HEWLETT® PACKARD

HP Exemplar Technical Servers
S-Class and X-Class Systems Overview

Hewlett-Packard Exemplar Servers
S- and X-Class Systems Overview

Although the material contained herein has been carefully reviewed, Hewlett Packard Company (HP) does not warrant it to be free of errors or omissions. HP reserves the right to make corrections, updates, revisions or changes to the information contained herein. HP does not warrant the material described herein to be free of patent infringement.

Introduction 5

The Exemplar Technical Server Portfolio
- The Exemplar Technical Server Platforms 6
- S- and X-Class Systems 6
- Exemplar Roadmap 7
- HP-Intel Relationship 8

Exemplar System Architecture
- Introduction 11
- S-Class 12
- X-Class 18

I/O Subsystem
- Introduction 23
- Peripherals Subsystem 24
- Networking Subsystem 26

Exemplar Operating Environment
- Introduction 29
- SPP-UX Operating System 29
- Programming Environment 32
- Compiler Technology 34
- Application Development Tools 35

Packaging
- S-Class Packaging 41
- X-Class Packaging 42
- Scalability Options 42
- Index 43

Introduction

The Exemplar Technical Server Portfolio

The Exemplar Technical Server Portfolio from Hewlett-Packard addresses five major computing challenges that customers face today: compute, file, product data management, storage, and intranet applications. Each solution set in the portfolio is composed of systems, services, and software, allowing customers to tailor unique business solutions by selecting off-the-shelf portions of each of these server sets.

Supporting the Technical Server Portfolio compute servers are a range of platforms, ranging from the entry-level D-Class systems to the high-end X-Class supercomputing level systems. This document describes the architecture, system software, and tools of the high-end of the technical server family: the S- and X-Class. For more information on the Technical Server Portfolio please refer to the Hewlett-Packard Technical Server Product Brief.

The Exemplar family of scalable computing systems brings high performance to bear on solving problems that are too complex for workstations to solve. By using a combination of high-performance RISC processors, a high bandwidth distributed memory subsystem, and a scalable I/O subsystem, these platforms provide high levels of throughput for users, as well as reduced time-to-solution for large individual problems.

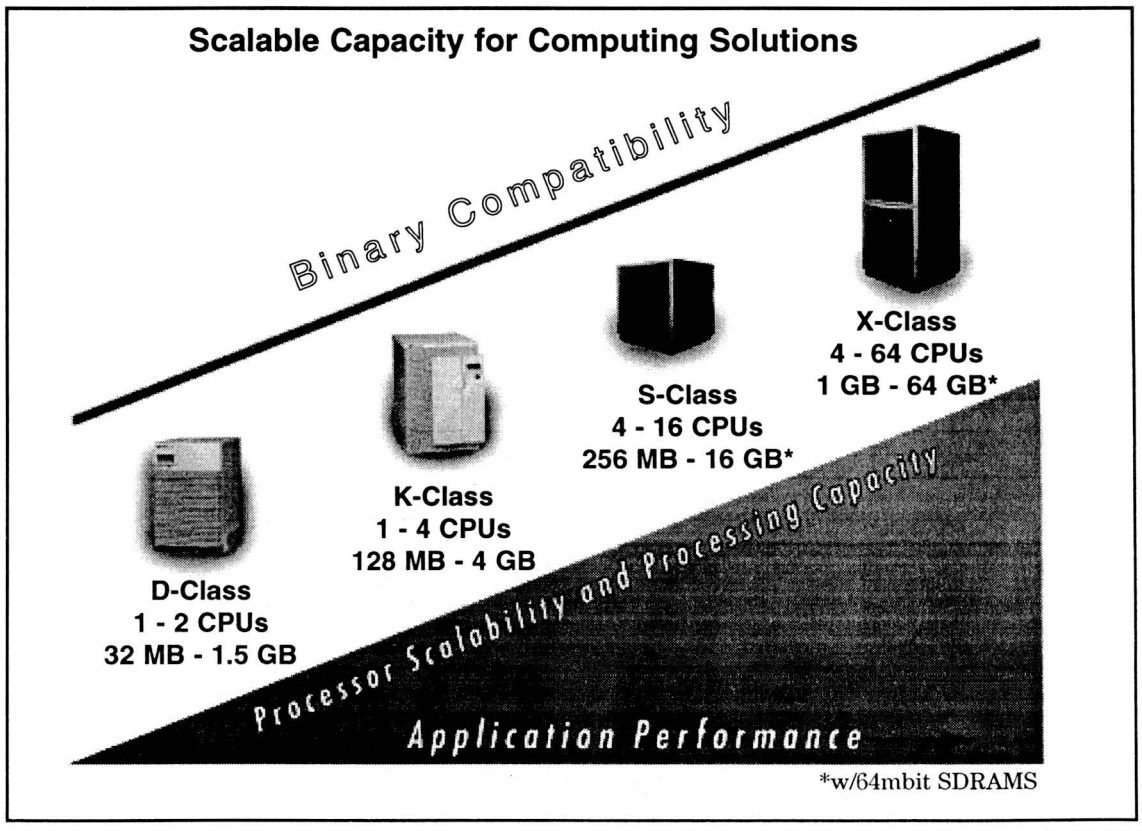

Figure 1-1. The Exemplar family of compute servers.

Through the use of components that are common with desktop systems, the Exemplar family exhibits price/performance similar to workstations and workstation servers. Through total compatibility—compatibility with user interfaces, application binary interfaces, and programming models—application software development and ownership costs are reduced. The systems fit neatly into an existing environment, and provide compute and file services for solving large problems in computer-aided engineering, petroleum exploration, computational chemistry and advanced research.

The Exemplar Technical Server Platforms

The Exemplar technical server product line, as shown in Figure 1-1, represents the broadest range of technical computing today. These systems feature:

- Complete up-and-down the line compatibility, permitting users to select the appropriate platform at the appropriate price-point without concern for application availability.
- The highest uni-processor performance in the industry, providing the highest levels of performance, including reduced time-to-solution and increased throughput simultaneously.
- Unmatched scalability–the ability to provide consistent price/performance over the entire range of each product line. This protects customers' investments in both hardware and software.
- Consistent programming model–all of these systems, regardless of the performance levels, present the same application programming environment. This greatly increases the number of "off-the-shelf" third-party applications that are available, and reduces porting and development costs.

S- and X-Class Systems

The Exemplar S- and X-Class platforms are the fourth and fifth generation of scalable Exemplar systems. The first generation, the SPP1X00 family, pioneered the use of a highly scalable operating system combined with a global shared memory (GSM) programming environment. The S- and X-Class systems retain full binary compatibility with previous SPP1X00 systems. For readers familiar with the SPP1X00 family, Figure 1-2 illustrates the magnitude of the improvements made in the S- and X-Class systems.

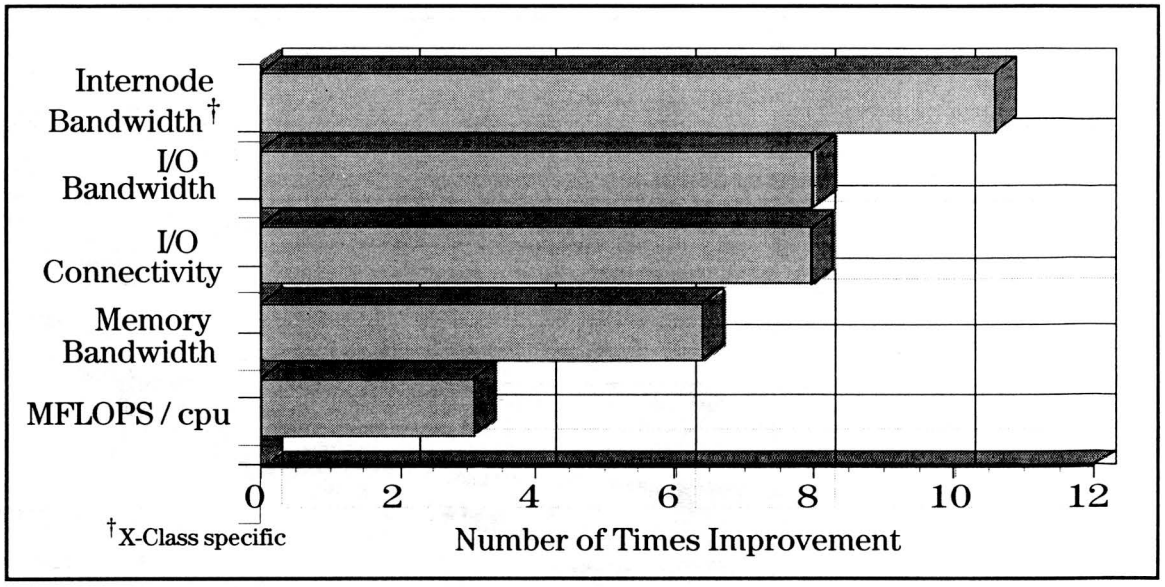

Figure 1-2. Comparison between the first generation and the S- and X-Class of Exemplar systems.

Exemplar Roadmap

Exemplar systems are able to encompass multiple generations of PA-RISC processors. Each Exemplar product generation is based on the latest generation PA-RISC processor—maintaining binary compatibility with previous generations while improving price/performance, lowering cost-of-ownership, and increasing functionality. Since 1994, customers have employed Exemplar systems in a variety of production and research environments.

All Exemplar system generations coincide with the initial availability of the newest PA-RISC architectures available (PA-7100, PA-7200 and PA-8XXX) and support an expanded system infrastructure to support the higher performance processors. Future generations of Exemplar systems will continue to track the latest processor architectures, including the Intel-HP processor architectures, and provide upward performance paths to customers.

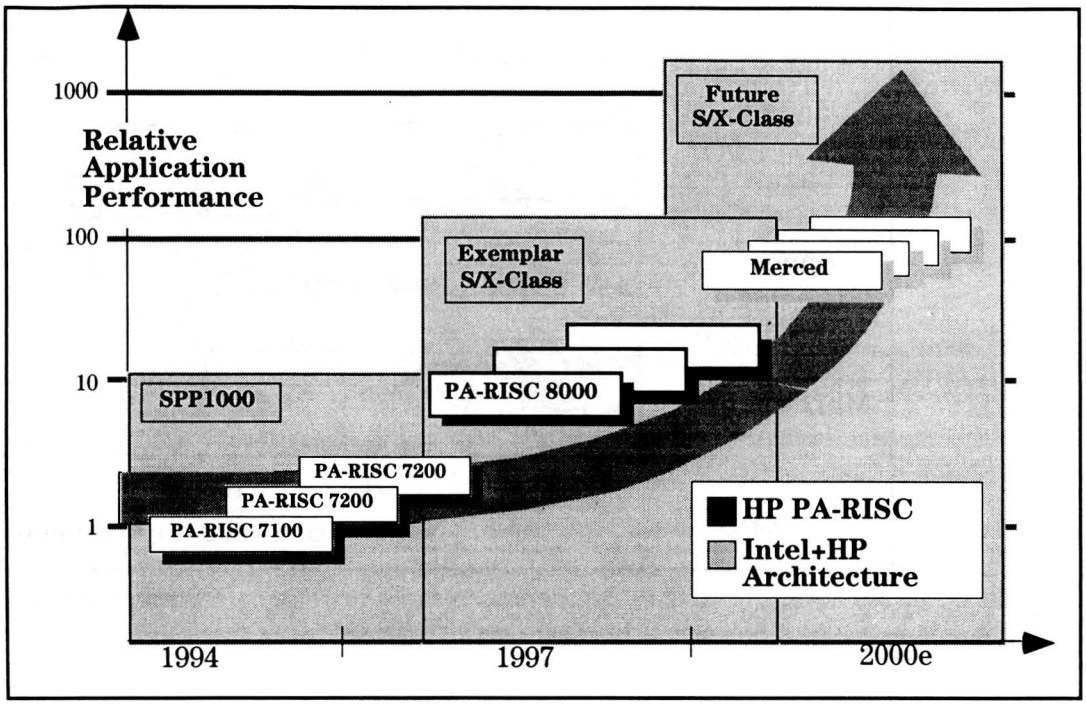

Figure 1-3. The Exemplar family protects customers' investment in technology.

