International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 09 | Sep 2021 www.irjet.net p-ISSN: 2395-0072

Realization of 8-Bit Pipelined RISC Processor using Verilog HDL

Ravi Hosamani[1] , Rakesh H. M. [2] , Rakesh B Shettar[3], Praveen Kumar Y G[4]

[1,2,3]Asst. Professor, Dept. of Electronics & Communication Engineering, K.L.E Institute of Technology, Hubballi, Karnataka, India

[4]Asst. Professor Dept. of ECE, Sri Siddhartha Institute of Technology, SSAHE, Tumkuru, India

---

Abstract - In many cases of processing units, the processor plays a critical role. A RISC processor is a type of central processing unit (CPU) that customises general instructions for improved performance and faster execution. It is cost-effective to design, test, and produce. This has aided in the technological field's implementation of RISC processors. Signal processing, convolutional applications, supercomputers such as K computers, and a broader base for smartphones are among its applications. This paper presents an 8-bit RISC processor with a pipeline, with the primary goal of improving performance and efficiency. The control unit, general-purpose registers, the arithmetic and logical unit, and shift registers are all part of it. Individual modules, such as general-purpose registers, ALUs, control units, memory units, and programme counters, are developed and tested using Verilog HDL. Finally, the process is created by putting together various components and then testing it through extensive simulation. Finally processed is developed by assembling individual modules and tested by exhaustive simulation.

Key Words: Central Processing Unit(CPU), Pipeline, RISC, Verilog.

1. INTRODUCTION

Over the last few decades, the reconfigurable processor's role in embedded system design has significantly improved. The advancement of the field-programmable gate array (FPGA) has made it possible to change or modify the processor architecture in real time. This enhanced feature minimises the cost each logic cell developed intentionally. The use of RISC processors is increasing across the board, thanks to significant advancements in silicon technology and falling integrated circuit costs. Only special load and store procedures are allowed on RISC CPUs. The remaining operations are carried out on a register-to-register basis. Because it permits instructions to be executed at a one-instruction-per-cycle rate, this makes the instruction set design straightforward and simple. Simple and transparent addressing modes allow for quick operand address computation. Signal processing, convolutional applications, commercial data processing, and real-time embedded systems are among the applications for RISC processors. Pipelining is a method of programming in which many instructions are performed at the same time. It's similar to parallelism on a single processor at the instruction level. Pipelining makes the proposed processor's architecture more efficient when tested. [1][2].

The suggested RISC architecture employs the clock gating approach. Clock power is an important part of overall dynamic power that must be addressed in order to reduce power usage. Clock-gating (ANDing) is accomplished by using Transfer Level (RTL), which dynamically terminates clock signals in unused modules of the entire hardware. This eliminates the waste of power caused by charging and discharging the clock signal at the unused gate. [5][6].

Verilog Hardware Description Language is used to create an 8-bit RISC CPU. The proposed processor is built on the Harvard architecture and features separate instruction and data memory. It includes a set of instructions that are straightforward, easy to understand, and concise. RISC architecture is required to reduce the complexity of CISC processor design. Pipelining is required to boost an ALU's speed by increasing instruction flow. [7].

The salient feature of the proposed processor is pipelining and clock gating. Pipelining helps in improved performance by executing each instruction at every clock cycle and clock gating helps in reduced power dissipation.

Jikku Jeemon (2015), An FPGA-based 8-bit RISC processor with pipelining and clock gating is designed in this study. All 29 instructions are confirmed on the Xilinx Spartan-6 SP605 Evaluation Platform, where the design is implemented. Because the processor only executes one instruction each clock cycle, the pipelining technique improves performance. [1].

Akshata S Patil, B G Shivaleelavathi (2017), in this paper 8-bit RISC processor is designed using Verilog HDL on an FPGA board. The processor can be very compact and, simple and clean to investigate and contains 24 instructions[2].

Aishwarya S H, Sujata Hiremath (2020), in this paper an 8-bit Reduced Instruction Set Computer (RISC) is Designed and Implemented. The Processor can execute with simplified Design for a greater number of instructions and obtain less path delay. RISC processors have a very large range of applications based on power consumption and speed[3].

2. METHODOLOGY

The designing of 8-bit Pipelined RISC Processor using Verilog HDL is carried out in this Section
Fig.1 shows the block diagram of the 8-Bit RISC Processor architecture. The 8-bit processor consists of the following blocks.

