

# An Evaluation of the Impacts of Network Bandwidth and Dual-Core Processors on Scalability

Ron Brightwell, Keith D. Underwood, and Courtenay Vaughan  
Sandia National Laboratories  
P.O. Box 5800, MS-1110  
Albuquerque, NM 87185-1110  
{rbbrigh, kdunder, ctvaugh }@sandia.gov

**Abstract**—A portion of the Cray Red Storm system at Sandia National Laboratories recently completed an upgrade of the processor and network hardware. Single-core 2.0 GHz AMD Opteron processors were replaced with dual-core 2.4 GHz AMD Opterons, while the network interface hardware was upgraded from a sustained rate of 1.1 GB/s to 2.0 GB/s (without changing the router link rates). These changes more than doubled the theoretical peak floating-point performance of the compute nodes and doubled the bandwidth performance of the network. This paper provides a analysis of the impact of this upgrade on the performance of several applications and micro-benchmarks. We show scaling results for applications out to thousands of processors and include an analysis of the impact of using dual-core processors on this system.

## I. INTRODUCTION

The emergence of commodity multi-core processors has created several significant challenges for the high-performance computing community. One of the most significant challenges is maintaining the balance of the system as the compute performance increases with more and more cores per socket. Increasing the number of cores per socket may not lead to a significant gain in overall application performance unless other characteristics of the

system – such as memory bandwidth and network performance – are able to keep pace. As such, understanding the impact of increasing the number of cores per socket on application performance is extremely important.

The Cray XT3 system was designed by Cray and Sandia National Laboratories specifically to meet system balance criteria that are necessary for effectively scaling several important DOE applications to tens of thousands of processors. Several large XT3 systems have been deployed and have demonstrated excellent performance and scalability on a wide variety of applications and workloads [1], [2]. Since the processor building block for the XT3 system is the AMD Opteron, existing single-core systems can be upgraded to dual-core simply by changing the processor.

The Red Storm system at Sandia, which is the predecessor of the Cray XT3 product line, recently underwent the first stage of such an upgrade on more than three thousand of its nearly thirteen thousand nodes. In order to help maintain system balance, the Red Storm network on these nodes was also upgraded from SeaStar version 1.2 to 2.1. The latest version of the SeaStar provides a significant increase in network bandwidth. Before conducting initial studies of single-core versus dual-core performance, the system software environment had to be enhanced to support multi-core compute nodes. Performance results from a small system

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

were encouraging enough to justify upgrading the full Red Storm system to dual-core nodes. With the first stage of the upgrade completed, we are now able to analyze the impact of the processor and network upgrade on application performance and scaling out to several thousand cores.

The main contribution of this paper is an in-depth analysis of application performance and scaling on a large-scale dual-core system. We provide results for multiple representative DOE applications and compare single-core versus dual-core performance on up to four thousand cores. Few such studies have been conducted on commodity dual-core systems at this scale, and we believe this is the first such study of the popular Cray XT3 platform. This paper also includes an analysis of the performance of the SeaStar 2.1 network using traditional micro-benchmarks. The Red Storm system is the first (and perhaps only) XT3 product to have the SeaStar 2.1 network, which will be the network deployed in the next-generation XT product. The combination of dual-core compute nodes with the SeaStar 2.1 network should provide a glimpse as to the level of performance and scaling that will be available in Cray's follow-on system. Also, this paper provides a description of the enhancements that were made to Sandia's Catamount lightweight kernel environment and Portals network stack to support dual-core XT3 systems.

The rest of this paper is organized as follows. In the next section, we describe how this study complements previously published research. Section II contains the specific details of the processor, network, and system software upgrade, including a detailed description of the necessary changes to Catamount. Micro-benchmark performance for the SeaStar 2.1 network is presented in Section III, while higher-level network benchmark results are shown in Section IV. Section V provides a detailed analysis of several real-world application codes, both in terms of application scaling and absolute system performance. We summarize the conclusions of this study in Section VI and close by discussing avenues of future work in

Section VII.

## II. OVERVIEW OF SYSTEM UPGRADE

While the goal of the Red Storm upgrade was to provide enhanced processing and network capabilities, it was also necessary to enhance the system software to leverage the dual-core capabilities. This section describes the hardware enhancements, their implications, and the software enhancements needed to leverage the new hardware.

### A. Hardware Upgrade

The Red Storm system was designed to accommodate both a processor and a network upgrade. The original system had 2.0 GHz single-core Opteron processors for a peak performance of 4 GFLOPs per node. It was anticipated that dual-core processors would become available in the lifetime of the machine, and the current upgrade is installing socket-compatible 2.4 GHz dual-core Opteron processors that have a peak performance of 9.6 GFLOPs per node — an improvement of almost  $2.5\times$ .

At Sandia's request, the board-level design placed the network chips on a mezzanine that could easily be replaced. Thus, along with the processor upgrade, the SeaStar 1.2 network chips are being replaced with SeaStar 2.1 network chips. The new network chips increase the sustained unidirectional bandwidth from 1.1 GB/s to 2.1 GB/s and the sustained bidirectional bandwidth from 2.2 GB/s to 3.6 GB/s. This increase helps to maintain the balance of the system, but the change is primarily a change in *injection* bandwidth — SeaStar 1.2 chips could receive up to 2 GB/s, but they could only send 1.1 GB/s. Notably, neither message latency or message rate properties of the SeaStar itself were changed. Similarly, the router link bandwidth has not increased. The router links, which could only be half subscribed by a SeaStar 1.2 node, can now be almost fully subscribed by a single SeaStar 2.1 node; thus, network contention will be significantly increased.

