Is High-Performance Reconfigurable Computing the Next Supercomputing Paradigm?

Tarek El-Ghazawi*
The George Washington University

1 Background

High-Performance Reconfigurable Computers (HPRCs) based on integrating conventional microprocessors and Field Programmable Gate Arrays (FPGA) have been gaining increasing attention in the past few years. With offerings from rising companies such as SRC and major high-performance computing vendors such as Cray and SGI, a wide array of such architectures is already available and it is believed that more and more offerings by others will be emerging. Furthermore, in spite of the recent birth of this class of high-performance computing architectures, the approaches followed by hardware and software vendors are starting to converge, signaling a progress towards the maturity of this area and making a room, perhaps, for standardization.

One of the reasons that made HPRCs attractive, is being synergistic systems that have the potential to exploit coarse-grain functional parallelism through conventional parallel processing, while exploiting fine-grain parallelism through direct hardware execution on the FPGAs. Furthermore, due to their hardware programmability, which allows changing the hardware to fit the underlying problem, HPRC systems have shown orders of magnitude improvement in performance, power, size and cost over conventional High-Performance Computers (HPCs). These benefits were particularly harnessed in compute intensive integer applications, such as cryptanalysis and sequence alignment. However, there have been doubts that the same benefits can be attained for general scientific applications. But it is worth noting that the trend in reconfigurable chip sizes and diversity of resources may be relieving some of those concerns. On the software side, due to the hardware reconfigurability of these machines, it is feared that domain scientists will have to learn how to design hardware if they were to use such machines effectively. Progress, though limited so far, has been made in improving the programmability in these systems.

*e-mail: tarek@gwu.edu

SC2006 November 2006, Tampa, Florida, USA
0-7695-2700-0/06 $20.00 ©2006 IEEE

2 Questions to the Panelists

In order to address the overarching question, posed by the title of this panel, with sufficient depth we have assembled a panel of experts who span academia, industry and government research labs and are well recognized in this filed with their outstanding accomplishments to share their thoughts, expertise and insights. In specific, the panelists will answer the following fundamental questions: 1) can FPGAs deliver order of magnitude performance gains in scientific floating-point applications in the foreseeable future? 2) can programming HPRCs become similar to that of HPCs in its level of difficulty? Can it avoid the need for hardware design skills and understanding? 3) Early systems had memory and I/O bottlenecks, will that sufficiently improve in the near future? 4) Some of the early systems imposed a fixed clock requirement on the reconfigurable hardware which can limit the performance, can that be avoided? 5) As processing gets faster, communication and other types of overhead become more pronounced, will that hinder the scalability of these machines? 6) What are the most critical developments in the industry/community that must happen in the near future to maintain the momentum and success of these machines?

Is High-Performance Reconfigurable Computing the Next Supercomputing Paradigm?

Dave Bennett*
Xilinx, Inc

It is clear to most that we are at a point where the direction of computing is about to change radically, and we are already seeing the beginnings of that change. Single core processors haven't gotten materially faster for over five years after four decades of doubling or more every two years or so. Yet the size of data storage continues to grow even faster. This has created an ever-widening gap between the amount of data being collected and stored and the computer power necessary to process it. Entire new industries are being created to exploit new technologies, such as bioinformatics, that create demand for still more computer horsepower.

Clusters are the current answer to the problem, but clusters have significant limitations of their own. First, the bandwidth required to synchronize multiple machines solving a single problem means that every additional computer brought to bear on the problem is only partially effective, and the more computers working on it the less effective each one is. Second, the power required by each processor is approximately the same even if only 10% of that power is being used to solve the problem. This means that these clusters require an enormous amount of infrastructure to deal with the heat and interconnectivity.

The traditional microprocessor suppliers are attempting to solve these problems by putting several processors on a single die. This allows faster interconnection between the processors (at least those on the die) and reduces the power required for the portions of the die (such as memory caches) that can be shared by the multiple processors. Heterogeneous devices, such as the Cell Processor, try to improve the efficiency of the overall system by providing a mixture of capabilities to enable parts of the problem to be mapped to that processor best suited to solve a specific aspect of the problem.

These approaches do offer improvement, but they are fundamentally limited because the interconnections of the buses and capabilities of the various processors are fixed at the time the device is manufactured and cannot be changed. Each problem being solved is unique, however, and will be more efficiently implemented with a system architecture tailored to suit its requirements. Some problems do large numbers of floating point calculations and don't require fast/wide connections to memory. Others are streaming large amounts of data through and do a lot of bit manipulations with no floating point. No fixed architecture is likely to solve a large cross-section of the problem space very efficiently.