Figure 1-3 shows how a customer's investment in the Exemplar product family is protected over time, while providing access to both evolutionary and revolutionary new technologies as they become available.

HP-Intel Relationship

On June 8, 1994, Intel and HP entered into a joint R&D program to collaborate in several areas. The program's intent is the development of technology which sets a new industry standard for processor performance. This effort, which involves collaboration on 64-bit microprocessor design, advanced compiler optimization techniques, and advanced high-performance integrated circuit (I.C.) process techniques, will enable state-of-the-art technology to be manufactured in an efficient and cost-effective manner. All of these elements are planned to come together to produce industry-leading performance by the end of the decade, well positioned for the demands of new classes of applications. Some of these applications can be characterized by new levels of interactivity in areas such as user interface-enriched multimedia communication, virtual reality, and low cost collaborative computer-based communications. And of course, it is naturally assumed that today's applications will achieve unsurpassed levels of performance when executed on systems based upon this new technology.

One element common to all technologies being evaluated is full binary compatibility with both HP PA-RISC and Intel processor families. This compatibility is a fundamental element of investment protection for customers with systems based upon both current and future processor families, and is an essential element of the collaboration. These technologies will provide customers with lower cost computing, pervasive and high-performing applications, and full enterprise interoperability.

Exemplar System Architecture

Introduction

The Exemplar S- and X-Class systems provide the benefits of high-performance scalable parallel processing (SPP) while maintaining a familiar programming model. This programming model allows the developer, compilers, and applications to view the system as a number of processors sharing a large physical memory and a number of high-bandwidth I/O ports. Logically, the system appears as a 4 to 64 processor MIMD (multiple instruction multiple data) shared memory system, as shown in Figure 3-1.

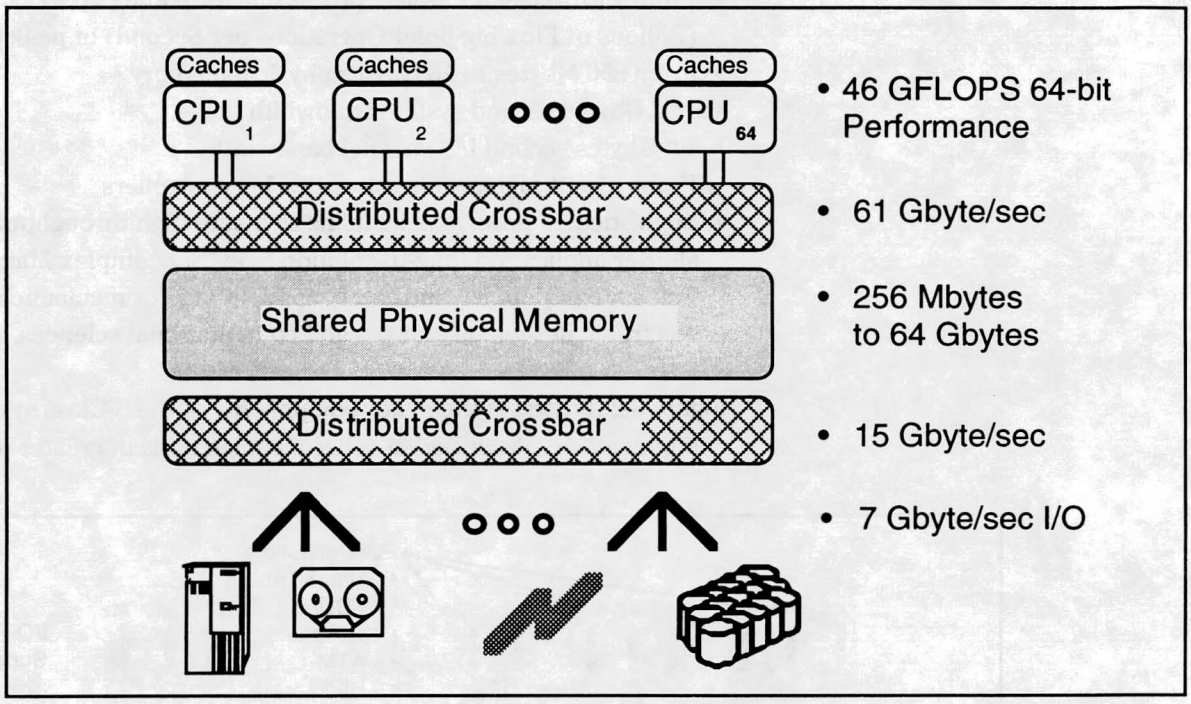

Figure 3-1. A view of the Exemplar programming model and user environment.

Underlying this view is sophisticated hardware and software that manages a globally shared memory (GSM) subsystem. GSM is a specific implementation of a CC-NUMA (cache coherent, non-uniform memory access) architecture. The GSM provides the scalability benefits of a distributed memory while maintaining the development environment of a shared memory system. In addition, the GSM automatically maintains complete data coherency throughout the system, relieving developers from the tedious chore of manually managing coherency. The GSM dynamically distributes data throughout the memory system, which increases application performance by eliminating excessive data movement.

S-Class

The mid-range Exemplar technical server is the S-Class system, a 4- to 16-way symmetric multiprocessor (SMP) with a highly scalable memory and I/O subsystem. The primary distinction. from SMP; between the S-Class system and a typical SMP system is the use of a memory cross-bar for data traffic from memory to/from the I/O system and processors. The use of a crossbar in the memory interconnect allows processors and the I/O subsystem simultaneous, non-blocked access to memory, which permits scalability not found in bus-based systems.

The S-Class system features:

- From 4 to 16 64-bit PA-8000 processors, providing up to 11.5 Gflops (Billions of Floating-point Operations per Second) of peak perfomance
- From 256 Mbytes to 16 Gbytes physical memory [1]
- 15.3 Gbytes/second system bandwidth
- 1.9 Gbytes/second I/O channel bandwidth
- From 1 to 24 high-performance PCI I/O controllers
- Sophisticated process scheduling to insure high throughput and shorter application time-to-solution (see "Subcomplex Management")
- Full suite of popular third-party applications for mechanical design, electronic design, oil and gas and computational sciences.

Figure 3-2 illustrates the major components of the S-Class system. These components are described in subsequent sections in greater detail.

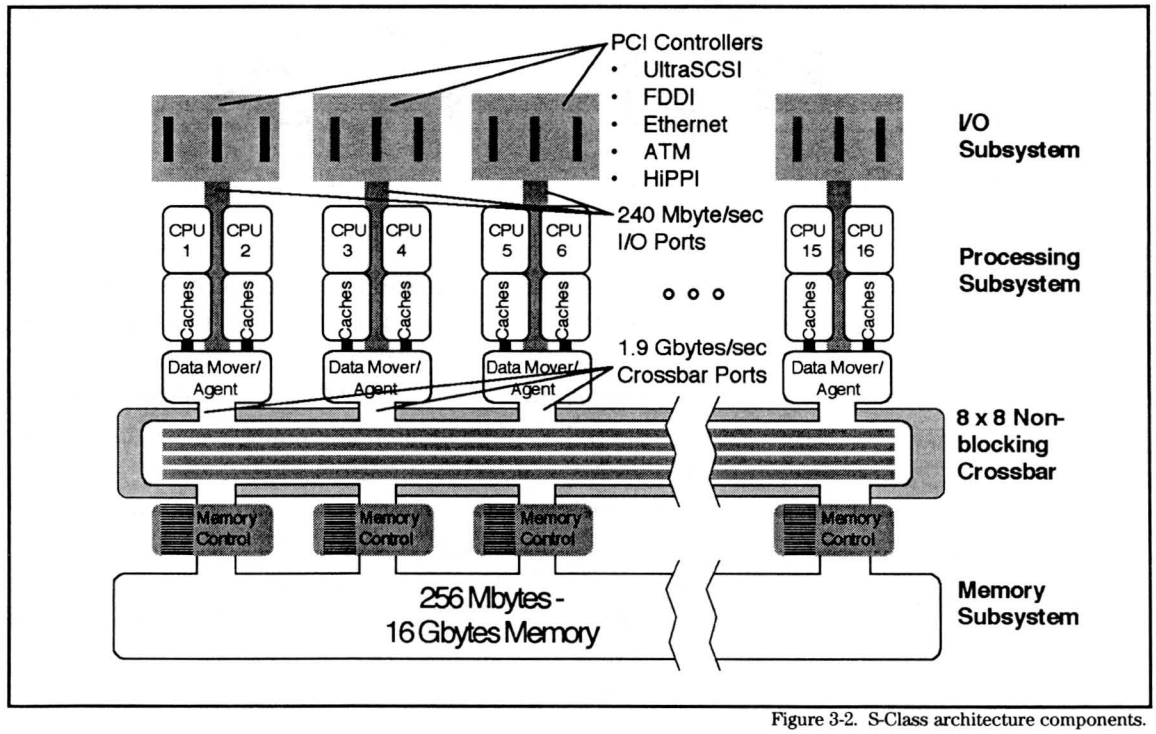

Figure 3-2. S-Class architecture components.

[1] *Configurations beyond four Gbytes will employ higher-density SDRAMS (64 Mbits), available in 1997.*

Crossbar

The S-Class memory subsystem is based on crossbar technology, which provides extremely high performance (15.3 Gbytes/second bandwidth) for a variety of applications. The crossbar provides non-blocking access from CPUs and I/O channels to the memory subsystem. The use of a crossbar prevents the performance drop-off associated with systems that employ a system-wide bus to handle memory and I/O traffic.

An S-Class system contains up to 16 processors, eight memory boards, and eight I/O channels with up to 24 PCI I/O controllers. Each of the eight crossbar ports on the processor side connects to a single *agent.* Each agent supports a pair of PA-8000 processors, a 240 Mbytes/second I/O channel and a DataMover (described in subsequent sections). On the memory side of the crossbar, each port connects to 4-way interleaved memory board.

The crossbar operates at 120 MHz (8.33 nanoseconds). The path width to the agent and memory controller chips is 64-bits. Thus the rated bandwidth of a crossbar port is 960 Mbytes/second, in each direction. The crossbar is non-blocking so that all ports can be operating at full bandwidth so long as their target ports are unique. It should be noted that for each of the 8 agents and 8 memory controllers there are two paths to the crossbar, an input and an output path. Thus, the aggregate bandwidth of the crossbar is 15.36 Gbytes/second.

Memory Subsystem

The S-Class system supports from 256 Mbytes to 16 Gbytes of SDRAM (Synchronous Dynamic Random Access Memory) physically distributed on two to eight memory boards (depending upon the packaging)[2].

