Titan: Early experience with the Cray XK6 at Oak Ridge National Laboratory

Arthur S. Bland, Jack C. Wells, Otis E. Messer, Oscar R. Hernandez, James H. Rogers
National Center for Computational Sciences
Oak Ridge National Laboratory
Oak Ridge, Tennessee, 37831, USA
[blandas, wellsjc, bronson, oscar, jrogers, at ORNL.GOV]

Abstract— **In 2011, Oak Ridge National Laboratory began an upgrade to Jaguar to convert it from a Cray XT5 to a Cray XK6 system named Titan. This is being accomplished in two phases. The first phase, completed in early 2012, replaced all of the XT5 compute blades with XK6 compute blades, and replaced the SeaStar interconnect with Cray's new Gemini network. Each compute node is configured with an AMD Opteron™ 6274 16-core processors and 32 gigabytes of DDR3-1600 SDRAM. The system aggregate includes 600 terabytes of system memory. In addition, the first phase includes 960 NVIDIA X2090 Tesla processors. In the second phase, ORNL will add NVIDIA's next generation Tesla processors to increase the combined system peak performance to over 20 PFLOPS. This paper describes the Titan system, the upgrade process from Jaguar to Titan, and the challenges of developing a programming strategy and programming environment for the system. We present initial results of application performance on XK6 nodes.**

Keywords-component; Titan, Jaguar, Cray XK6, hybrid computing

I. INTRODUCTION

Titan, a hybrid Cray XK6 system, is the third generation of major capability computing systems at the Department of Energy (DOE) Office of Science's Oak Ridge Leadership Computing Facility (OLCF) located at the Oak Ridge National Laboratory (ORNL). It is an upgrade of the existing Jaguar system [1] first installed at the OLCF in 2008. The initial upgrade from Cray XT5 to Cray XK6 compute nodes was accepted in February 2012 and consists of 18,688 compute nodes for a total of 299,008 AMD Opteron 6274 "Interlagos" processer cores and 960 NVIDIA X2090 "Fermi" Graphical Processing Units (GPU). The peak performance of the Opteron cores is 2.63 PFLOPS and the peak performance of the GPUs is 638 TFLOPS. In late 2012, the 960 NVIDIA X2090 processors will be removed and replaced with at least 14,592 of NVIDIA's next generation "Kepler" processors with a total system peak performance in excess of 20 PFLOPS.

ORNL is deploying Titan in 2012 as part of the DOE Office of Science's Leadership Computing Facility program in support of its basic and applied science missions. Examples of specific modeling and simulation areas where Jaguar and Titan are being and will be used include better understanding turbulent combustion, creating biofuels from cellulose, designing more efficient materials for photovoltaic cells, understanding the role of turbulence in combustion turbines, building a virtual nuclear reactor to increase power output and longevity of nuclear power plants, modeling the ITER prototype fusion reactor, predicting impacts from climate change, earthquakes and tsunamis, and improving the aerodynamics of vehicles to increase fuel economy. Access to Titan is available through several allocation programs. More information about access is available through the OLCF web site at: http://www.olcf.ornl.gov/support/getting-started/.

The OLCF worked with Cray to design Titan to be an exceptionally well-balanced system for modeling and simulation at the highest end of high-performance computing. The Cray XK6 nodes and the AMD Opteron processors double both the memory bandwidth and memory capacity per node as compared to the Jaguar Cray XT5 system it replaced. The system will be linked to an external global parallel file system by twice the number of I/O nodes and will use InfiniBand (IB) cards that provide at least twice the bandwidth of the IB cards in Jaguar. The file storage system is being acquired independently from the Titan system and will have at least twice the bandwidth and capacity of Jaguar's file system. The key new component of Titan is that a majority of the Cray XK6 nodes include an NVIDIA GPU application accelerator. In the November 2011 Top500 list [2] of the world's most powerful computers, 39 of the 500 computers on the list used application accelerators, including three of the five fastest computers.

This paper is organized as follows: The authors describe the Titan system architecture and our reasons for some of the choices we made in the design. We describe the process of upgrading Jaguar to Titan. We discuss the programming strategy for effectively using the GPU accelerators and the programming environment we have assembled for our users. Finally, we present some early performance results of applications on the first phase of the upgrade.

II. TITAN SYSTEM DESCRIPTION

Titan is the system being delivered as part of the OLCF-3 project, the third major architecture delivered for our users since the DOE Leadership Computing Facility program was created in 2004. The OLCF-1 system was Phoenix, a Cray X1 and X1E vector system. The OLCF-2 system was Jaguar, a series of machines built on the Cray XT system architecture ranging from a 25 TFLOPS XT3 in 2005 to the 2.3 PFLOPS XT5 placed in service in 2009. The OLCF-3 project upgraded the Jaguar XT5 system to Cray's XK6 architecture using the latest Opteron processor, Gemini interconnect, and most notably different from prior systems, NVIDIA's GPU application accelerators.

The Cray XK6 is an evolution of the Red Storm system architecture [3] developed through collaboration between Sandia National Laboratories and Cray, and first delivered in 2005 in Cray's commercial XT3 product. Through a series of evolutionary steps, the XT3 produced the XT4, XT5, and in 2010 the last in the XT series, the XT6. Each generation of the XT series used the same Cray SeaStar interconnect [4], but incremented the model number when a new generation of AMD's Opteron processor socket was used in the product. In each of these models, Cray delivered at least two revisions of the node board with an upgraded version of the processor.

A. *Node Architecture*

The XK6 node consists of an AMD Opteron 6274 "Interlagos" 16-core processor linked to an NVIDIA X2090 "Fermi" GPU. The AMD Interlagos processor is the first to use the new "Bulldozer" [5] module, each of which contains two integer processor cores that share a floating-point unit capable of generating eight 64-bit results per clock cycle. With eight floating-point units per processor and eight operations per clock cycle, the 2.2 GHz processors in Jaguar each have a peak performance of 140.8 GFLOPS. Each processor has four separate memory channels connected to DDR3-1600 memory providing up to 102.4 GB/s of memory bandwidth. The Interlagos processor connects to the NVIDIA GPU through a Hypertransport-3 (HT-3) to PCI-express version 2 conversion chip. The bidirectional bandwidth between the Interlagos processor and the GPU is 10.4 GB/s.

The NVIDIA GPU is mounted on an SXM form factor PCI-express card with 6 GB of GDDR5 memory. The first phase of the upgrade to Jaguar includes 960 compute nodes configured with NVIDIA Fermi GPUs [6]. The SXM module is a different form factor but functionally equivalent to the NVIDIA M2090 [7]. The Fermi GPU has 16 streaming multiprocessors (SM) and has a peak performance of 665 GFLOPS. The Fermi implements memory error correcting codes that can correct single-bit memory errors and report double-bit errors (SECDED), but at a cost of storing those codes in the GDDR5 memory. The memory error correcting codes decrease the available memory per GPU to approximately 5.25 GB. The memory bandwidth between a Fermi GPU and the GDDR5 memory is 177 GB/s with error correction turned off, and approximately 140 GB/s with error correction enabled.

