

An Overview of the Intel TFLOPS Supercomputer

Timothy G. Mattson, Microcomputer Research Laboratory, Hillsboro, OR, Intel Corp.
Greg Henry, Enterprise Server Group, Beaverton, OR, Intel Corp.

Index words: Supercomputer, MPP, TFLOPS.

Abstract

Computer simulations needed by the U.S. Department of Energy (DOE) greatly exceed the capacity of the world's most powerful supercomputers. To satisfy this need, the DOE created the Accelerated Strategic Computing Initiative (ASCI). This program accelerates the development of new scalable supercomputers and will lead to a supercomputer early in the next century that can run at a rate of 100 trillion floating point operations per second (TFLOPS).

Intel built the first computer in this program, the ASCI Option Red Supercomputer (also known as the Intel TFLOPS supercomputer). This system has over 4500 nodes, 594 Gbytes of RAM, and two independent 1 Tbyte disk systems. Late in the spring of 1997, we set the MP LINPACK world record of 1.34 TFLOPS.

In this paper, we give an overview of the ASCI Option Red Supercomputer. The motivation for building this supercomputer is presented and the hardware and software views of the machine are described in detail. We also briefly discuss what it is like to use the machine.

Introduction

From the beginning of the computer era, scientists and engineers have posed problems that could not be solved on routinely available computer systems. These problems required large amounts of memory and vast numbers of floating point computations. The special computers built to solve these large problems were called *supercomputers*.

Among these problems, certain ones stand out by virtue of the extraordinary demands they place on a supercomputer. For example, the best climate modeling programs solve at each time step models for the ocean, the atmosphere, and the solar radiation. This leads to astronomically huge *multi-physics* simulations that challenge the most powerful supercomputers in the world.

So what is the most powerful supercomputer in the world? To answer this question we must first agree on how to measure a computer's power. One possibility is to

measure a system's peak rate for carrying out floating point arithmetic. In practice, however, these rates are only rarely approached. A more realistic approach is to use a common application to measure computer performance. Since computational linear algebra is at the heart of many scientific problems, the de facto standard benchmark has become the linear algebra benchmark, LINPACK [1,7].

The LINPACK benchmark measures the time it takes to solve a dense system of linear equations. Originally, the system size was fixed at 100, and users of the benchmark had to run a specific code. This form of the benchmark, however, tested the quality of compilers, not the relative speeds of computer systems. To make it a better computer performance metric, the LINPACK benchmark was extended to systems with 1000 linear equations, and as long as residual tests were passed, any benchmark implementation, tuning, or assembly coding was allowed. This worked quite well until computer performance increased to a point where even the LINPACK-1000 benchmark took an insignificant amount of time. So, about 15 years ago, the rules for the LINPACK benchmark were modified so any size linear system could be used. This resulted in the MP-LINPACK benchmark.

Using the MP-LINPACK benchmark as our metric, we can revisit our original question: which computer is the most powerful supercomputer in the world? In Table 1, we answer this question showing the MP-LINPACK world record holders in the 1990's.

All the machines in Table 1 are massively parallel processor (MPP) supercomputers. Furthermore, all the machines are based on *Commercial Commodity Off the Shelf* (CCOTS) microprocessors. Finally, all the machines achieve their high performance with scalable interconnection networks that let them use large numbers of processors.

The current record holder is a supercomputer built by Intel for the DOE. In December 1996, this machine, known as the *ASCI Option Red Supercomputer*, ran the MP-LINPACK benchmark at a rate of 1.06 trillion floating point operations per second (TFLOPS). This was the first time the MP-LINPACK benchmark had ever been

Year	System	Number of Processors	MP-LINPACK GFLOPS
1990	Intel iPSC®/860 [2]	128	2.6
1991	Intel DELTA [3]	512	13.9
1992	Thinking Machines CM-5 [4]	1024	59.7
1993	Intel Paragon® [5]	3744	143
1994	Intel Paragon [5]	6768	281
1996	Hitachi CP-PACS [6]	2048	368
1996	Intel ASCI Option Red Supercomputer	7264	1060
1997	Intel ASCI Option Red Supercomputer	9152	1340

Table 1: MP-LINPACK world records in the 1990's. This data was taken from the MP-LINPACK benchmark report [7].

run in excess of 1 TFLOP. In June 1997, when the full machine was installed, we reran the benchmark and achieved a rate of 1.34 TFLOPS.

In Table 2, we briefly summarize the machine's key parameters. The numbers are impressive. It occupies 1,600 sq. ft. of floor-space (not counting supporting network resources, tertiary storage, and other supporting hardware). The system's 9,216 Pentium® Pro processors with 596 Gbytes of RAM are connected through a 38 x 32 x 2 mesh. The system has a peak computation rate of 1.8 TFLOPS and a cross-section bandwidth (measured across the two 32 x 38 planes) of over 51 GB/sec.

Getting so much hardware to work together in a single supercomputer was challenging. Equally challenging was the problem of developing operating systems that can run on such a large scalable system. For the ASCI Option Red Supercomputer, we used different operating systems for different parts of the machine. The nodes involved with computation (compute nodes) run an efficient, small operating system called Cougar. The nodes that support interactive user services (service nodes) and booting services (system nodes) run a distributed UNIX operating system. The two operating systems work together so the

user sees the system as a single integrated supercomputer. These operating systems and how they support scalable computation, I/O, and high performance communication are discussed in another paper in this Q1'98 issue of the *Intel Technology Journal* entitled *Achieving Large Scale Parallelism Through Operating System Resource Management on the Intel TFLOPS Supercomputer* [8].

When scaling to so many nodes, even low probability points of failure can become a major problem. To build a robust system with so many nodes, the hardware and software must be explicitly designed for Reliability, Availability, and Serviceability (RAS). All major components are hot-swappable and repairable while the system remains under power. Hence, if several applications are running on the system at one time, only the application using the failed component will shut down. In many cases, other applications continue to run while the failed components are replaced. Of the 4,536 compute nodes and 16 on-line hot spares, for example, all can be replaced without having to cycle the power of any other module. Similarly, system operation can continue if any of the 308 patch service boards (to support RAS functionality), 640 disks, 1540 power supplies, or 616

Compute Nodes	4,536
Service Nodes	32
Disk I/O Nodes	32
System Nodes (Boot)	2
Network Nodes (Ethernet, ATM)	10
System Footprint	1,600 Square Feet
Number of Cabinets	85
System RAM	594 Gbytes
Topology	38x32x2
Node to Node bandwidth - Bi-directional	800 MB/sec
Bi-directional - Cross section Bandwidth	51.6 GB/sec
Total number of Pentium® Pro Processors	9,216
Processor to Memory Bandwidth	533 MB/sec
Compute Node Peak Performance	400 MFLOPS
System Peak Performance	1.8 TFLOPS
RAID I/O Bandwidth (per subsystem)	1.0 Gbytes/sec
RAID Storage (per subsystem)	1 Tbyte

Table 2: System parameters for the ASCI Option Red Supercomputers. The units used in this table and throughout the paper are FLOPS = floating point operations per second with M, G, and T indicating a count in the millions, billions or trillions. MB and GB are used for a decimal million or billion of bytes while Mbyte and Gbyte represent a number of bytes equal to the power of two nearest one million (2^{20}) or one billion (2^{30}) respectively.

interconnection facility (ICF) back-planes should fail.

