PetaOp/Second FPGA Signal Processing for SETI and Radio Astronomy

Aaron Parsons[1], Donald Backer[1], Chen Chang[1], Daniel Chapman[1], Henry Chen[1],
Patrick Crescini[1], Christina de Jesus[1], Chris Dick[2], Pierre Droz[1], David MacMahon[1],
Kirsten Meder[1], Jeff Mock, Vinayak Nagpal[1], Borivoje Nikolic[1], Arash Parsa[1], Brian Richards[1],
Andrew Siemion[1], John Wawrzynek[1], Dan Werthimer[1], Melvyn Wright[1]

(Invited Paper)

Abstract—Our group, the Center for Astronomy Signal Processing and Electronics Research (CASPER), seeks to speed the development of radio astronomy signal processing instrumentation by designing and demonstrating a scalable, upgradeable, FPGA-based computing platform and software design methodology that targets a range of real-time radio telescope signal processing applications. This project relies on a small number of modular, connectible hardware components and open-source signal processing libraries which can be reused and scaled as hardware capabilities expand. We have demonstrated the use of 10 Gb Ethernet packetization and switches to manage high-bandwidth inter-board communication. Using these tools, we have built spectrometers, correlators, beamformers, VLBI data recorders, and many other applications. Future directions for the development include a fully packetized scalable correlator, additional library and toolflow development, and a next generation of modular FPGA-based hardware.

I. INTRODUCTION

THERE is a growing need for high-performance real-time Digital Signal Processing (DSP) in radio astronomy applications such as beam forming, correlation, spectroscopy, radio frequency interference (RFI) mitigation, and pulsar de-dispersion. In particular, new generations of radio telescopes (e.g. the Allen Telescope Array (ATA), the Combined Array for Research in Millimeter-wave Astronomy (CARMA), the Precision Array for Probing the Epoch of Reionization (PAPER), and the Square Kilometer Array (SKA)), employ large numbers of small antennas and are relying on DSP hardware to phase and correlate them to achieve new levels of sensitivity and resolution. These systems have computational complexity and cross-connect which scales as $O(N^2)$ with the number of antennas.

As an example of the scale of these problems, the computation requirement for correlating the full ATA bandwidth ($B = 11GHz$) for their proposed $N = 350$ dual-polarization antenna build-out, using an efficient frequency-multiply (FX) architecture, and a modest 500 KHz channel width (with number of channels $F = 2200$) is given by (1).

$$2B[Nlog_2(F)(10\ OPs) + (N(N+1)/2) \times$$
$$\times 4(8\ OPs)] = 44\ PetaOPs\ per\ second \qquad (1)$$

[1] University of California, Berkeley, Berkeley CA, 94720, USA
email: aparsons@astron.berkeley.edu
[2] Xilinx Corporation

Currently in radio astronomy, high-performance DSP instruments are highly specialized, with complex hardware being designed and built for narrowly tailored applications. Instruments take 3-10 years to design, construct, and debug, and by the time they are deployed, their technology has often been made obsolete by the Moore's Law growth of the electronics industry. This development cycle could be shortened by taking advantage of commodity hardware and interconnect, and by developing signal processing libraries which are device independent. Unfortunately, the computational requirements of many applications are far beyond the 64 to 128 MHz single-feed bandwidth capabilities of the general purpose computing clusters which have been the traditional commodity solution to radio astronomy signal processing [1].

In the last decade, however, FPGAs have come to fruition as a commodity hardware technology with the flexibility to address many different computing needs with a small number of board designs.

II. FPGAs AS COMMODITY SIGNAL PROCESSING HARDWARE

Field-Programmable Gate Arrays (FPGAs) are large-scale configurable logic devices with commercial applications that promote commodity pricing. These devices are the keystone technology for developing flexible signal processing hardware. Their reprogrammability and flexibility places them in a middle ground between custom hardware and flexible software: gateware. The data-flow processing nature of DSP algorithms matches the stream-based computation model used on FPGAs, with throughput locked to the system clock rate.

