Memory Scalability Evaluation of the Next-Generation Intel Bensley Platform with InfiniBand *

Matthew J. Koop Wei Huang Abhinav Vishnu Dhabaleswar K. Panda

Network-Based Computing Laboratory
Department of Computer Science and Engineering
The Ohio State University
{koop, huanwei, vishnu, panda}@cse.ohio-state.edu

Abstract

As multi-core systems gain popularity for their increased computing power at low-cost, the rest of the architecture must be kept in balance, such as the memory subsystem. Many existing memory subsystems can suffer from scalability issues and show memory performance degradation with more than one process running. To address these scalability issues, Fully-Buffered DIMMs have recently been introduced. In this paper we present an initial performance evaluation of the next-generation multi-core Intel platform by evaluating the FB-DIMM-based memory subsystem and the associated InfiniBand performance. To the best of our knowledge this is the first such study of Intel multi-core platforms with multi-rail InfiniBand DDR configurations. We provide an evaluation of the current-generation Intel Lindenhurst platform as a reference point. We find that the Intel Bensley platform can provide memory scalability to support memory accesses by multiple processes on the same machine as well as drastically improved inter-node throughput over InfiniBand. On the Bensley platform we observe a 1.85 times increase in aggregate write bandwidth over the Lindenhurst platform. For inter-node MPI-level benchmarks we show bi-directional bandwidth of over 4.55 GB/sec for the Bensley platform using 2 DDR InfiniBand Host Channel Adapters (HCAs), an improvement of 77% over the current generation Lindenhurst platform. The Bensley system is also able to achieve a throughput of 3.12 million MPI messages/sec in the above configuration.

1. Introduction

Over the past few years high-performance clusters of commodity hardware connected through high speed interconnects have largely replaced those with proprietary hardware. At this same time, computing power has increased rapidly, with multi-way SMP systems and the recent introduction of multi-core systems. As the number of computation processes on a single machine has increased, the memory requirements have likewise increased. Moreover, for scalable computation in cluster environments, high-speed interconnects such as InfiniBand [1], are emerging to take advantage of recent technology advancements, like PCI-Express [4], to achieve low latency of less than 2μs and high bandwidth of 20 Gbits/sec. This added network capability creates an additional burden on the memory subsystem, especially when multiple processes on a compute node are fully utilizing the network. Current-generation memory subsystems often face scalability issues to meet this increasing need for memory performance. Both increased memory capacity and throughput are required to scale for multi-core systems, however, traditional memory controllers require each additional memory module to be electrically connected to a shared bus, thus creating a tradeoff between speed and capacity [2].

To address these scalability issues and reduce the need for this tradeoff, the Fully-Buffered Dual Inline Memory Module (FB-DIMM) standard [3] has been introduced as a possible solution. Support for FB-DIMM is included in the next-generation multi-core platform from Intel, codenamed "Bensley." In this paper, we carry out a detailed evaluation of the memory subsystem and the associated impact on network performance with InfiniBand. We compare both the next-generation Bensley platform and the current-generation Intel Lindenhurst platform with these metrics. We first evaluate the impact of different FB-DIMM memory configurations using a varying number of channels and DIMMs. We also evaluate the memory subsystem using multiple processes performing memory intensive operations on a single system to emulate loaded conditions. We also include network-level communication metrics to evaluate the impact of the memory subsystem on InfiniBand performance. Our results show a significant advantage in scal-

*This research is supported in part by Department of Energy's grant #DE-FC02-01ER25506, National Science Foundation's grants #CNS-0403342 and #CNS-0509452; grants from Intel, Mellanox, Cisco, Sun Microsystems, and Linux Networx; Equipment donations from Intel, Mellanox, AMD, Advanced Clustering, IBM, Apple, Appro, Microway, PathScale, Silverstorm and Sun Microsystems.

able throughput and capacity in all measures with the Bensley platform. On the Bensley platform we observe a 1.85 times increase in aggregate write bandwidth over the Lindenhurst platform. Results using dual InfiniBand DDR show the Bensley platform is able to achieve 2.80 GB/sec uni-directional bandwidth, 4.55 GB/sec bi-directional bandwidth, and a MPI message rate of up to 3.12 million/sec. The Lindenhurst platform by comparison reached only 2.57 GB/sec for both uni-directional and bi-directional bandwidth and 2.07 million MPI messages/sec.

