International Journal of Engineering Research and General ScienceVolume 2, Issue 5, August – September 2014
ISSN 2091-2730

Design of 16-bit Data Processor Using Finite State Machine in Verilog

Shashank Kaithwas[1], Pramod Kumar Jain[2]

[1] Research Scholar (M.Tech) SGSITS

[2] Associate Professor, SGSITS

E-mail- shashankkaithwas09@gmail.com

Abstract— This paper presents design concept of 16—bit Data processor. Design methodology has been changing from schematic to Hardware Descriptive Language (HDL) based design. Data processor has been proposed using Finite State Machine (FSM). The state machine designed for the Data Processor can be started from any state and can jump on any state in between. The key architecture elements of Data processor such as, Arithmetic Logic Unit (ALU), Control Unit and Data-path are being described. Functionalities are validated through synthesis and simulation process. Besides verifying the outputs, the timing diagram and interfacing signals are also track to ensure that they adhere to the design specification. The Verilog Hardware Descriptive Language gives access to every internal signal and designing Data Processor using this language fulfils the needs for different high performance applications.

Keywords– HDL-Hardware Descriptive Language, FSM-Finite State Machine, ALU-Arithmetic Logic Unit, Control Unit, Data-path, Data Processor, Verilog Hardware Descriptive Language.

INTRODUCTION

Processors are the heart of all "smart" devices, whether they be electronic devices or otherwise. Their smartness comes as a direct result of the decisions and controls that processors makes. There are generally two types of processor: general purpose processors and dedicated processors. General-purpose processors such as the Pentium CPU can perform different tasks under the control of software instructions. General purpose processors are used in all personal computers. Dedicated processors also known as application-specific integrated circuits (ASICs) and are designed to perform just one specific task. For example, inside the cell phone, there is a dedicated processor that controls its entire operation. The embedded processor inside the cell phone does nothing, but controls the operation of the phone. Dedicated processors are therefore, usually much small and not as complex as general purposes processors.

The different parts and components fit together to form the processor. From transistor, the basic logic gates are built. Logic gates are combined together to form either combinational circuits or sequential circuits. The difference between these two types of circuits is only in the way the logic gates are connected together. Latches and flip-flops are the simplest forms of sequential circuits, and they provide the basic building blocks for more complex sequential circuits. Certain combinational circuits and sequential circuits are used as standard building blocks for larger circuits, such as the processor. These standard combinational and sequential components usually are found in standard libraries and serve as larger building blocks for the processors. Different combinational and sequential components are connected together to form either the data path or the control unit of a processor. Finally, combining the data path and the control unit together will produce the circuit for either a dedicated or general processor.

However, they are used in every smart electronic device such as the musical greeting cards, electronic toys, TVs, cell phones, microwave ovens and anti-lock break systems in car. Although the small dedicated processors are not as powerful as the general-purpose processors, they are being sold and used in a lot more places then the powerful general-purpose processors that are used in personal computers.

DESIGN OF MODULE

This contains designing of important processor module such as ALU, Data path and Control circuit.

ARITHMETIC AND LOGICAL UNIT (ALU)

The arithmetic-logic unit (ALU) performs basic arithmetic and logic operation which are controlled by the opcode. The result of the execution of the instruction is written to the output. Designing of ALU is done for arithmetic operation such as addition, subtraction, multiplication, increment, decrement etc. The inputs are 16-bit wide with type unsigned. Figure 1 shows ALU Block Diagram:-

