Editors: Volodymyr Kindratenko, kindr@ncsa.uiuc.edu
Pedro Trancoso, pedro@cs.ucy.ac.cy

KEENELAND: BRINGING HETEROGENEOUS GPU COMPUTING TO THE COMPUTATIONAL SCIENCE COMMUNITY

By Jeffrey S. Vetter, Richard Glassbrook, Jack Dongarra, Karsten Schwan, Bruce Loftis, Stephen McNally, Jeremy Meredith, James Rogers, Philip Roth, Kyle Spafford, and Sudhakar Yalamanchili

The Keeneland project's goal is to develop and deploy an innovative, GPU-based high-performance computing system for the NSF computational science community.

The Keeneland project—named for a historic thoroughbred horse racing track in Lexington, Kentucky—is a five-year Track 2D grant awarded by the US National Science Foundation (NSF) in August 2009 for the development and deployment of an innovative high-performance computing system. The Keeneland project is led by the Georgia Institute of Technology (Georgia Tech) in collaboration with the University of Tennessee at Knoxville and Oak Ridge National Laboratory. The initial delivery system, an HP Linux cluster with 360 Nvidia Fermi graphics processors, is now operational and is being used to prepare software tools and applications for the full-scale system, which is due to be deployed in 2012.

Keeneland is organized into two primary deployment phases. The first phase provides a moderately sized, initial delivery system (KID) to develop software tools for GPU computing and to prepare applications to exploit GPUs effectively. During 2012, Keeneland's second phase will provide a full-scale system for production use by computational scientists. The Keeneland Full Scale (KFS) system will be similar to the KID system in terms of hardware and software—that is, it will be a Linux cluster based on commodity next-generation CPUs, next-generation GPUs, HPC interconnect, programming environment, and tools. The KFS system is expected to have at least two times KID's total number of GPUs, with the expected performance improvements of the next-generation CPUs and GPUs. KFS will be an Extreme Digital Science and Engineering Discovery Environment (XSEDE) resource available to a broad set of users. Although there's currently a community movement in HPC toward this type of architecture, a critical component of the Keeneland project is to develop software that will let users take advantage of its unique capabilities. We also aim to reach out to teams developing applications that might map well to this innovative architecture.

Motivation

Heterogeneous architectures have recently emerged in response to the limits on improving single-node performance in traditional architectures; the primary factor influencing these limits is energy efficiency. In the case of GPUs,[1,2] these heterogeneous architectures were initially designed to support a fixed pipeline of graphics operations (such as rasterization), and consequently, architects could specialize them to be exceedingly efficient at those operations. However, it was only after a small set of early adopters began using GPUs for general-purpose computation more than a decade ago[3,4] that hardware features, algorithmic techniques, and programming systems—such as Cg,[5] CUDA,[6] and OpenCL[7,8]—started to emerge to make GPUs available to a wider audience.

Most recently, GPUs such as Nvidia's Fermi[1] have added critical features, including much-improved performance on IEEE double-precision arithmetic and memory error detection and correction, that make these architectures even more relevant to large-scale computational science. Compared to the alternatives, these new features, when combined with the original capabilities of GPUs, provide a competitive platform for numerous types of computing, such as media processing, gaming, and scientific computing, in terms of raw performance (665 Gflops/s per Fermi) and energy efficiency.

Not surprisingly, these trends have garnered the attention of researchers, vendors, and HPC customers. Beyond the Keeneland project, a substantial number of very large GPU-based systems have already been deployed.

 Copublished by the IEEE CS and the AIP 1521-9615/11/$26.00 © 2011 IEEE

Examples include China's Tianhe-1A, Nebulae at the National Supercomputing Centre (NSCS) in Shenzhen, Tokyo Tech's Tsubame 2, and several other systems in the US including Dirac at Lawrence Berkeley National Laboratory, Lincoln at the National Center for Supercomputing Applications (NCSA), and Edge at the Lawrence Livermore National Laboratory. Notably, the Chinese Tianhe-1A system at the NSCS in Tianjin achieves a performance of 2.57 petaflops/s on the TOP500 Linpack benchmark (www.top500.org), which was #1 in the world in November 2010.