The remaining part of the paper is organized as follows: In Section 2, we provide a brief overview of memory subsystem design and the Bensley platform and an overview of the InfiniBand Architecture. We will briefly describe the experimental setup in Section 3. Our performance evaluation of the memory subsystem and the effect of different memory configurations is presented in Section 4. In Section 5 we evaluate the effect of the memory subsystem on inter-node communication. Finally, Section 6 finishes with conclusions and future work. To the best of our knowledge, this is the first extensive evaluation of FB-DIMM in an InfiniBand cluster environment.

2. Background

In this section we present the background of two areas. We start with a brief overview of the Intel Bensley platform with Fully-Buffered DIMMs. This is followed by a brief introduction to InfiniBand.

2.1. Intel Bensley Platform with Fully-Buffered DIMMs

As multi-core processors are becoming more prevalent, the demand for additional memory capacity has increased. The current design of DDR and DDR2 memory controllers use a shared-bus, often referred to as a "stub bus," which has an inherent limitation that prevents increasing capacity at the same time as speed. As a shared medium, the bus becomes a significant problem as additional capacity is needed since each additional module must be electrically connected to the memory controller. In existing designs a tradeoff is required between increasing speed and increasing capacity [2].

Fully-Buffered DIMMs (FB-DIMMs) [3] are part of an emerging memory standard to avoid this tradeoff of speed and capacity. By using serial point-to-point lanes, the shared bus can be removed. In this design, each memory module contains a buffer that allows communication to pass through a serial channel to other DIMMs on the same channel. Using this design, the memory controller only needs to write to the first buffer, not directly to the destination memory module. As these accesses can be buffered, multiple write requests can be proceed simultaneously. This serial architecture also allows a significantly lower pin count, which in turn allows additional memory channels and faster transmission rates. The FB-DIMM specifications allow for at least 6 channels, each with up to 8 slots each for a total capacity of 192 GB. The design of the FB-DIMM memory modules is general enough to support new memory module speeds such as DDR3, although the current designs use DDR2. The Intel Bensley platform is the next-generation Intel server platform that encompasses the FB-DIMM memory controller, as provided by the Blackford chipset, Intel I/O Acceleration Technology, Intel Virtualization Technology, and the dual-core "Dempsey" processor. For the purposes of this paper we will focus on the memory controller performance.

2.2. Overview of InfiniBand Architecture

The InfiniBand Architecture specifies a high-speed switched network fabric for interconnecting processing nodes and I/O nodes. InfiniBand has grown in popularity in system-area networks due to its high-performance and open standard. It provides many advanced features such as Remote Direct Memory Access (RDMA), atomic operations, and multicast. InfiniBand Host Channel Adapters (HCAs) are generally available in two 4x speeds, Single-Data Rate (SDR) with a data rate of 8Gbits/sec and Double-Data Rate (DDR) with a data rate of 16Gbits/sec. For this paper we will be using two DDR HCAs to stress the memory subsystem.

3. Experimental Testbed

Our experimental testbed consists of two InfiniBand clusters. Our Bensley cluster contains SuperMicro X7DB8+ servers, each with two 8x PCI-Express slots. Each node contains two 3.2 GHz Intel Xeon "Dempsey" dual-core processors and a FB-DIMM-based main memory. This hardware contains 4 memory channels, each with 4 FB-DIMM slots. For comparison, we use a Lindenhurst cluster that consists of SuperMicro X6DH8-G2 servers, each with two x8 PCI-Express slots. Each node in the cluster contains two single-core Intel Xeon CPUs running at 3.4 GHz and with up to 4GB of DDR2 memory. In both clusters, all machines contain two dual-port Mellanox MT25208 InfiniBand DDR HCAs and are connected through a Mellanox MTS-2400 DDR InfiniBand switch for MPI-level performance results. Linux with kernel 2.6.16.2 is used on both clusters with InfiniBand support provided through the 1.0 branch of OpenIB/Gen2 stack [9]. The Intel Compiler v9.0 is used for compilation of all benchmarks.

