WHITE PAPER

Technical Advances in the SGI® UV™ Architecture

June, 2012

TABLE OF CONTENTS

1.0 Introduction 3

2.0 SGI UV 2000 Architecture 4
2.1 SGI UV 2000 Compute Blade 4
2.1.1 UV ASIC Functionality 5
2.1.1.1. Global Register Unit (GRU) 6
2.1.1.2. Active Memory Unit (AMU) 7
2.2 SGI UV 2000 Interconnect Topology 8

3.0 Integrated System Functionality 9
3.1 The MPI Offload Engine (MOE) 9
3.1.1 Reduced CPU Overhead and Latency for MPI_Send 9
3.1.2 Fast Barriers and Reductions in Hardware 9
3.1.3 Accessing the MOE 10
3.2 Large-scale Memory and Data Intensive Computation 10
3.3 Massive I/O Capabilities 10
3.4 Large-scale Compute Enhancements 11
3.5 Enhanced Reliability 11

4.0 Conclusion 12

1.0 Introduction

Coherent shared memory (CSM) architectures—in which all processors can access all memory directly—offer significant advantages for a broad class of applications including scientific and engineering applications, real-time complex event processing, very large databases (VLDBs) and advanced business applications. SGI is the industry leader in the development of CSM platforms that deliver exceptional processing, memory and I/O capabilities for high performance computing (HPC) and data-intensive tasks.

The SGI UV 2000 is the sixth generation of the company's scalable global shared memory architecture. SGI started shipping CSM systems with the 64-core SGI Origin® systems in 1996. The new SGI GSM platform is the UV 2000 system, which scales to 4,096 threads and 64 terabytes (TB) of memory. These systems are built using the SGI NUMAlink® interconnect that provides the high-bandwidth, low-latency, coherence-optimized functionality required by GSM systems. The NUMAlink interconnect fabric can also be used for efficient communication between OS instances, enabling scaling up to many thousands of CPU cores for shared memory applications as well as MPI applications and those developed using partitioned, global address space compilers like OpenMP or Unified Parallel C.

Following on the prior generation SGI UV 1000 family, SGI UV 2000 adds new architectural capabilities which enhance application scaling and performance, independent of the programming model or models employed. These systems also leverage the proven economy of the Intel® Xeon® processor E5-4600 product family and standard Linux® while maintaining compatibility with previously written applications.

The key benefits of the SGI UV 2000 architecture include:

- **Massive In-Core Computation.** The SGI UV 2000 allows much larger and more detailed models and simulations of physical systems, or any large data set, to be entirely memory resident.
- **Massively Memory Mapped I/O.** For applications that are bound by random I/O accesses on large data sets, the SGI UV 2000 with the Intel® Xeon® processor E5-4600 product family offers up to a 1,000x performance increase by enabling entire datasets to be brought into main memory.
- **Highly Efficient Application Scaling and Message Passing.** SGI UV 2000 utilizes an array of advanced hardware and software features to offload thread synchronization, data sharing and message passing overhead from CPUs — accelerating critical tasks by up to 100x. These features benefit all programming styles, and for Message Passing Interface (MPI) applications they are collectively referred to as the "MPI Offload Engine," or MOE.
- **Greatly Simplified Application Load Balancing.** In cluster computing environments, each node completes its own threads and then waits until all other nodes complete their assigned tasks. The global shared memory available with the SGI UV 2000 with the Intel® Xeon® processor E5-4600 product family allows processors that finish early to also work on other threads, since each processor has access to all data and synchronization points through globally shared memory.
- **Smooth Scaling of Application Size and Complexity.** In most cluster environments, applications run on a fixed number of nodes, each with a fixed amount of CPU cores and memory. Applications run into a "wall" when they exceed the fixed amount of memory per core or node in the cluster. Conversely, applications running on an SGI UV 2000 do not run into a memory induced wall. Instead, they scale smoothly by drawing on additional memory distributed throughout the system.
- **Petascale System and Application Scalability.** In addition to global shared memory support where all resources are shared by a single copy of the operating system, SGI UV 2000 provides an even larger Globally Addressable Memory (GAM) which enables systems to be built that extend beyond the shared memory reach of any given processor or OS implementation. Advanced coherence functionality and atomic operations that increase efficiency of GSM environments also allow single MPI and partitioned global address space (PGAS) applications to scale to many thousands of cores and 8 petabytes (PB) of memory using efficient GAM capabilities.

