



IBM

High Performance Computing for

AIX

Using

InfiniBand

White Paper

October, 2007

George Chochia

Frank Johnston

John Lewars

IBM Systems and Technology Group

Poughkeepsie, New York



Abstract

In this paper, we provide an overview and describe the components and performance of an HPC solution for AIX® on POWER™ using InfiniBand® as the cluster interconnect. The paper contains performance results for several communication benchmarks running on IBM System p™ servers with the IBM Host Channel Adapter driving InfiniBand switches, and on IBM JS21 Blade Center servers. We provide some guidelines for end users on the various environmental settings that can be tuned to improve the performance of applications. The specific measurements are for the MPI communication protocols. The MPI measurements were done with blocking and non-blocking sends and receives between pairs of MPI tasks using a benchmark which we call Spark. Selected results for some industry standard benchmarks are also included.

This paper includes performance data for up to 4-link configurations.

Contents

Abstract	2
Overview.....	4
InfiniBand for Technical Computing	5
Communication Software Stack	9
Supported Cluster Configurations	6
Scaling Characteristics.....	9
Details of Software components	9
MPI Performance	9
MPI Latency.....	10
MPI Bandwidth	10
Application Benchmarks.....	12
NAS Parallel Benchmarks.....	12
HPC Challenge Benchmark	13
Conclusion.....	14
More Information.....	15
Appendix A. Description of Measured Environments	15
Appendix B. MPI tuning variables	17
Appendix C. Determining Device Driver Level	18

Overview

IBM's objective is to provide an "industrial strength" high performance computing offering for AIX on POWER using InfiniBand switches as the high-speed cluster interconnect. The InfiniBand interconnect technology is particularly well suited for scalable high bandwidth, low latency applications such as technical computing. InfiniBand is a relatively new technology architected in the 21st century that incorporates advancements in high-speed networking and cluster computing that were developed recently.

The components described here constitute a complete high performance computing solution consisting of existing IBM program products modified to take advantage of InfiniBand. All components were integrated and tested together to develop a quality product. This testing included the IBM program products and the InfiniBand components.

This solution includes:

- The IBM GX dual-port 4X IB HCA for POWER5™ processor-based servers or the Cisco PCI-Express 4X IB HCA for JS21 servers.
- Cisco InfiniBand Switches
- IBM Power Architecture® technology-based servers: System p5™ 575, 550 Express and 550Q Express or PowerPC® 970MP BladeCenter JS21 servers
- IBM General Purpose File System™ (GPFS™) for AIX
- Tivoli® Workload Scheduler LoadLeveler® for AIX
- IBM Parallel Environment for AIX on POWER
- IBM Parallel Engineering Scientific Subroutine Library
- IBM Cluster Systems Manager for AIX

These components were integrated and qualified on servers running AIX on POWER. IBM MPI and LAPI interfaces are layered over the IB user verbs library; IBM's GPFS file-system uses the IPoIB driver.

The hardware components for POWER5 processor-based solutions consist of the IBM GX dual-port 4X IB HCA's and Cisco IB switches. The System p5 POWER processor-based servers supported are the models best for technical computing: the p5-575, p5-550 Express and p5-550Q Express. Not all POWER5 processor-based hardware types were measured.

New types of HPC clusters based on IBM BladeCenter JS21 server running AIX connected to a Cisco InfiniBand switch via Cisco PCI-E 4x adapters are now supported.

The switches are available in configurations from 24 ports to 96 ports. The largest network configuration supported for p5-575 clusters is four parallel 96-port networks. JS21 clusters are limited to a single network.

InfiniBand for Technical Computing

InfiniBand switches were designed for use in Clos networks, sometimes referred to as a fat-tree topology; this makes them ideal for technical computing. Clos networks have high bisectional bandwidths and provide multiple paths between most points of the network. Over the past decade, high performance computing and interconnect vendors have been providing Clos networks as the high-speed networks connecting computing servers as part of supercomputers. The InfiniBand standard provided by the InfiniBand Trade Association (IBTA) brings standardization to this concept. Standardization allows for multiple vendors to provide interoperability and complementary products to the end users. Today InfiniBand enjoys a growing number of vendor building related products. Some examples are storage products that attach directly to the InfiniBand switch and gateways that route traffic between InfiniBand and Ethernet or Fibre Channel. These products coupled with its inherent high bandwidth and low latency provides the necessary elements for creating technical computing clusters with InfiniBand.

Communication Software Stack

The IBM MPI communication software stack includes: IB adapter driver, IB verbs library and IPoIB driver, IBM Parallel Environment (including POE, MPI and LAPI), Tivoli Workload Scheduler LoadLeveler and GPFS.

For IBM MPI user space access to InfiniBand, the Verbs library is dynamically linked into the application allowing communication over the InfiniBand Fabric.

