Comparative Performance Analysis of RDMA-Enhanced Ethernet

Casey B. Reardon[1], Alan D. George[1], and Clement T. Cole[2]
[1]*Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL*
[2]*Ammasso, Inc., Boston, MA*
{reardon,george}@hcs.ufl.edu, clemc@ammasso.com

Abstract

Since the advent of high-performance distributed computing, system designers and end-users have been challenged with identifying and exploiting a communications infrastructure that is optimal for a diverse mix of applications in terms of performance, scalability, cost, wiring complexity, protocol maturity, versatility, etc. Today, the span of interconnect options for a cluster typically ranges from local-area networks such as Gigabit Ethernet to system-area networks such as InfiniBand. New technologies are emerging to bridge the performance gap (e.g. latency) between these classes of high-performance interconnects by adapting advanced communication methods such as remote direct-memory access (RDMA) to the Ethernet and IP environment. This paper provides an experimental performance analysis and comparison between three competing interconnect options for distributed computing: conventional Gigabit Ethernet; first-generation technologies for RMDA-enhanced Gigabit Ethernet; and InfiniBand. Results are included from basic MPI-level communication benchmarks and several application-level benchmarks on a cluster of Linux servers and show that emerging technologies for low-latency IP/Ethernet communications have the potential to achieve performance levels rivaling costlier alternatives.

1. Introduction

In the past decade, with the introduction and increasing use of computational clusters of servers or workstations based on commercial off-the-shelf (COTS) technologies, high-performance computing (HPC) has become part of the mainstream in the physical, engineering, and life sciences for purposes such as simulation, in commercial environments for purposes such as transaction processing, etc., targeting a broad range of applications. The distributed, non-shared memory machines in these clusters typically intercommunicate via explicit message passing and leverage a variety of middleware services. However, since the ascension to dominance of these distributed-memory machines, they have continually suffered from difficulty in programming (e.g. due to limits in tools and services), as well as interconnect price and performance. The HPC community has focused its efforts on solving the programming problem by developing MPI, the Message Passing Interface [1], while the hardware community has developed new and better interconnects with larger throughput, lower latency, CPU offload, etc. Together, these communities developed a software layer to make the difference in those interconnects less noticeable by mapping MPI onto a lower-level application programming interface (API) for clusters called uDAPL, the user-mode Direct Access Provider Library [2].

With these tools in place, a new cost-effective networking solution is emerging, where Ethernet devices are equipped with Remote Direct-Memory Access (RDMA) capabilities that promise increased performance at modest cost. In this paper, we provide a background into the operation of RDMA and the iWARP protocol. A first-generation iWARP implementation is presented, the Ammasso AMSO 1100, an RDMA-based NIC, or RNIC. The performance of this device is compared to the performance of two existing and widely accepted high-performance interconnect technologies, InfiniBand and conventional Gigabit Ethernet, using a set of MPI benchmarks and applications.

The remainder of this paper is organized as follows. Section 2 provides an overview of RDMA and its components, including the RNIC under study. Section 3 describes the tools and benchmarks used in our experiments. The results of the experiments are presented in Section 4. Section 5 provides an analysis of the experimental results, and Section 6 summarizes this work and cites conclusions.

2. Remote Direct-Memory Access

Just as direct memory access for traditional disk and tape controllers allows a memory transfer without the intervention of the host CPU, RDMA permits data to be moved from the memory subsystem of one system on the

network to another without involving either of the host CPUs with the data copies, memory protection, and housekeeping computations such as CRC, etc. While the ideas behind RDMA are not new, its widespread acceptance has been limited historically by the requirement of new and different networking infrastructures. However, new standards are enabling RDMA to be implemented using Ethernet as the physical layer and TCP/IP as the transport. This approach combines all of the performance and latency advantages of RDMA while maintaining the cost, maturity, and standardization benefits of Ethernet and TCP/IP.

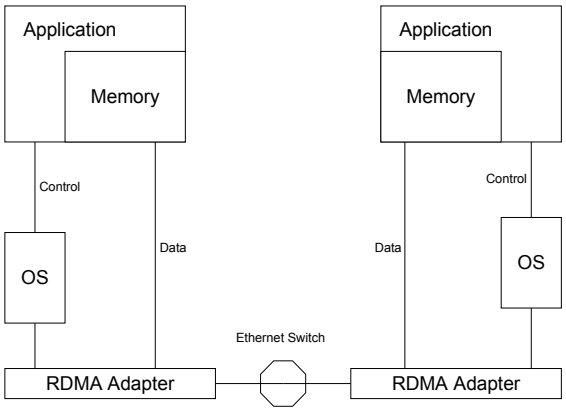


Figure 1: RDMA Concept

Figure 1 illustrates the data and control paths that an application uses when performing a RDMA transfer with an RNIC, such as the AMSO 1100. Once per connection, application memory is registered with the local RNIC and is locked down by the operating system. Using the control path, the application requests the operating system and the adapter to create and exchange a cookie, called a Steering Tag or *STag*, which will be used later during the data transfer. The STag identifies to each RNIC the memory region involved in the data exchange. The cost of this setup is paid at connection start-up but amortized across the life of the application. The one-time creation of the connection is used to ensure the availability of both sides of the transfer, and provide memory protection guarantees, etc.