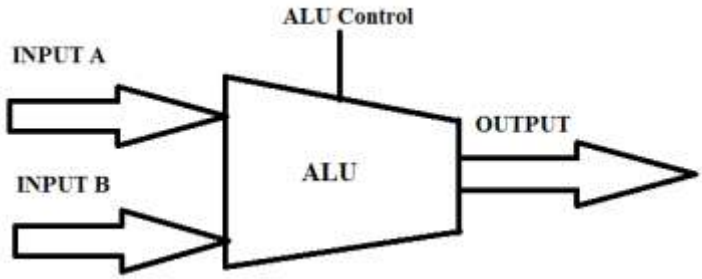

Fig 1. Block Diagram ALU

CONTROL CIRCUIT

The control unit is a sequential circuit in which its outputs are dependent on both its current and past inputs. This history of past inputs is stored in the state memory and is said to represent the state of the circuit. Thus, the circuit changes from one state to next when the content of memory changes. Depending on the current state of the circuit and the input signals, the next-state logic will determine what the next state ought to be by changing the content of the state memory. Hence, a sequential circuit executes by going through a sequence of states. Since the state memory is finite, therefore the total number of different states that the circuit can go to is also finite. This is not to be confused with the fact that the sequence length can be infinitely long. However, because of the reason of having only a finite number of states, a sequential circuit is also referred to as a Finite State Machine (FSM). . Figure 2 shows the block diagram of the control unit.

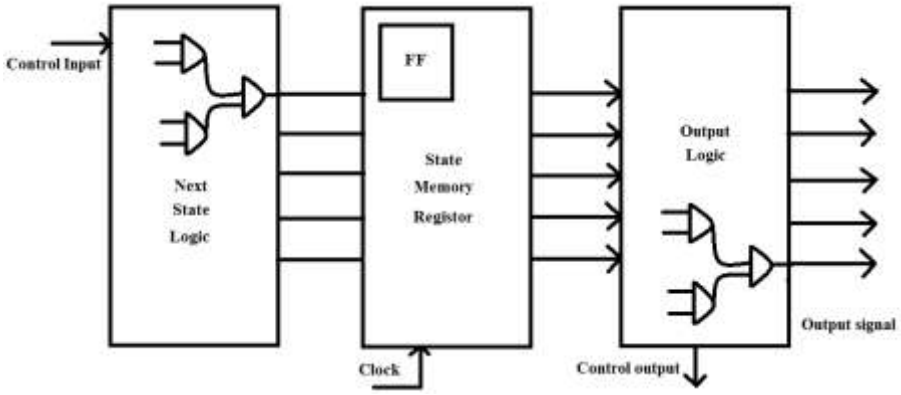

Fig 2. Block Diagram of Control Circuit

DATAPATH

The design of functional units for performing single, simple data operations, such as the adder for adding two numbers or the comparator for comparing two values is described. However, for adding a million numbers, there is no need to connect a million minus one adder together. Instead, take a circuit with just one adder and to use it a million time. A data path circuit allows to do just that, i.e., for performing operations involving multiple steps. Figure 3 shows a simple data path using one adder to add as many numbers as desired. In order for this to be possible, a register is needed to store the temporary result after each addition. The temporary result from the register is feed back to the input of the adder so that the next number can be added to the current sum.

