Performance Analysis of the SiCortex SC072

Brian J. Martin, Andrew J. Leiker, James H. Laros III and Doug W. Doerfler
Sandia National Laboratories
Albuquerque, NM

Abstract

The world of High Performance Computing (HPC) has seen a major shift towards commodity clusters in the last 10 years. A new company, SiCortex, has set out to break this trend. They have created what they claim to be a balanced cluster which makes use of low-power MIPS processors and a custom interconnect in an effort to avoid many of the bottlenecks plaguing most modern clusters. In this paper, we reveal the results of preliminary benchmarking of one of their systems, the SC072. First, we ran a collection of microbenchmarks to characterize the performance of interprocessor communication. Next, we ran some real applications relevant to high performance computing and compared performance and scalability to a typical commodity cluster. Lastly, we examine and compare the performance per watt of the SiCortex system to a commodity cluster.

1 Introduction

Recently, the most popular high performance computing solution has been the commodity cluster, which employs a large number of commodity processors linked together with a commercially available interconnect. This trend has largely been fed by high performance, low priced processors available for the personal computing market, as well as the advancement of a variety of open source software. SiCortex [5], a relatively new entrant in the HPC market, recently introduced a line of all-in-one clusters. They seek to avoid the inefficiencies that arise from clustering a large number of commodity parts not built for high performance computing. SiCortex claims to be the first company to engineer a cluster "from the silicon up" to create a balanced system, balancing processor speed with power consumption and communication speed in order to maximize application performance per dollar, per watt, and per square foot [11]. We benchmarked the smallest system in the lineup, the SC072("Catapult"), a 72 processor machine with the form factor of a typical desktop tower. Although the SC072 is not their largest cluster, it is representative of their unique architecture and design philosophies. In this paper we analyze the performance of the SC072 on a series of micro-benchmarks and compare the application performance and scalability of the SC072 with a typical commodity cluster. To characterize the communication performance of the interconnect, we ran a series of microbenchmarks, both from the Pallas suite and Sandia. In addition, we employed several applications to reveal performance and scalability of the SC072 and contrast between the SiCortex system and a commodity cluster. We analyzed the data gathered from these applications in several different ways, including an examination of the claims made by SiCortex regarding better performance per watt compared to a typical commodity cluster. Section 2 outlines previous work done in this area. Section 3 discusses the architecture of the SC072 and what makes it unique as well as contrasts it with a commodity cluster. A discussion of the microbenchmarking tools used takes place in section 4, and a review of the applications used to measure performance and scalability is covered in section 5. Results are presented and analyzed in sections 6 and 7, and a look at the performance per watt takes place in section 8. Our conclusions based on the data collected are presented in section 9.

2 Related Work

Preliminary analysis of the SiCortex systems has mainly been performed by the engineers at SiCortex, due in part to their recent entrance into the HPC market. Publications and technical summaries provided by SiCortex can be found in a series of white papers [6]. In addition, analysis of the zero-copy remote direct memory access (RDMA) implementation has been done by SiCortex through the use of HPCC RandomRing and Pingpong benchmarks [13]. Other than the RDMA analysis done by SiCortex and a few published white papers, analysis of the SiCortex systems has been largely non-existent. Therefore, this paper strives to provide an accurate and unbiased performance analysis of the SiCortex systems.

Table 1: Test Platform Summary

System	**SiCortex SC072**	**Generic Commodity Cluster**
Processor	500 MHz MIPS64	2.2 GHz AMD Opteron Processor x86-64
Single Core Peak Floating Point Rate	1 GFLOPS	4.4 GFLOPS
Interconnect	Custom	Myricom Myri10G
Interconnect Topology	Degree-3 Kautz Graph	Clos
Compiler	PathScale version 3	Gnu Compiler Collection 3.4.3
Power Consumption per Socket	15 Watts	85.3 Watts
MPI Implementation	MPICH2	MPICH-MX

3 Architecture

The SiCortex SC072 resides in a desk-side case, plugs into a typical 100-120V electrical outlet, and draws less than 300W of power. On the inside, however, it houses twelve compute nodes, each of which is a six-way symmetric multiprocessor, containing six low-power 500 MHz 64-bit MIPS® processor cores. Each core has a peak double precision floating-point rate of 1GFLOPS, giving the entire system 72 GFLOPS of peak performance [11]. To support these processors, the system houses 48GB of memory. One of the defining features of the SiCortex line of clusters is their unique interconnect [12]. The fabric topology in the SiCortex is a unique system based on a degree-3 directed Kautz graph [10]. This Kautz topology means that the diameter of the network grows logarithmically with the number of nodes even as the degree of the network remains fixed. The fabric links can support large message bandwidth of 2GBytes/second, and since there are a total of six fabric links per node, three exit links and three entrance links, bandwidth between nodes is up to 6Gbytes/second. The SC072 runs a custom build of Linux on each of it's compute nodes, placing it on equal footing with most commodity clusters in the availability of many open source software solutions and expertise with the system. For message passing, the SiCortex includes a custom message passing interface (MPI) implementation which is based on MPICH2 and optimized for the architecture. RDMA protocol takes effect at message sizes greater than 1024 bytes, and an MPI send-receive implementation is used for message transfers below 1024 bytes. The RDMA is essentially implemented through the DMA Engine; it is one of three interconnect components. SiCortex includes several compilers available for use on their system, including the PathScale and GNU compiler suites, both containing C, C++ and FORTRAN compilers. All applications in this paper used the PathScale compiler suite, as it is optimized to the SiCortex architecture. The resource management was taken care of on the SC072 through use of the Simple Linux Utility for Resource Management (SLURM) with the default production settings.

The cluster we used in comparison to the SiCortex is a typical commodity cluster which uses 256 2.2 GHz AMD Opteron processors linked together by a Myrinet network in a Clos topology. For all of the data gathered, this cluster was limited to 72 cores or less in order to provide a core-to-core comparison between the two systems. The job management on this cluster was handled by the Portable Batch System(PBS) in a production environment.

4 Microbenchmark Overview

In our analysis of the SiCortex system, several microbenchmarks were used to characterize SiCortex's unique communication system. Of the microbenchmarks, two were developed at Sandia National Laboratories and three were obtained from the Pallas Microbenchmark tool suite. A brief description of each is presented below.