## B. Software Upgrade

Compute nodes on the Cray XT3 system run a lightweight kernel called Catamount [3], which is a third-generation operating system developed by Sandia and the University of New Mexico (UNM). Sandia and Cray worked jointly to provide Catamount for the XT3, and Sandia provided enhancements to Cray to support dual-core compute nodes. The lowest-level network programming interface on the XT3 is called Portals [4], which was also developed jointly by Sandia and UNM. As with Catamount, Sandia and Cray have worked closely to provide the implementation of Portals for the SeaStar. The following describes the changes that were necessary to Catamount, the parallel runtime system, and to Portals to support running parallel applications on dual-core compute nodes.

Catamount consists of three components: the Quintessential kernel (QK), the Process Control Thread (PCT), and the parallel application loader, called yod. The QK and the PCT work together to manage the resources available on a compute node. The QK provides all of the mechanisms to manage hardware resources, while the PCT, which is a privileged user-level process, sets the policies for managing those resources. Yod plays the part of the parallel application launcher, similar to mpirun. Yod is a service-node application that communicates with a PCT to create a user-level process on a compute node.

The QK is the lowest level of the operating system. Logically, it sits closest to the hardware and performs services on behalf of the PCT and user-level processes. The QK supports a small set of tasks that require execution in supervisor mode, including servicing network requests, interrupt handling, and fault handling. If the interrupt or fault is caused by the application, control is turned over to the PCT for handling. The QK also fulfills requests made by the PCT, including running processes, context switching, virtual address translation and validation. However, the QK does not manage the resources on a compute node. It simply provides the

necessary mechanisms to enforce policies established by the PCT and to perform specific tasks that must be executed in supervisor mode.

The PCT is a privileged user-level process that performs functions traditionally associated with an operating system. It has read/write access to all memory in user space and is in charge of managing all operating system resources, including starting and scheduling processes and memory management. When the QK starts the PCT, the remainder of physical memory is included in the PCT's heap. When the PCT starts a process, it allocates space for the process from its heap. The user process is encapsulated within the PCT's address space, which enables debugging and core dump processing.

There are several important features of Catamount that differentiate it from traditional full-featured operating systems like Linux. First, Catamount does not support virtual memory. However, it does support virtual addressing to provide protection between privileged and non-privileged processes. Secondly, the default allocation scheme for a compute node is to commit all available resources to a process. This means that when a process is started, its stack size and heap size are fixed and consume all of available memory. If a process requires a larger stack space than the default or if the user wants to start multiple processes on a node, the user must explicitly tell the process loader how much stack or heap to allocate.

Support for dual cores in Catamount is implemented by virtual node mode (VNM). VNM makes individual nodes appear as two separate nodes to the parallel application. However, since the number of processors is the only resource that has been doubled, all other resources, including node memory and the network, must be shared. This mode does not support any form of shared memory communication between the two application processes on a node. All communication must still be performed using network commands. The implementation of the network transfer may use shared memory, but this is not exposed to the user-level processes. From the application

perspective, there are two totally independent processes on the node just as if they were on separate nodes.

The changes to the QK to support multiple cores fell mainly into two categories: adding a dimension to control variables and initializing the second CPU at boot time. The QK does not have a particular awareness of VNM, but it does treat the processors in a master-slave fashion. Since all PCT execution and scheduling occurs on the first processor, the second processor has a 'wait-for-work' loop in the QK. Running an application process on the first CPU involves a context switch from the PCT. Continuing a process on the second CPU involves simply clearing the flag that allows the processor to come out of the wait-for-work loop.

For VNM, process load as seen from yod is mostly unchanged. A single copy of the executable is fanned out to the PCTs. This does restrict the dual cores to having a single choice of binary on each node. The PCT does use only one copy of the text section of the process on each node, but it allocates two copies of every other section for the two processes. Both processes on a node then begin independently and the second process migrates to the second CPU by making a system request. It then notifies the PCT to switch schedulers and the VNM scheduler lets the processes run as appropriate. There is no true scheduling between them.

### III. MICRO-BENCHMARK RESULTS

We used two micro-benchmarks to compare the Seastar 1.2 and Seastar 2.1 network interfaces. We began with a simple ping-pong benchmark to measure the latency between nodes. We also looked at the impacts of the upgrade on message rate and peak bandwidth. To measure peak bandwidth, we used a benchmark developed by the Ohio State University, which posts several messages (64) at the receiver and then sends a stream of messages from the sender.

#### A. Ping-Pong Latency and Bandwidth

Figure 1(a) shows the impact of the upgrade on ping-pong latency. While the Seastar 2.1 did not incorporate any features that should reduce latency, the move from a 2.0 GHz processor to a 2.4 GHz processor reduced ping-pong latency by a full microsecond. This is because the majority of Portals processing happens on the host processor in an interrupt driven mode. Similarly, placing two processors on one node currently means that both the send and receive work happens on one processor<sup>1</sup> and this increases latency by over 2 microseconds. Future versions of the Cray software are supposed to short-circuit the trip through the Seastar to drastically reduce latencies between two cores on one node.

The step in latency between 16 and 32 bytes is attributable to the need to take an extra interrupt for messages that do not fit in the header packet (messages greater than 16 bytes). This step has been a constraint in the Red Storm system, but should go away when Cray implements a fully offloaded Portals[5].

#### B. OSU Streaming Bandwidth

In Figure 1(b), the improvements in the Seastar 2.1 are evident as a two-fold increase in sustained, streaming bandwidth. We can also see an almost 20% improvement in message rate that is provided by the 20% boost in the processor clock rate. The significant degradation in message rate for having two processes on one node (in the dual-core case) is attributable to the same source as the latency increase in Figure 1(a) — competition for processing resources on the node. Again, this reduction in messaging rate would largely evaporate with an offloaded Portals implementation.

