Open access Journal International Journal of Emerging Trends in Science and Technology

Risc Processor Using VHDL

Autors
Dr. S. L. Lahudkar[1], Amit M. Mankar[2], Sudesh D. Bhong[3]
Department of Electronics & Telecommunication,
JSPM's, Imperial College Of Engineering & Research,
Savitribai Phule Pune University, Pune, India
Email: *amitmankar69@gmail.com, sudesh.bhong@gmail.com*

ABSTRACT
Reduced Instruction Set Computer (RISC) is become a design theory that has accepted in engineering applications. Optimizing gate capacity and performance of FPGA kit. This devices provides complex logic systems to be performed and implemented on a single device. So the main aim of this project is to design, study and implement an 8-bit Reduced Instruction Set Computer processor using XILINX Spartan 3E tool. The elevate feature of Spartan-3E reduces the cost per logic cell designed. The prominent feature of the RISC processor is that this processor is very simple. RISC processor support load/store architecture (Von Neumann). The main components of processor include the Arithmetic Logic Unit, Control unit, Shifter, and Rotator.

Keywords- *RISC, PIPELING, VHDL, SPARTAN 3E TOOL.*

I. INTRODUCTION

In every life development is taking place at a very rapid rate. Market is becoming more and more customer driven. Customer wants system designed tailor made to suit his needs. This is putting burden and stress on the embedded system designer. Choosing appropriate processor and supporting peripherals is becoming very critical. Considering this aspect and the growth in the field of VLSI design, so the days are no longer when the people will start design their own processor

The selected project topic is aiming towards the same direction. The core is developed by using the very high speed integrated circuit hardware description language (VHDL) or (VHSICHDL), which offers the freedom of adding specific, advance hardware blocks for various operations. The effects on area and power consumption as well as computational power are analyzed.

This paper points towards the implementation of a RISC processor core that can be used as a single processor as well as a guide for a multiprocessor access for SoC designs.

As given above, most SoC architectures have harsh concern the area of module and power optimize. For this reason, we have chosen a well-suited architecture with small demands on both of these aspects. The Microchip PIC-Core architecture has sufficient performance for a broad variety of embedded applications. Therefore, we designed the Core in the hardware description language VHDL, a processor core that is binary compatible to the Microchi

II. LITERATURE REVIEW

According to [1] The concept of VHDL code is realized using the Spartan 3E XC3S100E by interfacing the input & output with the system through the USB. Also this paper addresses PWM control is verified using CRO.

According to [2] Increasing performance and gate capacity of recent FPGA devices permits complex logic systems to be implemented on a

single programmable device. So the main aim of this paper is to design Reduced Instruction Set (RISC) processor using XILINX Spartan 3E tool. The module process and solve the problem of area, dissipation of power and propagation delay are analyzed.

According to [3] This project is to write a VHDL behavioral model, evolve test-bench and mimic the behavior. Analyze instruction data path, decoder module function, MIPS instruction format and design theory based on RISC instruction set using pipeline process.

III. METHODOLOGY

Here design of architecture of an 8-bit RISC processor is shown in Figure. This architecture consists of control unit, shifter, rotator and arithmetic logic unit. The processor is designed with Von Neumann architecture. In this separate data bus and separate address bus is provided between processor and memory. Instruction and data are fetched in linear order so that the inactivity occur between the machine cycles can be reduced.

Four stages of pipelining have been integrated in the design which provides the speed of operation. The pipelining stages are Instruction Fetch, Instruction Decode, Instruction Execute and Write back. In Instruction fetch, the instruction and the necessary data are taken from the memory. Whereas in Instruction decode, the instruction and data that are drawn from the memory are separated call up the components and the data path so as to execute

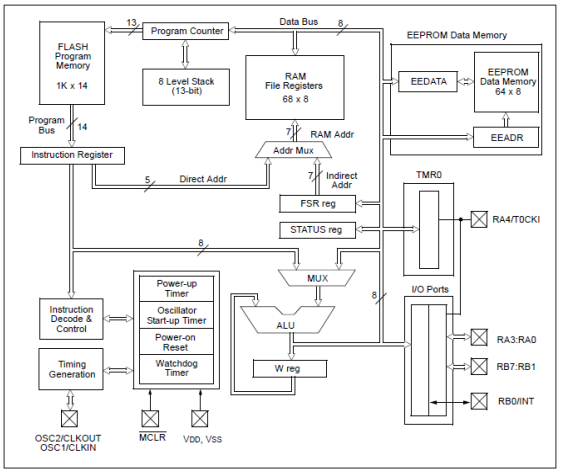

Figure.1 Architecture Diagram of 16F84A

And finally in execution stage, the instruction is executed, the data is employ and the result is stored.

IV. ALGORITHMS

A. The control unit reads the op-code and instruction bit and then creates control signals as outputs that trigger the respective components and data path to operate the correct task. It has two instruction decoder that decode the instruction bits and the unscramble output of the control unit is deliver as control signal either into Arithmetic logic unit (ALU) or Universal shifter or Barrel shift rotator.

B. The operands are acknowledged from register A and register B by the ALU. Depending on the control signal from the control unit the ALU performs either arithmetic or logic operations.

