AN SSC PUBLICATION

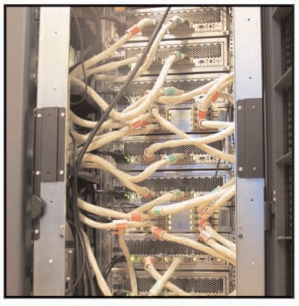
Link LANs securely with FreeS/WAN

LINUX JOURNAL

The Monthly Magazine of the Linux Community • **FEBRUARY 2003**

Behind the ALTIX 3000

SGI's new 64-Processor, 64-Bit NUMA system



USB Serial Drivers for 2.6

Replacing Microsoft Exchange

Choosing web development tools

Libraries go GPL with Koha

Large-scale mail with Postfix and LDAP

USA $5.00 CAN $6.50
www.linuxjournal.com
02
0 71486 03102 4

Scaling Linux to New Heights: the SGI Altix 3000 System

With 64 processors and 512GB of memory, SGI now holds a claim on the title of world's most powerful Linux system. **BY STEVE NEUNER**

SGI recently debuted its new 64-bit, 64-processor, Linux system based on the Intel Itanium 2 processor—a significant announcement for the company and for Linux. This system marks the opening of a new frontier as scientists working on complex and demanding high-performance computing (HPC) problems can now use and deploy Linux in ways never before possible. HPC environments continually push the limits of the operating system by requiring larger numbers of CPUs, higher I/O bandwidth and faster and more efficient parallel programming support.

Early on in the system's development, SGI made the decision to use Linux exclusively as the operating system for this new platform. It proved to be a solid and very capable operating system for the technical compute environments that SGI targets. With the combination of SGI NUMAflex global shared-memory architecture, Intel Itanium 2 processors and Linux, we were breaking records long before the system was introduced.

The new system, called the SGI Altix 3000, has up to 64 processors and 512GB of memory. A future version will offer up to 512 processors and 4TB. In this article, we explore the hardware design behind the new SGI system, describe the software development involved to bring this new system to market and show how Linux can readily scale and be deployed in the most demanding HPC environments.

Hardware and System Architecture Background

The SGI Altix 3000 system uses Intel Itanium 2 processors and is based on the SGI NUMAflex global shared-memory architecture, which is the company's implementation of a non-uniform memory access (NUMA) architecture. NUMAflex was introduced in 1996 and has since been used in the company's renowned SGI Origin family of servers and supercomputers based on the MIPS processor and the IRIX 64-bit operating system. The NUMAflex design enables the CPU, memory, I/O, interconnect, graphics and storage to be packaged into modular components, or bricks. These bricks can then be combined and configured with tremendous flexibility to match a customer's resource and workload requirements better. Leveraging this third-generation design, SGI was able to build the SGI Altix 3000 system using the same bricks for I/O (IX- and PX-bricks), storage (D-bricks) and interconnect (router bricks/R-bricks). The primary difference in this new system is the CPU brick (C-brick), which contains the Itanium 2 processors. Figure 1 shows the types of bricks used on the SGI Altix 3000 system. Figure 2 depicts how these bricks can be combined into two racks to make a single-system-image 64-processor system.

