INNOVATION
FOR RESULTS™

White Paper

The SGI® Message Passing Toolkit

Optimized Performance Across the Altix Product Line

Table of Contents

This paper describes the SGI Message Passing Toolkit (MPT), an MPI implementation tailored to take advantage of the unique features of SGI Altix hardware. SGI MPT minimizes message latency and delivers optimized application performance across the entire Altix product line. SGI MPT performance, SGI MPT features, as well as some of the available tools for SGI MPT environments are discussed.

1.0 Introduction

The Message-Passing Interface (MPI) is a library specification enabling communication between the processors of a multi-processor computer or the hosts of a compute cluster during the execution of a parallel application. The SGI® Message Passing Toolkit (MPT) is an optimized MPI implementation that has been designed to deliver maximum performance across the entire SGI Altix product line (SGI Altix 4700, SGI Altix 450, SGI Altix ICE, SGI Altix XE).

With built-in support for a variety of possible interconnects including shared memory, NUMAlink, InfiniBand™, and TCP/IP sockets, SGI MPT automatically chooses the best available interconnect to achieve maximum performance, and can utilize more than one interconnect option as appropriate; shared memory may be used for communication within a host while communication to other hosts uses InfiniBand.

As a result, applications built with MPT will run correctly and optimally on different systems and clusters with varying interconnect configurations. On SGI Altix ICE and SGI Altix XE platforms, SGI MPT communicates over InfiniBand and scales across clusters of thousands of processor cores.

On the SGI Altix 450 and SGI Altix 4700 systems, MPT can use SGI NUMAlink to support a maximum of 512 processor sockets (1,024 processor cores) under one instance of Linux—a single system image (SSI). Clusters of SSI systems can scale to thousands of processors using low latency NUMAlink as the cluster interconnect. Single systems and clusters leverage global shared memory to keep latencies at a minimum while providing a streamlined SHMEM API implementation and user APIs for direct use of global shared memory.

SGI MPT is designed to deliver the highest possible performance through support for low-latency programming algorithms, targeted support for important MPI-2 features, and the right toolset for performance analysis and overall usability.

This white paper explores SGI MPT performance, unique SGI MPT features, and some of the more valuable tools available for use with SGI MPT.

2.0 SGI MPT Performance

The primary factor that dictates SGI MPI performance is message latency—the time required to send a message across an interconnect to another node. In this section we look at latency results for SGI Altix systems using SGI MPT, as well as some of the MPT features that contribute to performance across the Altix product line. A later section examines some of the unique features available on Altix systems using SGI NUMAlink.

2.1 Message Latency

2.1.1 Altix Global Shared Memory Systems

SGI Altix 4700 and SGI Altix 450 systems incorporate the shared-memory NUMAflex® architecture-- the industry's most scalable memory architecture, offering up to 128TB of globally addressable memory in a system. Global shared memory allows access to all data in the system's memory directly and efficiently, without having to move data through I/O or networking bottlenecks. The impact can be dramatic, including:

- Significantly lower MPI latency on NUMAlink versus InfiniBand
- Large memory space that allows problems to be solved in memory instead of out of core
- Direct memory access is orders of magnitude faster than I/O access to disk
- A larger SSI that enables OpenMP scaling far beyond any other system
- Support for the low latency SHMEM API via MPT
- Quasi-SMP style programming through global shared memory segments. SGI MPT supports an allocator that creates and sets up for sharing a memory segment that MPI processes can reference much like UNIX System V memory segments on SMP systems.

The SGI Altix 4700 and SGI Altix 450 are comprised of modular blades—interchangeable compute, memory, I/O and special purpose blades for configuration flexibility. The innovative blade-to-NUMAlink™ architecture enables users to mix and match eight standardized blade choices for system right-sizing.

Observed MPI latencies for Altix NUMAlink systems range between 1 and 2.3 microseconds. Latency remains low, even for communication between distant processors. Figure 1 illustrates this point.

