Performance Analysis and Evaluation of Mellanox ConnectX InfiniBand Architecture with Multi-Core Platforms *

Sayantan Sur Matthew J. Koop Lei Chai Dhabaleswar K. Panda

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

Abstract

InfiniBand is an emerging networking technology that is gaining rapid acceptance in the HPC domain. Currently, several systems in the Top500 list use InfiniBand as their primary interconnect, with more being planned for near future. The fundamental architecture of the systems are undergoing a sea-change due to the advent of commodity multi-core computing. Due to the increase in the number of processes in each compute node, the network interface is expected to handle more communication traffic as compared to older dual or quad SMP systems. Thus, the network architecture should provide scalable performance as the number of processing cores increase.

ConnectX is the fourth generation InfiniBand adapter from Mellanox Technologies. Its novel architecture enhances the scalability and performance of InfiniBand on multi-core clusters. In this paper, we carry out an in-depth performance analysis of ConnectX architecture comparing it with the third generation InfiniHost III architecture on the Intel Bensley platform with Dual Clovertown processors. Our analysis reveals that the aggregate bandwidth for small and medium sized messages can be increased by a factor of 10 *as compared to the third generation InfiniHost III adapters. Similarly, RDMA-Write and RDMA-Read latencies for 1-byte messages can be reduced by a factor of* 6 *and* 3*, respectively, even when all cores are communicating simultaneously. Evaluation with communication kernel Halo reveals a performance benefit of a factor of* 2 *to* 5*. Finally, the performance of LAMMPS, a molecular dynamics simulator, is improved by 10% for the in.rhodo benchmark.*

*This research is supported in part by Department of Energy's grant #DE-FC02-06ER25749 and #DE-FC02-06ER25755; NSF grants #CNS-0403342 and #CCF-0702675; grants from Intel, Mellanox, Cisco, Sun Microsystems, and Linux Networx; Equipment donations from Intel, Mellanox, AMD, Advanced Clustering, IBM, Appro, Microway, PathScale, Silverstorm and Sun Microsystems.

1 Introduction

High-Performance computing has witnessed a tremendous growth and acceptance over the last decade primarily due to the availability of "Commodity Clusters". Commodity clusters leverage high volume off-the-shelf market products to provide very good cost-performance ratios. The performance of these clusters hinges upon the networking technology which interconnects the compute and I/O nodes. The InfiniBand Architecture [8] is an industry standard interconnection technology which aims to provide low-latency and high-bandwidth communication. Current generation InfiniBand products (InfiniHost III), from Mellanox [1] provide low latency (2μs) and high bandwidth (1400 MBps unidirectional).

Over the past several years, it has been shown that power consumption and heat dissipation issues restrict the clock frequencies in modern microprocessors. The only feasible way the computation power of a processor can keep on increasing as per Moore's Law [6], is by increasing the number of processing cores on the chip. Modern processors from Intel, AMD, Sun and IBM have multiple cores on a chip. Thus, the architecture of the compute nodes has been fundamentally altered. Modern commodity clusters have appeared which employ multi-core processors. The communication requirements of "multi-core nodes" (having multiple multi-core chips) are different from the more common two/four way SMP machines which were common in commodity clusters. One critical difference being that many processes can potentially communicate with remote processes using

the same network interface. The design of the network interface is thus crucial for achieving high and balanced performance across all the cores of a multi-core node.

The ConnectX Architecture [10] is a new InfiniBand network interface from Mellanox. It succeeds the InfiniHost III architecture which is one of the most widely deployed InfiniBand Host Channel Adapter (HCA). In this paper, we carry out an in-depth performance analysis of ConnectX architecture comparing it with the third generation InfiniHost III architecture on the Intel Bensley platform with Dual Clovertown processors. Our analysis reveals that the aggregate bandwidth for small and medium sized messages can be increased by a factor of 10 as compared to the third generation InfiniHost III adapters. Similarly, RDMA-Write and RDMA-Read latencies for 1-byte messages can be reduced by a factor of 6 and 3, respectively, even when all cores are communicating simultaneously. Evaluation with communication kernel Halo reveals a performance benefit of a factor of 2 to 5. Finally, the performance of LAMMPS, a molecular dynamics simulator, is improved by 10% for the `in.rhodo` benchmark.