In late 2012, the NVIDIA "Kepler" GPU will become generally available and will replace the Fermi GPUs in Titan. It will use the same SXM form factor board with 6 GB of GDDR5 memory. The specific details of the HPC version of the Kepler GPU architecture have not been released; however, an architecture document [8] describing the GeForce 680 version of the Kepler GPU shows the single precision floating point performance as almost twice that of the Fermi processor and describes some of the features of the Kepler architecture.

B. *Gemini Interconnect*

One of the key differences between the Cray XK6 and prior generation XT systems is the Gemini interconnect [9]. Instead of a SeaStar ASIC for each node, each Gemini custom ASIC connects two nodes to the 3-D torus interconnect. All of the cables and backplane interconnections between node boards are the same for SeaStar and Gemini based system. The only difference is the mezzanine card on the node boards. The mezzanine card is a separate printed circuit board that attaches to the base XK6 node board and contains either the SeaStar or Gemini ASIC along with any support circuitry and the interconnections between the SeaStar or Gemini chips. This feature allowed ORNL to upgrade from an XT5/SeaStar system to an XK6/Gemini system while reusing the cabinets, cables, and backplanes. Figure 1 shows the functional layout of the Gemini ASIC

.

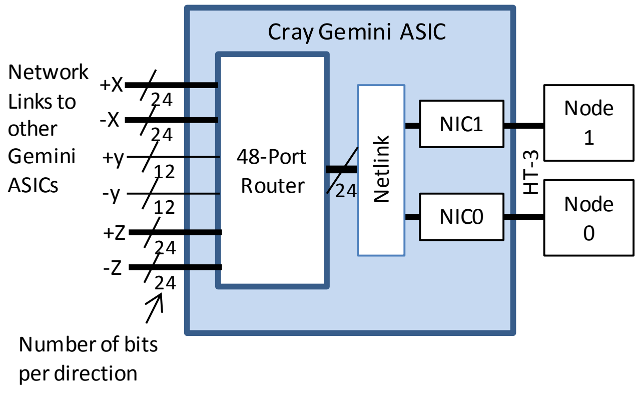

Figure 1 - Functional layout of the Gemini ASIC (Courtesy of Cray Inc.)

In describing a 3-D torus, it is convenient to describe the three directions as "X", "Y, and "Z", referring to the common directions in a coordinate system. Each Gemini ASIC therefore has six directional links with the positive and negative directions in the X, Y, and Z directions. The ±X and ±Z links have twice as many wires as the ±Y cables because the X and Z directions have two physical cables, replacing two separate positions in the SeaStar torus with one position in the Gemini torus and taking advantage of the extra cables to increase the bandwidth, as shown in Figure 2.

Some of the links between Gemini ASICs go over cables, and some of the links are implemented as traces on the Gemini mezzanine card or in the XK6 cabinet backplane. The length of the cables also varies depending on the direction in the torus. In Titan, the X dimension of the torus goes from cabinet to cabinet in the same row of cabinets on the computer room floor; therefore the number of Gemini ASCIs in the X dimension is equal to the number of cabinets in a row. The Y dimension cables go from row to row with the two Gemini ASICs on a mezzanine card also connected in the Y dimension. The result is that the number of Gemini ASICs in the Y dimension is equal to two times the number of rows. The Z dimension stays within an XK6 cabinet. There are 48 Gemini ASICs in a cabinet, two per XK6 blade.

The 24 Gemini 0 ASICs of each blade in a cabinet are wired together in the Z dimension as are the 24 Gemini 1 ASICs. The result is that Titan's 200 cabinets, each with 24 blades and 48 Gemini ASICs are wired into a three dimensional torus with 25 Gemini ASICs in the X dimension, 16 Gemini ASICs in the Y dimension and 24 Gemini ASICs in the Z dimension for a total of 9,600 Gemini ASICs, each of which is connected to two nodes.

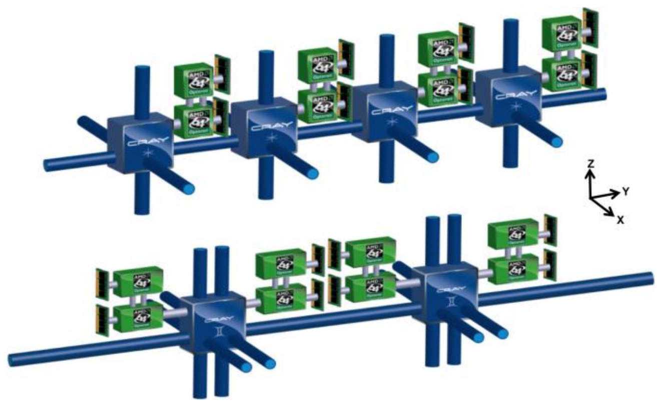

Figure 2 - Logical layout of interconnect ASICs in the SeaStar (top) and Gemini (bottom) networks on a node board (Courtesy of Cray Inc.)

The Gemini links have differing bandwidths depending on the direction and if the link goes over a cable, mezzanine card trace, or through the backplane that connects blades in a cabinet. Table 1 shows the bandwidths of each of the link types. Titan's 3-D torus dimension listed above is (25 x 16 x 24). The cross section bandwidth in the X dimension is 75Gbps * 16 Geminis in the Y * 24 Geminis in the Z * 2 directions per cable * 2 torus paths / 8 bits/byte = 14,400 GB/s. Using the same formula the cross section bandwidth in the Y dimension is 11,250 GB/s and in the Z dimension is 24,000 GB/s. The bisection bandwidth is the smallest of these three at 11,250 GB/s. Table 2 presents measured latency and bandwidth numbers from the XK6 as compared to the XT5 system.

Link Type	Bits per direction	Clock Rate	Bandwidth per Direction
X and Y cables	24	3.125 Gb/s	75 Gb/s
Y cables	12	3.125 Gb/s	37.5 Gb/s
Y mezzanine	12	6.25 Gb/s	75 Gb/s
Z backplane	24	5.0 Gb/s	120 Gb/s

Table 1 – Titan's torus link bandwidths across Gemini [10].

C. *System Description*

Titan is built from 200 Cray XK6 cabinets arranged as 8 rows of 25 cabinets per row. Each cabinet is powered by a 480 volt, three-phase circuit and when fully configured with maximum memory and the highest speed processors and accelerators, is rated to consume a maximum of 54 KVA per cabinet. When configured with the Kepler GPUs, Titan is expected to draw a peak of as much as 10 MVA of power and a more typical 7-8 MVA when running an average mix of applications.

The cabinets for Titan are cooled using Cray's ECOphlex™ technology in which liquid R-134a refrigerant is vaporized by the heat exiting the Cray XK6 cabinet. The vapor is then condensed with a chilled water heat exchanger to efficiently capture the heat from the system.

Titan's cabinets contain a mix of I/O and compute blade designs. There are a total of 4,800 blades, each containing four nodes. 4,672 of the blades are Cray XK6 compute blades. Each of the XK6 compute nodes has a single AMD Interlagos processor and four DDR3-1600 8 GB memory modules. In addition, at least 3,648 will also have NVIDIA Kepler GPUs. 1,024 XK6 node boards will initially be Opteron-only, with an open slot available for expansion.

