

# THE DESIGN AND APPLICATION OF A HIGH-END RECONFIGURABLE COMPUTING SYSTEM

Chen Chang, John Wawrzyniek, Pierre-Yves Droz, Robert W. Brodersen  
{chenzh, johnw, droz, rb}@eecs.berkeley.edu  
Department of Electrical Engineering and Computer Sciences  
University of California, Berkeley

## Abstract

This paper summarizes our current effort in the BEE2 project to design and construct a high-end reconfigurable computer (HERC) system based solely on field programmable gate arrays (FPGAs) as the processing elements. FPGAs offer many important advantages over conventional microprocessors, such as flexible arithmetic precision, higher computational density per unit silicon area, and lower power consumption. The programmable interconnect structure unique to FPGA technology has made it possible to tailor a HERC system, such as the BEE2 system, on a per-problem basis to best take advantage of task specific data-flow, memory access patterns, and node-to-node communication patterns.

The BEE2 project is a coordinated attack on the elements needed to demonstrate a practical, cost-effective, high-end reconfigurable computer: the design of a processing module to be used as the building block for a family of high-end reconfigurable computers; the development of several programming models; and the demonstration of the efficiency of the machine on a set of applications drawn from various research projects at Berkeley Wireless Research Center (BWRC), ranging from high-performance DSP and communication systems to traditional scientific computing. Preliminary analyses show that on selected DSP applications, when compared to a microprocessor-based system with similar power consumption and cost, BEE2 can provide over 10 times more computational efficiency.

**Keywords:** Reconfigurable Computing, FPGAs, High-End Computing, Super-Computer.

## 1 Introduction

A high-end reconfigurable computer (HERC) is a machine with supercomputer-level performance configured on a per-problem basis to match the structure of the algorithm and data-flow of a computing task. In such a machine, all data-paths, control paths, memory ports and controllers, and communication channels and controllers, can be programmed to match the needs of a particular application.

In the BEE2 project we seek to design,

construct, and program a HERC in a set of application areas. The hardware system is completely built out of commercial common-off-the-shelf (COTS) components. The first batch of two prototype BEE2 system have been assembled and currently in use at the Berkeley Wireless Research Center (BWRC) [1].

### 1.1 Target application domains

There are several computationally intensive problems central to the scientific objectives of BWRC. These problems act as an application benchmark set and design drivers for the specification of the machine architecture and its associated software mapping tools. The applications fall into four broad categories: the design of novel wireless communications systems, high-performance real-time digital signal processing, real-time scientific computation and simulation, and acceleration of computer aided-design (CAD) tools.

Many on-going advanced wireless communication system research projects at BWRC have utilized various advanced design techniques, including sophisticated encoding/decoding techniques (i.e, Turbo codes, LDPC codes, etc), software defined radio (SDR) design, cognitive spectral reuse, multi-mode operation (i.e., GSM, CDMA, TDMA, etc) , and multiple antenna MIMO (multiple-inputs-multiple-outputs) systems, all of which require ever increasing digital processing capability and flexibility at minimal power consumption, especially for mobile applications. The validation of these complex systems requires in-system emulation of the physical layer processing over hundreds to thousands of packets/frames. This emulation would take weeks on conventional microprocessor based computers, and would far exceed the real-time requirement of the analog subsystems. To address these issues, the predecessor of the BEE2 system, the Berkeley Emulation Engine [2], was designed in 2001 and has been in active service at BWRC in the past two years, as a real-time emulator of novel wireless communication ASICs (Application Specific Integrated Circuits). The BEE2 system not only increases the emulation speed by a factor of five over its predecessor, but also provides a scalable solution for ever-increasing emulation capacity requirements. The BEE2 system has been designed

to utilize a high-bandwidth low latency crossbar network to integrate up to 244 modules, each with an emulation capacity of up to 10 million ASIC gates for a total system capacity of over 2 billion ASIC gates. Hence the BEE2 system will be capable of real-time emulation from a single ASIC or SOC to a complete ad-hoc sensor network with thousands of individual nodes.