All of these examples are scalable heterogeneous architectures that leverage commodity components: multinode computing systems with a high-performance interconnection network, where each node contains more than one type of processing device, such as a traditional CPU and a graphics processor. Most experts expect this trend to continue into the foreseeable future, given the requirements and constraints of HPC. Even with this tremendous progress over the past few years, the adoption of GPUs by the wider computational science community faces several hurdles. These hurdles include programmability; portability; consistent performance; limitations of architectural interfaces; and the fact that some application features, such as performing I/O, simply won't perform well on current GPU architectures.

Keeneland

The Keeneland project's first phase is underway; it includes the acquisition and operation of the initial delivery system, development of software tools, preparation of applications for GPU computing, and an assessment of the fast-changing technologies.

Table 1. The KID Configuration.

Node architecture	HP ProLiant SL390 G7
CPU	Intel Xeon X5660 (Westmere)
CPU frequency	2.80 Ghz
CPU cores per node	12
Host memory per node	24 Gbytes
GPU architecture	Nvidia Tesla M2070 (Fermi)
GPUs per node	3
GPU memory per node	18 Gbytes (6 Gbytes per GPU)
CPU/GPU ratio	2:3
Interconnect	InfiniBand QDR (single rail)
Total number of nodes	120
Total CPU cores	1,440
Total GPU cores	161,280

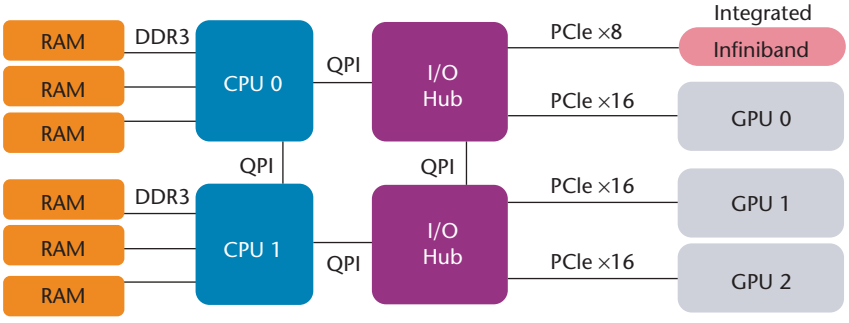

Figure 1. The HP SL390 G7 Node Architecture provides two host CPUs, up to three GPUs, and an integrated Infiniband QDR HCA.

Architecture

KID has been installed and operating since November 2010. As Table 1 shows, KID's configuration is rooted in the scalable node architecture of the HP ProLiant SL390 G7. In particular, as Figure 1 shows, each node has two Intel Westmere CPUs, three Nvidia M2070 Fermi GPUs with 6 Gbytes of memory each, 24 Gbytes of host main memory, and a Mellanox Quad Data Rate (QDR) InfiniBand Host Channel Adapter (HCA). Overall, the system has 120 nodes with 240 CPUs and 360 GPUs; the installed system has a peak performance of 201 Tflops in seven racks (or 90 square feet, including the service area).

More specifically, in the HP SL390, memory is directly attached to the CPU sockets, which are connected to each other and the Tylersburg I/O hubs via Intel's Quick Path Interconnect (QPI). GPUs are attached to the node's two I/O hubs using Peripheral Component Interconnect Express (PCIe). The theoretical peak for QPI's unidirectional bandwidth is approximately 12.8 Gbytes/s, and for PCIe x16 it is approximately 8.0 Gbytes/s. With these two I/O hubs, each node can simultaneously supply three full x16 PCIe links to the GPUs and an x8 link to the integrated Infiniband QDR HCA.

This design avoids contention and offers advantages in aggregate node bandwidth when the three GPUs and the HCA are used concurrently, as they often are in a scientific system. In contrast, other architectures frequently use a PCIe-switch-based approach, and the switch can quickly become a performance bottleneck.

Figure 2. The KID system as installed in Keeneland's data center. The compact 201 teraflops/s system requires only seven racks and 90 square feet of floor space.

Nevertheless, with this PCIe-switch-based approach, vendors are currently offering systems with the highest number of GPUs per node.