- **Efficient Application Development.** Developing threaded applications, MPI applications or PGAS applications on the SGI UV 2000 enables rapid development and large problem solution in the early stages of the parallelization process. Using these systems, applications that will ultimately run on thousands of CPU cores and access tens of terabytes of memory can be solved when only moderate levels of parallelism have been developed — proving algorithms early in the development cycle and shortening the time it takes to generate first results.
- **Lower Cost Through Efficient Memory Utilization.** In cluster systems, each node has a copy of the operating system, I/O buffer caches, and additional space for message passing overhead and duplicated data structures. Each node is also typically configured to have a large amount of memory per core, just in case a large memory application is assigned to that node. These two factors combined, lead to large amounts of excess memory being purchased for cluster systems, greatly inflating their costs. In contrast, the global shared memory architecture in the SGI UV 2000 only requires a single OS and a single buffer cache which reduces that amount of memory overhead. And since every application can access the entire memory, threads that need more memory than is available on their local nodes directly utilize memory resident on other nodes, greatly reducing the total amount of memory needed.
- **Simplified Administration.** The SGI UV 2000 enables large collections of compute, memory and storage resources to be managed together, significantly reducing the complexity and cost of system administration.

2.0 SGI UV 2000 Architecture

The SGI UV 2000 is a scalable global shared memory system based on the SGI NUMAlink interconnect. Physically, these systems feature a compact blade design, an architecture that supports a large number of processors in a global shared memory configuration running a single copy of standard Linux and the ability to share memory across multiple system images across SGI NUMAlink connections.

The SGI UV 2000 is designed to extend the industry-leading scalability of SGI shared memory systems in all dimensions — processors, memory, interconnect and I/O. The current release supports up to 2,048 cores (4,096 threads) and 64TB of memory—4x the memory of the prior generation UV — per single system image (SSI). Much larger multi-partition systems are supported with shared global memory address out to multiple petabytes, taking the company's leadership in shared memory systems to new heights. Configuration options enable the creation of systems that are optimized for compute capability (maximum core count), maximum total memory, system bisection bandwidth or maximum I/O.

SGI UV 2000 distinguishes itself from standard cluster approaches by tightly integrating with the Intel® Quick Path Interconnect (QPI). This integration provides both global shared memory and efficient access to that memory by working at the cache line level over the entire system consisting of tens to tens of thousands of blades. This cache-line-oriented approach contrasts sharply with cluster based approaches that are optimized for transferring large amounts of data over an InfiniBand interconnect connected to an I/O channel.

The SGI UV 2000 architecture also integrates multiple computing paradigms into a single environment based on industry standard Intel® Xeon® processors. The design increases the efficiency of Intel® Xeon® scalar CPUs by providing vector memory operations, a rich set of atomic memory operations, and tight integration of application-specific hardware such as GPUs, digital signal processors and programmable hardware accelerators.

2.1 SGI UV 2000 Compute Blade

A standard compute blade of SGI UV 2000 contains two processor sockets, each capable of supporting either 4-, 6- or 8-core Intel® Xeon® processor E5-4600 product family. As shown in Figure 1, each socket has direct connections to memory, plus one Intel® QuickPath Interconnect (QPI) connection. These connections

allow processors to communicate with each other, with the SGI developed UV ASIC, and with optional external I/O connections through. By altering the number and type of processors, memory and I/O expansion options, each blade can have anywhere from 8 to 16 cores with 32 to 512 of memory and I/O, providing complete configuration flexibility.

Figure 1. Block-level diagram of SGI UV compute blade.

