A Reconfigurable HEC Platform

R. W. Brodersen, J. Wawrzynek, V. P. Srini, A. Vladimirescu, D. P. Orofino, J. Hwang
C. Chang, B. Richards, K. Camera, H. So, N. Zhou
University of California at Berkeley, Xilinx Inc., MathWorks Inc.

1 Introduction

Ever since general purpose CMOS processors based on Von Neumann architecture (stored program, sequential execution, and time multiplexing hardware) became the underlying fabric of high end computing (HEC), the basic strategy to improve performance has been twofold: 1) use CMOS circuits that have the maximum performance on a single chip using the most advanced technology available, and 2) assemble N of these chips along with memory attempting to achieve an N times computational increase.

We feel that a new approach to HEC using hardware reconfigurable chips for matching the computation in an application to the hardware resources and a structural programming model using discrete-time based block diagram (DTBD) can provide higher performance at lower cost and power consumption while maintaining the general purpose nature of computation. After detailed justification of this approach, Section 2 presents actual results of an experimental high-performance computer constructed at UC Berkeley based on reconfiguration. Section 3 proposes a scalable solution using the advances in technology.

1.1 The obsolescence of Von Neumann era strategies

It has been over 50 years since Von Neumann first developed the processor architecture in widespread usage today. The experience and software tools that have been developed with this architecture have resulted in an enormous inertia that continues to hinder the consideration of any fundamentally different approach. Until recently the Von Neumann strategy had a number of advantages. First, it provided a means to easily exploit the cost, power and performance improvements made possible by the technology advancements characterized by Moore's law. Second, the software development (at least for the individual processors) was built on a wide experience base as well as a number of legacy programs that required the underlying Von Neumann computational model. The extension to parallel computers with Von Neumann nodes has been an active area for many years and many important applications have been parallelized.

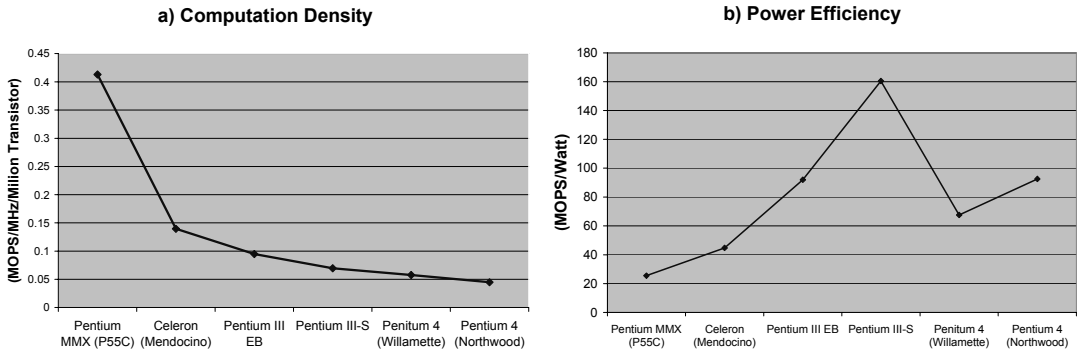

Figure 1: a) Computation Density and b) Power Efficiency vs. technology for Intel Microprocessors

However, it is amazing that an approach that was developed over 50 years ago is still so dominant, in particular because the basic technology assumptions which were present when Von Neumann developed his approach are no longer valid. In particular, one of the strategies he proposed was to time multiplex the computational hardware because it was extremely expensive. While it was true 50 years ago, it clearly is no longer valid. In fact, the basic strategy of time multiplexing hardware has resulted in power and area inefficiencies which are causing the basic Von Neumann architecture to be unable to exploit the

improvements provided by Moore's law scaling. Figure 1 shows the computation density and power efficiency vs. technology for Intel microprocessors. Figure 1a) shows the computation density in MOPS per MHz per million transistors to be decreasing with technology.

The basic strategy for improving the performance of individual Von Neumann processors has been to increase the clock rate, because it has been found that improving parallelism at the processor level through instruction set parallelism, multiple issues, and multiple threads, quickly reaches a point of diminishing returns. The result of this clock rate increase has been to increase the power density in the computational portion of the processor to the point that the heat can not be extracted fast enough. To avoid component damage, the performance obtained from each individual processor must be limited. This is a fundamental change in that power now limits performance, rather than circuit speed. The power efficiency graph for Intel microprocessors, shown in Figure 1b, decreases with technology. This problem will also affect HEC platforms which are based on the highest performance processors, for this power problem only gets worse as large numbers of processors are used. For example, if a GigaOp/sec processor consumes approximately 50W (assuming an energy efficiency of 100 MOPS/Watt), then 100 TeraOps will require 1 MegaWatt, not including the energy for cooling. Likely, the solution will be to avoid these leading edge processors for future HEC platforms, but instead use ones that have lower performance and thus lower power consumption. However, backing off to lower performance processors means that a Von Neumann based fabric will cease to exploit all the advantages of technology scaling.