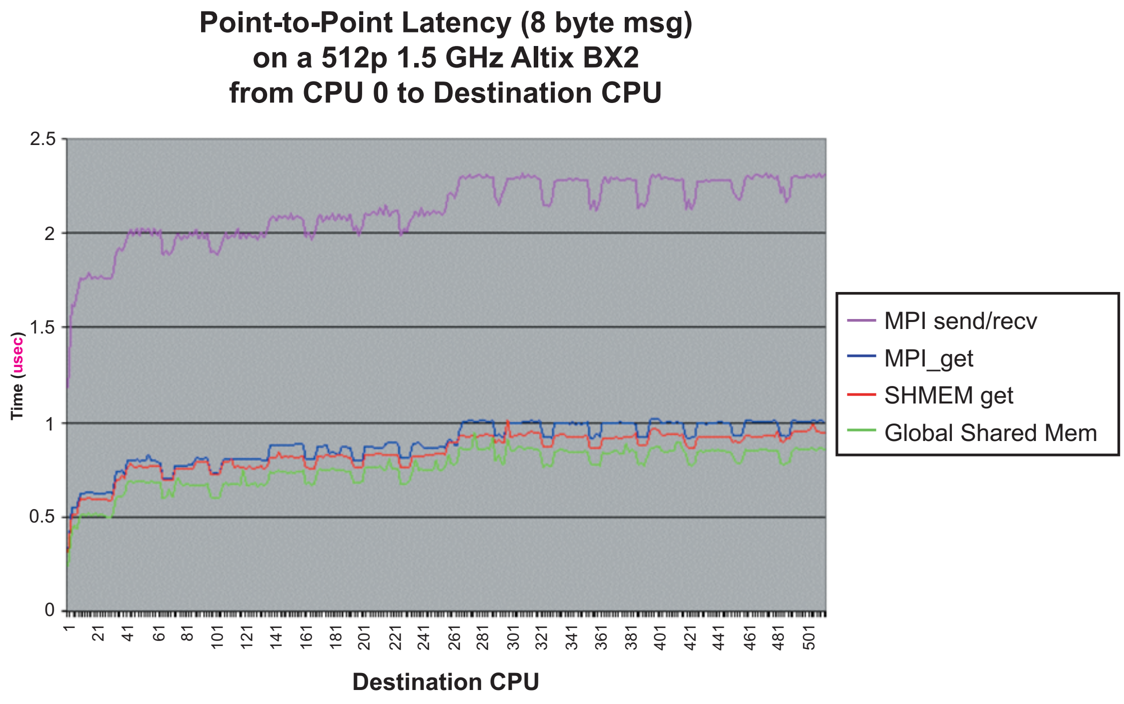

Figure 1. Latencies for various MPI functions versus SHMEM and Global Shared Memory.

2.1.2 Altix In niBand Systems

The SGI Altix ICE and SGI Altix XE server family utilizes industry-standard Dual- and Quad-core Intel® Xeon® processors. Either option is designed to support clusters of thousands of hosts using InfiniBand as the cluster interconnect.

The SGI Altix ICE platform is an integrated blade cluster system designed for demanding scale-out workloads. The SGI Altix ICE platform employs a blade architecture designed by SGI specifically to meet the unique needs of the HPC market. It delivers scalability, manageability, reliability, and price/performance.

SGI Altix XE rounds out the Altix family, with a choice of economical cluster-targeted servers that lead the industry in price/performance and low TCO (total cost of ownership). Altix XE extends SGI's technological innovations to the value-focused segment of the HPC market.

For small messages (8 bytes) MPT on SGI Altix ICE and SGI Altix XE achieve a message latency of approximately 3.4 microseconds running the Ohio State University MPI benchmark. For processors that are far apart, this result peaks at about 4.3 microseconds. For large messages (256KB) latency is approximately 124 microseconds. These results are consistent with the best results for other MPI implementations running on the same hardware and approach the interconnect limits. With the addition of single copy transfers and multi-rail support, SGI MPT achieves peak point-to-point bandwidth of 2400 MB/sec for 100 MB messages versus 900 MB/sec for the MVAPICH MPI implementation. These enhancements have shown performance improvements in many industry applications. (Both tests were run on the same platform: Altix ICE with 3.0GHz Xeon processors.)

2.2 Single Copy Transfers

Single copy transfers, also known as zero copy transfers, eliminates the extraneous copying that can occur when sending MPI messages. The process that calls MPI_Send specifies a user array that contains the data. The receiving process calls MPI_Recv and specifies a data array to receive the data. The SGI MPI library transfers the data from the sender's user buffer to the receiver's user buffer without any intermediate library buffering of the data.

The means by which this occurs is implementation specific. On InfiniBand systems, this is accomplished via use of RDMA functions with memory registration. On NUMAlink systems single copy transfers are accomplished with an optimized, processor-driven bcopy that references a mapping to memory that is set up by use of the XPMEM software stack.

2.3 In niBand Multi-Rail Support

A rail is an independent InfiniBand network. Instead of adding more endpoints to an existing network, a multi-rail system adds parallel networks to achieve higher bandwidth. Multi-rail is the standard configuration for Altix ICE. It is available as an option with Altix 4700 and Altix XE systems. Most other MPI implementations do not offer multi-rail support.