Keeping track of the status of such a large scalable supercomputer and controlling its RAS capabilities is a difficult job. The system responsible for this job is the Scalable Platform Services (SPS). The design and function of SPS are described in another paper in this issue of the *Intel Technology Journal* entitled *Scalable Platform Services on the Intel TFLOPS Supercomputer* [9].

Finally, a supercomputer is only of value if it delivers super performance on real applications. Between MP-LINPACK and production applications running on the machine, significant results have been produced. Some of these results and a detailed discussion of performance issues related to the system are described in another paper in this issue of the *Intel Technology Journal* entitled *The Performance of the Intel TFLOPS Supercomputer* [10].

In this paper, we describe the motivation behind this machine, the system hardware and software, and how the system is used by both programmers and the end-users. The level of detail varies. When a topic is addressed elsewhere, it is discussed only briefly in this paper. For example, we say very little about SPS. When a topic is not discussed elsewhere, we go into great detail. For example, the node boards in this computer are not discussed elsewhere so we go into great detail about them. As a result, the level of detail in this paper is uneven, but, in our opinion, acceptable given the importance of getting the full story about this machine into the literature.

Why Build a TFLOPS Supercomputer?

Even with the benefits of using CCOTS technology throughout the system, a TFLOPS supercomputer is not cheap. In this era of tight government funding, it takes a compelling argument to fund such a machine. The compelling argument in this case was the maintenance of the U.S. nuclear stockpile. It is up to the DOE to maintain this stockpile without nuclear testing; science-based testing will be used instead.

Science-based testing involves a combination of laboratory experiments and computer simulations. The data from experiments plus the results from past nuclear tests will be fed into massive computer simulations that will provide the necessary information to maintain a safe and reliable nuclear stockpile.

DOE scientists have determined that they can only run these simulations if they have 100 TFLOPS computers. Given enough time, the computer industry will create 100 TFLOPS supercomputers. The DOE, however, cannot wait for industry to build these machines; they are needed early in the next century.

In response to this need, the DOE launched a five year, 900 million dollar program in 1995 to accelerate the development of extreme scale, massively parallel supercomputers with the goal of having a 100 TFLOPS computer early in the next century. This program is called

the Accelerated Strategic Computing Initiative or ASCI [13]. Scientists from the three major DOE weapons labs: Sandia National Laboratories (SNL), Los Alamos National Laboratory (LANL), and Lawrence Livermore National Laboratory (LLNL) are involved in this program.

The ASCI program will produce a series of machines leading to the 100 TFLOPS machine. The goal is to take advantage of the aggressive evolution of CCOTS technology to build each generation of ASCI supercomputer for roughly the same cost.

The first phase of the ASCI program has two tracks corresponding to the two philosophies of how to build such huge computers: ASCI Red and ASCI Blue. The red track uses densely packaged, highly integrated systems similar to the MPP machines Intel has traditionally built [2,3,5]. The blue track uses clusters of high end, off-the-shelf systems. By considering the successes and failures from each track's initial machines, the direction would be clear for subsequent machines and the path to the 100 TFLOPS goal.

The ASCI red machine was built by Intel and has been in production use at Sandia National Laboratories in Albuquerque, New Mexico since late 1996. Contracts have been awarded for two ASCI blue machines. At Los Alamos National Laboratory, the so-called *ASCI Blue Mountain* [14] machine is being built by Silicon Graphics Inc. This machine will be based on a future version of the Origin 2000 computer and should be operational by mid 1999. At Livermore National Laboratory, the *ASCI Blue Pacific* [15] machine is being built by IBM. This machine will be based on the SP* system with a newer communication switch and a node board with eight PowerPC microprocessors. This machine will be operational in late 1998 or early 1999. The ASCI blue machines will have peak performances in the range of three TFLOPS.

Before leaving this section, we need to address one critical requirement for all of the ASCI machines. According to DOE security regulations, the hard disks used when carrying out classified computing must be permanently and physically isolated from the outside world. In other words, if a disk has *ever* had classified data on it, that disk can *never* be attached to an outside network. Hence, the ASCI supercomputers must have disks that can be switched so the machine can be used for both classified and unclassified computing. Later in this paper, we will address how the ASCI Option Red Supercomputer met this requirement.

The ASCI Option Red Supercomputer: Hardware Overview

The ASCI Option Red Supercomputer is a Massively Parallel Processor (MPP) with a distributed memory Multiple-Instruction, Multiple Data (MIMD) architecture.

All aspects of this system are scalable including the aggregate communication bandwidth, the number of compute nodes, the amount of main memory, disk storage capacity, and I/O bandwidth.

In the following sections, we will discuss the major hardware components used to implement the ASCI Option Red Supercomputer. We will begin with a quick introduction to the Intel Pentium Pro processor. We will follow this with a discussion of the two kinds of node boards used in the system: Eagle and Kestrel. Finally, we will talk about the Interconnection Facility (ICF) used to connect the nodes together and the packaging of all these

parts into a full machine.

Figure 1 is a diagram of the ASCI Option Red Supercomputer as it sits at SNL in Albuquerque, New Mexico. The machine is organized into a large pool of

