HEWLETT® PACKARD

V2500 SCA
HP-UX System Guide

V2500 SCA
HP-UX System Guide

HEWLETT® PACKARD

Manufacturing Part Number: A5532-96003

E1299

Legal Notices

The information in this document is subject to change without notice.

Hewlett-Packard makes no warranty of any kind with regard to this manual, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Warranty. A copy of the specific warranty terms applicable to your Hewlett-Packard product and replacement parts can be obtained from your local Sales and Service Office.

Restricted Rights Legend. Use, duplication or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 for DOD agencies, and subparagraphs (c) (1) and (c) (2) of the Commercial Computer Software Restricted Rights clause at FAR 52.227-19 for other agencies.

HEWLETT-PACKARD COMPANY
3000 Hanover Street
Palo Alto, California 94304 U.S.A.

Use of this document and any supporting software media (CD-ROMs, flexible disks, and tape cartridges) supplied for this pack is restricted to this product only. Additional copies of the programs may be made for security and back-up purposes only. Resale of the programs in their present form or with alterations, is expressly prohibited.

This software is based in part on the Fourth Berkeley Software Distribution under license from the Regents of the University of California.

Printing History.

December 1999, Edition 1.
Author: Adam E. Schwartz.

The document printing date and part number indicate its current edition. The printing date will change when a new edition is printed. Minor changes may be made at reprint without changing the print date. The document part number will change when extensive changes are made.

Document updates may be issued between editions to correct errors or document product changes. To ensure that you receive the updated or new editions, subscribe to the appropriate product support service. See your HP sales representative for details.

Contents

Figures

Tables

1 Introduction to HP V-Class Servers

New levels of system scaling are available to you when using Hewlett-Packard's V-Class SCA servers. The HP Scalable Computing Architecture (SCA) gives you uncommonly powerful capabilities to adjust your system's configuration or capacity as your needs change.

The HP-UX operating system manages V-Class SCA system resources as a single, unified system image. An HP V2500 SCA server may contain from 8 to 128 processors, and I/O and memory resources may likewise be scaled.

This book describes how to operate, configure, and administer V-Class SCA servers with the HP-UX operating system.

From a user's perspective, a V-Class SCA server is a standard HP-UX environment that has the same commands and features as on other HP systems.

System administration tasks are, in most cases, handled on V-Class SCA systems as they are on other systems. However, you will notice differences in the way you boot and reboot V-Class systems. You will also see some new considerations when configuring the HP-UX kernel, and other system features, for maximum performance.

A little knowledge of the underlying hardware can be useful as you perform various system administration tasks, especially tasks that affect how HP-UX or the hardware is configured.

Related Information

This section lists other sources for information on the topics covered in this chapter.

Other Books

These books provide details on topics addressed in this chapter.

- *Service Support Processor Guide* — This book provides details about the Service Support Processor workstation.
- *Operator Guide: HP 9000 V2500 SCA Server* — This book covers V2500 SCA system components, booting, hardware configuration, and other topics.

Web Sites

Additional information is available at the following sites on the Web.

- `http://docs.hp.com/` — The HP Technical Documentation home page, which provides free online access to many publications.

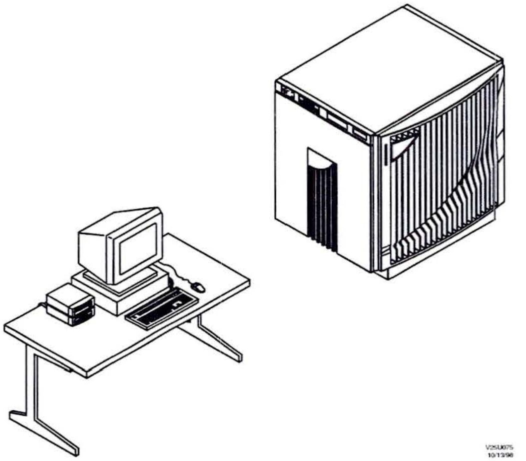
HP V-Class System Overview

Each HP V-Class server has one or more V-Class cabinets and a Service Support Processor, which is connected to the cabinets. HP V-Class cabinets contain all the memory, CPU, and I/O resources provided by a V-Class server.

The Service Support Processor is used for managing the cabinets in a V-Class system. It is an HP workstation that contains software for booting and helping configure and manage a V-Class system's hardware. It also is used to log HP-UX console messages and certain V-Class system status messages.

This book covers HP V-Class system management tasks. Some of these tasks you perform using the Service Support Processor. Other tasks, including most of the tasks that you perform after HP-UX has booted on the server, are managed directly from HP-UX running on the V-Class system's cabinets.

Figure 1 **A Basic V-Class System: The Service Support Processor and V-Class Cabinet**



Each V-Class system may contain up to 128 processors (from 8 to 32 PA-RISC CPUs per cabinet), up to 128 Gbytes of memory (up to 32 Gbytes per cabinet), and a large amount of I/O resources, such as disks, tape devices, network connections, and so on.

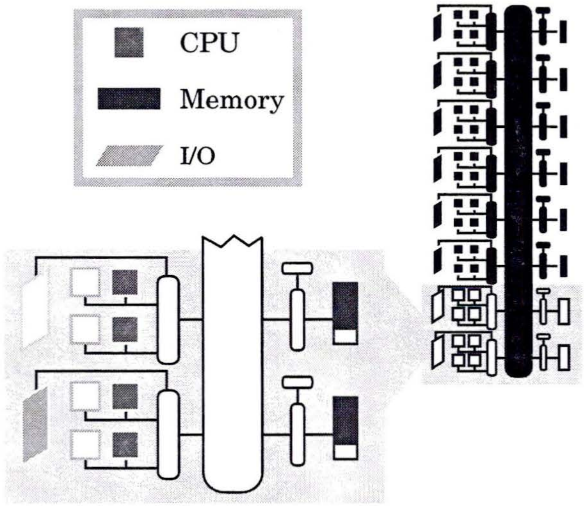
One unique feature of HP V-Class servers is that they are highly scalable machines: you can adjust the amounts of CPU, memory, and I/O as your computing needs change.

All hardware in a V-Class system is tightly interconnected, both within a cabinet and across the cabinets in multiple-cabinet systems. This enables you to scale your system while maintaining a high level of performance and a single, large, globally shared cache-coherent memory.

A single HP-UX system image is provided for each HP V-Class server, regardless of the number of processors, memory, I/O resources, or cabinets, thus simplifying system management.

The basic building blocks of V-Class servers consist of HP PA-RISC processors, memory, and I/O capabilities, as shown below. The interconnecting hardware connects the components to form a single system.

Figure 2 **Building Blocks of a V-Class Cabinet**



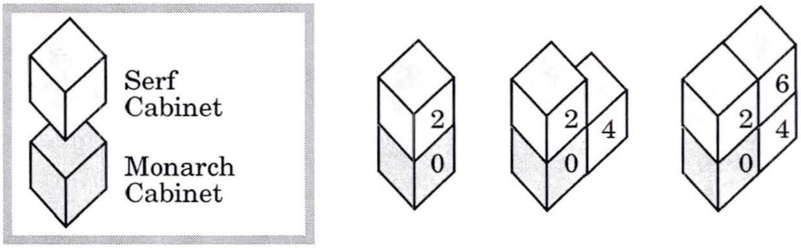
In the portion of the cabinet shown in Figure 2, four of the eight CPU slots shown are occupied, and more memory and I/O hardware can be added as well.

Once a cabinet is filled to capacity, you can add additional cabinets to expand the server's capacity. You also can connect several partially-filled cabinets to form a multiple-cabinet system, following configuration rules.

Supported HP V2500 System Configurations

Figure 3 shows multiple-cabinet server configurations. Each cabinet's hardware system ID number (cabinet ID) is marked on the various cabinets of the four systems shown below.

Figure 3 **Supported Multiple-Cabinet V-Class Server Configurations**

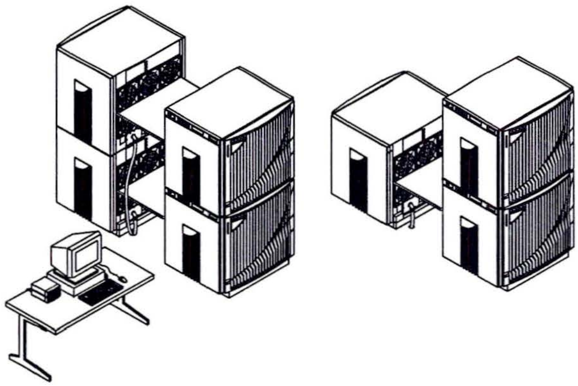


Cabinet ID 0 is always the *monarch cabinet* where the system's boot devices are found. The other cabinets (cabinet IDs 2, 4, and 6) are *serf cabinets* that contribute additional resources to the overall multiple-cabinet server.

All cabinets in a multiple-cabinet V2500 SCA server are managed by the Service Support Processor as a single server *complex*.

Only one Service Support Processor workstation is needed to manage a multiple-cabinet system. A Service Support Processor can also manage several server complexes, as shown in Figure 4, where it supports both a four-cabinet and a three-cabinet V2500 SCA complex.

Figure 4 **Service Support Processor with Two V2500 SCA Server Complexes**



Configuration Rules for V2500 SCA Hardware

Each cabinet in an HP V2500 SCA server complex may be partially or completely configured with processors, memory, and I/O components. Up to four cabinets may be configured in a V2500 SCA complex.

Adding or removing cabinets from a V2500 SCA server complex requires shutting down HP-UX running on the server, and requires reconfiguring hardware cable connections, cabinet firmware settings, and Service Support Processor software settings. Refer to HP service personnel for details on adding and removing cabinet from your V2500 SCA complex.

Supported V2500 SCA server hardware configurations include the following restrictions.

- All eight memory boards must be installed in each cabinet.

  Up to 32 Gbytes of memory may be installed per cabinet, with up to 128 Gbytes possible in a four-cabinet server.

  Each memory board must have the same configuration, and all cabinets must have the same memory configuration. Supported configurations include one-fourth, one-half, and full population of the memory boards. Single-cabinet servers also can be populated to three-fourths capacity. For details see Chapter 8.

- A minimum of eight processors must be installed in each cabinet.

  Up to 32 CPUs may be installed per cabinet, with up to 128 CPUs possible in a four-cabinet server.

  The same number of processors must be present in all cabinets. Processor boards normally have the same number of processors. For example, an eight-processor system may have eight single-CPU processor boards, or four dual-CPU processor boards. Some mixed-capacity processor board configurations are supported.

- Some I/O devices must be on cabinet ID 0, including the boot device, root volume group, primary swap space, and crash dump space.

  Up to 28 PCI cards may be installed per cabinet, with up to 112 PCI cards possible in a four-cabinet server. Each cabinet has up to eight PCI *I/O channels*, with three or four cards per channel.

  See Chapter 7 for additional I/O details.

Listing the Server Hardware Configuration

You can list the server hardware configuration using HP-UX commands and other methods described in later chapters of this book.

A summary of all configured hardware is displayed as part of the server boot-time output. This configuration summary is presented in the cabinet ID 0 console (the complex console). The following console output shows a summary of a two-cabinet server with 16 CPUs, 8192 Mbytes of memory, and 12 PCI I/O channels available.

```
-------------------------------------------------------------------------------
                       Multi-node Configuration Summary
===============================================================================
NODE 0
      UART?  Yes
      CORE   MAC address 0:a0:d9:0:c3:cf   IP# 15.99.111.166 (0x0f636fa6)
      JTAG   MAC address 0:a0:d9:0:c3:a8   IP# 15.99.111.116 (0x0f636f74)
      MEMORY 4096 MB memory installed      1024  MB CTI cache configured
      CPUs   0,2,8,10
             16,18,24,26
      PACs   0,1,2,3,4,5,6,7               MACs   0,1,2,3,4,5,6,7
      TACs   0,1,2,3,4,5,6,7               PCIs   0,1,2,3,4,5,6,7
-------------------------------------------------------------------------------
NODE 2
      UART?  No
      CORE   MAC address 0:a0:d9:0:c6:51   IP# 15.99.111.167 (0x0f636fa7)
      JTAG   MAC address 0:a0:d9:0:c5:ea   IP# 15.99.111.117 (0x0f636f75)
      MEMORY 4096 MB memory installed      1024  MB CTI cache configured
      CPUs   0,2,8,10
             16,18,24,26
      PACs   0,1,2,3,4,5,6,7               MACs   0,1,2,3,4,5,6,7
      TACs   0,1,2,3,4,5,6,7               PCIs   0,1,4,5
-------------------------------------------------------------------------------

8192 MB memory installed        2048  MB CTI cache configured (total, all nodes)
```

The following methods are ways to list a V2500 SCA server's CPU, memory, and I/O resources using HP-UX and other methods.

- **Processor Resources**—As V2500 SCA server hardware boots, the server boot-time output to the cabinet ID 0 console summarizes the processor configuration for all cabinets.

  The `mpsched -s` and `ioscan -Cprocessor` HP-UX commands list all processors currently configured on the server.

  The `CPU` Boot Console Handler (BCH) menu command prints the processor configuration for the cabinet only. Use the `RC` BCH command to list processors on other (non-0) cabinets; for example, `RC 2 CPU` prints cabinet ID 2's processor configuration. See Chapter 2 for details on the BCH menu.

- **Memory Resources**—The V-Class server boot-time output gives details about the physical memory installed on the server's cabinets and the memory's configured use.

  Using the `dmesg` HP-UX command provides one way of checking the installed physical memory available to HP-UX and programs.

```
# /usr/sbin/dmesg | grep -i Physical

    physical page size = 4096 bytes, logical page size = 4096 bytes

    Physical: 62664704 Kbytes, lockable: 47529976 Kbytes,
available: 54536464 Kbytes
```

  See Chapter 8 for details on using SAM, `dmesg`, and other methods for checking an SCA server's memory.

- **I/O Resources**—Both HP-UX commands and commands available from the Boot Console Handler (BCH) menu provide you ways to list I/O components such as devices and controllers.

  Using the `SEA` and `IO` BCH menu commands, you can list SCSI devices connected to a cabinet before HP-UX boots. See Chapter 2 for details on the BCH menu.

  The `ioscan` and `lanscan` HP-UX commands list I/O hardware, and `bdf` lists filesystems. See Chapter 7 for details on I/O.

Other HP Multiprocessor Architectures

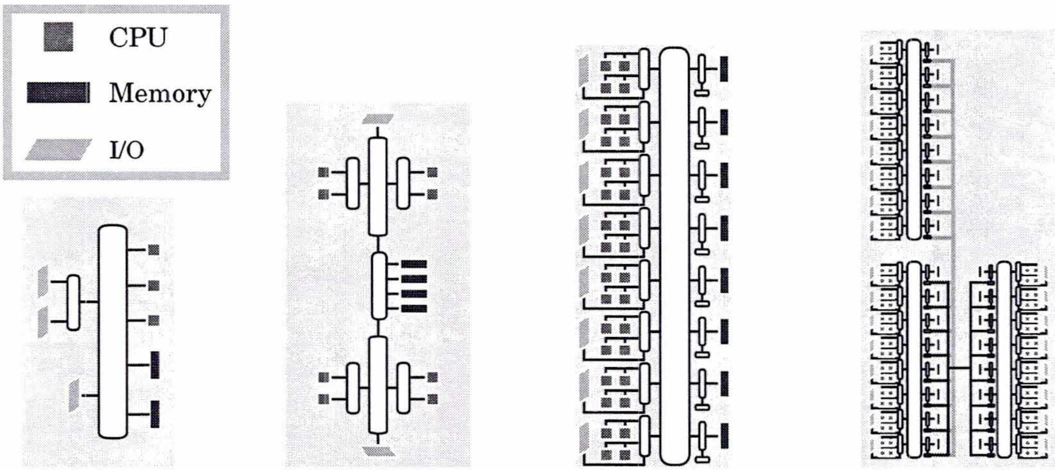
Hewlett Packard offers a wide variety of servers and workstations that run the HP-UX operating system. Despite differences among hardware architectures, HP-UX provides a compatible and consistent environment across the various system models.

It is the HP-UX *system image* as provided through the HP-UX kernel that defines how the hardware and operating system are presented to you and the workloads and utilities you run on the system.

Examples of several differing server architectures are shown in Figure 5. Two are bus-based architecture, which have central bus connections shared by CPU, memory, and input/output resources.

In Figure 5 the bus-based architectures differ from the HP V-Class architecture, which is centrally connected by a crossbar. The crossbar allows the individual CPU, memory, and input/output activities to operate simultaneously and independently, providing greatly increased system throughput and supporting large amounts of resources.

Figure 5 **Comparisons of Multiprocessor Architectures**



A Bus-based Architecture

Dual-bus HP N-Class Architecture

Crossbar-based HP V-Class Architecture

Multiple-crossbar HP V-Class SCA Architecture

Though architecturally different, all the systems in Figure 5 appear similar and are compatible when running HP-UX.

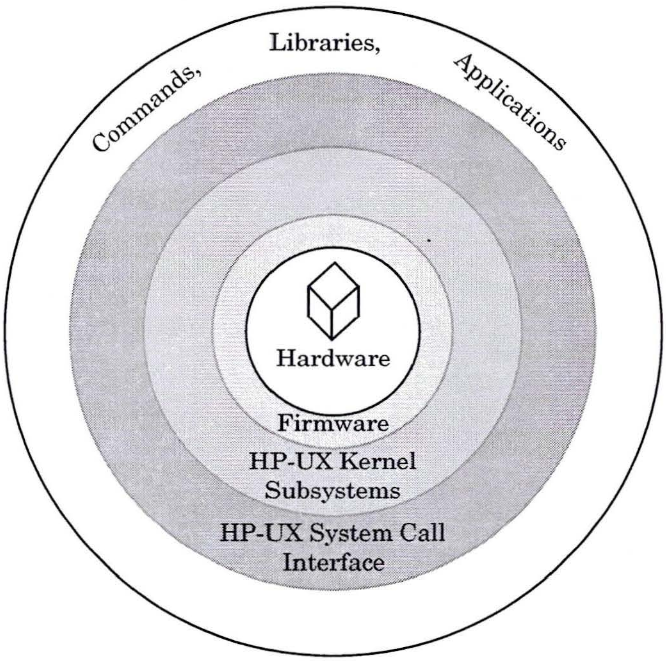
HP-UX Kernel Features for Multiprocessor Systems

The HP-UX operating system has features that can take maximum advantage of a system's resources, regardless of the system model that HP-UX runs on.

The HP-UX kernel gives you access to system resources, such as CPU, memory, and input/output devices. The HP-UX kernel also controls and manages these resources in a consistent way.

Figure 6 shows an overview of the HP-UX operating environment. The firmware helps boot the system and facilitates kernel control of the hardware during run time. HP-UX kernel subsystems communicate with and control the system hardware. HP-UX system calls are the part of the kernel that let you communicate with the kernel subsystems (and, thus, the hardware).

Figure 6 **HP-UX Operating Environment**



When you enter an HP-UX command, or run an application, the command or application makes system calls, which communicate with kernel subsystems. The kernel subsystems access and control the hardware resources needed.

You can configure the HP-UX kernel by adjusting kernel parameters, which specify how the operating system manages resources. Your modifying parameters in effect tunes HP-UX to optimally use the system resources for the sorts of workloads you run.

HP-UX Multiprocessor and SCA Features

The following kernel features are some of the ways HP-UX manages multiprocessor system resources. SCA system issues are also listed.

- The DLKM (Dynamically Loadable Kernel Modules) feature is not supported for the HP-UX 11.10 release.
- Kernel text must be less than 128 Mbytes because of kernel text replication across multiple cabinets on V2500 SCA servers. Normally this is not an issue because kernel text occupies much less than 128 Mbytes.
- The HP-UX kernel is highly parallelized and efficiently scales operating system functions on multiprocessor platforms.
- A single HP-UX kernel controls all hardware and software on a server. Multiple-cabinet HP V-Class servers are likewise managed by a single kernel that provides a single system image and a single set of shared resources.
- The 64-bit version of HP-UX, which V-Class servers run, allows HP-UX to process much larger applications and data sets, and can address a large, shared physical memory.
- Each processor has its own HP-UX data structures (such as the run queue, notion of priority and current process, and so on), which enable the HP-UX scheduler and the load balancer to ensure that processors in a server are efficiently and equitably used.
- Kernel threads provide for kernel-level parallelization of applications.

- Support for *gang scheduling* enables HP-UX to efficiently schedule parallel applications. The gang scheduling feature can significantly improve parallel application performance, especially for shared-memory programs running on heavily-loaded systems.
- Thread and process *launch policies* determine how HP-UX distributes work among an SCA system's locality domains. Each V-Class server cabinet is considered to be a locality domain. HP-UX uses default launch policies unless you specify a thread-launch policy or process-launch policy. See Chapter 5 for details.
- All processes have access to all available memory and I/O in the system.

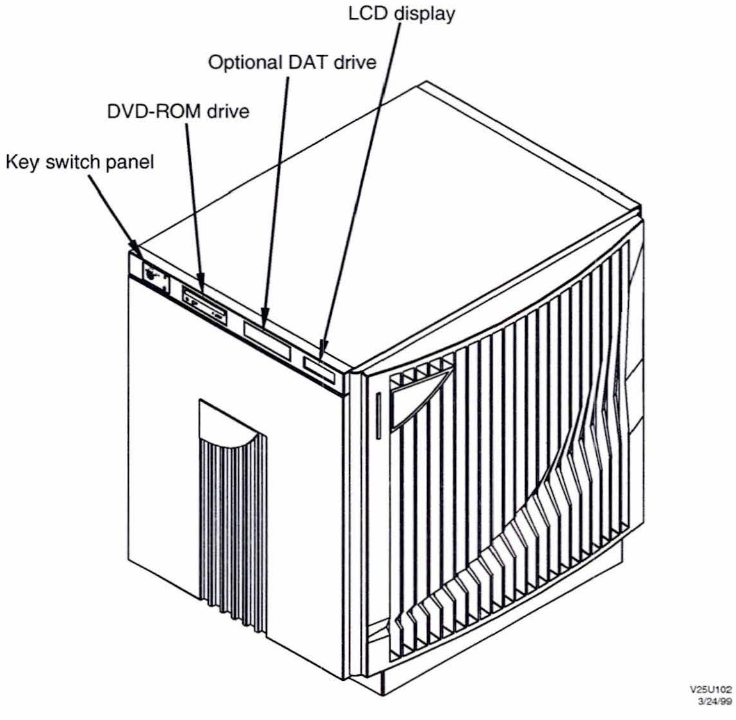
The HP-UX 11.10 Release Notes provides more details about HP-UX 11.10 SCA functionality. See the file /usr/share/doc/11.10RelNotes.

HP V2500 Cabinet Interface

The HP V2500 server cabinet includes an operator control panel on the cabinet's left side. The front of the cabinet includes the power on light bar, and the cabinet's rear provides an escape for air flow.

Figure 7 shows the operator control panel, with its key switch panel (for turning power on and off), LCD status display, DVD-ROM drive (for using CD media), and optional DAT drive.

Figure 7 **V2500 Cabinet Operator Control Panel**

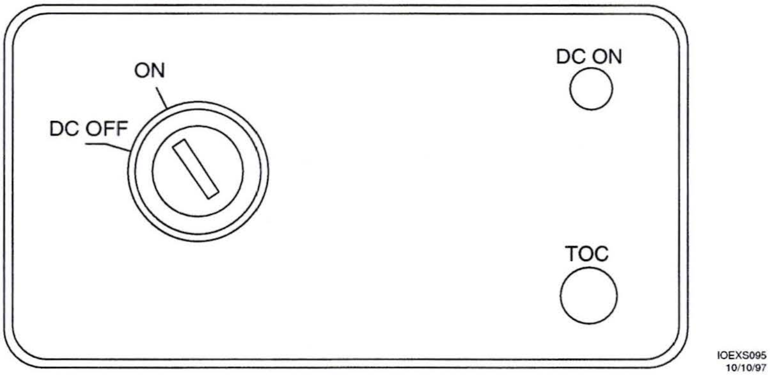


Various parts of the operator panel are described in the following sections.

Key Switch Panel

The key switch panel is located on the operator panel's left side, as shown in Figure 7 on page 27. The key switch panel includes a two-position key switch, a DC ON LED, and a TOC (Transfer Of Control) button, shown below in Figure 8.

Figure 8 **V2500 Cabinet Key Switch Panel**



Key Switch

The key switch has two positions:

- **DC OFF**

  DC power is not applied to the cabinet. Placing the key switch in this position is the normal method for turning off a cabinet's power.

- **ON**

  DC power is applied to the cabinet. When a cabinet is powered on, Power-On Self Test (POST) begins executing on the cabinet, brings up the cabinet hardware from an indeterminate state, and if necessary performs initial multiple-cabinet synchronizations. POST then calls OBP. See Chapter 2 for more details on system booting.

DC ON LED

When it is lit, the LED indicates that DC power has been applied to the cabinet.

TOC Button

The TOC (Transfer Of Control) button is a recessed switch for resetting the system. Pressing the TOC button causes a system-wide level 4 reset, which results in a TOC and a crash dump (if crash dump is enabled, see Chapter 7). Only one TOC should be initiated per system, including on multiple-cabinet systems.

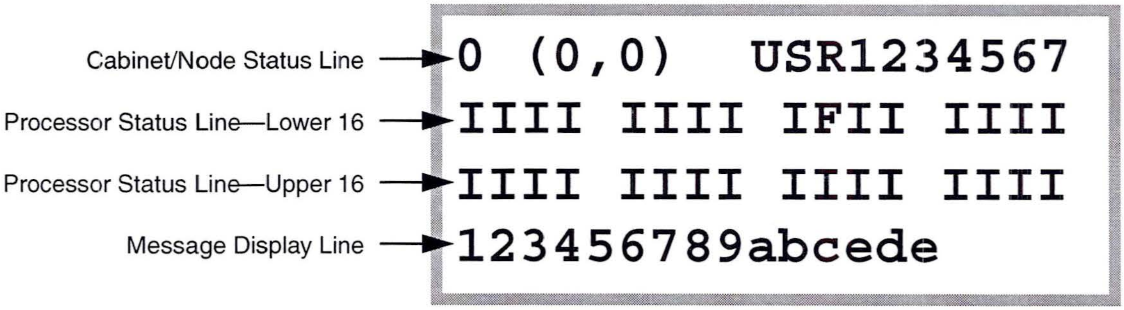
LCD (Liquid Crystal Display)

Each cabinet's Liquid Crystal Display (LCD) display is located on the right of the operator panel, as shown in Figure 7 on page 27. The LCD is a 20-character by 4-line liquid crystal display.

You can print the LCD contents using the `lcd` command on the Service Support Processor. The `lcd` command prints the LCD for all cabinets in a multiple-cabinet server, providing a way to remotely monitor the server.

Figure 9 shows the display and lists what each line on the display indicates.

Figure 9 **V2500 Cabinet Front Panel LCD**



When the operator panel key switch is turned on, the LCD powers up but is initially blank. Power-On Self Test (POST) takes about 20 seconds to start displaying output to the LCD.

The following sections explain the output shown in Figure 9.

Cabinet/Node Status Line

The cabinet status line shows the cabinet ID in decimal form and in CTI topology (X, Y) form, and shows the server's complex serial number. The complex serial number is the same for all cabinets in a multiple-cabinet V2500 SCA server complex. The cabinet ID is in the LCD's upper left corner, and the complex serial number is in the upper right corner.

Processor Status Lines

The processor status lines show the current run state for each CPU in the cabinet. The first line shows the lower 16 processors (those in the "A" slots) and the second line is the upper 16 processors (in the "B" slots).

Following POST (Power-On Self Test), the LCD shows each processor's run-time status using the codes listed in Table 1.

Table 1 **Processor Run-Time Status LCD Codes**

Status	Description
R	RUN: running HP-UX or performing system initialization operations.
F	FORTH: the processor running Forth when OBP is booted.
I	IDLE: processor is in an idle loop, awaiting a command.
M	MONARCH: the primary processor during POST initialization.
H	HPMC: processor detected a High Priority Machine Check (HPMC).
T	TOC: processor has detected a transfer of control (TOC).
S	SOFT_RESET: processor has detected a soft reset.
D	DEAD: processor has failed initialization or self-test.
d	DECONFIG: processor has been deconfigured by POST or the user.
-	EMPTY: empty processor slot.
?	UNKNOWN: processor slot status is unknown.

Message Display Line

The message display line shows the POST initialization progress. This is updated by the cabinet's monarch processor. The system console for each cabinet also shows details during POST. See the *post*(1M) manpage on the Service Support Processor for LCD processor and message display line codes.

Special Features of the HP V-Class Hardware Architecture

Many features distinguish HP V-Class servers from other systems. Some notable features covered in this book include:

- The HP V-Class Service Support Processor.

  The Service Support Processor is a workstation that you use for booting HP V-Class systems and for performing some configuration and diagnostic work.

  For details see Chapter 3, “Service Support Processor Workstation”.

- Multiple-cabinet system configurations.

  You can scale an HP V2500 SCA system to include up to four cabinets. Each cabinet can contain 32 processors, 32 Gbytes of memory, and 28 PCI I/O cards.

- Tight system interconnection.

  Hewlett-Packard’s V2500 SCA interconnecting hardware provides a very tightly interconnected system both within server cabinets and across cabinets in multiple-cabinet V2500 SCA systems.

The rest of this section gives an overview of the interconnecting hardware in HP V-Class systems.

System Interconnections

Interconnecting hardware in V-Class servers joins the various CPUs, the memory, and input/output devices into a single, tightly-interconnected system.

This section provides a simplified high-level view of HP V-Class servers’ interconnecting hardware.

Each cabinet contains the following interconnecting hardware:

- Processor agents—Provide CPU and input/output connections.

  Each processor agent serves as the bus connection for a subset of the system's CPUs, in addition to serving as a connection for PCI input/output controllers.

  Because of the processor agents each CPU receives only the necessary messages from the rest of the system. This helps achieve greater overall system bandwidth.

- Crossbar—Is the central interconnection for all components in a cabinet.

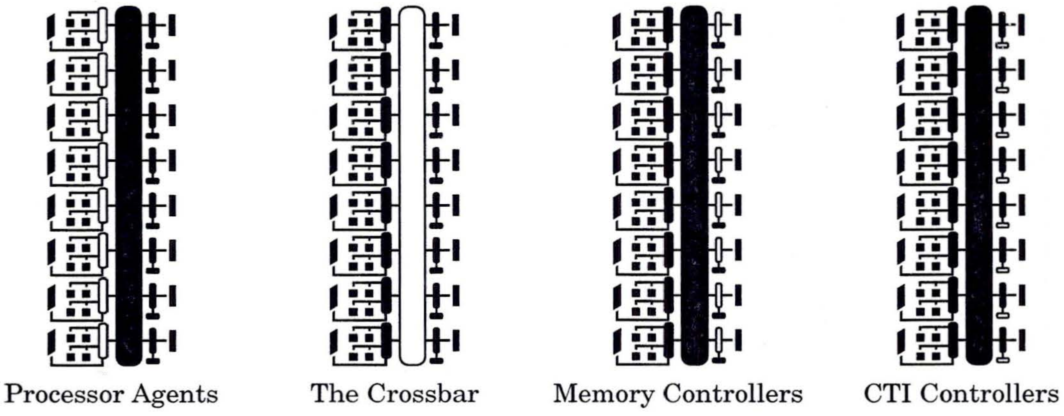
  The crossbar transfers CPU and I/O messages and data to and from memory. See "HP V2500 HyperPlane Crossbar" on page 33 for details.

- Memory controllers—Provide cache-coherent access to memory.

- CTI controllers—Join cabinets together and provide routes to memory on remote cabinets, if any. See "HP V2500 CTI Connections" on page 34 for details.

Figure 10 shows these major interconnecting components.

Figure 10 Interconnecting Hardware in HP V-Class Systems



The following sections, "HP V2500 HyperPlane Crossbar" and "HP V2500 CTI Connections", describe how V-Class servers access memory and shows the roles played by the interconnecting hardware.

HP V2500 HyperPlane Crossbar

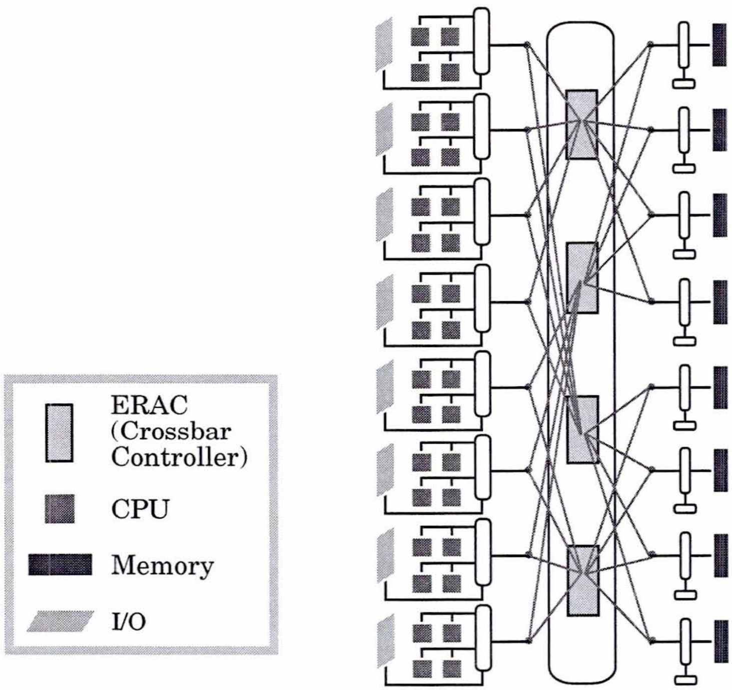
The primary interconnecting component within each HP V2500 cabinet is the HyperPlane Crossbar. The crossbar connects CPUs and I/O to memory within a cabinet.

As Figure 11 shows, the V2500 HyperPlane Crossbar has four Exemplar Routing Access Controllers (ERACs) each of which connects to four processor agents and four memory controllers. Each ERAC has 16 ports, four send and four receive on each side.

The V2500 HyperPlane Crossbar is a non-blocking 8x8 crossbar that simultaneously supports eight send messages and eight receive messages.

All V2500 memory controllers and processor agents connect to two separate ERACs, thus making the entire system's memory addressable by all processors and I/O devices in the system.

Figure 11 **V2500 HyperPlane Crossbar Connections**



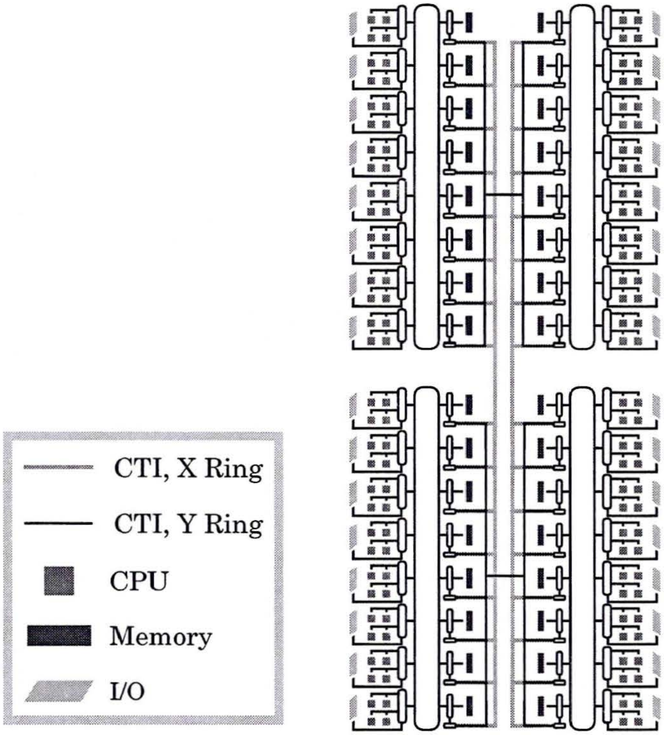
HP V2500 CTI Connections

All cabinets in a multiple-cabinet HP V2500 server configuration are tightly connected using HP's Coherent Toroidal Interconnect (CTI) technology. HP's CTI technology is an extension of the IEEE Scalable Coherent Interface (SCI) standard. CTI connections are used for the following purposes:

- Accessing memory across cabinets.
- Forwarding I/O requests to remote cabinets.
- Signaling interrupts and other communications across cabinets.

CTI cables connect the CTI controllers on different cabinets. An overview of a four-cabinet V2500 server CTI connections is shown in Figure 12.

Figure 12 **Four-Cabinet V2500 Server CTI Cable Connections**



Each CTI controller connects to a CTI controller on a remote cabinet. Two cables connect each pair of controllers, providing send (local-to-remote) and receive (remote-to-local) CTI connections.

Send and receive CTI connections are provided in two dimensions on each controller, for a total of four connections per controller possible.

Two dimensions of CTI connections are possible. Y-dimension cables connect between cabinets 0 and 2, and between cabinets 4 and 6. X-dimension cables connect cabinets 0 and 4, and cabinets 2 and 6.

CTI messages first travel the X dimension, if present and necessary, then travel the Y dimension. As a result, a CTI request may travel different outgoing and return paths.

In a two-cabinet server, cabinets 0 and 2 are connected by Y-dimension CTI cables only. For a three-cabinet server, cabinet 0 has Y-dimension CTI connections to cabinet 2 and X-dimension CTI connections to cabinet 4.

For Y-dimension connections, CTI cables connect to their counterparts on the remote cabinet. For example, the CTI cable connects from Y-send on memory board 0 to the remote cabinet's Y-receive on memory board 0.

For X-dimension connections, CTI cables connect to the opposite controller on the remote cabinet. This means—for X-dimension CTI connections—memory boards connect in the following pairs: 0 and 2, 1 and 3, 4 and 6, and 5 and 7.

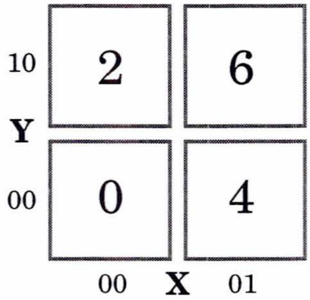
For more details on CTI cable connections refer to qualified HP service personnel.

Numbering of HP V2500 SCA Cabinets

Each cabinet's location in an HP V2500 SCA server complex determines its cabinet ID, as shown in Figure 13. Under this cabinet numbering scheme, HP-UX uses the cabinet ID's middle two bits as the top two bits of hardware paths. See Chapter 7 for details on SCA hardware paths.