For MPI-level evaluation we use MVAPICH[6, 8], a popular MPI-1 implementation over InfiniBand designed and developed at the Ohio State University. To fully exploit the throughput of the memory subsystem, we use the "multi-rail" feature of MVAPICH [5], which allows data transfer on multiple HCAs and ports.

4. Performance Evaluation of the Memory Subsystem

In this section we evaluate the memory subsystem performance of the next-generation Intel Bensley platform. We

also carry out performance comparisons between the Bensley platform and the current-generation Intel Lindenhurst platform.

We evaluate the memory subsystem from two main aspects:

- Impact of different memory configurations on the Bensley platform: Various memory channel configurations are possible given a set number of memory DIMMs. As an example, DIMMs can be distributed evenly across different subchannels in a round robin assignment or block configuration, where all channels may not have equal number of DIMMs. In Section 4.1, we quantitatively evaluate the impact on memory access latency and throughput of distributing DIMMs among multiple channels and adding additional DIMMs to each channel.

- Comparison of Intel Bensley and Lindenhurst systems: In Section 4.2 we compare memory access latency and throughput. We also study the performance degredation when a machine is under a heavy load.

Our experiments are based on *lmbench* 3.0-a5 [7], an open-source benchmark suite designed to evaluate UNIX system performance metrics. We focus on the memory read latency and the memory bandwidth benchmarks (including read, write, and copy) provided by the test suite.

The memory read latency benchmark reports back-to-back load latency. This latency is the time required to fetch a single byte from memory. To evaluate different levels of the memory subsystem, such as the L1 cache, L2 cache, and the main memory, *lmbench* creates an array of a given size and iterates over it, fetching the bytes with a user-defined stride. An array of a small size completely fits into cache and shows cache-level performance, while larger array sizes (above 2MB) do not fit into cache and evaluate main memory latency. To ensure a cache miss on each access, we use a stride size of 512 bytes, a value larger than the cache line size.

The read memory bandwidth benchmark begins with allocating 16MB of memory, setting all bytes to zero, and reading the data as integer loads and adds. With the speed of modern processors, the dominant term amongst the load and integer addition is the memory load time. The write memory bandwidth benchmark functions similarly, writing to the memory as a series of integer stores.

In order to evaluate memory performance scalability with system load, we introduce the terms *unloaded* and *loaded memory performance*. Unloaded memory performance is directly reported by *lmbench*. Loaded memory performance evaluation is conducted with multiple processes running the benchmarks. In the case of loaded latency, a read memory throughput test is run at the same time as the memory latency benchmark. For loaded memory bandwidth, multiple processes run the same bandwidth benchmark simultaneously and the aggregated throughput is reported.

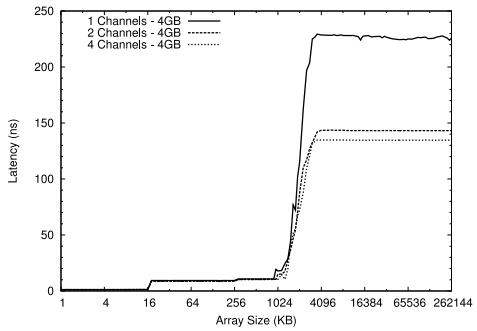


Figure 1. Main Memory Access Latency with Varying Numbers of Channels

4.1. Impact of FB-DIMM Memory Configurations

As mentioned earlier, our Bensley systems support 4 memory channels and up to 4 DIMMs per channel. In this section, we study the impact of different memory configurations on performance:

- Distributing DIMMs across multiple channels
- Adding more DIMMs per channel

4.1.1 Distributing DIMMs among Multiple Channels

In this section we fix the total amount of memory to 4GB (4 DIMMs of 1GB size) and populate the memory in the following configurations to study the memory access latency and bandwidth:

