Compaq High Performance
Technical Computing

AlphaPowered

AlphaServer SC: Scalable Supercomputing

COMPAQ

Notice

AlphaServerSC: Scalable Supercomputing

Technical Paper
prepared by
High Performance Technical Computing Group

(First Edition, July 2000)

Document number: 135D-0900A-USEN

Contents

AlphaServer SC: Scalable Supercomputing

Abstract:

With the introduction of Compaq alphaServer SC family of supercomputers, some of the fastest systems on the planet wear a Compaq label. This is clearly shown by two of these new systems debuting in the top 25 supercomputers, as shown by the June, 2000 release of the independent Top 500 Supercomputer Sites list at http://wwwtop500.org.

While these systems are the first supercomputers to wear the compaq name, Compaq technonogy powers 22 of the 50 fastest supercomputers in the world. In comparison, Sun Microsystems and Hewlett-Packard each placed a single system in the top *150* systems.

This paper explores the new AlphaServer SC family of supercomputers. It covers the history of supercomputing at Compaq, the architecture and building blocks of the AlphaServer SC family, and presents the unique scalability and upgradability of these systems.

While supercomputer discussions often focus on raw hardware and peak performance, the real issues revolve around software and management. This paper delves into a range of software-related topics, ranging from parallel programming libraries and development tools through the technologies that allow managing hundreds of processors as a single system. And, of course, the challenge of debugging a program running on 500 processors. . . .

The AlphaServer SC family represents the future of high performance computing; standard processors and system building blocks, high performance interconnects, standard software interfaces and a strong *systems* focus.

Help us improve our technical communication. Let us know what you think about the technical information in this document. Your feedback is valuable and will help us structure future communications. Please send your comments to: Russ.Doty@compaq.com

COMPAQ

Introduction

This paper explores the Compaq AlphaServer SC family of supercomputers. It provides a detailed explanation of the hardware, software, and development environment for these systems. It is intended to serve as an introduction to the AlphaServer SC family, to provide as understanding of the design and philosophy and capabilities of these systems, and to position them in the Compaq family. After reading this paper, you will be able to determine if the AlphaServer SC family is a good solution to your technical computing needs and how it will fit into your computing environment.

The AlphaServer SC family of supercomputers leverages Compaq's technology to deliver flexible, powerful systems offering unique scalability and upgradability. By using a combination of existing processor and system designs, high performance interconnect and innovative software, Compaq is able to deliver true supercomputer performance at affordable (in supercomputer terms!) price points. In addition, the AlphaServer is built on the newest industry-standard software models, providing excellent portability of applications.

As computer simulation becomes the tool of choice in research, engineering and industry, the size and sophistication of problems is growing dramatically and the time allowed to solve these problems is shrinking. High Performance Technical Computing (HPTC) is required to solve these large problems in an acceptable time. As computer models grow more viable as an alternative to physical experimentation, the economic benefits of high performance systems grow. High Performance Technical Computing has moved from an esoteric field to a broad based requirement.

Compaq is a leader in developing and delivering high performance systems. With the introduction of the AlphaServer SC family, Compaq is delivering everything from low cost desktop systems to the most powerful supercomputers. Compaq is not insisting on a single technical path for HPTC – a wide range of technologies are supported, including traditional SMP, clusters and large distributed processing systems.

A common characteristic of all approaches used by Compaq is to develop building blocks that meet the demands of HPTC and then combine them in a variety of ways. This approach enables Compaq to meet a broad range of performance needs at effective price points and with a high degree of software compatibility.

Perhaps the most significant aspect of Compaq's efforts is the system approach taken. Raw hardware is not useful unless it can be exploited by applications, managed and maintained by system managers, and upgraded and enhanced with technology advances. Users, system managers, and business professionals working with large computer systems are very aware of these issues! Compaq's development efforts are focused as much on software, system management and operations, maintenance and upgradability as on raw hardware performance.

The system approach becomes critical when you consider that AlphaServer SC systems will scale to 4,000 processors in the year 2001, and to 16,000 processors by 2004.

Key Features

- Up to 512 667MHz Alpha 21264a (EV67) processors in 1999
- Up to 4,096 700+MHz Alpha 21264a (EV67) processors in 2001
- Over 16,000 Alpha 21364 (EV7) Microprocessors within three years
- More than 6 GB/sec of bisection bandwidth
- Less than 3µ seconds of user-level latency
- Optimized MPI and Shmem parallel processing libraries
- Comprehensive parallel development software
- Very Large Memory Support
- Hot swap and hot add components
- Redundant remote and local console subsystem options
- High availability and reliability features engineered throughout
- "Single System" management model
- Cluster and parallel file systems

HPTC Requirements

High performance technical computing has traditionally placed three primary demands on system architectures: floating point computation (specifically 64-bit double precision floating point), memory bandwidth and memory size.

All of these elements are connected. Technical computing is usually used to model and simulate physical phenomena, using a variety of mathematical equations. As models grow larger and more sophisticated, data sets grow dramatically in size. For example, a 100x100x100 element model contains 1 million elements. Assuming that this model contains four data elements, each a 64-bit value, the model consumes 32 MB of memory - a reasonable size for even desktop systems today.

A 100x100x100 grid is quite small, and limits the accuracy of many models. If the grid resolution is increased by a factor of 10 - which can still be directly displayed on a computer screen - the model becomes 1,000x1,000x1,000, or 1 billion elements. Using the same four data elements per grid point, the raw data for this model now consumes 32 gigabytes of memory! Processing this data requires moving it from memory into the CPU - thus demanding high memory bandwidth. And, of course, there is the actual computation performed on the data.

Supercomputers are designed to handle this massive workload. These machines have been designed for maximum speed, using fast processors, high performance memory and I/O subsystems. They provide support for multiple giga-bytes of physical memory and use techniques such as vector processing to deliver high performance. Supercomputers have pioneered other technologies, such as parallel processing with the ability to apply hundreds – or even thousands – of processors to a single task. Supercomputers have used special processor, interconnect and system designs to maximize performance at all cost – or seemingly to maximize performance at maximum cost.

Many of the concepts pioneered by supercomputers have migrated into the computing mainstream. Microprocessors are constantly increasing their clock speed, with samples currently exceeding 1 GHz. Many, including the Alpha processor, support 64-bit addressing and large memories. Some processors, most notably the Alpha family, are providing memory bandwidth approaching that of traditional supercomputers. In fact, the cache bandwidth of the Alpha 21264 processor exceeds the memory bandwidth of traditional supercomputers.

Newer techniques, including superscalar designs that issue several instructions at a time and fully pipelined floating point units, provide the performance of vector processing with much greater flexibility.

AlphaServer SC History and Strategy

In 1992, Digital Equipment Corporation (acquired by Compaq in 1998) released the first generation Alpha 21064 processor and systems built with this processor. Providing leadership performance — which it has retained through today and will continue to provide — the Alpha processor was used in a wide range of systems. The fundamental design and architecture of the 21064 was well suited to HPTC, and it proved popular.

At the same time, Massively Parallel Processing (MPP) was emerging as an alternative to traditional vector supercomputers. MPP systems were developed using both custom processors (such as the Thinking Machines CM-5) and standard processors. Cray Research, the traditional leader in vector supercomputing, understood the need to offer an MPP system.

Cray and Digital equipment entered into a partnership to deliver an MPP system. This system was designed to address the needs of the high end of the supercomputing market. Digital Equipment provided the Alpha processor, and Cray developed an extremely high performance interconnect to connect hundreds or even thousands of Alpha processors together into a single system. And, of course, since Cray understood *systems issues*, they developed a complete infrastructure addressing needs such as I/O, software development tools and system management tools.

The result was the highly successful Cray T3D family, a milestone in the evolution of supercomputing. These systems proved popular for a wide range of applications. They combined the performance and capabilities of high performance standard processors with the ability to effectively harness hundreds or thousands of processors for a single job, and pioneered the development of software techniques and tools that are widely used today.

When the second generation Alpha 21164 processor was released, Cray incorporated it into an upgraded system, the T3E. Many of the systems on today's Top500 list (the 500 most powerful computers in the world, see http://www.top500.org) are T3E's.

In 1997, Cray Research was acquired by SGI. SGI elected to discontinue development of the T3 line. When Compaq released the third generation Alpha 21264 processor, Cray/SGI did not incorporate it into the T3. Cray/SGI is not offering their T3 users an upgrade path.

In April of 2000, the Cray supercomputer business unit was sold by SGI to Tera Computer Company. Tera has named the combined company Cray, Inc. The impact of this transfer on product directions is not known.

Compaq decided to directly offer our own supercomputer, exploiting the latest developments in processors, system design and supercomputer architectures. The result is the *AlphaServer SC system.*

The AlphaServer SC system has many similarities to the T3 family. It is built on the Alpha processor with a high performance, low latency interconnect and scales to systems incorporating thousands of processors. It offers a software environment similar to the T3. Common characteristics include the UNIX operating system, programming languages and other tools. Many aspects of the AlphaServer SC systems are designed for compatibility with the T3: in addition to the industry standard MPI programming interface, the AlphaServer SC system supports the Cray T3 *shmem* parallel programming interface and Cray mathematical libraries.

The result is higher performance than offered by the T3D or T3E systems and simplified applications portability from Cray T3 systems. Moving applications from the T3 to AlphaServer SC system typically does not require major re-work of code. In addition, the AlphaServer SC System offers a much more flexible growth path.

Beyond it's compatibility with the T3 family, AlphaServer SC System provides the best supercomputing solution available today. AlphaServer SC systems are built around the latest broad-based high performance computing concepts: it is a distributed processing system using standard interfaces, most

notably *MPI* and *Posix Threads*. It supports all applications that run on other Compaq AlphaServer systems. It uses standard software, including Compaq Tru64 UNIX and TruCluster V5. It provides a rich development environment, including all standard languages, Fortran90, HPF and OpenMP. It supports Compaq's Enterprise Toolkit, which allows a user running a Windows NT desktop system to develop and debug AlphaServer applications from a Microsoft Visual Studio environment.

In addition, the AlphaServer SC family provides an exceptionally clean and flexible growth and upgrade path. Unlike traditional systems that require all processors to be identical, the AlphaServer SC system is designed to allow new processors and new systems to be added without replacing existing components. The AlphaServer SC system is designed to support multiple generations of Alpha processors — starting with the 21264, adding the 21364 (EV7) and then adding the 21464 (EV8). Within the 21264 family, processors running at different speeds are supported in the AlphaServer SC system.

The result of this deliberate design decision is the ability to grow an AlphaServer SC system over a multi-year period through incremental upgrades rather than replacement. These incremental upgrades use the latest technology available at each point in time, delivering the highest performance available. It is absolutely **not** necessary to discard existing components or to upgrade existing components to a common version in order to expand an AlphaServer SC system.

The design flexibility of the AlphaServer SC system does not result in any performance compromises. AlphaServer SC systems deliver the full performance of the AlphaServer compute nodes that they are built on, as well as high bandwidth, low latency, low overhead communication between nodes.

The combination of support for industry standard applications and programming interfaces, highest levels of system performance, an extremely clean growth and upgrade path, greatly simplified system administration and the raw technical computing power of the Alpha processor combine to make the AlphaServer SC family the best choice for high-end technical computing needs.

Having explored the rich technical heritage that lead to the development of the AlphaServer SC family, we will now examine it in detail. We will cover the architecture of the system from a hardware and software perspective, the implementation of the system, and the ways that the combination of standard interfaces with high performance implementations of the underlying hardware and software yields today's best supercomputing environment.

AlphaServer SC Systems Architecture

AlphaServer SC systems provide a distributed memory environment, where each node has its own memory and address space, and the address space is not shared between nodes. These systems are programmed using a *message passing* model, which involves exchanging blocks of data between *cooperating processes*. This is in contrast to the shared memory model, where processes update data in memory without requiring the cooperation of other processes using the same memory.