The UV node blade carries a custom ASIC developed by SGI (referred to as the NUMAlink 6 hub) that is the cornerstone of the UV platform enables the creation of scalable global shared memory systems. The UV implements the NUMAlink 6 protocol, memory operations and associated atomic operations that provide much of the system capabilities including:

- Cache-coherent global shared memory that runs industry standard Linux OS and unmodified industry standard applications
- Offloading time-sensitive and data-intensive operations from processors to increase processing efficiency and scaling
- Scalable, reliable, fair interconnection with other blades via NUMAlink 6
- Petascale global addressable memory with low-latency synchronization
- MPI Offload Engine (MOE) that integrates a number of advanced features to reduce message passing overhead on processors and increase application and system scalability

2.1.1 UV ASIC Functionality

The UV ASIC was developed at SGI, and links the cache-coherent Intel® QPI interconnect found on Intel® Xeon® processor E5-4600 product family with the larger cache-coherent NUMAlink environment that extends across the full SGI UV 2000 system. However, the UV ASIC does much more than extend cache coherency to more processor cores; it provides other functionality that enables efficient system operation for petascale systems.

The UV ASIC has four major portions plus additional functionality to manage directories and snoop acceleration. The four major units are:

- The Global Register Unit (GRU) that extends cache coherency from the blade to the entire NUMAflex environment and provides other memory related functionality
- The Active Memory Unit that supports atomic memory operations which accelerate key synchronization activities

- The Processor Interconnect which implements two Intel® QPI interfaces that connect to Intel® Xeon® E5-4600 product family with an aggregate bandwidth of approximately 32GB/s; to the Intel® Xeon® processors, the Processor Interconnect makes the UV ASIC look like another memory controller on the QPI, but in this case one that is managing the rest of the physical and virtual memory on the system
- The NUMAlink Interconnect which implements four external NUMAlink 6 ports with an aggregate bandwidth of approximately 40GB/s

2.1.1.1 Global Register Unit (GRU)

The UV ASIC GRU supports a number of key features that enable the SGI UV 2000 to scale to more than 256,000 CPU cores and eight petabytes of memory. These features include:

- **Extended Addressing.** Memory management in the UV 2000 architecture uses a two-tiered approach. Within a compute blade, memory management is handled by the integrated memory controller on the Intel® Xeon® chip, while the GRU operations of the UV ASIC provide cache coherent memory access between blades and also extend the available address range. Intel® Xeon® processors used in this system have a 64 terabyte physical address space limit. The NUMAlink 6 ASIC not only enables single systems to support this entire physical addressing limit, but it extends the physical address space of standard Intel Xeon CPU to 53 bits and the virtual space to 60 bits for up to 8 petabytes of global memory address space across multiple partitions or nodes in a NUMAlink 6 system. NUMAlink 6 memory management functions do not interfere with fast memory access within a compute blade.
- **External TLB with Large Page Support.** A translation lookaside buffer (TLB) improves the speed of virtual to physical address translation. Intel® Xeon® CPUs include TLBs that translate all virtual references into physical addresses for memory attached to a processor. The external TLB in the UV ASIC provides the virtual to physical address translation for memory that is physically resident across the NUMAlink interconnect. To the processors on the SGI UV 2000 blade, the UV ASIC looks like another memory controller that happens to map an enormous amount of memory. The TLB on the UV ASIC also provides TLB shoot-down capabilities to increase the speed with which a large memory job, such as very large databases, can be started up or shut down.
- **Page Initialization.** With petascale systems, memory initialization can be a major performance bottleneck. The UV ASIC and associated NUMAlink 6 enhancements allow pages of memory to be initialized in a distributed manner with little or no involvement from system CPUs. By off-loading memory initialization from CPUs onto the distributed network of UV ASICs, initialization times that would have taken minutes can be completed in seconds.
- **BCOPY Operations.** Block copy (BCOPY) operations provide asynchronous mechanisms to copy data from one memory location to another. This is commonly found when replicating read-only data structures within threaded applications to avoid contention or in message passing environments to send an actual message from one blade to another. SGI shared memory systems already have the lowest message passing latency available partially because BCOPY is used to move aligned data structures that represent the majority of bytes transferred. The BCOPY functionality in the UV ASIC extends this functionality by also offloading the copying of unaligned portions of data transfers from CPUs and making BCOPY accessible from user space, reducing message passing overhead.