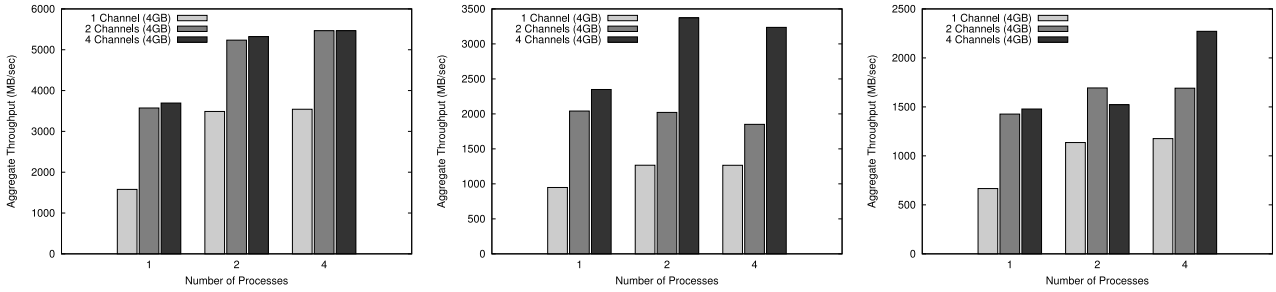
- 1 channel with 4 DIMMs
- 2 channels each with 2 DIMMs
- 4 channels each with 1 DIMM

Figure 1 shows the latency as reported by *lmbench* with increasing array sizes. For smaller arrays of less than 2MB size show the performance of L1 and L2 cache hits, since the entire array fits into cache. When the data has to be fetched from main memory, the latency reduces as the number of channels is increased, because the memory controller can fetch memory in parallel from the different channels through memory interleaving. The main memory byte latency for a single channel is 57% higher than that of two channels. The difference in latency between two channels and four channels is a more modest 6% improvement.

The performance benefit of populating DIMMs into multiple channels is also observed in Figure 2, which shows the memory bandwidth for read, write, and copy operations all increase with more channels being used. We also show the *loaded memory performance*, the aggregated throughput with 2 and 4 processes running the benchmarks simultaneously. We see similar trends except for the 2 process case, where the memory copy bandwidth drops with 4 channels. We are still investigating the reason for this drop.

4.1.2 Adding Additional DIMMs per Channel

In the previous section we have shown that using multiple memory channels can significantly increase memory subsystem performance. The Bensley systems, however, support up to 16 memory DIMMs, so multiple DIMMs per



(a) Aggregate Read Bandwidth (b) Aggregate Write Bandwidth (c) Aggregate Copy Bandwidth

Figure 2. Bensley Memory Throughput for Varying Numbers of Channels

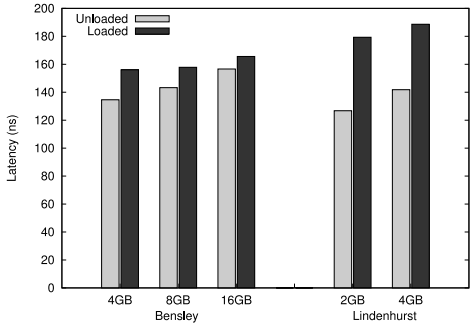


Figure 3. Bensley/Lindenhurst Access Latency Comparison

channel are required to reach the maximum memory capacity. In this section we study the impact of adding additional DIMMs per channel. We use 4 memory channels in all cases and increase the number of DIMMs per channel from 1 to 2 and 4, for a total of 4GB, 8GB and 16GB of total memory, respectively.

From our experimental results we observed that with increasing numbers of DIMMs per channel the latency increases for each additional set of DIMMs. The access latency when all 4 slots per channel are populated is approximately 22ns higher than in the case when only a single slot per channel is used and around 9 ns higher than utilizing 2 slots per channel. Thus, there is around a 7% overhead for adding 1 additional DIMMs per channel. This additional time is needed to reach the last DIMM in the channel, due to the serial connection.

We observe no significant differences by varying the number of DIMMs and evaluating the read, write, and copy throughput performance.

4.2. Performance Comparison of Bensley and Lindenhurst Memory Subsystem

In this section, we compare the memory subsystem performance of the Bensley FB-DIMM architecture with a traditional DDR2 memory controller on the Lindenhurst platform, under both *loaded* and *unloaded* cases.