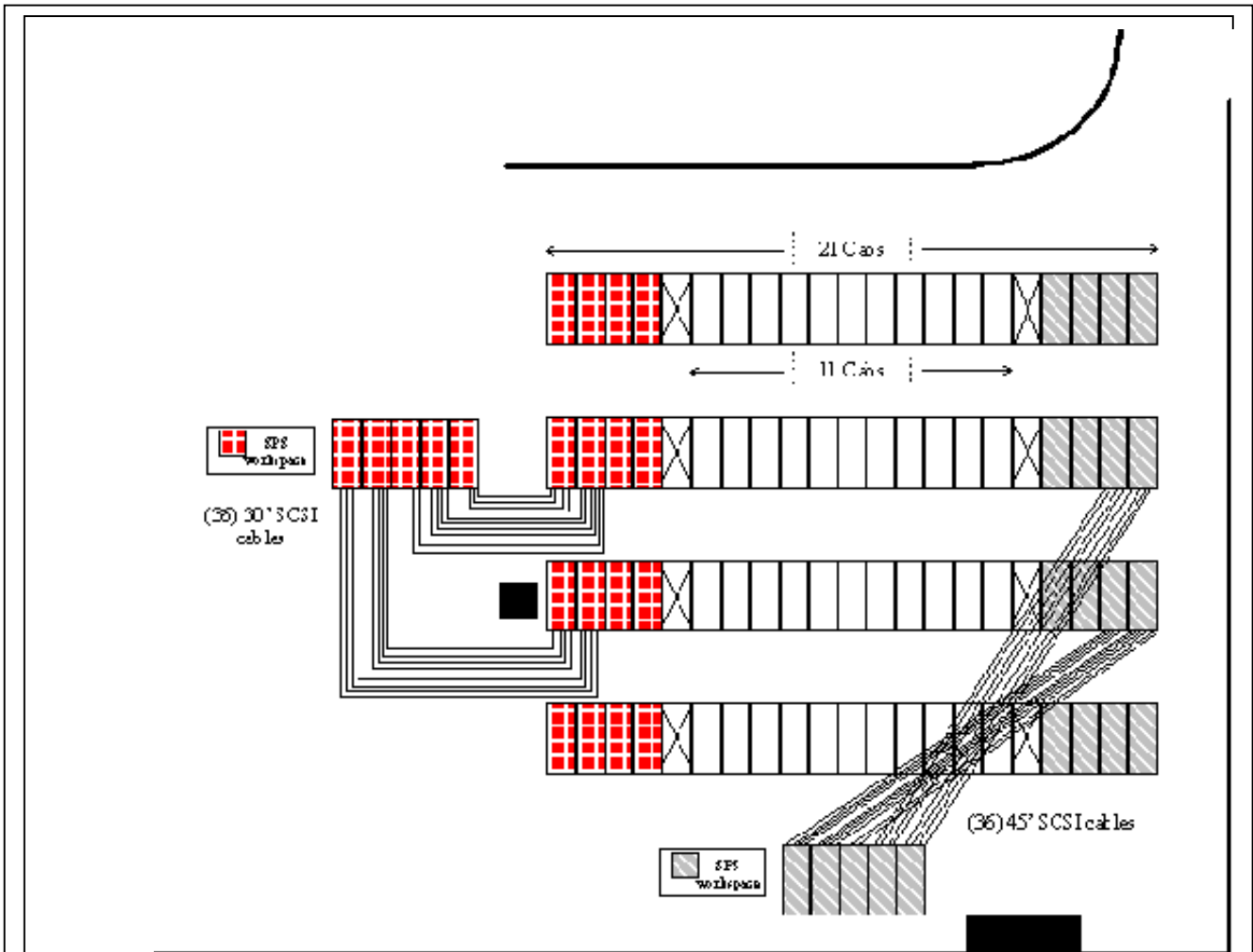


Fig. 1: Schematic diagram of the ASCI Option Red supercomputer as it will be installed at Sandia Nat. Laboratories in Albuquerque NM. The cabinets near each end labeled with an X are the disconnect cabinets used to isolate one end or the other. Each end of the computer has its own I/O subsystem (the group of 5 cabinets at the bottom and the left), and their own SPS station (next to the I/O cabinets). The lines show the SCSI cables connecting the I/O nodes to the I/O cabinets. The curved line at the top of the page show the windowed-wall to the room where the machine operators will sit. The black square in the middle of the room is a support post.

compute nodes in the center, two distinct blocks of nodes at either end, and two separate one-Tbyte disk systems. The end-blocks and their disk systems can be isolated from the rest of the machine by disconnecting the X-mesh cables in the *disconnect cabinets* (marked with an X in Figure 1). This design satisfies DOE security requirements for a physically isolated classified disk system while assuring that both disk systems are always available to users.

Rectangular meshes are needed, hence the number of cabinets set up for isolation must be the same in each row on each end. The most likely configuration would put disconnect cabinets four cabinets over from each end, but this can be varied to meet customer needs. Depending on which types of node boards are used in which slots, this would yield a 400 GFLOPS stand-alone system.

The Pentium Pro Processor

The Intel Pentium Pro processor [11] is used throughout the ASCI Option Red Supercomputer. The instruction set for the Pentium Pro processor is basically the same as the IA-32 instructions used on a Pentium® processor. Unlike the Pentium processor, however, the Pentium Pro processor doesn't directly execute the IA-32 instructions. These complex instructions are broken down

at runtime into simpler instructions called micro-operations (or *uops*). The uops execute inside the Pentium Pro processor with the order of execution dictated by the availability of data. This lets the CPU continue with productive work when other uops are waiting for data or functional units.

The Pentium Pro processor can complete up to three uops per cycle of which only one can be a floating-point operation. The floating-point unit requires two cycles per multiply and one cycle per add. The adds can be interleaved with the multiplies so the Pentium Pro processor can have a floating point result ready to retire every cycle. The processors used in the ASCI Option Red Supercomputer run at 200 MHz so the peak floating point rate is 200 MFLOPS.

The Pentium Pro processor has separate on-chip data and instruction L1 caches (each of which is eight KBytes). It also has an L2 cache (256 Kbytes) packaged with the CPU in a single dual-cavity PGA package. All cache lines are 32 bytes wide.

The Pentium Pro processor bus supports memory and cache coherency for up to four Pentium Pro processors. In the ASCI Option Red Supercomputer, however, we used only two processors per node.

