The Cray® XC™ Series
Scalability Advantage

Duncan Roweth, senior principal engineer
Mark Atkins, engineer
Kim McMahon, staff engineer

Cray Inc.
## Table of Contents

- Executive Summary ................................................................. 3
- Introduction .................................................................................. 4
- Collectives – Intracnode and Internode Performance Optimizations ........................................ 8
- Memory Use – Dynamic Allocation Minimizes Memory Footprint ............................................. 11
- Job Startup Times – Improving System Utilization and Efficiency ........................................... 14
- Power Consumption – Improved Time to Solution Improves TCO ......................................... 15
- Conclusions ................................................................................. 16
Executive Summary

To provide breakthrough advantages in performance and scalability, Cray® XC™ series supercomputers integrate an HPC-optimized Aries™ interconnect with a high-bandwidth, low-diameter network topology called Dragonfly, providing state-of-the-art performance on the most demanding applications. The paper describes aspects of the Cray XC system that was designed for scalability, enabling Cray to deliver systems like “Trinity.”

The Trinity supercomputer is designed to provide computational capability for the demanding workloads of the National Nuclear Security Administration (NNSA). The XC series project is managed and operated by the Alliance for Computing at Extreme Scale (ACES), a partnership between Los Alamos National Laboratory and Sandia National Laboratories. The Trinity system is located in Los Alamos at the Nicholas C. Metropolis Center for Modeling and Simulation.

This paper addresses important scalability metrics and then clearly demonstrates that the example Trinity system performs well in these critical areas, including job startup times, memory footprint, collective latency, power consumption, energy-to-solution and the affordability of high-bandwidth networks.

As applications increase in size the behavior of collective operations can limit performance. The Cray XC system accelerates these key operations, both between nodes and within each node. That allows more CPUs to be used on a given problem, reducing time to solution.

The memory footprint of the communication stack is increasingly important as job sizes increase. Cray has recently demonstrated the ability to run MPI jobs with 2 million ranks. In this paper we present results of a comparison that show the Cray XC system’s memory use per process being 30 times smaller than that of EDR Infiniband. This metric is particularly important when considering the limited amounts of fast on-package memory in modern CPUs.

This paper presents results on startup times for large parallel jobs. Startup times typically rise with the job size, meaning that more CPUs are tied up for longer. This paper presents results of new work minimizing startup times, improving system utilization and efficiency at scale.

The cost of electricity is an increasingly important element of TCO for all large scale systems. Network energy consumption is widely thought to be a small proportion of the total. This paper demonstrates that using highly integrated network components and a low-diameter Dragonfly network can provide substantial savings.

The sheer scale and high global bandwidth of XC systems enable Cray customers to study new problems that simply would not run on other systems, opening the door to discovery.
Introduction

This white paper considers important aspects of the scalability of Cray® XC™ systems, their Aries™ interconnect and the associated software. It includes performance data measured on the “Trinity” supercomputer system.

Work that formed part of the Defense Advanced Research Projects Agency’s (DARPA) High Productivity Computing Systems (HPCS) Cascade program, from which the Cray XC system was developed, combined with work undertaken for the Trinity system has resulted in a platform that is well suited to running MPI applications at scale. Results are presented for tests using in excess of 300,000 MPI ranks. Early testing on the Cray XC series Xeon Phi™ node included jobs with up to 2 million MPI ranks.

The Trinity supercomputer at Los Alamos National Laboratory is designed to provide increased computational capability for the NNSA Nuclear Security Enterprise. Trinity is a Cray XC system with a mix of Intel® Xeon® (previously codenamed Haswell) and Intel® Xeon Phi™ (previously codenamed Knights Landing, or “KNL”) processors. The first phase, installed in 2015, comprises 9,420 Intel Xeon compute nodes, each with 128 GB of DDR4 memory. The Trinity system will increase in size, to more than 19,000 nodes, during 2016 with the addition of 9,984 Intel Xeon Phi nodes. Peak performance of Trinity is in excess of 40 PF.