This is why reconfigurable hardware is required for optimum implementation of algorithms used in high-performance computing applications. Many if not most of the high-end supercomputers are incorporating FPGAs into their systems. FPGAs are a more effective solution because the architecture can be defined at runtime. Both connectivity and processing capabilities can be tailored to suit the needs of the algorithm being implemented. In particular, the largest part of the die area of an FPGA is consumed by the configurable routing and this is important because in general, the biggest limiting factor of any implementation of a parallel algorithm is data passing. This is why FPGAs will always offer significant opportunities of improvement over any microprocessor with fixed communication buses.

The majority of the silicon used by a microprocessor is not actually operating on each cycle because there is generally only one instruction being executed at a time, and each of those instructions rarely use a significant fraction of the capability of the processor. Then too there are special instructions that make use of special purpose hardware designed to appeal only to a particular vertical market (such as video gamers) but imposed on all other users because the cost of developing a device specifically for that market is more than the potential saving. FPGAs offer a big advantage here because not only can all of the hardware be executing on each clock cycle, but also the exact mix of computational elements can be tuned to precisely that required by the application.

For these reasons, among others, reconfigurable computing offers significant advantages over fixed computing and these advantages are overwhelming in the area of high-performance computing. So while reconfigurable computing may not be for everyone, for those interested in maximum performance it is going to be required.

*e-mail: dave.bennett@xilinx.com

SC2006 November 2006, Tampa, Florida, USA
0-7695-2700-0/06 $20.00 ©2006 IEEE

Reconfigurable Computing: The Future of HPC

Daniel S. Poznanovic*
SRC Computers, Inc.

Abstract

What will the impact of Reconfigurable Computing be on High Performance Computing? This paper defines Reconfigurable Computing, considers the issues and promises, and projects the impact on HPC.

Keywords: Reconfigurable Computing, architecture, HPC

1 Introduction

The question is: What will the impact of Reconfigurable Computing be on High Performance Computing?

To answer this question we must first discuss Reconfigurable Computing (RC) and understand the nature of this computing architecture. RC is a relatively new form of computing, so we must expect some evolution and change. Therefore, we must also project the future direction of underlying technology to properly assess its impact on RC. As will any new technology, there are issues and there are promises. The impact on HPC will be directly related to the success of dealing with issues, and realizing the promise.

2 Reconfigurable Computing

Reconfigurable Computing is a style of computing based upon the use of processors whose circuitry is defined by the algorithms that is to be run in the processor. In other words the algorithm defines the structure of processor. A RC processor can take the form of a traditional von Neumann instruction processor with an instruction set tailored to an application space. A RC processor may also be a direct execution logic processor without an instruction set and based on data flow architecture. Both structures have validity and the future RC processor could be a mix. The powerful fundamental concept is that the application defines the processor.

For this kind of flexibility the chip technology implementing RC processors must allow reconfigurability. The Field Programmable Gate Array (FPGA) is the most often used chip for RC. FPGAs are inherently parallel, operate at power and heat dissipation levels much lower than microprocessors, and are a commodity chip.

3 Promises and Issues

The promises of RC are many. The underlying FPGA technology has significant growth potential in capacity and clock rate, while microprocessor chips are hitting a limit in performance. An application specific processor can more likely achieve peak performance, since compromises need not be made and parallelism can be high. The heat dissipation and power requirements of FPGAs are significantly less than microprocessors. Coupled with achievable parallelism within the chip, petascale performance is possible at lower operating cost and smaller footprint.

There are issues. The tools for implementing RC on FPGAs have traditionally been logic designer's tools, not programming tools. HPC application are developed by programmers not hardware designers. FPGAs run at clock rates that are a fraction of the microprocessor, and therefore require a high degree of parallel execution to achieve high performance. Programming languages have been designed for von Neumann processors. An application's potential parallelism cannot always be expressed in current languages. Historically, RC processors have been accelerator boards that attach via I/O interfaces, and were not integrated into a balanced system. Without balanced data access and compute, RC will not meet HPC requirements. And of course HPC needs floating point, which is resource demanding in RC.

4 Conclusion

Though there are issues to be overcome, the potential for RC is very high. Programming tool development is underway and the path for success is understood. The FPGA roadmap makes even double precision floating point feasible at performance levels well above microprocessor. RC relieves the growing pressure on operating expense yielding superior FLOPS/watt and FLOPS/dollar. RC will positively influence HPC, will become a significant part of HPC solutions, and quite possibly will dominate HPC.