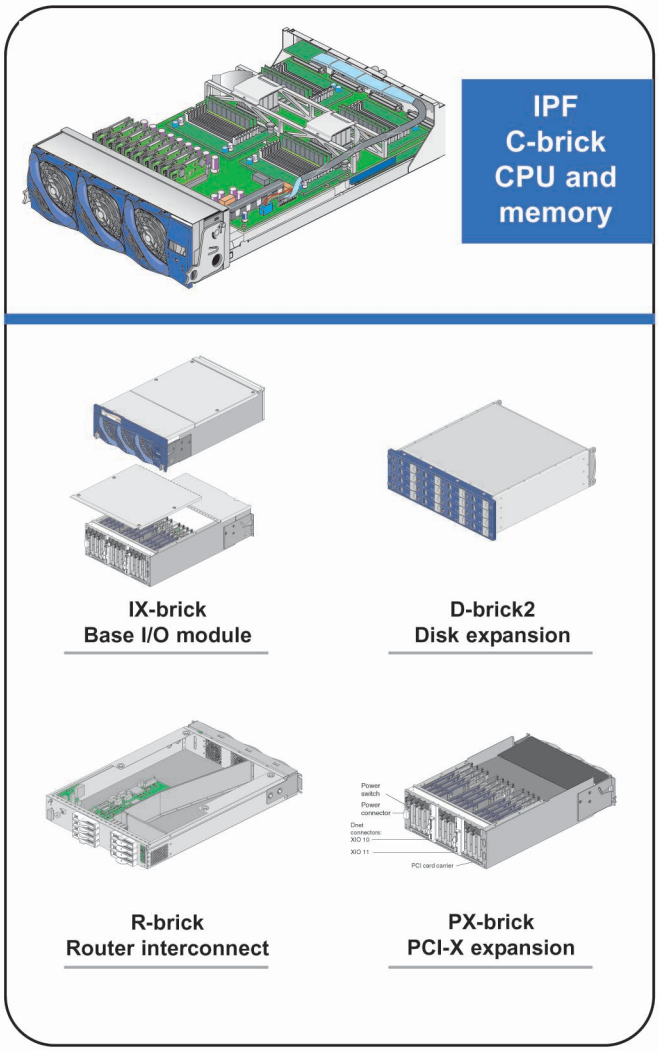


Figure 1. NUMAflex Brick Types

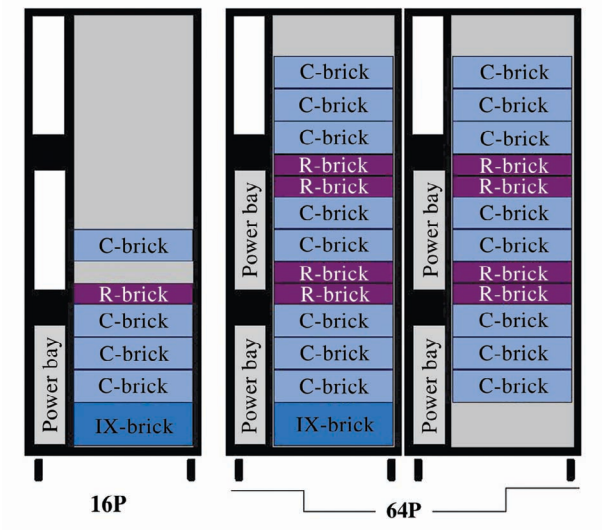


Figure 2. Two Possible NUMAflex Configurations

Preparing Linux for a New Hardware Platform

On a well-designed and balanced hardware architecture such as NUMAflex, it is the operating system's job to ensure that users and applications can fully exploit the hardware without being hindered due to inefficient resource management or bottlenecks. Achieving balanced hardware resource management on a large NUMA system requires starting kernel development long before the first Itanium 2 processors and hardware prototype systems arrive. In this case, we also used the first-generation Itanium processors for making the CPU scaling, I/O performance and other changes to Linux necessary for demanding HPC environments.

The first step in preparing the software before the prototype hardware arrives is identifying, as best you can, the necessary low-level hardware register and machine programming changes the kernel will need for system initialization and runtime. System manufacturers developing custom ASICs for highly advanced systems typically use simulation software and tools to test their hardware design. Before hardware was available, we developed and used simulators extensively for both the system firmware and kernel development to get the system-level software ready.

When the original prototype hardware based on first-generation Itanium processors arrived, it was time for power-on. One of the key milestones was powering the system on for the first time and taking a processor out of reset, then fetching and executing the first instructions from PROM.

Figure 3. SGI engineers celebrate power-on success.

After power-on, the fun really began with long hours and weekends in the hardware "bring-up" lab. This is where hardware, diagnostic and platform-software engineers worked together closely to debug the system and get the processor through a series of important milestones: PROM to boot prompt, Linux kernel through initialization, reading and mounting root, reaching single-user mode and then going into multi-user mode and then connecting to the network. After that, we did the same thing all over again with multiple processors and multiple nodes—typically pursued in parallel—with

SGI ALTIX 3000 C-BRICK BLOCK DIAGRAM AND SPECIFICATIONS

Each C-brick for the SGI Altix 3000 system contains up to four Itanium 2 processors, up to 32GB of commodity memory and two "SHUB" ASICs. Each SHUB interfaces to two Itanium 2 processors, along with memory, I/O devices and other SHUBs. Processors on one node communicate with memory and processors on remote nodes via the SHUB and NUMAlink, the NUMAflex interconnect technology. The SHUB provides a single coherence domain using a mix of snoopy and directory-based coherence/communication protocols. One or two interconnect planes between nodes can be configured for higher performance and improved fault tolerance. Figure A shows a block diagram of a C-brick with Itanium 2 processors.