The remaining part of the paper is organized as follows: In Section 2 we provide an overview of the new ConnectX architecture. We present a performance evaluation of this new architecture compared with the older generation InfiniHost III architecture in Section 3. In Section 4 we discuss related work in this area. Finally, we conclude the paper in Section 5.

2 Overview of the ConnectX Architecture

ConnectX is the fourth generation InfiniBand Host Channel Adapter (HCA) from Mellanox Technologies [1]. It uses PCI-Express v2.0, backwards compatible with v1.1, to connect to the host. It provides two network ports to connect to the network fabric. Each port can be independently configured to be used either as 4X InfiniBand or 10 Gigabit Ethernet. This independent configuration is made possible due to the layered architecture of the HCA. The architecture of the HCA is shown in Figure 1. In this paper, however, we do not evaluate the performance characteristics of the 10 Gigabit Ethernet mode; instead we focus completely on the InfiniBand implementation of ConnectX. To the best of our knowledge, the 10 GigE mode is not yet enabled by the ConnectX firmware and drivers.

The ConnectX architecture is designed to improve the processing rate of incoming packets. Compared to the previous InfiniHost III architecture, it has more advanced packet processing capabilities. In order to effectively use these capabilities, ConnectX has advanced scheduling engines which can assign processing duties

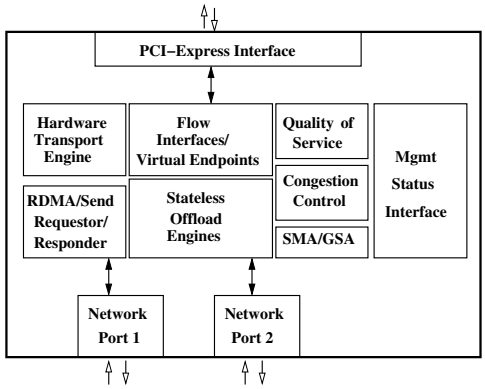


Figure 1. ConnectX HCA Architecture, courtesy Mellanox [10]

(like protocol processing, data integrity checks, etc.) to idle processing elements. The scheduling of packet processing is done directly in hardware, without firmware involvement in the critical path. These enhancements to the ConnectX architecture are expected to improve its performance on multi-core nodes when multiple processes are communicating at the same time, generating many simultaneous network messages.

The ConnectX architecture also features advanced processing capabilities for Work Queue Elements (WQEs). This capability is reflected in a reduced latency difference between RDMA Write and Send/Receive transport modes. The difference between the two modes on ConnectX is just 0.2μs, as opposed to 1.15μs on the earlier InfiniHost III architecture. This enhancement is expected to improve the performance of Shared Receive Queue (SRQ) which enables scalable buffer management in the Reliable Connected (RC) mode. SRQs are widely used in high-performance MPI implementations over InfiniBand [12, 5].

The ConnectX architecture even goes beyond the InfiniBand specification to enable an "Enhanced InfiniBand" mode which alleviates a major concern about memory scalability of Reliable Connected (RC) transport on very large scale multi-core clusters. This mode is called **S**calable **R**eliable **C**onnected transport (SRC). Normally, connection oriented transports require $O(n^2)$ connections (for the entire parallel application), where n is the number of processes. The advent of multi-core architecture has allowed many processes to be located inside one node. The SRC transport leverages this fact to reduce the total number of reliable connections required across different nodes. Using this SRC transport, it is sufficient to just have *one* connection between a set of processes on one node to another set of processes on a remote node. This is made possible due to a slight protocol change. In SRC, the sender side can specify the

destination SRQ in the packet. The receiver is then able to de-multiplex packets to the correct SRQ. In addition, the sending processes on one node can share the same send queue of the SRC. Thus, just one connection between two nodes suffices to connect all processes within them using a Reliable Connection. [1] ConnectX has a host of other features, and due to space constraints, we could not describe all of them. Readers are encouraged to peruse [10] for a list and usage of various features of ConnectX.