The use of SDRAMs permits the memory subsystem to operate at a higher clock frequency than regular DRAMS, and at higher effective bandwidths. SDRAMs are increasingly being employed in systems at all levels (including PCs), resulting in a much higher performance memory subsystem at lower cost.

In order to easily explain the advantage of SDRAM we should first develop an abstract representation of standard DRAMs. We can consider DRAM to have two components: the DRAM core and the pins. An address is presented to the DRAM via the pins, the core is then accessed to get the requisite data, which are then supplied on the output pins. Immediately after the core is accessed, it must recycle. While the recycling is taking place, the DRAM cannot respond to any new requests received on the pins.

[2] *Configurations beyond four Gbytes will employ higher-density SDRAMS (64 Mbits), available in late 1997.*

For SDRAM we can add one additional component to the DRAM abstraction. Intervening between the pins and the core is a set of registers. Now when the address is presented to the SDRAM, it accesses the core but it extracts more information than there are output pins. This "extra" data is staged in the registers. After the core is accessed it begins to recycle, just as for standard DRAMs. While the recycling is taking place, the additional data that is staged in the registers can be extracted from them by "clocking" the registers. Thus the central notion is to amortize the expensive (in terms of time) access to the SDRAM core over the ability to get more data out.

Physically, memory is implemented using DIMM (Dual In-line Memory Module) technology. DIMM technology greatly improves memory performance by preserving memory interleaving as memory is added to the system in odd increments (non-powers of two). Memory interleaving is controlled by the number of memory boards, not by the number of DIMMs. Each memory controller provides 4-way interleaving so that a system configured with all eight memory controllers will provide full 32-way interleaving, regardless of the amount of physical memory supported.

Processing Subsystem

The S-Class system supports from 4 to 16 PA-8000 processors, an I/O subsystem, and specialized data movement hardware by means of modules called *Processing Resource Blocks* (PRBs). As illustrated in Figure 3-3, each PRB consists of two PA-8000 processors, an I/O channel, and a DataMover. These are explained in further detail below.

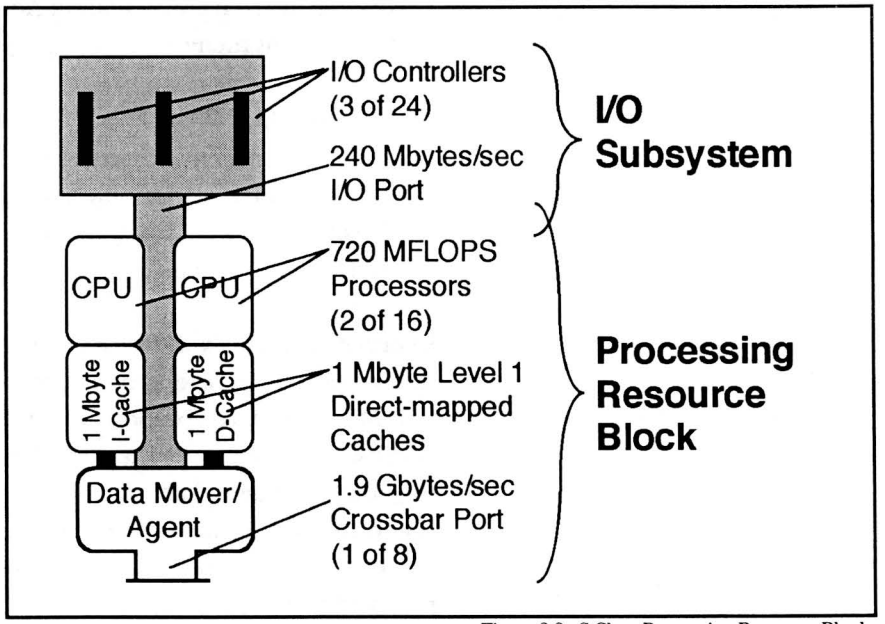

Figure 3-3. S-Class Processing Resource Block

PA-8000 Processor

The S-Class is designed from the ground-up for the HP PA-8000 processor, which offers 720 MFLOPS (Millions of Floating-point Operations per Second) of 64-bit superscalar RISC performance. This processor offers up to 4-way superscalar processing yielding four operations per clock and has the following architectural features:

- One Mbyte instruction and one Mbyte data primary caches. The architecture is designed to easily incorporate larger caches when available.
- Two 64-bit floating-point processing units each for multiply/add, divide/square root, integer, and shift/merge function (see Figure 3-4)
- Dual load/store units [3]
- 56 entry instruction reorder buffer (IRB)
- 10 outstanding memory requests
- Speculative execution
- Directed prefetch
- Static and dynamic branch prediction
- Three clock maximum latency to cache (one when pipelined)
- Dual-ported cache
- 64-bit addressing

The PA-8000 is the first chip to implement the PA-RISC 2.0 instruction set architecture. However, it retains full binary compatibility with the previous members of the PA-RISC family, which include the PA-7100 and PA-7200. The primary purpose of the 2.0 architecture is to support 64-bit integers and 64-bit addressing. Performance increases and new functionality are also an important and integral part of the new instruction set architecture. Some of the new features include variable size pages, new floating-point operations and better branching in both displacement and prediction.

The PA-8000 is a true 64-bit chip with native 64-bit integers. It supports a flat 64-bit virtual address space, although the chip exports 40-physical address bits. This corresponds to one Tbyte of directly addressable memory. In order to provide compatibility with previous members of the PA-RISC line, the chip supports a control bit to enable either narrow or wide address modes. In narrow address mode, only 32-bit physical addresses are used, thus providing backward compatibility with the PA-7100 and PA-7200 processors.

The PA-8000 is a decoupled architecture. This means that the instruction decode logic is not integrated with the functional units' pipeline logic. This allows the chip to partially decode instructions well in advance of the instruction's actual execution by the functional unit(s). Decoded instructions are staged in queues within the chip. The PA-8000 can have up to 56 instructions in progress at any given time.

[3] *Unique to the PA-8000*

The processor can issue up to four instructions per clock cycle. In order to sustain super-scalar performance to the greatest degree possible the PA-8000 incorporates two independent floating-point functional units, two independent divide and square-root functional units, two independent 64-bit integer ALUs, two shift/merge units (although separate from the integer ALUs, only two of the possible four instruction mixes for these units can be issued per clock), and two independent load/store units. In the important case of the floating-point functional units, each is capable of issuing a multiply/add instruction on every clock. Thus the peak floating-point performance of the chip is four times its clock rating of 180 MHz. Note that the multiply/add is a compound instruction–it is one instruction for which two floating-point operations are initiated. The multiply/add instruction has a three-cycle latency, but under pipelined conditions it delivers a result every clock.

The cache architecture of the PA-8000 is similar to the predecessors in the PA-RISC line. There are single level primary caches (one Mbyte instruction and one Mbyte data); there is no secondary cache. The data cache is direct-mapped using a write-back strategy. The caches are physically located off-chip in synchronous RAMs (SRAMS). The data cache can deliver data with a worst-case latency of three clock cycles, although this latency is usually hidden by pipelining and out-of-order execution (which is discussed below). The instruction cache can deliver four instructions per clock to the processor.

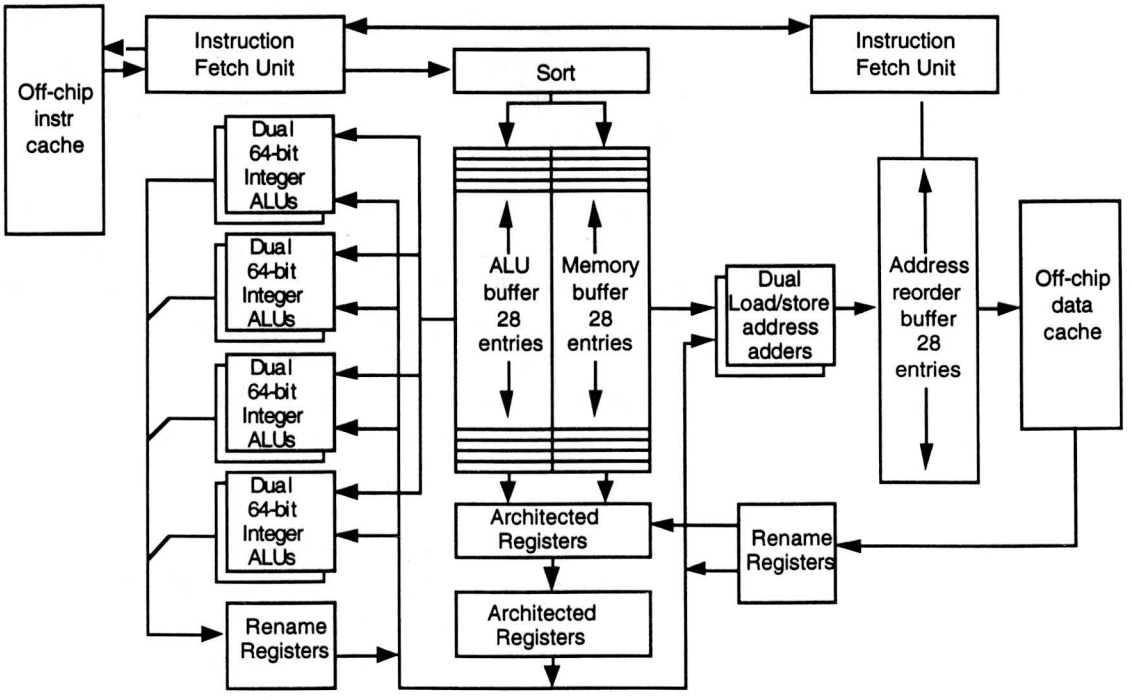

Figure 3-4. Overview of the Hewlett-Packard PA-8000 Proces.sor

In order to use the rich set of functional units, the processor contains a 56-entry instruction reorder buffer (IRB), a dual-ported data cache and the ability to fetch four instructions per clock from the large instruction cache. The processor can hold up to 56 instructions in the buffer and issue them when the requisite data and functional unit(s) are ready. The data dependencies for the instructions in the IRB buffers are known, and when the data and requisite functional units are available the instruction is dispatched to the functional units.

As a result of the decoupled architecture, the PA-8000 allows for out-of-order execution. Given that there are instructions in the instruction reorder buffer, they can be issued by the processor when registers, data and functional units are available. It is not necessary that instructions execute in strict program order, however, instructions are always retired in program order. (Instruction retire is the formal term for the completion of an instruction.) Leaving the actual fine-grained (i.e., instruction level) scheduling of instructions up to the processor is important when the processor permits both speculative execution and a large number of outstanding load requests. This is a complementary approach for CC-NUMA (Cache-Coherent Non-Uniform Memory Access) architectures such as the X-Class system. This means that the PA-8000 supports requests over the processor interconnect, concurrent with requests to data located within the local cache or memory. The result is a maximum amount of latency hiding.

A powerful feature of the PA-8000 is its ability to perform speculative execution. Speculative execution involves having the processor "guess" about the path of execution and execute those instructions in that path. If the guess is incorrect, the speculatively executed instructions are discarded. Speculative execution is performed in several cases. The first and most important case is on branches. The PA-8000 has a sophisticated mechanism to predict which path will be taken by a branch. The branch prediction indicates the instruction sequence (i.e., which branch it believes will be taken) to execute and those instructions are (speculatively) executed. If the branch is mispredicted those instructions are simply discarded since they have not yet been retired.