The node architecture exemplifies the heterogeneity trends and has one of the highest number of GPU counts per node in the November TOP500 list. The HP SL390 design has significant benefits over the previous generation architecture, but also exhibits multiple levels of non-uniformity.[9] In addition to traditional non-uniform memory access (NUMA) effects across the two Westmere's integrated memory controllers (see Figure 1), the dual I/O hub design introduces non-uniform characteristics for data transfers between host memory and GPU memory. These transfers will perform better if the data traverses only one QPI link (such as a transfer between data in the memory attached to CPU socket 0 and GPU 0) than if it traverses two QPI links (such as a transfer between data in the memory attached to CPU socket 0 and GPU 1 or GPU 2).

In addition, KID's GPUs include other features that can greatly affect performance and contribute to non-uniformity. For instance, each GPU contains error-correcting code (ECC) memory. ECC memory is desirable in a system designed for scalable scientific computing. Enabling ECC gives some assurance against these transient errors, but results in a performance penalty and adds yet another complexity to the GPU memory hierarchy.

Software Tools

As mentioned earlier, one of the risks in using these new heterogeneous platforms involves decreased programmer productivity. On the Keeneland project, we've started collaborating with many vendors and initiated research activities to address this challenge. In particular, software tools currently developed under the Keeneland project include scientific libraries, performance and correctness tools, and virtualization system software.

For scientific libraries, the University of Tennessee at Knoxville's Matrix Algebra on GPU and Multicore Architectures (Magma) project[10] is developing adaptive, dense linear algebra libraries that exploit GPUs and CPUs simultaneously. Magma is similar to Lapack, but includes support for heterogeneous architectures.

For performance and correctness tools, Georgia Tech's Ocelot project[11] offers a modular, dynamic compilation framework for heterogeneous systems, providing various backend targets for CUDA programs and analysis modules for the PTX virtual instruction set. Ocelot currently allows CUDA programs to be executed on Nvida GPUs, AMD GPUs, and x86-CPUs without recompilation. Furthermore, Ocelot supports the construction of a range of correctness and performance tools, such as a memory checker that detects unaligned and out of bounds memory accesses.

For system software and virtualization, Georgia Tech is developing a framework for integrating GPUs into existing infrastructures for virtualization.[12] This infrastructure will be useful for checkpointing and migration in virtualized systems, as well as for load balancing and debugging.

Technical Assessment

Because the architectures and software for heterogeneous computing are changing rapidly, we have an ongoing effort to evaluate different architectures and software stacks. For our assessment, we've developed the Scalable Heterogeneous Computing (SHOC) Benchmark Suite.[13] SHOC is a collection of programs designed to test the ability of GPUs and other OpenCL devices for scalable scientific computing. SHOC has benchmarks at three levels of complexity, which measure device "feeds and speeds," important scientific kernels, and portions of full applications. SHOC also includes a stability test to validate and stress new heterogeneous architectures during procurement and installation.

Application Readiness

An important aspect of Keeneland is our applications outreach and readiness activities. In particular, we're working with early adopters to ensure that their applications work well on Keeneland. Moreover, we're aggressively pursuing those applications that might make efficient use of GPUs, but whose developers haven't yet made the jump to rewriting their applications. Our team is surveying,

contacting, modeling, and, in some cases, assisting the applications teams with porting their code to this new architecture.

Science and Applications

During our acceptance testing of the KID system in November 2010, we evaluated the system's performance with various applications and kernels for functionality, performance, and stability. The KID acceptance test included tests for all system components including the CPU, GPU, interconnect, and storage. However, for the sake of brevity, we include only the relevant GPU results here. Unless otherwise noted, the benchmarks and applications were built using the Intel 11.1.073 compilers, Intel's Math Kernel Library (MKL), OpenMPI 1.4.3, and CUDA 3.2RC.

High-performance Linpack (HPL) for TOP500 has become the dominant benchmark for high-performance computing due to its ubiquity, portability, and legacy. It has several characteristics that make it perform well on a GPU-based cluster, including reliance on dense linear algebra operations--exactly the type of regular, throughput-oriented problems that GPUs excel at solving.