4.1 Pallas Microbenchmarks

The Pallas microbenchmarks (version 2.2.1) are a suite of tools capable of characterizing the message passing interconnect on high performance computers. They include point-to-point, collective, and parallel transfer benchmarks. We utilized the pingpong, allreduce, and sendrecv benchmarks as they are used extensively in Sandia applications. Performance analysis for point-to-point communications was accomplished with the pingpong microbenchmark, which measures the startup and throughput as a message is sent between two processors. MPI_Send() and MPI_Recv(), blocking communication functions, form the bulk of the pingpong microbenchmark. The allreduce microbenchmark is a collective benchmark that measures the average time to reduce a vector with MPI_Allreduce(). Lastly, the sendrecv tool was chosen as the parallel transfer benchmark. A chain like communication pattern and MPI_Sendrecv() are the underpinnings of the sendrecv tools analysis capabilities. Its bidirectional bandwidth results characterize the interconnects bandwidth capabilities for communication intensive work.

4.2 Sandia Microbenchmarks

Analysis performed on the SiCortex communication system with Sandia based microbenchmarks was done through the use of the Host Processor Overhead (HPO) analysis program [1] and a modified streaming analysis program; both benchmarks were originally developed at Sandia. The HPO microbenchmark provides a picture of the total overhead and application availability on a single processor while communication is taking place. Overhead is the total processor time spent on MPI related tasks during communication. Application availability, on the other hand, is the percentage of time available for computational work during communication. As noted in citation [8], high application availability and low overhead can remedy the negative affects of a high latency, low bandwidth interconnect. MPI_Isend() and MPI_Irecv(), both non-blocking, allow communication and application work to overlap in the HPO analysis, thereby producing a realistic estimate of MPI related overhead for the send and receive calls. The final microbenchmark used for analysis was the streaming bidirectional microbenchmark. It is much like the sendrecv benchmark; however, it characterizes the interconnect somewhat differently and gives a much better estimate of bisection bandwidth. It utilizes the MPI_Sendrecv() and floods the fabric links with messages between processes for one second. Bisection bandwidth is then found based upon the total number of bytes sent in that time period.

5 Application Overview

For performance analysis and scalability, three applications were used to compare the SiCortex to a generic commodity cluster. They are described below.

5.1 HPCCG: Simple Conjugate Gradient Benchmark Code

The HPCCG micro-application is a simple partial differential equation solver and preconditioned conjugate gradient solver that solves a linear system on a beam-shaped domain. It generates a 27-point finite difference matrix for a 3D chimney domain on an arbitrary number of processors. This open source software was designed to be scalable up to thousands of processors, making it a sound software choice for analyzing scalability of a system. This software was designed to be a weak scaling code, meaning that, given the same input, the problem size doubles as the number of processors doubles. Benchmarking data in this paper was taken with version 0.5 of HPCCG, using the reported total MFLOPS from the output of the program. HPCCG is licensed under the GNU LGPL [2].

5.2 phdMesh

phdMesh is a micro-application designed to perform parallel geometric proximity search, dynamic load balancing, parallel synchronization, and other common operations on parallel, heterogeneous and dynamic unstructured meshes. The data analyzed was taken from the amount of time that the program spent performing the parallel geometric search per step [4].

5.3 LAMMPS

LAMMPS is an open source molecular dynamics simulator available under the GNU general public license. The May 21, 2008 release of LAMMPS is the version being used for our analysis [3]. Within the LAMMPS package, atomic and molecular models constitute the principal scientific tools; however, the package also has application benchmarks incorporated. The application benchmarks originate from the models themselves. As the models scale linearly, the benchmarks prove to be excellent scaling analysis tools for high performance computers. Of the five application benchmarks available for performance analysis, we chose two, the Lennard Jones liquid benchmark and the Rhodospin Protein benchmark. The two benchmarks were chosen for the dissimilarity in simulation methods and the difference in computational expense, as Rhodospin Protein is more computational and communication intensive. Both benchmarks allow for weak and strong scaling. The combination of benchmark and scaling type provide various pictures of the systems scalability, such as characteristics of the system's communication or computation scalability. Or more importantly, the overall balance of the system's scaling.

6 Microbenchmark Results

SiCortex microbenchmark results were attained through the treatment of the SiCortex cluster as a non-production environment, allowing microbenchmarking to take place without outside influence. In addition, all core allocations were

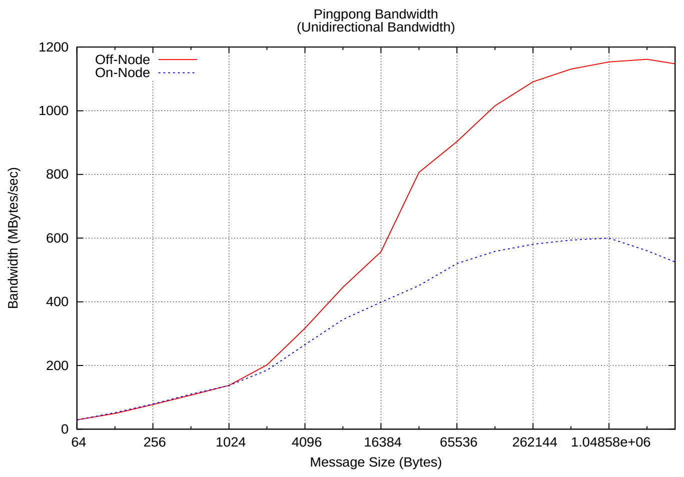

Figure 1: Pingpong Microbenchmark Bandwidth

handled by the SiCortex default implementation of SLURM, and all results are the average of multiple message transmissions.

6.1 Pingpong Results

Latency and bandwidth data are presented for both on and off node two processor core allocations. Both the on and off node results correlate with previous work published by SiCortex. However, previous work done by SiCortex exhibited erratic behavior at a message size of 64kB, see citation [13]; however, our results indicate SiCortex improved 64kB characteristics. For a message size of 1024 bytes, figures 6.1 and 6.1 demonstrate increased bandwidth performance and decreased latency. Performance changes at 1024 bytes are common among other communication systems, but the MPI to RDMA protocol change at this stage warrants inquiry as to its effect on the SiCortex's performance. Up to a 512 byte message size, latency values are under five microseconds, which is considerably good. Finally, on-node bandwidth performance is lower than off-node bandwidth performance; this can be attributed to the overload of on-node system memory due to the significant number of reads and writes, as noted in citation [13].

