Design of Optimized Heterogeneous Spectrometers for Radio Astronomy
Applications using Xilinx FPGAs

B.RAMESH NAIDU¹, N.BHOJANNA², S.KARUNAKAR REDDY³, P.SRAVANTI⁴
¹PG Scholar, Dept of VLSI System Design, Jagruti Institute of Engineering and Technology, Hyderabad, India, Email: ramesh0444@gmail.com.
²Asst Prof, Dept of ECE, Jagruti Institute of Engineering and Technology, Hyderabad, India, E-mail: bhoj401@gmail.com.
³Assoc Prof, Dept of ECE, ACE Engineering College, Hyderabad, India, E-mail: skreddy_acsec@rediffmail.com.
⁴Asst Prof, Dept of ECE, Matrusri Engineering College, India, E-mail: sravanisangnith@gmail.com.

Abstract: Radio astronomy is a non-commercial, passive user of the radio frequency spectrum. It is also the branch of fundamental scientific research that provides us with the most useful data on the origins of our universe. This paper is about a heterogeneous design, allowing us to benefit from the strengths of FPGAs. A design called the Packetized Astronomy Signal Processor, or PASP, is run on the FPGA. PASP splits up the large band into smaller bands that can be processed using off the shelf servers. The sub bands are put into packets on the FPGA and sent over a gigabit Ethernet switch to a cluster of servers. The servers receive the data from the switch and process it using spectroscopy software provided in the software package or special purpose application software written by the user and linked into the provided packet processing infrastructure. In this paper a scaled down version of spectrometers are implemented for radio astronomy applications using VHDL for Xilinx FPGA platform. The input signal is taken from Direct Digital Frequency Synthesis (DDFS) within the FPGA module. A block shall be implemented inside the FPGA to read from serial DDFS. The digitized DDFS output inside the FPGA is divided into two channels. In each channel poly-phase filter bank (PFB) shall be implemented. The output of PFB is given to 64 point FFT. Both the channels spectrum output is given to packetize. This output can be transmitted by any Ethernet device.

Keywords: DDFS, PASP, PFB, PACKETIZER.

I. INTRODUCTION
Spectrometer is a term that is applied to instruments that operate over a very wide range of wavelengths, from gamma rays and X-rays into the far infrared. So the systems which are working at the normal frequencies cannot handle all these instruments. The instruments generated with this package use a heterogeneous design, allowing us to benefit from the strengths of FPGAs. The FPGA board is able to sample and process very high bandwidths that a single CPU or GPU would not be able to manage; once the FPGA has split up the band then a platform that is easier than an FPGA to program but still provides high compute power. A design called the Packetized Astronomy Signal Processor, or PASP, is run on the FPGA. PASP splits up the large band into smaller bands that can be processed using off the shelf servers. The sub bands are put into packets on the FPGA and sent over a gigabit Ethernet switch to a cluster of servers. The servers receive the data from the switch and process it using spectroscopy software provided in the software package or special purpose application software written by the user and linked into the provided packet processing infrastructure. There are four modules in this paper namely DDFS, PFB, FFT, and PACKETIZER.

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 re-programmability and flexibility places them in a middle ground between custom hardware and flexible software: gate ware. The data-flow processing nature of DSP algorithms matches the stream-based computation model commonly used on FPGAs, with throughput locked to the system clock rate. These elements can provide over 10 times more computing throughput than a DSP-based system with similar power consumption and cost, and over 100 times that of a microprocessor-based system, as a result of the disparity between the inherently sequential execution model of microprocessors and the spatially parallel execution model of a hardware implementation.

Furthermore, because of their simple hardware structure, FPGAs scale naturally with each successive generation of silicon processing technology, resulting in the fact that FPGAs are on a faster Moore’s Law track than CPUs. Based on current projections for FPGA computing technology, the SKA computational requirement (on the order of 100 pet
operations per second) will be feasible by 2009, and implementable by 2011, with an estimated cost of $20 million USD per 800 MHz IF channel. FPGAs may be the answer for creating DSP hardware with the flexibility to be widely adopted in radio astronomy instrumentation, but Moore’s Law growth still dictates that hardware will need to be redesigned every few years. A solution is needed which minimizes the effort of redesign—a solution which minimizes the number of hardware modules which must be redesigned, and abstracts algorithms from hardware so that changing hardware affects only broad-scale implementation choices, not algorithm selection.