The reduction in peak bandwidth when two processes are on one node stems from competition for HyperTransport bandwidth. The transmit DMA engine must perform a HyperTransport read to obtain data. This consumes

<sup>1</sup>System calls are proxied to one processor in a node.

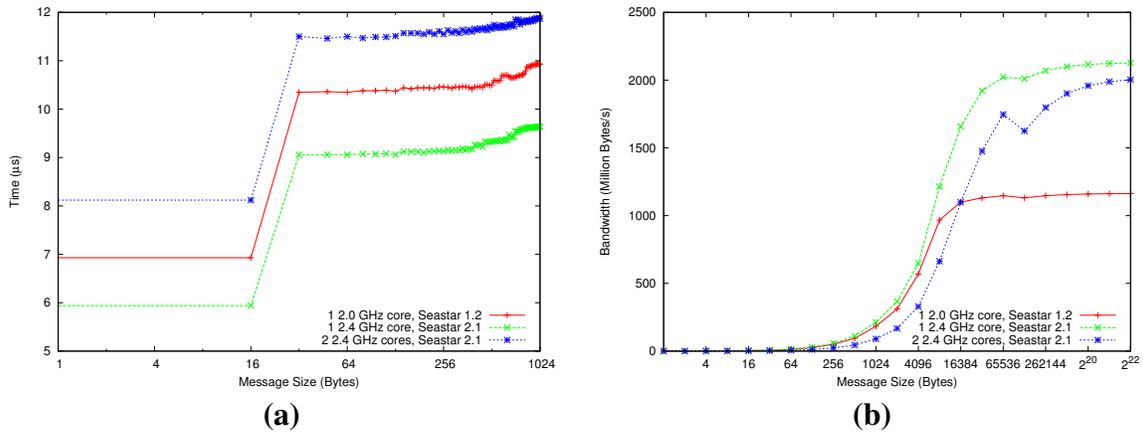


Fig. 1. Ping-pong latency (a) and streaming bandwidth (b)

some of the bandwidth that would otherwise be used to receive data.

#### IV. HIGHER-LEVEL BENCHMARKS

While micro-benchmarks provide some interesting insights into the network properties of individual nodes, higher-level benchmarks can provide much more insight into overall system-level behaviors. Thus, we have included results from the Pallas MPI benchmark suite[6] and from the HPC Challenge benchmark suite[7].

##### A. Pallas

The Pallas benchmarks provide a slightly higher level view of system-level network performance by measuring the performance of numerous MPI collectives. Figure 2 presents data for three collectives that are particularly common in scientific applications: Alltoall, Allreduce, and Reduce. In each of the three, the performance of the collectives using small sizes is dominated by latency. As such, the upgrade provides an advantage that is commensurate with the improvements in latency that are seen with the faster processors. Furthermore, moving to two cores per socket *does not* suffer from the sacrifice in latency experienced by a ping-pong test between cores in a single socket. This is because most of the collectives involve (at most) one or two communications between the cores in one socket and not contention at every phase of the algorithm.

At larger sizes, point-to-point bandwidths dominate for both the Allreduce and Reduce. Thus, the upgraded platform continues to see remarkable improvements from the improvement in bandwidth. In contrast, Alltoall performance at large message sizes can become constrained by bisection bandwidth. Thus, the curves start to converge. A remarkably bad region can actually be seen in the Alltoall curve where using both cores on a single socket causes significant degradation in performance. This is likely caused by a particularly poor allocation that causes regular synchronized link contention between the cores in a socket. The collective algorithms should generally be tuned to better match the topology of the machine with a particular focus on considering the impact of multi-core.

##### B. HPC Challenge

The HPC Challenge benchmarks provide a much broader view of system performance than simply network measurements. Performance measurements cover processor performance (HPL), memory performance (STREAMS), small message network performance (RandomAccess), network bisection bandwidth performance (PTRANS), and a coupled processor/network test (FFT). While far short of a real application, in its baseline form, this benchmark suite captures many more aspects of system performance than other benchmarks.

