Evaluation of RDMA Over Ethernet Technology for Building Cost Effective Linux Clusters

Michael Oberg*, Henry M. Tufo*†, Theron Voran*, and Matthew Woitaszek*

* University of Colorado, Boulder
† National Center for Atmospheric Research
theron.voran@colorado.edu

Abstract. Remote Direct Memory Access (RDMA) is an effective technology for reducing system load and improving performance. Recently, Ethernet offerings that exploit RDMA technology have become available that can potentially provide a high-performance fabric for MPI communications at lower cost than other competing technologies. The goal of this paper is to evaluate RDMA over gigabit Ethernet (ROE) as a potential Linux cluster interconnect. We present an overview of current RDMA technology from Ammasso, describe our performance measurements and experiences, and discuss the viability of using ROE in HPC applications. In a series of point-to-point tests, we find that the RDMA interface provides higher throughput and lower latency than legacy gigabit Ethernet. In addition, even when functioning in non-RDMA mode, the ROE cards demonstrate better performance than the motherboard network interfaces. For application benchmarks, including LINPACK and a climate model, the Ammasso cards provide a speedup over standard gigabit Ethernet even in small node configurations.

1 Introduction

Remote Direct Memory Access (RDMA) is quickly becoming a necessity in performance-critical networking. Modern HPC interconnects such as Infiniband, Myrinet, and IBM’s Federation technology make use of RDMA to achieve throughput approaching hardware bandwidth. The use of RDMA in these interconnects reduces the cost of data movement by eliminating redundant copies throughout the network path, and reduces overall resource utilization. Without the use of RDMA methods, network transfers compete with the local system for the available memory bandwidth. For these transfers, each local copy requires traversing the memory bus twice. Typically, data is received from the physical media and copied into a device buffer, then copied into an operating system buffer, then finally copied into the application memory. These multiple memory copies are a large bottleneck in HPC systems especially, because memory speeds have not increased at the same rate as CPU and interconnect speeds. For this reason, with current line rates of 10Gbps and higher, non-RDMA network transfers consume significant amounts of the available memory bandwidth and result in the system CPU(s) stalling on memory accesses.

This situation is compounded by the additional processing requirements for TCP/IP data transfers. Typical TCP/IP network stacks are implemented in the host operating system, which requires the host CPU to perform the necessary per packet processing, checksum operations, and handling of all incoming data through system interrupts. TCP Offload Engines (TOE) have been created to attempt to lessen the impact on the host CPU related to these factors, but these are implemented as separate add-on cards which incurs additional costs and increases cluster complexity. In addition, these devices require the use of specially designed Application Specific Integrated Circuits (ASICs) which keeps costs high, and yet ultimately only provide a marginal benefit [12].

To address these issues, many high-performance clusters use interconnects such as Infiniband, Myrinet, and SCI, which could be classified as custom RDMA systems. While their performance is very high, they can be quite costly and generally require a separate network infrastructure. The cost-effective (and only) alternative so far has been gigabit Ethernet, which has one-quarter to one-eighth the throughput of custom RDMA interconnects and up to two orders of magnitude higher latency. The high latencies of gigabit Ethernet make it particularly ill-suited for fine-grained parallel applications, and the limited throughput restricts its usability for other portions of the cluster infrastructure such as storage. Although gigabit Ethernet can be a relatively poor choice of interconnect for HPC clusters, budget constraints and other project objectives often lead to it being the chosen solution.

RDMA over Ethernet (ROE) provides another option by bridging this gap between the low price but low performance provided by an Ethernet infrastructure, and the higher cost yet higher performing interconnects. By offloading TCP/IP processing to hardware, and performing direct memory access operations without involving the operating system, an ROE adapter provides higher throughput and lower latency than previously available through traditional gigabit Ethernet. The other advantage of ROE is that it leverages the ubiquity of existing gigabit infrastructure. Current datacenters and clusters can therefore support ROE operations in the existing network, without requiring additional bridging or multi-network configurations.