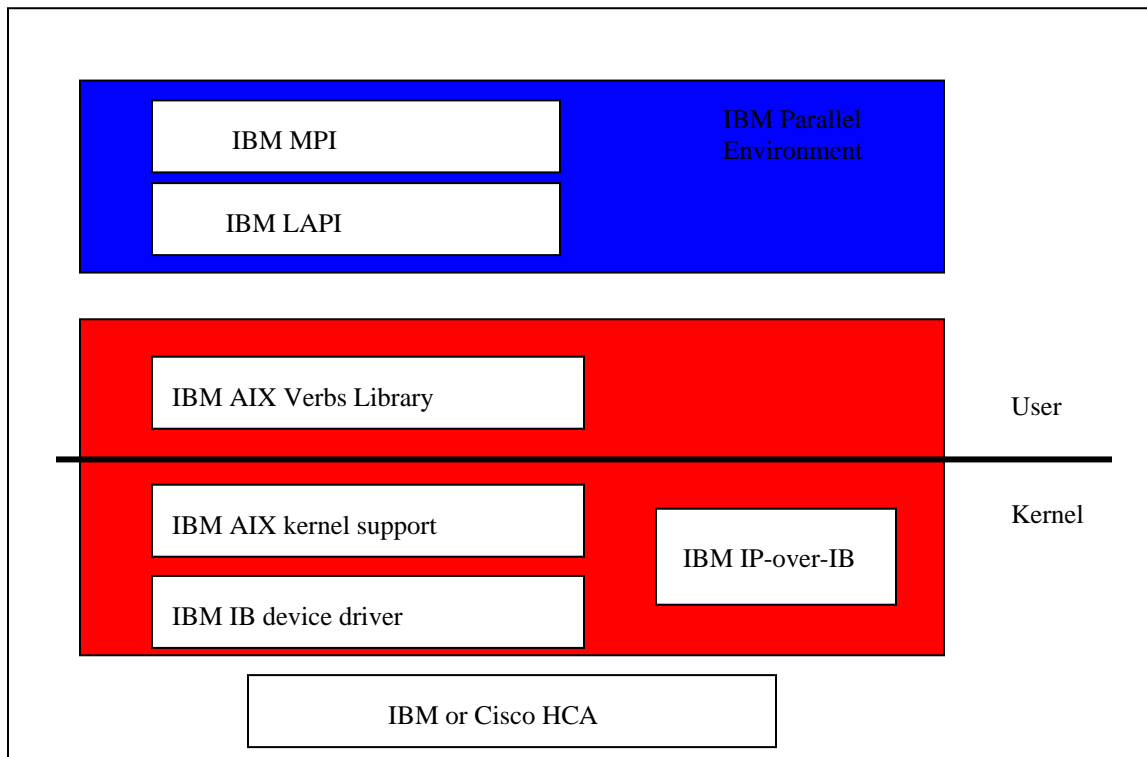


Figure 1- HPC Software Stack Structure

IBM's LAPI transport layer uses the unreliable datagram (UD) transport of InfiniBand. UD provides a scalable mechanism for running applications to thousands of tasks since the amount of InfiniBand and system resources scales well as the number of tasks in an MPI job increases. A goal of our implementation was to keep the memory footprint devoted to the communications software low, and UD accomplishes this.

Supported Cluster Configurations

Supported servers include POWER5 processor-based servers: p5-550 Express and p5-575 and IBM BladeCenter JS21 server. The p5-575 is available as a 1.9 GHz 16-core or a 2.2 GHz 8-core. The p5-550 Express is available as a 2- or 4-core system with 1.65 GHz, 1.9 GHz or 2.1 GHz processor cores.

A building block of a p-575 server in a Dual-Core Module (DCM) packaging the dual-core POWER5+™ chip with shared L2 cache and bus interfaces is shown in Figure 2. p5-575 1.9GHz server has 16 cores in eight DCMs

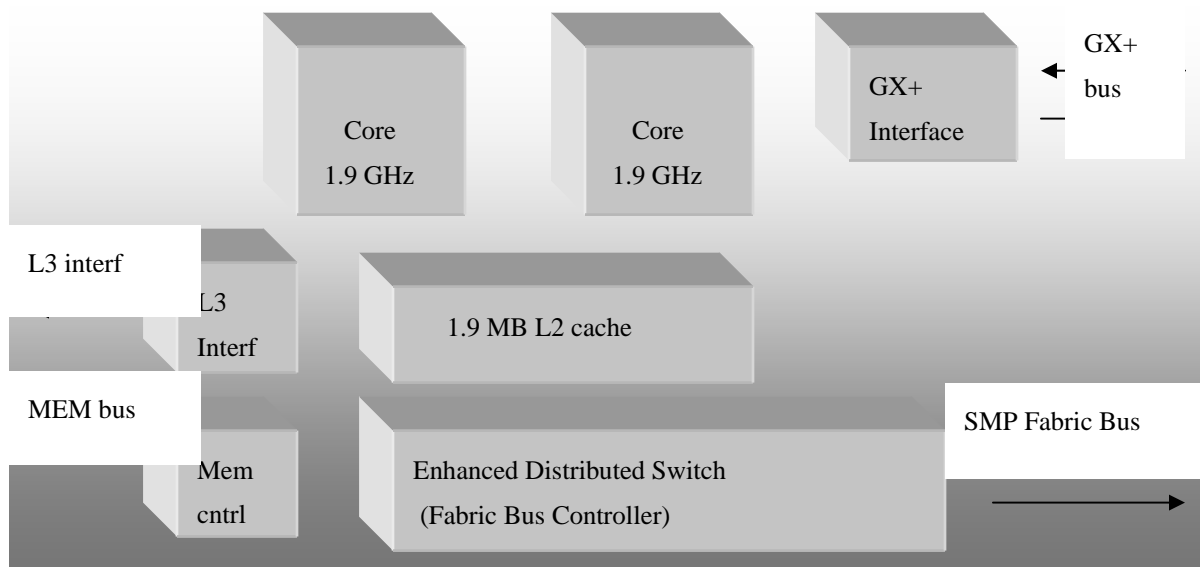


Figure 2 Logical view of a Dual-Core POWER5 Module

The IBM GX dual-port 4X IB HCA has two InfiniBand 4X ports and can attach to two separate InfiniBand switch networks. Currently two IB HCA adapters are supported per p5-575 and one adapter per p5-550. The HCA plugs into a slot in the IBM POWER processor-based server known as a “GX” slot. A GX slot connects directly into the POWER processor chip’s I/O bus without passing through an I/O bridge. This reduces the message latency.