Step	Life of an MPI Send Message in SGI Altix 4700 Systems	Step	Life of a Message with SGI UV Systems
1	Send Fetchop fetch-and-increment request to remote node	1	Place Payload in GRI register and issue message send instruction
2	Obtain response; ID queue slot location		
3	Issue queue slot store to the remote node; this sends read-before-write request remote node		
4	Read/hold remote cache line in local node; insert data in cached line		
5	Upon queue cache-line polling, write data back to remote node		

Figure 2: Acceleration of MPI send on the SGI UV 2000 compared to SGI Altix 4700 systems. BCOPY is used to transmit the actual data from one blade to another without CPU involvement.

- **Scatter/Gather Operations.** Vector memory operations such as fixed stride and list-driven scatter/gather memory transfer operations are processed directly by the GRU in the UV ASIC. These operations allow random addresses to be accessed without the need to carry whole cache lines around the network, improving effective bandwidth and reducing latency. Vector memory operations assemble a vector of related, but not contiguous, data elements into contiguous locations within the global shared memory address space, or distribute values from contiguous locations to specified memory locations anywhere within the shared address space. Vector scatter/gather operations have a long history of increasing achievable performance by optimizing memory transfer patterns, and scatter/gather operations implemented in the UV ASIC make these optimizations available to the Intel® Xeon® processors, creating cache-friendly data structures that can be processed with little or no CPU stalls.
- **Update Cache for AMOs.** The update cache allows for globally accessed variables, like barriers and contended locks, to spin locally on a copy of an SGI-managed AMO variable, reducing hot spots in the home memory. Contended locks and reduction variables can see up to a 100x acceleration in access times, which becomes critical for very large problems spanning thousands or tens of thousands of processor cores.

2.1.1.2 Active Memory Unit (AMU)

Two types of atomic memory operations (AMO) are supported within the AMU. The first are simply replacements for AMOs implemented by Intel® Xeon® CPUs. A user can switch back and forth between Intel® AMOs and AMOs implemented on the UV ASIC with the concept that uncontended variables are best accessed via AMOs on the processor socket and reused within its cache, while contended variables and those used for application wide barriers and reductions should be accessed via the AMOs implemented as part of the UV platform and reused within the UV ASIC update cache, reducing coherence overhead.

- **AMO Cache in Coherent Memory.** All AMOs are implemented in standard cache coherent user memory so no special resources need to be reserved. By using standard memory, applications have much higher limits in the number of AMOs that they utilize, with the AMO cache on the UV ASIC providing fast access to active variables across the NUMAlink.
- **Update Multicasting.** The update multicasting greatly improves the latency and repeat rate of update cache based AMOs. This is especially useful for barriers, collectives and contended locks.
- **Message Queues in Coherent Memory.** Just as is done with AMOs, message queues are implemented in standard system memory so users can set up as many queues as they would like.

2.2 SGI UV 2000 Interconnect Topology

SGI UV 2000 can be interconnected using a number of topologies, depending on system size and goals. A single SGI UV 2000 blade enclosure supports up to eight blades which are interconnected in a 3d enhanced hypercubetopology with a maximum of two NUMAlink hops between any node (see Figure 3a). A single rack UV 2000 system can be configured with the 12 ports available in blade-level routers into a 5d enhanced hypercube topology. As shown in Figure 3b, systems with from 3 to 16 blade enclosures (6 to 256 CPU sockets) are interconnected in a two level cross-bar connected 3d enhanced hypercube using the NUMAlink 6 links available in each blade plus additional rack-top NUMAlink routers. Larger configurations are achieved using 128-socket (2 rack) building block groups connected via this same 3d crossbar-connected enhanced hypercube.

