Can NIC Memory in InfiniBand Benefit Communication Performance? — A Study with Mellanox Adapter *

Jiesheng Wu Amith R. Mamidala Dhabaleswar K. Panda
Department Computer Science and Engineering,
The Ohio State University
{wuj, mamidala, panda}@cis.ohio-state.edu

Abstract

This paper presents a comprehensive micro-benchmark performance evaluation on using NIC memory in the Mellanox InfiniBand adapter. Three main benefits have been explored, including non-blocking and high performance host/NIC data movement, traffic reduction of the local interconnect, and avoidance of the local interconnect bottleneck. Two case studies have been carried out to show how these benefits can be utilized by applications. In the first case in which the NIC memory is used as intermediate communication buffer for non-contiguous data communication, lower CPU overhead and better latency are attained. In the second case, a common communication building block, communication forwarding chain, *has been studied. Our results show that using the NIC memory can achieve a factor of up to 2.2 improvement over the conventional approach.*

To the best of our knowledge, this is the first such study to demonstrate the benefits of NIC memory in InfiniBand adapter.

1. Introduction

Rapid advances in high performance network interconnects have improved hardware performance substantially in recent years. High performance communication protocols with features such as *user-level networking* and *remote direct memory access* (RDMA) enable applications to realize the peak performance permitted by the network hardware. However, the local interconnect within a server, such as the peripheral component interconnect (*PCI*) bus, becomes a bottleneck in applications since all data being sent over the network must be transferred across this interconnect, particularly for those networks which provide bandwidth comparable to or even higher than the local interconnect bus. The situation becomes even worse when multiple devices may share the same bus. Reducing the traffic across the local interconnect can alleviate this problem.

A significant amount of memory can be attached with the network interface cards (*NIC*) of modern high-speed networks. We call this memory *NIC memory* in this paper. Besides the portion used to accommodate the firmware and various data structures, a modest amount of memory can be left for applications. Using the NIC memory has the potential to reduce the local interconnect traffic, to avoid the possible local interconnect bottleneck [18, 19, 11], and to realize the network hardware capacity.

This paper presents a comprehensive micro-benchmark performance evaluation on using the NIC memory in the Mellanox InfiniBand Adapter. We proposed and designed a set of micro-benchmarks to characterize the NIC memory communication, including *local host/NIC data movement*, *unidirectional latency and bandwidth*, *bidirectional bandwidth*, *uni/bi-directional bandwidth with two ports*, and *hotspot tests*. One of our objectives is to use these benchmarks to explore the potential benefits and various limiting factors of using the NIC memory. This paper also shows how we can take advantage of these benefits. Two case studies have been carried out: (1) Using the NIC memory as the intermediate communication buffers for non-contiguous data communication; and (2) A communication forwarding chain, in which the intermediate nodes receive and forward data to its next node, as common in creating multicast trees. Our results show that significant improvement can be attained using the NIC memory.

The rest of the paper is organized as follows. Section 2 presents background and related work. In Section 3, we describe details of our micro-benchmarks and their performance results of using the NIC memory. Section 4 presents two case studies. In Section 5, we discuss our experience using the NIC memory, its implication with PCI Express, and our future work. Section 6 presents our conclusions.

2. Background

2.1. InfiniBand and Mellanox InfiniBand Adapter

The InfiniBand Architecture [3] defines a System Area Network for interconnecting both processing nodes and I/O nodes. It provides a communication and management infrastructure for both inter-processor communication and I/O. In an InfiniBand network, processing nodes and I/O nodes are connected to the fabric by Channel Adapters (CA). There are two kinds of adapters: Host Channel Adapters (HCA) and Target Channel Adapters (TCA). HCAs sit on processing nodes. TCAs connect I/O nodes to the fabric.

The InfiniHost MT23108 is the Mellanox second generation channel adapter [7]. As shown in Figure 1, this adapter is a single chip dual-port InfiniBand 4X HCA. It supports 64-bit PCI-X interface up to 133 MHz. Each port is connected to the IB fabric using 4x full duplex links with the bit rate of 10 Gb/s. The adapter has a high-speed double-data rate (DDR) interface to external memory. This memory is often called *HCA local memory*. In this paper, we call it NIC memory. The memory interface is 266 MHz DDR-SDRAM, the actual operating frequency is 250 MHz. The architecture supports up to 16 GB of memory (4 DIMMs at 4 GBytes each). The HCA core is capable of full wire speed transmissions over InfiniBand links. The core features a full hardware implementation of the InfiniBand architecture with Hardware Transport Engine that drastically reduces host CPU overhead on communication. This adapter also provides an Application Specific Programmable Packet Engine (ASPPE) and internal InfiniRISC process which allows application specific processing for different TCA implementations.