The remaining 128 blades are Cray XIO boards which contain four service and I/O nodes used to boot the system, as login nodes for interactive use, for network nodes to connect Titan to the internet, and as Lustre routers to link Titan to the Spider external file system. The initial Spider configuration has a bandwidth of 240 GB/s and capacity of 10 petabytes. The OLCF expects to more than double both the bandwidth and capacity of the Spider file system in 2013.

Test	XK6 Performance	XT5 Performance
MPI Unidirectional Bandwidth	5.87 GB/s (x dim)	1.62 GB/s
	3.47 GB/s (y dim)	
	6.09 GB/s (z dim)	
MPI Bidirectional Bandwidth	10.6 GB/s (x dim)	2.93 GB/s
	5.40 GB/s (y dim)	
	10.5 GB/s (z dim)	
MPI ping-pong, zero byte latency, nearest neighbors	1.50 μsec (x dim)	6.20 μs
	1.70 μsec (y dim)	
	1.55 μsec (z dim)	
MPI ping-pong, zero byte latency, across full machine	3.96 μs	8.70 μs
MPI All Reduce	22.41 μs	147 μs
MPI Barrier	24.80 μs	112 μs

Table 2 – MPI performance benchmarks, comparing Jaguar's XK6 performance with XT5 performance measured at the October 2009 Istanbul upgrade.

III. UPGRADING JAGUAR TO TITAN

The upgrade of Jaguar to Titan is being accomplished in two phases, as mentioned earlier. The first phase replaced all of the XT5 compute blades with XK6 compute blades. The second phase will add the NVIDIA Kepler GPUs to the node boards once those processors are commercially available in late 2012. A complicating factor for the upgrade is that Jaguar is a production computer with users depending on it for their work. Taking Jaguar down for a long period of time for the upgrade was not a viable option for the OLCF. The XT and XK 3-D Torus networks and the external Lustre file system, known as Spider, that the OLCF developed for Jaguar provide the flexibility to upgrade Jaguar in phases, always keeping a 1+ PFLOPS portion of Jaguar available for the users, except during the final test of the combined system.

In the first phase of the upgrade, the 200 cabinet Jaguar system was divided into two sections, the first with 104 cabinets and the second with 96 cabinets.

The X-dimension torus cables were removed between the two sections of the system so that one section could be upgraded, while the other section remained in operation for the users. This was necessary because the SeaStar and Gemini networks are electrically incompatible, even though they use the same cables and connectors. The 256 Service and I/O (SIO) nodes in Jaguar are distributed throughout the cabinets of the system and link to the Spider file system's object storage servers through a switched DDR IB network. Thus, when the system was segmented, the bandwidth to the file system remained proportional to the number of nodes, and each of the SIO nodes could reach the entire file system through the IB network fabric. Finally, the OLCF has maintained a portion of the Spider file system for testing, and the portion of the system that was being upgraded used this part of the file system. The result was that the 96 cabinet section of Jaguar was initially upgraded to the XK6 blades and run as a separate system, while we worked through the acceptance tests on that portion of Jaguar. The remaining 104 cabinets were connected to the production Spider file system and continued in operation for our users.

Once the 96 cabinet upgrade and testing was complete, the users were moved to the newly upgraded section by remounting the production Spider file system on the upgraded Cray XK6 nodes. We then replaced the blades in the 104 cabinet side of the system and ran diagnostics on that section of the system. Now, with both segments running the Gemini interconnect, we could recombine the two halves of the system for final acceptance. This was the only time during the upgrade where users could not run on the system. We needed to verify that the Gemini network worked properly at this scale since no other system of this size had previously been built using Gemini technology. After a few days of dedicated system testing that verified the functionality and performance of the system, we began a stability test running a fixed set of applications chosen to represent a substantial portion of the workflow on Jaguar and Titan. After a few more days, the stability test continued with selected users allowed to run applications to harden the system. The upgrade started in early October 2011 and the final acceptance test was completed in February 2012

IV. Programming Strategy for Titan

The Center for Application Acceleration Readiness (CAAR) is a collection of application teams, vendor partners, and tool developers brought together by the OLCF to produce the successful port of six representative applications to Titan, ensuring this suite of codes is capable of day-one scientific output when Titan is put into production, as well as serving as a laboratory to establish a set of best practices for other application ports. Succinctly, the mission of CAAR is to explore sustainable methods to uncover and exploit hierarchical parallelism in modern HPC codes, thereby ensuring portable performance on current and near-future platforms. Five of the six CAAR applications are discussed in some detail in Section VI.

It has been a common practice in HPC to refactor codes as architectures changed from vector supercomputers to symmetric multiprocessing systems to massively parallel cache-based machines. During each of these epochs, considerable effort was expended to exploit that facet of the new architecture that held the most promise for performance and scalability: codes on vector machines were re-engineered to minimize scalar operations, arrays were massaged to fit in data caches, and loops were rewritten and otherwise transformed to minimize node-to-node communication on MPPs. All of these changes could disrupt (what were often originally quite intuitive) data structures and program flow. One of the goals of the CAAR is to closely examine codes that broadly represent the types of simulation activity at the center and to more thoughtfully explore how the programming model should be modified in response to not only the Titan system, but a wider set of current and future systems.

Many modern codes in use on Jaguar today rely wholly on domain decomposition via MPI for parallelization. The Titan hybrid multicore architecture will demand that this level of parallelism be augmented with SMP-like and vector-like parallelism. Importantly, the most profitable path forward for exploiting the hybrid multi-core nodes will likely not be simply relegating some portion of the parallel work that is currently de-composed via MPI to a thread-level decomposition on the accelerator. For example, an explicit grid code should not expect to see performance improvement by parallelizing the same operations over node-local patches of grid and "coarsening" the level of the MPI decomposition [11]. Rather, those operations that are performed serially in the MPI-only code will need to be parallelized via a threading mechanism or, perhaps, local MPI communicators on the node, and those operations within the implied loop nests will profit from vector-like parallelization via the GPU's. This discovery or, perhaps in some case, the rediscovery of hierarchical levels of parallelism in current codes will form the heart of a successful programming model on Titan.

The CAAR codes have provided ample opportunity for us to begin exploring exactly how we will expose this parallelism to engage the new hardware. It is important to note that current notional designs for exascale machines will also rely primarily on ubiquitous, but hierarchical, parallelism to achieve high performance. Concomitantly, the kind of code transformations being undertaken on CAAR codes today will be beneficial on a host of current and near-future platforms when they are completed, whether or not those platforms use GPUs. The general notions of increasing data locality and exposing the maximum amount of inherent parallelism in codes will have positive impacts on performance for these codes on Titan, Sequoia, Mira, Stampede, Blue Waters, and other leadership scale systems.

V. Programming Environment

A Programming Environment (PE) is defined as the collection of software that supports the application development cycle for one or more programming models. A typical PE consists of compilers, programming languages, libraries, debuggers, and performance tools unified within a common infrastructure.