6.2 Sendrecv Results

The sendrecv benchmark was run both on and off node. Off-node jobs ranged from two to twelve nodes, where on-node jobs ranged from two to six cores. Fig-

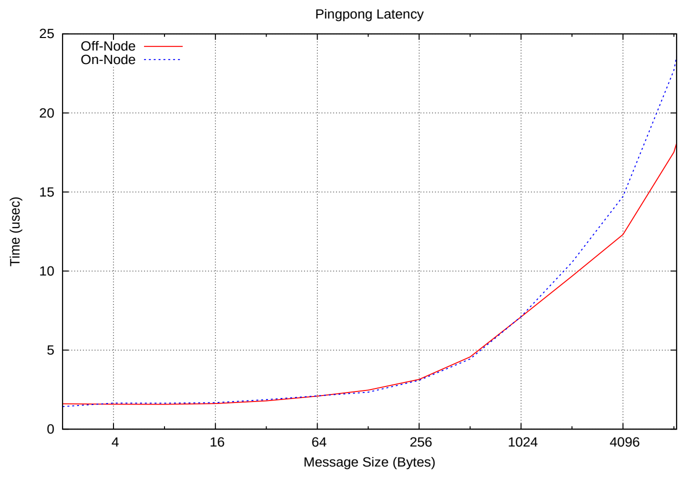

Figure 2: Pingpong Microbenchmark Latency

ure 6.2 shows the results obtained from the two and twelve off-node jobs. Other node allocation results fall somewhere in-between these two lines. For all off-node allocation sizes, the distribution of messages across multiple fabric-links, rather than one, brought about an increase in performance at a message size greater than 64kB. Boosts in performance at this stage can be directly attributed to MPI protocol change; messages greater than 64kB are spread across fabric links in partitions of 64kB, while messages 64 kB and smaller are passed on one fabric-link [13]. Interestingly, achievement of maximum bidirectional bandwidth for off-node jobs could only be obtained with message sizes greater than 4 MB. Lastly, figure 6.2 reveals an increase in the number of cores for an on-node job produces a predictable reduction in bidirectional bandwidth.

6.3 Allreduce Results

Allreduce results are presented for power-of-two core allocations up to 64 cores, along with a 40 core job. Power-of-two core allocations exhibited excellent performance, while non-power-of-two core allocations did not. For example, the 40 core allocation performed 17000 microseconds slower than a 64 core allocation at a vector reduction size of 16 MBytes. The prime factor for this deviation is the inherent dependence of collective communication algorithms on power-of-two allocations. Consequently, core allocation size plays a key role in performance and a smaller job size doesn't necessarily signify optimum timings, as the 40 core allocation demonstrates. Figure 5 demonstrates the power-of-two dependence

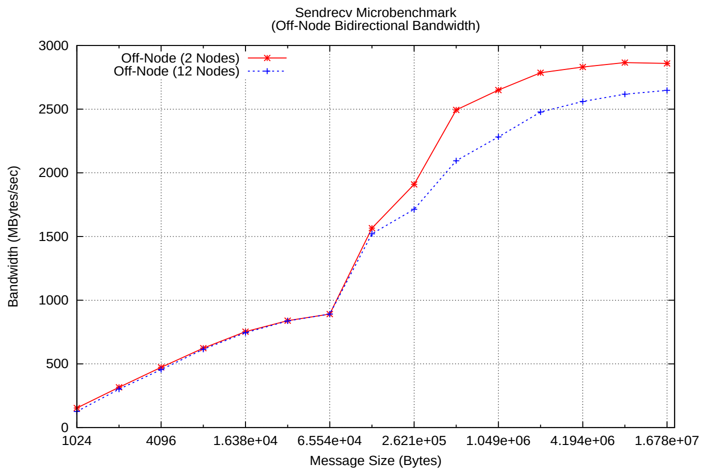

Figure 3: Sendrecv Microbenchmark Off-Node Performance

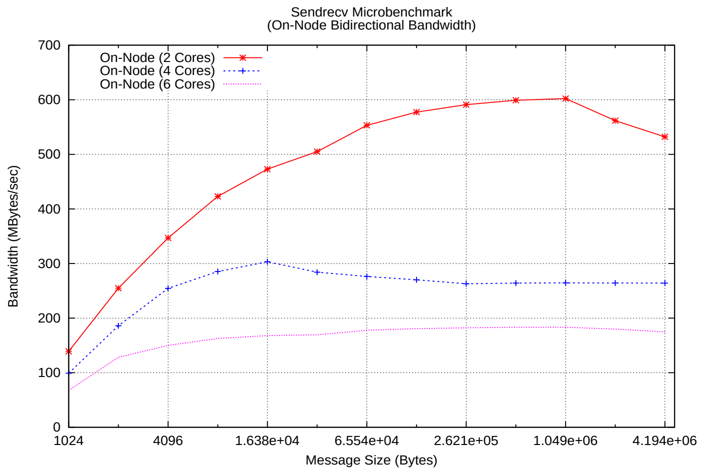

Figure 4: Sendrecv Microbenchmark On-Node Performance

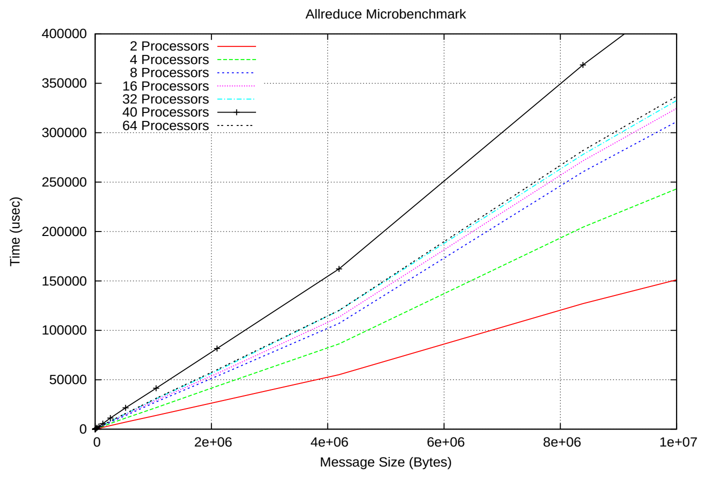

Figure 5: Allreduce Microbenchmark Results

for collective communications on the Sicortex interconnect. Finally, increased performance with greater core allocations indicates the SiCortex's scalability for communication intensive programs.

6.4 Host Processor Overhead

Overhead and application availability results were obtained for both an MPI Send and Receive function calls. Overhead data shown in figure 6.4, for both function calls, behaves linearly. In comparison to other systems, overhead performance on the Sicortex system is marginal at best [9]. Application availability for MPI_Isend in figure 6.4 exhibits erratic behavior, including a plunge at 1024 Bytes and a peak availability, 94.4%, at 512 KBytes. The unpredictable nature of the MPI_Isend application availability induced questions regarding the validity of the results; however, our analysis was performed multiple times and deviations were so small that results proved to be valid. Furthermore, a dependence on the RDMA protocol change is present at 1024 Bytes. In contrast to MPI_Isend, application availability for MPI_Irecv is unfavorable for all message sizes. Overall, high overhead and low application availability are common for message transfers greater than 1 MB, but performance yields below 1 MB are generally only good for the MPI_Isend function.