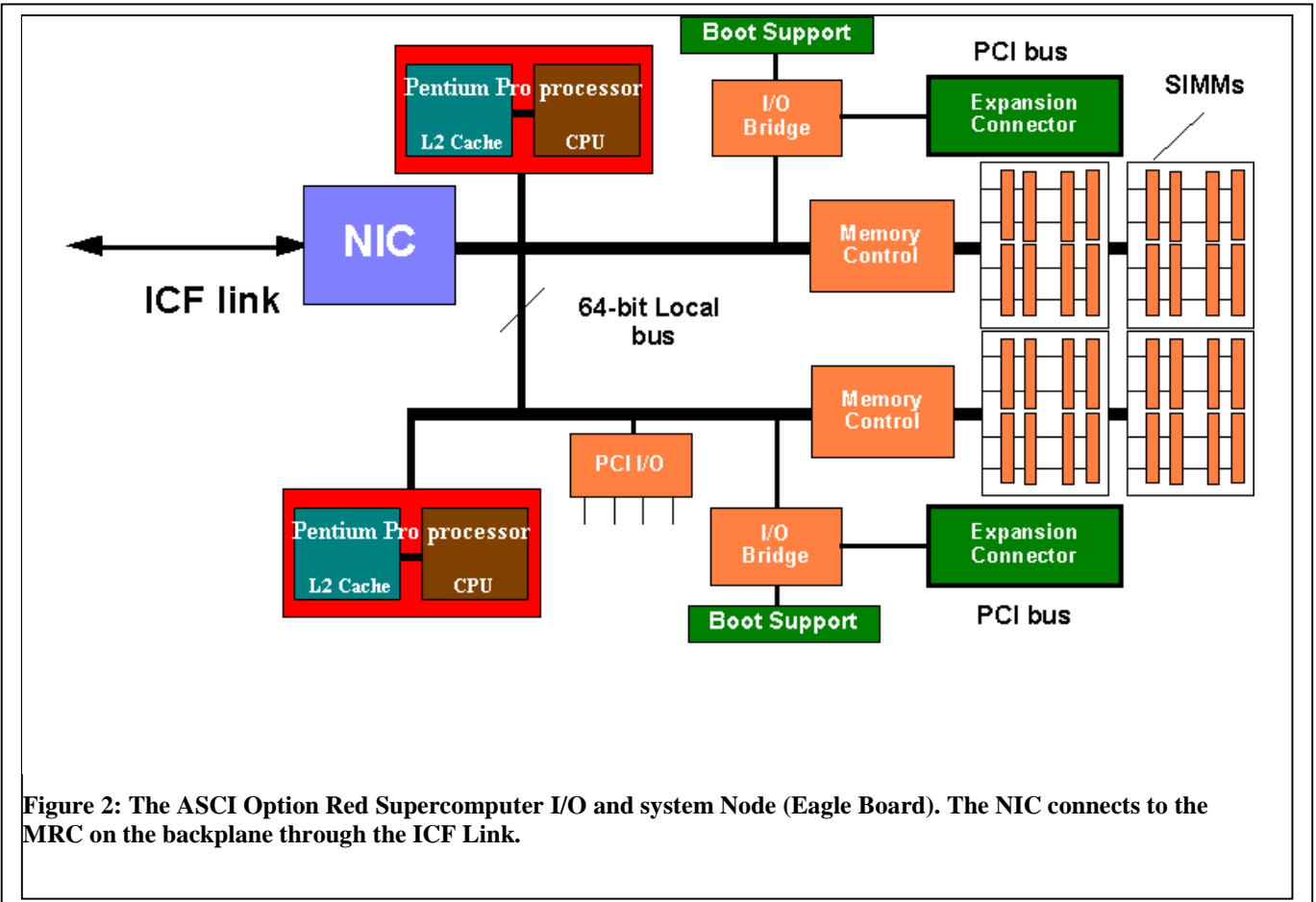


Figure 2: The ASCI Option Red Supercomputer I/O and system Node (Eagle Board). The NIC connects to the MRC on the backplane through the ICF Link.

The Eagle Board

The node boards used in the I/O and system partitions are called *Eagle* boards. Figure 2 shows a block diagram for an Eagle board. Each node includes two 200 MHz Pentium Pro processors. These two processors support two on-board PCI interfaces that each provide 133 MB/sec I/O bandwidth. One of the two buses can support two PCI cards through the use of a 2-level riser card. Thus, a single Eagle board can be configured with up to three long-form PCI adapter cards. CCOTS PCI adapter boards can be inserted into these interfaces to provide Ultra-SCSI, HiPPI, ATM, FDDI, and numerous other custom and industry-standard I/O capabilities. In addition to add-in card capabilities, there are base I/O features built into every board that are accessible through the front panel. These include RS232, 10 Mbit Ethernet, and differential FW-SCSI.

Each Eagle board provides ample processing capacity and throughput to support a wide variety of high-performance I/O devices. The throughput of each PCI bus is dictated by the type of interface supported by the PCI adapter in use, the driver software, and the higher-level protocols used by the application and the "other end" of the interface. The data rates associated with common I/O devices fit well within the throughput supported by the PCI bus. Ultra-SCSI, for example, provides a hardware rate of 40 MB/sec. This rate can easily be supported by

CCOTS PCI adapters.

The memory subsystem is implemented using Intel's CCOTS Pentium Pro processor support chip-set (82453). It is structured as four rows of four, independently-controlled, sequentially-interleaved banks of DRAM to produce up to 533 MB/sec of data throughput. Each bank of memory is 72 bits wide, allowing for 64 data bits plus 8 bits ECC, which provides single bit error correction and multiple bit error detection. The banks are implemented as two 36-bit SIMMs, so industry standard SIMM modules can be used to provide 64 Mbytes to one Gbytes of memory.

The Kestrel Board

Kestrel boards (see Figure 3) are used in the compute and service partitions. Each Kestrel board holds two compute nodes. The nodes are connected through their Network Interface Chip (NIC) with one of the NICs connecting to a Mesh Router Chip (MRC) on the backplane. Each node on the Kestrel board includes its own boot support (FLASH ROM and simple I/O devices) through a PCI bridge on its local bus. A connector is provided to allow testing of each node through this PCI bridge. The FLASH ROM contains the Node Confidence Tests, BIOS, plus additional features needed to diagnose board failures and to load a variety of operating systems. Local I/O support includes a serial port, called the "Node Maintenance Port." This port interfaces to the system's

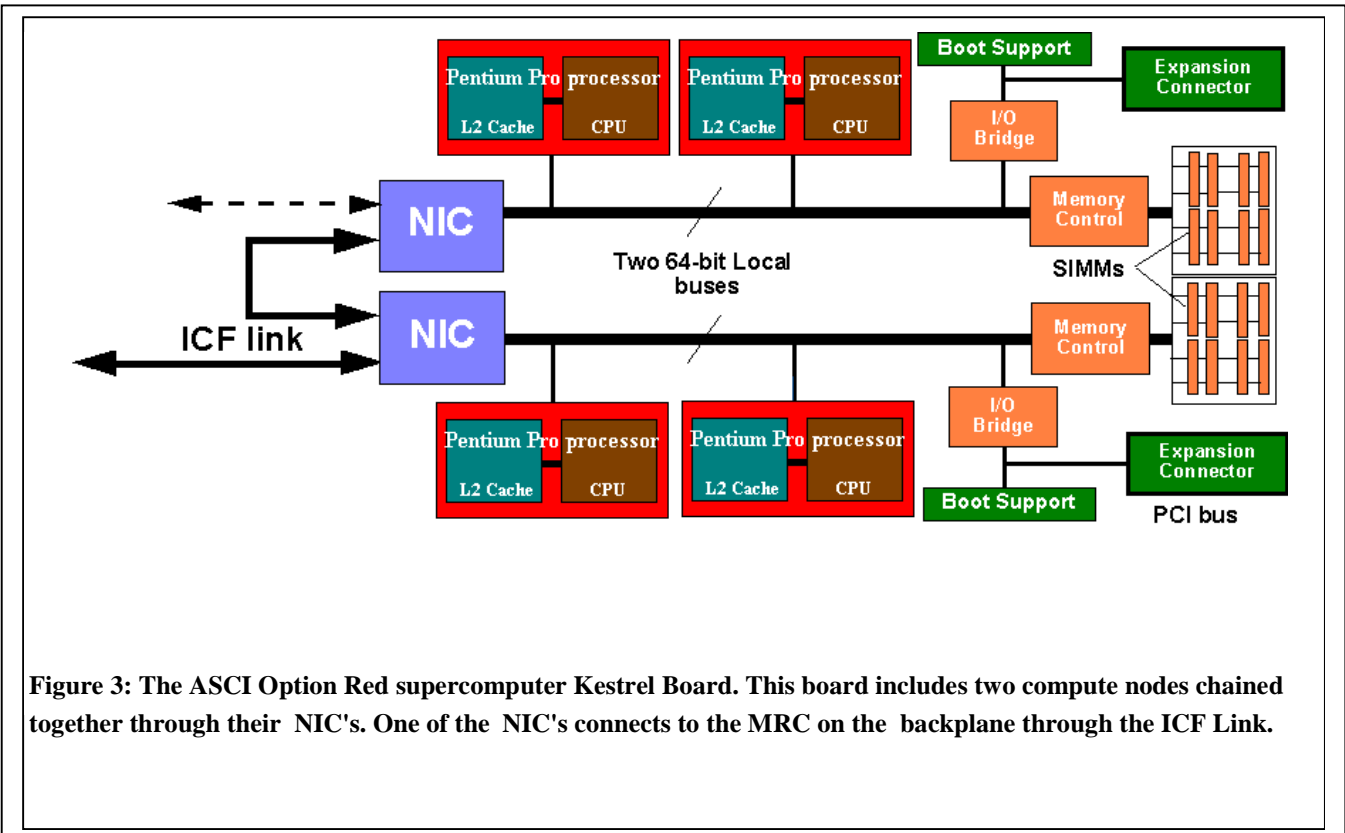


Figure 3: The ASCI Option Red supercomputer Kestrel Board. This board includes two compute nodes chained together through their NIC's. One of the NIC's connects to the MRC on the backplane through the ICF Link.

internal Ethernet through the PSB on each card cage.