During the design of Titan and the initial application work in CAAR, ORNL tools developers encountered a major challenge: there was no production-ready PE for rapidly porting codes to a hybrid GPU-based system that could meet petascale-level performance. As a result, the OLCF assessed the current state-of-the-art in order to select a suite of tools necessary to build an effective PE ecosystem that met the demanding needs of Titan's applications. In addition to commonly available commodity tools, our ecosystem consists of highly customized compilers, performance tools, debuggers, source code analyzers, and GPU libraries specifically built for Titan. Table 3 provides a list of tools in the Titan PE, an asterisk marks tools that were customized and enhanced specifically for Titan.

Compilers	Performance Tools	GPU Libraries	Debuggers	Source Code
Cray* PGI CAPS-HMPP* Pathscale NVIDIA GNU Intel	CrayPAT* Apprentice* Vampir* Vampir-Trace* TAU HPCToolkit CUDA Profiler	MAGMA* CULA* Trillinos*	DDT* NVIDIA gdb	HMPP Wizard*

Table 3 - Tools supported in Titan's Programming Environment

The CAAR suite of codes are a diverse bunch and the number and types of programming tools used to port them to the Titan architecture are at least as varied. Recent state-of-the-art approaches included hand-coded CUDA and the use of compiler directives. Although CUDA is most common and, at this time, most mature technology for development, several experiences in CAAR and our desire to produce software that exhibits portable performance for a range of platforms leads us to strongly advocate the use of compiler directives.

After giving careful consideration to the "three Ps", portability, productivity and performance, we decided that a hybrid programming model, one where programs can be written to support different levels of abstraction in terms of communication libraries (or PGAS languages), combined with shared memory directives (i.e. OpenMP) and/or accelerator programming APIs (OpenACC, HMPP) and languages (OpenCL, CUDA), would allow application developers to take maximum advantage of Titan with minimal porting effort. The goal of this model is to express parallelism in the codes at the most appropriate level and optimize the compilers, libraries, and other tools to generate high quality code for the underlying hardware. The most technically challenging aspect was to design a standard GPU accelerator directive API, as none existed when we began this effort. To accomplish this we worked closely with NVIDIA, the Cray compiler group, CAPS-Enterprise, and PGI. This effort led to the *de-facto* standardization of OpenACC, [12] a set of compiler-based directives targeted at NVIDIA GPUs. As a result, Cray, CAPS, and PGI support OpenACC with NVIDIA leading this effort and NVIDIA CEO Jen-Hsun Huang announcing OpenACC during his keynote address at SC'11.

OpenACC is a subset of the respective vendors' directives, providing a common interface. OpenACC should be fully implemented (as currently specified) before Titan reaches production. Experience with the transition from vector codes to message-passing codes twenty years ago proves the benefits of common, open standards for programming models. Much greater efficiency of effort was achieved following the wide adoption of the MPI standard. An open standard for the utilization of accelerators holds a similar promise.

For all these reasons, we feel that a directive-based approach to code refactoring is the most straightforward and profitable path for future development. Nevertheless, as CUDA is readily available and relatively mature (especially when compared to some of the directive-based approaches), it has been used heavily in the initial porting activities undertaken by many groups to this point. And this mature role serves an important purpose of helping the development of directives-based compiler technology by elucidating the capabilities of the accelerator hardware, i.e., giving the directives approach a target to meet and exceed. Interoperability of CUDA (including CUDA Fortran) and directives will, therefore, be an important consideration going forward. Congruence of memory models between CUDA, directives, and accelerated libraries is perhaps the most immediate and impactful place where interoperability must be preserved.

A handful of compilers currently implement directives for GPU programming:

- HMPP (CAPS Enterprise)
- pgf90 (The Portland Group)
- ftn (Cray)

Each of these compilers has been applied to one or more CAAR codes at some point during our OLCF-3 project. The directive-based approach provides:

- Incremental porting/development,
- Fast prototyping: quickly produce accelerated code segments,
- Increase in productivity: few code modifications to produce accelerated code,
- Transferability to different architectures (CPU, GPUs, FPGAs), and
- Opportunity for tools to assist the user in generating, and debugging directives, and help with performance analysis.

Of the three compiler vendors, we opted to work most closely with CAPS-Enterprise because their GPU directive solution, which uses a source-to-source strategy for the translation of GPU directives, was the most advanced in terms of features, functionality, performance and portability across platforms. CAPS also provided ORNL with a framework for ORNL's scientists to help them understand how the directives are translated to OpenCL, CUDA or other future accelerator standards. Enhancements to the CAPS-Enterprise GPU directives include: support for data distribution and unified virtual addressing among GPUs/CPU, pointers inside GPU kernels, inter-procedural data mirrors, C++, OpenMP and GPU library compatibility. Additionally, we worked with the vendor CULATOOLS and the University of Tennessee to improve the CULA and MAGMA GPU library support for Fortran. These enhancements stemmed from application studies and their needs.

Another challenge for Titan's PE was to provide a scalable, hybrid aware debugger. Our goal was to provide a debugger capable of handling petascale-runs, while summarizing (reducing) and presenting meaningful information to the user. By employing sophisticated tree topologies, Allinea, working along with ORNL, deployed the first petascale-level debugger, DDT, for Jaguar. Field-tested on development codes at ORNL, DDT has been shown to scale up to over 200,000 cores. Allinea has applied a co-design methodology, working closely with compiler vendors that support the languages for Titan, to include GPU directives and languages. As a result, DDT was designed from the ground up for large-scale hybrid systems. In addition to MPI/OpenMP parallel programs, DDT is fully supported on NVIDA GPUs. A scalable mechanism was designed to debug and visualize stacks, and to merge variable stacks with the same values. In addition, DDT is now capable of stepping into GPU directives regions. The debugger was extended with CUDA memory debugging capabilities, allowing the user to visualize the physical memory layout of their codes and its distribution over both the host and accelerator memory.

The main performance analysis tools for hybrid parallel applications are the Vampir toolset and CrayPat. The Vampir toolset consists of three major components: the Open Trace Format library (OTF), VampirTrace, and Vampir. VampirTrace is used in the pre-run phase to prepare the application source code to gather events during a run. At runtime, VampirTrace processes the events and passes them to the OTF library.

The visualization and analysis component of the Vampir toolset are the Vampir client and server respectively. The purpose of the server is to analyze the trace in parallel and aggregate enough main memory on the compute nodes to open/load the trace. The client visualizes the data analyzed and transferred by the server, and offers a variety of methods to interact with the trace, e.g. scrolling, zooming, and highlighting areas of interest.

We selected the Vampir toolset for Titan's PE, because its framework was designed to be highly scalable. To enhance the Vampir toolset for Titan, we identified four major areas of improvement: support for GPU performance tracing, improved I/O, tracing scalability, and the user interface presentation of traces of large-scale runs. One of the more important enhancements has been improving MPI behavior of VampirServer, which now enables a user to effectively analyze a trace of an application which utilizes the entire Titan system. Furthermore, a new display was implemented to directly compare two or more traces. This display enables the time-wise alignment of the traces, and it enables the visualization of common Vampir displays next to each other.