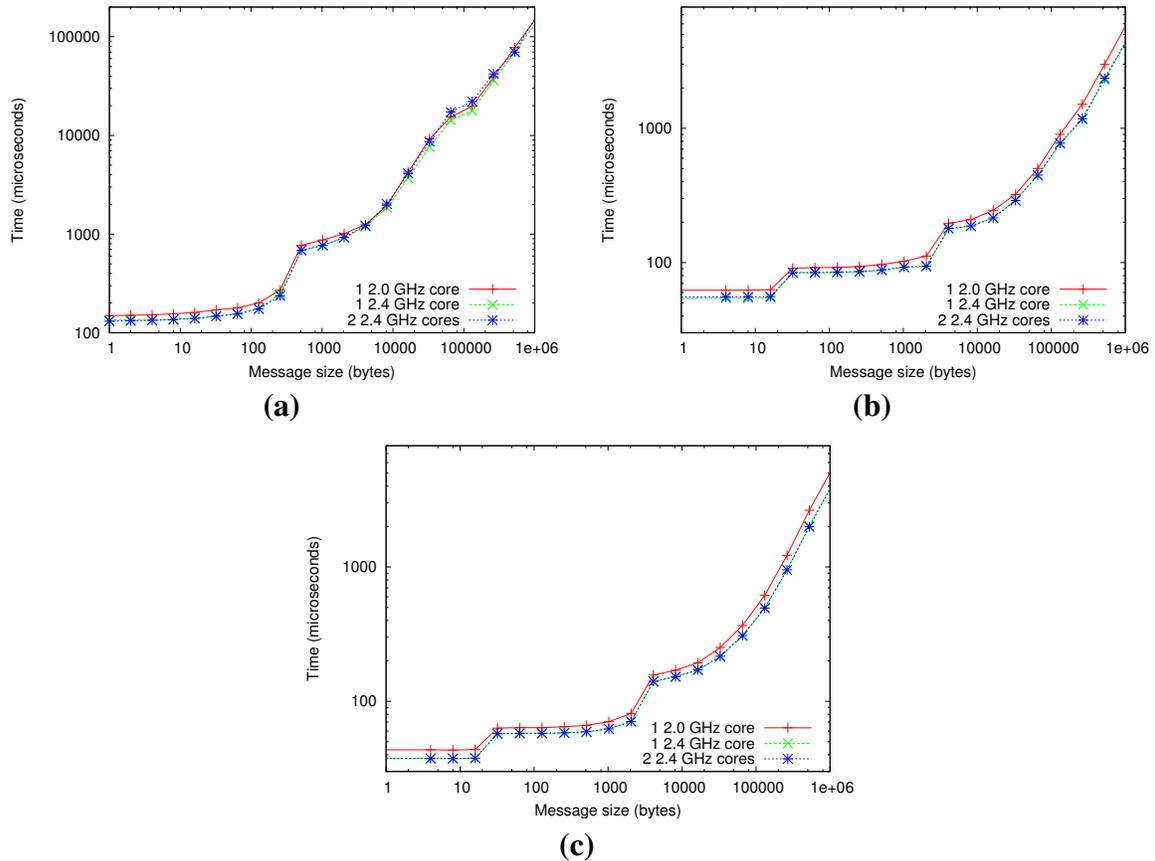


Fig. 2. 64 node Pallas benchmarks for (a) Alltoall, (b) Allreduce, and (c) Reduce

Figure 3 presents the performance of the more processor centric HPC metrics. As expected, moving from 2.0 GHz processors to 2.4 GHz processors provided approximately a 20% boost per core for HPL. FFT received a smaller (10%) gain, as it is a more memory bound code. In contrast, STREAMS actually lost a bit of performance. This is not surprising since the benchmarks used the configurations tuned for the slower processors, which may not be optimal for the faster processor.

Figure 3 also presents dual-core performance data along with a *percent improvement per socket* achieved by using two cores instead of one (while keeping the problem size constant). HPL is clearly the largest winner with an 80 to 90% gain. FFT achieves a somewhat surprising 20 to 40% win, indicating that the second cache is providing somewhat of a benefit. Finally, STREAMS achieves a remarkable 10 to 30%

improvement. A growing trend in microprocessors is an inability to sustain higher bandwidth due to the excessive relative memory latency. Here we see evidence that two independent issue streams in one socket can extract significantly more memory bandwidth from the memory controller.

The PTRANS and RandomAccess components of the HPC benchmark are shown in Figure 4. Both of these benchmarks are much more network centric and measure aspects of the network that were not improved by the upgrade. PTRANS focuses on bisection bandwidth, which did not change since the router links did not change. Likewise, the MPI message rate, which is measured by RandomAccess, did not change with the network upgrade. RandomAccess did, however, see a significant gain from the almost 20% gain in MPI message rate that was provided by the processor