In this paper, we present a performance evaluation of ROE cards from Ammasso, Inc. [1]. The remainder of this paper is organized as follows: Section 2 presents relevant background, describing the Ammasso gigabit Ethernet RDMA technology and its relation to other cluster interconnects. Section 3 describes our testing environment and methodology. Section 4 presents results examining the performance characteristics of the RDMA cards in two-node configurations with PingPong and PingPing tests, and Section 5 presents application benchmark results for a 4-node cluster. The final sections describe future work and conclusions.

2 RDMA Over Ethernet

Since the mid-1990's, Ethernet has been the dominant LAN technology and, more recently, has become a commodity component found in virtually every Linux cluster. Other LAN standards, such as FDDI and token ring, have largely been replaced with Ethernet. The original Beowulf cluster [11] used multiple 10Mbps (1.25MB/s)

Ethernet networks to achieve a level of performance, scalability and cost that demonstrated the feasibility of utilizing these types of clusters for certain classes of applications at dramatically reduced cost. Most current cluster designs use gigabit Ethernet in their design for filesystem and management functions, even if a separate high performing interconnect is purchased for MPI traffic. The high-performance networks, such as SCI, Infiniband, Myrinet, and even 10-Gigabit Ethernet (10GbE), can provide high throughput and low latency but at substantial cost. A 64-bit, 133MHz PCI-X bus is only able to obtain a peak transfer rate of 1066MB/s (8.5Gbps), and thus the 10Gbps interconnects require state of the art technologies to reach their performance potential and to reduce system-induced latencies, such as the new PCI-Express or Hypertransport [4].

Ammasso addresses these limitations by providing RDMA technology over gigabit Ethernet using a custom protocol [3]. The RDMA method is wrapped in TCP/IP packets and sent over Ethernet frames. To avoid the TCP/IP overhead previously discussed, the entire network stack is implemented in hardware. This provides a direct, zero-copy network path between two user-space tasks on separate systems. The Ammasso adapter can communicate with other Ammasso adapters using this proprietary protocol, but also maintains a legacy interface for standard TCP/IP communication. The main advantage to this design is that a single interconnect fabric can serve all the needs of a cluster, multiplexing high performance MPI communication with the typical system and user TCP/IP traffic. This design also aids in datacenter infrastructure, as an Ammasso card can service requests from equally equipped systems at a high rate, while still maintaining compatibility with the remainder of the datacenter and the standard gigabit Ethernet based systems.

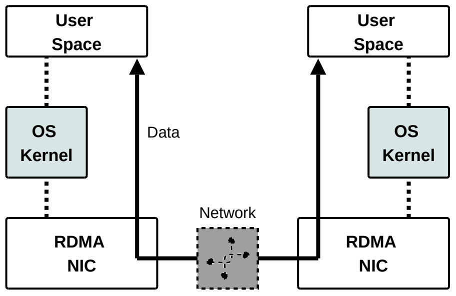

Fig. 1. Data transfers between two RDMA Ethernet adapters

The Ammasso RDMA protocol bypasses the operating system to transfer data between two systems, thereby eliminating the need for local system calls for either of the two processes involved in remote communication (see Fig. 1). This also eliminates the redundant copies typically required by the operating system for a network transfer. The application initiates the transfer and passes the control information through a standard TCP/IP session to the partner process, which sets up a receive buffer in the local application memory space that directly receives the incoming RDMA operation.

The Ammasso hardware functions as a TCP offload engine, implementing a custom TCP/IP stack in hardware. This allows for TCP offload processing, efficient algorithm implementation independent of the operating system, and the possibility of implementing future APIs through firmware updates. Under this hybrid scheme, the legacy interface provided by the Ammasso card is functionally identical to a traditional Ethernet interface, handling the Ethernet layer of the OSI stack, with the remainder of the TCP/IP stack implemented by the operating system. The RDMA interface, on the other hand, is not known to the host operating system as a network interface, and is only available to an application through provided libraries.