3 Experimental Evaluation

In this section we describe our experimental evaluation of the ConnectX architecture. Our experiments are primarily designed to determine how the new ConnectX architecture performs on multi-core platforms as compared to the older InfiniHost III architecture. We begin our evaluation with InfiniBand-level micro-benchmark study. We have re-designed the publicly available performance benchmarks "perftest" in the OpenFabrics stack [13] to work with multiple pairs of processes across different nodes. We study the performance of both RDMA-Write and RDMA-Read with increasing number of pairs of communicating processes on each node. Then we move on to evaluation with the communication kernels (Halo) [15] and a molecular dynamics simulator (LAMMPS) [14].

Evaluation Platform: Our evaluation cluster consists of a four-node Intel Bensley platform. We use dual Intel Clovertown 2.33GHz, quad-core processors. The nodes have 4GB of main memory (FBDIMMs). The platform is equipped with three x8 PCI-Express slots. In one slot we have the new Mellanox ConnectX cards which operate at DDR speed (20Gbps signalling rate), and in the other slot we have a InfiniHost III dual-port MT25218 card. This card also operates at DDR speed. Both the cards are connected to different, yet identical MTS2400, Mellanox 24-port InfiniBand switches. The InfiniHost III card has firmware version 5.2.0 and OpenFabrics 1.1 distribution. The ConnectX card (MT25408) has firmware version 2.0.139 and operates with new OpenFabrics drivers which are based on OpenFabrics 1.2 distribution. The kernel version used was 2.6.20-rc5 and the distribution is RedHat AS4(U4).

3.1 Multi-Pair Write Latency

In this test, we measure the RDMA-Write latency at the InfiniBand verbs level (Gen2 interface) as the number of communicating pairs between multi-core platforms increase. The OpenFabrics stack provides a single pair RDMA-Write latency benchmark, called "ib_rdma_lat". We re-designed this test to run on multiple processes by using the MPI interface. Each process starts up and uses MPI calls to synchronize and exchange connection information. To eliminate the possibility of the MPI transactions having any impact on the experiment, we used MPICH-1.2.7 [7] over a standard fast-ethernet network card. In multi-pair tests, it is very important to have very fine-grain synchronization between the pairs of processes in one node. This is to make sure that when each process issues the InfiniBand send, the other processes are sending at the same exact time; otherwise contention effects are not measured. To achieve this, we used System V shared memory. A leader process on a node creates the shared memory segment to which every process attaches. This shared region is then used to synchronize all the processes at a fine grain.

The results of the multi-pair write latency test are shown in Figures 2(a), 2(b) and 2(c). We observe from Figure 2(a) that the InfiniHost III architecture cannot provide very good latency as the number of communicating pairs on a node increases. This test does not involve a lot of Queue-Pairs (QPs). At maximum only eight are used and there is sufficient memory at the network interface for caching all eight QP contexts. Hence, the possibility of QP cache context misses is ruled out. The InfiniHost III adapter as such is capable of providing lower latencies when only one pair of processes is communicating, thus the basic processing of transport related activities should not be the bottleneck either. It appears to us that the reason for increased latencies for multiple pairs communicating could be due to either slow or serial processing of incoming WQEs (Work Queue Elements) by the InfiniHost III architecture.

