2
0
0
9

R&D 100

CNW

Catamount N-Way Lightweight Kernel

Sandia National Laboratories

SUBMITTING ORGANIZATION

Sandia National Laboratories
PO Box 5800
Albuquerque, NM 87185-1319
USA
Ron Brightwell
Phone: (505) 844-2099
Fax: (505) 845-7442
rbbrigh@sandia.gov

AFFIRMATION: I affirm that all information submitted as a part of, or supplemental to, this entry is a fair and accurate representation of this product.

Ron Brightwell

JOINT ENTRY

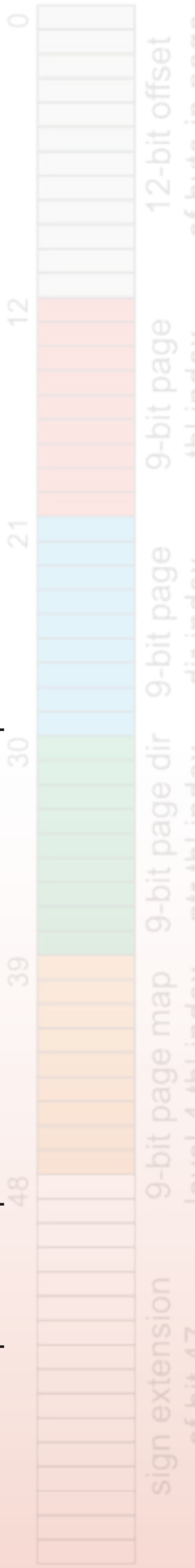
Operating Systems Research
1527 16th NW #5
Washington, DC 20036
USA
Trammell Hudson
Phone: (240) 283-1700
Fax: (843) 971-9774
hudson@osresearch.net

PRODUCT NAME

Catamount N-Way (CNW) Lightweight Kernel

BRIEF DESCRIPTION

CNW is an operating system that exploits existing features of multi-core processors to deliver significant improvements in data access performance for today's parallel computing applications.



*Cover: CNW is Sandia's operating system for the Cray Red Storm supercomputer.
Photo by Randy Montoya, Sandia National Laboratories.*

PRODUCT FIRST MARKETED OR AVAILABLE FOR ORDER

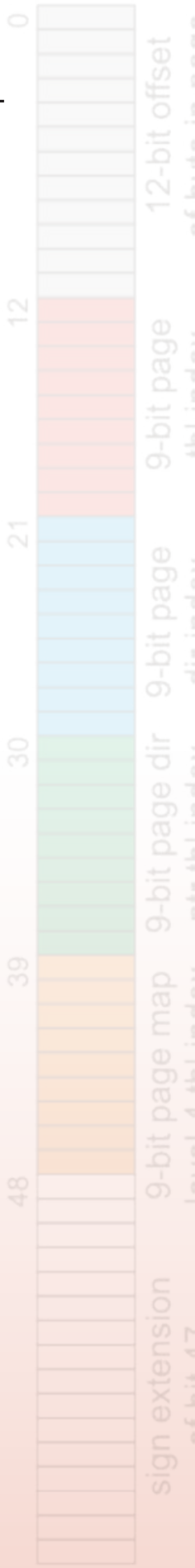
CNW was deployed as the production compute node operating system on Sandia National Laboratories' Cray Red Storm computer in May 2008.

INVENTORS OR PRINCIPAL DEVELOPERS

Ron Brightwell
Principal Member of Technical Staff
Sandia National Laboratories
PO Box 5800
Albuquerque, NM 87185-1319
USA
Phone: (505) 844-2099
Fax: (505) 845-7442
rbrigh@sandia.gov

Kurt Ferreira
Limited Term Member of Technical Staff
Sandia National Laboratories
PO Box 5800
Albuquerque, NM 87185-1319
USA
Phone: (505) 844-0433
Fax: (505) 845-7442
kbferre@sandia.gov

James Laros
Principal Member of Technical Staff
Sandia National Laboratories
PO Box 5800
Albuquerque, NM 87185-1319
USA
Phone: (505) 845-8532
Fax: (505) 845-7442
jhlaros@sandia.gov



Suzanne Kelly
Distinguished Member of Technical Staff
Sandia National Laboratories
PO Box 5800
Albuquerque, NM 87185-1319
USA
Phone: (505) 845-9770
Fax: (505) 845-7442
smkelly@sandia.gov

Kevin Pedretti
Senior Member of Technical Staff
Sandia National Laboratories
PO Box 5800
Albuquerque, NM 87185-1319
USA
Phone: (505) 844-1399
Fax: (505) 845-7442
ktpedre@sandia.gov

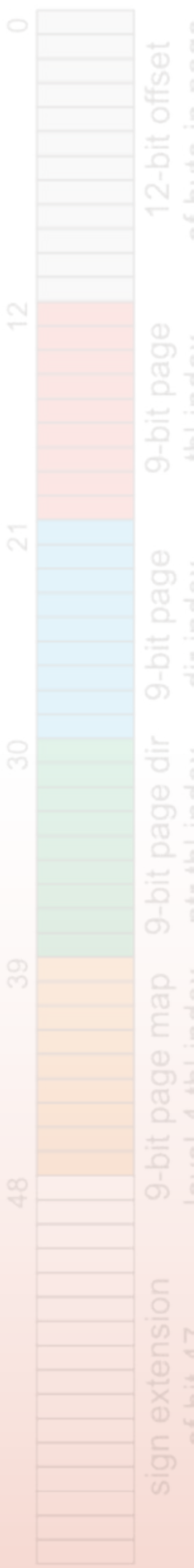
James Tomkins
Senior Scientist (retired)
Sandia National Laboratories
PO Box 5800
Albuquerque, NM 87185-1319
USA
Phone: (505) 845-7249
Fax: (505) 845-7442
jltomki@sandia.gov

