HW/SW Co-design for Reconfigurable Ultrasonic System-on-Chip Platform

Pramod Govindan, Spenser Gilliland, Thomas Gonnot and Jafar Saniie
Department of Electrical and Computer Engineering
Illinois Institute of Technology,
Chicago, Illinois 60616

***Abstract*- Efficient partitioning of hardware and software is essential for an optimized system design with a reduced development time. A reconfigurable Ultrasonic System-on-Chip Hardware (RUSH) platform has been developed to allow flexibility in system development for real-time ultrasonic signal processing applications via hardware-software (HW/SW) co-design. In this paper, we analyze how the various components within the RUSH system are split into hardware and software. Hence RUSH provides an industry standard development platform for both hardware and software designers.**

I. INTRODUCTION

The Reconfigurable Ultrasonic System-on-Chip Hardware (RUSH) platform consists of a Xilinx Virtex-5 (XC5VLX110T) FPGA which embeds a Microblaze processor within the FPGA firmware. This enables RUSH to support hardware-software (HW/SW) co-design. The RUSH platform helps the ultrasonic research community to conduct real-time signal processing experiments for non-destructive evaluation (NDE) and imaging applications. The platform includes a 12-bit ADC (Maxim MAX1215N 2012) which captures ultrasonic signals at a sampling rate up to 250 MSPS. The reconfigurable hardware within the FPGA is used to implement computationally-intensive signal processing algorithms. The embedded processor is capable of executing platform independent C-code. This provides the flexibility to reuse a substantial portion of available software, which in turn reduces the system development time. Reuse of hardware is achieved by integrating IP cores with the processor through the use of common bus architecture. Fig.1 shows the ultrasonic test setup for NDE and imaging applications using the RUSH platform.

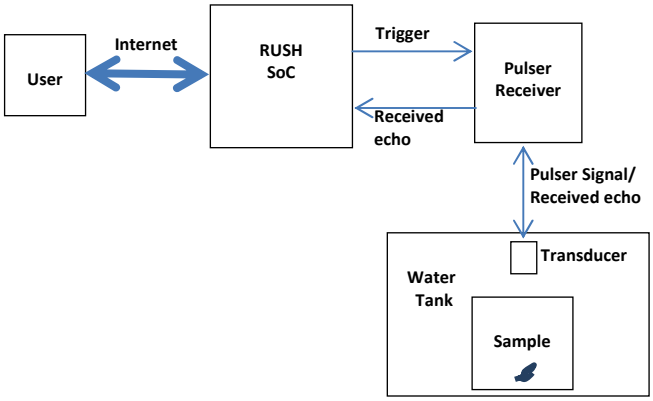

Fig. 1 Ultrasound testing setup using RUSH platform

Apart from the FPGA, RUSH integrates a Maxim MAX1215N ADC Evaluation Kit (Maxim1215N 2012), a Maxim MAX5874 DAC Evaluation Kit (Maxim5874 2012), and two Maxim MAX1536 Power Supply Evaluation Kits.

The FPGA System-on-Chip (SoC) implements the following Xilinx cores: A Multi-Port Memory Controller (MPMC), Universal Asynchronous Receiver Transmitter (UART), gigabit Ethernet, System Advanced Configuration Environment (Sys ACE) compact flash controller, timer, clock generators and a Microblaze processor.

Some of the ultrasonic signal processing applications implemented on RUSH are real-time ultrasonic flaw detection with a high repetition rate using split-spectrum processing, real-time embedded implementation of chirplet signal decomposition and real-time coherent averaging for improving signal-to-noise ratio (SNR). Split spectrum processing was implemented as a HW/SW co-design having parallel & pipelined multiplications and additions for high throughput.

This paper is organized as follows. Section II provides the HW/SW co-design description of the RUSH platform. Section III explains the ADC design strategies of the RUSH platform. Section IV presents hardware architecture details of the RUSH platform. Section V provides software details of RUSH platform. Section VI concludes this paper.

II. HW/SW CO-DESIGN

By using the RUSH platform, the user can explore the full design space including software only, hardware only and hardware/software co-design. It provides a common platform for both hardware and software designers. The system is developed by using the IP Core system in the Xilinx Embedded Development Kit (EDK). The EDK system implements a common bus based on the IBM Processor Local Bus (PLB) architecture. The PLB connects the various IP cores within the FPGA SoC. The ability to run Linux provides a common industry standard Application Programming Interface (API) to software designers.