Fast processors require high bandwidth data access to memory. The speed of low-cost large capacity memory (DRAM) has historically lagged behind the processor clock rate because memory technology has been optimized for capacity (chip area) rather than speed. Memory hierarchy has been widely used to reduce the memory data access time. A consequence of the heavy time multiplexing inherent in the Von Neumann model is that a large amount of intermediate state must be stored in fast on-chip memory. This memory further reduces the area available for computation, therefore only a relatively small area of a processor chip is actually performing computation.

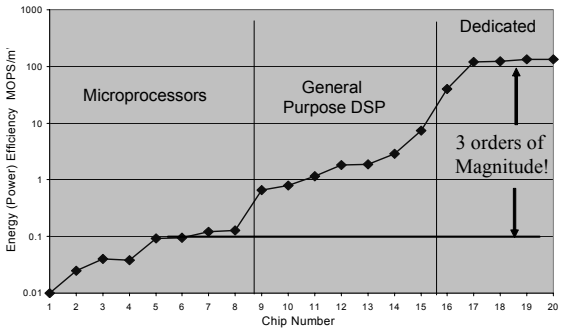

Figure 2: Computational Energy Efficiency of Various Processors

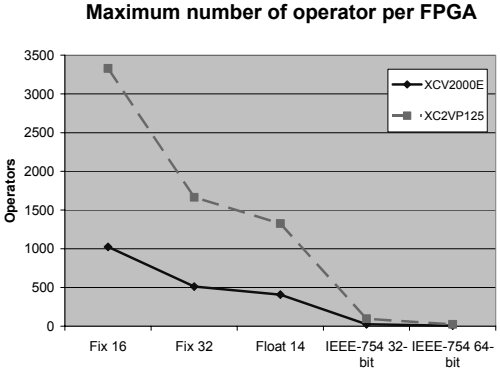

Figure 3: FPGA Capacity for Various Precision Operators

To understand the gap between optimal hardware and what is provided by the present Von Neumann solutions it is instructive to compare the MOPS per Milliwatt for a number of recent circuits that have been presented in the International Solid State Circuits Conference, the primary conference for presenting the latest results in integrated circuit designs. In Figure 2, there are 20 different chips which range from 0.18-0.25 micron technology indicating the computational energy efficiency expressed in MOPS/ Watt. There are 3 classes of architectures, starting from general purpose processors which are a mix of mainstream processors including Power PC's, Pentiums and Alphas which are labeled microprocessors, through DSP's which have a higher level of parallelism through to dedicated chips which have the highest throughputs and levels of parallelism similar to what is achieved in Field Programmable Gate Arrays (FPGAs). While the 3 orders of magnitude shown here are not achieved with FPGA's because of the overhead of reconfigurable interconnect, it does indicate the underlying

capability of the technology. The performance of state-of-the-art FPGA's is on the order of the efficiencies represented by the best DSP's.

1.2 A possible solution

It is clear that a fundamentally different set of strategies and a new computational model is required if the inherent inefficiencies of the Von Neumann era approach is to be avoided. An approach is needed that exploits the increased density of logic and memory capacity as provided by technology scaling. This increase in logic and memory capacity should be used to provide spatial parallel computation at the chip level to achieve high levels of performance, rather than attempting to do it by time multiplexing a small amount of hardware. Since the application focus for HEC is fundamentally general purpose computing, the particular configuration of the parallel architecture needs to be fully reconfigurable. Furthermore, it is desired that the spatial parallelism be exploited at all levels in the application (instruction, thread, task, user).

A strategy for the basic unit to be replicated in an HEC platform that meets the above requirements uses chips that have the ability to be configured to implement any arbitrary computational structure. The goal of mapping an algorithm into a spatial computational structure is to minimize the need for memory and maximize the number of parallel computational units. This mapping is analogous to compilation in a Von Neumann processor. The present commercial realization of this technology is Field Programmable Gate Arrays (FPGAs). FPGAs were originally designed to replace small amounts of random logic; however, because they exploit the density increases of the technology so well, they have now increased in capability to the point where they offer the highest computational density of any programmable chip. The basic architecture of an FPGA is composed of 100's of thousands of primitive logical blocks which have a fully reconfigurable network that allows the logical blocks to be arbitrarily interconnected. For example, to implement a fully IEEE compliant floating point unit requires 1500 blocks, while a 16-bit integer adder only requires 16 blocks.