When an actual data transfer is needed during the life of the application, the application writes to privately allocated queues directly mapped into the local address space. This approach allows the application to provide a specific starting offset for the transfer and the size needed, as well as initiate the transfer, and can significantly accelerate remote data transfers for several reasons:

1. Adding a work request does not require a system call or other kernel context switch. However, this user-space application action forces the RNIC to start the transfer without any other help needed by the local kernel. System designers call this type of operation *kernel bypass*.
2. Unlike a traditional TCP/IP network stack, the user data is never copied into kernel buffers or rearranged into serial streams by the kernel, thus providing *zero-copy* semantics. Removing copies removes transfer time and latency, but more importantly it directly affects system performance. As Talpey notes, avoiding data copies adds two times the line rate to available memory bandwidth of the processor [3]. As memory bandwidth in modern architectures has rapidly become a scarce resource, gaining that much memory bandwidth is a major benefit.
3. As noted, the combination of avoiding data copying and context switching significantly reduces the latency needed to deliver the message.
4. Finally, since the hardware is performing the transfer, the application can immediately be freed to take advantage of the local processor as it sees fit, and thus achieve the benefits of *CPU offload*.

Transfer completion semantics can be either a traditional poll or an asynchronous *event* notification. The RNIC's *Verbs* layer provides the ability to use either of these mechanisms, and various applications are free to mix or match notification schemes as defined by the application designer.

2.1 iWARP Components

Figure 2 is a diagram of the layered components of iWARP including the two optional portability layers, MPI [1] and DAPL [2]. The physical media layer is currently implemented using a standard Gigabit Ethernet.[+] On top of the physical interconnect is layered a conventional TCP/IP protocol stack. Since the lowest two layers are 100% orthodox, iWARP can be managed (i.e. switched, routed, and debugged) using existing and mature tools, hardware, and software. It is the layers built on top of TCP that makes iWARP interesting. We will review these in succession.

The Markers with PDU Alignment or MPA layer is the framing layer. Given that the goal is direct data placement to and from user memory (i.e. copy avoidance), the upper layers need to accurately identify user data payload from overhead (i.e. headers). When data is delivered by the network in order, alignment is not an issue since the segment lengths are known and can be used to find the physical segment boundaries. However, to

[+] We expect that in the future the physical layer will migrate from Gigabit Ethernet to 10 Gigabit Ethernet when the cost of the latter technology running over CAT 5/6 copper cables becomes commercially viable.

support out-of-order arrival, a deterministic mechanism needs to be added so that a receiver can find the physical data segments. Thus, a marker is inserted at 512-byte intervals to locate the header, which in turn can be found by examining the TCP sequence number.

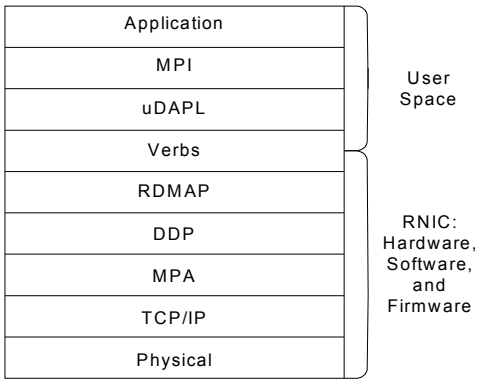


Figure 2: iWARP Layers

The Direct Data Placement or DDP layer is the placement and ordering layer. It is this layer that adds and examines the STags from and to memory objects. It provides the mechanics to register and unregister memory as well as DDP send and post operations. The DDP layer is tightly integrated with the RDMAP layer. It is also a client to the facilities that the MPA layer provides.

Like DDP, the RDMA Protocol or RDMAP layer has functions that manipulate memory addresses and provide send/read/write semantics. However, this layer is where the physical protection is enforced and that higher-level functions, such as operation completion semantics, are maintained. See Bailey and Talpey for more details [4].

Instead of trying to define a specific application interface for RDMA, the iWARP designers defined actions called *Verbs* for systems developers to map into specific actions [5]. The iWARP Verbs specification defines the *actions* provided by the RNIC (hardware, firmware, and software) when the host and operating system interact with it. The specification does not define the host software or host-to-RNIC interface API [6]. The primary concept used by the Verbs layer is the concept of the application sending *work requests* (WR) to the RNIC over a *queue pair* (QP). A QP is a pair of work queues for sending (SQ) and receiving (RQ) work to and from the RNIC. Asynchronous notification is handled via the completion queue (CQ).