John VanDyke
Distinguished Member of Technical Staff
Sandia National Laboratories
PO Box 5800
Albuquerque, NM 87185-1319
USA
Phone: (505) 845-7248
Fax: (505) 845-7442
jpvandy@cs.sandia.gov

Courtenay Vaughan
Senior Member of Technical Staff
Sandia National Laboratories
PO Box 5800
Albuquerque, NM 87185-1319
USA
Phone: (505) 845-7277
Fax: (505) 845-7442
ctvaugh@sandia.gov

Robert Ballance
Principal Member of Technical Staff
Sandia National Laboratories
PO Box 5800
Albuquerque, NM 87185-0807
USA
Phone; (505) 284-9361
Fax: (505) 844-6307
raballa@sandia.gov

Trammell Hudson
Operating Systems Research
1527 16th NW #5
Washington, DC 20036
Phone: (240) 283-1700
Fax: (843) 971-9774
hudson@osresearch.net

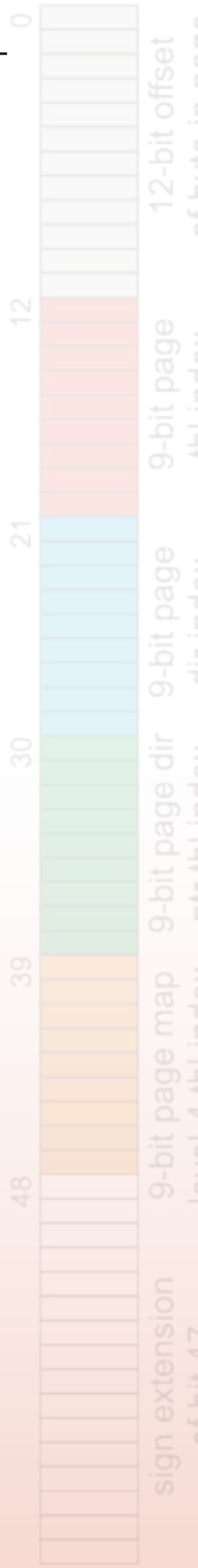


PRODUCT PRICE

The CNW software is licensed to Cray, Inc., at a non-disclosed price.

PATENTS OR PATENTS PENDING

None.



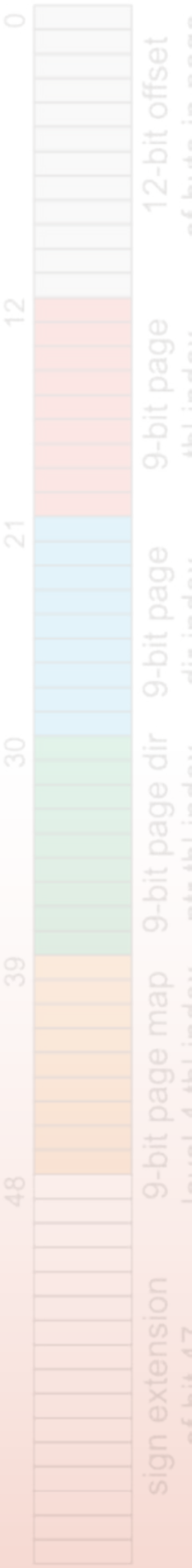
PRODUCT'S PRIMARY FUNCTION

The CNW operating system leverages the existing hardware capabilities of multi-core processors to deliver significant improvements in data access performance for today's parallel computing applications. CNW provides enhanced data access capabilities beyond other equivalent operating systems by employing a new technique that targets memory bandwidth, arguably the most important area of performance in scientific parallel computing.

The impact of commodity multi-core processors on the general computing community has been extensive. One can no longer assume that an application will run faster on successive generations of processors unless the application has been parallelized to take advantage of the increasing core count. Realizing the ramifications of this shift, commodity processor vendors such as Intel and AMD, along with software vendors such as Microsoft, have invested tens of millions of dollars specifically to research strategies for effective utilization of multi-core processors.

The parallel computing community has also been adversely impacted by multi-core processors. While the increase in computation power and density from multi-core processors is encouraging, the majority of scientific parallel computing applications depend as much on memory subsystem performance as on compute performance. To make the problem worse, the message-passing-based model upon which nearly all scalable parallel applications are based decreases the limited memory bandwidth available to a processor. Thus, as the core count increases, the rate at which data can be given to the individual cores for processing decreases.

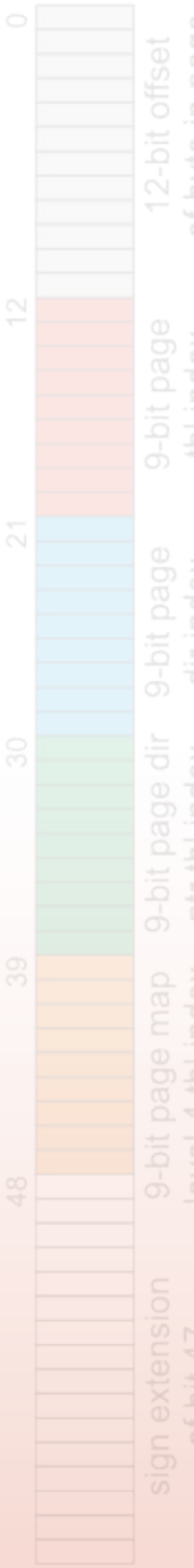
In an effort to address this potentially severe performance limitation, several organizations in the high-performance computing community are beginning to invest heavily in redesigning and rewriting existing parallel applications using alternatives to the existing message-passing-based programming model and traditional programming languages. For an organization such as the US Department of Energy, which has invested nearly $4B over the last 13 years in the development and maintenance of message-passing-based modeling and simulation applications in the Advanced Simulation and Computing Program alone, the prospect of such an undertaking is both pressing and daunting.