*e-mail: poz@srccomp.com

SC2006 November 2006, Tampa, Florida, USA
0-7695-2700-0/06 $20.00 ©2006 IEEE

Is High-Performance Reconfigurable Computing the Next Supercomputing Paradigm?

Allan J. Cantle *
Nallatech Ltd.

Reconfigurable computing, based on FPGA technology, has been around for over a decade and a growing community has been researching and leveraging the advantage of this technology for many years. The success of FPGA technology has primarily been in the domain of the embedded computing market where users are focussed on one application and are comfortable with using low level "hardware based solutions" to solve their computing problems.

Conversely the mainstream world of computing insists on high level software language design flows that are easily portable and maintainable. Additionally, most compute intensive applications are standardised around IEEE754 Double Precision Floating Point maths, that until recently have not been compatible with FPGA-based computing technologies.

2006 marks a year where FPGA-based technology and tool suites have evolved to the point where they are commercially viable for the HPC software community. The benefits include groundbreaking price/performance levels and reduced power consumption. Additionally HPC applications can be relatively easily ported without having to tolerate challenging development cycles.

Whilst 2006 marks a significant milestone in the adoption of FPGA-based technology for HPC, there are still some significant hurdles to get over to ensure wider adoption. These hurdles include:

1. Training the software community to think in fine grained parallelism and pipelining techniques.
2. Development of advanced tools to aid item 1 above.
3. 100% IEEE754 Floating Point compatibility.
4. Persuading the community to use Single or Integer Precision where appropriate.
5. Creating accelerator customer success stories.

I am of the opinion that the most significant wins in the future of FPGA-based high performance computing will be achieved when users port their complete algorithms to the FPGA. This will produce the most efficient implementation of the algorithm.

Today, this is a leap too far for the existing High Performance Computing community and therefore we need a bridge strategy that can demonstrate success with minimal effort. Each small step will introduce computer scientists to new programming techniques that provide incremental business benefits.

From a competitive standpoint, it is now widely accepted that the world of computing must go parallel in order to continue to benefit from a continuation of Moore's law. This fact has driven the whole community to agree that the face of computing will fundamentally change in the next decade.

To this end "all the balls are in the air" with regards to what processing technology will win in this changing world. Reconfigurable Computing technology with FPGAs have been driven from a hardware perspective whereas Microprocessors and Multi-Cores are driven from a software perspective.

In the end it is almost certain that marketing skill and financial muscle will dictate the winning players in the next decade of computing. Are Xilinx and Altera just Mino's in the fierce ocean of computing or will the likes of Microsoft or the Linux Community conclude that FPGA-based Reconfigurable Computing is the "new way" and place their cards on the table.

Ones thing is certain; exciting times lie ahead in the computing community and there will be plenty of opportunity and business to be made from experimenting with every alternative processing technology.

*e-mail: a.cantle@nallatech.com

SC2006 November 2006, Tampa, Florida, USA
0-7695-2700-0/06 $20.00 ©2006 IEEE

Challenges for Reconfigurable Computing in HPC

Keith D. Underwood*
Sandia National Laboratories†

In practice, reconfigurable computing currently refers to the use of FPGAs for computing applications. While FPGAs have real performance potential, they have yet to enter widespread use in HPC. For narrowly focused domains using single precision floating-point, FPGAs are likely to become a compelling solution in the near future; however, most HPC facilities have a broad application base using double precision floating-point. Thus, to truly penetrate HPC, FPGAs will have to overcome several hurdles:

Performance: Application developers expect at least an order of magnitude improvement in overall application performance to motivate a change in paradigm. With distributed memory parallel computers so widely entrenched, the barrier to entry may be even larger as application developers are reluctant to maintain and validate multiple different versions of a single application. FPGAs have not been demonstrated to yield anywhere near this advantage.

Cost: FPGA based accelerators are phenomenally expensive by HPC standards. Given an HPC node that costs $5,000, adding an FPGA can easily cost $10,000. This seriously impacts the cost/performance advantage of the FPGA.

Architecture: Most systems still treat the FPGA as a subordinate accelerator with a *subroutine* mentality. This leads to a mentality where either the FPGA or the hosting processor is executing a single serial stream. The FPGA needs to become a full peer.

Algorithms: Much as new algorithms and parallelization strategies were needed with the rise of the massively parallel processing (MPP) paradigm, new algorithms will be needed to leverage the unique capabilities of FPGAs while interacting with microprocessors as peers. This means partitioning the application differently than any system in existence today.

Portability: When application developers create a new application, they expect it to run everywhere. One of the fundamental enablers of the widespread adoption of MPPs was the rise of the MPI standard. With MPI, a single program would run on any MPP (and, now, commodity clusters) with minimal porting effort. FPGAs are a long way from this goal.

