Guest Editor's Introduction: Special Issue on High-Performance Computing with Accelerators

David A. Bader, *Fellow*, *IEEE*, David Kaeli, *Fellow*, *IEEE*, and Volodymyr Kindratenko, *Senior Member*, *IEEE*

1 Introduction

It is our honor to serve as guest editors of this special issue of the *IEEE Transactions on Parallel and Distributed Systems* (*TPDS*) on the use of accelerators in high-performance computing. There is a renewed interest in designing and building computer systems based on special-purpose chip architectures. Many research groups already have deployed experimental systems in which Field-Programmable Gate Arrays (FPGAs), Graphics Processing Units (GPUs), the Cell Broadband Engine (Cell/B.E.), and ClearSpeed, to name a few, are used as coprocessors or application-specific accelerators to speed up the execution of computationally intensive codes, and the community is starting to use AMD's Fusion, and NVIDIA's Fermi chips. A few of these efforts have resulted in the deployment of large-scale systems, such as Los Alamos National Laboratory's RoadRunner, which is based on AMD Opteron nodes accelerated with IBM's PowerXCell processor, and the Chinese National University of Defense Technology Tianhe-1 system, which combines Intel Xeon nodes with AMD GPU accelerators. Both of these systems are currently at the top of the most powerful supercomputers list (TOP-500): #3 and #7, respectively. High performance computer designers are turning toward accelerators to increase performance, reduce power requirements, and enable the most challenging applications. To realize the potential of these new systems, however, much remains to be done on the software side as the scientific computing community is only beginning to understand the intricacies and interactions between the new hardware, execution models, software architectures, development processes, and the application transformations necessary to utilize the available resources effectively.

2 Overview of the Special Issue on Accelerators

We are pleased to present this special issue containing 12 high quality contributions that discuss a range of different accelerator architectures and applications that can take advantage of these devices. This issue covers a range of accelerator issues that highlight how these devices can be used effectively to break down computational barriers in challenging applications. Independent of the architecture employed, we must first identify both task-level parallelism and data-level parallelism in order to map a common set of operations to parallel hardware.

FPGAs have been used successfully to offload computational kernels—the function of the FPGA tends to be statically defined, and the accelerator hardware is typically memory mapped. This model of computation has been able to achieve application speedups of 5X-10X in many common applications (e.g., FFTs, convolution), though it is limited by off-chip pin bandwidth and can only change function through entire device reprogramming.

GPUs were originally designed for render graphics; the computer gaming community has driven the rapid advancement in the performance and core densities of high end GPUs. Given that these devices were designed for a very different class of computing task (i.e., graphics rendering), and given the lack of general purpose programming languages for the GPU, high performance computing manufacturers and integrators were reluctant to install these devices in their high performance systems. It was not until much more mature programming environments, such as NVIDIA's CUDA and Khronos' OpenCL became available that we began to see the rampant adoption of GPUs as a standard element in most high performance systems.

In terms of hardware architectures, the IBM Cell/B.E. (jointly designed with Sony, Toshiba, and IBM) provides a rather unique design point, with a main Power Processing Engine (PPE) core connected by an Element Interconnection Bus to eight Synergistic Processing Elements (SPEs) that are essential for acceleration. The design provides for communications through a cache coherent direct memory access (DMA) protocol. While the Cell/B.E. has been used in a number of acceleration configurations, the Los Alamos RoadRunner system provides impressive capability through its demonstration as the first supercomputer to achieve petaflop performance.

Graphics processing units (GPUs) are quickly becoming the most commonly deployed accelerator platform by the high performance community. Both AMD and NVIDIA provide very impressive solutions. Only recently have these manufacturers recognized that there is a growing market for these cards in the high performance computing domain, and they are now producing cards that lack a video output (i.e.,

- *D.A. Bader is with the College of Computing, Georgia Institute of Technology, Atlanta, GA 30332.*
- *D. Kaeli is with the Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115.*
- *V. Kindratenko is with the National Center for Supercomputing Applications, University of Illinois at Urbana-Champaign, Urbana, IL 61801.*

For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org.