4.2.1 Latency

The results of the memory read latency benchmark are reported in Figure 3. We show the loaded and unloaded memory performance for each of the platforms when equipped with increasing numbers of DIMMs. When the systems are unloaded, the memory read latency on the Lindenhurst platform with 2 DIMMs is 12ns lower than the 4 DIMM Bensley configuration. This latency, however, on Lindenhurst increases 15ns when an additional 2 DIMMs are added. In contrast, the Bensley memory latency increases only 9ns when 4 additional DIMMs are added, one to each channel.

To demonstrate the loaded memory latency, we run two processes, one performing a read memory throughput test to provide a load on the system and another to perform a read memory latency test. As shown in Figure 3, the memory read latencies increase for both the Bensley and the Lindenhurst platforms due to memory contention. The latency increases nearly 40% on average for Lindenhurst in comparison to a 10% increase on Bensley. This shows that the slightly higher unloaded FB-DIMM latency, created by serial access, is largely offset by the other features of the architecture, such as allowing accesses to proceed in parallel.

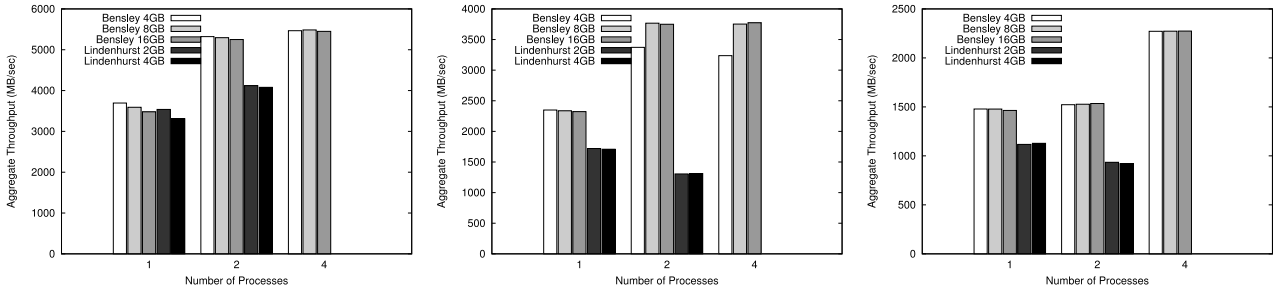
4.2.2 Memory Throughput

In this section, we evaluate the memory throughput of both the Lindenhurst and Bensley platforms with varying numbers of DIMMs. The read, write, and copy benchmarks are the same as described in the previous section.

Examining the results presented in Figure 4, we see that for unloaded case (1 process accessing the memory), the Bensley and Lindenhurst platforms give nearly the same read, write and copy bandwidth. This changes when 2 processes are running concurrently; the Bensley platform shows improved aggregated memory read bandwidth of 44% in contrast to a smaller 16% improvement on the Lindenhurst platform. On the Lindenhurst platform, the two process write and copy operations perform 25% and 19% worse, respectively, than the single process case The Bensley platform, however, scales to 4 concurrent processes without degradation of aggregate performance for both of these metrics.

5. Network Performance Evaluation

We present the evaluation of the Bensley and Lindenhurst platforms with InfiniBand in this section. For this evaluation we use MVAPICH, which supports multi-rail configurations and provides different scheduling policies for



(a) Aggregate Read Bandwidth (b) Aggregate Write Bandwidth (c) Aggregate Copy Bandwidth

Figure 4. Bensley/Lindenhurst Memory Throughput Comparison

data transfer. We perform various communication micro-benchmarks to evaluate the performance of the Bensley memory subsystem with InfiniBand. To stress the memory subsystem, we use two DDR InfiniBand HCAs per node.

To simulate various system loads we run 1, 2, and 4 process pairs for the microbenchmarks. In a *n* process pair benchmark, there are *n* processes on one node sending data to *n* processes on another node with a one-to-one mapping. For the single process pair case we use an even striping policy where small messages are sent with a round robin schedule and larger messages are striped across all available HCA ports. For two and four process cases, we use a process binding schedule with processes are assigned to HCA ports in a round-robin mapping.

In evaluating the results of the micro-benchmarks, it should be noted that InfiniBand requires communication buffers to be registered. Thus, for messages of size lesser than 8KB, MVAPICH uses a copy-based approach where the message is copied into a pre-registered buffer, making the memory copy bandwidth very important. Messages larger than 8K will use a zero-copy RDMA-based approach where the user buffer is registered and is more dependent on write memory bandwidth.