The state of the art FPGA's in the most advanced technology are able to implement up to 86,000 MOPS (millions of operations per second), where an operation is a 16 bit add equivalent or 9,600 MFLOPS for full IEEE 754 single precision floating point operations. This is achieved by implementing a large number of simple processing elements on the same FPGA (spatial parallelism), with each processing element running at 100 MHz. Due to the fact that the basic FPGA architecture directly exploits the logic density improvements from technology scaling, the performance of an FPGA increases by a factor of 2 with each new process generation. In fact, FPGA's are now used by IC fabrication foundries to drive their newest process development because of the regular structure of an FPGA and ease of designing into a new process generation.

Since each new process generation reduces the minimum feature size by a factor of 0.7, this results in a 2.4 times increase in computational throughput (assuming 2 times more units and approximately a 1.2 times increase in clock rate). The relatively low clock rates (< 250 MHz) protects the FPGA solution from the power limited computational limits being seen by modern Von Neumann processors and matches with the large capacity memory speed.

While the performance of a single FPGA is impressive, to achieve HEC computational requirements, arrays of FPGA's will be required. Instead of taking the current approach of clusters of multiprocessors, a large capacity virtual FPGA is constructed out of an array of densely connected physical FPGAs. To program a single FPGA, the algorithms of an application are expressed in Discrete Time based Block Diagrams (DTBD). These diagrams are spatially partitioned and directly mapped to the logic elements on the FPGA using automated hardware synthesis tools.

A unique characteristic of fully reconfigurable processors is that the computation architecture can be optimized for each application, and therefore the performance that can be achieved is near the maximum performance limit of the array. Because the reconfigurable interconnect is an integral part of the computing fabric, the machine is very flexible in its communications. This flexibility allows the system

to emulate any number of different communication and synchronization strategies, such as message passing, static scheduled interconnection patterns, and other ad hoc schemes on a task specific basis. Furthermore, the actual arithmetic can be optimized for each application. For legacy applications, it is possible to provide IEEE compatible arithmetic, but for many applications a more optimal arithmetic can be used as well. This optimization can provide another 3 order of magnitude increase in computational density, as shown in Figure 3. To exploit this capability optimally will require "compiler" like optimizations, which determine the accuracy of the arithmetic which is required. There has been some initial work in this area, but it remains relatively unexplored.

Modern FPGA chips typically run at a couple of hundreds of MHz, and offer large amounts of parallel I/O pins (currently up to 1200 for Xilinx Virtex II Pro series). The clock rate matches the current speed of DRAM technology and the large number of I/O pins offer the solution to the high memory bandwidth requirement by supporting up to 16 independent 8-byte-wide DDR memory channels. In addition, the high throughput multi-giga bit transceivers available on FPGAs offer enough bandwidth to construct an array of FPGAs that are locally interconnected to their neighbors such that the off-chip interconnection behaves similar to the on-chip interconnection. This allows a straightforward extension of the single FPGA programming model to arrays of FPGAs through the addition of FPGA level spatial partitioning.

2 Berkeley Emulation Engine Platform

At UC Berkeley we have implemented an array composed of 20 FPGAs (Xilinx Virtex-E 2000), shown in Figure 4, which has the overall capability of 600,000 MOPS or 20,000 MFLOPS while running at 60 MHz. An automated design flow in routine use compiles and directly "compiles" applications described using MathWorks' Simulink/Stateflow/Matlab tools directly onto the FPGA array. The array is in use by students in classes and researchers at UCB, as well as by users at a number of other Universities over the internet through an ethernet interface into the array. The programming model while particularly optimal for stream based programming, also can support finite difference and vector based computations. While it has been primarily used for real-time emulations of communication systems and other signal processing applications, the architectural flexibility can also straightforwardly support the following type of computations at near peak array performance:

- Linear algebra and Finite Difference Time Domain (FDTD) for Computational Fluid Dynamics (CFD), electromagnetic wave propagation
- Analog and hybrid digital/analog circuit simulation
- Systems biology, bioinformatics, gene/protein expression
- Image recognition, video processing