FPGA elements can provide more than 10 times the computing throughput of a DSP chip-based system with similar power consumption and cost, and more than 100 thatn of a microprocessor-based system [2], as a result of the disparity between the inherently sequential execution model of microprocessors and the spatially parallel execution model of a hardware implementation. For lower bandwidth, less computationally intensive applications, CPUs are the clear winners in terms of cost, development time, and design reusability because of the mature state of software compilers and commodity pricing. However, at moderate bandwidths and as the processing re-

quirements per channel increase, FPGAs become the clear winners in terms of cost and power consumption. Furthermore, because their simple structure scales naturally with successive generations of silicon processing technology, FPGAs are on a faster Moore's Law track than CPUs [2].

FPGAs are not the best solution for all scales of DSP applications, but they do occupy a vital middle ground between processor-scale computing and static applications with economies of scale.

III. Hardware Modules

FPGAs provides moderate scale DSP hardware with the flexibility to be widely adopted, but Moore's Law growth still dictates that hardware will need to be redesigned every few years. To minimize the effort of redesign between generations of hardware, a solution is needed which minimizes the number of hardware modules that must be redesigned and abstracts algorithms so that signal processing libraries are portable between hardware generations.

Hardware modularity suggests that boards have consistent interfaces in order to be connectible with an arbitrary number of heterogeneous components to meet the computing needs of an application. Upgrading a component should not change the way in which components are combined in the system. A modular system architecture can reduce overall cost and design time, and closely track the early adoption of state-of-the-art IC fabrication by FPGA vendors. The Berkeley Emulation Engine (BEE2) system is one of the first attempts at providing a scalable, modular solution for high-performance DSP applications [2]. Originally designed for emulating ASICs and large multiprocessor systems, the BEE2 has been conscripted for radio astronomy applications in a collaboration between the Berkeley Wireless Research Center (BWRC), the UC Berkeley Radio Astronomy Laboratory, and the UC Berkeley SETI group. Our group currently has three FPGA boards and one digitization board, which have been integrated into our toolflow.

Internet Break Out Boards (IBOBs, Fig. 1) are primarily responsible for packetizing ADC data according to the 10 Gb Ethernet protocol. Each board provides two ZDOK connectors for I/O card attachment, and two CX4 connectors which output 10 Gb Ethernet packetized data. Data packetization and serialization are performed by a Xilinx XC2VP50 FPGA which provides 232 18×18-bit multipliers, two PowerPC CPU cores, and over 53,000 logic cells. This FPGA, along with 36 Mbit of on-board ZBT SRAM, allows the IBOB to perform a significant amount of data preprocessing before handing off to the BEE2.

We designed an ADC board (Fig. 1) to mate directly to an IBOB board through ZDOK connectors for high-speed serial data I/O. Analog data is digitized using an Atmel AT84AD001B dual 8-bit ADC chip which can digitize two streams at 1 Gsample/sec or a single stream at 2 Gsample/sec. This board may be driven with either single-ended or differential inputs.

The BEE2 board (Fig. 1) is a high-end signal processing engine that integrates 500 Gops/sec of computa-
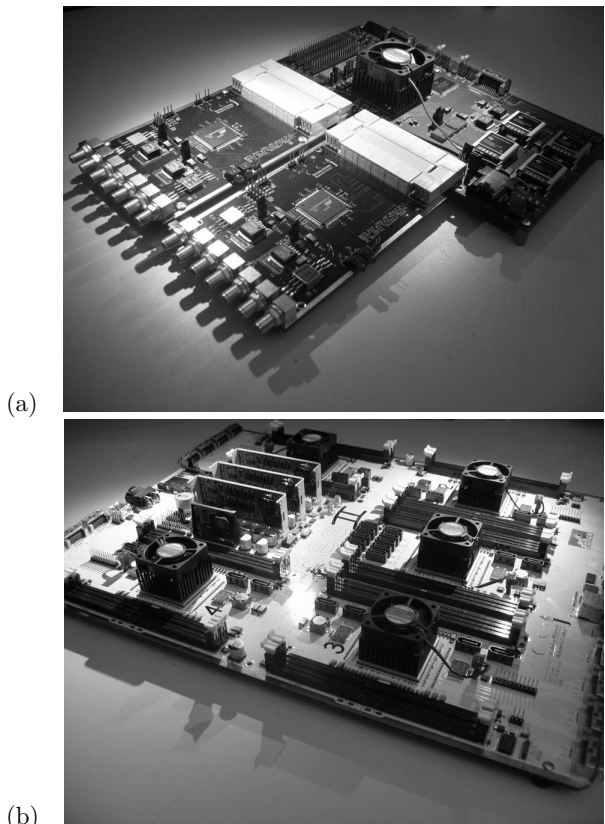