Figure 3 shows KID's HPL scaling results compared to the theoretical hardware peak performance and the aggregate measured double-precision, general matrix-matrix multiplication (DGEMM) performance across GPUs. For these results, we used Nvidia's implementation of HPL, version 9. In this version, each MPI task dynamically splits work between the GPU and CPU (using multithreaded MKL). The best performance on the KID node architecture requires three MPI tasks per node, where each task controls one GPU and uses four cores via MKL. We experienced highly variable single-node performance, presumably due to the dynamic load balancing. However, the primary bottleneck for HPL performance on KID was the relatively small size of our host memory. HPL performance is highly dependent on problem size, and nodes with a larger memory configuration are known to realize a higher percentage of peak performance. However, our configuration is sufficient for most of our real-world scientific applications.

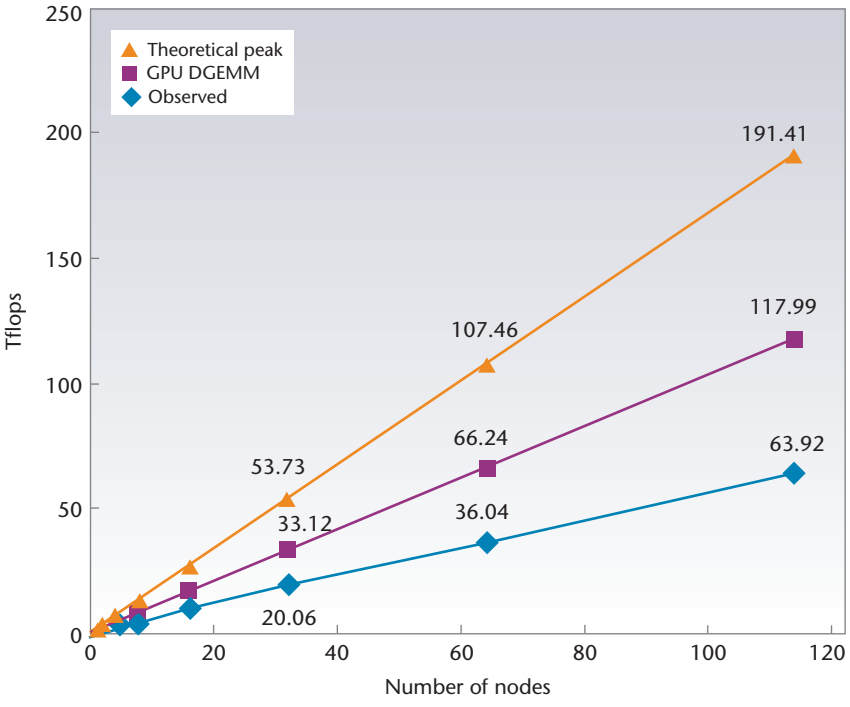

Figure 3. High-performance Linpack (HPL) performance on KID.

In addition, we captured the power usage of our HPL experiments using real-time monitoring of the entire cluster. With this information, we were able to calculate the energy efficiency of KID to be 677 megaflops per watt, which placed it at the #9 ranking on the November Green500 list (www.green500.org/lists/2010/11/top/list.php). Interestingly, eight of the current top-10 systems on this list use GPUs.

Next, we ran many real-world applications on the system, including both CPU-only applications and applications that had been ported to use GPUs. Among these applications, we found that the Groningen Machine for Chemical Simulation (Gromacs)[14] and the Nanoscale Molecular Dynamics (NAMD)[15] biomolecular modeling applications immediately performed well on the system without additional performance tuning. Both applications are used for biomolecular modeling. Table 2 shows performance results for these applications on KID nodes. For Gromacs, we used the *dhfr-impl-2nm* benchmark on a single node. We found that a single M2070 GPU outperforms a single CPU thread by 52 times, and outperforms a fully populated socket by 10.9 times. For NAMD, we used the *apoa1* problem running in parallel on four nodes and realized a 6.6 times speedup when utilizing the GPUs.

Finally, during this early acceptance phase, we also ported and successfully ran the main kernel (Fast Multipole Method) for a blood flow simulation application on KID; the results were presented at the 2010 International Conference on High Performance Computing, Networking, Storage, and Analysis, where this paper was awarded the SC10 Gordon Bell prize.[16]

Table 2. Performance of a single simulation step for Gromacs and NAMD benchmarks on KID nodes.

