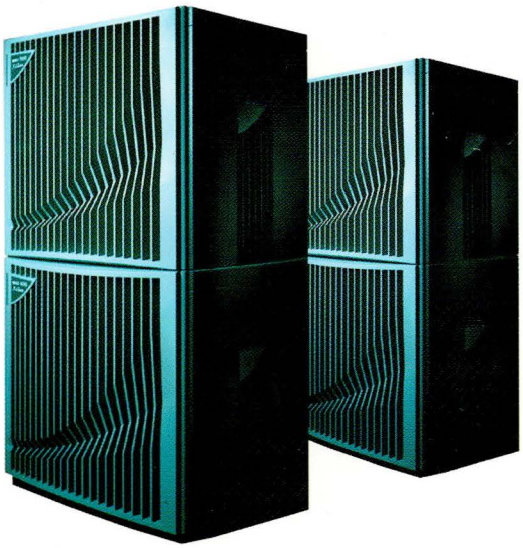
HEWLETT® PACKARD

S-Class Servers



Exemplar User's Guide

First Edition

Hewlett-Packard Company
Convex Division
3000 Waterview Parkway
P.O. Box 833851
Richardson, TX 75083-3851
United States of America

Exemplar User's Guide

S-Class Servers

B5655-90003

First Edition

January 1997

Hewlett-Packard Company
Convex Division
Richardson, Texas
United States of America

Exemplar User's Guide

S-Class Servers

B5655-90003

Notice

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

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

This entire book is recyclable.

Printed in the United States of America

Revision Information for

Exemplar User's Guide

S-Class Servers

Edition	Document No.	Description
First	B5655-90003	Initial release, January 1997.

Contents

Figures

Tables

How to use this guide

Purpose and audience

This book provides an introduction to using Hewlett-Packard (HP) Exemplar S-Class and X-Class servers. An overview of the hardware is given along with descriptions of the software that runs S-Class and X-Class servers.

The SPP-UX operating system, which runs on Exemplar S-Class and X-Class servers, SPP-UX utilities for determining the system configuration and monitoring server performance, and programming languages and tools are covered in this book.

Organization

This guide is organized as follows:

- Chapter 1, "Introduction"—Provides an introduction to Exemplar S-Class and X-Class technical servers, briefly describes their unique features and uses, and introduces commands and details of interest to users and programmers.
- Chapter 2, "Overview of S-Class and X-Class servers"—Provides a detailed overview of S-Class and X-Class server hardware and the operating system and file system. The various types of memory also are discussed.
- Chapter 3, "Determining the system configuration"—Describes methods to get details about an S-Class or X-Class server's hardware and software configuration.
- Chapter 4, "Monitoring system use"—Lists ways to monitor how an S-Class or X-Class server is being used, including load, process, memory, and user information.
- Chapter 5, "Programming tools, languages, and libraries"—Gives an overview of the programming languages, programming tools, and libraries available for use on S-Class and X-Class servers. A brief discussion of programming methods also is given.

Notational conventions

This section discusses notational conventions used in this book.

`Bold monospace`

In command examples, text shown in **`bold monospace`** identifies user input that must be typed exactly as shown.

`Monospace`

In paragraph text, `monospace` identifies command names.

In command examples, `monospace` identifies command output, including error messages.

Italic

In paragraph text, *italic* identifies new and important terms and titles of documents.

KEYCAP

In paragraph text, text shown in **KEYCAP** indicates keyboard keys you must press to execute the command. For example, **RETURN** refers to the carriage return key.

Two **KEYCAP** terms separated by a hyphen indicate two keys that you must press simultaneously. For example, **CTRL-d** indicates that you must press the **d** key while holding down the **CTRL** key.

Note

A Note highlights supplemental information.

Associated documents

The *Guide to Exemplar Documentation* (B5655-90001) lists and describes all documentation available to support HP Exemplar S-Class and X-Class Servers.

The following hardcopy documents may be of interest to Exemplar S-Class and X-Class application developers:

- *HP Fortran 90 Programmer's Reference* (B5876-90001) is a complete language reference. It also covers compiler options, compiler directives, and library information.
- *HP Fortran 90 1.0 Programmer's Notes* (B5876-90002) provides usage information, including how to compile and link, suggestions and tools for migrating to HP Fortran 90, and how to call C and SPP-UX routines from HP Fortran 90.
- *Exemplar C and Fortran 77 Programmer's Guide* (B5600-90002) describes how to use the Exemplar C and Fortran 77 compilers.
- *Exemplar Programming Guide* (B5600-90001) describes efficient shared-memory programming techniques using the

SPP1000-Series compilers (/usr/convex/bin/cc and /usr/convex/bin/fc).

- *C++ Programming Guide* (B5630-90001) describes how to use the Exemplar C++ compiler.
- *CXdb Quick Reference* (B5639-90001) covers the more frequently-used features of the CXdb debugging tool.
- *CXpa Reference* (B5639-90002) provides both introductory and reference information for using the CXpa performance analyzer.
- *CXtrace User's Guide* (B4539-90003) has information about CXtrace, a trace-based performance analysis tool.
- *Exemplar User's Guide* (B5655-90003) provides user-level information about using HP Exemplar S-Class and X-Class technical servers.
- *HP MPI User's Guide* (B6011-90001) describes Hewlett-Packard's Message Passing Interface (HP MPI) product. MPI is a library of C- and Fortran-callable routines for message-passing programming.
- *HP PVM User's Guide* (B5885-90001) describes Hewlett-Packard's Parallel Virtual Machine (HP PVM), which is a library of C- and Fortran-callable routines for message-passing programming.
- *HP FORTRAN/9000 Programmer's Reference* (B3906-90002) and *HP FORTRAN/9000 Programmer's Guide* (B3906-90001) describe the HP FORTRAN 77 programming language on HP-UX.
- *HP C/HP-UX Reference Manual* (92453-90024) presents reference information on the C programming language as implemented by Hewlett-Packard.
- *HP C/HP-UX Programmer's Guide* (92434-90002) contains detailed discussions of selected C topics.

For more information on Exemplar S-Class and X-Class hardware and software, you can order these books from Hewlett-Packard:

- *SPP-UX System Administration Guide* (B5655-90002) describes how to configure, control access to, and manage the file system of the SPP-UX operating system. Also describes the subcomplex manager, system accounting, and printer management.
- *Exemplar Architecture: S-Class and X-Class Servers* (A4716-90001) provides technical information and optimization applications for Exemplar S-Class and X-Class architecture.
- *Exemplar Networking Guide: S-Class Servers* (B5655-90004) describes tasks required to configure and administer

networking hardware and software based on the type of network at your site.

- *HP-UX 10.0 File System Layout Whitepaper* (March 1995) explains the reasoning for and impacts of the new file system layout and system startup/shutdown model. An ASCII file is located on your Exemplar S-Class Technical Server in /usr/share/doc/filesys.txt.
- *NQS System Administration Guide: Exemplar S-Class Servers* (B5589-90001) describes basic NQS concepts, configuration, and maintenance.
- *OpenBoot Quick Reference: Exemplar S-Class Servers* (B5655-90008) is the standard reference for OpenBoot PROM (OBP).
- *FlexLM End User Manual* (B5655-90010) provides a licensing overview for system administrators and end users. Topics covered include: the license file and license administration tools.
- *Managing software with SD-UX* (B2355-90080) introduces Software Distributor (SD). Covers distributing and administering software, as well as creating software packages with SD.

Ordering documents

To order additional copies of this document or other documents listed in the section "Associated documents", send requests to:

Hewlett-Packard Company
Convex Division
Customer Service
P.O. Box 833851
Richardson TX 75083-3851 USA

Please include the order number (*xxxxx-9xxxx*) or the exact title of the document.

Technical assistance

If you have questions that are not answered in this book, contact the Hewlett-Packard Convex Technical Assistance Center (TAC) at the following locations:

Within the continental U.S., call 1 (800) 952-0379.

From Canada, call 1 (800) 345-2384.

All other locations, contact your local Hewlett-Packard office.

You can also use the `contact` utility, if you would like to report any problems you may have with HP Fortran 90 or its associated documentation. For more information refer to the contact(1) man page.

Introduction 1

This book provides an introduction to Hewlett-Packard (HP) Exemplar S-Class and X-Class technical servers. It describes the Exemplar hardware and software environment and covers how you can use it for running software and developing programs.