In contrast to the emulation applications in the first category, where the BEE2 system is part of the design automation tool, the targeted high-performance real-time DSP applications use the BEE2 system as the final implementation. One of the key areas in this domain is radio telescope beamforming, spatial correlation, and wide-band fine resolution spectroscopy for large antenna arrays, such as the Allen Telescope Array. Despite the low numeric precision requirements (typically 4 to 32 bit fixed-point), these applications can easily consume from 1 tera operations-per-second (TOPS) to over 1 peta operations-per-second (POPS), to meet the processing requirements of several GHz of continuous RF bandwidth over hundreds physical antennas. The hard real-time throughput and the high performance requirements make it next to impossible to implement these applications on conventional micro-processors with their nondeterministic execution model. The data-flow processing nature of these algorithms matches the stream-based computation model commonly used on reconfigurable devices, with throughput directly locked to the system clock rate. The radio astronomy community has been using FPGA or ASIC for these applications. However, due to the diverse scientific requirements from one telescope system to another, and the fact that to date most of the existing radio telescope engineering development have been within the scientific research community, almost all of the existing implementations in this area have been custom designs with little reusability from one implementation to another. The BEE2 system is one of the first attempts at providing a scalable, modular, economic solution for a range of high performance radio telescope DSP applications. Furthermore, the BEE2 system provides a unique multi-user environment, much like in a conventional PC cluster, where many users can share a common pool of computing resources, with guaranteed computational throughput. Other applications in this domain include advanced video compression algorithms, video transcoding, image processing for structure-from-motion, obstacle avoidance, general navigation in autonomous vehicles, and real-time hyper-spectral imaging.

Scientific Computing involves a set of computing tasks traditionally solved using

supercomputers. One example of interest to BWRC is 3D electromagnetic field simulation for antenna design. This problem, as with many other large scale simulations of physical systems, is characterized by large systems of partial differential equations, often involving large regular or adaptive grid structures. The conventional methods require Fast-Fourier Transforms (FFT) and operations on large matrices and typically employ double-precision floating-point number representations. Traditionally these problems are not solved in real-time; however, real-time processing of this class of problem would offer significant new applications. Furthermore, we believe many opportunities exist for innovation on the algorithms and mapping of these problems to reconfigurable machines.

Acceleration of Computer Aided Design (CAD) Tools involves the development of techniques for the use of the BEE2 system to speedup the tool-flows for ASIC and FPGA design. Of primary concern is acceleration of SPICE-level circuit simulation. Also, existing low-level FPGA tool flows (placement and routing) are currently too slow to be practical for HERC systems. Current placement and routing cycle times of hours are unacceptable for reconfigurable systems with 100's of FPGAs, potentially each with a unique configuration. We believe that BEE2 like HERC machines can be used to accelerate their own tools. Some early work on hardware-assisted fast routing and simulated annealing for FPGA placement shows promise along these lines and will guide our work.

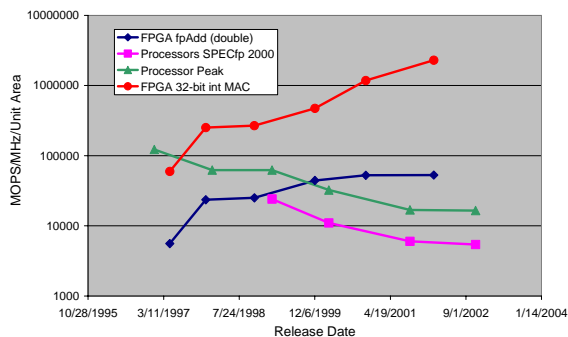
## **1.2 FPGAs as Computing Elements**

Modern day supercomputers are built as clusters of commodity microprocessors. These systems typically demonstrate peak system performance (on artificial benchmarks) in the 100's of GFLOP/s to 10's of TFLOP/s range. The key idea behind the success of this class of machines is the adoption of commodity components, namely off-the-shelf microprocessors and memory modules. In most cases, the low-volume production of supercomputers makes it difficult to justify the cost and time for custom processor development. Also, using commodity components enables rapid technology refresh and performance scaling as new components come to market using new process technology.