Figure 13 **Numbering of V2500 Cabinet Hardware**

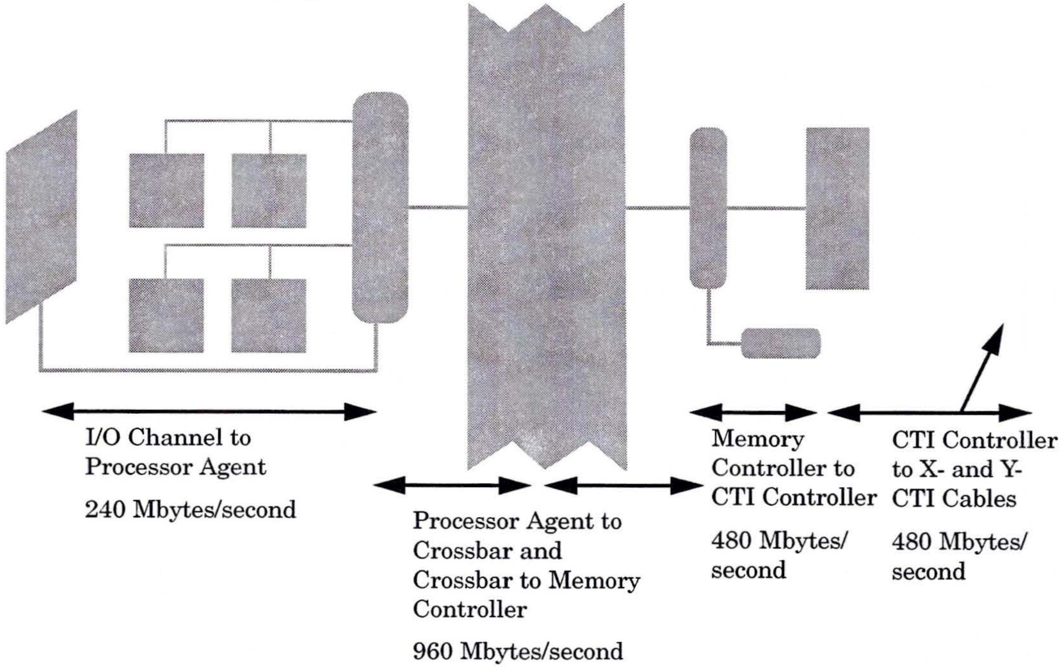
X Y	ID
00 00	0
00 10	2
01 00	4
01 10	6



System Throughput

Figure 14 shows the throughput for major components within an HP V2500 SCA server cabinet.

Figure 14 **Throughput within a V2500 SCA Server**



Hardware configuration decisions you make can depend in part on the bandwidth the programs running on your system require.

- If CPU-to-memory bandwidth is important you may want to limit each HP V2500 cabinet to 16 CPUs (two processors per agent). Fewer CPUs allows for fewer contentions for interconnecting hardware such as processor agents, the crossbar, and memory controllers.
- If I/O bandwidth is important, you may want to place all non-performance storage devices in a single I/O channel, and stripe disk devices across the remaining I/O channels (one mass storage device or a few disk devices per PCI card, and one PCI card for each remaining channel). Creating a filesystem using striped devices provides a high-bandwidth, high-performance filesystem.

- For additional I/O bandwidth, you can stripe devices across I/O channels in the various cabinets in multiple-cabinet V2500 SCA servers. It is possible to create a striped filesystem using devices in multiple different cabinets.

2 Booting and Rebooting

This chapter includes procedures for powering on and booting a V-Class server, and has details on on monitoring and configuring the boot process. Server rebooting and shutdown processes also are covered.

Related Information

This section lists other sources for information on topics covered in this chapter.

HP-UX Manpages

These HP-UX 11.10 manpages provide details on this chapter's topics.

- *shutdown*(1M) — Put the system in single-user mode or halt or reboot the system. The `shutdown` command properly terminates all currently running processes before rebooting or halting the system.
- *reboot*(1M) — Halt or reboot the system. The `shutdown` command is the preferred command for performing these tasks.

Service Support Processor Manpages

These manpages are available on the Service Support Processor.

- *do_reset*(1) — Perform one of four levels of "reset" on the specified V2500 cabinet or cabinets (nodes) within a V2500 server.
- *sppconsole*(1) — Provides a console interface for the specified V2500 cabinet, or by default for cabinet ID 0, the server's monarch cabinet.

Files in the Directory /usr/share/doc

File listed below is available on disk in the directory /usr/share/doc.

- /usr/share/doc/start_up.txt — *HP-UX 10.X Startup and Configuration*.

Web Sites

Additional information and online books are available on the Web.

- `http://docs.hp.com/` — The HP Technical Documentation home page, which provides free online access to many publications.

Overview of HP V2500 System Booting

The procedure for booting an HP V-Class system is similar to that used for other HP Series 800 systems. A few differences exist due to the set of firmware that runs on V-Class servers and the V-Class Scalable Computing Architecture (SCA) design, which enables multiple V2500 cabinets to form a single system.

Both the Service Support Processor workstation and the V-Class cabinets are involved in system startup and booting HP-UX on V-Class servers. The Service Support Processor is the central point of system management during the startup and power down processes. Details on the Service Support Processor are provided in Chapter 3, "Service Support Processor Workstation".

The firmware on V-Class servers includes different components than on other HP-UX servers. However, the main firmware interface, the Boot Console Handler (BCH) menu, provides a familiar, straightforward interface for managing a system before HP-UX boots. The BCH menu is available through the V-Class console interface.

You can safely power off V-Class server cabinets during the firmware boot process, or at any point before HP-UX is booted. After HP-UX begins booting, you must safely shut down HP-UX before powering off.

In multiple-cabinet V2500 SCA systems, each V2500 cabinet runs its own copy of the firmware. Normally during booting you only need to interact with the BCH menu on cabinet ID 0, which serves as the *monarch* or master cabinet. Other cabinets (IDs 2, 4, and 6, if present) run the same set and version of firmware, but are *serf* cabinets under control of the monarch cabinet once OBP starts. Firmware is covered in detail in Chapter 4, "V-Class Firmware Components".

Each HP V2500 cabinet is a *locality domain* that contains processor, memory, and I/O resources that are available throughout the whole system when HP-UX is booted. The hardware, firmware, and HP-UX operating system features collectively provide the SCA functionality that provides a single system image for up to four interconnected V2500 cabinets.

Only I/O devices on cabinet ID 0 may be used as the boot device for a V-Class server. Likewise, only logical volumes that reside entirely on cabinet ID 0 may be used as the boot device.

Powering On and Starting Up V2500 SCA Servers

Starting up HP-UX on V-Class servers involves the following steps. Various procedures for booting and rebooting V-Class servers are covered in detail in "Procedures" on page 54.

1 Powering on and booting the Service Support Processor workstation.

2 Powering on the V-Class server cabinets.

   On multiple-cabinet V2500 SCA systems, the serf cabinets (cabinet IDs 2, 4, and 6, if present) and the monarch cabinet (cabinet ID 0) must all be powered on within two minutes of each other. This ensures that all cabinets synchronize following Power-On Self Test.

3 Searching for and selecting a device on V-Class cabinet ID 0 from which to boot HP-UX, as needed.

   The Boot Console Handler (BCH) menu's SEARCH command searches for bootable devices connected on the cabinet. Only devices on cabinet ID 0 may be booted.

   You can designate an I/O device to be the primary (`PRI`) or alternate (`ALT`) boot device by using the BCH menu's PATH command.

4 Issuing the BCH menu's BOOT command to boot HP-UX on the V-Class server.

   You can set the server to automatically boot HP-UX if you have also set a primary boot device (`PRI`). The BCH menu provides the AUTO BOOT option, which causes the server to automatically boot HP-UX from the primary boot device when AUTO BOOT is set to `ON`.

For details on ways to adjust how your server behaves during the startup process see the following section, "Customizing Server Boot Behavior" on page 46.

Using the BCH Menu and BOOT Command

The Boot Console Handler (BCH) menu is the main interface for selecting boot devices and options before HP-UX boots on a V-Class server. The BCH menu includes several options for searching for boot devices, booting, and customizing the server's boot-time behavior.

For more details on the BCH menu and Forth commands see Chapter 3.

BCH Menu Overview

The following output shows the Boot Console Handler (BCH) menu. The BCH menu is available from the V-Class console for cabinet ID 0.

```
Command                         Description
-------                         -----------
AUto [BOot|SEArch|Force ON|OFF] Display or set the specified flag
BOot [PRI|ALT|<path> <args>]    Boot from a specified path
BootTimer [time]                Display or set boot delay time
CLEARPIM                        Clear PIM storage
CPUconfig [<cpu>] [ON|OFF|SHOW] (De)Configure/Show Processor
DEfault                         Set the system to defined values
DIsplay                         Display this menu
ForthMode                       Switch to the Forth OBP interface
IO                              List the I/O devices in the system
LS [<path>|flash]               List the boot or flash volume
PASSword                        Set the Forth password
PAth [PRI|ALT|CON] [<path>]     Display or modify a path
PDT [CLEAR|DEBUG]               Display/clear Non-Volatile PDT state
PIM_info [cpu#] [HPMC|TOC|LPMC] Display PIM of current or any CPU
RemoteCommand node# command     Execute command on a remote node
RESET [hard|debug]              Force a reset of the system
RESTrict [ON|OFF]               Display/Select restricted access to Forth
SCSI [INIT|RATE] [bus slot val] List/Set SCSI controller parms
SEArch [<path>]                 Search for boot devices
SECure [ON|OFF]                 Display or set secure boot mode
TIme [cn:yr:mo:dy:hr:mn[:ss]]   Display or set the real-time clock
VErsion                         Display the firmware versions
[0] Command:
```

The brackets preceding the "Command" prompt indicate the cabinet ID on which the BCH menu is running (cabinet 0 in this case). When booted on a single-cabinet server, BCH does not display the cabinet information.

BOOT Command Overview

For booting HP-UX on a V-Class server the BCH menu provides the BOOT command.

Syntax

The BOOT command's syntax is as follows.

- `BOot [PRI|ALT|` *path arguments* `]`

  Where the command can be specified as `BOOT` or abbreviated as `BO`.

  The BOOT command is optionally followed by the hardware path of the device to boot. If no path is specified, the `PRI` path is assumed.

  You can specify the entire hardware path (for instance `4/2/0.2.0`), or if the `PRI` or `ALT` variables are defined you can specify them. Note that, when specifying `PRI` or `ALT`, you specify both the *path* and any defined *arguments* associated with the variable.

  Permitted *arguments* include all `hpux` bootstrap arguments. These arguments are normally not used. If specified they are passed to `hpux`, the Secondary System Loader (SSL) that loads HP-UX. See *hpux*(1M) for details.

  For example, the `ls` argument lists the contents of the /stand directory for a device, allowing you to view any alternate vmunix kernel files present. The `ls` argument does boot HP-UX.

  For *arguments* you also can specify an alternate kernel file, causing the specified kernel to be booted rather than the default, /stand/vmunix.

Examples

- `BOOT`

  This command boots the device located at the primary (`PRI`) hardware path.

- `BOOT PRI`

  This boots the device at the hardware path `PRI`.

- `BOOT PRI vmunix.prev`

  Boots the /stand/vmunix.prev kernel on the `PRI` boot device.

- `BOOT` *path*

  Boots the device at the hardware path specified by *path*.

- `BOOT` *path* `ls`

  Lists—but does not boot—the /stand directory on the device *path*.

Customizing Server Boot Behavior

The V-Class server's Boot Console Handler (BCH) menu provides options for customizing how the server behaves during the boot process. For example, you can set the server to automatically proceed to boot HP-UX by setting the AUTO BOOT option to ON and setting the PRI boot path variable.

Boot variable settings are stored in non-volatile memory (NVRAM) that resides on the utilities board of each V-Class server cabinet. The boot variable settings are saved permanently until you modify them using the BCH menu.

Once a boot variable is modified, its new setting is used the next time the server boots.

On multiple-cabinet V-Class servers the boot variable settings for the monarch cabinet (cabinet ID 0) determine the server's boot-time behavior.

Table 2 lists the main variables that affect the server booting process.

Table 2 **Boot Variables for HP V-Class Servers**

Variable	Description of Boot Options
`AUTO BOOT`	[`ON`|`OFF`] If set to `ON`, the server automatically boots HP-UX from the primary (`PRI`) device during system startup or reset. When set to `OFF` the server boots to the BCH menu interface.
`AUTO SEARCH`	[`ON`|`OFF`] If set to `ON`, the server searches for and lists all bootable I/O devices.
`AUTO FORCE`	[`ON`|`OFF`] If set to `ON`, then OBP allows HP-UX to boot even if one or more cabinets does not complete Power-On Self Test (POST). When set to `OFF`, all cabinets must successfully pass POST for OBP to permit the server to boot HP-UX.
`BOOTTIMER`	[*seconds*] Sets the number of seconds for the system to wait before booting. This is used to allow external mass storage devices to come online.

Variable	Description of Boot Options
`OS`	Must always be set to `HP-UX` for HP V-Class servers.
`PATH`	`[PRI|ALT|CON]` [*path*] Sets the path for the primary boot device (`PRI`), alternate boot device (`ALT`), or console (`CON`).
`SECURE`	`[ON|OFF]` Sets (`ON`) or disables (`OFF`) secure boot mode. If set, the boot process cannot be interrupted. Only useful if `AUTO BOOT` is `ON`.

Overview of Setting Boot Path Variables

Using the BCH menu's PATH command, you can display or set the boot path variables. This lets you set the default (`PRI`) boot device. The `PRI` and `ALT` variables provide a quick way to list the hardware paths of bootable I/O devices.

If `PRI` is set, its device path is used as the boot device for AUTO BOOT.

Syntax

The PATH command's syntax is as follows.

- `PAth [PRI|ALT|CON]` [*path*] [*arguments*]

  Where the command can be specified as `PATH` or abbreviated as `PA`.

  The path (*path*) specified must refer to a device on cabinet ID 0 only.

  The `PRI` path variable is used for the primary or default boot device, and `ALT` is an alternate boot device. `CON` is the path for the console and usually is not modified (normally it is set to `15/1`).

  If no path is specified, either the specified variable is printed or all variable settings are printed. When both a variable and *path* are specified, the variable is set to the specified *path*.

  Boot arguments (*arguments*) also can optionally be specified. These arguments are passed to the `hpux` bootstrap program. Normally arguments are not required.

  For example, the argument `-is` causes HP-UX to boot in single-user mode. See *hpux*(1M) for more details.

Examples

- `PATH`

  This prints the hardware paths settings for all path variables.

- `PATH PRI 4/2/0.2.0`

  This sets the primary (`PRI`) boot path variable to `4/2/0.2.0`.

- `PATH ALT`

  Prints the setting for the alternate (`ALT`) boot path variable.

- `PATH ALT -is`

  Sets the boot arguments for the alternate boot path variable to be `-is`, causing HP-UX to boot in single-user mode. Normally boot arguments are not set, and the system boots at run level three (`-i3`).

Monitoring System Booting

Windows on the Service Support Processor workstation provide views of booting and other activity on the V-Class server. Using these windows, you can check the hardware status of V-Class cabinets and can monitor server startup and shutdown activities.

See Chapter 3, "Service Support Processor Workstation" for more details on the Service Support Processor environment.

The `/spp/scripts/sppconsole` script gives access to the V-Class server's console and the Boot Console Handler (BCH) menu when it is available during startup. For details see below or see the *sppconsole*(1) manpage, available on the Service Support Processor.

The following windows are the main Service Support Processor windows.

- V-Class message window—Reports V-Class cabinet hardware status every 60 seconds. This status comes from the ccmd daemon that runs on the Service Support Processor.

  This window also serves as the HP-UX console (/dev/console) for the Service Support Processor.

- V-Class console window—Reports information as the V-Class server is initialized and booted. This is the window that provides BCH menu access.

  The V-Class console window serves as the HP-UX console (/dev/console) for the server's cabinet ID 0, the monarch cabinet.

  In multiple-cabinet V-Class servers, a unique console is provided for each cabinet until the cabinets synchronize, at which point cabinet ID 0 is responsible for the entire system's console output.

- Service Support Processor login windows—Provide HP-UX login shell access for the Service Support Processor. From these windows you can run commands and scripts on the Service Support Processor for booting and configuring the V-Class server.

  Normally you will log in and access the Service Support Processor using the `sppuser` account.

You can access the various Service Support Processor windows either when directly using the Service Support Processor workstation or when remotely logged in.

During remote log in access such as `rlogin` and `telnet` provide, you initially access the Service Support Processor through a standard login shell.

Obtaining V-Class Console Access

To access the V-Class console after logging in to the Service Support Processor, run the `/spp/scripts/sppconsole` script.

You normally will just need to access the cabinet ID 0 console, the primary system console. This is the default and is provided by specifying `0` or nothing when invoking `sppconsole`.

The `sppconsole` script allows you to optionally specify a cabinet (or "node") number to indicate which V2500 cabinet the console should correspond to. For example, `sppconsole 2` gives you console access for cabinet ID 2 in your current terminal window.

For access to all cabinets' consoles, the `/spp/bin/consolebar` program brings up a new window with buttons listing all cabinets in the server. Each of the buttons initiates a new `xterm` window running `sppconsole` for the specified cabinet.

For details see the *sppconsole*(1) manpage or the *consolebar*(1) manpage, which are on the Service Support Processor.

Using V-Class Console Commands and Access Modes

Two access modes are provided the V-Class consoles: read-only ("spy") mode, and interactive ("control") mode. Many users can simultaneously have read-only access to a V-Class console, but only one person can have interactive control.

Table 3 lists the V-Class console commands, which let you change the console access mode, exit the `sppconsole` program, and do other things.

Table 3 **V-Class Console Window Commands**

V-Class Console Key Sequence	Description
`^Ecf`	Force (take) control of the console.
`^Ecs`	Spy mode: release control of the console.
`^Ecw`	List who is connected to the console.
`^Ec?`	Help: list the console key sequences.
`^Ec.`	Exit the console program.

For example, typing **Control-e c f** gives you read and write control of the console, and typing **Control-e c .** (period) exits the console. See the *sppconsole*(1) manpage on the Service Support Processor for more details and options.

Rebooting, Shutting Down, and Resetting HP V2500 Servers

This section gives an overview of V-Class system rebooting and related issues. Both V-Class HP-UX and V-Class firmware rebooting and reset issues are addressed here.

See also "Procedures" on page 54 for additional V-Class server resetting, rebooting, and shut down procedures.

Rebooting or Shutting Down HP-UX on HP V2500 Servers

You can reboot or shut down HP-UX running on a V-Class server by using the `/sbin/reboot` and `/sbin/shutdown` commands on the server. Using the `shutdown` command is recommended, because it properly terminates all currently running processes. For details see the *reboot*(1M) or *shutdown*(1M) manpage.

Resetting HP V-Class Server Hardware

The `/spp/bin/do_reset` command resets the V-Class hardware. The `do_reset` command is run from the Service Support Processor, causes OBP to reboot, and halts all activity on the V-Class server cabinets involved. For details see the *do_reset*(1) manpage on the Service Support Processor or the section "do_reset Command Overview" that follows.

NOTE — Before running `do_reset` from the Service Support Processor you should shut down HP-UX running on the V-Class server, if possible, to avoid losing data.

Four levels of system reset are provided by `do_reset`, from level 1 (the default) to level 4 (Transfer of Control).

On multiple-cabinet V-Class server configurations, you can reset all cabinets or just selected cabinets with `do_reset`. Larger systems with additional cabinets take more time to reset.

do_reset Command Overview

For performing a hardware reset of a V-Class server, the `/spp/bin/do_reset` command is provided on the Service Support Processor. More details are in the *do_reset*(1) manpage.

Syntax

The `do_reset` command's syntax is as follows.

- `do_reset [`*node_id*` | all] [`*level*`] [`*boot_option*`]`

  If `do_reset` is specified with no arguments then the default level 1 reset of all cabinets is performed, rebooting the server to the Boot Console Handler (BCH) menu,

  Either a cabinet ID (*node_id*) or the keyword `all` can be specified to indicate which cabinets are to be reset. If a cabinet ID is specified, the reset level also must be specified.

  Reset levels (*level*) of 1 through 4 are supported and are specified as numbers. Level 1, the default, provides a hard reset and reboots to the BCH menu. A level 4 reset provides a Transfer of Control (TOC), equivalent to pressing the TOC button on a V-Class cabinet.

  A TOC (`do_reset 0 4`) should specify both the cabinet ID and level. See Chapter 4 for crash dump details.

NOTE: Only one request for a level 4 reset (TOC) should be issued at a time. This ensures that the server properly completes the reset process.

  Various *boot_options* are supported for level 1 resets. Normally only trained HP service personnel use these. See *do_reset*(1).

Examples

- `do_reset`

  This performs a level 1 reset of all cabinets, resetting the server and rebooing to the BCH menu.

- `do_reset 0 4`

  Performs a level 4 reset of the server complex. This causes a Transfer of Control (TOC) that initiates a crash dump of the system, if crash dump is configured. The firmware on cabinet ID 0 resets all cabinets in the server, and reboots the server complex. See *savecrash*(1M) for crash dump details.

Procedures

This section has procedures for powering on, booting, rebooting, and shutting down V-Class servers.

Powering On HP V2500 Servers and the Service Support Processor

When powering on a V-Class system, power on the Service Support Processor and log in first, then power on the server's cabinets. All cabinets in a multiple-cabinet server must be powered on within two minutes of each other to properly synchronize.

Step One — Power on the Service Support Processor.

Step Two — Log in to the Service Support Processor using the `sppuser` account.

Step Three — Power on the V-Class server's serf cabinets (cabinet IDs 2, 4, and 6).

Step Four — Power on the V-Class server's monarch cabinet (cabinet ID 0).

You can safely power off any V-Class cabinet when it is running firmware and has not booted HP-UX. On multiple-cabinet servers this may cause cabinet synchronization problems, which are resolved by resetting all cabinets.

Booting HP-UX on HP V2500 Servers

The procedure for booting HP-UX on a V-Class server is as follows.

Step One — If needed, power on the Service Support Processor and the V-Class server cabinets.

For details see the procedure "Powering On HP V2500 Servers and the Service Support Processor".

Step Two — Access the BCH menu "Command" prompt.

Step Three — Select a device on cabinet ID 0 to boot.

Step Four	Select an HP-UX kernel on the device. See "Finding Bootable Devices and HP-UX Kernels" on page 57 for details.
Step Five	Issue the BOOT command from the BCH menu.

Rebooting HP-UX on HP V2500 Servers

Rebooting from HP-UX shuts down the operating system and reboots the hardware, before proceeding to re-boot the HP-UX operating system.

On V2500 SCA servers, this causes the Power-On Self Test (POST) and OBP boot processes to proceed as if from server start-up.

If, for example, AUTO BOOT is set and PRI is set to a bad (unbootable) device, the server reboot will not proceed past the BCH menu. This is because a bad device (`PRI`) is selected during the hardware reboot process, even though a valid device was previously selected when HP-UX booted.

The basic server reboot process is as follows.

Step One	If the server is running HP-UX, log in to the server. If HP-UX is not running on the server, refer to the procedure "Booting HP-UX on HP V2500 Servers".
Step Two	Check activity on the server and warn users of the impending server reboot. If HP-UX is hung, to reboot HP-UX you may need to reset the server. See the procedure "Resetting HP V2500 Server Hardware".
Step Three	Change to the root (/) directory.
Step Four	Issue the command to reboot HP-UX running on the server. You can use the *reboot*(1M) or *shutdown*(1M) command. If using `shutdown`, use the `-r` option to shut down the system and reboot automatically. The recommended method is to use the `shutdown` command, which properly terminates all currently running processes.

Shutting Down HP-UX on HP V2500 Servers

If HP-UX is not running on a V-Class server, you can safely power it off.

The following procedure is for shutting down and halting HP-UX on a V-Class server.

Step One — Log in to the V-Class server.

Step Two — Check activity on the server and warn users of the impending server reboot.

Step Three — Change to the root (/) directory.

Step Four — Issue the command to shut down and halt HP-UX.

You can use the *shutdown*(1M). Specify the `-h` option, which the `shutdown` command support for halting HP-UX after shutting down.

Upon successful server shutdown, you should see the following message in the V-Class "complex console" for cabinet ID 0.

```
System has halted

OK to turn off power or reset system

UNLESS "WAIT for UPS to turn off power" message was
printed above
```

Resetting HP V2500 Server Hardware

The following procedure is for resetting HP V2500 server hardware. If the server is hung, skip Step One.

Step One — Shut down HP-UX on the V2500 server.

This involves logging in to the server and issuing the shutdown -h command. For details see the procedure "Shutting Down HP-UX on HP V2500 Servers".

If the V-Class server is hung and you can not log in and shut down HP-UX, you can proceed with Step Two and may want to perform a level 4 reset at Step Three.

Step Two — Access a Service Support Processor login shell.

You can do this directly at the Service Support Processor workstation, or by remotely logging in with a `telnet` or `rlogin` session.

Step Three — Issue the `do_reset` command from a Service Support Processor login shell.

By default, `do_reset` performs a level 1 reset of all cabinets. See the *do_reset*(1) manpage on the Service Support Processor for details.

A level 4 reset (`do_reset 0 4`) performs a Transfer of Control (TOC) that resets the server and initiates the HP-UX crash dump process for it, if configured.

Pressing the TOC button on a V-Class server cabinet has the same effect as a level 4 `do_reset`.

Issue only one level 4 reset to ensure a proper TOC.

Normally a level 1 reset (the default for `do_reset`) is performed.

Finding Bootable Devices and HP-UX Kernels

Step One — If needed, power on the Service Support Processor and V-Class server cabinets.

For details see the procedure "Powering On HP V2500 Servers and the Service Support Processor".

Step Two — Access the BCH menu "Command" prompt.

Step Three — Issue the SEARCH command to search for bootable devices.

Note that the `PRI` and `ALT` boot path variables may have been set, indicating preferred devices for booting. You can print their settings with the PATH command.

Step Four — Issue the BOOT command with the `ls` argument to list HP-UX kernel and other files in the specified device's /stand directory.

The format for this command is: `BOOT` *path* `ls`, where the path (*path*) may be `PRI`, `ALT`, or a device path listed by the SEARCH command.

The default kernel for a device is /stand/vmunix. Other kernel files in /stand typically have variations of the "vmunix" name, such as vmunix.prev or vmunix.old.

Setting Boot Variables

Step One — If needed, power on the Service Support Processor and V-Class server cabinets.

For details see the procedure "Powering On HP V2500 Servers and the Service Support Processor".

Step Two — Access the BCH menu "Command" prompt.

Step Three — Print the current variable setting by entering the BCH command with no arguments.

The following BCH commands support customizable boot options. For details see Table 2 on page 46.

AUTO — Supports several options for automating boot behavior, including BOOT, SEARCH, and FORCE.

BOOTTIMER — The time for the system to wait before booting, allow mass storage devices to come online.

PATH — Allows setting the primary (`PRI`) and alternate (`ALT`) boot path variables.

SECURE — Sets or disables secure mode for auto boot.

Step Four — Set the variable to the desired value by entering the command, its arguments (if applicable), and value.

The AUTO and PATH commands accept arguments and values, BOOTTIMER and SECURE accept values only.

Examples of using the BCH menu to print and set boot PATH variables are provided in "Overview of Setting Boot Path Variables" on page 47.

Example Boot-Time Output

The following output is sample boot output from a V2500 SCA server's cabinet ID 0 console. This shows a two-cabinet V-Class server's boot process, from the initial Power-On Self Test (POST) to the Boot Console Handler (BCH) menu prompt.

The text below is marked with numbers, which correspond to the descriptions that follow the sample output.

```
                          (1)
POST Hard Boot on [0:PB5R_A]

HP9000/V2500 POST Revision 2.0.0.3, compiled 1999/07/01 12:21:15 (#0046)

Probing CPUs: PB0L_A PB1R_A PB4L_A PB5R_A PB0L_B PB1R_B PB4L_B PB5R_B

Completing core logic SRAM initialization.
                          (2)
Starting main memory initialization.
  Probing memory: MB0L MB1L MB2R MB3R MB4L MB5L MB6R MB7R
  Installed memory: 4096 MBs, available memory: 4096 MBs.
  Initializing main memory.
  Parallel memory initialization in progress.
                   r0          r1          r2          r3
  PB5R_A MB0L [:::: ____] [:::: ____] [____ ____] [____ ____]
  PB1R_A MB1L [:::: ____] [:::: ____] [____ ____] [____ ____]
  PB4L_A MB2R [:::: ____] [:::: ____] [____ ____] [____ ____]
  PB0L_A MB3R [:::: ____] [:::: ____] [____ ____] [____ ____]
  PB0L_B MB4L [:::: ____] [:::: ____] [____ ____] [____ ____]
  PB1R_B MB5L [:::: ____] [:::: ____] [____ ____] [____ ____]
  PB4L_B MB6R [:::: ____] [:::: ____] [____ ____] [____ ____]
  PB5R_B MB7R [:::: ____] [:::: ____] [____ ____] [____ ____]
 Building main memory map.
Main memory initialization complete.
                          (3)
Starting multinode initialization.
 Collecting memory configuration from nodes: 0,2
 Initializing ERI rings for node 0,2
 Synchronizing nodes: 0,2
 Initializing CTI cache.
  Parallel CTI cache initialization in progress.
                  r0          r1          r2          r3
  PB5R_A MB0L [LLLL ____] [MMMM ____] [____ ____] [____ ____]
  PB1R_A MB1L [LLLL ____] [MMMM ____] [____ ____] [____ ____]
  PB4L_A MB2R [LLLL ____] [MMMM ____] [____ ____] [____ ____]
  PB0L_A MB3R [LLLL ____] [MMMM ____] [____ ____] [____ ____]
  PB0L_B MB4L [LLLL ____] [MMMM ____] [____ ____] [____ ____]
  PB1R_B MB5L [LLLL ____] [MMMM ____] [____ ____] [____ ____]
  PB4L_B MB6R [LLLL ____] [MMMM ____] [____ ____] [____ ____]
  PB5R_B MB7R [LLLL ____] [MMMM ____] [____ ____] [____ ____]
```

```
 Synchronizing nodes: 0,2
 Verifying remote memory access.
 Enabling Time of Century synchronization routing.
 Synchronizing nodes: 0,2
Multinode initialization complete.

Booting OBP          (4)

OBP Power-On Boot on [0:2]

Node 0 OBP attempting to synchronize with node 2.
                       (5)
Node(s) { 2 } now synchronized.

-------------------------------------------------------------------------------
                       PDC Firmware Version Information
                   (6)      PDC_ENTRY version 4.2.0.6
                             POST Revision: 2.0.0.3
          OBP Fieldtest Release 4.2.0.6, compiled 99/07/03 13:49:18  (3)
                   SPP_PDC Release 2.0.0 (04/12/99 18:55:48)

-------------------------------------------------------------------------------
                        Multi-node Configuration Summary
===============================================================================
NODE 0
      UART?  Yes
      CORE   MAC address 0:a0:d9:0:c3:cf    IP# 15.99.111.166 (0x0f636fa6)
      JTAG   MAC address 0:a0:d9:0:c3:a8    IP# 15.99.111.116 (0x0f636f74)
      MEMORY 4096 MB memory installed       128  MB CTI cache configured
      CPUs   0,2,8,10
             16,18,24,26
      PACs   0,1,2,3,4,5,6,7                MACs   0,1,2,3,4,5,6,7
      TACs   0,1,2,3,4,5,6,7                PCIs   0,1,2,3,4,5,6,7
-------------------------------------------------------------------------------
NODE 2
      UART?  No
      CORE   MAC address 0:a0:d9:0:c6:51    IP# 15.99.111.167 (0x0f636fa7)
      JTAG   MAC address 0:a0:d9:0:c5:ea    IP# 15.99.111.117 (0x0f636f75)
      MEMORY 4096 MB memory installed       128  MB CTI cache configured
      CPUs   0,2,8,10
             16,18,24,26
      PACs   0,1,2,3,4,5,6,7                MACs   0,1,2,3,4,5,6,7
      TACs   0,1,2,3,4,5,6,7                PCIs   0,1,4,5
-------------------------------------------------------------------------------

8192 MB memory installed        256  MB CTI cache configured (total, all nodes)
                       (7)
Primary boot path     = 1/0/0.6.0
Primary boot arguments =
Alternate boot path = 1/0/0.6.0
Alternate boot arguments =
Console path          = 15/1
Keyboard path         = 15/1
```

```
System is HP9000/800/V2500 series
                           (8)
Autoboot and Autosearch flags are both OFF or we are in HP core mode.
Processor is entering manual boot mode.

Command                           Description
-------                           -----------
AUto [BOot|SEArch|Force ON|OFF]   Display or set the specified flag
BOot [PRI|ALT|<path> <args>]      Boot from a specified path
BootTimer [time]                  Display or set boot delay time
CLEARPIM                          Clear PIM storage
CPUconfig [<cpu>] [ON|OFF|SHOW]   (De)Configure/Show Processor
DEfault                           Set the system to defined values
DIsplay                           Display this menu
ForthMode                         Switch to the Forth OBP interface
IO                                List the I/O devices in the system
LS [<path>|flash]                 List the boot or flash volume
PASSword                          Set the Forth password
PAth [PRI|ALT|CON] [<path>]       Display or modify a path
PDT [CLEAR|DEBUG]                 Display/clear Non-Volatile PDT state
PIM_info [cpu#] [HPMC|TOC|LPMC]   Display PIM of current or any CPU
RemoteCommand node# command       Execute command on a remote node
RESET [hard|debug]                Force a reset of the system
RESTrict [ON|OFF]                 Display/Select restricted access to Forth
SCSI [INIT|RATE] [bus slot val]   List/Set SCSI controller parms
SEArch [<path>]                   Search for boot devices
SECure [ON|OFF]                   Display or set secure boot mode
TIme [cn:yr:mo:dy:hr:mn[:ss]]     Display or set the real-time clock
VErsion                           Display the firmware versions
[0] Command:               (9)
```

In the output shown above, the following activities are taking place during the HP V2500 SCA server hardware boot process.

1 The Power-On Self Test (POST) firmware begins server hardware initialization. One instance of POST runs on each cabinet in a multiple-cabinet server.

2 POST initializes the current cabinet's memory (cabinet ID 0, in the example).

3 POST performs an initial synchronization of all cabinets ("nodes"), and configures CTI cache memory on all cabinets.

   See "Observing Memory Configuration during System Booting" on page 184 of Chapter 8 for details on POST's memory reports.

4 The OBP (Open Boot PROM) firmware begins executing; one instance of OBP runs on each cabinet in a multiple-cabinet server.

5 OBP synchronizes all cabinets ("nodes"), using the Node Address Resolution Protocol. Following this, OBP on cabinet ID 0 takes control of all cabinets in the system.

6 OBP prints a summary of system firmware and hardware that represents all cabinets ("nodes") in the system.

7 OBP prints the boot path variable settings.

   In many cases, cabinet ID 0's boot-time variables determine the system's boot-time behavior. The cabinet ID 0 settings for AUTO BOOT, AUTO SEARCH, and PRI establish what occurs after OBP synchronizes all cabinets.

   However, each cabinet uses its own `hypernode-bitmask`, `cti-cache-size`, `node-local-size`, and AUTO FORCE boot-time variable settings.

8 OBP prints the server hardware model, and prints cabinet ID 0's boot-time variable settings for AUTO BOOT and AUTO SEARCH.

   If AUTO BOOT is set to OFF, then the Boot Console Handler menu is presented.

9 The Boot Console Handler (BCH) menu and its "Command" prompt are displayed. Before HP-UX is booted, this is the main HP V2500 server interface for selecting options and choosing which device is booted.

   The BCH menu, like all output in this example, is available from cabinet ID 0's console window.

More details about V2500 SCA server firmware are given in Chapter 4.

For information about the Service Support Processor environment, including details on console window ("sppconsole") access, see Chapter 3.

3 Service Support Processor Workstation

The Service Support Processor workstation is the main point for managing and configuring a V-Class system before and during its HP-UX boot process.

This chapter covers the Service Support Processor user accounts, tools, and work environment.

Related Information

This section lists other sources for information on the topics covered in this chapter.

Service Support Processor Manpages

These manpages are available on the Service Support Processor.

- *sppconsole*(1)
- *lcd*(1)

Other Books

These books provide more details on topics addressed in this chapter.

- *Service Support Processor Guide* — This book provides details about the Service Support Processor workstation.
- *Operator Guide: HP 9000 V2500 SCA Server* — This book covers V2500 SCA system components, booting, hardware configuration, and other topics.

Web Sites

Additional information is available at the following Web sites.

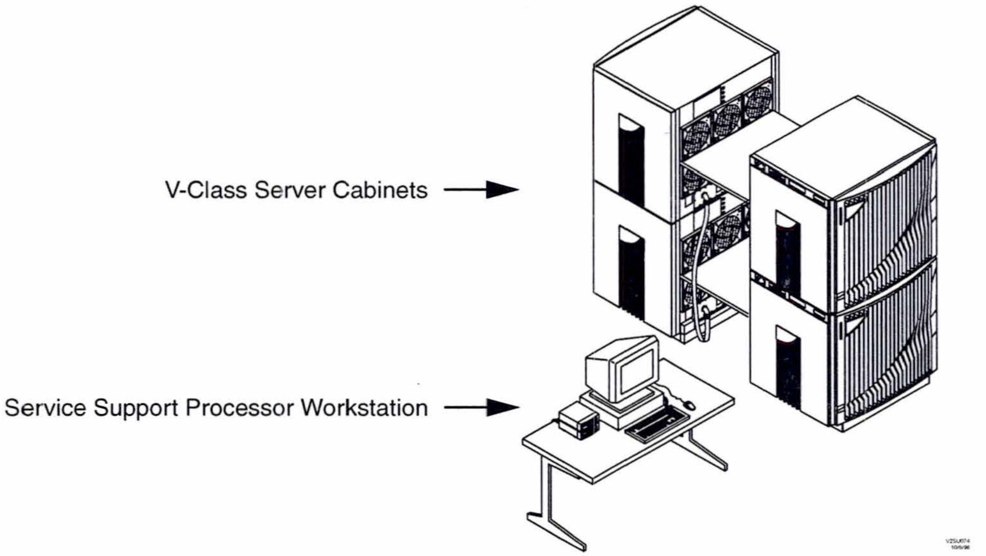
- `http://docs.hp.com/` — The HP Technical Documentation home page, which provides free online access to many publications.
- `http://docs.hp.com/hpux/systems/` — System hardware documentation, including V-Class publications.

Overview of the Service Support Processor Environment

The Service Support Processor provides a single point for monitoring an HP V-Class server as it boots, for configuring how a server's hardware resources are used, and for booting and monitoring HP-UX as it runs on the server.

Other areas in which the Service Support Processor is used include diagnostics and testing, which are usually performed by HP hardware service personnel.

Figure 15 **Service Support Processor and V-Class Cabinets**



Each Service Support Processor can manage one or more V2500 SCA server *complexes*. A server complex is a single bootable system image, such as a single-cabinet or multiple-cabinet V2500 server. See "Listing and Changing Server Complex Connections" on page 83 for more details.