Figure 2(b) on the other hand shows that the ConnectX architecture is able to provide almost the same latency for small messages for up to 8-pairs of communicating processes. The basic improvement in latency stems from the fact that in ConnectX architecture, it is possible to include the WQE inside the doorbell. The doorbell informs the network interface of incoming requests and is sent using PIO (Programmed Input/Output). For very small messages, the data is itself included inside the WQE, this technique is called "inlining". In summary, in the ConnectX architecture, the WQE includes the message data (for small messages), and is sent using PIO over to the network interface. Thus, in the critical path, there are no DMA operations for very small messages. This is in contrast to InfiniHost III, which supports "inline" messages, but the WQE itself has to be DMA'd from host memory after the doorbell request is received by the network interface.

[1] This feature is not enabled in the first generation firmware for ConnectX. Accordingly, we did not evaluate the performance and memory reduction of this mode in this paper.

As the number of pairs increase (from one to eight), we observe that the latency does not increase for the ConnectX architecture. This may be attributed to faster or more parallelized methods of handling incoming WQEs by the network interface. At 256 bytes, with 8-pairs communicating, we see a jump in the latency. For 128 bytes, the multi-pair latency is 1.86 μs, whereas for 256 bytes it is 5.87 μs. For 256 byte messages, although the message data is "inline", the size of the WQE exceeds the PIO limit, the doorbell does not include the WQE. Upon receiving the doorbell, the network interface has to simultaneously fetch eight WQEs using DMA. This limit for using PIO for WQEs can be configured according to the application behavior. Latency sensitive applications using mainly small messages may benefit from using PIO over DMA. We believe that the number of outstanding reads (which actually execute in parallel) on the PCI Express bus might be limited on the specific chip-set used in our experiments, leading to this jump. For this test, the PCI bus had 256 byte payload (configurable in the BIOS). When we changed this setting to 128 byte payload + coalescing (in the BIOS), we saw that the jump in latency for 256 bytes was muted, to 4.0 μs. Thus, we believe that this jump might be related to the specific chip-set used, and not with the ConnectX architecture itself.

Overall, the ConnectX architecture provides over 6 times improvement for messages below 256 bytes over the InfiniHost III architecture on multi-core nodes using 8-pairs communicating simultaneously.

3.2 Multi-Pair Read Latency

In this test we measure the impact on Read latency with increasing number of communicating partners on a multi-core platform. The basic test is the same as the write test as mentioned in Section 3.1. Only, instead of RDMA-Write operations, RDMA-Read operations are performed. The results of this experiment are shown in Figures 2(d), 2(e) and 2(f). We observe from Figure 2(d) that the Read latency on the InfiniHost III architecture is high (6μs). In case of ConnectX, it is much lower (1.8μs) as shown in Figure 2(e). As mentioned in the previous section, much of the latency improvement comes from the fact that WQEs are now included inside the doorbell, eliminating the need for initiating DMAs in the critical path (on the sender side). However, on the receiver side, one DMA is still required to fetch the user data from main memory. Thus, RDMA Reads are a bit more expensive than RDMA Writes (for small messages). With ConnectX, the Read latencies for two-pairs is nearly the same as that for one-pair. However, the latencies do increase with the number of pairs. This may be attributed to the fact that the chip-set may not execute in parallel as many reads on the bus as issued by ConnectX, as mentioned in the previous section. We analyzed the ConnectX firmware configuration (file: `fw-25408-rel.mlx`, parameter: `pcie_max_outstanding_read_requests`), and found that by default, the firmware issues up to 16 concurrent outstanding reads on the PCIe bus. It is left up to the specific chip-set to execute as many in parallel as possible, which in our system seems to be limited to lower than eight.