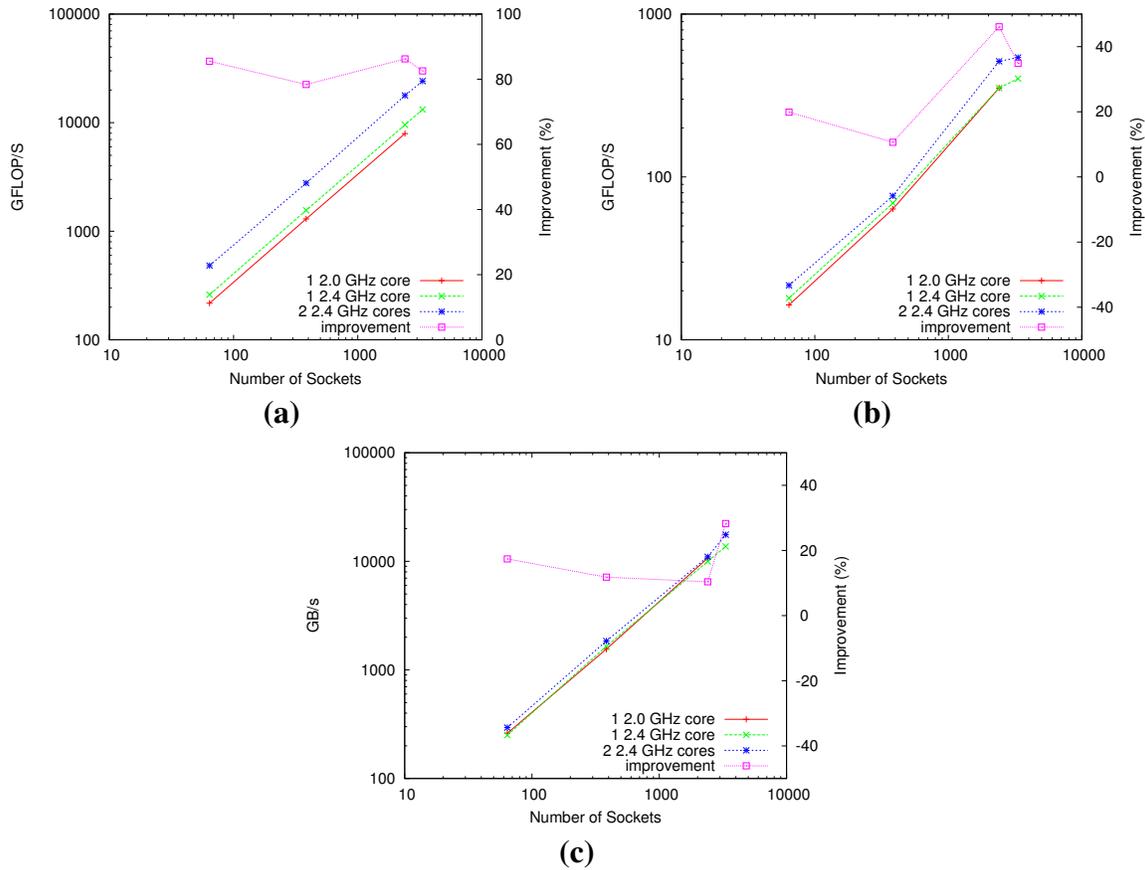


Fig. 3. (a) HPL, (b) FFT, and (c) STREAMS results

upgrade.

Both the PTRANS and RandomAccess benchmarks also show mixed results from using the second core in the socket. In the case of PTRANS, a *good* placement of MPI ranks on the nodes would yield much better performance because more communications would be “on node”; however, the allocation algorithm currently in use in the Cray software stack is particularly poor. This is particularly evident in the case where using dual-cores yields a large performance loss. RandomAccess gets no gains in any cases from the second core. At large scales, the additional traffic to the same socket is negligible. Furthermore, two cores now contend for one network interface, which gives each one less than half of the message throughput.

## V. APPLICATIONS

We considered the impact of the Red Storm upgrade on three applications from three general perspectives. The applications include SAGE, PARTISN, and CTH. Each of these are commonly used application level benchmarks within the DOE complex.

### A. SAGE

SAGE, SAIC’s Adaptive Grid Eulerian hydrocode, is a multi-dimensional, multi-material, Eulerian hydrodynamics code with adaptive mesh refinement that uses second-order accurate numerical techniques [8]. It represents a large class of production applications at Los Alamos National Laboratory. It is a large-scale parallel code written in Fortran 90 and uses MPI for inter-processor communications. It routinely runs on thousands of processors for months at a time.

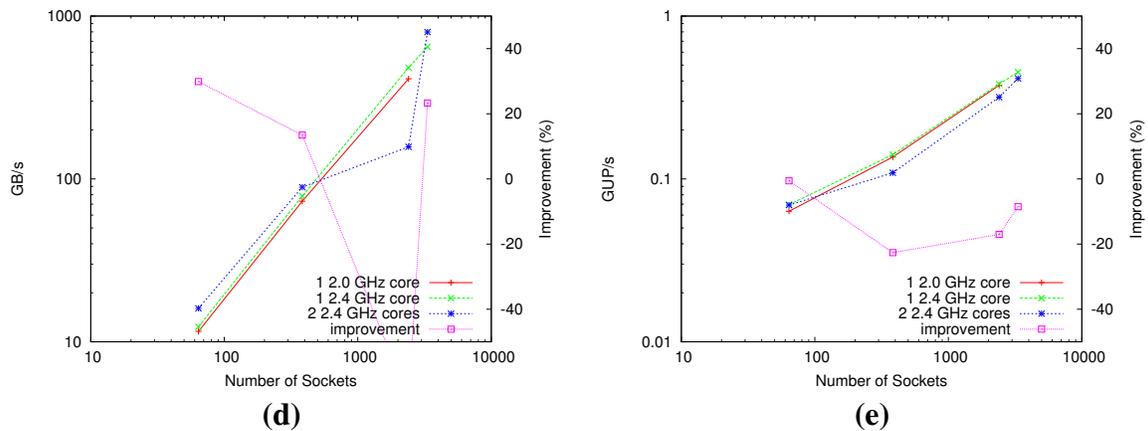


Fig. 4. (a) PTRANS and (b) RandomAccess results

## B. PARTISN

The PARallel, Time-dependent SN (PARTISN) code package is designed to solve the time-independent or dependent multigroup discrete ordinates form of the Boltzmann transport equation in several different geometries. It provides neutron transport solutions on orthogonal meshes with adaptive mesh refinement in one, two, and three dimensions. A multi-group energy treatment is used in conjunction with the  $S_n$  angular approximation. A significant effort has been devoted to making the code run efficiently on massively parallel systems. It can be coupled to nonlinear multi-physics codes that run for weeks on thousands of processors to finish one simulation.

## C. CTH

CTH is a multi-material, large deformation, strong shock wave, solid mechanics code developed at Sandia. CTH has models for multi-phase, elastic viscoplastic, porous and explosive materials. Three-dimensional rectangular meshes; two-dimensional rectangular, and cylindrical meshes; and one-dimensional rectilinear, cylindrical, and spherical meshes are available. It uses second-order accurate numerical methods to reduce dispersion and dissipation and to produce accurate, efficient results. CTH is used for studying armor/anti-armor interactions, warhead design, high explosive initiation physics, and weapons safety

issues.

## D. Scalability Impact

When analyzing the impact of the upgrade, there are three ways to think of the data. The first viewpoint considers the impact of moving from Seastar 1.2 to Seastar 2.1 on application scalability. Another viewpoint asks the question: how is overall scalability impacted by having two cores share a connection to the network? Finally, we can consider the performance improvements in terms of the *performance gain per socket*.