*This research is supported in part by Department of Energy's Grant #DE-FC02-01ER25506, and National Science Foundation's grants #EIA-9986052, #CCR-0204429, and #CCR-0311542.

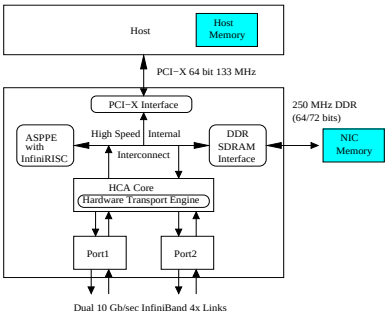

Figure 1. MT23108 HCA Architecture and System Interface.

A full-duplex InfiniBand 4x link supports an ideal bidirectional data rate up to 20 Gb/s. The link is 8b/10b encoded, the usable data bandwidth is 16 Gb/s (or 2000 MBytes/s as indicated in the rest of this paper). Unless stated otherwise, the unit *MB or MBytes* in this paper is an abbreviation for 1000, 000 bytes. The total InfiniBand link data bandwidth of two ports are 4000 MBytes/s. The maximum DDR HCA memory bandwidth is 2000 MBytes/s. The PCI-X 133 bus peak bandwidth is 1064 MBytes/s.

2.2. Using the NIC Memory

The NIC memory in the Mellanox MT23108 HCA serves several purposes [7]. One of them is used as general (non-cacheable) system memory. The Mellanox software API, *VAPI* [8], provides convenient interface for applications to use the NIC memory as the general system memory. Host CPUs can access the NIC memory either through PIO or DMA, HCA considers the NIC memory as an extension to the system memory. VAPI supports both Remote Direct Memory Access (RDMA) Write and Read. The NIC memory is considered in the same way as the system memory in RDMA operations.

2.3. Related Work

There has been several research work on the performance of NIC memory and how to take advantage of the NIC memory for other interconnects. Petrini *et al.* [11, 12] have studied the performance of the NIC memory on the Quadrics network. Liu *et al.* [4] have studied the host memory to host memory communication performance over three networks: InfiniBand, Myrinet, and Quadrics. Buntinas *et al.* [1] and Yu *et al.* [20] have studied using the NIC memory on the Myrinet network to design efficient collective operations. Kim *et al.* [19] have proposed a network interface data caching on the Myrinet network. Yocum *et al.* [18] have proposed a payload caching technique in the network intermediaries on the Myrinet network. Goferey *et al.* [2] have proposed an off-processor I/O architecture to move data between the disk and the Myrinet NIC memory directly over the PCI bus.

Our work differs from the previous work in several important aspects. First, our work focuses on the NIC memory communication over InfiniBand network. To the best of our knowledge, this is the first such work. Second, our work is based on the InfiniBand channel adapter and the 64 bit 133MHz PCI-X bus. Third, the InfiniBand adapter is quite different from Myrinet and Quadrics cards studied in previous work. Lastly, our micro-benchmarks are comprehensive and helpful to explore possible benefits and bottlenecks in the network interconnects.

3. Micro-benchmarks and Performance

In this section, we proposed a set of micro-benchmarks to explore different aspects of the NIC memory communication performance. It includes the following benchmarks: *local host/NIC DMA data movement*, *latency and bandwidth between two nodes*, *bidirectional bandwidth between two nodes*, *uni/bi-directional bandwidth between two nodes with two ports*, and *hotspot tests*.

We conducted our tests on the following cluster. A cluster of 8 SuperMicro SUPER X5DL8-GG nodes, each with dual Intel Xeon 3.0 GHz processors, 512 KB L2 cache, PCI-X 64-bit 133 MHz bus, and connected to Mellanox InfiniHost MT23108 DualPort 4x HCAs. The nodes are connected using the Mellanox InfiniScale 24 port switch MTS 2400. The kernel version used is Linux 2.4.22smp. The InfiniHost SDK version is 3.0.1 and HCA firmware version is 3.0.1. The Front Side Bus (FSB) runs at 533MHz. The physical memory is 1 GB of PC2100 DDR-SDRAM memory.