Because this book provides summary information about the topics it covers, you should refer to the books listed in the section "Associated documents" on page xii for more detailed information.

Exemplar S-Class and X-Class servers

Exemplar S-Class and X-Class servers are capable of solving problems too complex or too large to be solved on workstations. They provide high levels of throughput and a reduced time-to-solution for large problems. They also are compatible with other HP PA-RISC platforms, including HP workstations and other HP servers. This is partly because of processor compatibility and partly due to compatible operating systems among the platforms.

Exemplar S-Class and X-Class servers have unique hardware features for supporting multiple processors (up to 64 CPUs) and greater I/O and memory capacities than found on other machines. Likewise, the SPP-UX operating system includes both complete HP-UX functionality and additional features for running programs in parallel on multiple processors and managing the large memory and I/O capacities.

A benefit of Exemplar S-Class and X-Class servers is that they are scalable—the amount of processor, memory, and I/O hardware can be adjusted as computing needs demand. Also, regardless of how large or small the hardware configuration is, a single system image is maintained by SPP-UX, thus giving users a view of the server as a single machine.

The Exemplar S-Class and X-Class environment

As already mentioned, features of S-Class and X-Class servers include multiple processors, a large shared memory, the SPP-UX operating system, and the ability to add hardware as needs increase. S-Class and X-Class servers also can be quickly configured (or reconfigured) to share server resources among different groups of users.

An overview of the Exemplar environment is given in Chapter 2. The overview includes both the hardware environment (the processors, physical memory, and rest of the architecture) and software environment (including the operating system, file system, and virtual memory). Also in Chapter 2 is a discussion of subcomplexes, which determine the processors and memory available to various users and programs.

Determining an Exemplar server's configuration is covered in Chapter 3. You can get information about:

- The server's name and internet address
- Number of processors
- Memory capacity and cache allocation
- Subcomplex configuration(s) and name(s)
- Available disk space
- Other details

Monitoring performance and programming

Monitoring the use of S-Class and X-Class servers is the focus of Chapter 4. You can use several SPP-UX utilities to find out how heavily a server's processors are being used, which programs are running, how many threads a process has spawned, and who is logged in using the server.

Programming tools available for S-Class and X-Class servers are briefly described in Chapter 5. Several programming languages (Fortran 77, Fortran 90, C, and C++) are available, in addition to optimized libraries, message passing libraries (PVM and MPI), and performance analysis and debugging tools.

Overview of S-Class and X-Class servers

2

This chapter provides an introduction to the HP Exemplar S-Class and X-Class technical servers and the SPP-UX operating system and file system, which run on S-Class and X-Class servers.

Topics covered in this chapter include the following:

- An overview of the HP Exemplar server family
- Descriptions of the S-Class and X-Class architectures, operating system, and file system
- A discussion of the different categories of memory
- Explanations of how S-Class and X-Class servers' resources can be partitioned (and restricted) for different uses by using subcomplexes

Chapter 3 covers utilities for getting information about a server's hardware and software configuration. Utilities for monitoring system use are listed in Chapter 4. Programming languages, debugging tools, performance tools, and libraries are covered in Chapter 5.

Exemplar technical server family

HP's Exemplar family of computing systems includes four classes of scalable high-performance servers: D-Class, K-Class, S-Class, and X-Class. Exemplar machines have one or more PA-RISC processors, high-bandwidth distributed memory subsystems, and scalable I/O subsystems. Table 1 is an overview of Exemplar servers' features.

Table 1 Exemplar technical server family

Class	CPUs	Memory	I/O
D-Class	1 to 2	32 MB to 1.5 GB	8 slots maximum (Ethernet, SCSI, SCSI-2, parallel, and RS-232)
K-Class	1 to 4	128 MB to 4 GB	8 slots maximum (Ethernet, SCSI-2, parallel, and RS-232)
S-Class	4 to 16	256 MB to 16 GB[1]	1 to 24 PCI controllers
X-Class	4 to 64	1 GB to 64 GB[2]	1 to 96 PCI controllers

1. 16 GB maximum using 64-mbit SDRAMs. The limit using 16-mbit SDRAMs is 4 GB.
2. 64 GB maximum using 64-mbit SDRAMs. The limit using 16-mbit SDRAMs is 16 GB.

Exemplar servers are highly interoperable among each other and with HP desktop workstations. They are based upon the same processor architecture (HP's PA-RISC) and provide compatible operating environments (HP-UX on D-Class and K-Class, and SPP-UX on S-Class and X-Class).

The rest of this book covers Exemplar S-Class and X-Class servers only.

Exemplar S-Class and X-Class architecture

This section gives an overview of the Exemplar S-Class and X-Class architectures.

Scalability is a major feature of the S-Class and X-Class architectures: as computing needs increase, the number of processors, amount of memory, and I/O capabilities can be increased. Regardless of how the hardware is configured, S-Class and X-Class servers maintain a single SPP-UX operating system image.

A complete S-Class or X-Class server consists of at least one hypernode. As shown in Figure 1, a hypernode includes processors (up to 16 PA-8000 CPUs), memory, I/O interfaces, and hardware for interconnecting these components. S-Class servers

are single-hypernode machines; X-Class servers can be scaled to multiple-hypernode configurations.

The HP Exemplar S-Class server supports from 4 to 16 PA-8000 processors, from 256 MB to 16 GB of physical memory, and from 1 to 24 high-performance PCI I/O controllers (for network, disk, tape, or other use). Another major component is the crossbar, which provides the processors and I/O controllers access to and from physical memory.

HP Exemplar X-Class servers may consist of one to four hypernodes and are technically capable of being scaled further. Each X-Class hypernode may be configured in the same manner as an S-Class server, allowing for a total of up to 64 CPUs, 64 GB of physical memory, and 96 PCI I/O controllers in a four-hypernode X-Class server.

Figure 1 shows an overview of a single hypernode.

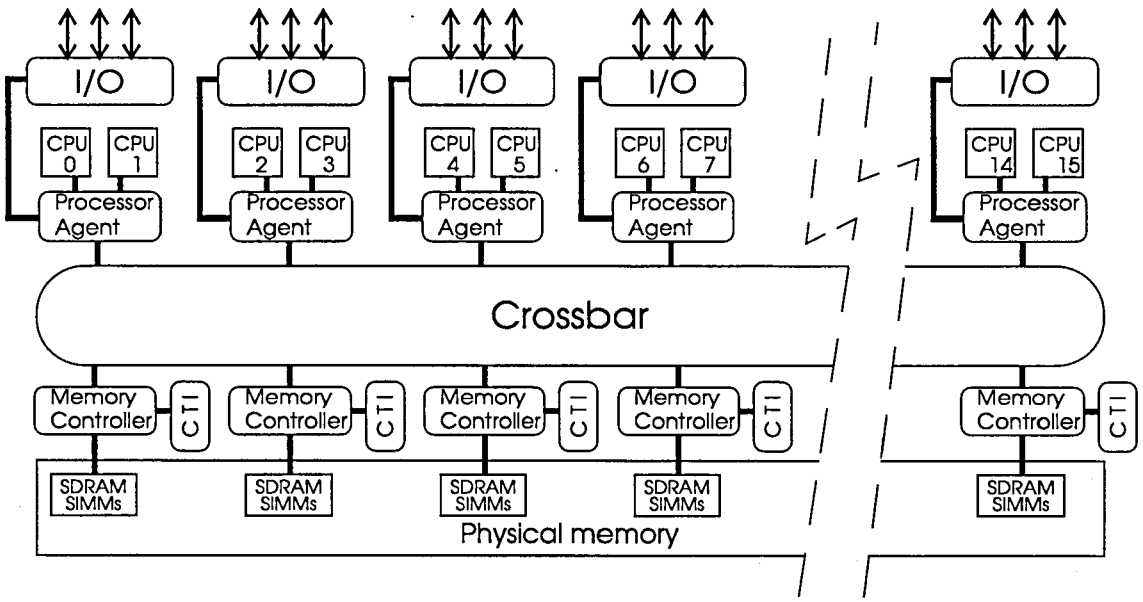


Figure 1 Exemplar S-Class and X-Class hardware overview (one hypernode)

Figure 1 depicts a single hypernode with the maximum number of processors (16 processors: CPU 0–CPU 15) and I/O ports. A hypernode can have fewer processors or fewer I/O ports present. In addition, a server's memory boards may be partially or fully populated with memory.