CATAMOUNT N-WAY LIGHTWEIGHT KERNEL

2009

Here's how CNW provides an innovative, cost-effective solution: Rather than explore costly and complex changes to applications, CNW attacks the problem at the operating system level. Using the existing features of a multi-core processor to cut the memory bandwidth requirement of message-passing-based applications in half, CNW provides new capabilities that significantly increase the raw performance of critical message-passing operations. These improvements greatly help maintain the viability of existing parallel applications on quad-core processors and become increasingly more important on successive generations of multi-core processors.



PRODUCT'S COMPETITORS

In addition to CNW, the ten fastest parallel computers in the world on the Top 500 list (www.top500.org) run the following other operating systems:

- Linux – open-source UNIX operating system
- CNK – IBM lightweight kernel modeled after CNW predecessors
- Microsoft Windows

None of these operating systems natively provides the enhanced technology for shared data access for message-passing-based parallel applications that CNW supplies.

COMPARISON MATRIX

For message-passing-based parallel applications running on a node composed of multi-core processors, CNW provides the following key capabilities relative to its competitors.

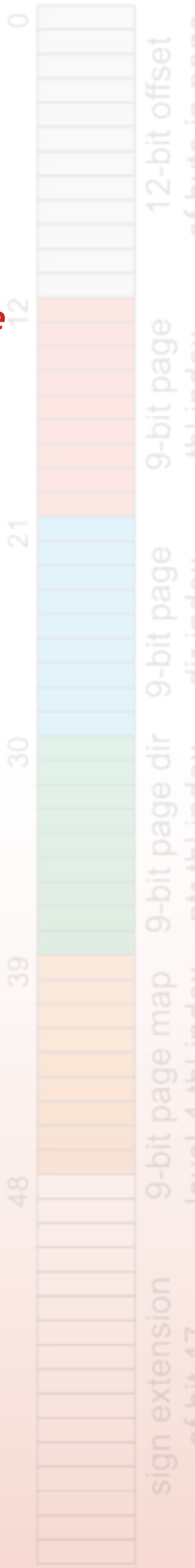
CAPABILITY	CNW	LINUX	CNK	WINDOWS
Single-copy message passing	Yes	No	No	No
In-place collective operations	Yes	No	No	No
Threaded collective operations	Yes	No	No	No
Support for one-sided operations	Yes	No	No	No
Preserves investment in existing applications	Yes	No	No	No

HOW PRODUCT IMPROVES ON COMPETITION

CNW is superior to the competitive operating systems for the following reasons:

- CNW cuts the required memory bandwidth for intra-node message passing in half.
- CNW enables in-place collective operations, which significantly reduces the memory bandwidth requirement of important message-passing operations.
- CNW enables threaded reduction operations, which provides a significant performance optimization for a critical message-passing operation.
- CNW enables high-performance, one-sided data movement operations.
- CNW leverages the existing capabilities of a multi-core processor.
- CNW enables these features in approximately 25 lines of operating system code.
- CNW helps preserve the multi-billion dollar investment made in scientific parallel applications.

CNW helps preserve the multi-billion dollar investment made in scientific parallel applications.