The most widely used interface is *MPI* (Message Passing Interface). MPI is a specification for the software API which allows processes in a parallel program to exchange information with each other. MPI does not specify a physical implementation; MPI may be implemented using ordinary Ethernet or with exotic hardware acceleration. The underlying *data transport mechanism* is hidden from the application — the application does not know or care what the transport is. This greatly simplifies programming and porting of applications. The transport mechanism does, however, have a massive impact on performance.

AlphaServer SC systems support the distributed memory model through a highly optimized implementation of MPI that uses Elan hardware, software and switches (described in detail later in this paper) to provide extremely low latency and high bandwidth communications. This hardware accelerated communication incurs very little system overhead; the Alpha system simply writes its data to a location in memory. The Elan interconnect handles the entire communications task, and uses hardware accelerated Direct Memory Access (DMA) to interact with the host Alpha processor.

Beyond the base interprocess communication capabilities, the AlphaServer SC system has implemented a broad set of hardware and software functions to support *Single System Management*. The goal of Single System Management is to provide a total environment that appears like a single large multiprocessor system. This includes aspects such as system management, job scheduling, system identification, storage integration, file system sharing, resource management, partitioning, user management, and similar capabilities.

Connection Models

Three models are commonly used for connecting systems: flat bus, fully connected crossbar switch and tree structure. For HPTC applications, the flat bus is immediately rejected because of its fixed bandwidth, which as shared by all systems; this simply does not provide adequate performance. Fully connected crossbar switches and tree structure interconnects are widely used in computing; each has strengths and drawbacks. AlphaServer SC systems introduce a new model, the *fat tree*, which combines the characteristics of both fully connected crossbar and tree structures.

Fully connected crossbar switches provide a direct pairwise connection between any two nodes. In fact, the fully connected crossbar provides a simultaneous direct connection between *every pair* of two nodes.

The drawback of a fully connected crossbar switch is that the required internal connections increase as the square of the number of external connections:

External Connections (ports)	Internal Connections
4	16
8	64
10	100
16	256
32	1,024
64	4,096

Figure 1

It is clear from this table that the number of internal connections grows tremendously for fully connected crossbar switches which support large numbers of ports. Supporting large numbers of ports incurs high complexity and cost. Cost increases in terms of the size of the silicon chip required to implement the switch, and also in terms of the number of *signal pins* required to support each port. In addition, latency increase as the number of internal connections grow.

An additional complexity in the design of crossbar switches is whether they are *blocking* or *non-blocking.* The simplest design allows internal communication paths to be allocated to a data transfer, and unavailable for other use until the transfer completes – this is a *blocking* design. In a non-blocking design, on the other

hand, internal resources are available to multiple data transfers. Non-blocking designs provide higher performance but incur greater cost and complexity.

For a variety of reasons, the "sweet spot" for crossbar switches is eight ports. Eight ports is large enough to be useful, small enough to implement economically, and a manageable level of complexity for chip design. Assuming 32 signal pins are required for each port, an eight port crossbar would need a 300 pin package. This type of chip package is widely available at low cost. Using the same assumptions, a 16 port crossbar would require a 600 pin package, which is much more expensive. A 32 port crossbar would required over 1,200 pins, which is extremely expensive and at the state of the art for current semiconductor packaging technology.

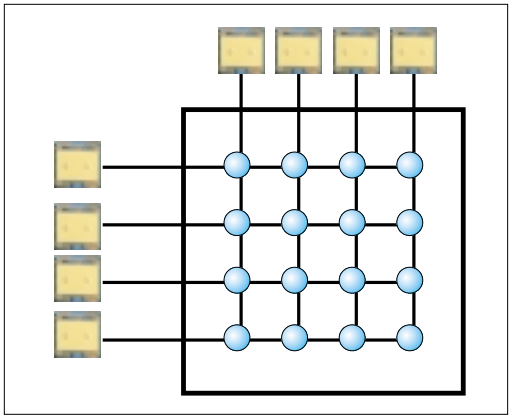


Figure 2: 8-port Crossbar Switch

Thus, the most common implementation is to use eight port crossbar switches. This allows connecting up to eight nodes directly together. The logical next step is to connect the crossbar switches directly to each other, allowing larger numbers of nodes to be connected.

The most common way of doing this is with a *tree structure.* One familiar example of a tree structure is a binary tree, where each internal node has one connection up and two connections down.

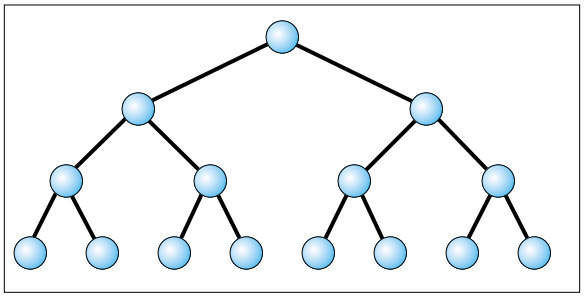


Figure 3: Binary Tree

A binary tree is constructed of the equivalent of three port switches. Using larger switches allows connecting more nodes in a shorter tree. For example, using four port switches results in a tree that expands by a factor of three for each level, looking like:

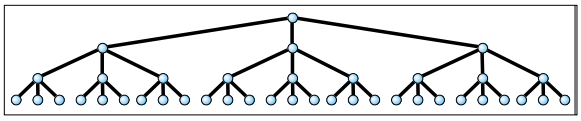


Figure 4: Trinary Tree

If eight way switches are used, the tree expands by a factor of seven for each level. Thus, a simple two level tree can support 56 nodes:

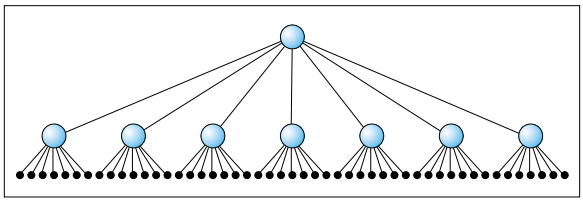


Figure 5: Tree Structure built with 8-way crossbar switches

This sort of tree structure is commonly encountered in local area networks. Many Beowulf Clusters are built using Ethernet switches cascaded together exactly as shown in this illustration. (Note: Ethernet switches are readily available supporting 48 or more ports. These tend to become internally complex, and Ethernet is certainly **not** a high performance interconnect for HPTC!)

The drawback of a tree structure is that the available *communication bandwidth* does not grow as the number of nodes increases – there is a fixed bandwidth shared by all nodes. When high performance systems are required to work with a fixed communication bandwidth, bandwidth quickly becomes the primary performance bottleneck. Such systems don't scale to support large numbers of processors.

An elegant solution is to construct a tree structure with more internal connections — a ***fat tree.*** Such a fat tree has several advantages. First, bandwidth scales with increasing numbers of processors. In fact, the bisection bandwidth of a fat tree is identical to the bisection bandwidth of a fully connected crossbar switch. (Bisection bandwidth is the common measurement of communication bandwidth inside large systems. It is the bandwidth available for pair-wise communication between processors or compute nodes.)

At the same time, the internal connection count of a fat tree grows much more slowly, at the rate of O (n *log* n) (where n is the number of ports), as opposed to the crossbar switch growth rate of O (n^2).

The AlphaServer SC system uses eight port crossbar switches, each implemented as a single integrated circuit, to construct the fat tree interconnect. These switches, their packaging, and the design of the 128 port switch are described later in this document.

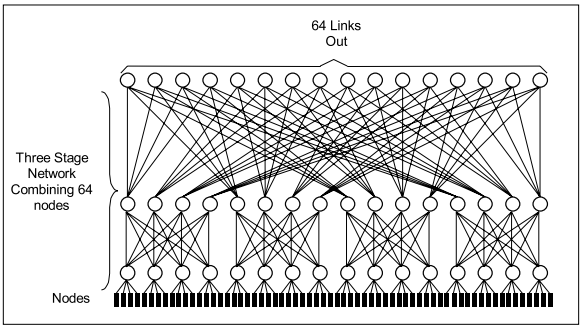


Figure 6: AlphaServer SC Fat Tree Topology

A 64-node network is illustrated. 64 systems are connected to stage 1, 64 links connect stage 1 to stage 2, and similarly stage 2 to stage 3, 64 links are available at the *top* for expansion. The bi-directional nature of the links localizes traffic to a sub-tree large enough to span both nodes. The uplinks in the top stage of a switch network can either be used for expansion or they can be used as additional downlinks doubling the number of nodes that can be connected without reducing the bisectional bandwidth.

A Fat Tree topology uses many more crossbar switches than an equivalent binary tree – and provides much better latency and bandwidth scaling. Communication between the four nodes connected to a single switch is performed locally in the switch. The latency is the latency of the single switch, captured as a *latency factor* of 1.

Connecting 16 nodes involves 8 switches connected in two layers. The multiple connections between the first layer and the second layer provide several times the bandwidth of a single layer one/layer two connection. This combination of local switching between nodes connected to a single switch and multiple paths between switches delivers the same bisection bandwidth as a fully connected crossbar, and typically delivers lower average latencies.

Communicating between two nodes on different switches involves going into the source switch, going up to the second level switch, and then going down to the target switch. This involves three switch operations, resulting in a latency factor of 3. Note that the connectivity allows the each of the second level switches to talk to any of the first level switches.

Nodes	Switching Layers (l)	Network Switches (n/4*1)	Latency Factor
4	1	1	1
16	2	8	3
64	3	48	5
256	4	256	7

Figure 7: Scaling Characteristics of AlphaServer SC system interconnect

The Fat Tree topology allows the node count to grow by a factor of four for each additional layer of switches. As shown in the table, a three layer switch can support up to 64 nodes, and a four layer switch can support up to 256 nodes.

The combination of multiple layers of switches and rich connection between switches enables multiple routes for transferring data. The switches incorporate multiple routing algorithms, which are designed to minimize latency, maximize bandwidth across the entire switch, and adapt to changing loads.

The 128-Port switch used in AlphaServer SC systems is a four-level switch. Design and packaging considerations resulted in a 128 port switch, instead of the theoretical 256 port limit.

AlphaServer SC Interconnect Performance

Having discussed the theoretical bandwidth and scaling issues of the AlphaServer SC Interconnect, we will explore its delivered performance in real systems.

Measured latency in AlphaServer SC system is roughly 3.6 microseconds using SHMEM and 5.1 microseconds using MPI. These latency figures are for process to process communication between the most widely separated nodes in a fully configured system (Node 0 and Node 127). These are not theoretical hardware specifications; they are measured results using the system the way actual applications do.

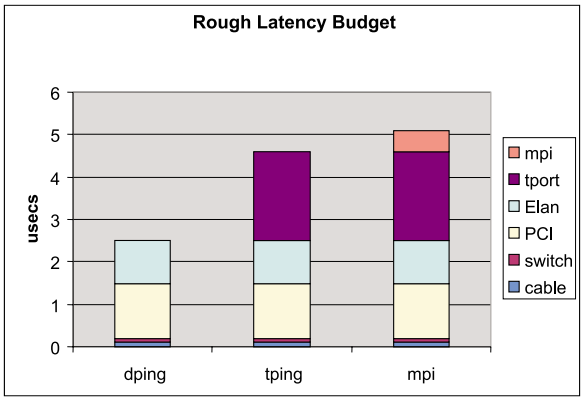


Figure 8: AlphaServer SC Interconnect Latency (measured)

Within these latency figures, it is interesting to examine what the various factors contribute. As shown in the attached chart, the largest factors are tport, PCI latency, and the Elan Adapter Card. The Switch contributes almost no latency, and the relatively long interconnect cables (up to 10 meters long) are a very small factor. It is also interesting to note that the MPI implementation on the AlphaServer SC system is very efficient, and imposes little latency.