Figure 1. The Trinity system at Los Alamos National Laboratory

Compute Node Architecture Flexibility Optimized to Support Diverse Arithmetically Intense and Memory Bandwidth Intense Applications

Trinity is the first instantiation of an Advanced Technology system, as defined by the NNSA’s Advanced Simulation and Computing (ASC) Program, one that will establish the technological foundation necessary to build exascale computing environments. It must achieve a balance between usability and performance on current simulation codes while supporting adaptation to new computing technologies and programming methodologies. The NNSA workload includes a wide mix of codes, some of which have high arithmetic intensity and some of which are memory bandwidth limited. Many of the codes run at high scale, making the Cray XC system with Intel Xeon CPUs an excellent platform for these codes.

1 For information on the Trinity system see the Trinity home page or the Trinity to Trinity page tracking the history of innovation at Los Alamos National Laboratory.
Memory bandwidth of the Intel Xeon nodes is 150 GB/s. Memory bandwidth of each Intel Xeon Phi node is 450 GB/s to on-package memory and 120 GB/s to DRAM. The on-package memory can be configured to operate as a cache of the DRAM, or as a separate segment from which processes make explicit allocation. The Cray Linux® Environment (CLE) is unique and flexible in enabling this mode selection to be made by the user on a per-job basis.

A popular and successful Xeon Phi porting strategy is to begin by running existing applications with the Xeon Phi on-package memory configured as a cache and to limit the number of processes per node to keep the cache hit ratios high. This approach provides immediate benefit from the high bandwidth of the on-package memory. Application developers can then make incremental changes to improve use of threading within their codes to increase the number of active cores. The Cray compiler, performance analysis and porting tools, Reveal™ in particular, help in this task. They can also be used to identify frequently (or infrequently) used arrays that might be best explicitly placed in fast on-package (or slower DRAM) memory.

**Cray® DataWarp™ I/O Acceleration Introduces Unique Support for Burst Buffer Storage**

The Trinity system is one of the first to use integrated nonvolatile “burst buffer” storage. Embedded within the high-speed fabric are nodes with attached solid-state disk drives. Cray’s DataWarp burst buffer capability will allow for accelerated checkpoint/restart performance and relieve much of the pressure normally loaded on the back-end file system storage. In addition, the burst buffer will support new workflows such as in-situ analysis. The burst buffer storage tier on Trinity provides 3.7 PB of usable capacity with a sustained bandwidth of 3.3 TB/s. The file system is provided by Cray® Sonexion® storage arrays. This tier provides 78 PB of usable capacity with a sustained bandwidth of 1.45 TB/s.

**One Common High-Performance Interconnect Across Compute, I/O and Burst Mode Blades**

The XC series nodes are connected using the Aries interconnect common to all Cray XC systems. A PCI-Express 3.0 x16 interface connects each node to an Aries network interface card (NIC). Aries uses a high-density system-on-chip (SoC) design in which each ASIC provides four high-performance NICs, a 48-port switch and a load balancer that connects the four NICs to eight of the switch ports. Cray XC systems are constructed from four-node blades (see Figure 2) with one Aries ASIC per blade.

![Figure 2. XC series Xeon Phi (KNL) compute blade](image)
Injection bandwidth is 10 GB/s for each node in the Trinity system. It is interesting to compare the bandwidth-to-flop (BW/flop) ratios of Trinity and those of other large ASC systems that have been recently announced (see Table 1) to understand trade-offs in computation, bandwidth and communications. Higher BW/flop ratios facilitate scalability to larger numbers of nodes whereas lower BW/flop ratios require more computation to take place within each node.

The spread of BW/flop ratios is indicative of the mixed workload on these systems and the different priorities attached to inter-process communication by the various vendors. The BW/flops ratio of the Intel Xeon nodes in Trinity is three times that of the Intel Xeon Phi nodes in Trinity, which in turn is approximately five times that of the Summit (IBM) nodes. BW/flop of the Summit nodes is equivalent to that of a cluster of Xeon nodes connected using 10 gigabit Ethernet – this system has been configured to run highly computationally intensive jobs with little communication. The Xeon nodes in Trinity are configured for memory bandwidth-intensive jobs with frequent communication. The Xeon Phi nodes in Trinity sit in between, with higher peak computational performance than the Trinity Xeon nodes and better BW/flop ratios than those in Summit. Published data on the Aurora (Intel/Cray) system suggests a BW/flop ratio close to that of the Trinity Xeon nodes or 2.5 times that of the Trinity Xeon Phi nodes.