While the lowest level of work is defined in terms of queue pairs, actual TCP/IP connections must also be managed. It is the Verbs layer that exports that management, usually referred to as the connection manager or CM. Simply put, connection *endpoints* define the passive and active side of a connection, containing IP address, protocol type, port number, and some protocol state. The application calling the verbs needs to associate the endpoint with an RDMA QP during the connection set-up to allow standard RDMA WRs to use the connection. The CM maintains all connection state used by the RNIC. The endpoints are managed internally by the CM and RNIC, and are not directly exposed to the client application.

2.2 Ammasso AMSO 1100

The RDMA-capable Ethernet NIC featured in our experiments is the AMSO 1000 from Ammasso, Inc. [6] The AMSO 1100 is a first-generation RNIC and implements the iWARP, user and kernel DAPL, and MPI specifications [5, 2, 1]. The hardware provides a traditional IEEE-conformant MAC layer interface, a PCI-X interface to the host, all memory transfer operations (data placement and motion), CRC calculation, and the internal performance-sensitive states of the TCP state machine. Embedded microprocessors on board the RNIC working in conjunction with the host provide connection management, options, and error management. Figure 3 is a functional diagram of the AMSO 1100.

The Ammasso implementation of MPI used in this investigation is based on MPI 1.2.5 from Argonne National Lab plus selected bug fixes later released in 1.2.6. The hardware interface extensions developed by UC Berkeley and Ohio State, originally developed to call an InfiniBand-based RDMA implementation, were modified to directly call the Ammasso iWARP RDMA Verbs. This implementation uses traditional TCP sockets for the MPI control.

3. Benchmarks and Configuration

To evaluate and analyze the performance of the emerging RDMA technologies versus existing networking solutions, several sets of communications and distributed computing benchmarks were selected for our experiments. All of these benchmarks are based on MPI. First, the Pallas MPI Benchmark suite (PMB) was employed for low-level MPI performance evaluation. Two application-level benchmark suites were then employed, Gromacs and the NAS Parallel Benchmark (NPB) suite. All of these benchmarks were executed on the same cluster of servers using three different high-performance communication networks: conventional Gigabit Ethernet, RDMA-based Gigabit Ethernet, and InfiniBand. Each of the network interfaces on each server for each network was housed in a PCI-X bus slot.

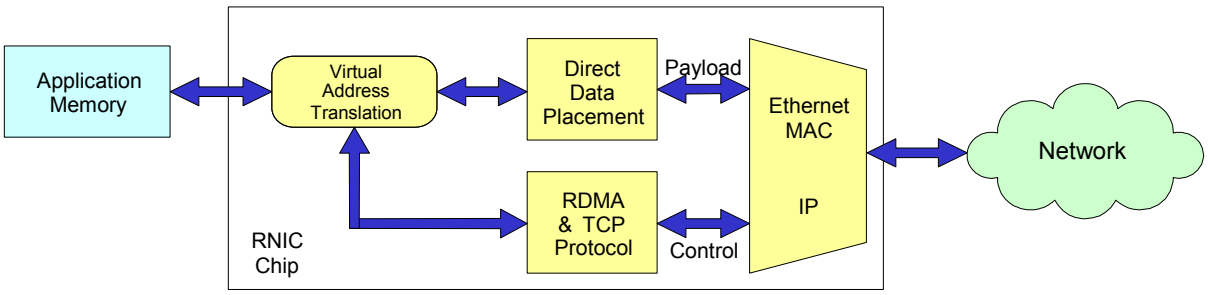


Figure 3: Block Diagram of AMSO 1100 RNIC

To obtain a raw, low-level performance comparison between the networks, the Pallas MPI Benchmarking suite was selected [7]. Pallas is a useful benchmarking suite for several reasons. A multitude of tests are available to the user to evaluate any common MPI function, and these tests are easily customizable. The two tests used from this suite were PingPong and SendRecv. With PingPong, the measured round-trip time is divided by two to calculate the one-way latency. Throughput for the PingPong test is then derived by dividing the message size by the one-way latency. SendRecv forms a periodic chain of nodes, where each node simultaneously sends messages to the next node in the chain and receives messages from the previous node. This benchmark is effective in testing the bidirectional bandwidth between hosts. Latency values are derived by the time it takes all nodes to complete a send to the other, and bandwidth is derived again by dividing the message size by the measured latency.