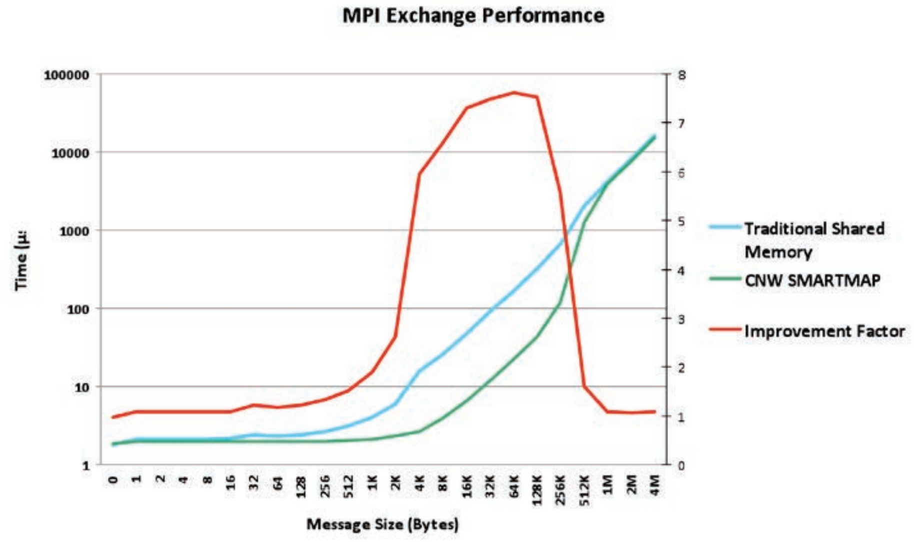


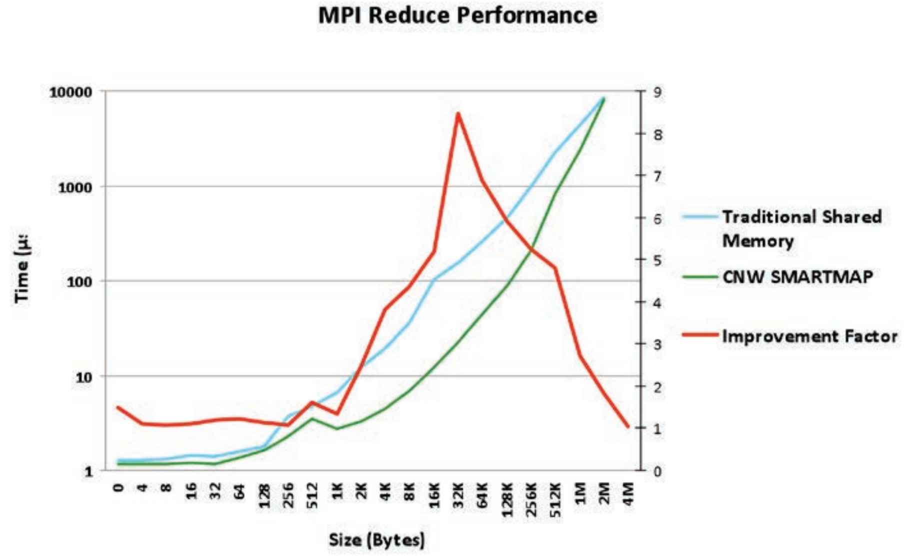
CNW makes processes running on a multi-core processor behave as if both processes were running in their own independent address space and as threads that run within a single address space. For message-passing applications, this capability creates significant performance improvements for the raw data transfer mechanism used to communicate between processes. It also provides tremendous gains in performance for an important set of fundamental operations that require multi-process communication and coordination. CNW achieves these capability enhancements with only approximately 25 lines of additional operating system code.

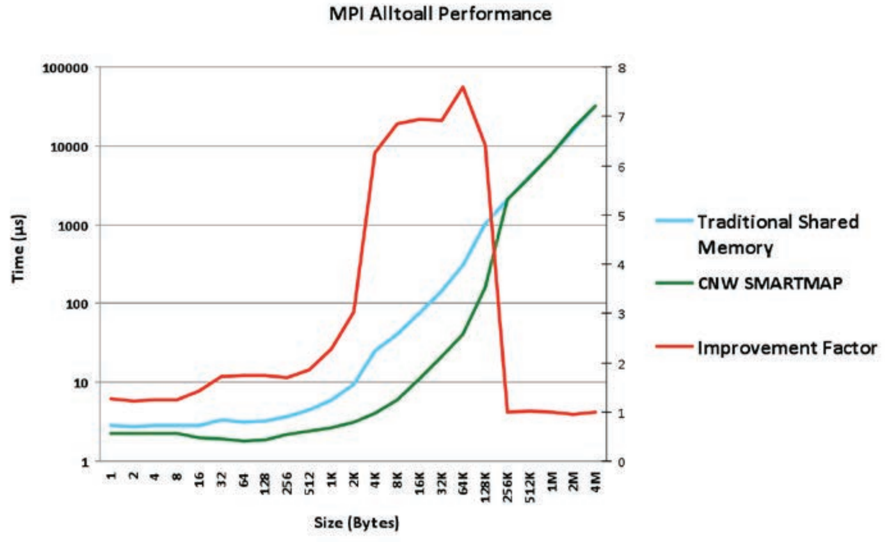
For example, the following three graphs are from the Intel MPI Benchmark Suite running on a 2.2 GHz Quad-Core AMD Opteron processor. The plots show the performance of using a traditional shared memory approach that CNW's competitors use versus the CNW SMARTMAP approach (SMARTMAP allows the application processes on a multi-core processor to directly access each others' memory without the overhead of kernel involvement). The red line shows the

CATAMOUNT N-WAY LIGHTWEIGHT KERNEL

2009


MPI Exchange Performance

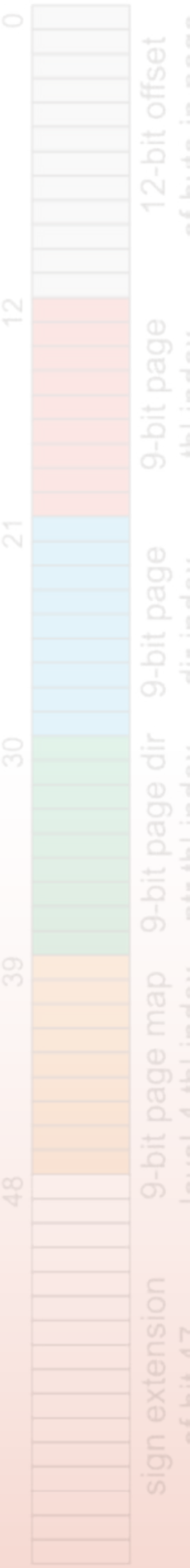

MPI Reduce Performance


MPI Alltoall Performance

CNW achieves these capability enhancements with only approximately 25 lines of additional operating system code.

improvement factor. So, for 64 KB messages, CNW is more than seven times faster for message-passing interface (MPI) exchanges than a standard shared memory approach. The second graph is the threaded reduction operation (MPI_Reduce), and the last graph is a complete exchange (MPI_Alltoall). An important consideration for this data is that this is only on a quad-core processor; we expect the performance to increase as the core count increases.