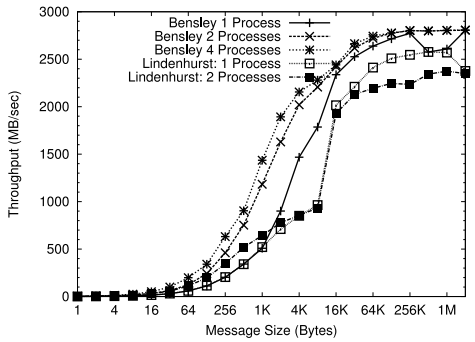


Figure 5. Uni-Directional Bandwidth

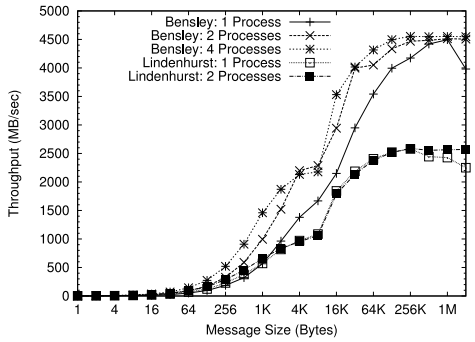


Figure 6. Bi-Directional Bandwidth

5.1. Uni-Directional Bandwidth

In the uni-directional bandwidth benchmark, we evaluate the bandwidth of each platform by varying the message size. This benchmark performs a throughput test between sender and receiver pairs where for each message size the sender sends a window of messages and waits for an acknowledgment from the receiver. This step is repeated for a few thousand iterations.

Figure 5 shows the aggregated bandwidth of multiple processes performing network communication to processes on another node. In the figure we present results for 1, 2 and 4 process pairs for Bensley systems and compare them with the 1 and 2 process pair results of the Lindenhurst systems. On the Lindenhurst systems, using two processes improves the aggregated throughput for small to medium-sized messages, however, the peak bandwidth for two process pairs is worse than that of the single process pair. This matches the drop in aggregate memory write bandwidth for the Lindenhurst platform as shown in Section 4.2. On the Bensley platform, the aggregated bandwidth increases consistently for small messages with an increasing number of communication pairs. Comparing with the Lindenhurst 2 process case, the aggregate bandwidth is nearly double for the 16K message size. The Bensley system shows further scalability in the 4 process pairs case, with a significant performance improvement over two process pairs. The 4 process case saturates the InfiniBand fabric at around 32K to around 2810 MB/s, an improvement of 28% over the peak bandwidth achieved by the Lindenhurst systems.

5.2. Bi-Directional Bandwidth

The bi-directional bandwidth benchmark is similar in design to the uni-directional bandwidth benchmark presented earlier. However, in this case, both sides of the communication send as well as receive data from each other. As done in the previous test, the above step is performed for a few thousand iterations.

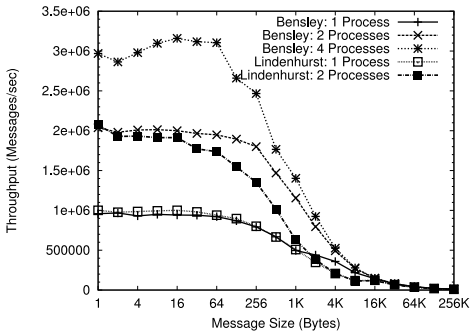


Figure 7. Messaging Rate

Figure 6 compares the performance of the systems using the bi-directional bandwidth test. We notice that for nearly all message sizes the bi-directional bandwidth achieved by the Bensley platform is much greater than the Lindenhurst platform. For the Lindenhurst platform, at a message size of 1KB the 2 process case is only 15% greater than the 1 process case. The Bensley platform by comparision increases performance by 75% from 1 to 2 process pairs and 45% from 2 to 4 process pairs. For peak bandwidth, we note the Lindenhust platform achieves 2565 MB/sec, only 100 MB/sec better than the uni-directional bandwidth peak bandwidth. The Bensley system, however, is able to reach over 4550 MB/sec, which is an improvement of 62% over the uni-directional benchmark result.