The other performance tools that we support in Titan are CrayPAT and Cray Apprentice. CrayPat is a high-level performance analysis tool for identifying opportunities for optimization by providing a set of experiments and reports that measure how fast an application is running and where the performance bottlenecks are found. CrayPat is a scalable and low-overhead performance tool because it relies on sampling-based technology and/or binary instrumentation. Cray Apprentice provides a user interface to visualize the application's performance results. For Titan, CrayPAT and Apprentice were enhanced to support GPUs performance analysis.

Application	(a) XK6 vs. XK6 (no GPU)	(b) XK6 vs. XE6
S3D	1.5	1.4
DENOVO	3.5	3.3
LAMMPS	6.5	3.2
WL-LSMS	3.1	1.6
CAM-SE	2.6	1.5

Table 4 – Early-status XK6 performance benchmark ratios showing the relative effectiveness of GPU acceleration on five application codes from the CAAR suite: (a) Accelerated XK6 performance relative to XK6 without acceleration, (b) Accelerated XK6 performance relative to XE6 (dual Interlagos) performance. In each case, the new, refactored code is used in measuring performance, that is, we use the best code for each hardware option. XK6 performance results achieved on OLCF's development partition Titandev. XE6 performance results achieved on CSCS's Cray XE6, Monte Rosa [13].

VI. RESULTS TO DATE

The six primary applications selected for inclusion in the CAAR effort are:

- S3D – Direct numerical simulation of compressible, reacting flows for combustion science [14],
- DENOVO – A high-fidelity neutron transport code for nuclear reactor design [15,16] ,
- LAMMPS – A molecular dynamics general statistical mechanics based code applicable to bioenergy problems [17],
- WL-LSMS – A first principles density functional theory code (local density approximation) used to study magnetic materials [18],
- CAM-SE – A highly scalable atmospheric model that is part of the Community Climate Systems Model [19], and

- NRDF – A non-equilibrium radiation diffusion code designed to serve as a test bed for accelerated approaches to adaptive-mesh refinement (AMR) [20].

In the following, we present, for five of the six CAAR applications, brief introductions to the application, the strategy used for porting to the Cray XK6, early performance results on Jaguar's XK6 development partition (i.e., "Titandev"). We also highlight targeted stretch research goals for these applications on Titan over the near term.

A. *S3D*

Accelerated computing will allow combustion simulations to be performed with higher chemical complexity, or a greater range of length scales, (implying higher Reynolds number and greater turbulence.) On a petaflops-scale system, a lifted ethylene jet flame simulation at *Re* = 10,000 is feasible. While this was a significant achievement for a turbulent flame simulation, the Reynolds number is not sufficiently high to rule out artifacts of the transition between a laminar and turbulent flow. An accelerated system will allow a 50–100% increase in Reynolds number (*Re* = 15,000–20,000) and enable the simulation of homogeneous, fully developed turbulence beyond the theoretically predicted transition. Such a simulation will enable the exploration of turbulent combustion attributes that exhibit Reynolds number independence, and hence relevant for engines where the Reynolds number is higher by orders of magnitude. A higher chemical complexity will enable the simulation of renewable and carbon-neutron automobile fuels such as oxygenated bio-fuels (e.g., dimethyl ether, iso-butanol). The chemical reaction model for such fuels would require between 60 and 100 chemical species compared to the 22 species ethylene chemistry that is feasible on non-accelerated petascale systems.

S3D is a compressible Navier-Stokes flow solver with reacting flows for the full mass, momentum, energy and species conservation equations. The code uses detailed chemistry and molecular transport models to perform direct numerical simulation (DNS) of combustion. In S3D, the conservation equations are integrated in time using a low-storage explicit fourth order Runge-Kutta integrator with error estimator. Discretization of the equations is achieved using a structured Cartesian mesh and 8th-order finite-difference methods to evaluate the spatial derivatives. The three-dimensional spatial domain is decomposed in a regular structured grid topology. The halo around the subdomains necessary for spatial differentiation is obtained through non-blocking MPI communication. S3D is written in Fortran, and has been exhaustively tested with the PGI compiler. MPI, ACML, and I/O libraries are used. S3D consumed the second-largest compute resource on Jaguar over the period 2010-2011, consuming 6% of the total workload.

The S3D benchmark problem for development is a 3-dimensional DNS of Homogeneous Charge Compression Ignition (HCCI) combustion in high-pressure stratified turbulent 52 species n-heptane/air reaction mechanism, 1200^3 grid points, and 48^3 grid points per node. HCCI Engines have the potential for high efficiencies (diesel-like) but with low soot and NO_X emissions. On the Jaguar XT5, this problem would require approximately 225M core-hours. Larger chemical mechanisms are being developed for more complex fuels (e.g., 73 species bio-diesel or 99 species iso-octane mechanisms) for stretch science goals.

S3D has been transformed into an Hybrid MPI/OpenMP/OpenACC code. This refactoring consists of a significant rewrite to identify high-level parallel structures and the communication was redesigned to achieve overlap of computation and communication. The application is so structured that simple instructions at compile time can result in an executable for a multi-core system using OpenMP for parallelization on the node or for an accelerated node using OpenACC to direct the compiler to generate code for the accelerator. In the later case, all intensive computation is performed on the accelerator.

The outcomes of S3D refactoring have been positive: initial results showed it is faster on Cray XT5 by 2x. The GPU acceleration is achieved with minimal overhead using OpenACC directives. At this early stage, XK6 performance (Opteron + GPU) exceeds XE6 performance (dual Opteron) by a factor of 1.4, and accelerated XK6 exceeds the performance of non-accelerated XK6 by a factor of 1.5. (See Table 4.)

B. *DENOVO*

High-fidelity neutron transport is a necessary component of nuclear reactor design. Current simulation technology decomposes the problem into three steps that are connected in a semi-empirical manner. These steps require extensive tuning from experimental data to extrapolate correctly between them. In order to move to *ab initio* modeling and simulation, high-fidelity 3-D models and codes are required. Commercial power reactors consist of thousands of fuel pins assembled into hundreds of lattice bundles. For boiling water reactors, each bundle requires approximately 5-10 M CPU-hours to simulate on the Jaguar XT5. Thus, the XT5 was not capable of performing full core, 3-D *ab initio* simulations. Truly predictive reactor analysis modeling and simulation will require computing resources beyond petascale.

Denovo solves the first-order form of the linear Boltzmann radiation transport equation using the discrete ordinates (S_N) method and is an important component in the DOE Innovation Hub entitled Consortium for Advanced Simulation of Light Water Reactors (CASL) lead by ORNL. Denovo uses uniform and non-uniform Cartesian grids. The fundamental solution methods are:

- Arnoldi, power, and Rayleigh-Quotient iteration for eigenvalue solution;
- Multigrid-preconditioned GMRES or Transport two-grid preconditioned, Gauss-Seidel over energy;
- DSA-preconditioned GMRES, or BICGSTAB for within-group transport equations;
- PCG and Super-LU solvers used for diffusion solve in DSA pre-conditioner; WLA algorithm used to provide discontinuous updates for Finite Element spatial schemes.

