ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 2, Issue4, April 2014

RISC Processor Design in VLSI Technology Using the Pipeline Technique

Rakesh M.R

M.Tech student, Department of ECE, Canara Engineering College, Mangalore, Karnataka, India

Abstract: This paper presents the design and implementation of a pipelined 9-bit RISC Processor. The various blocks include the Fetch, Decode, Execute and Store result to implement 4 stage pipelining. Harvard Architecture is used which has distinct program memory space and data memory space. The only load and store is used to communicate with data memory. RISC using pipeline makes CPI as 1 and improves speed of execution. Verilog Language is used for coding purpose. The proposed architecture is then simulated using Modelsim.

Keywords: RISC features, Pipelining, ALU, Booth multiplier, Barrel shifter

I INTRODUCTION

Reduced Instruction Set Computers (RISCs) are now use for all type of computational tasks. In the area of scientific computing, RISC workstations are being increasingly used for compute task such as DSP, DIP etc. RISC concepts help to achieve given levels of performance at significantly lower cost than other systems. Pipelined RISC improves speed and cost effectiveness over the ease of hardware description language programming and conservation of memory and RISC based designs will continue to grow in speed and ability.

The main features of RISC processor are the instruction set can be hardwired to speed instruction execution. In the present work, the design of a 4-bit data width Reduced Instruction Set Computer (RISC) processor is presented. It has a complete instruction set, program and data memories, general purpose registers and a simple Arithmetical Logical Unit (ALU) for basic operations. In this design, most instructions are of uniform length and similar structure, arithmetic operations are restricted to CPU registers and only separate *load* and *store* instructions access memory. The architecture supports 8 instructions to support Arithmetic, Logical, Shifting, and load -store operations.

Verilog HDL has evolved as a standard hardware description language. A hardware descriptive language is a language used to describe a digital system. HDL's allows the design to be simulated earlier in the design cycle in order to correct errors or experiment with different architectures. Designs described in HDL are technology-independent, easy to design and debug, and are usually more readable than schematics, particularly for large circuits. More recently Verilog is used as an input for synthesis programs which will generate a gate-level description for the circuit. The simulator which is used for the language is Xilinx ISE and Modelsim. Verilog is capable of describing simple behaviour. Machine cycle instructions allow the processor to handle several instructions at the same time. The processor can work at a high clock frequency and thus yields higher speed. This paper is about design of a simple RISC processor and synthesizing it. The RISC architecture follows single-cycle instruction execution.

II. RISC PROCESSOR ARCHITECTURE

RISC architecture has been developed as a result of the 801 project which started in 1975 at the IBM T.J.Watson Research Center and was completed by the early 1980s. RISC architecture starts with a small set of most frequently used instructions which determine the pipeline structure of the machine enabling fast execution of those instructions in one cycle. The IBM was the first company to define the RISC (Reduced Instruction Set Computer) architecture in the 1970s. This research was further developed by the universities of Berkeley and Stanford to give basic architectural models.

A. CISC v/s RISC

Processors have traditionally been designed around two Philosophies: Complex Instruction Set Computer (CISC) and Reduced Instruction Set Computer (RISC).

The CISC concept is an approach to the Instruction Set Architecture (ISA) design that emphasizes doing more with each Instruction using a wide variety of Addressing modes, the Instructions are of widely varying lengths and execution times thus demanding a very complex Control Unit, which tends to large chip area.

On the other hand, the RISC Processor works on reduced number of Instructions, fixed Instruction length, more general-purpose registers, load-store architecture and simplified Addressing modes which makes individual instructions execute faster, achieve a net gain in performance and an overall simpler design with less silicon consumption as compared to CISC.

Nowadays RISC is one of major important device in computer systems because of following points,

- RISC instructions are executed in single clock cycle, while most of CISC requires more than one clock cycle.
- RISC system is more popular.

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 2, Issue4, April 2014

- The technology RISC processors support the floating point data type.
- RISC machines mostly uses hardwired unit.
- RISC machines require lesser time for its design implementation.
- RISC processor consumes less power
- RISC machines use load and store instruction to access data from memory.