Reliability: An FPGA system that delivers an order of magnitude improvement over the best current MPP will have many more parts than the MPP. Many of those parts will be FPGA devices that are sensitive to soft-errors and are hard to detect errors in (see work by Quinn and Graham at LANL). The corresponding decrease in system reliability will be unacceptable for major HPC procurements.

One of FPGAs largest claims to fame is perhaps their biggest weakness: the performance advantage is not compelling for most HPC applications. They also add a high cost and a major change in programming paradigm. When reconfigurable computing yields more than an order of magnitude performance improvement, at a comparable cost, with comparable reliability, and with comparable portability, then it may motivate application developers to do the development necessary to see widespread adoption of reconfigurable computing systems.

*e-mail: kdunder@sandia.gov

†Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

SC2006 November 2006, Tampa, Florida, USA
0-7695-2700-0/06 $20.00 ©2006 IEEE

Reconfigurable Computing - Are We There Yet?

Rob Pennington*
National Center for Supercomputing Applications

The use, potential and realized, of high performance computing as a tool for scientific exploration has grown to the point where computational cycles are a fundamental requirement for research programs across a broad range of science and engineering disciplines. This requirement will continue to grow as research programs grow in scope and sophistication and as new programs begin to need such resources to accomplish their goals.

These pressures are mounting and are driving the interest in pushing the limits of current technologies and in examining new technologies for high performance computing. This includes reconfigurable computing as well as other technologies. The task in high performance computing is to reduce the time and effort involved in developing new applications or moving to new platforms, reduce the total execution time for scientific and engineering research programs dependent on HPC capabilities and reduce the physical infrastructure necessary to support HPC resources.

Adapting current HPC applications to operate efficiently on a reconfigurable platform is a labor intensive task, generally requiring detailed knowledge of the algorithm, the implementation of the algorithm and the platform to be used. Currently, it may take several months of concentrated effort by a team with specialized knowledge of each of these aspects to create an efficient version of the application for a reconfigurable platform starting from highly optimized code for a microprocessor based platform. The similarities of this effort to the early phases of past technology transitions, such as the adoption of MPPs and clusters, is significant. The development and widespread adoption of advanced tools that minimize the development time while presenting a familiar environment to the programmer is essential for broad adoption.

The scale of applications requirements is driving the increased size of many new systems. This increased size translates into larger numbers of nodes needed to respond to the pressures for more cycles in more immediate ways, including reduced turnaround time for batch type jobs, rapid response for event driven applications and interactive response for data visualization driven areas. The effective use of small numbers of reconfigurable components and platforms is the immediate goal of many current efforts. In order for reconfigurable systems to have a significant impact in HPC, they will need to be usable and useful at larger scales to greatly decrease the time required to execute large scale applications. To make a significant difference, they should provide an order of magnitude increase in performance/decrease in execution time.

One promise of reconfigurable computing is the opportunity to achieve high levels of efficiency in the execution of applications at low power and space requirements. This efficiency will be increasingly necessary as the cost of housing and powering petascale and even larger systems over the coming decade will exceed the available HPC budgets of many institutions and programs. New machine rooms for petascale HPC systems being designed to provide power measured in tens of megawatts and floor space measured in tens of thousands of square feet.

The potential exists for reconfigurable computing in the current environment. The task is to use applications to drive the technology to demonstrate the usefulness of the technology in an efficient and cost effective manner.

*e-mail: robp@ncsa.uiuc.edu

SC2006 November 2006, Tampa, Florida, USA
0-7695-2700-0/06 $20.00 ©2006 IEEE

Reconfigurable Computing: The Road Ahead

Duncan Buell*
University of South Carolina

It has been true since reconfigurable computing (RC) began in the late 1980s that one could not merely argue about the possibility of improved performance from RC; one had to demonstrate improved performance. For that, once need a physical machine as well as some demonstration applications. For the demonstrations to succeed, one needed a programming environment that would allow the applications to be implemented.

It is an exciting thought that we now have a number of vendors (SRC, Cray, SGI, Nallatech, Linux Networx, at least) offering general purpose machines that have reconfigurable hardware. We also have at least two programming approachesthe all-in-one of SRC's proprietary compiler and a general language tool from Mitrion (and others?). Both approaches build on the fact that FPGAs, the building blocks of current reconfigurable computing, are large and powerful chips.

A final push toward reconfigurable computing is the possibility that power and cooling issues will make huge improvements in performance less necessary, and that modest improvements (a single order of magnitude?) will be sufficient if the power and cooling requirements for the entire system are an order of magnitude smaller than would be the demand from conventional machines.