SGI MPT utilizes multiple InfiniBand rails to perform message pathway distribution and message striping. Message pathway distribution is done by strategically mapping individual routes (source to destination) to the available rails. Since routes are mapped to different rails, more aggregate bandwidth is available in situations where many MPI processes are communicating at the same time. SGI MPT performs message striping by sending portions of large messages on each rail in parallel with the effect of nearly doubling the effective MPI point-to-point bandwidth.

2.4 Optimized Collectives

Both the MPI-1 and MPI-2 specifications include collective functions that provide simultaneous communications between all processors in a communicator group. SGI has optimized many of these collective functions to streamline performance:

- Alltoall collective functions and other collectives with large messages sizes are optimized on both NUMAlink and InfiniBand via single copy data transfers.
- Barriers are used to synchronize all communicating processors. Barriers are optimized on NUMAlink systems by use of a fetchop tree algorithm. On InfiniBand systems they are optimized by using shared memory to synchronize MPI processes within the same host.
- Reduction functions are used to perform a global operation (such as sum, max, etc.) across all members of a group. These functions are accelerated by combining an optimized shared memory reduction within the node with a recursive doubling algorithm for the inter-node phases of the reduction.

2.5 Tunable Environment Variables

SGI MPT includes a large number of tunable environment variables. While many of these are part of the MPI specification, SGI has introduced a number of variables that allow you to control process placement and message buffering to fine tune performance.

Process Placement:

- **MPI_DSM_DISTRIBUTE:** Activates NUMA job placement mode. Ensures that each SGI MPI process gets a unique CPU and physical memory on the node with which that CPU is associated.
- **MPI_DSM_CPULIST:** Specifies a list of CPUs on which to run an MPI application.
- **MPI_DSM_PPM:** Sets the number of MPI processes per memory node.

Message Buffering:

- **MPI_DEFAULT_SINGLE_COPY_BUFFER_MAX:** If the size of the cumulative data to be transferred is greater than this value, then MPI will attempt to send the data directly between the processes' buffers and not through intermediate buffers.
- **MPI_IB_SINGLE_COPY_BUFFER_MAX:** Similar to the previous variable except applies specifically to InfiniBand.
- **MPI_BUFFER_MAX:** Specifies a minimum message size, in bytes, for which the message will be considered a candidate for single-copy transfer.
- **MPI_DEFAULT_SINGLE_COPY_OFF:** Disables the single-copy mode. (By default, single copy is enabled.)
- **MPI_BUFS_PER_PROC:** The number of 16KB buffers to allocate for each MPI process.
- **MPI_BUFS_PER_HOST:** The number of 16KB shared message buffers to allocate for each host.

3.0 SGI MPI Features

A number of additional features supported by MPT contribute to the usability of the SGI implementation.

3.1 MPI-2 Content

The SGI MPT implementation is compliant with the 1.0, 1.1, and 1.2 versions of the MPI Standard specification. In addition, SGI has implemented a number of frequently requested features from MPI-2:

- **MPI I/O**. The optimizations required for efficiency in a parallel computing environment (such as grouping, collective buffering, and disk-directed I/O) can only be implemented if the parallel I/O system provides a high-level interface supporting partitioning of file data among processes and a collective interface supporting complete transfers of global data structures between process memories and files. Further efficiencies can be gained via support for asynchronous I/O, strided accesses, and control over physical file layout on storage devices. MPI I/O provides these facilities.
- **MPI Thread Compliance and Safety.** MPI may be implemented in environments where threads are not supported or their use is undesirable. Therefore, MPI implementations are not required to be thread compliant. However, MPI-2 defines optional thread compliance functions. When threads are in use, each thread in a process can issue MPI calls, but threads are not separately addressable. A message sent to a process may be received

by any thread in the process. There are four options for the level of thread support. MPT supports all four options.

- **MPI_THREAD_SINGLE.** Only one thread executes
- **MPI_THREAD_FUNNELED.** Only the main thread will make MPI calls.
- **MPI_THREAD_SERIALIZED.** Only one thread at a time may make MPI calls.
- **MPI_THREAD_MULTIPLE.** Multiple threads may call MPI without restriction. (MPT requires libmpi_mt.so for this option.)