The sppuser Account

The primary account to use on the Service Support Processor is the `sppuser` account, which is installed as part of the Test Station Software package.

The default password for `sppuser` is **`spp user`**, with a space between “spp” and “user”.

When you log in as `sppuser`, the user environment is configured for managing a V-Class server. The command line prompt, command and manpage paths, and other details are set up for using the specialized V-Class management software.

For direct Service Support Processor workstation log ins, the `sppuser` account’s graphical user environment (using HP’s CDE) is likewise tailored for V-Class server management.

While some detailed configurations require `root` account access, you can use the `sppuser` account for nearly all purposes.

The root Account

In most situations you should not use the `root` account on the Service Support Processor. However, if you run the `ts_config` utility for configuring or reconfiguring multiple-cabinet V2500 server complexes you’ll need `root` access.

The default password for the `root` account is **`serialbus`**.

Directories and Files

The following directories on the Service Support Processor provide the programs and files needed, in addition to HP-UX 10.20, to use the workstation for supporting various V-Class server configurations.

- /users/sppuser — The home directory for the `sppuser` account.
- /spp — The primary directory for files distributed as part of the Service Support Processor software package.

  (Note that the name, “spp”, stands for “Scalable Parallel Processor”.)
- /spp/bin — Programs for supporting V-Class and Service Support Processor work, including `diag_version`, `flash_info`, `jf-node_info`, `ts_config`, and others.

Other utilities are supplied in the /spp/firmware directory, see below.

- /spp/data — Directory containing log files, configuration data, diagnostic files, and more. Server-specific data is stored within this directory in a subdirectory with the same name as the server's complex (see below).
- /spp/data/*complex* — Directory where raw log, configuration, and other data specific to a server complex is stored.

  Each server complex's directory includes the following files, among others.

  - consolelog0, consolelog2, consolelog4, consolelog6 — Console logs for the V-Class cabinets.

    Each cabinet (ID 0, ID 2, ID 4, ID 6) logs its activity to a separate console file (consolelog0, consolelog2, and so on). Once the the server boots OBP, normally only cabinet ID 0 will have activity to be logged.
  - event_log — Log of all event information collected by `ccmd`.

    Records system reset and Transfer of Control (TOC, crash dump) events, among others.
  - hard_hist — File where hard errors are logged.
- /spp/firmware — Directory containing firmware files. These files are downloaded to the V-Class server. If firmware needs to be reinstalled, these files are used by the installation programs and scripts.
- /spp/man — Online manpages for scripts and utilities in the Service Support Processor software package.
- /spp/scripts — Scripted programs, including `lcd` and `sppconsole`.

A few other files on the Service Support Processor are configured as part of its role in supporting the V-Class server, most notably the /etc/hosts file, which `ts_config` modifies. However, you should only need to access files in the /spp/bin, /spp/scripts, and /users/sppuser directories under most normal circumstances.

Service Support Processor Connections to HP V2500 Cabinets

Connections between the Service Support Processor workstation and the V-Class server cabinets provide you communications for console, boot, and diagnostic activities.

The V2500 utilities board on each cabinet includes a console port, boot LAN, and diagnostic LAN that connect to the Service Support Processor and are used for system monitoring, booting, and other operations.

- Diagnostic (JTAG) LAN — Provides diagnostic connections for various utilities including the `ccmd` daemon that reports current cabinet status to the "test station console".

  Each cabinet has a diagnostic IP address, such as 15.99.111.116, and a diagnostic name, such as *complex*-0000 (both for cabinet ID 0, in this example).

- Boot (Core) LAN — Provides OBP boot connections for firmware and utilities.

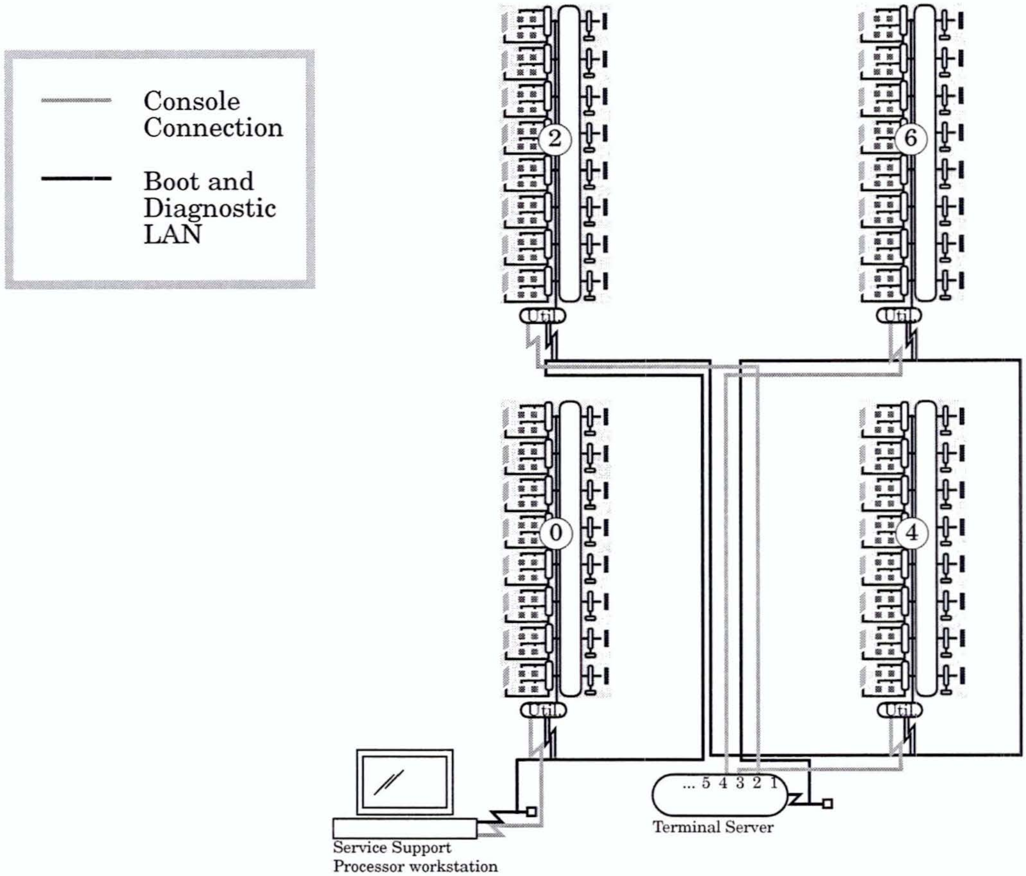
  Each cabinet also has an boot IP address, such as 15.99.111.166, and a boot name, such as obp-*complex*-000 (both for cabinet ID 0, in this example).

- Console (Serial) connection — The serial connection provides V-Class console output. This output, for each cabinet, may be viewed by using the `sppconsole` utility. It also is logged to the /spp/data/*complex*/consolelog*n* files.

  Cabinet ID 0's serial port connects to the Service Support Processor's serial port 1. Other cabinets connect through a terminal server.

A single-cabinet V2500 server connects directly to the Service Support Processor. Multiple-cabinet V2500 servers have connections to both the Service Support Processor and a terminal server, as shown in Figure 16.

Figure 16 **Service Support Processor V2500 Server Connections**



The console port on cabinet ID 0's utilities board connects to the Service Support Processor, and console ports on cabinet IDs 2, 4, and 6 connect to the terminal server (normally port numbers 2, 3, and 4, respectively).

If a Service Support Processor is connected to multiple V2500 SCA complexes, the terminal server connections may differ and additional ports may be used. Port 1 is used only when configuring the terminal server.

The boot and diagnostic LAN connects between, and is terminated at, the Service Support Processor and the terminal server. Between these two points the LAN runs in sequence to cabinet IDs 0, 2, 4, and 6.

Figure 16 on page 69 illustrates the physical cables used for multiple-cabinet console, boot, and diagnostic connections.

Figure 17 shows how OBP reports these connections during the server boot process. The connections are summarized as part of the server's Multi-node Configuration Summary during boot, shown below.

Figure 17 Service Support Processor Connections Reported During Boot

```
                                 Multi-node Configuration Summary
=============================================================================
NODE 0
Console (Serial) --> UART?  Yes
        Boot LAN --> CORE   MAC address 0:a0:d9:0:c3:cf   IP# 15.99.111.166 (0x0f636fa6)
  Diagnostic LAN --> JTAG   MAC address 0:a0:d9:0:c3:a8   IP# 15.99.111.116 (0x0f636f74)
                     MEMORY 4096 MB memory installed      1024  MB CTI cache configured
                     CPUs   0,2,8,10
                            16,18,24,26
                     PACs   0,1,2,3,4,5,6,7               MACs   0,1,2,3,4,5,6,7
                     TACs   0,1,2,3,4,5,6,7               PCIs   0,1,2,3,4,5,6,7
-----------------------------------------------------------------------------
NODE 2
Console (Serial) --> UART?  No
        Boot LAN --> CORE   MAC address 0:a0:d9:0:c6:51   IP# 15.99.111.167 (0x0f636fa7)
  Diagnostic LAN --> JTAG   MAC address 0:a0:d9:0:c5:ea   IP# 15.99.111.117 (0x0f636f75)
                     MEMORY 4096 MB memory installed      1024  MB CTI cache configured
                     CPUs   0,2,8,10
                            16,18,24,26
                     PACs   0,1,2,3,4,5,6,7               MACs   0,1,2,3,4,5,6,7
                     TACs   0,1,2,3,4,5,6,7               PCIs   0,1,4,5
-----------------------------------------------------------------------------
```

The "UART" line refers to the console connection. "Yes" indicates a direct serial connection and "No" indicates the cabinet's console connects though the terminal server. The boot (Core) and diagnostic (JTAG) connections also are summarized.

Using Special Commands on the Service Support Processor

In addition to the HP-UX 10.20 command set, several commands and scripts are provides for your use on the Service Support Processor.

Table 4 **Service Support Processor Commands and Scripts**

`/spp/bin/consolebar`	Provides an X Window interface with buttons for creating console windows for all configured cabinets.
`/spp/bin/do_reset`	Halts all activity and resets all cabinets (the default) or the specified cabinet. Supports four reset levels.
`/spp/bin/diag_version`	Prints the version of the installed Test Station Software package.
`/spp/bin/get_node_info`	Prints current V2500 cabinet configuration details for configured cabinets. The `-A` option prints all cabinets.
`/spp/bin/jf-ccmd_info`	Prints information about V-Class cabinets connected to the diagnostic LAN, including unconfigured cabinets.
`/spp/bin/jf-node_info`	Prints details about V-Class cabinets in a complex. Note that this utility can interfere with other utilities using the server's diagnostic LAN.
`/spp/bin/flash_info`	Prints the versions of firmware components installed on the specified server cabinet (default is cabinet ID 0).
`/spp/bin/set_complex`	Sets the default complex name for the current shell; useful on Service Support Processors connected to more than one V2500 SCA complex.
`/spp/bin/ts_config`	Recommended configuration utility for server and complex configurations, including CTI cache configuration. Requires root access for reconfiguration.

`/spp/bin/xconfig`	Specialized hardware configuration utility, for use by trained hardware support personnel. Can result in unsupported and unusable server configurations.
`/spp/scripts/lcd`	Prints the contents of the cabinet LCD for all cabinets in the current complex (the default) or the specified cabinet and/or complex.
`/spp/scripts/sppconsole`	Connects to and initiates a console for the specified cabinet, or for the complex console (cabinet ID 0, the default). To disconnect, type **Control-e c** . (period).

More details on the commands and scripts in Table 6 above are provided in their manpages, available on the Service Support Processor.

Other scripts and utilities are also provided on the Service Support Processor. Tools not listed in Table 4 are for detailed hardware diagnostic and various installation purposes, and are recommended for use by trained hardware service personnel only.

Powering On the System

You should turn on the Service Support Processor power and allow it to boot before powering on the V-Class server's cabinets. This allows you to use the Service Support Processor to monitor and control your V-Class server as it boots and as the server is used.

You can reboot the Service Support Processor workstation without affecting V-Class operations after the server has booted HP-UX.

The following steps are the sequence for powering on an HP V-Class server and its Service Support Processor:

Step One — Power on the Service Support Processor and allow it to boot HP-UX.

You then can log in to the Service Support Processor using the `sppuser` account.

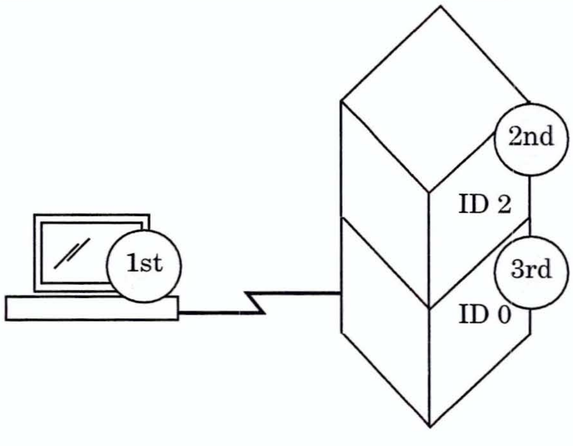
Step Two — Power on the *serf cabinets* if your system has more than one cabinet.

Serf cabinets include cabinet IDs 2, 4, and 6.

Step Three — Power on the *monarch cabinet*. All cabinets should be powered on within two minutes of each other.

The *monarch* is cabinet ID 0, and has a direct serial port connection with the Service Support Processor.

Figure 18 **Power-on Sequence for V-Class and Service Support Processor**



After the cabinets in a V2500 system are powered on, each cabinet's ID is displayed in the upper left corner of its LCD. All cabinet must be powered on within two minutes of each other in order to synchronize during Power-On Self Test (POST).

If one or more of the server's cabinets already is powered on when you boot the Service Support Processor, you can reset all cabinets after logging in to Service Support Processor `sppuser` account. Use the `/spp/bin/do_reset all 1` command after powering on all remaining cabinets.

Accessing Service Support Processor Windows

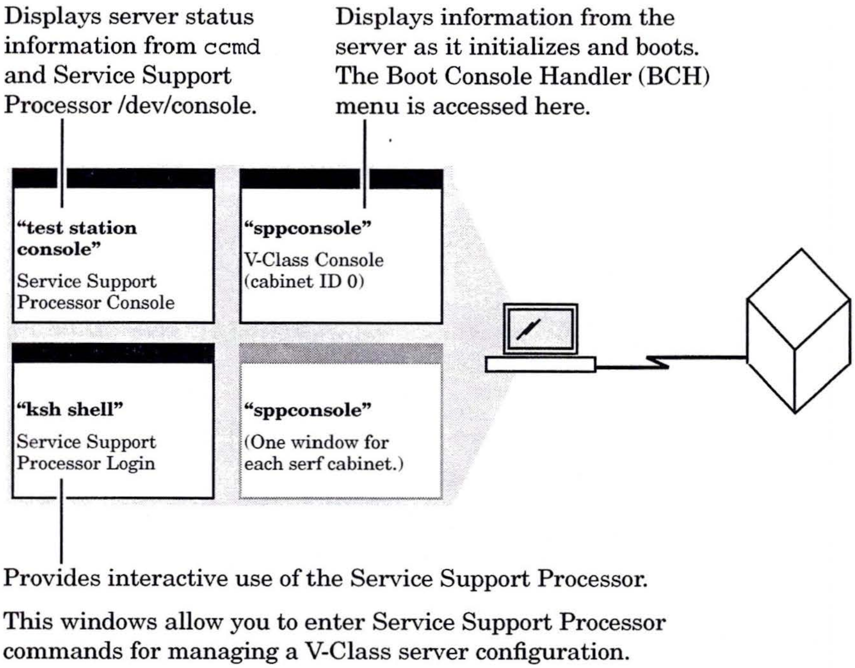
The default `sppuser` account provides three types of windows when you log in from the HP CDE (Common Desktop Environment) login prompt at the Service Support Processor workstation.

The `sppuser` CDE setup provides the following windows:

- "sppconsole" Window — Displays a single V-Class cabinet's console information and—for cabinet ID 0—provides interactive access to the Boot Console Handler (BCH) menu. All sppconsole activity is logged in the console log files on the Service Support Processor. Cabinet ID 0 is the server's "complex console".
- "test station console" Window — Displays the Service Support Processor's HP-UX /dev/console, and V-Class status from `ccmd`.
- "ksh shell" Window — Interactive HP-UX login for `sppuser` on the Service Support Processor.

Figure 19 on page 76 shows the default `sppuser` window environment.

Figure 19 **Windows Available from the Service Support Processor**

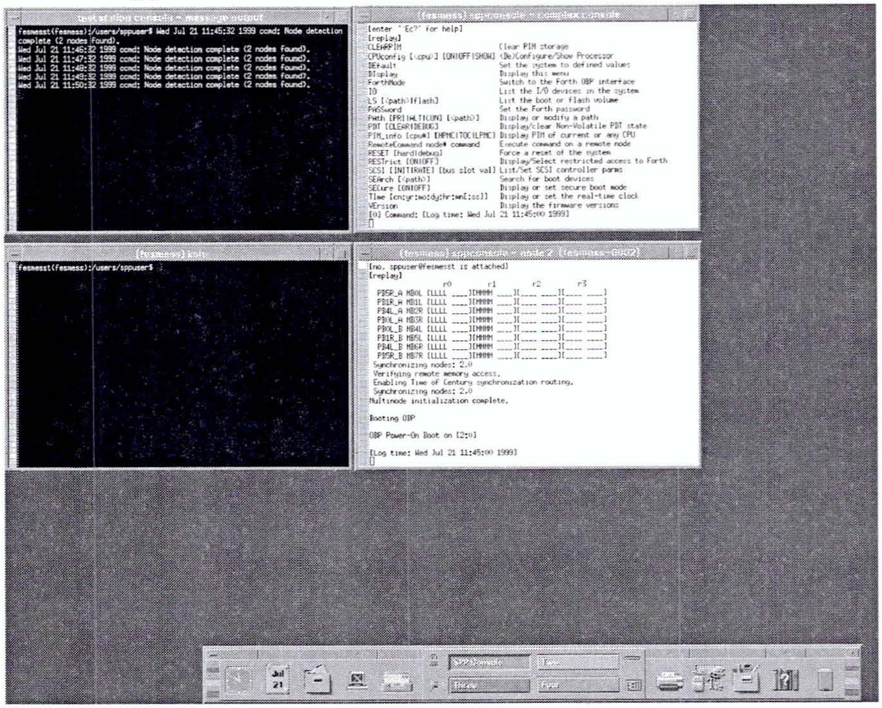


Directly Logging In to the Service Support Processor

When you log in directly to the Service Support Processor using the `sppuser` account, the HP Common Desktop Environment configuration provides the windows described in Figure 19.

Figure 20 shows a screen shot of the actual `sppuser` environment, for a Service Support Processor connected to a two-cabinet V-Class server.

Figure 20 **Service Support Processor: Direct Access**



Remotely Logging In to the Service Support Processor

You can remotely log in to the Service Support Processor workstation to manage and configure a V-Class server complex.

You can accomplish this by using the `rlogin` or `telnet` commands, if connecting from an HP-UX or other Unix environment.

From a Microsoft Windows environment, use a telnet program or X Window system emulator and initiate a terminal window for the Service Support Processor workstation.

If you have terminal-only access, you can use the `sppconsole` command to access the V-Class consoles for the various cabinets. But you cannot run graphical utilities such as `consolebar` or `ts_config`.

If you remotely connect to the Service Support Processor from an X Window environment, you can achieve a multiple-window environment similar to that provide by the Service Support Processor's CDE setup.

The following steps provide a basic remote Service Support Processor environment, when connecting from an X Window environment.

- Log in to the Service Support Processor workstation as `sppuser`.
- Set the `DISPLAY` environment variable to your local X Window server. You also may need to use `xhost` to permit remote connections.

  ```
  $ DISPLAY=myws:0 ; export DISPLAY
  ```

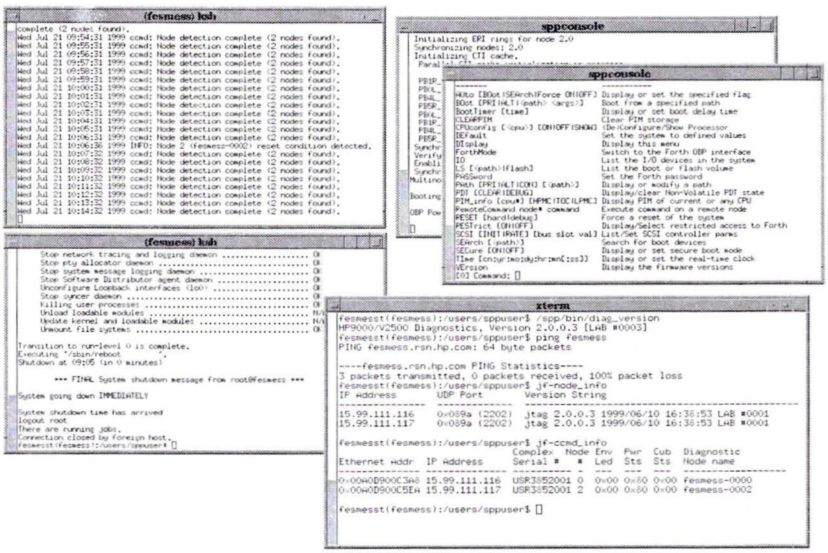
  Many utilities also accept the `-display` *displayname* option for specifying a specific X Window server from the command line.

- Run `/spp/bin/consolebar` for access to V-Class cabinet consoles, or issue the `xterm -e sppconsole` command.
- Issue `xterm -C` to monitor the "test station console", if desired.

Normally you will run the `xterm`, `ts_config`, and other X Window applications in the background by appending `&` to the command line.

Figure 21 shows a remote session with two Service Support Processor shells at the bottom, two V-Class consoles, and a "test station console".

Figure 21 **Service Support Processor: Remote Access**



Configuring HP V2500 Hardware from the Service Support Processor

Configuring and deconfiguring specific V2500 server hardware components is possible by using the `xconfig` utility, which resides on the Service Support Processor.

NOTE The `xconfig` utility should be used cautiously, if at all. Its use can create unsupported or unusable hardware configurations.

Key tasks you perform through `xconfig` include:

- Viewing a V-Class server's hardware configuration.

  `xconfig` lets you view which hardware components are present in a V-Class server and how those components are configured.

- Setting the CTI cache size for each cabinet (only on multiple-cabinet servers).

- Adjusting the amount of available memory, and configuring or deconfiguring memory.

  You can set the total amount of memory by specifying its size, and can configure and deconfigure memory by selecting specific rows of physical memory or by selecting individual DIMMs.

- Deconfiguring other hardware resources, such as CPUs or interconnecting components. In most circumstances you should not need to use this functionality.

You should use `xconfig` only before you boot HP-UX on your V-Class server. Any configuration changes require rebooting the server hardware.

The `xconfig` program gives you several capabilities not provided by other utilities. However, using `ts_config` is recommended for any necessary server reconfigurations that do not require trained hardware support staff.

Procedures

The following basic procedures are for creating and manipulating Service Support Processor windows.

Using the V-Class Console Window

The V-Class console window (`sppconsole`) restricts interactive access to one user at a time, but permits additional users to view the console in *spy mode*.

Table 5 lists the key sequences for using the V-Class console window.

Table 5 **V-Class Console Window (`sppconsole`) Commands**

V-Class Console Key Sequence	Description
`^Ecf`	Force (take) control of the console.
`^Ecs`	Spy mode: release control of the console.
`^Ecw`	List who is connected to the console.
`^Ec?`	Help: list the console key sequences.
`^Ec.`	Exit the console program.

For example, typing **Control–e c f** gives you control of the V-Class console, and typing **Control–e c s** gives up control and enters spy (read-only) mode.

Creating New Service Support Processor Windows

To modify or create new Service Support Processor windows you can use the `sppconsole`, `consolebar`, and `xterm` commands from the Service Support Processor; see Table 6 for details.

Table 6 **Commands for Creating Service Support Processor Windows**

Service Support Processor Command	Description

`/spp/scripts/sppconsole`
or
`/spp/bin/consolebar`
	These scripts create V-Class console windows for communicating with and monitoring V-Class servers:
	• The `sppconsole` script provides a V-Class console interface in the current Service Support Processor window.
	• `consolebar` displays a window with buttons for creating a new V-Class console window for any of the cabinets connected to the Service Support Processor.

`/usr/bin/X11/xterm`	Creates a Service Support Processor login window.
`/usr/bin/X11/xterm -C`	Creates a Service Support Processor console window (also called the “test station console”).

Entering BCH Menu and Forth Commands

The Boot Console Handler (BCH) menu is available for entering various booting and server configuration commands after the server hardware is powered on and boots OBP.

The BCH menu is available through the cabinet ID 0 console. See "Accessing Service Support Processor Windows" on page 75 or Chapter 2 for details.

The BCH menu also provides access to Forth mode, a lower-level command mode. Normally you should not need to access Forth mode, as it is possible to create unsupported system configurations using it.

You can specify an individual Forth command from the BCH menu by using the `FC` BCH menu command. For example, the `FC printenv node-local-size` BCH command issues the `printenv` Forth command from the BCH menu.

To enter Forth mode, use the `FM` BCH menu command. The `FM` command changes from the BCH menu interface to the Forth mode interface.

To return to the BCH menu from Forth mode, use the `mm` Forth command.

In the following example, Forth mode is accessed using the `FM` BCH menu command. Then the `printenv` Forth command is issued, and finally the `mm` Forth command is issued to exit Forth mode and return to the BCH menu.

```
[0] Command: FM
[0:2] ok printenv node-local-size
Parameter Name:  node-local-size

Current Value:

  128 MB

Default Value:

  128 MB
[0:2] ok mm
```

Listing and Changing Server Complex Connections

Each Service Support Processor can connect to multiple V2500 SCA server *complexes*, where each complex may consist of up to four V2500 SCA cabinets.

The `sppuser` account's command prompt displays both the Service Support Processor name and the current server complex name.

```
bingo-t(bingo):/users/sppuser$
```

The prompt above shows that the current complex is "bingo" and the Service Support Processor name is "bingo-t".

The `COMPLEX_NAME` environment variable indicates the V2500 server complex with which the current login shell is connected.

To change server complexes, use the `set_complex` Service Support Processor command. In the following example, the `set_complex` command changes the current complex from "lewey" to "hewey". Following this change, the `COMPLEX_NAME` environment variable also is set to "hewey".

```
hewey-t(lewey):/users/sppuser$ set_complex
COMPLEX_NAME = [Select from: hewey, lewey] hewey
hewey-t(hewey):/users/sppuser$ printenv COMPLEX_NAME
hewey
```

Examples

The examples in this section use commands and scripts available on the Service Support Processor. These are distributed as part of the Test Station Software distribution.

Printing Cabinets' LCD Output

The `lcd` command displays the contents of each cabinet's LCD.

```
fesmesst(fesmess):/users/sppuser$ lcd

Complex fesmess, node 0
|                     |
|I I       I I        |
|M I       I I        |
|abc                  |

Complex fesmess, node 2
|                     |
|I I       M I        |
|I I       I I        |
|abc                  |
```

See the section "LCD (Liquid Crystal Display)" on page 29 of Chapter 1 for details about the `lcd` command's output.

Printing the Test Station Software Version

The following example prints the Test Station Software package version using diag_version.

```
fesmesst(fesmess):/users/sppuser$ /spp/bin/diag_version
HP9000/V2500 Diagnostics, Version 2.0.0.3 [LAB #0003]
```

Listing V2500 Cabinet Connections

A list of all V-Class cabinets connected to the Service Support Processor is produced by `jf-ccmd_info`.

```
fesmesst(fesmess):/users/sppuser$ jf-ccmd_info
                                  Complex    Node Env  Pwr  Cub  Diagnostic
Ethernet Addr  IP Address         Serial #     #  Led  Sts  Sts  Node name
-------------- -------------- ---------- - ---  ---  ---  ------------------
0x00A0D900C3A8 15.99.111.116  USR3852001 0  0x00 0x80 0x00 fesmess-0000
0x00A0D900C5EA 15.99.111.117  USR3852001 2  0x00 0x80 0x00 fesmess-0002
```

The next command, `jf-node_info`, lists only the cabinets configured to the current server complex (the "fesmess" complex).

Note that the `jf-node_info` command has the potential to interfere with any diagnostic activity taking place on cabinets connected to the Service Support Processor. See *jf-node_info*(1) for details.

```
fesmesst(fesmess):/users/sppuser$ jf-node_info
IP Address        UDP Port        Version String
---------------   -------------   ----------------------------------------------
15.99.111.116     0x089a (2202)   jtag 2.0.0.3 1999/06/10 16:38:53 LAB #0001
15.99.111.117     0x089a (2202)   jtag 2.0.0.3 1999/06/10 16:38:53 LAB #0001
```

4 V-Class Firmware Components

This chapter describes V-Class firmware components. These components control the V-Class cabinet hardware from the time you power on the hardware until an HP-UX kernel is selected and loaded for booting.

The firmware also has a smaller role managing server hardware after HP-UX has booted.

Related Information

This section lists other sources for information on the topics covered in this chapter.

HP-UX Manpages

These HP-UX 11.10 manpages provide details on this chapter's topics.

- *pdc*(1M) — General coverage of processor-dependent code firmware.
- *hpux*(1M) — The SSL, which loads the HP-UX kernel during boot.

Service Support Processor Manpages

These manpages are available on the Service Support Processor.

- *post*(1M) — V-Class hardware and memory initialization firmware.
- *flash_info*(1) — Reads firmware revision information from flash ROM.

Files in the Directory /usr/share/doc

Files listed below are available on disk in the directory /usr/share/doc.

- /usr/share/doc/start_up.txt — *HP-UX 10.X Startup and Configuration*.

Other Books

These publications provide details on topics addressed in this chapter.

- *Service Support Processor Software Release Notes* — Includes details on installing and upgrading V-Class firmware and Service Support Processor software.

Web Sites

Additional information is available at the following sites on the Web.

- `http://docs.hp.com/` — The HP Technical Documentation home page, which provides free online access to many publications.

Overview of HP V2500 Firmware Components

Firmware is the software used for initial startup and booting of HP V-Class server cabinets. The firmware controls the system from the point at which you turn on each cabinet's power to the point where you initiate the booting of an HP-UX kernel.

Firmware is stored in the utilities board of each cabinet on a V-Class server. After a cabinet is powered on, firmware is copied to the cabinet's physical memory.

The V-Class firmware initially is distributed as files stored on the Service Support Processor in the /spp/firmware directory. During the Test Station Software package installation and initial V-Class server configuration, the firmware files are downloaded from /spp/firmware to the V-Class server utilities boards.

All cabinets that make up a V-Class server must be running the same firmware release.

Checking Firmware Versions

You can check which firmware release is installed on a V-Class server's cabinets by running the `/usr/bin/flash_info` command. By default, `flash_info` lists firmware versions for cabinet ID 0.

To compare the firmware versions on each cabinet within a multiple-cabinet V-Class server, list the firmware version for each cabinet (`flash_info 0`, `flash_info 2`, etc.)

Identical firmware components must be on all cabinets in a multiple-cabinet system. Also, the obp and pdc_entry versions must be identical within and across cabinets.

Installing or Updating Firmware

Updating the firmware on a V-Class server's cabinets involves loading files from the Service Support Processor /spp/firmware directory onto the V-Class cabinet's utilities boards. After this, several firmware variables (NVRAM settings on the utilities boards) are set to properly configure the system as single server.

The V-Class firmware installation, upgrade, and configuration process is performed as part of the Test Station Software package installation process.

Normally, only trained HP hardware support staff perform firmware installations and upgrades.

HP V-Class Firmware Components

While the booting process on HP V-Class servers and other Series 800 systems are similar, some of the firmware components involved differ:

- V-Class servers include several firmware components: POST (used for the Power-On Self Test), OBP (Open Boot PROM), and others. Other Series 800 servers include only PDC firmware.
- V-Class servers provide a single menu level of commands during the boot process. You can access a lower level of commands via the Forth Mode menu command. Forth mode is a maintenance interface that provides direct interaction with the OBP Forth System.
- V-Class servers do not support the "Initial System Loader" (ISL). Instead, OBP loads the "Secondary System Loader" (SSL, the `hpux` bootstrap program) directly.
- V-Class servers do not have "I/O Dependent Code" (IODC). Instead the function of IODC is provided by "Fcode device packages" that OBP compiles from I/O controller ROM at boot time. The IODC interface is emulated through a firmware component.

Despite these minor differences, you should find that the boot menus, messages, and overall boot process are essentially the same.

The next section describes how the firmware components interact when booting HP-UX on a fully configured V-Class system.

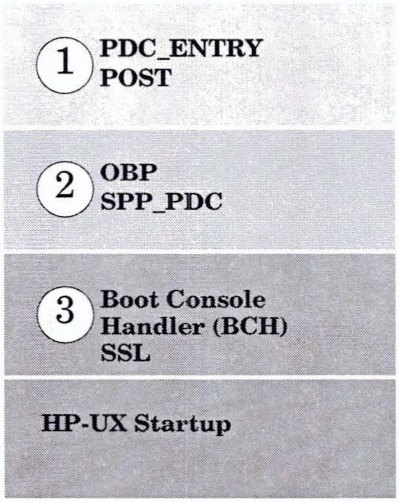
HP V2500 Firmware During the Boot Process

The major V-Class firmware components are shown below in Figure 22. These components interact among each other as an HP V-Class server boots.

You can view the overall V-Class HP-UX boot process as a series of three steps, as shown in Figure 22. The "SSL" and "HP-UX Startup" components in Figure 22 represent the HP-UX boot process itself.

Descriptions of Figure 22's components are given in the following list.

Figure 22 **Overview of HP V-Class Firmware Components**



On multiple-cabinet V-Class servers each cabinet boots its own copy of the firmware (steps 1 and 2 in Figure 22). Once OBP is booted, the various cabinet synchronize and cabinet ID 0 takes control of the multiple-cabinet server complex.

- Step One: PDC_ENTRY and POST

  PDC_ENTRY dispatches control of the machine, normally to POST.

  POST (Power-On Self Test) — Performs minimal CPU and core diagnostics to ensure the essential system components are present and are functioning. It also performs some memory initializations and configurations. On multiple-cabinet servers, an instance of POST executes simultaneously on each cabinet.

- Step Two: OBP and SPP_PDC

  OBP (Open Boot PROM) — Performs system configuration and provides booting support.

  On multiple-cabinet servers, OBP also engages in a Node Address Resolution Protocol that synchronizes all cabinets. This OBP cabinet synchronization determines which cabinets are present and establishes each cabinet's hardware configuration (all aspects except I/O). It then exchanges configuration data among all cabinets.

  OBP assigns a port on the cabinet's utilities board as the system console. It also probes all installed and enabled I/O devices, and either reads the boot device path, or automatically searches for the boot path.

  SPP_PDC (Processor-Dependent Code) — Is layered on top of OBP.

- Step Three: Boot Console Handler (BCH) and the SSL

  Boot Console Handler (BCH) — The BCH menu lets you select boot devices and options. You can interact with the system, modify its configuration, and issue commands by using the BCH menu interface. See "Using the BCH Menu and BOOT Command" on page 43 of Chapter 2 for details. This menu is bypassed if AUTO BOOT is ON.

  SSL (Secondary System Loader) — Exists as a LIF file named "HPUX" on the boot volume. The SSL loads the HP-UX kernel into memory and provides file I/O for reading /etc/ioconfig. See the *hpux*(1M) manpage for details.

After V-Class server startup has taken pace, including selecting a boot device and loading the SSL, the HP-UX startup process occurs. The HP-UX startup process:

- Initializes hardware and software devices.
- Initializes HP-UX kernel data structures and interfaces.
- Locates, checks, and mounts the root file system.
- Starts the `/sbin/init` process, which reads the file `/etc/inittab` and starts the rest of user space processes. For details see the *init*(1M) manpage.

Once HP-UX has booted, of course, you can log in to the server from an HP-UX login prompt.

Procedures and Examples

For details on installing or upgrading V-Class firmware see the release notes for the Test Station Software package.

Listing V-Class Firmware Components

The procedure for listing the installed V-Class firmware versions involves using the `/spp/bin/flash_info` command or the `jf-node_info -c` command.

To list firmware for a server cabinet other than cabinet ID 0, specify the cabinet number on the `flash_info` command line.

```
fesmesst(fesmess):/users/sppuser$ /spp/bin/flash_info 0

Node : 0 (fesmess-0000)

  Program Name           Version           Date         Build Level
-----------------------------------------------------------------
pdcfl                    2.0.0.3         1999/07/01         0005
post                     2.0.0.3         1999/07/01         0046
rdr_dumper               2.0.0.3         1999/07/01         0004
test_controller          2.0.0.3         1999/07/01         0004
mem3000                  2.0.0.3         1999/07/01         0003
eri3000                  2.0.0.3         1999/07/01         0003
cpu3000                  2.0.0.3         1999/07/01         0003
io3000                   2.0.0.3         1999/07/01         0004
diodc                    2.0.0.3         1999/07/01         0004
obp                      4.2.0.6
pdc_entry                4.2.0.6
```

The firmware versions on your system may differ from those shown above. The obp and pdc_entry versions must match within a cabinet. All firmware components must match versions across all cabinets.

Many of the firmware components are used for diagnostics purposes. Others (such as post, obp, and pdc_entry) are described in “HP V2500 Firmware During the Boot Process” on page 91.

Printing the Hypernode-Bitmask Setting

Each V2500 SCA cabinet has a hypernode-bitmask variable setting stored in the server's NVRAM. The hypernode-bitmask indicates which cabinets are configured to be in a multiple-cabinet V2500 SCA server complex.

When a V2500 server boots, the POST (Power-On Self Test) firmware on each cabinet checks the hypernode-bitmask NVRAM setting and attempts to synchronize with the cabinets that the hypernode-bitmask lists.

Figure 23 **V2500 SCA hypernode-bitmask Settings**

hypernode-bitmask	Cabinet IDs 7654 3210	
1	0000 0001	One-cabinet V2500 server
5	0000 0101	Two-cabinet V2500 server
15	0001 0101	Three-cabinet V2500 server
55	0101 0101	Four-cabinet V2500 server

You can check the hypernode-bitmask setting for a server by using the `FC printenv` Boot Console Handler (BCH) menu command.

In this example, the hypernode-bitmask setting on cabinet ID 0 is 5. This indicates that the server is configured as a two-cabinet server including cabinet IDs 0 and 2.

```
[0] Command: FC printenv hypernode-bitmask
Parameter Name:  hypernode-bitmask

Current Value:

  5

Default Value:

  0
```

5 SCA Programming and Process Management

Hewlett-Packard's Scalable Computing Architecture (SCA) allows you to combine multiple resource localities (*locality domains*) to form a single server running a single instance of the HP-UX operating system.