DataMover

To increase I/O and DMA performance, the S- and X-Class includes dedicated hardware for data movement between memory locations. This hardware is called the DataMover and is included with each processing subsystem as part of the processing agent (see Figure 3-3). The DataMover handles single-threaded data copies at a sustained 450 Mbytes/second. The DataMover also plays an important role in increased performance in Explicit Parallel Programming situations (See "Programming Models").

Support of the DataMover is included in the operating system and compilers through system calls, PVM (Parallel Virtual Machine), and MPI (Message-passing Interface) programming model libraries. Thus the user will realize immediate benefits of this new hardware implementation on the Exemplar architecture without changing source code or recompiling.

I/O Channel

Each PRB supports a single 240 Mbytes/second channel to the I/O system. In the S- and X-Class systems, each I/O channel supports a dedicated PCI bus. The I/O subsystem is described further in the section "I/O Subsystem".

X-Class

The X-Class system complements the S-Class, and extends the range of performance of these systems into the high-end supercomputing range. The X-Class system is based on multiple symmetric multiprocessing (SMP) units called "hypernodes". Each hypernode is essentially an S-Class system, as described earlier, with interconnect hardware to maintain the global shared view of memory and complete system cache coherency.

X-Class features include:

- From 16 to 64 64-bit PA-8000 processors, providing up to 46 GFLOPS (Billions of Floating-point Operations per Second) of peak performance
- A proven, tiered CC-NUMA architecture for scalability
- From 1 to 64 Gbytes physical memory [4]
- 61.4 Gbytes/second system bandwidth
- 7.6 Gbytes/second I/O channel bandwidth
- From 1 to 96 high-performance PCI I/O controllers
- Sophisticated process scheduling to insure high throughput and shorter application time-to-solution (see "Subcomplex Management")
- Full suite of popular third-party applications for mechanical design, electronic design, oil and gas and computational sciences.

The X-Class system is supported by a CC-NUMA memory subsystem, each level of which is optimized for data sharing. The first level consists of traditional SMP memory; the second level of the memory subsystem is created by tying first level memories through a high-performance interconnect. This interconnect, called CTI (Coherent Toroidal Interconnect), provides multiple rings in two directions (x and y) for high bandwidth and fault-resilience. The system thus becomes multiple symmetric multiprocessing hypernodes connected by the CTI, and appears to the developer and users and a single, globally shared-memory multiprocessor system.

[4] *Configurations beyond four Gbytes will employ higher-density SDRAMS (64 Mbits), available in 1997.*

This tiered memory subsystem is optimal for several reasons:

- The low-latency shared memory of a hypernode can effectively support fine-grained parallelism within applications, thus improving performance. This is often accomplished by simply recompiling the program with the automatic parallelizing compilers.
- The two levels of memory latency will likely be the model of the future: as the ability to increase the number of gates within semiconductors evolves, multiple CPUs will likely be placed on a single die. Thus, the first level of the hierarchy will be multiple CPUs sharing a common memory; the second level of the memory hierarchy will be "off-chip". Additionally, as multiprocessor workstations are introduced, clusters of these workstations will follow the same model.
- This system organization is a superset of a cluster of workstations, traditional experimental MPP systems, and SMP systems. Processors within a hypernode are tightly coupled to support fine-grained parallelism. Hypernodes implement coarser-grained parallelism with communication through shared memory and/or explicit message-passing mechanisms.

The Exemplar architecture is scalable up to 512 processors and 512 Gbytes of physical memory. The architecture is a building block for solutions requiring very high capacity and performance.

X-Class Interconnect (CTI)

A low-latency interconnect called CTI (Coherent Toroidal Interconnect) connects multiple hypernodes. CTI is derived from the IEEE standard 1596-1992, SCI (Scalable Coherent Interface). This interconnect combines high-bandwidth with low-latency to provide system-wide coherent access to shared memory. Explicit message-passing (EMP) applications also use the CTI for inter-hypernode communication.

CTI is implemented as a series of point-to-point unidirectional links. If we consider a single interconnect, say the x-direction, then it can be considered as a unidirectional ring. The ring is actually comprised of the eight independent data paths from each of the eight interface controllers in each hypernode. Each link between a pair of hypernodes for each of the eight paths is completely independent from all others, permitting interleaved access across the rings (similar to interleaved memory requests). An important performance issue is that the CTI controllers permit multiple outstanding requests.

Consistent with SCI the CTI uses a split transaction protocol. Consider an example in which a specific hypernode requests a cache line from a remote hypernode. The first use of the interconnect is to send a request packet out over the interconnect. (Without loss of generality it can be assumed both hypernodes are on the same ring). The request packet travels over the ring, passing through intervening CTI controllers on the non-target hypernodes in the path, until the target hypernode sees and removes the packet from the interconnect. The memory subsystem on the target hypernode then undertakes the actions needed to retrieve the cache line from its local memory. Concurrent with its local memory subsystem access is the sending, by the target hypernode, of an acknowledgment packet to the requesting hypernode. When the target hypernode has retrieved the cache line, it constructs a response packet that contains the data. This response packet is placed on the interconnect and is sent to the requesting hypernode. The final part of this transaction is the sending of an acknowledgment packet from the requesting hypernode to the sending hypernode. The "split" is that the request packet is separate and distinct from the response packet.

Owing to this split transaction protocol and the concurrency of the acknowledgment packets (which are "invisible" from a latency perspective), there is no notion of a nearest neighbor or of "hops." If the requesting hypernode and target hypernode are physically near each other in the downstream direction, then the request packet has a short path and the response packet a longer path. Likewise, if the two hypernodes are far apart from each other, in the sense of directional separation on the unidirectional ring, then the request packet has a long path and the response packet's path is short. Regardless, the combined path length of the request packet and the response packet is a full circuit around the ring.

Note that the X-Class system provides hardware support for global shared memory access, whereas a system without this feature can only emulate shared memory by moving pages from node to node under software control. This reduces the amount of overhead involved in parallel applications.

CTIcache

To minimize memory request latencies across the CTI, each hypernode contains a cache of memory references that have been made over the interconnect to other hypernodes. This is referred to as the CTIcache. Any data that has been moved from another hypernode into the processor cache on the hypernode, and is still resident in a processor cache, is guaranteed to also be encached in the CTIcache. Consequently the CTIcache directory information can be used to locate any global data that is currently encached by the hypernode. The CTIcache is physically indexed and tagged with the global physical address.

The system guarantees cache coherence between multiple hypernodes; two or more hypernodes that map the same global address will get a consistent view. This is done by maintaining a linked sharing list that contains a list of all the hypernodes sharing each cache line, or the hypernode that exclusively owns the cache line. The system keeps a record of which processors have encached each line in the CTIcache, so that interconnect coherency requests can be forwarded to the appropriate hypernodes containing the processors affected.

I/O Subsystem

Introduction

The I/O subsystem is highly scalable and distributed across the system. Each pair of processors within a hypernode supports a 240 Mbytes/sec interface to a 32-bit PCI subsystem. Each PCI interface is capable of supporting three PCI controllers as shown in Figure 4-1.

Each intelligent I/O port is capable of Direct Memory Access (DMA) transfers directly to and from any physical memory in the system–including memory in other hypernodes in the X-Class. This eliminates CPU involvement in data transfers, reserving them for user work. It also streamlines data transfers for such things as large disk blocks and high-speed network connections. As discussed in Section 3, these I/O operations are enhanced by the DataMover.

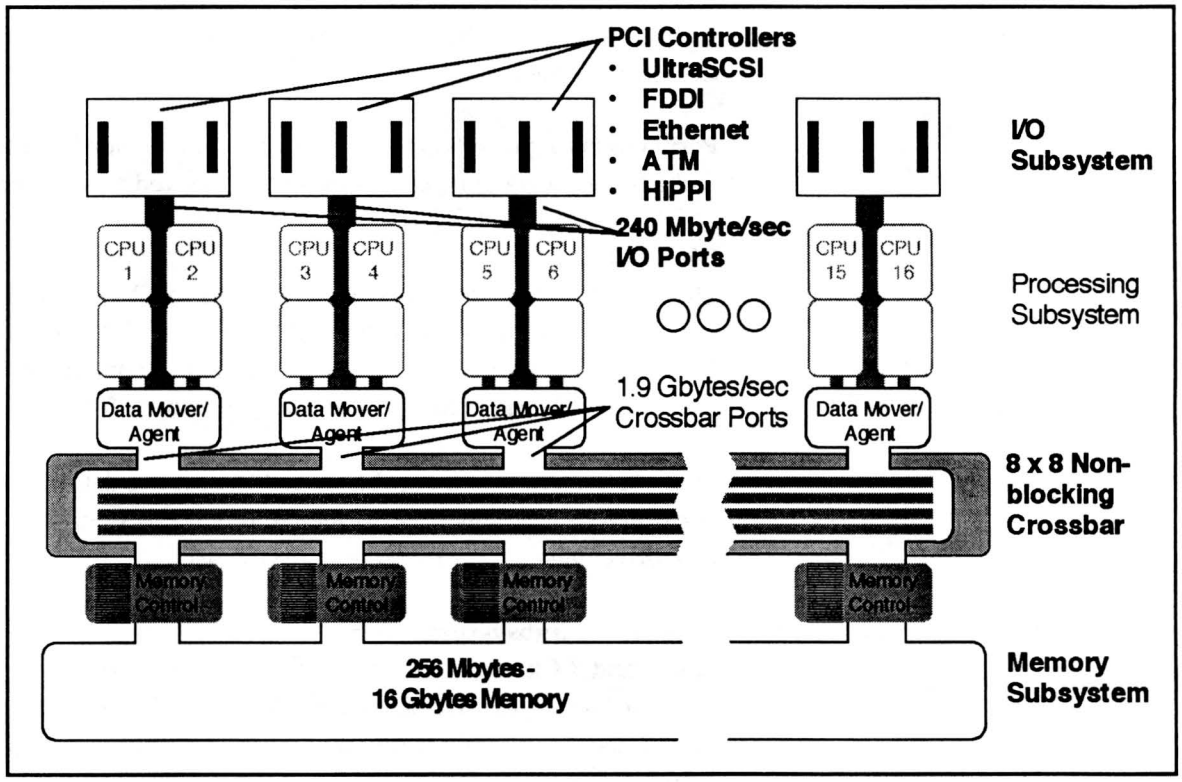

Figure 4-1. S- and X-Class I/O Subsystem

In addition, high-performance connectivity options, such as FDDI, ATM, and HiPPI, support direct connections to other servers and high-performance networks. The I/O subsystem supports a broad range of automated tape libraries as well as industry leading high-performance and high-capacity tape devices.

The I/O subsystem is physically distributed across any or all the hypernodes in the system. Over the interconnect and the crossbar, peripherals and network interfaces have access to any memory unit in the system. Similarly, any processor can access any filesystem or device mounted on any hypernode. Across all hypernodes in a fully configured system, the user has available an aggregate of 7.6 Gbytes/sec of DMA I/O bandwidth to system memory.

Peripherals Subsystem

The Exemplar peripherals are supported through industry standard PCI. The PCI Ultra SCSI host adapter provides the fastest possible SCSI interconnect currently available at 40 Mbytes/sec. This adapter provides an ideal interface to high-performance disk array subsystems and high-performance tape drives.

Peripherals supported on these host adapters include Ultra SCSI disks and a broad range of tape devices and automated tape libraries.