Next, the protein simulator Gromacs was considered. The Gromacs benchmark was first developed at the Department of Biophysical Chemistry at Groningen University [8]. The software is used to simulate molecular dynamics for systems with up to millions of particles. A series of molecular systems is distributed by the developer to benchmark the performance of Gromacs on a system. We studied three of these protein systems: Villin, DPPC, and LZM. Gromacs was selected as a benchmark set for several reasons. For one, it is very sensitive to latency, and not simply bandwidth, thereby rendering it useful for studying the impact of RDMA-capable networks. Gromacs executes by distributing the atoms in the simulated system across all active nodes. The time is then broken down into many steps, each representing a small fraction of a picosecond of simulation time, depending upon the system. Synchronization must occur with all nodes before proceeding to the next step. This frequent communication is heavily impacted by network latencies. A second reason that Gromacs was chosen as a benchmark was for its computational demand. Any advantage the network architecture offers in terms of offloading communication processing should become evident during these benchmark runs.

Developed by the NASA Advanced Supercomputing Division, the NPB 2.4 benchmark suite has been used for performance evaluation of many parallel computer systems and networks. The different tests within this suite vary much more widely in their behavior as compared to the three Gromacs tests evaluated. The two benchmarks selected from the NPB suite and used herein are the integer sort (IS) and the conjugate gradient (CG) [9].

The Lambda cluster of the HCS Laboratory at the University of Florida was used as the experimental testbed for this study. This cluster consists of 16 server nodes, each containing dual AMD Opteron 240 processors, a Tyan Thunder K8S motherboard with PCI-X bus slots, and 1GB of memory, and operating with Suse Linux 9. Only one of the two CPUs per node is used in these experiments. For the conventional Gigabit Ethernet network, each server uses its on-board Broadcom PCI-X BCM5704C controller, and LAM 7.1.1 is used for MPI [10]. For the RDMA-based Gigabit Ethernet network, each server has an AMSO 1100 RNIC from Ammasso connected in a PCI-X slot. A version of MPICH-2.1.5 is distributed with the AMSO 1100 cards, and this is used to implement MPI. In both of these cases, nodes are connected to one another through a Nortel Baystack-5510 Gigabit Ethernet switch. For the InfiniBand network, each server is outfitted with a Voltaire 400LP Host Channel Adapter (HCA) connected to a Voltaire 9024 InfiniBand Switch Router. The switch and all HCAs operate using the 4X (i.e. 10 Gb/s) InfiniBand fabric. MVAPICH-0.9.5 from Ohio State University is used to implement MPI over InfiniBand [11]. All tests were conducted by researchers at the University of Florida.

4. Experimental Results

Figures 4(a) and 4(b) show the results of the PMB PingPong tests, with both throughput and latency data. The results are the average of three test runs for each case, and only the data for messages sizes up to 1 KB are shown. As the message size grows beyond 1 KB, the test becomes bandwidth-oriented, which of course obviously favors the extra throughput the 10 Gb/s InfiniBand offers.

