------------------------------------------
28-AUG-2002 17:43:47.09 DRIVER       NORMAL   HUBDRIVER
        Message: Find a driver for DeviceClass/DeviceSubClass = 0x2/0x0

28-AUG-2002 17:43:47.09 UNKNOWN      NORMAL   UCM UNKNOWN DEVICE
        Message: VENDOR_ID = 4483.PRODUCT_ID = 16392.RELEASE_NUMBER =
        256.BUS_NUMBER = 3.PATH = 1.1.4.4.0.0.DEVICE_CLASS =
        2.DEVICE_SUB_CLASS = 0.DEVICE_PROTOCOL = 0.NUMBER_OF_INTERFACES =
        2.NUMBER_OF_CONFIGURATIONS = 2.MANUFACTURER_STRING
        = Compaq Computer Corp., Inc..PRODUCT_STRING = Compaq USB
        Modem.CONFIGURATION_NUMBER = 0.

28-AUG-2002 17:43:47.24 DRIVER       NORMAL   HUBDRIVER
        Message: hub_configure_device Unable to find Interface Driver


28-AUG-2002 17:43:47.24 DRIVER       NORMAL   HUBDRIVER
        Message: Find a driver for InterfaceClass/InterfaceSubClass/
        Protocol = 0xff/0xff/0xff

28-AUG-2002 17:43:49.17 UCM          CRITICAL ucm_config_request
        Message: %SYSTEM-W-DEVEXISTS, device unit already exists

28-AUG-2002 17:43:50.17 DRIVER       NORMAL   HUBDRIVER
        Message: Configured device TXA3 using driver SYS$YCDRIVER:

28-AUG-2002 17:43:56.11 DRIVER       NORMAL   HUBDRIVER
        Message: Device on bus 3 at port 1 bus tier 4 can exceed the
        bus power available

UCM> exit
```

Note that the display on your screen might be somewhat different from the one you see here.

The last message in this example, which is in bold type, indicates that there is insufficient power in the hub to supply the device. Therefore, UCM will not configure the device.

If no entry for the device is in the generic list, the log displays what is known about the device. If an error caused the failure, the error code is listed in the log.

2.3 UCM Usage Summary

The Universal Serial Bus (USB) Configuration Manager (UCM) utility allows you to connect a computer to a variety of USB devices using a single four-wire cable.

Format

UCM

Description

To invoke UCM, enter UCM at the DCL command prompt ($):

```
$ UCM

UCM>
```

At the UCM> prompt, you can enter any of the UCM commands described briefly in Section 2.4 and in more detail in the following sections.

Alternatively, you can enter UCM commands at the DCL prompt. For example:

```
$ UCM RELOAD
$
```

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

2.4 UCM Commands

The following table summarizes the UCM commands.

Command	Description	Privilege Required
ADD DEVICE	Allows you to add a new device to the collection of known USB devices.	SYSPRIV
DELETE DEVICE	Allows you to remove a device from the collection of known devices.	SYSPRIV
EXIT	Exits the UCM utility.	None
HELP	Provides online help information for using the UCM commands.	None
MODIFY DEVICE	Modifies the unit number or flags of an entry in the permanent list. The changes take effect immediately.	SYSPRIV
RELOAD	Reads the generic and permanent lists from disk.	SYSPRIV
RESTART	Restarts the configuration server.	CMKRNL
SET LOG/NEW	Creates a new version of the event log file.	OPER
SHOW DEVICE	Displays configured and unconfigured devices that are connected to the USB.	None
SHOW EVENTS	Displays events in the event log file.	None

ADD DEVICE

Allows you to add a new device to the collection of known USB devices.

Requires SYSPRIV privilege.

Format

ADD DEVICE device-name:

Parameter

device-name:
The name of the device whose characteristics are to be added. The device name has the form *ddcu*, where:

dd is the device code—for example, LP. (The driver name corresponds to the device code; in this case, the driver name would be SYS$LPDRIVER.)

c is the controller designation A through Z; unless UCM specifies a different letter, all USB devices are A.

u is the unit number (0 through 9999).

OpenVMS device names are made up of the two-character device code, followed by the controller designation, then by the unit number (which can be 1 to 4 characters long), then by a colon (:).

Qualifiers

/BUS_NUMBER=number
Specifies the USB bus number of the device. This parameter is required to identify a particular device on a system that has multiple USB buses. If you do not use this qualifier, the bus number defaults to zero.

The number can be any number from 0 through 25.

/PATH=(n1[.n2.n3.n4.n5.n6])
Specifies the path to the device on the bus. The path is used to uniquely identify a device if the device does not have a serial number. The path specification is a series of six or fewer nonzero numbers, where:

n1 is the number of the port on the root hub (at tier 0).

n2 through *n6* are port numbers for downstream hubs at tiers 1, 2, 3, 4, and 5. (If you do not specify trailing zeros, the UCM server supplies them.)

For example, /PATH=1.4.3 indicates that the device is plugged into port 3 of the second tier hub, which is plugged into port 4 of the first tier hub, which in turn is plugged into the root hub 1.

For a more detailed explanation of path specifications, see Figure 2–2 and the text that introduces the figure.

/UNIT_NUMBER=number
Unit numbers can be between 0 and 9999. By default, UCM selects the next available unit number. This qualifier allows you to change the unit number to suit your needs.

Example

```
$  UCM


    Universal Serial Bus Configuration Manager, Version V1.0

    UCM> SHOW DEVICE /UNCONFIGURED

    DEVICE
    DEVICE_TYPE                     TENTATIVE
    DEVICE_NAME_ROOT                AG
    UNIT_NUMBER                     0
    BUS                             1
    PATH                            1.0.0.0.0.0
    END_DEVICE

    UCM> ADD DEVICE AGA0:

    UCM> SHOW DEVICE /PERMANENT /FULL AGA0:

    DEVICE
    DEVICE_TYPE                     PERMANENT
    DEVICE_NAME_ROOT                AG
    UNIT_NUMBER                     0
    DRIVER                          SYS$AGDRIVER.EXE
    BUS_NUMBER                      1
    PATH                            1.0.0.0.0.0
    HID_USAGE_DATA                  65540
    BEGIN_INTERFACE
    HID_USAGE_DATA                  65540
    END_INTERFACE
    END_DEVICE

    UCM>
```

In this example, the first UCM command SHOW DEVICE/UNCONFIGURED, indicates that the device has not yet been configured. It displays only the information that appears in the generic list: the device name root, the unit number, the bus, and the path.

After the ADD DEVICE command, the second SHOW DEVICE command, with the /PERMANENT and /FULL qualifiers, displays the information in the permanent list. The list includes the name of the driver assigned to the device, the bus number, and the Human Interface Device (HID) usage data number, which is used to configures devices in the HID interface class. Examples of HID devices are keyboards, mice, joysticks, and so on.

DELETE DEVICE

Allows you to remove a device from the the permanent list.

Requires SYSPRIV privilege.

Format

DELETE DEVICE device-name:

Parameters

device-name:
The name of the device whose characteristics are to be deleted. The device name has the form *ddcu*,
where:

dd is the device code—for example, LP. (The driver name corresponds to the device code; in this case, the driver name would be SYS$LPDRIVER.)

c is the controller designation A through Z; unless UCM specifies a different letter, all USB devices are A.

u is the unit number (0 through 9999).

OpenVMS device names are made up of the two-character device code, followed by the controller designation, then by the unit number (which can be 1 to 4 characters long), then by a colon (:).

Example

```
$ UCM


   Universal Serial Bus Configuration Manager, Version V1.0

   UCM> SHOW DEVICE /PERMANENT AGA0:

   DEVICE
   DEVICE_TYPE                    PERMANENT
   DEVICE_NAME_ROOT               AG
   UNIT_NUMBER                    0
   BUS                            1
   PATH                           1.0.0.0.0.0
   END_DEVICE

   UCM> DELETE DEVICE AGA0:

   UCM> SHOW DEVICE /PERMANENT AGA0:

   %USB-E-NOSUCHDEV, Device name or device unit not found

   UCM>
```

In this example, the first SHOW DEVICE AGA0: command displays information about the device that is in the permanent list. After the DELETE DEVICE AGA0: command, the second SHOW DEVICE AGA0: command displays an error message indicating that the device is no longer in the permanent list.

EXIT

Stops the execution of UCM and returns control to DCL command level. You can also press Ctrl/Z to perform the same function.

Format

EXIT

HELP

Provides online help for using the UCM commands.

Format

HELP [command-name]

Parameter

command-name
The name of a UCM command. When you enter the HELP command with a command name, UCM displays a list of all the command keywords used with the command.

Example

```
UCM> HELP RESTART



   RESTART

        Restarts the configuration server.  This command should be used
        only if the server is no longer responding to configuration requests
        or if the client cannot get the server to respond to commands.  Use
        of this command requires the CMKRNL privilege.