Each pair of processors connects to a port on the crossbar by means of a processor agent. This processor agent also serves as the

connection for the processors' associated I/O port, if one exists. The processor agent includes a specialized component called the DataMover, which is dedicated to moving data between memory locations. The DataMover helps increase I/O performance, can speed up memory accesses, and can be of significant benefit to programs that use message-passing libraries such as PVM and MPI.

Each I/O port can have up to three PCI controllers connected. The I/O ports are capable of directly transferring data to and from any physical memory in the system, including memory on other hypernodes. This eliminates CPU involvement in data transfers, allowing them to be dedicated to running user programs.

On the other side of the crossbar, each port has a memory controller that is the interface to physical memory. Also connected to each memory controller is a CTI controller (Coherent Toroidal Interconnect controller). The CTI controller is used in multihypernode X-Class servers to provide all hypernodes access to a single shared memory, which is distributed among all the server's hypernodes. The CTI controller is not used in S-Class servers.

The crossbar is nonblocking, so all ports can simultaneously operate at full bandwidth. (That is, all ports on the processor side can be served their requests for memory—without delay—as long as the requests are made to unique ports on the memory side.)

The SPP-UX operating system

SPP-UX is the operating system that runs on S-Class and X-Class Exemplar technical servers.

SPP-UX provides compatibility with the HP-UX 10.01 operating system: nearly all HP-UX commands (system utilities) are supported by SPP-UX, and HP-UX 10.01 applications can run on SPP-UX servers without being recompiled.

SPP-UX also includes features not in HP-UX, such as utilities for configuring and monitoring S-Class and X-Class servers, and features that permit parallel applications to execute efficiently on the hardware.

Figure 2 shows an overview of the Exemplar operating environment. The items shown in the shaded area are components of the SPP-UX operating system; the other items (applications, compilers, and tools) run on top of the operating system.

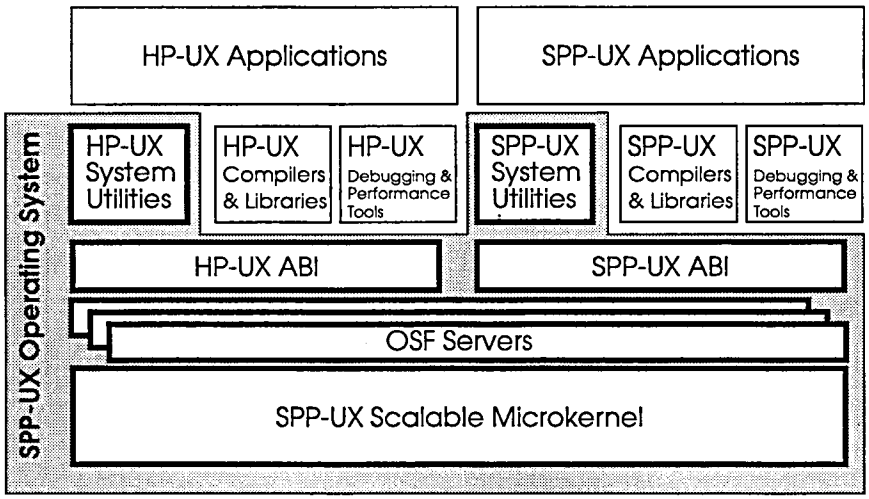


Figure 2 SPP-UX operating environment

As shown in Figure 2, the most basic SPP-UX functionality is provided through the microkernel. The microkernel is responsible for the scheduling of processors, virtual memory, and other operations. On S-Class servers, there is only one microkernel. On X-Class servers, each hypernode runs its own copy of the mach microkernel (thus providing improved performance by "scaling" the system's operations across hypernodes).

OSF servers run as needed to manage the file system, networking, subcomplexes, and processes. Other OSF servers can be added for additional functionality. On X-Class servers, OSF servers may be distributed across hypernodes to help eliminate resource bottlenecks.

The two ABIs (Application Binary Interfaces) shown in Figure 2 provide the mechanisms for handling system calls that are made by HP-UX and SPP-UX compilers, applications, and system utilities (such as `ls`, `ps`, `pot`, `sysinfo`, and other commands).

SPP-UX provides HP-UX system utilities as well as SPP-UX utilities. HP-UX utilities include nearly all HP-UX commands. A few of the specialized HP-UX utilities, such as the audio tools, are not appropriate on S-Class and X-Class servers (which do not have audio hardware). SPP-UX utilities provide features and commands not in the HP-UX operating system. Table 2 lists some of the utilities that distinguish SPP-UX from HP-UX.

Table 2 SPP-UX utilities not in HP-UX (partial listing)

SPP-UX command	Summary
`/usr/bin/mpa`	Modifies the attributes of a program for execution.
`/usr/bin/pot`	Displays and updates information about the threads on the system.
`/usr/bin/sod`	Displays format information about SOM and ESOM objects.
`/usr/bin/sysinfo`	Prints system information.
`/usr/bin/syspic`	Monitors performance of the system and provides real-time graphical representation.
`/usr/bin/chkpnt` `/usr/bin/restart`	Checkpoints (`chkpnt`) and restarts (`restart`) execution of a process or process family.

File system

This section describes the SPP-UX file system and summarizes where you can find various types of files.

The term *file system* refers to the organization of files and directories. The SPP-UX file system is essentially the same as that used by HP-UX, although some files (such as the kernel files) differ between the two operating systems, and SPP-UX has additional files, as shown in Table 2.

In SPP-UX, files are organized by category. Operating system files are kept separate from application files, configuration files are kept separate from executable files, private (local) files are separate from shared (networked) files, and areas of the file system that may change in size are separate from areas that remain a static size.

Table 3 lists the main directories found at the root level on all SPP-UX systems. System administrators may create additional root-level directories on individual systems, but the directories in Table 3 should be sufficient for nearly all purposes.

Table 3 SPP-UX root-level directories

Directory	Contents
/dev	Device files
/etc	Configuration files
/home	Users' home directories
/mnt	Mounting point for local file systems
/net	Mounting point for remote file systems
/opt	Optional products, including applications, compilers, and 3rd-party software
/sbin	SPP-UX system administration tools
/stand	SPP-UX kernel files needed at boot time
/tmp	Temporary "scratch" files
/usr	Most SPP-UX operating system files
/var	Variable-sized files, such as data and log files used by applications

The main directories important to users and programmers include the following:

/etc

The /etc directory contains mostly system configuration and administration files, and it does not include any user-executable files. However, the following files are of interest to all users:

/etc/skel/.cshrc
/etc/skel/.exrc
/etc/skel/.login
/etc/skel/.profile

These files provide the system default settings for users' configuration files. When a new user account is created these files are automatically copied to the new user's home directory.

/etc/PATH

This file contains the default system list of directories to search for executable programs.

/etc/MANPATH

The MANPATH file contains the default list of directories searched for man pages when the `man` command is used.

/etc/SHLIB_PATH

This file contains a list of directories where shared libraries can be found.

/home

This directory contains users' home directories. It is private (not networked) and is dynamic (its files may change in size). Users are free to use this area of the file system as they wish.

/opt

This directory contains applications, compilers, and other optional products (programs that are not required for S-Class and X-Class servers to operate).

Files in the /opt directory are organized by product. Executables are installed in /opt/*product*/bin, libraries are placed in /opt/*product*/lib, and so on.

/usr

The /usr directory includes a variety of subdirectories, including:

/usr/bin

Operating system user commands.

/usr/contrib

Unsupported software distributed with SPP-UX, such as `scinfo` and `scname`.

/usr/include

Header files.

/usr/lib

Object code and object code libraries.

/usr/local

Software contributed by local users.

For more details on the file system layout refer to the file /usr/share/doc/filesys.txt. A PostScript version of this file is available as /usr/share/doc/filesys.ps.

Subcomplexes

Exemplar S-Class and X-Class servers provide a way to balance the use of processors and memory by dividing the system's resources into units called *subcomplexes*. A subcomplex's configuration determines which processors and memory are available to programs that run on that subcomplex.

In effect a subcomplex is a "logical server" that contains a portion of the server's processors and memory. When several subcomplexes are established, a single Exemplar server can act as multiple smaller servers. Every S-Class and X-Class server has at least one subcomplex and may contain as many subcomplexes as there are processors in the system.

If a subcomplex contains more than one processor, multiple programs can run on different processors within the subcomplex, or an individual program can split up its tasks to run them in parallel.