Disk Subsystem

The S- and X-Class disk subsystem provides reliable, cost-effective mass storage through disk arrays of 3.5-inch and 5.25-inch form factor Ultra SCSI disks. A fully configured system can support terabytes of disk storage for sites that require a large amount of storage.

Disks in the subsystem may be striped to improve data transfer performance. Striping is performed in the SPP-UX integrated stripe disk driver, and is thus transparent to the user.

Disk Array Subsystem

The S- and X-Class systems support a high-density disk array subsystem for high availability. The array includes up to 20, high-density, SCSI-2 disk drives for a maximum of 64 GB storage capacity. The subsystem supports:

- Redundant components to eliminate single points of failure: controllers, disks, power supplies and fans
- RAID 0, 1, 1/0, 3, and 5 levels supported in the same configuration for maximum flexibility

- Dual-active storage controllers offering automatic failover to significantly improve data availability
- Global hot spare capability for unattended disk rebuild, more economical than conventional dedicated spare drive approach
- Mirrored write cache protected by battery backup for data integrity in the event of a power failure
- User-configurable disk rebuild rate to minimize impact on I/O performance
- Convenient on-line maintenance for hot repair of drives, storage controllers, power supplies and fans

In addition, through the use of system-level striping, multiple disk arrays may be striped for higher performance.

Tape Drives and Automated Tape Libraries

The S- and X-Class systems are excellent platforms for applications that require movement of massive amounts of information to/from off-line and nearline storage. The S- and X-Class systems support popular and industry-standard tape transports, automated cartridge systems (ACS), and legacy tape transports for existing media. These devices are supported through UltraSCSI PCI controllers on the S- and X-Class I/O channels, and include:

3480/3490 18-Track cartridge tape transport. The 3480/3490 transport is compatible with the IBM 3480/3490 tape subsystem. The transport mounts in the 19" Exemplar external peripheral rack and supports an optional automatic cartridge loader (ACL). The 3480/3490 transport is appropriate for accessing legacy data sets, system backup, and/or transferring data from other systems.

StorageTek Timberline 9490 Cartridge Subsystem. The TimberLine transports support high-performance 36-track tapes for batch processing, hierarchical storage management, and system backup. TimberLine is a component of StorageTek's NearlinePlus performance-based tape library strategy.

StorageTek Wolfcreek 9360 Automated Cartridge Subsystem. The Wolfcreek automated cartridge system (ACS) is an entry-level robotic tape library system used in conjunction with one or more Timberline transports to provide operatorless access to multiple terabytes of data.

StorageTek Redwood D3 Transport. Redwood is a high-capacity, high-performance helical scan tape transport that supports very high density (10, 20 and 50 Gbyte) tapes with high transfer rates.

StorageTek PowderHorn Automated Cartridge Subsystem.
PowderHorn is a high-end automated tape cartridge library that supports enterprise-wide information management functions. The PowderHorn ACS may be used in conjunction with the Redwood transport subsystem to provide access to 20 to 300 Tbytes of data.

Digital Linear Tape DLT4700 Subsystem. The DLT4700 subsystem is a mid-range digital linear tape subsystem with a stacker that is ideal for fully automated backup and recovery and economical near-line storage. The transport includes a 1.5 Mbytes/sec data transfer rate per drive (can be doubled to 3.0 Mbytes/sec assuming 2:1 data compression); up to four drives running in parallel; HP patented robotics for leading library performance; and up to 1.9 Tbytes capacity (assuming 2:1 data compression); and bar code scanning to facilitate unattended backups.

Networking Subsystem

S- and X-Class networking options are designed to meet the connectivity needs of a broad range of users. Users requiring connections to the most commonly used interfaces such as 10 Mbits/sec Ethernet, and FDDI can seamlessly integrate the systems into their network infrastructure.

High-performance network interfaces such as 155 Mbits/sec ATM and 100 MBytes/sec HiPPI offer balanced, high bandwidth networking options for distributed applications and global access to corporate information.

The systems offer an additional feature called Scalable Networking. As more network interfaces are added to keep pace with system and user demands, the architecture guarantees that network throughput will scale with this increased capacity.

Ethernet

The S- and X-Class systems support one or more industry-standard Ethernet connections via a single PCI controller. This 100Base-T product uses a single RJ45 unshielded twisted pair (UTP) LAN connection that will autosense and operate at either 100 Mbits/sec using 100BaseT or 10 Mbits/sec using 10BaseT. This product can be used as a normal 10BaseT Ethernet adapter when connected to a 10BaseT hub, or it can be used as a 100 Mbits/sec 100BaseT adapter when connected to a 100BaseT hub.

FDDI

The I/O system supports industry-standard FDDI (Fiber Distributed Data Interface). FDDI offers the next level of performance beyond Ethernet in local area network (LAN) standards, with a theoretical peak transfer rate of 100 megabits per second (Mbps). FDDI can span up to two kilometers between network nodes and can support up to 500 connections on a network, and cover a total distance of 100 kilometers. On-board station management is fully integrated on the host adapter. The Exemplar implementation is a single-attached fiber optic controller using a single PCI interface.

ATM

ATM (Asynchronous Transfer Mode) is an emerging communications technology that provides an increase in network speed and capacity compared to conventional network technologies. ATM improves the performance of today's network applications and provides a scalable roadmap for bandwidth increases required for future network growth.

The I/O system implements a 155 megabits per second OC-3 ATM interface using one PCI controller card. Switched virtual circuit capability providing classic IP services are fully integrated into the scalable networking infrastructure.

HiPPI

HiPPI (High Performance Parallel Interface) is an ANSI standard (X3T9.3/88), very efficient, simplex point-to-point link capable of transferring data over copper twisted-pair cables at up to 800 megabits (100 megabytes) per second for distances of up to 25 meters. Data is transferred in bursts of 1 to 256 32 bit words. One or more bursts make up a packet, while one or more packets make up a connection. HiPPI interfaces are used primarily as a high speed networked data channel between computer systems.

Exemplar HiPPI offers a peak bandwidth of 800 Mbits/sec. HiPPI functionality includes HiPPI-LE, HiPPI-FPM, HiPPI-PH and HiPPI-SC capabilities. SPP-UX IP services are layered on to this base HiPPI infrastructure, providing *ftp* and *rcp* utilities. Both utilities support the larger block sizes necessary for high performance file transfers across SPP-UX HiPPI. SPP-UX HiPPI also supports NFS (Network File System) so users can share a common base of files across different systems on the network.

Exemplar Operating Environment

Introduction

The goal of the Exemplar operating environment (EOE) is two-fold: to provide broad availability of high-performance, scalable applications, and to flexibly deploy the power of the system when and where it is required in the enterprise. The development capabilities that HP provides are designed to automatically parallelize even legacy codes (the so-called "dusty deck" FORTRAN programs in particular). The toolset also empowers the programmer to develop applications which are optimized for the architecture.

To facilitate the porting of existing applications, and the development of new ones, the Exemplar environment provides both familiar shared-memory and explicit message-passing (EMP) development paradigms. The result is a single programming model that supports two programming styles that can be combined if desired. The availability of both styles means that developers can port applications in a manner which is conducive to the application. The programming model for Exemplar systems provides a full range of support for the underlying architecture. For the X-Class system, the CC-NUMA architecture can be fully exploited with easy-to-use programming extensions. Fully automatic parallelization, combined with the hardware's ability to dynamically distribute data, provides immediate benefit to existing C and FORTRAN shared memory programs. PVM (Parallel Virtual Machine) and MPI (Message-passing Interface) message-passing is provided for existing programs written in that style. Programmers can utilize features supporting explicit parallel thread management, explicit synchronization and explicit data distribution.

SPP-UX Operating System

SPP-UX Architecture

The S- and X-Class operating system, SPP-UX, is a scalable, micro-kernel-based operating system designed specifically to provide high performance to a wide variety of applications in an Exemplar environment. Applications may be single-threaded HP-UX binary executables, moderately parallel applications compatible with common SMP systems, or highly parallel applications. Applications may employ shared-memory and/or message-passing parallelization to increase performance. Figure 5-1 illustrates the application support layers of SPP-UX.

On X-Class systems SPP-UX runs a microkernel on each hypernode. The microkernel provides fundamental kernel functionality such as virtual memory and scheduling of processors. The majority of the operating system is implemented in server tasks and in protected modules that run in user space. These components provide the standard functions of a full-featured operating system, such as file system management, device management and network services. The operating system components communicate through special mechanisms optimized for the Exemplar hardware.

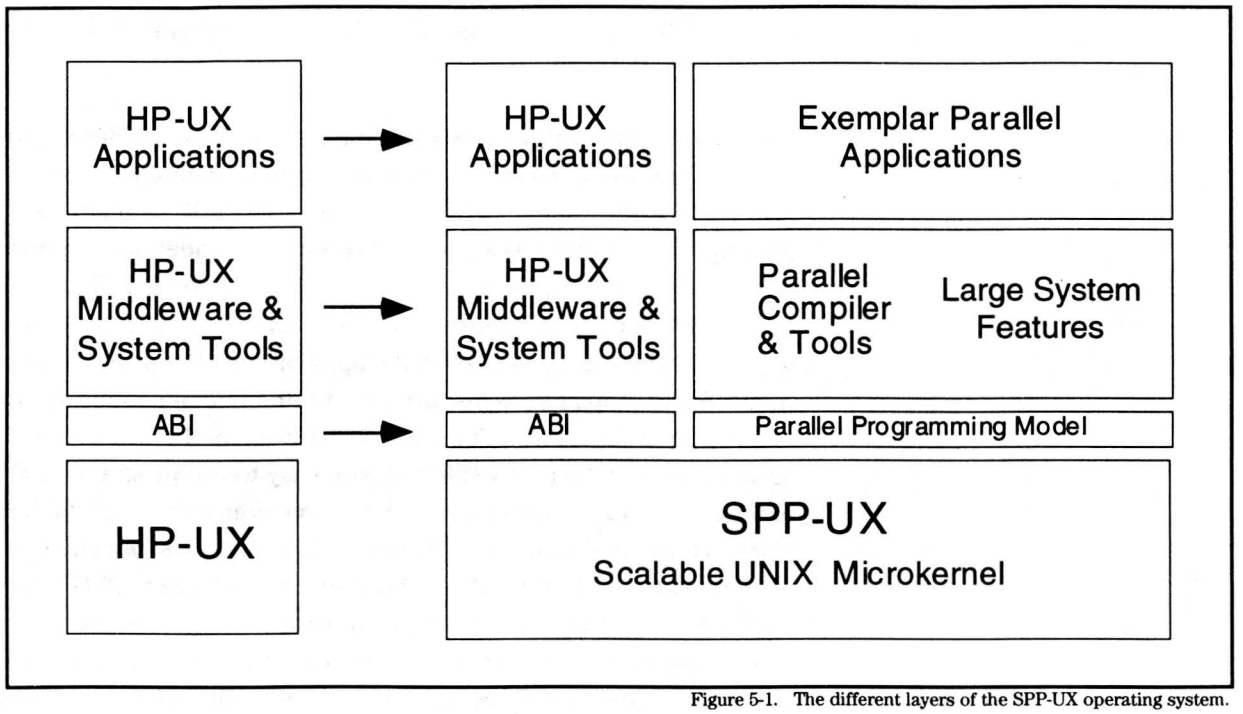

Figure 5-1. The different layers of the SPP-UX operating system.