<table>
<thead>
<tr>
<th>System</th>
<th>Vendor</th>
<th>Year of Installation</th>
<th>Network Bandwidth</th>
<th>Bandwidth / Flop</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trinity Xeon</td>
<td>Cray</td>
<td>2015</td>
<td>10 GB/s</td>
<td>0.00849</td>
</tr>
<tr>
<td>Trinity Xeon Phi</td>
<td>Cray</td>
<td>2016</td>
<td>10 GB/s</td>
<td>0.00328</td>
</tr>
<tr>
<td>Summit</td>
<td>IBM</td>
<td>2018/19</td>
<td>25 GB/s</td>
<td>† 0.00063</td>
</tr>
<tr>
<td>Aurora</td>
<td>Intel / Cray</td>
<td>2018/19</td>
<td>25 GB/s</td>
<td>† 0.00714</td>
</tr>
</tbody>
</table>

Table 1. Bandwidth/flops ratios for recently announced ASC systems. † Based on estimated peak performance.

**Scalable HPC Network with Option to Reduce Cost**

All Cray XC systems use the Dragonfly network. Dragonfly groups of 384 nodes are constructed from 96 blades distributed over six chassis in a pair of cabinets. An electrical all-to-all network, constructed from backplane links, connects the blades in each chassis (see Figure 3). A second all-to-all network, constructed using electrical cables, connects the six chassis in each group (see Figure 3). The group-level interconnect is common to all large Cray XC systems. The full system is constructed by connecting Dragonfly groups with an optical all-to-all network (see Figure 4). The system network needs to be optical because of the long cable lengths (up to 50 meters in the Trinity system). High-speed active optical cables (AOCs) remain relatively expensive. Cray customers have the optimization option to taper down the optical network elements to trade off bandwidth versus the expense of optical cables.

---

2 For information on the Aurora (Intel/Cray) system at Argonne National Laboratory see the Aurora home page or a press article collecting material from various sources.
4 For information on Cray XC systems see the Cray white paper “Cray XC Series Network.”
5 In July 2016 the market price for an active optical cable operating at 14 GB/s was around $300.
Optimized to Address Real-World Bandwidth-Intensive Communications Patterns

Bisection bandwidth, the worst case bandwidth available between two halves of a system, is a widely used network metric, although it does not reflect the communication patterns used in HPC applications. The all-to-all network used in XC systems has roughly twice the global bandwidth as it has bisection. This is because half the AOCs cross the bisection and the other half provide connectivity among the groups in each half (see Figure 4). The XC series structure mirrors that of bandwidth-intensive communication patterns such as all-to-all.

In the Trinity XC series system each of the 55 two-cabinet groups is connected to each of the other groups using a pair of 12X AOCs, each rated at 37.5 GB/s. Global bandwidth is 5.2 GB/s per node, high for a system of this size, when compared to the injection bandwidth of 10 GB/s. Global bandwidth for the system as a whole is 111 TB/s; 1,512 of the 2,970 AOCs cross the bisection, and aggregate bisection bandwidth is 56 TB/s.

Reducing Network Costs While Still Providing Full Global Bandwidth

The Dragonfly design used in Cray XC systems can provide full global bandwidth for system sizes of up to 241 groups and 92,544 nodes, approximately four times the size of Trinity. It does so with just one Aries SoC per blade and 2.5 AOCs per four-node blade. There are no external director switches, and half the number of AOCs as an equivalent fat-tree network (see Table 2). The Dragonfly design trades
expensive optical links for low-cost electrical links within a cabinet. All global traffic must take two optical hops in a fat-tree network. In a Dragonfly such traffic takes just one global hop when minimally routed. These simple differences underlie why Dragonfly is so much more cost effective than fat-tree in systems.