3.1. Local Host/NIC Data Movement

Using NIC memory requires data movement between the local NIC memory and the local host memory. This data movement can be achieved by either DMA or PIO. The PIO performance is normally very low (its results are not shown here). We designed a test to show the DMA performance using VAPI RDMA Write over a *local* communication Queue Pair. The data source and sink buffers have four combinations: host to host, nic to nic. nic to host, and host to nic. When both buffers are in the host memory, `memcpy(2)` is normally used. The results of memcpy are also shown for comparison.

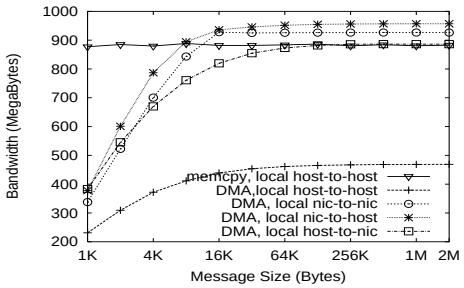

Figure 2. Local Host/NIC Data Movement.

Figure 2 shows results of the above five cases. The peak memory copy bandwidth without cache effect is 882 MB/s. The DMA bandwidth is 468 MBytes/s for the host to host case, 927 MBytes/s for the nic to nic case, 956 MBytes/s for the nic to host case, and 886 MBytes/s for the host to nic case. The last two cases are often referred to HCA DMA write and HCA DMA read. It can be observed that performance of HCA DMA write and read is comparable to or even better than that of the host memory copy on our testbed which has a PC2100 DDR-SDRAM host memory. In addition, the DMA write and read operations have very low CPU overhead, which can benefit communication performance, as to be seen in later sections.

3.2. Latency and Bandwidth between Two Nodes

The latency test is performed in a ping-pong fashion between two nodes. RDMA Write with Immediate data is used to send data and to help both processes detect the arrival of data. Similarly, the source and sink buffers on two nodes can be either in the host memory or the NIC memory or both. Four cases are considered: a local host buffer and a remote host buffer (*host-to-host*); a local NIC buffer to a remote NIC buffer (*nic-to-nic*); a local NIC buffer to a remote host buffer (*nic-to-host*); and a local host buffer to a remote NIC buffer (*host-to-nic*).

Figure 3 shows the latency reduction from the host-to-host case when the NIC memory is used. The reduction

increases when the message size increases in both nic-to-nic and nic-to-host. This is because the reduction of PCI-X bus traffic also increases as the message size increases. The improvement factor using the NIC memory is shown in Figure 4. The host-to-host results are considered as the base. Overall, the nic-to-nic and nic-to-host cases have 7% to 14% improvement. The host-to-nic case performs slightly better than the host-to-host case. There is not much improvement in this case because the HCA DMA read on the RDMA Write initiator side is a bottleneck, as shown in Figure 2.

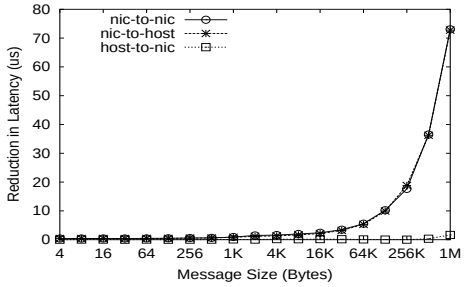

Figure 3. Reduction in Latency of RDMA Write.

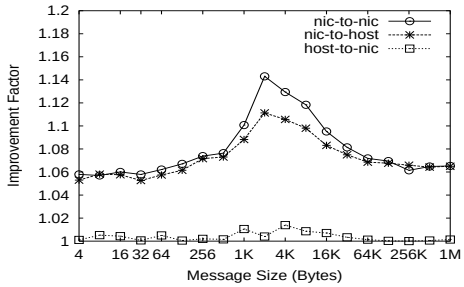

Figure 4. Improvement Factor of RDMA Write Latency.

The RDMA Write bandwidth test is a one-sided pushing bandwidth test as defined in [4]. The window size used in our test is 1000. Figure 5 shows the bandwidth results. The peak bandwidth is around 886 MBytes/s for the host-to-host case, around 944 MBytes/s for the nic-to-nic case, around 944 MBytes/s for the nic-to-host case, and around 886 MBytes/s for the host-to-nic case. The improvement comes from the traffic reduction in PCI-X bus on the RDMA Write initiator side in the nic-to-nic and the nic-to-host cases. Both cases have 7–30% improvement. The host-to-nic and host-to-host cases preform very closely for large messages. For small messages, the host-to-nic has 7–12% improvement.