This approach has multiple advantages:

- Scalability. The components of the operating system are distributed and replicated across the machine. This allows the operating system to scale with system requirements, avoiding a common bottleneck to performance.
- Modularity. By splitting operating system functions into distinct modules, new features, performance enhancements or personalities may be added.
- Performance. Employing multiple servers provides additional performance.

For example, running a file server on each hypernode allows I/O operations to take place in parallel across the system.

SPP-UX has many features that satisfy the needs of data centers and enterprise systems, including 64-bit features like a large file system (which supports files and/or filesystems of one Terabyte in size) and 64-bit data types and pointers. Other features include batch job scheduling, checkpoint/restart, and subcomplex management—a powerful, flexible resource allocation capability.

Sub-Complex Manager

One of the unique features of SPP-UX is subcomplex management. As shown in Figure 5-2, subcomplexes control the allocation of processor and memory resources on an Exemplar system. Subcomplexes are defined and managed by the system administrator through a GUI-based subcomplex manager or automated scripts using a UNIX-based command-line interface. The subcomplex manager allows the system administrator to graphically view the subcomplexes, change them, and experiment with new configurations on-line.

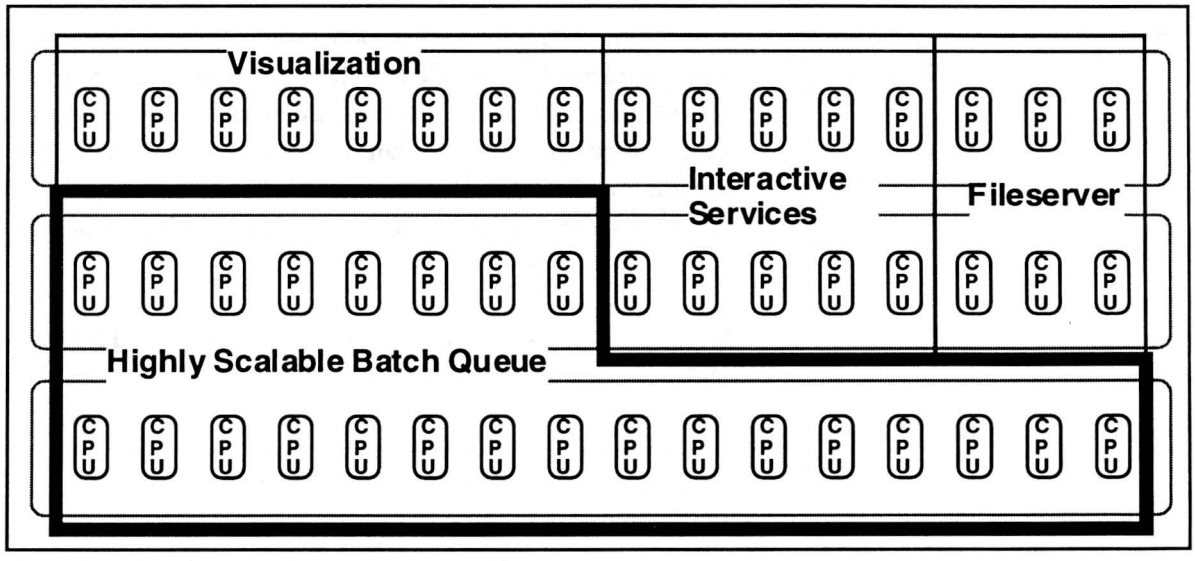

Figure 5-2. The Subcomplex manager allows flexible partitioning of resources.

The use of the subcomplex manager allows the system manager to efficiently allocate processors and/or memory to executing processes.

Benefits include:

- Subcomplexes may be configured to allow multiple workgroups or departments to fairly share the machine. This facilitates the sharing of the machine during periods of peak load. During off-peak hours, resources that might otherwise go idle may be redistributed to users that are able to take advantage of them. This provides customers with greater problem solving power while keeping cost-of-ownership low.
- Subcomplexes are dynamically configurable. Quiescent processors may be reassigned to another subcomplex without rebooting the system.
- Subcomplexes may be defined which separate interactive time-sharing processes with other processes, for example, a batch queue dedicated to highly parallel jobs. The result is excellent response time for a large number of interactive sessions while dedicating other resources to important batch-type functions.

Programming Environment

Shared-Memory Parallelism

Shared memory parallelism (SM) is beneficial when the application can be broken up in small pieces (fine grain parallelism). Shared memory parallelism embodies the characteristics of conventional serial programming, enhanced to provide parallel execution. A programmer wishing to use SM parallelization to increase the performance of an application has only to recompile the program with one of the automatic parallelizing compilers. The SM style is best suited for porting existing serial or shared-memory parallel programs and for developing new programs in a shared memory style.

When parallelization options are invoked, the S- and X-Class compilers will automatically parallelize data-independent loops and Fortran 90 array expressions. Parallelization is achieved by dividing the work of the loops among threads of execution, one per processor in the subcomplex. Code generation is such that a process automatically adapts itself to the number of processors available to the user at runtime—the number does not have to be supplied to the compiler.

When a program compiled for SM parallelization begins execution, there is one thread of execution per available processor. The number of processors may be specified by the user or assigned by the run-time libraries at the time the program is invoked. Initially all threads are idle

except for thread zero, which executes alone until it encounters a parallel construct. It then activates, through procedure calls to the compiler's run-time libraries, all the idle threads in the process. If the threads are not currently executing on a processor, the microkernel will schedule them for execution. At the end of the parallel construct the threads synchronize at a barrier, and all threads except thread zero go idle.

The biggest benefit to shared-memory parallelization is that it is often easy to accomplish, and often requires no modifications to source code whatsoever. The HP Exemplar compilers are able to detect loop-level shared-memory parallelization and generate proper parallel applications.

Explicit Message-passing Parallelism

Message-passing parallelism is most beneficial when the data within the application can be divided up and the computation which occurs on the divided data is large. An explicit message-passing (EMP) program is typically many single-threaded processes, each executing on its own processor. The processes coordinate their mutual operations and share data values through explicit message-passing. The user specifies the communication of data values between the processes using a message-passing library (e.g., PVM and MPI). Within a subcomplex, message-passing is done through shared memory, providing extremely high bandwidth and low latency communication between processes. Across subcomplexes and to other machines, PVM uses conventional UNIX sockets. EMP programs are inherently parallel, and unless explicitly coordinated by message-waiting, all processes execute independently. In a conventionally coded message-passing program, all variables are private to each process. Synchronization among the processes occurs explicitly through message-passing.

The message-passing libraries on the Exemplar systems are PVM (Parallel Virtual Machine) and MPI (Message-passing Interface). Both are accessible from C and FORTRAN programs. Each library determines the fastest means of communication, either shared memory or sockets, based on where the communicating processes are executing.

Explicit message-passing through standard libraries has the advantage of running across multiple, often heterogeneous, systems (since the standard libraries are supported on many different platforms). The disadvantage is that the source code must necessarily be modified, and parallelism must be determined by the developer at a high level.

Hybrid SM and EMP Parallelism

Hybrid programs are also supported such that message-passing can be performed among several multi-threaded, shared memory programs. This permits programs that have been designed for EMP, and written to use a defacto standard EMP library, like PVM, to be immediately moved to the S- and X-Class. The individual EMP processes may themselves be parallelized, resulting in extremely high-performance EMP applications.

Compiler Technology

Introduction

The S- and X-Class systems support popular programming languages and extensions, including FORTRAN 77, Fortran 90, C and C++. The compilers allow the user to migrate applications from other platforms and develop new applications to take maximum advantage of the S- and X-Class architecture. The goal of the Exemplar compilers is to increase the overall performance of the application, while maintaining high productivity and application portability. As shown in Figure 5-3, the compilers provide a hierarchy of automatic optimizations, ranging from instruction-level optimization through coarse-grained task-level parallelization. The lowest level of optimization is at the code generation level. Here, as much scalar performance as possible is extracted using advanced RISC optimization techniques, including instruction scheduling, software pipelining, and tiled global register allocation. Data locality provides additional single CPU performance by performing loop blocking to improve cache utilization, as well as classic machine-independent optimizations.

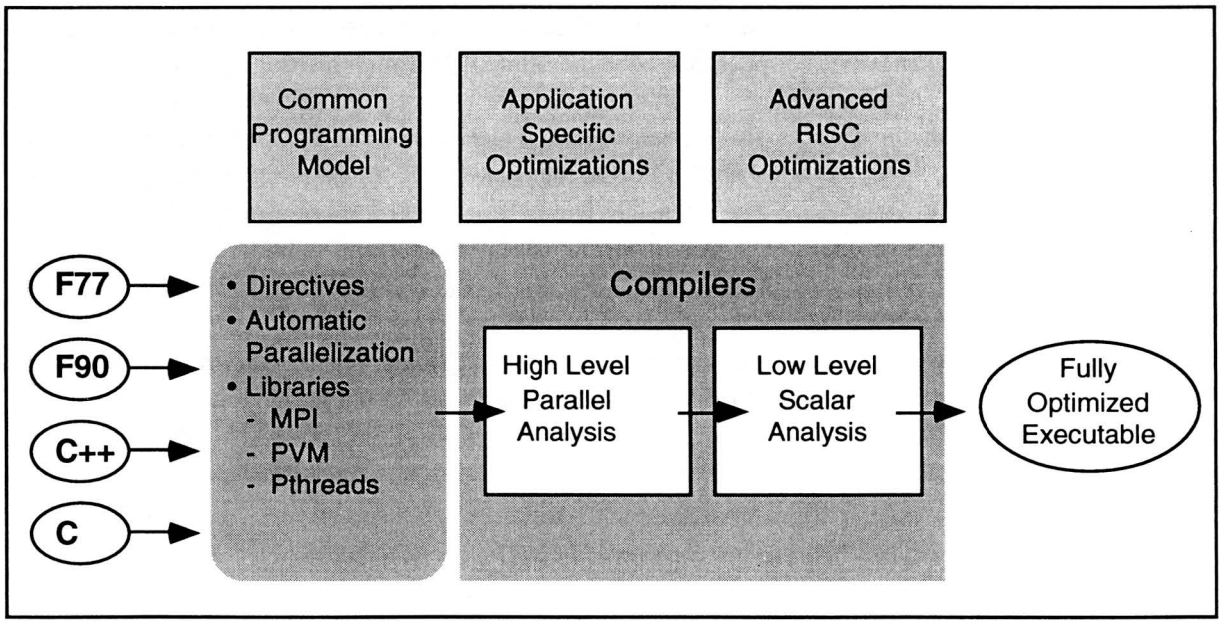

Figure 5-3. Exemplar compilers architecture.

The next level of performance is achieved by compiling for automatic parallelization, which enables loop-level parallelization on data independent loops. Typically, this form of parallelism results in as many threads of parallelism for a particular process as are allowed at the time of execution. The compilers also support directives/pragmas which allow the user to explicitly control parallelization and data distribution. For example, a directive may be used to indicate that a loop should be run with one thread per hypernode allocated. This is used to parallelize an outer loop in which each iteration of the loop could spawn additional threads for an inner loop, providing multiple levels of parallelization.

The Exemplar S- and X-Class FORTRAN compiler adheres to the American National Standard programming language FORTRAN 77, X3.9-1978, ISO 1539-1980(E). The default language interpretations are FORTRAN 77 with Exemplar extensions and certain features of the International FORTRAN Standard, ISO/IEC 1539.1991, which is identical to the ANSI Fortran 90 programming language, ANSI X3.198-1992.