The pieces are thus in place for success or failure. The SRC approach promises better performance, but at the cost of portability. The general-language approach should be expected to offer poorer performance but perhaps a better plan for dealing with legacy code and migrations across multiple platforms.

Will this somewhat avant-garde approach to computing finally pay off? I honestly don't know, but I do believe that it will finally get an honest test of its capability.

*e-mail: buell@sc.edu

SC2006 November 2006, Tampa, Florida, USA
0-7695-2700-0/06 $20.00 ©2006 IEEE

Opportunities and Challenges with Reconfigurable HPC

Alan D. George*
University of Florida

Conventional systems for high-performance computing (HPC) have grown into mammoth structures with concomitant requirements for power and cooling, where reliability is a challenge due to the massive number of components involved. A new approach, known as reconfigurable computing (RC), has come to the forefront as an important paradigm for the HPC field, often in concert with microprocessor-based computing. With RC, the full potential of underlying electronics in a system may be better realized in an adaptive manner. Preliminary success with RC has been achieved in a variety of areas such as signal and image processing, cryptology, communications processing, data and text mining, and global optimization, for a variety of platform types, from high-end systems on earth to mission-critical systems in space.

At the heart of RC, field-programmable hardware in its many forms has the potential to revolutionize the performance and efficiency of systems for HPC as well as deployable systems in high-performance embedded computing (HPEC). One ideal of the RC paradigm is to achieve the performance, scalability, power, and cooling advantages of the "Master of a trade," custom hardware, with the versatility, flexibility, and efficacy of the "Jack of all trades," a general-purpose processor. As is commonplace with components for HPC such as microprocessors, memory, networking, storage, etc., critical technologies for RC can also be leveraged from other markets to achieve a better performance-cost ratio, most notably the FPGA. Each of these devices is inherently heterogeneous, being a predefined mixture of configurable logic cells and powerful, fixed resources.

Many opportunities and challenges exist in realizing the full potential of reconfigurable hardware for HPC. Among the opportunities offered by field-programmable hardware are a high degree of on-chip parallelism that can be mapped directly from dataflow characteristics of the application's defining parallel algorithm, user control over low-level resource definition and allocation, and user-defined data format and precision rendered efficiently in hardware. In realizing these opportunities, there are many vertical challenges, where we seek to bridge the semantic gap between the high level at which HPC applications are developed and the low level (i.e. HDL) at which hardware is typically defined. There are also many horizontal challenges, where we seek to integrate or marry diverse resources such as microprocessors, FPGAs, and memory in optimal relationships, in essence bridging the paradigm gap between conventional and reconfigurable processing at various levels in the system and software architectures.

Success will come from both revolutionary and evolutionary advances. For example, at one end of the spectrum, internal design strategies of field-programmable devices need to be reevaluated in light of a broad range of HPC and HPEC applications, not only to potentially achieve a more effective mixture of on-chip fixed resources alongside reconfigurable logic blocks, but also as a prime target for higher-level programming and translation. At the other end of the spectrum, new concepts and tools are needed to analyze the algorithmic basis of applications under study (e.g. inherent control-flow vs. data-flow components, numeric format vs. dynamic range), and new programming models to render this basis in an abstracted design strategy, so as to potentially target and exploit a combination of resources (e.g. general-purpose processors, reconfigurable processors, special-purpose processors such as GPUs, DSPs, NPs, etc). While attempting to build highly heterogeneous systems composed of resources from many diverse categories can be cost-prohibitive, and a goal of uni-paradigm application design for multi-paradigm computing may be extremely difficult to perfect, one of the inherent advantages of RC is that it promises to support these goals in a more flexible and cost-effective manner. Between the two extremes of devices and programming models for multi-paradigm computing, many challenges await with new concepts and tools - compilers, core libraries, system services, debug and performance analysis tools, etc. These and related steps will be of paramount importance for the transition of RC technologies into the mainstream of HPC.

Many logistical challenges and opportunities also await this field. For instance, we can benefit greatly by recognizing and leveraging the overlap of interests between the heretofore often separate activities in HPC and HPEC. While platforms and applications vary between them, much synergy can be attained by exploiting their common bonds in terms of both vertical and horizontal challenges for RC. Also, we must apply lessons learned (often painfully) in fields of computing that preceded RC, such as the importance of standard interfaces and the proper balance between open and proprietary solutions, in terms of innovation versus widespread success.

*e-mail: ageorge@ufl.edu

SC2006 November 2006, Tampa, Florida, USA
0-7695-2700-0/06 $20.00 ©2006 IEEE