International Journal of Electronics Communication and Computer Engineering
Volume 3, Issue 1, ISSN 2249 –071X

Hardware Software Co-design for Low Cost Embedded Systems Based on FPGA

Sandeepkumar R. Pandey, Dr.R.S Pande, Prof.P.A Dwaramwar, J.B. Zalke,
Department of Electronics Engineering, RCOEM, Nagpur (India).
{pandeys, panders, dwaramwarpa, zalkej}@rknec.edu

***Abstract* — The fundamental building blocks of a SoC (System on Chip) are its intellectual property (IP) cores, which are reusable hardware blocks designed to perform a particular task. For realizing SoC application, IP used may be customized to save silicon Area. With the abundance of proprietary and free IPs, and customized software, significant improvement in the overall system performance can be achieved. Low cost FPGAs, such as Spartan family from Xilinx, can be used effectively to demonstrate power and purpose of hardware software co-designed systems. In this paper, we establish, from the experimental results, that at the cost of small silicon area, one can get significant improvement in speed, by properly conceiving Hardware Software (HW/SW) Co-design. Entire System is implemented using Microblaze soft-core from Xilinx and tested on Spartan 3E Starter kit with Xilinx EDK (Embedded Development Kit).**

***Key Words* — FPGA, SoC, intellectual property (IP) cores, hardware software co-design**

I. Introduction

Ever-increasing embedded system design complexity combined with reduced time-to-market window has revolutionized the embedded system design process. The traditional design techniques (independent hardware and software design) are now being challenged when heterogeneous models and applications are getting integrated to create a complex system on chip. In hardware-software co-design, designers consider trade-off in the way hardware and software components of a system work together to exhibit a specified behavior, given a set of performance goals and technology.

II. FPGA Structure & Soft Core

A typical modern FPGA (Fig. 1) provides the designer with programmable logic blocks that contain the pool of combinatorial blocks and flip-flops to be used in the design.

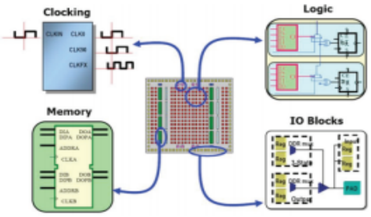


Fig.1 Internal structure of a generic FPGA (courtesy Xilinx, Inc.)

In addition, vendors acknowledge the fact that logic is often used in conjunction with memory, and typically include variable amounts of static Random Access Memory (RAM) inside their chips. Clock conditioning has also become commonplace, and support in the form of Delay Locked Loops (DLLs) and Phase Locked Loops (PLLs) is also provided inside the same silicon chip.

With such a huge computational competency provided by FPGAs, coupled with large number free/proprietary soft IP cores available, realization of SoCs have been feasible. The hardware/software co-design community challenges and opportunities in designing high performance computing (HPC) systems are explored [1]

The MicroBlaze embedded processor soft core is a reduced instruction set computer (RISC) optimized for implementation in Xilinx Field Programmable Gate Arrays (FPGAs). Fig. 2 shows a functional block diagram of the MicroBlaze core. Basic architecture is fixed while the optional blocks may/may-not be used [2].

III. Hardware-Software Co-design & System Architecture

Fig. 3 shows how a Finite Impulse Response (FIR) filter could be implemented in two platforms. While the DSP needs large number of clock ticks to calculate an output sample, the FPGA generates a new sample for every clock cycle. Even if DSP chips can be clocked faster than FPGAs, the difference is in no case larger than a factor of 10. If one adds that many such filters can exist concurrently and interact inside the same FPGA, it is easy to see that DSPs are no match for FPGAs in high performance signal processing applications [3]. Cost paid for inherent parallelism in FPGA is silicon area.