(a)

(b)

Fig. 1. (a) IBOB and ADC FPGA/digitizer modules (b) The Berkeley Emulation Engine (BEE2) FPGA board

tional power with high-speed I/Os. Its compute power is provided by its 5 Xilinx XC2VP70 Virtex-II Pro FPGAs, each containing 384 multipliers, two PowerPC CPU cores capable of running Linux, and over 74,000 configurable logic cells. Each FPGA can be connected to 4 GB of DDR2-SDRAM, and external interfaces are available through eighteen 10 Gbps Ethernet connections, as well as 100 Mbps Ethernet and RS232 serial ports.

IV. Board Interconnect and Toolflow

Hardware modularity depends on a standard communication interface between boards. For high-speed interboard communication, we implement two standard interfaces which are both compatible with CX4 hardware connectors and transmit data at 10 Gbps. The first of these interfaces, an asynchronous serial protocol called the X (ten) Attachment Unit Interface (XAUI), provides point-to-point communication for a relatively modest cost in FPGA resources. The second interface is an open-source module we developed which packetizes data according to the 10 Gb Ethernet standard. The use of a widely adopted communication standard (and the several industry experts we consulted agree that 10 GbE will soon be the most widely adopted) for inter-board communication allows the use of commercial off-the-shelf switches for

data routing, and eliminates the need for designing custom backplanes and protocols, and the associated expense in NRE. Furthermore, the flexibility of high-bandwidth packet-switched communication interfaces allows hardware components to be easily added, removed, swapped, and scaled without affecting intercommunication between other boards in the system.

The development of a full-featured DDR2 memory controller, combined with a CompactFlash-based file system has allowed us to support a standard (Debian) Linux installation on the BEE2 board. Running Linux on the FPGA CPUs makes available a plethora of standard software libraries for use on these boards. Notably, our boards can now support SSH logins, web servers, standard GCC-compiled programs, and programs written in any other programming language. Besides providing file and process management tools, Linux allows programmers to work in a standard environment using familiar tools to interact with compiled FPGA designs. Recently, we have extended this Linux framework to the IBOB using an SRAM-based memory controller and a file system on an SD-flash card.

On top of this Linux framework, we run BORPH, which extends the Linux operating system to provide kernel support for FPGA resources [3, 4]. This framework supersedes the TinySH shell and allows hardware processes on FPGA resources to be run in the same fashion as software processes on conventional processor-based systems. BORPH creates a virtual file system representing the memories and registers defined on an FPGA, allowing real-time communication with running hardware processes from Linux. This system has helped speed the verification of hardware designs as a debugging tool, and is used as a primary interface for low- to moderate-bandwidth data I/O in final systems. Because it emulates standard file-I/O interfaces for FPGA resources, BORPH is agnostic with regard to programming language, and thus allows programmers to use the programming tools with which they are most familiar.

Using Simulink and Xilinx System Generator, we abstract the physical FPGA fabric into a set of library blocks for interfacing to hardware components such as ADCs, DRAM, and other FPGAs. Additionally, we have wrapped Xilinx EDK into our toolflow for connecting registers and BRAMs on FPGA fabric to buses which are accessible by the on-chip CPU. All of these components are represented by a set of System Generator blocks, and abstract away low-level details such as pin locations, addresses, handshaking, and clock-boundaries. We can extend our toolflow to other boards by creating blocks which implement the new low-level interfaces to hardware specific to that board. These hardware interface blocks can be simple insertion points for implementing hardware interfaces, or they can implement functional models useful for verifying design behavior before it is compiled into hardware. Our toolflow ensures that a standard package of debugging tools, hardware interfaces, software drivers, and data access points are available on all compiled designs.