For reactor calculations, the state (angular-flux moments) will consist of 500-1,500M cells, 20–250 groups, and 4–36 moments, yielding a solution vector consisting of O(10^{12}) unknowns. Denovo is written in C++ with F95 kernels, and builds on Jaguar with the PGI and GNU compilers. It uses the following external libraries, some of which are optional: MPI, BLAS/LAPACK, GSL, Trilinos, HDF5, Silo, Parallel SuperLU, BRL-CAD, SPRNG.

The acceleration strategy is to port the energy-angle sweep to the accelerator. This routine consumes 80% to 99% of the runtime. The acceleration of this component has focused on exposing more thread-level parallelism and locality of memory reference for multiple problem dimensions. GMRES, the second highest consumer of runtime, can run on the accelerator via Trilinos, but only for modest-size problems due to large memory requirements. To support acceleration of the sweep algorithm, the code has implemented an alternative parallelism approach for energy groups. The Koch-Baker-Alcouffe (KBA) wavefront decomposition sweep pipelining is limited by the number of angles per octant. We have used the multi-level energy decomposition to provide more opportunities for parallelizing the sweep. In this multi-level scheme, the number of pipelines is the product of groups and angles per octant, and will provide the GPU with large sections of work each iteration step, allowing scaling to over 100K cores with opportunities for enhanced parallelism on accelerators. This redesigned sweep kernel was rewritten in CUDA/C++ and now runs on either CPU or GPU.

The outcomes of DENOVO sweep redesign have been positive: initial results showed CPU-only code is faster on Cray XT5 by 2x. At this early stage, XK6 performance (Opteron + GPU) exceeds XE6 performance (dual Opteron) by a factor of 3.3x. (See Table 4.)

As a development problem, we target neutron transport simulations on Titan using 256 million spatial grid cells, 256 energy groups to resolve the neutron spectra, and 256 discrete ordinates to resolve the angular dependence of the radiation field. This will produce a single state vector of approximately 33TB. To be useful, the steady-state solution for this system will have to be computed in less than 12 wall-clock hours. For a stretch science goal on a 20 PFLOPS to 30 PFLOPS Titan, we have targeted 3-D, full reactor core simulations of fully self-consistent neutron transport on real engineering reactor models that could be used to address the challenge problems identified for the CASL project with an acceptable runtime within one week.

C. LAMMPS

Molecular Dynamics (MD) is a general statistical mechanics based approach that lets scientists peer into the motion of individual atoms in a way that is not possible in laboratory experiments. The various force fields available allow MD to be used to study a wide variety of chemical, physical, materials and biological systems, and the force field computations are generally very amenable for acceleration. The force fields typically involve bonded, non-bonded van der Waals, and long-range electrostatic interactions between atoms. The majority of the simulation time is in the short-range force-field calculation and many have a very similar functional form. The nature of these kernels allows for a straightforward approach to buffering input and output functions overlapped with computation on the accelerator. For MD simulations that require the computation of long-range forces, scaling is typically limited by particle-mesh approximations to the Ewald summation. These scale as $M\log(M)$, where the number of mesh points, M, is typically on the order of the number of atoms in the simulation. This algorithm uses a full 3-D FFT and thus, involves all-to-all communication patterns.

LAMMPS is a molecular dynamics code that is used to computationally model condensed-matter and biological systems [17]. LAMMPS solves the classical N-body problem of atomistic modeling, i.e., numerically solves Newton's Second Law. The Velocity Verlet algorithm is the most commonly employed time integrator. The non-bonded van der Waals interactions are evaluated within a spherical cutoff to reduce the computational work. A neighbor list is used to track the pairs of interacting atoms over successive time steps. Long-range electrostatics are evaluated using a Particle-Particle Particle-Mesh (PPPM) solver. In this effort, we are exploring alternative long-range algorithms better suited for massively parallel simulations on next generation architectures. This includes approaches that replace the 3-D FFT's with grid-based algorithms that reduce inter-process communications.

Given this grid-based approach, the parallelization strategy is spatial domain decomposition. The elements to be accelerated include the short-range force calculations, neighbor list builds, and computation of the long-range electrostatics. Readying LAMMPS for acceleration at large scales required modifying the code to expose data parallelism to the GPU and to exploit concurrent CPU/GPU calculations and data transfer where possible. The OLCF-3 accelerated code builds with OpenCL or CUDA. The OLCF-3 development efforts for hybrid multicore systems now allows for (a) acceleration for building neighbor lists, (b) acceleration of over 21 force-field models, (c) acceleration of PPPM long-range electrostatics, (d) concurrent CPU/GPU utilization, and (e) portability to many accelerators and CPUs.

As a development problem for present purposes, we target a liquid crystal benchmark, the Gay-Berne potential [17] for biaxial ellipsoid particles, often used to represent liquid-crystal mesogens and for aspherical coarse-grain models. An attractive feature of this benchmark problem is its high-arithmetic intensity: greater than 99% of the floating-point operations are available for acceleration. At this early stage, XK6 performance (Opteron + GPU) exceeds XE6 performance (dual Opteron) by a factor of 3.2, and accelerated XK6 exceeds the performance of non-accelerated XK6 by a factor of 6.5. (See Table 4.)

The stretch science target for the OLCF-3 effort with LAMMPS is to understand membrane fusion mechanisms, biological systems of the order of 10^6–10^9 coarse grain particles, using computational domains large enough to remove finite-size effects. Membrane fusion is the merging of two biological membranes in a controlled manner, and is an integral part of the normal life cycle of all living organisms. The requirement for proper solvation, the need to eliminate artifacts from interactions between periodic images, and the ability to capture long-wavelength modes all lead to a minimum requirement of 850,000,000 coarse-grained particles for these simulations, roughly 1000× in particle number and spatial extent compared to what could be performed on XT5 instantiation of Jaguar. The goal is to perform this simulation in fewer than 5 wall-clock days using the entire Titan system.

D. *WL-LSMS*

Nanostructured magnetic materials are well recognized as providing important opportunities for the control of the magnetic state and its response to applied fields with positive impacts on technology. At the center of these opportunities is the possibility to manipulate the magnetic state at the level of atoms and electrons, and thereby gain control over fundamental magnetic interactions. However, these systems also pose fundamental challenges due to the inhomogeneity and defects intrinsic to their reduced dimensionality, making the construction of simplified models, e.g., extended Heisenberg models, and the extraction of parameters from experiment or from theory of bulk systems difficult or impossible. These difficulties are further compounded by the fact that it is the finite-temperature and field-dependent response that are most important. At the most fundamental level, these difficulties and challenges require approaches that enable us to predict from first principles the finite-temperature free energy, thermodynamic fluctuations, and magnetization dynamics, fully accounting for the complexities of the underlying quantum-mechanical interactions of the electrons from which finite-temperature magnetic states then emerge. The density-functional-theory based Wang-Landau (FP-WL) method we are developing is just such an approach.