The IBM BladeCenter JS21 is available in 2-core 2.7 GHz or 4-core 2.5 GHz configurations. Each core is a 64-bit PowerPC 970MP processor. A chassis has room for fourteen blades. More information can be found in the IBM Redpaper: “IBM BladeCenter

JS21 Technical Overview and Introduction”. Each BladeCenter JS21 is connected to a Cisco InfiniBand Switch Module by a Cisco 4x PCI-E adapter. There could be one or two switch modules per chassis. Each switch module has eight links connecting it to a Cisco InfiniBand switch. Two switch modules provide the equivalent of fourteen links connectivity from the JS21 blades to the InfiniBand switch, one link per PCI-E adapter card. The switches are available in configurations from 24 to 96 ports and only a single switch network is supported. Figure 3 shows the IBM BladeCenter JS21 logical data flow.

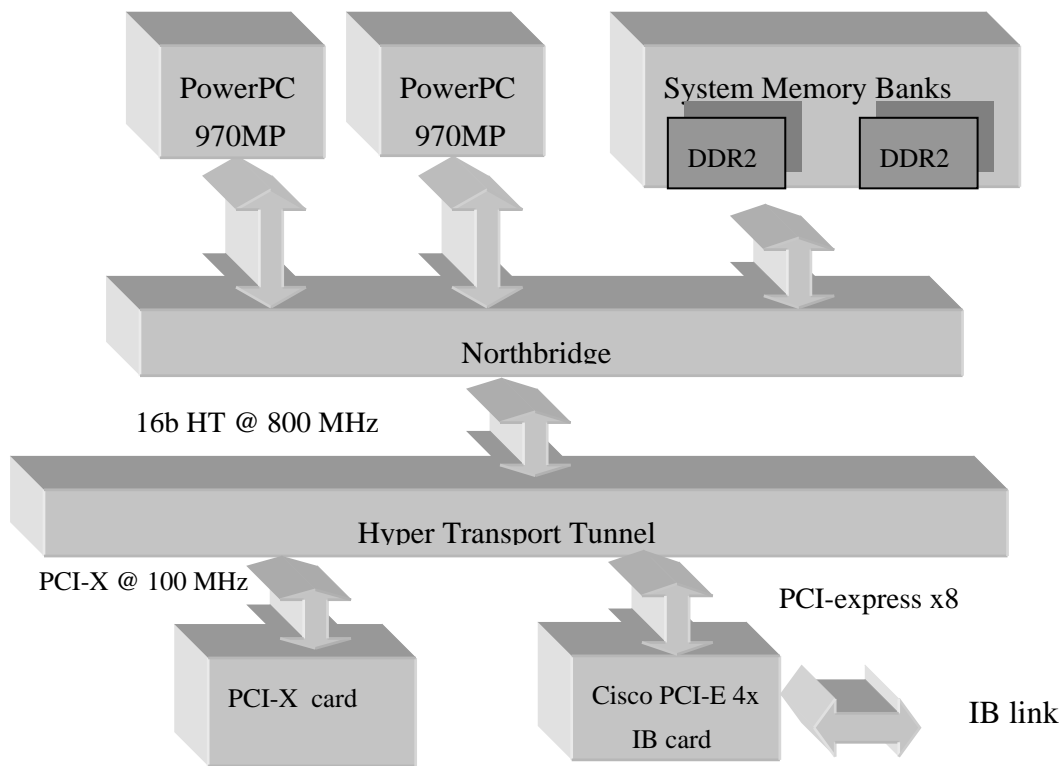


Figure 3 BladeCenter JS21 logical data flow

A single network, Figure 4, dual network, Figure 5, and quad networks, Figure 6, configurations are supported. JS21 is supported only with single port per Cisco PCI-E adapter in a single network configuration. BladeCenter JS21 can communicate with p5-575 and p5-550 via IPoIB, but running MPI user space jobs in heterogeneous configurations which include blades is not supported.

Dual networks provide somewhat more bandwidth than a single network using one adapter per node. Quad networks use both GX slots in the p5-575 servers. Each link has to go to a different network or subnet. We do not support multiple links per subnet per server. The performance using quad networks improves over dual networks; details are provided in the MPI performance section.