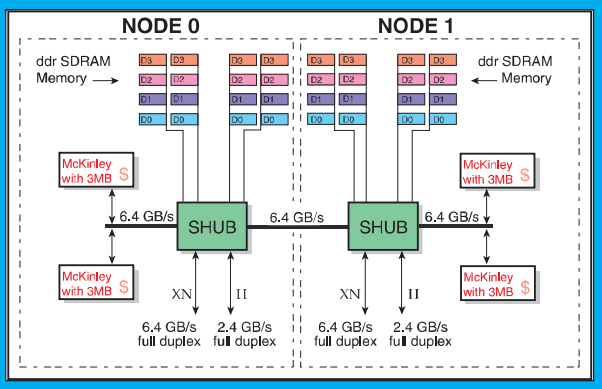


Figure A. Each C-brick contains two nodes connected by NUMAlink.

Each C-brick was designed to contain two nodes and two SHUBs, so that each SHUB will have no more than two processors on its front-side bus (FSB). For each node, the processor's FSB bandwidth is 6.4GB per second; memory bandwidth is 10.2GB per second; aggregate interconnect bandwidth is 6.4GB per second; and I/O aggregate bandwidth is 4.8GB per second. By limiting the number of processors on the FSB to two, combined with the bandwidth capacities mentioned, the SGI Altix 3000 system design ensures plenty of available bandwidth not only for each processor but also throughout the system to avoid congestion or bottlenecks.

several other bring-up teams at other stations that trail closely behind the lead team's progress.

Figure 4. During bring-up, a hardware engineer, a PROM engineer and an OS engineer discuss a bug.

Once we had Linux running on the prototype systems with first-generation Itanium processors, software engineers could proceed with ensuring that Linux ran and, in particular, scaled well on large NUMA systems. We built and used numerous in-house, first-generation Itanium-based systems to help ensure that Linux performed well on large systems. By early 2001, we had succeeded in running a 32-processor Itanium-based system—the first of its kind.

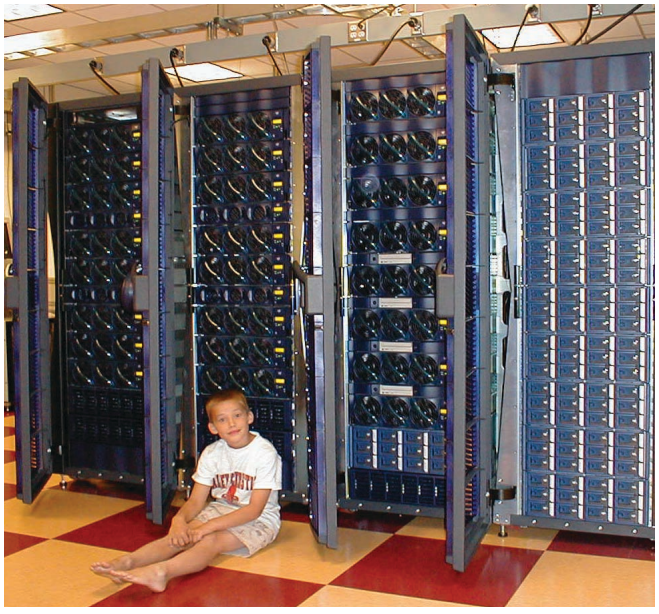


Figure 5. The author's son in front of an early 32-processor Itanium-based system, Summer 2001.

These first-generation Itanium-based systems were key in having Linux ready for demanding HPC requirements. Well before the first Itanium 2 processors were available from Intel, the bulk of the scaling, I/O performance and other changes for Linux could be developed and tested.

As one group of SGI software engineers was busy working on performance, scaling and other issues, using prototypes with first-generation Itanium processors, another team of hardware and platform-software engineers was getting the next-generation SGI C-brick with Itanium 2 processors ready for power-on to repeat the bring-up process all over again.