- A.L.U: The Arithmetic Logic Unit is connected to the Accumulator, ALU performs the operations such as AND, OR, ADD, XOR, SUB.
- Accumulator: It is an 8-bit register, ALU, I/O, increment, decrement, complement, rotates right and left operations take place through it.
- Program Counter Unit: It is an 18-bit wide register that contains the address of the instruction being executed at the current time.
- Control Unit: It takes the input and source clock of the FPGA and generates 56 control signals and 3 clock signals for the proper working of all the modules.
- Instruction and Data Memory: Instruction memory is 16-bit and the data memory is 8-bit which provides read and write control.
- Register Set: It contains 8-bit registers R0, R1…. R7, which can be used for storing data that is frequently used.
- Interrupt Module: It contains two external interrupts I0 and I1. I0 has higher priority and I1 has lower priority.

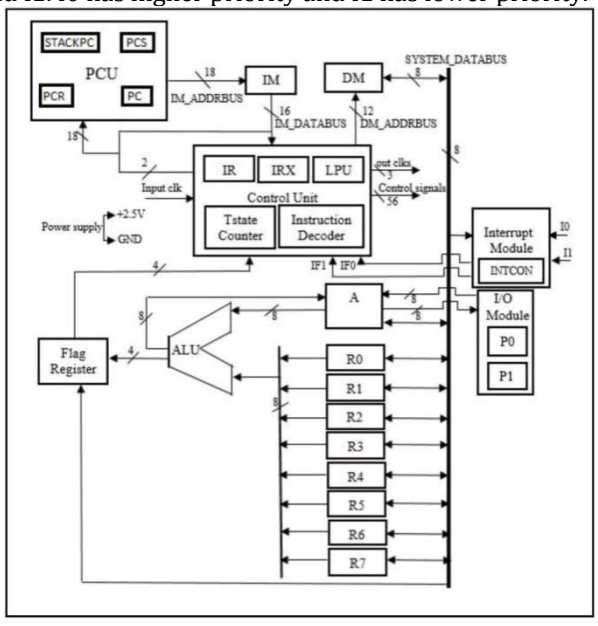

Fig.1 Proposed Architecture

It consists of the following functional units –

1) **Arithmetic and logic unit**
As the name suggests, it performs arithmetic and logical operations like Addition, Subtraction, AND, OR, etc. on 8-bit data.

2) **General-purpose register**
There are 4 general purpose registers in this processor, i.e. R0, R1, R2, R3. Each register can hold 8-bit data.

3) **Program counter**
It is an 8-bit register that stores the memory address of the next to be executed instruction. When an instruction is executed, the microprocessor increments the programme so that it counterpoints to the memory address of the next instruction to be executed.

4) **Temporary registers**

These are 8-bit register OP1 and OP2, which holds the temporary data of arithmetic and logical operations.

5) **Instruction register and decoder**

It is a 16-bit register. When an instruction is fetched from memory then it is stored in the Instruction register. The instruction decoder in Control Unit decodes the information present in the Instruction register.

6) **Timing and control unit**

It gives the microprocessor a timing and control signal so that it can perform operations. The timing and control signals that control external and internal circuitry are listed next. READ, WRITE, and ALU are the control signals.

7) **Address bus and data bus**

The data bus is a bidirectional bus that carries the operands for different operations whereas the address bus is a unidirectional bus that carries the location to be accessed. It is used to transfer the data & Address I/O devices.HDLs have numerous pluses compared to customary schematic-based designs. Designs can be defined at a very abstract level by the use of HDLs. RTL description can be done by the designers without choosing a specific fabrication technology. CAD tools like logic synthesis tools can be used for converting the design to any fabrication technology. If a new technology emerges, designers need not remodel their circuits. They simply input the RTL description to the logic synthesis tool and create a new gate-level netlist. The logic synthesis tool also optimizes the circuit in area and timing for the new technology. Any Verilog program begins with a keyword–called a “module.” A module is a name given to any system considering it as a black box with input and output terminals as shown in Fig. 2. The terminals of the module are referred to as ‘ports’. The ports attached to a module can be of three types:

- **Input ports** through which one gets entry into the module; they signify the input signal terminals of the module. x
- **Output ports** through which one exits the module; these signify the output signal terminals of the module.

