This is an exciting time for high-performance computing (HPC) on novel architectures. The IBM PowerXCell 8i-based Roadrunner supercomputer recently acquired by Los Alamos National Laboratory (LANL) is the first to achieve a petaflop per second on the Linpack benchmark; it’s based on a new, more powerful version of the chip found in the Sony PlayStation. Moreover, Nvidia and AMD are releasing next-generation graphical processors that support mas-
sively parallel double-precision computations and are providing software support for new programming models that are much easier than OpenGL-based code development. In this issue of Gise, we explore these technologies—among others—in a way that will hopefully help you assess how these developments could impact your own work.

**Accelerators for HPC**

High-performance computers, or supercomputers, are distinguished by their ability to perform a very large number of computations in a short amount of time as compared to, say, personal computers. Yet, even supercomputers can benefit from application accelerators, which dramatically shorten the execution time for many HPC applications. Today, the scientific computing community is evaluating several types of accelerators, most notably field-programmable gate arrays (FPGAs), graphics processing units for general-purpose computations (GPGPUs), Sony-Toshiba-IBM’s Cell Broadband Engine (Cell/B.E.), and the ClearSpeed attached processor. Several HPC vendors offer systems that include accelerators as an integral part of their product lines—for example, the SGI reconfigurable application-specific computing (RASC) architecture uses FPGAs, the Cray XT5, uses vector processors and FPGA accelerators, and IBM’s hybrid system architecture uses a PowerXCell as a co-processor. Several Beowulf PC clusters are reportedly outfitted with FPGAs, ClearSpeed, and GPGPU accelerators.

The jury’s still out on which of these computational accelerator technologies will dominate the field because each one brings to the table a different mix of benefits and challenges:

- **FPGAs** have demonstrated unsurpassed performance and power efficiency in applications with limited numerical precision, such as image and signal processing, encryption, and bioinformatics, but their use in floating-point applications generally encounters some obstacles (however, the new chips substantially accelerate floating-point codes as well).
- **GPGPUs** have gained significant popularity in the past two years with the introduction of the Nvidia G80 chip architecture and compute unified device architecture (CUDA) C compiler. The initial G80 architecture provided support for 32-bit numerical types, thus, only those applications that satisfactorily execute with 32-bit arithmetic could exploit this technology. However, with the recent introduction of the Nvidia GT200 architecture and AMD stream processors, this limitation no longer exists.
- **The Cell Broadband Engine** was originally designed as a processor for the Sony PlayStation game console and digital content delivery systems, but from the start, the scientific computing community recognized its performance potential for compute-intensive applications, particularly for single-precision floating-point codes. Its latest version—PowerXCell 8i—is at the core of LANL’s Roadrunner, the world’s fastest supercomputer, and has substantially improved double-precision floating-point performance.
- **Although** the previously listed processors weren’t designed as HPC accelerators per se, the ClearSpeed attached processor was. It natively supports double-precision floating-point arithmetic and is designed to execute application kernels that lend themselves to vector processing.

The scientific computing community has been optimistic but cautious in embracing these new architectures. As with any new and unproven technology, accelerator-based HPC systems present significant challenges. Software developers must deal with unfamiliar programming models, fine-grained parallelism, an intimate understanding of accelerator architecture, and new programming tools and languages. Experience shows that developing applications from scratch for any of these new architectures isn’t trivial, whereas porting existing applications is even more difficult. Furthermore, a good number of integration challenges remain to ensure that libraries and tools developed around novel architectures can be incorporated (and combined freely) in non-novel application code according to established software engineering principles (such as separation of concerns and good modular structure).

In this special issue of Gise, application scientists share their experience in moving scientific codes to accelerator-based computer systems.

**The Articles**

In “Moving Scientific Codes to Other Multicore Microprocessor CPUs,” Paul R. Woodward and his students from the University of Minnesota’s Laboratory for Computational Science and Engineering present a method for implementing numerical algorithms for scientific computing for execution on the Cell/B.E. architecture using a gas dynamics algorithm as an example. The authors propose a two-stage code transformation strategy in which the original Fortran code is first translated into a fully pipelined implementation, which
is then transformed into the Cell/B.E. implementation. They performed many of the transformations manually, but an effort is under way to build automated code transformation tools to assist in at least the most tedious of the code transformations. The authors report achieving 91.2 Gflops performance on a dual Cell/B.E. blade system.

In “Graphical Processing Units for Quantum Chemistry,” Ivan S. Ufimtsev and Todd J. Martinez from the University of Illinois’s Department of Chemistry report on an effort to implement direct self-consistent field (SCF) electronic structure calculations on a GPGPU accelerator platform. In contrast to Woodward’s work, the authors implemented their application from scratch, starting from theory and best algorithms known to date. In particular, they examine different mapping schemes for electron repulsion integral kernels and report up to 80x performance improvements as compared to the leading electronic structure codes running on modern microprocessors. They also assess the adequacy of the single-precision accuracy for quantum chemistry applications.

In “Computing Models for FPGA-Based Accelerators,” Martin C. Herbordt and his students from Boston University’s Department of Electrical and Computer Engineering share their experience in implementing molecular modeling applications on FPGAs. The authors examine several computational models appropriate for FPGA-based implementation and show how to map different applications onto one of them. They argue that when an appropriate computational model is applied, performance improvements up two orders of magnitude are possible.

In the closing article, “QPACE: Quantum Chromodynamics Parallel Computing on the Cell Broadband Engine,” researchers from several European institutions present an overview of the architecture of a novel massively parallel computer called QPACE that’s optimized for applications in lattice quantum chromodynamics (QCD). The machine to be built is a 3D torus of identical processing nodes based on the PowerX-Cell 8i processor tightly coupled using an FPGA-based application-optimized network processor. The authors present a performance analysis of lattice QCD codes on QPACE and corresponding hardware benchmarks.

Many high-quality papers were submitted and peer reviewed for this special issue, so it was especially difficult to select the final four articles. These ones made the cut because they cover both the breadth (variety of architectures and applications) and depth (application implementation details) of the subject. We hope that these articles give a frank account of the state of the art in the field and will motivate further developments.

Volodymyr Kindratenko is a senior research scientist at the US National Center for Supercomputing Applications at the University of Illinois. His research interests include HPC and special-purpose computing architectures. Kindratenko has a DSc in analytical chemistry from the University of Antwerp. Contact him at kindr@ncsa.uiuc.edu.

George K. Thiruvathukal is an associate professor of computer science at Loyola University Chicago. His research interests include programming languages, operating systems, distributed systems, architecture and design, computing history, and enhancing science and computing education with emerging technologies. Thiruvathukal has a PhD from the Illinois Institute of Technology. Contact him at gkt@cs.luc.edu.

Steven Gottlieb is a professor of physics at Indiana University. His research is in lattice QCD, and he’s been using parallel computers for more than 20 years, with a collection of coffee mugs to match. Gottlieb has a PhD in physics from Princeton University. Contact him at sg@iub.edu.