One example of an SCA system is a multiple-cabinet HP V2500 server, where each cabinet comprises a locality domain.

The HP-UX SCA extensions described here can help you to achieve better performance for your applications by exploiting the available resources on SCA systems.

HP-UX SCA features allow you to perform the following tasks:

- Specify which processors and memory are used by a program, including which locality domain (such as a V2500 cabinet) a thread's or process's memory and CPUs are allocated from.
- Specify the *launch policy* used when threads and processes are created. The SCA launch policies are HP-UX process management extensions for systems with multiple locality domains.
- Retrieve information about the current hardware topology.
- Inquire about thread and process *locality bindings*.

These features are supported through system calls, library extensions, and the new `mpsched` HP-UX utility.

The features described here apply to HP V2500 SCA systems running HP-UX 11.10, and may behave differently on other systems or other releases of HP-UX.

Related Information

This section lists other sources for information on the topics covered in this chapter.

HP-UX Manpages

The manpages listed below provide details on topics covered here.

- *gang_sched*(7)
- *mpsched*(1), *rtprio*(1), and *rtsched*(1)
- *ps*(1) and *top*(1)

Files in the Directory /usr/share/doc

Files listed below are available on disk in the directory /usr/share/doc and give details on topics covered here. The ASCII text files are listed below, and other file formats are available.

- sca_pm.txt—The *HP-UX SCA Programming and Process Management White Paper*.
- proc_mgt.txt—The *HP-UX Process Management White Paper*.
- mp.txt—The *HP-UX MultiProcessing White Paper*.

Web Sites

Additional information is available at the following sites on the Web.

- `http://docs.hp.com/`

  The HP Technical Documentation home page. This site provides online access to HP-UX documentation, V-Class system hardware documentation, related development tools manuals, and other information.

- `http://docs.hp.com/hpux/os/`

  HP-UX documentation including books, white papers, and other HP-UX reference sources.

Overview of HP-UX SCA

Hewlett-Packard's SCA programming and process management extensions include the HP-UX features and interfaces described in this section.

SCA Features

HP-UX SCA programming and launch features provide the following capabilities.

- Inquiry Features—Provide system hardware topology information, thread and process binding information, and thread-launch and process-launch policy information.

  You can print or retrieve information about the system hardware configuration using the `mpsched` utility, the mpctl() system call, and pthread library features. These interfaces also allow for retrieving information about thread and process bindings and launch policies.

  For example, `mpsched -s` prints the current system's configuration, including the number of localities and processors per locality.

  For details see "mpsched SCA Utility" on page 120, "mpctl() System Call" on page 122, and "pthread Library SCA Support" on page 126.

- Targeting and Binding—Provide locality domain, processor, and memory targeting and binding.

  You can specify the locality domain or processor on which a process runs by using the `mpsched` command, the mpctl() system call, and the pthread interfaces pthread_processor_bind_np() and pthread_ldom_bind_np(). For details and examples see "Working with Locality Domains" on page 99.

  You also can specify the locality from which a process' memory is allocated via the mmap() and shmget() system calls. Details are given in "SCA Memory Targeting Policies" on page 118.

- Launch and Scheduling—Provide launch policies for threads and processes, and support for gang scheduling of threads and processes.

You can set the launch policy for a process with the `mpsched` utility, the mpctl() system call, and the pthread interface pthread_launch_policy_np(). This is covered in "Using SCA Launch Policies" on page 105.

Gang scheduling of threads and processes is supported through the `mpsched -g` option and the `MP_GANG` HP-UX environment variable. Details are provided in the *gang_sched*(7) manpage and in the section "Gang Scheduling of Threads and Processes" on page 113.

SCA Interfaces

Components of this SCA programming and command interfaces include system calls, library extensions, and the `mpsched` utility. These are listed below.

- HP-UX system calls.

  The mpctl(), mmap(), and shmget() HP-UX system calls include support for using SCA resources, as briefly described in the previous section. For details, refer to the *mpctl*(2), *mmap*(2), and *shmget*(2) manpages, or see "mpctl() System Call" on page 122 and "SCA Memory Targeting Policies" on page 118.

- Library extensions.

  Both the pthread and Message Passing Interface (MPI) libraries include extensions for binding threads and processes to localities or processors, and for setting the launch policy. Details on this SCA support are in the *pthread*(3T) and *mpi*(3) manpages and in the section "pthread Library SCA Support" on page 126.

- HP-UX `mpsched` utility.

  Using `mpsched`, you can dictate the locality domain or processor on which a process executes. `mpsched` allows you to do this either by binding a process to a particular locality or processor, or by setting the launch policy for the process.

  You also can print system configuration details using the `mpsched -s` option. The `mpsched` utility is described in the *mpsched*(1) manpage and in the section "mpsched SCA Utility" on page 120.

Working with Locality Domains

SCA extensions to the HP-UX process management and memory management subsystems let you specify how system resources from various localities are used.

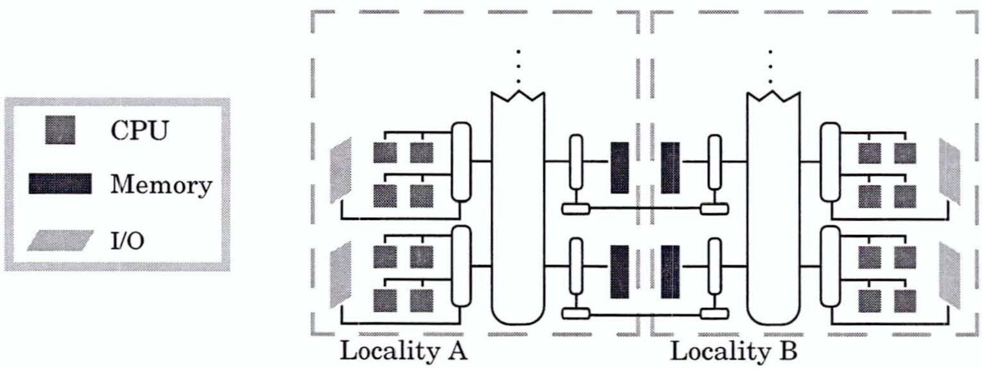
Each locality domain is a set of resources—including processors, memory, and I/O—that comprise a fundamental building block of the system. All resources within a locality have an equal latency to memory.

In a V2500 SCA server with multiple cabinets, each cabinet is a locality domain and all localities, or cabinets, form a single system. Future HP hardware architectures may include multiple localities within a single cabinet, and may include additional levels of memory latency.

When using the SCA launch policies, threads and processes are bound to the localities in which they begin execution. Details are in "Using SCA Launch Policies" on page 105. By default, HP-UX attempts to allocate memory from the same locality as the processor requesting the memory. Details are in "SCA Memory Targeting Policies" on page 118.

Figure 24 shows a conceptual overview of a two-cabinet V2500 SCA server. Portions of each locality are shown, including processors, memory, I/O, and interconnecting components. More details about V2500 hardware are given in Chapter 1.

Figure 24 **Overview of Localities in V2500 SCA Servers**



HP V2500 Memory Latencies across Localities

The term locality domain refers to a set of resources that comprise a building block of a system. More details about localities are given in the section "Working with Locality Domains" on page 99.

On V2500 SCA servers, each cabinet is a locality that provides a uniform memory latency or access time for the processors within the locality. A processor accessing memory within the same locality takes less time to read and write using local memory rather than remote memory.

Local V2500 memory accesses take approximately 502 nanoseconds. Remote V2500 memory accesses take a minimum of 1550 nanoseconds and can take longer depending on the system configuration and load.

Remote memory accesses take longer because of the additional hardware components involved when accessing memory in a remote locality.

Using specific HP-UX SCA launch policies or locality bindings can potentially benefit your programs' performance. For example, a shared-memory parallel program can benefit when all the program's threads, processes, and data reside in the same locality, because threads can more quickly share data stored in physical memory from the same locality domain.

Accessing I/O devices across localities involves a much less significant latency penalty for crossing locality domains. In many cases, the slight additional time to complete remote I/O requests is not measurable or noticeable.

Migration of Threads and Processes Across Localities

In HP-UX 11.10, each SCA locality has its own HP-UX load balancer that manages the load of all processors within the locality, and HP-UX provides a load balancer for managing the load across all locality domains in the system. The load balancers help manage the system throughput.

HP-UX avoids migrating threads and processes across localities if at all possible. However, in some situations system load imbalances cannot be resolved by migrating jobs among the processors within a single locality, and migrating to a processor in different locality is the best way to improve processor loads and system throughput.

HP-UX does not migrate some programs across localities. Programs not migrated across localities include: gang-scheduled programs, programs bound to processors, programs bound to locality domains, and programs launched using thread-launch and process-launch policies.

HP-UX keeps track of the original locality domains in which threads and processes are launched. The original locality domain is the preferred location for a thread or process to run throughout its lifetime.

If migrating a thread or process is required to improve system throughput, HP-UX attempts to minimize the number of localities into which the thread or process is migrated. If possible, HP-UX migrates the thread or process from its original locality to only one other locality.

To help minimize the overhead of migrating threads and processes to different localities, when migration is necessary HP-UX selects the eligible thread or process with the smallest working data set to be migrated.

HP-UX does not migrate memory among localities. When a program is migrated, only the processor is different.

HP-UX Numbering of Locality Domains and Processors

Each locality domain in a V-Class system has a unique locality ID number that HP-UX and applications use for identifying it. The locality IDs are assigned sequentially and may differ from the V2500 physical cabinet ID on multiple-cabinet systems, as shown in Table 7.

For single-cabinet V2500 systems, both the physical cabinet ID and the locality ID are 0.

The processor IDs (CPU IDs) that HP-UX assigns for each locality also are assigned sequentially. For instance, an HP-UX system with five CPUs includes processor IDs 0, 1, 2, 3, and 4. HP-UX processor numbering differs from the numbering used by a system's firmware to identify physical CPUs. For details on processor numbering see the section "HP-UX Numbering of Processor IDs" on page 102.

Table 7 shows the physical cabinet numbers, locality domain IDs, and HP-UX processor numbers for a four-cabinet V2500 server with 128 processors, and a four-cabinet V2500 server with 64 processors.

Table 7 **V2500 Cabinet, Locality, and Processor IDs**

Physical Cabinet ID	Locality Domain ID	HP-UX CPU IDs (64-CPU System)	HP-UX CPU IDs (128-CPU System)
0	0	0 to 15	0 to 31
2	1	16 to 31	32 to 63
4	2	32 to 47	64 to 95
6	3	48 to 63	96 to 127

Table 7 shows how HP-UX sequentially numbers both locality domain IDs and CPU (processor) IDs, starting with 0.

Unlike CPU IDs, HP-UX *hardware paths* for processors are constant, based on the CPU's location within a V2500 SCA server. See the discussion following Table 15 on page 152 of Chapter 7 for details.

HP-UX Numbering of Processor IDs

HP-UX 11.10 uses the following processor numbering (CPU ID) scheme for HP V2500 servers, including multiple-cabinet V2500 SCA configurations.

- HP-UX numbering of CPU IDs differs from the processor numbering used by the OBP (Open Boot PROM) and POST (Power-On Self Test) firmware. The HP-UX `ioscan` processor numbering also differs.
- HP-UX numbers the processors in a lower-numbered locality before numbering the processors in a higher-numbered locality.

  For example, in a server with 16 processors in two localities, locality ID 0 has CPU IDs 0–7, and locality ID 1 has CPU IDs 8–15.
- HP-UX gives the lowest CPU ID in each locality to the locality's *monarch processor* (the first processor to report back during a system hardware boot). All other CPUs are numbered in the order in which they report back during boot.
- HP-UX CPU IDs may differ at each system boot.

  For example, the CPU given processor ID 5 on one boot may be numbered processor ID 0 on a subsequent boot, depending on the order in which CPUs report back as the system boots.

- When a CPU in a cabinet (locality) is deconfigured or fails to report as system hardware boots, HP-UX adjusts the numbering of CPUs in system's localities accordingly.

  For example, after 2 of the 32 CPUs are deconfigured from cabinet 0, locality 0 would contain processors 0 to 29, locality 1 would have processors 30 to 61, locality 2 would have processors 62 to 93, and so on.

Using mpsched Support for Localities

The mpsched utility provides you ways to print a system's locality information and to bind threads and processes to a locality. Complete details about mpsched are in the *mpsched*(1) manpage and in the section "mpsched SCA Utility" on page 120.

The following mpsched -s example displays the processor configuration for a 32-processor system with two locality domains.

```
# mpsched -s
System Configuration:
=====================
Locality Domain Count: 2
Processor Count: 32
Domain    Processors
------    ----------
   0        0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
   1       16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31
```

You can bind a process to a locality using the mpsched -l option. The following example launches a.out to run in locality domain 0.

```
# mpsched -l 0 a.out
```

You can modify the locality binding for a running process by using the mpsched -l and -p options. To get the process ID (PID), use the top command or the ps command. In the following example, the mpsched command binds process ID 2493 to locality 1.

```
# mpsched -l 1 -p 2493
```

For more features and details of mpsched, see "mpsched SCA Utility" on page 120.

Programming with SCA Locality Support

You can include HP-UX SCA functionality in programs by using pthread library features and by using the mpctl(), mmap(), and shmget() system calls.

The mpctl() system call provides a programming interface for retrieving information about localities, for setting locality binding, as well as for setting launch policies. See "mpctl() System Call" on page 122 for details.

The mmap() and shmget() HP-UX system calls provide programming interfaces for binding memory use to a system's localities. This is covered in the section "SCA Memory Targeting Policies" on page 118.

The pthread library also supports SCA launch policies and supports processor binding and locality binding. See the section "pthread Library SCA Support" on page 126.

The following general guidelines may be useful when using HP-UX SCA programming features supported by HP-UX 11.10.

- Use the sysconf() call to check for HP-UX support of SCA programming extensions. Check for support by passing the `_SC_CCNUMA_PM` variable to sysconf(). For details see *sysconf*(2).
- To guarantee that threads or processes do not migrate across localities, bind them to locality domains or processors. You can do this either by using SCA launch policies or targeting and binding features.
- Do not use specific numbers to refer to locality IDs or CPU IDs. Instead, retrieve current system locality and processor information by using mpctl() and pthread features (such as the GETFIRST, GETNEXT, and other arguments).
- Write programs to handle other inquiry function return values; this ensures they will be compatible with future releases and systems.
- Program your applications to go parallel first, then initialize memory. This can help ensure that memory is allocated from the same locality domain as the requesting thread.
- Consider using SCA launch policies to distribute work in various ways across a system's localities and processors, rather than hand-coding launch and distribution methods. This allows you to create algorithms that are independent of the number of localities or processors on any given system.

Using SCA Launch Policies

HP-UX 11.10 provides several thread-launch policies and process-launch policies for distributing work in various ways among an SCA system's localities. These launch policies determine how HP-UX selects the locality domains into which it launches threads or processes.

HP-UX SCA thread-launch and process-launch policies are independent of one another. Launch policies are inherited by child threads and processes that a parent thread or process creates, as described in "Scope of Launch Policies and Policy Trees" on page 110.

You can establish the thread-launch policy and process-launch policy by using the `mpsched` command's `-T` and `-P` options, respectively, or by passing arguments to the mpctl() system call. The pthread_launch_policy_np() library call supports thread-launch policies.

The SCA launch policies, except the None policy, guarantee that HP-UX binds threads and processes to the locality domains in which they start execution. This locality binding causes HP-UX to not migrate the threads and processes across localities.

HP-UX 11.10 supports the following SCA thread-launch and process-launch policies. Detailed descriptions are in Table 8 on page 107.

- **None**—Is the default launch policy for threads or processes that are not gang-scheduled. See "Default Launch Policies" on page 106 for details.
- **Round Robin**—Launches by alternating the thread or process placement among all localities until all localities are selected once, then starts over as needed.
- **Fill First**—Fills a locality by placing threads or processes in the locality until all processors are selected. It then spills over to another locality, as needed. Once all localities are filled, starts over as needed.

  This is the default policy for gang-scheduled threads and processes.
- **Packed**—Places all threads or processes in the same locality; it does not spill over.
- **Least Loaded**—Places each thread or process in the locality that is least-loaded at the time of its creation.

Default Launch Policies

By default, for jobs that are not gang-scheduled, HP-UX launches threads and processes using a launch policy called None. The None policy implies that HP-UX is free to launch threads and processes as it determines is best for system performance.

The HP-UX gang scheduler uses a default launch policy of Fill First when launching gang-scheduled threads and processes within a single gang. HP-UX maintains a separate fill count for each gang.

No locality binding is established by the None policy, so HP-UX may migrate threads and processes that use this policy.

You should not rely on the None launch policy to provide any specific behavior, because it may change from release to release or from hardware platform to hardware platform.

If you need to guarantee that a specific launch behavior is maintained when running a program on a variety of systems, specify an SCA thread-launch policy or process-launch policy for the program.

The HP-UX 11.10 release of SCA features on V2500 systems uses the following methods for launching threads and processes using the None launch policy:

- **Threads**—Are first placed by the default HP-UX 11.10 thread-launch policy (None) into the current locality. Then, if there are more threads than processors in the current locality, threads spill over into another locality, which is selected for highest performance. This is equivalent to a Fill First thread-launch policy with no binding.
- **Processes**—Are placed by the default HP-UX 11.10 process-launch policy (None) into the locality with the fewest threads and processes currently in the run queue. This is equivalent to a Least Loaded process-launch policy with no binding.

The None policy may perform differently on other hardware platforms and in future HP-UX releases. All SCA policies are described in "Descriptions of SCA Launch Policies" on page 107.

Descriptions of SCA Launch Policies

The SCA launch policies supported in the HP-UX 11.10 release are listed in Table 8.

HP-UX supports these policies as thread-launch policies and as process-launch policies. The policies are inherited by child threads and processes as described in "Scope of Launch Policies and Policy Trees" on page 110.

All launch policies permit the processors in a server's localities to be *oversubscribed*. That is, the number of threads and processes in an application may exceed the number of processors in a single locality, or may exceed the total number of processors in a server.

The Packed policy causes threads and processes to be launched only within a single locality. All other policies distribute threads and processes among all locality domains using various methods.

Table 8 SCA Thread-Launch and Process-Launch Policies

Launch Policy Description	Interfaces

None
None is the default policy for launching threads and processes that are not gang-scheduled.
Under the None policy, HP-UX tries to launch threads and processes in a manner that is best for performance. Do not rely on this policy for a specific launch behavior, as None may differ on future HP hardware platforms and HP-UX releases.
	pthread_launch_policy_np() support:
	`PTHREAD_POLICY_NONE_NP`
	
	mpctl() system call support:
	`MPC_SETPROCESS_NONE` and `MPC_SETLWP_NONE`

Round Robin
The Round Robin policy launches each new thread or process in a round-robin (alternating) fashion across all locality domains.
Under Round Robin, the newly created thread or process is launched on the least loaded locality, and each subsequent thread or process launches round-robin into remaining localities, each time selecting the least loaded of the remaining localities. Once all localities are used, round-robin starts over, again allowing placement in all localities and beginning with the least loaded locality.
	`mpsched` policy name: `RR`
	
	pthread_launch_policy_np() support:
	`PTHREAD_POLICY_RR_NP`
	
	mpctl() system call support:
	`MPC_SETPROCESS_RR` and `MPC_SETLWP_RR`

Launch Policy Description	Interfaces

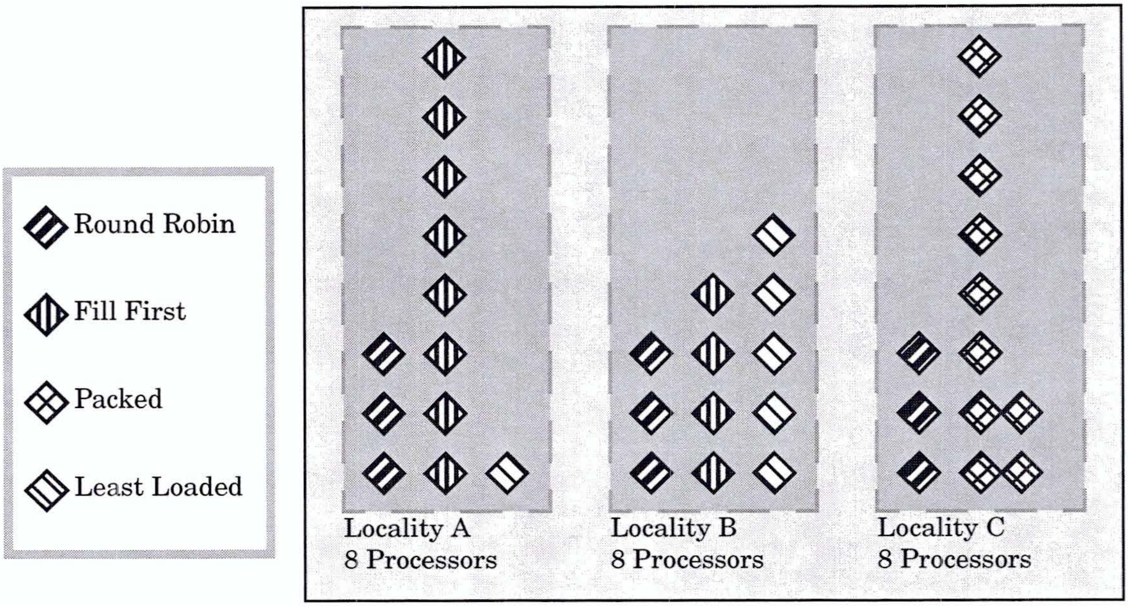
Fill First
The Fill First policy launches each new thread or process into the same locality until the number of launched threads or processes equals the number of processors in the locality. Threads or processes then spill over into the next locality and launch there, as needed, until it is filled. Once all localities are filled, the Fill First policy starts over. When this occurs, all locality domains are again available for placement and Fill First selects the least loaded locality, fills it as needed, then selects the least loaded of the remaining localities.
Fill First is the default policy for programs that are gang-scheduled.
	`mpsched` policy name: `FILL`
	
	pthread_launch_policy_np() support:
	`PTHREAD_POLICY_FILL_NP`
	
	mpctl() system call support:
	`MPC_SETPROCESS_FILL` and `MPC_SETLWP_FILL`

Packed
The Packed launch policy causes all of a program's threads or processes to be launched into the same locality.
Under the Packed policy, a locality domain can become oversubscribed. That is, the number of threads and processes may exceed the number of processors in the locality. Threads and processes launched under the Packed policy do not spill over into other localities.
	`mpsched` policy name: `PACKED`
	
	pthread_launch_policy_np() support:
	`PTHREAD_POLICY_PACKED_NP`
	
	mpctl() system call support:
	`MPC_SETPROCESS_PACKED` and `MPC_SETLWP_PACKED`

Least Loaded
The Least Loaded policy launches newly created threads or processes into the least loaded locality domain in the system at the time of creation.
For HP-UX 11.10, the least loaded locality domain is the one with the lowest load average within the last second, based on the number of active runnable threads and processes (excluding those in a short-term I/O wait).
	`mpsched` policy name: `LL`
	
	pthread_launch_policy_np() support:
	`PTHREAD_POLICY_LEASTLOAD_NP`
	
	mpctl() system call support:
	`MPC_SETPROCESS_LEASTLOAD` and `MPC_SETLWP_LEASTLOAD`

Sample uses of SCA launch policies are shown in Figure 25, which represents a three-locality system with four jobs running. Each job in Figure 25 has been launched using a different HP-UX SCA launch policy. This sample system has 24 processors, eight processors in each locality.

Figure 25 **Distribution of Threads and Processes by SCA Launch Policies**



The sample placement of the four jobs in Figure 25 assumes that each job is launched in its entirity before the next is launched (Round Robin first, then Fill First, Packed, and Least Loaded). The actual placement of threads and processes would depend on the order in which the jobs are launched and other activity on the system.

In Figure 25, the Round Robin job is distributed equally across all the system's localities. The Fill First job launches into a locality (Locality A) until all the locality's processors are selected, then spills over into another locality (Locality B). The Packed job launches into a single locality (Locality C) and can oversubscribe that locality's processors. Finally, the Least Loaded job launches each of its members into the locality with the lowest current processor utilization.

Scope of Launch Policies and Policy Trees

HP-UX maintains which thread-launch and process-launch policies are in effect for threads and processes by using *policy trees*.

A policy tree is a collection of threads or processes that share the same SCA launch policies and *launch count* details. The launch count and the thread-launch or process-launch policy are maintained by the policy tree's *root*.

HP-UX 11.10 supports two types of policy trees:

- **Thread Policy Trees**—Are maintained as part of the thread structure and include a thread-launch policy and launch count.
- **Process Policy Trees**—Are maintained as part of the process structure and include a process-launch policy and launch count.

HP-UX refers to the root of the associated policy tree when a pthread_create() call creates a new thread, or when a fork() call creates a new process. HP-UX launches the new thread or process using the thread-launch or process-launch policy specified by the root. HP-UX also updates the policy tree's launch count.

HP-UX maintains the launch count to track the number and location of threads or processes launched within a policy tree. Every policy tree has its own launch count.

HP-UX establishes a new policy tree, which has a new launch count and may have a different launch policy, under the following circumstances:

- When a thread changes the thread-launch policy, such as when calling the pthread_launch_policy() routine or mpctl() system call.

  The calling thread is the new policy tree's root.

- When a process has its process-launch policy changed, such as when mpctl() is called.

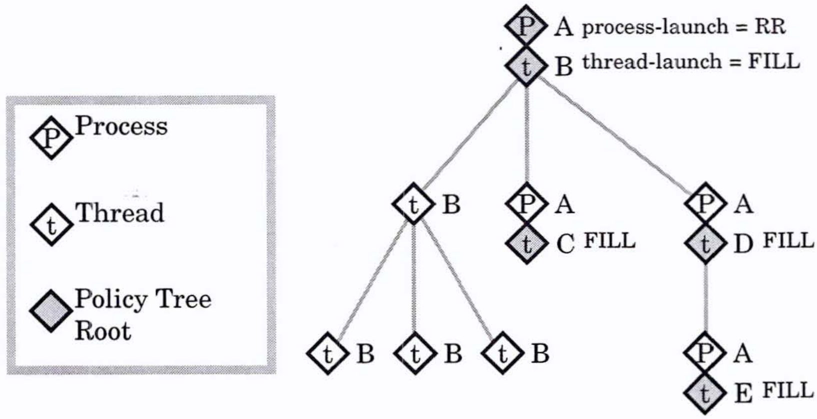
  The process is the new policy tree's root. Any thread in a process can change the process-launch policy.

- When a thread calls fork() to create a process.

  The new process is a member of the same process policy tree that the thread's process belongs to. But the new thread created as part of the new process is the root of a new thread policy tree. The new thread policy tree inherits the thread-launch policy of the calling thread.

Figure 26 shows a sample process, the threads and processes it creates, and the associated policy trees and SCA launch policies.

Figure 26 **Sample HP-UX Launch Policy Trees**



The first process in Figure 26 is launched with a Round Robin process-launch policy and a Fill First thread-launch policy. As a result, the root of the process policy tree (labeled A in Figure 26) establishes a Round Robin process-launch policy, and the root of the thread policy tree (B) establishes a Fill First thread-launch policy.

The thread created by the initial thread in Figure 26 is a member of the same thread policy tree, and HP-UX launches it using the Fill First thread-launch policy. Likewise, threads created by the secondary thread also are members of the same thread policy tree (B).

For the two child processes created by the initial thread in Figure 26, HP-UX creates one process and one thread for each. The child processes are members of the same policy tree (A) as the thread's parent, and are launched using the Round Robin process-launch policy. The child processes' threads, however, are the roots of new thread policy trees. These new thread policy trees (C and D) inherit the Fill First launch policy from the creating thread's policy tree (B).

Finally, in Figure 26, the thread in policy tree D calls fork() to create yet another child process. This new process is a member of the same policy tree as the thread's process (A). The new thread that is created by the fork() call is the root of a new thread policy tree (E) which inherits the thread-launch policy from the thread's policy tree (D).

Combining Launch Policies with Gang Scheduling

The HP-UX gang scheduling feature enables you to force HP-UX to concurrently schedule all threads or processes in an application for execution.

By default, the HP-UX gang scheduler launches threads and processes using a Fill First launch policy. You cannot override the default gang scheduling launch policy, but you can dictate the initial locality domain into which the gang scheduler launches threads and processes.

See "Gang Scheduling of Threads and Processes" on page 113 for details.

Performance Benefits of SCA Launch Policies

Benefits of the HP-UX SCA thread-launch and process-launch policies depend on the characteristics of the workload being launched.

- Round Robin and Least Loaded launch policies provide methods of spreading work across the system, potentially reducing contentions for resources. Round Robin spreads work evenly. Least Loaded spreads work to areas of lowest current CPU utilization.

  Least Loaded generally is useful for CPU-intensive applications where processor availability is most critical. Round Robin is more appropriate for applications that are memory-bandwidth intensive, but generate little communication traffic among the threads and processes.

- Fill First and Packed launch policies provide methods of allowing threads and processes to share memory. Packed ensures all threads and processes remain in the same locality. Fill First allows threads or processes to spill over to other localities after all CPUs in the initial locality are subscribed.

  The Fill First and Packed policies are appropriate for applications with a significant amount of communication among the threads and processes.

Assigning different thread-launch and process-launch policies may be appropriate for certain programs. For example, a program that creates 4 processes, each of which spawns 10 threads, may benefit from a Round Robin process-launch policy and a Packed thread-launch policy.

Gang Scheduling of Threads and Processes

The HP-UX gang scheduler permits a set of MPI processes, or multiple threads from a single process, to be scheduled concurrently as a group. The gang scheduler may only be used with timeshare processes. It is not supported for use with real-time processes. As of the HP-UX 11.10 release, the HP Process Resource Manager (PRM) does not support gang scheduling.

The gang scheduler is engaged only when a gang consists of multiple threads. For a pthread application, this is when a second thread is created. For an MPI application, it is when a second process is added.

Using Gang Scheduling

You can enable and disable gang scheduling by setting the `MP_GANG` environment variable to `ON` or `OFF`. Any MPI or pthread application to execute and find `MP_GANG` set to `ON` will enable gang scheduling for the application.

All uppercase letters must be used when specifying the `MP_GANG` environment variable name and its setting (`ON` or `OFF`).

You also can launch an application with gang scheduling enabled by using the `mpsched -g` option. When using this method, the `mpsched` process is not considered to be a gang member.

Details on HP-UX gang scheduling are available in the *gang_sched*(7) manpage. See also the *mpsched*(1) manpage and the description of the `mpsched -g` option.

Performance Benefits and Issues

The HP-UX gang scheduling feature can significantly improve parallel application performance in loaded timeshare environments that are *oversubscribed* or *partially oversubscribed*.

Oversubscription occurs when the total number of runnable threads and processes exceeds the number of processors in the system (or locality). A system is considered partially oversubscribed when one locality is oversubscribed and another locality is undersubscribed.

By concurrently scheduling threads in a gang, HP-UX gang scheduling can provide low-latency interactions among a gang's threads in shared-memory applications.

Gang scheduling a process involves system overhead due to synchronizing the execution of all members of the gang. As a result, gang-scheduled threads and processes involve more system time than non-gang-scheduled threads and processes.

The additional system time for gang-scheduled processes increases with the number of threads and processes in the gang. For processes and threads that do not benefit from gang scheduling, the system overhead may add to the execution time. However, for applications benefiting from gang scheduling, the system time costs are outweighed by gang scheduling performance benefits.

Gang Sizes

To have concurrent execution of all members, a gang cannot consist of more threads and processes than available processors.

When a gang's size equals the number of processors in the system, the following occurs:

- New threads or processes are not added to the gang.
- The gang remains intact and continues to be gang scheduled.
- The spill-over threads are scheduled with the regular timeshare policies.
- If threads in the gang exit (thus making room available), the spill-over threads are not added into the gang. However, newly created threads are added into the gang when room is available.

Note that for MPI applications, all processes are allocated statically at the beginning of execution.

Gang Membership

For MPI applications that are gang-scheduled, all processes in the application are members of the same gang.

For gang-scheduled pthread programs, all threads within a process are members of the same gang. However, when a non-MPI program creates a child process, a new gang is formed for the child if the parent is gang-scheduled. The child process does not become a member of its parent's gang.

Gang scheduling is an inherited process attribute, so processes created by a fork() call inherit the gang scheduling attribute.

Gang Scheduler Launch Policies

By default, the HP-UX gang scheduler uses its own implementation of the Fill First launch policy when placing gang-scheduled threads and processes. Programs that are not gang-scheduled by default use a launch policy called None.

Under the gang scheduler's Fill First launch policy, when a new gang-scheduled process is launched, it is placed in the locality domain with the fewest gangs or lowest gang load. When all processors in the initial locality are assigned threads or processes from the gang, the gang scheduler selects a second locality into which it launches threads or processes.

Gang Scheduler Load Balancing and Bindings

Locality bindings can be used in conjunction with gang scheduling. See the next section, "Specifying the Locality Placement of Gangs".

Load balancing among gangs occurs only within a locality domain—gangs are not balanced or migrated across localities.

For gangs that span multiple locality domains, HP-UX schedules all gang members to execute concurrently—just as occurs for gangs whose members are contained within a single locality.

The HP-UX gang scheduler attempts to give an equalized percentage of CPU time to members of all gangs. In doing this, HP-UX rebalances the CPU assignments of gang members within locality domains that are oversubscribed. This rebalancing occurs every few seconds.

When a single gang executes on a system, the gang's threads are bound to processors and are not migrated to different processors or localities.

In an oversubscribed (or partially oversubscribed) system with multiple gangs, gang members are periodically migrated to different processors within the same locality.

For a gang that spans multiple localities, it is possible for some of the gang's members to remain bound to processors (in an undersubscribed locality) and other members to migrate among processors (in a second, oversubscribed locality with multiple gangs).

Gang-scheduled threads and processes are migrated to different processors only to accommodate members from another gang, as part of the gang scheduler's effort to provide equalized CPU time among gangs.

As mentioned earlier, gangs and gang members currently are never migrated across localities.

Specifying the Locality Placement of Gangs

You can specify a locality binding to be applied to a process that will be gang scheduled. When doing so, you override the gang scheduler's default initial locality selection. The locality domain you specify is used as the initial locality into which gang-scheduled threads and processes are placed.

When combining gang scheduling with a locality binding, you must specify the locality placement before gang scheduling is enabled.

If a locality binding is requested after gang scheduling is enabled, then gang scheduling is disabled for the process.

The HP-UX gang scheduler always uses its default launch policy (a Fill First policy), which will select additional localities if the gang has more threads and processes than processors in the initial locality domain.

Example

The following command line establishes the initial locality for a gang-scheduled process.

```
# mpsched -l 1 mpsched -g ./potato -n 5
```

Above, the first `mpsched` command sets a locality binding (to locality 1), and the second `mpsched` command enables gang scheduling. These are applied to the `potato` program.

SCA Timeshare and Real-Time Support

Your jobs can be executed using either of the two classes of scheduling priorities that are available in HP-UX: timeshare priority and real-time priority.

SCA launch policies can be used with both real-time and timeshare processes. The HP-UX gang scheduling feature, however, can be used only with timeshare processes.

NOTE

Real-time threads and processes can not be gang-scheduled.

A real-time processes can potentially prevent one or more gangs from executing during its life because of the real-time process' higher scheduling importance. To avoid this, make sure long-running real-time processes do not run in the same locality as gang-scheduled processes.

By default, HP-UX user processes are scheduled using the HP-UX timeshare scheduler. Several varieties of real-time schedulers are optionally available for executing jobs with a real-time priority, which has greater scheduling importance.

Real-time processes, unlike timeshare processes, are not subject to priority degradation. You can schedule real-time threads and processes using either the POSIX or HP-UX real-time schedulers; for details refer to *rtprio*(1) and *rtsched*(1).

SCA Memory Targeting Policies

The SCA memory targeting policies let you request that HP-UX allocate memory from the same locality domain as the requesting thread. The SCA memory targeting policies are supported by the mmap() and shmget() system calls.

Note that if memory cannot be allocated from the requested (or current) locality, memory from the next closest locality is used to fulfill the request.

These memory policies cause the targeted memory regions to be allocated from the specified locality, regardless of the faulting thread's location. Paging has no affect on the locality of targeted memory regions—the memory region remains in the targeted locality domain.

The default memory allocation policy is first touch. That is, memory for a page fault is allocated from the locality domain in which the fault occurs. The first touch policy applies to data, text, stack, and shared memory. Under this policy, after a memory region is paged out it can be paged back into any locality domain based on first touch.

SCA Memory Support in mmap() and shmget()

You can specify the locality domain from which a memory region is allocated when using the mmap() and shmget() routines. These routines include flags for targeting memory regions to be allocated from a specified locality.

The mmap() routine includes the `MAP_LOCAL` flag, which targets memory to be allocated from the same locality domain as the calling thread. Likewise, the shmget() routine provides the `IPC_MEM_LOCAL` flag, for targeting shared memory to the same locality as the calling thread.

Should there not be enough memory available in the current locality to satisfy the request, memory from the next closest locality will be allocated to satisfy the request.

SCA extensions to these routines are described below. See the *mmap*(2) and *shmget*(2) manpages for more details.

Table 9 Memory Routines Supporting SCA Localities

Routine and Syntax	SCA Extensions and Example

Routine
mmap()
 	
Syntax
`#include <sys/mman.h>`
`caddr_t mmap(caddr_t addr,`
`    size_t len, int prot,`
`    int flags, int fildes,`
`    off_t off);`
	SCA Extensions
	`MAP_LOCAL` is a new mmap() `flag` value for allocating memory from the same locality domain as the calling thread.
	
	Example
	`caddr_t  addr1`
	`size_t   len`
	`...`
	`addr1 = mmap((caddr_t)0,`
	`    (size_t)len, PROT_READ |`
	`    PROT_WRITE, MAP_SHARED |`
	`    MAP_VARIABLE | MAP_ANONYMOUS`
	`    | MAP_LOCAL, -1, (off_t)0);`

Routine
shmget()
 	
Syntax
`#include <sys/shm.h>`
`int shmget(key_t key, size_t size,`
`    int shmflg);`
	SCA Extensions
	`IPC_MEM_LOCAL` is a new shmget() `shmflg` value for targeting shared memory to the same locality domain as the calling thread.
	
	Example
	`int shmid`
	`...`
	`shmid = shmget(IPC_PRIVATE, 1024,`
	`    IPC_CREAT | IPC_MEM_LOCAL);`

mpsched SCA Utility

The `mpsched` HP-UX utility provides a command-line interface for applying launch policies, for binding threads and processes to locality domains and processors, and for printing details about system hardware and running processes.

Table 10 lists supported `mpsched` features. For more details see the *mpsched*(1) manpage.