V. DSP Libraries

In order for the signal processing algorithms we develop to be useful in a variety of applications, it is important that they be parameterized such that they be customizable for size, behavior, and speed. This requirement adds complexity to the initial design of these libraries, but dramatically enhances their applicability and potential for longevity as hardware evolves. This design principle has an added feature that it decreases the time necessary for testing by allowing developers to debug scale models of systems which are behaviorally identical to the larger systems and are derived from the same parameterization code.

All modules in our libraries operate on a vector-warning architecture whereby a single bit signal (called a sync pulse) is passed along with a data stream, and is active the clock-cycle before the first valid data appears on that stream. This sync pulse enables library components to phase themselves correctly to the data stream and to remove any effect pipeline delays might have on downstream modules, effectively allowing modules to be swapped in to and out of designs without affecting the timing of other modules. Data samples are interpreted as 2s complement, fixed-point numbers in the range $[-1, 1)$. Modules that enable the magnitude of samples to grow (such as the FFT), have selectable down-shifting or overflow detection to prevent bit growth.

A. The FFT Library

The computational core of our FFT library is an implementation of a radix-2 biplex pipelined FFT [5] capable of analyzing two independent, complex data streams using one quarter the FPGA resources of commercial designs [6]. This architecture takes advantage of the streaming nature of ADC samples by multiplexing all of the butterfly computations of an FFT stage into a single physical butterfly core. When used to analyze two independent streams, this butterfly core outputs valid data every clock for 100% utilization efficiency.

To analyze bandwidths higher than the native clock rate of our FPGA, we developed a Wideband FFT architecture wherein biplex and parallel FFT cores are combined to create a FFT that uses no additional buffering over an equivalently sized biplex FFT core, and that still achieves 100% butterfly utilization efficiency. A 2^F channel FFT performed on time samples arriving 2^N in parallel may be decomposed into 2^N parallel biplex FFTs of length 2^{F-N} followed by a 2^N channel direct FFT that uses time-multiplexed twiddle-factor coefficients.

We have written modules for performing two real FFTs using each half of a biplex FFT. A more detailed discussion of this technique is available in Brigham, 1998 [7]. This Real FFT module has the same bandwidth flexibility as our standard parameterized Wideband FFT.

Our library combines the above cores under a unified interface which has been parameterized to implement an FFT of length 2^F with 2^N parallel time samples per clock, for arbitrary F and N. Additional parameters include pipelining controls and fixed point precision. Al-

though this library was developed primarily to be coupled with the Polyphase Filter Bank (PFB) library we discuss next, it has been used in stand-alone applications for fine-resolution spectroscopy applications such as a 128 million channel SETI spectrometer.

B. The Polyphase Filter Bank Library

The Polyphase Filter Bank (PFB) is an efficient implementation of a bank of evenly spaced, decimating digital FIR filters. The PFB algorithm decomposes these filters into a single convolution followed by a Discrete Fourier Transform [8, 9]. Since the Discrete Fourier Transform has already been highly optimized algorithmically, this results in an extremely efficient implementation. Alternatively, the PFB may be regarded as an improvement on the FFT which uses additional hardware in the form of a phased FIR front-end filter to improve the filter response of each spectral sample in the FFT. Using selectable, parameterized windowing functions, our design allows for the adjustment of the out-of-band rejection, passband ripple/roll off, and absolute width of each filter. Because of its near-ideal filter shapes, our PFB library has already seen widespread use in the radio astronomy community in applications such as 21 cm hydrogen surveys [10], pulsar surveys [1], antenna arrays [11], Very Long Baseline Interferometry (VLBI), and others.

C. The X Engine Library

In collaboration with Lynn Urry of UC Berkeley's Radio Astronomy Lab, we have developed and implemented a parameterized module for computing and accumulating baselines in an FX correlation architecture. In this correlator architecture, spectral data for each antenna is computed using our PFB design, and each of our "X Engine" modules is responsible for handling all of the baseline data for a single frequency. These dedicated-frequency modules have an advantage over dedicated-baseline modules in that they require only same number of modules as antennas (each module can handle $\frac{1}{N}$ spectral channels), and each module may be split across multiple FPGAs.