In absolute terms, the latency of the AlphaServer SC interconnect is very good. The extremely low latency of the switch provides excellent scaling for large systems.

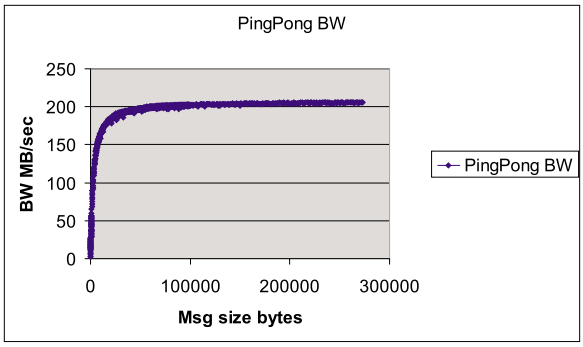


Figure 9: Delivered MPI Bandwidth across AlphaServer SC Interconnect

The other key factor in interconnect performance is bandwidth. Bandwidth was measured by doing “ping pong” transfers of data – a block of data was sent back and forth between two nodes. Again, the measurements were performed between the two most distant nodes (Node 0 and Node 127). Again note that this is measured bandwidth, using MPI, between two processes – this is not a theoretical hardware specification.

This chart shows the common behavior of data communications. There is an overhead associated with each data transfer operation, regardless of the amount of data transferred. Sending 1 byte of data incurs the same overhead as sending 1,000 bytes of data or even 1,000,000 bytes. This analysis is slightly simplified, but a good first order approximation. Sending 1,000 bytes of data a byte at a time requires 1,000 data transfer operations. Sending the same 1,000 bytes in a 1,000 byte block requires a single data transfer operation.

The result is that small data packets result in a low delivered bandwidth.

The AlphaServer SC Interconnect is quite efficient. It is capable of delivering over 200 MB/sec sustained data transfer rates, and provides excellent bandwidth with relatively small packet size.

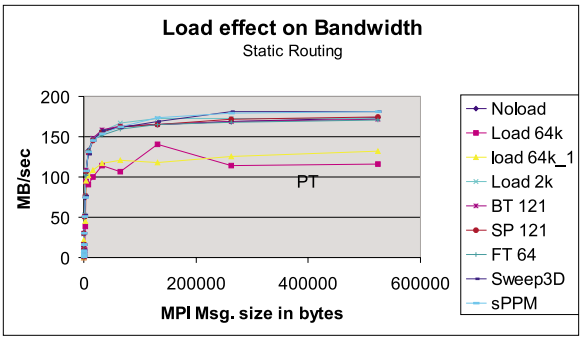


Figure 10: Behavior of AlphaServer SC Interconnect under load

Another key performance metric for real systems is behavior under load. An interconnect that works well when only two nodes are communicating, but degrades badly when many nodes are communicating, is not especially useful for large systems.

Testing bandwidth under load proved somewhat challenging. It was necessary to develop special programs to synthetically saturate the interconnect to produce significant degradation. This chart again shows ping pong communication between Node 0 and Node 127. The difference is that now Nodes 1 through 126 are all communicating with each other as rapidly as possible. Under this load, the delivered bandwidth does go down slightly. Measured bandwidth falls from slightly over 200 MB/sec on an unloaded system to 170-180 MB/sec on a heavily loaded system.

This is a surprisingly low level of degradation, and demonstrates that AlphaServer SC systems have a very robust interconnect. The combination of low latency, high bandwidth, and robust behavior under heavy workloads means that AlphaServer SC systems are not just large systems – they are large *fast* systems when running real applications.

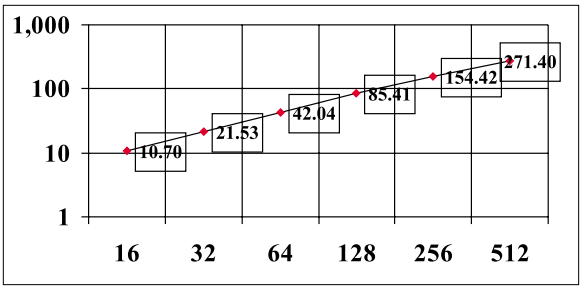


Figure 11: Linpack NxN scaling across AlphaServer SC systems

As a final proof point, consider Linpack. Linpack is the commonly used performance measure for high performance systems. It is a challenging metric for large systems, as performance improvements usually get smaller as processor counts increase. The combination of AlphaServer compute nodes, the AlphaServer SC Interconnect, and optimized compilers and math libraries produces near linear scaling at high processor counts.

In general, AlphaServer SC systems deliver a much higher portion of their peak performance than other systems.

AlphaServer SC Hardware Architecture

From a hardware perspective, an AlphaServer SC system consists of four elements: processors, compute nodes, interconnect and storage.

Processors are the familiar Alpha processors; specifically, the Alpha 21264a processor (in the current generation of AlphaServer SC system). These are programmed using all of the familiar languages, tools, math libraries and applications.

The processors are combined into *compute nodes*, which are multiprocessor AlphaServer systems. The nodes used in the first generation AlphaServer SC system are the four processor AlphaServer ES40 systems; future generations will use larger nodes.

The *interconnect* that allows the compute nodes to communicate with each other is the heart of AlphaServer SC systems. The AlphaServer SC Interconnect is a high bandwidth, low latency, low overhead, hardware accelerated communications interface designed to support inter-processor communications. It connects up to 128 compute nodes today, with plans to scale to 4,000+ nodes over the next several years.

Storage on AlphaServer SC systems must meet the demands of high performance and high capacity. Capacity demands are in the terabytes, and continue to grow. AlphaServer SC systems address this through Compaq's StorageWorks storage area networks (SAN). Storage works supports fibre channel connection of multiple host systems to high capacity, high performance, high availability RAID controllers. The hardware connectivity is augmented through the Cluster File System and Parallel File System (both are described in a later section). Finally, the AlphaServer SC architecture supports a combination of local storage and global (SAN) storage.

Alpha 21264 Processor

The 21264 is a 64-bit superscalar microprocessor with out-of-order and speculative execution. Out-of-order execution implies that instructions can execute in an order that is different from the order that the instructions are fetched. In effect, instructions execute as soon as possible. This allows for faster execution since critical path computations are started and completed as soon as possible. In addition, the 21264 employs speculative execution to maximize performance. It speculatively fetches and executes instructions even though it may not know immediately whether the instruction will be on the final execution path. This is particularly useful, for instance, when the 21264 predicts branch directions and speculatively executes down the predicted path. The sophisticated branch prediction in the 21264 coupled with speculative and dynamic execution extracts the most instruction parallelism from applications. Branch prediction and speculative and dynamic execution are part of the hardware run-time environment; applications take advantage of these capabilities without modification or re-compilation.

The 21264 memory system is another enabler of the high performance levels of the 21264. Large on chip and off-chip caches provide for very low latency /high bandwidth data access – the L1 cache delivers over 10.6 GB/sec of bandwidth. The 21264 added a *memory prefetch* instruction, which allows data to be requested before it is used. Previously, a register was used every time data was fetched from memory. The prefetch instruction allows data to be requested from (relatively) slow main memory so that it will be resident in cache when it is actually used. The Alpha compilers have been enhanced to issue these prefetch commands before the data is needed – this greatly reduces effective memory latency and dramatically improves memory performance. In addition, many memory references can be serviced in parallel to all caches in the 21264 as well as to the off-chip memory system. This allows for very high bandwidth data access.

The Alpha 21264 processor is described in considerable detail in a separate technical paper *Exploring Alpha Power for Technical Computing.* This paper is available from Compaq in hardcopy under part # 11CF-0500B-WWEN, or from the web at http://www.compaq.com/hpc.

AlphaServer ES40 Compute Nodes

The ES40 nodes are shared memory symmetric multi-processor (SMP) systems built with a high performance crossbar switch for maximum performance. The ES40 is an ideal building block for AlphaServer SC systems. It uses the Alpha 21264a processor, provides 5.2 GB/sec of memory bandwidth, supports two independent 64-bit/33 MHz PCI buses and supports up to

32 GB of memory. It combines powerful processing, high performance memory subsystems and fast I/O. The use of two independent 64-bit PCI buses allows one 64-bit PCI bus to be dedicated to cluster communication and the second 64-bit PCI bus to be used for other I/O operations such as storage.

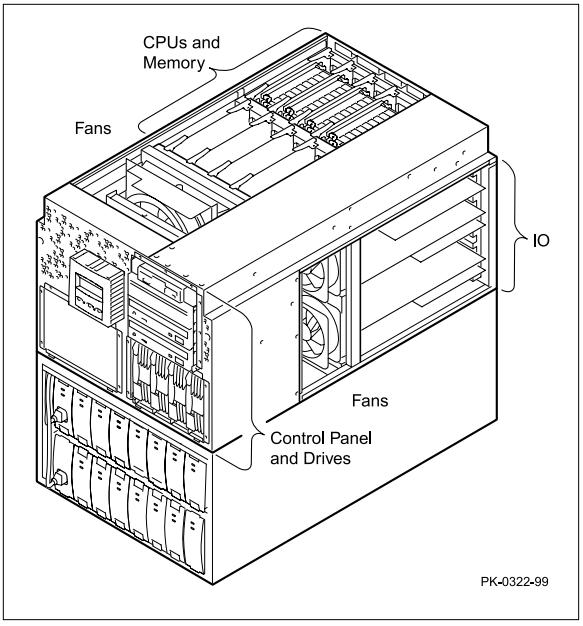


Figure 12: AlphaServer ES40

The AlphaServer ES40 is described in considerable detail in the previously mentioned technical paper, *Exploring Alpha Power for Technical Computing.*

Parallel programming on the ES40, with its shared memory model, is typically done using *threads.* This is a familiar programming model which provides a powerful mechanism for communicating between the processes making up a parallel job. The threads implementation used is the Compaq Tru64 UNIX *Pthreads* package. This is a full implementation of the industry standard Posix Threads specification, and is used on all Alpha systems running Tru64 UNIX.

As threads provide a low-level programming interface, they can be quite complex to use. The preferred way to develop software is to use OpenMP. OpenMP is a higher level interface that allows the compilers to do the detailed work of implementing parallel code. Compaq compilers support the OpenMP specification, simplifying parallel development.

AlphaServer SC Interconnect

The AlphaServer SC Interconnect consists of three hardware components: a Elan Adapter Card that plugs in to the AlphaServer compute node, a 16 port or 128 port interconnect switch, and an ultra high frequency interconnect link cable.

AlphaServer SC Elan Adapter Card

The Compaq AlphaServer SC Elan Adapter Card is a high performance network interface card, based on the Quadrics Elan communications device. Each node in a Compaq AlphaServer SC system requires one Elan Adapter Card. The card provides a single high speed link for connection to AlphaServer SC high speed network switches, such as the AlphaServer SC 16-port Switch and AlphaServer SC 128-port Switch.

The Elan Adapter Card is an intelligent device that implements communications protocols internally. It uses a hardware DMA interface, and is able to transfer data with minimal system overhead. Benchmark results show a typical CPU utilization of less than 2% for MPI communications through the Elan Adapter Card, compared to 35% CPU utilization seen when running GigaBit Ethernet.

The Elan Adapter Card is capable of transferring data at over 360 MB/s, when used on a system capable of supporting 64 bit/66 MHz PCI. The AlphaServer ES40 implements dual 64 bit/33 MHz PCI controllers, each providing a peak performance of 267 MB/s. Delivered MPI bandwidth of the Elan Adapter Card on an AlphaServer ES40 has been measured at well over 200 MB/s. As the Elan Adapter Card can totally saturate the PCI bus, it is normally configured as the only device on a PCI bus. The dual PCI bus design of the AlphaServer ES40 allows it to effectively support the Elan Adapter Card and other high performance devices, such as Fibre Channel storage.