While subcomplexes do share some hardware resources (such as I/O devices and the crossbar) each processor belongs to only one subcomplex and memory cannot be shared across subcomplexes.

Exemplar subcomplexes are created and managed by software. Only certain users, such as the system administrator, have the authority to set and change subcomplex configurations.

Subcomplex permissions

Each subcomplex has a set of permissions that are similar to Unix file permissions. The permissions (read, write, and execute) can be set to apply to individual users, groups of users, or all users.

Read permission allows for reading a subcomplex's configuration information. Write permission enables the subcomplex's process scheduling policies to be changed. Execute permission allows for processes to be run on the subcomplex.

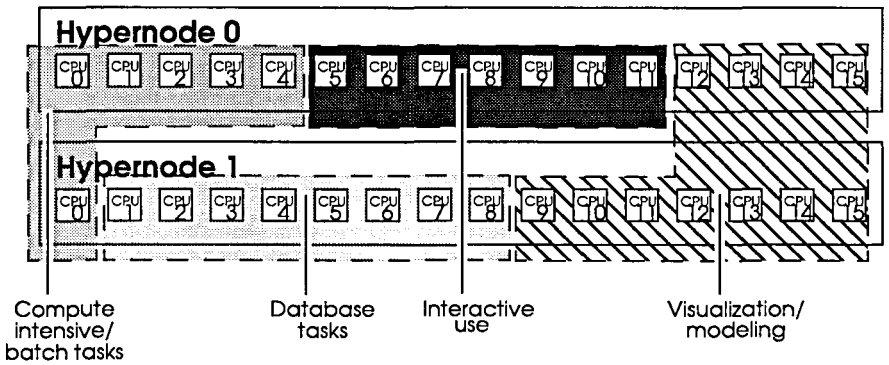
Reconfiguring subcomplexes

An Exemplar's subcomplexes can be reconfigured as needed to balance the loads on various parts of the server. When a subcomplex is reconfigured it may be assigned different processors, or it may be allocated more or less memory or processors.

In many cases a server can be reconfigured while it remains running; this permits the server to be tailored to users' needs without interruption.

Exemplar S-Class and X-Class servers can be configured in thousands of different ways to provide sufficient resources for various situations. Figure 3 shows two examples.

Daytime configuration



Nighttime configuration

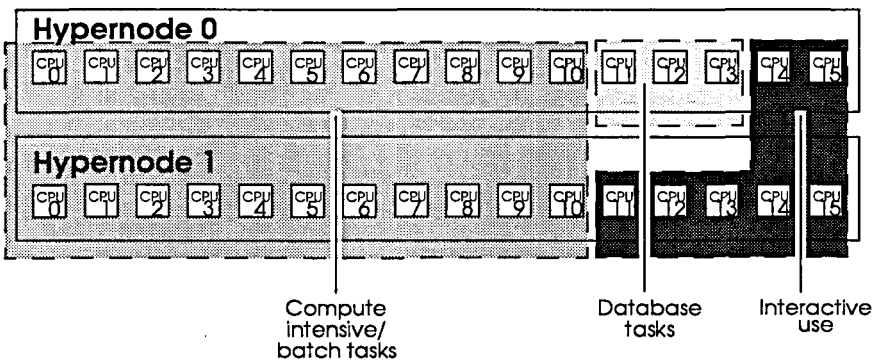


Figure 3 Sample subcomplex configurations for an X-Class server

In Figure 3, a 32-CPU X-Class server is configured for different purposes during the daytime and nighttime. Note that Figure 3 shows just the CPU allocation—but not the memory allocation—for the subcomplexes.

In this example, the daytime configuration provides 6 CPUs for intensive batch processing, 8 CPUs for database tasks, 7 CPUs for interactive use (general user access to the system), and the remaining 11 CPUs for visualization and modeling purposes. In contrast, the nighttime configuration dedicates 22 CPUs to intensive batch processing, provides a different set of 7 CPUs for interactive use, and has 3 CPUs for database tasks.

Memory

This section describes memory on Exemplar S-Class and X-Class servers. Three categories of memory are discussed: cache memory, physical memory, and virtual memory.

Cache memory, which is described in the following section, provides a faster way to read or write data that is expected to be handled in the near future.

Physical memory is the actual memory hardware. Virtual memory is managed by the SPP-UX microkernel and consists of the memory space available to programs. Physical and virtual memory are covered in sections that follow.

Figure 4 presents a simplified high-level overview of memory use on S-Class servers and X-Class servers. (Hypernode-local and global memory is covered in the section "Virtual memory".)

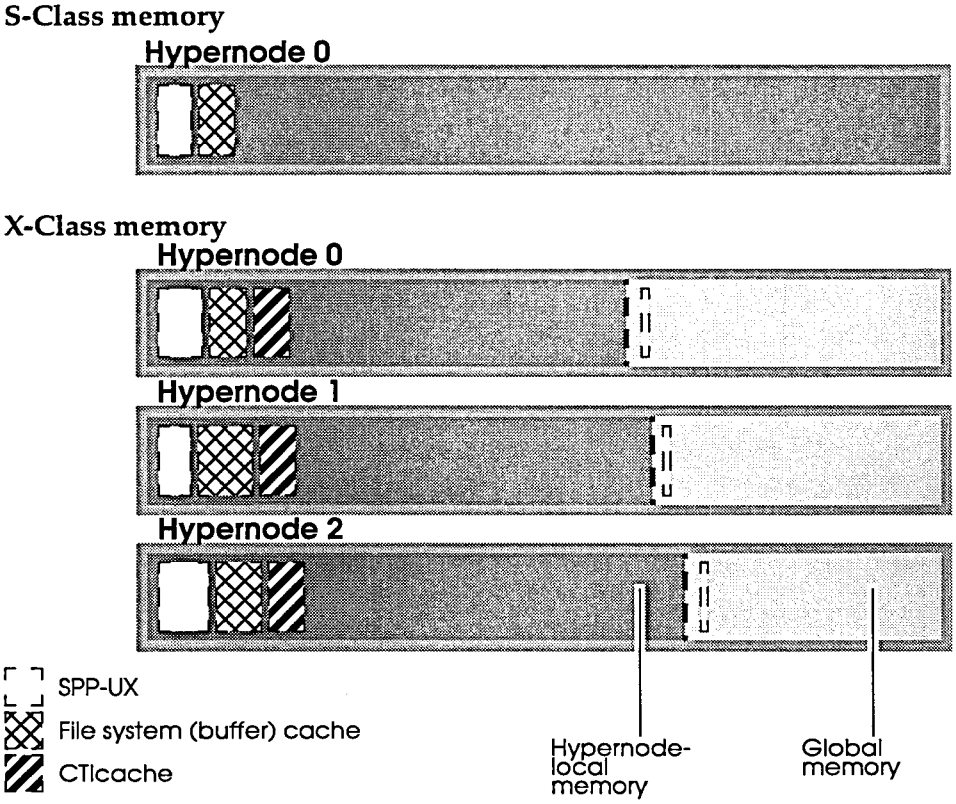


Figure 4 Overview of S-Class and X-Class memory

As shown in Figure 4, virtual memory on S-Class servers consists entirely of "hypernode-local" memory, part of which is occupied by the SPP-UX operating system and the buffer cache, and the remainder of which is available for user programs.

On X-Class servers, virtual memory is partitioned into "hypernode-local" memory and "global" memory. SPP-UX occupies some of the hypernode-local memory and a small portion of the global memory on each hypernode. The buffer cache and CTIcache also occupy portions of hypernode-local memory on each hypernode. The remainder of the virtual memory is available to user programs.

Cache memory

Cache memories are used to temporarily store data in an area—the cache—that provides faster access than the data's permanent storage area, such as disk space or, on X-Class servers, memory on a remote hypernode. Items that are encached (stored in cache memory) are typically expected to be requested for reading or writing in the near future.

Cache memory on S-Class and X-Class servers includes the file system cache and the processor caches. X-Class servers also have a CTIcache.

For detailed information about caches and their optimal use on S-Class and X-Class servers refer to the *Exemplar Programming Guide*. The following are brief descriptions of how cache memories are used.

- **Processor caches**—The processor caches are off-chip instruction and data caches. These caches are used by PA-8000 processors to reduce the latency of instruction and data fetches. The instruction and data caches are separate from the physical memory, and each may be from 1 MB to 4 MB in size.
- **CTIcache**—On multihypernode X-Class systems only, part of each hypernode's memory is dedicated as CTIcache. The CTIcache is used to encache global data fetched from other hypernodes.
- **File system cache (buffer cache)**—Each hypernode has part of its memory dedicated as a file system cache. The file system cache is used to encache items read from disk and items that are to be written to disk.