B. RISC features

The main features of RISC processor are the instruction set can be improved to speed instruction execution. No microcode is needed for single cycle execution. All instructions are fixed bit in length. This simplifies the instruction fetch mechanism since the location of instruction boundaries is not a function of the instruction type. The processor has small number of addressing modes. Only load and store instructions access memory, load/store instructions operate between memory and a register. The machine cycle time is minimized. The fixed size of the instructions allows the instructions to be easily piped. RISC provides a flexible and expandable architecture that maximizes performance from any given semiconductor technology. RISC includes extensions to RISC concepts that help achieve given levels of performance at significantly lower cost than other systems. The design process is very fast and cost effective. In this design, most instructions are of uniform length and similar structure, arithmetic operations are restricted to CPU registers and only separate *load* and *store* instructions access memory. The Instruction cycle consists of four stages namely fetch, decode, execute and Store. After every instruction fetch, Control Unit generate signals for the selected Instruction. The architecture supports 8 instructions to support Arithmetic, Logical, Shifting and Load-store operations.

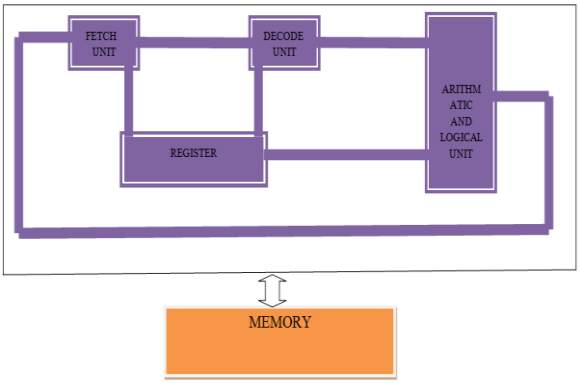

Fig1: Block diagram of simple RISC processor

The Block diagram of a 9 - bit RISC processor is shown in Figure1. The RISC processor architecture consists of Arithmetic Logic Unit (ALU) (including arithmetic and logical units, barrel shifter, booth multiplier), control unit (CU), Memory and Register File. RISC processor is designed with load/store architecture, meaning that all operations are performed on operands held in the processor registers and the main memory can only be accessed through the load and store instructions.

C. Pipelining

Instruction and data are fetched in sequential order so that the latency incurred between the machine cycles can be reduced. For increasing the speed of operation RISC processor is designed with four stage pipelining. The pipelining stages are Instruction Fetch (IF), Instruction Decode (ID), Execution (EX) and Store result (ST).

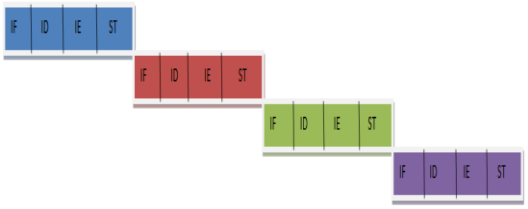

Fig2: instruction execution without pipelining technique

Fig3: instruction execution without pipelining technique

The figure2 shows the execution of instruction without taking concept of the pipeline. Then there will be long time required for processor to execute set of instruction. So CPI will be very high.

The figure3 shows the execution of instruction by using the technique called pipelining. Execution time required for processor can be reduced and speed of execution increases. So CPI can reduce.
Let us take each stage of pipeline technique as,

Instructions fetch cycle (IF):
Send the program counter (PC) to memory and fetch the current instruction from memory. Update the PC to the next sequential PC by adding one to the PC.

Instructions decode/register fetch cycle (ID):
Decode the instruction and read the registers corresponding to register source specifiers from the register file. Decoding is done in parallel with reading registers, which is possible because the register specifiers are at a fixed location in RISC architecture. This technique is known as *fixed-field decoding*.

Execution (EX):
The ALU operates on the operands prepared in the prior cycle, performing one of three functions depending on the instruction type. Register-Register ALU instruction: The ALU performs the operation specified by the ALU opcode on the values read from the register file.

Store result (ST):
Register-Register ALU instruction or Load instruction: Write the result into the register file, whether it comes from the memory system (for a load) or from the ALU (for an ALU instruction).

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 2, Issue4, April 2014

The function of the instruction fetch unit is to obtain an instruction from the instruction memory using the current value of the Program counter (PC) and increment the PC value for the next instruction. The main function of the instruction decode unit is to use the 9-bit instruction provided from the previous instruction fetch unit to index the register file and obtain the register data values. The instructions opcode field bits are sent to a control unit to determine the type of instruction to execute. The type of instruction then determines which control signals are to be set and function that Execute unit is to perform, thus decoding the instruction.