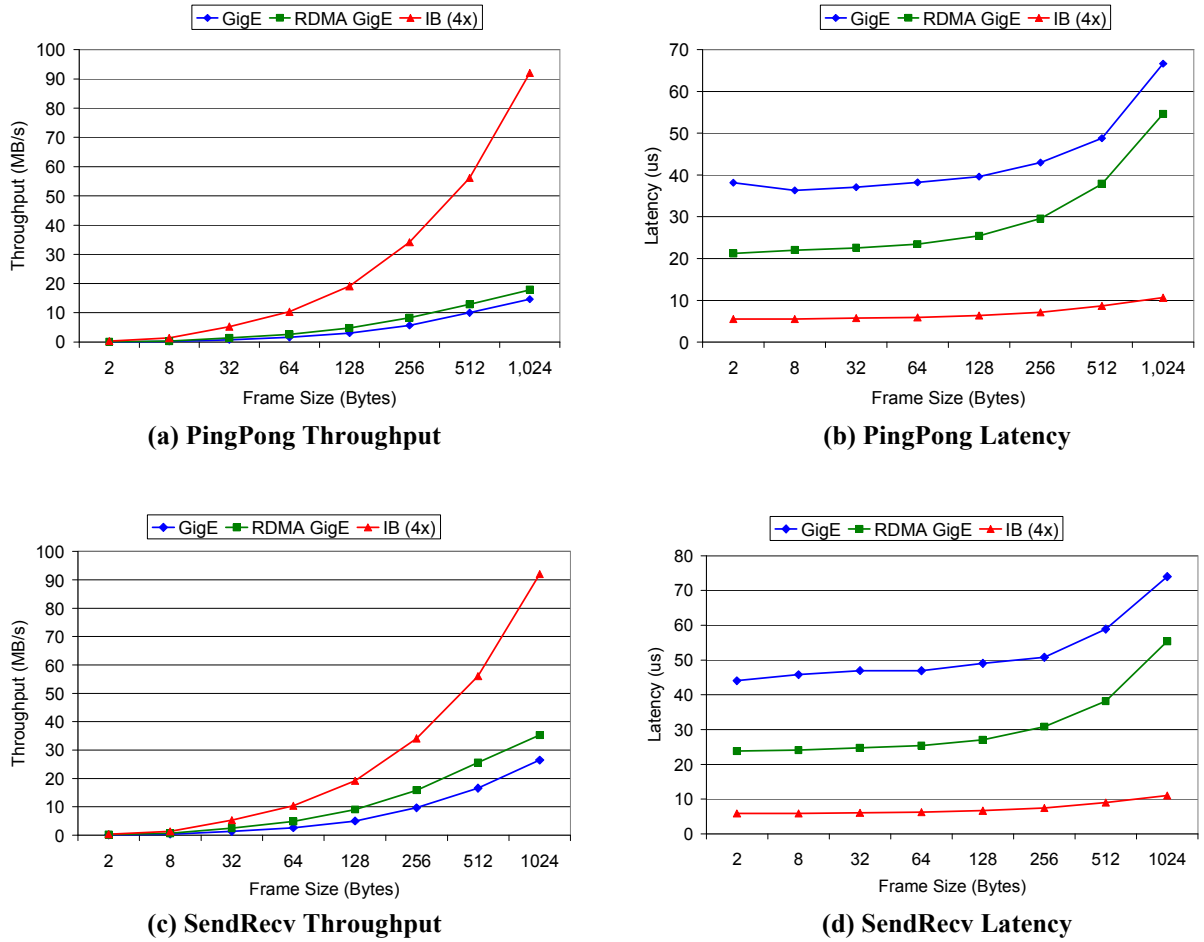


Figure 4: PMB PingPong and SendRecv Performance Results

The bandwidth advantage of InfiniBand grows larger for messages of increasing size, diverging such that eventually these graphs become skewed, our initial interests for this study are primarily with communications using small- to medium-sized messages, and thus only messages up to 1KB in size are included here. Figure 4(a) shows that InfiniBand does provide the highest throughput as expected, but additional throughput is also observed with the RDMA Ethernet over the convention Ethernet. This modest increase in measured bandwidth is a direct result of the lower latencies shown in Figure 4(b). For latency, InfiniBand again shows the best results, with latencies as low as 6 µs. However, for many small messages, the RDMA Ethernet provides latencies approximately 15 µs less than those observed with conventional Ethernet, with latencies for the RDMA Ethernet dropping as low as 20 µs.

The results from the two-node SendRecv benchmark, shown in Figures 4(c) and 4(d), are similar to those in PingPong. While InfiniBand outperforms the two Ethernet networks, the RDMA Ethernet bridges the gap, which provides more of an advantage over conventional Ethernet than it did in PingPong. RDMA Ethernet latencies are about half of those observed with conventional Ethernet up to 128 bytes, often over 20 µs better. It is important to note that these are MPI latencies, and thus they include the overhead and processing that is associated with MPI.

As mentioned previously, three systems were tested in the Gromacs simulator for application-level experiments. Figure 5 plots the performances of the Villin, LZM, and DPPC tests from Gromacs. Each test spans 5 nanoseconds of simulation time, and the results in the graphs represent the amount of simulated time the cluster can complete in one day. First, for the Villin system, before the system size grows beyond 8 nodes we can see the performance with the RDMA-based Ethernet network is approximately half-way between the performance of InfiniBand and conventional Ethernet. As the system size grows beyond 8 nodes, InfiniBand begins to pull away in performance, and as the system size grows the difference is expected to become more dramatic since the InfiniBand system can perform 50% more simulation steps than the RDMA Ethernet system, and 75% more than the