Similar improvement is also achieved in RDMA Read latency and bandwidth. Details can be found in [15].

3.3. Bidirectional Bandwidth

Characterizing the interconnect bidirectional communication performance can give applications more hints to understand their communication performance. The bidirectional bandwidth test is carried out in a way similar to that of the unidirectional test as mentioned in Section 3.2. The difference is that both sides send data at the same time.

Figure 6 shows the bidirectional bandwidth results. The peak bidirectional bandwidth is 936 MBytes/s in the host-to-host case, 1859 MBytes/s in the nic-to-nic case, 1857 MBytes/s in the nic-to-host case, and 1751 MBytes/s in the host-to-nic case. Compared to the unidirectional bandwidth results in Figure 5, the bidirectional bandwidth using the NIC memory is almost doubled.

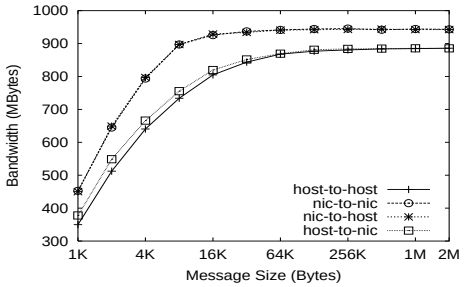

Figure 5. Unidirectional Bandwidth of RDMA Write.

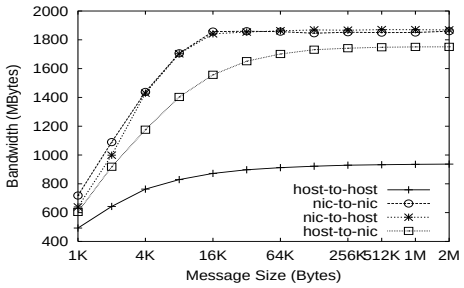

Figure 6. Bidirectional Bandwidth of RDMA Write.

In the host-to-host case, the bidirectional bandwidth is limited by the PCI-X bus which supports an ideal bandwidth up to 1064 MBytes/s. In the nic-to-nic case, there is no PCI-X traffic on both sides. Its bidirectional bandwidth is only limited by the HCA adapter and the IB link. Given an ideal bidirectional data rate up to 2000 MBytes/s over a 4x IB link and the NIC memory bandwidth of 2000 MBytes/s, our results show that the MT23108 adapter itself is capable of realizing almost the peak bandwidth permitted by the hardware with one port when the NIC memory is used. In the nic-to-host case, there is only one way PCI-X bus traffic on both sides. Process 1 performs nic-to-host RDMA Write which incurs PCI-X traffic only on the process 2 side. Similarly, process 2 performs nic-to-host RDMA Write which incurs PCI-X traffic only on the process 1 side. The bidirectional bandwidth can exceed the PCI-X bus limitation. The same reason can be applied to the host-to-nic case.

3.4. Uni/Bi-directional Bandwidth Using Two Ports

The MT23108 adapter has two ports. To explore the maximum communication performance and possible communication bottlenecks, we carried out the unidirectional and bidirectional bandwidth tests using two ports. Each message is striped evenly on two ports.

Figure 7 shows the unidirectional bandwidth results using two ports. The peak bandwidth is 855 MBytes/s in the host-to-host case, 1686 MBytes/s in the nic-to-nic case, 961 MBytes/s in the nic-to-host case, and 855 MBytes/s in the host-to-nic case.

Although the two ports together support a maximum of 2000 MBytes/s unidirectional bandwidth, the PCI-X bus becomes a bottleneck in the host-to-host, nic-to-host, and host-to-nic cases. Compared to the unidirectional bandwidth using one port, the bandwidth in the host-to-host and host-to-nic cases using two ports is slightly lower. This is because the number of RDMA Write operations across the PCI-X bus on the RDMA Write initiator side is doubled due to striping. In the nic-to-nic case, the bandwidth using two ports is increased to 1686 MBytes/s from 944 MBytes/s using one port, The PCI-X bus bottleneck is avoided.