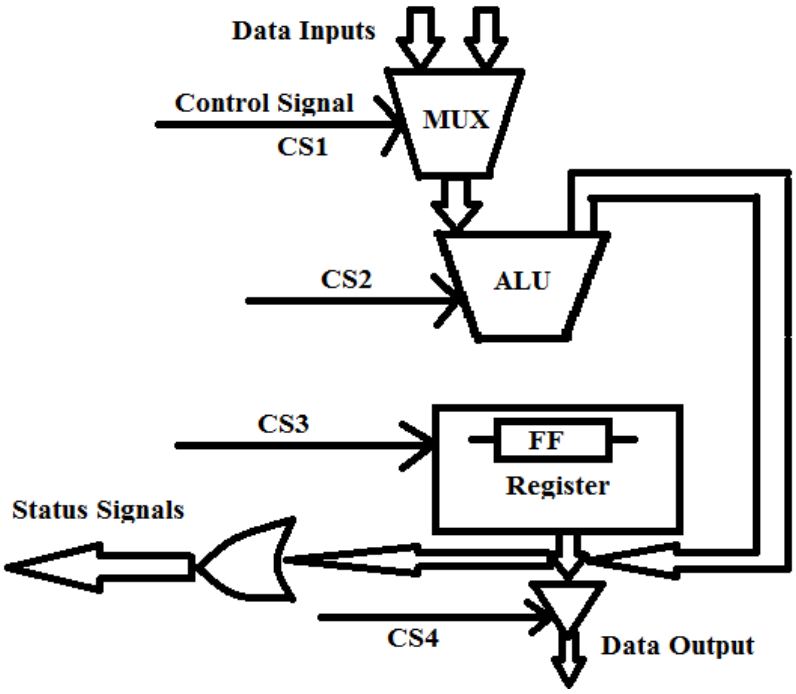

Fig 3. Block Diagram of Datapath

PROCESSING UNIT

Datapath and control unit together makes processing unit. Control circuit provides essential control signals to the Datapath unit for required operations. And the Datapath is the part concerning the flow of data, to be manipulated, transmitted or received. Functional Diagram of Processing Unit is shown in figure 4.

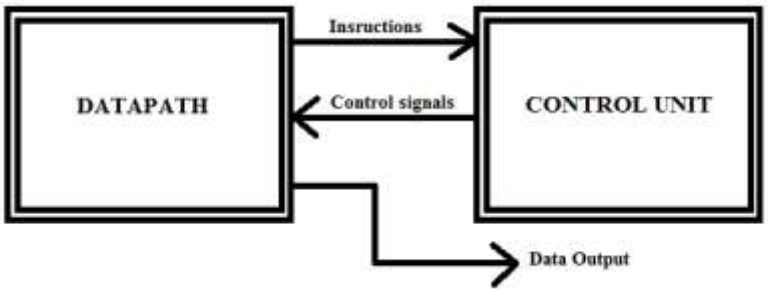

Fig 4. Functinal Diagram of Processing Unit

STATE MACHINE DIAGRAM

Figure 5 and 6 shows the State Diagram for processor having four states and eight states respectively. This paper presents the 16 states data processor which can be designed easily sighting these two state diagrams. The two state machines starts from State0 if the RESET is set to logic 1 and if the RESET is forced to logic 0 then depending on the value of START the state machines changes to next states also we can switch to any state from the present state of the state machine. If the value of START does not change then the machine will remain on the present state. For each state a particular opration is assigned.