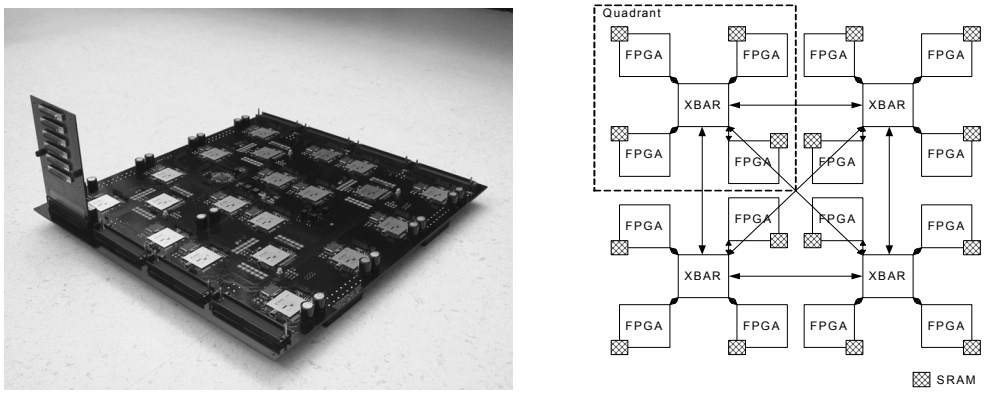

Figure 4: The 0.6 TeraOp BEE FPGA Array & Topology Diagram

For some of these applications, HEC implementations demand vector- and matrix-based representation of the system equations or processing elements in order to capture and exploit the parallelism inherent in such systems. The fine-grained parallel architecture uses optimized architectures to solve factorizations of very large matrices or to perform Gaussian Elimination (GE). The performance of GE on BEE for various matrix sizes is shown in Figure 5. At the execution rate of 1.2 GFLOPS, each 0.18 micron technology Xilinx Virtex-2000E FPGA can achieve the performance of the current state-of-art 0.13 micron technology Intel Itanium 2 processor which is 2 process generations more advanced. On a FPGA with the same fabrication technology, such as the Xilinx Virtex-II Pro 125, over 9.6 GFLOPS can be achieved in each FPGA, which is 6 times higher performance than the Itanium 2 processor.

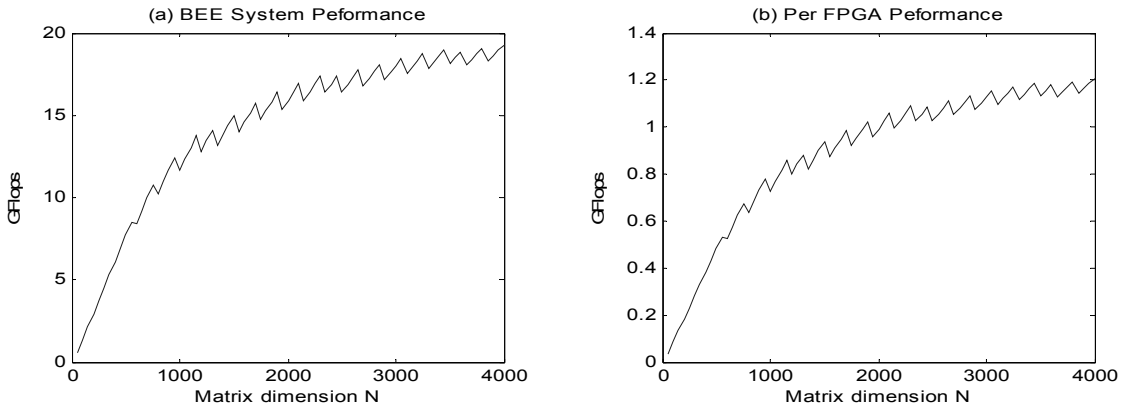

Figure 5: BEE System and per FPGA Performance

3 Scaling up the BEE to HEC Performance Levels

To see the capability possible with larger FPGA arrays, the parameters of a possible HEC will be given using the most advanced FPGAs currently available with a straightforward scaling of the BEE architecture using only local interconnect. These new chips have over 125,000 primitive logic elements, 10Mbits of on-chip SRAM, 1040 user I/O pins, and twenty 2.5 Gigabit/second transceivers for interconnect. Each of these FPGA's can implement up to 48 floating point processing units and the large number of I/O pins allow connection of to up 12 Gbytes of off-chip DIMM memory. Each of these memories is capable of 3.2 GBps data bandwidth so that all 48 floating point processors can operate without any limitation from memory bandwidth constraints. A parallel version of the GE algorithm was designed for analysis and the estimated performance is shown in Figure 6 for a scaled up system composed of different numbers of FPGAs.