The Elan Adapter Card has performance headroom to support future generations of AlphaServer SC systems.

The Elan Adapter Card has the following features.

- Custom design 100MHz dedicated I/O processor
- 8KBytes on board cache
- MMU (Memory Management Unit) with hardware tablewalk and 16 entry TLB (translation lookaside buffer)
- DMA (Direct Memory Access) engine
- 64MBytes SDRAM
- Byte wide, 400MHz bi-directional links with 400 MB/sec bandwidth

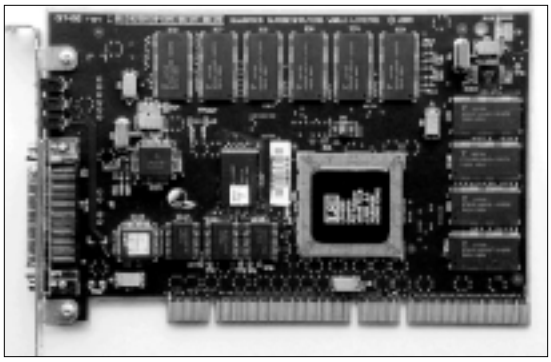


Figure 13: Elan Adapter Card

The Elan Adapter Card is a Universal, short, 64 bit PCI card which conforms to PCI specification 2.1. It is supplied with a Quadrics link cable for connecting to the network switch.

The Elan Adapter Card communicates through custom protocols that are designed to fully exploit the performance of the Elan Adapter Card. These protocols are not available to applications programs; applications communicate through the Elan Adapter Card by using standard MPI and shmem programming interfaces.

Compaq AlphaServer SC High-Level Switch Cards

Compaq AlphaServer SC Interconnect data networks are built from network switch cards housed in a either a 16 port low profile standalone enclosure or a 128 port scalable switch chasis. The network switch cards use crossbar switching technology to provide point to point connections, scalable bandwidth and low latency.

The fundamental building block of the network switch is the Quadrics crossbar switch chip. This chip is an 8-way cross-bar switch used to construct a multi-stage network of links connecting all nodes. The switch configuration allows simultaneous connectivity between all nodes.

Each adaptor provides access for nodes to connect to the previously described fat tree network constructed from 8-way crossbar switches. Each switch chip rolls up the functionality of eight of the unidirectional two way switches with provision for two virtual channels to minimize contention. Bandwidth is constant at each stage of the network, and there are as many links out (for expansion) as there are processors. Larger networks are constructed by taking four (sub-) networks and connecting them with a higher stage of switches.

The data network can be replicated to increase bisectional bandwidth (using the nodes multiple independent PCI buses) and increase the tolerance to failures. Each node can have 2 or more network adaptors connecting them to separate layers (sometimes called rails). This functionality is planned for future versions of AlphaServer SC systems.

AlphaServer SC 16-Port Switch Card

The AlphaServer SC 16-Port Switch Card is a network switch card built from eight Quadrics crossbar switch components. These eight switches are arranged in a two stage ***fat tree*** topology, to produce a 16-way switch.

AlphaServer SC High-Level Switch Card

The AlphaServer SC High-Level Switch Card is a network switch card containing a single Quadrics crossbar switch component. It provides the 8 ports in the top stage of an AlphaServer SC 128-port switch. Its 8 links connect a given link from each of the AlphaServer SC 16-Port Switch Card front cards

AlphaServer SC 16-Port Switch

The AlphaServer SC 16-Port Switch is a standalone, high speed switch unit. It provides Quadrics link ports for connecting up to sixteen separate computer systems, each fitted with an Elan Adapter Card network adapter. The unit contains a single AlphaServer SC 16-Port Switch Card. The AlphaServer SC 16-Port Switch Card switch card uses eight Quadrics crossbar switch components. These eight switches are arranged in a two stage, ***fat tree*** topology to create a 16-way switch.

The 16-Port Switch has a 2U high, 370mm deep, screened enclosure, suitable for mounting in standard 19” equipment racks. It is air cooled, with the air flowing from front to back through the unit.

AlphaServer SC 128-Port Switch

The AlphaServer SC 128-Port Switch is a high speed 128 port switch unit. It provides Quadrics link ports for connecting up to 128 separate computer systems, each fitted with an Elan Adapter Card network adapter.

The crossbar switches are connected in a 3 stage ***fat tree*** topology. The unit contains eight slots for AlphaServer SC 16-Port Switch Cards. Each 16-Port Switch Card has 16 Quadrics link ports for connecting up to sixteen separate computer systems, making 128 ports in total. The link format is byte wide in each direction at 400MHz.

The 128-Port Switch has a 17U high, 450mm deep, screened enclosure, suitable for mounting in a 19" equipment rack. It is air cooled, with the air flowing from front to back through the unit.

The power consumption is 1.2KW. The 128-Port Switch uses a separate 4U rack mounted redundant power supply, accommodating three 600W power supply modules for N+1 redundancy. The switch unit requires that any two of these power supply modules be functioning correctly.

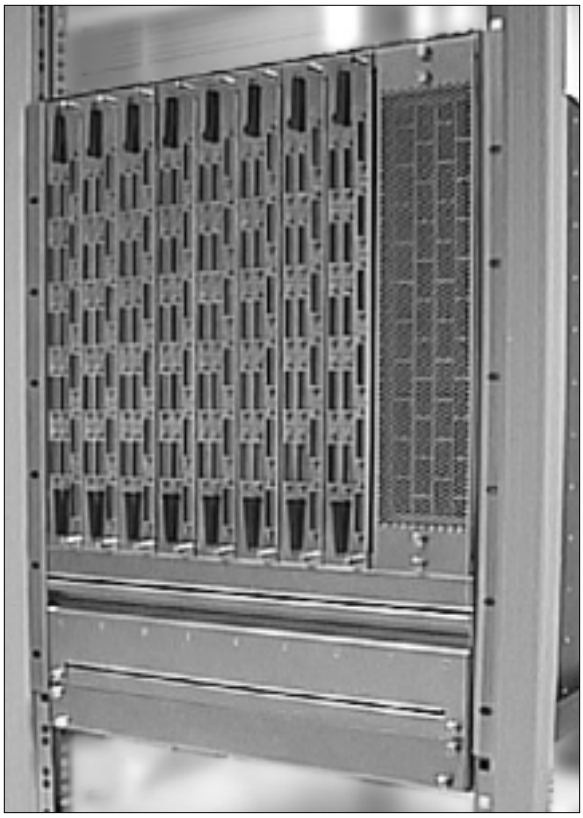


Figure 14: AlphaServer SC 128-Port Switch Mounted in Rack

AlphaServer SC Internal Communications

Networks

Each AlphaServer ES40 compute node is connected to the following networks:

AlphaServer SC Interconnect

The interconnect provides the high-speed message passing and remote memory access capability for parallel applications. Multiple rails of this network can exist. Multiple rails are used to increase aggregate throughput and to reduce queuing delays on systems with large numbers of CPUs per SMP node.

Internal management network

This Ethernet network is used to monitor and control all of the major system components (system nodes, data switches, and so on). This traffic is separated from the data network to avoid perturbing parallel application performance. If the AlphaServer SC Management Server is present as the Root Console, it is connected to the internal management network as well to the external site network. This allows the AlphaServer SC Management Server to act as external portal for control of the system.

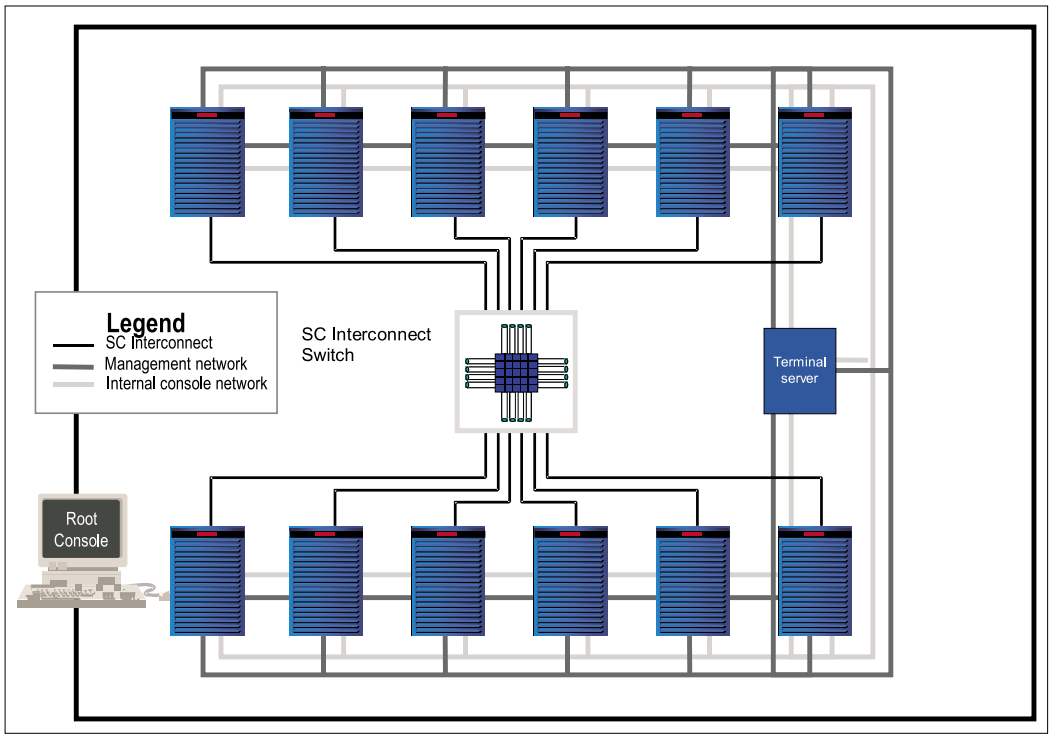
Internal console network

The serial port of each system node is connected to a terminal server. The terminal servers are connected to the internal management network. This configuration enables each node's console port to be accessed through IP over the management network. This facility provides management software with access to a node's console subsystem (for boot, power control, configuration probes, firmware upgrade, and so on).

Node Local Storage

Node local storage is internal to a node's cabinet. Typically, it used to store data that is intimately associated with the node itself; this includes the node's bootable operating system, swap space, and node temporary files. This internal storage should be considered volatile; it is not a system goal to make this storage highly available. All of the system data (the bootable operating system) can be automatically regenerated in case of complete hard disk failure. Data stored on local storage is node-specific and the loss of a node should not have an impact on the data requirements of the rest of the system.

Figure 15: AlphaServer SC System Components



Each node is configured with two internal drives. Under normal operation, the secondary drive holds a copy of the primary drive's boot partition. A node can run with a single drive. During the upgrade process, the secondary drive stores the boot partition for the upgraded operating system. This allows fast reversion to the original operating system version, if necessary.

The internal drives are configured at system installation time, but they can be reconfigured by the administrator.

Applications can use data that is stored on file systems on the internal drives, on the understanding that this data is not highly available.

External Storage

External storage is used to hold file systems served to the rest of the system (and potentially, to other external clients). External storage is also used to hold system files and customer data. Any number of nodes, from two nodes to all nodes, can be configured as file server nodes with external storage.

External storage is highly available through the use of redundant paths, RAID, and the failover capabilities of the Cluster File System (CFS). External storage and associated file systems will remain available and accessible if any single component fails, because of the following:

- Each storage subsystem is connected to at least two hosts
- Storage is RAID-protected
- Physical disks are connected to dual RAID controllers
- System storage is configured as external storage for availability.