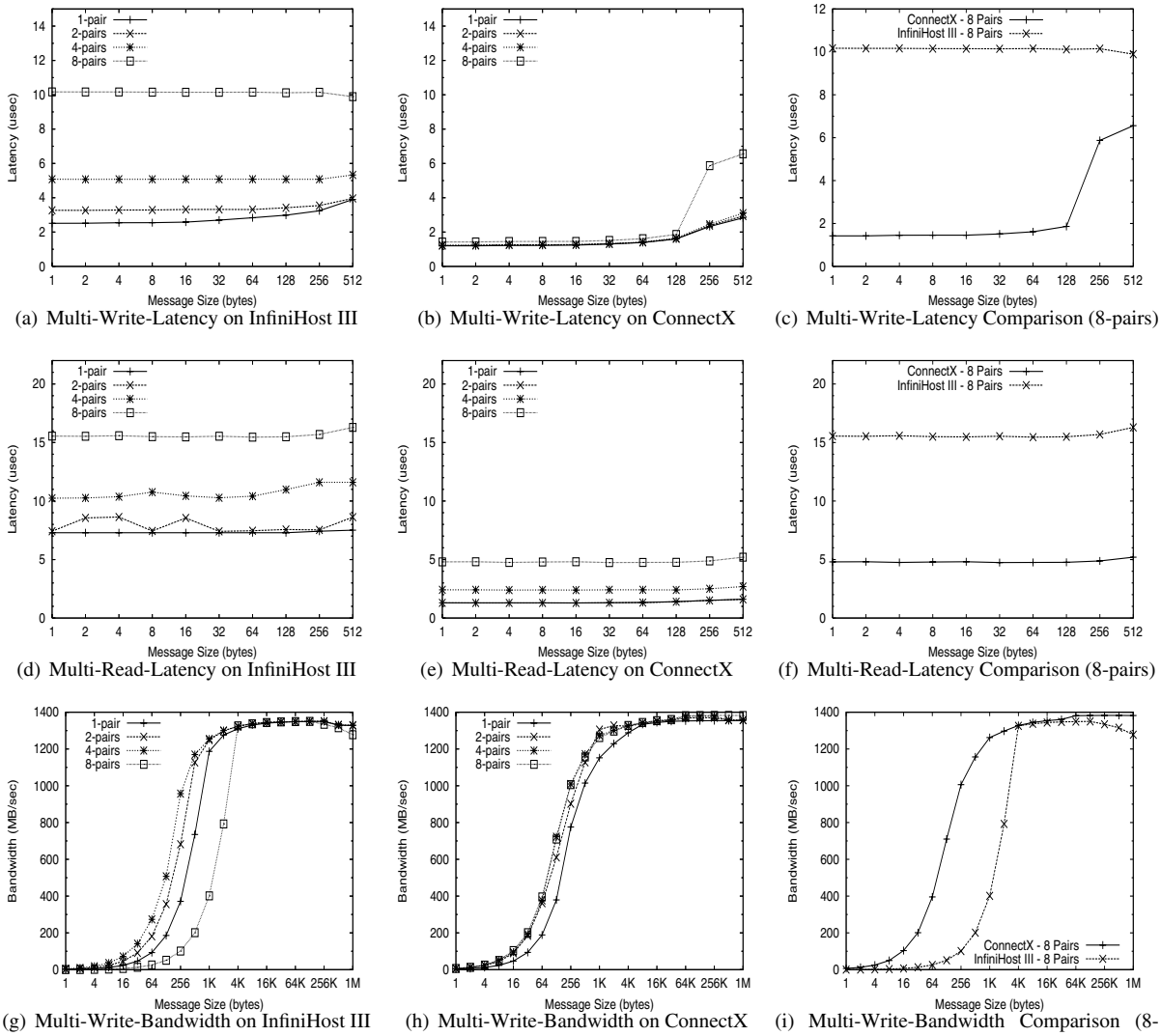
Overall, ConnectX outperforms InfiniHost III by a factor of 3 for message sizes 1-512 bytes when multiple pairs of processes are communicating.