The system can be optimized by partitioning the design into hardware and software in different ways. For example, we can have software implementation for an algorithm on Microblaze processor, use hardware accelerator for functions such as FFT and iFFT, and use software look-up tables for trigonometric and exponential functions. The hardware-software partitioning within the FPGA of the RUSH system is shown in Fig. 2. The signal processing algorithms are implemented on FPGA. The FPGA system is

978-1-4673-5208-6/13/$31.00 ©2013 IEEE

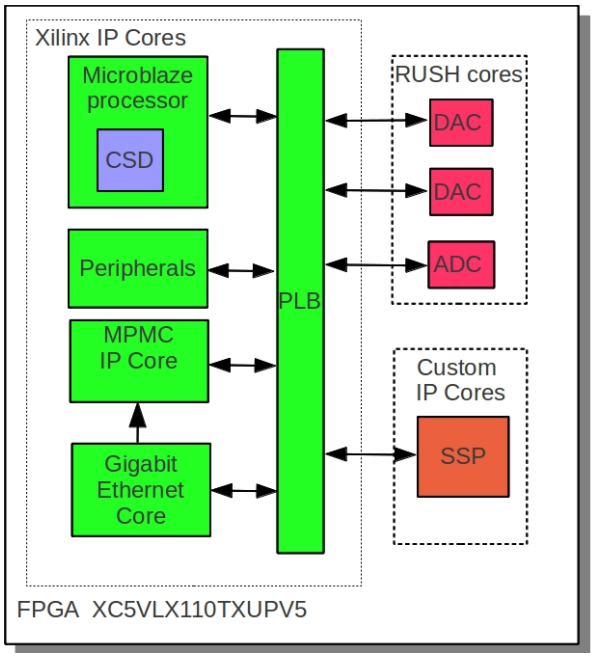

Fig. 2 HW/SW partitioning on FPGA within RUSH platform

divided into 3 sections - Xilinx IP cores, custom IP cores and RUSH cores. The Xilinx IP cores are the existing IP cores provided by Xilinx which includes the Microblaze processor, MPMC core, gigabit Ethernet core and peripherals (for e.g., UART, timer, Sys ACE compact flash controller). Custom IP cores are the hardware components developed using Xilinx IP cores like the FFT module and Xilinx primitives such as DSP48 modules, LUTs and BRAMs. The Split Spectrum Processing (SSP) core is an example of a custom IP core. RUSH cores are those developed completely by the RUSH designers by using hardware description languages and corresponding software. Here the RUSH cores (DAC and ADC) are developed using VHDL. Chirplet Signal Decomposition (CSD) is completely implemented in software on the Microblaze processor.

The MPMC provided by Xilinx (Xilinx 2011) is the bridge which connects DDR2 memory. The gigabit Ethernet controller connects to the processor using the PLB interface and an SDMA (Soft Direct Memory Access) PIM (Personality Interface Modules) of the MPMC. The RUSH system communicates with the user using TCP/IP on top of gigabit Ethernet. The RUSH system provides wired gigabit Ethernet communication with a theoretical maximum of 118.75 MB/s which is very close to the line rate of 125 MB/s.

III. ADC DESIGN STRATEGIES

The ultrasonic signal operating range is from 20 KHz to 20 MHz. The sampling rate of 250 MSPS allows enough bandwidth to capture all the received ultrasonic signals. The maximum throughput of the ADC is 375 MB/s (assuming a 250 MHz clock). The ADC uses LVDS (low voltage differential signaling) for interfacing to the FPGA. The

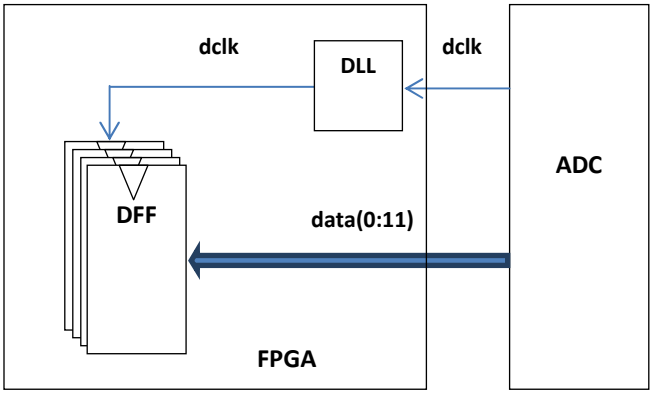

Fig. 3 ADC – FPGA interconnection

ADC delay and the wire length decide the maximum operating frequency. The ADC edge rate is significant with respect to the length of wire, so the line is treated as a transmission line. The transmission line has a delay of 170 ps/inch according to the measurement obtained in this study. The clocking delay has been compensated by using a recovered clock in addition to a separate clock domain operating within the FPGA. With this arrangement a sampling rate of 200 MSPS has been achieved.