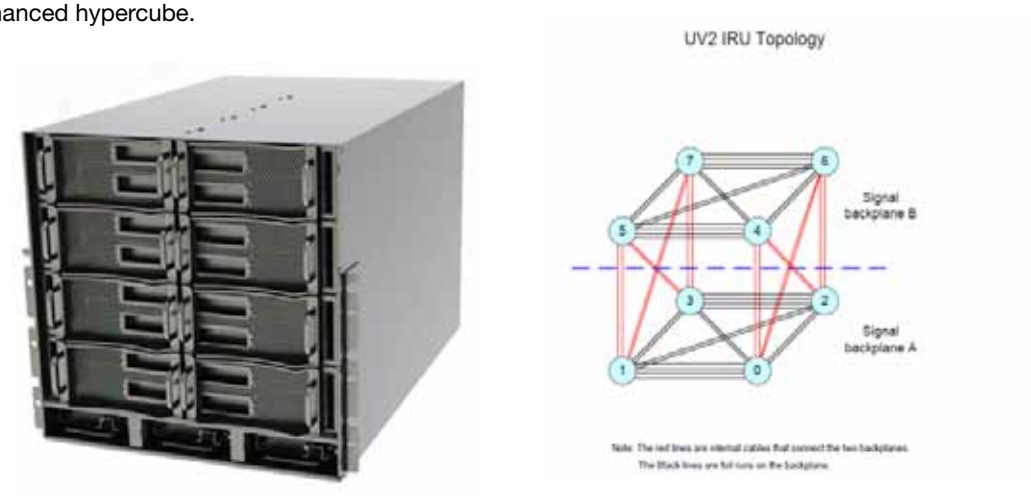

Figure 3a: SGI UV 2000 blade enclosure, 16 socket interconnect topology, four enclosures per rack.

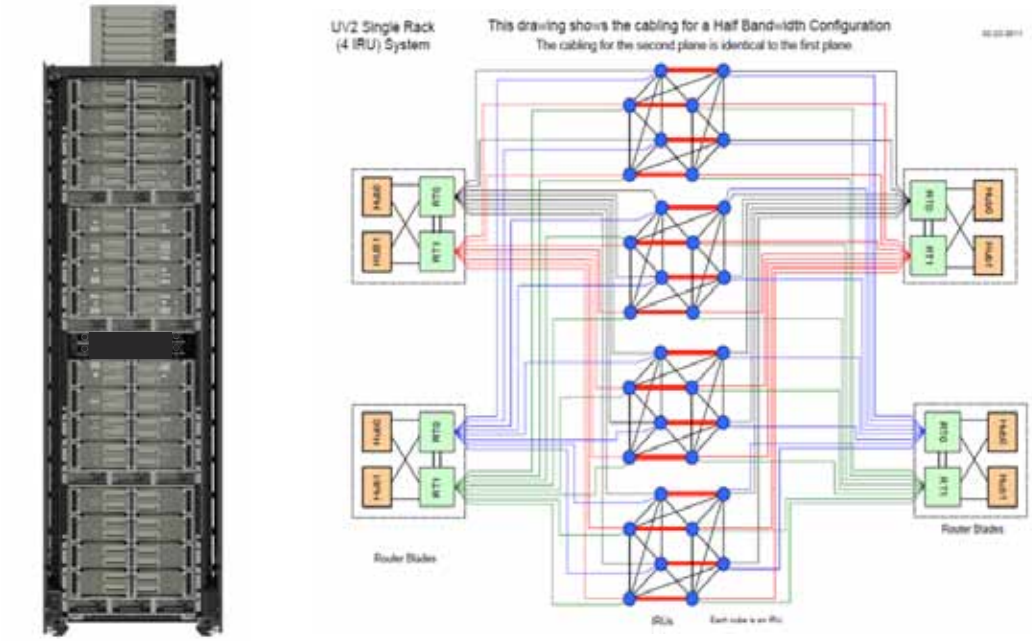

Figure 3b. 64 socket (up to 512 core,16 TB) configuration of SGI UV 2000 using 4 external NUMAlink 6 router blades (only ¼ of cables are shown). 4 rack 256 socket systems are created with blade enclosures and 32 external router blades in the same three-level router topology.