Figure 5(a) and (b) highlight the scalability of PARTISN in the diffusion and transport sections, respectively. Scalability is clearly not a problem with the move to faster processors. Furthermore, on larger problem sizes, the dual core processors show very little scalability impact as the number of processors increases; however, the problem is very sensitive to the contention for resources (both the network interface and memory bandwidth) that arises when two cores are placed in one socket. Another interesting note is in the jaggedness of the lines in Figure 5(b). Parallel efficiency should be approximately monotonic; however, we see numerous points where it increases as we move to larger numbers of processors. These increases typically occur because of particularly good mappings between the problem specification and the way the allocator places the MPI ranks on the physical nodes. These

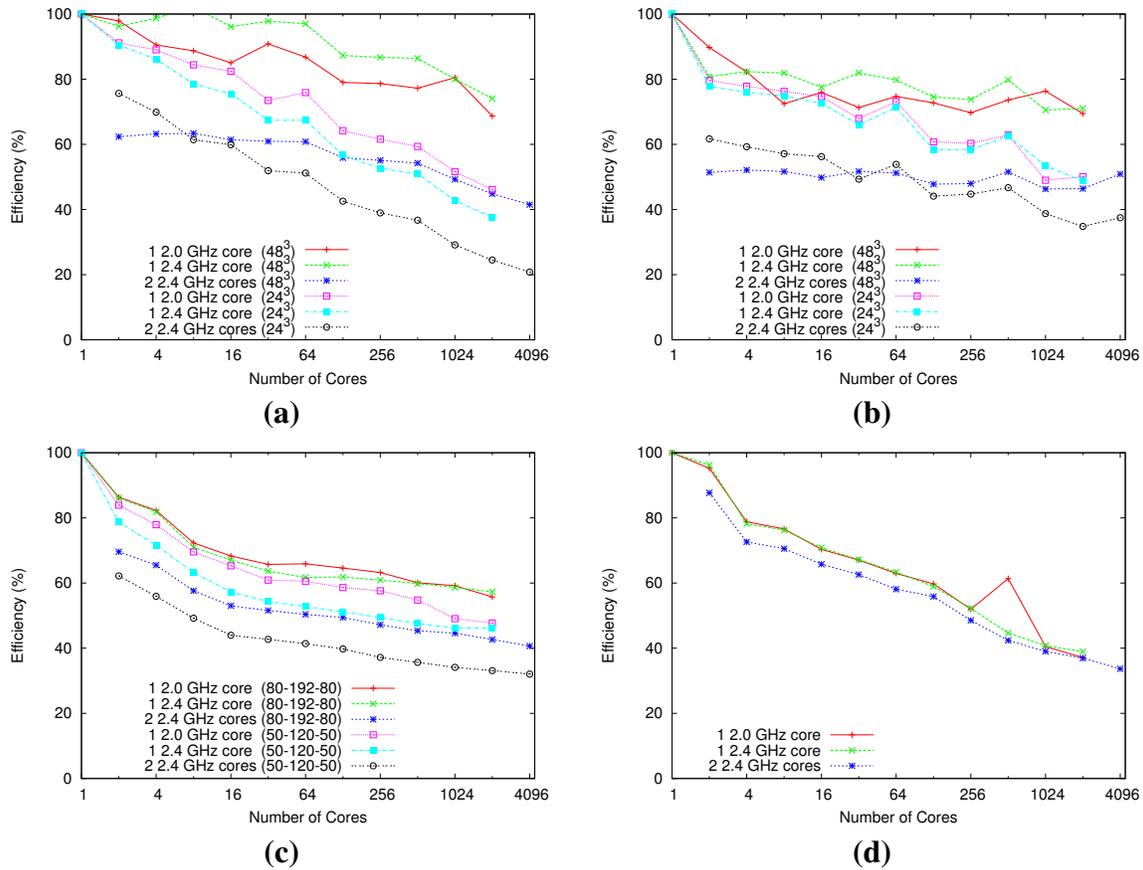


Fig. 5. Parallel efficiency impacts on PARTISN (a) diffusion and (b) transport portions and parallel efficiency impacts on (c) CTH and (d) SAGE

data points could be greatly smoothed by moving to a better allocation algorithm[9].

Figure 5(c) shows scalability results for CTH on two scaled speedup problem sizes (problem size shown is per node). Notably, the overall upgrade was neutral in terms of scalability for a single core per node at the larger problem size. Similarly, moving to two cores per node takes an overall parallel efficiency drop as two MPI tasks contend for memory bandwidth; however, that drop is constant and overall scalability is relatively unimpacted. At the smaller problem size, however, we see one of the weaknesses of the upgrade in that the upgraded processors do not scale quite as well as the slower processors. There are two sources of this issue. Foremost, the processor performance improved by more (20%) for the smaller problem than for the larger problem (10%). In addition, MPI latency did not improve as much as MPI bandwidth,

which impacts applications with smaller messages (in this case, from a smaller problem size).

Like CTH, SAGE scales extremely well on the upgraded machine, as indicated in Figure 5(d). In fact, the parallel efficiency of the single core and dual core systems begins to converge at large scales as scaling issues begin to dominate single processor performance issues. Thus, it is clear for both SAGE and CTH that contention for the network interface is not a particular issue. The only anomaly is at 512 nodes where the pre-upgrade machine has a “magic data point” that is reproducible. As with the PARTISN transport problem, these points periodically arise as the application mapping to the nodes hits a particularly good configuration (or bad configuration in the case of PARTISN).