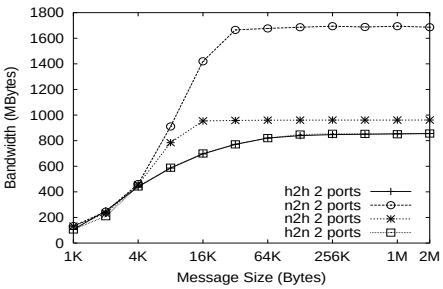

Figure 7. Unidirectional Bandwidth of RDMA Write using Two Ports.

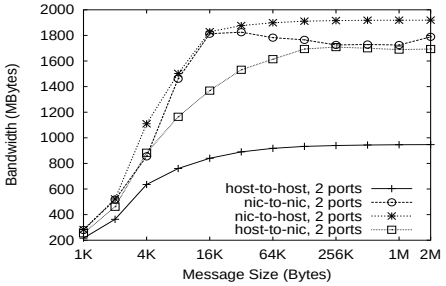

Figure 8. Bidirectional Bandwidth of RDMA Write using Two Ports.

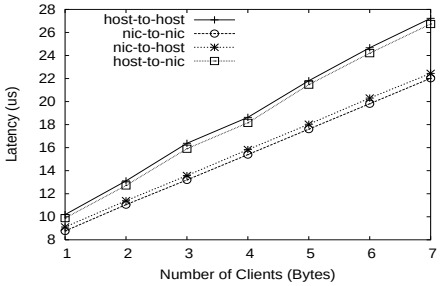

Figure 9. Hotspot Write Test Results with 2048-byte Messages.

Figure 8 shows the bidirectional bandwidth results using two ports. The peak bandwidth is 946 MBytes/s in the host-to-host case, 1788 MBytes/s in the nic-to-nic case, 1919 MBytes/s in the nic-to-host case, and 1693 MBytes/s in the host-to-nic case.

These results demonstrate interesting limiting factors in each case. The PCI-X bus becomes a bottleneck in the host-to-host case. Although two ports together support a maximum bidirectional bandwidth up to 4000 MBytes/s, the NIC memory with a peak bandwidth of 2000 MBytes/s becomes a limiting factor in the nic-to-nic case. In the nic-to-host case, all writes go to the host memory through the PCI-X bus, and all reads come from the NIC memory. The PCI-X bus becomes the limiting factor. The ideal bidirectional bandwidth in this case is two times of the PCI-X bus bandwidth. The nic-to-host case performs better than the nic-to-nic case when two ports are used. We suspect that this is because only reads on the NIC memory occur in the nic-to-host case, while a mix of reads and writes occur in the nic-to-nic case. The performance of the HCA memory interface may be not optimized for the intermixing of reads and writes. Similarly, in the host-to-nic case, its ideal bandwidth is two times of the PCI-X bus bandwidth. The lower bandwidth than the nic-to-nic and the nic-to-host cases is because of the lower performance of the HCA DMA read as discussed in Section 3.2.

3.5. Hotspot Tests

Applications often have many-to-one or one-to-many communication patterns and therefore generate hotspot traffic. To characterize the ability of network interconnects to handle hotspot traffic, we designed a hotspot write test and a hotspot read test. In the hotspot write test, a master process writes data to several slave nodes one by one using RDMA Write. The time when the data arrives on the last slave node is reported. This communication pattern is often used in I/O severs when multiple clients read files simultaneously.

Figure 9 shows latency results of the hotspot write test. The message size is 2048 bytes. The nic-to-nic and nic-to-host cases achieve around 20% improvement over the host-to-host and host-to-nic cases. Details of the hotspot read test can be found in [15].

4. Case Studies

In this section, we first summarize the potential benefits of using NIC memory. Then we present two case studies to demonstrate how applications can take advantage of these benefits.

4.1. Potential Benefits of Using NIC Memory

Results in Section 3 demonstrate the following potential benefits. First, non-blocking and high performance local host/NIC data movement enables applications to use the NIC memory as an intermediate buffer for communication. It not only provides performance comparable to the host memory copy bandwidth as shown in Figure 2, but also provides the non-blocking semantics and thus uses less CPU compared to the blocking semantics of the host memory copy.

Second, using NIC memory can reduce the PCI-X bus traffic. This reduction can improve communication performance of basic latency, bandwidth, and hotspot communication. Figures 3, 5, and 9 show these benefits.

Third, using NIC memory can help applications avoid the PCI-X bus bottleneck and realize higher performance permitted by network hardware. The results of the unidirectional bandwidth test using two ports (Figure 7), bidirectional bandwidth using one and two ports (Figures 6 and 8) demonstrate this benefit. In the rest of this section, two case studies are presented to show how applications can take advantage of these benefits.