        Format

          RESTART

    Additional information available:

     Qualifiers
     /CONFIRM

   RESTART Subtopic?
```

The HELP RESTART command describes the command, shows its format, and indicates what additional information is available, such as qualifiers. It then prompts you to enter the name of the /CONFIRM qualifier to display information about this qualifier.

MODIFY DEVICE

Allows you to modify the path and unit number of a device in the permanent list. The changes take place immediately.

Requires SYSPRIV privilege.

Format

MODIFY DEVICE device-name:

Parameter

device-name:
The name of the device whose characteristics are to be modified. The device name has the form *ddcu*, where:

dd is the device code—for example, LP. (The driver name corresponds to the device code; in this case, the driver name would be SYS$LPDRIVER.)

c is the controller designation A through Z; unless UCM specifies a different letter, all USB devices are A.

u is the unit number (0 through 9999).

OpenVMS device names are made up of the two-character device code, followed by the controller designation, then by the unit number (which can be 1 to 4 characters long), then by a colon (:).

Qualifiers

/BUS_NUMBER=number
Specifies the USB bus number of the device. This parameter is required to identify a particular device on a system that has multiple USB buses. If you do not use this qualifier, the bus number defaults to zero.

The number can be any number from 0 through 25.

/PATH=(n1[.n2.n3.n4.n5.n6])
Specifies the path to the device on the bus. The path is used to uniquely identify a device if the device does not have a serial number. The path specification is a series of six or fewer numbers,
where:

n1 is the number of the root hub (at tier 0).

n2 through *n6* are port numbers for downstream hubs at tiers 1, 2, 3, 4, and 5.

For example, /PATH=1.4.3 indicates that the device is in turn plugged into port 3 of the second tier, which is plugged into port 4 of the first tier, which is plugged into the root hub 1.

/UNIT_NUMBER=number
Unit numbers can be between 0 and 9999. By default, the configuration code selects the next available unit number. This qualifier allows you to change the unit number to suit your needs.

Example

```
$  UCM


    Universal Serial Bus Configuration Manager, Version V1.0

    UCM> SHOW DEVICE /UNCONFIGURED

    DEVICE
    DEVICE_TYPE                     TENTATIVE
    DEVICE_NAME_ROOT                AG
    UNIT_NUMBER                     0
    BUS                             1
    PATH                            1.0.0.0.0.0
    END_DEVICE

    UCM> ADD DEVICE AGA0:

    UCM> MODIFY DEVICE AGA0:/UNIT=9999

    UCM> SHOW DEVICE /PERMANENT /FULL AGA9999:

    DEVICE
    DEVICE_TYPE                     PERMANENT
    DEVICE_NAME_ROOT                AG
    UNIT_NUMBER                     9999
    DRIVER                          SYS$AGDRIVER.EXE
    BUS_NUMBER                      1
    PATH                            1.0.0.0.0.0
    HID_USAGE_DATA                  65540
    BEGIN_INTERFACE
    HID_USAGE_DATA                  65540
    END_INTERFACE
    END_DEVICE

    UCM>
```

The first SHOW DEVICE command displays information from the generic list about the unconfigured AG device. The ADD DEVICE command adds the device to the permanent list, and the MODIFY DEVICE command changes the unit number of the device. The second SHOW DEVICE command displays this change.

RELOAD

Forces the configuration server to reload the configuration data from the generic and permanent device files and to rebuild the lists. This allows you to add a new device type and lets the server find out about it without restarting UCM.

Requires SYSPRIV privilege.

Format

RELOAD

RESTART

Restarts the configuration server.

Note

Use this command only if the server no longer responds to configuration requests or client commands.

Requires CMKRNL privilege.

Format

RESTART

Qualifiers

/CONFIRM (default)
/NOCONFIRM
Asks you to confirm the restart of the configuration server. If you answer yes, the configuration server is restarted. If you answer no, the operation is not performed.

Example

```
$  UCM


    UCM> RESTART

    Restart UCM Server? [N]: yes

    Waiting for UCM Server image to exit....
    Waiting for UCM Server image to restart....
    %USB-S-SRVRRESTART, Identification of new UCM Server is 00000217

    UCM>
```

Following the RESTART command, UCM prompts you to confirm this command. The system assigns a new identification number to the UCM server when it restarts.

SET LOG

Tells the configuration server to create a new log file. You must use the /NEW qualifier with this command.

Requires OPER privilege.

Format

SET LOG /NEW

Qualifier

/NEW
Creates a new SYS$MANAGER:USB$UCM_EVENTS.DAT file. This qualifier is required with the SET LOG command.

SHOW DEVICE

Displays information about devices.

Format

SHOW DEVICE device-name:

Parameter

device-name:
The name of the device whose characteristics are to be displayed. The device name has the form *ddcu*, where:

dd is the device code—for example, LP. (The driver name corresponds to the device code; in this case, the driver name would be SYS$LPDRIVER.)

c is the controller designation A through Z; unless UCM specifies a different letter, all USB devices are A.

u is the unit number (0 through 9999.)

OpenVMS device names are made up of the two-character device code, followed by the controller designation, then by the unit number (which can be 1 to 4 characters long), then by a colon (:).

Display_Qualifiers

/BRIEF (default)
Displays summary information for each device.

/FULL
Displays complete information for each device.

Selection_Qualifiers

/ALL (default)
Displays all device entries, including those that the /CONFIGURED, /GENERIC, /PERMANENT, /PHYSICAL, and /UNCONFIGURED qualifiers display.

/CONFIGURED
Displays all the devices connected to the bus that have been configured successfully.

/GENERIC
Displays the devices that are on the generic device list.

/PERMANENT
Displays the devices for which the system automatically loads device drivers if the devices are plugged in.

/PHYSICAL
Displays the devices that are connected to the bus even if drivers for these devices are not loaded.

/UNCONFIGURED
Displays devices that are attached to the bus and that have drivers, but that do not have entries in the permanent list. (These are also known as tentative devices.)

You must execute an ADD DEVICE command to make these devices part of the permanent list. Once the drivers have been added, the device is automatically configured the next time it is plugged in.

Example

```
$  UCM


    UCM> SHOW DEVICE /PERMANENT /FULL DNA3:

    DEVICE
    DEVICE_TYPE                     PERMANENT
    DEVICE_NAME_ROOT                DN
    UNIT_NUMBER                     3
    DRIVER                          SYS$DNDRIVER.EXE
    USB_CONFIG_TYPE                 INTERFACE
    VENDOR_ID                       3519
    PRODUCT_ID                      768
    RELEASE_NUMBER                  4352
    BUS_NUMBER                      1
    PATH                            1.0.0.0.0.0
    DEVICE_CLASS                    0
    DEVICE_SUB_CLASS                0
    DEVICE_PROTOCOL                 0
    NUMBER_OF_INTERFACES            1
    CONFIGURATION_VALUE             2
    NUMBER_OF_CONFIGURATIONS        1
    SERIAL_NUMBER                   2B0301060D97A4C8
    MANUFACTURER_STRING             QTS
    PRODUCT_STRING                  USB 2.0 ATAPI Bridge
    CONFIGURATION_NUMBER            0
    BEGIN_INTERFACE
    INTERFACE_CLASS                 8
    INTERFACE_SUB_CLASS             6
    INTERFACE_PROTOCOL              80
    END_INTERFACE
    END_DEVICE

    UCM>
```

In this example, the SHOW DEVICE command displays complete information about DNA3:.

SHOW EVENTS

Displays important events that occur on the USB bus. Data displayed can include information about device events, such as removals, connections, unrecognized devices, new devices, and so on.

Format

SHOW EVENTS

Qualifiers

/BEFORE=time
Selects events that occurred before the specified time. You can specify *time* as an absolute time, as a combination of absolute and delta times, or as the keyword TODAY (default), TOMORROW, or YESTERDAY. Times are expressed in standard OpenVMS date/time format.

/OUTPUT=file-name
Writes the selected events to the specified file. By default, output is sent to the current SYS$OUTPUT device (usually your terminal).

You cannot use the /OUTPUT qualifier with the /PAGE qualifier.

/PAGE
/NOPAGE (default)
Controls how information is displayed. /PAGE displays events on one screen at a time.

You cannot use the /PAGE qualifier with the /OUTPUT qualifier.

/SINCE=time
Selects only those events that occurred on or after the specified time. You can specify time as absolute time, as a combination of absolute and delta times, or as the keyword TODAY (default) or YESTERDAY.

/TYPE=event-type
Selects only the specified type of events. Valid event-types are the following:

ALL	All event-types (default).
CONFIGURED	Device was recognized and configured.
DECONFIGURE	Device was removed from the bus.
DRIVER	Driver events.
UCM	UCM server events.
UNCONFIGURE	Device was recognized but not configured.
UNKNOWN	Event type is unknown.

/VALUE=event-number
Selects only the events specified by the event number. In a future version of this product, you will be able to use this qualifier as an alternative to the /TYPE qualifier for events that do not have an assigned keyword.

Example

```
$  UCM


   Universal Serial Bus Configuration Manager, Version V1.0

   UCM> SHOW EVENTS /SINCE=YESTERDAY

   USB EVENT LISTING
   -----------------
   Date        Time        Type         Priority Component
   ------------------------------------------------------------------------
   31-JUL-2002 11:46:20.76 DRIVER       NORMAL   HUBDRIVER
      Message: Find a driver for DeviceClass/DeviceSubClass = 0x9/0x0     1)

   31-JUL-2002 11:46:20.76 DRIVER       NORMAL   HUBDRIVER
      Message: Configured device UCM0 using driver SYS$HUBDRIVER:         2)

   31-JUL-2002 11:46:21.06 DRIVER       NORMAL   HUBDRIVER
      Message: Find a driver for DeviceClass/DeviceSubClass = 0x0/0x0     3)

   31-JUL-2002 11:46:21.31 UNKNOWN      NORMAL   UCM UNKNOWN DEVICE
      Message: VENDOR_ID = 3519.PRODUCT_ID = 768.RELEASE_NUMBER =
      4352.BUS_NUMBER = 1.PATH = 1.0.0.0.0.0.DEVICE_CLASS =
      0.DEVICE_SUB_CLASS = 0.DEVICE_PROTOCOL = 0.NUMBER_OF_INTERFACES =
      1.NUMBER_OF_CONFIGURATIONS = 1.SERIAL_NUMBER =
      2B0301060D97A4C8.MANUFACTUR                                         4)

   31-JUL-2002 11:46:21.31 DRIVER       NORMAL   HUBDRIVER
      Message: Find a driver for InterfaceClass/InterfaceSubClass/
      Protocol = 0x8/0x6/0x50                                             5)

   31-JUL-2002 11:46:21.46 DRIVER       NORMAL   HUBDRIVER
      Message: Configured device DNA3 using driver SYS$DNDRIVER:          6)