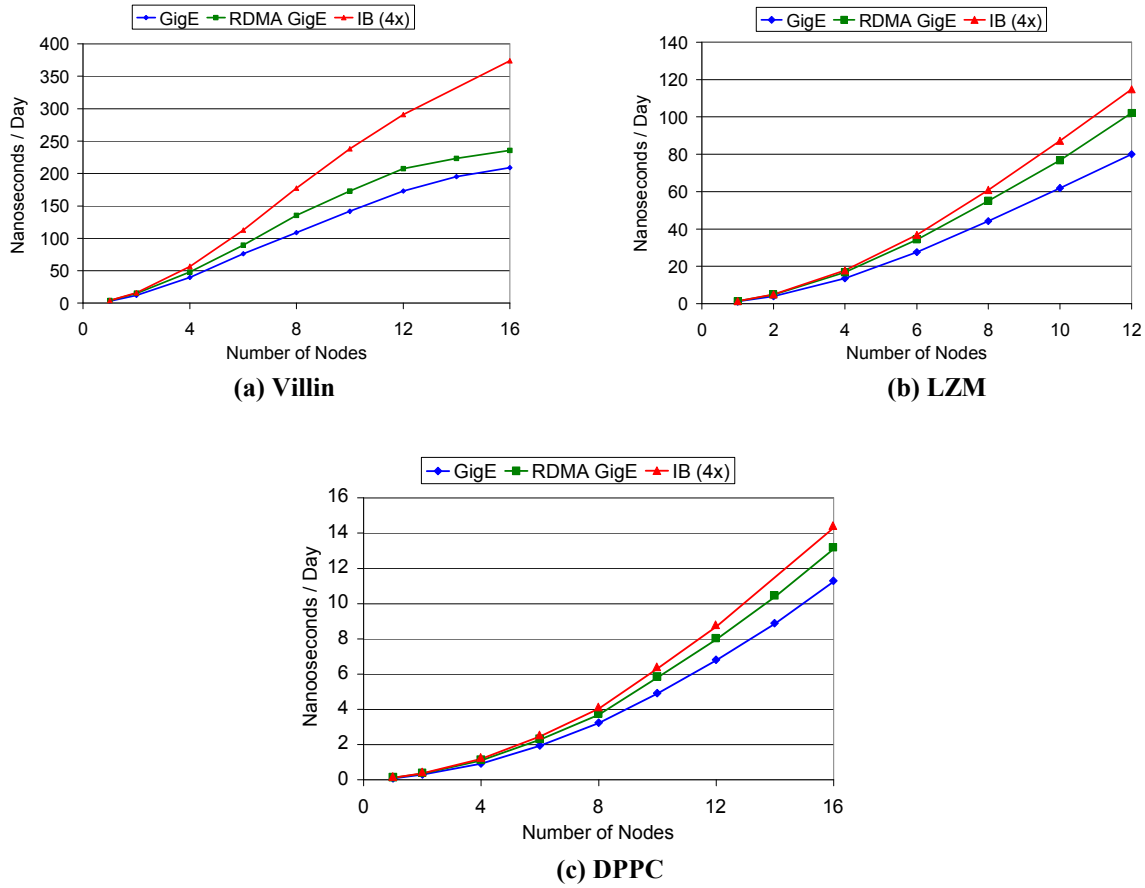


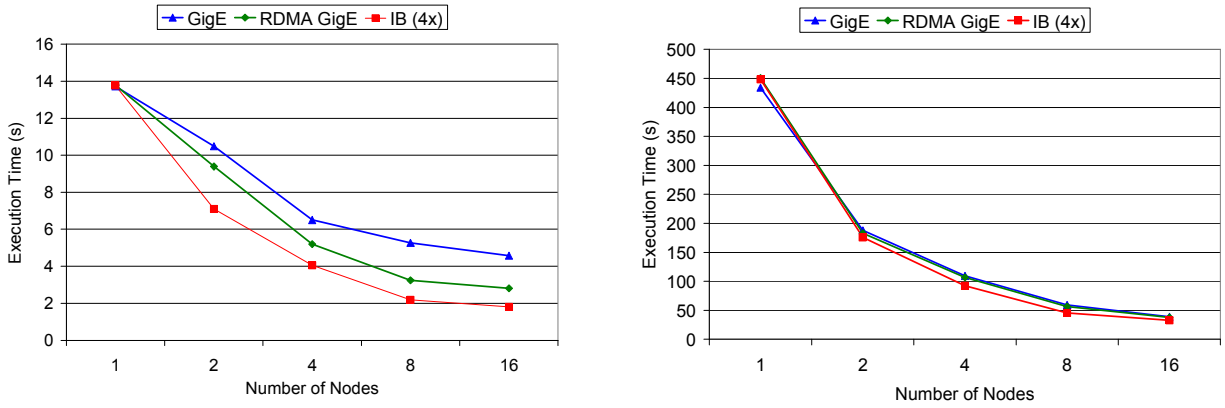
Figure 5: Gromacs Performance Results

conventional Ethernet system in a given period of time. For the RDMA Ethernet system, it increases the performance gap between itself and conventional Ethernet at every system size until 14 nodes is reached.

In the LZM benchmark, the behavior is quite different than in Villin. The LZM system itself is only capable of scaling up to 12 nodes, and thus no results are available for larger node counts. The RDMA Ethernet scales more closely with InfiniBand than it does with conventional Ethernet, even for the larger node counts used. There is barely any noticeable difference in performance between InfiniBand and RDMA Ethernet until the system size reaches 8 nodes. This is not the case with the convention Ethernet network, as it begins to fall off in performance with a size of 4 nodes. By the time the system size reaches 12 nodes, RDMA Ethernet provides over a 25% increase in performance over conventional Ethernet, and slightly over 10% less than InfiniBand.