3 Testing Setup and Methodology

We performed two series of tests to determine the performance of the Ammasso RDMA Ethernet interfaces in our environment. The first series of tests examined the characteristics of the interconnect, directly linking two identical servers using both a crossover cable and a dedicated commodity gigabit Ethernet switch. The second series of tests were executed on a small (4-node, 8-CPU) cluster using the Ammasso cards and the dedicated commodity switch.

The systems used for this evaluation were three identical IBM x345s, each with dual 3.06GHz Intel Xeon processors and 2.5GB of RAM, and a dual 3.4 GHz Intel Nocona system with 8 GB RAM. All of these systems used Hyper-Threading (HT) to present 4 virtual CPUs in the OS scheduler although only the two physical CPUs were able to handle interrupts. For all systems, the Ammasso NIC used a 64-bit PCI-X 133 slot on a dedicated bus, and the Ethernet NICs were connected to each other via either a crossover cable or a dedicated Dell 5224 24-port gigabit Ethernet switch. The baseline gigabit Ethernet adapter was the ubiquitous built-in Intel 82546EB using the e1000 Linux kernel module. The operating system was SuSE SLES9, running kernel 2.6.5-7.151-smp.

Ammasso provides three RDMA-enabled software layers (DAPL, SDP and MPI), but for these tests only the MPI interface was explored. The lack of availability and maturity of DAPL software tests and utilities made consistent and rigorous testing of this component difficult, and with our interest primarily in the applicability of the Ammasso NIC in HPC applications, we focused on the MPI implementation. For the MPI tests, the Intel MPI Benchmark (formerly known as the Pallas Benchmark or ‘PMB’) was utilized [6]. The primary Pallas MPI tests used for this analysis were the PingPong and PingPing tests. These tests are useful for finding the message startup time and the throughput of a given interconnect. This particular implementation of the PingPong test sends messages from 1 byte to 4MB, increasing by powers of two. For 1 to 32k byte message sizes, the timing result was the average of 1000 messages; for 64k and above the number of messages sent is such that total message volume is held constant at 4MB. The PingPing test is similar, except each process posts an MPI_Isend, then each receives the data simultaneously with MPI_Recv. Thus PingPing measures performance in the presence of network contention. These two schemes provide an accurate measure of the interconnect characteristics in the short byte ranges, and sufficient runs were taken for each configuration to give confidence that temporal and unrelated system effects were not skewing the results.

The utility we used to determine the system resources necessary to drive the Ethernet interfaces was *mpstat*. This utility is part of the *sysstat* package, which is part of the *sar* suite of monitoring utilities [10]. The base interrupt load is the interrupts per second from the 'timer' interrupt, which occurs at the frequency specified in `include/asm-i386/params.h`, which for our systems was 1000 Hz. In this paper, all reported statistics for the interrupts per second include this base 1000 interrupts per second, and therefore reflect the total interrupt load on the system.

For the system resource utilization tests, we ran an initial set of 5 runs. From these, we determined the total number of runs required to give an estimation of the mean within a 95% confidence interval. All rates are averaged using a harmonic mean, and all times and percentages were calculated using an arithmetic mean. Unless otherwise noted, latencies are reported for the 0-byte message size as calculated by the benchmark. The measurements for the built-in Intel NIC varied much more than the Ammasso interfaces, and therefore we had to run approximately 4x as many tests on the built-in NIC as for the Ammasso interfaces in order to achieve a 95% confidence level in these means.

4 Simple Network Performance Test Results

For our primary analysis, we compare the throughput, latency, CPU load, and interrupt rate for the three interfaces: Ammasso RDMA, Ammasso legacy, and built-in Intel. The RDMA interface performed the best, in both latency and throughput, but the high CPU interrupt load due to the polling implementation may impact CPU-intensive applications.