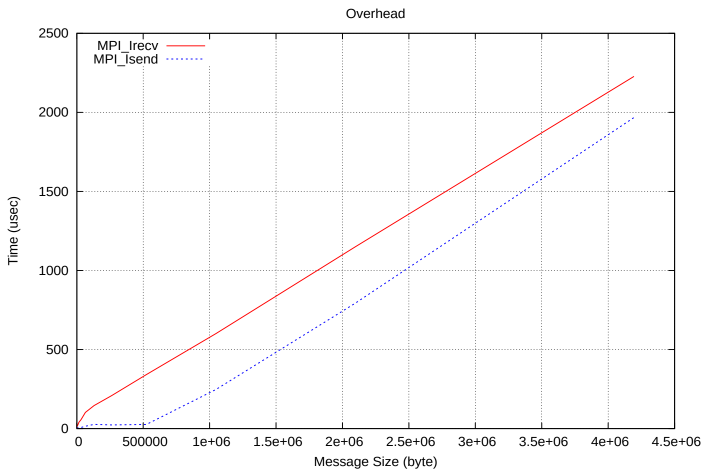

Figure 6: Host Processor Overhead Microbenchmark Results: Overhead

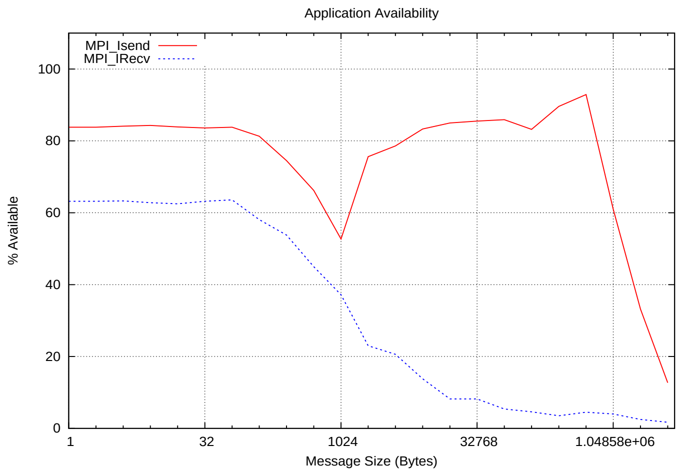

Figure 7: Host Processor Overhead Microbenchmark Results: Application Availability

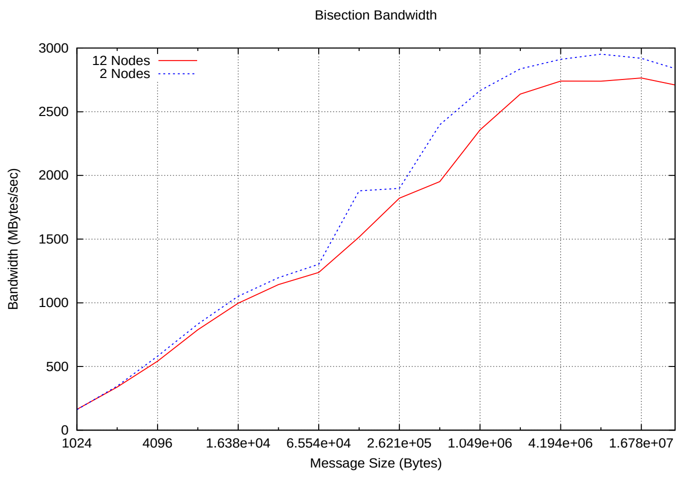

Figure 8: Bisection Bandwidth Microbenchmark Results

6.5 Bisection Bandwidth

The bisection bandwidth microbenchmark shown in figure 8 performed similarly to the Pallas sendrecv microbenchmark. As expected, the bisection bandwidth microbenchmark obtained a greater peak bidirectional bandwidth, a consistent 2.95 Gbytes to a sendrecv bandwidth of 2.86 GBytes. These bandwidth values place the interconnects bidirectional bandwidth capabilities under 3 GBytes, which is noteworthy. Lastly, maximum bandwidth was only obtained at message sizes greater than 4 MBytes, which also occurred in the sendrecv program; this tendency to not obtain high bandwidth until extremely large message sizes is not common among other interconnects.

7 Application Results

All applications presented in this section were tested in conditions consistent with a production environment. All applications were compiled with a high level of optimization consistent on both of the two platforms presented. All processor to MPI task core allocations were done using SLURM on the SiCortex system, and PBS on the x86 cluster.

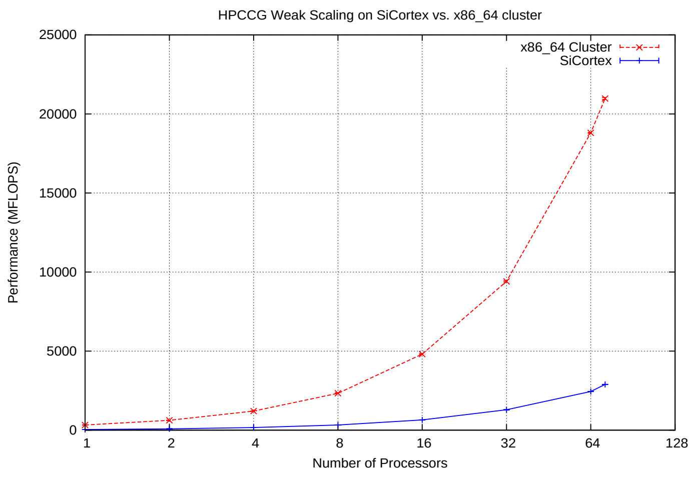

Figure 9: HPCCG Weak Scaling: Raw Data

7.1 HPCCG Results

A weak scaling study was accomplished using HPCCG. A problem size of 64 x 64 x 64 elements for each core was used. For each data point HPCCG was executed three times, and the results were averaged and plotted. The error bars on the plots represent the high and low point. Note the error bars are barely visible on the charts, showing consistency in the results.

Figure 7.1 shows how the clusters performed on HPCCG. It is clear that as far as performance goes, the Opteron cluster outperforms the SiCortex machine due to the large difference in processor speed. However, as we can see in figure 7.1, the SiCortex shows better scalability up to 72 cores, maintaining over 95% efficiency, whereas the Opteron cluster falls to 88% efficiency. It appears the balanced approach taken by SiCortex helps it mantain performance efficiency up to full capacity for HPCCG.