3.3 Multi-Pair Aggregate Write Bandwidth

In this test, we measure the effect on maximum aggregate RDMA-Write bandwidth as the number of communicating pairs on a multi-core platform increases. This test is re-designed along the same lines as described in Section 3.1. As before, this test was derived from the OpenFabrics "ib_rdma_bw" test. Figures 2(g), 2(h) and 2(i) show the results of this experiment. We observe that the aggregate bandwidth obtained by InfiniHost III decreases by almost 4 times when the number of communicating pairs increases to 8. However, with ConnectX architecture, the aggregate bandwidth is not impacted at all as the number of communicating pairs increases. From our results it is clear that even one pair of communicating processes can achieve close to the maximum bandwidth possible, this is a very encouraging result. The encapsulation of WQEs inside the doorbells and the use of PIO is the reason scalable performance can be achieved with multiple pairs communicating at the same time. As the processors get faster and the I/O bus speeds increase, PIO for small messages is much more efficient than setting up DMA. In addition, ConnectX uses much more efficient scheduling and parallelized scheduling engines to quickly schedule incoming WQEs and process them. Comparing with the InfiniHost III adapter (Figure 2(i)), there is around 10 times improvement in aggregate bandwidth for 256-byte messages.

3.4 Halo Benchmark

The Halo Benchmark [15] simulates an NLOM (Naval Research Laboratories, Layered Ocean Model) 2-D exchange for a N by N sub domains $\{N = 0, \cdots, 1024\}$. The authors of this benchmark indicate that the performance of this benchmark is indicative of the suitability of a particular platform for executing layered ocean model (NLOM) codes. The benchmark is particularly latency sensitive. The Halo benchmark is written using MPI, thus to execute this benchmark, we used MVAPICH [12], which is an open-source MPI implementation over InfiniBand. MVAPICH is based



(a) Multi-Write-Latency on InfiniHost III (b) Multi-Write-Latency on ConnectX (c) Multi-Write-Latency Comparison (8-pairs)

(d) Multi-Read-Latency on InfiniHost III (e) Multi-Read-Latency on ConnectX (f) Multi-Read-Latency Comparison (8-pairs)

(g) Multi-Write-Bandwidth on InfiniHost III (h) Multi-Write-Bandwidth on ConnectX (i) Multi-Write-Bandwidth Comparison (8-pairs)

Figure 2. Performance Comparison on Multi-Pair Communication Benchmarks

on MVICH [9] and MPICH [7]. MVAPICH is currently used by more than 525 organizations around the world. MVAPICH-0.9.9-beta2 was used for this evaluation. The MPI processes were distributed in a cyclic manner. This means that every alternate process is on a different node. The four process runs are in a $4x1$ configuration, where four is the number of nodes and one process per node. The eight process runs are in $4x2$ configuration and so on.

The results of our experiments are shown in Figures 3(a) and 3(b). Figure 3(a) shows the performance of Halo on 32 processes with increasing number of tiles. The tile size corresponds to the size of the messages sent. The performance of ConnectX is relative to that of InfiniHost III. All bars in the figures are normalized to 1.0 for InfiniHost III. We observe that for small number of tiles, the performance is 5 times better than that of InfiniHost III. In the scalability graph 3(b), we fix the number of tiles to two and increase the number of processes. In this figure we can observe that due to the low latency of ConnectX, it achieves a factor of 2 to 5 better performance than InfiniHost III.

3.5 LAMMPS Benchmark

Large-scale Atomic/Molecular Massively Parallel Simulator (LAMMPS) [14] is a classical molecular dynamics simulator from Sandia National Laboratories. There are several benchmark datasets available from the LAMMPS website which the authors indicate, are meant to span a range of simulation styles and computational expense for molecular-level interaction forces. LAMMPS reports the "Loop Time" for a particular benchmark as a measure of CPU time required to simulate a set of interactions. In our experimental analysis, we used the Rhodospin protein benchmark from the LAMMPS website. As indicated in Section 3.4 MVAPICH-0.9.9-beta2 was used for this evaluation and the similar process distribution. The results of our experiments are shown in Figure 3(c). This benchmark has significant amount of computation, so the performance of InfiniHost III and ConnectX is almost similar for up to 8 processes. For 16 processes, however, ConnectX outperforms InfiniHost III by almost 10%. This is explained by the factor of 10 difference in aggregate bandwidth for small and medium messages for 8-pair communication as seen in Figure 2(i).