Figure 1 illustrates a trend in computational density of microprocessors in comparison to FPGAs. The graph shows the evolution of peak processor computation density over time, through the previous six generations of Intel desktop processors. This data is specific to Intel microprocessors but is typical of all modern day super-scalar microprocessor

architectures. It clearly shows the inability of microprocessors to efficiently turn increasing die area and speed into useful computation. Over the same period, the peak computational density of FPGAs has surpassed Moore's law. Because of their simple hardware structure, each successive generation of FPGA scales naturally with process technology. Also, the use of an increasing number of metal layers for on-chip interconnections, and the inclusion of dedicated functional blocks, has further allowed FPGA devices to scale in peak performance.

Of course, comparing processor and FPGA computational density is not entirely fair, as conventional processors come with a programming model and mature compilation tools, while computation on FPGAs often requires laborious hand-mapping of applications. However, it is the goal of this research to make computing with reconfigurable devices similar in convenience to that of programming microprocessors. Our experience with the Berkeley Emulation Engine (BEE) has demonstrated that, at least within the DSP domain, high user-productivity and efficient application mapping is possible.



**Figure 1: Computational density comparison between FPGA and Intel processors**

Besides the decreasing computational density of microprocessors over time, another troubling characteristic of high-end microprocessor-based systems is the widening gap between memory speed and processor speed. This gap has led to the incorporation of several layers of caches in these systems. The unpredictable latency through the cached memory hierarchy makes it difficult to meet hard real-time requirements and makes programming large multi-processors extremely complex.

A high-end computer based on FPGAs allows a high degree of spatial parallelism and circuit specialization within nodes, resulting in increased performance density even at significantly lower clock rates than microprocessors. The lower clock rate results in reduced power and a better match to the

speed of synchronous DRAM memory systems. Furthermore, multiple independently addressed memory banks can be provided per processing node, resulting in significantly higher memory bandwidth per node than on microprocessor systems, as well as deterministic memory latency as a result of removing caches. By interfacing FPGAs directly to the communication network, the low-level configurability of FPGAs permits a high degree of flexibility in the network—allowing, for instance, circuit switched routing for some applications and packet switched dynamically routed messages in others. Predictable memory and network latency permit static scheduling of memory accesses and data transfers in real-time applications.

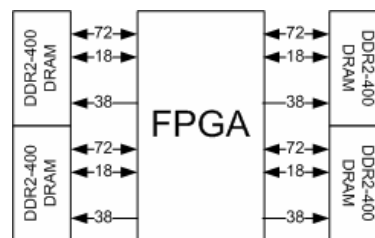
## 2 The BEE2 system

BEE2 hardware system design, just like any other large parallel computer systems, consists of three primary components: processing elements, memory elements, and interconnects. On the system level, processing elements are the FPGA chips; memory elements are the external DRAM modules locally attached to each of the FPGAs; interconnects consists of local connections, that link local FPGAs on the same PCB board, as well as global connections that link multiple boards into a unified system. The main difference of the BEE2 design from traditional parallel computer system design is that the processing elements are FPGA chips rather than microprocessors. In addition to the primary components, BEE2 also incorporates a range of secondary system components for bootstrap, clock distribution, power regulation, and thermal regulation.

The BEE2 system organizes the primary components on three levels of hierarchy: basic computing elements, compute modules, and global communication networks.

### 2.1.1 Basic Computing Elements

As shown in Figure 2, Each computing element comprises one Xilinx Virtex 2 Pro 70 FPGA chip [3] and four DDR2 240-pin DRAM DIMMs. Each DIMM supports a maximum capacity of 1GB.



**Figure 2: basic computing element**

Because of the large number of I/O pins required to simultaneously accommodate both four independent banks of DRAMs and the high bandwidth inter-FPGA connections; we use the largest FPGA package available for the Virtex 2 Pro family, the FF1704 Flip-Chip Fine-Pitch BGA package. Currently two chips can use this package, the XC2VP70 and XC2VP100. Table 1 compares the feature difference between the two chips. The XC2VP70 part is used in the current design. In future usages, XC2VP100 parts can be populated with no change of the PCB board.