    1-AUG-2002 11:16:07.71 DECONFIGURED NORMAL   HUBDRIVER
      Message: Deconfiguring device on bus 1 at port 1 bus tier 1 usb
      address 2                                                           7)
```

Note that the display on your screen might be somewhat different from the one you see here.

Numbers in the example correspond to the following explanations:

1. UCM saw the root hub.
2. UCM configured the root hub.
3. UCM saw a new device.
4. Data about the new device in number 3 is displayed.
5. For the device in number 3, UCM attempted to locate a drive based on device interface data.
6. UCM found and loaded the driver for the device in number 3.
7. The device specified in the message line was unplugged.

3

Managing Workloads With Partitions

OpenVMS customers use systems that support hard and soft partitions in many different ways. To most effectively use these systems, customers can decide which configuration options best meet their computing and application needs.

This chapter describes how to use hard and soft partitions to ensure that applications run as efficiently as possible on new AlphaServer systems.

3.1 Using Hard and Soft Partitions on OpenVMS Systems

Hard partitioning is a physical separation of computing resources by hardware-enforced access barriers. It is impossible to read or write across a hard partition boundary, and there is no resource sharing between hard partitions.

Soft partitioning is a separation of computing resources by software-controlled access barriers. Soft partitions (subpartitions in some contexts) enable the sharing of hardware resources among several operating systems. Read and write access across a soft partition boundary is controlled by the operating system or applications. OpenVMS Galaxy is an implementation of soft partitioning.

How customers choose to partition their new AlphaServer ES or GS series systems depends on their computing environments and application requirements. When planning for partitioning, the amount of memory required by the applications and which operating system to run should be considered. When deciding how to configure an OpenVMS system that supports partitioning, look at the following questions:

- How many hard partitions do I need?
- How many soft partitions do I need?
- How small can I make the partitions?

Multiple hard partitions provide the maximum hardware security between partitions. A single soft partition running in a hard partition is equivalent to its running in a dedicated machine.

Multiple soft partitions within a given hard partition allow sharing resources such as CPUs and memory, and provide performance advantages.

3.2 OpenVMS Partitioning Guidelines

Using partitions on AlphaServer ES47/ES80/GS1280 systems is similar to using partitions on GS80/160/320 systems.

When deciding whether to use hard or soft partitions on the new AlphaServer ES or GS series systems, note the following:

- For AlphaServer ES47/ES80 and GS1280 systems, a network connection to the console or direct console access is needed.
- You can have multiple soft partitions within a hard partition.

- As an example of the licensing policy, you only need one cluster license for the entire AlphaServer ES or GS series system. It does not matter how many instances exist or how they are clustered, internally or externally. For additional information on licensing, see the Licensing section of these Release Notes and the licensing policy itself.

Note

In an OpenVMS Galaxy computing environment, MOP (Maintenance Operations Protocol) Booting is only supported on Instance 0.

3.3 Partitioning for the AlphaServer ES47/ES80/GS1280

- Hard partitions should be on *building block boundaries*, either on quad (QBB), system (SBB), or subsystem (SSBB) building blocks. GS80/160/320 systems use QBBs, and ES47/ES80/GS1280 systems use SBBs or SSBBs. A hard partition can contain more than one SBB or QBB. For high availability, HP recommends that hard partitions be created on quad or system building block boundaries.

  Only the GS1280 can use subsystem building blocks (SSBBs). A subsystem building block in a GS1280 is a 2P building block within an 8P building block.

  The AlphaServer ES47/ES80 system uses two-processor (2P) system building blocks, and the GS1280 uses eight-processor (8P) SBBs. A 2P SBB includes some I/O, but with an 8P SBB, the user must supply external I/O connections. Table 3–1 shows the two types of SBBs with their variants.
- Soft partitions do not have to be on building block boundaries.

Table 3–1 System Building Blocks

System	SBB Type	Model	Maximum Number of Hard Partitions
ES47/ES80	2P (2x1)	2	1
	2P (2x2)	4	2
ES80	2P (2x3)	6	3
	2P (2x4)	8	4
GS1280	8P (8x1)	8	4 - using 2P SSBBs
	8P (8x2)	16	8 - using 2P SSBBs
	8P (8x4)	32	16 - using 2P SSBBs
	8P (8x1)	8	1 - using 8P SBBs
	8P (8x2)	16	2 - using 8P SBBs
	8P (8x4)	32	4 - using 8P SBBs

Setup for partitions must follow setup rules.

Hard Partition Rules:

Each hard partition must have:

- Console access with the MBM or telnet
- An I/O drawer per partition (an internal drawer on the ES47/80 is sufficient)
- To avoid single points of failure, you must set up on system building block boundaries.

Required Console

Hard partitioning capability on the ES47/ES80/GS1280 Series requires a minimum of the V6.5 console set, which is available at the AlphaServer firmware web site:
`http://ftp.digital.com/pub/Digital/Alpha/firmware/readme.html`

Note that this web site address is case sensitive.

Required Patch Kit

Running an ES47/ES80/GS120 Series system configuration in a hard or soft partition other than 0 requires a minimum of the VMS731_PARTITIONING-V0100 kit.

Soft Partition Rules:

Each soft partition must have:

- Console access with the MBM or telnet
- An I/O drawer per partition (an internal drawer on the ES47/80 is sufficient)
- A primary CPU
- Private and shared memory for the operating system and applications. Shared memory is not necessary for independent instances but is required to create a Galaxy.

For maximum fault isolation and availability, ES47/ES80/GS1280 systems should be hard partitioned on system building block boundaries. For the ES47/ES80, this means on the 2P SBB boundary. For the GS1280, it is the 8P SBB boundary. When configured on system building block boundaries, hard partitions will have no single points of failure for the entire system. Power and cooling are self-contained within the hard partition. The interprocessor links will be turned off. This is the most robust form of partitioning. Note that a robust hard partition may contain multiple system building blocks.

For the GS1280 system, it is possible to hard partition the system building blocks into subsystem building blocks (SSBBs). An 8P SBB may be partitioned down to the 2P level. These hard partitions are separately powered and offer the ability to power off a dual-CPU module if needed for repair. This level of partitioning does not offer the fault isolation and robustness of a system that is partitioned on 8P SBB boundaries.

For hard partitions at the 8P or 2P SBB level, an individual serial console line per hard partition is supported. When an 8P SBB is subpartitioned into multiple hard partitions, the serial console can only connect to one subpartition at a time. If simultaneous access to all subpartition consoles is needed, then telnet sessions across the management LAN must be used. Section 3.3.1 describes the partition setup process, and Section 3.3.2 illustrates configuration setup with two examples.

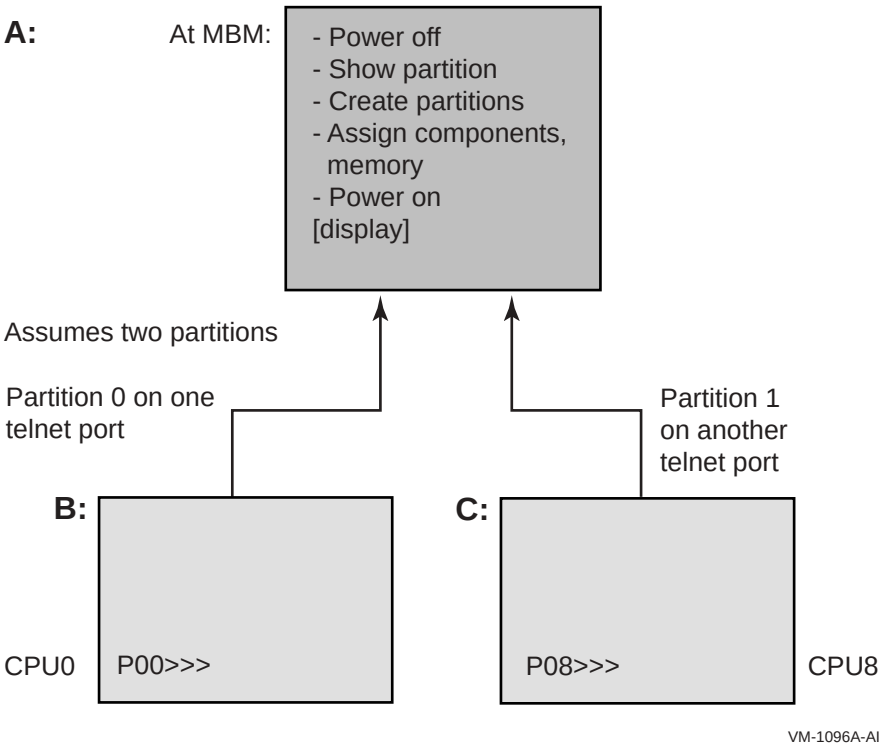
3.3.1 Process for Partition Setup

The basic process for setting up hard and soft partitions at the Management Backplane Module (MBM) is the following:

1. Create hard and soft partitions.
2. Assign CPU, IO, and memory resources to the partitions.
3. Power on CPUs and connect to consoles.
4. Check and reset console environment variables if necessary.

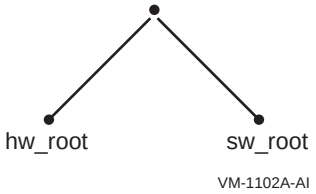
This process is illustrated in Figure 3–1. Starting with the new MBM firmware, T2.1-8, the `save partition` command is still implemented but no longer required. Each time a `create` or `delete` partition command is executed, an implicit `save` commits the changes to non-volatile storage. Older versions of MBM firmware still require an explicit `save`, otherwise, the changes are only temporary and are lost if AC power is taken away or a `reset -micro` is done.

Figure 3–1 Partitioning Sequence



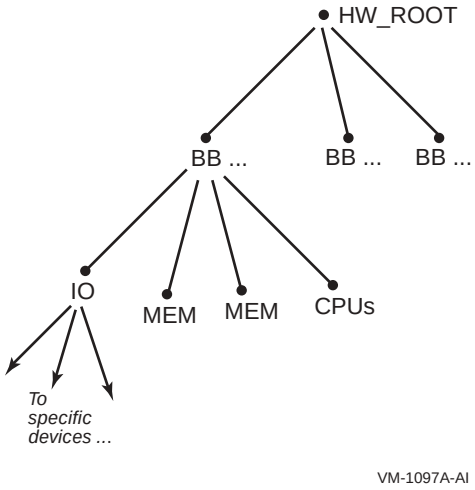
Physical hardware and ownership relationships are represented as branches of a single configuration tree in each system. Partitions, both hard and soft, can be thought of as ownership containers representing the accessibility and visibility of all resources in the configuration. The top level configuration branch shown in Figure 3–2 includes both the hardware and software configuration trees.

Figure 3–2 System Configuration Tree



In the hardware configuration tree shown in Figure 3–3, the physical nature of the box is delimited. Each bullet or filled circle represents a **node** in the tree (where node is not an individual computer but a point of connection in the tree). From the hardware root, the tree divides to the building blocks, and within each building block to the major system categories such as memory, input/output, and CPU. At a lower level in the configuration tree, within I/O, for example, the tree branches to individual devices on the system, and so on. Each node in the tree has a definition that includes its parent, its siblings, its children, and its ownership.

Figure 3–3 Hardware Configuration Tree



The scope of a partition is always on a branch, up or down, but never across branches. A soft partition, as shown in Figure 3–4, always looks up its tree for potentially available resources. The hard partition owns assigned resources that are available to all nodes below it.

In general, resources that are in use by a specific instance of an operating system are owned by the soft partition that represents that partition in the configuration tree. The cooperative **push model** in effect with multiple instances in a hard partition dictates that resources owned by a given instance can only be given away, not taken. A resource owned by nodes further up the tree may be used

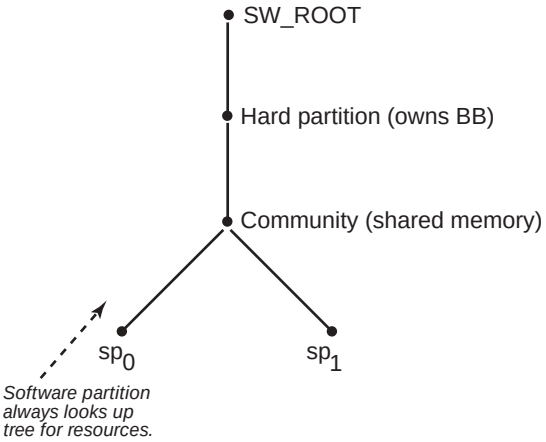
cooperatively (for example, shared memory at the community level), assigned down the tree to a specific soft partition, or back up the tree where it becomes available to potentially multiple soft partitions.

Direct assignment of resources between soft partitions is called migration; only CPUs can move between soft partitions. The migration partitioning feature is not specific to Galaxy: CPUs can be migrated from any soft partition to another in the same hard partition without direct communication between them. All management of CPU resources is initiated with OpenVMS DCL `SET/STOP CPU` commands.

Migration Restriction

You cannot migrate a CPU that has attached IO. To check if there is attached IO, either check the physical configuration for a hose attached to a CPU, or use the MBM show partition output: in the IOPs section, a CPU with attached IO has a dashed line; the CPU ID is the corresponding PID in the CPUs section of the show partition output. (See Section 3.3.2 for an example.)

Figure 3–4 Software Configuration Tree



3.3.2 Configuration Examples

The following example illustrates setup on a 32P system for hard partitions. For information on using GCM, the Graphical Configuration Manager, to view and manage a partitioned configuration, see the GCM chapter of the *OpenVMS Alpha Partitioning and Galaxy Guide*.

For information on the server management command line interface (CLI), see the CLI reference manual available at the following location:

`http://h18002.www1.hp.com/alphaserver/gs1280/gs1280_tech.html`

or by following the links from http://www.hp.com -> Servers -> AlphaServer Systems -> AlphaServer SC, GS, ES and DS Systems -> AlphaServer GS1280 -> support documentation -> CLI reference.

Note

The server management CLI is case insensitive.

Hard and soft partition names are limited to 19 characters (alphanumeric and underscore only).

Setup on a 32P GS1280 System

The following example shows setup of three hard partitions on a 32-processor system. Two partitions have eight processors each; the other partition has 16 processors. The partitions are assigned on 8P SBB boundaries.

In this example, the consoles can be accessed on telnet ports 323, 324, and 325. The hard partitions are called *part0*, *part1* and *part2*; each is created with a maximum of 16 CPUs. The subpartition type is soft. CPU and IO resources are then assigned to each partition. The assign component command assigns the entire SBB to the component; in the example, each drawer in the cabinet has 8 processors.

Please refer to the CLI_Reference.pdf manual mentioned earlier for changes to these commands.