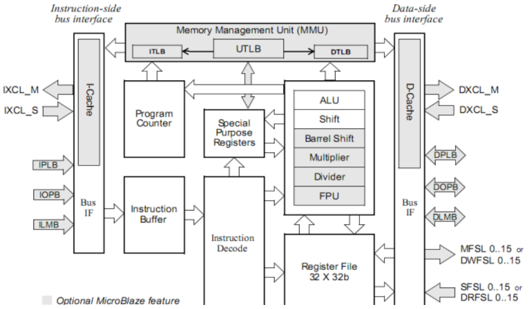


Fig.2 Microblaze Core Block Diagram

Fig.4 is the complete system architecture that is proposed for this work. The fixed feature set of the processor(Microblaze) includes:

- Thirty-two 32-bit general purpose registers
- 32-bit instruction word with three operands and two addressing modes
- 32-bit address bus
- Single issue pipeline

MicroBlaze uses Big-Endian bit-reversed format to represent data. The hardware supported data types for MicroBlaze are word, half word, and byte.

All MicroBlaze instructions are 32 bits and are defined as either Type A or Type B. Type A instructions have up to two source register operands and one destination register operand. Type B instructions have one source register and a 16-bit immediate operand. Type B instructions have a single destination register operand. Instructions are provided in the following functional categories: arithmetic, logical, branch, load/store, and special. Refer to [2] for more information on these instructions.

The MicroBlaze core is organized as Harvard architecture with separate bus interface units for data and instruction accesses. The following three memory interfaces are supported: Local Memory Bus (LMB), the IBM Processor Local Bus (PLB) or the IBM On-chip Peripheral Bus (OPB).

The MicroBlaze PLB interfaces are implemented as byte-enable capable 32-bit masters. The MicroBlaze OPB interfaces are implemented as byte-enable capable masters. The LMB is a synchronous bus used primarily to access on-chip block RAM. It uses a minimum number of control signals and a simple protocol to ensure that local block RAM are accessed in a single clock cycle[5].

The MicroBlaze floating point unit is based on the IEEE 754 standard:

- Uses IEEE 754 single precision floating point format, including definitions for infinity, not-a-number (NaN), and zero.
- Supports addition, subtraction, multiplication, division, comparison, conversion and square root instructions
- Implements round-to-nearest mode
- Generates sticky status bits for: underflow, overflow, and invalid operation.

The Timer/Counter is organized as two identical timer modules as shown in Fig.6. Each timer module has an associated load register that is used to hold either the initial value for the counter for event generation, or a capture value, depending on the mode of the timer.

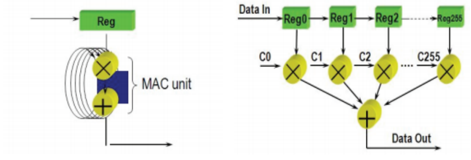


Fig. 3 FIR filter comparison between DSP and FPGA

There are three modes that can be used with the two Timer/Counter modules:

- Generate mode
- Capture mode
- Pulse Width Modulation (PWM) mode.

In the Generate mode, the value in the load register is loaded into the counter. The counter, when enabled, begins to count up or down, depending on the control word in the Timer Control Status Register (TCSR) [7] can be referred for complete description of the Timer modes.

IV. Experimental Setup

Fig.5 is the Embedded Development Kit (EDK) Tools Design Flow [5]. The tools provided with EDK are used for embedded design process, as illustrated in Fig.5

Hardware Development: The hardware platform consists of one or more processors and peripherals connected to the processor buses. EDK captures the hardware platform in the Microprocessor Hardware Specification (MHS) file.

Software Development: A software platform is a collection of software drivers and, optionally, the operating system on which to build any application. The software image created consists only of the portions of the Xilinx library used in embedded design. EDK captures the software platform in the Microprocessor Software Specification (MSS) file.

To verify the correct functionality of hardware platform, a simulation model is created and run on a Hardware Design Language (HDL) simulator.