1045-9219/11/$26.00 © 2011 IEEE Published by the IEEE Computer Society

they are compute engines, not rendering engines). The communication between the CPU (the host) and the GPU (the device) is through a custom device driver. When a CUDA or OpenCL code gets compiled, the code generated is sent as a string to the AMD or NVIDIA device driver, where the code is just-in-time compiled for the specific GPU installed. GPUs can provide impressive speedups for a variety of applications —typical speedups range from 50X-500X.

Independent of which accelerator platform is chosen, the underlying execution characteristics of the target applications play a critical role in our ability to effectively utilize an accelerator effectively. As we will see in the articles included in this special issue, some applications are limited by the amount of divergent and unpredictable control flow that may be fundamental to the algorithm being used. Others applications may possess irregular memory access patterns which can limit memory efficiency.

3 In This Issue

This special issue contains a collection of papers that nicely cover the hardware accelerator design space, especially if we consider the recent focus on many-core graphics processors and gaming chipsets, as well as applications that exploit these devices.

Engineers and scientists enjoy the ease of use of programming using an interpreted language framework. These environments allow them to quickly carry out computations effectively without significant programming effort. Some commercial examples of these environments include Matlab and Octave, which are especially well suited for working with matrix data. In "Accelerating the Execution of Matrix Languages on the Cell Broadband Engine Architecture," Raymes Khoury, Bernd Burgstaller, and Bernhard Scholz look at offloading Octave computations onto an IBM PowerXCell processor. They report on speedups of up to 12X over a dual core X86 CPU. This paper provides a typical example of the kind of benefits that can be enjoyed by moving computation a many-core accelerator.

In "Cyclic Reduction Tridiagonal Solvers on GPUs Applied to Mixed-Precision Multigrid," Dominik Göddeke and Robert Strzodka extend previous work on mixed precision iterative solvers for sparse linear equation systems by implementing the entire algorithm on the GPU. The authors present a new GPU implementation of a cyclic reduction scheme for solving tridiagonal systems in parallel and use it as a line relaxation smoother for the GPU-based multigrid solver. They show that the resulting mixed precision schemes are always faster than double precision alone and outperform tuned CPU solvers by nearly an order of magnitude.

In "A Framework for Evaluating High-Level Design Methodologies for High-Performance Reconfigurable Computers," Esam El-Araby, Saumil G. Merchant, and Tarek El-Ghazawi consider the impact of the programming model on our ability to effectively exploit accelerators provided on the reconfigurable Cray XD1 system. In this paper, the authors use examples of imperative programming, functional programming, and dataflow programming, and develop a set of metrics that captures both the programming effort involved and the resulting performance improvements achieved when utilizing.

In the paper "Hybrid Core Acceleration of UWB SIRE Radar Signal Processing," Song Jun Park, James A. Ross, Dale R. Shires, David A. Richie, Brian J. Henz, and Lam H. Nguyen leverage both NVIDIA and AMD GPUs to explore the range of GPU accelerator options to speed up a key signal processing applications used in military radar systems. The target application in this study was to provide acceleration for real-time obstacle detection, which was only achieved through the adoption of a GPU in their final design.

A Quantum Monte Carlo application has been redesigned and implemented to work on several application accelerators in "Comparing Hardware Accelerators in Scientific Applications: A Case Study." Rick Weber, Akila Gothandaraman, Robert J. Hinde, and Gregory D. Peterson present design methodologies and demonstrate performance improvements of the code on NVIDIA GPUs, ATI graphics accelerators, and Xilinx FPGAs as compared with a baseline CPU implementation. The authors also consider OpenCL multicore and GPU implementations and demonstrate the OpenCL application portability between these platforms, albeit at a performance cost.