The memory subsystem on an individual compute node is implemented using Intel's CCOTS Pentium Pro processor support chip-set (82453). It is structured as two rows of four, independently-controlled, sequentially-interleaved banks of DRAM to produce up to 533 MB/sec of data throughput. Each bank of memory is 72 bits wide, allowing for 64 data bits plus 8 bits ECC, which provides single bit error correction and multiple bit error detection. The banks are implemented as two 36-bit SIMMs, so industry standard SIMM modules may be used. Using commonly available 4 and 8 MByte SIMMs (based on 1Mx4 DRAM chips) and 16 and 32 MByte SIMMs (based on 4Mx4 DRAM chips), 32 MB to 256 MB of memory per node is supported. The system was delivered with 128 Mbytes/node.

Interconnection Facility

The interconnection facility (ICF) is shown in Figure 4. It utilizes a dual plane mesh to provide better aggregate bandwidth and to support routing around mesh failures. It uses two custom components: NIC and MRC. The MRC sits on the system back-plane and routes messages across the machine. It supports bi-directional bandwidths of up to 800 Mbytes/sec over each of six ports (i.e., two directions for each X, Y, and Z port). Each port is composed of four virtual lanes that equally share the total bandwidth. This means that as many as four message streams can pass through an MRC on any given port at any given time. This reduces the impact of communication contention and leads to a more effective use of total system bandwidth.

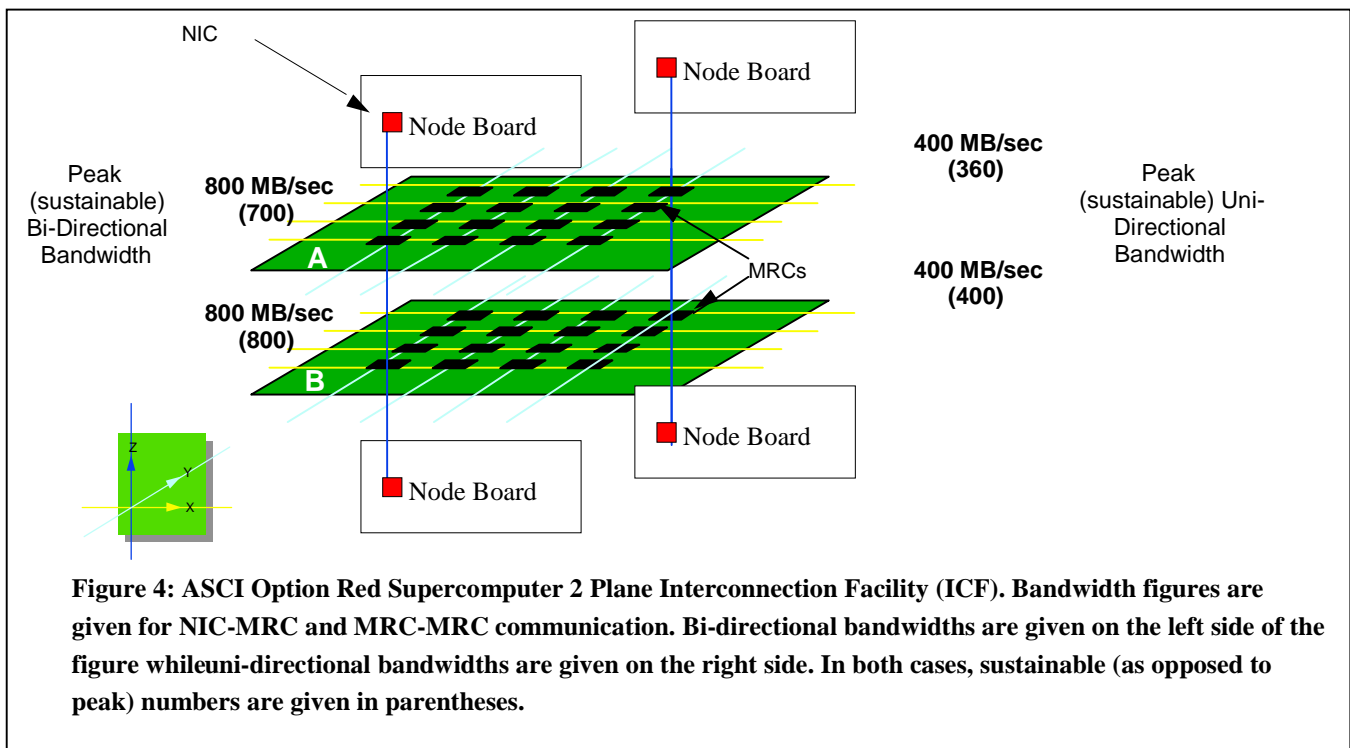
The NIC resides on each node and provides an interface between the node's memory bus and the MRC. The NIC can be connected to another NIC to support dense packaging on node boards. For example, on the Kestrel board, the NIC on one node, the *outer* node, is connected to the NIC on the other node, the *inner* node, which then connects to the MRC. Contention is minimized in this configuration since the virtual lane capability used on the MRCs was included on the NICs.

Putting the Pieces Together

Boards and an ICF are not enough to build a working supercomputer. In this section we very briefly address how these components are packaged into a full system.

The boards and ICF components are packaged into cabinets and organized into a full system. Each cabinet contains a power supply, four card cages, and a fan unit. The fan unit sits on top of the basic cabinet making the full assembly eight feet tall. A card cage holds any combination of eight Kestrel or Eagle node boards. The card cage also holds two MRC cards with four MRCs per card. The pair of MRC cards implements a cube topology and provides the back and front planes of the split plane mesh and the connection points for the node boards.

Associated with each card cage is the PSB. The PSB monitors the system resources within the card cage and communicates this information back to a system management console over a separate Ethernet network. The cabinets are connected in the X direction with cables between adjacent cabinets. For the Y direction, cables run from the top of one cabinet, down through the floor, and



into the bottom of the corresponding cabinet in the next row. This cabling connects the nodes into a 38x32x2 topology.

I/O and network nodes are usually placed near either end of the full machine. As mentioned earlier, the I/O nodes are Eagle nodes since they provide more room for PCI connections. These nodes are connected with standard SCSI cables to the disk RAIDS in separate cabinets.

The User's View of the ASCII Option Red Supercomputer

To the user, the ASCII Option Red Supercomputer has the look and feel of a UNIX-based supercomputer. When users log onto the machine, they get a familiar UNIX prompt. Files are presented to the user and manipulated in the standard UNIX way. Programs are built with *make* and shell scripts provide compiler interfaces that link in the special compilers and libraries needed to build parallel programs. It is only when the user submits a parallel job that the system deviates from a standard workstation environment—but even this deviation is slight.