A. Hardware Modularity

Hardware modularity requires that a small number of components with consistent interfaces be connectible with an arbitrary number of identical components to meet the computing needs of an application ("computing by the yard"), and that upgrading/revising a component does not change the way in which components are combined in the system. Modular system architecture can provide orders of magnitude reduction in overall cost and design time, and will 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, economic solution for high-performance radio telescope DSP applications. Originally designed for high-end reconfigurable computing applications such as DSP and ASIC design, the BEE2 has been conscripted for radio astronomy applications in collaboration between the Berkeley Wireless Research Center (BWRC), the UC Berkeley Radio Astronomy Laboratory, and the UC Berkeley SETI group.

The BEE2 system consists of three hardware modules developed by graduate students Chen Chang and Pierre Droz of BWRC: the main BEE2 processing board, a high-speed ADC board for data digitization, and an IBOB for high-speed serial communication between the two boards. Communication between hardware modules takes place over standard 10 Gbit Ethernet protocol, allowing for the eventual integration of commercial switches and processors. Any of these boards may be upgraded separately to use the latest FPGA chips, and all of them may be upgraded together to take advantage of advancements in inter-board communication. These three boards may be combined to provide ample computational resources for any radio astronomy application: spectral analysis, antenna correlation, band extraction, and back-end analysis. This modular hardware platform provides astronomers with the ability to connect as many boards as necessary to meet the needs of their application.

B. Gate Ware Reusability

The advantages of flexible, upgradeable hardware architecture for radio astronomy signal processing cannot be realized without a set of reusable libraries for quickly implementing signal processing algorithms in FPGAs. These libraries and their underlying algorithms must be abstracted from the hardware involved in order to support changes and upgrades in hardware technology, and to be of independent use to the reconfigurable computing community. The viability of developing such libraries has been already demonstrated: several of the original libraries we developed were targeted for the SERENDIP V board we designed as a first implementation of a multipurpose, FPGA-based signal processing engine for radio astronomy. This board and associated libraries have proven useful in several applications for prototype antenna arrays, and others. Most importantly, the libraries developed for these applications, which include designs for Poly phase Filter Banks (PFBs), Fast Fourier Transforms (FFTs), Phase accumulators, digital mixers, digital oscillators, quadrature baseband down-converters, and FIR filters, were able to be ported to the new BEE2 architecture without modification. An important tool which has helped make it possible to write reusable gate ware libraries is the Xilinx System Generator package for the Math works Simulink language. Much as a C compiler translates platform invariant ASCII code into processor-specific byte-code, Simulink translates designs written using a standard set of FPGA components into chip-specific VHDL or Verilog that is synthesized into a final chip configuration.

Using Simulink and Xilinx System Generator, we abstract the physical FPGA fabric into a set of parameterizable library blocks for implementing signal processing algorithms, and for interfacing with abstracted hardware specific components such as ADCs, DRAM, and other FPGAs. 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; it is important that algorithms be expandable to take advantage of the inevitable increase in chip resources. This design principle has an added feature that it decreases the time necessary for testing by allowing one to debug scale models of systems which are behaviorally identical to the larger systems and are derived from the same parameterization code.

III. DESIGN IMPLEMENTATION OF HIGH LEVEL INSTRUMENT ARCHITECTURE

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 as shown in Fig.1. Data samples are interpreted as 2s complement, fixed-point numbers in the range [-1, 1) module that enable the magnitude of samples to grow (such as the FFT), have selectable down-shifting or overflow detection to prevent bit growth.

The output frequency $f_{out} = M \times (f_{clk})/2^N$
And resolution = $(f_{clk})/2^N$

**B. The Fast Fourier Transform**

The FFT is the faster version of the Discrete Fourier Transform (DFT). The FFT utilizes some clever algorithms to do the same thing as the DFT, but in less time. A DFT decomposes a sequence of values into components of different frequencies but FFT compute same results more quickly. The computational core of our FFT library is an implementation of a radix-2 biplex pipelined FFT capable of analyzing two independent, complex data streams using one quarter the FPGA resources of commercial designs. 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 $2^F$ parallel biplex FFTs of length $2^N$ 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. 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 $2^N$ in parallel per clock, for arbitrary F and N. Additional parameters include pipelining controls and fixed point precision. Although this library was developed primarily to be coupled with the Poly phase 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.

**C. The PolyPhase Filter Bank**

The Poly phase Filter Bank (PFB) is an efficient implementation of a bank of evenly spaced, decimating digital FIR filters. It means PFB used to Down converting the high frequency signals to low frequency signals. The PFB algorithm decomposes these filters into a single convolution followed by a Discrete Fourier Transform. 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, pass band 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 21cm hydrogen surveys, pulsar surveys, antenna arrays, Very Long Baseline Interferometer (VLBI), and others.