Several options were analyzed to choose the interconnection between the ADC and FPGA, due to multiple issues. The connections between the ADC and the FPGA are located across multiple FPGA input/output banks; which can cause non-uniform delays for each input. So, the clock-skew and clock delay has to be controlled properly. The Virtex-5 device has a dedicated on-chip Digital Locked Loop (DLL) circuit which provides zero propagation delay and low clock skew between clock signals distributed throughout the FPGA. By monitoring a sample of the DLL output clock, the DLL can compensate for the delay on the routing network, effectively eliminating the delay from the external input port to the individual clock loads within the FPGA. DLL is used for the incoming clock signal as shown in Fig. 3. In this case, the skew present at the first D-type Flip Flop (DFF) is eliminated by using the DLL. This ensured proper operation of the interface.

For the implementation of coherent averaging, a preprocessing block is designed which provides for precise synchronization between the ultrasonic pulser and data acquisition capture logic. The preprocessing block in the ADC within the FPGA as shown in Fig. 4 provides a highly accurate synchronization between the data acquisition capture clock and the excitation trigger pulse applied to the ultrasonic transducer. The configuration registers programmed by the user via PLB interface are synchronized to the ADC clock domain by a synchronizing circuit. Thus the pulse generator, capture logic and the preprocessing logic will receive the synchronized versions of the configuration register values.

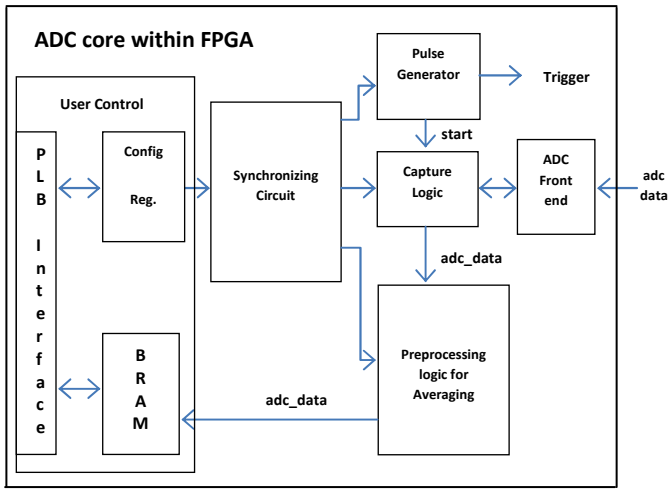

Fig. 4 Synchronization in the ADC core within FPGA

In order to meet timing requirements, the coherent averaging block is parallelized by a factor of four as represented in Fig. 5. The concurrent nature of this design helps to reliably meet timing requirement of the system. The coherent averaging block uses the block RAMs (BRAM) which is faster than the distributed RAM. In Xilinx FPGAs, BRAMs are faster but have a one cycle delay for access. It takes 2 cycles for the accumulation. The first cycle is a read cycle and the second is a write cycle. During the read cycle, the BRAM exports the value to a register. During the write cycle, the incoming data is added to the value in the register and stored back in the BRAM.

IV. SIGNAL PROCESSING HARDWARE ARCHITECTURE

SSP algorithm is implemented inside the FPGA fabric. The SSP hardware has three major modules - FFT followed by a basic one-zero windowing algorithm to split the spectrum, inverse FFTs (iFFT) to obtain frequency-diverse signals and pass-through absolute minimum post-processing