Two variants of the Microblaze soft core are implemented using Xilinx EDK. First implementation (Fig.4) is the pure software implementation of following equation (1) without FPU (Floating Point Unit).

$$y = \sum_{n=0}^{N-1} \mathrm{c[n].\,x[n]} \quad \text{……….} \ (1)$$

In second implementation (Fig.4), the dedicated FPU (Floating Point Unit) along with Microblaze is put such that same computation are done and no modification in the software written to compute Eq. (1).

Following is the Snippet of C (only dummy shown here) code written in SDK.

```
void main()

{               int i,j;
                float sum;
                unsigned long int val=0;
                xil_printf("Input signal is:\n");
                print_signal(x);
                xil_printf("timer is starting");
                TLR0=0x000000;
                TCSR0=0x0020;
                TCSR0=0x0080;
                for(i=0;i<=M-1;i++)
{

for(j=0;j<=M-1;j++)

{              --statements for computation for eq.(1)
                --exit after computation & send timer count to -------
                ----PC through UART
```

The time required to do the computation is calculated by first implementation (Pure Software) and then for Hardware software co-designed system i.e. second implementation with FPU. Numbers of clock cycles required for computation are displayed on the hyper-terminal of PC for both the case.

V. Result & Conclusion

Form a sample run of the complete system with reference clock frequency 50MHz; results shown in Fig.8 are obtained. Same is tabulated in Table I.

Device Utilization summary shows: (for Implementation2)

Total Number Slice Registers	*:*	*2701 out of 9312*	*29%*
Number of RAMB16s	*:*	*11 out of 20*	*55%*
Number of BUFGMUXs	*:*	*5 out of 24*	*20%*
Number of DCMs	*:*	*2 out of 4*	*50%*
Number of BSCANs	*:*	*1 out of 1*	*100%*

Fig.7 Shows the complete floor-planned design.

In this EDK-based HW/SW design, the comparison results have shown significant improvement of speed (26 times) for HW/SW co-design system at the expense of a little hardware (1.27 times).

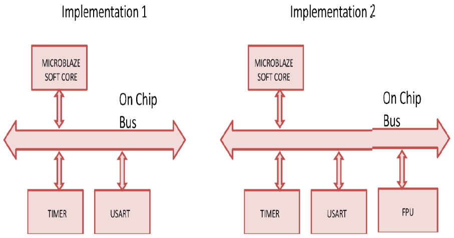


Fig.4 Microblaze based system implementation.

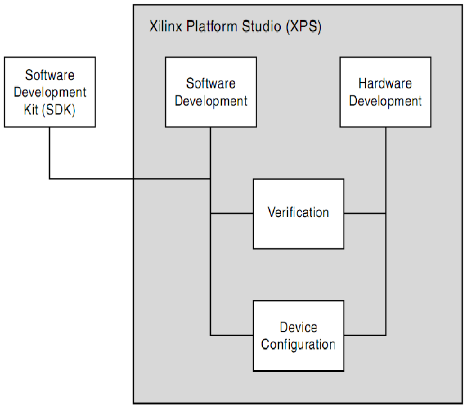


Fig.5 Basic Embedded Design Process Flow.

A benchmark of the automatic personal recognition systems available today points that most of them rely on purely software solutions running under personal computers or microprocessor platforms [4]. However, this architecture can result inappropriate when the complexity of the application increases and real-time characteristics are required for the functionality. The advances made in VLSI offer the hardware-software co-design methodology as a challenging alternative solution.

The introduction of the field programmable logic devices into the system allows the partitioning of any application into hardware and software tasks: those complex and time-consuming computational tasks can be accelerated by synthesizing them into the hardware core blocks (FPGA), while the rest of less computationally expensive tasks can be kept under the control of the software block (Microblaze Soft Core). Therefore, a solution based on hardware-software co-design is proposed in this work and significant improvement is demonstrated.

The right HW/SW partitioning of the application allows the efficient implementation of the system, from which it is possible to reach the timing requirements for critical applications.