7.2 phdMesh Results

For phdMesh, both weak and strong scaling studies were accomplished. The data gathered for analysis comes from the amount of time performing a parallel geometric search per step. For strong scaling, a 4x8x4 mesh of 128 gears was used. Weak scaling was done with 2 gears per core arranged in an appropriate 3D mesh. For each data point, phdMesh was run three times and the average was plotted, with the high and low showing up as error bars on the plot. Again,

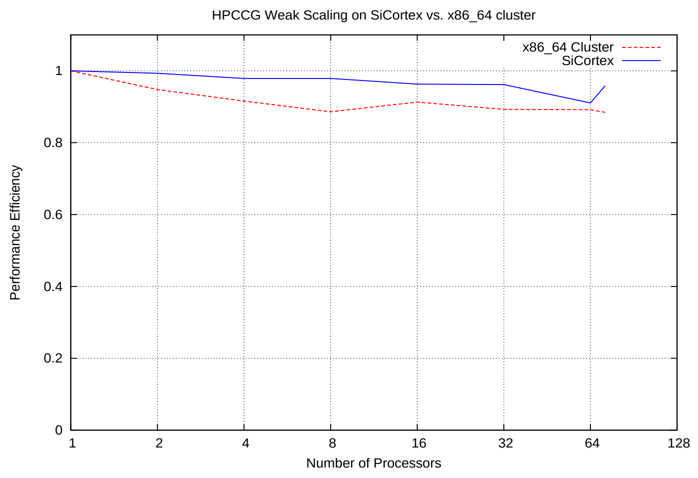

Figure 10: HPCCG Weak Scaling: Raw Data

the error bars are barely visible on the plot, showing consistent results.

In strong scaling, as we can see in Fig. 7.2, the x86 cluster completed the task significantly faster due to it's greater per-core performance. Unlike HPCCG, though, the scaling was almost identical on the Opteron cluster as the SiCortex system. This can be seen in Fig. 7.2(the plots are almost identical). Note that in Fig. 7.2, the graph is normalized so for each of the systems '1' represents the amount of time one core took searching, and all other times are relative.

Similar results to those seen in the strong scaling study are seen in the weak scaling study. The Opteron cluster performs better than the SiCortex in general, simply because of the difference in raw computational ability per processor. Again, the scaling of the two systems are nearly identical, as seen in Figure 7.2.

On phdMesh, unlike HPCCG, we see almost identical scalability between the x86 cluster and the SiCortex cluster up to 64 cores. Both systems scale fairly well and consistently up to 64 cores with phdMesh.

7.3 LAMMPS Results

Strong and Weak Scaling results are presented for the two LAMMPS benchmarks chosen, Lennard Jones and Rhodospin Protein. Three runs were utilized on both clusters in all LAMMPS evaluations; the final result is the normalization of the minimum time obtained for each run at each core allocation size, with normalization based on single core run times. Power-of-two core allocations up