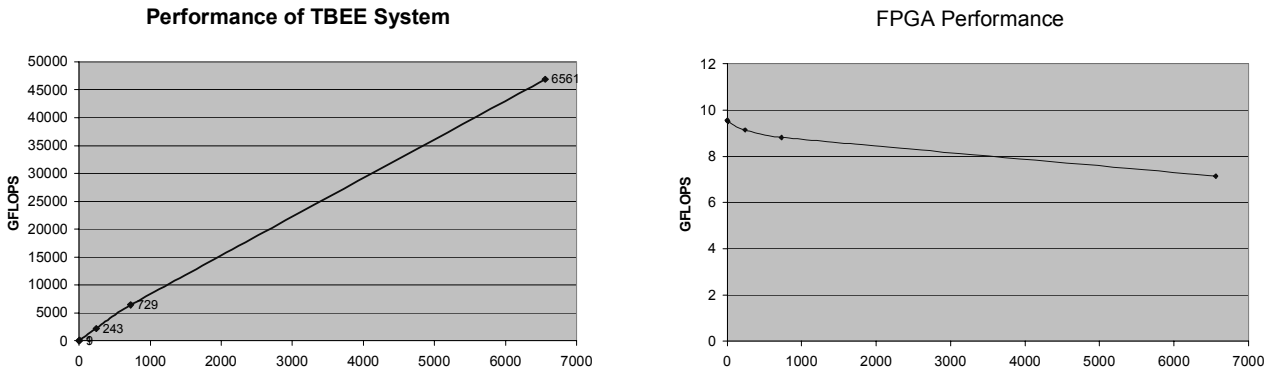

Figure 6: Performance of a Scaling up the BEE array for Gaussian Elimination

More information can be found at http://bwrc.eecs.berkeley.edu/Research/BEE

Contact Information

R. W. Brodersen
Title: Professor, EECS Dept.
Affiliation: University of California at Berkeley
Postal Address: 2108 Allston Way, Suite 200,
Berkeley, CA 94704
Email: rb@bwrc.eecs.berkeley.edu
Phone: (510) 666-3110

J. Wawrzynek
Title: Professor, EECS Dept.
Affiliation: University of California at Berkeley
Postal Address: 631 Soda Hall
Berkeley, CA 94720-1776
Email: johnw@cs.berkeley.edu
Phone: (510) 643-9434

V. P. Srini
Title: Research Associate
Affiliation: University of California at Berkeley
Postal Address: 2108 Allston Way, Suite 200,
Berkeley, CA 94704
Email: Srini@eecs.berkeley.edu
Phone: (510) 333-8271

D. P. Orofino
Title: Sr. Manager of DSP Devlopment
Affiliation: MathWorks Inc.
Postall Address: 3 Apple Hill Drive
Natick, MA 01760
Email: don@mathworks.com
Phone: (508) 647-7568

J. Hwang
Title: Sr. Manager, Vertical Products
Affiliation: Xilinx Inc.
Postal Address: 2100 Logic Drive
San Jose, CA 95124
Email: jim.hwang@xilinx.com
Phone: (408) 879-4883

A. Vladimirescu
Title: Visiting Faculty, EECS Dept.
Affiliation: University of California at Berkeley
Postal Address: 2108 Allston Way, Suite 200,
Berkeley, CA 94704
Email: Andrei@bwrc.eecs.berkeley.edu
Phone: (510) 666-3102

C. Chang
Title: Graduate Student Researcher
Affiliation: University of California at Berkeley
Postal Address: 2108 Allston Way, Suite 200,
Berkeley, CA 94704
Email: chenzh@bwrc.eecs.berkeley.edu
Phone: (510) 666-3121

B. Richards
Title: Technical Staff
Affiliation: University of California at Berkeley
Postal Address: 2108 Allston Way, Suite 200,
Berkeley, CA 94704
Email: Richards@bwrc.eecs.berkeley.edu
Phone: (510) 666-3107

K. Camera
Title: Graduate Student Researcher
Affiliation: University of California at Berkeley
Postal Address: 2108 Allston Way, Suite 200,
Berkeley, CA 94704
Email: kcamera@bwrc.eecs.berkeley.edu
Phone: (510) 666-3112

H. So
Title: Graduate Student Researcher
Affiliation: University of California at Berkeley
Postal Address: 2108 Allston Way, Suite 200,
Berkeley, CA 94704
Email: skhay@bwrc.eecs.berkeley.edu
Phone: (510) 666-3127

N. Zhou
Title: Graduate Student Researcher
Affiliation: University of California at Berkeley
Postal Address: 441 Soda Hall,
Berkeley, CA 94704
Email: normzhou@cs.berkeley.edu
Phone: (510) 643-8229