Figure 6. First power-on of the Itanium 2-based C-brick.

By mid-2002, the bring-up team had made excellent progress, from power-on of a single processor to running a 64-processor system. The 64-processor system with Itanium 2 processors again marked the first of its kind. All this, of course, was with Linux running in a single-system image.

Throughout this whole process, we passed any changes in Linux or bugs found back to the kernel developers for inclusion in a future release of Linux.

A Closer Look at Linux on Big Iron

Other Linux developers often ask, "What kind of changes did you have to make to get Linux to run on that size system?" or "Isn't Linux CPU scaling limited to eight or so processors?" Answering these questions involves examining further what SGI is using as its software base, the excellent changes made by the community and the other HPC-related enhancements and tools provided by SGI to help make Linux scale far beyond the perceived limit of eight processors.

On the SGI Altix 3000 system, the system software consists of a standard Linux distribution for Itanium processors and SGI ProPack, an overlay product that provides additional features for Linux. SGI ProPack includes a newer 2.4-based Linux kernel, HPC libraries highly tuned to exploit SGI's hardware, NUMA tools and drivers.

The 2.4-based Linux kernel used on the SGI Altix 3000 system consists of the standard 2.4.19 kernel for Itanium processors (kernel.org), plus other improvements. These improvements fall into one of three categories: general bug fixes and platform support, improvements from other work occurring

within the Linux community and SGI changes.

The first category of kernel changes is simply ongoing fixes to bugs found during testing and the continued improvements for the underlying platform and NUMA support. For these changes, SGI works with the kernel team's designated maintainer to get these changes incorporated back into the mainline kernel.

The second category of kernel improvements consists of the excellent work and performance patches developed by others within the community that have not been accepted officially yet or were deferred until the 2.5 development stream. These improvements can be found on the following VA Software SourceForge sites: "Linux on Large Systems Foundry" (large.foundries.sourceforge.net) and the "Linux Scalability Effort Project" (sourceforge.net/projects/lse). We used the following patches from these projects: CPU scheduler, Big Kernel Lock usage reduction improvements, dcache_lock-usage reduction improvements based on the Read-Copy-Update spinlock paradigm and xtime_lock (gettimeofday) usage reduction improvements based on the FRlock locking paradigm.

We also configured and used the Linux device filesystem (devfs, www.atnf.csiro.au/people/rgooch/linux/docs/devfs.html) on our systems to handle large numbers of disks and I/O buses. Devfs ensures that device path names persist across reboots after other disks or controllers are added or removed. The last thing a system administrator of a large system wants is to have a controller go bad and have some 50 or more disks suddenly renumbered and renamed. We have found devfs to be reliable and stable in high-stress system environments with configurations consisting of up to 64 processors with dozens of Fibre Channel loops with hundreds of disks attached. Devfs is an optional part of the 2.4 Linux kernel, so a separate kernel patch was not needed.

The third category of kernel change consists of improvements by SGI that are still in the process of getting submitted into mainline Linux, were accepted after 2.4 or will probably remain separate due to the specialized use or nature of the patch. These open-source improvements can be found at the "Open Source at SGI" web site (oss.sgi.com). The improvements we made included: XFS filesystem software, Process AGGregates (PAGG), CpuMemSets (CMS), kernel debugger (kdb) and a Linux kernel crash dump (lkcd).

In addition, SGI included its SCSI subsystem and drivers ported from IRIX. Early tests of the Linux 2.4 SCSI I/O subsystem showed that our customers' demanding storage needs could not be met without a major overhaul in this area. While mainstream kernel developers are working on this for a future release, SGI needed an immediate fix for its 2.4-based kernel, so the SGI XSCSI infrastructure and drivers from IRIX were used as an interim solution.

Figures 7–9 illustrate some of the early performance improvements that were achieved with Linux on the SGI Altix 3000 system using the previously described changes. Figure 7 compares XFS to other Linux filesystems. (Note, for a more detailed study on Linux filesystem performance, see "Filesystem Performance and Scalability in Linux 2.4.17", 2002 USENIX Annual Technical Conference, which is also available at oss.sgi.com). Figure 8 compares XSCSI to SCSI in Linux 2.4, and Figure 9 shows CPU scalability using AIM7.