To submit a parallel job, the user issues a command with a typically cryptic UNIX style name ("yod"). Parameters to "yod" control the runtime factors effecting a computation such as how many and which nodes to use. Once the job is running, the user can monitor the job with a command called "showmesh." This command graphically displays the compute nodes visible to the user and the jobs running on the machine. In principle, it is not that different from the familiar "ps" command used to monitor a job on a UNIX workstation. A standard NQS environment is available to submit batch jobs.

To a programmer, the machine looks like a typical MPP supercomputer. The programs running on each node use standard sequential languages. To move data between nodes, the programs explicitly call routines from a message-passing library (MPI or NX in most cases). When developing software, building programs, or other interactive operations, the computer will have the look and feel of a UNIX workstation.

Fortran77, Fortran90, C, and C++ compilers from Portland Group Incorporated (PGI) are available on the system. In addition, interactive debuggers and performance analysis tools that work with and understand the source code for each of these languages are provided. For data-parallel programming, the HPF compiler from PGI is provided.

While message passing (either explicitly or implicitly through HPF) is used between nodes, shared memory mechanisms are used to exploit parallelism on a node. The user has three multiprocessor options. First, the second processor can be completely ignored. Alternatively, it can

be used as a communication co-processor. Finally, a simple threads model can be used to utilize both processors on a single application. The threads model is accessed through compiler-driven parallelism (using the *-Mconcur* switch) or through an explicit dispatch mechanism referred to as the *COP* interface.

The *COP* interface lets a programmer execute a routine within a thread running on the other processor. Global variables are visible to both the *COP* thread and the parent thread. To use *COP*, the programmer passes *COP* the address of the routine, the address of a structure holding the routine's input parameters, and an output variable to set when the routine is completed. The *COP*'ed routine can not make system calls (including calls to the message-passing library).

The debugger on the ASCII Option Red Supercomputer is a major re-implementation of the *IPD* [16] debugger developed for the Paragon XP/S supercomputer. The debugger has been designed to scale up to the full size of the ASCII Option Red Supercomputer. It includes both graphical and command line interfaces. The debugger's command line interface has been designed to mimic the *DBX* interface where ever possible.

The performance analysis tools use the counters included on the Pentium Pro processor and on the NIC. The counters on the Pentium Pro processor let users track a range of operations including floating point operation counts, cache line loads, and data memory references. Each Pentium Pro processor has two counters so only two independent events can be counted at one time. The NIC has ten independent counters.

We anticipate that the applications on this system will run for many hours or even days. Hence, even a system mean time between failure in excess of our target (>50 hours) will not be sufficient. Therefore, a check-point/restart capability will be provided. Automatic check-pointing is exceedingly difficult to implement on systems as large as this one. Hence, applications will need to assist the check-pointing by putting themselves into a clean state prior to explicitly invoking a check-point. (A clean state is one where the communication network does not hold any message-passing state for the application being check-pointed.) The I/O bandwidth will be sufficient to check-point the entire system memory in approximately five minutes.

The ASCII Option Red Supercomputer: Software

The user view of the ASCII Option Red Supercomputer is shown in Figure 5. This view is maintained by the system software which organizes the system into four logical partitions:

- **Service Partition**—provides an integrated, scalable "host" that supports interactive users,

application development, and system administration. An operator can vary the number of service nodes at boot time depending on the demand for interactive services.

- **I/O Partitions**—implement scalable file and network services.
- **Compute Partition**—contains the nodes dedicated to running parallel applications.
- **System Partition**—supports initial system booting. The boot node has its own independent RAID and Ethernet connections. A second Eagle node configured as part of the I/O partition can be cross-connected with the boot RAID to provide an on-line spare to the boot node.

In normal operation, one of the sets of disconnect cabinets will cut the system in two. In this case, each side will see the logical partition model outlined in Figure 5.

In the following sections, we describe the key features of the system software on the ASCI Option Red Supercomputer. We start with the two operating systems used on the system. We then describe the *portals*

mechanism used for high performance communication.

The Operating Systems

Each partition in the system places different demands on the operating system. One operating system that met the needs of all of the partitions could have been developed. This would guarantee, however, that the operating system did all jobs adequately but none of them particularly well. We took a different approach and used multiple operating systems tuned to the specific needs of each partition.

The service, I/O, and system partitions are directly visible to interactive users. These partitions need a familiar, UNIX operating system. We used Intel's distributed version of UNIX (POSIX 1003.1 and XPG3, AT&T System V.3 and 4.3 BSD Reno VFS) developed for the Paragon® XP/S supercomputer. The port of the Paragon OS to the ASCI Option Red Supercomputer resulted in an OS we call the TFLOPS OS. The TFLOPS OS presents a single system image to the user. This means that users see the system as a single UNIX machine despite the fact that the operating system is running on a distributed collection of nodes.