This architecture allows maximum scalability by dividing computational resources along both axes of the $N \times N$ matrix of calculations to be performed. This enables the correlation of any number of antennas and frequency channels to be mapped into modules of any size. In practice, of course, one must account for the overhead of communication interfaces between modules, which can range from the very modest (on-board, parallel communication) to the more expensive (10 Gb Ethernet packetization). This architecture also has the feature of attaining 100% multiplier efficiency by multiplexing data in each stage.

D. Other Libraries

In addition to the above-mentioned libraries, we have developed code for implementing frequency-selectable digital oscillators, decimating Finite Impulse Response (FIR) and Cascaded Integrator Comb (CIC) filters that operate on arbitrary input bandwidths, accumulating vectors in BRAM, SRAM, and DRAM, and reordering sammples into any arbitrary order using in-place buffering (single buffers the size of the vector being reordered).

VI. PROJECTS

A. A 128 Million Channel SETI Spectrometer

We have developed a SETI spectrometer for use by NASAs Jet Propulsion Laboratory. This application required the analysis of a selectable 200 MHz band at better than 2 Hz resolution for anomalous narrow-band emission. This 128 Million channel spectrometer was implemented on an IBOB module connected to one BEE2 module. This project was an important demonstrator of the design-flow and hardware connectivity which are instrumental to the rapid development of applications on our modular hardware. In the development of this spectrometer, every hardware component and interface on every board, with the exception of the ZBT SRAM on the IBOB, was tested. The reusability of the FFT and PFB libraries, which were originally developed for the SERENDIP V architecture, was also demonstrated. This instrument is currently in operation at NASAs Goldstone Deep Space Communications Complex.

B. An FX Correlator

In the first phase of correlator development for the PAPER [11] array (with applications to the ATA), we have implemented an 8 antenna correlator on 4 IBOBs and 1 BEE2 which bypassed the need for inter-IBOB spectrum synchronization by using IBOB compute resources only for digital band extraction (mixing, filtering, and decimation). PFB spectral decomposition for the antennas was moved into four FPGAs on the BEE2, and correlation was performed in the fifth. In this phase, the number of spectral channels was limited to 256 as a result of the chip resources required for buffering and accumulating the entire $N_{freq} \times N_{base} \times N_{stokes}$ complex vector of data output by the correlator. Furthermore, we limited the number of Stokes parameters calculated to to reduce by one half the number of multipliers required in the X Engines. This limitation came as a result of the architecture-simplifying decision we made of placing all correlation processing in one FPGA.

Our second stage of development is a 16 station packetized, full Stokes, FX correlator. This project will demonstrate innovative and crucial technologies needed for the next generation of radio telescopes such as the ATA, CARMA, the SKA, and the next generation PAPER array.

C. A Wide-Bandwidth Spectrometer for Space Flight

We are currently developing a digital spectrometer for NASA's MARVEL mission to Mars (Mars Volcanic Emission and Life). The bandwidth and spectral resolution of this spectrometer will also make it useful for a space-based study via microwave emission of the Earth's global warming and climate change. It will feature a 1 GHz bandwidth and 4096 spectral channels with a power consump-

tion under 10 Watts. Polyphase filter banks will ensure an out-of-band rejection of 80 dB on each channel. This spectrometer is being prototyped on an IBOB board, and will be respun into a radiation tolerant ASIC connected to a similarly hardened 2 Gsps National ADC083000 8bit ADC.

D. Other Projects

In collaboration with Harvard CFA, we have developed an SMA beamformer using the IBOB, and working with MIT Haystack, we have developed a front-end channelizer for the Mark V VLBI data recorder. The SERENDIP V board is currently being used to perform a survey of hydrogen in the galaxy at the Arecibo radio telescope. For GALFA (Galactic Arecibo L-band Feed Array), the Virtex II 6000 FPGA uses a 8192 point PFB to perform a detailed spectral analysis of a 7 MHz band centered on the 21 cm line. Simultaneously, a 256 point PFB is used on a 100 MHz band to identify large scale standing waves and remove them from the 7 MHz band used for the survey. The SERENDIP V board has also been used in projects for doing holography, correlation, front-end channelization for pulsar de-dispersion cluster computing, and Radio Frequency Interference (RFI) mitigation.