4.2. Intermediate Communication Buffers for Non-Contiguous Data Communication

Intermediate communication buffers are often used in applications and communication libraries such as MPICH [14] to hold data which is sent or received. MPI Datatype communication is a typical example [14, 16].

Suppose a process wants to send M non-contiguous blocks to another process. The block size is S bytes. Two approaches are often used. The first one copies the total $M \times S$ bytes data into an intermediate buffer and then sends all data out once. The second one uses RDMA Write Gather operation. In the first approach, the intermediate buffer can be pre-registered before hand, only the copying cost must be paid. The NIC memory can be used in the first approach as the intermediate communication buffer. We call these cases as *host-memory pack* and *nic-memory pack*, respectively. The second approach may pay the memory registration and deregistration costs. Its performance depends on how often these buffers are reused [4, 16].

In the following test, 8 blocks are considered. The block size varies from 8 KBytes to 64 KBytes. The gap size be-

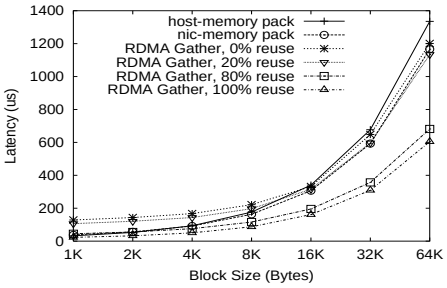

Figure 10. Latency.

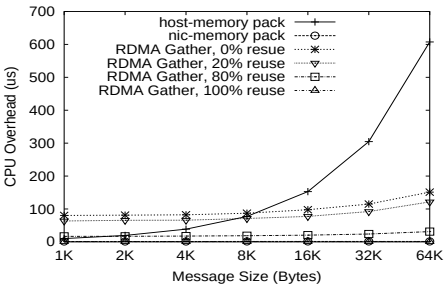

Figure 11. CPU Overhead.

tween two blocks equals to the block size. 0%, 20%, 80%, and 100% buffer reuse ratioes are considered.

The nic-memory pack approach achieves performance comparable to or better than the host-memory pack. This is because they take almost the same time to pack data and the nic-memory pack approach can take benefit of the nic-to-host communication as shown in Figure 3. The nic-memory pack approach also beats the RDMA Gather approach when the buffer reuse ratio is low.

Figure 11 shows the CPU overhead in the above test. The memory copy cost in the host-memory pack approach incurs high CPU overhead. The memory registration and deregistration in the RDMA Gather approach also incurs high CPU overhead. However, the nic-memory pack approach uses little CPU, indicating that the whole communication time can be overlapped with other useful computation.

4.3 Communication Forwarding Chain

Communication forwarding chain is a frequently used pattern in applications and communication libraries, i.e., node 1 sends data to node 2, node 2 receives data and sends data to its next node, and so on. A set of nodes form a communication chain and the intermediate nodes receive and forward data. It can serve as a "building block" to construct complicated communication algorithms. For example, this pattern is often used in tree-based collective communication algorithms [13].

We designed a test to show how we can use the NIC memory in the intermediate nodes to accelerate the forwarding. Two approaches are considered. In the host-memory-based approach, the intermediate nodes receive data into their host memory and then send data out. Communication occurs between the host memory of these nodes. Data goes through the PCI-X bus two times. In the NIC-memory-based approach, the intermediate nodes first receive data into the NIC memory, then forward data to the next node and upload data into its host memory. Communication between the first two nodes is from the host memory to the NIC memory, from the NIC memory to the NIC memory between the intermediate nodes, and from the NIC memory to the host memory between the last two nodes. The time the last node takes to receive data is reported.

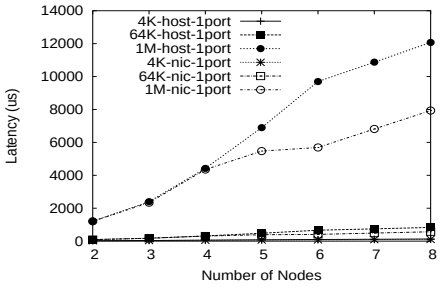

Figure 12. Latency Comparison between the Host-memory-based and NIC-memory-based Forwarding Chain Using One Port.