Ultimately, CNW provides the ability for the majority of existing scientific parallel applications to more effectively use multi-core processors, thereby extending the lifetime of current applications and potentially avoiding costly application-level modifications.

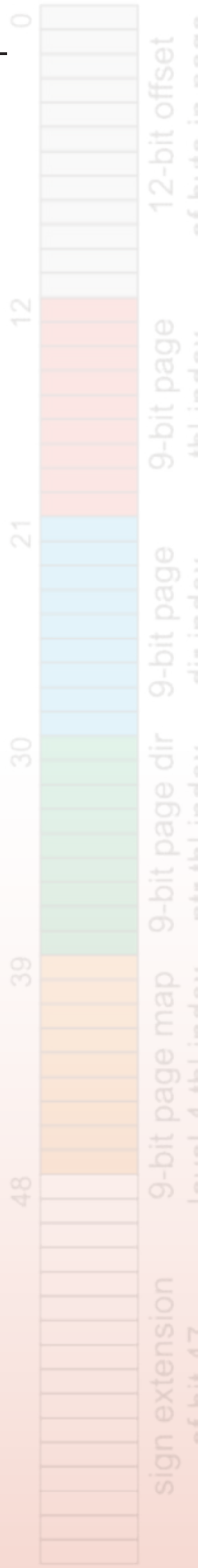


PRODUCT'S PRINCIPAL APPLICATIONS

CNW manages the resources of the compute node of a massively parallel processing system.

OTHER APPLICATIONS

The multi-core processor optimizations in CNW have been implemented and demonstrated for one family of commodity multi-core processors (X86-64), but a similar approach is viable on other multi-core processors, such as the POWER processors from IBM and the SPARC processors from Sun Microsystems. The CNW approach for optimizing intra-node data transfers would be straightforward to implement in other lightweight or embedded operating systems.



CATAMOUNT N-WAY LIGHTWEIGHT KERNEL

R&D 100 Entry

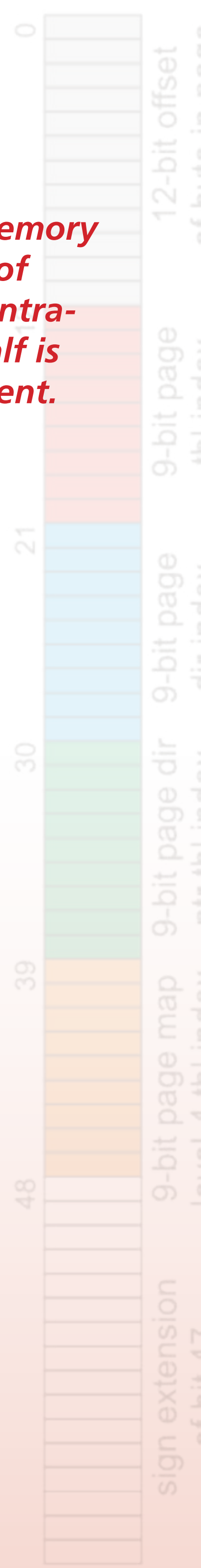
2009

SUMMARY

The Catamount N-Way (CNW) lightweight kernel is an example of how a small innovation in operating system implementation can dramatically impact the capabilities of existing parallel computing applications. CNW offers significant performance improvements for the fundamental operations on which these applications rely. The investment in developing the multi-core enhancements for CNW pales in comparison to the multi-million dollar investment that some organizations are undertaking to maximize application performance on multi-core processors.

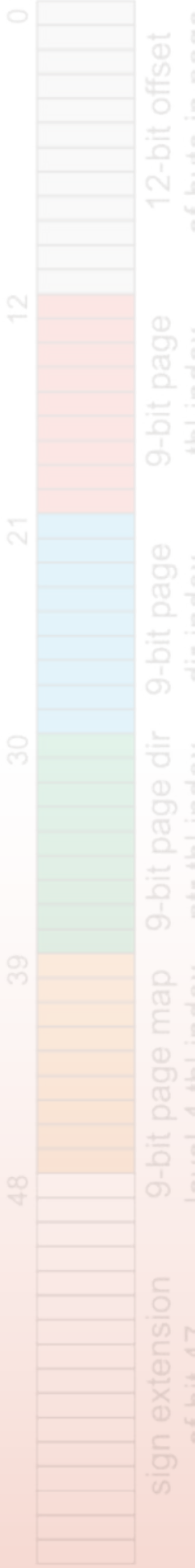
...the ability to cut the memory bandwidth requirement of parallel applications for intra-node data transfers in half is an exceptional achievement.

Experimental data show significant performance advantages for all key low-level data sharing capabilities upon which nearly every existing scientific parallel computing application relies. When considering the continued increase in the core count of commodity processors, together with the corresponding stress placed on the memory subsystem to keep pace, the ability to cut the memory bandwidth requirement of parallel applications for intra-node data transfers in half is an exceptional achievement. Further, CNW's ability to provide new capabilities that significantly increase the raw performance of several key group-based communication operations is an extraordinary and noteworthy accomplishment.