C. After the execution of the instruction, the result is always stored in the accumulator register or A register.

D. Input is taken from source register A and is either loaded or shifted in right or left direction planted on the control lines activated by the control unit.

E. The shifted data is saved in the accumulator register. Input data is prone from source register A and rotated/shifted N number of times based on the op-code given from the control unit.

F. The rotated data is stored in the acc. register.

V. SIMULATION & RESULT

In the simulation part first MIPS single-cycle VHDL operation was complete. The second task

is to pipeline the MIPS processor. Pipelining is a mandatory feature in RISC processors which technique used to improve both clock speed and overall performance. Pipelining allows a processor to work on different steps of the instruction at the same time more instruction can be executed in a shorter period of time. The data path is divided into different modules, where each module must wait for the previous one to finish before it can execute, thereby completing one instruction in one long clock cycle. When the RISC processoris pipeline, same time a single clock cycle each one of those modules or stages is in use at exactly the same time executing on multiple instructions in parallel. Figure.2 shows an example of a MIPS single-cycle pipelined (a.) versus a MIPS pipelined implementation (b.).

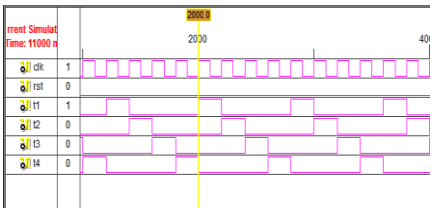

Figure.2 Test Bench

The pipelined implementation is faster. The four modules are Instruction Fetch, Instruction Decode, Execution, and Write Back. The Hardware Discripitive Language design becomes a 2-level hierarchy. The high-level of the hierarchy is a structural VHDL file which connects the all components of the single-cycle implementation, while the lower-level contains the behavioral of VHDL models.

A. INSTRUCTION FETCH UNIT

The aim of this unit is to fetch an instruction from the memory using the current value and increment the next instruction. For getting the instruction memory we has to implement byte addressing to access the register and word address.

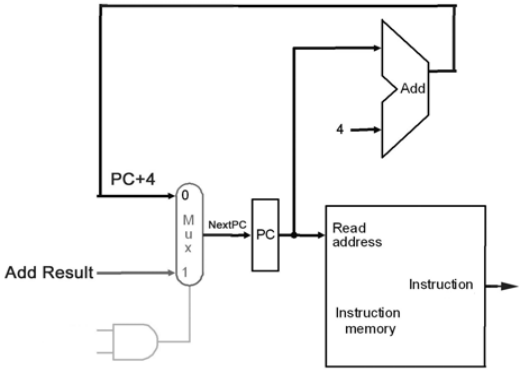

Figure.3 Fetch Unit

B. INSTRUCTION DECODE UNIT

The function of this unit is to use the 8-bit instruction given from the previous instruction. Fetch unit to obtain the register data values.

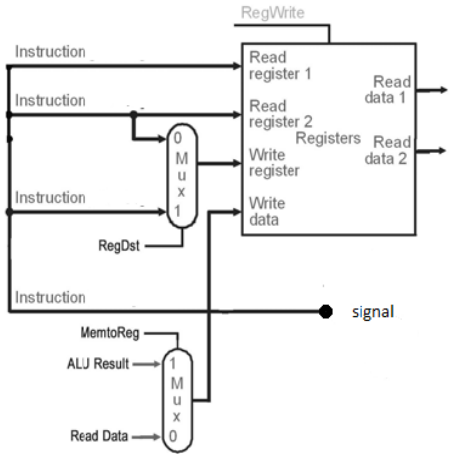

Figure.4 Instruction Decode Unit

However with our design of 8-bit data width, our implementation uses the instruction bits [7 – 0] bits instead of sign extending the value. The logic component to be implemented in VHDL include several multiplexrs and the register file

C. CONTROL UNIT

The control unit of the RISC processorreads the instruction op-code and decodes the instruction to produce 9-control signals to be used in the remaining modules. The RegDst control signal figure out which reg is written to the register file. The function of Jump control signal is to selects the jump address into the PC. The Branch control signal which select the branch address sent into the PC. The Memread control signal is affirm

during a load instruction when the data memory is read to load a register with its memory contents.

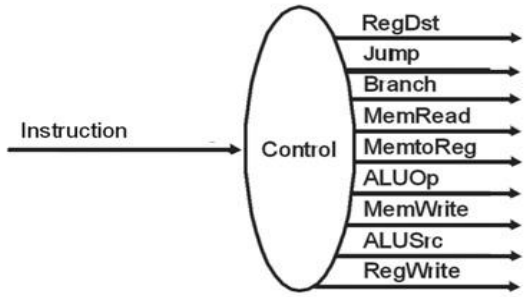

Figure.5 Controller Unit

The Memory to Register control signal check whether the data memory output or ALU result is written to the register file. The ALU Output signals determine the ALU performs. The Memory Write control signal is use when a registers value is saved in the data memory. The ALUSrc control signal determines if the ALU second operand comes from the register file or the sign extend. The RegWrite control signal is asserted when the register file needs to be written.