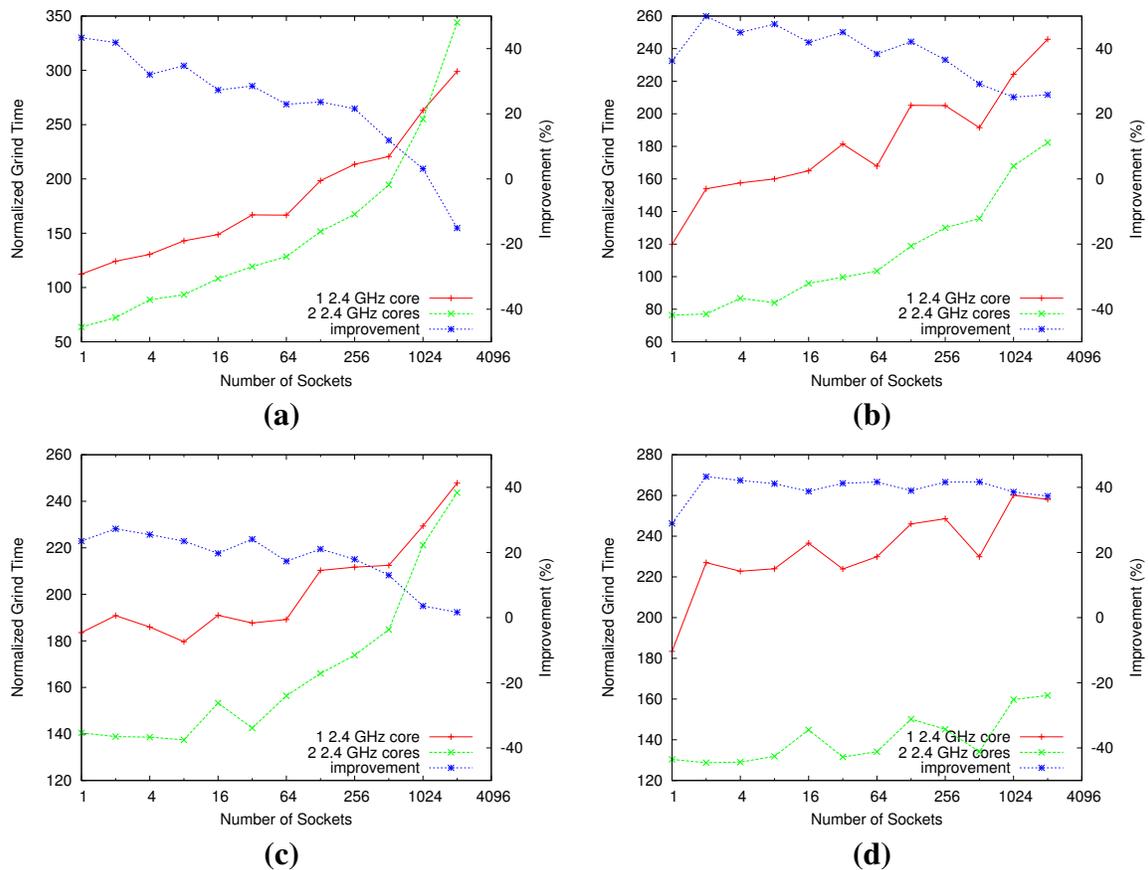


Fig. 6. Percent improvement for PARTISN diffusion (a, c) and transport (b, d) portions for  $24^3$  (a, b) and  $48^3$  (c, d) problems

### E. Dual-Core Improvement

Another way to consider the upgrade is based on the improvement offered by using dual core processors instead of single core processors. This view holds the problem size per socket constant (since the memory size is constant) and graphs absolute performance in terms of time on the left axis and percent improvement on the right axis versus the number of *sockets* on the X-axis. The PARTISN diffusion problem (Figure 6(a, c)) sees improvements of 20 to 40% on small numbers of sockets; however, at large scale, the use of dual core processors offers very little advantage. In fact, there can even be a slight performance loss associated with the drastic increase in MPI tasks needed to support two cores in each socket! In stark contrast, the transport problem (shown in Figure 6(b, d)) achieves a consistent 25 to 50% performance improvement from

using the second core.

Much like the diffusion portion of PARTISN, CTH receives an impressive 20 to 30% improvement per socket by using dual core processors. This is impressive because the number of MPI tasks has doubled, but the total work has not. Thus, the work per time step per core is reduced by a factor of two when using dual core processors in these cases.

## VI. CONCLUSIONS

This paper has described the dual-core and network upgrade to the Red Storm system at Sandia National Laboratories. This is the first Cray XT3 system to be upgraded to dual-core Opteron processors and SeaStar 2.1 network chips.

Network micro-benchmarks show that half round-trip latency has decreased by nearly 15% due to improvements in the performance of the Opteron. Peak unidirectional bandwidth has

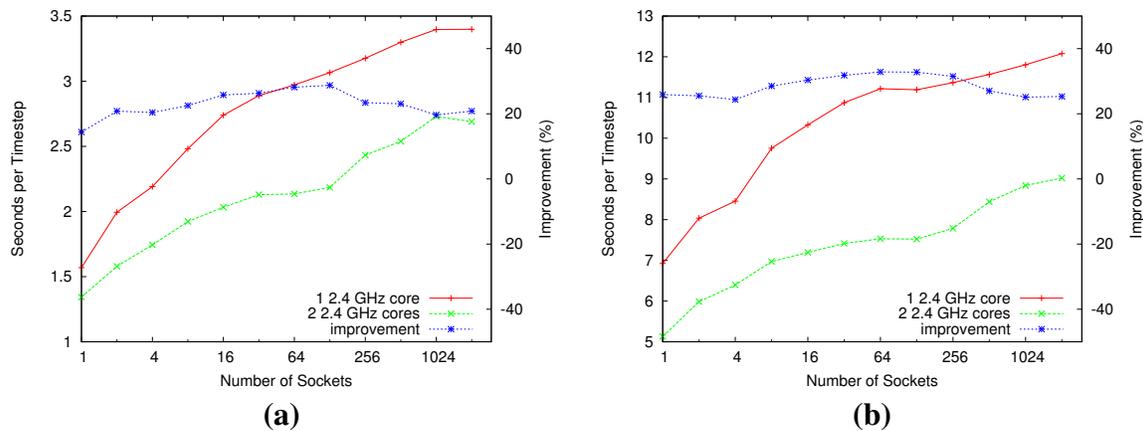


Fig. 7. Percent improvement for CTH sizes 50x120x50 (a) and 80x192x80 (b)