The Ammasso RDMA interface provided the lowest 0-byte message latency, followed by the Ammasso legacy interface, and then the motherboard Ethernet controller (see Fig. 2 and Fig. 4). Similarly, the Ammasso RDMA interface provided the highest effective throughput, followed by the Ammasso legacy and the motherboard controllers (see Fig. 3 and Fig. 5). In addition to these tests with the standard 1500-byte MTU, we also tried testing 9000-byte MTUs, but all three interfaces performed poorly. Other evaluations of network interconnects have reported similar results where larger MTUs decrease throughput [9]. Ammasso reported that this was an unsupported configuration, so we do not report results for the larger MTUs.

The higher throughput and lower latency, however, comes at a potential cost. Use of the Ammasso RDMA interface consumes a higher percentage of available CPU cycles, followed by the legacy and built-in interfaces (see Fig. 6). The Ammasso legacy interface generates the most interrupts per second, dwarfing both the built-in and RDMA interfaces. It should also be noted that the Ammasso cards were 100% stable during our testing using both the legacy and RDMA interfaces. A detailed description of the results for each interface is provided below.

4.1 Interface Performance Results

The built-in Intel 82546EB NIC on the IBM x345's represents the baseline performance for a typical motherboard-mounted gigabit Ethernet interface. For the PingPong test, the built-in interface achieved a 0-byte latency of 86.99 µsec with the crossover cable and 85.85 µsec with the switch. The throughput achieved for the 4MB message size over the crossover cable was 50.39 MB/s, and was 45.07 MB/s when run through the switch.

The Ammasso legacy interface achieved a 0-byte latency of 38.45 µsec and 66.79 MB/sec throughput using a crossover cable, and 44.90 µsec 0-byte latency and 66.40 MB/s throughput using the switch.

The Ammasso RDMA interface demonstrated impressive performance in both latency and throughput, providing a one-way latency of 17.62 µsec on the PingPong test with the crossover cable and 23.11 µsec with the switch. The throughput reached 110.21 MB/s over the crossover cable, and 110.07 MB/s over the switch.

The PingPing results were very similar to the PingPong results in the relative performance of the interfaces. There was noticeable degradation in the throughput for the built-in Intel and Ammasso Legacy interfaces, which is about half of the throughput achieved with each interface in the PingPong test. The throughput of the Ammasso RDMA card degraded about 20% with PingPing.

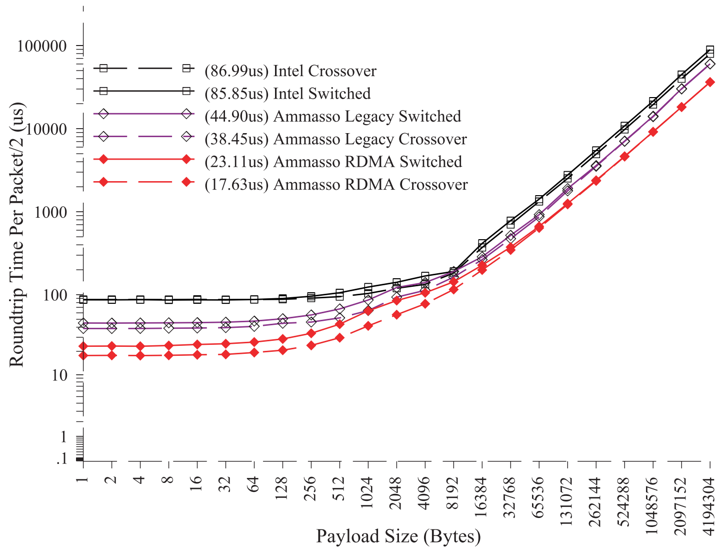

Fig. 2. Latency measured with the Pallas PingPong test for the RDMA, legacy, and built-in interfaces in both crossover and switch configurations for 1500-byte MTU, with HT enabled

4.2 System Interrupt and CPU Utilization

The Ammasso legacy interface, which provides higher throughput and lower latency than the built-in standard gigabit Ethernet interface in the x345s, uses less CPU load but generates far more interrupts/sec than the two other interfaces at an observed 44,432.68 interrupts/sec.