D. Packetizer
In order to dynamically balance the work distribution, the packetizer uses pull architecture: when workers are ready for further processing they ask the packetizer for a next packet. FFT outputs are given to the packetizer block. We are using 64-po int fft, so that we can get 64 sub-band values (complex) as result of fft operation. These fft outputs are given to magnitude calculator block to get the magnitudes of corresponding bands. Magnitude calculator takes the complex term (real imaginary) and calculates the maximum value from given real, imaginary values. It gives the output max+min/2 as magnitude of corresponding band.

The outputs of Magnitude calculator are given to Packetizer module, where adding of header and footer for payload (message) are done to form a packet. It adds header(24bit) at the starting of payload, payload consist a of 64 sub-band magnitudes each of 23 bit from Magnitude calculator followed by footer(24bit).

In our implementation we are using “AABBCC” as header information to indicate starting of packet and “DDEEFF” as footer information to indicate the end of packet. Before adding header and footer information it will add footer first to indicate unwanted information. This architecture allows maximum scalability by dividing computational resources along both axes of the N × 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.

E. 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 samples into any arbitrary order using in-place buffering (single buffers the size of the vector being reordered).

IV. EXPERIMENTAL RESULTS
This paper consists of all the software and hardware results observed in the experiment. The results include snapshots of each and every module individually with all the inputs, outputs and intermediate waveforms.

A. Simulation Results

- It consists of clock signal indicated by clk_8x which is 10 MHz as shown in fig.3.
- The clock signal clk_1x represents decimated clock which is (10/8)MHz.
- The state counter is common counter for all operations.
- The test_sig is combination of two frequency signals at 6th bin and 8th bin.
- The polyphase_coeff_val signal is filter coefficient 512 each of 10 bit.
- The coeff_signal_val is the product signal of test_sig and polyphase_coeff_val.

BRAM outputs and FFT inputs:
- There are two clock signals for bram1 and bram2 when the clock is low one the bram sends the signal.
- Bram1_portb_data and bram2_portb_data signals send data to fft.
- Selected_fft_in_full is combined signal of bram1 and bram2 at low clock as shown in fig.4.
Design of Optimized Heterogeneous Spectrometers for Radio Astronomy Applications using Xilinx FPGAs

Spectrum Analysis:
- The mag- signal gives the magnitude of 64 sub bands where peaks appear at 6th, 8th bin and (64-6)th, (64-8)th bins due to symmetry as shown in fig.5.
- Since the input frequencies are given at that sub bands.

Packet Formation:
- The signal sig_packeted_fft_data is the packet formatted output data as shown in fig.6.
- With header (AABBCC) and footer (DDEEFF) information appended.
- The information bits are located in each of 23 bit in the 64 bins.

B. Chip Scope Results

Input Signal Bus Plot:
- Input signal bus plot for spectrum analysis as shown in fig.7.
- The signal is a cosine signal which is a combination of two input frequency signals with desired peaks at 6th and 8th bins respectively.
- Due to symmetry of cosine signal the peaks will also appear at 58th and 56th bins.
- The signal test_signal indicates the input bus plot which is marked.

FFT Bus Plot:
- The combined signal of two dual ports BRAM IP cores at low clock signal.
- The signal is indicated by fft_in in the bus plot as shown in fig.8.
Spectrum Analysis Bus Plot:
- The output signal which has two peaks at given input frequencies 6th and 8th bin appear at the output.
- Due to symmetry the peaks also appear at 56th and 58th bin respectively.
- The signal is indicated by fft_out_mag in the bus plot as shown in fig.9.

Fig.9. Spectrum.

Packet Formation:
- The signal output data is the packet formatted output data.
- The header (AABBCC) and footer (DDEEFF) information appended with every 64 bins of information data as shown in fig.10.

Fig.10. Packet data.

V. CONCLUSION AND FUTURE SCOPE

In this paper, we describe a radio astronomy instrument that is easily reconfigured to suit a variety of applications. This style of instrument design can be extended to a heterogeneous cluster running multiple processing algorithms at the same time. Many algorithms require the data to be broken up into sub bands before it can be processed by the server which can be done on the same FPGA. Using multicast packets, multiple servers can subscribe to the same sub bands generated on by PASP and process them in different ways. This style of instrument design greatly accelerates time to science for many projects. Separating the implementation of the instrument from the hardware specification has created a design that works well for a variety of computational resources and applications. As resources improve, the instrument can improve along with them, providing the opportunity to do new science that wasn’t possible before.

VI. REFERENCES