The WL-LSMS application combines classical statistical mechanics for the atomic moment directions with a constrained, first-principles calculation of the associated energies. For the statistical mechanics part, the Wang-Landau method is used to calculate the thermodynamic density of states. The first-principles problem is solved using local spin-density-functional theory formulated in a real space multiple scattering formalism (LSMS). The main computational effort (>80%) in the code involves dense linear algebra for complex numbers (matrix inversion). Additionally, when the full interaction is considered (i.e., the “full-potential method”, systems of coupled ordinary differential equations on a 1-D grid become important. WL-LSMS is primarily a Fortran code (mostly F77 some F90), with C++ for the Wang-Landau driver. The code compiles with PGI. Libraries used include BLAS, LAPACK, HDF5, and BOOST. On the XT5 system, parallelization was achieved over two levels: (1) over Wang-Landau Monte-Carlo walkers, and (2) over each atom in an individual LSMS energy calculation by associating each atom with a separate MPI process.

The main computational effort in LSMS is concentrated in a single subroutine that inverts a single block of a double complex dense matrix. This kernel makes extensive use of a few BLAS Level 3 and LAPACK routines. Therefore, the strategy for acceleration is to leverage developments in accelerated linear algebra libraries, e.g., cuBLAS, CULA, or libSci_acc. Other important elements, necessary to enhance the parallelism expressed on the accelerator, are to (1) utilize OpenMP in the CPU sections of the code, and (2) restructure the code by moving communications outside of the energy loop. The outcomes of our WL-LSMS refactoring have been positive. At this early stage, XK6 performance (Opteron + GPU) exceeds XE6 performance (dual Opteron) by a factor of 1.6, and accelerated XK6 exceeds the performance of non-accelerated XK6 by a factor of 3.1. (See Table 4.)

The scientific target for the Titan simulations is to calculate both the magnetization and the free energy for magnetic materials. 5 million CPU-hours are required on the Cray XT5 today to calculate only the free energy for a single 250-atom iron supercell. Determining the magnetization for this supercell in addition to the free energy would require more than 50 million CPU-hours on the XT5, making the calculation intractable. The aim of the WL-LSMS Titan port is to reduce this cost back to the present cost of a one-dimensional density of states calculation, i.e., 5M core-hours.

E. *CAM-SE*

Coupled models of the physical climate system have been the traditional projection tools used in climate research studies. These models have historically been used to predict the consequences of climate change given assumptions about the carbon cycle and emissions of greenhouse gases, sulfate aerosols, black carbon, and other atmospheric constituents. These models need to be further developed and evaluated to provide more robust and reliable, multi-decadal, probabilistic predictions at the regional scale of climate variability and change. Examples include extensions to the physical climate system that provide an accurate description of evolving chemical and biogeochemical processes that affect atmospheric composition. Improvements to existing modeling frameworks will be essential for estimating impacts on natural systems and important components of the socioeconomic infrastructure related to energy, water, and carbon. The atmospheric component will require unprecedented large horizontal and vertical resolution, and the dynamical core will require numerical approximation techniques that can exploit both fine and coarse-grained parallelism.

HOMME (High-Order Method Modeling Environment) is a promising framework for solving the atmospheric primitive equations in spherical geometry. It has already been integrated into the CESM (Community Earth Systems Model), a comprehensive Earth system model used nationwide by climate researchers. CAM (Community Atmosphere Model) is the atmospheric component of CESM, and includes sub-components that predict the time evolution of the large-scale flow (the dynamical core), and physical parameterizations that predict the effects of sub-grid scale phenomena. HOMME is one of the supported dynamical cores in CAM, and the combined model is called

CAM-SE (Community Atmosphere Model – Spectral Element).

HOMME is one of the most scalable dynamical cores currently under development. It uses an improved formulation of the spectral-element method that locally conserves both mass and energy and maintains numerical stability with an isotropic hyper-viscosity term (i.e., a fourth-order derivative term). Many previous generation dynamical cores needed to apply special techniques to solve the so-called "pole problem," (specific to latitude-longitude grids) where converging meridians near the north and south poles otherwise cause numerical instabilities. To eliminate the pole problem, HOMME uses an equal-angle, cubed-sphere grid to discretize the horizontal dimension, as the element areas are very nearly equal across the entire gird. This is a more scalable approach than traditional filter-based methods, because non-scalable polar filters are no longer needed. In the vertical dimension, CAM-SE uses a pressure-based, terrain-following coordinate system. CAM-SE allows for full two-dimensional domain decomposition and is the first version of CAM that exactly conserves dry mass without the use of mass fixers and conserves energy.

The baseline parallelism in HOMME, prior to refactoring, is across spectral elements using MPI. Maximum parallelism is achieved with one MPI task per element. A data transpose is performed every model time step to transfer data from HOMME data structures to physical parameterization data structures. In the physics packages, vertical columns of data are treated independently without communication in the horizontal directions. CAM-SE runs on a variety of architectures and compiler environments. Examples include PGI and Pathscale compilers on the ORNL XT5 system, xlf90 on the IBM Power and Blue Gene systems, and the Lahey compiler on generic Linux clusters.

The acceleration strategy involves first identifying computationally expensive parts of HOMME. The performance profile was relatively flat, and none of the computationally expensive parts of the calculations in the code were in libraries. However, with the addition of Mozart chemistry, the tracer advection now spikes in its contribution to run time due to 106 tracer species. Also, the addition of Mozart chemistry is useful to help quantify aerosol effects on uncertainties concerning climate sensitivities and to project changes in air quality with climate change. Hybrid OpenMP/MPI parallelism is implemented in all of CAM-SE, i.e., in the physical parameterizations and in the HOMME dynamical core. Porting the physics packages is largely not an option because they usually exhibit strong vertical dependencies that are difficult to thread over vertical indices. In order to keep CUDA FORTRAN kernels separate from the rest of the code, they are grouped together into a module. When acceleration is turned on at compile time, the GPU kernels are called in place of the original routines. The current port of these key kernels in CAM-SE has used CUDA FORTRAN. Over the long term, increased use of compiler directives, e.g., OpenACC, will be desirable to enhance portability and sustainability of the code.

Refactoring CAM-SE has lead to code that is more efficient and more scalable on homogeneous multi-core (CPU) architectures as a result replacing the vertical remap algorithm entirely and optimization of others for memory utilization. Initial results showed CPU-only code is faster on Cray XT5 by 1.7x, and the GPU-accelerated code is 3.7x faster than the original baseline overall. Using the new, re-factored code, "apples-to-apples" comparison of XK6 performance (Opteron + GPU) exceeds XE6 performance (dual Opteron) by a factor of 1.5, and accelerated XK6 exceeds the performance of non-accelerated XK6 by a factor of 2.6. (See Table 4.)

Incorporation of the HOMME dynamical core into a CESM simulation including Mozart tropospheric chemistry with 106 constituents at high spatial resolution (14 km horizontal grid) is the scientific simulation goal for the Titan port. To meet accepted throughput goals, a 100-year climate simulation with this collection of parameters must achieve 3.3 simulated years per wall-clock day.