The observed interrupt rate with the RDMA interface is not significantly different that the base interrupt load of 1000 interrupts/sec, so the data verifies the assertion that the RDMA interface does not generate interrupts, which was expected due to the OS bypass nature of the RDMA design. However, the RDMA cards generated the highest CPU load of the three interfaces tested during network transfers, saturating an entire CPU on the node. This high CPU utilization was to some degree expected, as the RDMA MPI interface is tuned for low latency [3], and the library routinely polls to determine if data is available to achieve such low latency.

Using a simple C MPI benchmark, we found that the MPI library generates high CPU load only during blocking MPI_Recv operations. Thus, the use of the RDMA MPI library should not reduce the number of cycles available to the application, as the increased load is introduced only during stalls for blocking communications. Since this is implemented in the MPI library that is linked into the application, the increased load allows the program to continue as soon as the data is available. Thus, the impact of this CPU load is isolated to the portions of the application doing MPI communication, and in tightly coupled applications would be offset by the gains from the improved latencies. According to Ammasso, it would be possible to build a DAPL interface to the RDMA card that could be tuned to provide the latency and CPU utilization balance desired, and would still have the same throughput characteristics as the current RDMA interface.

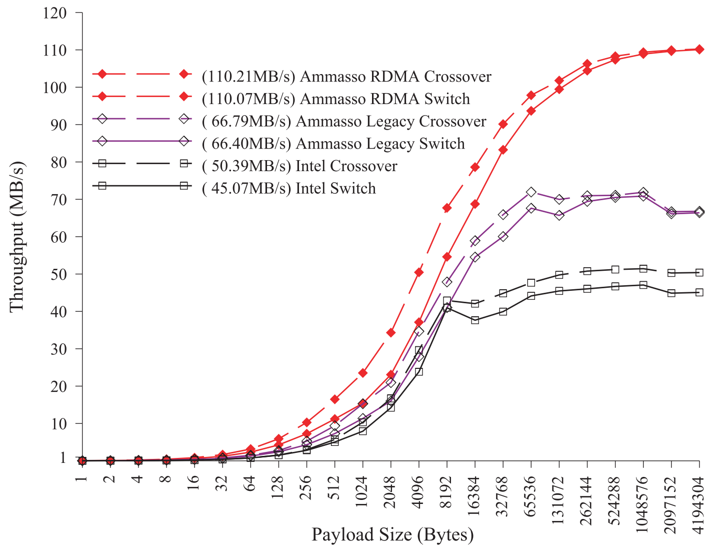

Fig. 3. Throughput with the Pallas PingPong test for the RDMA, legacy, and built-in interfaces in both crossover and switch configurations for 1500-byte MTU, with HT enabled

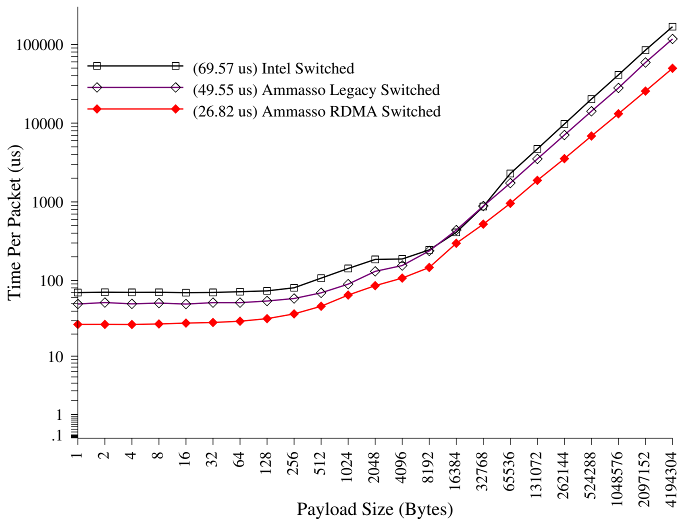

Fig. 4. Latency measured with the Pallas PingPing test for the RDMA, legacy, and built-in interfaces with switch configurations for 1500-byte MTU, with HT enabled