```
Welcome - GS1280 Server Manager - T2.1-5
MBM> create partition -hp part0 255 soft
MBM> create partition -hp part1 255 soft
MBM> create partition -hp part2 255 soft
MBM> assign component -hp part0 -cabinet 0 -drawer 0 sbb
MBM> assign component -hp part1 -cabinet 0 -drawer 1 sbb
MBM> assign component -hp part2 -cabinet 0 -drawer 2 sbb
MBM> assign component -hp part2 -cabinet 0 -drawer 3 sbb

MBM> show partition

------------------------------------------------------------------
Hard Partition : HP Name = part0, HP No.= 0, SP count = 2
Attributes     : max CPUs = 16, SP type = soft, Non-stripe
Physical Memory: 16384MB (16.000GB)

 Community Memory: 0MB (0.000GB)

 Sub Partition: HP Name = part0, HP No.= 0
                SP Name = Default_SP, SP No.= 0
                State = Not Running, Telnet port = 323

  Assigned Memory: unspecified

  CPUs:
      Cab  Drw  CPU   (NS,EW)   PID    Type
       0    0    0    ( 0,0 )     0    Non-primary
       0    0    2    ( 0,1 )     2    Non-primary
       0    0    4    ( 0,2 )     4    Non-primary
       0    0    6    ( 0,3 )     6    Non-primary
       0    0    1    ( 1,0 )     1    Non-primary
       0    0    3    ( 1,1 )     3    Non-primary
       0    0    5    ( 1,2 )     5    Non-primary
       0    0    7    ( 1,3 )     7    Non-primary
```

```
  IOPs:
             SBB                              PCI Drawer
      Cab  Drw  IOP   (NS,EW)           Cab  Drw  IOR
       0    0    0    ( 0,0 ) ---------  2    4    0
       0    0    2    ( 0,1 )
       0    0    4    ( 0,2 )
       0    0    6    ( 0,3 )
       0    0    1    ( 1,0 )
       0    0    3    ( 1,1 )
       0    0    5    ( 1,2 )
       0    0    7    ( 1,3 )

 Sub Partition: HP Name = part0, HP No.= 0
                SP Name = Free_Pool, SP No.= 255
                State = Not Running
  Free Memory: 0MB (0.000GB)

  CPUs: None

  IOPs: None

-----------------------------------------------------------------
Hard Partition : HP Name = part1, HP No.= 1, SP count = 2
Attributes     : max CPUs = 16, SP type = soft, Non-stripe
Physical Memory: 16384MB (16.000GB)

 Community Memory: 0MB (0.000GB)

 Sub Partition: HP Name = part1, HP No.= 1
                SP Name = Default_SP, SP No.= 0
                State = Not Running, Telnet port = 324

  Assigned Memory: unspecified

  CPUs:
      Cab  Drw  CPU   (NS,EW)   PID    Type
       0    1    0    ( 2,0 )    0    Non-primary
       0    1    2    ( 2,1 )    2    Non-primary
       0    1    4    ( 2,2 )    4    Non-primary
       0    1    6    ( 2,3 )    6    Non-primary
       0    1    1    ( 3,0 )    1    Non-primary
       0    1    3    ( 3,1 )    3    Non-primary
       0    1    5    ( 3,2 )    5    Non-primary
       0    1    7    ( 3,3 )    7    Non-primary

  IOPs:
             SBB                              PCI Drawer
      Cab  Drw  IOP   (NS,EW)           Cab  Drw  IOR
       0    1    0    ( 2,0 ) ---------  2    5    0
       0    1    2    ( 2,1 )
       0    1    4    ( 2,2 )
       0    1    6    ( 2,3 )
       0    1    1    ( 3,0 )
       0    1    3    ( 3,1 )
       0    1    5    ( 3,2 )
       0    1    7    ( 3,3 )

 Sub Partition: HP Name = part1, HP No.= 1
                SP Name = Free_Pool, SP No.= 255
                State = Not Running
  Free Memory: 0MB (0.000GB)

  CPUs: None

  IOPs: None
```

```
-----------------------------------------------------------------
Hard Partition : HP Name = part2, HP No.= 2, SP count = 2
Attributes     : max CPUs = 16, SP type = soft, Non-stripe
Physical Memory: 36864MB (36.000GB)

 Community Memory: 0MB (0.000GB)

 Sub Partition: HP Name = part2, HP No.= 2
                SP Name = Default_SP, SP No.= 0
                State = Not Running, Telnet port = 325

  Assigned Memory: unspecified

  CPUs:
      Cab  Drw  CPU   (NS,EW)   PID    Type
       0    2    0    ( 0,4 )    0    Non-primary
       0    2    2    ( 0,5 )    2    Non-primary
       0    2    4    ( 0,6 )    4    Non-primary
       0    2    6    ( 0,7 )    6    Non-primary
       0    2    1    ( 1,4 )    1    Non-primary
       0    2    3    ( 1,5 )    3    Non-primary
       0    2    5    ( 1,6 )    5    Non-primary
       0    2    7    ( 1,7 )    7    Non-primary
       0    3    0    ( 2,4 )    8    Non-primary
       0    3    2    ( 2,5 )   10    Non-primary
       0    3    4    ( 2,6 )   12    Non-primary
       0    3    6    ( 2,7 )   14    Non-primary
       0    3    1    ( 3,4 )    9    Non-primary
       0    3    3    ( 3,5 )   11    Non-primary
       0    3    5    ( 3,6 )   13    Non-primary
       0    3    7    ( 3,7 )   15    Non-primary

  IOPs:
              SBB                              PCI Drawer
      Cab  Drw  IOP   (NS,EW)            Cab  Drw  IOR
       0    2    0    ( 0,4 ) ---------  2    6    0
       0    2    2    ( 0,5 )
       0    2    4    ( 0,6 )
       0    2    6    ( 0,7 )
       0    2    1    ( 1,4 )
       0    2    3    ( 1,5 )
       0    2    5    ( 1,6 )
       0    2    7    ( 1,7 )
       0    3    0    ( 2,4 ) ---------  2    7    0
       0    3    2    ( 2,5 )
       0    3    4    ( 2,6 )
       0    3    6    ( 2,7 )
       0    3    1    ( 3,4 )
       0    3    3    ( 3,5 )
       0    3    5    ( 3,6 )
       0    3    7    ( 3,7 )

 Sub Partition: HP Name = part2, HP No.= 2
                SP Name = Free_Pool, SP No.= 255
                State = Not Running
  Free Memory: 0MB (0.000GB)

  CPUs: None

  IOPs: None

MBM> save partition
```

The system is then powered on and diagnostics run.