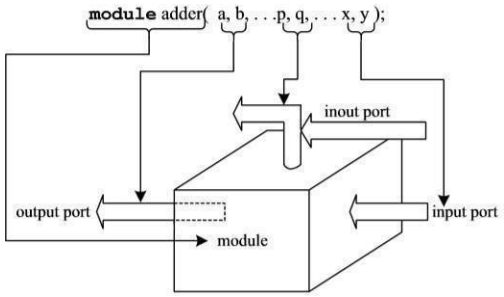

Fig .2 black box module with ports

- **Inout ports:** These are terminals through which signals are inputted to the module sometimes; at some other times signals are outputted from the module[2].

3. RESULTS

No Operation (NOP) instruction, the memory is provided with the binary equivalent of the instruction. Here, the binary equivalent NOP which is 0000 given. The output is as shown in fig.3, as it is no operation the fetch for the next instruction is high, and the program counterpoints to the next instruction.

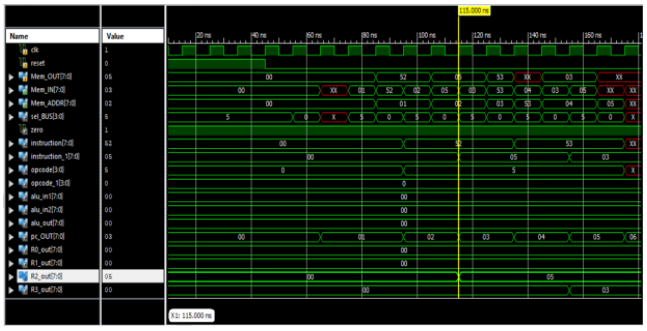

Fig.3 Results of NOP(0000) instruction

For, Subtract Operation (SUB) instruction, the memory is provided with the binary equivalent of the instruction. Here, the binary equivalent SUB which is 0010 given. The output is as shown in fig. 4, the operands are stored in two registers and are subtracted.

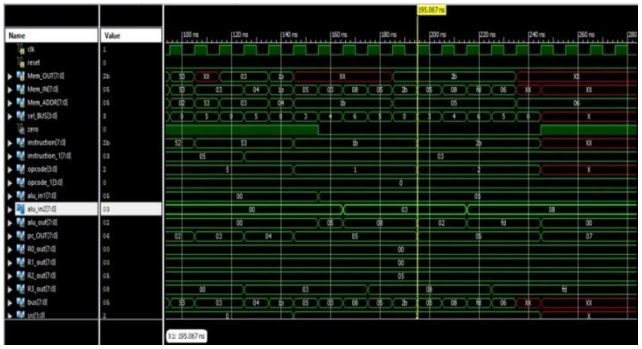

Fig. 4 Results of SUB (0010)instruction

For, Add Operation (ADD) instruction, the memory is provided with the binary equivalent of the instruction. Here, the binary equivalent ADD which is 0001 given. The output is as shown below, the operands are stored in two registers and are added. The below shown output is obtained with pipelining by giving multiple inputs in the memory file for getting output with pipelining. The Results are shown in fig. 5.

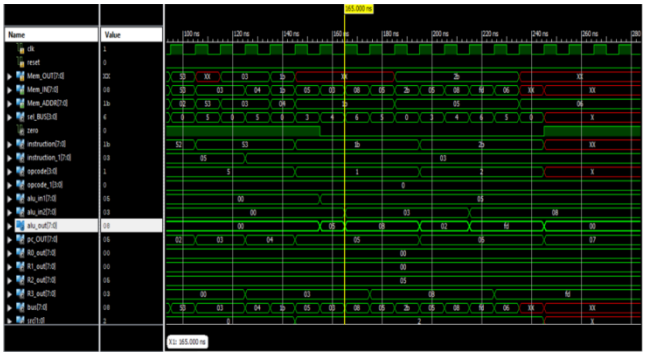

Fig. 5 Operation ADD (0001) instruction

For, AND Operation instruction, the register R2 is given with input 0101, this binary value is stored in register R2 and the next fetch is high to fetch the next instruction in the memory, that is to store 0011 in R2. Then the instruction is passed with operands R2 and R3. The output also shows pipelining as mentioned in the previous test case but with different inputs given. The output is shown below.