<table>
<thead>
<tr>
<th></th>
<th>Dragonfly</th>
<th>Fat-tree</th>
</tr>
</thead>
<tbody>
<tr>
<td>NICs</td>
<td>18,840†</td>
<td>18,840</td>
</tr>
<tr>
<td>Switch enclosures</td>
<td>0</td>
<td>70</td>
</tr>
<tr>
<td>Switch ASICs</td>
<td>4,710</td>
<td>3,240</td>
</tr>
<tr>
<td>Backplane electrical links</td>
<td>36,000</td>
<td>38,880</td>
</tr>
<tr>
<td>Cabled electrical links</td>
<td>36,000</td>
<td>9,420</td>
</tr>
<tr>
<td>Optical links</td>
<td>12,000</td>
<td>22,380</td>
</tr>
</tbody>
</table>

Table 2. Component counts for Trinity-sized system with 50 percent taper. Component counts are in terms of number of links. The Cray XC system uses electrical cables carrying three links and active optical cables carrying four links to reduce cost.

†These NICs form part of the Aries SoC. InfiniBand NICs are separate devices.

A fat-tree network the size of Trinity would not have been affordable. The Dragonfly network used in the Cray XC system is the only full-bandwidth network proven at anything like this scale. Other large systems, such as “Jaguar” (a Cray® XT™ system), “Blue Waters” (a Cray® XE™ system) and “Sequoia” (an IBM® Blue Gene/Q® system) use torus networks with much-reduced global bandwidth.

Collectives – Intranode and Internode Performance Optimizations

MPI collective operations are widely used to coordinate computation and communication in HPC applications. The collective API provides users with portability between systems while allowing implementors to optimize performance and resource usage at scale. To progress communications independently of the processor and accelerate overall performance, the Aries NICs include a collective engine that offloads execution of latency-sensitive collectives.

For some common MPI operations on a system the size of Trinity, such as a barrier or global reduction, operations can be embedded within the Dragonfly network and the arithmetic units in the Aries NICs, eliminating main CPU involvement and improving performance. As an example, a barrier or global reduction is executed using a three-stage tree of branching ratio 32. Global reductions occur in two phases. First is a ready phase in which the leaf nodes join the reduction and the reduction operator is applied as data flows up the tree toward the root node (see Figure 5). Then comes a multicast phase in which the result of the reduction flows from the root down the tree and back to the leaf nodes. Progression of the reduction is managed by the NICs, as is application of the reduction operators. Aries supports the full range of MPI reductions (except MPI_PROD) operating on 32-bit and 64-bit integer and floating-point data types. The reductions are executed by an arithmetic unit in each NIC – there is no main CPU involvement. The Cray white paper “Cray XC Series Network” provides details of the collective engine.

Aries reduction and barrier primitives are non-blocking. Processes on each node provide their contribution to the reduction and check back later for the result. Many reductions can be progressed in parallel (up to 128 per job for a radix-32 tree over MPI_COMM_WORLD) allowing for optimized
implementation of the MPI non-blocking collectives. An MPI application running on Trinity might have up to 301,440 ranks when running on the Haswell nodes or up to 640,560 ranks when running on the KNLs.\(^6\)

**Offload of Collectives to Aries Accelerates Performance**

Figure 6 shows latencies for single-word MPI_Allreduce, MPI_Barrier and MPI_Bcast operations measured on Trinity. Measurements were made for increasing numbers of nodes using 32 processes per node.

\(^6\) Trinity uses the 68-core version of KNL.
Cray Local Shared-Memory Optimizations Help Intranode Performance

While it might seem that performing a collective operation over the processes on one node is intrinsically easier than performing the same operation over all of the nodes in a system the size of Trinity, once a hardware offload is in place to accelerate the internode collectives the time taken within each node becomes significant. The MPI_BARRIER times shown in Figure 6 are split roughly 50-50 between intranode and internode operations at high node counts. Cray has developed local shared-memory optimizations for the MPI collectives that significantly improve performance on high-core-count nodes.