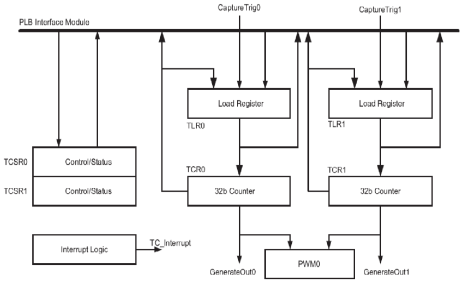
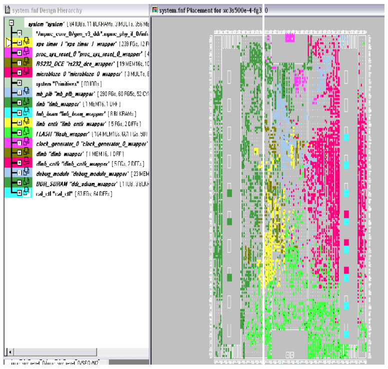


Fig.6 XPS Timer/Counter Detailed Block Diagram.


system.fnf Design Hierarchy
system.fnf Placement for xc3s500e-4-fg3[illegible]0

system "system" [84 IOBs, 11 BLKRAMs, 3 MULTs, 356 ME
/mpmc_core_0/gen_s3_ddr.mpmc_phy_if_0/infi
xps timer 1 "xps timer 1 wrapper" [239 FGs, 12 F
proc_sys_reset_0 "proc_sys_reset_0_wrapper" [4
RS232_DCE "rs232_dce_wrapper" [19 MEM16s, 10
microblaze 0 "microblaze 0 wrapper" [3 MULTs, 8
system "Primitives" [00 IOBs]
mb_plb "mb_plb_wrapper" [290 FGs, 60 FG5s, 52 CY
ilmb "ilmb_wrapper" [1 MEM16, 1 DFF]
lmb_bram "lmb_bram_wrapper" [8 BLKRAMs]
ilmb cntlr "ilmb cntlr wrapper" [5 FGs, 2 DFFs]
FLASH "flash_wrapper" [104 MEM16s, 601 FGs, 501
clock_generator_0 "clock_generator_0_wrapper"
dlmb "dlmb wrapper" [1 MEM16, 1 DFF]
dlmb_cntlr "dlmb_cntlr_wrapper" [5 FGs, 2 DFFs]
debug_module "debug_module_wrapper" [23 MEM
DDR_SDRAM "ddr_sdram_wrapper" [1 IOB, 3 BLK
cal_ctl "cal_ctl" [63 FGs, 64 DFFs]
proc_sys_reset_0/proc_sys_reset_0/SEQ/N2

Fig.7 Floor-Plan of Complete System.

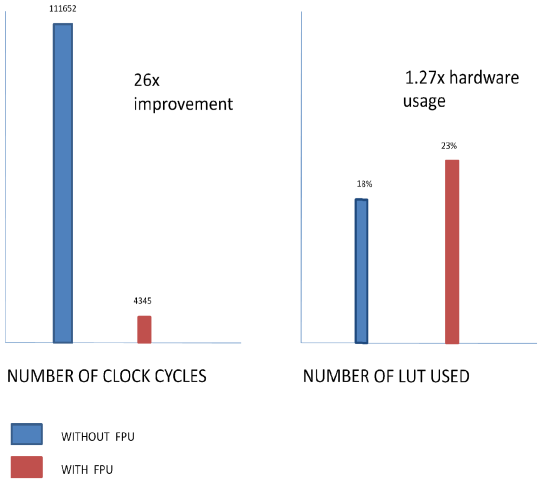


Fig.8 First Implementation Vs. Second Implementation. (Result showing 26 times improvement in computation time and 1.27 times more hardware requirement for hardware software co-designed system)

Sr. No.	System implementation	Number of LUT used for Spartan-3E FPGA.(%)	Number of Clock cycles required for computation	Reference clock frequency
1	With FPU	29 %	4345	50MHz
2	Without FPU	18 %	111652	50MHz

Table. I Results obtained from experimental setup.