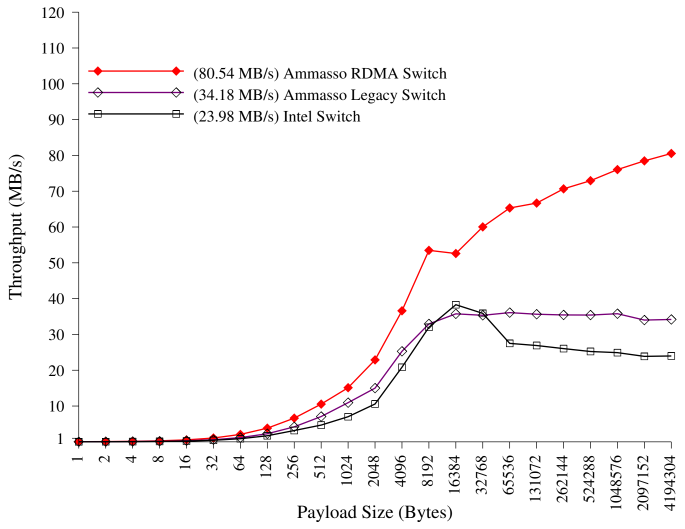

Fig. 5. Throughput with the Pallas PingPing test for the RDMA, legacy, and built-in interfaces in both crossover and switch configurations for 1500-byte MTU, with HT enabled

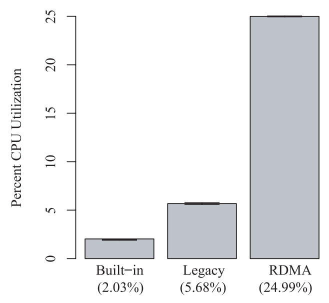

Fig. 6. 1500 byte MTU CPU utilization with 95% confidence interval bars. Note: with HT enabled, 25% is 100% of a single physical CPU.

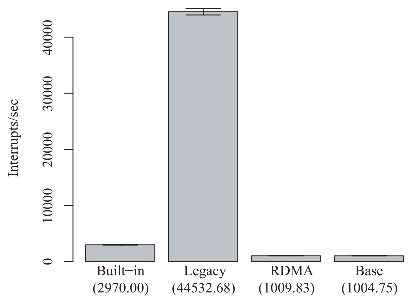

Fig. 7. Interrupts per second for 1500 Byte MTU with 95% confidence interval bars.

5 Application Benchmark Results

In addition to the simple point-to-point tests used to examine the network characteristics, we also ran two application benchmarks to determine the effect of the increased throughput and lower latency on real MPI programs. On the 4-node, 8-CPU cluster described previously, we ran the LINPACK benchmark employed by Top500.org to rank supercomputing systems worldwide and the HOMME climate model developed by the Computational Science Section at the National Center for Atmospheric Research.

5.1 LINPACK

To determine the effect of the interconnect on the LINPACK benchmark [5], we ran LINPACK on all three interconnects and identified the test configurations that provided the best results for the Ammasso RDMA and legacy interfaces. We then examined the behavior of all three interconnects for those cases. The RDMA version had the best performance when running LINPACK test WR00C2C2 with 8 blocks on a 4x2 distribution, and the Ammasso legacy interface had the best performance when running WR00C2L2 (also with 8 blocks on a 4x2 distribution). For these two cases, we ran LINPACK five times each using all three interconnects, computed average execution time, and calculated speedup compared to the built-in motherboard interface. In both cases, both RDMA and the Ammasso legacy interface provided noticeable speedup between 1.16 and 1.48 (see Table 1).