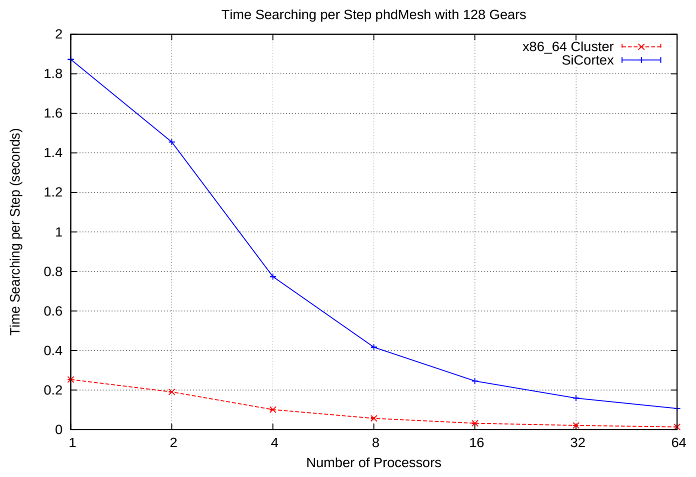

Figure 11: phdMesh Strong Scaling: Raw Data

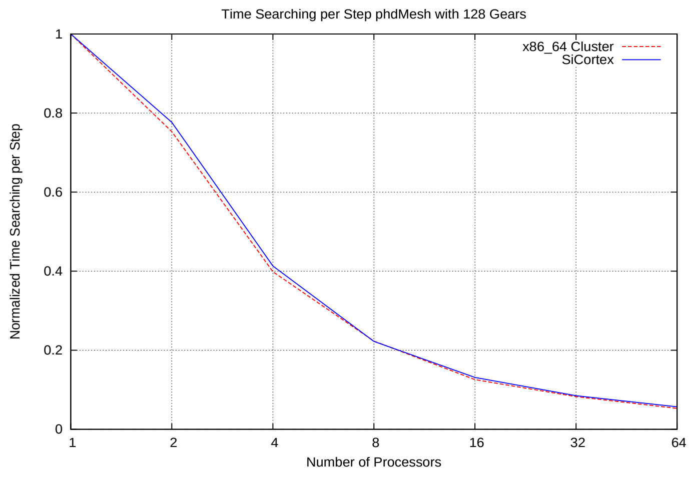

Figure 12: phdMesh Strong Scaling: Normalized Data

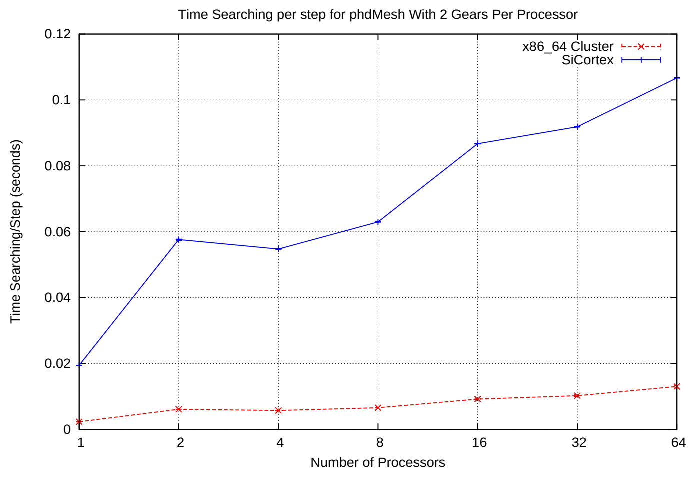

Figure 13: phdMesh Weak Scaling: Raw Data

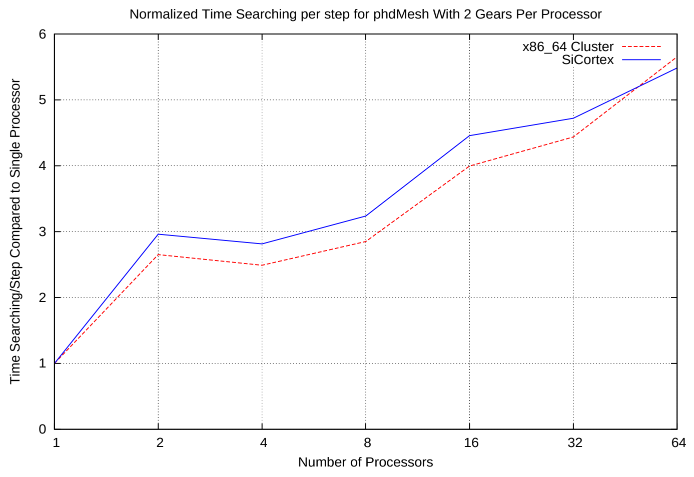

Figure 14: phdMesh Weak Scaling: Normalized Data

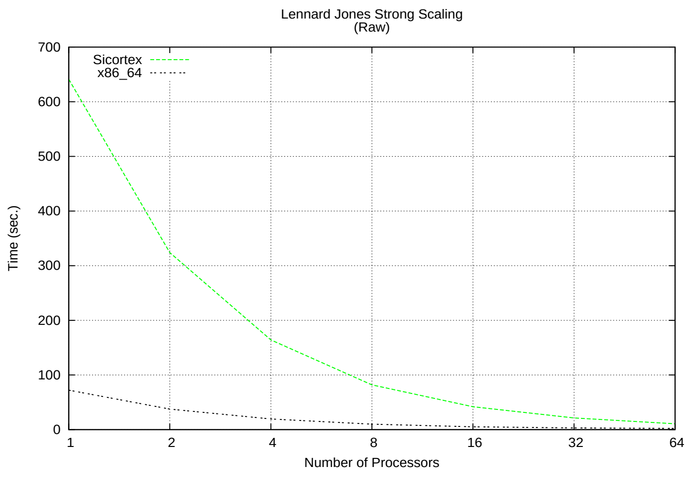

Figure 15: Lennard Jones Strong Scaling: Raw Data

to 64 processors were selected as the job sizes.

Problem size was set at 32000 atoms for Rhodospin Protein strong scaling. The same problem size was originally used for Lennard Jones strong scaling analysis; however, this problem size proved to be too small a problem for the Linux cluster which demonstrated poor and uncharacteristic performance. To combat the problem, a problem size of 4194304 atoms was implemented for LJ, "Lennard Jones", strong scaling on both clusters. Figure 7.3 demonstrates the SiCortex's greater scalability versus those of a generic Linux cluster for LJ strong scaling, a non-computationally intensive benchmark. Correspondingly, the computationally intensive Rhodo, "Rhodospin Protein", strong scaling favored the faster generic Linux cluster for small core allocations; however, scaling performance on the generic cluster begins to fall as core allocations grow larger, which is shown in figure 7.3. Not to our amazement, the SiCortex scaled better than the generic Linux cluster for non-computationally intensive weak scaling (LJ microbenchmark) and the generic Linux cluster demonstrated good scaling for the computationally intensive weak scaling (Rhodo microbenchmark), see figures 7.3, 7.3. Weak scaling, in this instance, was performed by increasing the problem size proportionally to the number of processors allocated. Although the generic cluster clock times are consistently lower than the SiCortex for weak scaling, the SiCortex nearly catches the generic cluster at larger allocation sizes for strong scaling. Combined, the weak and strong scaling results demonstrate a tendency for the SiCortex to scale better for less computationally intensive programs.