Figure 7 shows the time taken by MPI_Allreduce for increasing numbers of nodes using the standard algorithm (blue), the standard algorithm with a local shared-memory optimization (red), the standard algorithm with offload of the network phase to Aries (green) and the fully optimized version (purple) using both the Aries offload and the local shared-memory optimization. The benefits of both network and local shared-memory optimizations on addressing latency and enabling scaling are clear.

![Figure 7. MPI_Allreduce latencies measured on Trinity with 32 MPI ranks per node](image)

Optimizing MPI Code Execution for Many-Core Architectures

Code optimization is a little different for multi-core versus many-core architectures, and Cray invests in addressing both so that users can focus on their application results. By reducing the overhead of excessive function calls and branch execution, overall application performance can be improved for many-core processors like the Intel Xeon Phi.

Since MPI is typically all scalar code there are many branches, lots of small functions and function calls using pointers. This code does not run as well on Xeon Phi (even when adjusting to the slower CPU frequency), which has a smaller branch target buffer than the Intel Xeon processor. Cray is optimizing the critical path as much as possible for Intel Xeon Phi. This involves more in-lining of small functions, use of a many-core-specific memory and hand-optimizations that avoid taking branches in the critical code paths. These enhancements are made within the Cray MPI library, and users are not required to make code changes.
As we look to future processors with larger numbers of energy efficient cores, the local shared-memory element of the MPI collectives becomes an increasingly important element of the total time. Figure 8 shows single-node MPI_Allreduce latencies for Xeon Phi nodes (with KNL model 7250 CPUs) with up to 68 processes per node using the standard algorithm (green) and the Cray shared-memory optimization (red). The lower core speed and the increased number of cores adversely affect MPI performance within a Xeon Phi node, but the Cray shared-memory optimizations reduce the impact of these factors, bringing the time taken for the intranode element of the MPI_Allreduce close to that of Xeon nodes (with model E5-2699 v4 CPUs) running the same code (blue).

![Figure 8. Single-node MPI_Allreduce latency for KNL and Xeon nodes. Local shared memory optimization (SHM).](image)

**A Record MPI Test at Scale**

Early testing of the Trinity system included a record-breaking run of Cray MPI using 2,001,150 MPI ranks running on 50 KNL cabinets. This test used 8,894 nodes with 225 MPI ranks per node — one MPI rank for nearly every KNL hardware thread in the system.

**Memory Use – Dynamic Allocation Minimizes Memory Footprint**

The amount of memory used by the MPI stack can be a significant barrier to scalability. Each process maintains some state for every other process in the job. The total amount of memory used for interprocessor communication is the sum of the memory used for MPI virtual channel (VC) structures that maintain peer-to-peer state, plus the memory used by the process management interface (PMI), plus the memory used by the transport layer (GNI), and the additional state shared between the processes on a node (MPI rank-to-node mappings, etc.).

The latest versions of Cray MPI have reduced the MPI memory footprint by implementing a dynamic virtual channel feature in which memory for VC structures is allocated only when one rank makes direct contact with another – rather than allocating these structures statically during MPI_Init.
Additionally, optimizations to MPI_Alltoall and MPI_Alltoallv use connectionless RMA-based transfers, removing their requirement to use VC structures. These changes significantly reduce the MPI footprint, especially for applications that use a mix of local point-to-point communications in conjunction with global collectives.

**Cray XC with Cray MPI Uses 30 Times Less Memory per Peer than EDR InfiniBand**

Figure 9 compares memory used by the MPI stack for a 32-process-per-node job running on a Cray XC system and an EDR Infiniband cluster. This like-for-like comparison tracks the free memory per node as the job starts, as it returns from MPI_Init() and as it completes an MPI_Alltoall (A2A) operation. For each line, the vertical offset shows the amount of memory required to run any MPI job, and the slope shows the cost of each additional process. The results (Table 3) show a linear fit to the measured data. The dynamic connection (DC) transport, which is new in EDR Infiniband, reduces the cost of each peer substantially, but EDR Infiniband (with MXM library) still uses 30 times more memory per peer than the Cray XC system (with MPI Toolkit, MPT). Unreliable datagram (UD) remains the most economical InfiniBand transport, with base memory use four times that of the XC system and the cost of each peer 21 times that of the XC system.