D. EXECUTION UNIT

The RISC processor contains ALU which execute the operation given by the ALU-output signal. The branch address is compute by W+4 to the sign extended immediate field shifted left by a separate adder.

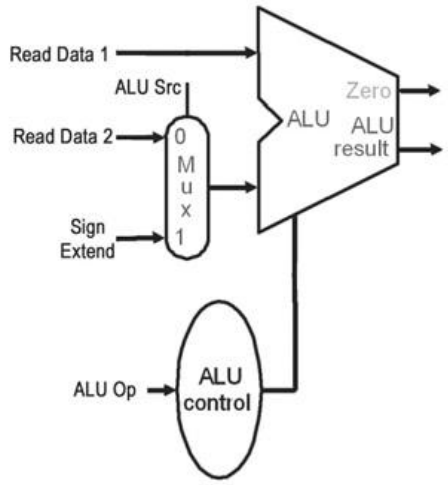

Figure.6 Execution Unit

VI. DATA MEMORY UNIT

The data memory unit is only get by the Von-Neuman architecture. The memory read signal which asserts by memory load instruction and uses the ALU result as an address of data memory. The read output data is write into the register file. A store instruction collects the memory write signal and writes the data value read from a register into the computed memory address. The complete VHDL code used to create the memory state of the MIPS single-cycle processor.

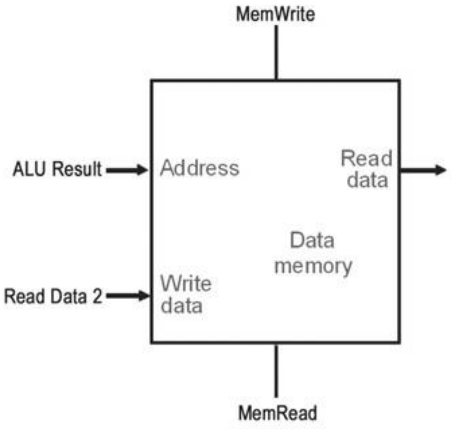

Figure.7 Data Memory Unit

VII. FPGA Module

The Spartan®-3 family of Field-Programmable Gate Arrays is specifically designed to meet the needs of high volume, cost-sensitive consumer electronic applications. The eight-member family offers densities ranging from 50,000 to 5,000,000 system gates, The Spartan-3 family are a superior alternative to mask programmed RSICs. FPGAs avoid the high initial cost, the lengthy development cycles, and the inherent inflexibility of conventional RSICs.

Also, FPGA programmability permits design upgrades in the field with no hardware replacement necessary, an impossibility with RSICs

VII. ADVANTAGES

A. The Processor Achieves Higher Performance,
B. Lower Area,
C. Compact in size.
D. Low power dissipation.
E. Achieve highest system speed .
F. More than 20K times reprogrammable.
G. Fast concurrent programming.
H. Reconfigurable of logic is possible.
I. Reduce serial I/O power ,cost and complexity with the word first 90 nm FPGA's.
J. Easy upgrade to new specifications.

VIII. APPLICATIONS

A. Portable Gaming Kits,
B. A Systolic Core To Perform Mathematical Computations,
C. Solving Polynomial And Differential Equations.

IX. ACKNOWLEDEMENT

"Inspiration and guidance are invaluable aspect for every student in his life". We would also like to take this opportunity to express our honour and respect to express deep sense of regards towards our guide **Dr. S. L. Lahudkar** Sir of Electronics & Telecommunication Department, for his encouragement, excellent guidance and precious suggestion throughout our work and his co-operation for successful completion of this project. Finally, we would thanks to all those who directly or indirectly made a contribution to completion of this project work.

VII. CONCLUSION

RISC Processor has design and implement on Xilinx Spartan 3E FPGA tools. This easily represented in the Xilinx ISE Design Suite. This project is easy to understand. It executes all the instructions in one clock cycle including jumps, returns from subroutines and external accesses.

RISC processor of 8-bit with 35 instruction set has design. Every instruction is executed in one clock cycles with 4-stage pipelining. The processor achieves higher performance lower power dissipation & lower area.

REFERENCES

1. Andrzej Kos, FPGA implementation of RISC Microcontroller‖18th International Conference on MIXDES, pp. 323-238, 16-17 March 2011.
2. Design Processor Using VHDL http://cegt201.bradley.edu/projects/proj2001/vhdlrisc/
3. Gray, J., Designing a Simple FPGA-Optimized RISC CPU and System-on-a-Chip, 2000.
4. IEEE Standard VHDL Language Reference Manual.IEEE, New York, NY, 2002. IEEE Standard 1076-2002.
5. VHDL Programming [Douglas Perry]
6. Brown, Richard, "A Microprocessor Design Project in an Introductory VLSI Course".
7. VHDL Programming http://www.asic-world.com/vhdl/first1.html
8. VHDL source for simple processor http://www.cse.wustl.edu/~dzar/class/260/hw/cpu.html
9. J.L. Hennessy, "VLSI Processor Architecture," IEEE Trans. Computers, vol. C-33, no. 12, Dec. 1984, pp. 1221-1246.