Physical memory

Physical memory is the memory hardware, connected to memory controllers, that makes up part of S-Class and X-Class hypernodes (see Figure 1). A hypernode can have 16 GB of physical memory when all memory banks are populated with 64-mbit SDRAM

memory modules (or up to 4 GB when 16-mbit SDRAMs are used).

Virtual memory

Virtual memory is the memory space as seen by a program running on an S-Class or X-Class server. There are two types of virtual memory—hypernode-local and global—each of which is divided into various classes.

When a process is started it is given a 4 GB virtual address space, which cannot be shared across processes and cannot be shared across subcomplexes, unless it is allocated using the `mmap` or `shmget/shmat` routines (see the man pages for mmap(2), shmget(2), and shmat(2) for details).

Depending on a system's configuration, the virtual memory space can be larger than the available physical memory space (with disk space making up the difference as necessary).

Hypernode-local memory and global memory

Access to hypernode-local memory is restricted to the part of a subcomplex that is on the hypernode where the memory resides. Hypernode-local data may exist as a single copy or may exist in multiple copies; see Table 4 on page 16 for details. On S-Class servers, all virtual memory is hypernode-local.

Global virtual memory is accessible throughout a subcomplex, including all hypernodes that make up the subcomplex. Global data exists in a single copy, which is distributed in different ways across hypernodes. On X-Class servers, part of the virtual memory is hypernode-local and part is global (unless 0 K is allocated for global memory).

Classes of virtual memory

Both hypernode-local and global virtual memory are divided into classes; each class provides a different type of data distribution.

Hypernode-local memory includes thread-private memory and node-private memory, two methods of sharing data within hypernodes. Global virtual memory includes near-shared, far-shared, and block-shared memory; these provide ways to share data across hypernodes.

Table 4 and Figure 5 describe and illustrate how the various virtual memory classes differ.

Note

Virtual memory classes are normally assigned automatically to variables by Exemplar compilers. Programmers can also designate variables' memory classes manually. See the *Exemplar Programming Guide* for details.

Table 4 lists local and global memory classes and descriptions of how data in those classes is distributed.

Table 4 Types and classes of virtual memory

Hypernode-local memory	Thread-private	Every thread of a process has its own private copy of the data.
	Node-private	A copy of the data is provided for each hypernode in the subcomplex. All threads in a hypernode share a copy of the data.
Global memory	Near-shared	A single copy is accessible by any thread on any hypernode in the subcomplex. Is located on one particular hypernode in the subcomplex.
	Far-shared	A single copy is accessible by any thread on any hypernode in the subcomplex. Is distributed in 4-K blocks across all hypernodes in the subcomplex in round-robin fashion.
	Block-shared	A single copy is accessible by any thread on any hypernode in the subcomplex. Is distributed in contiguous blocks equally among all hypernodes in the subcomplex. Must be dynamically allocated by the program at run time.

Figure 5 illustrates how data is distributed differently when it belongs to the various memory classes. This example shows how a 32-K variable occupies the memory space of a 10-processor subcomplex that spans 3 hypernodes.

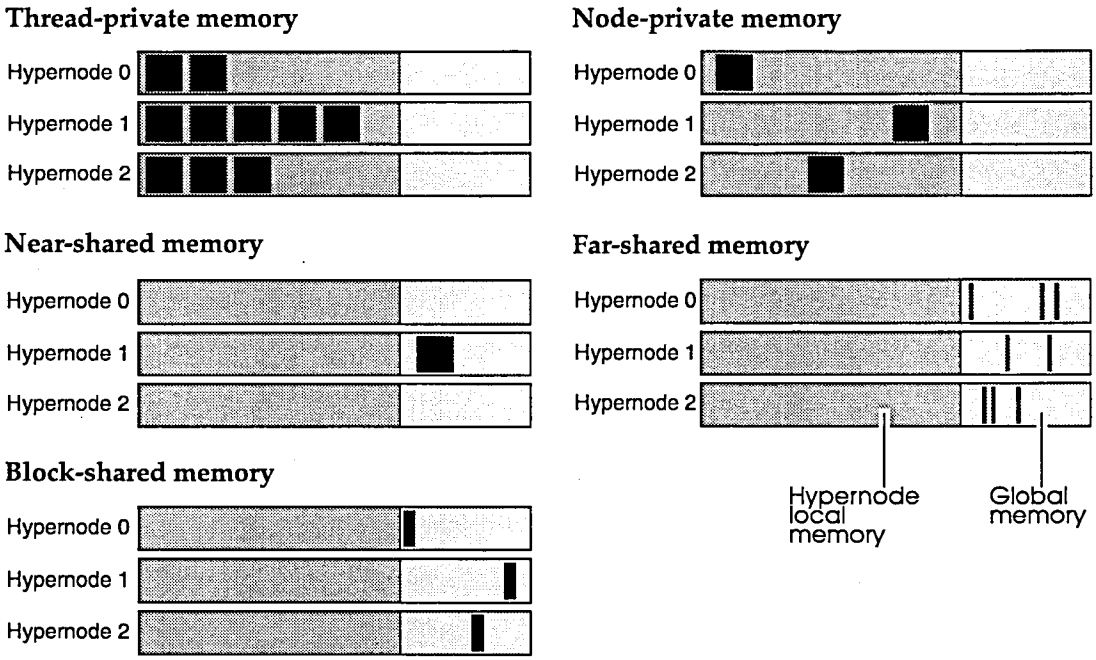


Figure 5 Classes of virtual memory

As Figure 5 shows, when the variable is assigned to thread-private memory every thread is given its own private copy of the variable, for a total of 10 copies that occupy 320 K. (This assumes the program runs 10-way parallel; if the program were run 5-way parallel there would be a total of 5 copies that occupy 160 K.)

Similarly, when the variable is assigned to node-private memory, every hypernode is given its own copy of the variable, for a total of 3 copies that occupy 96 K. (If the program were run across 2 hypernodes, there would instead be 2 copies that occupy 64 K.)

When the variable is assigned to near-shared, far-shared, or block-shared memory, only one copy exists for a total of 32 K, regardless of the number of threads or hypernodes involved in the program's execution. In these cases, the one copy of the variable is distributed across the hypernodes on which the program is run, as described in Table 4.

Determining the system configuration

3

This chapter lists the utilities and methods you can use to gather information about an Exemplar server's hardware and software configuration.

Utilities for monitoring system use are listed in Chapter 4. Programming languages, debugging tools, performance tools, and libraries are covered in Chapter 5.

SPP-UX utilities

The following utilities permit users to get information about Exemplar S-Class and X-Class machines.

Hardware and subcomplex utilities

SPP-UX provides utilities for getting information about an S-Class or X-Class server's hardware configuration. For examples of using these utilities see the section "Hardware and subcomplex configuration" on page 21.

Table 5 SPP-UX hardware and subcomplex utilities

Utility	Description
`/usr/bin/hostname`	Display the current host system's name
`/etc/ping`	Sends "echo request" packets to a host system; can be used to obtain a machine's IP address

Table 5 (continued) SPP-UX hardware and subcomplex utilities

Utility	Description
`/usr/contrib/bin/scinfo` `/usr/contrib/bin/scname`	Unsupported utilities for printing a subcomplex's name and other information
`/usr/sbin/scm`	Subcomplex Manager utility—provides full descriptions of the current system configuration
`/usr/bin/sysinfo`	Print system information, including processor, memory, SPP-UX, and subcomplex configurations

Software utilities

The utilities in Table 6 can be used to get information about software installed on S-Class and X-Class servers.

For examples of using these utilities see the section "Software configuration" on page 26.

Table 6 SPP-UX utilities for software information

Utility	Description
`/usr/bin/chatr`	View (or change) a program's internal attributes
`/usr/bin/file`	Determine file type
`/usr/sbin/mnm`	Print current SPP-UX memory utilization
`/usr/bin/mpa`	Print (or modify) a program's execution attributes
`/usr/bin/size`	Print section sizes of object files
`/usr/bin/sod`	SOM/ESOM object file dump utility
`/usr/sbin/swlist`	Display information about installed software products
`/usr/bin/sysinfo`	Print system information, including SPP-UX details, subcomplex configurations, and other details

Hardware and subcomplex configuration

This section includes examples and details for using SPP-UX utilities to get information about Exemplar S-Class and X-Class servers.