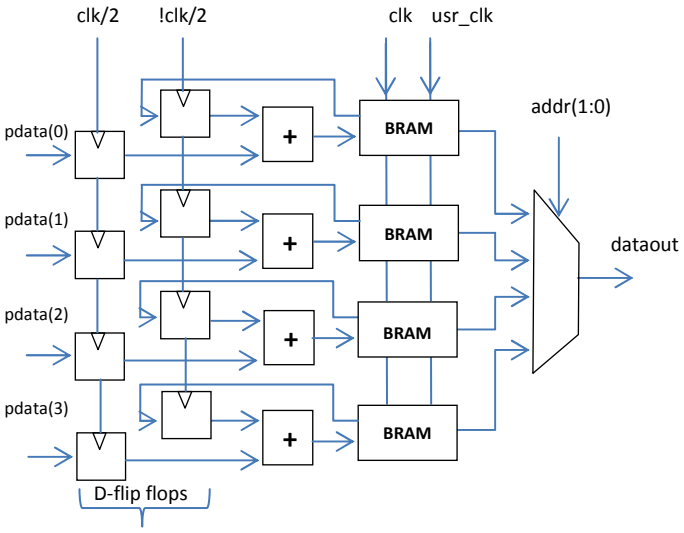

Fig. 5 ADC coherent averaging block

to improve the visibility of target echo in the presence of clutter. The pass-through absolute minimization block is used to reduce the noise while minimizing the impact on the signal of interest. Fig. 6 shows the binary decision tree used in the post-processing block. Each processing element (PE) performs a binary decision on which of the two incoming values is the absolute minimum. Then, it passes the value to the output. The post-processing block is generic, and can handle any number of input channels. Each PE implements a pass-through absolute minimization.

The number of channels and the maximum number of samples can be modified in the EDK during synthesis. The SSP algorithm implementation can use up to twelve channels and 4096 samples and is limited by the number of DSP48 resources available on the FPGA. DSP48 blocks inside the FPGA provide constant multiply and accumulate (MAC) as well as two-operand multiply and division capabilities. The main components used in the SSP algorithm are FFT IP cores. For the SSP processing, the number of FFT IP cores required is equal to the number of channels for iFFT and one additional core for FFT. The major challenge in this implementation was to optimize the area of iFFT implementation. The FFT/iFFT core (Xilinx Radix 2-Lite FFT IP Core - FFT 2012) provided by Xilinx is used here because it is one of the most resource efficient FFT implementation. The latency of this IP core at a transform size of 4096 points is 57,539 cycles; and it uses two DSP48 components, seven BRAMs and approximately 250 slices.

There are control registers to store information like the number of samples to use, the start frequency, bandwidth and the step between the bands of the SSP algorithm. As shown in Fig. 7, the dual port memory is accessible by the SSP algorithm block and also by the processor via PLB. The processor stores the samples into the memory via PLB interface. These samples are loaded into the SSP algorithm block. After processing, the results are stored back to the memory. A data valid signal (ssp_data_out_valid in Fig. 7) is used to select between reading from and writing to the memory by SSP block.

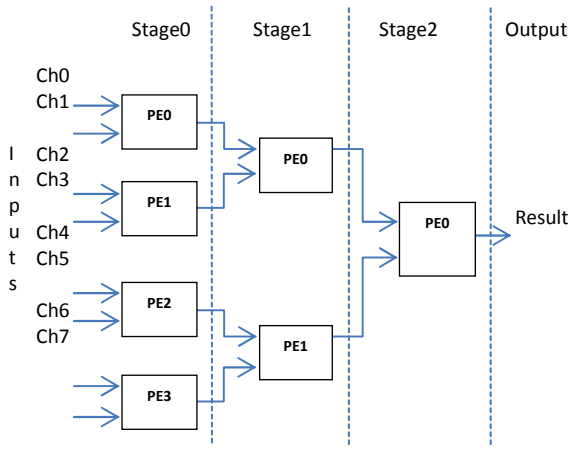

Fig. 6 SSP pass-through absolute minimization post processing

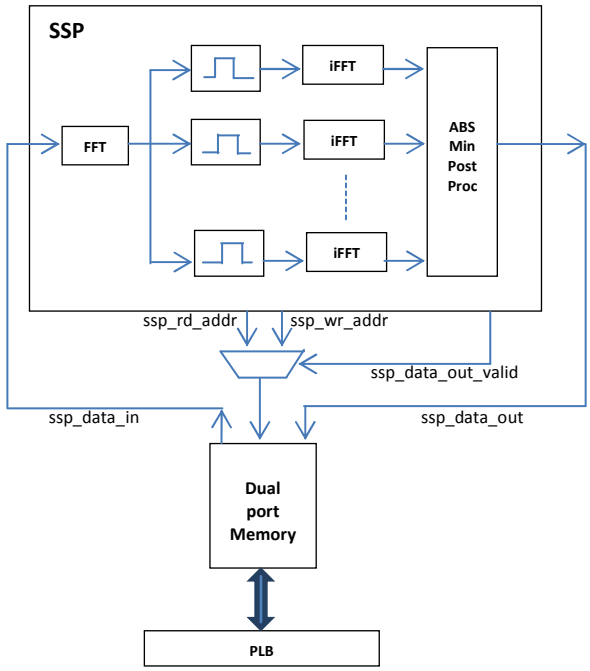

Fig. 7 SSP implementation within RUSH platform

V. SOFTWARE ARCHITECTURE

The Microblaze processor is a 32 bit RISC CPU which can be customized to include an FPU (Floating-Point Unit) and streaming interface to either fast simplex link or AXI (Advanced eXtensible Interface) stream interfaces. To balance the resource usage and the performance, 12 KB data and 4KB instruction caches were chosen. The cache is backed by the MPMC which interfaces to the 256 MB DDR2 memory.