5.3. Messaging Rate

The messaging rate benchmark evaluates the throughput of each platform to send messages of varying sizes. This benchmark performs a throughput test similar to the unidirectional bandwidth benchmark, however, instead of reporting bandwidth it reports the number of messages transfered per second. It is important to assess the throughput in this manner as many applications make frequent use of small messages. For evaluating scalability, the benchmark allows more than one pair of processes to run simultaneously, with aggregated performance being reported.

Figure 7 compares the aggregated messaging rate capabilities of the Lindenhurst and Bensley platforms. For very small messages ($\leq$ 4 bytes), the Lindenhurst and Bensley platforms have a nearly identical peak message rate of 1 million messages/sec and 2 million messages/sec for one and two processes pairs, respectively. The performance difference of the two systems diverges for larger message sizes. At a message size of 512 bytes the two process result for the Lindenhurst system has dropped to a 52% increase over the single process case while the Bensley systems still show an improvement of 100%. We also note the scalability of the Bensley systems by increasing the number of process pairs to 4, which increases the aggregated messaging rate to 3 million packets/sec and converges to the level of the two process case only after 4K message size when the InfiniBand fabric is becoming saturated.

6. Conclusions and Future Work

In this paper, we have carried out a detailed evaluation of the memory subsystem of both the the next-generation Bensley platform and the current-generation Lindenhurst platform. We also evaluated the impact of the memory subsystem on network performance with InfiniBand. We first evaluated the impact of different FB-DIMM memory configurations using a varied number of channels and DIMMs. We then evaluated the memory subsystem using multiple processes performing memory intensive operations on a single system to emulate loaded conditions. Our evaluation completed with network-level communication metrics to evaluate the memory subsystem impact on InfiniBand performance with PCI-Express. Our results have shown a significant advantage in scalable throughput and capacity in all measures with the Bensley platform. On the Bensley platform, we observed a 1.85 times increase in aggregate write bandwidth as compared to the Lindenhurst platform. Inter-node results with 2 DDR HCAs have shown, the Bensley platform is able to achieve 2.80 GB/sec uni-directional bandwidth, 4.55 GB/sec bi-directional bandwidth, and a MPI message rate of up to 3.12 million/sec. The Lindenhurst platform by comparison reached only 2.57 GB/sec for both uni-directional and bi-directional bandwidth and 2.07 million MPI messages/sec.

In the future we plan to continue working on assessing the scalability of the Bensley architecture by profiling real-world applications on a larger cluster and observing the effects of contention in multi-core architectures.

References

[1] InfiniBand Trade Association. InfiniBand Architecture Specification, Volume 1, Release 1.0. http://www.infinibandta.com.

[2] Intel. Fully-Buffered DIMM Technology Moves Enterprise Platforms to the Next Level. http://www.intel.com/technology/magazine/computing/Fully-buffered-DIMM-0305.htm, March 2005.

[3] JEDEC Solid State Technology Association. Preliminary FB-DIMM Specification. http://www.jedec.org/.

[4] J. Liu, A. Mamidala, A. Vishnu, and D. K. Panda. Evaluating InfiniBand Performance with PCI Express. *IEEE Micro*, 25(1):20–29, 2005.

[5] J. Liu, A. Vishnu, and D. K. Panda. Building multirail infiniband clusters: Mpi-level design and performance evaluation. In *SC '04: Proceedings of the 2004 ACM/IEEE conference on Supercomputing*, page 33, 2004.

[6] J. Liu, J. Wu, S. P. Kini, P. Wyckoff, and D. K. Panda. High Performance RDMA-Based MPI Implementation over InfiniBand. In *17th Annual ACM International Conference on Supercomputing (ICS '03)*, June 2003.

[7] L. W. McVoy and C. Staelin. lmbench: Portable tools for performance analysis. In *USENIX Annual Technical Conference*, pages 279–294, 1996.

[8] Network-Based Computing Laboratory. MVAPICH: MPI for InfiniBand on VAPI/Gen2 Layer. http://nowlab.cse.ohio-state.edu/projects/mpi-iba.

[9] OpenFabrics Alliance. OpenFabrics. http://www.openfabrics.org/, April 2006.