Table 1. LINPACK speedup over built-in motherboard by interconnect, with HT enabled

Case	Configuration	Ammasso Legacy	Ammasso RDMA
I	WR00C2C2 6000 8 4 2	1.30	1.48
II	WR00C2L2 6000 8 4 2	1.16	1.27
III	WR00R2C2 6000 8 4 2	1.21	1.22

For the LINPACK benchmark, an unfavorable partitioning or solution method destroys any benefit provided by the network. In these cases, all three interconnects fared poorly. These bad configurations usually required over 50 seconds to complete as opposed to the 20-30 seconds for our system's best results. However, for the solution methods which produced the cluster's best runs, the difference between the three interfaces was obvious and impressive. These results are similar to those shown by another research group working with Ammasso to benchmark their applications [7].

5.2 HOMME

To observe the real-world speedup on a representative application, we ran a series of tests using the HOMME application on all three interconnects, with and without HT enabled on the nodes. The High Order Method Modeling Environment (HOMME) is a global atmospheric modeling framework developed at the National Center for Atmospheric Research (NCAR). We used the NSF 05-625 benchmark [8] variation of HOMME, which runs a Baroclinic instability simulation. HOMME was chosen in this investigation for its portability and scalability, proven on platforms such as IBM's Blue Gene/L.

We ran the HOMME benchmark using a variation of the "tiny" test case, running the Baroclinic instability simulation with 216 elements (N_e=6), 64 points per element (N_p=8), and 96 vertical levels for 200 time-steps. We gathered timing data for 4 and 8 processor runs, with 5 trials for each. The average time per time-step was used to compute the relative speedup to the motherboard NIC, as listed in Table 2.

Table 2. HOMME speedup over built-in motherboard by interconnect, with and without HT enabled

	Processors	Ammasso Legacy	Ammasso RDMA
With HT	4	1.065	1.403
	8	1.072	1.541
Without HT	4	1.044	1.360
	8	0.985	1.228

HOMME requires "nearest neighbor" boundary exchanges during each iteration and our tests showed that communication accounted for approximately 33-50% of the iteration time according to the code’s internal timing, depending on the combination of interface and number of processes per node. The Ammasso RDMA interface showed good speedup compared to the motherboard and Ammasso Legacy adapters, up to 1.541 with 8 processors (2 per node) with HT enabled. Interestingly enough, the RDMA performance without HT enabled is worse than with HT enabled, presumably because there are extra hardware threads available to deal with polling for the RDMA driver. However, performance was slightly better without HT enabled for the built-in motherboard and Ammasso legacy interfaces when looking at time per time-step of HOMME.

6 Analysis of Results / Recommendations

The Ammasso RDMA adapter demonstrated better throughput and latency performance than the typical non-RDMA gigabit adapters in a commodity Ethernet environment. In addition to the surprising performance gains over Ethernet provided by the RMDA interface, one unanticipated aspect of the results is the performance of the legacy interface. We expected the performance of this interface to be similar to the performance of other gigabit cards, assuming that the hardware performance had already been optimized and the limitations remained in the OS and low-level x86 system architecture. These results show that a highly tuned hardware implementation of a gigabit NIC can outperform standard adapters, and that RDMA methods can drastically improve the utilization of an Ethernet fabric.

The current Ammasso MPI design and implementation is suitable for cluster applications, and could be considered as a low-cost option for any cluster designed around gigabit Ethernet. By bridging the price and performance gap between standard gigabit Ethernet and the high-performance interconnects, Ammasso's RDMA Ethernet implementation is a viable alternative implementation that provides high performance while exploiting the commodity Ethernet infrastructure already deployed in today’s datacenters.

Other groups have examined the behavior of RDMA (using the Ammasso technology) over a wide area, using latency-inducing software routers to simulate the behavior of a wide area network. Their results show that even with the increased latencies induced by the wide-area fabric, the RDMA operations were beneficial in reducing CPU utilization, increasing overall throughput, and providing resiliency to heavy load conditions [13].

