# IMPLEMENTATION OF 16-BIT MICROPROCESSOR WITH RISC ARCHITECTURE ON AN FPGA USING VERILOG

W. A. D. P. B. Senarathne*, W. A. S. Wijesinghe

*Department of Electronics, Wayamba University of Sri Lanka, Kuliyapitiya, Sri Lanka*
*piyumibuddhini@gmail.com**

## ABSTRACT

Today's world most essential part of the various applications, system is embedded with processing unit. One of the most reliable, low power consumption way to implement those systems are using Field Programmable Gate Arrays (FPGAs). When FPGAs are used, output can decide while designing process and devices can reprogrammable. So, it's more accurate if using FPGAs. In this project comprise the designing process and hardware implementation of simple 16-bit Reduced Instruction Set Computer, called RISC processor on a FPGA which can perform a basic ALU operations. For the implementation Xilinx Spartan 3E development board was used that is coming up with high features. This 16-bit processor runs on Xilinx XC3S500E Spartan 3E FPGA. Synthesized using Xilinx ISE 14.7 Webpack, simulated using ISim simulator. Verilog language used for modeling components. The source code of the processing system is very simple and compacted. In this architecture I/O, addresses are transferring based on stack machine system that is LIFO (Last-In First-Out) sequence. This paper presents, Small processor implemented on a FPGA that is used RISC architecture on the design. All the instructions were in same size, so that it may be fetched in a single operation. The instruction set adopted here is extremely simple that gives an insight into the kind of hardware which should be able to execute the set of instructions properly. The processor's architecture features 16 bit instruction words.

Keywords: FPGA, RISC architecture, ALU, CPU, Xilinx, Verilog

## 1. INTRODUCTION

### 1.1 Field-Programmable Gate Array (FPGA)

FPGA is a Programmable Logic Device (PLD) and also a kind of IC with high performances. FPGAs are the most complex and density one among PLD categories that allow more complex logic implementations and can be used for implementation of digital circuits that use equivalent of several million logic gates. Users can configured FPGAs according to specifications which are they want. FPGAs used in high-speed custom digital applications, where designs tend to be more constrained by performance rather than cost.

FPGA has a bunch of configurable logic blocks (CLBs) decorate within a switching matrix and can be used as simple as a logic gates or as complex as a microprocessor. When FPGAs combined with processors, complex digital systems can be implemented on FPGAs easily [1,2]. Because of the FPGAs contain very large number of logic resources, a simple processor can be implemented on a FPGA, using Verilog that is the simplest and most popular HDL which can describes circuits. GDL compilers provide a gate map, not a computer executable file. After provide the gate map it download to the device. Advantages are fast design and better verification, provide the timing information[3]. FPGA architecture basically consists Configurable Logic Cells (CLB), I/O Blocks (Input/Output) and interconnections wires/switch matrix.

CLB is the main structure of FPGAs that consists of one Lookup Table (LUT), one D-Flip Flop and one 2 to 1 Multiplexer. Modern FPGAs contain up to hundreds of thousands of CLBs. Lookup tables are small memories that fulfill logic operations. Interconnections of logic cells are provides routing between the logic blocks, switch matrix formed data paths similar to the installed program on the FPGA. Switch matrix provides switching between interconnections depending on the logic and I/O pads which are used for the get inputs and take the output from the device[3].