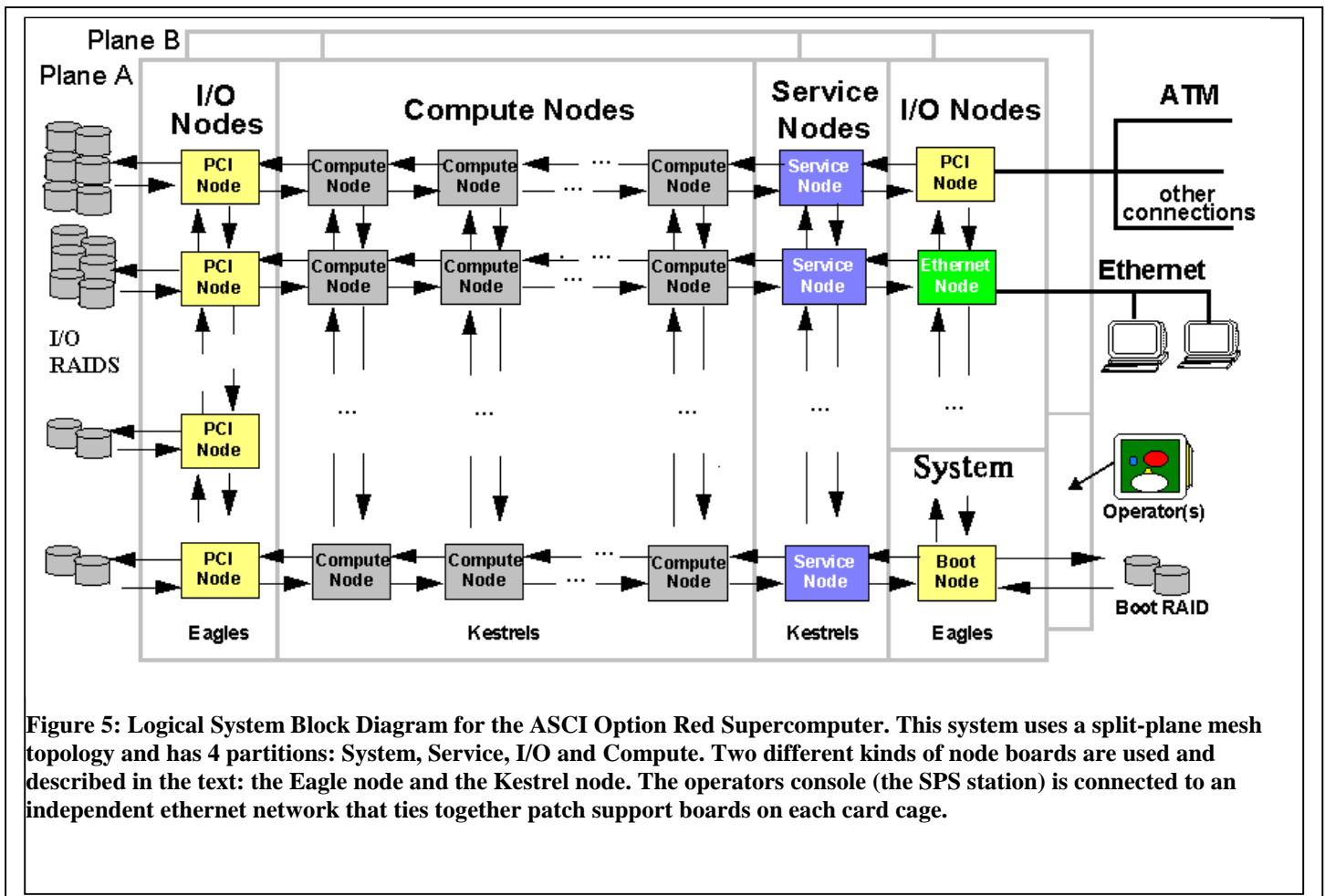


Figure 5: Logical System Block Diagram for the ASCI Option Red Supercomputer. This system uses a split-plane mesh topology and has 4 partitions: System, Service, I/O and Compute. Two different kinds of node boards are used and described in the text: the Eagle node and the Kestrel node. The operators console (the SPS station) is connected to an independent ethernet network that ties together patch support boards on each card cage.

The compute partition has different needs. Users only run parallel applications on these nodes, not general interactive services. Furthermore, these nodes are the most numerous so the aggregate costs of wasted resources (such as memory consumed by an OS) grows rapidly. Therefore, for the compute partition, we wanted an operating system that was small in size, very fast, and provided just those features needed for computation.

On our Paragon XP/S supercomputers, we had great success with SUNMOS [17] –a light-weight operating system from Sandia National Laboratories and the University of New Mexico. For the ASCI Option Red Supercomputer, we decided to work with their next light weight OS (Puma [3]). We call our version of Puma, *Cougar*.

Cougar easily meets our criteria for a compute partition operating system. It is small (less than half a megabyte), of low complexity, and scalable. Cougar can be viewed in terms of four entities:

- host Operating System
- quintessential kernel (Q-Kernel)
- process control thread (PCT)
- application

Since it is a minimal operating system, Cougar depends on a host OS to provide system services and to support interactive users. For the ASCI Option Red Supercomputer, the host OS is the TFLOPS OS running in the service partition. The Q-Kernel is the lowest level component of Cougar. All access to hardware resources comes from the Q-Kernel. Above the Q-Kernel sits the process control thread (PCT). This component runs in user space and manages processes. At the highest level are the users' applications.

Cougar takes a simple view of a user's application. An application is viewed as a collection of processes grouped together and identified by a common group identifier. Within each group, a process is assigned a group rank which ranges from 0 to $(n-1)$ where n is the number of processes. While the PCT supports priority multi-tasking, it is anticipated that most users will run only one application process per node.

Memory integrity in Cougar is assured by a *hierarchy of trusts* ranging from the Q-Kernel to the PCT to the application. At each level in the hierarchy, lower levels are trusted but higher levels are not trusted. Hence, an application cannot corrupt the PCT or Q-Kernel while a flawed Q-Kernel can corrupt anything.

The Quintessential Kernel or Q-Kernel

There is one Q-Kernel running on each of the nodes in the compute partition. It provides access to the physical resources of a node. For example, only the Q-Kernel can directly access communication hardware or handle interrupts. In addition, any low-level functions that need to

be executed in supervisor mode are handled by the Q-Kernel. This includes memory management, context switching, and message dispatch or reception.

The Q-Kernel is accessed through a series of system traps. These are usually invoked by an application or a PCT, but they can also arise from an exception (e.g., a floating point exception) or an interrupt (e.g., timer or communication interrupts).

The Q-Kernel does not set the policy for how a node's resources will be used. Rather, it performs its low-level tasks on behalf of the PCT or a user's application. This design keeps the Q-Kernel small and easy to maintain.

The Process Control Thread

The Process Control Thread (PCT) sits on top of the Q-Kernel and manages process creation, process scheduling, and all other operating system resources. While part of the operating system, the PCT is a user-level process meaning that it has read/write access to the user address space. There will typically be only one PCT per node.

Most of the time, the PCT is not active. It only becomes active when the Q-Kernel receives an application exception (a process blocks itself by a call to the `quit_quantum()` routine) or in response to certain timer interrupts.

When a PCT becomes active, it first checks to see if the most recently suspended process has any outstanding requests for the PCT. Once these requests are resolved, it handles requests from any other PCTs. Finally, it tries to execute the highest priority, run-able application.

ASCI Option Red Message Communication Software: Portals

Low-level communication on the ASCI Option Red Supercomputer uses Cougar portals [12]. A portal is a window into a process's address space. Using portals, a process can write-to or read-from a special address subspace on another process. This address space is user-accessible meaning copying between kernel space and user space is avoided. Since extra copy operations increase communication latencies, portals support low-latency communication. In addition to low latencies, portals provide high-performance asynchronous transfers with buffering in the application's data space.

Application programs will have access to portals, but we expect most applications will use higher level message-passing libraries. A variety of message-passing protocols will be available on top of portals. The preferred library is MPI. This is a full implementation of the MPI 1.1 specification.

To support applications currently running on Paragon supercomputers, we implemented a subset of the NX message-passing library on top of portals. Since both MPI

and NX run on top of portals, latency and bandwidth numbers will be comparable for applications utilizing either library.

System RAS Capabilities

We have set aggressive reliability targets for this system. The key target is that a single application will see a mean time between failure of greater than 50 hours. Some applications (e.g., CTH [5]) have already exceeded this target. In the aggregate, however, we expect far more from our system. Our target is to have the system in continuous operation for greater than four weeks with no less than 97% of the system resources being available. In order to meet these targets, the system includes sophisticated Reliability, Availability, and Serviceability (RAS) capabilities. These capabilities will let the system continue to operate in the face of failures in all major system components.

Three techniques are used to meet our system RAS targets. First, the system includes redundant components so failures can be managed without swapping hardware. For example, the dual plane ICF uses Z-X-Y-Z routing so a bad mesh router chip can be by-passed without causing system failure. In addition, the system will include on-line spare compute-nodes that can be mapped into the system without interruption.

