International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2015): 78.96 | Impact Factor (2015): 6.391

Design of Advanced 64-Bit RISC Processor using Verilog HDL

P. N. B. Harika

M.Tech.(VLSI & Embedded Systems), Chirala Engineering College Affiliated to JNTU, Kakinada

Abstract: ***The Reduced Instruction Set Computer or RISC is a microprocessor design principle that favors a smaller and simpler set of instructions that all take same amount of time to execute. RISC architecture is used across a wide range of platforms from cellular phones to super-computers. In this paper the behavioral design and functional characteristics of 64- bit RISC processor is proposed, which utilizes minimum functional units without compromising in performance. The instruction word length is 17-bit wide. The processor supports 23 instructions. It has 12 general purpose registers. Each register can store 64-bit data. The processor has 64-bit ALU capable of performing arithmetical and logical operations. The processor also incorporates a flag register which indicates carry, zero and parity status of the result. All the modules in the design are coded in verilog. The design entry and synthesis is done using Xilinx ISE 14.1 tool.***

Keywords: RISC, Instruction set, ALU, Verilog, Xilinx ISE 13.2 tool.

1. Introduction

RISC-The reduced instruction set computer, or 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. The most common RISC microprocessors are ARM, DEC Alpha, PA-RISC, SPARC, MIPS, and IBM's PowerPC. The idea was inspired by the discovery that many of the features that were included in traditional CPU designs to facilitate coding were being ignored by the programs that were running on them. Also these more complex features took several processor cycles to be performed. Additionally, the performance gap between the processor and main memory was increasing. This led to a number of techniques to streamline processing within the CPU, while at the same time attempting to reduce the total number of memory accesses. When the controller design become more complex in CISC and the performance was also not up to expectations, people started looking on some other alternatives. It had been found that when a processor talks to the memory the speed gets killed.

So the one improvement on CPI was to keep the instruction set very simple. Simple in not the way it works but the way it looks. That's why we have very few instructions in any typical RISC architecture where processor asks data from memory probably not other than Load and Store. We avoid keeping such addressing modes. The Complexity of controller design has been overcome with the help of operands and op-code bits fixed in instruction register. At the end the 4 stage pipelining added a new dimension in the speed just with the help of some additional registers. Now what pipeline does is it increases throughput by reducing CPI. The instruction can be executed effectively in one clock cycle. The pipelining in any kind of architecture took birth from the inherent parallelism and the idle states of components.

Features which are generally found in RISC designs are:

Over many years, RISC instruction sets have tended to grow in size. Thus, some have started using the term "load-store" to describe RISC processors, since this is the key element of all such designs. Instead of the CPU itself handling many addressing modes, load-store architecture uses a separate unit dedicated to handling very simple forms of load and store operations. CISC processors are then termed "register-memory" or "memory-memory". Today RISC CPUs (and microcontrollers) represent the vast majority of all CPUs in use. The RISC design technique offers power in even small sizes, and thus has come to completely dominate the market for low-power "embedded" CPUs. Embedded CPUs are by far the largest market for processors. RISC had also completely taken over the market for larger workstations for much of the 90s. After the release of the Sun SPARC station the other vendors rushed to compete with RISC based solutions of their own. Even the mainframe world is now completely RISC base.

2. Proposed Processor Architecture

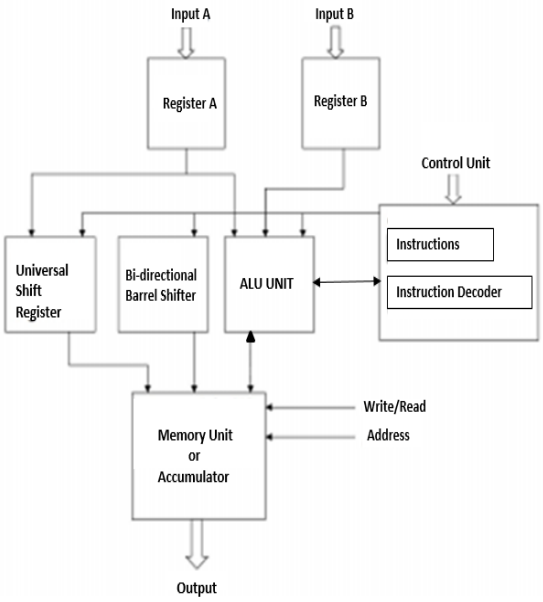


Figure 1: 64-Bit processor architecture

The architecture of 64 bit RISC processor has been shown in the figure 1. It comprises of Control unit, general purpose

register, ALU, Barrel shifter, universal shift register and accumulator. The control unit consists of two registers i.e. instruction register and instruction decoder. Instruction and data are fetched sequentially in order to reduce the latency in the machine cycle. Pipeline structure has been incorporated that further utilizes three execution cycle fetch, decode and execute. This pipeline structure helps in enhancing the speed of operation. In fetch cycle, instruction and relevant data are inferred from the memory while in decode cycle, instruction and data drawn from the memory are bifurcated to activate component and data path for execution and in the execution cycle instruction is executed, data is manipulated and result is stored in the accumulator.