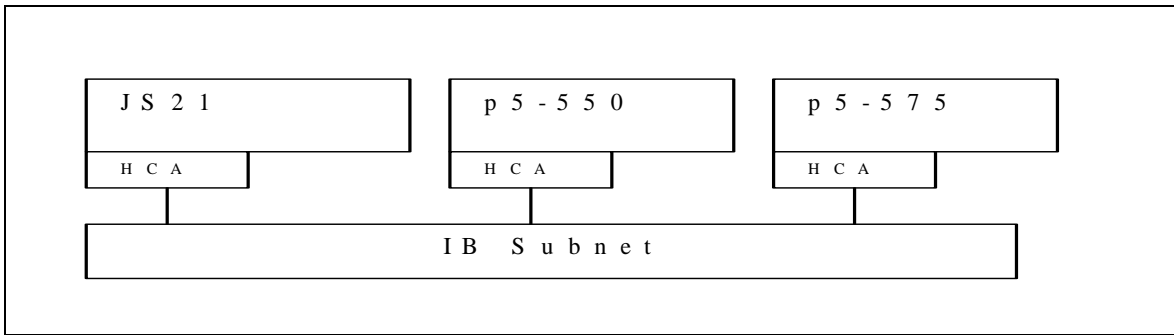


Figure 4 Single Network

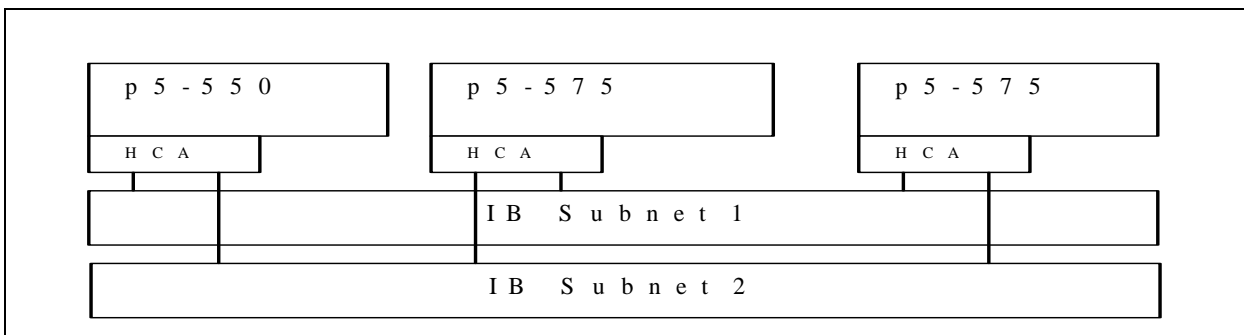


Figure 5 Dual Networks

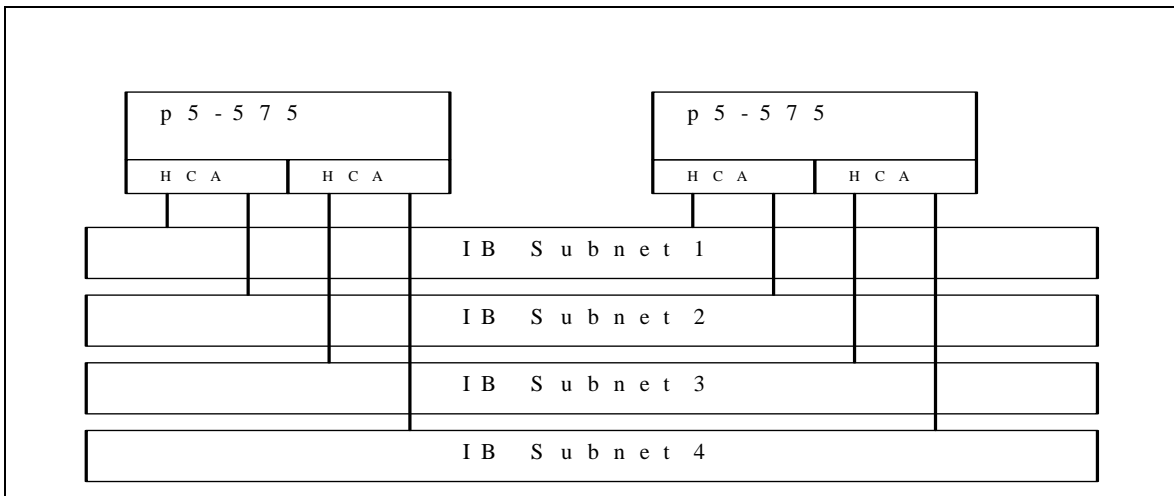


Figure 6 - Quad Networks

Switches are available with 24 to 96 4X ports. The Cisco SFS 7000P Server Switch has 24 4X ports and the Cisco SFS 7008P Server Switch comes with between 48 and 96

ports depending on the number of line interface modules (LIM) ordered. Each LIM contains 12 4X ports.

Scaling Characteristics

The scaling limits of the p5-575 based offering allow up to four parallel switch networks of 96 endpoints each. This allows up to 96 p5-575 POWER servers to be in a cluster with jobs up to 1,536 tasks. The limit was determined by virtue of 96 endpoints being the largest single switch supported by our offering.

The scaling limits of the JS21 based offering allow up to 12 BladeCenter chassis to be connected to a 96 port Cisco switch. Each chassis can hold up to 14 blades, for a maximum of 168 blades. If each of the 168 blades had four cores, then it would be possible to run jobs with up to 672 tasks.

The other important consideration is how the applications scale as they contain more tasks spread over more nodes of the cluster. The model picked for task-to-task communications scales well in terms of InfiniBand resources, Queue Pairs, and system memory. The amount of memory used by the IB support per task is constant in our model and does not grow with the number of tasks. Not all communication models using InfiniBand can achieve this efficient use of memory for very large jobs.