The Exemplar S- and X-Class C compiler supports full ANSI C as well as the Kernighan and Ritchie derivative of the language. The compiler performs extensive loop-level, scalar, and machine specific optimizations, including global register allocation and instruction scheduling. The Exemplar S- and X-Class Fortran 90 Compiler adheres to the American National Standard programming language Fortran 90, X3.198-1992 and ISO Programming Language Fortran, ISO/IEC 1539.1991. The supported language includes Exemplar extensions for enhanced performance and compatibility with other vendors' extensions.

The Exemplar S- and X-Class C++ compiler is a complete implementation of the C++ programming language described in "The Annotated C++ Reference Manual" by Ellis and Stroustrup. The supported language includes Exemplar extensions.

Application Development Tools

Introduction

CXtools is a complete and powerful set of tools that aid programmers in debugging, testing, and optimizing highly parallel applications. This high-productivity tool kit includes three visually oriented products: CXdb, the parallel debugger; CXpa, the parallel performance analyzer; and CXtrace, the parallel event analyzer.

CXDB

CXdb is a full-featured debugger for serial and parallel applications, and permits robust and correct debugging of fully optimized code. CXdb contains a graphical user interface and numerous features that support debugging at the source code level, and is capable of debugging multi-threaded and optimized applications. Application source statements are internally mapped to the associated optimized machine instructions, which prevents erroneous debugging of event locations. CXdb maps multiple threads within a parallel application on a single source or disassembly window. Activity within either of these windows is indicated by highlighting the active expressions or statements within the source code, as shown in Figure 5-4.

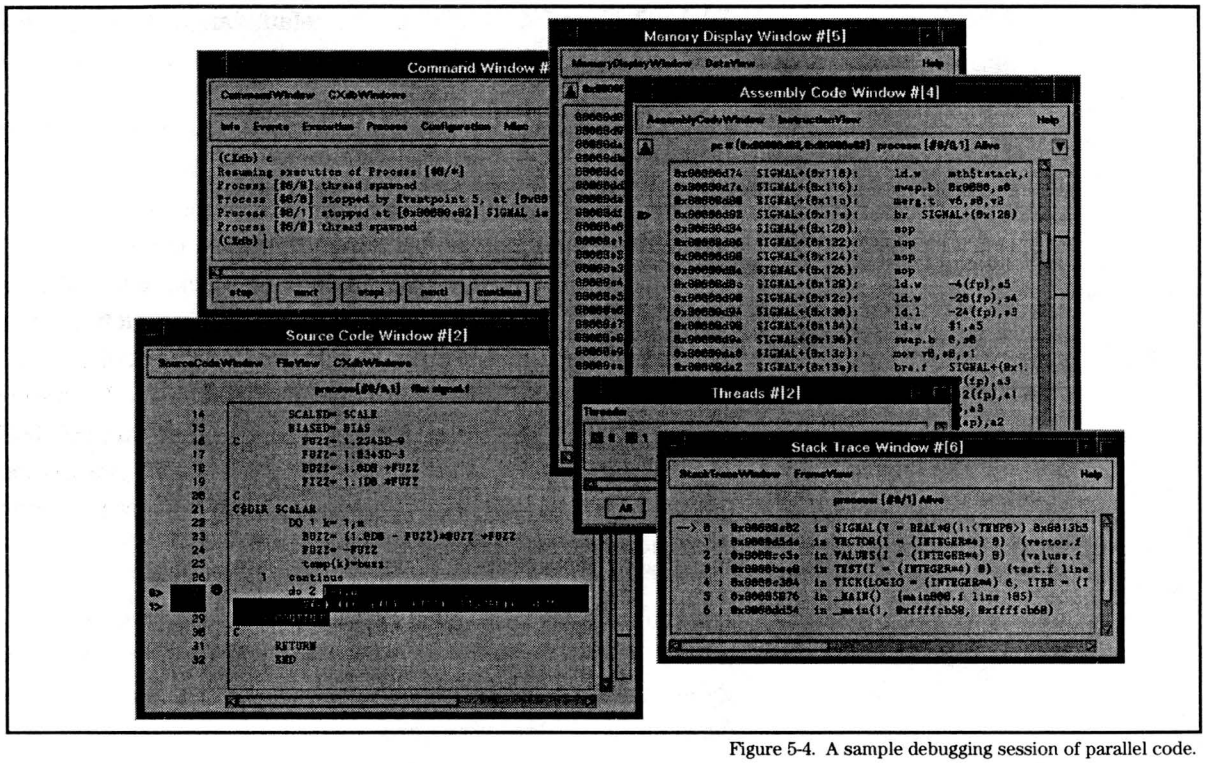

Figure 5-4. A sample debugging session of parallel code.

CXdb allows complete control over the threads in an application; one can select which thread(s) are to be associated with any of the windows, as well as breakpoints within an individual thread. CXdb provides mechanisms to view thread activity (e.g., number of threads) across an entire application. Facilities are provided to navigate through a volume of threads, displaying source code associated with different areas of activity, and repositioning the source code window from one area of activity to the next. This allows parallel application developers to quickly examine variables and states when the application reaches an exception or a breakpoint.

CXPA

CXpa is an interactive profiler which allows flexible profiling of threads and message-passing applications. CXpa is a "reductionist" or summary profiler–the data from the application is summarized for presentation to the user. CXpa contains a graphical user interface that provides detailed profiles of optimized loops, loop nests, and routines. As shown in Figure 5-5, profiling information is provided at the routine, loop, parallel region, and basic block levels to help the developer pinpoint and diagnose performance problems. CXpa is tightly integrated with the Exemplar compilers, allowing it to display profiling information on the optimized code generated by the compiler.

CXpa can provide memory latency and cache hit information through performance monitors in the hardware; providing execution information without intrusive library calls. Users are able to selectively monitor private memory, shared memory accesses, global shared memory (CC-NUMA) accesses, or any combination of these metrics.

CXpa can be used by the developer to insure that each processor has executed an equal amount of the application (load balancing), or determine which routine is utilizing the majority of the CPU time, for further optimization.

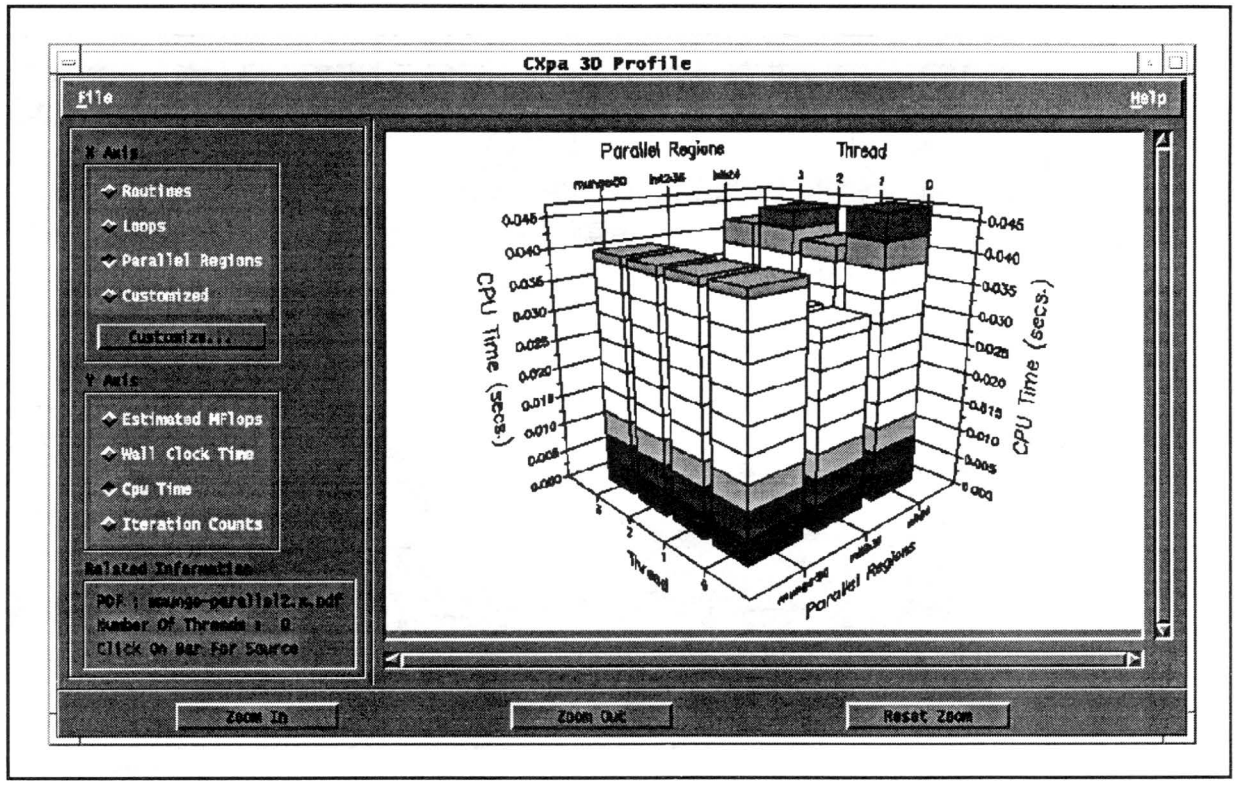

Figure 5-5. Sample output from the CXpa performance analyzer.

Besides providing a number of profiling metrics, CXpa provides two- and three-dimensional bar charts to easily identify "hot" routines or loops. Two dimensional bar charts display profiling data (wallclock time, cache misses, memory latency, etc.) aggregated across all threads in the application. Three dimensional bar charts display profiling on a per-thread (and on a loop or routine) basis. This allows users to determine how evenly their application is distributed across multiple threads. All displays feature "source click-back" in which source code can be examined by clicking portions of the performance charts.

CXtrace

CXtrace is an event-based profiler which can profile message-passing applications, or any other application where the developer has supplied "events". For message-passing applications (using PVM or MPI), the various API calls to either library are defined as events. CXtrace allows the developer to determine the order of events, and the latencies between events for these types of applications. When the application is run, selected time-stamped events are emitted to a log file. A visualizer is then used to graphically depict events in the program. An example of a six-way parallel program is shown in Figure 5-6.

As shown, CXtrace can be used to isolate inefficient message-passing within an application.