CONTACT PERSON

Robert W. Carling, Director
Sandia National Laboratories
PO Box 969
Mail Stop 9405
Livermore, CA 94551-0969
USA
Phone: (925) 294-2206
Fax: (925) 294-3403
rwcarli@sandia.gov



APPENDICES ITEMS

APPENDIX ITEM A

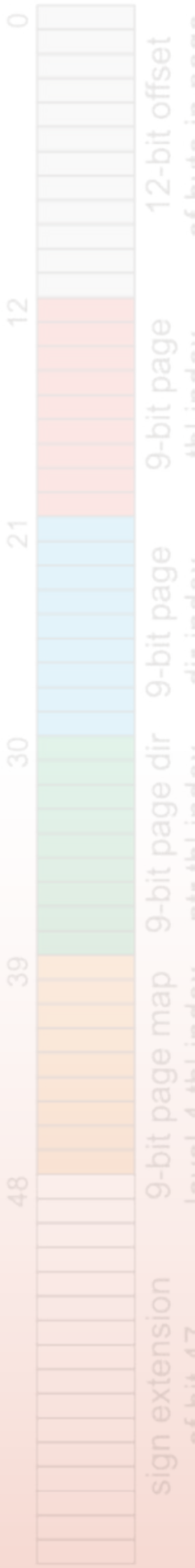
Red Storm: A New Dimension in Computing Capability

APPENDIX ITEM B

SMARTMAP: Operating System Support for Efficient Data Sharing Among Processes on a Multi-Core Processor (Excerpt)

APPENDIX ITEM C

References Used in This Submission



APPENDIX ITEM A

RED STORM

A NEW DIMENSION IN COMPUTING CAPABILITY

The 40 teraOPS Red Storm supercomputer at Sandia National Laboratories is critical to meeting NNSA deliverables for the Nuclear Weapons Stockpile. Red Storm scored the highest on two of the five High Performance Computing Challenge (HPCC) benchmarks.

Early science runs on Red Storm during acceptance testing on the machine included ocean and climate simulations that scaled to the maximum available 10,368 processors.

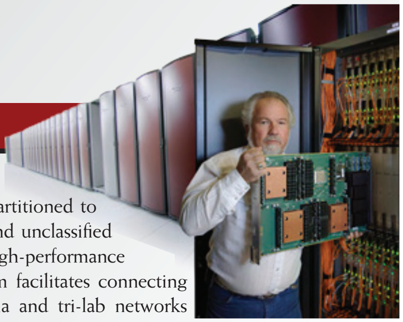
Red Storm is the result of a partnership between Sandia National Laboratories and Cray, Inc. The SeaStar™ interconnect chip is key to the three-dimensional mesh topology that enables increased efficiency and scaling for engineering applications.

All three NNSA laboratories are running large-scale weapon simulations for the stockpile on Red Storm, including calculations to support qualification of the W76-1.

Introducing Red Storm to the news media at Sandia National Laboratories, NNSA Administrator Linton Brooks said, *"Some of my colleagues in the scientific community now say that scientists in the future will grow up thinking there is theory, there is experiment, and there is simulation—three ways in which we advance scientific knowledge."* —February 8, 2006

Advanced Simulation & Computing™
NNSA
National Nuclear Security Administration

Introducing Red Storm



Red Storm is a massively parallel processing (MPP) supercomputer at Sandia National Laboratories/New Mexico that recently joined the ranks of the Advanced Simulation & Computing (ASC) supercomputers monopolizing world computing records. Red Storm was uniquely designed by Sandia and Cray, Inc., to address the highly complex nuclear weapons stockpile computing problems that particularly characterize the simulations required by an engineering laboratory such as Sandia. Red Storm allows modeling and simulation of complex problems in nuclear weapons stockpile stewardship that were only recently thought impractical, if not impossible.

ASC researchers at Los Alamos and Lawrence Livermore are also finding it a valuable resource.

Red Storm is partitioned to support classified and unclassified operations. Its high-performance Input/Output system facilitates connecting with external Sandia and tri-lab networks and storage. Red Storm is scalable from one cabinet to hundreds up to 10,000 processors. The architecture is scalable to greater than 100 teraOPS.

Sandia's collaboration with Cray supports commercialization of the technology. This not only increases national competitiveness in supercomputing, but results in a wide user knowledge base to detect and fix errors and problems.

What Features Make Red Storm Unique?

The ASC computing resources offer different advantages for different kind of applications. Red Storm specializes in MPP problems that require considerable interaction and coordination of the high-volume AMD processors running an application.

Red Storm was constructed of commercial off-the- shelf parts supporting the custom IBM-manufactured SeaStar interconnect chip. The interconnect chips, one of which accompanies each of 10,368 AMD Opteron™ compute node processors, make it possible for Red Storm's processors to pass data to one another efficiently while applications are running. The interconnect is also key to the three-dimensional mesh that allows 3-D representations of complex problems. Red Storm holds a world record in visualization and two of the High Performance Computing Challenge (HPCC) benchmarks, PTRANS and RandomAccess.

A flat architecture is another unique feature. A simple design means that information can pass more directly from processor to processor without having to pass through many levels of processors in a complex hierarchy. The Catamount operating software runs the application with a user-friendly Linux system serving as the user interface.

How Does Red Storm Excel?

Red Storm excels in the three HPCC benchmarks that are particularly relevant to the kinds of problems it was designed to address: STREAM, PTRANS, and RandomAccess. It has the highest performance in the latter two when compared to the current baseline submissions for the HPCC benchmarks.