The control unit accepts the opcode and generate the signal that triggers the components and data path to work accordingly and perform the desired function. The control unit has two instruction decoders. These two decoders decode the instruction bits and direct the signal to either into ALU, universal shift register or barrel shift rotator. The operands are received from register A or register B. Upon receiving the operands from registers and the decoded instruction bits arithmetic and logical unit perform arithmetic and logical functions. Universal shift register and barrel shift rotator receives the input from register A and depending upon the decoded information perform the desired operation of either shifting or rotation and the result is stored in the accumulator register.

3. Bidirectioanl Barrel Shifter

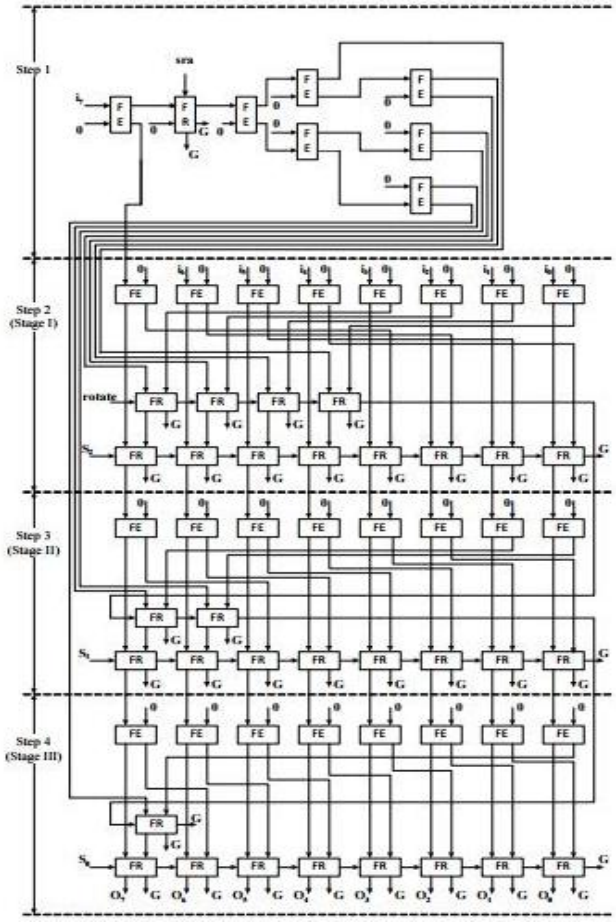


Figure 2: Bi-directional Barrel Shifter

A reversible bidirectional arithmetic and logical shifter is capable can perform logical right shifting, arithmetic right shifting, logical left shifting and arithmetic left shifting operations. The barrel shifter performs the various operations such as logical right shift, logical left shift etc. depending on the values of sra, sla and lef t control signals. In the proposed design, the input data is represented as i7, i6, i5, i4, i3, i2, i1, i0 while the shift value is controlled by select signals represented as S2S1S0.

The design of a reversible arithmetic and logical shifter can be divided into four modules:

(i) Data reversal control unit-I, (ii) Arithmetic right shift control unit, (iii) Shifter unit which consists of three sub-modules that performs Stage I, Stage II and Stage III operations discussed later, (iii) Arithmetic left shift control unit, (iv) Data reversal control unit-II. The reversible design of the modules of the reversible bidirectional arithmetic and logical shifter along with their working are explained as follows:

Table: Operation performed by a (n,k) reversible bidirectional arithmetic and logical shifter

The functional verification of all the proposed reversible design methodologies for the barrel shifters are done in Verilog HDL.

Operation performed	Control signal values		
Logical right shift	sra=0	sla=0	left=0
Arithmetic right shift	sra=1	sla=0	left=0
Logical left shift	sra=0	sla=0	left=1
Arithmetic left shift	sra=0	sla=1	left=1

4. Pipelining

Pipelined architectures have advantages over conventional processor design.

Throughput of the processor can increase up to 270%. The cost effectivity is enhanced by using a single set of peripheral.