```
MBM> power on -all
[2003/04/16 08:05:31]
~PCO-I-(pco_04) Powering on partition. HP: part0
[2003/04/16 08:05:32]
~PCO-I-(pco_03) Powering on partition. HP: part1
[2003/04/16 08:05:32]
~PCO-I-(pco_01) Powering on partition. HP: part2
[2003/04/16 08:05:51]
~PCO-I-(pco_04) Running diagnostics on HP: part0
[2003/04/16 08:05:55]
~PCO-I-(pco_03) Running diagnostics on HP: part1
[2003/04/16 08:06:00]
~PCO-I-(pco_01) Running diagnostics on HP: part2
[2003/04/16 08:06:50]
~PCO-I-(pco_04) Diagnostics completed on HP: part0
[2003/04/16 08:06:50]
~PCO-I-(pco_04) HP:part0 SP:Default_SP Primary is NS:0 EW:0
 which is cab:00 drw:0 cpu:0 [2003/04/16 08:06:51]
~PCO-I-(pco_04) Loading SRM on Primary for HP: part0,
 SP: Default_SP. [2003/04/16 08:06:54]
~PCO-I-(pco_03) Diagnostics completed on HP: part1
 [2003/04/16 08:06:54]
~PCO-I-(pco_03) HP:part1 SP:Default_SP Primary is NS:2 EW:0
 which is cab:00 drw:1 cpu:0 [2003/04/16 08:06:54]
~PCO-I-(pco_03) Loading SRM on Primary for HP: part1,
 SP: Default_SP. [2003/04/16 08:06:55]
~PCO-I-(pco_04) Powered On HP:part0
[2003/04/16 08:06:59]
~PCO-I-(pco_03) Powered On HP:part1
[2003/04/16 08:07:24]
~PCO-I-(pco_01) Diagnostics completed on HP: part2
[2003/04/16 08:07:25]
~PCO-I-(pco_01) HP:part2 SP:Default_SP Primary is NS:0 EW:4
 which is cab:00 drw:2 cpu:0 [2003/04/16 08:07:25]
~PCO-I-(pco_01) Loading SRM on Primary for HP: part2,
 SP: Default_SP. [2003/04/16 08:07:29]
~PCO-I-(pco_01) Powered On HP:part2
MBM>
```

3.4 OpenVMS Galaxy Support

OpenVMS Galaxy software controls operating system instances through shared memory; it implements a resource-sharing association between instances in multiple soft partitions. Multiple independent operating system instances can run in multiple soft partitions without Galaxy. For information about OpenVMS Galaxy concepts, refer to the *HP OpenVMS Alpha Partitioning and Galaxy Guide*.

The configuration tree enables the manipulation of resources up and down the tree within each hard partition. The tree defines a hierarchy of physical connectivity and resource ownership. Hardware resources can be assigned ownership at one of several levels, although resources are allocated and used by an **instance**. An instance is an operating system running within a soft partition.

Software partitions govern system resources. CPUs are used only when owned by a soft partition and manipulated by the instance running there; however, CPUs may be owned by higher levels of the configuration tree—making them visible and available to all instances equally. If boot ownership is not set before a platform box is powered up, and a CPU module is later added to the system, that CPU is owned by the hardware partition to which it was added, and it is programmatically assignable to any of the soft partitions within that hard partition. (The ES47/ES80/GS1280 does not support hot swap of components.)

As soon as an instance of the operating system boots, all its resources are exclusively owned by the soft partition to which it is assigned: only that instance can manipulate its state characteristics. Consequently, it is important to consider the initial allocation of CPUs at powerup —even those that do not currently exist – to provide the best division of resources when they become available.

To create multiple soft partitions within a single hard partition, use standard partitioning procedures, as previously described, to create Galaxy configurations with instances running in these soft partitions. Figure 3–5 illustrates how memory is used in a Galaxy instance. Soft partitions require the OpenVMS minimum for memory.

The Galaxy ID is within the hard partition and can span the hard partition. That is, if you have two hard partitions and you run Galaxy in both, each Galaxy will have its own unique Galaxy ID. Keep this in mind when you use network management tools; they will also see two Galaxy environments when two Galaxy IDs exist.

Figure 3–6 illustrates four hard partitions (0, 1, 2, 3), each of which has a unique name such as hp_0. Within hard partition 0, a Galaxy is shown that contains two soft partitions, the left-most sp_1 and sp_0. Other Galaxies could exist in each of the other hard partitions, as could multiple independent operating system instances if none of them creates or joins a Galaxy. You can run an operating system instance in any soft partition.

There is one Galaxy per community because the shared memory used by the Galaxy is owned by the community. This makes it visible and accessible to all instances in the Galaxy membership. All instances running in soft partitions below the community node are potentially eligible to join the Galaxy there. Member nodes may take advantage of the shared resources controlled by the Galaxy. Although independent instances are still able to assign and migrate CPUs, only members of the Galaxy can take advantage of the benefits of shared memory.

Figure 3–5 How Memory is Used

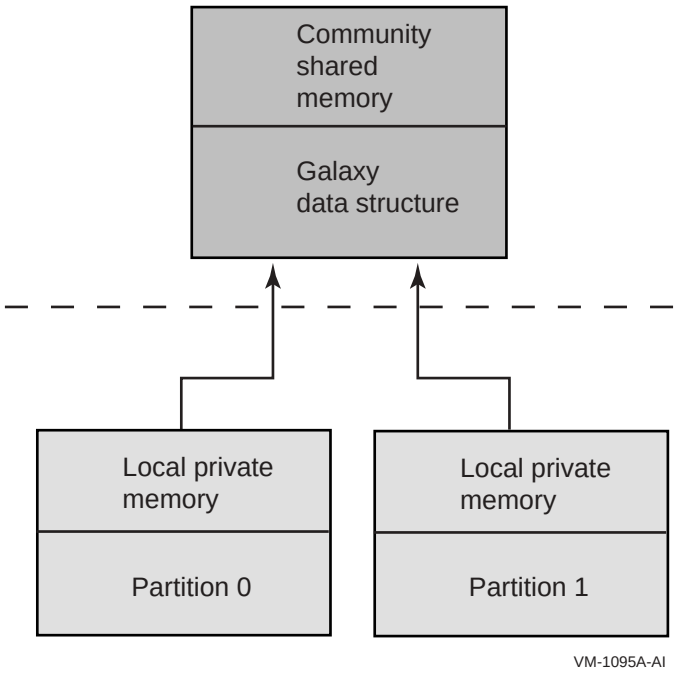
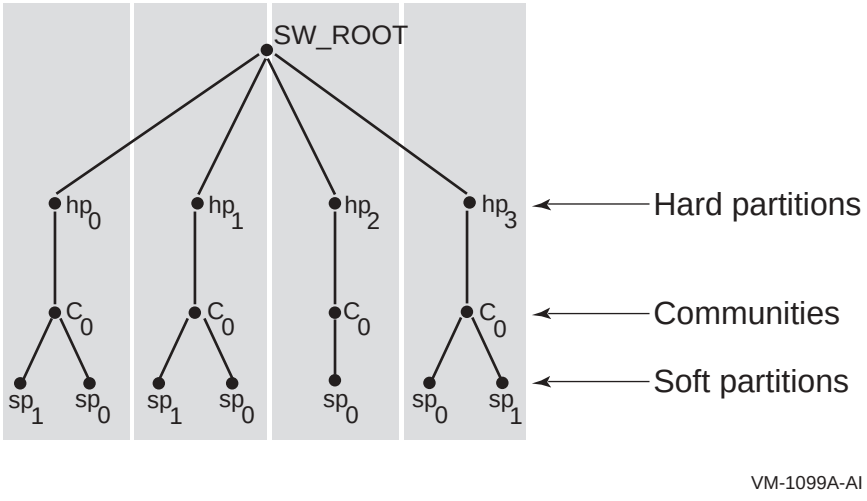


Figure 3–6 Soft Partitioning in the Configuration Tree



There can be only one community per hard partition.

4

Creating an OpenVMS Galaxy on AlphaServer ES47/ES80/GS1280 Systems

This chapter describes the process of creating an OpenVMS Galaxy computing environment on AlphaServer ES47/ES80/GS1280 systems.

4.1 Step 1: Choose a Configuration and Determine Hardware Requirements

Unlike previous AlphaServer platforms, on the ES47/ES80/GS1280, Galaxy partitions are set up outside the SRM console and there are no environment variables to set.

OpenVMS Alpha Version 7.3-1 supports the maximum configurations for the AlphaServer systems listed in Table 4–1. Some shared memory and private memory is used for each partition; the minimum amount of memory is as required by OpenVMS.

Table 4–1 ES47/ES80/GS1280 Configurations

ES47	ES80	GS1280/8	GS1280/16	GS1280/32
2 instances	4 instances with embedded IO	2 instances	4 instances (4 IO7 ports required)	8 4P instances (8 IO7 ports required)
2, 2P SBBs	4, 2P SBBs	1, 8P SBB	2, 8P SBBs	4, 8P SBBs
4 CPUs	8 CPUs	8 CPUs	16 CPUs	32 CPUs

4.2 Step 2: Set Up the Hardware

When you have acquired the necessary hardware for your configuration, follow the procedures in the appropriate hardware manuals to assemble it.

4.3 Step 3: Create a System Disk

Decide whether to use:

- one system disk per instance or
- a cluster common-disk

4.4 Step 4: Install OpenVMS Alpha

No special installation procedures are required to run OpenVMS Galaxy software. Galaxy functionality is included in the base operating system and can be enabled or disabled using the console command and system parameter values described later in this chapter.

For more information about installing the OpenVMS Alpha operating system, see the *OpenVMS Alpha Version 7.3–1 Upgrade and Installation Manual.*

4.4.1 Required Patches

Check for current revision numbers and how to obtain firmware patches at the following web location:

`http://h18002.www1.hp.com/alphaserver/gs1280/gs1280_tech.html`

or by following the links from `http://www.hp.com` -> Servers -> AlphaServer Systems -> AlphaServer SC, GS, ES and DS Systems -> AlphaServer GS1280 -> support documentation -> firmware updates.

4.4.2 OpenVMS Galaxy Licensing Information

In a Galaxy environment, the OPENVMS-GALAXY license units are checked during system startup and whenever a CPU reassignment between instances occurs.

If you attempt to start a CPU and there are insufficient OPENVMS-GALAXY license units to support it, the CPU remains in the instance's configured set but is stopped. You can subsequently load the appropriate license units and start the stopped CPU while the system is running.

For additional information on licensing, refer to Section 1.20 and Section 1.21.

4.5 Step 5: Set Up Partitions

When you have installed the operating system, you can set up the Galaxy partitions as shown in the examples in this section.

4.5.1 AlphaServer GS1280 Example

This example for an AlphaServer GS1280 assumes you are configuring an OpenVMS Galaxy computing environment with:

1 hard partition
3 soft partitions
256M of shared memory

In this example, there are three soft partitions in a single hard partition on a 32P GS1280. The first partition comprises the first two 8P drawers. The second partition has one 8P drawer, as does the third partition. There are 256 megabytes of shared memory. As seen in the show partition display, the consoles for the various partitions can be accessed via telnet ports. Note that the show partition display will change after a power-on command to show which CPU was selected as primary.

Note

Use the assign command only to set up shared memory. Private memory cannot be assigned.

Part A: Setting Up the Galaxy