Server name and address

The `hostname` utility returns the name of the server. In the following example, the server's name is "abednego".

```
% hostname
abednego
```

You can use the `ping` utility to get a server's IP address. In the following example, `ping` is used to get information about the host abednego. In this case the server's name is "abednego", its full name is "abednego.x.hp.com", and its IP address is 15.99.219.13.

```
% ping abednego
PING abednego.x.hp.com: 64 byte packets
64 bytes from 15.99.219.13: icmp_seq=0. time=1. ms
64 bytes from 15.99.219.13: icmp_seq=1. time=1. ms
64 bytes from 15.99.219.13: icmp_seq=2. time=1. ms
64 bytes from 15.99.219.13: icmp_seq=3. time=0. ms
64 bytes from 15.99.219.13: icmp_seq=4. time=1. ms
^C
----abednego.x.hp.com PING Statistics----
5 packets transmitted, 5 packets received, 0% packet loss
round-trip (ms)  min/avg/max = 0/0/1
```

To stop `ping` from sending packets type **CTRL-C**, or use the `-n` option to limit the number of packets sent. For more details see the ping(1M) man page.

Number of processors

The `sysinfo` utility has two options for getting a count of the processors in an S-Class or X-Class server; some of the other `sysinfo` options are also covered in this chapter. For complete details see the sysinfo(1) man page.

`sysinfo -cpu_count` prints the total number of CPUs in the server. The following example shows the output when run on a 16-CPU server.

```
% sysinfo -cpu_count
16
```

`sysinfo -cpu` prints detailed information for every CPU in a server. Each CPU is listed on its own line, along with information

about which node it is on, whether the CPU is running and whether it is part of the SPP-UX mach kernel's master or server set, and which subcomplex it is assigned to.

This example shows a three-subcomplex server where CPUs 0-5 are in the "System" subcomplex, CPUs 6-11 belong to the "Math" subcomplex, and CPUs 12 to 15 are in the "Batch" subcomplex.

```
% sysinfo -cpu
node   cpu    flags                                  subcomplex
0      0      Master, Server, Running                System
0      1      Server, Running                        System
0      2      Server, Running                        System
0      3      Server, Running                        System
0      4      Server, Running                        System
0      5      Server, Running                        System
0      6      Server, Running                        Math
0      7      Server, Running                        Math
0      8      Server, Running                        Math
0      9      Server, Running                        Math
0      10     Server, Running                        Math
0      11     Server, Running                        Math
0      12     Server, Running                        Batch
0      13     Server, Running                        Batch
0      14     Server, Running                        Batch
0      15     Server, Running                        Batch
```

Memory availability and cache allocation

Memory on Exemplar S-Class and X-Class servers is used in part for cache memories. Also, on multihypernode X-Class servers, memory is split between node-private and global memory.

The `syspic` graphical system utility has facilities for tracking a server's memory use and buffer cache use. For details refer to the program's Help menu.

The `mnm` utility displays information about a server's memory utilization. It prints the total and free amount of hypernode local (node private) memory and global memory for each hypernode and shows the server's load averages. For a list of `mnm`'s command-line options enter `mnm -h`.

```
% /usr/sbin/mnm -h
/usr/sbin/mnm [-c <x> ] [-d] [-n] [-s <time>] [-p]
        -c <x>     - How many times to display
        -d         - Don't use Curses
        -h         - Display this help
        -n         - Don't display headers
        -p         - Print number of pages instead of Kbytes
        -s <time>  - Seconds to sleep, 0 = no sleep Can use decimal eg. 0.25
```

The `sysinfo` command has several options for getting details about an Exemplar server's memory configuration.

`sysinfo -memc` prints memory statistics for the entire server, including all nodes and all subcomplexes.

```
% sysinfo -memc
COMPLEX MEMORY     max    allocated     free
global              0M           0M       0M
node private     3686M         480M    3206M
buffer cache      409M
network cache       0M
total            4096M         480M    3206M
```

`sysinfo -memn` prints memory statistics for one node of a server. (On S-Class servers the `-memc` and `-memn` options provide the same information.)

```
% sysinfo -memn
NODE  0 MEMORY     max    allocated     free
global              0M           0M       0M
node private     3686M         479M    3207M
buffer cache      409M
network cache       0M
total            4096M         479M    3207M
```

`sysinfo -shmem` prints information about System V shared memory parameters; see the sysinfo(1) man page.

Subcomplex configuration and name

An Exemplar S-Class or X-Class server's processors and memory can be divided among one or more subcomplexes.

You can print a server's subcomplex configuration using several SPP-UX utilities. Each of the utilities (`scinfo`, `scm`, `scname`, and `sysinfo`) presents subcomplex information in different formats.

The `scm` utility also can show a graphical presentation of a server and its subcomplexes. See Table 7 for details.

Table 7 SPP-UX utilities for subcomplex information

Utility (and options)	Description
`scinfo`	An unsupported utility that prints information about the current subcomplex, including: • Subcomplex name • Subcomplex number • Number of CPUs in the subcomplex • Number of nodes the subcomplex spans
`scm`	Displays graphical representations of the server and subcomplex configuration. When invoked with no options, `scm` runs in interactive mode with a graphical user interface; for details refer to the scm(1) man page or the application's Help menu
`scm -c`	Prints a full description of the current system configuration, including the following for each subcomplex: • Subcomplex ID number • Subcomplex name • Subcomplex permissions (read, write, execute) • The user and group ID of the subcomplex's owner • Subcomplex scheduling policy • Subcomplex restrictions (if any) on memory use • List of processors in the subcomplex • The amount of global memory allocated for the subcomplex
`scm -s`	Prints the names of all subcomplexes for which the user has read permission
`scname`	An unsupported utility that prints the name of the current subcomplex
`sysinfo -cpu`	Prints the subcomplex assignment and other details for a server's CPUs. For each CPU the following information is printed: • Which node it is on • Whether the CPU is running • Whether it is part of the SPP-UX mach kernel's master or server set • Which subcomplex it is assigned to

Disk space

The `bdf` command displays the amount of free disk space available on a file system or set of file systems. This command works identically in both the HP-UX and SPP-UX operating systems. Two examples of the `bdf` command are given here. For complete information see the bdf(1m) man page.

In the following example, the `bdf` command is given the current directory, represented by a period (.), as its argument. This causes `bdf` to report usage statistics for the file system that contains the current directory. (In this example the current directory is /users/kaputnik, which is printed by the `pwd` command.)

```
% pwd
/users/kaputnik
% bdf .
Filesystem              kbytes     used    avail capacity  Mounted on
/dev/dsk/sd5a          1983928   340904  1444624     19%    /users
```

As shown above, the file system is at 19% capacity (it is 81% unused) and the mount point for the file system is /users. The file system has a total capacity of 1983928 kilobytes, of which 340904 kilobytes are used.

Note

The `bdf` command does not account for any disk space reserved for swap space. For this reason the amount of used and available disk space may not equal the file system's total capacity.

The example below gives a report of free space on all mounted file systems. When no arguments are given to `bdf`, reports for all file systems are printed.

```
% bdf
Filesystem              kbytes     used    avail capacity  Mounted on
/dev/dsk/sd0a           867078   519945   260425     67%    /
/dev/dsk/sd5b          1983928   189304  1596224     11%    /scratch
/dev/dsk/sd5a          1983928   340904  1444624     19%    /users
/dev/dsk/sd4a          7807920    91512  6935616      1%    /work1
/dev/dsk/sd3a          7807920  3612296  3414832     51%    /work2
/dev/dsk/sd6a          3967928  1418744  2152384     40%    /work3
```

Software configuration

This section describes ways to get information about the software installed on Exemplar servers. You can display software information including:

- The version number for SPP-UX libraries and executables
- A listing of the optional software installed on a server
- Settings for system tunables
- Other details about the size, properties, and contents of files, executables, objects, and libraries

SPP-UX version

To display the current operating system version use the `sysinfo` command. Entering `sysinfo` with no arguments displays the versions of: the SPP-UX kernel, the SPP-UX server, and the Architectural Interface Library (AIL—see the ail(3x) man page for details).