Table 10 **mpsched Utility SCA Features**

SCA Feature and Description	`mpsched` Options and Examples
Launch Policies Apply SCA launch policies to threads or processes.	• `mpsched -P` *policy* `a.out` Applies the specified process-launch *policy* to the process `a.out` and processes it creates. • `mpsched -T` *policy* `a.out` Applies the specified thread-launch *policy* to threads created by the process `a.out`. The *policy* can be `RR`, `LL`, `FILL`, or `PACKED`.
Locality Domain and Processor Binding Bind and unbind processes to locality domains or processors.	• `mpsched -u -p` *pid* Unbinds the process/command specified by *pid* from any processor or locality bindings. • `mpsched -c` *processor* `...` Binds the process/command to the specified *processor*. Either a process ID (`-p` *pid*) or command may be specified. • `mpsched -l` *locality* `...` Binds the process/command to the specified *locality* domain. Either a process ID (`-p` *pid*) or command may be specified. `mpsched -s` prints *processor* and *locality* values.

SCA Feature and Description	`mpsched` Options and Examples
Gang Scheduling Enable gang scheduling for a process.	The `-g` option enables gang scheduling for the process/command being launched by `mpsched`. No other options can be specified with `-g`. To combine gang scheduling with launch policies, see the section "Combining Launch Policies with Gang Scheduling" on page 112.

System Hardware Topology
Print system hardware locality domain and processor details.
	`mpsched -s` prints the system's locality domain and processor configuration, including all processor and locality identifying numbers.
	The following shows a two-locality system with 16 processors.
	`#` **`mpsched -s`**
	`System Configuration:`
	`=====================`
	`Locality Domain Count: 2`
	`Processor Count: 16`
	`Domain    Processors`
	`------    ----------`
	`   0        0   1   2   3   4   5   6   7`
	`   1        8   9  10  11  12  13  14  15`

Thread and Process Inquiry Print processor binding information and launch policy information.	`mpsched -q -p` *pid* prints any locality or processor binding for the specified process ID (*pid*). This also prints any process-launch policies for the process. Thread-launch policies are not reported. The following show output for a process with locality binding and the Packed process-launch policy. `#` **`mpsched -q -p2480`** `Pid 2480 has locality domain based binding with a packed process launch policy`

mpctl() System Call

The mpctl() system call provides a means of determining how many processors and locality domains are installed in a system, and getting detailed information about them.

Using mpctl(), you also can bind threads ("light-weight processes") and processes to run on specific processors or localities. mpctl() also supports SCA thread-launch and process-launch policies.

See the *mpctl*(2) manpage for more details.

Syntax

```
#include <sys/mpctl.h>

int mpctl(mpc_request_t request, spu_t spu, pid_t pid);

int mpctl(mpc_request_t request, spu_t spu,
    lwpid_t lwpid);

int mpctl(mpc_request_t request, ldom_t ldom,
    pid_t pid);

int mpctl(mpc_request_t request, ldom_t ldom,
    lwpid_t lwpid);
```

SCA Extensions to mpctl()

Extensions to the mpctl() call for supporting SCA systems include new `request` values, and allow both processor (`spu`) and locality domain (`ldom`) arguments to be specified. More detailed descriptions of these `request` arguments are available in the *mpctl*(2) manpage.

The SCA `request` arguments provide you the following capabilities.

- Bind threads and processes to specified processors and locality domains.

  These `request` arguments allow advisory and forced binding of threads and processes to processors, and allow thread and process binding to locality domains: `MPC_SETPROCESS`, `MPC_SETPROCESS_FORCE`, `MPC_SETLWP`, `MPC_SETLWP_FORCE`, `MPC_SETLDOM`, `MPC_SETLWPLDOM`.

- Set the launch policy for threads and policies.

  These `request` arguments set process-launch policies: `MPC_SETPROCESS_RR`, `MPC_SETPROCESS_FILL`, `MPC_SETPROCESS_PACKED`, `MPC_SETPROCESS_LEASTLOAD`, `MPC_SETPROCESS_NONE`.

  The corresponding `request` arguments for thread-launch policies are: `MPC_SETLWP_RR`, `MPC_SETLWP_FILL`, `MPC_SETLWP_PACKED`, `MPC_SETLWP_LEASTLOAD`, `MPC_SETLWP_NONE`.

  These launch policies are described in the *mpctl*(2) manpage and in the section "Using SCA Launch Policies" on page 105.

- Inquire about the system hardware topology.

  The following `request` arguments provide topology information.

  For these `request` arguments, both the `ldom` and `pid` arguments are ignored: `MPC_GETNUMSPUS` (number of processors), `MPC_GETFIRSTSPU` (ID of first processor), `MPC_GETNEXTSPU` (ID of processor following `spu`), `MPC_GETCURRENTSPU` (ID of current processor), `MPC_GETNUMLDOMS` (number of locality domains), `MPC_GETFIRSTLDOM` (ID of first locality domain).

  For these `request` arguments, the `pid` argument is ignored: `MPC_GETNEXTLDOM` (ID of locality domain following `ldom`), `MPC_LDOMSPUS` (number of processors in locality specified by `ldom`), `MPC_SPUTOLDOM` (locality ID of the locality that contains the processor specified by `spu`).

- Inquire about thread and process bindings and launch policy settings.

  These `request` arguments retrieve information about thread and process bindings and launch policies: `MPC_GETPROCESS_BINDVALUE`, `MPC_GETLWP_BINDVALUE`, `MPC_GETPROCESS_LAUNCH`, `MPC_GETLWP_LAUNCH`.

  For specific details on SCA launch policies see the *mpctl*(2) manpage and the section "Using SCA Launch Policies" on page 105. The mpctl() return values for these request arguments are listed in Table 11.

SCA Launch and Binding Inquiry Values

The mpctl() system call provides several `request` arguments for inquiring about thread-launch and process-launch policies and bindings. Both processor and locality domain binding information can be retrieved using mpctl().

Table 11 lists the mpctl() `request` arguments for inquiring about bindings and launch policies, and associated return values.

Table 11 mpctl() SCA Launch and Binding Inquiry Values

mpctl() `request` Argument and Description	Return Values
`MPC_GETPROCESS_BINDVALUE` Returns the current binding type for the process specified by `pid`. The `spu` argument is ignored.	• `MPC_SPU_FORCED_BINDING`—Forced processor binding. • `MPC_SPU_BINDING`—Advisory processor binding. • `MPC_LDOM_BINDING`—Locality domain binding. • `MPC_NO_BINDING`—No binding.
`MPC_GETLWP_BINDVALUE` Returns the current binding type for the thread ("light-weight process") specified by `lwpid`. The `spu` argument is ignored.	• `MPC_SPU_FORCED_BINDING`—Forced processor binding. • `MPC_SPU_BINDING`—Advisory processor binding. • `MPC_LDOM_BINDING`—Locality domain binding. • `MPC_NO_BINDING`—No binding.

mpctl() request Argument and Description	Return Values
MPC_GETPROCESS_LAUNCH Returns the current process-launch policy for the process specified by pid. The ldom argument is ignored.	• MPC_LAUNCH_POLICY_RR—Round Robin process-launch policy. • MPC_LAUNCH_POLICY_FILL—Fill First process-launch policy. • MPC_LAUNCH_POLICY_PACKED—Packed process-launch policy. • MPC_LAUNCH_POLICY_LEASTLOAD—Least Loaded process-launch policy. • MPC_LAUNCH_POLICY_NONE—None (the default process-launch policy).
MPC_GETLWP_LAUNCH Returns the current thread-launch policy for the thread (“light-weight process”) specified by lwpid. The ldom argument is ignored.	• MPC_LAUNCH_POLICY_RR—Round Robin thread-launch policy. • MPC_LAUNCH_POLICY_FILL—Fill First thread-launch policy. • MPC_LAUNCH_POLICY_PACKED—Packed thread-launch policy. • MPC_LAUNCH_POLICY_LEASTLOAD—Least Loaded thread-launch policy. • MPC_LAUNCH_POLICY_NONE—None (the default thread-launch policy).

pthread Library SCA Support

HP extensions to the pthread library provide SCA support for binding threads to locality domains, for specifying the launch policies for threads, and for inquiring about system topology details.

pthread Locality Domain Binding and Inquiry Support

The pthread_ldom_bind_np() routine provides a way to bind a thread to a specified locality domain. You also can retrieve the current locality assignment and can break any locality assignment using this routine.

For pthread programs, using pthread routines for SCA functionality is recommended rather than using the mpctl() system call.

Once a thread is bound to a locality domain, HP-UX will not migrate it across locality boundaries when performing system load balancing activities.

Syntax

```
#include <pthread.h>

int pthread_ldom_bind_np(pthread_ldom_t *answer,

    pthread_ldom_t ldom, pthread_t tid);
```

Description

The pthread_ldom_bind_np() routine binds the thread specified by `tid` to the locality identified by `ldom`. You also can unbind or inquire about a thread's locality binding. The result of the call is returned in `answer`.

On successfully completing, pthread_ldom_bind_np() returns 0. Otherwise an error number is returned; see the *pthread_ldom_bind_np*(3T) manpage for details.

For `tid`, you can specify the value `PTHREAD_SELFTID_NP` to refer to the calling thread.

The following `ldom` values allow inquiring about or disabling locality domain bindings.

- `PTHREAD_LDOMNOCHANGE_NP`—When this value is passed as the `ldom` argument, the current locality domain assignment is read only, and is returned in `answer`.
- `PTHREAD_LDOMFLOAT_NP`—When this is passed as `ldom`, any specific locality domain binding is broken, allowing the thread `tid` to run in any locality that HP-UX chooses.

pthread SCA Launch Policy Support

The pthread interface for specifying a thread-launch policy is the pthread_launch_policy_np() routine. Using this interface you can set any of the SCA launch policies to be in effect, or can retrieve a thread's launch policy.

See the section "Using SCA Launch Policies" on page 105 for more details on thread-launch and process-launch policies.

Syntax

```
#include <pthread.h>

int pthread_launch_policy_np(int request, int *answer,

    pthread_t tid);
```

Description

The pthread_launch_policy_np() routine sets or retrieves the thread-launch policy for the thread specified by `tid`. If retrieving the current launch policy, `answer` stores the result as described below, otherwise `answer` is ignored.

On successfully completing, pthread_launch_policy_np() returns 0. Otherwise an error number is returned; see the *pthread_launch_policy_np*(3T) manpage for details.

For `tid`, you can specify the value `PTHREAD_SELFTID_NP` to refer to the calling thread.

Setting Thread-Launch Policies

The following `request` values are supported for specifying the thread-launch policy.

- None (`PTHREAD_POLICY_NONE_NP`)— The system tries to launch threads in a manner that provides the best performance, though applications should not rely on this providing any specific launch behavior.

  This is the default policy for launching threads and processes.

- Round Robin (`PTHREAD_POLICY_RR_NP`)—Alternate among all localities until all localities have been selected once, then start over as needed.
- Fill First (`PTHREAD_POLICY_FILL_NP`)—Fill a locality first, then spill over to another locality, as needed. Once all localities are filled start over as needed.
- Packed (`PTHREAD_POLICY_PACKED_NP`)—Place all threads in the same locality; do not spill over.
- Least Loaded (`PTHREAD_POLICY_LEASTLOAD_NP`)—Place each thread in the locality that is least-loaded at the time of the thread's creation.

Retrieving Thread-Launch Policies

The pthread_launch_policy_np() routine supports retrieving the current launch policy for the thread `tid` by specifying a `request` value of `PTHREAD_GET_POLICY_NP`. This returns the thread-launch policy by setting `answer` to one of the following values.

- `PTHREAD_POLICY_NONE_NP`—None (the default launch policy).
- `PTHREAD_POLICY_RR_NP`—Round Robin thread-launch policy.
- `PTHREAD_POLICY_FILL_NP`—Fill First thread-launch policy.
- `PTHREAD_POLICY_PACKED_NP`—Packed thread-launch policy.
- `PTHREAD_POLICY_LEASTLOAD_NP`—Least Loaded thread-launch policy.

To continue working on future releases, applications that use `PTHREAD_GET_POLICY_NP` should be written to handle other return values.

pthread System Hardware Inquiry Support

Several pthread library routines support getting details about the current system hardware topology.

The pthread_num_ldoms_np(), pthread_ldom_id_np(), pthread_num_ldomprocs_np(), and pthread_spu_to_ldom_np() functions provide ways to retrieve information about locality domains on the system.

These pthread routines are covered in the following sections.

Syntax

```
#include <pthread.h>

int pthread_num_ldoms_np(void);
```

Description

The pthread_num_ldoms_np() routine returns the number of locality domains currently installed on the system.

Syntax

```
#include <pthread.h>

int pthread_ldom_id_np(int request,

    pthread_ldom_t *answer, pthread_ldom_t ldom);
```

Description

The pthread_ldom_id_np() routine obtains the locality domain ID of a specific locality domain on the system, and returns the locality domain ID in `answer`.

The `request` parameter determines the precise action taken by the routine. The `request` argument may be either of the following.

- `PTHREAD_GETFIRSTLDOM_NP`—Stores the locality ID of the first locality in the system in `answer`. The `ldom` argument is ignored.
- `PTHREAD_GETNEXTLDOM_NP`—Stores in `answer` the locality ID for the next locality domain in the system after `ldom`.

Syntax

```
#include <pthread.h>

int pthread_num_ldomprocs_np(int *answer,

    pthread_ldom_t ldom);
```

Description

The pthread_num_ldomprocs_np() routine determines the number of processors currently installed in the locality domain specified by `ldom`. Returns the number of CPUs in `answer`.

Syntax

```
#include <pthread.h>

int pthread_spu_to_ldom_np(pthread_spu_t spu,

    pthread_ldom_t *ldom);
```

Description

The pthread_spu_to_ldom_np() routine returns the ID of the locality domain containing the processor specified by `spu`. The locality domain ID is returned in `ldom`.

6 Configuring HP-UX Kernel Parameters

This chapter includes suggestions for setting HP-UX kernel parameters for various uses of HP V2500 SCA servers.

You do not need to reconfigure kernel parameter settings after installing HP-UX 11.10. Optional parameter settings are suggested in the sections that follow. By default, HP-UX 11.10 is tuned for V2500 SCA servers when installed.

Related Information

This section lists other sources for information on the topics covered in this chapter.

HP-UX Manpages

These HP-UX 11.10 manpages provide details on this chapter's topics.

- *kmupdate*(1M) — Update the default HP-UX kernel file and files associated with the kernel, or update specified kernel modules.
- *mk_kernel*(1M) — Build a bootable HP-UX kernel and/or kernel modules.
- *sam*(1M) — The menu-driven System Administration Manager program (SAM).

HP-UX Files and Utilities

Files listed below are available on disk in the directories shown.

- /usr/sam/lib/kc/tuned — Directory containing tuned parameter set files used by SAM.
- `/usr/lbin/sysadmin/system_prep` — Utility script to extract a system file from /stand/vmunix or other kernel specified by `-k`.

Other Books

These publications provide details on topics addressed in this chapter.

- *HP-UX Tuning and Performance* — Describes concepts, tools, and methods for tuning HP-UX.
- SAM Online Help — Online help available from the **Help** menu.
- *Managing Systems and Workgroups* — Covers HP-UX system administration tasks, including kernel configuration.

Web Sites

Additional information is available at the following sites on the Web.

- `http://docs.hp.com/` — The HP Technical Documentation home page, which provides free online access to many publications.
- `http://docs.hp.com/hpux/os/` — HP-UX documentation, including *HP-UX 11.0 Configurable Kernel Parameters.*

Overview of HP-UX Kernel Parameters

HP-UX configurable kernel parameters specify how system hardware resources are used by HP-UX and the programs and users it supports.

By setting kernel parameters to values appropriate for your type of server use, you can configure HP-UX to make optimal use of your V-Class server.

You can set HP-UX parameters to specific numeric values, or to computed values by specifying formulas. For example, the following are three ways of specifying 8 Mbytes: 8388608, 0x800000, and 8*1024*1024.

Multiple-Cabinet Server Parameter Settings

The following settings are notable initial kernel parameter settings for multiple-cabinet V2500 SCA servers. These settings are configured when installing HP-UX 11.10 on multiple-cabinet V2500 servers.

- `maxuprc`—Maximum number of user processes.

  Initial V2500 SCA value: 256

- `maxusers`—The MAXUSERS value, used in various kernel formulae.

  Initial V2500 SCA value: 256

- `max_thread_proc`—Maximum number of threads allowed in each process.

  Initial V2500 SCA value: 500

- `maxswapchunks`—Maximum number of swap chunks. The size of each swap chunk is defined by the swchunk kernel parameter, which specifies the number of 1 Kbyte blocks. The initial V2500 SCA value depends on swap space configured.

Note that the `maxswapchunks` parameter setting (and the `swchunk` setting) may need to be adjusted based on your system's swap space and crash dump needs.

The kernel parameter settings listed above are appropriate as initial, minimum settings for a two-cabinet V2500 server.

You may need to increase the `maxuprc` and `maxusers` values, depending on your server use configuration. The setting for these parameters should be based on the number of processors installed on your system and the number of user processes you expect to simultaneously execute.

In general, `maxuprc` and `maxusers` should be set to *at least* four times the number of processors. For heavily used systems, these parameters should be increased to even larger values.

Also, in general, it is best to set `maxuprc`, `maxusers`, and `max_thread_proc` to larger values to allow processes and threads to run.

Suggested Parameter Tunings for Workloads

Suggested HP-UX kernel configurations are provided for the following types of V-Class server use.

- Scientific and Technical Use — Servers running NASTRAN, Abaqus, mechanical and electrical design, fluid dynamics, and other workloads. These workloads have very large data sets and long processing times.

  See "Parameter Settings for Technical Workloads" on page 136 for details.

- Dedicated Commercial Workload and Data Processing Use — Servers whose use is restricted for online transaction processing (OLTP), running Oracle, and running other data processing workloads.

  See "Dedicated Commercial Server Configuration" on page 138 for details.

- Mixed Interactive and Data Processing Use — Servers used both for interactive user log-ins and for OLTP/data processing workloads.

  See "Mixed-Use Commercial Server Configuration" on page 139 for details.

Tuned Parameter Sets

Tuned parameter sets provide settings for collections of kernel parameters.

Several different parameter sets are provided for configuring HP-UX for using HP servers in different situations. However, for HP V-Class servers, only the "V-Class Technical Server" tuned parameter set is appropriate for certain workloads. See "Parameter Settings for Technical Workloads" on page 136.

You can access tuned parameter sets through the **Actions** menu of the System Administration Manager (SAM) Configurable Parameters subarea. These sets also are stored as files in the directory /usr/sam/lib/kc/tuned.

Configuration Utilities

You can perform the HP-UX configuration tasks described here by using the System Administration Manager (SAM) utility (`/usr/sbin/sam`) or the `kmtune` utility (`/usr/sbin/kmtune`).

For details on configuring kernel parameters using `kmtune`, see the *kmtune*(1) manpage. This document only covers using SAM.

You can use SAM's Configurable Parameter subarea to perform most of the procedures described in this chapter. To access this subarea perform the following actions: (1) run SAM, (2) enter the Kernel Configuration area by double-clicking its icon, and (3) double-click the Configurable Parameter icon.

Listing Kernel Parameters

System administrators may list HP-UX kernel parameter settings using the SAM utility (`/usr/sbin/sam`) or the `kmtune` utility (`/usr/sbin/kmtune`). A complete list is provided through SAM's Configurable Parameters subarea. Users who do not have root permission should use `kmtune`.

Running `kmtune` with no options lists all kernel parameters and their current values. To list only specified kernel parameters use the `-q` option. For example, `kmtune -q vps_ceiling` lists the the `vps_ceiling` parameter's setting.

See the *kmtune*(1M) manpage for details.

Parameter Settings for Technical Workloads

Scientific and technical workloads have very large data sets and may have long processing times. Examples include NASTRAN, Abaqus, mechanical and electrical design applications, and fluid dynamics applications.

The "V-Class Technical Server" tuned parameter set provides HP-UX kernel parameter settings for running such workloads on HP V-Class servers.

NOTE

The parameter settings suggested here may require adjustments for optimal performance on your systems.

When tuning HP-UX for multiple-cabinet V2500 SCA servers, refer also to "Multiple-Cabinet Server Parameter Settings" on page 133.

The following list includes key kernel parameters for V-Class servers running scientific and technical workloads:

- `hfs_ra_per_disk`
- `hfs_max_ra_blocks`
- `maxdsiz`
- `maxdsiz_64bit`
- `maxssiz`
- `maxssiz_64bit`
- `vps_ceiling`

These parameters and suggested settings are described in the following table, Table 12.

Note that, on HP-UX 11.10 systems and other systems using HP JFS 3.3, the `vxfs_ra_per_disk` and `vxfs_max_ra_kbytes` parameters are ignored. Use the `/sbin/vxtunefs` utility instead; see the *vxtunefs*(1M) manpage for details.

Table 12 **HP-UX Parameter Settings: Technical Servers**

HP-UX Parameter	Recommended V-Class Technical Setting	Description
`hfs_ra_per_disk`[a]	64	Amount of filesystem read-ahead (in kilobytes) for HFS filesystems.
`hfs_max_ra_blocks`[a]	8	Maximum number of read-ahead blocks the kernel may have outstanding for a single HFS filesystem.
`maxdsiz`	1342177280 (0x50000000 hexadecimal)	Maximum size (in bytes) of a 32-bit process's static data storage segment.
`maxdsiz_64bit`	17179869184 (0x400000000 hexadecimal)	Maximum size (in bytes) of a 64-bit process's static data storage segment.
`maxssiz`	8388608 (0x800000 hexadecimal)	Maximum size (in bytes) of a 32-bit process's dynamic storage segment, also called the "stack size".
`maxssiz_64bit`	1073741824 (0x40000000 hexadecimal)	Maximum size (in bytes) of a 64-bit process's dynamic storage segment, also called the "stack size".
`vps_ceiling`	64 (0x40 hexadecimal)	Maximum variable page size (in kilobytes) the kernel can select.

a. You should set the hfs_ parameters manually using the SAM utility. They are not part of the "V-Class Technical Server" tuned parameter set

Parameter Settings for Commercial Workloads

Commercial workloads include Oracle, online transaction processing (OLTP) applications, decision support (DSS) applications, and other data processing applications.

You can tune kernel parameter in SAM to configure HP-UX for running commercial workloads on V-Class servers. Two HP-UX configurations are described in the next sections.

NOTE

The parameter settings suggested here may require adjustments for optimal performance on your systems.

When tuning HP-UX for multiple-cabinet V2500 SCA servers, refer also to “Multiple-Cabinet Server Parameter Settings” on page 133.

Dedicated Commercial Server Configuration

The parameter tunings in Table 13 may provide a good initial HP-UX configuration for HP V-Class servers dedicated to running commercial data processing applications. These systems provide limited, if any, interactive user access

Table 13 **HP-UX Parameter Settings: Dedicated Commercial Workloads**

HP-UX Parameter	Setting	Description
`bufpages`	(nproc*3)	Number of buffer pages.
`dbc_max_pct`	2	Maximum dynamic buffer cache size (percent of memory).
`dbc_min_pct`	2	Minimum dynamic buffer cache size (percent of memory).

HP-UX Parameter	Setting	Description
maxuprc	((nproc*9)/10)	Maximum number of user processes.
maxusers	200	The MAXUSERS value; used in various kernel formulae.
nproc	((maxusers*3)+64)	Maximum number of processes.

Mixed-Use Commercial Server Configuration

For HP V-Class servers running data processing applications as well as miscellaneous other applications, the tunings in Table 14 may provide a good initial HP-UX configuration.

These systems may provide interactive log-ins for user access in addition to commercial OLTP/DSS application support.

Table 14 **HP-UX Parameter Settings: Mixed-Use Commercial Workloads**

HP-UX Parameter	Setting	Description
bufpages	0	Number of buffer pages.
dbc_max_pct	20	Maximum dynamic buffer cache size (percent of memory).
dbc_min_pct	5	Minimum dynamic buffer cache size (percent of memory).
maxuprc	((nproc*8)/10)	Maximum number of user processes.
maxusers	400	The MAXUSERS value; used in various kernel formulae.
nproc	(((10*maxusers)/3)+128)	Maximum number of processes.

Creating HP-UX Kernels for V-Class Servers

This section has instructions for using the System Administration Manager (SAM) utility to configure HP-UX for HP V-Class servers.

Follow **Step 1–Step 4** to configure parameters and install the kernel using the SAM utility.

Follow **Step 1–Step 6** to configure HP-UX in SAM and manually install the kernel.

An alternative to using SAM to modify a kernel is using the `kmtune` utility; for `kmtune` information see *kmtune*(1).

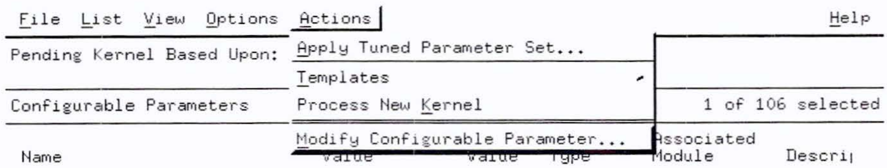
Step 1. Run SAM and enter the Configurable Parameters subarea.

- The SAM executable program is `/usr/sbin/sam`.
- First enter the Kernel Configuration area by double-clicking its icon.

- Then enter the Configurable Parameters subarea.

Step 2. Apply the appropriate kernel parameter settings.

You can modify specific configurable kernel parameters, and can apply a tuned parameter set to modify several parameters at once. You use the SAM **Actions** menu for both applying a parameter set and for modifying individual parameters.



- To apply a tuned parameter set, perform the following tasks.

  Select **Apply Tuned Parameter Set** from the **Actions** menu, and then select a tuned parameter set from the list and click the **OK** button.

  The "V-Class Technical Server" parameter set is for servers that primarily run scientific and technical applications, which may manipulate large amounts of data and require long processing times. See "Parameter Settings for Technical Workloads" on page 136 for more information.

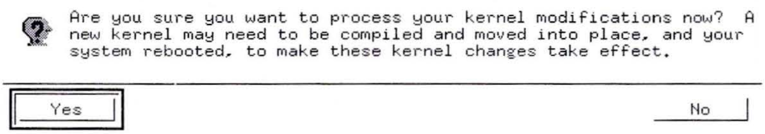
- To modify specific kernel parameter settings, perform the following tasks.

  Select a kernel parameter from the list in the Configurable Parameters subarea, then select **Modify Configurable Parameter** from the **Actions** menu. Enter a formula or value for the parameter, and click the **OK** button.

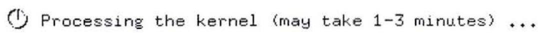
  For suggested kernel parameter settings for various purposes, refer to the following sections: "Parameter Settings for Technical Workloads" on page 136, "Dedicated Commercial Server Configuration" on page 138, and "Mixed-Use Commercial Server Configuration" on page 139.

Step 3. Process the new kernel.

- Select **Process New Kernel** from the **Actions** menu.
- Click the **YES** button in response to the following question.

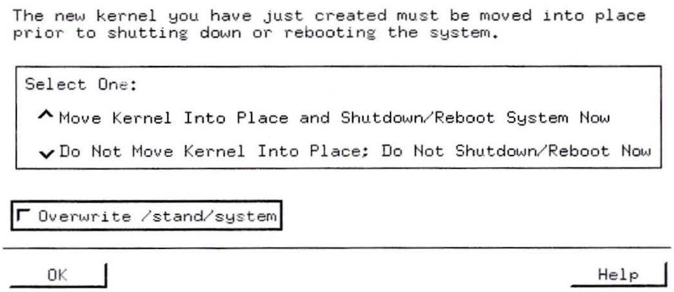


- Wait for SAM to finish processing the new kernel.



Step 4. Select a method for installing the new kernel.

After SAM processes the new kernel, you are presented with the following options.



These options are described below.

- To immediately reboot the system, select both “Move Kernel Into Place” and “Overwrite /stand/system” and click the **OK** button.

  This causes SAM to take the following actions:

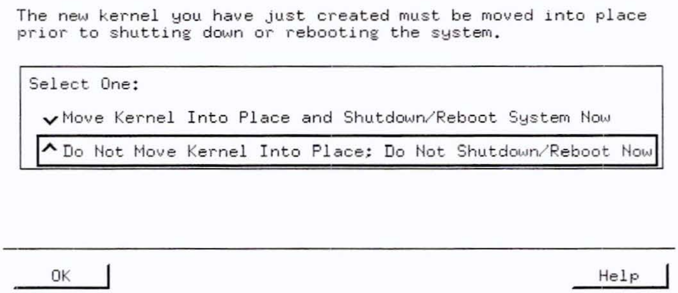
  – Run `kmupdate` to cause the new kernel files to be moved into place when the system reboots.

  – Move the new kernel configuration file into place (if “Overwrite /stand/system” is selected). This is recommended because some utilities expect /stand/system to correspond to the current kernel.

- Reboot the system immediately.

When the system reboots, the new kernel files are moved to the standard locations (/stand/vmunix, /stand/dlkm) and are used to boot the system.

- To not reboot the system, select "Do Not Move Kernel Into Place" and click the **OK** button. This allows you to install the new kernel manually at your discretion.



Selecting "Do Not Move Kernel" causes the new kernel files to be left in the directory /stand/build. This includes the following files:

vmunix_test—The new kernel file.

dlkm.vmunix_test—The directory containing any Dynamically Loadable Kernel Module (DLKM) files.

system.SAM—The new kernel description file.

You should not move these files, except as described in **Step 5** below. The new kernel files are required during the `kmupdate` kernel update process.

Step 5. Manually set the system to use the new kernel when rebooted.

This step is *not needed* if you had SAM move the kernel into place and reboot.

You can schedule the new kernel files to be moved into place by using the `kmupdate` utility and moving the new kernel description file into place.

Use the `kmupdate` utility to schedule the new kernel to be used starting the next time the system boots.

The following example shows the `kmupdate` command and output. For details see the *kmupdate*(1M) manpage.

```
# /usr/sbin/kmupdate /stand/build/vmunix_test
  Kernel update request is scheduled.

  Default kernel /stand/vmunix will be updated by
  newly built kernel /stand/build/vmunix_test
  at next system shutdown or startup time.
```

Move the new kernel description file to /stand/system, as shown in the command line below.

```
# mv /stand/build/system.SAM /stand/system
```

It is highly recommended to move this file because system utilities expect /stand/system to correspond to the current kernel.

Step 6. Set the system to reboot.

This step is *not needed* if you had SAM move the kernel into place and reboot.

Use the `reboot` command to reboot the system.

In the example below, the reboot command sets the system to reboot in ten minutes and print a message. For details see the *reboot*(1M) manpage.

```
# reboot -t +10 -m "Rebooting for kernel update."
Shutdown at 13:52 (in 10 minutes) [pid 1315]

        *** System shutdown message from root@arete-e ***

System going down in 10 minutes
        ... Rebooting for kernel update.
```

When the system reboots, the new kernel files are moved into place and are used to boot the system, as shown by the following output.

```
HP-UX Start-up in progress
    __________________________
    Configure system crash dumps ............................ OK
    Mount file systems ...................................... OK
    Update kernel and loadable modules ......................

Reboot of system has been requested -- rebooting...
```

7 HP SCA Input/Output Management

On multiple-cabinet V2500 SCA servers, input/output (I/O) operates with a few restrictions on certain device locations. This chapter covers those restrictions as well as new hardware path numbering considerations.

Overviews of I/O-related commands, utilities, and processes also are given in this chapter.

Related Information

This section lists other sources for information on the topics covered in this chapter.

HP-UX Manpages

These HP-UX 11.10 manpages provide details on this chapter's topics.

- *ioscan*(1M), *lanscan* (1M)— Scan and list system devices.
- *swapinfo*(1M) — Print device and filesystem paging information.
- *crashconf*(1M) — Print settings and configure system crash dumps.

Files in the Directory /usr/share/doc

Files listed below are available on disk in the directory /usr/share/doc.

- file_sys.txt — *HP-UX 10.0 File System Layout.*
- jfsacl.ps — *Using JFS Access Control Lists.*

Other Books

The following books provide details on topics addressed in this chapter.

- *Managing Systems and Workgroups* — Covers HP-UX system administration tasks, including disk and filesystem management.
- *Configuring HP-UX For Peripherals* — Describes how to configure peripheral devices supported on HP-UX.
- SAM Online Help — Online help that is available from the **Help** menu.
- *VERITAS File System System Administrator's Guide* — Covers HP JFS (VxFS 3.3) administration.
- *Disk and File Management Tasks on HP-UX* — See the `http://www.hp.com/retailbooks/` Web site for details.

Web Sites

Additional information is available at the following sites on the Web.

- `http://docs.hp.com/` — The HP Technical Documentation home page, which provides free online access to many publications.

V2500 SCA I/O Overview

V2500 SCA servers consist of multiple resource localities (*locality domains*) that HP-UX manages and provides as a single, unified system. As HP-UX begins to boot all I/O devices in all localities are made available though the HP-UX operating system.

Before HP-UX begins booting on a V2500 server, each server cabinet is aware of only the I/O devices directly connected to it.

Some restrictions and differences exist in how I/O is managed and made available on multiple-cabinet V2500 SCA servers.

- For booting HP-UX on a V2500 SCA server, only the I/O devices connected to cabinet ID 0 are available to be selected as boot devices.

  If you have an interface to another of the server's cabinets, such as cabinet ID 2, cabinet ID 4, or cabinet ID 6, you can list those cabinet's I/O devices using the Boot Console Handler (BCH) menu. However, only devices on cabinet ID 0 can be used to boot a V2500 server.

- Once the HP-UX boot process begins on a multiple-cabinet V2500 SCA server, all I/O devices are known throughout the server. Boot devices must reside on cabinet ID 0, but the network interface lan0 may reside on any cabinet.

- Logical Volume Manager (LVM) volume groups can be mirrored on multiple-cabinet HP V2500 systems, but both the original and mirrored volume groups must reside on the same cabinet.

- Although volume groups can span across multiple cabinets, performance benefits of this depend on the I/O bandwidth of your application. The boot device, vg00, and any crash dump volume groups must reside entirely on cabinet ID 0.

- HP-UX filesystem buffer cache is configured on V2500 SCA systems as it is on other HP-UX servers. However, on V2500 SCA systems you should consider how much memory is used as buffer cache, and which localities (V2500 cabinets) the memory is taken from. See Chapter 8 for details on HP-UX buffer cache memory use.

Bootable Devices on V2500 SCA Servers

Multiple-cabinet V2500 SCA servers can boot hard disk and CD-ROM devices connected to cabinet ID 0, the monarch cabinet. Booting is supported through the Boot Console Handler (BCH) menu, which is available after system power-on and before HP-UX booting begins.

For the HP-UX 11.10 release, only SCSI devices are supported for booting.

Any disk or volume group with HP-UX 11.0 installed that resides entirely on the monarch cabinet can be selected as the boot disk. A BCH menu command, `SEARCH` finds and prints bootable devices on V2500 servers.

When installing HP-UX 11.10, you can boot the operating system from the CD-ROM media. This is accomplished by selecting the CD-ROM device path on cabinet ID 0 after the compact disc is loaded. The CD-ROM device typically is located at path `4/2/0.0.0` if on V2500 cabinet ID 0.

Forwarding of I/O Requests across SCA Localities

HP V2500 SCA servers can include up to four "resource localities" where I/O devices and other resources may reside. Each locality corresponds to one of the V2500 cabinets that makes up the server.

HP-UX 11.10 provides *I/O forwarding* functionality that enables remote I/O requests made in one locality to be forwarded to another locality where the requested I/O device resides. This happens when a program running in one locality accesses a device in another locality.

For example, a program running on cabinet ID 2 can write to a disk located on cabinet ID 0 without causing the program to be relocated to the locality where the disk resides. This is a performance benefit as it allows the program to remain running in the same location.

In specific rare cases when a program opens or closes a device, a brief temporary program migration may occur as needed to complete the I/O request, rather than I/O forwarding. Ordinarily this type of device access does not occur within user applications.

Hardware Paths on V2500 SCA Servers

Hardware paths provide a unique way for HP-UX to refer to hardware components, such as CPUs, memory, and I/O devices, and indicate the location of those components within a system.

You can use the `/usr/sbin/ioscan` utility to display hardware paths; see the *ioscan*(1M) manpage for details and options.

The hardware path notation for system hardware devices uses the same format on HP V2500 SCA servers as on other HP 9000 hardware architectures.

The first field of the hardware path indicates where a component is located within the server. See Table 15 on page 152. The discussion following Table 15 describes how to determine a CPU's precise location. See "Determining the Physical Location of V2500 SCA I/O" on page 167 to precisely find where I/O devices reside.

Most I/O disk devices have hardware paths in the format *a/b/c.d.e*, as discussed in "General I/O Device Hardware Path Format" on page 153.

The hardware paths for CPUs and memory consist of just one digit, *a*, which indicates the component's location.

Multiple-Cabinet Hardware Paths

On multiple-cabinet V2500 SCA servers, the first field of the hardware path indicates the cabinet in which a hardware component resides. The top two bits of the hardware path match the middle two bits of the V2500 server cabinet ID. (See Chapter 1, Figure 13 for cabinet ID details.)

Figure 27 **Hardware Path Bits and Corresponding Cabinets**

Path Bits	V2500 Cabinet and Fields
`00`*xx xxxx*	Cabinet 0—Fields 0/ to 63/
`01`*xx xxxx*	Cabinet 2—Fields 64/ to 127/
`10`*xx xxxx*	Cabinet 4—Fields 128/ to 191/
`11`*xx xxxx*	Cabinet 6—Fields 192/ to 256/

As Figure 27 shows, the hardware components on each cabinet of a V2500 server are numerically grouped together. This grouping is apparent in listings produced by the `/usr/sbin/ioscan` utility.

Table 15 lists the numbers assigned for the first field of HP-UX hardware paths on V2500 SCA cabinet IDs 0, 2, 4, and 6.

Table 15 **Hardware Path Numbering for V2500 Cabinets**

V2500 Cabinet ID	First Field of Hardware Path	Description of Hardware Component
Cabinet ID 0	0–7	PCI I/O bus bridges (card cages).
	8	Memory.
	15	Core utilities board.
	16–47	Processors (PA-RISC CPUs).
Cabinet ID 2	64–71	PCI I/O bus bridges (card cages).
	72	Memory.
	79	Core utilities board.
	80–111	Processors (PA-RISC CPUs).
Cabinet ID 4	128–135	PCI I/O bus bridges (card cages).
	136	Memory.
	143	Core utilities board.
	144–175	Processors (PA-RISC CPUs).
Cabinet ID 6	192–199	PCI I/O bus bridges (card cages).
	200	Memory.
	207	Core utilities board.
	208–239	Processors (PA-RISC CPUs).