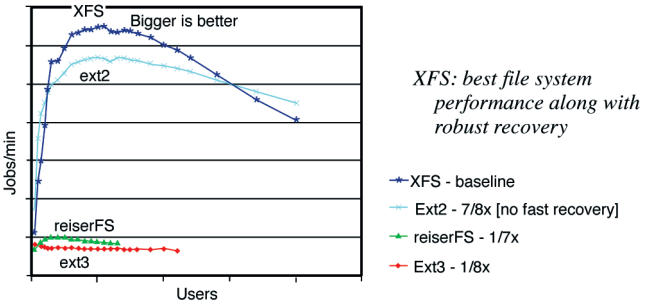


Figure 7. Filesystem performance comparison: AIM7 multi-user kernel workload, 2.4.17 kernel; 28 P Itanium prototype, 14GB, 120 disks; work-in-progress, interim example; varied filesystems only, but includes SGI enhancements and SGI-tuned kernel.

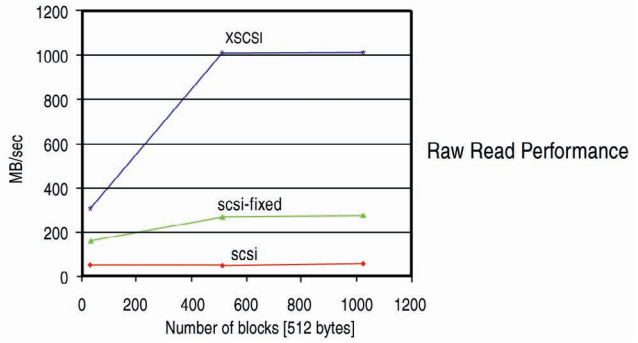


Figure 8. Linux XSCSI performance example: work-in-progress, interim example using 2.4.16 kernel; 120 processes reading from 120 disks (through driver only).

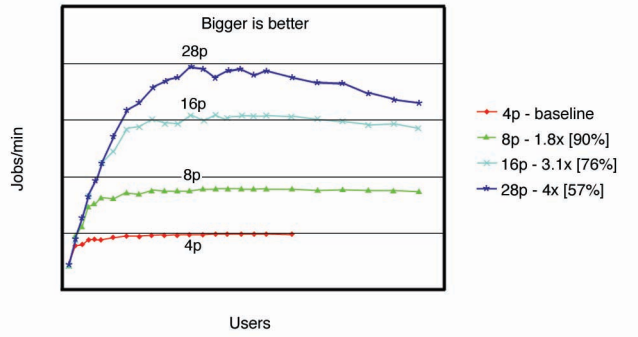


Figure 9. CPU scaling example with AIM7: AIM7 multi-user kernel workload, 2.4.16 kernel; work-in-progress, interim example; SGI enhancements and SGI-tuned kernel.

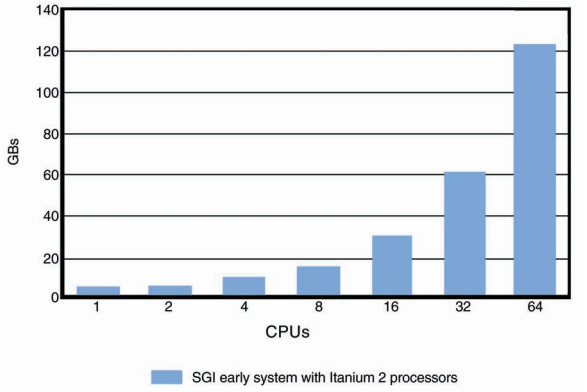


Figure 10. Near-linear STREAM Triad scalability up to 64 processors.

ALREADY SOLVING REAL-WORLD PROBLEMS

The following examples show the performance of three HPC scientific applications on an SGI Altix 3000 system running Linux. These examples are FASTA for bioinformatics, Gaussian for computational chemistry and STAR-CD for computational fluid dynamics. All testing was conducted by SGI.