In the following example, the Exemplar server is running SPP-UX version 5.1.

```
% sysinfo
SPP-UX_mk      5.1 L34 tentacle:/OOW/OOW_5_1 [SPP1_FAST]
SPP-UX_server 5.1 L34 tentacle:/OOW/OOW_5_1 [SPP1_FAST]
SPP-UX_ail     5.1 L34 tentacle:/OOW/OOW_5_1 [SPP1_FAST]
```

The AIL provides interfaces to cache management, synchronization, the DataMover, and other features. The AIL version should match the kernel and server versions.

Library and executable versions

This section covers ways to display the version number for libraries and executable programs. For information about the various types of libraries see Chapter 5.

Some programs have a `-v` command-line option for printing the program version.

The `what` utility extracts and prints version information, if possible, from executable programs and libraries. The following examples show the `what` command printing information about the Fortran 90 library libblas.a (the 10.20.05 version) and the executable program `netscape` (Netscape 2.0, Motif version 1.2.2).

```
% what /opt/fortran90/lib/libblas.a
/opt/fortran90/lib/libblas.a:
        HP-UX libblas.a PA1.0 960802 (091335)  B3906AA/B3908AA B.10.20.05
```

```
% what /usr/local/bin/netscape
/usr/local/bin/netscape:
        Netscape 2.0/export, 23-Jan-96; (c) 1995,1996 Netscape Communications
Corp.
        OSF/Motif Version 1.2.2
```

See the what(1) man page for more information.

Installed software

Installed software includes all files, such as applications, compilers, libraries, and other optional products that are not required for Exemplar servers to operate.

You can use several methods for finding out what software is installed on an S-Class or X-Class server. This section briefly describes the `swlist`, `whereis`, and `which` SPP-UX utilities and recommends directories to inspect. If these methods do not answer your questions about installed software you can also contact your system administrator.

The `swlist` utility lists software products that were installed using the `swinstall` program. Most software packages are installed with `swinstall`, including SPP-UX and most compilers, libraries, and other optional products. For details refer to the swlist(1m) man page.

If you are interested in optional software regardless of how it was installed, you can use `ls` to list it. Listing the contents of the /opt, /usr/local/bin, and /usr/contrib/bin directories reveals what is in the primary locations for optional software products.

The `whereis` utility locates programs, man pages, and source files whose name you specify. `whereis` can find programs that are stored in directories not listed in the PATH environment variable. In the following example both the "syspic" executable and man page are located.

```
% whereis syspic
syspic: /usr/bin/syspic /usr/share/man/man1/syspic.1
```

When `whereis` reports more than one executable for a name, the `which` command can determine the one that would be executed. For details see the whereis(1) man page.

The `which` command searches for the file that would be executed if the specified name were given as a command. `which` displays the absolute path of the file or, if the name is aliased, displays the name's alias, as shown in the following examples.

```
% which scinfo
/usr/contrib/bin/scinfo

% which ls
ls:      aliased to ls -F
```

For more information see the which(1) man page.

Tunables information

SPP-UX system tunables control various components of the operating system. "Tunables" (tunable parameters) can be adjusted by settings in the file /os/tunables, which is maintained by the system administrator.

Tunable parameters control aspects of the server's behavior in different events, the manner in which system logging is performed, file and process services, and other SPP-UX features. For some tunables of interest to users and programmers see Table 8.

All tunable parameters have default values, which are listed in the tunables(4) man page. These values take effect unless a parameter is given a different value in the file /os/tunables.

Tunable parameters are established at SPP-UX boot time. For a complete list of SPP-UX tunables see the tunables(4) man page. Table 8 describes some of the user- and process-related tunables.

Table 8 SPP-UX system tunables (partial listing)

Parameter name	Description
buffer_cache_percent	The file system cache (buffer cache) for each node, specified as a percent of physical memory
dfldsiz	The default number of bytes in a process's data segment
maxdsiz	The maximum number of bytes in a process's data segment
maxfiles	The maximum number of files a process can have open at once
maxssiz	The maximum number of bytes in a process's stack
maxuprc	The maximum number of processes a user can have at once

Table 8 (continued) SPP-UX system tunables (partial listing)

Parameter name	Description
maxusers	The maximum number of simultaneous users
msgmax	The maximum number of queued messages a user can have

Device (disk), file, and directory sizes

To print the amount of available disk space on the current disk (or all available disks) use the `bdf` command. See the bdf(1m) man page or the section "Disk space" on page 25.

The `ls` command has options for listing the sizes of files. The `-l` option gives listings in long format, which includes files' sizes in bytes. In this example, `ls` reports that the file libU77.a is 64112 bytes in size.

```
% ls -l /opt/fortran90/lib/libU77.a
-r--r--r--   1 root     10          64112 Sep 24 15:30 /opt/fortran90/lib/libU77.a
```

See the ls(1) man page for more information.

The `du` command gives the number of 512-byte blocks allocated for the files, directories, or both that you specify. For example, `du` would display the following for the file libU77.a.

```
% du /opt/fortran90/lib/libU77.a
128     /opt/fortran90/lib/libU77.a
```

For libU77.a, `du` indicates it occupies 64K (128 512-byte blocks).

In the following example, `du` prints the amount of disk space the current directory occupies (including all files and subdirectories it contains). The `-s` option forces `du` to print only summary information. In this case the directory occupies a total of 2304K (4608 512-byte blocks).

```
% du -s .
4608    .
```

For more details refer to the du(1) man page.

File, object, and executable attributes

You can get information about files and executable programs using several SPP-UX utilities. These utilities can reveal, for example, whether a file is an ASCII text file, a source code file, an

executable program, a file in SOM or ESOM format, or whether it has some combination of these attributes.

Table 9 lists utilities for printing a file's attributes and information about its contents. The `chatr` utility and `mpa` utility also can change a file's attributes. For more details about the utilities in Table 9 refer to their man pages. For examples of the `chatr`, `mpa`, and other utilities see also the *Exemplar C and Fortran 77 Programmer's Guide.*

Table 9 SPP-UX file attribute utilities

Utility	Description
`chatr`	By default, prints a file's magic number and file attributes to standard output; also can be used to change a program's internal attributes.
`file`	Performs a series of tests on a file in an attempt to classify it. Classifications include ASCII text, language source file, SOM or ESOM executable file, and other classes.
`mpa`	Print (or modify) a program's execution attributes.
`nm`	Displays an object file's symbol table (the list of symbol names referenced or defined in the object file).
`size`	Prints the size of the text, data, and bss (uninitialized data) sections for an object file, along with the total size of the object file.
`sod`	Displays object files in a human-readable form. The `-a` option displays all headers, which contain information such as the minimum and maximum number of CPUs and the memory types.

Monitoring system use

4

This chapter describes ways to monitor how Exemplar S-Class and X-Class servers are being used. Getting information about the following is covered in this chapter:

- How heavily a server is being used
- When the server was booted (how long it has been up and running)
- What programs, processes, and threads are running
- How memory (including cache memory) is being used
- Which users are logged in

The available utilities are listed and described briefly in this chapter. You can find more detailed documentation in the man pages for the utilities.

Utilities for getting information about a server's hardware and software configuration are listed in Chapter 3. Programming languages, debugging tools, performance tools, and libraries are covered in Chapter 5.

Process, thread, load, and server monitoring

SPP-UX includes utilities for monitoring the programs running on an S-Class or X-Class server, as well as utilities for monitoring the amount and duration of a server's use.

Table 10 lists utilities for getting details about the processes running on a server, the threads running on the server, the server's average load over time, how long a server has been up and running, or combinations of these statistics.

Refer to the man pages for more details about the utilities in Table 10, including additional command-line options and ways to customize the output generated by the utilities. For detailed information about the `syspic` graphical performance monitor refer to that program's Help menu.

Table 10 SPP-UX process, thread, and load utilities

Utility	Description
`pot, top`	`pot` displays and updates information about the threads running on a server. Allows the user to specify which information is displayed and how it is sorted. `top` display and update information about the top processes on the system.
`ps, cnx_ps`	`ps` prints information about selected processes and has options to specify which processes and what information to report. `cnx_ps` includes two additional options: `-T` for providing thread information and `-s` for showing processes from a given subcomplex.
`sysinfo`	Provides a wide variety of system information. System load options include the following: • `sysinfo -lc` prints the mean load average of the entire system. • `sysinfo -ls` prints the load average for all subcomplexes, or for a particular subcomplex if its name is given.
`syspic`	Has a graphical user interface for displaying and, optionally, recording performance details. Recorded statistics can also be played back. Provided a way to monitor the system load and a wide variety of other activities.
`uptime`	Prints the current time, the length of time the system has been up, the number of users logged on to the system, and the average number of jobs in the run queue over the last 1, 5 and 15 minutes.