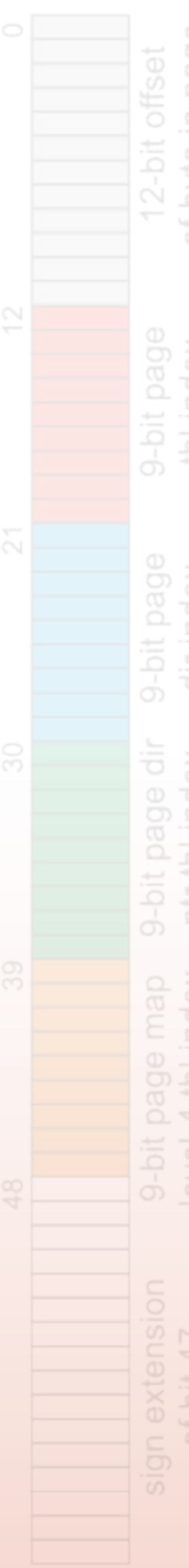
- STREAM measures sustainable memory bandwidth. Red Storm's higher memory bandwidth keeps the processors from being starved for data and makes the Central Processing Unit more efficient.
- PTRANS is a useful measure for the total communications capacity of the internal interconnect. This high score means that data can flow freely between the 10,368 Red Storm processors without bottlenecks or congestion.
- RandomAccess indicates the performance in moving individual data elements as opposed to long arrays of data. Red Storm has the ability to coordinate the many interactions of various processors.

What Are Red Storm's Applications?

Red Storm's primary use is in U.S. nuclear stockpile work: designing new replacement components, virtual testing of components under hostile, abnormal, and normal conditions, and assisting in weapons engineering and weapons physics.

For more information, contact Sandia ASC communications: Reeta Garber (ragarbe@sandia.gov), Michael Townsend (mtowns@sandia.gov).

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.



APPENDIX ITEM B (Excerpt)

SMARTMAP: Operating System Support for Efficient Data Sharing Among Processes on a Multi-Core Processor

Ron Brightwell and Kevin Pedretti
Scalable System Software Department
Sandia National Laboratories
Albuquerque, New Mexico 81785–1319
{rbbrigh,ktpedre}@sandia.gov

Trammell Hudson
Operating Systems Research
1527 16th NW #5
Washington, DC 20036
hudson@osresearch.net

***Abstract*—This paper describes SMARTMAP, an operating system technique that implements fixed offset virtual memory addressing. SMARTMAP allows the application processes on a multi-core processor to directly access each other's memory without the overhead of kernel involvement. When used to implement MPI, SMARTMAP eliminates all extraneous memory-to-memory copies imposed by UNIX-based shared memory strategies. In addition, SMARTMAP can easily support operations that UNIX-based shared memory cannot, such as direct, in-place MPI reduction operations and one-sided get/put operations. We have implemented SMARTMAP in the Catamount lightweight kernel for the Cray XT and modified MPI and Cray SHMEM libraries to use it. Micro-benchmark performance results show that SMARTMAP allows for significant improvements in latency, bandwidth, and small message rate on a quad-core processor.**

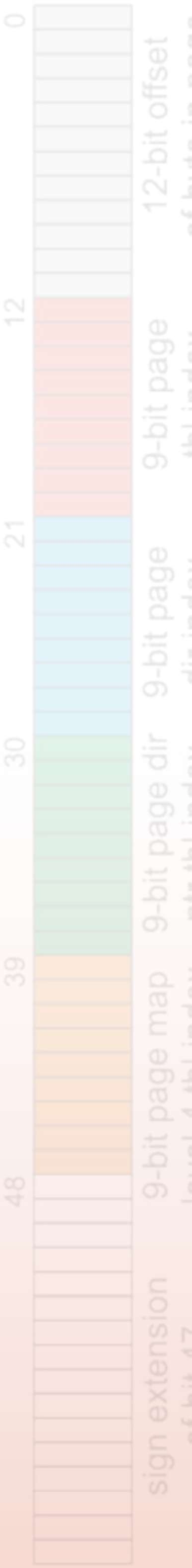
I. Introduction

As the core count on processors used for high-performance computing continues to increase, the performance of the underlying memory subsystem becomes significantly more important. In order to make effective use of the available compute power, applications will likely have to become much more sensitive to the way in which they access memory. Applications that are memory bandwidth bound will need to avoid any extraneous memory-to-memory copies. For many applications, the memory bandwidth limitation is compounded by the fact that the most popular and effective parallel programming model, MPI, mandates copying of data between processes. MPI implementors have worked to make use of shared memory for communication between processes on the same node. Unfortunately, the current schemes for using shared memory for MPI can require either excessive memory-to-memory copies or potentially large overheads inflicted by the operating system (OS).

In order to avoid the memory copy overhead of MPI altogether, more and more applications are exploring mixed-mode programming models where threads and/or compiler directives are used on-node and MPI is used off-node. Unfortunately, the complexity of shared memory programming using threads has hindered both the development of applications as well as the development of thread-safe and thread-aware MPI implementations. The initial attractiveness of mixed-mode programming was tempered by the additional complexity induced by finding multi-level parallelism and by initial disappointing performance results [1], [2], [3]. Recently, however, unpublished data on mixed-mode applications suggest more encouraging results on multi-core processors.

In this paper, we introduce a scheme for using fixed-offset virtual address mappings for the parallel processes within a node to enable efficient direct access shared memory. This scheme, called Simple Mapping of Address Region Tables for Multi-core Aware Programming, or SMARTMAP, achieves a significant performance increase for on-node MPI communications and eliminates all of the extraneous memory-to-memory copies that shared memory MPI implementations incur. SMARTMAP can also be used for more than MPI. It maps very well to the partitioned global address space (PGAS) programming model and can be used to implement one-sided get/put operations, such as those available in the Cray SHMEM model. This strategy can also be used directly by applications to eliminate the need for on-node memory-to-memory copying alto-

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.



gether.