In this paper, using a four-stage pipelining in the architecture of a contemporary processor, the design illustrates a 30% decrease in output of a single task when four tasks are concurrently executed.

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).

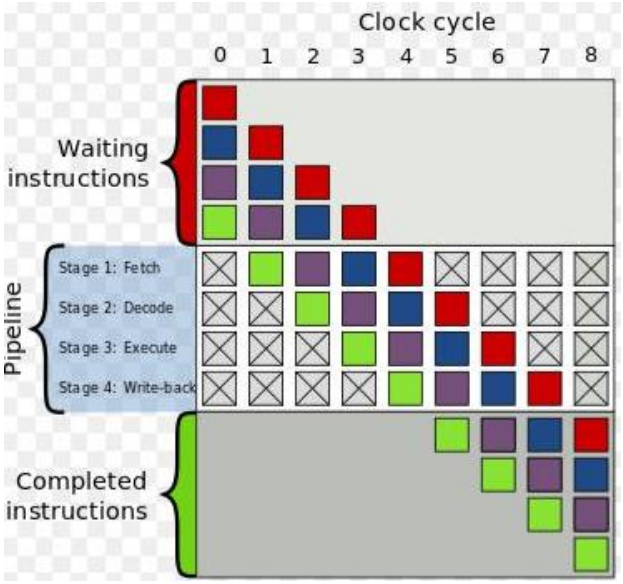
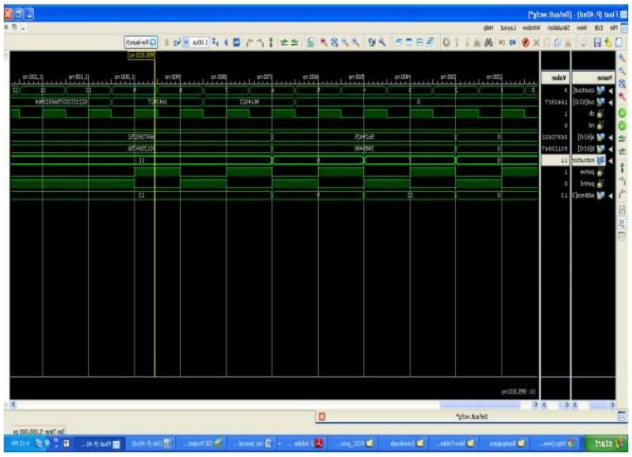


Figure 4: Stage PIPELINING

5. Expected Simulation and Synthesis Results



6. Power Analysis

Whereas power consumption of view this RISC Processor consumes 0.177mW within the frequency range of 1000MHz.

7. Conclusion

A 64 bit RISC processor with 16 instruction set has been designed. Every instruction has been executed in one clock cycle with 3 stage pipelining. Verification has been endeavored by exhaustive simulation. Power consumption also reduced by using clocking techniques, the processor can be used for mathematical computation in portable calculators as well as in gaming tool kit, signal processors, vending machines, etc. Here arithmetic, logical and shifting programs are created in verilog using Xilinx 13.2 Software. Simulated output results are compared with the expected results which we considered at the timing of design.

References

[1] B. Rajesh Kumar, Ravisaketh and Santha Kumar, 2014, “Implementation of A 16-bit RISC Processor for Convolution Application”, Research India publications, pp 441-446.
[2] Aneesh, R.; Jiju, K. "Design of FPGA based 8-bit RISC controller IP core using VHDL", India Conference (INDICON), 2012 Annual IEEE, On page(s): 427 - 432
[3] Ferdous, T. "Design and FPGA-based implementation of a high performance 32-bit DSP processor", Computer and Information Technology (ICCIT), 2012 15th International Conference, on page(s): 484 – 489
[4] R Uma, Design and performance analysis of 8 bit RISC processor using Xilinx tools, March-April 2012, ”, International Journal of Engineering Research and Applications (IJERA), pp 053-058.
[5] MD.Shabeena Begum, M.Kishore Kumar, “FPGA based implementation of 32 bit risc processor”, International Journal of Engineering Research and Applications (IJERA), pp 1148-1151
[6] Mr. Sagar P. Ritpurkar, Prof. Mangesh N. Thakare, Prof. Girish D. Korde ,” Review on 32-bit MIPS RISC Processor using VHDL”, IOSR Journal of Electrical and Electronics Engineering (IOSR-JEEE), PP 46-50
[7] Anjana R & Krunal Gandhi, “VHDL Implementation of a MIPS RISC Processor”, August 2012, International Journal of Advanced Research in Computer Science and Software Engineering, pp 83-88