```
Welcome - GS1280 Server Manager - V2.1-8
MBM> create partition -hp hp0 255 soft
MBM> create partition -hp hp0 -sp sp0
MBM> create partition -hp hp0 -sp sp1
MBM> create partition -hp hp0 -sp sp2
MBM> assign component -cab 0 -drawer 0 sbb -hp hp0 -sp sp0
MBM> assign component -cab 0 -drawer 1 sbb -hp hp0 -sp sp0
MBM> assign component -cab 0 -drawer 2 sbb -hp hp0 -sp sp1
MBM> assign component -cab 0 -drawer 3 sbb -hp hp0 -sp sp2
MBM> assign mem -hp hp0 256mb -com
MBM> show partition

------------------------------------------------------------------
Hard Partition : HP Name = hp0, HP No.= 0, SP count = 4
Attributes     : max CPUs = 255, SP type = soft, Non-stripe
Physical Memory: 69632MB (68.000GB)

 Community Memory: 256MB (0.250GB)

 Sub Partition: HP Name = hp0, HP No.= 0
                SP Name = sp0, SP No.= 0
                State = Not Running, Telnet port = 323

  Assigned Memory: unspecified

  CPUs:
      Cab  Drw  CPU   (NS,EW)   PID    Type
       0    0    0    ( 0,0 )     0    Non-primary
       0    0    2    ( 0,1 )     2    Non-primary
       0    0    4    ( 0,2 )     4    Non-primary
       0    0    6    ( 0,3 )     6    Non-primary
       0    0    1    ( 1,0 )     1    Non-primary
       0    0    3    ( 1,1 )     3    Non-primary
       0    0    5    ( 1,2 )     5    Non-primary
       0    0    7    ( 1,3 )     7    Non-primary
       0    1    0    ( 2,0 )     8    Non-primary
       0    1    2    ( 2,1 )    10    Non-primary
       0    1    4    ( 2,2 )    12    Non-primary
       0    1    6    ( 2,3 )    14    Non-primary
       0    1    1    ( 3,0 )     9    Non-primary
       0    1    3    ( 3,1 )    11    Non-primary
       0    1    5    ( 3,2 )    13    Non-primary
       0    1    7    ( 3,3 )    15    Non-primary

  IOPs:
              SBB                              PCI Drawer
      Cab  Drw  IOP   (NS,EW)            Cab  Drw  IOR
       0    0    0    ( 0,0 ) ---------  2    0    0
       0    0    2    ( 0,1 )
       0    0    4    ( 0,2 )
       0    0    6    ( 0,3 )
       0    0    1    ( 1,0 )
       0    0    3    ( 1,1 )
       0    0    5    ( 1,2 ) ---------  1    4    0
       0    0    7    ( 1,3 )
       0    1    0    ( 2,0 ) ---------  2    1    0
       0    1    2    ( 2,1 )
       0    1    4    ( 2,2 )
       0    1    6    ( 2,3 )
       0    1    1    ( 3,0 )
       0    1    3    ( 3,1 )
       0    1    5    ( 3,2 )
       0    1    7    ( 3,3 )
```

```
Sub Partition: HP Name = hp0, HP No.= 0
               SP Name = sp1, SP No.= 1
               State = Not Running, Telnet port = 324

 Assigned Memory: unspecified

 CPUs:
     Cab  Drw  CPU   (NS,EW)   PID    Type
      0    2    0    ( 0,4 )    0    Non-primary
      0    2    2    ( 0,5 )    2    Non-primary
      0    2    4    ( 0,6 )    4    Non-primary
      0    2    6    ( 0,7 )    6    Non-primary
      0    2    1    ( 1,4 )    1    Non-primary
      0    2    3    ( 1,5 )    3    Non-primary
      0    2    5    ( 1,6 )    5    Non-primary
      0    2    7    ( 1,7 )    7    Non-primary

 IOPs:
             SBB                              PCI Drawer
     Cab  Drw  IOP   (NS,EW)            Cab  Drw  IOR
      0    2    0    ( 0,4 ) ---------  2    2    0
      0    2    2    ( 0,5 )
      0    2    4    ( 0,6 )
      0    2    6    ( 0,7 )
      0    2    1    ( 1,4 )
      0    2    3    ( 1,5 )
      0    2    5    ( 1,6 )
      0    2    7    ( 1,7 )

Sub Partition: HP Name = hp0, HP No.= 0
               SP Name = sp2, SP No.= 2
               State = Not Running, Telnet port = 325

 Assigned Memory: unspecified

 CPUs:
     Cab  Drw  CPU   (NS,EW)   PID    Type
      0    3    0    ( 2,4 )    8    Non-primary
      0    3    2    ( 2,5 )   10    Non-primary
      0    3    4    ( 2,6 )   12    Non-primary
      0    3    6    ( 2,7 )   14    Non-primary
      0    3    1    ( 3,4 )    9    Non-primary
      0    3    3    ( 3,5 )   11    Non-primary
      0    3    5    ( 3,6 )   13    Non-primary
      0    3    7    ( 3,7 )   15    Non-primary

 IOPs:
             SBB                              PCI Drawer
     Cab  Drw  IOP   (NS,EW)            Cab  Drw  IOR
      0    3    0    ( 2,4 ) ---------  2    3    0
      0    3    2    ( 2,5 )
      0    3    4    ( 2,6 )
      0    3    6    ( 2,7 )
      0    3    1    ( 3,4 )
      0    3    3    ( 3,5 )
      0    3    5    ( 3,6 )
      0    3    7    ( 3,7 )

Sub Partition: HP Name = hp0, HP No.= 0
               SP Name = Free_Pool, SP No.= 255
               State = Not Running
 Free Memory: 0MB (0.000GB)

 CPUs: None

 IOPs: None

MBM> power on -all
 [...]
```

Part B: Output from Instance 0:

```
~PCO-I-(pco_01) Powering on partition. HP: hp0
starting console on CPU 0
initialized idle PCB
initializing semaphores
initializing heap
initial heap 700c0
memory low limit = 54c000 heap = 700c0, 1fffc0
initializing driver structures
initializing idle process PID
initializing file system
initializing timer data structures
lowering IPL
CPU 0 speed is 1150 MHz
create dead_eater
create poll
create timer
create powerup
entering idle loop
access NVRAM
Get Partition DB
hpcount = 1, spcount = 4, ev7_count = 32, io7_count = 5
hard_partition = 0
IO7-100 (Pass 3) at PID 8
IO7 North port speed is 191 MHz
Hose 32 - 33 MHz PCI
Hose 33 - 66 MHz PCI
Hose 34 - 66 MHz PCI
Hose 35 - 4X AGP
IO7-100 (Pass 3) at PID 0
IO7 North port speed is 191 MHz
Hose 0 - 33 MHz PCI
Hose 1 - 66 MHz PCI
Hose 2 - 66 MHz PCI
Hose 3 - 4X AGP
IO7-100 (Pass 3) at PID 5
IO7 North port speed is 191 MHz
Hose 20 - 33 MHz PCI
Hose 21 - 33 MHz PCI
Hose 22 - 33 MHz PCI
Hose 23 - 4X AGP
0 sub-partition 0:   start:00000000 00000000   size:00000000 80000000
PID 0 console memory base: 0, 2 GB
1 sub-partition 0:   start:00000004 00000000   size:00000000 80000000
PID 1 memory: 400000000, 2 GB
2 sub-partition 0:   start:00000008 00000000   size:00000000 80000000
PID 2 memory: 800000000, 2 GB
3 sub-partition 0:   start:0000000c 00000000   size:00000000 80000000
PID 3 memory: c00000000, 2 GB
4 sub-partition 0:   start:00000020 00000000   size:00000000 80000000
PID 4 memory: 2000000000, 2 GB
5 sub-partition 0:   start:00000024 00000000   size:00000000 80000000
PID 5 memory: 2400000000, 2 GB
6 sub-partition 0:   start:00000028 00000000   size:00000000 80000000
PID 6 memory: 2800000000, 2 GB
7 sub-partition 0:   start:0000002c 00000000   size:00000000 80000000
PID 7 memory: 2c00000000, 2 GB
8 sub-partition 0:   start:00000040 00000000   size:00000000 80000000
PID 8 memory: 4000000000, 2 GB
9 sub-partition 0:   start:00000044 00000000   size:00000000 80000000
PID 9 memory: 4400000000, 2 GB
10 sub-partition 0:   start:00000048 00000000   size:00000000 80000000
PID 10 memory: 4800000000, 2 GB
11 sub-partition 0:   start:0000004c 00000000   size:00000000 80000000
```