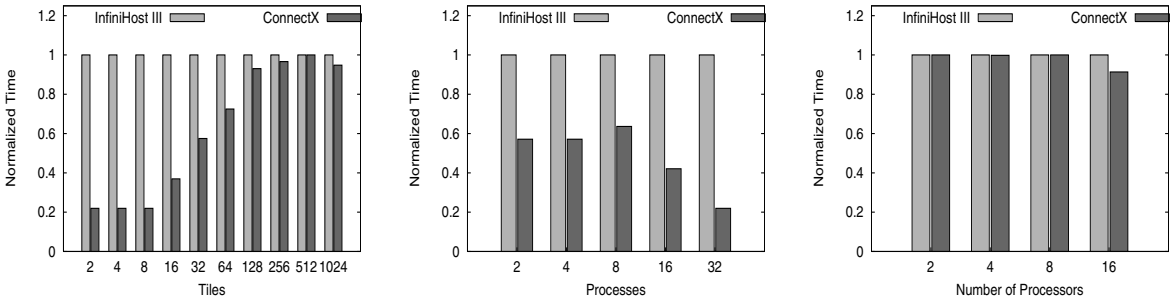
4 Related Work

In this section we discuss related work in this area. QLogic (previously PathScale) are the makers of InfiniPath, another type of InfiniBand adapter. InfiniPath was evaluated in [3] by Dickman et al. and in [2] by Brightwell et al. The InfiniPath adapter also claims to have superior performance on multi-core systems. However, the fundamental technology is different from what is used in either InfiniHost III or ConnectX. InfiniPath relies on spending host CPU cycles to achieve good performance. In other words it is an "On-load" architecture, as opposed to InfiniHost III and ConnectX which are "Offloaded" architectures. In this study we only focus on comparing two adapters, InfiniHost III and ConnectX which are both Offload architectures. Myrinet MPICH-MX [11] is another alternative to InfiniBand. Their unique approach of "partial-offload" results in very good latency and bandwidth. However, to the best of our knowledge, MX does not expose a lower-level one sided messaging model to applications. Applications must be written using an MPI like interface (MX) which includes message tag matching (required for MPI). In [4], Doerfler et al. showed that MPICH-MX has low overhead for a posted send and receive. Greater "application availability" was also reported in the same study. In this paper, we compare the performance of the InfiniHost III and ConnectX architectures at both network and MPI levels specifically on multi-core architectures.

5 Conclusion

In this paper, we present an in-depth network-level performance evaluation of the new Mellanox ConnectX architecture on multi-core platforms. Through our experiments and analysis, we show that the ConnectX architecture is very well suited for modern multi-core platforms. With the increasing number of cores inside a node, there is a need for the network-interface to provide balanced and scalable performance to all communicating processes. The ConnectX architecture provides excellent performance on multi-core platforms through more advanced processing capabilities and fine-grain scheduling of transport related operations on the processing units. In addition, it supports an "Enhanced InfiniBand" mode which proposes, although yet to be enabled by firmware, a new transport called Scalable Reliable Connection (SRC) which aims to alleviate major concerns about memory scalability in large-scale multi-core clusters.

Our analysis and comparison with InfiniHost III architecture reveals that on the Intel Bensley platform with dual Clovertown processors, aggregate bandwidth provided by ConnectX for small and medium sized messages can be increased by a factor of 10 as compared to the third generation InfiniHost III architecture. Similarly, RDMA-Write and RDMA-Read latencies for 1-byte messages can be reduced by a factor of 6 and 3, respectively, even when all cores are communicating simultaneously. Evaluation with communication kernel Halo reveals a performance benefit of a factor of 2 to 5.