**Table 1: Xilinx Virtex 2 Pro Family FPGA Major Feature Comparison**

Features	XC2VP70	XC2VP100
Logic cells	74,448	99,216
PowerPC cores	2	2
MGTs	20	20
Block RAM (Kbits)	5,904	7,992
18bit Multipliers	328	444
DCMs	8	12
Max User I/Os	996	1,040

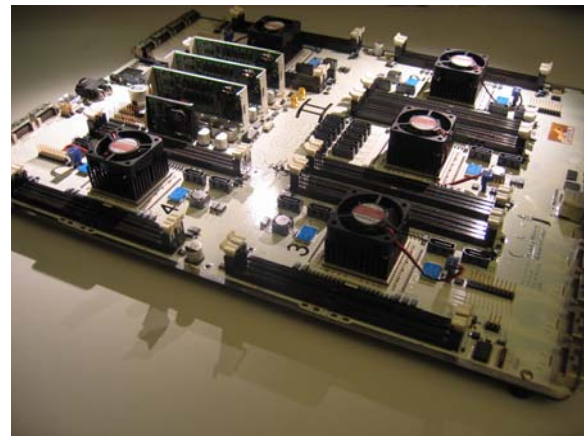
In comparison to the original BEE design, where each FPGA only has 1 external 1 MB ZBT-SRAM chip running at 100MHz with a 32-bit data interface, on the BEE2 system, each FPGA has four independent DRAM channels, each running at 200MHz (400DDR) with a 72-bit data interface (for non-ECC 64-bit data width). Therefore, peak aggregate memory bandwidth is 12.8 GBps per FPGA—32 times more than the BEE external memory bandwidth. However, each memory access in general will incur both RAS and CAS latency of about 6 cycles, that is twice the time of SRAM latency. Although DRAM, with its large memory capacity, is preferred in most computing applications, QDR SRAM modules are the better choice for latency sensitive applications that do not require DRAM capacity. To accommodate these applications, special QDR SRAM DIMMs with the same DDR2 DRAM 240-pin DIMM form factor can be used to connect to the FPGA.

We chose DDR2 memory instead of regular DDR is mainly because of the additional features offered by DDR2 memory, such as 50% lower power consumption, on-die-termination, and additive latency. With supply and I/O voltage at 1.8V instead of 2.5V, DDR2 memory consumes only half of the power of DDR. With both on-die termination (ODT) and output calibration (OCD), signal integrity has been improved to support data rate up to 800MHz. Another new feature of DDR2 is the additive latency, which enables the CAS command to be moved to the

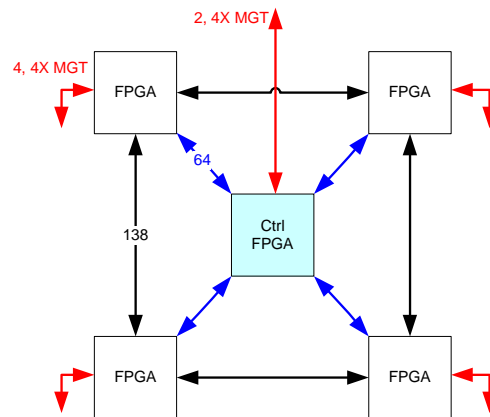
cycle right after RAS, therefore fully pipeline the data access when sequentially read/write through the memory banks. Currently Xilinx Virtex 2 Pro FPGAs support a DDR2 memory interface at speeds up to 400MHz. The data rate is largely limited by the FPGA fabric speed rather than the external signal integrity.

### 2.1.2 Compute Module

Each compute module consists of four basic compute elements and one control element. The control element is similar to compute elements, but with additional global interconnect interfaces and controls signals to the secondary system components. As shown in Figure 4, the connectivity on the compute node can be classified into two classes: on-board LVCMOS connections and off-board MGT connections.



**Figure 3: BEE2 Compute Module Photo**



**Figure 4: Compute module connectivity**

The local mesh connects the four compute FPGAs on a two-by-two 2D grid. Each link between the adjacent FPGAs on the grid has 138 physical single-ended circuits designed to achieve up to 150MHz DDR using the LVCMOS signal standard,

with aggregate bandwidth of 41.1 Gbps per link. From the control FPGA, there are four down links, one for each of the computing FPGA. Each downlink has 64 physical single-ended circuits running up to 150MHz DDR, for a total of 19.2 Gbps per link. These direct FPGA-to-FPGA mesh connections provide a high-bandwidth low latency connection for the FPGAs on the same compute module (e.g. on the same PCB board); so all five FPGAs can be aggregated to form a virtual FPGA with five times the capacity.