External Support Systems

Using the external network, the system can connect to a variety of external support servers, for example, file servers and development systems. Alternatively, it is possible to configure an internal partition (consisting of some subset of the computational nodes) to provide login/compile services and user file directories.

External Network Connections

Compute nodes can also be connected to external networks. The number of nodes used to provide such connections and the type of network interconnect used can be specified. Standard system interconnects (FDDI, Ethernet, Gigabit Ethernet, ATM and HiPPI) are supported.

IP aliasing is used to present an AlphaServer SC system with multiple external interconnects as a single network entity.

Systems can also be configured with an optional front end. The *AlphaServer SC* Management Server is a server running the *Tru64 Unix* operating system, but it does not have to be the same server type as the system nodes. If this node is present, it can be configured to do the following:

- Act as a Remote Installation Server (RIS) for the installation process
- Serve user home directories
- Act as a development server
- Host the central management functions and daemons for the system

Root Consoles

The system is also equipped with a graphics terminal that acts as the root console (the console on system nodes 0 or 1). This is used during the initial installation process to install and configure the root node. After the installation procedure, console access to all nodes is possible via management software. At this stage, the root console can be used to log in to the system.

AlphaServer SC Software Architecture

Compaq *AlphaServer* SC System Software provides the utilities and libraries for the operation and management of the SC systems. This software includes capabilities which supply the infrastructure for extreme network performance of parallel applications. In addition, the software provides users and system administrators with highly available single system management (SSM) of the nodes of the system, and thus minimizes the effort and complexity of both the administration and the usage of the system.

AlphaServer SC System Software organizes a system into a small number of cluster file system domains of up to 32 nodes per CFS domain. Each CFS domain provides a single namespace for files and directories, including a single root file system that is shared by all members of the domain. A Parallel File System can be layered on each cluster file system domain. The SC System Software also includes a cluster alias feature for the Internet protocol suite (TCP/IP) so that a domain appears as a single system to its network clients. Because, under SC System Software, the nodes in a domain share system and common system configuration files, management tasks need to be performed only once within the domain, rather than repeatedly for each individual node. The cluster manages as a single entity, rather than disparate cooperating systems, in effect, the cluster is "the system". This means tasks such as software installation and upgrades are now performed once. For example, you install Compaq FORTRAN just once per SC domain rather than once on each node. In addition, most network applications need to be configured only once for the domain.

AlphaServer SC System Software arranges a system into parallel execution partitions. The partitions are managed and scheduled by a job management facility, a part of RMS, which implements RMS scheduling of parallel jobs, partition management, user quotas and job accounting. These functions may be further enhanced through the integration of other resource and batch-management products such as LSF (Load Sharing Facility) from Platform Computing or the public domain Portable Batch System (PBS)

The system software, including the AlphaServer SC System Software, the *Tru64 UNIX* Operating System, and the Advanced File System Utilities, resides primarily on storage systems served by one of two designated system nodes of each of the domains of the *SC* system. The other node is capable of serving the storage system to provide load balancing and in case of failover. System utilities are provided for easy replication of system files between domains, thereby further simplifying overall management of the entire SC system.

Single system management, performance visualization, and hardware diagnostics are each done from centralized user interfaces, which extend the system management utilities of *Tru64 UNIX*.

AlphaServer SC System Software includes the drivers and low-level libraries for the *AlphaServer* SC Elan cards. The low-level Shmem library is provided for direct access to the memory of remote processes. In addition, a highly optimized MPI message-passing library is included.

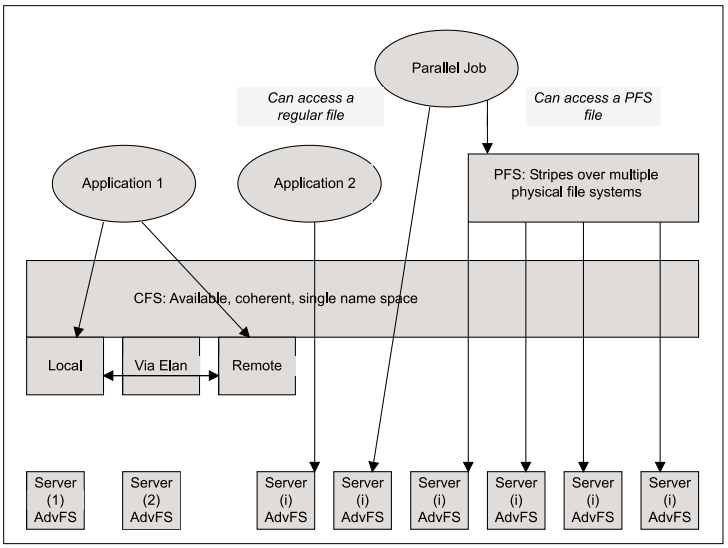


Figure 16: File Systems

File Systems

Through the system software, AlphaServer SC systems provide access to three primary file systems:

• Cluster File System (CFS)

• Parallel File System (PFS)

• Externally Served Network File System (NFS)

These file systems are layered on top of AdvFS, the Tru64 UNIX Advanced File System. These file systems are illustrated below:

Cluster File System (CFS)

CFS is a file system service that integrates all of the underlying file systems within a CFS domain. CFS does not provide disk-structure management; it uses the capabilities of the serving file system for this. The underlying serving file system used is the standard Compaq AdvFS product, with no changes to on-disk structures.

CFS is a POSIX and X/Open compliant file system. CFS provides the following capabilities:

A single coherent name space

The same pathname refers to the same file on all nodes. A file system mount on any node is a global operation and results in the file system being mounted at the same point on all nodes.

Global root

The point of name space coherency is at the root of the file system and not at a subordinate point; therefore, all files are global and common. This enables all nodes to share the same files, for example, system binaries, and global configuration and administration files.

Failover

Because the file system capability is global, CFS will detect the loss of a service node. CFS will automatically move a file service from a failed node to another node that has a path to the same storage. In-flight file system operations are maintained.

Coherent access

Multiple accesses of the same file will give coherent results. (Though this mode of access is less common with high-performance applications, and incurs a performance penalty, it is essential for enterprise applications such as database access and transaction processing.)

Client/Server file system architecture

Each node's local file system acts a server to other nodes. Each node is also a client of other nodes.

Support for node-specific files with the same pathname on each node

This is implemented through a Context Dependent Symbolic Link (CDSL) - a file link with a node identifier in the link name. CDSL is a feature of CFS. The node identifier is evaluated at runtime and can be resolved to a node-specific file. This can be used to provide, for example, a node-specific */tmp* directory. This feature support is used to provide node-unique files and to optimize for local performance.

The diagram below illustrates a sample CFS file hierarchy. The member boot partitions are mounted on node local storage. The system components of the root partiton will be mounted on external storage on one of the system file servers. Other nodes serving external storage can mount their file systems at other points within the hierarchy; this is illustrated by */a* and */b* in the diagram.

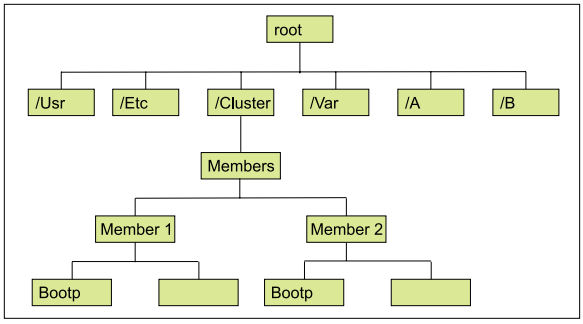


Figure 17: Sample *AlphaServer* SC CFS File Hierarchy

The operation of CFS is described in detail in the Compaq *AlphaServer* SC System Administration Guide.

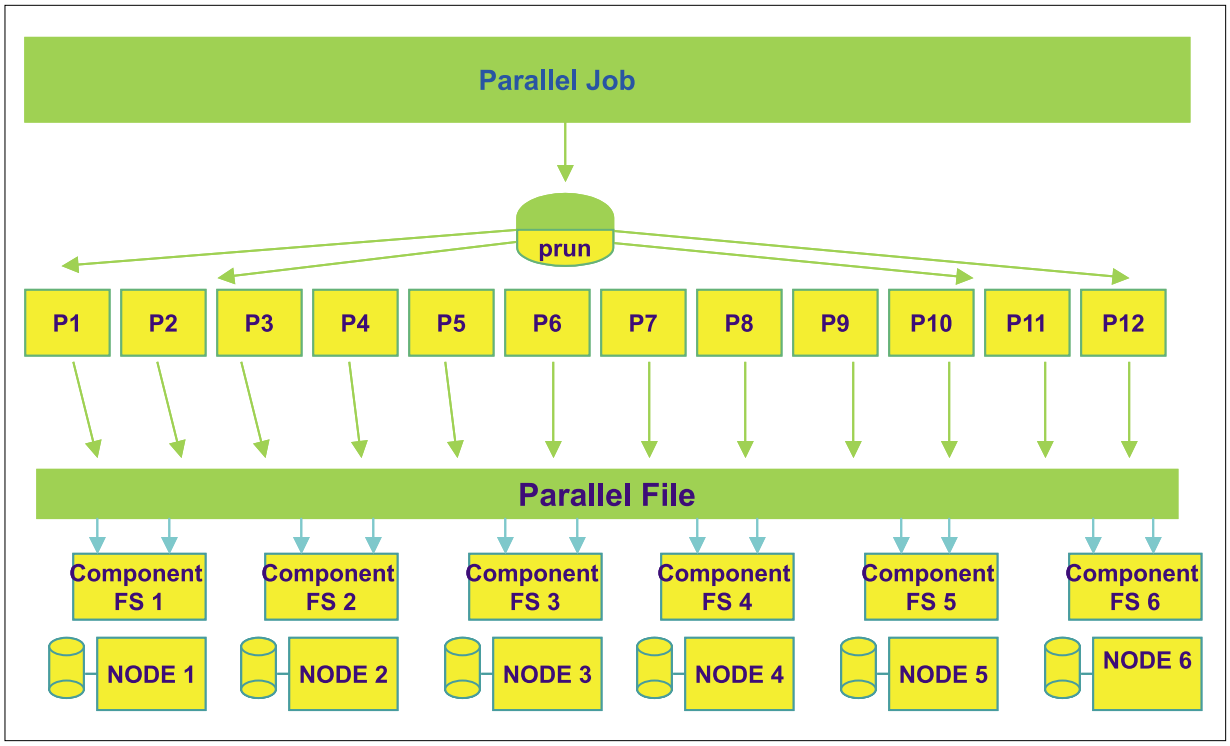


Figure 18: Parallel File System

Parallel File System (PFS)

CFS provides the ability to scale the total file system bandwidth for accesses to files on different servers. PFS delivers scalable bandwidth capability to a single job within a CFS domain by striping the data of a single parallel file system over multiple underlying file systems. This allows accesses from different processes within a single parallel job to proceed in parallel and to be served by concurrent file servers. Contrast this with multiple processes accessing the same file served by a single file server.

When a parallel file system is created by an administrator, the constituent file systems and the default stripe size are specified.

For the programmer, PFS uses POSIX-compliant syntax and semantics. It supports additional file system commands (implemented as *ioctls*) that allow a job to interrogate and modify attributes of the file system. For example, determining which file system components are local to a process, modifying the stripe size of a file, and so on. This enables a parallel job to optimize its I/O with respect to the underlying PFS structure. The operation of PFS is described in detail in the Compaq *AlphaServer* SC System Administration Guide.

Externally Served Network File System (NFS)

Just as with any SMP server, *AlphaServer* SC systems can mount file systems exported by other file servers. This means that user home directories can be located outside the *AlphaServer* SC system, and data on other servers can be accessed. Multiple external file systems can be mounted. It is possible to mount different external servers on different *AlphaServer* SC computational nodes.