In "Accelerating Pairwise Computations on Cell Processors," an open-source software library for accelerating pairwise computations on the Cell processor is presented. Such computations are needed in many application domains; in a general case, they have a computational complexity of $O(n^2)$. For sufficiently large n, Cell implementation of pairwise computations becomes a challenge due to the limited amount of memory available in the synergistic processing elements. Abhinav Sarje, Jaroslaw Zola, and Srinivas Aluru present a scheduling algorithm based on the particular tile decomposition schema that allows maximization of the data reuse on Cell and demonstrate the performance of their implementation on several applications.

A high-level directive-based language for CUDA programming is presented in "*hi*CUDA: High-Level GPGPU Programming." Tianyi David Han and Tarek S. Abdelrahman describe a set of new directives (pragmas) and present a prototype compiler that translates a C program with the *hi*CUDA directives to a CUDA program. A set of CUDA benchmarks is used to demonstrate the effectiveness of the developed compiler.

In "Design and Performance Evaluation of Image Processing Algorithms on GPUs," In Kyu Park, Nitin Singhal, Man Hee Lee, Sungdae Cho, and Chris W. Kim consider a range of Image Processing algorithms running on NVIDIA GPUs. They compare algorithms taken from four different domains: 3D imaging, feature extraction, image compression, and computational photography. The result is a general set of metrics for evaluating the suitability of image processing algorithms for a GPU.

In the article titled "Exploiting Memory Access Patterns to Improve Memory Performance in Data-Parallel Architectures," Byunghyun Jang, Dana Schaa, Perhaad Mistry, and David Kaeli demonstrate that by applying a set of profile-guided memory transformations on a GPU, very significant performance benefits can be reaped. The paper

presents a model for characterizing the memory access patterns present in the data structures accessed in a kernel, and then applies transformations that improve both our ability to use vectorization (for AMD GPUs) and memory coalescing (for NVIDIA GPUs).

In "Automatic Generation of Multicore Chemical Kernels," John C. Linford, John Michalakes, Manish Vachharajani, and Adrian Sandu extend a kinetic preprocessor tool used in atmospheric modeling codes to generate chemical kinetics code for scalar architectures to support the generation of the numerical solution of chemical reaction network problems for NVIDIA GPUs, Cell processor, and multicore processors. The article presents a comparative performance analysis of chemical kernels from two atmospheric community codes with the kernels generated by the proposed tool across different accelerator platforms.

In "Accelerating Wavelet Lifting on Graphics Hardware Using CUDA," Wladimir J. van der Laan, Andrei C. Jalba, and Jos B.T.M. Roerdink presented a fast and extendable to any number of dimensions wavelet lifting schema implemented on NVIDIA GPUs. The authors also provide a theoretical performance model which is in good agreement with the experimental observations.

In "Assessing Accelerator-Based HPC Reverse Time Migration," Mauricio Araya-Polo, Javier Cabezas, Mauricio Hanzich, Miquel Pericas, Félix Rubio, Isaac Gelado, Muhammad Shafiq, Enric Morancho, Nacho Navarro, Eduard Ayguade, José María Cela, and Mateo Valero present an implementation of Reverse Time Migration seismic imaging technique on Cell, NVIDIA GPU, and SGI RC100 FPGA platforms and compare development methodologies and performance improvements across these platforms. They also provide a performance prediction analysis for the Convey HC-1 FPGA-based system. Three-dimensional stencil computations are at the core of the Reverse Time Migration algorithm; their efficient implementation on the three accelerator-based platforms is the main subject of this paper.

We received a large number of high-quality submissions and it has been a challenge to select a subset of the best papers for inclusion in this special issue. We would like to thank the reviewers whose thorough and thoughtful reviews made this task much easier.