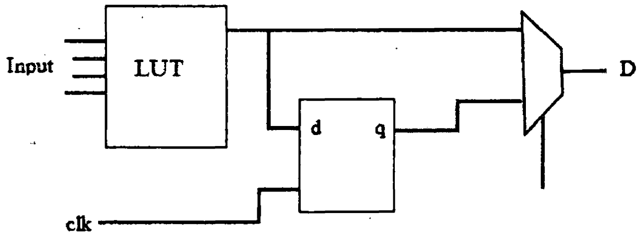

Figure 1: Basic FPGA Logic Block

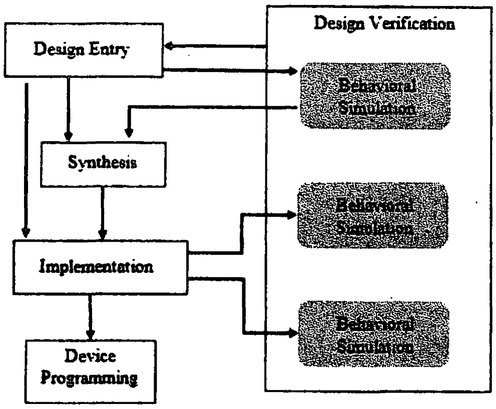

Figure 2: FPGA Design Flow

Simplified design flow of the FPGA is shown in the figure 4. Program should synthesis once design files have been created. Then implement to the board. This consist three steps, translate, map, place and route. Translate process combines all the input netlists and constraints to a logic design file. This file stored in named UCF files. Map process divides the whole circuit with logical elements into sub blocks such that they can be fit into the FPGA logic blocks. Then the place and route process places the sub blocks from the map process into logic blocks according to the constraints and connects the logic blocks. After these three steps design must be loaded on the FPGA. Also behavioral and function simulation are on the design flow. Behavioral simulation can be performed on Verilog design. Since this is a very fast simulation designer allows to change the code if want. Functional simulation gives information about the logic operation of the circuit[4, 5].

This paper presents a very simple 16-bit processor on a FPGA. Verilog language is used to implement this processor and Reduced Instruction Set Computer (RISC) architecture used for the CPU design. RISC processor uses simplified instructions for high performance, faster execution and reduce delay in execution. Also stack machine architecture used for the internal data and address flow. Stack machine uses LIFO sequence[4].

## 2. METHODOLOGY

Reduced Instruction Set Computer (RISC), is a microprocessor CPU design philosophy that favors a smaller and simpler set of instructions that all take about the same amount of time to execute. This strategy based on the insight that simple instruction set and higher performances per second. Also these more complex features took several processor cycles to be performed. Reduce the total number of memory accesses. Features are uniform instruction encoding, a homogeneous register set, Simple addressing modes, few data types supported in hardware. Since the earliest day of computing by common ways of overlapping are pre-fetching, pipelining and superscalar operation which are speed-up techniques$^6$.

In this research work, implement a simple, scalable, portable 16-bit processor in RISC architecture using Verilog language on a FPGA board. For the hardware implementation Verilog HDL language was used. The Spartan-3E Starter Kit board is used to this study which have the unique features of the Spartan-3E FPGA family and provides a convenient development board for embedded processing applications.

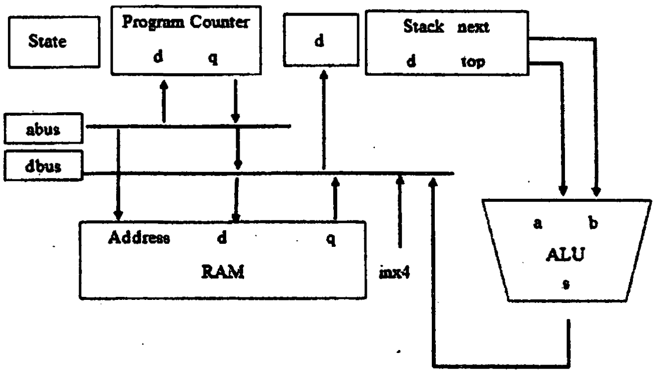

Figure 3: Processor Design Flow

According to the above diagram several blocks combined together for the implement. Arithmetic Logic Unit (ALU) block doing all the arithmetic and logic part of the processor. When inputs enter it sends to the RAM block and stored there. Address bus and data bus also stored in RAM. Address bus send to the program counter that is a register in the processor which contain addresses of the instruction being execute current time. Data bus going to the stack machine block. Here, stack machine working principal is output the data that comes last. And output data that comes earlier last one. All the blocks combined together and programmed in the CPU block. CPU unit designed based on Finite State Machine (FSM) that running state block. In this research also used above design shown in figure 5.

Programed mini CPU system in separate files including alu.v, dpram.v, stackm.v, statef.v, signed_mult.v and files were simulated and synthesized. All 5 files were combined in the minicpu.v file and run the program.

## 3. RESULTS & DISSCUSSION

After the program designed, create test bench file to get the timing control diagrams from ISim modulator and get results. Following timing control wave forms got as the result from the processor system. Only added here ALU and state machine results.