Details of Software components

- IBM Parallel Environment (PE) for AIX on POWER has functions to run MPI and LAPI jobs over user-space and IP communications stacks over the InfiniBand interconnect. For user-space jobs on POWER5 processor-based clusters, multiple links can be used to provide fault tolerance and higher aggregate bandwidth. If one link fails, its traffic is routed to the other link in the adapter.
- Tivoli Workload Scheduler LoadLeveler provides the ability to schedule jobs over InfiniBand's user-space and IP stacks.
- Parallel ESSL for AIX on POWER was tested running jobs over the InfiniBand Fabric.

MPI Performance

In this section, we restrict ourselves to the performance of MPI as measured using benchmarks using only MPI communications. No computation is involved in the benchmarks used to measure peak communication performance under different conditions. All hardware used was configured to have the maximum number of cores and enough memory installed to achieve peak memory performance. The p5-575 machines are 16-core SMP servers based on IBM POWER5 technology. Both 2-core 2.7 GHz and 4-core 2.5 GHz BladeCenter JS21 servers were tested.

MPI Latency

Table 1 shows MPI PingPong latency measurements. Appendix A describes the test environment. These measurements were run using a thread-safe MPI library, and C interface to MPI. Most measurements in the literature are for non-thread-safe libraries.

Node Type	# Cores	# Links	No. of MPI Tasks/Node	Latency in μs
1.9 GHz p5-575	16	1	1	4.67
		4	2	4.86
		4	4	4.91
		4	8	5.17
		4	16	5.34
2.5 GHz JS21	4	1	1	8.08
		1	2	8.10
		1	4	8.71
2.7 GHz JS21	2	1	1	7.87
		1	2	8.02

Table 1 - MPI Latency measurements

For the 1.9 GHz 16-core p5-575 nodes with four links, PingPong latency with 16 MPI tasks on each node is 5.34 μ s. In this case, each link is effectively shared by four MPI tasks. This is an indication of the fact that a single adapter is able to manage multiple communication streams very efficiently.

MPI Bandwidth

Tables 2, 3a, 3b, and 4 show the MPI bandwidth results for several message sizes. Large, 16 MB pages were used for these measurements. The distinction between PingPong and unidirectional bandwidth is that in a PingPong measurement, each task alternately sends and receives data of the same size. The time is measured over a send/receive pair on a node, and divided by two. In a unidirectional measurement, some tasks only send and some tasks only receive data. The time is measured over a loop of sends or receives and divided by the loop count. For unidirectional measurements, internal buffering of the message by the MPI library may artificially reduce the measured time and hence inflate the bandwidth for small messages. During a measurement of exchange bandwidth, each task has non-blocking sends and non-blocking receives outstanding at the same time. The exchange bandwidth is higher than the unidirectional

bandwidth because, in the case of exchange, data can flow through the adapter in two directions, rather than just one.

Node Type	No. of Cores	No. of Links	Tasks per Node	MPI PingPong Bandwidth (MB/sec)		
				Size=8K	Size=128K	Size=2048K
p5-575 1.9 GHz	16	1	1	410	771	891
JS21 2.5 GHz	4	1	1	273	643	828
JS21 2.7 GHz	2	1	1	278	641	824

Table 2 - MPI PingPong bandwidth for selected message sizes

Node Type	No. of Cores	No. of Links	Tasks per Node	MPI Unidirectional Bandwidth (MB/sec)		
				Size=8K	Size=128K	Size=2048K
p5-575 1.9 GHz	16	1	1	786	753	880
			2	883	925	940
			4	873	931	941
			8	875	931	941
p5-575 1.9 GHz	16	2	1	796	758	889
			2	1368	1327	1510
			4	1734	1780	1859
			8	1634	1787	1772
p5-575 1.9 GHz	16	4	1	796	757	889
			2	1594	1513	1778
			4	2700	2658	3030
			8	3494	3600	3753

Table 3a - MPI Unidirectional bandwidth for selected message sizes

Note that unidirectional bandwidth for 2 MB messages with two MPI tasks per link scales almost linearly in going from one to two to four links on the 1.9 GHz p5-575.

Node Type	No. of Cores	No. of Links	Tasks per Node	MPI Unidirectional Bandwidth (MB/sec)		
				Size=8K	Size=128K	Size=2048K
JS21 2.5GHz	4	1	1	710	637	843
			2	747	833	921
			4	785	942	953
JS21 2.7 GHz	2	1	1	708	631	833
			2	748	826	908

Table 3b - MPI Unidirectional bandwidth for selected message sizes

Node Type	No. of Cores	No. of Links	Tasks per Node	MPI Exchange Bandwidth (MB/sec)		
				Size=8K	Size=128K	Size=2048K
p5-575 1.9 GHz	16	1	1	782	1360	1582
			2	1435	1818	1844
			4	1566	1841	1873
			8	1590	1841	1873
p5-575 1.9 GHz	16	2	1	775	1365	1621
			2	1417	2238	2487
			4	2403	2866	3045
			8	2908	3080	3131
p5-575 1.9 GHz	16	4	1	776	1365	1621
			2	1536	2726	3270
			4	2822	4474	4975
			8	4827	5834	6190
JS21 2.5 GHz	4	1	1	486	862	1007
			2	789	1122	1185
			4	1132	1178	1133
JS21 2.7 GHz	8	1	1	496	872	998
			2	790	1120	1135

Table 4 - MPI Exchange bandwidth for selected message sizes

Note that exchange bandwidth for 2 MB messages with two MPI tasks per link does not double in going from one link to two links (on the same adapter) on the 1.9 GHz p5-575. In this case, the dual link exchange bandwidth is limited by the adapter. However the exchange bandwidth nearly doubles in going from two links on one adapter to four links on two adapters on the 1.9 GHz p5-575.