![Figure 9. MPI memory use for Cray XC system and EDR InfiniBand. Values plotted are per MPI rank.](image)

It should also be noted that the memory used by the Cray MPI stack is the same before and after the MPI_Alltoall call. This is not the case for the other implementations, the Infiniband RC implementation in particular.

---

7 R-squared values range from 0.96 to 0.99, indicating a high-quality linear fit.
<table>
<thead>
<tr>
<th></th>
<th>Base cost (MB)</th>
<th>Cost per peer MPI_Init (KB)</th>
<th>Cost per peer MPI_Alltoall (KB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cray XC system</td>
<td>23</td>
<td>0.16</td>
<td>0.16</td>
</tr>
<tr>
<td>EDR Infiniband UD</td>
<td>90</td>
<td>3.36</td>
<td>3.41</td>
</tr>
<tr>
<td>EDR Infiniband RC</td>
<td>132</td>
<td>4.84</td>
<td>92.8</td>
</tr>
<tr>
<td>EDR Infiniband DC</td>
<td>141</td>
<td>4.84</td>
<td>7.72</td>
</tr>
</tbody>
</table>

Table 3. MPI memory use for Cray XC system and EDR InfiniBand. Values are per MPI rank.

The comparison made in Figure 9 and Table 3 is for small systems, but even on such systems the memory consumed by the MPI stack can be significant. With 32 processes on each of 56 nodes, Cray MPI requires 960 MB per node; UD, the most economical of the InfiniBand transports, requires 2,880 MB per node.

Figure 10 illustrates per-node memory use by version 7.2.2 of the Cray MPI stack (the release in use on most systems today) and the recently improved version 7.2.4 developed for Trinity and similar systems, where memory use is shown as a function of job size for jobs with 32 processes per node. Per-peer space required by the MPI library has been reduced from 624 to 204 bytes. Another 384 bytes is allocated dynamically as each peer is used for the first time. This and other changes made for the XC series bring the total amount of memory used by the communications library for a 301,248-process job (one process per core on Trinity) down to 89 MB per rank.

Figure 10. Cray MPI memory use, 32 processes per node

Equivalent data for 601,920 MPI ranks, obtained by running 64 processes per node on 9,405 nodes, shows memory usage of 135 MB per rank. This test simulated 64 processes per KNL node by running one MPI rank per hyperthread on the 32-core Intel Xeon processor nodes.

Cray MPI tracks and displays MPI's internal memory usage, reporting minimum and maximum high-water mark by rank or a consolidated summary for the job. This information can be generated for any job by setting an environment variable. The latest version of Cray MPI is MPI 3.1 compliant, with the
exception of dynamic process management support. The Argonne MPICH version 3.2rc1 was merged into version 7.3.0 of MPT (the Cray MPI toolkit) in December 2015. Leveraging MPICH Application Binary Interface (ABI) compatibility, this new feature simplifies maintenance and reduces deployment costs by allowing the same executable to run on Cray XC series or Cray CS cluster systems as on any other Linux cluster, without wrappers or other modifications. The reductions in MPI memory use made for Trinity in MPT version 7.2.4 are now available for all Cray XC systems.

Job Startup Times – Improving System Utilization and Efficiency

Job startup times are an important aspect of scalability. Startup times rise with the job size, meaning that more CPUs are tied up for longer, reducing both the efficiency of large jobs and overall system utilization. Work on improving the efficiency of job startup in the Cray MPI toolkit has resulted in the data measured on Trinity and shown in Figure 11. The time taken to start a 301,248-process job with 32 MPI ranks per node over all of Trinity was just 12 seconds. This startup time includes the completion of MPI_Init() and MPI communication from every rank in the job.