D. Main modules

Let us consider modules of RISC processor. The main parts of processor is shown in Figure4 and are explained bellow

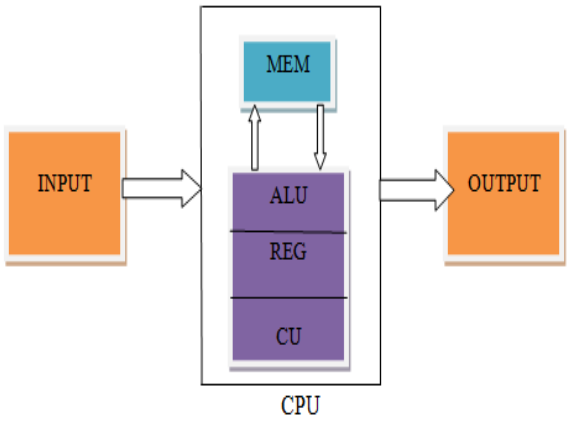

Fig4: Basic main modules of processor

These are following basic main parts of the processor:

Control Unit:

It manages the sequence and timing of events carried out within the processor. The control unit of the RISC processor examines the instruction opcode bits and decodes the instruction to generate eight control signals.

Registers:

Holds values of internal operation, such as the address of the instruction being executed and the data being processed i.e. Program Counter Register, Status Register.

Separate program and data memory:

The program memory also called as ROM which contains instructions of the processor. Each instruction is 9 bit and there are 8 instructions are there.

The data memory which also called as RAM which is temporary memory and used to store the data values need for processor and data values coming from processor after processing.

Load and store:

Processor which communicates with memory only by using load and store instruction. Load instruction which load the data value from memory to register and store instruction which store the value from register to RAM memory.

Arithmetic Logic Unit (ALU):

The arithmetic/logic unit (ALU) executes all arithmetic and logical operations. Arithmetic operations either take two registers as operands. The result is stored in a third register.

The arithmetic/logic unit can perform arithmetic operations or mathematical calculations like addition, and subtraction and also performs logical operations include Boolean comparisons, such as AND, OR, XOR, NAND, NOR and NOT operations.

Barrel Shifter:

A barrel shifter is a digital circuit that can shift a data word by a specified number of bits in one clock cycle. It can be implemented as a sequence of multiplexers and in such an implementation the output of one Mux is connected to the input of the next Mux in a way that depends on the shift distance. For example, take a four-bit barrel shifter, with inputs A, B, C and D. The shifter can cycle the order of the bits *ABCD* as *DABC*, *CDAB*, or *BCDA*; in this case, no bits are lost. That is, it can shift all of the outputs up to three positions to the right. The barrel shifter has a variety of applications, including being a useful component in microprocessors (alongside the ALU. A *barrel shifter* is a combinational logic circuit with *n* data inputs, *n* data outputs, and a set of control inputs that specify how to shift the data between input and output. A barrel shifter that is part of a microprocessor CPU can typically specify the direction of shift, the type of shift and the amount of shift.

Fig5: block diagram of barrel shifter

In this paper the four bit data _in taken with clock and reset inputs. The data_out is the shifting for every clock cycle and it shows the barrel shifter operation. The block diagram of barrel shifter with input and output is shown in Figure5. The simulation result of the barrel shifter is shown in Figure6.

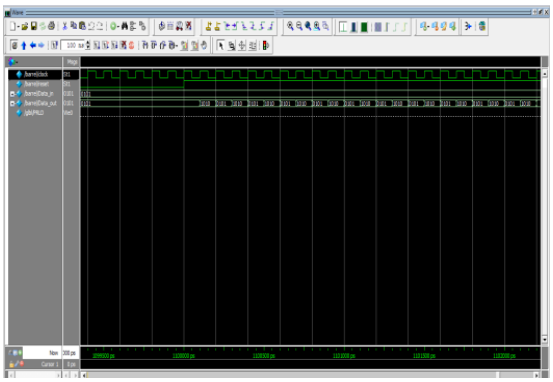

Fig6: simulation waveform for barrel shifter

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 2, Issue4, April 2014

Booth's Multiplier:

Booth's multiplication algorithm is a multiplication algorithm that multiplies two signed binary numbers in two's complement notation. Booth used desk calculators that were faster at shifting than adding and created the algorithm to increase their speed. The area and speed of the multiplier is an important issue, increment in speed results in large area consumption and vice versa. Multipliers play vital role in most of the high performance systems. Performance of a system depends to a great extent on the performance of multiplier thus multipliers should be fast and consume less area and hardware. For this one multiplier is used with Booth's Algorithm. The two main advantages of this algorithm are speed and the ability to do signed multiplication (using two's complement) without any extra conversions.