- **MPI Process Spawn.** MPI-1 applications are static: processes can't be created or deleted after an application has started. MPT implements features from the MPI-2 specification which allow an application to spawn new processes as necessary.
- **One-sided Communication.** Normal send/receive communication requires matching operations by the sender and receiver. In some cases, processes may not know which data in their own memory needs to be accessed or updated by remote processes. MPT implements the MPI_PUT and MPI_GET calls of MPI-2, allowing a process to read and write the memory of another process many times between synchronization points in a program.
- **Language Bindings.** The earlier MPI specifications did not include bindings for C++ or Fortran 90. SGI MPT includes support for C++ (except for C++ interfaces for MPI-2 functions) and partial support for Fortran 90.
- **Process Communication via MPI Ports.** Establishing process communication where none existed before and where there is no parent/child relationship can be complex. MPT implements the MPI-2 feature which allows a server process to establish a port to which client processes can connect.

3.2 Array Services

Secure Array Services which is part of SGI ProPack provides the same functionality as Array Services, but uses stronger encryption and security algorithms. It includes administrator commands, libraries, daemons, and kernel extensions that simplify the execution, monitoring and control of MPI programs across a cluster. Array Services provides the notion of an array session, which is a set of processes that can run on different cluster nodes. With Array Services, an array session handle (ASH) is used to logically group related processes that may be distributed across multiple systems, facilitating accounting and administration.

Array Services utilities let an administrator query and manipulate distributed array applications, and dramatically simplify common operations. For instance, with many MPI implementations, starting an application on a cluster can be awkward and complicated. Array Services makes the task simple and straightforward.

4.0 SGI MPT on Altix NUMAlink Systems

User programs and libraries on SGI Altix 450 and SGI Altix 4700 systems have direct, global access to memory across the whole system. This feature allows SGI MPI more freedom to implement efficient data sharing algorithms. The SGI Altix 4700 system can be administered as a single large SMP system with up to 2,048 processors, or as a cluster of such hosts.

All hosts in a cluster are individual partitions within a larger NUMAlink interconnect domain. The Linux software distribution installed on Altix systems is augmented by SGI software that provides memory sharing within and across hosts (XPMEM), as well as optimized libraries and other features. The software layers related to memory access on NUMAlink are shown in figure 2.

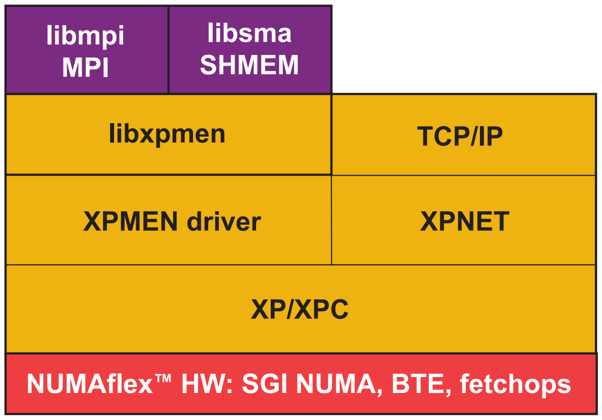

Figure 2. Software/hardware stack for NUMAlink interconnect.

SGI programs attach all memory in the static and common blocks segments, the stack segments, and the heap segments of an MPI process into the virtual address space of every other MPI process in the job. This allows free access to MPI queues and data structures, as well as user data areas passed as send and receive buffers or targeted by put or get operations or accessed via global pointers into globally shared data regions.

4.1 SHMEM™ Programming Model

MPI extends the explicit parallel programming capabilities of MPI through the addition of the SHMEM parallel programming model. The SHMEM library provides the fastest interprocessor communication using data passing or one-sided communication techniques. The SHMEM library also contains a facility for assigning global pointers that allow data in another cooperating process to be accessed directly via load/store for communication or synchronization.

In addition, the SHMEM library includes a number of highly optimized functions for collective operations such as global reductions. Since it can be implemented very efficiently on globally addressable shared- or distributed-memory systems, use of this library improves communication latency by an order of magnitude over optimized MPI implementations on SGI NUMAlink systems. Some of the one-sided communications concepts introduced in SHMEM have been incorporated into the MPI-2 specification.

4.2 Global Shared Memory Support

SGI Altix servers have global shared memory. This means that memory segments can be allocated on or striped across memory nodes anywhere in the system and be referenced via ordinary load/store instructions. MPT provides a global shared memory allocator called GSM_Alloc() that may be called within MPI or SHMEM programs to allocate remote or distributed memory segments.

When the ability to directly load and store to remote memory is correctly exploited, most software overhead is eliminated yielding substantially reduced effective latencies.