To precisely locate a CPU, correlate its hardware path with its POST identifier. Subtract the hardware path from the base processor hardware path (16, 80, 144, or 208) for the cabinet. This is equivalent to the CPU's OBP number; the corresponding POST identifier specifies its location.

For example, the CPU that has a hardware path of `18` is reported by OBP as CPU `2` (18-16) on cabinet ID 0. POST indicates this CPU is `PB1R_A` (on board "PB1R" in position "A"). See Table 16 on page 158 for details.

General I/O Device Hardware Path Format

The following format is the general hardware path description for I/O devices:

a/b/c.d.e

where:

a — Is the major "hardware module". This first component, *a*, indicates the location and V2500 SCA cabinet on which the device resides. See Table 15 on page 152.

The hardware module includes items such as the PCI I/O bus bridge (card cage) where a device is located, and other primary hardware components, including CPU, memory, and the core utilities board ("core I/O").

b — Is the card slot where the component is located.

This includes PCI I/O cards such as SCSI adapters, lan cards such as FDDI adapters, as well as the built-in serial and Ethernet utilities board components.

c — Is the function (normally `0`) of the card.

For single-function cards such as most SCSI and network cards, this value will always be `0`. Fibre Channel adapters can have varying, non-`0` function identifiers.

d — Is the target (such as SCSI ID) for the hardware component. The `IO` Boot Console Handler (BCH) command reports targets above 9 using letters (a=10, b=11, and so on).

e — Is the logical unit number (LUN) for the device.

This normally is `0` for disks. A multiple-tape device or disk array, for example, could support multiple LUN values.

For example, a disk on cabinet ID 0 could have a hardware path of:

```
1/2/0.9.0
```

The above disk is on cabinet ID 0, is connected to I/O card cage 1, slot 2, and has a target (SCSI ID) of 9.

Fibre Channel I/O Device Hardware Path Format

The hardware path for a Fibre Channel I/O device has more fields than the paths for an individual disk device.

The hardware path format for Fibre Channel I/O devices is:

a/b/c.d.e.f.g.h.i

where:

a/b/ — Is the I/O card cage (*a*) and the card slot (*b*) where the Fibre Channel device is located.

The first component, *a*, indicates the location and V2500 SCA cabinet on which the device resides. See Table 15 on page 152 for details.

c.d.e — Is the function (*c*), target (*d*), and logical unit number (*e*) for the Fibre Channel device.

A target value of `5` indicates a network device. A target value of `8` indicates a disk device.

f.g.h.i — Is additional information.

For example, the following example is a hardware path for a Fibre Channel device connected to cabinet ID 0.

```
4/1/0.8.0.1.0.2.0
```

The above Fibre Channel device is a disk connected to cabinet ID 0, I/O card cage 4, card slot 1.

Configuring System Paging (Swap)

The HP-UX operating system uses paging (swap) space to implement its virtual memory scheme, allowing disk space to effectively extend the amount of memory available to applications running on the system.

Note that here and elsewhere the terms *paging* and *swap* are used interchangeably, although HP-UX implements virtual memory by paging rather than swapping.

Swap considerations for HP V2500 SCA servers include the following issues.

- For V2500 SCA swap configuration, a rough guideline is 1 Gbyte of swap for every 8 Gbytes of available physical memory. Note that some installed memory is not available to HP-UX; see Chapter 8 for details.
- Secondary swap devices can reside on any cabinets within a V2500 SCA server, but the primary swap area must be available during system boot and typically is on the same disk as the root file system (on cabinet ID 0).
- On SCA servers the location (locality) of swap space does not significantly affect performance. That is, a process that pages (uses swap space) on a disk residing on a remote locality does not necessarily perform worse than if it were paging to a local disk.
- Also on SCA servers, when a process is entirely paged out of memory, it pages out and back in to the same locality.

You can configure system swap space using the SAM application (`/usr/sbin/sam`) or by editing the /etc/fstab file as described in the *fstab*(4) manpage.

The `/usr/sbin/swapinfo` command prints system paging information, and `/usr/sbin/swapon` allows you to adjust the swap space configuration.

Primary swap space typically is on the same disk as the root file system. Secondary swap space may reside on any device or file system within the server. Network-mounted (NFS) devices must not be used for swap.

When secondary swap is configured to be on a separate device and to be the same amount as the primary swap, interleaved swapping takes place. HP-UX writes to the primary and secondary swap spaces simultaneously, thus increasing swapping speed.

You can enable both devices and file systems as swap space. By enabling a device for paging, the device can be accessed directly without going through the file system during paging activity. This is significantly faster than paging through a file system.

Paging through a file system, while slower that device swap, allows space on the device that is not being used for paging to be used as file system space.

A third type of swap space is pseudo-swap space, which involves using system memory as swap space, rather than a file system or device. Pseudo-swap is enabled or disabled by the `swapmem_on` HP-UX kernel parameter. See "Configuring I/O-Related HP-UX Kernel Parameters" on page 170.

The amount of swap space available on system disk devices is determined by the contents of the file /etc/fstab, and is not affected by HP-UX kernel parameter configurations. See Table 17 on page 170 for configurable kernel parameters related to I/O and paging activities.

For details on configuring system paging/swap devices see the *swapon*(1M), *swapinfo*(1M), and *fstab*(4) manpages. Information also is provided in the book *Managing Systems and Workgroups*.

Using and Configuring Crash Dump

The HP-UX crash dump process allows you to capture selected portions of a system's physical memory (or all of it) to disk if the system crashes. The resulting crash dump image can be useful in determining the cause of the crash.

For V2500 SCA systems, HP-UX 11.10 provides a new crash dump format and requires that crash dump devices, like boot devices, reside entirely on cabinet ID 0 of the system and be SCSI devices.

Early crash dumps are not supported for multiple-disk root volume groups. However, for all systems, crash dump is supported once the HP-UX startup (init) process configures system crash dump.

HP-UX 11.10 SCA Crash Dump Overview

In HP-UX 11.10 the new SCA crash dump format and accompanying utility set are provided for performing crash dumps on multiple-cabinet V2500 SCA servers.

Crash dumps taken on multiple-cabinet V2500 SCA systems require the HP-UX 11.10 crash dump utilities. The HP-UX 11.10 utilities can read both SCA crash dumps and non-SCA system crash dumps.

Single-cabinet V2500 server crash dumps are written in the non-SCA crash dump format, which can be read using HP-UX 11.0 utilities.

The multiple-cabinet SCA crash dump format is used only when a crash dump is taken on a V2500 SCA server that is booted in a multiple-cabinet configuration.

A full crash dump of a V2500 SCA server includes all memory in the HP-UX physical memory table. The CTI cache is not included in crash dumps because it is not part of the HP-UX memory table. See Chapter 8 for details on memory.

A selective crash dump also is an option on V2500 SCA systems, causing only selected portions of memory to be dumped. See “Types of Crash Dump” on page 162.

Hard Physical Addresses (HPAs)

A unique HPA (Hard Physical Address) is assigned to each processor in a V2500 SCA server. During crash dump, the HPAs displayed by HP-UX and firmware components unambiguously identify a processor within the multiple-cabinet server complex.

For example, in the following console 0 output, the HPA indicates which processor initiated the TOC on the system.

```
******* Unexpected TOC. Processor HPA FFFFFFFC'000A0000 *******
```

The above processor's HPA (`0xfffffffc.000a0000`) corresponds to a CPU on cabinet 0. This processor is identified by OBP as processor `2`. The POST firmware identifies this processor as `PB1R_A`, which indicates that the processor resides on processor board "1R" in the board's "A" position.

Note that HP-UX may assign a processor a different CPU ID (as reported by `mpsched -s`) every time HP-UX is booted, but the HP-UX hardware path (as reported by `ioscan -Cprocessor`) remains constant.

See "Hardware Paths on V2500 SCA Servers" on page 151 for hardware path details. Also see "HP-UX Numbering of Locality Domains and Processors" on page 101 of Chapter 5 for HP-UX CPU ID information.

The POST and OBP firmware on each cabinet identify their corresponding processors using the same identifiers, but HPAs are unique for all processors throughout the system.

Table 16 **HP V2500 SCA Processor HPAs and Firmware Identifiers**

POST	OBP	Cabinet 0 HPA	Cabinet 2 HPA	Cabinet 4 HPA	Cabinet 6 HPA
PB0L_A	0	0xfffffffc.00020000	0xfffffffc.20020000	0xfffffffc.40020000	0xfffffffc.60020000
PB0R_A	1	0xfffffffc.00030000	0xfffffffc.20030000	0xfffffffc.40030000	0xfffffffc.60030000
PB1R_A	2	0xfffffffc.000a0000	0xfffffffc.200a0000	0xfffffffc.400a0000	0xfffffffc.600a0000
PB1L_A	3	0xfffffffc.000b0000	0xfffffffc.200b0000	0xfffffffc.400b0000	0xfffffffc.600b0000
PB2L_A	4	0xfffffffc.00120000	0xfffffffc.20120000	0xfffffffc.40120000	0xfffffffc.60120000
PB2R_A	5	0xfffffffc.00130000	0xfffffffc.20130000	0xfffffffc.40130000	0xfffffffc.60130000
PB3R_A	6	0xfffffffc.001a0000	0xfffffffc.201a0000	0xfffffffc.401a0000	0xfffffffc.601a0000

POST	OBP	Cabinet 0 HPA	Cabinet 2 HPA	Cabinet 4 HPA	Cabinet 6 HPA
PB3L_A	7	0xfffffffc.001b0000	0xfffffffc.201b0000	0xfffffffc.401b0000	0xfffffffc.601b0000
PB4L_A	8	0xfffffffc.00220000	0xfffffffc.20220000	0xfffffffc.40220000	0xfffffffc.60220000
PB4R_A	9	0xfffffffc.00230000	0xfffffffc.20230000	0xfffffffc.40230000	0xfffffffc.60230000
PB5R_A	10	0xfffffffc.002a0000	0xfffffffc.202a0000	0xfffffffc.402a0000	0xfffffffc.602a0000
PB5L_A	11	0xfffffffc.002b0000	0xfffffffc.202b0000	0xfffffffc.402b0000	0xfffffffc.602b0000
PB6L_A	12	0xfffffffc.00320000	0xfffffffc.20320000	0xfffffffc.40320000	0xfffffffc.60320000
PB6R_A	13	0xfffffffc.00330000	0xfffffffc.20330000	0xfffffffc.40330000	0xfffffffc.60330000
PB7R_A	14	0xfffffffc.003a0000	0xfffffffc.203a0000	0xfffffffc.403a0000	0xfffffffc.603a0000
PB7L_A	15	0xfffffffc.003b0000	0xfffffffc.203b0000	0xfffffffc.403b0000	0xfffffffc.603b0000
PB0L_B	16	0xfffffffc.00060000	0xfffffffc.20060000	0xfffffffc.40060000	0xfffffffc.60060000
PB0R_B	17	0xfffffffc.00070000	0xfffffffc.20070000	0xfffffffc.40070000	0xfffffffc.60070000
PB1R_B	18	0xfffffffc.000e0000	0xfffffffc.200e0000	0xfffffffc.400e0000	0xfffffffc.600e0000
PB1L_B	19	0xfffffffc.000f0000	0xfffffffc.200f0000	0xfffffffc.400f0000	0xfffffffc.600f0000
PB2L_B	20	0xfffffffc.00160000	0xfffffffc.20160000	0xfffffffc.40160000	0xfffffffc.60160000
PB2R_B	21	0xfffffffc.00170000	0xfffffffc.20170000	0xfffffffc.40170000	0xfffffffc.60170000
PB3R_B	22	0xfffffffc.001e0000	0xfffffffc.201e0000	0xfffffffc.401e0000	0xfffffffc.601e0000
PB3L_B	23	0xfffffffc.001f0000	0xfffffffc.201f0000	0xfffffffc.401f0000	0xfffffffc.601f0000
PB4L_B	24	0xfffffffc.00260000	0xfffffffc.20260000	0xfffffffc.40260000	0xfffffffc.60260000
PB4R_B	25	0xfffffffc.00270000	0xfffffffc.20270000	0xfffffffc.40270000	0xfffffffc.60270000
PB5R_B	26	0xfffffffc.002e0000	0xfffffffc.202e0000	0xfffffffc.402e0000	0xfffffffc.602e0000
PB5L_B	27	0xfffffffc.002f0000	0xfffffffc.202f0000	0xfffffffc.402f0000	0xfffffffc.602f0000
PB6L_B	28	0xfffffffc.00360000	0xfffffffc.20360000	0xfffffffc.40360000	0xfffffffc.60360000
PB6R_B	29	0xfffffffc.00370000	0xfffffffc.20370000	0xfffffffc.40370000	0xfffffffc.60370000
PB7R_B	30	0xfffffffc.003e0000	0xfffffffc.203e0000	0xfffffffc.403e0000	0xfffffffc.603e0000
PB7L_B	31	0xfffffffc.003f0000	0xfffffffc.203f0000	0xfffffffc.403f0000	0xfffffffc.603f0000

Crash Dump Operation on V2500 SCA Servers

In general, the crash dump process operates on multiple-cabinet V2500 SCA systems as it does on other HP-UX servers. The crash dump process is managed by the PDC firmware running on each V2500 cabinet.

To monitor crash dump activity, use the server's cabinet ID 0 console, which displays system status messages.

You can initiate a crash dump on a V2500 SCA system by performing either of the following actions:

- Pressing the TOC button on an HP V2500 server's cabinet. See Chapter 1 for V2500 cabinet details.
- Issuing a level 4 reset command (`do_reset 0 4`) from the server's Service Support Processor workstation. See Chapter 3 for Service Support Processor details.

Both methods cause a Transfer of Control (TOC) that initiates a crash dump. You should issue only one TOC per crash dump, to ensure that a proper TOC takes place and the crash dump properly completes.

On multiple-cabinet V2500 SCA servers, a TOC causes the same behavior as a panic.

When a TOC occurs, the server gives you the option to select the type of crash dump that will be performed. Either a full dump, selective dump, or no dump can be chosen. The following output is presented in the server's cabinet ID 0 console:

```
*** The dump will be a SELECTIVE dump:  1024 of 7128 megabytes.
*** To change this dump type, press any key within 10 seconds.
```

Once an SCA crash dump begins, the system dumps all cabinet 0 memory in the HP-UX physical memory table, with selective options considered as appropriate. Then the cabinet private memory on cabinet IDs 2, 4, and 6 is dumped, per any selective options and as space allows.

If not enough space exists on the crash dump file system, the HP-UX `savecrash` utility saves as much as will fit in the available space. See *savecrash*(1) for details.

The following sample output is for a crash dump of a two-cabinet V2500 SCA server. The following files were saved to a crash dump directory within /var/adm/crash on the V2500 SCA server.

```
# ls
INDEX           image.10.3.gz  image.2.2.gz   image.4.1.gz   image.8.1.gz
image.1.1.gz    image.10.4.gz  image.2.3.gz   image.5.1.gz   image.9.1.gz
image.10.1.gz   image.11.1.gz  image.2.4.gz   image.6.1.gz   vmunix.gz
image.10.2.gz   image.2.1.gz   image.3.1.gz   image.7.1.gz
```

The contents of the INDEX file for this crash dump is shown below. The header information indicates that the system (hewey) is a V2500 model HP 9000 server. The “numnodes 2” line indicates that hewey is a two-cabinet server. The “monarchnode 0” line shows that cabinet 0 is the monarch cabinet, as is always the case for HP V2500 SCA servers.

```
comment   savecrash crash dump INDEX file
version   4
hostname  hewey
modelname 9000/800/V2500
panic       , isr.ior = 0'10340083.0'a4a93108
dumptime  938745456 Thu Sep  30 20:37:36 MDT 1999
savetime  938746557 Thu Sep  30 20:55:57 MDT 1999
release   @(#) $Revision: vmunix:    vw: ph_ic11_i80    selectors:  'BE11.10_IC11'  Mon Jul 26 12:45:15
PDT 1999 $
memsize   6354370560
chunksize 134217728
numnodes  2
monarchnode 0
module    /stand/vmunix vmunix 19009240 822749276
node      image.1.1 0x00000000 0x0000000000000000 0x0000000000007000 0x0000000000000000
0x0000000000004bff 1874117153
node      image.2.1 0x00000000 0x0000000000000000 0x000000007ffe000 0x0000000000005000
0x00000000000109ff 2420961175
node      image.2.2 0x00000000 0x0000000000000000 0x000000007ff9000 0x0000000000010a00
0x00000000000189f7 1879931149
node      image.2.3 0x00000000 0x0000000000000000 0x000000007ff9000 0x00000000000189f8
0x00000000000209ef 2245309789
node      image.2.4 0x00000000 0x0000000000000000 0x0000000005088000 0x00000000000209f0
0x000000000003ffff 2913215836
node      image.3.1 0x00000000 0x0000000000000000 0x0000000000001000 0x0000000000005000
0x0000000000007fff 3018728591
image     image.4.1 0x0000000000000000 0x000000001ead000 0x0000000000080000 0x00000000000bffff
878292939
image     image.5.1 0x0000000000000000 0x0000000000008000 0x0000000000200000 0x000000000023ffff
2532515601
image     image.6.1 0x0000000000000000 0x0000000000007000 0x0000000001008000 0x000000000103ffff
525066776
image     image.7.1 0x0000000000000000 0x000000000327a000 0x0000000001080000 0x00000000010bffff
1461808238
image     image.8.1 0x0000000000000000 0x0000000000008000 0x0000000001200000 0x000000000123ffff
2532515601
node      image.9.1 0x00000002 0x0000000000000000 0x0000000000007000 0x0000000000000000
0x0000000000004bff 711793027
node      image.10.1 0x00000002 0x0000000000000000 0x000000007ffe000 0x0000000000005000
0x00000000000109ff 3953187251
node      image.10.2 0x00000002 0x0000000000000000 0x000000007ff9000 0x0000000000010a00
0x00000000000189f7 1879931149
node      image.10.3 0x00000002 0x0000000000000000 0x000000007ff9000 0x00000000000189f8
0x00000000000209ef 2245309789
node      image.10.4 0x00000002 0x0000000000000000 0x0000000005088000 0x00000000000209f0
0x000000000003ffff 2913215836
node      image.11.1 0x00000002 0x0000000000000000 0x0000000000001000 0x0000000000005000
0x0000000000007fff 3018728591
```

The Service Support Processor workstation records system crash dump, reset, and other events in the event_log file for the server complex. For example, the following event_log entry is for the sample crash dump of `hewey`, a two-cabinet V2500 server.

```
+++>
<Thu Sep 30 20:37:26 1999>  info:0x454a3003
do_reset:2.0.0.3:../do_reset.c:305
complex: usr3852003  (hewey)    node: 0
Transfer of Control sent to node 0.

****
```

This file is saved as /spp/data/*complex*/event_log on the Service Support Processor. See Chapter 3 for details on the Service Support Processor and its files and utilities.

Types of Crash Dump

A full crash dump, selective dump, or no dump may be performed following a system crash. If no dump is selected, of course, nothing is saved to disk. The `crashconf -v` command lists the currently selected and unselected classes of memory for dumping.

The crashconf file and `crashconf` command are both available for configuring HP-UX crash dumps. They both are discussed later in "Crash Dump Utilities and Files". See the /etc/rc.config.d/crashconf file or *crashconf*(1M) manpage.

When a selective crash dump is performed, only the most potentially useful portions of memory are dumped. You can optionally customize the lists of memory classes included and excluded from selective dumps.

If a full crash dump is performed, all memory is dumped—except, on V2500 SCA servers, areas containing CTI cache memory.

Crash Dump Device Configuration

On V2500 SCA systems, all crash dump devices must reside entirely on cabinet ID 0. Note that early crash dumps (that is, before HP-UX init configures crash devices) are not supported for systems with multiple-disk root volumes.

To establish which devices are configured for crash dump you can edit the /etc/fstab file, use the SAM utility (`/usr/sbin/sam`), or use the `crashconf` command.

Configuring dump devices using SAM (equivalent to modifying the /stand/system file) requires rebuilding the HP-UX kernel and rebooting the system. See the SAM online help for details.

Modifying the /etc/fstab file and using `crashconf` to reconfigure the set of crash dump devices can be done without rebooting the system. These changes can be implemented immediately while the system remains running. See the *crashconf*(1M) manpage or "Crash Dump Utilities and Files" below. See also the *fstab*(4) manpage.

Note that if /etc/fstab will be relied upon to configure crash dump devices upon every system boot, the /etc/rc.config.d/crashconf file should include the line `CRASHCONF_READ_FSTAB=1`.

Crash Dump Utilities and Files

HP-UX crash dump tools include the following three utilities:

- `/sbin/crashconf` — Configure system crash dumps.
- `/sbin/crashutil` — Manipulate (copy, format) crash dump data.
- `/sbin/savecrash` — Save a crash dump of the operating system.

See the *crashconf*(1M), *crashutil*(1M), and *savecrash*(1M) manpages.

The following files can affect crash dump configuration:

- /etc/fstab — File system entries, can specify crash dump devices. See the *fstab*(4) manpage for the format for crash dump entries.

  Note that if a device is to be used for both swap and crash dump purposes, both a "swap" entry and a "dump" entry must be provided in the system's /etc/fstab file.

- /etc/rc.config.d/crashconf — Crash dump system configuration file.

The file /etc/rc.config.d/crashconf is used to configure crash dump during HP-UX startup. You can edit this file to customize how the crash dump process behaves on your system. For instance, this file can set which pages are included and excluded by default for partial crash dumps, and it can set whether /etc/fstab is scanned on boot for dump devices.

To modify a crash dump configuration while HP-UX is running use the `/sbin/crashconf` command. Specifying `crashconf` *device* adds the specified device to the list of configured crash dump devices.

The `crashconf -a` command causes /etc/fstab to be scanned and all crash dump devices found there to be configured for crash dump use.

The `crashconf` command's `-r` option can also be specified to cause new crash dump settings to replace, rather than add to, any existing crash dump settings.

Using Logical Volume Manager (LVM)

The main logical volume restriction on HP V2500 SCA systems is that the root volume group may not span across V2500 cabinets. Note, also, that a volume group that is selected for booting, configured for crash dump, configured as primary swap space, or serving as vg00 must reside entirely on cabinet ID 0.

On multiple-cabinet servers, volume groups can be mirrored but both the original and mirrored volume groups must be on the same cabinet.

HP-UX volume group and logical volume management tasks are performed using the same utilities and processes as on other HP 9000 servers. HP-UX 11.10 adds SCA-related restrictions to the `lvlnboot`, `vgextend`, `vgscan`, and `vgimport` commands.

In general, except for the restrictions mentioned above, volume groups may reside anywhere (on any cabinets) within a multiple-cabinet V2500 server. Volume groups also can span across cabinets to include volumes that physically reside on more than one cabinet.

The `/usr/sbin/lvlnboot` command prepares a logical volume to be the root, boot, primary swap, or dump volume. This command requires that all physical volumes in the volume group reside on the same cabinet. See *lvlnboot*(1M) for details.

The `/usr/sbin/vgextend` command checks whether the volume group being extended is bootable. If so, the new physical volumes being added are checked to ensure that they will not cause the volume group to span across cabinets. See the *vgextend*(1M) manpage.

The `/usr/sbin/vgscan` and `/usr/sbin/vgimport` commands build /etc/lvmtab entries based on information retrieved from on-disk LVM data structures. These commands check whether the volume groups being imported are bootable. If so, the physical volumes are checked to ensure that they do not span across cabinets. See the *vgscan*(1M) and *vgimport*(1M) manpages.

More details on HP LVM are in *Managing Systems and Workgroups*.

Listing the I/O Configuration

HP V2500 SCA firmware commands and HP-UX 11.10 commands are available for listing V2500 SCA server I/O configurations, both before and after the server boots HP-UX.

The following Boot Console Handler (BCH) menu commands allow you to list server I/O information before HP-UX has booted.

- `SEARCH` — Searches for and lists bootable I/O devices.
- `IO` — Lists the I/O devices connected to the cabinet. Note that devices residing on other cabinets are not listed.

  Also note that BCH commands report targets above 9 using letters (a=10, b=11, and so on). See "General I/O Device Hardware Path Format" on page 153.
- `RC # IO` — By specifying the cabinet ID (#, either 2, 4, or 6), allows listing the I/O devices on cabinets other than the monarch cabinet.
- `FC banner` — Prints the server's firmware banner information, including details about the number of PCI I/O busses available.

The following HP-UX commands provide details about the current HP-UX server I/O configurations.

- `/usr/bin/bdf` — Displays information about all normally mounted file systems and their available free space. See *bdf*(1M).
- `/sbin/crashconf -v` — Displays the current crash dump configuration. See *crashconf*(1M).
- `/usr/sbin/swapinfo` — Prints information about device and file system paging space. See *swapinfo*(1M).
- `/usr/sbin/lanscan` — Displays LAN device configuration and status. See *lanscan*(1M).
- `/sbin/ioscan` — Scans system hardware and usable I/O system devices and lists the results. See *ioscan*(1M). Requires root access.

Additional options and arguments for the above HP-UX commands are listed in their associated manpages.

Determining the Physical Location of V2500 SCA I/O

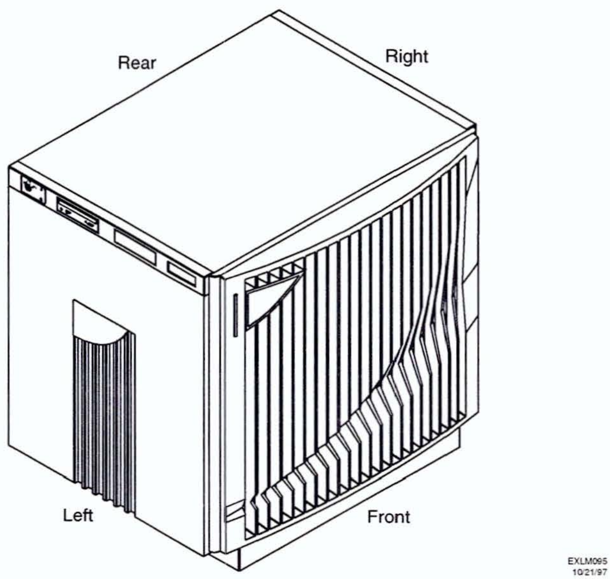
The I/O card cages (PCI bus controllers) in HP V2500 SCA servers are numbered based on which V2500 cabinet they reside in. The first component of the hardware path, such as reported by the HP-UX `ioscan` utility, indicates the cabinet where it resides.

Figure 29 on page 168 shows the PCI bus numbers and card cage locations for a multiple-cabinet server. The I/O card cages are accessible from either the top-left or the bottom-right sides of each V2500 cabinet.

The I/O card cages in a single-cabinet server are numbered from 0 to 7. This numbering also is used for the card cages in cabinet 0 of a multiple-cabinet server.

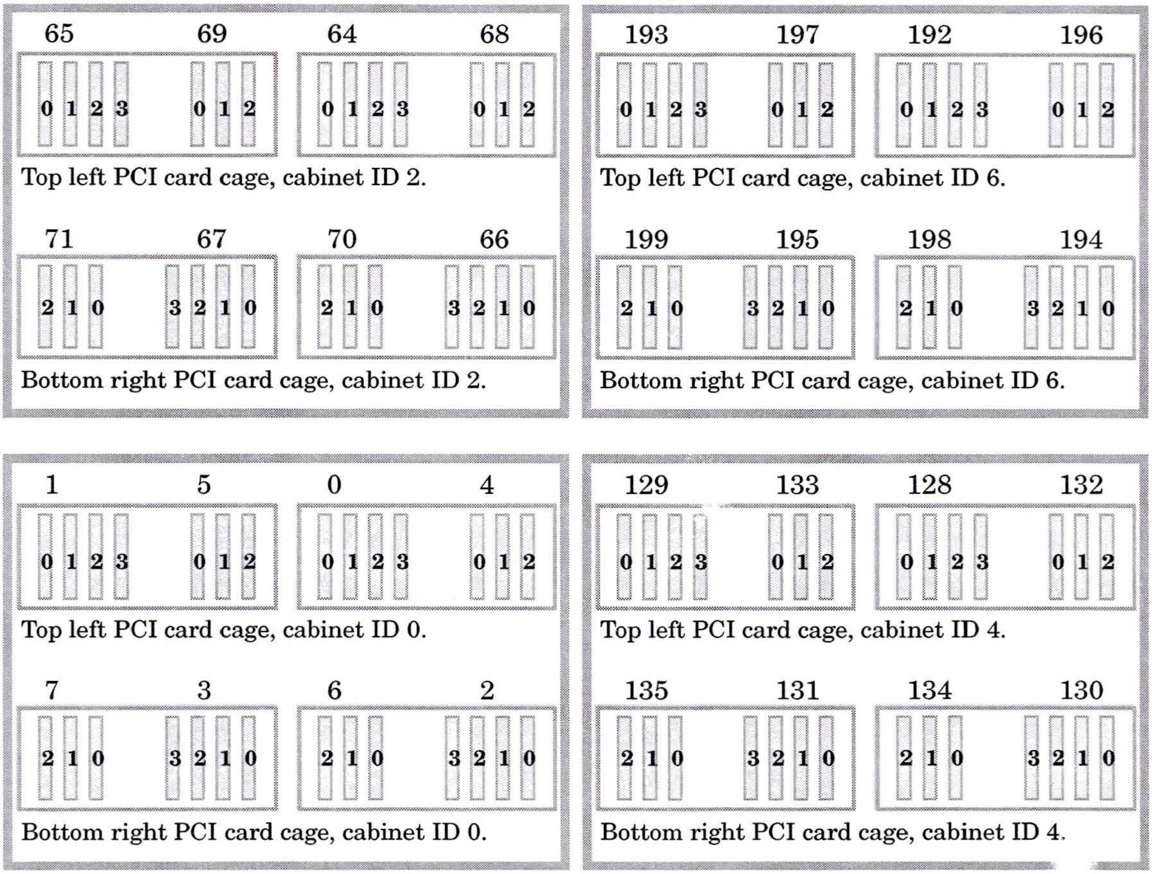
For cabinet ID 2, the PCI bus numbers are from 64 to 71. PCI busses on cabinet ID 4 are from 128 to 135, and cabinet ID 6 devices are numbered from 192 to 199.

Figure 28 **HP V2500 Server Cabinet Sides**



The PCI bus and card slot numbering for multiple-cabinet servers is illustrated in Figure 29 below.

Figure 29 **Numbering and Locations of Multiple-Cabinet V2500 PCI I/O**



I/O card cages are on each cabinet's upper left and lower right sides. Figure 28 on page 167 shows that the front of a V-Class cabinet has a light bar (indicating power is on or off), and the cabinet's left side has the LCD and key switch. Air flow exits the rear of each V-Class cabinet.

A device with a hardware path of

```
193/1/0.2.0
```

resides on cabinet ID 6, card cage 193, slot 1, and has a SCSI ID of 2. For details on multiple-cabinet V-Class server hardware numbering, see "Multiple-Cabinet Hardware Paths" on page 151.

Supported I/O Cards and Devices for V2500 SCA Servers

This section briefly lists the I/O cards and devices supported for HP V2500 SCA servers. Additional cards and devices may also be made available as they are tested and approved for HP V2500 SCA servers running HP-UX 11.10.

For more details on supported cards and devices, refer to the HP-UX 11.10 release notes or to HP sales and support personnel.

The following types of I/O cards are supported by HP-UX 11.10 on V2500 SCA systems.

- Tachyon Fibre Channel
- HVD FWD SCSI
- LVD Ultra2 SCSI
- 10/100 Base T Ethernet
- 1000 Base SX Gigabit Ethernet
- FDDI

The following I/O devices are supported by HP-UX 11.10 on V2500 SCA systems.

- Fibre Channel Mass Storage Peripherals
- HVD SCSI Mass Storage Peripherals
- LVD SCSI Mass Storage Peripherals
- Tape Drives
- Tape Libraries
- DVD and CD-ROM Drives

Configuring I/O-Related HP-UX Kernel Parameters

By adjusting the settings for various configurable HP-UX kernel parameters you can modify HP-UX system I/O capabilities and performance characteristics. Table 17 lists many of the configurable kernel parameters related to I/O performance.

Note that, on HP-UX 11.10 systems and other systems using JFS 3.3, the `vxfs_ra_per_disk` and `vxfs_max_ra_kbytes` parameters are ignored. Use the `/sbin/vxtunefs` utility instead; see the *vxtunefs*(1M) manpage for details.

See Chapter 6, "Configuring HP-UX Kernel Parameters", for details on HP-UX kernel parameter configuration.

Table 17 HP-UX Parameters: I/O-Related Kernel Configuration

HP-UX Parameter Name	Description and Parameter Settings
`allocate_fs_swapmap`	Defines whether the kernel at boot time preallocates sufficient kernel data structures for file system paging (swap). When `allocate_fs_swapmap` is 1, preallocates kernel data structures. Otherwise, allocates as needed.
`bufpages`	Defines the number of 4 Kbyte memory pages in the file system buffer cache. To enable dynamic buffer cache allocation, set `bufpages` to 0. Maximum depends on system memory limits.
`create_fastlinks`	Configures whether the system uses fast HFS symbolic links. When `create_fastlinks` is set to non-zero (1) it causes the system to create HFS symbolic links in a manner that reduces disk accesses for each symbolic link in a pathname lookup.
`dbc_max_pct`	Defines the maximum percentage of memory to be used by dynamic buffer cache. From 2 to 90.

HP-UX Parameter Name	Description and Parameter Settings
`dbc_min_pct`	Defines the minimum percentage of memory to be used by dynamic buffer cache. From 2 to 90.
`fs_async`	Specifies synchronous or asynchronous writes of file system data structures to disk. Setting `fs_asynch` to 1 allows asynchronous disk writes; 0 uses synchronous disk writes only.
`hfs_max_ra_blocks`	Sets the amount of filesystem read-ahead (in kilobytes) for HFS filesystems.
`hfs_ra_per_disk`	Places a ceiling on the maximum number of read-ahead blocks that the kernel may have outstanding for a single HFS filesystem.
`maxswapchunks`	The maximum amount of paging (swap) space that can be configured, system-wide. The total configurable swap space is defined as `maxswapchunks` times `swchunk` times 1024 bytes.
`maxvgs`	Specifies the maximum number of volume groups on the system.
`nbuf`	Determines maximum total number of file system buffers on the system. Set `nbuf` to 0 to enable dynamic buffer cache. Maximum depends on system memory limits.
`nswapdev`	Specifies the number of disk devices that can be enabled for device paging (swap).
`nswapfs`	Specifies the maximum number of file systems that can be enabled for file system paging (swap).
`swapmem_on`	Enables or disables the reservation of pseudo-swap, which is system memory made available for use as paging (swap) space, in addition to swap space on disk. Setting `swapmem_on` to 0 disables pseudo-swap, and 1 enables it.
`swchunk`	The chunk size to be used for paging (swap) purposes. `swchunk` should remain at the default, 2048.

Procedures and Examples

This section has brief I/O-related procedures and examples. See also "Listing the I/O Configuration" on page 166.

Listing Crash Dump Space

The `crashconf -v` HP-UX command lists the configured crash dump settings and crash dump devices. These must be on cabinet ID 0.

```
# /sbin/crashconf -v
CLASS          PAGES  INCLUDED IN DUMP  DESCRIPTION
--------  ----------  ----------------  -------------------------------------
UNUSED       1652283  no,  by default   unused pages
USERPG          8518  no,  by default   user process pages
BCACHE         98386  no,  by default   buffer cache pages
KCODE           2092  no,  by default   kernel code pages
USTACK           305  yes, by default   user process stacks
FSDATA            57  yes, by default   file system metadata
KDDATA         41196  yes, by default   kernel dynamic data
KSDATA        125355  yes, by default   kernel static data

Total pages on system:          1928192
Total pages included in dump:    166913

DEVICE         OFFSET(kB)   SIZE (kB)  LOGICAL VOL.  NAME
------------  ----------  ----------  ------------  -------------------------
 31:0x006000      260960     1048576   64:0x000002  /dev/vg00/lvol2
                          ----------
                             1048576
#
```

Listing Paging (Swap) Space

The swapinfo HP-UX command lists the currently configured swap space and usage statistics.

```
# /usr/sbin/swapinfo
              Kb       Kb       Kb   PCT  START/       Kb
TYPE       AVAIL     USED     FREE  USED   LIMIT RESERVE  PRI  NAME
dev      1048576        0  1048576    0%       0       -    1  /dev/vg00/lvol2
reserve        -    72284   -72284
memory   5611336    70736  5540600    1%
#
```

Listing Available Boot Devices

Before HP-UX boots, use the SEA BCH menu command to list only the bootable devices.

```
[0] Command: SEA
Searching for Devices with Bootable Media.
Sel  Path            Device Type (autoexecute file?)
-------------------------------------------------------------------------------
P0   1/0/0.6.0       Disk   : SEAGATE ST19171W       HP09            (yes)
[0] Command:
```

Only devices on cabinet ID 0 may be booted.

Listing SCSI I/O Devices Before HP-UX Boots

To list I/O devices before the HP-UX operating boots on a V-Class server you can use the IO BCH menu command. The IO command only lists SCSI devices.

On a multiple-cabinet V-Class server, to list I/O devices on a cabinet other than cabinet ID 0, use the RC BCH command to issue the IO command on the remote cabinet. For example, RC 2 IO, as shown in the second example below.

```
[0] Command: IO
4/2/0
Target 0
```

```
  Unit 0   Removable CD-ROM    TOSHIBA CD-ROM XM-5701TA1557

1/0/0
Target 6
  Unit 0   Disk      SEAGATE ST19171W        HP09
Target b
  Unit 0   Disk      SEAGATE ST19171W        HP09

[0] Command: RC 2 IO
69/0/0
Target 2
  Unit 0   Removable CD-ROM    TOSHIBA CD-ROM XM-3301TA0272
Target 6
  Unit 0   Disk      SEAGATE ST19171W        HP01

65/2/0
Target 3
  Unit 0   Disk      SEAGATE ST19171W        HP09
Target 4
  Unit 0   Disk      SEAGATE ST19171W        HP06
Target 5
  Unit 0   Disk      SEAGATE ST19171W        HP06
Target 6
  Unit 0   Disk      SEAGATE ST19171W        HP06

65/0/0

[0] Command:
```

8 HP SCA Memory Management

On multiple-cabinet V-Class servers, the server's memory exists in multiple locations: some physical memory resides on each cabinet within a multiple-cabinet V2500 SCA server.

Although HP-UX manages a single, large pool of memory, there are new considerations when managing memory, including configuring CTI cache and node-local memory.

This chapter covers these and other memory-related issues.

Related Information

This section lists other sources for information on the topics covered in this chapter.

HP-UX Manpages

These HP-UX 11.10 manpages provide details on this chapter's topics.