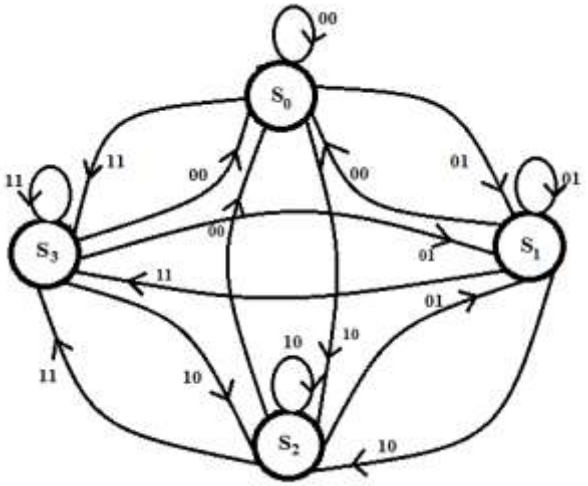

Fig 5. State Machine Diagram having 4 states

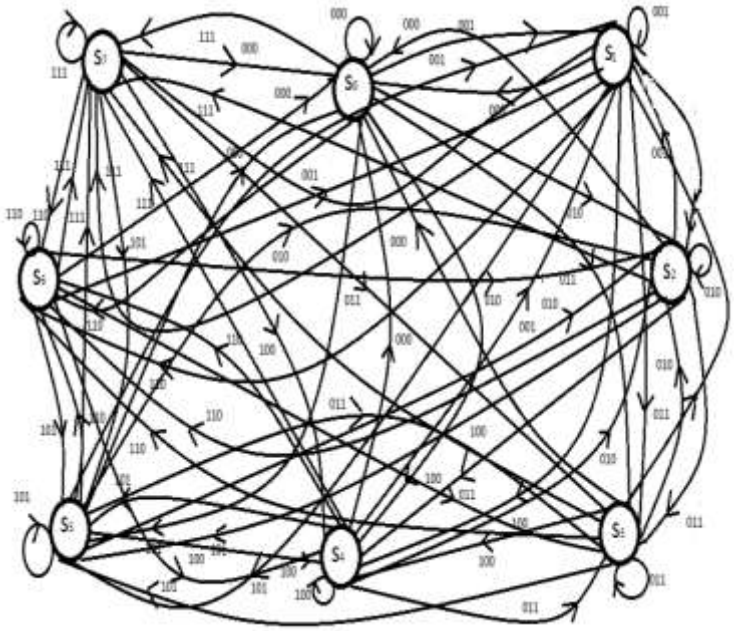

Fig 6. State Machine Diagram having 8 states

DATA PROCESSORS SPECIFICATIONS

Table I:- shows the various specifications according to the opcodes

OPCODE	OPERATION	SPECIFICATION
0000	a + b	zout is assigned the value of a+b
0001	a - b	zout is assigned the value of a-b
0010	a + 1	Incremented value of a assign to zout
0011	a - 1	Decremented value of a assign to zout
0100	a OR b	zout is assigned the value of a or b
0101	a AND b	zout is assigned the value of a and b
0110	NOT a	zout is assigned the value of not a

0111	NOT b	zout is assigned the value of not b
1000	a NAND b	zout is assigned the value of a nand b
1001	a NOR b	zout is assigned the value of a nor b
1010	a XOR b	zout is assigned the value of a xor b
1011	a XNOR b	zout is assigned the value of a xnor b
1100	a << 1	Shifted left value of a assign to zout
1101	a >> 1	Shifted right value of a assign to zout
1110	b << 1	Shifted left value of b assign to zout
1111	b >> 1	Shifted right value of a assign to zout

RTL (Register-transfer Level) GENERATION

The RTL (Register-transfer Level) view of the Verilog code is shown in figure 7:-

