Design and Implementation of an FPGA Based System on Chip (SoC) For Embedded Control Application

Salman Jafri[1], Dr.Muhammad Ali Ismail[2]

Department of Electrical Engineering,
[1]Usman Institute of Technology, Engineering, UIT, Karachi, Pakistan
[2]Department of Computer & Information Systems Engineering, NED University of Engg. & Technology, Karachi, Pakistan

Abstract
FPGAs offer a great prototyping medium for complex digital systems. FPGAs focus on the hardware implementation of digital systems where the circuit is specified in a Hardware Description Language (HDL) like Verilog or VHDL. Verilog coding, debugging and verification takes time. To save time and also implement efficient and high performance FPGA based systems, the system can be divided into software tasks running on a softcore processor (coded in HDL) and delegating the high performance application specific tasks to hardware accelerators (hardware tasks). This technique is termed as Hardware / Software Codesign. It speeds up design of FPGA based systems because a small subset of highly specific tasks need to be coded in HDL and the rest of the application is easily handled by the Software running on the softcore processor. This paper represents design and implementation of an FPGA based System on Chip (SoC) consisting of a Xilinx PicoBlaze soft processor core interfaced with HDL coded hardware modules. The design is extensible and more modules can be integrated in future to incorporate highly specialized hardware tasks. The system has been implemented on a Spartan 6 based FPGA board.

Key words:
RISC Architecture; System on Chip; UART; FPGA

1. Introduction

Different implementations of FPGA based Systems on Chip exist in literature. Authors in [1] discuss a system on Chip based on the Xilinx Microblaze Softcore processor. Open source processor based systems like [2] use the OR1200 processor as the softcore processor for the platform. The RISCV softcore processor [3] has also gained popularity in optimized embedded control applications. A Nios-II powers the SoC for an embedded web server in [4].

The major consideration in using softcore processors is area/resource efficiency in terms of the resources of the FPGA. Most of the above cited implementations are resource hungry and complex in implementation and take up a lot of valuable FPGA resources. Additionally softcore processors like Xilinx Microblaze [1] need to be purchased as an intellectual property (IP). This adds to the cost and complexity of developing systems for embedded control applications. To alleviate the problems of licensing and to build a resource-efficient SoC, our SoC uses the Xilinx PicoBlaze ™ softcore processor. PicoBlaze is free, its HDL source is freely available (so it can be modified for customized processor implementations). Most importantly it is resource-efficient and can be implemented on modest FPGA boards. PicoBlaze might not be fast for some demanding applications but the idea of our system is that hardware/software co-design may be used to add power to the lean and mean PicoBlaze whereby specialized high speed hardware modules may tackle the specialized high performance tasks while normal priority tasks may be easily implemented on Software running on the PicoBlaze softcore processor. This approach has two major advantages:

1. The system is very efficient and resource friendly.
2. The system can expand to accommodate high performance hardware modules.

The rest of the paper is organized as follows: Section II explains System architecture, Section III highlights system implementation, Section IV explains system working, Section V presents hardware implementation results for the FPGA and Section VI concludes the paper.

2. System Architecture

The block diagram of the developed SoC is shown in Figure 1. The processing unit which is a PicoBlaze softcore processor is interfaced through a bus network with its external interfaces. The SoC has three interfaces- a general purpose Input/output interface, a UART based serial communication interface and an 8 bit Analogue to Digital Convertor (ADC) interface. The System on Chip is implemented on a Xilinx Spartan 6 based FPGA kit. The PicoBlaze, UART and General purpose IO ports have been implemented in hardware using Verilog HDL while the ADC module is incorporated by the PIC16F877A microcontroller and interfaced to the FPGA board. This approach offers added flexibility and it is explained in Section III.

Manuscript received May 5, 2019
Manuscript revised May 20, 2019

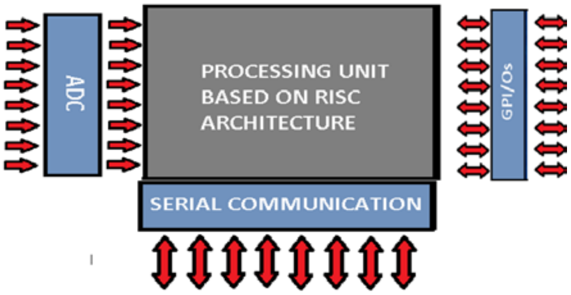

Fig. 1 The architecture of the System on Chip Module

3. System Implementation

The SoC consists of the following modules:

A. Xilinx PicoBlaze(KCPSM6)

The processing unit of the System on chip module is based on Xilinx PicoBlaze implementation named KCPSM6. Figure 2 shows the Kcpsm6 module of the SoC indicating the various signals to be connected. The code memory is addressed by using the address bus and the instruction size is 18 bits. The processor can handle interrupts and can access upto 256 input and output ports. It has an 8 bit RISC architecture. KCPSM 6 is optimized for Spartan 6 series FPGA kits and 7 series FPGA kit by Xilinx. For Spartan 6 FPGA it has a KCPSM6 code assembler [1]. Figure 2 show RTL representation of the KCPSM6 controller, when implemented in Verilog HDL. One of the most important thing to notice is the 18 bit instruction pin. All the information are defined by 18 bit instructions. It takes KCPSM6 two cycles to execute an instruction.