- *vmstat*(1) — Reports statistics on process, virtual memory, CPU, and other activity.
- *mmap*(2) — Map pages of memory.
- *shmget*(2) — Get shared memory segment.
- *swapinfo*(1M) — Print device and filesystem paging information.
- *mpsched*(1) — Control the processor or locality domain on which a specific process executes.

Files in the Directory /usr/share/doc

File listed below is available on disk in the directory /usr/share/doc.

- mem_mgt.txt — *HP-UX Memory Management* white paper.

Other Books

These books provide more details on topics addressed in this chapter.

- *Managing Systems and Workgroups* — Covers HP-UX system administration tasks.

Web Sites

Additional information is available at the following sites on the Web.

- `http://docs.hp.com/` — The HP Technical Documentation home page, which provides free online access to many publications.

Overview of V2500 SCA Memory

The following issues relate to memory use, performance, and management on multiple-cabinet V2500 SCA servers.

- All V2500 cabinets in an SCA system must have the same memory configuration. Supported configurations for SCA servers include one-fourth, one-half, or full population of memory boards. Single-cabinet systems also can be populated to three-fourths memory capacity.

- In a multiple-cabinet V2500 SCA server, each cabinet (or *locality*) contains memory that is available throughout the entire server.

  A processor takes longer to access memory from a remote locality than from memory that resides in the same locality.

- Some physical memory from each locality (cabinet) is dedicated as CTI cache memory and as node-local memory. The CTI cache and node-local memory are configured when the server boots and are not available to HP-UX applications.

  The node-local memory stores firmware and replicated kernel text. CTI cache memory encaches data fetched and prefetched from remote memory. The remaining memory is available to HP-UX and programs.

- Programs that use archive libraries and non-shared text perform better in SCA environments than those using shared libraries and text.

  Use the `-N` linker option for non-shared text, or the `-a archive` linker option to link using archive libraries.

- Memory errors, like other V2500 hardware errors, may be logged in the following files in the /spp/data/*complex* directory on the Service Support Processor: event_log, hard_hist, and the complex's consolelog files (consolelog0, consolelog2, and so on).

  If a memory error results in a memory DIMM being deconfigured, the POST firmware deallocates the corresponding DIMM on all cabinets. It then reconfigures all cabinets to the largest possible supported memory configuration (one-fourth or one-half full, for SCA systems).

- HP-UX data structures can occupy 20% or more of cabinet ID 0 memory.
- The mmap() and shmget() routines include new flags for targeting local memory localities. See Chapter 5 for details.
- If file system buffer cache is configured for HP-UX running on a multiple-cabinet V2500 SCA server, the system memory use can differ significantly for static and dynamic buffer caches. See "Using Filesystem Buffer Cache Memory" on page 186.
- When an application runs on a processor in a locality, if possible HP-UX allocates memory from the same locality for the application's use. However, HP-UX allocates memory from a remote locality if memory from the same locality is not available.
- Primary swap space must be on cabinet ID 0. For more detail about paging (swap) configuration see Chapter 7.

More details on these issues are provided in the sections that follow.

Memory Localities on V2500 SCA Servers

On a multiple-cabinet V-Class server, there is both local memory and *remote memory*. When a processor accesses memory that physically resides on a remote cabinet (remote memory), it takes several hundred nanoseconds longer than accessing memory on the same cabinet as the processor (local memory).

Local memory accesses take approximately 502 nanoseconds, while accessing remote memory takes 1550 nanoseconds or longer.

The additional time to access remote memory is due to the latency involved when the memory request travels across both local and remote hardware and back again to the requesting processor. (The requesting processor is the CPU running the program using the memory.)

You can enhance memory performance on multiple-cabinet servers by dedicating part of each cabinet's memory as CTI cache memory for storing local copies of remote memory. Details are provided in the section "Using CTI Cache Memory" on page 187.

Listing Memory Localities

You can list a V2500 SCA server's memory localities using the `ioscan` or the `mpsched` HP-UX commands. `mpsched` is new in HP-UX 11.10.

The `ioscan -Cmemory` option lists the hardware paths for all physical memory on the system. For example, the following command lists memory on a two-cabinet V2500 SCA server.

```
# ioscan -Cmemory
H/W Path  Class   Description
==============================
8         memory  Memory
72        memory  Memory
```

The `mpsched -s` option lists the system's hardware topology, including the number of localities and processors in the localities. The following example lists details for a two-locality, 16-processor V2500 SCA server.

```
# mpsched -s
System Configuration:
=====================
Locality Domain Count: 2
Processor Count: 16
Domain    Processors
------    ----------
0         0   1   2   3   4   5   6   7
1         8   9  10  11  12  13  14  15
```

Types of Memory on V2500 SCA Servers

Not all installed physical memory is available to HP-UX or programs running on V2500 SCA servers. Some memory is dedicated as CTI cache and node-local memory.

Physical memory on V2500 SCA servers includes the following categories:

- **CTI cache memory**—Used to encache data fetched from remote localities. One-eighth or more of each cabinet's physical memory may be dedicated as CTI cache memory. See "Using CTI Cache Memory" on page 187 for details.
- **node-local memory**—Used by the firmware on each cabinet, and for replicating HP-UX kernel text across cabinets. You must allocate 128 Mbytes per cabinet as node-local memory. See "Configuring CTI Cache and Node-Local Memory" on page 188.
- **HP-UX memory**—The remaining physical memory on all cabinets is available to the HP-UX operating system and user programs.

HP-UX data structures occupy memory in cabinet ID 0 that is equivalent to 5–7% of all available physical memory. On a four-cabinet system, 20–28% of cabinet ID 0 memory is occupied by kernel data structures.

HP-UX filesystem buffer cache can occupy memory in cabinet ID 0, and may use other cabinet's memory as well.

See "Checking the Server Memory Configuration" on page 183. See also "Using Filesystem Buffer Cache Memory" on page 186.

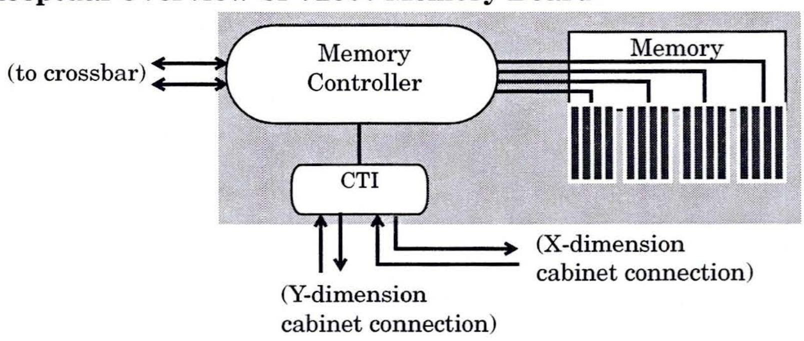
V2500 SCA Memory Architecture Details

All physical memory installed in V2500 SCA servers must only consist of 88-bit DIMMS. Three DIMM sizes are supported for use in V2500 servers: 32 Mbyte, 128 Mbyte, and 256 Mbyte DIMMs.

Through the V2500 server's memory access controllers, each memory board provides separate read and write access to the memory DIMMs. Up to 16 DIMMs may be installed per board, providing up to 256-way memory interleaving per cabinet when all memory boards are fully populated.

Slots for DIMMs on each memory board are conceptually grouped in four quadrants. Each quadrant, as Figure 30 shows, has a separate connection to the memory controller. All quadrants should have the same memory configuration; this also provides good performance by interleaving memory within a memory board.

Figure 30 **Conceptual Overview of V2500 Memory Board**



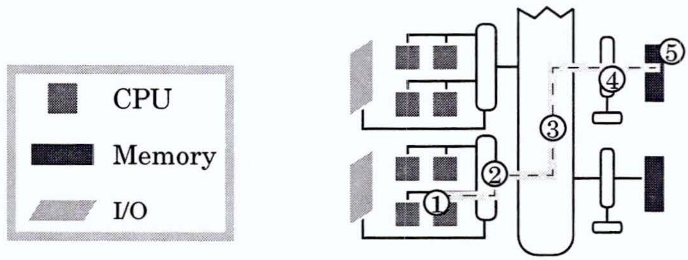
Memory also is interleaved across memory controllers. This allows separate controllers and separate parts of the V2500 crossbar to simultaneously access memory on across all eight memory controllers.

Varieties of V2500 SCA Memory Accesses

This section covers how programs access memory on multiple-cabinet HP V2500 SCA servers. Memory accesses may occur in any of three ways: local accesses (Figure 31), remote accesses (Figure 32), or CTI cache accesses (Figure 33).

Figure 31 shows the steps taken access local memory.

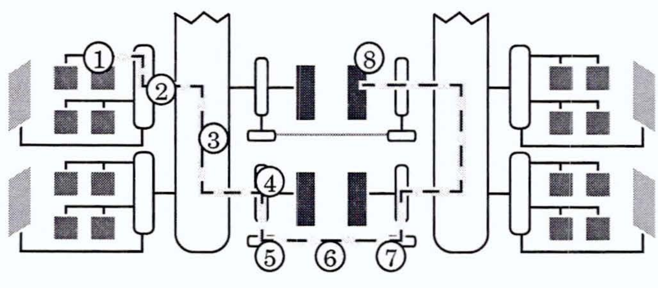
Figure 31 **Memory Access: Data Local to a Cabinet**



In Figure 31, the CPU (1) requests memory not found in its data cache, so the request travels through the processor agent (2) over the crossbar (3) to a memory controller (4) and to the physical memory (5) and then back to the requesting processor.

A remote memory access, as in Figure 32, below, takes longer than a local access.

Figure 32 **Memory Access: Data Remote to a Cabinet**

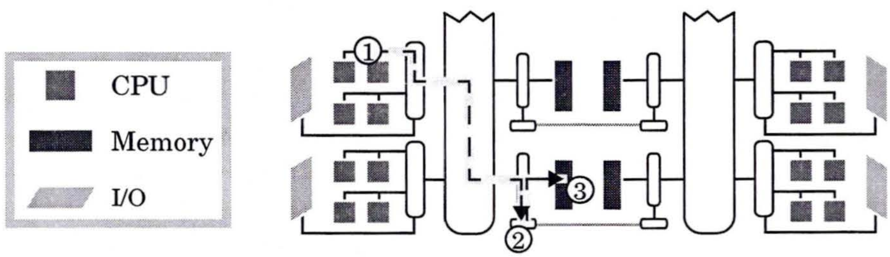


In Figure 32 the processor (1) requests memory not found in its data cache. To read the data from physical memory on a remote cabinet, the request travels to a CTI controller (2–5), across a CTI ring (6) to a remote CTI controller (7) and to the remote memory (8) and back to the requesting processor.

To avoid many remote memory accesses, some of each cabinet's physical memory is used exclusively as CTI cache.

Figure 33 shows a processor using data that is stored (encached) in the CTI cache. The CTI cache encaches both reads and writes to remote memory.

Figure 33 **Memory Access: Remote Data Stored in Local CTI cache**



In Figure 33 the requested memory physically resides on a remote cabinet but has been encached in the local cabinet's CTI cache. The request travels from the CPU (1) to a CTI controller (2) to the portion of the cabinet's physical memory used as CTI cache (3), which holds a local copy of the data.

Checking the Server Memory Configuration

To determine the amount of physical memory available to HP-UX and applications running on a V2500 SCA server, you can make an approximate calculation based on the CTI cache and node-local settings or can use HP-UX utilities to inquire about memory availability.

Note that HP-UX data structures and HP-UX filesystem buffer cache also occupy memory on cabinet ID 0 and possibly other cabinets.

You can observe the server memory configuration during the hardware boot process.

The following sections cover various methods for determining the memory configuration.

Approximately Determining the Available Physical Memory

You can roughly estimate the amount of physical memory available on a multiple-cabinet HP V2500 server by subtracting the total size of the CTI cache and node-local memory from the amount of memory installed.

The following formula gives this rough estimation of available memory:

total installed - CTI cache - "node-local" = available memory

The example below shows a sample calculation of approximately how much memory is available. The following calculation is for a two-cabinet server with 4 Gbytes (4096 Mbytes) of memory installed per cabinet and a 1024 Mbyte CTI cache and 128 Mbyte node-local size per cabinet.

8192 Mbytes - 2048 Mbytes - 256 Mbytes = 5888 Mbytes

Note that the 5888 Mbytes (6,029,312 Kbytes) is an approximate calculation. More physical memory may be available, as some node-local memory may be made available for use by HP-UX and user programs. However, not all this memory is available to user programs; some is occupied by HP-UX data structures and buffer cache.

Inquiring about Memory Availability Using HP-UX Utilities

You can view the current system memory use and availability using several HP-UX utilities. These utilities operate on V2500 SCA servers as they do on other HP hardware platforms.

The following examples show uses of the `dmesg` and `top` commands on a two-cabinet V2500 SCA server with 8 Gbytes (8192 Mbytes) of memory installed. Each cabinet in these examples has 1024 Mbytes CTI cache and 128 Mbyte node-local memory configured.

You can use the SAM (`/usr/sbin/sam`) utility to view the current memory availability. Select the Performance Monitors area, and the System Properties subarea. For the example system described above, SAM reports 6065.4 Mbytes of physical memory.

Using the `/usr/sbin/dmesg` HP-UX command provides a way to check the amount of physical memory available. For the example system, `dmesg` reports 6205440 Kbytes (6060 Mbytes) of physical memory. See the *dmesg*(1M) manpage for details.

```
# /usr/sbin/dmesg | grep -i Physical
    physical page size = 4096 bytes, logical page size = 4096 bytes
    Physical: 6205440 Kbytes, lockable:4444228 Kbytes, available:5109396 Kbytes
```

You also can use the `/usr/bin/top` command to check current system memory use and availability. The following line reports the amount of real and virtual memory, with the amount considered active in parenthesis.

```
Memory: 24084K (5716K) real, 18440K (9504K) virtual, 5017584K free  Page# 1/7
```

Though `top` does not summarize total installed physical memory, it does present current usage details. For details see the *top*(1) manpage.

Observing Memory Configuration during System Booting

As HP V2500 hardware boots after it is powered on, the POST firmware prints information to the cabinet ID 0 console (complex console). POST includes memory configuration details among the information it reports.

You can find the following memory information as part the cabinet 0 console's boot-time output.

- The installed physical memory DIMM configuration.

  During server start-up, the POST firmware reports on each installed memory DIMM using a pair of characters.

```
  PB1R_B MB0L [:::: ____][:::: ____][____ ____][____ ____]
  PB1R_A MB1L [:::: ____][:::: ____][____ ____][____ ____]
  PB4L_A MB2R [:::: ____][:::: ____][____ ____][____ ____]
  PB5R_A MB3R [:::: ____][:::: ____][____ ____][____ ____]
  PB0L_B MB4L [:::: ____][:::: ____][____ ____][____ ____]
  PB0L_A MB5L [:::: ____][:::: ____][____ ____][____ ____]
  PB4L_B MB6R [:::: ____][:::: ____][____ ____][____ ____]
  PB5R_B MB7R [:::: ____][:::: ____][____ ____][____ ____]
 Building main memory map.
Main memory initialization complete.
```

  Each pair of character indicates the amount of memory present: **..**=32 Mbytes, **::**=128 Mbytes, ||=256 Mbytes, __=empty, **##**=hardware deconfigured, and **$$**=software (user) deconfigured.

  For example, POST lists a 256 Mbyte DIMM as ||.

- The allocated use of installed memory.

  The POST firmware allocates HP V2500 SCA physical memory as either *CTI cache* memory or as *local* memory that HP-UX can use.

  Memory marked "L" is local memory, "C" is used for CTI cache, and "M" is mixed-use memory used in part as CTI cache and in part as local memory.

```
  PB4L_A MB0L [LLLL ____][MMMM ____][____ ____][____ ____]
  PB1R_A MB1L [LLLL ____][MMMM ____][____ ____][____ ____]
  PB0L_A MB2R [LLLL ____][MMMM ____][____ ____][____ ____]
  PB5R_A MB3R [LLLL ____][MMMM ____][____ ____][____ ____]
  PB0L_B MB4L [LLLL ____][MMMM ____][____ ____][____ ____]
  PB1R_B MB5L [LLLL ____][MMMM ____][____ ____][____ ____]
  PB4L_B MB6R [LLLL ____][MMMM ____][____ ____][____ ____]
  PB5R_B MB7R [LLLL ____][MMMM ____][____ ____][____ ____]
```

  POST also allocates memory as node-local memory but does not print the node-local allocation during system boot time.

- The total physical memory and CTI cache memory details.

  The boot-time output also includes a memory summary, such as the following text.

```
8192 MB memory installed       2048  MB CTI cache configured (total, all nodes)
```

Using Filesystem Buffer Cache Memory

HP-UX filesystem buffer cache memory can be allocated using memory from cabinet ID 0, or using memory from any cabinet, depending on whether static or dynamic buffer cache is configured for the system.

HP-UX allocates static buffer cache memory at start-up, and on HP V2500 SCA systems it allocates memory from cabinet ID 0.

HP-UX allocates dynamic buffer cache as it is needed. Typically the first allocations are from cabinet ID 0, and other dynamic buffer cache allocations use memory from any of the cabinets in a V2500 SCA system.

NOTE

The maximum HP-UX filesystem buffer cache size on an HP V2500 SCA system should be less than 1/(*number_of_cabinets**2) of the available memory. Otherwise, one of the system's cabinets may have all its memory used as buffer cache.

For example, on a two-cabinet V2500 SCA server the buffer cache should be less than one-fourth the available physical memory. If the server has 16 Gbytes of memory and 4 Gbytes of CTI cache configured, then allowing 1 Gbyte for node-local memory and HP-UX data structure use leaves about 5.5 Gbytes available per cabinet (11 Gbytes total). So, for this example system, a buffer cache size of less than 1.3 Gbytes is appropriate.

If all memory in a V2500 SCA cabinet is used by HP-UX and buffer cache, HP-UX allocates remote memory for processes running on the cabinet's CPUs. Some of the remote memory may also be encached using the local cabinet's CTI cache memory.

HP-UX filesystem buffer cache is configured using the `bufpages` and `nbuf` HP-UX kernel parameters. If dynamic buffer cache is used, the `dbc_min_pct` and `dbc_max_pct` parameters are set.

Using CTI Cache Memory

A portion of the memory on each cabinet in a V2500 SCA server is dedicated for use as CTI cache. The CTI cache is physical memory used for encaching memory requested from remote cabinets.

Single-cabinet servers should not have any memory dedicated as CTI cache, as it is unnecessary.

Once remote memory is encached in CTI cache, local CTI cache memory is used when the memory is subsequently accessed by programs running on the cabinet. Using CTI cache can help applications reduce the time taken to access data stored in remote physical memory.

CTI Cache Operation Details

CTI cache memory is configured at system boot time. The memory used for CTI cache is dedicated to be used only for encaching remote accesses; it is not available for other uses, such as use by HP-UX or applications.

CTI cache encaches data fetches but not instruction fetches.

For efficiency, when a CPU reads data from a remote cabinet's memory, additional data may also be fetched and encached into the local cabinet's CTI cache. This *prefetching* of data saves time should the additional data also be requested. Accessing memory from the CTI cache is faster than accessing it from a remote cabinet.

Prefetching remote memory also can reduce the overall communications traffic in the system, by effectively combining multiple remote memory requests into a single request.

The HP V-Class hardware keeps track of the various copies of data that exist in a system and keeps the copies synchronized as needed. In doing this, HP V2500 SCA servers maintain global cache coherency—both between a CPU's data cache and memory, and between a local cabinet's CTI cache and remote memory.

Selecting CTI Cache Size

On multiple-cabinet V2500 servers you must dedicate a portion of each cabinet's physical memory as CTI cache.

The minimum recommended CTI cache size for multiple-cabinet V2500 servers is 128 Mbytes per cabinet. Typical CTI cache sizes range from 128 Mbytes to 1024 Mbytes per cabinet, depending on the amount of remote memory accesses that typically occur for applications that are run on the server.

The amount of memory dedicated as CTI cache must be set to be the same for each cabinet in the V2500 server. So, for example, a 128 Mbyte per-cabinet CTI cache setting would consume a total of 256 Mbytes of memory for a two-cabinet V2500 server.

Configuring CTI Cache and Node-Local Memory

On V2500 SCA servers, both CTI cache and node-local memory are configured using some of the physical memory on each cabinet.

You can configure the CTI cache and node-local memory on a V2500 SCA server's cabinets by using the `ts_config` Service Support Processor utility.

The `xconfig` Service Support Processor utility also can set each V2500 cabinet's CTI cache, and the Boot Console Handler (BCH) Forth mode interface can configure both CTI cache and node-local memory settings.

NOTE

Use `ts_config` for configuring CTI cache and node-local memory.

Only trained HP support personnel should use the `xconfig` and Forth mode methods for configuring memory, as those methods can result in unsupported and unusable system configurations.

The `/spp/bin/ts_config` utility is available on the server's Service Support Processor. Basic details about the utility are provided in the *ts_config*(1) manpage and in the program's online help.

When you use `ts_config` the utility sets each cabinet's CTI cache and node-local memory size to the same designated setting, ensuring a proper configuration.

To configure CTI cache and node-local memory on a multiple-cabinet server (multinode complex) you must run `ts_config` as the `root` user. Also, HP-UX should not be running on the server because `ts_config` reboots the hardware as part of the configuration.

The following procedure is for configuring server memory.

1 As the `root` user, run the `/spp/bin/ts_config` utility from the Service Support Processor.

   You can log in as the `root` user using the `su` HP-UX command. If directly logged in to the Service Support Processor `sppuser` account, you can run `ts_config` as `root` using the CDE menu.

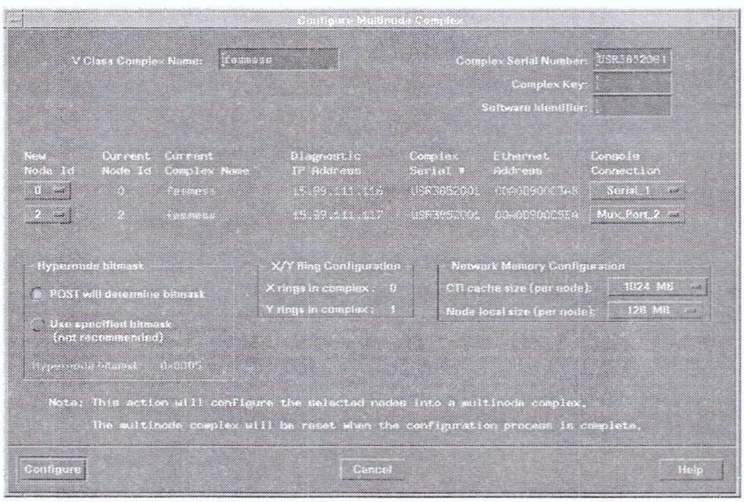
   If remotely accessing the Service Support Processor, you must set the `DISPLAY` environment variable to your local display before running `ts_config`, or use the `ts_config -display` option.

2 Select all the server's cabinets (nodes) from the list of cabinets.

3 Select **Configure Multinode Complex** from the **Actions** menu.

   `ts_config` then displays the window shown in Figure 34.

Figure 34 **The ts_config Multinode Configuration Window**



NOTE

Note that the CTI cache size listed by `ts_config` is not necessarily the current system setting. To check the current setting, see the section "Checking the Current CTI Cache Setting" on page 190.

4 Select the CTI cache size and node-local memory size using the pop-up menus in the "Network Memory Configuration" area.

   `ts_config` uses the settings you specify for every cabinet in the complex.

   The default "Node local memory" size of 128 Mbytes is required. See "Selecting CTI Cache Size" on page 188 for CTI cache recommendations.

5 Click **Configure** to begin the server configuration.

   Following the configuration, `ts_config` reboots the server. The reboot causes all configuration changes to be put into effect.

For more details, see the `ts_config` online help, manpage, or the *Service Support Processor Guide*.

Checking the Current CTI Cache Setting

You can check a server's current CTI cache configuration by monitoring the server's "complex console" boot-time output, which includes (among other information) the CTI cache configuration.

During the server boot and initialization process, the CTI cache is configuration is shown as follows. Each cabinet's CTI cache setting is listed, followed by the total server CTI cache.

See "Printing CTI Cache Settings" on page 192 for more procedures and examples for printing CTI cache settings.

Chapter 2 includes a complete boot output, with memory initializations.

The following boot-time output shows a summary of the system configuration for a multiple-cabinet V2500 server, including CTI cache settings.

```
-------------------------------------------------------------------------------
                        Multi-node Configuration Summary
===============================================================================
NODE 0
     UART?  Yes
     CORE   MAC address 0:a0:d9:0:c3:cf   IP# 15.99.111.166 (0x0f636fa6)
     JTAG   MAC address 0:a0:d9:0:c3:a8   IP# 15.99.111.116 (0x0f636f74)
     MEMORY 4096 MB memory installed      128  MB CTI cache configured
     CPUs   0,2,8,10
            16,18,24,26
     PACs   0,1,2,3,4,5,6,7               MACs   0,1,2,3,4,5,6,7
     TACs   0,1,2,3,4,5,6,7               PCIs   0,1,2,3,4,5,6,7
```

```
---------------------------------------------------------------------------------
NODE 2
      UART?  No
      CORE   MAC address 0:a0:d9:0:c6:51   IP# 15.99.111.167 (0x0f636fa7)
      JTAG   MAC address 0:a0:d9:0:c5:ea   IP# 15.99.111.117 (0x0f636f75)
      MEMORY 4096 MB memory installed      128  MB CTI cache configured
      CPUs   0,2,8,10
             16,18,24,26
      PACs   0,1,2,3,4,5,6,7               MACs   0,1,2,3,4,5,6,7
      TACs   0,1,2,3,4,5,6,7               PCIs   0,1,4,5
---------------------------------------------------------------------------------
8192 MB memory installed       256  MB CTI cache configured (total, all nodes)
```

Procedures and Examples

This section has additional procedures for listing a server's current and future CTI cache settings and simple examples.

Printing CTI Cache Settings

You can use the following two methods to print CTI cache settings. Monitoring a server's complex console output during boot time is described in "Checking the Current CTI Cache Setting" on page 190.

To print the current configuration as of the last time the server booted, retrieve the information from the server's complex console log (the consolelog0 file).

Use `grep` to strip the CTI cache configuration data, and use `tail` to list only the current cabinet and total-server data. (For a two-cabinet server use `tail -3`, for a three-cabinet server use `tail -4`, and so on.)

This results in a list of the individual cabinet CTI cache configurations plus the server's total configuration.

The command line below prints the current (last) CTI cache settings for the server complex "fesmess", a two-cabinet server.

```
$ grep "CTI cache configured" /spp/data/fesmess/consolelog0 | tail -3
      MEMORY 4096 MB memory installed      128  MB CTI cache configured
      MEMORY 4096 MB memory installed      128  MB CTI cache configured
8192 MB memory installed       256  MB CTI cache configured (total, all nodes)
```

Another method of checking CTI cache settings is to use the Boot Console Handler (BCH) menu to print the current CTI cache variable settings.

Note that while this method is a likely indicator of the current CTI cache size, it is not necessarily correct. The settings it prints (for example, 128 Mbytes in the example below) will be used the next time the server boots.

This first Boot Console Handler (BCH) command line below prints cabinet ID 0's CTI cache setting.

```
[0] Command: FC printenv cti-cache-size
Parameter Name:  cti-cache-size
Current Value:
  128 Megabytes
Default Value:
  0 Megabytes
```

The next BCH command line prints cabinet ID 2's CTI cache setting, which also will be used the next time the server boots.

```
[0] Command: RC 2 FC printenv cti-cache-size
Parameter Name:  cti-cache-size
Current Value:
  128 Megabytes
Default Value:
  0 Megabytes
[0] Command:
```

A HP-UX Installation

This appendix covers installing HP-UX 11.10 on V2500 SCA servers, as well as related Service Support Processor and V-Class firmware issues.

For complete details, refer to the *HP-UX 11.10 Installation and Configuration Notes* document, which is distributed with HP V2500 SCA systems and the HP-UX 11.10 CD-ROM media.

Related Information

This section lists other sources for information on the HP-UX installation tasks covered in this document.

Web Sites

Additional information is available at the following sites on the Web.

- `http://docs.hp.com/`

  The HP Technical Documentation home page.

  This site provides online access to HP-UX documentation, HP V2500 server hardware documentation, related development tools manuals, and other information.

- `http://www.software.hp.com/`

  HP's Software Depot Web site, a single source for software from HP.

Release Notes

Additional information is available in the following release notes.

- *Release Notes for HP-UX Release 11.10* — Describes the contents of the HP-UX 11.10 release, including its features and restrictions.

  The release notes are installed with HP-UX 11.10 as the file /usr/share/doc/11.10RelNotes.

- *V2500 Test Station Software Release Notice* — Covers installing the Service Support Processor software environment, including diagnostics, utilities, the `sppuser` login account used for service work, and other details. Also covers procedures for downloading and configuring HP V2500 SCA firmware.

About the HP-UX 11.10 Release

The HP-UX 11.10 release is distributed on three CD-ROMs:

- The *Install and Core OS CD.*

  The Install CD is used for installing HP-UX 11.10. See "Installing HP-UX 11.10" on page 206.

- The *Applications CD.*

  The Applications CD contains optional development and system management software products. See "Installing Optional Software Products" on page 218.

- The *Unlimited User License CD.*

This release provides new SCA programming and process management features, as well as the ability to support multiple V2500 cabinets as a single server running one instance of the HP-UX 11.10 operating system.

An overview of all HP-UX 11.10 features is provided in the Release Notes, which are installed as the file /usr/share/doc/11.10RelNotes.

Restrictions and V2500 System Prerequisites

Some new installation and initial set-up restrictions are introduced for HP-UX 11.10 on multiple-cabinet HP V2500 SCA servers.

- Before installing HP-UX 11.10 on your V2500 SCA server, you must have already installed (and upgraded, if necessary) the hardware and firmware to a supported V2500 SCA configuration.

  Hardware installation includes cabling together the V2500 server cabinets and connecting the cabinets to the Service Support Processor.

  The Service Support Processor must be running the SCA-supported *Test Station Software* release (Version 2.0 or later), and the corresponding V2500 firmware (Version 2.0 or later of POST) must be downloaded to all cabinets that make up the V2500 SCA server complex. The *Test Station Software Release Notice* explains this procedure.

You can check the Test Station Software version using the `/spp/bin/diag_version` command. The `/spp/bin/flash_info` command or Boot Console Handler (BCH) menu `VERSION` command prints firmware versions.

- The V2500 boot device, vg00, and crash dump partion must reside entirely on SCSI disks on cabinet ID 0 (the *monarch cabinet*).
- Swap space may include disks on non-monarch V2500 cabinets, but primary swap space must be entirely on cabinet ID 0.

NOTE

If upgrading an HP V-Class system to a V2500 SCA server configuration, first record the server's network and other details, as listed in "Pre-Install: Information Gathering" on page 200.

Also consider saving other system data, as described in "Pre-Install: Saving Files and Settings" on page 202.

Overview of Installation

The HP-UX installation process involves three phases: pre-install tasks, HP-UX installation, and post-install tasks. These are covered in the sections that follow.

You can perform the installation procedure using the standard `sppuser` account on the Service Support Processor workstation. Most installation operations are done through the cabinet ID 0 console (the "Node 0/Complex console"), which is available on the Service Support Processor.

The cabinet 0 console gives access to the Boot Console Handler (BCH) menu, which is used to select boot devices and perform basic firmware activities before HP-UX boots on a V-Class server. The Ignite-UX terminal interface is managed from the cabinet 0 console during installation.

Normally HP-UX 11.10 is installed on your V2500 SCA system at the factory before it is shipped. However, you can Cold-Install it yourself from the HP-UX 11.10 CD-ROM media.

New SCA Kernel and Tunable Settings

The HP-UX 11.10 release introduces a new *64-bit NUMA* kernel for SCA systems. (The term *NUMA* stands for non-uniform memory access, such as is provided by a multiple-cabinet V2500 SCA server which has physical memory on each cabinet.)

This brings to three the number of types of HP-UX install images: 32-bit, 64-bit, and 64-bit NUMA. The HP-UX install images are as follows:

- INSTALL — 32-bit kernel install image. Not supported at 11.10.
- WINSTALL — 64-bit kernel install image. Not supported at 11.10.
- VINSTALL — 64-bit NUMA kernel install image. Supported at 11.10 for V2500 SCA servers only.

Customized HP-UX kernel parameter configurations are established for V2500 SCA servers upon HP-UX 11.10 installation. The `maxuprc` and `maxusers` parameters are 256, `max_thread_proc` is 500. You may wish to adjust these kernel parameters further based upon your system configuration and application usage patterns.

The `maxswapchunks` parameter also is tuned during the install process, based on the larger swap space that may be configured.

Pre-Install Tasks

Tasks to perform before installing HP-UX 11.10 include gathering information you will supply during the HP-UX installation, and, if needed, saving files and system settings to be restored after the HP-UX 11.10 installation overwrites the target disk's contents.

Pre-Install: Information Gathering

You will need to supply the information in the following list when installing HP-UX 11.10.

This list includes the sizes for the /var/adm/crash and the swap and dump filesystems, V2500 server network details, and the hardware paths for the CD-ROM drive and the target install disk.

If you are installing HP-UX on an HP V2500 server for the first time, you must designate a unique host name, host IP address, and other network information for the V2500 server. Refer to your site network administrator for details.

If installing HP-UX 11.10 on an existing V2500 system runing HP-UX, use the commands listed below to print information about the server.

- Primary Swap and Dump Size: _________ Kbytes
  Recommended size: 1024 Mbytes (1048576 Kbytes) or larger.

  Use the `swapinfo -a` command to determine the current swap setting.

  The primary swap and dump size for a V2500 SCA system *does not* need to equal the system's physical memory size, which may be as large as 128 Gbytes.

- /var/adm/crash Volume Size: _________ Kbytes
  Recommended size: 2048 Mbytes (2097152 Kbytes).

  Use the `bdf /var/adm/crash` command to determine the current size of this filesystem, if it exists.

- Network Information
  - Host Name: ____________________

    The `uname -n` command prints the host name.
  - Host IP Address: ____________________

    The `nslookup` *name* command prints the host IP address.
  - Subnet Mask: ____________________

    The `grep SUBNET_MASK /etc/rc.config.d/netconf` command prints the subnet mask setting in the netconf file.
  - Default Gateway IP Address: ____________________

    The `netstat -r` command prints network routing information, including the "default" gateway IP address.
  - Domain Name: ____________________

    The /etc/resolv.conf file includes both the domain name and the IP address of the nameserver (DNS).

    Use the `cat /etc/resolv.conf` command to print the domain name and DNS IP address.
  - DNS IP Address: ____________________
  - NIS Domain Name: ____________________

    The `grep NIS_DOMAIN /etc/rc.config.d/namesvrs` command prints the NIS domain name.
- CD-ROM Path: ____________________

  You can determine the CD-ROM and install disk hardware paths by using the `SEARCH` or the `IO` Boot Console Handler (BCH) menu commands. `SEARCH` lists bootable devices, and `IO` lists all SCSI devices connected to a V2500 cabinet.
- Install Disk Path: ____________________

Pre-Install: Saving Files and Settings

This section lists optional pre-installation backup tasks.

If replacing a prior HP-UX release or re-installing 11.10, you should consider performing tasks listed here to avoid losing data or configuration tunings due to the Cold-Install of 11.10.

You can save system data using any of the following methods:

- Transfer the files to another system, using `ftp` (see the *ftp*(1) manpage).
- Copy the files to a disk or volume group on the system that will not be overwritten by the Cold-Install. Note that, following the installation, the disk or volume group must be configured back into the system.
- Save the files to a tape.

Creating a Backup Tape

You can perform a full backup using the `fbackup` command.

If using `fbackup` to back up your entire system to tape, you do not need to `umount` any imported file systems; `fbackup` does not cross NFS mounts unless you force it to.

The following command uses the normal tape location and performs a full backup:

```
fbackup -f /dev/rmt/0m -i / -v
```

See the *fmbackup*(1M) manpage for more details.

Saving Kernel Parameter Settings

If your V2500 system has customized HP-UX kernel parameter settings, you may want to save them to refer to them after HP-UX 11.10 has been Cold-Installed with its new parameter settings.

Use the `kmtune` command to list all current kernel parameter settings, and save the settings to a file on tape or a separate disk.

The following command line writes settings to /tmp/param.set.

```
/usr/sbin/kmtune > /tmp/param.set
```

You should reconfigure only selected tunables after the Cold-Install.

NOTE

Several HP-UX kernel parameters are specifically tuned during the HP-UX 11.10 installation to help enable SCA system use, including `maxuprc`, `maxusers`, `max_thread_proc`, and `maxswapchunks`.

See Chapter 6 for details.

Saving Filesystem Setup Details

The current filesystem setup may be useful to reference during the Cold-Install, or afterward when checking which filesystems and volumes are to be mounted on the new HP-UX 11.10 system.

Use the `crashconf -v` command to list crash dump settings and devices, and use the `swapinfo -a` command to list swap settings.

Use the `bdf` command to list current locally mounted filesystems.

The `bdf -t hfs` command lists HFS filesystems, and `bdf -t vxfs` lists VxFS filesystems.

The following commands write HFS and VxFS filesystem information, crash settings, and swap settings, to the file /tmp/fsys.txt.

```
bdf -t hfs > /tmp/fsys.txt
bdf -t vxfs >> /tmp/fsys.txt
crashconf -v >> /tmp/fsys.txt
swapinfo -a >> /tmp/fsys.txt
```

Note that `bdf`, `crashconf`, and `swapinfo` report sizes in Kbytes and Ignite-UX specifies them in Mbytes (1024 Kbytes per Mbyte).

Saving Customized Files

Consider saving any customized HP-UX files as well as any other files on the target disk to be used for installing HP-UX 11.10.

Files and directories to save may include those in the following lists among others.

- The `root` account's dot files. These can be specified as shown below.

  `/.[a-zA-Z]*`

- The /stand/system file.
- Customized files in the /etc and /var directories. These may include the following files among others.

/etc/*.conf

/etc/bootptab

/etc/checklist

/etc/exports

/etc/fstab

/etc/group

/etc/hosts

/etc/inittab

/etc/passwd

/etc/profile

/etc/services

/etc/shutdown.allow

/var/adm/inetd.sec

/var/spool/cron/crontab.root

- The following directories containing customized files. These may need to be saved in part or in entirity, if customizations involving them have occurred.

  /etc/rc.config.d

  /etc/sam/custom

  /usr/local

  /var/sam/lp

  The /home and /opt directories also may need to be partially or completely saved. Only the local home directories (those not imported from a remote system) and any /opt directories containing additional software you installed should be saved.

  You also may want to save any customized startup scripts in the /sbin/init.d directory.

As needed, copy the files and directories in the above lists to a directory in /tmp, and back up the directory to tape or another disk. When copying files and directories, use the `cp -p` command for files and `cp -pr` for directories. This preserves the ownership and access settings.