Benchmark	Configuration	Performance (ms)
Gromacs	1 CPU thread	39.3
	6 CPU threads	8.23
	1 GPU	0.75
NAMD	4 nodes, 12 CPU tasks	104
	4 nodes, 12 GPU tasks	15.8

Currently, many applications perform well on Keeneland, due to previous work by researchers to modify their codes to exploit GPUs.[17–19] The user community has responded positively to the availability of KID. In the first six months of operation, more than 70 projects and 200 users have requested and received access to KID. In the coming months, we'll be preparing for delivery of the final system, deploying GPU-enabled software, and engaging more applications teams so that they can use this innovative architecture to accelerate their scientific discovery, as NAMD, Gromacs, and others have.

Acknowledgments

Keeneland is funded by the US National Science Foundation's Office of Cyberinfrastructure under award 0910735. The Keeneland team includes members from the Georgia Institute of Technology, Oak Ridge National Laboratory, and the University of Tennessee at Knoxville.

References

1. J. Nickolls and W.J. Dally, "The GPU Computing Era," *IEEE Micro*, vol. 30, no. 2, 2010, pp. 56–69.
2. L. Seiler et al., "Larrabee: A Many-Core x86 Architecture for Visual Computing," *ACM Trans. Graphics*, vol. 27, no. 3, 2008, pp. 1–15.
3. J.D. Owens et al., "A Survey of General-Purpose Computation on Graphics Hardware," *Computer Graphics Forum*, vol. 26, no. 1, 2007, pp. 80–113.
4. M. Pharr and R. Fernando, *GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation (GPU Gems)*, Addison-Wesley Professional, 2005.
5. W.R. Mark et al., "Cg: A System for Programming Graphics Hardware in a C-Like Language," *ACM Trans. Graphics*, vol. 22, no. 3, 2003, pp. 896–907.
6. J. Nickolls and I. Buck, "Nvidia CUDA Software and GPU Parallel Computing Architecture," *Proc. Microprocessor Forum*, 2007.
7. Khronos Group, *OpenCL—The Open Standard for Parallel Programming of Heterogeneous Systems*, 2008, www.khronos.org/opencl.
8. J.E. Stone, D. Gohara, and G. Shi, "OpenCL: A Parallel Programming Standard for Heterogeneous Computing Systems," *Computing in Science and Eng.*, vol. 12, no. 3, 2010, pp. 66–73.
9. K. Spafford, J. Meredith, and J. Vetter, "Quantifying NUMA and Contention Effects in Multi-GPU Systems," *Proc. ACM 4th Workshop General Purpose Computation on Graphics Processors*, ACM Press, 2011; doi:10.1145/1964179.1964194.
10. H. Ltaief et al., "A Scalable High Performant Cholesky Factorization for Multicore with GPU Accelerators," *High Performance Computing for Computational Science–VECPAR*, Springer-Verlag, 2010, pp. 93–101.
11. A. Kerr, G. Diamos, and S. Yalamanchili, "A Characterization and Analysis of PTX Kernels," *Proc. IEEE Int'l Symp. Workload Characterization*, IEEE CS Press, 2009, pp. 3–12.
12. A.M. Merritt et al., "Shadowfax: Scaling in Heterogeneous Cluster Systems via GPGPU Assemblies," *Proc. 5th Int'l Workshop Virtualization Technologies in Distributed Computing*, ACM Press, 2011, pp. 3–10.
13. A. Danalis et al., "The Scalable Heterogeneous Computing (SHOC) Benchmark Suite," *ACM Workshop General-Purpose Computation on Graphics Processing Units (GPGPU)*, ACM Press, 2010, pp. 63–74.
14. E. Lindahl, B. Hess, and D. van der Spoel, "Gromacs 3.0: A Package for Molecular Simulation and Trajectory Analysis," *J. Molecular Modeling*, vol. 7, no. 8, 2001, pp. 306–17.
15. J.C. Phillips et al., "Scalable Molecular Dynamics with NAMD," *J. Computing in Chemistry*, vol. 26, no. 16, 2005, pp. 1781–1802.
16. A. Rahimian et al., "Petascale Direct Numerical Simulation of Blood Flow on 200K Cores and Heterogeneous Architectures (Gordon Bell Award Winner)," *Proc. Int'l Conf. High Performance Computing, Networking, Storage, and Analysis*, IEEE CS Press, 2010, pp. 1–11.
17. A. Alexandru et al., "Multi-Mass Solvers for Lattice QCD on GPUs," *Actaphysics*, 29 Mar. 2011; arXiv:1103.5103v1.
18. K. Esler et al., "Fully Accelerating Quantum Monte Carlo Simulations of Real Materials on GPU Clusters," *Computing in Science and Eng.*, vol. 13, no. 5, 2011.
19. G. Khanna and J. McKennon, "Numerical Modeling of Gravitational Wave Sources Accelerated by OpenCL," *Computer Physics Comm.*, vol. 181, no. 9, 2010, pp. 1605–1611.