Chirplet Signal Decomposition algorithm is implemented completely in software. The Microblaze processor executes this algorithm. The basis for the implementation is the algorithm discussed in [12]; however, the algorithm has been optimized by using pre-computation and estimation methods. Additionally, the algorithm has been altered to provide deterministic behavior. To produce a more deterministic algorithm, the estimation of time-of-arrival and center frequency of the signal are calculated by a finite step process as opposed to the iterative process mentioned in the FCT algorithm flow chart in [12].

The main performance impediment for CSD is the correlation operations. These operations require regeneration of chirplets multiple times with different parameters. This time-consuming iterative process can be avoided by pre-computing a table of these parameter values and utilizing this table to perform the parameter estimation operation. This approach reduces the computation time significantly. Furthermore, by using lookup tables for exponentiation and trigonometric functions like cosine, sine and tangent, the total execution time was substantially reduced with only a slight reduction in accuracy.

VI. CONCLUSION

The Xilinx Virtex-5 FPGA and the embedded microprocessor within the FPGA are the major components supporting Hardware-Software (HW/SW) co-design in the RUSH platform. Co-design brings in several challenges in system partitioning. RUSH being a reconfigurable platform provides sufficient flexibility to both the hardware and software designers in developing an efficient system within a short period. The modular design structure of this platform supports to upgrade the system by replacing the existing FPGA with latest high-performance FPGA devices as they become available.

REFERENCES

[1] J. Saniie and E. Oruklu, “Introduction to Special Issue on Novel Embedded Systems for Ultrasonic Imaging and Signal Processing”, *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 59, no. 7, pp. 1329-1332, July 2012 (invited paper).

[2] J. Saniie, E. Oruklu and S. Yoon, “System-on-Chip Design for Ultrasonic Target Detection Using Split-Spectrum Processing and Neural Networks”, *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control*, vol. 59, no. 7, pp. 1354-1369, July 2012.

[3] J. Saniie and D. T. Nagle, “Analysis of Order-Statistic CFAR Threshold Estimators for Improved Ultrasonic Flaw Detection”, *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control*, 39(5), pp. 618-630, 1992.

[4] J. Saniie, K. D. Donohue, D. T. Nagle and N. M. Bilgutay, “Frequency Diversity Ultrasonic Flaw Detection using Order Statistic Filters”, *Proceedings of IEEE Ultrasonics Symposium*, pp. 879-884 vol.2, 1988.

[5] J. Saniie, D. T. Nagle and K. D. Donohue, “Analysis of Order Statistic Filters Applied to Ultrasonic Flaw Detection using Split-Spectrum Processing”, *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control*, 38(2), pp. 133-140, 1991.

[6] J. Saniie, T. Wang and X. Jin, “Performance Evaluation of Frequency Diverse Bayesian Ultrasonic Flaw Detection”, *The Journal of the Acoustical Society of America*, 91, pp. 2034-2041, 1992.

[7] Y. Lu, E. Oruklu and J. Saniie, “Fast Chirplet Transform With FPGA-Based Implementation”, *IEEE Signal Processing Letters*, 15, pp. 577-580, 2008.

[8] Xilinx, 2012, Platform Specification Format Reference Manual v13.4, http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_4/psf_rm.pdf

[9] R. Demirli and J. Saniie, "Model-based estimation of ultrasonic echoes, part I: analysis and algorithms ", *IEEE Transactions on Ultrasonics Ferroelectrics and Frequency Control*, vol. 48, no. 3, pp. 787-802, May 2001.

[10] R. Demirli and J. Saniie, "Model-based estimation of ultrasonic echoes, part II: analysis and algorithms ", *IEEE Transactions on Ultrasonics Ferroelectrics and Frequency Control*, vol. 48, no. 3, pp. 803-811, May 2001.

[11] Y. Li, H. Fu and P. Kam, " Improved, Approximate, Time-Domain ML estimators of chirp signal parameters and their performance analysis ", *IEEE Transactions on Signal Processing*, vol. 57, no. 4, pp. 1260-1272, April 2009.

[12] Y. Lu, R. Demirli, G. Cardoso, and J. Saniie, “A successive parameter estimation algorithm for chirplet signal decomposition,” *IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control*, pp. 2121–2131, vol. 53, November 2006.