Nodes that mount external file systems act as clients to those external systems. These nodes also act as servers of the imported file system to other nodes within the *AlphaServer* SC system - that is, external file systems only need to be mounted by a single node to be available to all nodes.

External Networking and I/O

AlphaServer SC systems connect to external LANs using network interfaces on a subset of the nodes. Standard network interfaces are supported, including Fast Ethernet, FDDI, ATM, and HiPPI.

Any subset of nodes can support external interfaces, ranging from one node to all nodes.

AlphaServer SC systems employ a facility known as Cluster Alias (CA) that allows each CFS domain to be represented by a single IP address or hostname. Multiple aliases can be defined for each CFS domain. The administrator can define additional aliases and their constituent nodes. The administrator can also define the alias capabilities of the services that run on the system, as follows:

- A service can be designated to run on a single node only, with transparent failover to another node.
- A service can be defined to run on multiple nodes; the CA facility will load-balance between those nodes running the service.
- A service can be defined as accessible only through an IP alias.
- A service can be defined as accessible only through a real IP address.

Services establishing outward connections will, by default, use the default alias as their source address.

The Cluster Alias capabilities enable the system administrator to configure subsets of nodes to provide specific services in available single server mode or in multiserver load-balanced mode. Clients of these services do not need to be exposed to the multiple node nature of the *AlphaServer* SC system.

Job Management

The RMS job management facility allows the administrator to treat the total set of CPUs on the AlphaServer SC system as a single large pool. The administrator can subdivide the pool into smaller partitions for use by different classes of job.

When a user executes a job, the job management system is responsible for selecting the set of nodes that best fits the job's needs and for creating the requisite processes on each node.

The total computing space provided by the nodes can be subdivided into non-overlapping partitions. A partition is a collection of SMP nodes; it is not possible to subdivide the resources of a single SMP node between partitions. The system administrator can use partitions to allocate compute resources according to enterprise needs; for example, a system administrator may create partitions for development, production, and interactive work.

The system administrator can define multiple configurations, each with a different set of partitions. Different configurations can be applied, for example, for successive shifts. Partition types include "interactive" and "parallel". Login is disabled for nodes defined as being part of parallel partitions. Interactive nodes can be used for login and code development.

When a user runs a parallel job, a number of parameters can be specified, including:

- The partition that is to be used for the job.
- The number of processes in the job.
- The number of CPUs per process.

If sufficient CPU resources are available, the scheduler creates the required processes on the specified nodes. The processes that make up a parallel job are constrained to run on the set of CPUs specified by the RMS scheduler. If processes within the job create child processes, these child processes are subject to the same constraints. When a job is terminated or de-scheduled, all processes are handled in concert, including any child processes created after job startup. This ensures that CPU resources can be managed reliably.

The job management system supports the time-sharing of a partition between different long-running jobs: the system deschedules the set of processes that constitute the job and schedules the set of processes for the job that is to be run. The RMS scheduler works by giving the operating system scheduler no work to do; that is, it ensures that there are no other active processes competing for a CPU's cycles. This is known as "time-sliced gang scheduling." Ensuring that all processes of a parallel job make even progress is critical to job performance.

When a partition is designated as time-shared, the RMS scheduler will group jobs of equal priority using an administrator-specified time slice. (The time slice is specified in seconds or minutes.) When jobs have been grouped in this way, the scheduler will schedule a new job only if sufficient memory resources are available to run the new job and the existing jobs. Swapping and paging are not practical because of the large memory sizes and slow page rates involved. The job management system accounts for resource usage on a per-job basis; that is, total resource usage is aggregated. In addition, the job management system can be used for access control. The resources in each partition are managed by a Partition Manager, which mediates user requests, checking access permissions and resource limits before scheduling the user's jobs. RMS accounts for resource usage on a per-job basis. Total resource usage is aggregated. The RMS database is accessible through an SQL interface.

RMS has also been integrated with LSF (Load Sharing Facility), a workload management system from Platform Computing. LSF allows users to run sequential and parallel jobs by submitting them to batch queues, which dispatch jobs to the best available systems and partitions, according to scheduling policies set by the organization. LSF provides significant benefits, including increased throughput, fair sharing of resources among users, teams or projects, and increased workload reliability. LSF is available from Compaq, and directly from Platform. More information about LSF can be found at the URL: http://www.platform.com.

System Administrator

Tru64 UNIX SysMan Menu and SysMan Station together provide a single-system interface to management of the *AlphaServer* SC configuration, and may be used to determine the state of availability and connectivity in the system. A choice of graphical, Web-based, or command-line user interfaces makes management and file-manipulation tasks easier for the administrator, flexible for those with large configurations, and streamlined for users.

Most of the system management and configuration files used by the system are global and common. With the use of CFS, operations on these files only need to happen once for the changes to be visible to all nodes within each CFS domain. For example, the password file is global — adding a user to the password file automatically creates an account for the user on all nodes within each CFS domain. Apart from system installation and physical maintenance, the system can be managed remotely. The system can be managed and monitored as a single entity. Monitor display provides the ability to aggregate collections of components into a single entity. Any fault or attention state in a sub-component is reflected in the parent entity. The management user interface allows the system administrator to drill down from the wide system view into nodes and within a node.

System administration utilities can have a global CFS domain or local focus. When the global focus is selected, commands are applied to the entire domain. Frequently, this will mean updating a single file in global CFS space; at other times, daemons may need to be restarted on all nodes. When the local focus is selected, the administration command is applied to the specified node. This would apply, for example, to changing configuration settings on a network interface card on a particular node. The system management utilities are *locus aware*, that is, they are aware of whether they apply to the global or local context and will report an error if used incorrectly.

AlphaServer SC System Software includes highly automated system installation capabilities with supporting components, including console software. The capabilities provide an interface to the management functions that build the system software on the disks local to each node. They automate the setting of SRM (System Resources Manager) console parameters on all nodes. They are also used to set up and manage DECserver terminal servers into which the individual node console ports are connected, to manually connect to the node consoles and to log console output.

AlphaServer SC System Software can perform hardware error diagnostics and initiate remedial actions in the case of node failures in the *AlphaServer* SC system. Hardware-failure events are logged and can be monitored with the SysMan utilities. In addition, the events can also be configured to trigger actions such as sending email to the system administrator. When appropriate, events will initiate console functions that cause failed hardware components to automatically fail over to replacement components and then to automatically restart the failed hardware. Network behavior is controlled and monitored by the Switch Manager, which samples the network control interface, checking for network errors and monitoring performance. When network problems are found, events are posted and can be monitored.

Programming AlphaServer SC Systems

Introduction

Programmers can view an *AlphaServer* SC system as a pool of serial processors, a pool of shared memory SMPs, or a distributed memory multi-processor. Each of these views is supported by one or more programming models, compilers, libraries, and tools. Typically, programs will be written using Compaq FORTRAN, Compaq C, or Compaq C++. Compaq FORTRAN and Compaq C both include support for the industry-standard OpenMP directives, which are widely used for parallel development on shared-memory SMP systems

Most large-scale jobs that use the full capability of the *AlphaServer* SC system will be programmed using a message-passing model (multiple processes and address spaces) via the Message Passing Interface (MPI) library. In some cases, higher performance will be achieved through the deployment of a hybrid-programming model, where a set of shared address space threaded processes (written, for example, with OpenMP) pass messages *via* MPI. For the highest possible performance, but at the expense of additional complexity, programmers can use the shared-memory Shmem facility to take the fullest advantage of the *AlphaServer* SC Interconnect's extremely low-latency one-sided communication capability. Yet another alternative for parallel development and execution, the Compaq FORTRAN compiler includes High Performance Fortran (HPF).

RMS provides a set of commands for running parallel programs and monitoring their execution. The set includes utilities that determine what resources are available and commands that request allocation of resources.

More on Parallel Programs

A parallel program consists of a controlling process (prun) and a set of application processes distributed over the nodes in a partition. Each application process can have multiple threads running over one or more CPUs inside the AlphaServer ES40 compute nodes. RMS includes MPI and Shmem libraries that have been optimized for the AlphaServer SC Interconnect

The *prun* command sends a request to the Partition Manager to allocate the resources (CPUs and memory) required by the parallel program. Once the resources are available, the Partition Manager instructs the RMS per-node daemons to load an RMS process called rmsloader on each node.

Software Development and Tools

MPI

The MPI library is a standard message passing library for parallel applications. Using MPI, parallel processes cooperate to perform their task by passing messages to each other. MPI includes point-to-point message passing and collective operations between a user-defined group of processes

The *AlphaServer* SC MPI library is an optimized implementation of the MPI-1 specification and is based on MPICH Version 1.1.1 from Argonne National Laboratory and Mississippi State University. Fortran and C interfaces are provided. The *AlphaServer* SC MPI library is layered on top of tagged message passing routines that are especially designed for the *AlphaServer* SC Elan cards, and that cause highly efficient communication between nodes connected by the *AlphaServer* SC Interconnect. In addition, highly efficient communication within a node occurs through direct memory transfers done with minimal data movement. A number of environment variables can be set to help optimize the performance of MPI programs, and these are described in the software documentation.

Support is included for a large subset of the MPI-2 I/O interface via the ROMIO package from Argonne National Laboratory. Performance of the MPI-2 I/O operations will be related to the attributes of the Parallel File System.

The message-passing routines support the following:

- Blocking (synchronous), point-to-point send and receive
- Non-blocking (asynchronous), point-to-point send and receive
- Collective message-passing operations derived from the four primitives: broadcast, scatter, gather and reduce In addition to the communications routines, MPI provides the following categories of service:
- Environmental queries
- Timing information for measuring performance
- Profiling information for monitoring performance

The MPI library is layered on top of the tagged message passing routines provided by the Elan library. These routines make use of tagged message ports, known as tports, for point-to-point communications.

On a SMP node, the tagged message passing routines (and, hence, MPI) use shared memory segments to communicate between processes on the same node and the Compaq AlphaServer SC Interconnect to communicate between nodes.

Shmem

The Shmem library provides direct access (via put and get calls) to the memory of remote processes. A message passing library, such as MPI, requires that the remote process issue a receive to complete the transmission of each message; the Shmem library, by contrast, provides the initiating process with direct access to the target memory. The one-sided communication used by Shmem maps well onto the DMA hardware in the Compaq AlphaServer SC network adapter. A consequence of this is that Shmem latencies are even lower than MPI latencies.

Shmem provides the following categories of routine:

- Put routines write data to another process.
- Get routines read data from another process.
- Collective routines distribute work across a set of processes.
- Atomic routines perform an atomic fetch-and-operate, such as fetch-and-increment or swap.
- Synchronization routines order the actions of processes. For instance, the barrier routine might be used to prevent one process from accessing a data location before another process has updated that location. The Shmem programming model requires that you think about the synchronization points in your application and the communication that must go on between them.
- Reduction routines reduce an array to a scalar value by performing a cumulative operation on some or all of the array elements. For example, a summation is a reduction that adds all the elements of an array together to yield one number.

The Shmem library also includes a number of initialization and management routines.

Shmem routines provide high performance by minimizing the overhead associated with data passing requests, maximizing bandwidth and minimizing data latency (the time from when a process requests data to when it can use the data). By performing a direct memory-to-memory copy, Shmem typically takes less steps to perform an operation than a message-passing system. For example, in a generic message passing system, for a put operation the sender performs a send, then the receiver performs a receive; for a get operation, the requesting process sends a description of the data required, the sender acts on the request by sending the data, then the requesting process receives the data. By contrast, Shmem requires only one step: either send the data or get the data. However, additional synchronization steps are almost always required when using Shmem. For example, the programmer must ensure that the receiving process does not try to use the data before it arrives.