You also can use the `tar cvhpf` *file*`.tar` *file-list* command to save the files and directories listed in *file-list* to the *file*.tar file. This command saves the files, their associated permissions, and their group and owner settings. See the *tar*(1) manpage for details.

Installing HP-UX 11.10

This section covers the installation procedure for Cold-Installing HP-UX 11.10.

WARNING **Installing HP-UX 11.10 requires a Cold-Install of HP-UX, which will *overwrite everything* on the target disk.**

See "Restrictions and V2500 System Prerequisites" on page 197 for system prerequisites that must be met before installing HP-UX 11.10.

Also see "Pre-Install Tasks" on page 200 for tasks you may need to perform before installing HP-UX 11.10.

Step 1. Power on the Service Support Processor workstation and log in using the `sppuser` account.

The `sppuser` account is the main operator account on the Service Support Processor.

Step 2. Power on all V2500 cabinets in the system and any external storage devices.

Step 3. Use the server's cabinet ID 0 console ("Node 0/Complex console") and access the Boot Console Handler (BCH) menu.

The BCH menu looks like the following.

```
...
TIme [cn:yr:mo:dy:hr:mn[:ss]]    Display or set the real-time clock
VErsion                          Display the firmware versions
[0] Command:
```

Tips

- To access the console, use the `consolebar` or `sppconsole` command from the Service Support Processor. If directly logged in to the Service Support Processor, you can use the complex console window, or create one using the root menu (right mouse click on the workspace background).

- Gain interactive (write) access to the console. If needed, type **Control-e c f** to force write access.
- If HP-UX is booted on the server then log in as root, and shut down and reboot to OBP using the `shutdown -r` command.
- If AUTO BOOT is enabled on the server, interrupt it during the startup process to prevent the server from booting past OBP.

```
System is HP9000/800/V2500 series
Processor is starting the autoboot process.
To discontinue, press any key within 10 seconds.

process interrupted by user
```

Step 4. Insert the *Install and Core OS* HP-UX 11.10 CD in cabinet ID 0's CD-ROM drive.

You can find the cabinet ID in the cabinet LCD's upper left corner.

```
|0 (0,0)USR3852001|
|I-I-----F-I----- |
|I-I-----I-I----- |
|123456789abcde   |
```

Step 5. Check the device path of the CD-ROM drive and the target disk on which to install HP-UX 11.10.

Use the `SEARCH` BCH command to list bootable devices, or `IO` to list all SCSI devices; both commands list devices on cabinet ID 0 only.

```
[0] Command: SEA
Searching for Devices with Bootable Media.
Sel  Path           Device Type (autoexecute file?)
----------------------------------------------------------------------------
P0   4/2/0.0.0      CD-ROM : TOSHIBA CD-ROM XM-5701TA1557           (yes)
P1   1/0/0.6.0      Disk   : SEAGATE ST19171W        HP09           (yes)
[0] Command:
```

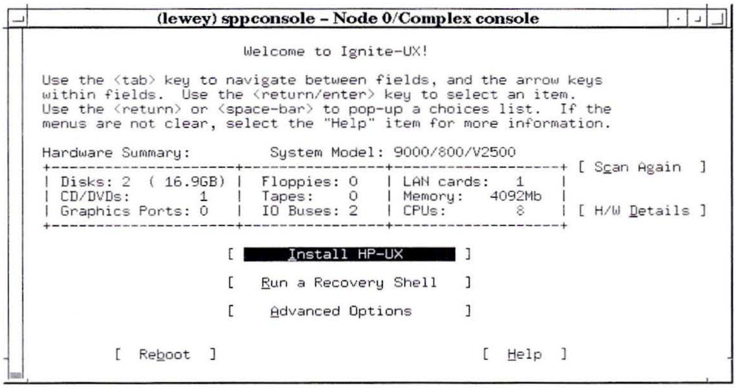
HP-UX 11.10 must be installed on cabinet ID 0 only. Everything on the target disk will be overwritten during the installation.

Step 6. Boot the HP-UX 11.10 CD from the CD-ROM device, using the `BOOT` BCH command.

```
[0] Command: BOOT 4/2/0.0.0
Device       : 4/2/0.0.0
File         : hpux
Arguments    : hpux (;0):VINSTALL
Loading      : hpux .......... 171000 bytes loaded.
```

Booting from the CD takes three minutes or more and initiates the Ignite-UX installation software.

Step 7. Access the main Ignite-UX screen and read about using the "Terminal interface" navigation keys.



```
(lewey) sppconsole - Node 0/Complex console

                         Welcome to Ignite-UX!

   Use the <tab> key to navigate between fields, and the arrow keys
   within fields.  Use the <return/enter> key to select an item.
   Use the <return> or <space-bar> to pop-up a choices list.  If the
   menus are not clear, select the "Help" item for more information.

   Hardware Summary:         System Model: 9000/800/V2500
   +-------------------+----------------+-------------------+ [ Scan Again  ]
   | Disks: 2  ( 16.9GB) | Floppies: 0   | LAN cards:   1   |
   | CD/DVDs:        1   | Tapes:     0  | Memory:   4092Mb |
   | Graphics Ports: 0   | IO Buses: 2   | CPUs:        8   | [ H/W Details ]
   +-------------------+----------------+-------------------+

                        [   Install HP-UX       ]

                        [  Run a Recovery Shell  ]

                        [   Advanced Options     ]

          [ Reboot  ]                              [  Help  ]
```

To use Ignite-UX you must access the console through a window that is 80 characters by 24 characters of size, or larger. If the window is too small you must resize it and have Ignite-UX recheck its dimensions.

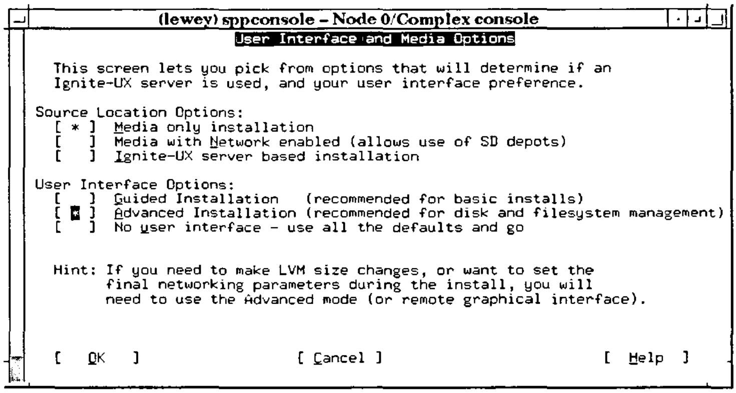
```
The dimensions your terminal window is reporting (80x23) is
insufficient to run the installation application.  It must be at
least 80x24 characters large.  If you are using a resizeable (X11)
terminal window, then you should resize it now.
Would you like recheck the dimensions now? ([y]/n):
```

Step 8. Select “Install HP-UX” and press the **Return** key.

To cancel the installation, select “Reboot” and press **Return**. This reboots the server and boots to OBP.

Step 9. Select “Media only installation” and “Advanced Installation”.

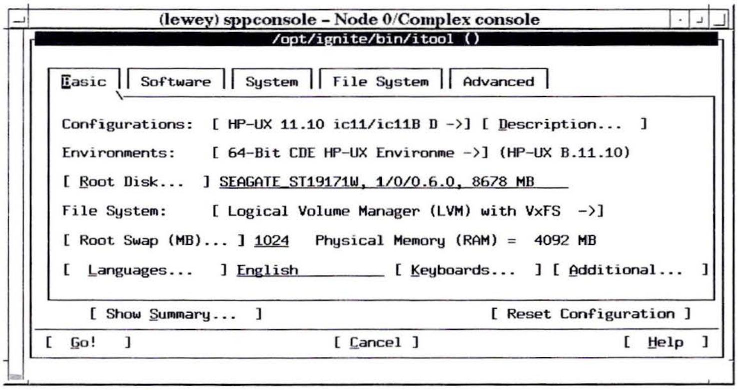
This installs from the CD-ROM (the default) and performs an Advanced install, which lets you customize the filesystems.



Step 10. Configure the HP-UX installation using the Ignite-UX "Basic" settings.

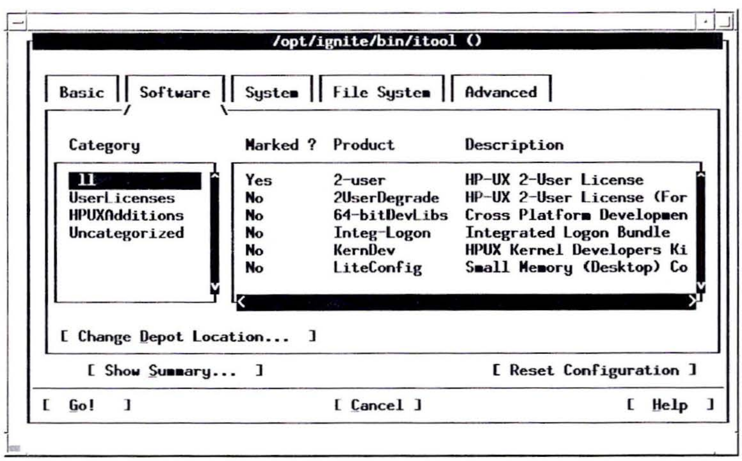
The "Basic" settings let you select the main software package for installation, among other options.

Set the "Root Disk" to be the device onto which HP-UX 11.10 should be Cold-Installed.

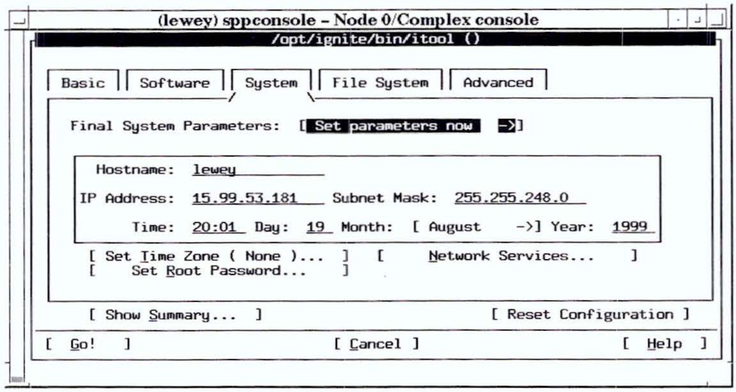


Step 11. Use the "Software" settings to select the software products to be installed.

The "All" category lists all products on the *Install and Core OS* CD-ROM that are available for installation. Only the products marked "Yes" are selected for installation. For the HP-UX 11.10 release, the "LiteConfig" product is not recommended.



Step 12. Configure the system network services, host name, time zone, and root password using the “System” Ignite-UX settings.



Refer to your site network administrator for the correct network settings. See “Network Information” on page 201 for details on determining settings, and for any settings you collected before starting the installation.

a Set the host name, IP address, and subnet mask.

b Select the time zone from the pop-up menu in the main “System” area.

c Set the `root` account’s password.

d Select the “Network Services” sub-area and set its parameters.

“Static Routes” (Gateway)—Establishes the destination, IP address, and hop count settings.

The destination can be “default”, the Gateway IP address is returned by the `netstat -r` command, and the hop count can be `1`.

DNS—Establishes the domain name and DNS IP address settings. The /etc/resolv.conf file includes this information.

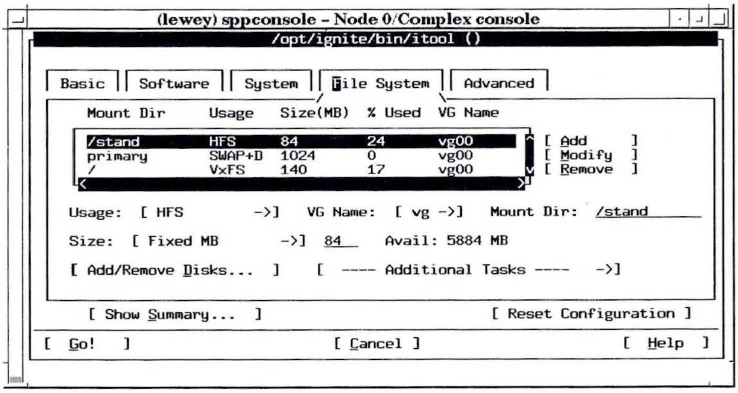
NIS—Establishes the NIS domain name. The /etc/rc.config.d/namesvrs file includes this information.

Setting XNTP may not be required.

Finally, select “OK” to complete the network services set-up.

Step 13. Configure filesystems using the “File System” Ignite-UX settings.

You can add new filesystems and modify existing filesystems. After modifying a filesystem (for example, changing the size of /stand) select “Modify” to save the change.



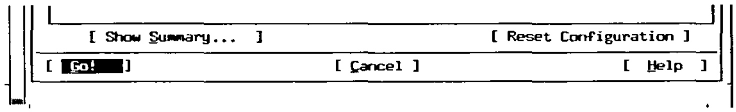
For V2500 SCA servers, you should consider making the following filesystem customizations.

- Increase the size of /stand to 200 Mbytes or more, if multiple kernels may be built.
- Set the primary swap/crash space to be 1024 Mbytes or larger.
- Create (add) a new /var/adm/crash filesystem for saving crash dump files. A suggested size is a “Fixed MB” size of 2048 Mbytes or larger.

  If sufficient space does not exist to create this filesystem during the HP-UX installation, consider adding it afterward.

  To create a new filesystem first enter the mount directory and size, then select “Add” to save it.

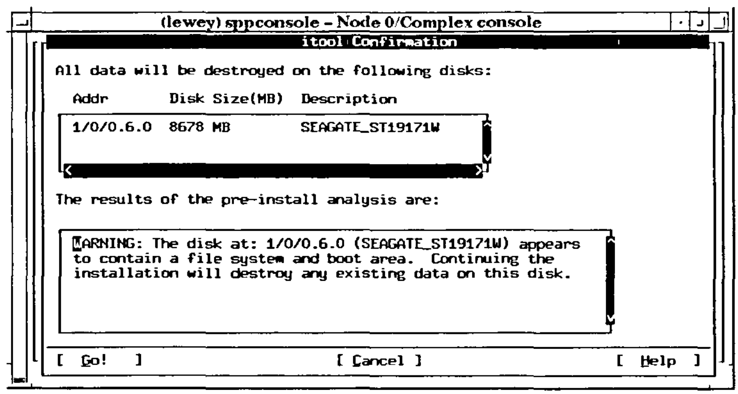
Step 14. Select “Go!” to finish the install configuration and begin installing HP-UX 11.10 as configured.



Step 15. Confirm whether you want to proceed with installing HP-UX 11.10 to the specified disk.

Check the path for the target install disk and confirm that it is correct. For HP-UX 11.10, all contents of the disk will be overwritten.

Also read the pre-install analysis summary.



During installation Ignite-UX displays output in the complex console.

The installation can take 25 minutes or more. After Ignite-UX installs HP-UX, it reboots the server hardware and automatically boots HP-UX 11.10, configures HP-UX and filesystems, restores AUTO BOOT to its previous setting, and reboots the server.

Step 16. After the installation completes and the system has rebooted, remove the install CD from the CD-ROM drive.

Following the installation:

- The primary boot path variable (the BCH `PRI` setting) is set to the install target disk, where HP-UX 11.10 has been installed.
- You can boot the newly installed operating system from the BCH menu by entering the `BOOT PRI` BCH command.

See the next section for optional post-installation tasks you may want to perform.

Post-Install Tasks

This section lists optional system configuration tasks you can perform after installing HP-UX 11.10 and booting HP-UX on an HP V2500 SCA server.

- **Install Optional Software**

  Optional development tools and system management software are available on the *Applications CD* that is distributed with HP-UX 11.10.

  See "Installing Optional Software Products" on page 218 for details.

- **Restore Prior System Configurations**

  If updating a pre-11.10 system with an HP-UX 11.10 Cold-Install, or if re-installing HP-UX 11.10, you may want to restore previous HP-UX configurations.

  See "Pre-Install: Saving Files and Settings" on page 202 for details.

- **Update System Manpages**

  You can update the manpage "whatis" database by running the `/usr/lbin/mkwhatis` script, as necessary. You also may want to run the `/usr/sbin/catman` command; see *catman*(1M) for details.

- **Configure HP-UX Kernel Parameters**

  HP-UX 11.10 has custom kernel parameter tunings when installed on V2500 SCA servers. See Chapter 6 for optional kernel parameter configuration details.

- **Configure Swap (Paging) Space**

  The `/usr/sbin/swapinfo -a` command prints current system paging space information.

  You can configure additional devices and logical volumes to be used as system paging devices (swap space) in several ways.

  The SAM application (`/usr/sbin/sam`) provides a method for configuring paging devices.

You also can edit the /etc/fstab to specify devices or file systems to be used for paging. See the *fstab*(4) manpage for details.

The `/usr/sbin/swapon` command enables devices or file systems to be used for paging purposes. You can either specifying a device on the `swapon` command line, or can specify the `-a` option to make available the devices listed in /etc/fstab.

- **Configure Crash Dump**

  The `/sbin/crashconf -v` command displays the current crash dump settings.

  You can configure additional disk devices to be used for crash dump in several ways, listed below.

  - Use the `/sbin/crashconf` *device* command to specify a device (*device*) to be used for crash dump purposes. Including the `-r` option causes the device to replace any currently specified crash dump device.

  - Edit the /etc/fstab file to define crash dump devices. Entries in the following format are enabled as crash devices during the HP-UX boot process.

    devicepath `/ dump defaults 0 0`

    The `crashconf -a` command adds (or replaces, if `-r` also is specified) crash devices using the devices defined in /etc/fstab entries.

  - Use the SAM application (`/usr/sbin/sam`) or edit the /stand/system file to configure dump devices into the kernel. This method requires rebuilding the HP-UX kernel and rebooting.

  On HP V2500 SCA systems the device(s), or logical volume(s), used for crash dump must reside entirely on cabinet ID 0. Early crash dump (before HP-UX init configures crash devices) is not supported for systems with multiple-disk root volumes.

  For more details on crash dump configuration see the *crashconf*(1M) manpage or refer to the *V2500 SCA HP-UX System Guide* or *Managing Systems and Workgroups*.

- **Configure Network Settings**

  System network settings should have been set when installing HP-UX, but if they were not properly set you must reconfigure them after booting the system.

  Contact your site's network administrator if needed for details.

  Use the `set_parms` command, as shown below, to reconfigure the booted system's network settings.

  ```
  /sbin/set_parms addl_netwrk
  ```

  The `set_parms` command gives you the option to configure the following aspects of networking for the server.

  – Subnetwork Mask and Default Gateway — These should have been set during install. Configure them, or enter **No** or click **Cancel** to not make any changes.

  – Domain Name System (DNS) — At this prompt or window, you can configure the server's domain name, DNS server host name, and DNS server address.

  – NIS client parameters — These also should be properly set. If so, enter **No** or click **Cancel** to not make changes.

Installing Optional Software Products

The *Applications CD* that is distributed with HP-UX 11.10 contains optional development tools and system management software.

Among the software products distributed on the *Applications CD* is the HP OnLineJFS product. Using OnLineJFS requires that you purchase a software license and enter a license key to enable its features. See “Installing the Optional HP OnLineJFS Product” on page 220 for details.

HP-UX 11.10 must be booted on the server in order to install optional software products. The following is the procedure for installing products from the *Applications CD*.

Step 1. Insert the *Applications CD* in cabinet ID 0’s CD-ROM drive.

You can find the cabinet ID in the cabinet LCD’s upper left corner.

```
|0 (0,0)USR3852001|
|R-R-----R-R----- |
|R-R-----R-R----- |
|123456789abcde   |
```

Step 2. Mount the *Applications CD*.

To install software from the *Applications CD*, you must mount the CD as a filesystem that HP-UX 11.10 can access.

a Determine the CD-ROM device name.

Use the `ioscan -funC disk` command to list disk devices, including the CD-ROM. The following sample output shows a CD-ROM on cabinet 0 whose device name is /dev/dsk/c1t0d0.

```
disk      4  4/2/0.0.0  sdisk CLAIMED      DEVICE        TOSHIBA CD-ROM XM
                           /dev/dsk/c1t0d0    /dev/rdsk/c1t0d0
```

b Create a mount point for the *Applications CD*, if one does not yet exist.

The mount point is a directory that HP-UX will use as an access point for the CD. Often a /CDROM directory is used. If this directory does not exist, create it using the `mkdir /CDROM` command.

c Use the `mount` command to mount the CD.

Using the `mount` command, specify the CD-ROM device name and mount point. For example, the following command mounts the /dev/dsk/c1t0d0 device as the /CDROM directory.

`mount /dev/dsk/c1t0d0 /CDROM`

See the *mount*(1M) manpage for details.

Step 3. Use `swinstall` to install software from the *Applications CD*.

The following example command runs `swinstall` to install software from the source mounted at /CDROM.

`swinstall -i -s /CDROM`

See the *swinstall*(1M) manpage for details.

Step 4. Select and install software from the *Applications CD*.

The `swinstall` program presents the following interface for selecting and installing software from the CD.

```
Starting the terminal version of swinstall...

To move around in swinstall:

- use the "Tab" key to move between screen elements
- use the arrow keys to move within screen elements
- use "Ctrl-F" for context-sensitive help anywhere in swinstall

On screens with a menubar at the top like this:

       -------------------------------------------------------
       |File View Options Actions                         Help|
       | ---- ---- ------- -------------------------------- ---|

- use "Tab" to move from the list to the menubar
- use the arrow keys to move around
- use "Return" to pull down a menu or select a menu item
- use "Tab" to move from the menubar to the list without selecting a menu item
- use the spacebar to select an item in the list

On any screen, press "CTRL-K" for more information on how to use the keyboard.

Press "Return" to continue...
```

Step 5. Unmount and eject the *Applications CD*.

You must unmount the CD before you can eject it from the CD-ROM drive. The CD is automatically unmounted whenever the server reboots.

Use the `umount` command to unmount the CD. For example, `umount /CDROM` will unmount the /CDROM filesystem. See the *umount*(1M) manpage for details.

Installing the Optional HP OnLineJFS Product

HP OnLineJFS is the advanced optional product for the VxFS File System. You can use the capabilities of HP OnLineJFS to perform certain key administrative tasks on mounted VxFS filesystems; this allows users on the system to perform their work uninterrupted.

These tasks include:

- Defragmenting a file system to regain performance.
- Resizing a file system.
- Creating a snapshot file system for backup purposes.

To install the HP OnLineJFS product, you first need to obtain a license key. You need to obtain a license key even if you have purchased and installed a previous version of OnLineJFS.

When you order the HP OnLineJFS product (B3929CA), you will receive the following:

- License Entitlement Certificate/ License Key Request Form
- JFS/OnLineJFS Version 3.3 for HP-UX 11.00 CD
- HP JFS 3.3 and HP OnLineJFS 3.3 Release Notes for HP-UX 11.00 Only

NOTE

You only need the License Entitlement Certificate/License Key Request Form for HP-UX 11.10. The other items (CD and Release Notes) apply only to HP-UX 11.00.

Obtaining a License Key for the HP OnLineJFS Product

The HP OnLineJFS product uses a runtime licensing mechanism. You must obtain a runtime license key from HP to install and use HP OnlineJFS.

When you order HP OnLineJFS you receive a License Entitlement Certificate. The License Key Request Form is on the back of the certificate.

NOTE

If you cannot locate these documents or have questions about licensing, contact **Licensing Services** at:

Hewlett-Packard World Wide Licensing Services

California, U.S.A.
Phone: 650-960-5111
fax: 650-960-5670
e-mail: hplicense@mayfield.hp.com
Hours: Monday through Friday, 8am-5pm PST

Grenoble, France
Phone: +33.(0)4.76.14.15.29
fax: +33.(0)4.76.14.25.15
e-mail: codeword_europe@hp-france-gen1.om.hp.com
Hours: Monday through Friday, 8am-5pm GMT +1

You can redeem your license as follows:

Step 1. Get the **`hardware/model name`** and **`hostid`** of the system(s) on which you will install HP OnLineJFS.

To obtain the **`hardware/model name`**, from the server system(s) run

```
uname -m
```

To obtain the **`hostid`**, from the server system(s) run

```
uname -i
```

Step 2. Fill out the License File Request Form for each system and fax it to HP at the appropriate number for you listed on the License Entitlement Certificate.

Alternatively, you may send the information to HP via e-mail or you may phone HP. Use the e-mail address or phone number listed above or on the License Entitlement Certificate.

You will receive your license key(s) by e-mail and fax within eight business hours.

Installing the HP OnLineJFS Product

OnLineJFS is installed from the *Applications CD* after HP-UX 11.10 has been installed from the *Install and Core OS CD*.

You can install HP OnLineJFS as follows:

Step 1. Check to see whether the HP JFS product has been installed.

JFS is installed as part of the HP-UX 11.10 core installation. Run the following command to check if HP JFS is installed:

```
swlist -l fileset JFS
```

If JFS is installed, the output will include the following:

```
# JFS      B.11.10    The Base VxFS File System
```

If you get an error message, JFS is not installed. Install the JFS fileset from the core media.

Step 2. Run the following command as root:

```
/sbin/vxlicense -c
```

At the prompt from `vxlicense`, type in the 23-digit license key you received from HP Licensing Services. (See the section "Obtaining a License Key for the HP OnLineJFS Product".)

Step 3. Enable the OnLineJFS features with the following command (run as the `root` user):

```
/sbin/fs/vxfs/vxenablef -a
```

NOTE: Normally `vxenablef` is run from /etc/inittab when the system boots. You only need to run it manually once, after running `vxlicense`.

Step 4. Register the HP OnLineJFS product (product number B3929CA) with `swinstall`.

First insert and mount the *Applications CD* that accompanies the HP-UX 11.10 release.

Then use the following `swinstall` command to install the OnLineJFS product:

```
swinstall -x autoselect_dependencies=false -s software-source B3929CA
```

where ***software-source*** is the physical media (for example, /CDROM) or depot containing the product.

After the installation, OnLineJFS features will be immediately usable; no reboot is need.

For more information about new features in this release of JFS and OnLineJFS, see the *Release Notes for HP-UX Release 11.10*.

Using HP OpenView IT/Operations

The HP OpenView IT/Operations (ITO) agent version A.05.30 is enabled to support HP-UX 11.10.

NOTE The HP IT/Operations software *is not* available on the HP-UX 11.10 Applications CD.

The ITO management server software component *is not supported* on HP-UX 11.10. In order to get the functionality to manage HP-UX 11.10 systems using ITO, you must install the ITO management server software either on an HP-UX 10.20 or HP-UX 11.0 system.

The ITO management server software includes the ITO agent software and is available by purchasing the appropriate media product:

- B6940AA—ITO HP-UX 10.20 media.
- B6040BA—ITO HP-UX 11.0 media.

or

- The corresponding December 1999, or later, Applications CD.

If you have an ITO support contract, you should already have the appropriate ITO CD media listed above.

HP IT/Operations Special Edition Online Help

A Web browser is not shipped with the HP-UX 11.10 release media. The OpenView IT/Operations Special Edition (ITO-SE) software—which is shipped as part of the GlancePlus PAK (B3701AA, B3699AA)—requires that a Web browser be configured in order to display online help.

You can use any Web browser, such as Netscape Navigator, to view ITO-SE online help. For HP-UX 11.10, you must install the browser manually following the HP-UX installation.

For directions on configuring a Web browser for viewing the online help, refer to the README file (/opt/perf/ReleaseNotes/Itose). This file is installed as part of the GlancePlus PAK software.

B Units of Measurement

This appendix has tables for the following units of measurement:

- “Time Units” on page 226.
- “Data Units” on page 227.

Time Units

This table lists units for measuring time.

Table 1 **Units for Measuring Time**

Unit	Time (American units)	Example
hour	60 minutes 3600 seconds	• Large workloads may take several dozen hours to complete.
minute	60 seconds	• Transferring data to or from remote systems or tape (DAT) devices may take several minutes.
second	1 second	• The time taken to access data across a network can be measured in milliseconds to seconds or more.
millisecond	0.001 seconds 10^-3 seconds One-thousandth of a second	• Reading data from a disk drive can take milliseconds: anywhere from less than one millisecond to more than ten milliseconds.
microsecond	0.000001 seconds 10^-6 seconds One-millionth of a second	• Communication from a CPU to an I/O device's control registers can be measured in microseconds.
nanosecond	0.000000001 seconds 10^-9 seconds One-billionth of a second	• Accessing memory that is not encached takes a few hundred nanoseconds. • A CPU's clock operating at 250 MHz will "tick" once every 4 nanoseconds.
picosecond	0.000000000001 seconds 10^-12 seconds One-trillionth of a second	• Internal communication within hardware (such as a circuit) can occur in picoseconds.

Data Units

The following table lists various units for measuring data.

Table 2 **Units for Measuring Amounts of Data**

Unit	Abbreviation	Size
byte	byte	8 bits
kilobyte	Kbyte	1024 bytes 2^10 bytes
megabyte	Mbyte	1024 Kbytes 2^20 bytes 1,048,576 bytes
gigabyte	Gbyte	1024 Mbytes 2^30 bytes 1,073,741,824 bytes
terabyte	Tbyte	1024 Gbytes 2^40 bytes 1,099,511,627,776 bytes
petabyte	Pbyte	1024 Tbytes 2^50 bytes 1,125,899,906,842,624 bytes
exabyte	Ebyte	1024 Pbytes 2^60 bytes 1,152,921,504,606,846,976 bytes

Examples

- The size of a CPU's data cache can be measured in megabytes (for example: 2 Mbytes) or kilobytes (for example: 2048 Kbytes).
- The amount of physical memory in a system often is measured in gigabytes (for example: 32 Gbytes).
- Likewise, the amount of disk space in a system often is measured in terms of gigabytes or, if a lot of disk space is available, terabytes.

Glossary

BCH (Boot Console Handler)

The menu-driven interface available on V-Class servers before HP-UX boots. Normally only cabinet ID 0 (the monarch cabinet) provides a BCH menu. Sometimes called the "OBP menu" or the "OBP interface".

binding

The ability to assign a specific processor or *locality domain* that HP-UX uses for running a program's threads or processes. Bindings are either mandatory (required) or advisory. Mandatory bindings guarantee that a program's threads or processes are not migrated to different processors or localities. Locality bindings always are mandatory.

See locality domain.

boot-time variables

Variables that configure the system hardware and its behavior before HP-UX boots. The boot-time variables are stored in non-volatile memory on the cabinet's utilities board. Also called "NVRAM variables".

See NVRAM, utilities board.

cabinet ID

The hardware identifier for an HP V2500 cabinet. Up to four V2500 cabinets can be combined into a single V2500 SCA system, using cabinet IDs 0, 2, 4, and 6. Cabinet ID 0 is the *monarch cabinet* and others are the *serf cabinets*. HP-UX does not refer to the cabinet ID, and instead references cabinets as *locality domains*.

See locality domain, monarch cabinet, serf cabinet.

child process

A new process created by a pre-existing process that calls the fork(2) system call. The new process is thereafter known to the pre-existing process as its child process. The pre-existing process is the parent process of the new process.

See parent process.

Cold-Install, Cold-Installing

A method of installing HP-UX that overwrites everything on the target disk. Also called a “scratch install” or “manual install”. Installing HP-UX 11.10 requires a Cold-Install.

complex

A collection of V2500 server cabinets that are collectively managed by the Service Support Processor as a single server. The whole V2500 SCA server complex boots a single instance HP-UX. Also called a “server complex” or “SCA complex”.

crash dump

The process of saving selected portions of a system’s physical memory (or all of it) to disk if the system crashes. The resulting crash dump image can be useful in determining the cause of the crash.

See TOC (Transfer of Control).

CTI (Coherent Toroidal Interconnect)

The method by which cabinets in a multiple-cabinet HP V2500 server are tightly interconnected. CTI connections are used for accessing memory, forwarding I/O requests, and signaling interrupts across cabinets. HP’s CTI is an extension of the IEEE SCI (Scalable Coherent Interface) standard.

CTI cache

Physical memory used to encache data fetched and prefetched from memory on remote cabinets. CTI cache memory is dedicated as cache and is not available to HP-UX or applications.

CTI controllers

The controllers, on all V2500 SCA memory boards, that are used to join cabinets together. CTI controllers provide routes to memory on remote cabinets. CTI cables connect CTI controllers across cabinets.

See CTI ring.

CTI rings

The connections among CTI controllers on different V2500 SCA cabinets. Also called “CTI cables”.

diagnostic LAN

The internal network that provides diagnostic connections for various utilities, including the ccmd daemon that reports current cabinet status to the “test station console”. The diagnostic LAN connects from the Service Support Processor to the V2500 SCA cabinets.

Also called the “JTAG LAN”.

Forth mode

A lower-level command mode available from the Boot Console Handler (BCH) menu. Normally you should not need to access Forth mode.

See BCH.

Gbyte

Gigabyte: 1024 Mbytes, 2^30 bytes, or 1,073,741,824 bytes.

HPA (Hard Physical Address)

A unique hardware identifier that specifies a particular hardware component, such as a processor, within a V2500 SCA complex.

HP JFS

The Journaled File System (JFS) implementation in HP-UX. On HP-UX 11.10 it is implemented using VxFS 3.3.

HPMC (High-Priority Machine Check)

A fatal machine error.

hypernode-bitmask

A variable setting stored in the server's NVRAM that indicates which cabinets are configured to be in a multiple-cabinet V2500 SCA server complex.

HyperPlane Crossbar

A switching device used in multiprocessor, shared-memory computer systems (such as HP V-Class servers) that connects processors to the various banks of memory in the system.

Kbyte

Kilobyte: 1024 (2^10) bytes.

launch count

A count that HP-UX keeps to track the number and location of threads or processes launched within a policy tree. Every thread policy tree and process policy tree has its own launch count.

launch policies

Policies used for determining how HP-UX distributes threads and processes among an SCA system's *locality domains*. (Each V-Class server cabinet is considered to be a locality domain.) HP-UX uses default launch policies unless you specify a thread-launch policy or process-launch policy. Specifying a launch policy also *binds* the effected threads or processes to their localities.

See binding.

level 4 reset

Provides a Transfer of Control (TOC), equivalent to pressing the TOC button on a V-Class cabinet. A TOC initiates a crash dump of the system, if crash dump is configured, and resets all server hardware.

See TOC.

locality domain

A set of resources—including processors, memory, and I/O—that comprise a fundamental building block of a system. All resources within a locality have an equal latency to memory.

HP-UX considers each V-Class server cabinet to be a locality domain. Locality domains are sequentially numbered by HP-UX, starting at 0; this differs from V2500 cabinet ID numbering.

See cabinet ID.

Mbyte

Megabyte: 1024 Kbytes, 2^20 bytes, or 1,048,576 bytes.

monarch cabinet

Cabinet ID 0 in a V2500 SCA server complex. The monarch cabinet is where the system's boot devices reside. As the server hardware boots, the OBP firmware on cabinet ID 0 takes control of the server complex.

See cabinet ID, serf cabinet.

node-local memory

Physical memory on each cabinet that is used for storing firmware and replicated kernel text. For HP-UX 11.10, 128 Mbytes must be dedicated as node-local memory. Node-local memory is not available to HP-UX or applications.

Nodemask

See hypernode-bitmask.

NVRAM (Non-Volatile Memory)

Memory on the utilities board of each V2500 cabinet that is used for storing boot-time settings and system configurations.

OBP

Open Boot PROM, the firmware component that performs system configuration and provides booting support.

On multiple-cabinet servers, OBP also engages in a Node Address Resolution Protocol that synchronizes all cabinets. Each cabinet runs its own copy of OBP, but OBP running on the monarch cabinet (ID 0) ultimately takes control.

OBP menu interface

See BCH (Boot Console Handler).

oversubscription

A condition when the number of threads and processes in an application exceeds the number of processors in a single locality domain, or exceeds the total number of processors in a server.

parent process

Whenever a new process is created by a currently-existing process (through the fork(2) system call), the currently existing process is said to be the parent process of the newly created process. Every process has exactly one parent process (except the init process), but each process can create several new processes by calling fork(). The parent process ID of any process is the process ID of its creator.

See child process.

policy tree

A collection of threads or processes that share the same SCA launch policies and launch count details. The launch count and the thread-launch or process-launch policy are maintained by the policy tree's root.

policy tree root

The root node of a policy tree, which maintains the launch count and the thread-launch or process-launch policy.

POST (Power-On Self Test)

The initial firmware component that performs minimal CPU and core diagnostics to ensure the essential system components are present and are functioning. It also performs some memory initializations and configurations. On multiple-cabinet servers, an instance of POST executes simultaneously on each cabinet.

process policy tree

See policy tree.

process-launch policy

See launch policies.

SCA (Scalable Computing Architecture)

A computing architecture that allow multiple locality domains to be combined to form a single, tightly-interconnected server managed by a single operating system.

HP-UX provides a single system image for each SCA system, regardless of the number of cabinets (localities) that comprise it.

See locality domain.

serf cabinet

In a multiple-cabinet V2500 SCA server complex, the serf cabinets are cabinet IDs 2, 4, and 6, if present. The serf cabinets contribute additional resources to a multiple-cabinet server, and are controlled by the monarch cabinet.

See cabinet ID, monarch cabinet.

Service Support Processor workstation

A workstation used for managing the cabinets in a V-Class system. It is an HP workstation that contains software for booting and helping configure and manage a V-Class system's hardware. It also is used to log HP-UX console messages and certain V-Class system status messages.

sppconsole

A program that provides a console interface for HP V2500 cabinets. Normally, only `sppconsole` access to cabinet ID 0 is needed, for accessing the Boot Console Handler (BCH) menu.

targeting

The ability to request that HP-UX allocate memory from the same *locality domain* as the requesting thread. HP's SCA memory targeting policies are supported by the mmap() and shmget() system calls.

When you *bind* a thread or process to a locality domain, the locality's memory is targeted for use.

See binding, locality domain.

Tbyte

Terabyte: 1024 Gbytes, 2^40 bytes, or 1,099,511,627,776 bytes.

Test Station Software package

The software that provides the Service Support Processor software environment, including diagnostics, utilities, the `sppuser` login account, and other files.

See Service Support Processor workstation.

thread policy tree

See policy tree.

thread-launch policy

See launch policies.

TOC (Transfer of Control)

A server hardware reset, equivalent to a level 4 reset. On multiple-cabinet V2500 SCA servers, a TOC causes the same behavior as a panic. When a TOC occurs, the server gives you the option to select the type of crash dump that will be performed.

See crash dump.

utilities board

The board in each V2500 cabinet that provides system booting and testing functions. It connects to the cabinet's core logic bus, system environmental sensors, and other test points, as well as the Service Support Processor workstation.

Index

M

N

HEWLETT® PACKARD

Order Part Number
A5532-90003
E1299

Printed on chlorine-free
bleached paper

Printed in Germany

A5532-96003