The use of different interconnect topologies based upon the size of the system maximizes bi-section bandwidth and minimizes furthest-node latency while managing deployment cost. In most cases, memory is read from the “local blade” or from blades that are topologically nearby. However, even in the worst case, the maximum MPI latency on a 32,768 socket (262,144 core) system is projected to be under two microseconds. For more details on MPI performance on SGI UV2000 please see our whitepaper entitled: “A Hardware-accelerated MPI Implementation on SGI UV 2000 Systems.”

3.0 Integrated System Functionality

Section 1 of this paper outlined some of the end user benefits achievable with SGI UV 2000, and section 2 outlined some of the specific architectural enhancements found in them. This section shows how specific architectural enhancements combine into five “collections” of higher-level functionality that deliver the benefits outlined in section 1.

- MPI Offload Engine (MOE)
- Petascale Memory and Data Intensive Computation
- Massive I/O
- Petascale Compute Enhancements
- Enhanced Reliability

3.1 The MPI Offload Engine (MOE)

The MPI Offload Engine, or MOE, is a set of functionality that offloads MPI communication workload from CPUs to the UV ASIC, accelerating common MPI tasks such as barriers and reductions across both GSM and GAM address spaces.

The MOE is similar in concept to a TCP/IP Offload Engine (TOE), which offloads TCP/IP protocol processing from system CPUs. The result is lower CPU overhead and memory access latency, allowing MPI applications to achieve better performance and scale to greater numbers of processors.

3.1.1. Reduced CPU Overhead and Latency for MPI_Send

With the SGI UV 2000 , all that is required to send an MPI message is for a CPU to place the payload information in a register on the GRU and issue a GRU message_send instruction. The GRU then takes over and transmits the data to a hardware-managed, memory-resident message queue slot on the appropriate remote compute blade. This significantly reduces the CPU’s workload, lowering message latency well below that seen on other hardware platforms and increasing single stream transfer rates — even when network path lengths are extremely long.

3.1.2. Fast Barriers and Reductions in Hardware

Both the MPI-1 and MPI-2 specifications include collective functions that provide simultaneous communications between all processors in a communicator group. SGI UV 2000 combines features of the AMU and update caches on the GRU to accelerate many of these important functions like barriers and reductions.

Barriers are used to synchronize all communicating processors and require that all MPI tasks first tell the master that they have reached the barrier, and then have the master inform all tasks that the barrier has been completed and that they can return to processing data. The AMU and update caches offload barrier updates from the CPU, accelerating update rates by up to 100x, while the GRU allows synchronization variable updates to be multicast to all processors in a communicator group, dramatically accelerating barrier completion.

Reduction functions are used to perform a global operation (such as SUM, MAX, etc.) across all members of a communicator group. The UV ASIC provides a number of functions in hardware that can be exploited to increase the speed of SGI UV 2000 reductions by 2-3x versus competing clusters and MPP systems.

3.1.3. Accessing the MOE

Applications that want to utilize the MOE combined capabilities will be able to do so by simply using the SGI Message Passing Toolkit (MPT) library which implements MPI. Other MPI libraries are also able to utilize the MOE by interfacing with a lower-level API which exposes key features in the GRU and AMU to user applications.

Lower level APIs can also be used to optimize shared memory applications and languages exploiting partitioned, global address space (PGAS) functionality. SGI is using this capability to provide compiler-level support for Unified Parallel C (UPC), a PGAS programming environment.

For further details on the benefits of running MPI applications on SGI UV 2000, please see our white paper, A Hardware-Accelerated MPI Implementation on SGI® UV 2000 Systems.

3.2 Large-scale Memory and Data Intensive Computation

SGI UV 2000 architecture will accommodate up to 64TB of coherent, global shared memory running under a single copy of Linux (the physical address limit of the Intel® Xeon® CPUs) and up to 8PB of globally addressable memory (the physical address space of the UV ASIC) directly accessible via user initiated GET and PUT operations, PGAS programming environments or MPI.

The GRU directly supports GET and PUT operations to move coherent snapshots of data from one GSM domain to another. The GRU also extends the number of memory references that can be outstanding from the individual blades into the larger NUMAlink 6 interconnect beyond the reach of an individual processor. This is especially important for achieving the highest scatter/gather performance possible.