Fig7: block diagram of booth multiplier

In this paper the four bit input, eight bit output with clock and reset inputs are taken to simulate the algorithm. The block diagram of booth multiplier with input and output is shown in Figure7. The simulation result of booth multiplier is shown in Figure8.

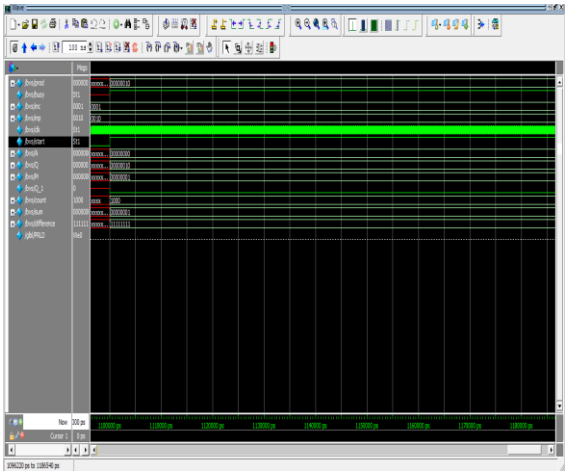

Fig8: simulation waveform for booth multiplier

E. Instruction Set

This processor which having some of instruction set such as: Arithmetic Instruction, Shift Instructions, and Load and store instructions. There are eight instructions each of 9 bit wide. Opcode of 3bit and three operands of each 2 bit. The instruction format is of register based. Addressing mode is register addressing mode. Opcode and corresponding operation of processor is shown in Figure9. Instruction format for Processor is shown in Figure10.

OPERATION		OPCODE
ADD	Addition of two register	000
SUB	Subtraction of two register	001
MUL	Multiplication of two register	010
INC	Increment of two register	011
DEC	Decrement of two register	100
SR	Register value Shift right	101
LD	Load from memory to register	110
ST	store to memory from register	111

Fig9: opcode and corresponding operation of processor

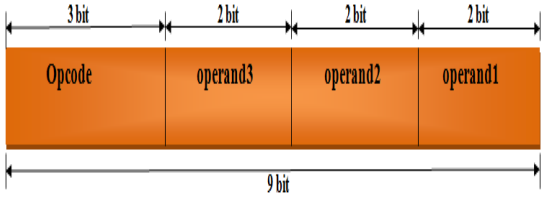

Fig10: instruction format for Processor

III. SIMULATION RESULT

The simulation of RISC processor is done by using the Modelsim simulator, designed using Verilog HDL in XILINX ISE. The RTL schematic for single RISC processor is shown in Figure11. The Technological schematic for single RISC processor is shown in Figure12.The simulation result of RISC processor which having 8 instructions of 9 bit each. Instead of common shifter and multiplier here we used the Barrel shifter and booth multiplier respectively. The simulation waveform is getting by using Modelsim simulator which consists of 8 instructions including barrel shifter and booth multiplier is shown in Figure 13.