```
PID 11 memory: 4c00000000, 2 GB
12 sub-partition 0:   start:00000060 00000000   size:00000000 80000000
PID 12 memory: 6000000000, 2 GB
13 sub-partition 0:   start:00000064 00000000   size:00000000 80000000
PID 13 memory: 6400000000, 2 GB
14 sub-partition 0:   start:00000068 00000000   size:00000000 80000000
PID 14 memory: 6800000000, 2 GB
15 sub-partition 0:   start:0000006c 00000000   size:00000000 78000000
PID 15 memory: 6c00000000, 1.875 GB
0 sub-partition 1:   start:00000080 00000000   size:00000000 80000000
1 sub-partition 1:   start:00000084 00000000   size:00000000 80000000
2 sub-partition 1:   start:00000088 00000000   size:00000000 80000000
3 sub-partition 1:   start:0000008c 00000000   size:00000000 80000000
4 sub-partition 1:   start:000000a0 00000000   size:00000000 80000000
5 sub-partition 1:   start:000000a4 00000000   size:00000000 80000000
6 sub-partition 1:   start:000000a8 00000000   size:00000000 80000000
7 sub-partition 1:   start:000000ac 00000000   size:00000000 7c000000
0 sub-partition 2:   start:000000c0 00000000   size:00000000 80000000
1 sub-partition 2:   start:000000c4 00000000   size:00000000 80000000
2 sub-partition 2:   start:000000c8 00000000   size:00000001 00000000
3 sub-partition 2:   start:000000cc 00000000   size:00000001 00000000
4 sub-partition 2:   start:000000e0 00000000   size:00000000 80000000
5 sub-partition 2:   start:000000e4 00000000   size:00000000 80000000
6 sub-partition 2:   start:000000e8 00000000   size:00000000 80000000
7 sub-partition 2:   start:000000ec 00000000   size:00000000 7c000000
0 community 0:   start:0000006c 78000000   size:00000000 08000000
1 community 0:   start:000000ac 7c000000   size:00000000 04000000
2 community 0:   start:000000ec 7c000000   size:00000000 04000000
total memory, 31.875 GB
probe I/O subsystem
probing hose 0, PCI
probing PCI-to-PCI bridge, hose 0 bus 2
do not use secondary IDE channel on CMD controller
bus 2, slot 0, function 0 -- usba -- USB
bus 2, slot 0, function 1 -- usbb -- USB
bus 2, slot 0, function 2 -- usbc -- USB
bus 2, slot 0, function 3 -- usbd -- USB
bus 2, slot 1 -- dqa -- CMD 649 PCI-IDE
bus 2, slot 2 -- pka -- Adaptec AIC-7892
probing hose 1, PCI
bus 0, slot 1, function 0 -- pkb -- Adaptec AIC-7899
bus 0, slot 1, function 1 -- pkc -- Adaptec AIC-7899
probing hose 2, PCI
probing PCI-to-PCI bridge, hose 2 bus 2
bus 2, slot 4 -- eia -- DE602-B*
bus 2, slot 5 -- eib -- DE602-B*
bus 0, slot 2 -- pga -- FCA-2354
probing hose 3, PCI
probing hose 20, PCI
probing PCI-to-PCI bridge, hose 20 bus 2
do not use secondary IDE channel on CMD controller
bus 2, slot 0, function 0 -- usbe -- USB
bus 2, slot 0, function 1 -- usbf -- USB
bus 2, slot 0, function 2 -- usbg -- USB
bus 2, slot 0, function 3 -- usbh -- USB
bus 2, slot 1 -- dqb -- CMD 649 PCI-IDE
bus 2, slot 2 -- pkd -- Adaptec AIC-7892
probing hose 21, PCI
bus 0, slot 2 -- pgb -- KGPSA-C
probing hose 22, PCI
probing PCI-to-PCI bridge, hose 22 bus 2
bus 2, slot 4 -- eic -- DE602-AA
bus 2, slot 5 -- eid -- DE602-AA
probing hose 23, PCI
probing hose 32, PCI
```

```
probing PCI-to-PCI bridge, hose 32 bus 2
do not use secondary IDE channel on CMD controller
bus 2, slot 0, function 0 -- usbi -- USB
bus 2, slot 0, function 1 -- usbj -- USB
bus 2, slot 0, function 2 -- usbk -- USB
bus 2, slot 0, function 3 -- usbl -- USB
bus 2, slot 1 -- dqc -- CMD 649 PCI-IDE
bus 2, slot 2 -- pke -- Adaptec AIC-7892
probing hose 33, PCI
bus 0, slot 1, function 0 -- pkf -- Adaptec AIC-7899
bus 0, slot 1, function 1 -- pkg -- Adaptec AIC-7899
probing hose 34, PCI
probing PCI-to-PCI bridge, hose 34 bus 2
bus 2, slot 4 -- eie -- DE602-B*
bus 2, slot 5 -- eif -- DE602-B*
bus 0, slot 2 -- pgc -- FCA-2354
probing hose 35, PCI
starting drivers
Starting secondary CPU 1 at address 400030000
Starting secondary CPU 2 at address 800030000
Starting secondary CPU 3 at address c00030000
Starting secondary CPU 4 at address 2000030000
Starting secondary CPU 5 at address 2400030000
Starting secondary CPU 6 at address 2800030000
Starting secondary CPU 7 at address 2c00030000
Starting secondary CPU 8 at address 4000030000
Starting secondary CPU 9 at address 4400030000
Starting secondary CPU 10 at address 4800030000
Starting secondary CPU 11 at address 4c00030000
Starting secondary CPU 12 at address 6000030000
Starting secondary CPU 13 at address 6400030000
Starting secondary CPU 14 at address 6800030000
Starting secondary CPU 15 at address 6c00030000
initializing GCT/FRU.....................................................
..................... at 54c000
Initializing dqa dqb dqc eia eib eic eid eie eif pka pkb pkc pkd pke pkf
 pkg pga pgb  pgc
AlphaServer Console T6.5-14, built on Jun 20 2003 at 14:52:48
P00>>>
```

Set environment variables separately for each partition:

```
P00>>> set ei*mode twi
P00>>> set bootdef_dev dka0
```

Part C: Output from Instance 1:

```
~PCO-I-(pco_01) Powering on partition. HP: hp0
starting console on CPU 16
console physical memory base is 8000000000
initialized idle PCB
initializing semaphores
initializing heap
initial heap 700c0
memory low limit = 54c000 heap = 700c0, 1fffc0
initializing driver structures
initializing idle process PID
initializing file system
initializing timer data structures
lowering IPL
CPU 16 speed is 1150 MHz
create dead_eater
create poll
create timer
create powerup
entering idle loop
access NVRAM
Get Partition DB
hpcount = 1, spcount = 4, ev7_count = 32, io7_count = 5
hard_partition = 0
IO7-100 (Pass 3) at PID 10
IO7 North port speed is 191 MHz
Hose 64 - 33 MHz PCI
Hose 65 - 66 MHz PCI
Hose 66 - 66 MHz PCI
Hose 67 - 4X AGP
0 sub-partition 0:   start:00000000 00000000   size:00000000 80000000
1 sub-partition 0:   start:00000004 00000000   size:00000000 80000000
2 sub-partition 0:   start:00000008 00000000   size:00000000 80000000
3 sub-partition 0:   start:0000000c 00000000   size:00000000 80000000
4 sub-partition 0:   start:00000020 00000000   size:00000000 80000000
5 sub-partition 0:   start:00000024 00000000   size:00000000 80000000
6 sub-partition 0:   start:00000028 00000000   size:00000000 80000000
7 sub-partition 0:   start:0000002c 00000000   size:00000000 80000000
8 sub-partition 0:   start:00000040 00000000   size:00000000 80000000
9 sub-partition 0:   start:00000044 00000000   size:00000000 80000000
10 sub-partition 0:   start:00000048 00000000   size:00000000 80000000
11 sub-partition 0:   start:0000004c 00000000   size:00000000 80000000
12 sub-partition 0:   start:00000060 00000000   size:00000000 80000000
13 sub-partition 0:   start:00000064 00000000   size:00000000 80000000
14 sub-partition 0:   start:00000068 00000000   size:00000000 80000000
15 sub-partition 0:   start:0000006c 00000000   size:00000000 78000000
0 sub-partition 1:   start:00000080 00000000   size:00000000 80000000
PID 16 console memory base: 8000000000, 2 GB
1 sub-partition 1:   start:00000084 00000000   size:00000000 80000000
PID 17 memory: 8400000000, 2 GB
2 sub-partition 1:   start:00000088 00000000   size:00000000 80000000
PID 18 memory: 8800000000, 2 GB
3 sub-partition 1:   start:0000008c 00000000   size:00000000 80000000
PID 19 memory: 8c00000000, 2 GB
4 sub-partition 1:   start:000000a0 00000000   size:00000000 80000000
PID 20 memory: a000000000, 2 GB
5 sub-partition 1:   start:000000a4 00000000   size:00000000 80000000
PID 21 memory: a400000000, 2 GB
6 sub-partition 1:   start:000000a8 00000000   size:00000000 80000000
PID 22 memory: a800000000, 2 GB
7 sub-partition 1:   start:000000ac 00000000   size:00000000 7c000000
PID 23 memory: ac00000000, 1.938 GB
0 sub-partition 2:   start:000000c0 00000000   size:00000000 80000000
1 sub-partition 2:   start:000000c4 00000000   size:00000000 80000000
2 sub-partition 2:   start:000000c8 00000000   size:00000001 00000000
```