The main contributions of this paper are:

- an OS virtual memory mapping strategy that allows direct access shared memory between processes on a multi-core processor
- a description of how this strategy can be used for on-node data movement between processes on a multi-core processor
- a detailed analysis of the performance impacts of using this strategy for MPI peer communication, MPI collective communication, and Cray SHMEM data movement operations

The rest of this paper is organized as follows. The next section provides background on the current approaches to using shared memory for intra-node data movement. In Section III, we describe the implementation of the SMARTMAP and its advantages over other approaches. Section IV provides a detailed description of the enhancements that we have made to MPI and SHMEM implementations on the Cray XT to use it. Section V presents performance results using several micro-benchmarks. Relevant conclusions of this paper are summarized in Section VI, and we close by discussing possible avenues of future work in Section VII.

II. Background

POSIX-based operating systems generally support shared memory capability through two fundamental mechanisms: threads and memory mapping. Unlike processes, which allow for a single execution context inside an address space, threads allow for multiple execution contexts inside a single address space. When one thread updates a memory location, all of the threads sharing the same address space also see the update. A major drawback of threads is that great care must be taken to ensure that common library routines are reentrant, meaning that multiple threads could be executing the same piece of code simultaneously. For non-reentrant functions, some form of locking must be used to ensure atomic execution. The same is true for data accessed by multiple threads – updates must be atomic with respect to one another or else difficult to debug race conditions will occur. Race conditions and fundamentally non-deterministic behavior make threads difficult to use correctly.

In memory mapping, cooperating processes request a shared region of memory from the operating system and then map it into their private address space, possibly at a different virtual address in each process. Once initialized, a process may access the shared memory region in exactly the same way as any other memory in its private address space. As with threads, updates to shared data structures in this region must be atomic.

Explicit message passing is an alternative to shared memory for intra-node data sharing. In message passing, processes pass messages carrying data between one another. No data is shared directly, but rather is copied between processes on an as necessary basis. This eliminates the need for re-entrant coding practices and careful updates of shared data, since no data is shared. The main downside to this approach is the extra overhead involved in copying data between processes.

In order to accelerate message passing, memory mapping is often used as a high-performance mechanism for moving messages between processes [4]. Unfortunately, such approaches to using page remapping are not sufficient to support MPI semantics, and general-purpose operating systems lack the appropriate mechanisms. The sender must copy the message into a shared memory region and the receiver must copy it out – a minimum of two copies must occur. It would be ideal if messages could be moved directly between the two processes with a single copy. This would be possible if all processes operated entirely out of the shared memory region, but this would amount to the processes essentially becoming threads, with all of their inherit problems. Furthermore, message passing APIs such as MPI allow message buffers to be located anywhere in an address space, including the process's data, heap and stack.

As of MPI 2.0, MPI applications may make use of both threads and memory mapping, although few MPI implementations provide full support for threads. More commonly, MPI implementations utilize memory mapping internally to provide efficient intra-node communication. During MPI initialization, the processes on a node elect one process to create the shared memory region and then the elected process broadcasts the information about the region to the other processes on the node (e.g., via a file or the sockets API). The other processes on the node then "attach" to the shared memory region, by requesting that the OS map it into their respective address spaces.

Note that the approach of using shared memory for intra-node MPI messages only works for the point-to-point operations, collective communication operations, and a subset of the MPI-2 remote memory access operations. Copying mandates active participation of the two processes involved in the transfer. Single-sided put/get operations, such as those in the Cray SHMEM programming interface, cannot be implemented using POSIX shared memory.

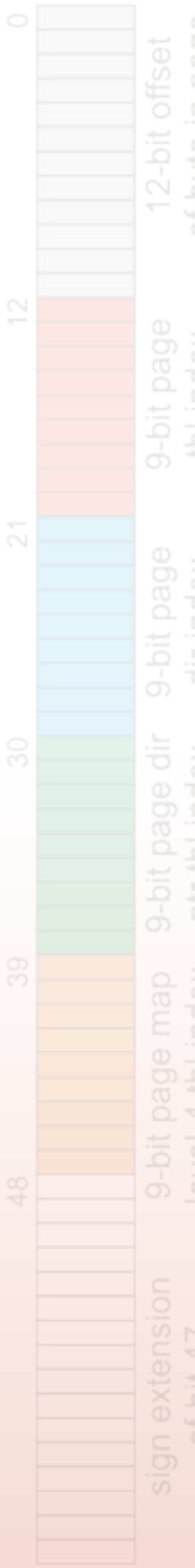
A. Intra-Node MPI

There are several limitations in using regions of shared memory to support intra-node MPI [5], [6], [7]. First, *APPENDIX ITEM C*

References Used in This Submission

Red Storm: A New Dimension in Computing Capability. National Nuclear Security Administration (NNSA) and Sandia National Laboratories' Advanced Simulation & Computing brochure, March **2006.**

SMARTMAP: Operating System Support for Efficient Data Sharing Among Processes on a Multi-Core Processor. Brightwell, R.; Pedretti, K.; Hudson, T. Proceedings of the 2008 IEEE/ACM International Conference on High-Performance Computing, Networking, Storage, and Analysis (SC'08), Austin Texas, November **2008**.



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.
SAND2009-1490P. Designed by the Sandia Creative Group. (505) 284-3181. SP•135391•03/09