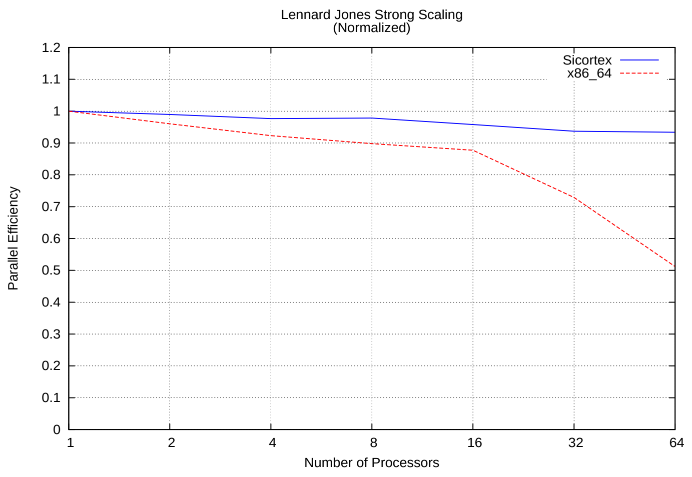

Figure 16: Lennard Jones Strong Scaling: Normalized Data

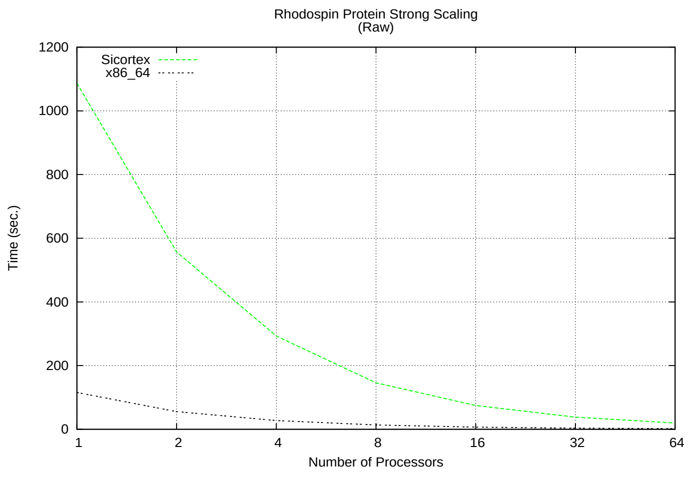

Figure 17: Rhodospin Protein Strong Scaling: Raw Data

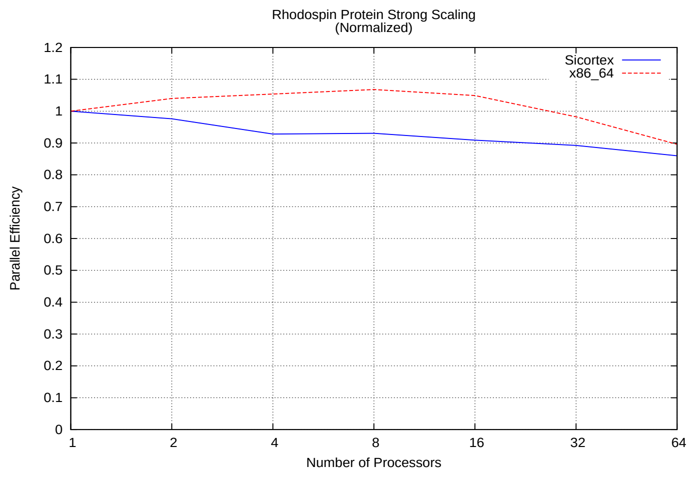

Figure 18: Rhodospin Protein Strong Scaling: Normalized Data

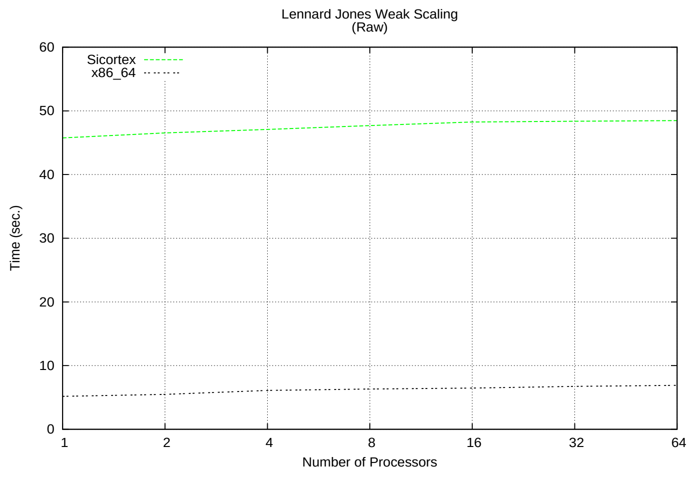

Figure 19: Lennard Jones Weak Scaling: Raw Data

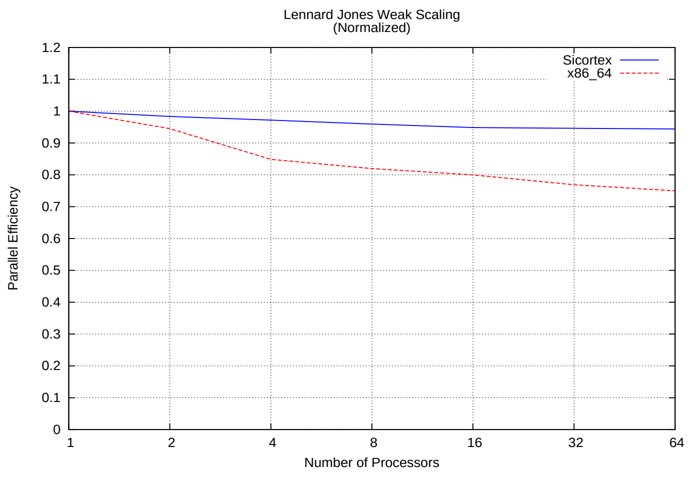

Figure 20: Lennard Jones Weak Scaling: Normalized Data

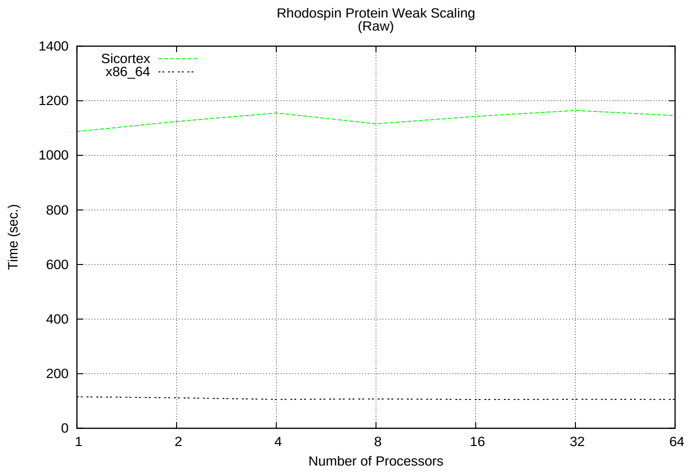

Figure 21: Rhodospin Protein Weak Scaling: Raw Data

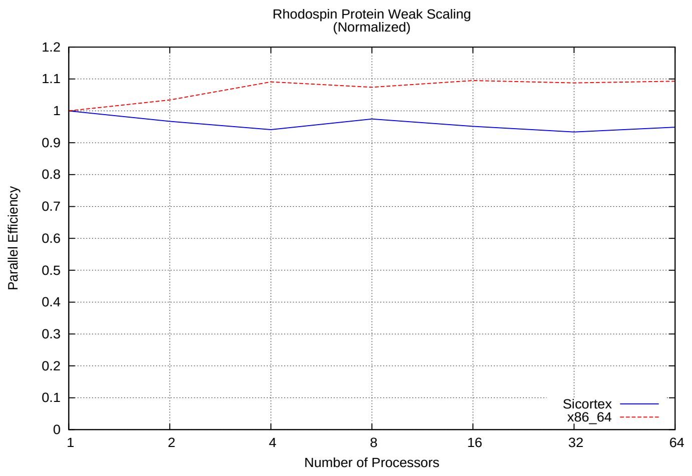

Figure 22: Rhodospin Protein Weak Scaling: Normalized Data

8 Performance per Watt

The motivation behind the use of the underpowered MIPS64 processors found in SiCortex systems is the fact that they take a low amount of power to run and provide a very high performance to watt ratio. Each processor in a SiCortex system consumes less than one watt, and each six-processor node, which includes memory and other components for those six processors, consumes less than 15 watts. In comparison, the commodity cluster's Opteron core used in this paper requires 85.3 Watts. Peak performance per watt on SiCortex systems is 322 MFLOPS per watt, an impressive number. In comparison, the average performance per watt of a system on the July 2008 version of the top500 list is 122 MFLOPS per watt [7].