The processing unit is optimized for efficient code execution and presents an efficient device with respect to resource utilization of the targeted FPGA (See Table 1). The interrupt handling feature of the processor allows sophisticated real-time applications to be built. This would cater for systems that require real-time response in the presence of peripherals with varying data transfer speeds. As shown in Figure 2, additional signals read_strobe and write_strobe are provided for peripheral interfacing and this offers a simplistic but powerful method for device interfacing. The input and output port are placed in separate address spaces and are each addressed by eight bit addresses. The sleep mode is provided for energy saving where dedicated watchdog timers may be added to the SoC to produce sophisticated power efficient systems with features like power saving program loops and wakeup on keystrokes. This adds further to deployment in power constrained embedded environments like drones. The RTL can be modified to incorporate different additional functions into the processing core like DSP (Digital Signal Processing) MAC (multiply and accumulate) instructions

The KCPSM6 version of Picoblaze is compatible for various latest kits by Xilinx like Virtex 6, 7 series FPGA (Artix 7 for instance) and Spartan 6. KCPSM6 only utilizes 26 Slices of an FPGA kit to operate. This results in fewer resources of the FPGA to be used [1]. Furthermore KCPSM6 supports programs up to 4K instructions. It has an additional bank of 16 registers, dynamic JUMP and CALL, user defined interrupt vector and constant-optimized output ports [1] [2]. We have implemented the Verilog HDL version of KCPSM6. This is yet another contribution of this work because KCPSM6 has mostly been implemented in VHDL in prior works.

The main motivation behind this Verilog Implementation was to enable an infra-sturcture for future enhancement and research of the SoC using Verilog HDL. This may allow future reseaerch activities on the developed system for example incorporating complex image processing algorithms using custom accelerators interfaced to the PicoBlaze System, or incorporating sound wave processing engines incorporating digital filtering and sonic refinement.

B. A Universal Asynchronous receiver / transmitter (UART)

The SoC consists of a UART for asynchronous serial communication. This module is used to interface the SoC. The UART communicates with the PC and is interfaced to the PicoBlaze. Commands can be given to the SoC as explained in Section C.

C. General Purpose I/O Interface

This interface is used to the output the PWM wave and also the analog to digital convertor module (ADC). This block can be thought of as the I/O connectgivity portion of the system. As a future extension multiple such systems may be interfaced or specialized hardware may be directly interfaced through the system bus of the PicoBlaze.

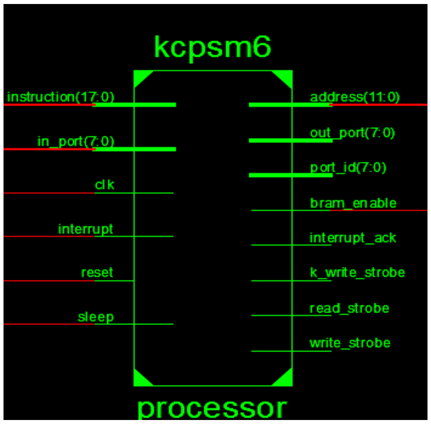

Fig. 2 RTL representation of KCPSM6

Figure 3 shows the schermatic diagram generated by Xilinx ISE using the RTL schematic tool. This schematic reveals the many components just discussed and shows the layout of the system as coded in the Verilog HDL.

LED block contains the block RAM containing the code to be executed by the PicoBlaze processor. The UART block connects to the PC for serial communication and is also interfaced to the PicoBlaze (KCPSM6). This allows the controlling of the SoC through the PC by using the PC terminal software.

The ADC block (not shown) is interfaced to the system and offers the interface of an eight channel 10 bit ADC. This offers optimum channels and resolution for embedded control applications of moderate size and complexity. This block may be useful in digital control applications and resolution may be traded off for frequency for high speed applications. To offer flexibility ADC is provided through a PIC16F877A microcontroller. This allows programmable flexibility for incorporating multi-channel ADC applications for complex applications like digital multi-channel oscilloscopes or logic analyzers.