MPI bandwidths on the 2.5 GHz JS21 and 2.7 GHz JS21 are similar because these machines use the same memory subsystem.

Application Benchmarks

NAS Parallel Benchmarks

The NAS Parallel Benchmark suite is a set of seven computational kernels to test scalability of parallel systems using MPI. Each kernel can be built in one of four Classes, using increasingly larger amounts of memory. For the data presented in Table 7, the Class C kernels were built and tested. This data is for up to four IB links per system. The data presented is the “Time in seconds” as listed by each kernel. All runs were done using 16 MB pages. The p5-575 test configuration consisted of 16-core nodes running at 1.9 GHz and booted in ST mode. The MASS mathematical function library was used, and

the programs were compiled with the IBM XLF compiler using `-O3` optimization, `-qarch=pwr5` and `-qtune=pwr5`. For the runs on JS21 clusters, the compiler options were changed to `-qarch=ppc970` and `-qtune=ppc970`. The JS21 clusters had one switch module per chassis. Note that BT and SP require the number of tasks to be a perfect square, so for these two codes 36 tasks were used instead of 32 and 121 tasks were used instead of 128.

System	# Links	# Tasks	# Nodes	Time in seconds							
				BT	CG	EP	FT	IS	LU	MG	SP
p5-575 1.9 GHz	1	32	2	54.3	16.5	14.7	25.5	2.11	50.1	4.37	59.4
	2	32	2	53.6	15.2	14.7	21.1	1.59	50.2	4.22	58.1
	4	32	2	53.1	14.8	14.7	17.3	1.19	50.2	4.13	57.1
	1	64	4	33.6	12.7	7.28	15.8	1.95	25.6	2.26	34.5
	2	64	4	32.8	10.6	7.32	12.5	1.35	25.3	2.09	33.1
	4	64	4	32.3	9.89	7.29	9.55	0.85	25.3	1.97	32.0
	1	128	8	19.1	8.11	3.67	8.64	1.08	14.7	1.50	20.8
	2	128	8	18.1	6.73	3.66	6.70	0.75	14.3	1.31	18.9
	4	128	8	17.4	6.19	3.67	5.02	0.48	14.3	1.19	17.6
JS21 2.5 GHz	1	32	8	106.3	31.5	11.5	35.7	3.10	79.3	17.6	219.6
	1	64	16	60.1	20.4	5.81	18.7	2.06	39.1	5.73	119.6
	1	128	32	32.7	13.1	2.93	10.7	1.13	23.1	3.21	51.6
JS21 2.7 GHz	1	32	16	78.4	23.8	10.6	22.3	1.88	60.0	10.1	130.4

Table 7 – NAS Class C Parallel Benchmark run times

HPC Challenge Benchmark

The HPC Challenge Benchmark is a relatively recent, comprehensive, single application that attempts to measure various characteristics of a parallel system.

All runs were completed using 16MB pages. Tables 8 and 9 show performance data for a single port (two link) and double port (four link) IB 4X adapter respectively. The source files were compiled with the IBM XLF compiler using `-O3` optimization and tuned for POWER5 architecture with `-qtune=pwr5` `-qarch=pwr5` compiler flags. The object code was linked with the IBM ESSL library. The number of computational threads in the job was equal the number of MPI tasks i.e. OpenMP threads were not used. The test environment consisted of eight p5-575 16-core nodes running at 1.9 GHz.

All nodes in the cluster were installed with 32 GB of memory and booted in ST mode. The HPL problem size, N, was set to 10000 for a 128 task job and 78000 for a 64 task job. The matrix block size, NB, was 224 in both cases.

N	PG	HPL	PTRANS	*DGEMM	*GUPS	GUPS	*STREAM	*FFT	FFT	NRL	NRBW	RRL	RRBW
4	16M	0.40	5.04	7.13	.0144	0.018	4.71	0.62	11.7	5.05	0.82	7.30	0.095
8	16M	0.77	8.50	7.27	.0146	0.030	4.84	0.59	22.4	5.19	0.80	8.00	0.081

Table 8: HPCC Performance on p575 – One IB Adapter, two links, 16 tasks/node

N	PG	HPL	PTRANS	*DGEMM	*GUPS	GUPS	*STREAM	*FFT	FFT	NRL	NRBW	RRL	RRBW
4	16M	0.404	8.86	7.13	.0147	0.015	5.00	0.61	16.6	5.10	1.10	7.02	0.101
8	16M	0.783	24.8	719	.0186	0.029	4.72	0.62	22.9	5.10	1.08	7.17	0.152

Table 9: HPCC Performance on p5-575 – Two IB Adapters, four links, 16 tasks/node

N	PG	HPL	PTRANS	*DGEMM	*GUPS	GUPS	*STREAM	*FFT	FFT	NRL	NRBW	RRL	RRBW
16	16M	0.442	4.91	8.40	.0043	0.015	0.81	0.51	6.55	7.10	.282	13.5	0.118
32	16M	0.851	7.25	8.10	.0043	0.028	1.00	0.49	11.5	6.98	.283	14.2	0.091

Table 10: HPCC Performance on JS21 2.5 GHz – One link, one switch module, four tasks/node

- N -- Number of Nodes
- PG -- Page Size (bytes)
- HPL -- High Performance Linpack (TFLOPS)
- PTRANS -- PTRANS (GB/s)
- *FFT -- Star FFT (GFLOPS)
- FFT -- FFT (GFLOPS)
- *DGEMM -- Star DGEMM (GFLOPS)
- *STREAM -- Star STREAM (GB/s)
- *GUPS -- Star GUPS (Giga Updates/s)
- GUPS -- GUPS (Giga Updates/s)
- NRL -- Naturally Ordered Ring Latency (us)
- NRBW -- Naturally Ordered Ring BW (GB/s)
- RRL -- Random Ordered Ring Latency (us)
- RRBW -- Random Ordered Ring BW (GB/s)

Conclusion

The IBM HPC stack supporting InfiniBand on AIX on POWER augments the previously available solution for AIX 5L™ with the High Performance Switch proprietary network. It brings a scalable, mature communication infrastructure to more application areas.