Jeffrey S. Vetter is the project director of Keeneland, as well as joint professor in the College of Computing at Georgia Institute of Technology and founding group leader of Oak Ridge National Laboratory's Future Technologies Group. Vetter has a PhD in computer science from the Georgia Institute of Technology. Contact him at vetter@computer.org.

Richard Glassbrook is the deputy project director for Keeneland. His research interests include process improvement. Glassbrook has an MS in atmospheric science from the State University of New York at Albany. Contact him at glassbrook@computer.org.

Jack Dongarra is a University Distinguished Professor of Computer Science in the University of Tennessee's Electrical Engineering and Computer Science Department and a member of the Distinguished Research Staff at Oak Ridge National Laboratory's Computer Science and Mathematics Division. He's also a Turing Fellow in the University of Manchester's Computer Science and Mathematics Schools, and an adjunct professor in Rice University's Computer Science Department. His research interests include numerical algorithms in linear algebra, parallel computing, advanced-computer architectures, programming methodology, and tools for parallel computers, as well as the development, testing, and documentation of high-quality mathematical software. Contact him at dongarra@eecs.utk.edu.

Karsten Schwan is a Regents' Professor in the College of Computing at the Georgia Institute of Technology, where he's also a director of the Center for Experimental Research in Computer Systems. His research interests include topics in operating systems, middleware, and parallel and distributed systems, focusing on information-intensive distributed applications in the enterprise and high-performance domains. Schwan has a PhD in computer science from Carnegie Mellon University. Contact him at schwan@cc.gatech.edu.

Bruce Loftis manages the Scientific Support Group at the National Institute for Computational Sciences. His research interests include environmental modeling, large-scale mathematical optimization, and large-scale distributed applications. Loftis has a PhD in civil engineering from Colorado State University. Contact him at bruce3@tennessee.edu.

Stephen McNally is a high-performance system administrator for the National Institute for Computational Sciences at the University of Tennessee/Oak Ridge National Lab. His research interests include GPU clustering, operating system standardization using configuration management tools and best practices, Linux virtualized systems, and high-performance networking. McNally has a BS in computer information systems from Carson-Newman College. Contact him at smcnally@utke.edu.

Jeremy Meredith is a computer scientist in the Future Technologies Group at Oak Ridge National Laboratory. His research interests include emerging computing architectures and large-scale visualization and analysis. Meredith has an MS in computer science from Stanford University. He is a recipient of the 2008 ACM Gordon Bell Prize and a 2005 R&D100 Award. Contact him at jsmeredith@ornl.gov.

Philip C. Roth is a member of the research and development staff at Oak Ridge National Laboratory, where he is a founding member of the Computer Science and Mathematics Division's Future Technologies Group. His research interests include performance analysis, prediction, and tools, with special emphases on scalability, automation, and emerging architectures. Roth has a PhD in computer science from the University of Wisconsin–Madison. Contact him at rothpc@ornl.gov.

Kyle Spafford is a computer scientist in the Future Technologies Group at Oak Ridge National Laboratory. His research interests include high-performance computing, with a focus on emerging architectures. Spafford has an MS in computer science from the Georgia Institute of Technology. Contact him at spaffordkl@ornl.gov.

Sudhakar Yalamanchili is the Joseph M. Pettit Professor of Computer Engineering in the Georgia Institute of Technology's School of Electrical and Computer Engineering. His research interests include the software challenges of heterogeneous architectures, modeling and simulation, and solutions to power and thermal issues in many-core architectures and data centers. Yalamanchili has a PhD in electrical and computer engineering from the University of Texas at Austin. Contact him at sudha@ece.gatech.edu.