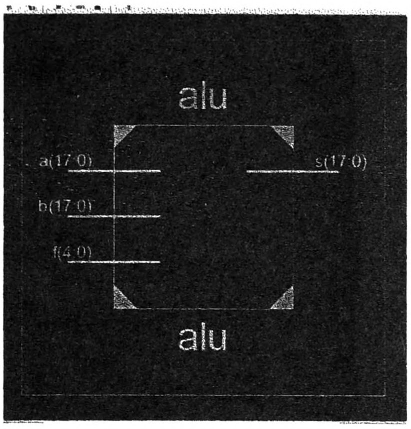

Figure 4: ALU top block

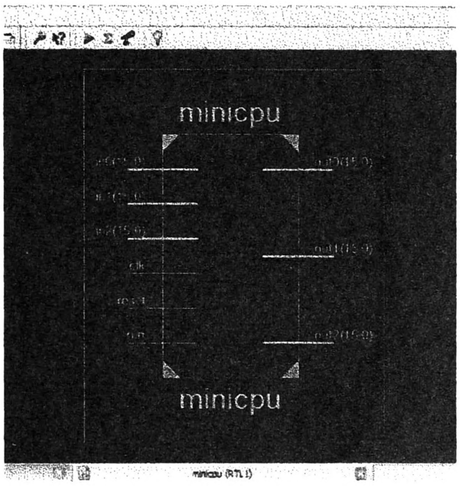

Figure 5: CPU Top Block

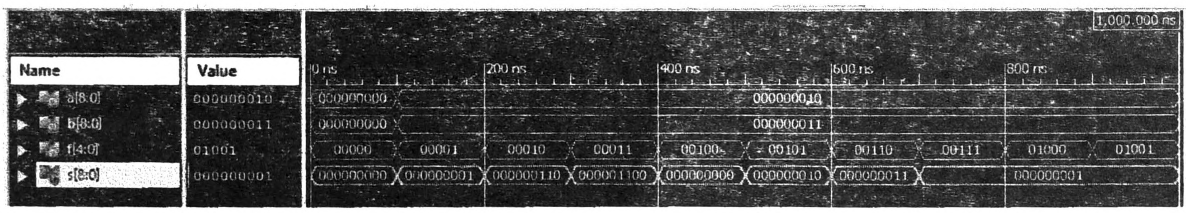

Figure 6: Simulation Result for the ALU

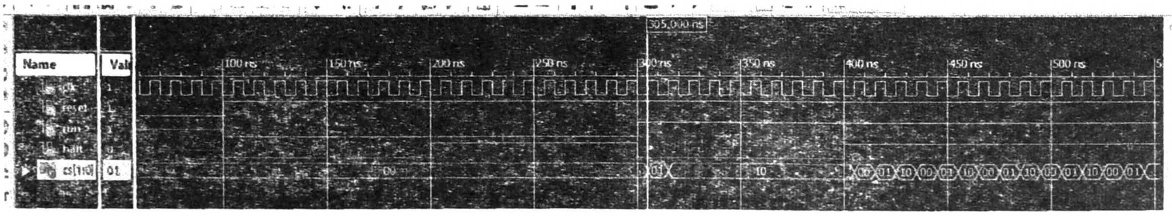

Figure 7: State Machine Simulation Result

## 4. CONCLUTION

This paper presents implementation of a simple, scalable, portable 16-bit processor in RISC architecture using Verilog language on a FPGA board. Design code synthesized using Xilinx ISE 14.7 Webpack and simulated using ISim simulator. The code for all modules were written using Verilog language and tested by applying test benches. All modules were executed without errors and expected results got from ISim modulator. Design implemented on Xilinx Spartan 3E development board. This implementation can be extended up to high bit values such as 32, 64, 128 bits.

## 5. REFERENCES

[1] M. Electronics, "Programmable Logic," [Online]. Available: http://www.mouser.com/applications/programmable-logic. [Accessed 2016 February 16].

[2] V. Prasanth and K. V. Rajeshkumar, "FPGA Based 64-Bit Low Power RISC Processor Using Verilog HDL," Surampalem, 2007.

[3] P. P. Chu, FPGA Prototyping by Verilog Examples, Xilinx Spartan - 3 Version, Clevelend: A JOHN WILEY & SONS, INC., PUBLICATION, 2008.

[4] S. Trimberger, "Effects of FPGA architecture on FPGA routing," in *DAC '95 Proceedings of the 32nd annual ACM/IEEE Design Automation Conference*, New York, 1995.

[5] S. Brown and J. Rose, "Architecture of FPGAs and CPLDs: A Tutorial," University of Toronto.

[6] F. Masood, "RISC and CISC - Computer Architecture".

[7] "The History of the Integrated Circuit," Nobelprize.org. Nobel Media AB,, 2014. [Online]. Available: http://www.nobelprize.org/educational/physics/integrated_circuit/history/. [Accessed 10 February 2016].

[8] S.palnitkar, Verilog HDL - Aguide to Digital and Synthesis, California: Sun Microsystems, 1996.

[9] S. Muthukrishnan and R. Priyadharsini , "32-Bit RISC and DSP System," *International Journal of Computer Science and Mobile Computing,* vol. 3, no. 12, p. 361–368, 2014.