VII. FPGA-based Beowulf-like Clusters

Our proposed architecture using FPGA DSP boards that communicate via packetized data routed through commercially available switches looks like a Beowulf cluster with reconfigurable hardware modules in the place of CPUs (Fig. 2). This architecture uses a small number of easily replaceable, upgradeable hardware modules connected with as many identical modules as necessary to meet the computational requirements of an application. In addition to having the advantage of built-in tolerance for hardware failure (data can simply be routed to other, identically reconfigured modules), this architecture can use multicast switches to allow many commensal experiments to run simultaneously on the same FPGA compute cluster.

However, to realize the full potential of this architecture, support for automatically partitioning designs across different FPGAs and boards should be integrated into our toolflow. It remains to be seen whether an efficient partitioning algorithm can be developed that divides designs based on knowledge of chip resources and communication interfaces. In addition to this problem, there are questions of how to allocate and manage resources on a multi-user reprogrammable FPGA cluster that require significant further research.

Fig. 2. A Beowulf-like FPGA compute cluster for radio astronomy signal processing.

References

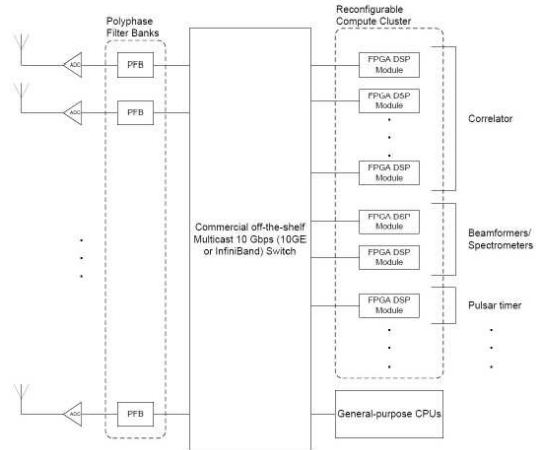[1] P. Demorest, R. Ramachandran, D. Backer, R. Ferdman, I. Stairs, and D. Nice, "Precision Pulsar Timing and Gravity Waves: Recent Advances in Instrumentation," in *Bulletin of the American Astronomical Society*, Dec. 2004, pp. 1598–+.

[2] C. Chang, J. Wawrzynek, and R. W. Brodersen, "BEE2: A High-End Reconfigurable Computing System," *IEEE Design and Test of Computers*, vol. 22, no. 2, pp. 114–125, Mar./Apr. 2005.

[3] H. So and R. Brodersen, "Improving Usability of FPGA-Based Reconfigurable Computers through Operating System Support," in *16th International Conference on Field Programmable Logic and Applications*, 2006.

[4] H. So, A. Tkachenko, and R. Brodersen, "A Unified Hardware/Software Runtime Environment for FPGA-Based Reconfigurable Computers using BORPH," *CODES+ISSS*, 2006.

[5] L. R. Rabiner and B. Gold, *Theory and application of digital signal processing*, Englewood Cliffs, N.J., Prentice-Hall, Inc., 1975. 777 p., 1975.

[6] C. Dick, "High-Performance 1024-Point Complex FFT/IFFT V2.0," Tech. Rep., 2000.

[7] E. O. Brigham, *The fast Fourier transform and its applications.*, Englewood Cliffs, N.J.: Prentice-Hall, 1988, 1988.

[8] R. Crochiere and L. R. Rabiner, *Multirate Digital Signal Processing*, Prentice Hall, 1983.

[9] P. P. Vaidyanathan, "Multirate Digital Filters, Filter Banks, Polyphase Networks, and Applications," in *IEEE*, Jan. 1990, vol. 78, pp. 56–93.

[10] C. Heiles, J. Goldston, J. Mock, A. Parsons, S. Stanimirovic, and D. Werthimer, "GALFA Hardware and Calibration Techniques," in *Bulletin of the American Astronomical Society*, Dec. 2004, pp. 1476–+.

[11] R. Bradley, D. Backer, A. Parsons, C. Parashare, and N. E. Gugliucci, "PAPER: A Precision Array to Probe the Epoch of Reionization," in *Bulletin of the American Astronomical Society*, Dec. 2005, pp. 1216–+.