Memory monitoring

The `sysinfo`, `syspic`, and `mnm` SPP-UX utilities are available for monitoring memory and cache use on Exemplar servers. For details see the section "Memory availability and cache allocation" on page 22.

User information

This section lists commands for getting information about the users logged on to an Exemplar server.

Some system monitoring utilities, such as the `pot` and `ps` commands, print details about users' activities on a server. Additional SPP-UX commands give complete lists of the users on a system and, in the case of `w` and `who`, what they are doing.

The `users` command lists the login names of the users currently on the system in a compact, one-line format.

The `who` and `w` commands print information about users, such as the user name, terminal, login time, and current activity. Details are given in the who(1) man page.

The `finger` command lists details about a specified user, including the following:

- Login name
- Full given name
- Terminal write status (if write permission is denied)
- Idle time
- Login time
- User's home directory and login shell

For complete information see the finger(1) man page.

Programming tools, languages, and libraries

5

This chapter gives an overview of the programming languages, programming tools, and libraries available on HP Exemplar S-Class and X-Class servers. A short summary of programming strategies is also provided.

For detailed information about programming Exemplar servers refer to the *Exemplar Programming Guide*, the *Exemplar C and Fortran 77 Programmer's Guide*, the *HP MPI User's Guide*, the *HP PVM User's Guide*, or other documents listed in the "Associated documents" section of this book's preface.

Utilities for getting information about a server's hardware and software configuration are listed in Chapter 3. Utilities for monitoring system use are listed in Chapter 4.

Programming strategies

There are three methods for creating programs that run on Exemplar S-Class and X-Class servers: shared-memory programming, message-passing programming, and hybrid shared-memory/message-passing programming.

Shared-memory programming

In the shared-memory paradigm, the compiler handles most optimizations and, if requested, provides parallelization. Many compiler directives and pragmas are available to enhance this programming method.

The *Exemplar Programming Guide* is the primary book on Exemplar shared-memory programming.

Message-passing programming

The message-passing paradigm involves using functions, such as those in the PVM and MPI libraries, to explicitly spawn parallel processes, share data among the processes, and coordinate activities among them. There is no shared memory in this paradigm—data that is shared is explicitly passed between processes.

The *HP MPI User's Guide* and the *HP PVM User's Guide* are the main books about message-passing programming for Exemplar servers.

Hybrid shared-memory/message-passing programming

Combining the shared-memory and message-passing paradigms allows multiple shared-memory programs to coordinate activities via message passing. By using this approach you can write the majority of a program in the shared-memory style and exploit the benefits of message-passing programming as well.

The *Exemplar Programming Guide* contains examples of both styles of programming. See also the *HP MPI User's Guide* and the *HP PVM User's Guide.*

Programming languages and compilers

The four main programming languages available for Exemplar S-Class and X-Class servers are: Fortran 77, Fortran 90, C, and C++.

Each of these languages is distributed with compilation tools (compilers and related utilities), collections of libraries, and man pages and complete reference books.

Additional programming tools and libraries are also available, as discussed in the following sections ("Programming tools" and "Libraries").

For details about the Exemplar programming languages, refer to the "Associated documents" section for books covering the Fortran 77, Fortran 90, C, and C++ languages and related subjects.

Programming tools

This section briefly covers four tools for debugging and analyzing Exemplar programs: CXdb, CXpa, CXtrace, and XMPI. More information is available in the man pages for these products. See also these products' documentation, which is listed in the "Associated documents" section.

CXdb

CXdb can debug C, C++, and Fortran programs. It provides both a graphical user interface and a line-mode interface for working with programs being debugged. CXdb has many features, including support for the following:

- Debugging of executable programs, core files, and active processes—including multithreaded program debugging features
- Single point of control for debugging multiple processes created with MPI applications
- Views of source code, assembly-language code, process memory, process stack frames, contents of various registers
- Interactive input and output from processes
- Breakpoints and tracepoints for routines, lines, and instructions in a program—including conditional breakpoints
- Watchpoints to monitor address ranges
- Information on program arguments, variables, expressions, arrays, and memory
- Running, re-running, stopping and continuing, and killing programs and processes

CXpa

CXpa is a runtime performance analysis tool for Fortran, C, and C++ programs that run on Exemplar S-Class and X-Class technical servers. CXpa provides three interfaces: an X/Motif graphical user interface (GUI), a character-oriented tty interface (line mode), and a batch interface for integration with scripts and Makefiles.

You can run your application in line mode or batch mode to collect profiling data, then use CXpa's GUI to view a graphical analysis of the data.

CXpa measures a program's entire execution time and reports the total time spent in individual routines, loops, and parallel loops. This produces profiling results that are more complete than statistical sampling.

Using CXpa, you can collect the following performance data for routines, loops, and compiler-generated parallel loops in your program:

- Wall clock time
- CPU time
- CPU/wall clock time (parallel efficiency)
- Execution counts
- Dynamic call graph
- Cache miss counts and latency time for memory accesses

CXpa also includes the ability to:

- Analyze profiling data in 2D and 3D graphs or text reports
- Instrument libraries and object files for routine-level profiling with `cxoi`, a separate utility shipped with CXpa
- Profile MPI and PVM applications
- Clickback to source code during analysis
- View performance data for individual threads or summed across all threads of a process

CXtrace

CXtrace is a performance analysis tool particularly useful for analyzing MPI and PVM message-passing programs. It can be used with programs coded in Fortran 77, C, or both. CXtrace's instrumentation and monitoring systems capture and visualize execution trace information.

XMPI

XMPI is an X/Motif graphical user interface for running applications, monitoring processes and messages, and viewing trace files. XMPI provides a graphical display of the state of processes within an HP MPI application.

XMPI is useful when analyzing programs at the application level (for example, examining HP MPI data types and communicators). Unlike other profilers and debuggers, you can run XMPI without having to recompile or relink your applications.

XMPI runs in one of two modes: postmortem mode or interactive mode. In postmortem mode, you can view trace information for each process in your application. In interactive mode, you can monitor process communications and take snapshots while your application is actually running.

Libraries

Libraries are included as part of the HP Exemplar programming language environments. Additional, optional message-passing libraries (PVM and MPI) and mathematical routine libraries (HP MLIB) are also available.

This section briefly discusses the differences between shared and archive libraries. It also gives a brief introduction to the PVM, MPI, and HP MLIB collections of libraries.

Shared and archive libraries

Two kinds of libraries are available: archive libraries and shared libraries. Archive library file names end with `.a`, and shared library file names end with `.sl`.

When the linker maps a reference to an archive library routine, the resulting executable contains its own copy of the library routine referenced in the program. An executable that uses only archive libraries is a "complete executable".

When a shared library is used by the linker to resolve an external reference, the executable does not contain a copy of the library routine but instead contains a linkage table that has the routine's address. An executable that uses a shared library is an "incomplete executable" (and nearly always is *much* smaller than an equivalent complete executable).

Shared library routines are shared among all processes that use the library.

PVM and MPI message-passing libraries

PVM and MPI are both standard specifications for interfaces to libraries of message-passing routines. Both provide a portable way to create parallel message-passing applications.

The HP PVM and HP MPI libraries are available for creating message-passing programs that run on Exemplar servers. Tools such as XMPI are also provided to facilitate tuning programs that use these library routines.

For details refer to the PVM and MPI documentation listed in the "Associated documents" section.

MLIB math libraries

The HP MLIB set of math libraries includes three main components: Scilib, Veclib, and Lapack. These libraries include

Fortran-callable (and C-callable) mathematical routines that are optimized for HP Exemplar parallel servers.

Scilib provides a look-alike implementation of the Scientific Library portion of Cray Research Incorporated's UNICOS Math and Scientific Library V5.0.

Veclib provides mathematical software and computational kernels for application programs involving arrays. Veclib incorporates BLAS, LINPACK, EISPACK, and other sets of routines.

Lapack is a collection of Fortran-callable subprograms that provides mathematical software for application programs involving linear algebra. (Lapack is an acronym for "Linear Algebra PACKage".)

Refer to the documentation for these MLIB packages for more details.

Index

Symbols

A

B

C

D

E

O

P

R

S

T

U

V

W

X

B5655-90003