4.3 Process Placement

Other important improvements include memory placement, process pinning, and CPUset support. When either solving a complex problem that needs large numbers of CPUs and memory or when multiple, unrelated problems are running simultaneously on the same large system, efficient resource allocation enables an application or batch job to complete within a predictable and consistent time period. SGI provides the commands **cpuset**, **dplace**, and **omplace** to ensure a particular workload or batch job can efficiently use the available CPU and memory resources and to help ensure multiple jobs can allocate and use the resources they require without interfering with each other. These tools can also prevent a smaller job from thrashing across a larger pool of resources than it can effectively use. Support for the utilities **dplace**, **cpuset**, and **omplace** is implemented using SGI's libcpuset library which provides the kernel support and library calls for implementing this functionality. Programmers can make explicit calls to the library to incorporate these functions directly into applications.

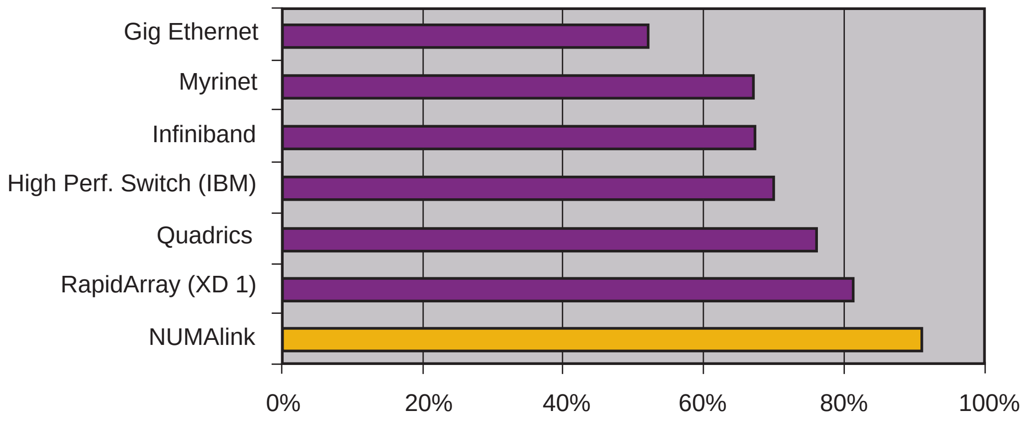

Figure 3. Interconnect efficiency for common MPI applications. Calculated as Linpack NxN Rmax/Rpeak.

5.0 Tools

A variety of tools are available that can help profile, debug, and execute SGI MPI applications. The following subsections describe several of the more useful ones from SGI and a variety of other sources.

- **Perfcatch.** The perfcatch utility is a useful lightweight tool for first pass MPI performance analysis that SGI provides with SGI ProPack. It runs an MPI program with a wrapper profiling library that prints MPI call profiling information to a summary file upon program completion.
- **TotalView.** The TotalView Debugger from TotalView Technologies is a comprehensive debugger with enhanced parallel and thread support for use with SGI MPI and other MPI implementations as well as other parallel programming models.
- **Intel Trace Analyzer and Collector.** Intel® Trace Analyzer and Collector provides information critical to understanding and optimizing MPI cluster performance by quickly finding performance bottlenecks with MPI communication. Version 7.1 now includes trace file comparison, counter data displays, and an MPI correctness checking library.
- **Workload Management.** Running MPI applications in a production environment requires effective workload management and job scheduling. SGI MPI applications can be managed by both PBS Professional from Altair and Platform LSF, the two leading workload management applications.

6.0 Conclusion

SGI MPT can be used to create parallel applications that make efficient use of SGI hardware in both multiprocessor and cluster environments. MPT delivers superior performance with message latency ranging from 1 to 2.3 microseconds on the Altix 450 and 4700 systems, and 3.4 microseconds on Altix XE and Altix ICE. Single copy transfer capabilities eliminate time-consuming memory copies to enhance bandwidth on all platforms. A targeted set of MPI-2 features add important capabilities over and above MPI-1.2.

The NUMAlink interconnect used on all SGI Altix 450 and SGI Altix 4700 systems makes it possible to share memory between cooperating hosts in a cluster. The SHMEM programming model offers an extension to MPT that allows programmers to improve latency by an order of magnitude relative to standard MPI communications.

A targeted set of tools for debugging, optimization, and workload management round out the SGI MPI offering to yield a productive development environment that streamlines application creation and execution.

sgi®

Corporate Office
1140 E. Arques Avenue
Sunnyvale, CA 94085
(650) 960-1980
www.sgi.com

North America +1 800.800.7441
Latin America +55 11.5185.2860
Europe +44 118.912.7500
Japan +81 3.5488.1811
Asia Pacific +61 2.9448.1463

4106 [04.22.2008] **J15375**