TotalView

TotalView is the source-level dbugger for Compaq *AlphaServer* SC systems. TotalView is licensed from Etnus. Their website is http://www.etnus.com and free 30-day demos are available from www.etnus.com/trybuy/totalview/index.html

TotalView has an easy-to-use graphical interface, along with a command line interface for scripting portions of your debug sessions, writing debug macros, or for simply debugging from a textual interface for those times when a GUI isn't convenient.

TotalView supports debugging of both serial and parallel programs. Not only does it have full support for C++ and F90, but also it supports more parallel programming models than any other debugger, including MPI, OpenMP, threads, shmem, and more.

TotalView runs on ths same node as you run prun. It starts a remote server process called the TotalView Debugger Server, tvdsvr, on each of the nodes used by your parallel program.

TotalView lets you control how you want to debug your program. You can debug at the program level, groups of processes, single process, or thread level.

TotalView has 5 types of breakpoints, providing addition flexibility and control over your program. TotalView 4.1 also introduces data watchpoints on the Compaq platforms. If you set a watchpoint on a variable in which you are interested, TotalView will notify you when that variable has changed. With so many types of breakpoints, and the addition of watchpoints, your debug session is under your control, not the debugger's.

TotalView cooperates with RMS to perform the following functions:

- Automatically acquire the processes spawned by prun at startup, before they enter the main program
- Attach to a parallel job started by prun and automatically acquire all of the processes in the job, wherever they reside in the machine.

TotalView lets you start a program under it's control, debug a core file, or attach to a running process (not started under TotalView's control) that appears to be hung.

If your program has large amounts of data, TotalView has many features to help you manage and analyze that data. For example, you can

- "dive" (click the right mouse button) over any construct to obtain more information about it
- view statistics about an array (min, max values, for example, or the number of elements that have values of NaN or INF)
- filter out data that you find uninteresting so that you only have to deal with the data you really need
- visualize your data
- sort data, or only view particular dimmensions of an array

TotalView is not sold by Compaq, but rather directly from Etnus. See the Etnus website for more information about TotalView, to obtain a free demo, or for pricing information.

Vampir

You can use Vampir version 2.0 and Vampirtrace version 1.5 to prepare, build, trace and analyze MPI programs running on Compaq *AlphaServer* SC systems. Vampir is a visualization and analysis tool for MPI programs. Vampirtrace is the Vampir MPI profiling library. The Vampir software is available from Pallas GmbH. Their web site is http://www.pallas.com

After installing Vampir, you link your MPI program with Vampirtrace. When you run the program, Vampirtrace uses the MPI profiling interface to gather information about the program's execution behavior. The information is kept locally in each processor's memory and saved in a trace file when the program exits. The trace file is then fed into Vampir for analysis.

Performance Visualizer

Performance Visualizer is a performance and resource-monitoring tool. Developers of parallel applications can use Performance Visualizer to monitor execution of their applications on the nodes of the *AlphaServer* SC system. With this information, they can redesign their applications to spread the load more evenly and reduce overall execution time. System managers can use Performance Visualizer by selecting one or more graphical displays to help them to quickly identify overloaded hosts, underutilized resources, active users, and busy processes.

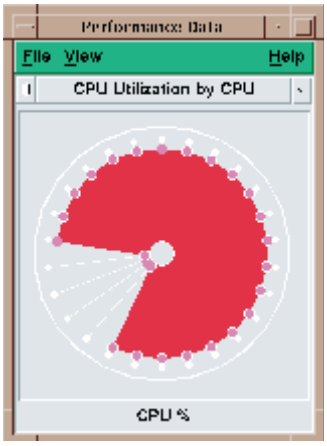


Figure 18: Performance Analyzer Kiviat Chart

LSF: Load Sharing Facility

LSF is a product of Platform Computing, available from Compaq and directly from Platform, with information available at http://www.platform.com. LSF, currently at version 4.0, is a workload anagement system that manages jobs across the partitions and computational nodes in an AlphaServer SC system, and across multiple systems in a network.

LSF uses a 5 step process:

1. Collect information on the status of all resources in all systems. A resource can be any hardware or software component, such as CPU load, available memory, available file space, application licenses, etc.
2. Choose the best available resource to process each job.
3. Schedule the job based on rules, policies and priorities. This may include holding a job until a certain license is available, preempting the queue for emergencies, ensuring fair sharing of resources, event driven scheduling, calendar driven scheduling and parallel scheduling.
4. Manage, monitor and balance all jobs until completion. This include restarting and requeing jobs if computational nodes fail or if resources become unavailable.
5. Notify users when jobs are completed and log job statistics for accounting and reporting. LSF extends the Tru64 UNIX job scheduler that runs on each compute node and the AlphaServer SC Resource Management System by providing global batch queuing and job management for an entire AlphaServer SC system, and clusters of SC systems.

Programmer's Toolkit

The *Developers' Toolkit for Compaq Tru64™ UNIX* provides a basic application development tool set for the Tru64 UNIX Operating System. This product combines the previous Developers' Toolkit and C Developers' Extensions products into a single solution.

The Developers' Toolkit for Tru64 UNIX is a prerequisite for all Compaq Tru64 UNIX development tools, languages, and environments. The Developers' Toolkit contains the following components:

- Compaq C for Tru64 UNIX, an ANSI conformant C Compiler

- Debuggers (Ladebug, dbx)
- The ATOM API
- Program analysis tools (profiling and performance analysis)
- Visual Threads (thread-related profiling and debugging)
- Porting Assistant
- Reordering tools (cord, om)

FORTRAN

Compaq FORTRAN supports the FORTRAN 66, FORTRAN 77, FORTRAN 90 and FORTRAN 95 standards. Compaq FORTRAN supports the multivendor OpenMP V1.1 specification, including support for directed parallel processing using OpenMP directives on SMP systems. Compaq FORTRAN supports most statement, library, and directives from the HPF-2 specification.

Compaq FORTRAN provides a multiphase optimizer that is capable of performing optimizations across entire programs. These optimizations include:

- Optimizations of arithmetic IF, logical IF and block IF-THEN-ELSE
- Removal of invariant expressions from loops
- Global allocation of general registers across program units.
- In-line expansion of statement functions and routines.
- Optimization of array addressing in loops.
- Deletion of redundant and unreachable code.
- Loop unrolling.
- Thorough dependence analysis
- Software pipelining to rearrange instructions between different unrolled loop iterations.
- Loop transformation optimizations that apply to array references within loops, including loop blocking, loop distribution, loop fusion, loop interchange, loop scalar replacement and outer loop unrolling.
- Speculative code scheduling, where a conditionally executed instruction is moved to a position before a test instruction and executed unconditionally. This reduces instruction latency stalls.

Compaq Extended Math Library

The Compaq Extended Math Library (CXML) is a set of mathematical subprograms that are optimized for the Alpha architecture. CXML includes:

- Basic Linear Algebra
- Linear System and Eigenproblem Solvers
- Sparse Linear System Solvers
- Sorting
- Random number generation
- Signal processing.

The Basic Linear Algebra library includes the industry standard Basic Linear Algebra Subprograms (BLAS) Level 1, Level 2 and Level 3. Also included are subprograms for BLAS Level 1 Extensions, Sparse BLAS Level 1 and Array Math Functions (VLIB).

The Linear System and Eigenproblem solver library provides the complete LAPACK package developed by a consortium of university and government laboratories. LAPACK uses blocked algorithms that are better suited to most modern computer architectures, particularly those with memory hierarchies (such as Alpha systems). LAPACK will supersede LINPACK and EISPACK for most users.

The Sparse Linear System library provides both direct and iterative sparse linear systems solvers. The direct solver package supports both symmetric and non-symmetric sparse matrices stored using the skyline storage scheme. The iterative solver package contains a basic set of storage schemes, preconditioners, and iterative solvers. The design of this package is modular and matrix-free, allowing future expansion and easy modification by users.

The Signal Processing library provides a basic set of signal processing functions. Included are one-, two- and three-dimensional FFTs, group FFTs, Cosine/Sine Transforms (FCT/FST), Convolution, Correlation and Digital Filters.

Many CXML subprograms are optimized for Alpha systems. Optimizations include traditional optimizations such as loop unrolling and loop reordering. CXML subprograms also provide efficient management of cache, using techniques such as:

- Reuse of data within registers to minimize memory accesses.
- Efficient cache management.
- Use of blocked algorithms that minimize translation buffer misses and unnecessary paging.

CXML supports SMP for improved performance. Key BLAS Level 2 and Level 3 routines have been modified to run in parallel on SMP systems. These parallel routines are supplied in an alternative library. The programmer can choose to link with either the parallel or serial library, depending on whether parallel support is required. Each library contains the complete set of routines.

Cray SciLib Portability Support

SCIPORT is Compaq's implementation of the Cray Research scientific numerical library, SciLib. SCIPORT provides 64-bit single precision floating point and 64-bit integer interfaces to underlying CXML routines for Cray users porting programs to Alpha systems. SCIPORT also provides almost all Cray Math Library and CF77 (Cray Fortran 77) Math intrinsic routines.

SCIPORT provides the following:

- 64-bit versions of all Cray SciLib single-precision BLAS Level 1, Level 2 and Level 3 routines.
- All Cray SciLib LAPACK routines.
- All Cray SciLib Special Linear System Solver routines.
- All Cray SciLib Signal Processing routines
- All Cray SciLib Sorting and Searching routines

Compaq KAP FORTRAN Optimizer

The Compaq KAP FORTRAN optimizer is a FORTRAN language source-to-source preprocessor that restructures code for improved performance on single and multiprocessor systems. The Compaq KAP optimizer performs an interprocedural dependence analysis, allowing it to use advanced optimizations safely. The Compaq KAP optimizer integrates with the Compaq FORTRAN compiler for seamless operation.

Major optimization capabilities include:

- Automatic and directed parallel decomposition for SMP, including
  - Automatic decomposition
  - Guided automatic decomposition
  - Directed decomposition
  - Combined automatic and directed decomposition
- Loop optimizations
- Memory management optimizations
- Scalar optimizations
- Function inlining
- BLAS recognition
- "Dusty Deck" transformations

When used for parallel decomposition, the Compaq KAP FORTRAN optimizers will:

1. analyze the source code, including KAP-specific directives.
2. identify the compute intensive loops that are candidates for parallelization. (Compaq KAP FORTAN can also parallelize FORTRAN 90 array operations.)
3. determine if the selected loops can be safely executed in parallel
4. transform the code, creating and managing the parallel sections.
5. insert all necessary synchronization points.

Compaq KAP FORTRAN optimizer will automatically insert OpenMP directives, taking advantage of the support for OpenMP parallel directives in Compaq FORTRAN.

In addition to increasing programming productivity, allowing the Compaq KAP optimizer to use its extensive analysis capabilities to automatically insert OpenMP directives has the advantage that it can also apply scalar optimizations to the loops.

Because a primary objective of the automatic parallel decomposition capability is to produce a correct parallel program, the Compaq KAP optimizer will skip

loops that cannot be properly analyzed, whether due to lack of information at compile time or to complex program structure. The advanced programmer can often improve parallel efficiency by guiding the optimizer in performing automatic decomposition, supplying missing information through the use of directives, assertions and command line options. This technique is referred to as *guided automatic decomposition.*

Directed decomposition requires the programmer to identify parallel regions and loops, insert parallel directives, and manage all necessary decomposition.

It is possible to do mixed *decomposition*, combining both directed and automatic decomposition within one program. This is done by defining parallel regions with directives, and then using the -concurrent option when compiling to automatically parallelize the remaining regions.

The Compaq KAP FORTRAN optimizer can selectively insert both function and subroutine code into the main code stream, an capability referred to as "inlining." Inlining a function allows the optimizer to include the function code in its interprocedural dependence analysis, resulting in a more thorough analysis and optimization. Inlining also eliminates call overhead and may enable better code scheduling by the compiler.