(a) Normalized Halo Performance on 32 Processes

(b) Halo Scalability (Tiles = 2)

(c) LAMMPS (in.rhodo)

Figure 3. Halo and LAMMPS Performance Comparison

Finally, the performance of LAMMPS, a molecular dynamics simulator, is improved by 10% for the `in.rhodo` benchmark. As future work, we intend to evaluate the ConnectX architecture thoroughly at the MPI level on a larger scale cluster. As the various features of ConnectX are enabled through firmware, we aim to evaluate them for their performance characteristics on modern clusters and propose MPI level design changes to best take advantage of them.

6 Acknowledgements

We would like to thank Mr. Gilad Shainer from Mellanox Technologies for having many informative technical discussions regarding ConnectX with us. We would also like to thank fellow members of the Network-Based Computing Laboratory: Abhinav Vishnu, Wei Huang, Qi Gao, Amith Mamidala, Gopal Santhanaraman, Sundeep Narravula and Karthikeyan Vaidyanathan for engaging in thought provoking discussions and help with setting up machines and debugging system related issues. Finally, we would like to thank the anonymous reviewers for their constructive criticism and helpful comments.

References

[1] Mellanox Technologies. http://www.mellanox.com.

[2] R. Brightwell, D. Doerfler, and K. D. Underwood. A Preliminary Analysis of the InfiniPath and XD1 Network Interfaces. In *Workshop on Communication Architecture for Clusters, held in conjunction with IPDPS*, 2006.

[3] L. Dickman, G. Lindahl, D. Olson, J. Rubin, and J. Broughton. PathScale InfiniPath: A First Look. In *13th Symposium on High Performance Interconnects (HOTI)*, Palo Alto, CA, 2005. IEEE Computer Society Press.

[4] D. Doerfler and R. Brightwell. Measuring MPI Send and Receive Overhead and Application Availability in High Performance Network Interfaces. In *13th European PVM/MPI Users' Group Meeting*, Bonn, Germany, 2006.

[5] E. Gabriel, G. E. Fagg, G. Bosilca, T. Angskun, J. J. Dongarra, J. M. Squyres, V. Sahay, P. Kambadur, B. Barrett, A. Lumsdaine, R. H. Castain, D. J. Daniel, R. L. Graham, and T. S. Woodall. Open MPI: Goals, concept, and design of a next generation MPI implementation. In *Proceedings, 11th European PVM/MPI Users' Group Meeting*, pages 97–104, Budapest, Hungary, September 2004.

[6] Gordon Moore. Moore's Law. http://www.intel.com/technology/mooreslaw/index.htm.

[7] W. Gropp, E. Lusk, N. Doss, and A. Skjellum. A High-Performance, Portable Implementation of the MPI, Message Passing Interface Standard. Technical report, Argonne National Laboratory and Mississippi State University.

[8] InfiniBand Trade Association. InfiniBand Architecture Specification. http://www.infinibandta.com.

[9] Lawrence Berkeley National Laboratory. MVICH: MPI for Virtual Interface Architecture. http://www.nersc.gov/research/FTG/mvich/ index.html, August 2001.

[10] Mellanox Technologies. ConnectX Architecture. http://www.mellanox.com/products/connectx_architecture.php.

[11] Myricom Inc. MPICH-MX. http://www.myri.com/scs/download-mpichmx.html.

[12] Network-Based Computing Laboratory. MVAPICH: MPI for InfiniBand. http://nowlab.cse.ohio-state.edu/projects/mpi-iba.

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

[14] S. J. Plimpton. Fast Parallel Algorithms for Short-Range Molecular Dynamics. *Journal of Computational Physics*, 117:1–19, 1995.

[15] A. J. Wallcraft. The HALO Benchmark. http://www.navo.hpc.mil/Navigator/Fall99_Feature.html.