The third Gromacs system, DPPC, shows less difference in interconnect performance than the previous two. The performance trends are very similar to those seen in the LZM benchmark. For system sizes less than 8 nodes, InfiniBand and RDMA Ethernet perform almost identically, while conventional Ethernet trails in all cases. Even as InfiniBand continues to rise in performance as the node count rises, RDMA Ethernet performs closer to the level of InfiniBand than it does to conventional Ethernet

Finally, two tests from the NPB suite were used as additional application benchmarks. The execution times from the Integer Sort (IS) and Conjugate Gradient (CG) are shown in Figure 6. The Class B data sets of NPB were used in both cases. The IS benchmark appears to be the more network-sensitive of these two benchmarks, as can be seen by the large differences in execution times in Figure 6(a). Applications with high communication and relatively less computation will be greatly affected by the network, and IS provides a perfect example. As the system size grows, RDMA Ethernet provides performance closer to InfiniBand than conventional Ethernet. While CG appears to be much less sensitive to the network environment, there is still a 30% difference in execution time in the 8-node system when comparing InfiniBand and conventional Ethernet. Here, RDMA Ethernet is nearly identical in performance conventional Ethernet.



(a) Integer Sort **(b) Conjugate Gradient**

Figure 6: NPB Performance Results

With all three interconnects under study, superlinear performance was observed for small system sizes. The bulk of the CG benchmark is spent calculating a matrix-vector product, with each element requiring two floating-point operations and three memory accesses [12]. Thus, memory accesses dominate the execution time of the benchmark. Scaling to additional nodes can reduce the average memory access time, as more data points will reside in faster caches. When this speedup outweighs the penalties from communication and parallelization overheads, superlinear performances will be observed.

5. Analysis of Results

Given the results seen in the previous section, it is clear that the choice of network can have a major impact on the performance of the system, particularly for applications sensitive to network latency. InfiniBand proved to perform the best in all circumstances of these network-oriented communication and application experiments, as expected. Meanwhile, the RDMA Ethernet showed varying performance gains in every scenario over conventional Ethernet, and at times it approached the overall performance of InfiniBand.

The raw, low-level results seen with the Pallas benchmarks are very useful in determining how and why the networks perform differently. First of all, any application that relies on high bandwidth will of course perform more favorably on InfiniBand than any Gigabit Ethernet network, since InfiniBand can offer throughput approaching 10 Gb/s. IS from the NPB suite is an example of this characteristic, since almost half of the messages sent during a typical run are over 1 MB in size.

When comparing the throughput of large messages between Gigabit Ethernet and RDMA Gigabit Ethernet, there is little difference. When the PMB PingPong test is expanded to include multi-megabyte messages, the two Ethernet networks are almost identical in terms of measured throughput. At the same time, most messages exchanged during HPC applications are not necessarily several megabytes in size, and many are much smaller, which is why our results focused on the latencies of smaller messages. This is an area where InfiniBand still excels, as expected for a high-performance, system-area network. However, the RDMA Ethernet provides a significant increase in performance in this area over conventional Ethernet. MPI latencies around 20 µs are significantly better than existing Ethernet devices for copper wire. The efficiency and performance gains of RDMA increase even more when bidirectional was considered in the SendRecv benchmark. The services provided to the processor by the network hardware allow the node to more efficiently handle a heavier load of network traffic.

The trends in performances seen in the low-level PMB tests carried over to the application-level experiments. In the latency-sensitive Gromacs system, RDMA Ethernet achieved performance levels almost matching those of InfiniBand in two of the three systems, and well above that of conventional Ethernet. In the Villin system, InfiniBand performs much better than the two Ethernet solutions, and appears to scale better as the system size grows. Of the three Gromacs systems, Villin is the least computationally intensive, while using more frequent system updates, thus more frequent messaging. Increased system sizes also lead to more frequent updates. These factors cause increased network traffic, which allows InfiniBand to benefit from the extra bandwidth it offers, especially as the system size grows. However, the RDMA Ethernet still performs roughly halfway between the other two networks for smaller system sizes.

The fact that the RDMA Ethernet almost matches InfiniBand performance on two of the three Gromacs systems is impressive. Not only is the RDMA Ethernet

competing with InfiniBand in handling the communication loads of these applications, the RDMA hardware is also providing enough processor off-loading to match that of the high-performance InfiniBand hardware. These two factors allow both networks to distance themselves from conventional Ethernet in terms of performance in most of these applications. Beyond the scope of this paper, further research is planned to closely examine the detailed effects of processor offloading by RDMA

The NPB benchmarks gave us an example of how much, and how little, the network can affect performance. For an 8-node system, IS showed the execution time more than doubling when going from InfiniBand to Gigabit Ethernet. Meanwhile, CG is more computationally intensive, and thus the services provided by a high-performance network deliver a smaller impact. The only significant differences in performance are seen by InfiniBand at sizes of 4 and 8 nodes, otherwise performance varies by less than 10%.