More Information

“IBM High Performance Computing for Linux on POWER Using InfiniBand” –

http://www-03.ibm.com/systems/p/library/wp_linux_lit.html

“Introduction to InfiniBand for IBM eServer™ pSeries® Servers (Redpaper)” –

<http://www.redbooks.ibm.com/abstracts/sq247351.html>

“IBM BladeCenter JS21 Technical Overview and Introduction” –

<http://www.redbooks.ibm.com/abstracts/redp4130.html>

“System p Clustering systems using InfiniBand hardware” –

<http://publib.boulder.ibm.com/infocenter/eserver/v1r3s/topic/iphau/infinbdpdf.pdf>

Appendix A. Description of Measured Environments

MPI latency measurements were carried out between a pair of the following System p5 nodes:

- p5-575 16-core running at 1.9 GHz, bus/adaptor ratio of 3:1
- IBM BladeCenter JS21 2-core running at 2.7 GHz
- IBM BladeCenter JS21 4-core running at 2.5 GHz

For the p5-575 16-core nodes there were two IBM GX dual-port 4X IB adapters installed with each port connected to a port in a different Cisco SFS 7000P switch. Environments that use only one link (port) are described as *sn_single*. Environments that use two or four links are *sn_all*. Only *sn_single* runs are possible on JS21 clusters. Latency measurements were made between nodes requiring the fewest number of switch hops. Each node had AIX V5.3, with device driver level 61 (see Appendix D). The MPI stack was Parallel Environment for AIX on POWER, Version 4.3. InfiniBand resources were allocated by LoadLeveler. MPI latency and bandwidth programs were compiled in 64-bit mode.

AIX used 16 MB pages for user data. MCM memory affinity was set by exporting MEMORY_AFFINITY=MCM environment variable.

IBM's MPI library implementation provides a number of tunables (environment variables) that can be controlled by the user for each application. Depending on the application's communication pattern, the use of these environment variables may impact performance.

Table A1 contains the MPI variable settings that were used to measure the MPI performance in this white paper:

Parameter	Setting
MP_EUILIB	us
MP_EUIDEVICE	sn_single (use links from one plane) sn_all (use links from all available planes)
MP_SINGLE_THREAD	Yes
MP_EAGER_LIMIT	65536
MP_CSS_INTERRUPT	No
MP_POLLING_INTERVAL	2000000000

Table A1: MPI environment variable settings

CPUs in System p5 servers are organized in modules called DCMs having one or two physical processor cores (the p5-575) or QCMs (the p5-550Q Express) having four processor cores. Each DCM or QCM has a number of memory banks local to it with shortest access latency and highest bandwidth compared to other memory banks. An adapter is connected to a GX+ bus attached to one of the DCMs (QCMs). Communication latency is minimal when CPU and GX+ bus are on the same DCM (QCM). To tell the OS that a process is to run on a certain DCM (QCM) one may use binding.

When running with MEMORY_AFFINITY=MCM, it is recommended to bind the parent process, for example a helper script, which invokes MPI program as this lets the OS allocate memory on a given MCM (QCM) before the process is scheduled to run. To accomplish binding in a predictive manner one has to know the rank of a task to be run on a node. For interactive jobs a host list defines task rank to node mapping: rank increments consecutively for a task in the next line starting from zero. The MP_CHILD environment variable provides task rank within the helper script before MPI program is executed. This information is sufficient for binding tasks to CPUs in a predictable manner. In order to remove the need for a cumbersome helper script, IBM has recently enhanced the functionality MP_TASK_AFFINITY variable in MPI/POE.

Another reason to use process binding on POWER5 processor-based systems is to prevent thread migration between DCMs (QCMs) during the run or to prevent two tasks from running on the same core in simultaneous multithreading (SMT) mode when the number of tasks is equal to the number of cores. We also used process binding when

running on JS21 systems to prevent processes from migrating between CPUs. Note that there are no memory affinity issues on JS21, since each blade has only a single memory domain. Also note that SMT is not supported on JS21.

The OS deals with logical POWER5 CPUs rather than physical ones. Each physical POWER5 CPU (core) is associated with one or two logical CPUs known to the OS a process could be bound to. Single logical CPU per core is called ST (Single Threaded) mode and two logical CPUs per core is called SMT (Simultaneous Multithreading) mode.

On AIX V5.3 operating systems, the root user can disable or enable SMT by using the `smtctl` command which can be done either dynamically on a running system or with node reboot. SMT mode is enabled by default on POWER5 systems, but SMT is not supported on JS21 systems .