All off-module connections use the Multi Gigabit Transceivers (MGTs) on the FPGA. Each individual MGT channels are configured in software to run at 2.5Gbps or 3.125Gbps with 8B/10B encoding. Our use of the MGTs on the compute module is divided into two classes. The primary high bandwidth solution channel-bonds every 4 MGTs together into a physical Infiniband 4X electrical connector, to form a 10Gbps full duplex (20Gbps total) interface. The 4X MGT connections are AC coupled on the receiving end to comply with the Infiniband specification. There are a total of 18 Infiniband connectors on each BEE2 modules—2 from the control FPGA, 4 from each of the 4 compute FPGAs—for a total of 360Gbps off-module communication bandwidth. These Infiniband interfaces can be used for direct inter-module communication, Infiniband switch connection, or direct I/O connection to external high-bandwidth real-time devices, such as gigahertz ADC and DAC.

Because each FPGA has 20 MGTs, exceeding that needed for the Infiniband interfaces, the remaining MGTs (4 for compute FPGA, 10 for control FPGA) are configured as low bandwidth device links using Serial ATA connectors. Due to the requirements of out-of-band signaling in serial ATA disk specification, these connections are not designed to directly connect to disk drives, but for in-chassis peripheral devices, such as audio and video capturing and display.

### 2.1.3 Global Communication Networks

The BEE2 system design supports a wide variety of global connection schemes to accommodate various application requirements. Figure 5 illustrates the basic three types of global communication networks: low latency 4-ary global communication tree, high-bandwidth non-blocking Infiniband crossbar, and 10/100Base-T Ethernet.

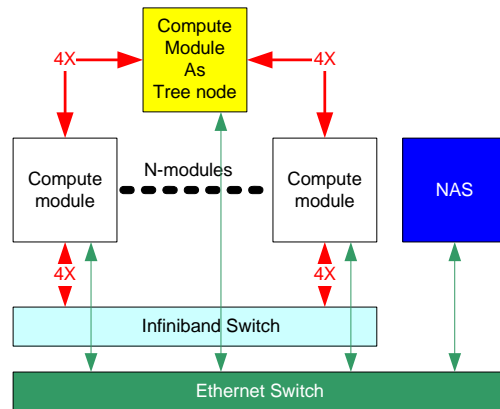


Figure 5: Global communication network

Each BEE2 compute module can be used as a global communication tree node, connecting up to 16 other compute modules as its leaves, and up to two independent parent nodes, as shown in Figure 6. This kind of tree communication network is useful for data aggregation or distribution. Using the 4X Infiniband physical connections, the compute modules can also form many other types of network topology, such as crossbar and 3D-mesh, as shown in Figure 7 and Figure 8 respectively.

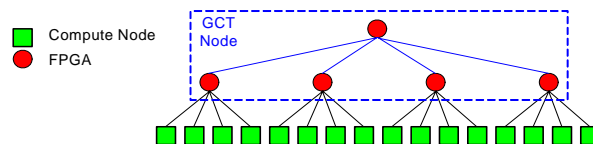


Figure 6: 4-ary global communication tree

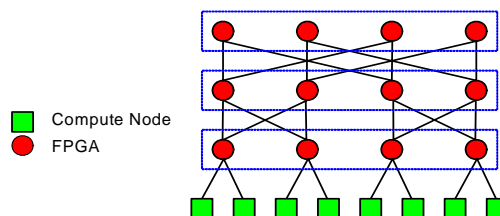


Figure 7: FPGA-based crossbar switch scheme

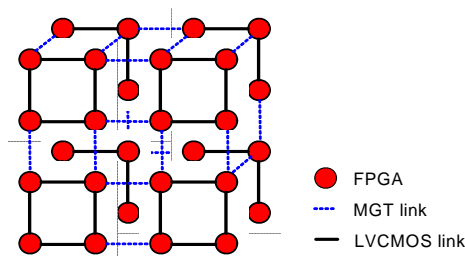


Figure 8: Alternative global 3D mesh connectivity

For applications require high bisection bandwidth random communication among many compute modules, the BEE2 system is designed to take advantage of the Infiniband crossbar switch

technology. The Infiniband network provides the highest throughput connection in the BEE2 system, with commercial switches currently reaching 244 4X ports in a single physical 9U chassis. Other computing and storage devices, such as computer/storage servers can also communicate directly with the BEE2 compute modules using the Infiniband network on the link or network layer. Due to the relatively large IP core required to implement the Infiniband and IP network layer, the connection to computer/storage server on the Infiniband network will occur through the control FPGA only, with the compute FPGA running only a subset of the link layer protocol for direct communication between modules, or to analog front-ends through separate digital interface cards.