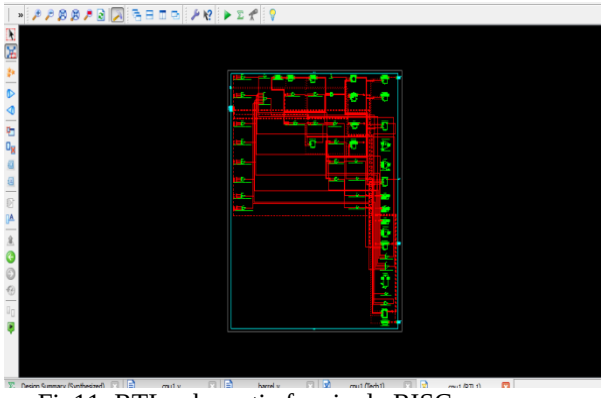

Fig11: RTL schematic for single RISC processor

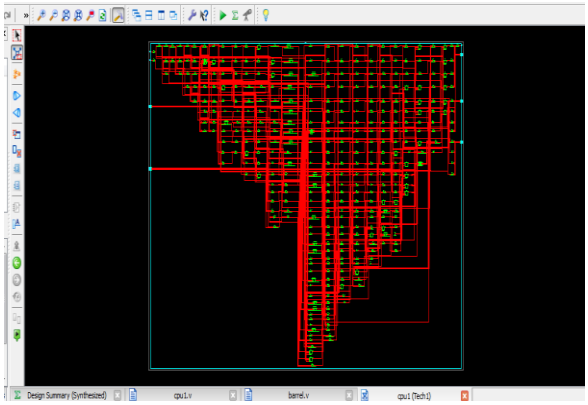

Fig12: Technology schematic for single RISC processor

ISSN (Online) 2321 – 2004
ISSN (Print) 2321 – 5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 2, Issue4, April 2014

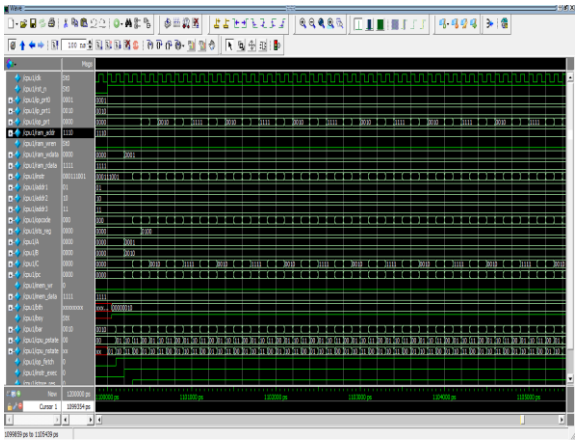

Fig13: waveform for single RISC processor

IV. CONCLUSION

In this paper the 9 bit RISC Processor core has been design and simulated in Xilinx ise13.1. The design has been achieved using Verilog and simulated with Modelsim simulator. Most of the goals were achieved and simulation shows that the processor is working perfectly. Every instruction is executed in one clock cycles with 4-stage pipelining. The design is verified through simulations.

REFERENCES

[1] R. Uma, Mar-Apr 2012, "Design and Performance Analysis of 8-bit RISC Processor using Xilinx Tool"

[2] Galani Tina G., Riya Saini and R.D.Daruwala, July-2013, "Design and Implementation of 32 – bit RISC Processor using Xilinx"

[3] Galani Tina R.D.Daruwala, February 2013, " Performance Improvement of MIPS Architecture by Adding New Features"

[4] Samiappa Sakthikumaran,S.Salivahanan and V.S.Kaanchana Bhaaskaran , June 2011, "16-Bit RISC Processor Design For Convolution Application",IEEE International Conference on Recent Trends In Information Technology, pp.394-397.

[5] Rohit Sharma, Vivek Kumar Sehgal, Nitin Nitin1, Pranav Bhasker, Ishita Verma , 2009, "Design And Implementation Of 64-Bit RISC Processor Using VHDL",UKSim : 11th International Conference on Computer Modeling And Simulation, pp. 568 – 573.

[6] Rupali S. Balpande and Rashmi S. Keote.2011, "Design of FPGA based Instruction Fetch & Decode Module of 32-bit RISC (MIPS) Processor, International Conference on Communication Systems and Network Technologies pp. 409 – 413

[7] Xiao Li, Longwei Ji, Bo Shen, Wenhong Li, Qianling Zhang, "VLSI implementation of a High-performance 32-bit RISC Microprocessor", Communications, Circuits and Systems and West Sino Expositions, IEEE 2002 International Conference on ,Volume 2, 2002 ,pp.1458 – 1461.

[8] Kusumlata Pisda, Deependra Pandey, "Realization & Study of High Performance MIPS RISC Processor Design Using VHDL", International Journal of Emerging trends in Engineering and Development, Volume 7, Issue 2, November 2012, pp. 134 – 139, ISSN: 2249 – 6149.

[9] Kirat Pal Singh, Shivani Parmar, "VHDL Implementation of a MIPS – 32 bit Pipeline Processor", International Journal of Applied Engineering Research, Volume 7, Issue 11, ISSN: 0973 – 4562.

BIOGRAPHY

Mr. Rakesh M.R received his B.E degree in Electronics and Communication from KVG College of Engineering Sullia in 2012. Currently he is pursuing M.Tech degree in Electronics at Canara Engineering College, Mangalore. His areas of interest are VLSI and Image Processing.