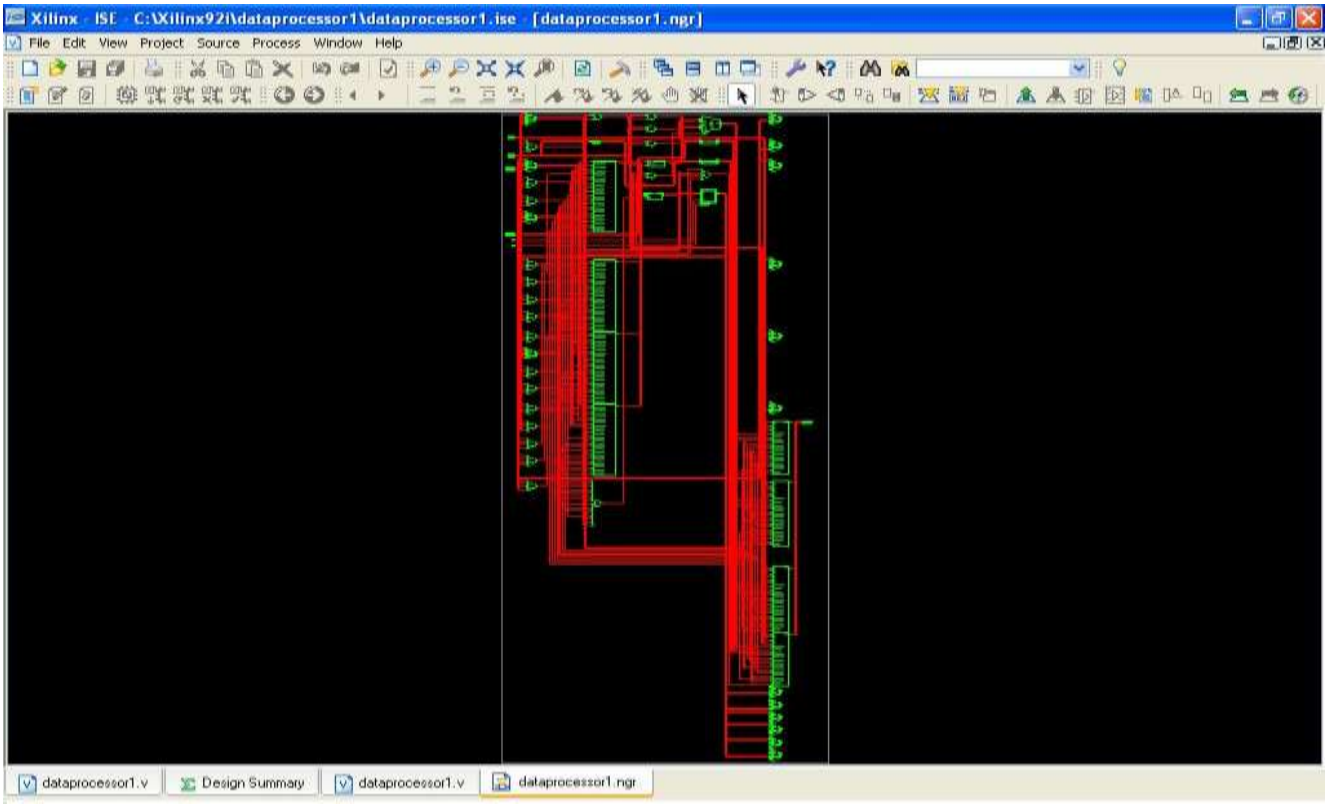

Fig 7 RTL view of the Data Processor

SIMULATION RESULTS

Design is verified through simulation, which is done in a bottom-up fashion. Small modules are simulated in separate testbenches before they are integrated and tested as a whole. The results of operation on the test vectors are manually computed and are referred to as expected results

By simulation for a and b where a=4 and b=2, zout gives following results shown in figure 8:

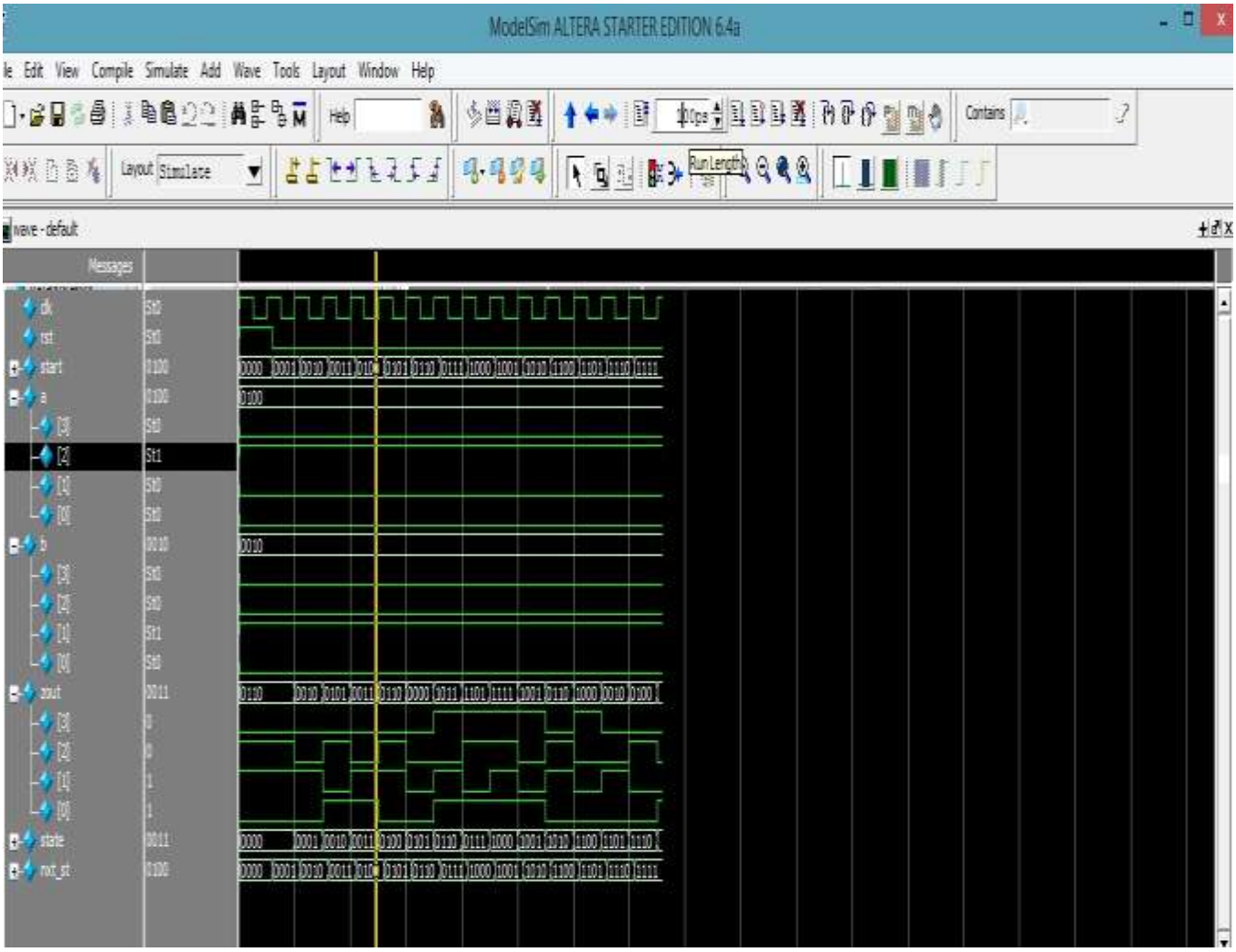

Fig 8. Simulation Results

ACKNOWLEDGMENT

We gratefully acknowledge the Almighty GOD who gave us strength and health to successfully complete this venture. We wish to thank lecturers of our college for their helpful discussions. We also thank the other members of the Verilog synthesis group for their support.

CONCLUSION

In this paper, we have proposed efficient Verilog coding verification method. We have also proposed several algorithms using different design levels. Our proposal have been implemented in Verilog using Xilinx 9.2a and Altera's Modelsim simulator, the RTL is generated in Xilinx 9.2a and the functionality has been checked in Modelsim simulator. the Data Processor design using Verilog is successfully designed, implemented and tested. Currently, we are conducting further research that considers the further reductions in the hardware complexity in terms of synthesis. Finally the code has been downloaded into Altera SPARTAN-3E: FPGA chip on LC84 package for hardware realization. Figure 9 shows the FPGA implementation of the design:-