The regular 10/100Base-T Ethernet connection, available only on the control FPGA, provides a lightweight easy-to-use network for user interface, low speed system control, monitoring, and data archiving purposes. The compute module has been designed to run Linux OS on the control FPGA with full IP network stack.

#### **2.1.4 Mechanical design**

Each BEE2 module is hosted in one of the 10 blades in a custom 8U rack-mount chassis. With 2 out of the 10 blades reserved for AC/DC power supplies, each chassis packs 8 BEE2 modules, hence a standard 42~48U rack can host up to 40 BEE2 modules. With each BEE2 modules designed to dissipate maximum 250Watt power, which is similar to a standard 1U dual-processor computer server, each rack has a power budget of 10 Kwatts (12 Kwatt AC input). Such as system with 40 BEE2 modules (200 FPGAs) is capable to deliver up to 16 TOPS (32-bit integer) or 2 TFLOPS, with up to 800GB DRAM, and over 7.2 Tbps full-duplex I/O bandwidth.

## **2.2 Programming Models**

Due to the diverse application domains targeted by the BEE2 system, any single programming model would not be optimal for all applications; hence the need for domain specific programming models that can fully exploit the computing power of the BEE2 system.

Currently the most mature programming model for the BEE2 system is the synchronous data flow model for DSP/communication applications, which has been the primary design methodology of the BEE system. Commercial tools, such as Mathworks' Matlab/Simulink, Xilinx System Generator, along with automation tools developed at BWRC, provide automatic mapping from high-level block diagrams

and state machine specifications to FPGA configurations. This programming model and tool flow has proven very successful on a variety of projects at BWRC [4] [5] [6], particularly in the areas of DSP and other datapath intensive streaming applications. To extend this model to support BEE2 specific hardware, stream-based design abstractions are currently being developed for external DRAMs and global communication networks.

As we extend our work into other computational domains, we will explore additional programming methodologies. One of primary interest is the message passing interface (MPI) standard [7], which has been the predominant programming model for existing microprocessor based supercomputers. We plan to implement the MPI library for our HERC as a means to ease the task of porting supercomputer applications. Many novel opportunities exist for specializing MPI functions in the reconfigurable fabric. The computational kernels and inner-loops of the programs can be accelerated using the reconfigurable fabric. For this we will rely on the use of predefined library elements, and on recent work in automatic C to hardware compilation. The PowerPC core on each FPGA provides a conventional RISC processor to run the non-accelerated parts of the program.

In a sense, MPI forms an abstract machine. It is the target for the application algorithm developer, and for us, the organizing structure for the communication circuits built in the reconfigurable fabric. We would like to explore other abstract machines architectures tailored for specific problem domains. For instance, these might include abstract architectures for regular grid problems and cellular-automata, and irregular graph problems, such as transistor-level circuit simulation. These abstract machine models will be the target for domain specific programming languages, and the organizing principle for automatic problem mapping and optimization.

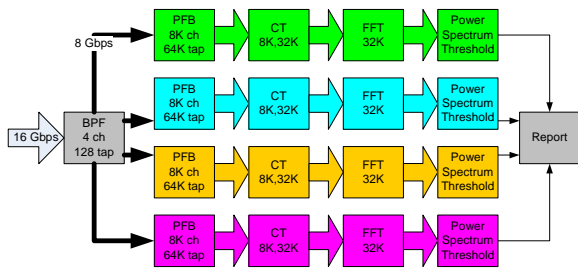
## **3 Preliminary results**

Currently, two BEE2 hardware modules have been manufactured and fully tested, with ten more modules in production. Meanwhile, we have developed a few demonstration applications to showcase the computing power of the BEE2 system. Limited by the scope of this paper, only one application in the high-performance DSP domain is presented here. All performance numbers are reported from the post place-and-route timing analysis of Xilinx ISE design tool suite.