SMT mode on POWER5 has interesting characteristics to consider for a parallel program. In many cases it may boost performance. Applications, like LINPACK, known to consume a large volume of so called hardware rename registers are best positioned to run in ST mode. Applications that are memory latency bound benefit from SMT mode. Posix threads or OpenMP should be used to exploit SMT feature to make total number of runnable threads equal to the number of logical CPUs.

Appendix B. MPI tuning variables

The following are descriptions on what the parameters used to tune for peak performance mean. Additional MPI parameters and their usage can be found in “Parallel Environment for AIX Operation and Use”.

MP_EUILIB=us: (This variable tells MPI to use IBM’s “user-space” protocol. `MP_EUILIB` can also be set to “ip” indicating that the MPI application should use the UDP/IP transport protocol.

MP_EUIDEVICE=sn_single: This variable specifies that each MPI task should use only a single adapter. LoadLeveler will assign tasks to adapters in a round robin manner. If a job has at least as many MPI tasks per node as there are adapters, all adapters on the node will be used. A setting of **MP_EUIDEVICE=sn_all** tells the scheduler to allocate adapter resources on each of the adapters on the node for each task of the parallel job.

MP_SINGLE_THREAD=yes: This variable tells MPI that each MPI task will make MPI calls from one and only one thread, even if the task spawns multiple threads during its execution. Setting this parameter to “yes” provides a hint to the transport protocol underneath that it can make use of smarter and more efficient locking mechanisms available in the Power Architecture™ and avoid paying the penalty of the heavyweight `pthread_mutex` locks. The `pthread_mutex` locks are necessarily heavyweight since they

are designed for the general case and have to handle issues like fairness in lock access and related queuing to enforce it. With this setting the transport protocol can use the hardware lock primitives directly without having to worry about fairness issues among the various user threads in the MPI process/task (since the user has indicated that there is only one user thread). It should be noted that the MPI implementation itself is multi-threaded and spawns threads to handle packet arrival interrupts and to drive retransmission of unacknowledged packets.

MP_EAGER_LIMIT=65536: This variable tells the MPI transport protocol to use the “eager” mode for message less than or equal to 65536 bytes. Under the “eager” mode, the sender will send the message without worrying about the receive actually being posted on the target. For messages shorter than the EAGER_LIMIT, the MPI transport does not send a rendezvous message to the target or wait for an O.K. from the receiver before sending the data. Hence the messages that are smaller than the EAGER_LIMIT will typically be faster if the corresponding mpi_recv has already been posted. If not the transport will incur an extra copy cost on the target as data will be staged through the early arrival buffers.

MP_BUFFER_MEM=64M: This allows the user to define the amount of space the transport should reserve for the early arrival buffers.

MP_CSS_INTERRUPT=no: This variable allows the users to control packet arrival interrupts. Setting this variable to ‘no’ implies that the application must be run in polling mode. This setting is appropriate for applications that have mostly synchronous communication. For applications with an asynchronous communication pattern it is more appropriate to set this variable to yes.

MP_POLLING_INTERVAL=200000000: This environment variable sets the number of iterations before the timer thread is woken up. Large number reduces interference of the timer thread which reduces latency of MPI collectives with short messages.

Appendix C. Determining Device Driver Level

To determine the device driver level, issue the following command at the (root user) prompt:

```
# lspp -l | grep -i hca
```

should return something like:

```
devices.chrp.IBM.lhca.rte
```

```
5.3.0.61 COMMITTED InfinBand Logical HCA Runtime
```

which, in this case, is level 61.



© IBM Corporation 2008

IBM Corporation
Marketing Communications
Systems Group
Route 100
Somers, New York 10589

Produced in the United States of America
October 2007
All Rights Reserved

This document was developed for products and/or services offered in the United States. IBM may not offer the products, features, or services discussed in this document in other countries.

The information may be subject to change without notice. Consult your local IBM business contact for information on the products, features and services available in your area.

All statements regarding IBM's future directions and intent are subject to change or withdrawal without notice and represent goals and objectives only.

IBM, the IBM logo, AIX, AIX 5L, eServer, General Parallel File System, GPFS, LoadLeveler, Power Architecture, POWER, POWER5, PowerPC, pSeries, System p, System p5, Tivoli are trademarks or registered trademarks of International Business Machines Corporation in the United States or other countries or both. A full list of U.S. trademarks owned by IBM may be found at <http://www.ibm.com/legal/copytrade.shtml>.

The Power Architecture and Power.org wordmarks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org.

Other company, product, and service names may be trademarks or service marks of others.

IBM hardware products are manufactured from new parts, or new and used parts. Regardless, our warranty terms apply.

Copying or downloading the images contained in this document is expressly prohibited without the written consent of IBM.

This equipment is subject to FCC rules. It will comply with the appropriate FCC rules before final delivery to the buyer.

Information concerning non-IBM products was obtained from the suppliers of these products or other public sources. Questions on the capabilities of the non-IBM products should be addressed with the suppliers.

This paper is intended to provide information regarding IBM's HPC solution for AIX on POWER using InfiniBand. It discusses findings based on configurations that were created and tested under laboratory conditions. These findings may not be realized in all customer environments, and implementation in such environments may require additional steps, configurations, and performance analysis. The information herein is provided "AS IS" with no warranties, express or implied. This information does not constitute a specification or form part of the warranty for any IBM or Cisco products.

The IBM home page on the Internet can be found at <http://www.ibm.com>.

The Power Systems home page on the Internet can be found at <http://www.ibm.com/systems/p>.

PSW03041-USEN-01