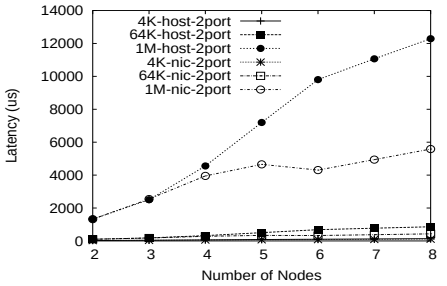

Figure 13. Latency Comparison between the Host-memory-based and NIC-memory-based Forwarding Chain Using Two Ports.

Figure 12 shows results for three message sizes of 4 KBytes, 64 KBytes, and 1 MBytes using one port. The NIC-memory-based approach can gain significant benefit from the higher communication performance using NIC memory. The total benefit is the sum of benefit between every two nodes.

Figure 13 shows results using two ports. The NIC-memory-based approach achieves much better performance than the host-memory-based approach. Performance is improved by a factor of up to 2.2. This is because the NIC-memory-based approach further takes full advantage of the high nic-to-nic unidirectional bandwidth over two ports as shown in Figure 7.

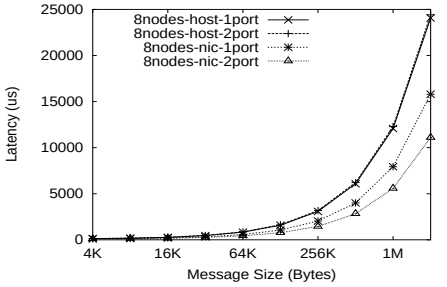

Figure 14. Latency Comparison between the Host-memory-based and NIC-memory-based Forwarding Chain With Different Message Sizes.

Figure 14 shows that the NIC-memory-based approach outperforms the Host-memory-based approach. It also can take advantage of using two ports, which the Host-memory-based approach can not due to the PCI-X bus limit. This case demonstrates how applications can take advantage of the benefits of the PCI-X bus traffic reduction and avoid the PCI-X bus limitation to realize higher performance permit-

ted by the network hardware in using the NIC memory.

5 Discussion and Future Work

The NIC memory communication performance depends on four components in the Mellanox Channel Adapter: the NIC memory interface, the internal interconnect, the Hardware Transport Engine, and the physical link, as shown in Figure 1. Compared to the host memory to host memory communication, the benefits of using the NIC memory are relevant to whether the local host interconnects such as PCI bus and PCI-X bus are the main bottleneck or one or more than one of the above four components are the main bottlenecks. In our studied Mellanox MT23108 adapter, the PCI-X bus is the main bottleneck. This is a common situation for high speed networks, partly because the network technologies have advanced faster than the local interconnect technologies in recent years. In addition, multiple devices may share the same local interconnect buses. Therefore, the benefits of using NIC memory we have explored are valid in many general cases.

The new local interconnect technology, *PCI Express* [10], has been proposed to address the issues associated with PCI-X. As one of its salient features, PCI-Express is a serial interconnect, providing improved reliability, full duplex transmission and simpler routing/cabling [5]. Mellanox has released their third generation HCA device (*InfiniHost III Ex MT25208*) [6] with an 8X PCI Express interface recently. The 8X PCI Express has 32 Gbps full duplex usable data bandwidth (40 Gbps bit rate). Since the MT25208 HCA local memory bandwidth is comparable to the one way data rate of 8X PCI Express, we expect that the unidirectional communication with involvement of the NIC memory must be comparable to the host memory communication, if not better. Therefore, non-blocking and high performance local host/NIC data movement is still beneficial. We also expect that bidirectional communication for the NIC memory to the host memory might be the best since the traffic is spread to PCI Express/the host memory bus and the NIC memory bus. Therefore, many benefits explored in our case studies can be applied to PCI Express as well. We plan to evaluate the NIC memory communication performance with PCI Express.

As our future work, we are also working on incorporating the ideas explored in our case studies to Datatype communication and collective communication in MVAPICH [9, 16] and network data server cache [17].

6 Conclusions

This paper has two main parts. In the first part, we presented a set of new micro-benchmarks and their use to explore the potential communication benefits of using the NIC memory in the Mellanox InfiniBand adapter and identify various performance limiting factors. In the second part, we conducted two case studies to show how applications can take advantage of these performance benefits. The main benefits include non-blocking and high performance local data movement between the host memory and the NIC memory, reduction of the PCI-X bus traffic, and avoiding the PCI-X bus limitation to realize higher performance permitted by the network hardware. Results of our case studies show that with careful design, application performance can be enhanced significantly by taking advantage of these benefits.