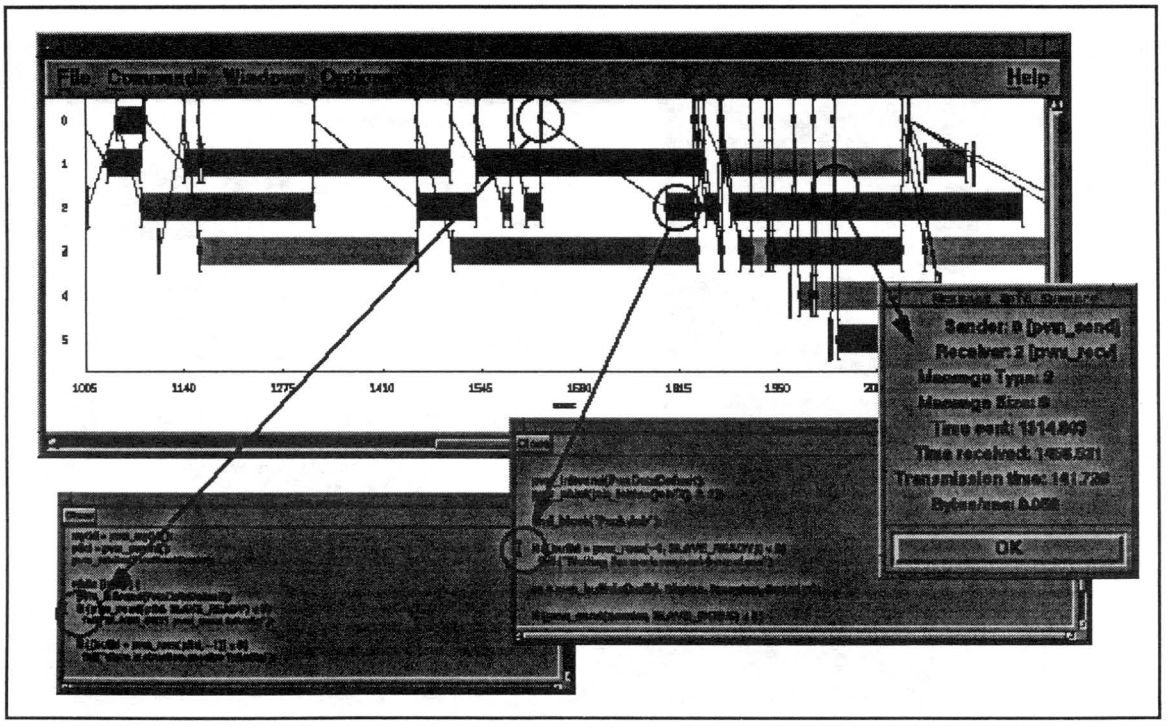

Figure 5-6. Output from CXtrace.

Figure 5-6 illustrates the computational and message-passing components of an integer sorting algorithm. Activity from different processors is displayed on separate timelines. Message-passing send and receives between processors are indicated by the lines linking different timelines. The display also points out the latencies inherent in the message-passing portions of the application. Disproportionate latencies or deviations in anticipated communication patterns, or event orderings indicating potential performance problems, become evident in such displays. Simple point and click operations allow the user to access message and process information as well as the source code associated with this information. Message information accessible through this interface includes: source and destination processor, time sent/time received, message size, and transmission rate. The information associated with a process include: source file, source routine, line number and time.

Packaging

Both the S- and X-Class systems are air-cooled and stand-alone (i.e., do not require a front-end). The systems are mechanically scalable; as additional performance and capabilities are required, additional memory, processors and I/O may be added. Further, the S-Class system is fully upgradable to the X-Class system.

S-Class Packaging

The S-Class is the mid-range member of the Exemplar Technical Server family, and is composed of one desk-height cabinet. The system offers from 4 to 16 CPUs, up to sixteen Gbytes of memory and 1.96 Gbytes/sec of I/O bandwidth. A basic S-Class system comes configured with four CPUs, 256 MBytes of memory, six PCI slots with one SCSI controller, one SCSI disk drive, and one 4mm DAT tape drive. This basic system can be easily upgraded to the full configuration shown in Figure 6-1. A full configuration may contain up to 16 CPUs, 16 Gbytes of physical memory, 24 PCI controllers, and up to 18 optional 3.5 inch SCSI devices (disks and tapes) mounted internally. External peripheral cabinets are available as an option.

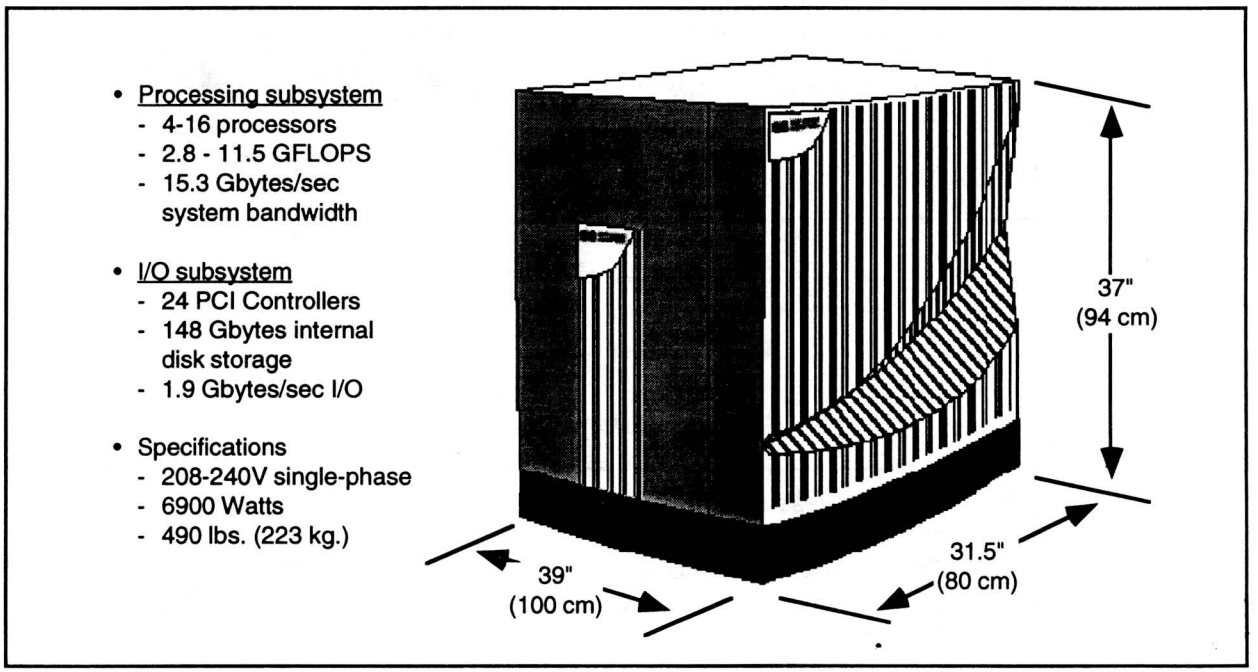

Figure 6-1. The S-Class provides compact, low-power packaging.

The S-Class is fully upgradable to an X-Class system. Thus, if computing needs grow and more than 16 CPUs, additional memory, or I/O is required, an upgrade may be ordered that includes cache coherency modules, the CTI interconnect and additional X-Class hypernodes.

X-Class Packaging

The X-Class offers from 16 to 64 CPUs, from 1 to 64 GBytes of memory, and nearly eight GBytes/sec of I/O bandwidth. As the system grows, additional cabinets are "clicked" together with the Coherent Toroidal Interconnect (CTI) and service cabling routed through cable channels between the cabinets. Cabinets are stacked in pairs as shown in Figure 6-2 to minimize floor space. Additional stacks or towers can be added to the configuration at any time. The X-Class does not require a raised floor environment.

- Processing subsystem
  - 16-64 processors
  - 11.5 - 46.1 GFLOPS
  - 76.8 Gbytes/sec system bandwidth
- I/O subsystem
  - 96 PCI Controllers
  - 592 Gbytes internal disk storage
  - 7.6 Gbytes/sec I/O
- Specifications
  - 208-240V single-phase
  - 6900 Watts / hypernode
  - 980 lbs. (446 kg.) / cabinet

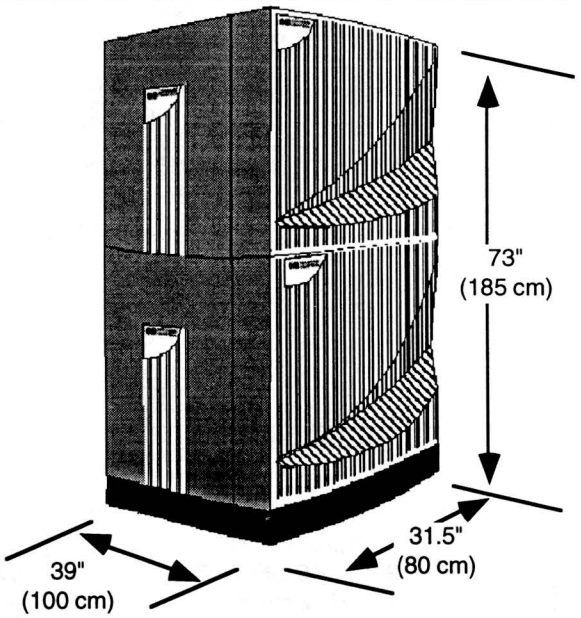

Figure 6-2. An X-Class system showing two hypernodes.

Scalability Options

The Exemplar systems provide scalability and configuration options beyond any architecture available to date. Table 6-3 provides a feature summary of upgrade levels the system grows.

Number of Processors	4	8	16	32	64
Peak Performance (GfLOPS) [7]	2.9	5.76	184.3	368.6	46.1
Max. Memory (Gbytes) [8]	16	16	16	32	64
System Bandwidth (Gbyte/sec)	15.36	15.36	15.36	34.56	76.8
I/O Bandwidth (MBytes/sec)	480	960	1,920	3,840	7,680
PCI Controllers	6	12	24	48	96

Table 6-3. Scalability Options

[7] *Assumes an initial clock rate of 180 MHz. Higher clock rates are expected in future systems.*

[8] *Configurations beyond four Gbytes will employ higher density SDRAMS (64 Mbits), available in 1997.*

Index

A

ATM · 24, 27

C

C · 29, 34
C++ · 34
cache: CTI · 19, 21; PA-8000 (architecture) · 15; PA-8000 (implementation) · 16; PA-8000 (sizes) · 15
CC-NUMA · 11, 17
Coherent Toroidal Interconnect · 19
CTI · 18, 42
CXdb ·36
CXpa · 37
CXtools · 35
CXtrace · 38

D

DataMover · 17, 19
D-Class · 5
decoupled architecture · 15
DIMM · 14
DMA · 23

E

Ethernet · 26
Exemplar operating environment (EOE): and automatic parallelization · 29; defined · 29; OS layers · 29
Exemplar Roadmap · 7
Exemplar Technical Server Portfolio · 5
explicit message-passing (EMP) · 19, 29, 33

F

FDDI · 27
FORTRAN 77 · 29, 35
Fortran 90 · 32, 35

G

globally shared memory · 11
GSM · 6 (see globally shared memory)

H

HiPPI · 24, 27
HP-UX: layers · 30
hypernode: description · 18

I

I/O subsystem · 23
instruction reorder buffer (IRB) · 15, 16

M

MIMD · 11
MPI · 18, 29
multiprocessor CPU chips · 19

N

networking options · 24

O

OC-3c · 27

P

PA-7100 · 7
PA-7200 · 7
PA-8000 · 15; features · 15
parallelism: and threads · 29; explicit message-passing (EMP) · 29; hybrid · 30; loop-level · 30; shared-memory · 29
PCI · 23, 27
PVM · 18, 29

R

RAID 0 · 24

S

Scalable Networking · 26
SCI (Scalable Coherent Interface) · 19
S-Class: crossbar · 13; description of · 12; distinction from SMP · 12; features · 12
SDRAMS: advantages · 13; defined · 13
SMP · 12, 18
SPP · 11
SPP1X00 · 6; S- and X-Class comparison with · 6
SPP-UX: layers · 30
SPP-UX: checkpoint/restart · 27; features · 27; large filesystems · 27
Striping · (see RAID 0)
Sub-Complex Manager · 31; benefits · 32

T

tiered memory · 19

U

Ultra SCSI · 24

X

X-Class: Description · 18; features · 18

Printed in U.S.A. 9/96
5965-5017E