In addition to inlining, the Compaq KAP optimizer can perform only the interprocedural dependence analysis on a function or subroutine. When this option is selected, the function or subroutine is temporarily inlined while dependence analysis is performed, and then is removed. This provides the advantage of improved interprocedural analysis without the code size expansion that results from inlining.

The Compaq KAP FORTRAN optimizers optionally provide an informational listing of the program, outlining the results of its processing. Information includes:

- Annotated original program listing
- Call tree of entire program
- KAP options used for each program unit
- Loop optimization table for each program unit
- Loop summary table
- Transformed program listing
- Footnotes detailing important actions taken or conditions that may require attention, such as data dependencies that inhibit parallel decomposition.
- Syntax Error/Warning messages
- Questions generated by the optimizer
- Summary of actions taken by the optimizer

"C"

Compaq C for Tru64 UNIX is a standards-compliant, multi-dialect, full-featured, and highly optimizing implementation of the C language specifically developed to exploit the 64-bit Alpha architecture.

It contains extended support for systems programming, parallel programming, and mathematical computing with features including:

- Support for the OpenMP parallel programming API
- Fine-grained control over optimization behavior with #pragmas and command-line options
- User-defined assembly language sequences using #pragmas and intrinsic functions that expand and optimize these instruction sequences in line
- #pragmas and command-line options for controlling data layout
- IEEE floating-point (with optional handling of exceptional values like NaN, INF, and denormals)
- Built-in functions to generate efficient code sequences for processor interlocked memory access
- Shared library support
- Program profiling in conjunction with pixie, prof, gprof and hiprof
- Run-time feedback optimization
- Inter-language calls with other Tru64 UNIX languages

C++

Compaq C++ for Tru64 UNIX is a native C++ compiler which directly generates optimized object code directly from C++ source (no intermediate translation to "C"). It is based on the ANSI/ISO C++ standard, and supports several dialects to simplify porting of applications. These dialects include:

- The *ms* dialect for maximizing compatibility with Microsoft's Visual C++ product.
- The c*front* dialect for compatibility with the AT&T cfront translator
- The *arm* dialect for compatibility with *The Annotated C++ Reference Manual* by Ellis and Stroustrup.
- The *standard* dialect for developers who want to write applications that comply with the ANSI/ISO standard.

Porting Assistant

The Porting Assistant is an integrated graphical toolset designed to reduce the time and cost of porting applications to Tru64 UNIX. It helps when porting applications written in C, C++, and Fortran from other UNIX platforms, such as SunOS, HP-UX, IBM/AIX, and ULTRIX, and from non-UNIX platforms, such as OpenVMS. The Compaq Porting Assistant is a Tru64 UNIX/Motif based toolset.

The Porting Assistant guides the developer through the porting process by suggesting a systematic, iterative porting approach. First, the Porting Assistant analyzes source files and locates changes that are potentially needed for the application to run on Tru64 UNIX. Then, through extensive, graphical hyperlinked information, it helps programmers understand what changes are needed and why they need to make them. Finally, it aids in making each change, either by using the integrated editor or through its global replace capability.

In analyzing software code, the Porting Assistant locates:

- 32-bit dependencies
- Conditional code that might also need Tru64 UNIX branching
- References to files that do not exist on Tru64 UNIX
- Calls to library functions that do not match Tru64 UNIX definitions or have different semantics on Tru64 UNIX
- Platform-dependent signal handling code
- Function/subroutine calls in Fortran that are inconsistent with their definitions
- Makefile actions that do not exist on Tru64 UNIX
- Important command options and arguments that differ between platforms

To help developers understand each porting problem, the Porting Assistant graphical hyperlinked information:

- Provides porting tips that complement Compaq's extensive porting guides
- Explains the relevance of each of the analysis steps
- Explains the semantic differences in functions on different vendor's platforms
- Provides detailed help on may individual diagnostics
- Provides hyperlinked access to relevant reference pages

When making code changes, the Porting Assistant:

- Provides developers with a choice of vi, emacs, or a Motif editor
- Steps developers through the problem areas one at a time, bringing each into the editor to be examined and changed
- Provides a global search/replace facility to make changes across files
- Filters out unwanted diagnostics
- Suggests what shareable libraries and archives to use
- Reruns analyses at users' request after changes are made
- Provides support for mapping Fortran compile command lines in a makefile to the equivalent Tru64 UNIX command lines
- Provides the addition of a keyword lookup function for the DXML Library, which includes commonly used Fortran math functions
- Provides support for mapping LINPACK and EISPACK routines to LAPACK routines

Program Analysis Tools

The ATOM API can be used to create simple or sophisticated tools. ATOM uses the target application program, an instrumentation file, and an analysis file to

create a new executable, that when run collects analysis data for a wide variety of purposes.

Tools are created by writing two sets of C routines: an instrumentation file and an analysis file. In the instrumentation file, the ATOM API is used to specify where calls to analysis routines should be inserted in an application. A single call to an ATOM routine creates a new executable. The analysis routines are called, at run time, by instructions ATOM inserted in the new executable. These routines run in the background of the application and analyze it for attributes such as quality and performance. While ATOM greatly expedites tool creation, it offers vast flexibility in designing the scope of the tool. Compaq has created several tools with ATOM, including hiprof, pixie, and Third Degree.

All these tools support shared libraries and threaded programs and are licensed with the Developers' Toolkit.

The Third Degree tool profiles heap memory for programs written in C and C++. It helps developers identify three memory bugs: memory leaks, reading uninitialized memory, and accessing an invalid memory address.

Pixie is a basic block profiler that allows instruction-level execution counts to be obtained. Developers can then feed this profiling data to the compiler and the cord or om reordering tools.

Hiprof provides procedure call information on the program that is similar to but more powerful than the gprof (cc -pg) profiler. It shows the amount of CPU time utilized by each primary procedure and the subordinate procedures they call.

Graphical Program Analysis Tools

The Graphical Program Analysis Tools (GPA) help application developers examine the run-time behavior of their applications on Tru64 UNIX.

The Graphical Program Analysis Tools can locate problems in the code, explain the problems using diagnostic messages, and even suggest necessary changes through reports. For example, using these tools, developers can:

- Instrument an application, run the application, and look at the memory usage data in one step or separate steps
- Find poorly tested areas in the code
- Locate and correct performance bottlenecks
- Find and fix memory leaks and memory errors
- Find and fix problems with writing past the ends of memory blocks
- Get information about all processes, including child processes, running on local or remote systems
- Invoke utilities and commands to monitor and manage the remote system
- Record all the process information to a file for later review

The Graphical Program Analysis Tools Are:

- **Process Viewer:** Graphically displays performance information about processes and child processes running on specified (local or remote) Tru64 UNIX systems.
- **Memory Profiler:** Graphically displays how an application uses memory over the course of its execution, for example, showing how much is allocated for various purposes and how frequently the application is allocating and deallocating memory. This tool helps developers understand if an application uses memory inefficiently, for example, fragmented memory allocations.
- **Performance Profiler:** Gathers, analyzes, and displays, in graphical form, run-time statistics about an application, such as CPU usage by function or line and test coverage.
- **Heap Analyzer:** Finds and displays in graphical form memory errors and memory leaks in an application.
- **Man Page Browser:** Used for searching, navigating, and printing reference pages (manpages) in a graphical, scrollable hypertext window.

Reordering Tools

The reordering tools rearrange procedures in an executable file to facilitate better cache mapping, reducing the instruction cache miss rates. The tools are as follows:

- cord – Rearranges procedures in an executable to facilitate better cache mapping
- om - A post-link optimizer that can optimize executables and reorder them

Enterprise Toolkit

The Compaq Enterprise Toolkit – UNIX Edition is a software product that extends Microsoft's Visual C++ or Compaq's Visual FORTRAN environments to enable development of UNIX applications. Visual C++ and Visual FORTRAN work with the Microsoft Visual Studio environment, a suite of software development tools for Windows operating systems. Using Enterprise Toolkit, you can use these tools to create UNIX applications that take advantage of the performance and scalability of 64-bit Compaq Tru64 UNIX Alpha technology.

Enterprise Toolkit uses an Intel-based system running Windows NT, Windows 95 or Windows 98 as a front-end system, working with an Alpha-based system running Tru64 UNIX as a back end, to develop Tru64 applications. Enterprise Toolkit allows use of a Windows development environment to write, compile, debug and tune applications for Alpha systems running Tru64 UNIX. The Enterprise Toolkit extends the Visual Studio graphical user interface, incorporating UNIX development functions into Visual Studio style toolbars, buttons, displays, wizards and online Help. Enterprise Toolkit uses native Tru64 UNIX tools on Alpha systems, such as FORTRAN, C and C++ compilers, linkers, and debuggers to do the actual work.

The Technical Programming Extensions for Enterprise Toolkit (TPE) is an add-in for Enterprise Toolkit that adds a number of useful features for FORTRAN programmers, including:

- Porting Assistant, to aid in moving programs from Cray, SGI, Sun and HP platforms to Tru64 UNIX.
- Support for the Compaq KAP FORTRAN optimizer.
- Toolbar buttons to launch the TotalView debugger and Pallas Vampir tools.
- Support for linking to math and parallel programming libraries such as CXML and KAP.

The Porting Assistant analyzes potential portability problem areas, including:

- Analyzes COMMON blocks and their use to flag potential data size and data alignment problems
- Locates and flags parallel directives (Cray, SGI and X3H5) and suggests equivalent OpenMP directives.
- Validates existing makefiles, checking for the existence of the compilers and tools used and making sure the options and arguments for each action are valid on Tru64 UNIX.
- Checks the consistency of calls and function definitions.
- Verifies that all include files exist within the path and reports any duplicate files.
- Flags conditional code sections that may contain platform dependent code.

Compaq AlphaServer SC systems *deliver*

- **Power.** Raw, brute-force, number-crunching power – hundreds of GigaFLOPS, hundreds of Gigabytes of memory, and terabytes of storage. The fastest processors, full 64-bit systems (64-bit processors, 64-bit Very Large Memory, 64-bit I/O, and 64-bit Compaq Tru64 UNIX), SMP and clustering technologies, fast I/O, and high system bandwidth.
- **Scalability.** SMP and high-performance interconnects provide efficient scaling for application performance, while compatibility with industry standards makes it easy to integrate into existing environments.
- **Compatibility.** For the ultimate in investment protection for your software, all Compaq AlphaServer systems – from the smallest workstation to the mightiest supercomputer – are completely binary-compatible.
- **Investment protection.** Every AlphaServer system offers virtually unlimited growth potential for years to come – a clear, cost-effective upgrade path, with migration-free hardware, software, and option upgrades.

What's more, you can choose the architecture, performance, and operating system that's best for you:

- The AlphaServer SC Series of supercomputers boast up to 512 Alpha processors and 4096 GB of memory managed as a single 665-GigaFLOPS system.
- AlphaServer GS Series includes five different models with eight to 32 Alpha processors and up to 256 GB of memory in an SMP environment.
- AlphaServer ES Series features up to four Alpha processors, 32 GB of memory, and crossbar technology to enable memory bandwidth of 5.3 GB/s.
- AlphaServer DS Series employs up to two Alpha processors, up to 8 GB of memory, and up to 5.3 GB/s memory bandwidth.
- Parallel servers, based on AlphaServer ES Series systems, deliver supercomputer power in a stable, mature architecture at a fraction of the cost of traditional supercomputers.
- High-performance clusters can be built from Compaq's range of AlphaServer systems, interconnects, and software –- or purchased as a complete, packaged solution.

To find out more about the benefits of using Compaq AlphaServer systems in high-performance computing environments like yours, call **800-457-8211,** visit **www.compaq.com/hpc**, or return the attached card.

Printed in U.S.A. Document #135D-0900A-USEN X.XM

COMPAQ