```
3 sub-partition 2:   start:000000cc 00000000   size:00000001 00000000
4 sub-partition 2:   start:000000e0 00000000   size:00000000 80000000
5 sub-partition 2:   start:000000e4 00000000   size:00000000 80000000
6 sub-partition 2:   start:000000e8 00000000   size:00000000 80000000
7 sub-partition 2:   start:000000ec 00000000   size:00000000 7c000000
0 community 0:   start:0000006c 78000000   size:00000000 08000000
1 community 0:   start:000000ac 7c000000   size:00000000 04000000
2 community 0:   start:000000ec 7c000000   size:00000000 04000000
total memory, 15.938 GB
waiting for GCT/FRU at 54c000 by CPU 0
probe I/O subsystem
probing hose 64, PCI
probing PCI-to-PCI bridge, hose 64 bus 2
do not use secondary IDE channel on CMD controller
bus 2, slot 0, function 0 -- usba -- USB
bus 2, slot 0, function 1 -- usbb -- USB
bus 2, slot 0, function 2 -- usbc -- USB
bus 2, slot 0, function 3 -- usbd -- USB
bus 2, slot 1 -- dqa -- CMD 649 PCI-IDE
bus 2, slot 2 -- pka -- Adaptec AIC-7892
probing hose 65, PCI
bus 0, slot 1, function 0 -- pkb -- Adaptec AIC-7899
bus 0, slot 1, function 1 -- pkc -- Adaptec AIC-7899
probing hose 66, PCI
probing PCI-to-PCI bridge, hose 66 bus 2
bus 2, slot 4 -- eia -- DE602-B*
bus 2, slot 5 -- eib -- DE602-B*
bus 0, slot 2 -- pga -- FCA-2354
probing hose 67, PCI
starting drivers
Starting secondary CPU 17 at address 8400030000
Starting secondary CPU 18 at address 8800030000
Starting secondary CPU 19 at address 8c00030000
Starting secondary CPU 20 at address a000030000
Starting secondary CPU 21 at address a400030000
Starting secondary CPU 22 at address a800030000
Starting secondary CPU 23 at address ac00030000
initializing GCT/FRU to 54c000
Initializing dqa eia eib pka pkb pkc pga
AlphaServer Console T6.5-14, built on Jun 20 2003 at 14:52:48
P16>>>set ei*mode twi
P16>> set bootdef_dev dka0
```

Part D: Output from Instance 2:

```
~PCO-I-(pco_01) Powering on partition. HP: hp0
starting console on CPU 24
console physical memory base is c000000000
initialized idle PCB
initializing semaphores
initializing heap
initial heap 700c0
memory low limit = 54c000 heap = 700c0, 1fffc0
initializing driver structures
initializing idle process PID
initializing file system
initializing timer data structures
lowering IPL
CPU 24 speed is 1150 MHz
create dead_eater
create poll
create timer
create powerup
entering idle loop
access NVRAM
Get Partition DB
hpcount = 1, spcount = 4, ev7_count = 32, io7_count = 5
hard_partition = 0
IO7-100 (Pass 3) at PID 18
IO7 North port speed is 191 MHz
Hose 96 - 33 MHz PCI
Hose 97 - 66 MHz PCI
Hose 98 - 66 MHz PCI
Hose 99 - 4X AGP
0 sub-partition 0:   start:00000000 00000000   size:00000000 80000000
1 sub-partition 0:   start:00000004 00000000   size:00000000 80000000
2 sub-partition 0:   start:00000008 00000000   size:00000000 80000000
3 sub-partition 0:   start:0000000c 00000000   size:00000000 80000000
4 sub-partition 0:   start:00000020 00000000   size:00000000 80000000
5 sub-partition 0:   start:00000024 00000000   size:00000000 80000000
6 sub-partition 0:   start:00000028 00000000   size:00000000 80000000
7 sub-partition 0:   start:0000002c 00000000   size:00000000 80000000
8 sub-partition 0:   start:00000040 00000000   size:00000000 80000000
9 sub-partition 0:   start:00000044 00000000   size:00000000 80000000
10 sub-partition 0:   start:00000048 00000000   size:00000000 80000000
11 sub-partition 0:   start:0000004c 00000000   size:00000000 80000000
12 sub-partition 0:   start:00000060 00000000   size:00000000 80000000
13 sub-partition 0:   start:00000064 00000000   size:00000000 80000000
14 sub-partition 0:   start:00000068 00000000   size:00000000 80000000
15 sub-partition 0:   start:0000006c 00000000   size:00000000 78000000
0 sub-partition 1:   start:00000080 00000000   size:00000000 80000000
1 sub-partition 1:   start:00000084 00000000   size:00000000 80000000
2 sub-partition 1:   start:00000088 00000000   size:00000000 80000000
3 sub-partition 1:   start:0000008c 00000000   size:00000000 80000000
4 sub-partition 1:   start:000000a0 00000000   size:00000000 80000000
5 sub-partition 1:   start:000000a4 00000000   size:00000000 80000000
6 sub-partition 1:   start:000000a8 00000000   size:00000000 80000000
7 sub-partition 1:   start:000000ac 00000000   size:00000000 7c000000
0 sub-partition 2:   start:000000c0 00000000   size:00000000 80000000
PID 24 console memory base: c000000000, 2 GB
1 sub-partition 2:   start:000000c4 00000000   size:00000000 80000000
PID 25 memory: c400000000, 2 GB
2 sub-partition 2:   start:000000c8 00000000   size:00000001 00000000
PID 26 memory: c800000000, 4 GB
3 sub-partition 2:   start:000000cc 00000000   size:00000001 00000000
PID 27 memory: cc00000000, 4 GB
4 sub-partition 2:   start:000000e0 00000000   size:00000000 80000000
PID 28 memory: e000000000, 2 GB
```

```
5 sub-partition 2:   start:000000e4 00000000   size:00000000 80000000
PID 29 memory: e400000000, 2 GB
6 sub-partition 2:   start:000000e8 00000000   size:00000000 80000000
PID 30 memory: e800000000, 2 GB
7 sub-partition 2:   start:000000ec 00000000   size:00000000 7c000000
PID 31 memory: ec00000000, 1.938 GB
0 community 0:   start:0000006c 78000000   size:00000000 08000000
1 community 0:   start:000000ac 7c000000   size:00000000 04000000
2 community 0:   start:000000ec 7c000000   size:00000000 04000000
total memory, 19.938 GB
waiting for GCT/FRU at 54c000 by CPU 0
waiting for GCT/FRU at 54c000 by CPU 0
waiting for GCT/FRU at 54c000 by CPU 0
waiting for GCT/FRU at 54c000 by CPU 0
waiting for GCT/FRU at 54c000 by CPU 0
waiting for GCT/FRU at 54c000 by CPU 0
waiting for GCT/FRU at 54c000 by CPU 0
waiting for GCT/FRU at 54c000 by CPU 0
waiting for GCT/FRU at 54c000 by CPU 0
waiting for GCT/FRU at 54c000 by CPU 0
waiting for GCT/FRU at 54c000 by CPU 0
waiting for GCT/FRU at 54c000 by CPU 0
waiting for GCT/FRU at 54c000 by CPU 0
waiting for GCT/FRU at 54c000 by CPU 0
waiting for GCT/FRU at 54c000 by CPU 0
waiting for GCT/FRU at 54c000 by CPU 0
waiting for GCT/FRU at 54c000 by CPU 0
waiting for GCT/FRU at 54c000 by CPU 0
waiting for GCT/FRU at 54c000 by CPU 0
probe I/O subsystem
probing hose 96, PCI
probing PCI-to-PCI bridge, hose 96 bus 2
do not use secondary IDE channel on CMD controller
bus 2, slot 0, function 0 -- usba -- USB
bus 2, slot 0, function 1 -- usbb -- USB
bus 2, slot 0, function 2 -- usbc -- USB
bus 2, slot 0, function 3 -- usbd -- USB
bus 2, slot 1 -- dqa -- CMD 649 PCI-IDE
bus 2, slot 2 -- pka -- Adaptec AIC-7892
probing hose 97, PCI
bus 0, slot 1, function 0 -- pkb -- Adaptec AIC-7899
bus 0, slot 1, function 1 -- pkc -- Adaptec AIC-7899
probing hose 98, PCI
probing PCI-to-PCI bridge, hose 98 bus 2
bus 2, slot 4 -- eia -- DE602-B*
bus 2, slot 5 -- eib -- DE602-B*
bus 0, slot 2 -- pga -- FCA-2354
probing hose 99, PCI
starting drivers
Starting secondary CPU 25 at address c400030000
Starting secondary CPU 26 at address c800030000
Starting secondary CPU 27 at address cc00030000
Starting secondary CPU 28 at address e000030000
Starting secondary CPU 29 at address e400030000
Starting secondary CPU 30 at address e800030000
Starting secondary CPU 31 at address ec00030000
initializing GCT/FRU to 54c000
Initializing dqa eia eib pka pkb pkc pga
AlphaServer Console T6.5-14, built on Jun 20 2003 at 14:52:48
P24>>>set ei*mode two
P24>>>set bootdef_dev dka100
```

4.6 Step 6: Boot the OpenVMS Galaxy

Changing the partition setup may affect console environment variable values; please check to ensure they are correct. To ensure that AUTOGEN reboots correctly when it needs to reboot the system after an initial installation and after a system failure or operator-requested reboot, set the BOOTDEF_DEV and BOOT_OSFLAGS variables on each of your Galaxy consoles before booting.

When the OpenVMS system is booted with Galaxy enabled, each soft partition can run a separate instance of the operating system as a Galaxy member.

At a point in the boot process, the instance will attempt to join the Galaxy configuration. If no Galaxy exists, the attempt triggers the creation of a Galaxy, generating a new Galaxy ID and initializing the structures in shared memory that all members in the resulting Galaxy will use. If a Galaxy does exist, the instance joins, connecting to the shared memory constructs and, through them, to the other member instances.

For a Galaxy instance on the ES47/ES80/GS1280:

- There can be only one primary CPU in any 2P building block.
- Either CPU of a 2P building block can be a primary in a Galaxy instance.
- Each Galaxy requires an IO port.
- Up to 8 Galaxy partitions are supported per hard partition.

When you have correctly installed the Galaxy firmware and configured the consoles, you can boot the initial Galaxy environment as follows for each Galaxy instance:

```
P00>>> B -FL 0,1 DKA100 // or whatever your boot device is.

SYSBOOT> SET GALAXY 1

SYSBOOT> CONTINUE
```

Congratulations! You have created an OpenVMS Galaxy.

Index

R

S

T

U