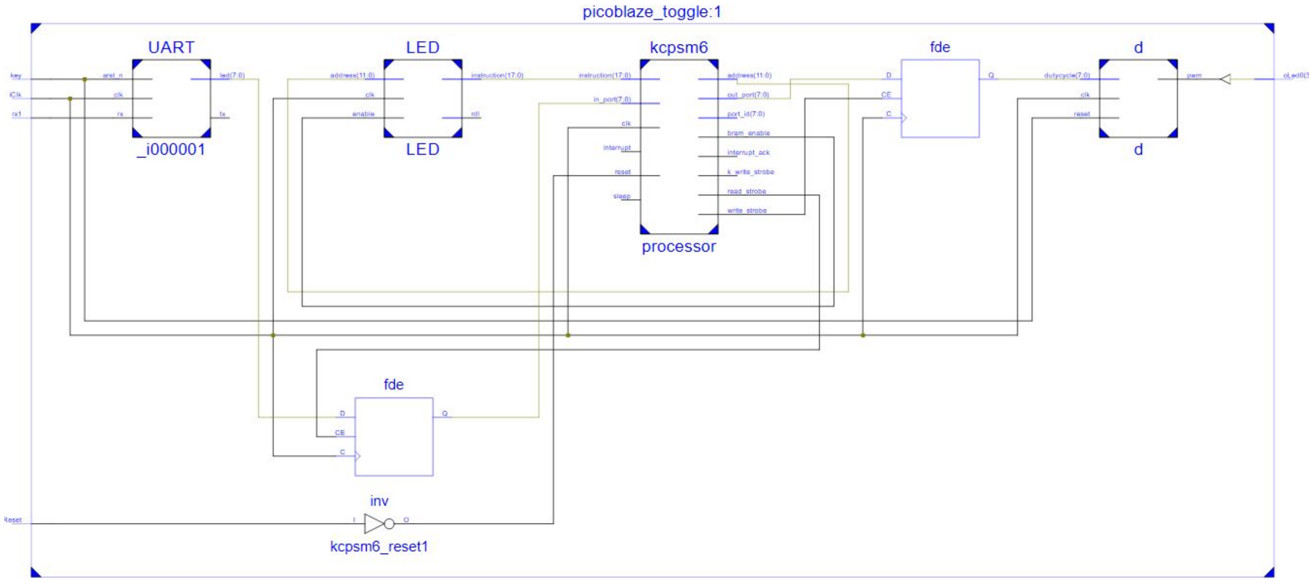

Fig. 3 RTL Schematic generated by the Xilinx ISE RTL Schematic tool

4. Working

The user inputs the desired duty cycle for the PWM wave from the serial connection through the Personal computer (PC). This input is received with the help of the UART in the SoC. In response,the PicoBlaze (KCPSM6) generates the respective PWM signal. This PWM signal drives a DC fan connected to the SoC. In another specialized application of the SoC, The ADC module is connected to a temperature transducer and provides the temperature reading to the SoC. As a result, the SoC can control the speed of the DC fan by PWM. These are demonstrative examples of the SoC and more complex applications and devices may be constructed by modifying the KCPCM6 Verilog HDL code and also incorporating and interfacing new HDL modules into the system. Figure 4 shows the implementation of the system. The actual results of synthesis of the SoC are presented in Table 1 where the highly optimized design of the SoC is evident. Figure 5 shows the output of the PWM unit of the SoC for a 50 % duty cycle. As is evident from the figure, accurate PWM waveform are being generated by the unit with the proper timing constraints. This is because the synthesis does not include any timing violations and latching elements.

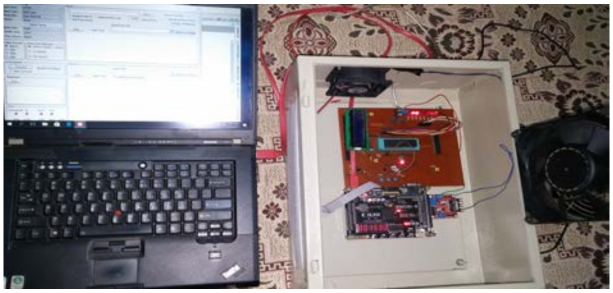

Fig. 4 The System on Chip showing the ADC module PCB

5. Results

Table 1 shows the resource utilization for the Spartan 6 FPGA as implementation results. One can see that the SoC only utilizes 1 % of slice registers and 3% each of slice LUTs, occupied slices and bonded IOBs. These figures translate into a very optimized SoC design for our modest Spartan 6 FPGA. They also show the potential for future growth to incorporate complex hardware modules in Verilog HDL. This is true for even for the modest Spartan 6 FPGA and the possibilities grow exponentially for the complex FPGAs like the Artix 7 and Vertex 7 Devices.