Moreover, we expect that the Ammasso cards will be useful in other applications, such as filesystems and servers, especially if the vendor introduces support for larger MTU sizes. Many system-level applications can benefit from reduced latency, especially NFS and parallel filesystems, where lock management often requires sending many small messages between large numbers of nodes.

7 Future Work

We would like to test scalability using larger clusters and examine the Ammasso technology for datacenter applications. The Ammasso road-map literature describes upcoming features such as support for larger MTU sizes that may improve the performance of common services such as NFS. As this technology becomes available, we would like to evaluate it with respect to our datacenter environment. In addition, we would like to examine the Sockets Direct Protocol (SDP) and DAPL interfaces to the RDMA cards. These APIs should offer additional application improvements, allowing system services to utilize the underlying RDMA protocols and the RDMA mode of the interface, improving latencies and throughput across a wider range of system software.

8 Acknowledgements

We would like to thank Michael Barbour of Ammasso, Inc., and Dirk Grunwald and Sean McCreary at the University of Colorado, Boulder, for their assistance in this investigation. University of Colorado computer time was provided by equipment purchased under NSF ARI Grant #CDA-9601817, NSF sponsorship of the National Center for Atmospheric Research, and a grant from the IBM Shared University Research (SUR) program.

References

1. Ammasso, Inc., “About Ammasso”. http://www.ammasso.com/about.htm
2. Ammasso, Inc., “The Ammasso 1100 High Performance Ethernet Adapter: User's Guide”, Revision 1.1a, 2004. http://www.ammasso.com/Ammasso_1100_UserGuide.pdf
3. Ammasso, Inc., “Understanding Remote Direct Memory Access (RDMA)”, Whitepaper. http://ammasso.com/Ammasso_RDMA_Whitepaper_05-14-2004.pdf
4. C. Bell and D. Bonachea and Y. Cote and J. Duell and P. Hargrove and P. Husbands and C. Iancu and M. Welcome and K. Yelick, “An Evaluation of Current High-performance Networks”, IPDPS, 2003. http://citeseer.ist.psu.edu/bell03evaluation.html
5. HPL - A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers, http://www.netlib.org/benchmark/hpl/
6. Intel MPI Benchmarks (formerly Pallas Bemchmark or ‘PMB’), http://www.intel.com/software/products/cluster/clustertoolkit/features.htm#mpi_benchmarks
7. Jacobs, C. Reardon, A. George, “Performance Evaluation of the Ammasso 1100 Low-Latency Gigabit Ethernet Adapter, Whitepaper. http://ammasso.com/Ufl_ammasso_paper.pdf
8. NSF Program Solicitation 05-625. “High Performance Computing System Acquisition: Torwards a Petascale Computing Environment for Science and Engineering.” http://www.nsf.gov/pubs/2005/nsf05625/nsf05625.htm
9. P. Gray and A. Betz, “Performance Evaluation of Copper-based Gigabit Ethernet Interfaces”, Proceedings of the 27th Annual IEEE Conference on Local Computer Networks, 2002.

10. SYSSTAT utilities, http://perso.wanadoo.fr/sebastien.godard/
11. T. Sterling and D. Savarese and D. J. Becker and J. E. Dorband and U. A. Ranawake and C. V. Packer, “Beowulf: A Parallel Workstation for Scientific Computation”, Proceedings of the 24th International Conference on Parallel Processing, Oconomowoc, WI, I:11-14, 1995. http://citeseer.ist.psu.edu/sterling95beowulf.html
12. P. Sarkar, S. Uttamchandani, and K. Voruganti, Storage over IP: Does hardware support help? Proceedings of the 2nd USENIX Conference on File and Storage Technologies, pages 231–244, San Francisco, CA, March 2003.
13. J. Hyun-Wook, S. Narravula, G. Brown, K. Vaidyanathan, P. Balaji, and D. K. Panda, Performance Evaluation of RDMA over IP: A Case Study with the Ammasso Gigabit Ethernet NIC, High Performance Interconnects for Distributed Computing, Raleigh, NC, July 2005