VII. CONCLUSION

2013 promises to be a landmark year for OLCF, as we anticipate making Titan, a new scalable, GPU/CPU hybrid multicore supercomputer available to our user community. This paper describes the Titan system architecture, the process of upgrading Jaguar, our recommended GPU/CPU hybrid programming strategy, and early performance results on the first phase of the upgraded Jaguar for a few, selected applications.

The bulk of users running on Titan will access this resource through the Innovative and Novel Computational Impact on Theory and Experiment (INCITE) program [21], a merit-based, peer-reviewed allocation program, which is jointly managed by OLCF and the Argonne Leadership Computing Facility. Titan will make an unprecedented contribution to the INCITE program in 2013, making available more than 20 PFLOPS and approximately 2 billion core-hours (i.e., 67 million node-hours) of computing capability.

Recognizing this sharp and large increase in capability, INCITE issued a Request for Information (RFI) to potential research teams to gain an early look at the demand for these new leadership-computing resources in advance of the Call for Proposals closing on 27 June 2012. The RFI responses reflect the diverse array of agencies that look to INCITE to achieve advances across the scientific spectrum: responses represent potential users from the Department of Energy (DOE) laboratories; academia; international respondents; and U.S. industry. One finding of the RFIs: the availability of INCITE's new capabilities, such as accelerated, hybrid computing in a high-performance, scalable network, is drawing new, high-impact communities, such as Genetics and Drug Discovery, to leadership computing, which have not historically participated in computing at the leadership scale.

With the advent of Titan in 2013, and a burgeoning interest in new scientific disciplines, the leadership-computing program promises to make a big impact in 2013. Whether that impact is to clean energy, scientific discovery, or industrial innovation, nearly every sector of computational research stands to benefit.

ACKNOWLEDGMENT

We are very happy to acknowledge the excellent work of the OLCF-3 acceptance team and each of the six CAAR teams without whom the results presented here would not be possible. We would like to thank the leadership and staff of the Swiss National Supercomputing Centre, CSCS for their assistance in accessing their Cray XE6 platform, Monte Rosa. We also thank Julia White, INCITE program manager, for her thoughtful review and editing of the paper. This research was funded by the U.S. Department of Energy and used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

REFERENCES

[1] Bland, A.S.; Kendall, R.A.; Kothe D.B.; Rogers, J.H.; Shipman, G.M.;,"Jaguar: The world's most powerful computer," Cray Users Group (CUG) 2009 Proceedings.

[2] Dongarra, J.; Mueur, H.; Simon, H.; Strohmaier, E.;, "Top500"; http://top500.org/lists/2011/11.

[3] Camp, W.J.; Tomkins, J.L.; "Thor's hammer: The first version of the Red Storm MPP architecture," Proceedings of the SC 2002 Conference on High Performance Networking and Computing, Baltimor, MD, November 2002.

[4] Brightwell, R.; Predretti, K.T.; Underwood, K.D.; Hudson, T.; "SeaStart Interconnect: balanced bandwidth for scalable performance," IEEE Micro, vol. 26, issue 3, pp. 41-57, May-June 2006.

[5] Butler, M.; Barnes, L.; Sarma, D. D.; Gelinas, B.;, "Bulldozer: An approach to multitreaded compute performance," IEEE Micro, vol. 31, issue 2, pp. 6-15, March-April 2011 doi: 10.1109/MM.2011.23.

[6] Nickolls, J.; Dally, W.; "The GPU computing era," IEEE Micro, vol. 30, issue 2, pp. 56-69, March-April 2010 doi: 10.1109/MM.2010.41.

[7] NVIDIA M2090: http://www.nvidia.com/docs/IO/43395/Tesla-M2090-Board-Specification.pdf.

[8] NVIDIA GeForce GTX 680 whitepaper: http://www.geforce.com/Active/en_US/en_US/pdf/GeForce-GTX-680-Whitepaper-FINAL.pdf.

[9] Alverson, R.; Roweth, D.; Kaplan, L.; , "The Gemini system interconnect," High Performance Interconnects (HOTI), 2010 IEEE 18th Annual Symposium on , vol., no., pp.83-87, 18-20 Aug. 2010 doi: 10.1109/HOTI.2010.23.

[10] [Cray, Inc.; "Hardware overview for Cray XE6 and Cray XK6 series systems"; H40-6081-D; pp. 90.

[11] Bronevetsky, G., Gyllenhaal, J., & de Supinski, B. 2008, in Lecture Notes in Computer Science, Vol. 5004, OpenMP in a new era of parallelism, ed. R. Eigenmann & B. de Supinski (Springer Berlin / Heidelberg), pp. 13–25.

[12] http://www.openacc-standard.org/

[13] Monte Rosa: http://user.cscs.ch/hardware/rosa_cray_xe6/index.html.

[14] Chen, J. H., Choudhary, A., de Supinski, B., DeVries, M., Hawkes, E. R., Klasky, S., Liao, W. K., Ma, K. L., Mellor-Crummey, J., Podhorski, N., Sankaran, R., Shende, S. & Yoo, C. S. 2009 Terascale direct numerical simulations of turbulent combustion using S3D. Computational Science and Discovery 2, 1-31.

[15] Evans, T. M., Stafford, A. S., Slaybaugh, R. N., & Clarno, K. T. (2010). Denovo: A New Three-Dimensional Parallel Discrete Ordinates Code in SCALE. Nuclear Technology, 171, 171–200.

[16] Wayne Joubert, "Developing a 3-D Sweep Radiation Transport Code for Large-scale GPU Systems," presented at SIAM Conference on Computational Science and Engineering, Reno, NV, Feb 28 – Mar 4 2011.

[17] Brown, W.M., Wang, P. Plimpton, S.J., Tharrington, A.N. Implementing Molecular Dynamics on Hybrid High Performance Computers - Short Range Forces. Computer Physics Communications. 2011. 182: p. 898-911.

[18] Eisenbach, M., "Future Proofing WL-LSMS: Preparing for First Principles Thermodynamics Calculations on Accelerator and Multicore Architectures", CUG 2011 Proceedings (2011) and references therein.

[19] Dennis, J.M., Edwards, J., Evans, K.J., Guba, O., Lauritzen, P.H., Mirin, A.A., St-Cyr, A., Taylor, M.A., and Worley, P.A., "CAM-SE: A scalable spectral element dynamical core for the Community Atmosphere Model", Int. J. High Perform. Comput. Appl. 74—89 (2012).

[20] Pernice. M., and Philip, B., "Solution of equilibrium radiation diffusion problems using implicit adaptive mesh refinement", SIAM J. Sci. Comp., 27 (2006).

[21] The INCITE program is open to U.S.- and non-U.S.-based researchers and research organizations needing large allocations of computer time, supporting resources, and data storage to pursue transformational advances in science, engineering, and computer science. Applications undergo a two-phase review process to identify projects with the greatest potential for impact and a demonstrable need for leadership-class systems to deliver solutions to grand challenges. To submit an application, please visit http://hpc.science.doe.gov for details about the proposal requirements.