All of these applications have shown that while InfiniBand performs the best, RDMA Ethernet can also provide a significant step up in performance over conventional Ethernet. In fact, in some cases, the RDMA will perform at almost the same level as InfiniBand. Having observed this outcome, RDMA Ethernet can be considered a cost-effective alternative to expensive high-performance networks such as InfiniBand for applications sensitive to latency, especially considering the significantly higher cost of InfiniBand HCAs coupled with the cost of the switching fabric for it. So, while InfiniBand is an attractive option for its ability to scale with larger systems, its cost may not make it a viable option to many customers. RDMA Ethernet can be implemented for significantly less with little or no change to the switching fabric. It can use already existing Ethernet infrastructure, and an RNIC like the one described in this paper is expected to sell for a fraction of the cost of an HCA. Therefore, in terms of cost versus performance, RDMA Ethernet promises to be an interesting and in some cases a very attractive solution for HPC networking.

6. Conclusions

RDMA-based Ethernet NICs show promise as a cost-effective network solution for high-performance parallel computing on clusters. An overview of the components and operation of RDMA, including the iWARP protocol, were presented. Experiments were conducted with a first-generation RDMA-based Ethernet card, the AMSO 1100, and compared against InfiniBand and conventional Ethernet network hardware on a 16-node cluster of Linux servers. While, as expected, InfiniBand performed the best in the low-level MPI tests and multiple application suites, the RDMA Ethernet showed significant performance improvement over conventional Ethernet, with some applications approaching the performance of InfiniBand. As performance results from experiments with emerging technologies indicate, RDMA and iWARP offer an attractive technology with significant potential for achieving increasingly high performance at low cost, supporting distributed applications that are sensitive to network performance. Taking advantage of existing Ethernet switching infrastructure makes the RNIC approach even more attractive, an approach that may well evolve into a common commodity among servers in distributed and parallel computing systems. One day in the not-too-distant future, one can foresee RDMA-capable Ethernet being provided by default on all servers and operating with iWARP at 10 Gb/s data rates and more.

7. References

[1] *MPI: A Message Passing Interface Standard v1.1,* The MPI Forum, http://www.mpi-forum.org/docs/mpi-11.ps.

[2] *User-Level Direct Access Transport APIs (uDAPL v1.2)*, uDAPL Homepage, DAT Collaborative. http://www.datcollaborative.org/udapl.html.

[3] T. Talpey, *NFS/RDMA*. IETF NFSv4 Interim WG meeting. June 4, 2003, http://ietf.cnri.reston.va.us/proceedings/03jul/slides/nfsv4-1.pdf.

[4] S. Bailey and T. Talpey, *The Architecture of Direct Data Placement (DDP) and Remote Direct Memory Access (RDMA) on Internet Protocols*, The IETF Internet Report, Feb. 2, 2005, http://ietfreport.isoc.org/ids/draft-ietf-rddp-arch-07.txt.

[5] R. Recio, *RDMA enabled NIC (RNIC) Verbs Overview*, RDMA Consortium, April 29, 2003, http://www.rdmaconsortium.org/home/RNIC_Verbs_Overview2.pdf.

[6] *Ammasso AMSO 1100*, Ammasso Inc., http://www.ammasso.com/Ammasso_1100_Datasheet.pdf.

[7] *Pallas MPI Benchmarks – PMB, Part 1*, Pallas GmbH., http://www.pallas.com.

[8] *Gromacs: The World's Fastest Molecular Dynamics*, Dept. of Biophysical Chemistry, Groningen University, http://www.gromcas.org.

[9] *NAS Parallel Benchmarks*, NASA Advanced Supercomputing Division, http://www.nas.nasa.gov/Software/NPB/.

[10] *LAM/MPI: Enabling Efficient and Productive MPI Development*, University of Indiana at Bloomington, http://www.lam-mpi.org/.

[11] *MVAPICH: MPI for InfiniBand over VAPI Layer*, Networked-Based Computing Lab, Ohio State University, June 2003, http://nowlab-cis.ohio-state.edu/projects/mpi-iba/.

[12] J. Boisseau, L. Carter, K. Gatlin, A. Majumdar, and A. Snavely, *NAS Benchmarks on the Tera MTA*, Proc. of Workshop on Multi-Threaded Execution, Architecture, and Compilers, Las Vegas, NV, February 1-4, 1998.