As the performance per watt data for the x86 cluster was unavailable, the data presented here is based on the wattage of a single socket and the performance of the 4 applications presented in section 7. Wattage for the x86 cluster socket encompasses the NIC, memory (4 DIMMS), and the Opteron processor. The NIC and memory power contributions are estimates based upon previous work done by one of the authors; altogether, the generic cluster socket consumes approximately 115 watts. For the SiCortex, one socket power consumption is 15 watts, which is the equivalent of one node. Weak and strong scaling 64 core runs were used for our analysis. With 64 core allocations, the x86 cluster used 64 sockets and the SiCortex used 11 sockets. Considering power analysis results, it is quite apparent the energy consumption for the SiCortex system is

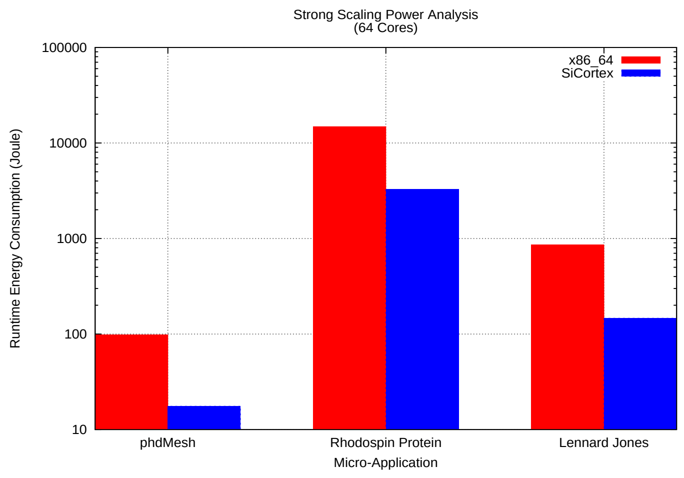

Figure 23: x86_64-SiCortex Strong Scaling Power Analysis

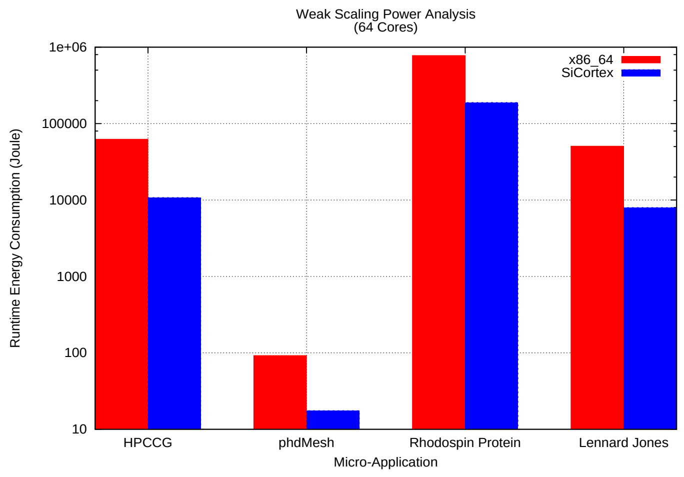

Figure 24: x86_64 - SiCortex Weak Scaling Power Analysis

low compared to the x86 cluster. It should also be noted that the SiCortex run time for equivalent core allocations was much higher. This would normally raise a dilemma as to whether or not the decreased power consumption is worth the increase in run time; however, the core allocation size on the SiCortex can be raised to levels that compete with lower allocation sizes on generic systems and still consume less power.

9 Conclusions

In order for SiCortex to be competitive in the HPC using the less computationally powerful MIPS processors, they need to show that the amount of cores needed to make up for computational ability in a large system will not cause a steep drop-off in performance. The results gathered largely reflect the claims that the system scales well to a reasonable number of processors. The microbenchmarks show that the communication system is capable of low-latency, high-bandwith data transfer on par with many popular commercial interconnects. This interconnect capability coupled with the slower clock rate of the MIPS processors provide more balance than is typically seen in a commodity cluster, preventing many of the communication bottlenecks prevalent in the world of high performance computing. The application benchmarks show us that the system is consistently scalable to a reasonable number of nodes on all applications tested. As a result, more of the advertised performance will be used at a high number of nodes. This was reflected in the fact that for all of the applications which analysis was performed on, performance efficiency per core never dipped below 87%. We compared these performance numbers to those of a typical commodity cluster in production, and we saw some advantages in scalability for the SiCortex system. The total performance of the commodity cluster's high-powered Opterons, however, outpaced the slower MIPS processors in the SiCortex. In some applications the commodity cluster showed the same or greater performance efficiency at a high number of processors than the SiCortex, but the SiCortex showed consistent scalability across all applications. The performance efficiency of the commodity cluster dropped as low as 51% on an appplication benchmarking study. In terms of performance per watt, we saw a huge advantage for the SiCortex, a big concern recently in the world of HPC due to operating costs and environmental impacts. In terms of pure processing power per node, the SiCortex MIPS64 nodes do not compete with today's modern consumer processors. However, our results demonstrate that their more balanced approach to HPC leads to consistent scalabilty and greater performance per watt than a typical commodity cluster.

10 Acknowledgements

We would like to express our gratitude to the Computer Science Research Institute (CSRI), the Student Internship Program (SIP), and our manager James Ang. In addition, we greatly appreciate the engineers at SiCortex for their help

in preparing the SC072 for analysis.

References

[1] *Host processor overhead (hpo).* Available http://www.cs.sandia.gov/smb/overhead.html.

[2] *Hpccg.* Available http://software.sandia.gov/mantevo/download.html.

[3] *Lammps molecular dynamics simulator.* Available http://www.lammps.sandia.gov.

[4] *phdmesh.* Available http://www.cs.sandia.gov/ maherou/.

[5] *Sicortex website.* Available http://www.sicortex.com.

[6] *Sicortex whitepapers.* Available http://www.sicortex.com/products/white_papers.

[7] *Top 500 computer sites.* Available http://www.top500.org.

[8] D. Doerfler, *An analsysis of the pathscale inc. infiniband host channel adapter, infinipath*, Tech. Rep. SAND2005-5199, Sandia National Laboratories, August 2005.

[9] D. Doerfler and R. Brightwell, *Measuring mpi send and receive overhead and application availability in high performance network interfaces*, in EuroPVM/MPI, 2006.

[10] W. Kautz, *"bounds on directed (d,k) graphs," theory of cellular logic networks and machines*, Tech. Rep. AFCRL-68-0668, Air Force Cambridge Research Laboratory, 1968. pp. 20-28.

[11] M. Reilly, L. Stewart, J. Leonard, and D. Gingold, *Sicortex technical summary*, April 2008. Available http://www.sicortex.com/whitepapers/sicortex-tech_summary.pdf.

[12] L. Stewart and D. Gingold, *A new generation of cluster interconnect*, April 2008. Available http://www.sicortex.com/whitepapers/sicortex-cluster_interconnect.pdf.

[13] L. Stewart, D. Gingold, J. Leonard, and P. Watkins, *Rdma in the sicortex cluster systems*, in EuroPVM/MPI, 2007.