Acknowledgments

We would like to thank Diego Crupnicoff from Mellanox for his valuable comments on the paper draft and answers to many of our questions about the Mellanox adapter. We are also thankful to nowlab fellows, Dr. Hyun-Wook Jin, Jiuxing Liu, Sayantan Sur, and Gopalakrishn Santhanaraman for their comments.

References

[1] D. Buntinas, D. K. Panda, and P. Sadayappan. Fast NIC-Based Barrier over Myrinet/GM. In *International Parallel and Distributed Processing Symposium (IPDPS)*, April 2001.

[2] P. Geoffray. OPIOM: off-processor I/O with myrinet. *Future Generation Computing System*, 18(4):491–499, 2002.

[3] InfiniBand Trade Association. InfiniBand Architecture Specification, Release 1.0, October 24, 2000.

[4] J. Liu, B. Chandrasekaran, W. Yu, J. Wu, D. Buntinas, S. P. Kini, D. K. Panda, and P. Wyckoff. Micro-Benchmark Performance Comparison of High-Speed Cluster Interconnects. *IEEE Micro*, 24(1):42–51, Janurary/February 2004.

[5] D. Mayhew and V. Krishnan. PCI Express and Advanced Switching: Evolutionary Path to Building Next Generation Interconnects. In *Hot Interconnects 11*, August 2003.

[6] Mellanox Technologies. Mellanox InfiniHost III Ex. `http://www.mellanox.com/products/`.

[7] Mellanox Technologies. Mellanox InfiniHost MT23108. `http://www.mellanox.com/products/`.

[8] Mellanox Technologies. Mellanox IB-Verbs API (VAPI), Rev. 0.95, March 2003.

[9] Network-Based Computing Laboratory. MVAPICH: MPI for InfiniBand on VAPI Layer. `http://nowlab.cis.ohio-state.edu/projects/mpi-iba/index.html`, January 2003.

[10] PCI-SIG. PCI Express Base Specification 1.0a. `http://www.pcisig.com/ specifications/pciexpress`.

[11] F. Petrini, W. chun Feng, A. Hoisie, S. Coll, and E. Frachtenberg. The quadrics network: High-performance clustering technology. *IEEE Micro*, 22(1):46–57, 2002.

[12] F. Petrini, S. Coll, E. Frachtenberg, and A. Hoisie. Performance Evaluation of the Quadrics Interconnection Network. *Journal of Cluster Computing*, 6(2):125–142, 2003.

[13] R. Thakur and W. Gropp. Improving the Performance of Collective Operations in MPICH. In *EuroPVM/MPI*, Oct. 2003.

[14] W. Gropp and E. Lusk and N. Doss and A. Skjellum. A High-Performance, Portable Implementation of the MPI, Message Passing Interface Standard.

[15] J. Wu, A. R. Mamidala, and D. K. Panda. Can NIC Memory in InfiniBand Benefit Communication Performance? – A Study with Mellanox Adapter. Technical report, Dept. of Computer Science and Engineering, The Ohio State University, March 2004.

[16] J. Wu, P. Wyckoff, and D. K. Panda. High Performance Implementation of MPI Datatype Communication over InfiniBand. In *International Parallel and Distributed Processing Symposium (IPDPS '04)*, April 2004.

[17] J. Wu, P. Wyckoff, D. K. Panda, and R. Ross. Unifier: Unifying Cache Management and Communication Buffer Management for PVFS over I nfiniBand. In *Proceedings of the IEEE/ACM Symposium on Cluster Computing and the Grid (CCGrid2004)*, April 2004.

[18] K. Yocum and J. S. Chase. Payload caching: High-speed data forwarding for network intermediaries. In *Proceedings of the General Track: 2002 USENIX Annual Technical Conference*, pages 305–317. USENIX Association, 2001.

[19] H. youb Kim, V. S. Pai, and S. Rixner. Increasing web server throughput with network interface data caching. In *Proceedings of the 10th international conference on architectural support for programming languages and operating systems (ASPLOS-X)*, pages 239–250. ACM Press, 2002.

[20] W. Yu, D. Buntinas, and D. K. Panda. High Performance and Reliable NIC-Based Multicast over Myrinet/GM-2 . In *the 2003 International Conference on Parallel Processing (ICPP'03)*, Oct. 2003.