In addition, UV 2000 directly addresses one of the critical issues in the use of petascale memory, variable initialization. The UV ASIC enables pages of memory to be initialized by the UV ASIC instead of by the CPUs, reducing startup times for large memory jobs from minutes to seconds.

3.3 Massive I/O Capabilities

Use of the Intel® QuickPath Interconnect (QPI) between processor sockets, I/O and the UV ASIC will eliminate bottlenecks and achieve aggregate I/O rates in excess of 1TB/second in a UV 2000 system.

The petascale compute and memory capabilities of the UV platform require comparable external I/O capabilities to move data to and from external storage and external networks. Each compute blade in a system can be configured with an I/O riser that provides a variety of I/O options. These include either Base I/O, low profile Gen3 X16 PCIe slots, full height Gen 3 X16 PCIe slots and on-board drive expansion (spinning disk or flash). The GSM design of SGI UV 2000 allows all I/O devices to be shared by all processors while high aggregate I/O rates as high as a theoretical 1 TB/sec can be supported by distributing physical connections across multiple blades that can be dedicated to I/O or share compute, I/O and memory functions.

However, the fastest I/O is no I/O, and the large memory capabilities of the UV 2000 can be utilized in several ways to eliminate or reduce physical I/O. First, an extremely large I/O buffer cache can be defined in system memory to increase I/O efficiency for applications such as out-of-core solvers or those that require large scratch files. Second, the multicast functionality of the GRU and GAM capability can be used to create multi-terabyte RAM disks with mirroring and write-through capabilities that run under separate copies of Linux. Mirroring and write-through capabilities provide increased reliability and can survive application and operating system crashes.

3.4. Large-scale System Compute Enhancements

Petascale applications involve computation and data structures distributed across thousands of nodes, and require efficient access to memory and rapid synchronization among threads or processes. The UV ASIC adds new memory access functionality such as distributed gather/scatter, coherent AMO update caches and asynchronous user-level memory-to-memory copies to provide efficient access to distributed data structures while allowing cache-line oriented CPUs to run with maximum efficiency. Advanced fairness capabilities have also been added to the NUMAlink protocol to ensure that large-message and small-message performance is maintained under heavy loading in petascale environments.

To support high-performance applications, SGI UV 2000 utilizes high-bandwidth NUMAlink 6 connections with multiple tiers of 16-port NUMAlink 6 routers to deliver a total bisection bandwidth of over 15 TB/s with an MPI latency of under two microseconds. The result is an architecture optimized for extremely low latency, high bandwidth access to both on- and off-blade memory resources.

3.5. Enhanced Reliability

A variety of hardware and software enhancements have been made to SGI UV 2000 for Intel® Xeon® processor E5-4600 product family to provide the reliability required for systems that scale to over 128K cores and 8PB of memory. To enhance reliability for petascale systems, the architecture includes extensive fault isolation, data path protection, monitoring and debugging functions to help ensure data integrity and prevent disruptions.

First, NUMAlink 6 protocols and the UV ASIC have been enhanced with additional error checking and retry capabilities to reduce transient communications errors by two orders of magnitude. Second, by offloading remote memory reads to the UV ASIC, failures that would have caused processor hangs can instead be retried or dealt with gracefully. Third, the UV ASIC provides safe mechanisms to communicate between nodes, even in the presence of node, memory or interconnect failures. And finally, system software has been enhanced to identify problematic nodes and memory and to remove them from the active pool of scheduled resources.

4.0 Conclusion

SGI UV 2000 is the sixth generation of global shared memory architectures from SGI, and was designed to increase application efficiency in highly scalable systems. Specific enhancements were identified after the careful study of a broad number of high performance technical and business applications, while a number of chip, protocol and system level enhancements were identified which would improve CPU performance, system scalability, reliability and manageability.

The features described in this paper provide an outline of the more critical capabilities developed for SGI UV 2000 and illustrate how they work together to create effective, scalable systems that can address today's largest and most demanding compute, memory and I/O-intensive problems.

Global Sales and Support: sgi.com/global