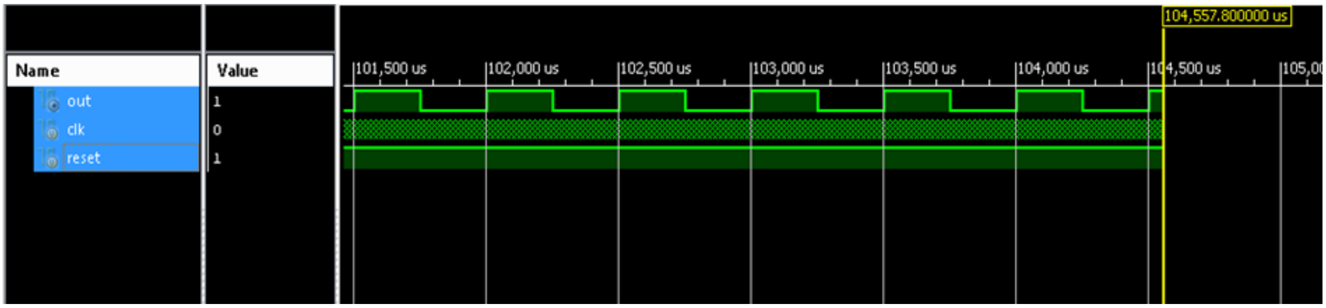

Fig. 5 Output of the PWM unit of the SoC.

Table 1: Resource Utilization of Spartan 6 FPGA

Resource	Used	Available	Utilization
Number of Slice Registers	157	11440	1%
Number of Slice LUTs	198	5720	3%
Number of Occupied Slices	57	1430	3%
Number of bonded IOBs	6	186	3%

Note that the user decides the duty cycle based on the temperature measured using the ADC channel which is also interfaced using the transmitter of the UART. The temperature value is received at another I/O pin of the System on Chip.

As a result the speed of the fan varies as the temperature increases or decreases. This leads to a controlled ventilation system which saves energy and reduces running cost as well. Various more complex applications can be interfaced using the system on chip.

5. Conclusion

The results show that our SoC is very resource friendly from the point of view of the Spartan 6 FPGA.The developed setup can be expanded to incorporate ore complex embedded control systems by including further HDL modules. Also, the inclusion of the ADC module of the PIC16F877A microcontroller into the system increases the possibilities even further. In a future derivative of the system, this ADC module might include some software based signal processing of its own e.g. digital filtering or sound processing algorithms. The possibilities expand even further for the high end Xilinx Devices like Virtex 7 and Artix 7

References

[1] B. Bengherbia, S. Chadli, M. O. Zmirli and A. Toubal, "A MicroBlaze based WSN sink node using XBee transceiver," 2016 8th International Conference on Modelling, Identification and Control (ICMIC), Algiers, 2016, pp. 831-834. doi: 10.1109/ICMIC.2016.7804229

[2] S. Xuan, J. Han, Z. Yu, Y. Ren and X. Zeng, "A configurable SoC design for information security," 2015 IEEE 11th International Conference on ASIC (ASICON), Chengdu, 2015, pp. 1-4. doi: 10.1109/ASICON.2015.7516998

[3] C. A. R. A. Melo and E. Barros, "Oolong: A Baseband processor extension to the RISC-V ISA," 2016 IEEE 27th International Conference on Application-specific Systems, Architectures and Processors (ASAP), London, 2016, pp. 241-242. doi: 10.1109/ASAP.2016.7760808

[4] P. G. Salunke and A. M. Sayyed, "Design of embedded web server based on NIOS-II soft core processor," 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), Chennai, 2016, pp. 488-492. doi: 10.1109/ICEEOT.2016.7755638

[5] L. Akçay, M. Tükel and S. B. Örs, "Implementation of an OpenRISC based SoC and Linux Kernel installation on FPGA," 2016 24th Signal Processing and Communication Application Conference (SIU), Zonguldak, 2016, pp. 1969-1972. doi: 10.1109/SIU.2016.7496153

[6] L. Janik, M. Novak, L. Hudcova, O. Wilfert and A. Dobesch, "LwIP based network solution for MicroBlaze," 2016 International Conference on Broadband Communications for Next Generation Networks and Multimedia Applications (CoBCom), Graz, 2016, pp. 1-4. doi: 10.1109/COBCOM.2016.7593513

[7] M. A. Jaballah, D. Mezghani and A. Mami, "Development of a MPPT controller for hybrid wind/photovoltaic system based on SOPC and NIOS II," 2015 16th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA), Monastir, 2015, pp. 700-706. doi: 10.1109/STA.2015.7505230

[8] D. Romeo, J. LaMagna, I. Hogan and J. C. Squire, "An Introduction to Soft-Core Processors and a Biomedical

Application," in IEEE Potentials, vol. 37, no. 2, pp. 13-18, March-April 2018. doi: 10.1109/MPOT.2017.2733341

[9] S. Tamimi, Z. Ebrahimi, B. Khaleghi and H. Asadi, "An Efficient SRAM-Based Reconfigurable Architecture for Embedded Processors," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 38, no. 3, pp. 466-479, March 2019. doi: 10.1109/TCAD.2018.2812118