The second RAS technique is to build the system so all major components are hot-swappable and can be repaired while the system continues to run. The compute nodes and the on-line spares, for example, can all be replaced without having to cycle the power of any other module. Similarly, system operation can continue if any of the 640 disks, 1540 power supplies, or 616 ICF backplanes should fail.

Finally, to manage these RAS features and to manage the configuration of such a large system, an active Monitoring and Recovery Subsystem (MRS) is included. At the heart of this system is a PSB—one per 8-board card cage. The PSB board monitors the boards in its card cage and updates a central MRS database using a separate RAS Ethernet network. This database will drive an intelligent diagnostic system and will help manage the replacement of system units. The console for the RAS subsystem is the SPS station. There is one of these connected to the RAS Ethernet network on each side of the machine (see Figure 1). As mentioned earlier, the SPS system is described in more detail in another paper in this issue of the *Intel Technology Journal* [4].

Conclusion

DOE scientists need computers that can deliver sustained TFLOPS in order to get their jobs done. To meet that need, the DOE created the ASCI program, a program that will produce a series of supercomputers ranging from 1 TFLOPS in 1996 to up to 100 TFLOPS

early in the next century. The first of these machines is the ASCI Option Red Supercomputer built by Intel.

The machine is a huge massively parallel supercomputer containing over 9200 Pentium Pro processors. It was this supercomputer (actually 3/4 of it) that first broke the 1 TFLOPS MP-LINPACK barrier (1.06 TFLOPS on December 7, 1996). Later, after the full machine was installed at Sandia National Laboratories, we broke our own record and ran the MP-LINPACK benchmark at 1.34 TFLOPS.

Supercomputers, however, aren't built to run benchmarks. A machine is justified by its ability to run applications that solve key problems. On this front, we have succeeded as well. As we built the machine, we used application benchmarks and full application programs to debug the machine. Hence, within hours of installing the machine, it was being used for production calculations. Actually, the first production calculations were carried out while it was still being built in the factory!

It is customary to close a research paper with a discussion of future plans. After we started building this supercomputer, our focus changed from providing MPP systems for end-users, to providing components and expertise to the broader high performance computing market. We learned a lot from our experiences, and this knowledge is being applied throughout Intel. Hence, the core technologies behind the machine will likely show up in future products.

Acknowledgments

The Intel ASCI Option Red Supercomputer is a result of the "blood, sweat and tears" of hundreds of Intel engineers over the course of a decade. It all started *with Intel Scientific Computers* back in the mid 1980's and ended with the *Scalable Systems Division* inside Intel's *Enterprise Server Group* in the late 1990's. This was our last MPP machine, and most of the people behind this computer have already moved on in their careers. We can't list all of you here so we must settle for a collective *thank you*. To these people, we hope you are doing well and that you take as much pride in this remarkable machine as we do.

*All other brands and names are the property of their respective owners.

References

- [1] J.J. Dongarra, J.R. Bunch, C.B. Moler, and G.W. Stewart, *Linpack Users' Guide*, SIAM, Philadelphia, PA, 1979.
- [2] E. Barszcz, "One Year with an iPSC/860," *Proceedings of the COMPCON Spring'91 conference*, p. 213, IEEE Computer Society Press, 1991.

- [3] S.L. Lillevik, "The Touchstone 30 Gigaflop DELTA Prototype," *Proceedings of the sixth Distributed memory Computer Conference*, p. 671, IEEE Computer Society Press, 1991.
- [4] P.S. Lomdahl, P. Tamayo, N. Gronbech-Jensen, D. M. Beazley, "50 Gflops Molecular Dynamics on the Connection Machine 5", *Proceedings of Supercomputing'93*, p. 520, IEEE Computer Society Press, 1993.
- [5] D.R. Mackay, J. Drake, T. Sheehan, B. Shelton, "Experiences in Programming Grand Challenge Applications on the Intel MP Paragon Supercomputer," *Proceedings of the Intel Supercomputer Users Group*, 1995. Available on the web at <http://www.cs.sandia.gov/ISUG>.
- [6] Y. Abei, K. Itakura, I. Boku, H. Nakamura and K. Nakazawa, "Performance Improvement for Matrix Calculation on CP-PACS Node Processor," in *Proceedings of the High Performance computing on the Information Superhighway conference (HPC_ASIA'97)*, 1997.
- [7] Dongarra, J.J., "Performance of various computers using standard linear equations software in a Fortran environment," Computer Science Technical Report CS-89-85, University of Tennessee, 1989, <http://www.netlib.org/benchmark/performance.ps>
- [8] S. Garg, R. Godley, R. Griffiths, A. Pffifer, T. Prickett, D. Robboy, S. Smith, T. M. Stallcup, and S. Zeisset, "Achieving Large Scale Parallelism Through Operating System Resource Management on the Intel TFLOPS Supercomputer," *Intel Technology Journal*, Q1'98 issue, 1998.
- [9] R. Larsen and B. Mitchell, "Scalable Platform Services on the Intel TFLOPS Supercomputer," *Intel Technology Journal*, Q1'98 issue, 1998.
- [10] G. Henry, B. Cole, P. Fay, T.G. Mattson, "The Performance of the Intel TFLOPS Supercomputer," *Intel Technology Journal*, Q1'98 issue, 1998.
- [11] Pentium Pro Processor technical documents, <http://www.intel.com/design/pro/manuals/>.
- [12] S.R. Wheat, R. Riesen, A.B. Maccabe, D.W. van Dresser, and T. M. Stallcup, "Puma: An Operating System for Massively Parallel Systems," *Proceedings of the 27'th Hawaii International Conference on Systems Sciences Vol II*, p. 56, 1994.
- [13] ASCI overview web site, <http://www.sandia.gov/ASCI>.
- [14] ASCI Blue Mountain web site, <http://www.lanl.gov/projects/asci/bluemtn>.
- [15] ASCI Blue Pacific web site, <http://www.llnl.gov/asci/platforms/bluepac>.
- [16] D. Breazeal, K. Callaghan, and W.D. Smith, "IPD: A Debugger for Parallel Heterogeneous systems", in *Proceedings of the ACM/ONR Workshop on Parallel and Distributed Debugging*, p. 216, 1991.
- [17] B. Traversat, B. Nitzberg and S. Fineberg, "Experience with SUNMOS on the Paragon XP/S-15," in *Proceedings of the Intel Supercomputer User's meeting*, San Diego, 1994.

Authors' Biographies

Timothy G. Mattson has a Ph.D. in chemistry (1985, U.C Santa Cruz) for his research on Quantum Scattering theory. He has been with Intel since 1993 and is currently a research scientist in Intel's Parallel Algorithms Laboratory where he works on technologies to support the expression of parallel algorithms. Tim's life is centered on his family, snow skiing, science and anything that has to do with kayaks. His e-mail is timothy_g_mattson@ccm2.hf.intel.com

Greg Henry received his Ph.D. from Cornell University in Applied Mathematics. He started working at Intel SSD in August 1993. He is now a Computational Scientist for the ASCI Option Red Supercomputer. He tuned MP LINPACK and the BLAS used there-in. Greg has three children and a wonderful wife. He plays roller hockey, soccer, and he enjoys Aikido and writing. His e-mail address is henry@co.intel.com.