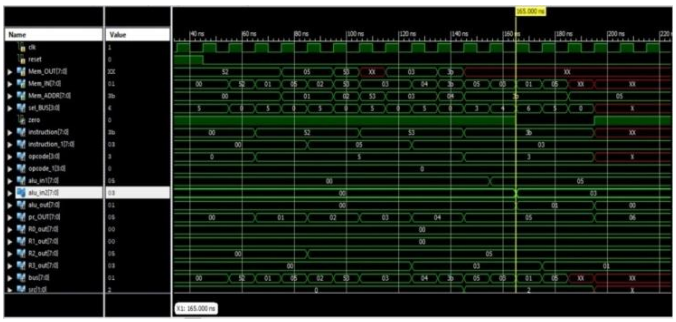

Fig. 6 AND Operation (0011) instruction

The design is tested for proper working by giving input as a series of various instructions to check the pipelining. Here the output for the instructions is shown in the below diagram.

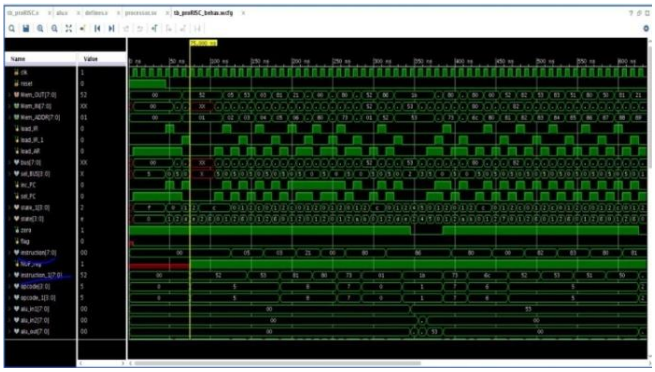

Fig.7. Result of various instructions in the pipeline

4. CONCLUSIONS

Signal processing, convolution applications, commercial data processing, smartphones, tablets, and real-time embedded systems just are a few of the uses for RISC processors. Verilog HDL is used to create an 8-bit RISC CPU. The Xilinx ISE 14.7 suite is used to verify the submitted design. Extensive simulation has been used to verify the results. The majority of the formatting instructions in this document were successfully completed. The design appears to be working properly, as evidenced by the results. This processor can be utilised in pocket calculators, gaming toolkits, and signal processors to perform mathematical computations. Wide range processors, such as 16-bit and 32-bit, will be used for a wider range of applications.A few more features can be added to this processor in the future.

REFERENCES

[1] JikkuJeemon, "Low Powered Pipelined 8-Bit RISC Processor Design and Implementation on FPGA", International Conference on Control Instrumentations, Communication and Computational Technologies (ICCICCT), pp.476-481, 2015.

[2] Akshatha S Patil, B G Shivaleelavathi, "Design and Implementation of Pipelined 8bit RISC Processor using Verilog HDL on FPGA", International Research journal of Engineering and Technology (IRJET), 2017, Vol.4, Issue 6, pp.1753-1756, June 2017.

[3] Aishwarya H S, SujathaHiremath, "Design of Low Power High Speed 8-bit RISC Processor", International Research journal of Engineering and Technology (IRJET), Vol. 7, Issue 6, pp.1051-1056, June 2020.

[4] RamandeepKaur, Anuj, "8 Bit RISC Processor Using Verilog HDL", Anuj et al Int. journal of Engineering Research and Applications, Vol. 4, Issue 3, pp.417422, March 2014.

[5] R. Uma, "Design and Performance Analysis of 8-bit RISC Processor using Xilinx Tool", International Journal of Engineering Research and Applications (IJERA), Vol.2, Issue 2, pp.053-058, Mar-Apr 2012. V. P. Veiko, Laser Assisted Microtechnology, 2nd ed., R. M. Osgood, Jr., Ed. Berlin, Germany: Springer-Verlag, 1998.

[6] R. Uma."Design and Performance Analysis of 8-bit RISC Processor using Xilinx Tool,"International Journal of Engineering.

[7] SeungPyoJung, JingzheXu, Donghoon Lee, Ju Sung Park, 2008, "Design And Verification Of 16 Bit RISC Processor", International SOC Design Conference.

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

[9] Pravin S. Mane, Indra Gupta, M. K. Vasantha, Implementation of RISC Processor on FPGA, 1-4244-0726-5/06, 2006 IEEE.

[10] Charles H. Roth, Jr., "Digital Systems Design using VHDL", The University of Texas at Austin. 2006 reprint, Thomson Asia Pte Ltd, Singapore

[11] Nazeih M. Botros, "HDL Programming VHDL and Verilog", 2009 reprint, Dreamtech press

[12] Kevin Skahill, "VHDL for Programmable Logic", Pearson education, 2006

[13] "Design Hubs", https://www.xilinx.com/support/documentation-navigation/design-hubs/vivado-design-suite.html