# Scali MPI - ScaMPI

## Overview

Scali's MPI implementation, ScaMPI™, is fully compliant with the MPI 1.2 specification from Message Passing Interface Forum, ensuring application compatibility. ScaMPI is designed to support scalable systems, focusing on sustained performance of systems up to hundreds of nodes. We are proud to offer a complete solution with extremely good latency numbers and excellent bandwidth in a thread-safe implementation. This is achieved by exploiting the *shared address space architecture* of SCI, Scalable Coherent Interface (IEEE Std. 1596-1992).

Scali's commitment to performance is combined with extensive RAS (Reliability, Availability and Serviceability) features required for large systems; ScaMPI continues to operate in an error free manner during cable exchanges, dynamic change of interconnect routing tables, interconnect reset etc.

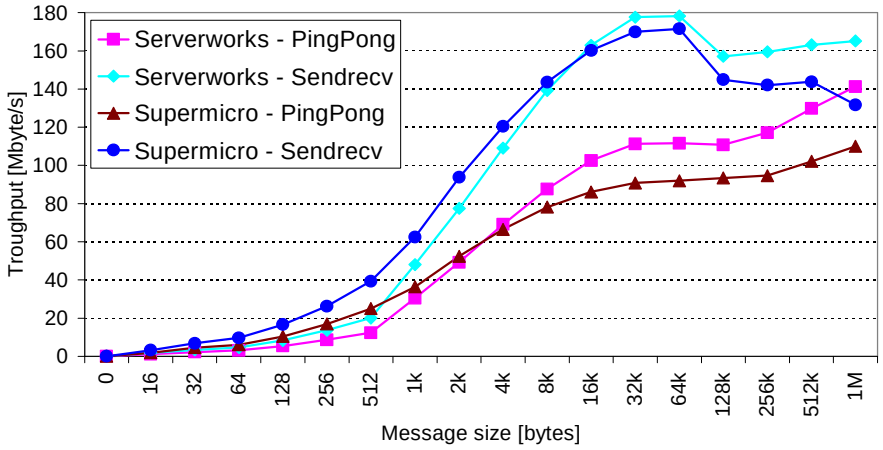
## Scalable Systems

Scaling an application to a large number of MPI processes, while keeping the problem size constant, requires more frequent exchange of messages which becomes smaller and smaller. This is the situation when the goal of the scaling is to reduce the execution time. ScaMPI has been developed with these applications in mind and provides extremely low latency for short messages, e.g. 4.5 µsec. for a zero byte message on a Supermicro MB's[1)]. Furthermore, ScaMPI supports multithreaded applications. Thus, it is possible to reduce the number of MPI processes from one per CPU to one per node in the system, hence improving communication performance, by using e.g. OpenMP or pthreads.

ScaMPI has also included a highly optimized implementation of MPI's collective operations. These are ideally suited for solving large problems, where high sustained bandwidth is an important requirement. ScaMPI's collective operations show close to linear performance scaling with growing system size.

## Features and benefits

- **Highly Optimized Implementation** - ScaMPI's message latency, measured as half the round-trip delay (*ping-pong-half*) of a zero length MPI message, is less than 5 µsec. Sustained bandwidth exceeds 180 MBytes/s over a 32 bit 66 MHz PCI bus.
- **Multi-Thread-Safe and Hot** - Multithreaded applications can fully exploit ScaMPI and multiple threads can simultaneously request services and conduct communication using ScaMPI.
- **Fault Tolerance** - ScaMPI operates transparently to transient networks errors, interconnect reset or change of routing tables.
- **Automatic selection of physical transport mechanism** - ScaMPI will automatically select the best suited transport medium for MPI messages; within a Symmetrical Multiprocessor node (SMP), shared memory will be used, whereas Scalable Coherent Interface -SCI - interconnect will be used between the nodes.
- **UNIX Command Line Replication** - The command line arguments to the application will automatically be provided to all MPI processes. This avoids tedious parsing and broadcast of parameters to the other MPI processes.
- **Exact Message Size Option** - Although not required by the MPI specification, ScaMPI has the option of verifying match between the effective received and transmitted message size.
- **MIMD Support** - ScaMPI supports the Multiple Instruction - Multiple Data model by having provision of launching different executables which constitute the whole MPI application.
- **Heterogeneous Clusters** - ScaMPI supports applications consisting of MPI processes hosted on nodes running different operating systems.
- **Graphical Frontend** - ScaMPI application can either be launched from *mpimon*, Scali's MPI start-up program, from the MPICH compatible *mpirun* scripts, or from the Scali graphical desktop.




- **Daemon launch** - ScaMPI uses a daemon to start the MPI processes. This enhances security and avoids use of the remote shell daemon (rshd).
- **Manual launch mode** - ScaMPI has the ability to start X terminals enabling manual start of selected MPI processes. This is useful if special trace-, profiling-tools, or debuggers are to be applied on selected portions of the MPI processes.
- **Built-in MPI call trace and timing support**.Through environment settings timing or calling parameters of MPI calls can be displayed. Single or groups of functions can be selected for monitoring using regular expressions.
- **Support for debuggers** - ScaMPI fully supports Etnus' TotalView distributed debugger. Arbitrary selected processes can alternatively be debugged using GNU gdb.

## Performance example

The figure above shows the performance of the Pallas MPI benchmark for one way and two way traffic between two PCs[1)] interconnected with SCI (Dolphin D330 SCI cards utilizing a 32 bit / 66 MHz PCI bus). In the *PingPong* test a message is sent back and forth between two nodes, while in the *Sendrecv* test the processes send and receives concurrently.

Pallas defines MByte as $2^{20}$ bytes.

## Platforms supported

| OS | Version | Architecture |
|---|---|---|
| Linux | RedHat 6.0-7.0 | i86pc / IA64 / Alpha |
| Linux | SuSe 6.4-7.0 | i86pc / IA64 / Alpha |
| Solaris | 2.6 or 7 | i86pc / UltraSPARC |

## Availability

Now

1) Test were run using Pallas MPI benchmark version 2.2 between two workstations

| System | Software | Motherboard | Processor(s) |
|---|---|---|---|
| Serverworks | SSP 2.1 & RedHat Linux 6.2 | Tyan Thunder 2500 (HE Serverworks) | Dual Pentium III 666 MHz (Coppermine) |
| Supermicro | SSP 2.1 & RedHat Linux 6.2 | Supermicro PIII - DME (840) | Dual Pentium III 733 MHz (Coppermine) |