Tests undertaken using 64 MPI ranks per node show a startup time of 22 seconds with 602,496 processes. These tests use a statically linked executable. Where dynamic shared libraries are used on a large system it is important to provision I/O resources that enable them to be loaded quickly.
Power Consumption – Improved Time to Solution Improves TCO

System power for Trinity is expected to vary between 8 MW and 10 MW depending on the application. With energy costs rising as a proportion of TCO\(^8\) it is increasingly important to consider energy to solution, the product of the time a job takes to run and its power consumption.

\[
\text{Energy to solution} = \text{time to solution} \times \text{system power} \times \text{power utilization efficiency}
\]

Each of these three elements requires detailed consideration to minimize the overall cost. Improvements in the efficiency element of the equation are important, but they must be achieved without increasing the time to solution, otherwise the potential improvement will be less than expected and possibly even negated.

Because it does compromise the other elements, reducing time to solution is the most effective element of minimizing energy to solution. As discussed elsewhere in this paper, improvements in scalability have direct benefit in reducing time to solution. The system consumes less energy on a given problem, and can complete more jobs with the same energy.

Power consumption is minimized for a given choice of processor through efficient power conversion and delivery within the system. A Cray XC system rack consumes approximately 80kW under heavy load, around 420W per dual-socket node. The Aries network plays an important part in reducing overall power consumption, at just 15W per node including the active optical links. This compares to around 47.5W per node for an equivalent-sized InfiniBand network.\(^9\) The design of the Cray XC system reduces network power consumption by 625kW on a Trinity-sized system, saving around $3.5 million in running costs over the system’s lifetime.

The power utilization efficiency (PUE) is a datacenter and system-specific multiplier reflecting the overheads of delivering power to a system and extracting the heat it generates. The Cray XC system uses 480V power to minimize the costs of distribution within the datacenter. Voltages are converted down to 48V for distribution within the rack and again to 1V close to the components. The XC system uses warm water cooling to minimize cooling costs. Their typical PUE ratios are in the range of 1.1 to 1.25, depending on the design of the datacenter.

The XC system’s design ensures that systems the size of Trinity operate efficiently with the minimum power required to fulfill their role.

---

\(^8\) A widely used rough order-of-magnitude estimate of electricity costs is $1 million per megawatt per year.

\(^9\) A system the size of Trinity requires a four-stage InfiniBand fat tree. Network power consumption for a large three-stage network is estimated to be 40.9W per node.
Conclusions

This paper demonstrates the scalability of key aspects of Cray XC systems. Work that formed part of the DARPA HPCS Cascade program, from which Cray XC system was developed, combined with work undertaken for the Trinity system, has resulted in a platform that is well suited to running MPI applications at scale. Network tests run on greater than 300,000 MPI ranks (32 per node) show excellent performance, as do those run with 64 MPI ranks per node undertaken in advance of KNL deployment. Early results from benchmarks, acceptance testing and early application use of Trinity are reported in Cray User Group presentations. The Trinity system performs well on a wide range of scalability metrics, including job startup times, memory footprint, collective latency, power consumption and energy-to-solution. The Aries interconnect, Dragonfly network and Cray software stack are key elements of Trinity. Many of the results reported here could not have been achieved without them; others could have been achieved only at disproportionately high cost.

© 2016 Cray Inc. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior permission of the copyright owners.

Cray, the Cray logo and Sonexion are registered trademarks, and XC, XT, XE, DataWarp and Reveal are trademarks, of Cray Inc. Other product and service names mentioned herein are the trademarks of their respective owners.

The Aries trademark is owned by Intel.

10 For information on early applications use on Trinity see the following papers presented at the 2016 Cray User Group meeting:

“Performance on Trinity (a Cray XC40) with Acceptance-Applications and Benchmarks,” Mahesh Rajan (Sandia National Laboratories); Nathan Wichmann, Cindy Nuss, Pierre Carrier, Ryan Olson, Sarah Anderson and Mike Davis (Cray Inc.); Randy Baker (Los Alamos National Laboratory); Erik Draeger (Lawrence Livermore National Laboratory); and Stefan Domino and Anthony Agelastos (Sandia National Laboratories)