The demonstration application is the 800MHz 1 billion channel SETI spectrometer, that was developed in collaboration with the SETI@HOME

project at UC Berkeley [8]. The SETI radio astronomy scientific requirements demand the spectrometer to have a spectral resolution of less than 1 Hz. The SERENDIP-4 instrument, deployed at Arecibo telescope in 1997, consists of 40 spectrum analyzer boards working in parallel to look at 168 million 0.6Hz channels over 100MHz bandwidth. The SERENDIP-5 instrument, currently under testing, can potentially produce 128 million 0.7Hz channels over the same 100MHz bandwidth in one board. The goal of the BEE2 system is to produce the same sub-Hz spectral resolution over 800MHz, hence deliver a 1 billion channel real-time spectrometer in a single BEE2 module.

As shown in Figure 9, the data processing of the spectrometer starts with 16 Gbps digital inputs from the radio telescope antenna ADC, where 4 bit I,Q samples of two polarization signals are digitized and sent through the Infiniband cable to the BEE2 modules's control FPGA. Using a 128-tap 4-channel polyphase filter bank, implemented in the control FPGA, the 800MHz complex dual polarization input signal stream is split into four 200MHz 8-bit complex dual-polarization streams, each carries ¼ of the 800MHz spectrum. Each of these streams is then sent through the LVCMOS links on the BEE2 module to each of the four compute FPGAs, where a local spectrometer is implemented to split the 200MHz input signal into 256 million 0.745Hz channels, then accumulated at each frequency bin for 10ms to increase the SNR of any potential signal. Finally the power spectrum is calculated, and bins with power over 20dB higher than the local average is reported back to the control FPGA, and then relayed to an external PC over the Ethernet interface.



**Figure 9: One billion channel spectrometer**

The most challenging part of the design is the 256 million channel spectrometer on each of the computer FPGAs. The polyphase filter bank (PFB) technique is used to improve the spectral isolation between adjacent channels. The spectrometer consists of three major steps: the 8K channel PFB, followed by a corner turn, then finally by the 32K point FFT. The 8K channel 64K-tap PFB splits the 200MHz bandwidth input signal into coarse bands of 24.4KHz

each. Since the PFB outputs data in channel order, a corner turn stage is required to reorder the data in time sample order before it can be fed to the 32K point FFT stage.

The corner turn stage is essentially an 8K by 32K matrix transpose. With each matrix element being the 16bit complex dual polarization signal sample, 8 byte per element, a total of 2GB is required to store the matrix. When double buffering is used for the matrix transpose, a total of 4GB memory capacity is needed, which is the maximum capacity of the BEE2 module per FPGA. At the sample rate of 200MHz, the corner turn module needs to read and write one element per cycle to keep up with the data rate, hence a total of 3.2GBps memory bandwidth is required, which is well below the maximum DRAM bandwidth of 12.8GBps per FPGA on BEE2.

The 32K point FFT and the 8K point FFT inside the 8K channel PFB are all implemented using the biplex pipelined FFT structure [9], where the two polarization complex signal are interleaved and fed continuously to the FFT on one signal sample per cycle basis, hence providing fully streamed implementation.

As shown in Table 2, the 256 million channel spectrometer uses all the block RAM on a Virtex-II Pro 70 FPGA, and about 50% of the other resources. When placed and routed on a XC2VP70-7-FF1704C speed grade chip, the maximum clock rate is reported to be 230MHz, well above the required 200MHz clock rate.

**Table 2: 256 million channel spectrometer resource utilization on XC2VP70**

Resource	Utilization (percentage)
Flip Flops	30,964 (46%)
LUTs:	13,326 (20%)
Slices	18,958 (57%)
Block RAMs	328 (100%)
MULT18X18s	128 (39%)

At the targeted 200MHz clock rate, each compute FPGA performs 28.8 Gmult/s and 39.2 Gadd/s, where the multiplications are done with 16-bit operands, and additions with 32 bit operands. With the control FPGA performing 32 Gmult/s and 32 Gadd/s, each BEE2 module delivers over 333 GOPS at a clock rate of 200MHz—more than 100 times the throughput of current state-of-art Intel Pentium-4 processor in terms of integer performance.