almost doubled, and peak bidirectional bandwidth has increased almost 80%. Improved performance of a single core of the Opteron has also improved small message throughput by over 20%.

We presented three perspectives on the upgrade. First, we considered single-core, 2.0 GHz Opterons with Seastar 1.2 parts and compared them to using a single 2.4 GHz core on each node with Seastar 2.1 parts. This provided insight into the improvements provided by the increase in network bandwidth. Second, we compared scaling efficiency of two cores per node in the upgraded system to the scaling efficiency of one core per node in the upgraded system. This provided a different look at the scaling impacts of network bandwidth in the system. Finally, we compared absolute performance per socket for using a single 2.4 GHz core with use of both cores to provide a look at the advantage obtained from a dual-core upgrade.

Our results showed that adding a second core provides from 20% to 50% performance boost to real applications on a fixed problem size per-socket basis. Furthermore, the results indicate that scalability is impacted relatively little by the upgrade. Most degradation in parallel efficiency is directly attributable to contention for resources caused by having two cores in one socket; however, with the doubling in MPI tasks that is typical of using dual-core proces-

sors, it is possible to see scaling effects that are detrimental to overall performance when running at the largest scale.

## VII. FUTURE WORK

The full system upgrade of the processors and network chips for Red Storm is scheduled to be completed before the end of the 2006 calendar year. In its largest configuration, the final system will have nearly twenty-six thousand processor cores, making it the largest commodity-based dual-core Opteron system in the world. In the Spring of 2007, Red Storm is scheduled to undergo an upgrade of the memory system. The current 333 MHz DDR memory will be upgraded to 400 MHz DDR, and all nodes in the system will have at least 6 GB of host memory.

There are a number of system software enhancements that are under development, but which have not yet been deployed for production dual-core systems. The manner in which Catamount handles system call traps to initiate messages has been changed to address fairness. Currently, a trap on the second core causes the first core to be interrupted so that the Qk can process the request. Ideally, a trap to send a message on the second core should be handled by that core, without disturbing the other core. In addition, message passing between two cores on a node has not been optimized to its fullest extent. There are planned enhancements that

can potentially reduce the latency of inter-node messages significantly.

As mentioned previously, the current Cray-supported implementation of Portals uses the host to process incoming messages. Sandia has developed an implementation of Portals that runs entirely on the SeaStar network interface [5]. This implementation does not use interrupts and uses no host processor cycles to process incoming messages. In addition to the demonstrated network performance improvement of this implementation, we expect it to have some significant benefits for dual-core systems.

## REFERENCES

- [1] J. S. Vetter, S. R. Alam, J. T. H. Dunigan, M. R. Fahey, P. C. Roth, and P. H. Worley, "Early evaluation of the Cray XT3," in *20th International Parallel and Distributed Processing Symposium*, April 2006, pp. 25–29.
- [2] A. Hoisie, G. Johnson, D. J. Kerbyson, M. Lang, and S. Pakin, "A performance comparison through benchmarking and modeling of three leading supercomputers: Blue Gene/L, Red Storm, and Purple," in *Proceedings of the IEEE/ACM International Conference on High-Performance Computing, Networking, Storage, and Analysis (SC'06)*, November 2006.
- [3] S. M. Kelly and R. Brightwell, "Software architecture of the light weight kernel, Catamount," in *Proceedings of the 2005 Cray User Group Annual Technical Conference*, May 2005.
- [4] R. Brightwell, T. Hudson, K. Pedretti, R. Riesen, and K. Underwood, "Implementation and performance of Portals 3.3 on the Cray XT3," in *Proceedings of the 2005 IEEE International Conference on Cluster Computing*, September 2005.
- [5] R. Brightwell, T. Hudson, K. T. Pedretti, and K. D. Underwood, "SeaStar interconnect: Balanced bandwidth for scalable performance," *IEEE Micro*, vol. 26, no. 3, May/June 2006.
- [6] *Pallas MPI Benchmarks*, <http://http://www.pallas.com/e/products/pmb/index.htm>.
- [7] P. Luszczek, J. Dongarra, D. Koester, R. Rabenseifner, R. Lucas, J. Kepner, J. McCalpin, D. Bailey, and D. Takahashi, "Introduction to the HPC challenge benchmark suite," March 2005, <http://icl.cs.utk.edu/hpcc/pubs/index.html>.
- [8] D. J. Kerbyson, H. J. Alme, A. Hoisie, F. Petrini, H. J. Wasserman, and M. Gittings, "Predictive performance and scalability modeling of a large-scale application," in *Proceedings of the ACM/IEEE International Conference on High-Performance Computing and Networking (SC'01)*.
- [9] M. A. Bender, D. P. Bunde, E. D. Demaine, S. P. Fekete, V. J. Leung, H. Mejer, and C. A. Phillips, "Communication-aware processor allocation for supercomputers," in *International Workshop on Algorithms and Data Structures*, August 2005.