REFERENCES

[1] Hu, X.S.; Murphy, R.C.; Dosanjh, S.; Olukotun, K.; Poole, S.; “Hardware/software co-design for high performance computing: Challenges and opportunities”, Hardware/Software Codesign and System Synthesis (CODES+ISSS), 2010 IEEE/ACM/IFIP.

[2] “MicroBlaze Processor Reference Guide, Embedded Development Kit EDK 10.1i”, UG081 (v9.0), Xilinx Inc.

[3] Uwe Meyer-Baese, Digital Signal Processing with Field Programmable Gate Arrays, 3rd ed. (Springer, 2007).

[4] D. Maio, D. Maltoni, R. Cappelli, J. L. Wayman, A. K. Jain, “FVC2004: Third Fingerprint Verification Competition”, Proceedings of ICBA 2004, LNCS 3072, pp. 1-7

[5] “Embedded System Tools Reference Manual, Embedded Development Kit, EDK 10.1”, Xilinx Inc.

[6] “PLBV46 Interface Simplifications, SP026 (v1.0) October 11, 2007”, Xilinx Inc.

[7] “LogiCORE IP XPSTimer/Counter (v1.02a), DS573 April 19, 2010”, Xilinx Inc.

AUTHOR’S PROFILE

Dr. Rajesh S. Pande has received his Ph.D. and M.Tech. degree from VNIT, Nagpur (India) and B.E. from Govt. College of Engineering, Jabalpur (India). His research interest include: *RF-MEMS, VLSI Design and VLSI Technology.*

He has published more than 20 papers in national, international conferences and Journals. He is a member of various technical organizations like IEEE, ISTE and IETE. He is closely associated with “**Indian Nanoelectronics Users Program (INUP)**”- a unique activity jointly run by the Centers of Excellence in Nanoelectronics at the Indian Institute of Technology, Bombay (IITB) and the Indian Institute of Science, Bangalore (IISC). He is recognized supervisor for Ph.D. in *Rashtrasant Tukdoji Maharaj Nagpur University.*

Currently he is working as VICE-PRINCIPAL and HEAD of the Electronics Engineering department, RCOEM, Nagpur (India).

Pravin A. Dwaramwar has received his M.Tech. from *Rashtrasant Tukdoji Maharaj Nagpur University*, Nagpur (India) and B.E. from VNIT, Nagpur (India). His research interest include: *RF-IC Design, CMOS analog VLSI Design and Power Electronics.*

He is currently working as ASSOCIATE PROFESSOR in Department of Electronics Engineering, Shri Ramdeobaba College of Engineering and Management (Formerly Known as Shri Ramdeobaba Kamla Nehru Engineering College), Nagpur.

Sandeep R. Pandey completed Graduation Degree (B.E in Electronics and Telecommunication) in 2006, from Nanded(India) and M.Tech in VLSI Design from *Rashtrasant Tukdoji Maharaj Nagpur University*, Nagpur in 2009. Fields of interest include: *Analog/Digital CMOS IC Design, Reconfigurable Computing, VLSI testing, Embedded systems.* He is currently working as ASSISTANT PROFESSOR in Department of Electronics Engineering, Shri Ramdeobaba College of Engineering and Management (RCOEM) (Formerly Known as Shri Ramdeobaba Kamla Nehru Engineering College), Nagpur.

Jitendra B. Zalke completed Graduation Degree (B.E in Electronic Design Technology) in 2006, from Nagpur (India) and M.Tech in VLSI Design from *Rashtrasant Tukdoji Maharaj Nagpur University*, Nagpur in 2009.

Fields of interest include: *Reconfigurable Computing, VLSI, Embedded systems.*
He is currently working as ASSISTANT PROFESSOR in Department of Electronic Design Technology, Shri Ramdeobaba College of Engineering and Management (Formerly Known as Shri Ramdeobaba Kamla Nehru Engineering College), Nagpur.