With the system power consumption of the billion channel spectrometer estimated at about 150 Watt per BEE2 module (including all memory and other subsystem power consumption), each BEE2 module consumes a similar amount of power as a PC desktop computer. Therefore in terms of computing

throughput per unit energy, the BEE2 system is also over 100 times better than conventional microprocessor based computers.

Currently the BEE2 module hardware cost is estimated at 20K USD when manufactured in several 10s quantity. Comparing to typical single Intel processor computer server hardware cost of 2K USD, the BEE2 still retains 10 times more computing throughput per unit hardware cost.

In applications that require less arithmetic precision, such as the radio astronomy spatial cross correlation computation, the BEE2 system can provide even more computation throughput by leveraging the fine-grain parallelism found on FPGAs. In the correlator design for 32 radio telescope each with 200MHz input bandwidth at 4 bit I,Q dual polarization, each BEE2 module can performance up to 250 billion complex multiply and accumulate (CMAC) operations per second, where multiplication are done with 4 bit operands and then accumulated by 22 bit adders. Each CMAC operation is equivalent to 8 real arithmetic operations, thus each BEE2 will perform 2 TOPS throughput—almost 1000 times more than any existing processors. In these cases, the BEE2 system can provide about 100 times more cost effective, and 1000 times more power efficient solution than any processor-based solutions.

## 4 Conclusion

BEE2 is our first attempt at creating a universal reconfigurable computing system that can target a wide range of application domains, starting from the high performance DSP domain where FPGAs are a proven technology, to the scientific computing domain, where the use of FPGAs still remains a relative novelty. With the introduction of modular FPGA computing platforms, such as BEE, and the improvements of domain specific programming models, we believe that the negative image of FPGA as a user unfriendly technology can be over-turned, and FPGAs can provide efficient solutions to a wide variety of high-end computing applications.

## 5 Acknowledgement

The BEE2-project radio astronomy application development is in collaboration with the SETI@HOME, SERENDIP project at UC Berkeley Space Science Laboratory (Dan Werthimer), as well as UC Berkeley Radio Astronomy Laboratory (Melvyn Wright). We would also like to thank Xilinx for generous donation of FPGAs, software tools, and engineering assistance. Many thanks to all the hard work by the students, and staff members on the BEE2

team: Nan Zhou, Yury Markovskiy, Zohair Hyder, Adam Megacz, Alexander Krasnov, Hayden So, Kevin Camera, Brian Richards, and Susan Mellers.

The BEE2 project is funded by MARCO, C2S2, and MURI programs, as well as BWRC industrial sponsor companies.

## 6 Reference

- [1] <http://bwrc.eecs.berkeley.edu>
- [2] C. Chang, K. Kuusilinna, B. Richards, and R.W. Brodersen, "Implementation of BEE: a Real-time Large-scale Hardware Emulation Engine," Proc. FPGA 2003, pp. 91-99, Feb. 2003.
- [3] <http://www.xilinx.com>
- [4] K. Kuusilinna, C. Chang, etc, "Real-time System-on-Chip Emulation," Chapter 10, Winning the SoC Revolution, p. 229-253, 2003.
- [5] C. Chang, K. Kuusilinna, B. Richards, A. Chen, N. Chan, R. W. Brodersen, B. Nikolić, "Rapid Design and Analysis of Communication Systems Using the BEE Hardware Emulation Environment," Proc. IEEE Rapid System Prototyping Workshop, June 2003.
- [6] K. Kuusilinna, C. Chang, M. J. Ammer, B. Richards, and R. W. Brodersen, "Designing BEE: a Hardware Emulation Engine for Signal Processing in Low-Power Wireless Applications," EURASIP Journal on Applied Signal Processing, special issue on Rapid Prototyping of DSP Systems, 2003.
- [7] <http://www.mpi-forum.org/>
- [8] <http://seti.berkeley.edu/>
- [9] R. F. Emerson, "Biplex Pipelined FFT," NASA Code 310-40-72-04, The Deep Space Network Progress Report 42-34