FASTA Bioinformatics Example

Although biochemistry and computational biology have existed since the mid-1980s, bioinformatics is a fairly new scientific realm, made possible by the convergence of data-intensive life sciences research, laboratory-automation technologies and the development of computer databases and algorithms that can rapidly organize, process and distribute enormous quantities of data. Bioinformatics is used to help bring new drugs to market faster, prevent genetic disease, develop disease- or drought-resistant crops, extend the shelf life of food, find alternatives to petroleum and create foods of the future that will help manage cholesterol or prevent cancer.

Sequence database searching is among the most critical areas of bioinformatics due to the rapidly growing volume of publicly available biological data. Classical algorithms like Smith-Waterman (T. F. Smith and M. S. Waterman, 1981, *Journal of Molecular Biology* 147:195–197) provide the most rigorous way to search biological databases for local sequence similarities. However, Smith-Waterman is computationally demanding, so parallelization is extremely important for making this algorithm efficient. One parallel Smith-Waterman implementation available is the FASTA bioinformatics package (W. R. Pearson, 1991, *Genomics* 11:635–650).

The Smith-Waterman algorithm from FASTA version 3.4 (ssearch34_t), which is available from the University of Virginia (alpha10.bioch.virginia.edu/fasta), was used to measure parallel performance on a 64-processor SGI Altix 3000 system. Smith-Waterman is parallelized with Pthreads, and the SGI ChemBio Applications team also further tuned the core algorithm to more fully exploit the capabilities of the SGI Altix 3000 system. As a result, the Smith-Waterman algorithm shows close to ideal scaling (dotted line represents ideal scaling) with speedups of approximately 59× when running on 64 processors.

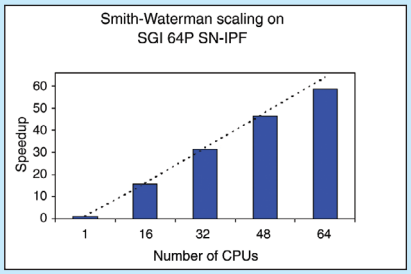


Figure I. FASTA scales nearly linearly.

Gaussian Computational Chemistry Example

National research laboratories, universities, pharmaceutical and biotechnology companies and chemical companies use computational chemistry applications like Gaussian 98 from Gaussian, Inc. (www.gaussian.com) for research in molecular energies, properties and reactions by modeling very large molecular systems using electronic structure methods.

In Figure II, the parallel scaling of Gaussian 98 on a widely used test from Gaussian's QA-suite is shown using an early prototype SGI Altix 3000 system and an earlier version of the Intel compilers targeting Itanium processors rather than Itanium 2 processors. The case corresponds to a Force calculation of the Valynomycin molecule (C54H90N6O18), using Density Functional Theory. The graph shows the parallel speedups for this test case, with the elapsed times in seconds. Even with the early tools and system versions used, the time to perform this calculation is reduced from about 4.5 hours to only 25 minutes when using 20 processors on an SGI Altix 3000 system.

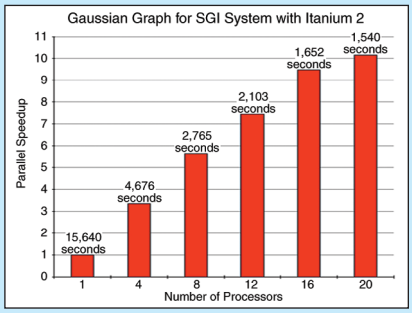


Figure II. Gaussian 98 Results

STAR-CD Computational Fluid Dynamics Example

Computational fluid dynamics (CFD) is used in many traditional industries such as automotive, aerospace and power generation. STAR-CD from Computational Dynamics Limited (www.cd-adapco.com) is one of the CFD technology leaders for fluid flow analysis. It can assist a CFD user from initial fundamental design, through parametric studies to optimization, using its advanced physical models and its ability to handle complex geometry using unstructured meshes. STAR-CD runs on all leading UNIX and NT platforms. Its parallel form, STAR-HPC, runs on shared-memory multiprocessor servers, massively parallel systems and clusters of workstations. STAR-HPC uses the MPI library to achieve a high degree of scalability. Using a prerelease version of STAR-CD on an SGI Altix 3000 system with the "Model A Class Dataset" for modeling turbulence flow on an automobile, Linux again demonstrates excellent processor scaling up to 64 processors based on early performance runs. According to published results at www.cd-adapco.com/support/bench/aclass.htm as of early November 2002, Linux even scaled better than two proprietary operating systems: Hewlett-Packard's HP-UX and IBM AIX.