David A. Bader
David Kaeli
Vlad Kindratenko
Guest Editors

David A. Bader received the PhD degree in 1996 from The University of Maryland, and his research is supported through highly competitive research awards, primarily from the US National Science Foundation (NSF), National Institutes of Health (NIH), the US Defense Advanced Research Projects Agency (DARPA), and the US Department of Energy (DOE). He is a full professor in the School of Computational Science and Engineering, College of Computing, at the Georgia Institute of Technology, and the Executive Director for High Performance Computing. Dr. Bader is a lead scientist in the DARPA Ubiquitous High Performance Computing (UHPC) program. He serves on the Research Advisory Council for Internet2, the steering committees of the IPDPS and HiPC conferences, and is the general chair of IPDPS 2010 and the chair of SIAM PP12. He is an associate editor for several high impact publications, including the *Journal of Parallel and Distributed Computing* (*JPDC*), *ACM Journal of Experimental Algorithmics* (*JEA*), *IEEE DSOnline*, *Parallel Computing*, and the *Journal of Computational Science*, and has been an associate editor for the *IEEE Transactions on Parallel and Distributed Systems* (*TPDS*). Dr. Bader's interests are at the intersection of high-performance computing and real-world applications, including computational biology and genomics and massive-scale data analytics. He has cochaired a series of meetings, the IEEE International Workshop on High-Performance Computational Biology (HiCOMB), coorganized the NSF Workshop on Petascale Computing in the Biological Sciences, written several book chapters, and coedited special issues of the *JPDC* and *IEEE TPDS* on high-performance computational biology. He is also a leading expert on multicore, manycore, and multithreaded computing for data-intensive applications such as those in massive-scale graph analytics. He has coauthored more than 100 articles in peer-reviewed journals and conferences, and his main areas of research are in parallel algorithms, combinatorial optimization, massive-scale social networks, and computational biology and genomics. Professor Bader is an IEEE fellow, an NSF CAREER Award recipient, and has received numerous industrial awards from IBM, NVIDIA, Intel, Sun Microsystems, and Microsoft Research. He served as a member of the IBM PERCS team for the DARPA High Productivity Computing Systems program, was a distinguished speaker in the IEEE Computer Society Distinguished Visitors Program, and has also served as Director of the Sony-Toshiba-IBM Center of Competence for the Cell Broadband Engine Processor.

David Kaeli received the BS and PhD degrees in electrical engineering from Rutgers University, and the MS degree in computer engineering from Syracuse University. He is an Associate Dean for the Undergraduate Program at the College of Engineering and a full processor on the electrical and computer engineering faculty at Northeastern University, Boston, Massachusetts. He directs the Northeastern University Computer Architecture Research Laboratory (NUCAR). Prior to joining Northeastern in 1993, he spent 12 years at IBM, the last seven at the T.J. Watson Research Center, Yorktown Heights, New York. Dr. Kaeli has published more than 200 critically reviewed publications, six books, and eight patents. His research spans a range of areas, including microarchitecture to back-end complers and database systems. His most recent work looks at the design, programming, and compilation for graphics processing units. He is an associate editor of the *Journal of Instruction Level Parallelism*, an associate editor of the *IEEE Transactions on Parallel and Distributed Systems*, the chair of the IEEE Technical Committee on Computer Architecture, and a member of CRA's Computing Consortium Council. Dr. Kaeli is an IEEE fellow and a senior member of the ACM.

Volodymyr Kindratenko received the DSc degree from the University of Antwerp (UIA), Belgium, in 1997 and the MS degree from the Vynnychenko State Pedagogical University (KDPU), Kirovograd, Ukraine, in 1993. He is a senior research scientist in the Innovative Systems Laboratory (ISL) at the National Center for Supercomputing Applications (NCSA) and a lecturer in the Department of Electrical and Computer Engineering (ECE) at the University of Illinois at Urbana-Champaign (UIUC). Prior to joining ISL, he was with the NCSA Experimental Technologies group, where he led research efforts on the development and application of advanced sensor and communication technologies for smart environments and high-end visualization and virtual reality systems for high-performance S&E applications. Dr. Kindratenko has coauthored more than 60 articles in peer-reviewed journals and conferences. His main research interests include high-performance computing, special-purpose computing architectures, and applications. He investigates the use of next-generation computational accelerators, such as FPGAs, Cell, and GPUs, for advanced science and engineering applications. He is a senior member of the IEEE and the ACM.