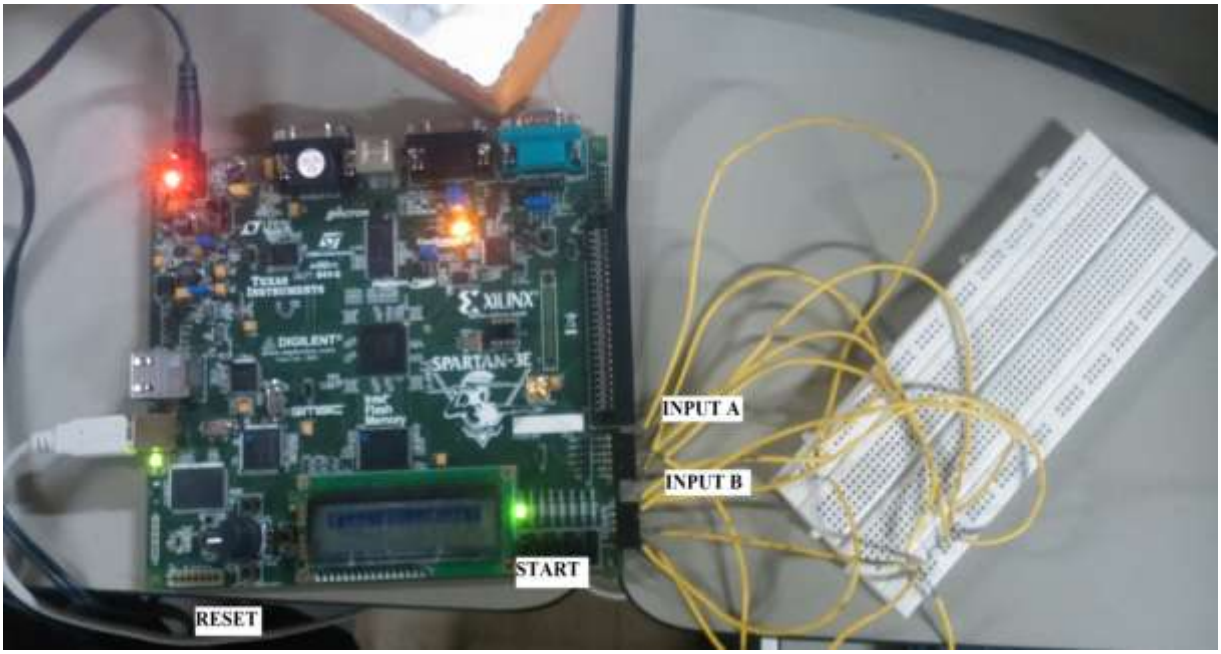

Figure 9. FPGA Implementation of the Design

REFERENCES:

[1] Development and Directions in Computer Architecture Lipovski, G.J. Doty, K.L. University of Texas Aug. 1978 Volume: 11, Issue: 8. (Pp.54-67)
[2] Andrei-Sorin F., Corneliu B., 2010 "Savage 16-16 bit RISC Architecture General Purpose Microprocessor" in Proc, IEEE Journal. (Pp.3-8)
[3] Venelin Angelov, Volker L., 2009 "The Educational Processor Sweet-16" in Proc. IEEE Conference. (Pp. 555-559)
[4] J. Eyre and J. Bier, DSP Processors Hit the Mainstream, IEEE Micro, August 1988. [5] Gordon Bell, "RISC: Back to the Future?" , Datamation, Vol. 32, No. 11, June 1, 1986 (Pp.96-108)
[6] Gin-der Wu Kuei-Ting Kuo, "Dual-ALU structure processor for speech reorganization" Publication Date: 24-26, April 2006
[7] Tseyun Feng, Dharma P. Agrawal, "A Microprocessor-controlled asynchronous circuit switching network", 1979 (Pp.202-215)
[8] Xiao Tiejun, Liu Fang, 2008 "16-bit Teaching Microprocessor Design and Application" in Proc. IEEE International Symposium on It in Medicine and Education. (Pp.160-163)
[9] Cross, J.E. and Soetan, R. A., 1988 "Teaching Microprocessor Design using the 8086 Microprocessor" in Proc. IEEE Journal.
[10] J. Bhaskar, Verilog HDL Synthesis, A Practical primer
[11] Douglas j. Smith, HDL Chip Design: A Practical guide for Designing, Synthesizing and Simulation ASICs and FPGAs using VHDL or Verilog. JUNE 1996.
[12] *James M. Lee,* Verilog Quickstart. Hardcover Published by Kluwer Academic Pub. MAY 1997.
[13] Fraunhofer IIS, "From VHDL and Verilog to System".www.iis.fraunhofer.de/bf/ic/icdds/arb_sp/vhdl.
[14] Bannatyne, R, 1998 "Migrating from 8 to 16-bit Processor" in proc. Northcon/98 conference. (Pp. 150-158)