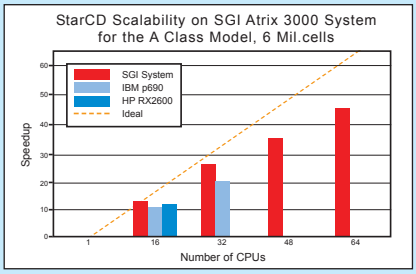


Figure III. STAR-CD results comparing Linux, HP-UX and AIX.

While SGI is focused more toward high-performance and technical computing environments—where the majority of CPU cycles is typically spent in user-level code and applications instead of in the kernel—the AIM7 benchmark does show that Linux can still scale well with other types of workloads common in enterprise environments. For HPC application performance and scaling examples for Linux, see the Sidebar "Already Solving Real-World Problems".

Figure 10 shows the scaling results achieved on an early SGI 64-processor prototype system with Itanium 2 processors running the STREAM Triad benchmark, which tests memory bandwidth. With this benchmark, SGI demonstrated near-linear scalability from two to 64 processors and achieved over 120GB per second. This result marks a significant milestone for the industry by setting a new world record among a microprocessor-based system, which was achieved running Linux within a single-system image! This impressive result also demonstrates that Linux can indeed scale well beyond the perceived limitation of eight processors. For more information on STREAM Triad, see www.cs.virginia.edu/stream.

When you look at the list of kernel additions included in SGI ProPack the list is actually surprisingly small, which speaks highly of Linux's robust original design. What is even more impressive is that many of these and other changes are already in the 2.5 development kernel. At this pace, Linux is quickly evolving as a serious HPC operating system.

Other Enhancements to Linux for HPC

SGI ProPack also includes several tools and libraries to help improve performance on large NUMA systems for solving a complex problem with an application that needs large numbers of CPUs and memory, or when multiple applications are running simultaneously within the same large system. On Linux, SGI provides the commands **cpuset** and **dplace**, which give predictable and improved CPU and memory placement control for HPC applications. These tools help unrelated jobs carve out and use the resources they each need without getting into each other's way or help prevent a smaller job from inadvertently thrashing across a larger pool of resources than it can effectively use. Therefore system resources are used efficiently and deliver results in a consistent time period—two characteristics critical to HPC environments.

Also, the SGI Message Passing Toolkit (MPT) in SGI ProPack provides industry-standard message passing libraries optimized for SGI computers. MPT contains MPI and SHMEM APIs, which transparently utilize and exploit the low-level capabilities within the SGI hardware, such as its block transfer engine (BTE) for fast memory-to-memory transfers and the hardware memory controller's fetch operation (fetchop) support. Fetchop support enables direct communication and synchronization between multiple MPI processes while eliminating the overhead associated with system calls to the operating system.

The SGI ProPack NUMA tools, HPC libraries and additional software support layered on top of a standard Linux distribution provide a powerful HPC software environment for big compute and data-intensive workloads. Much like a custom ASIC on hardware providing the "glue logic" to leverage and use commodity processors, memory and I/O parts, SGI ProPack software provides the "glue logic" to leverage the Linux operating system as a commodity building block for large HPC environments.

Conclusion

No one believed Linux could scale so well, so soon. By combining Linux with SGI NUMAflex system architecture and Itanium 2 processors, SGI has built the world's most powerful Linux system. Bringing the SGI Altix 3000 system to market involved a tremendous amount of work, and we consider it to be only the beginning. The aggressive standards-based strategy that SGI has for using Linux on Itanium 2-based systems is raising the bar on what Linux can do while providing customers an exciting, no-compromises alternative for large HPC servers and supercomputers. SGI engineers—and the entire company for that matter—are fully committed to building on Linux capabilities and pushing the envelope even further to bring more exciting breakthroughs and opportunities for the Linux and HPC communities.

Steve Neuner has been working in UNIX kernel development for the past 19 years at major computer manufacturers including MAI Basic Four, Sequent Computer Systems, Digital Equipment Corporation and SGI. Now with SGI, Steve is the Linux engineering director and has been working on Linux and Itanium-based systems since joining SGI four years ago.