2nd International Online Conference on Advanced Research in Biology, Engineering, Science and Technology (ICARBEST'16)
Organized by
International Journal of Advanced Research in Biology, Engineering, Science and Technology (IJARBEST)
19th February 2016

FPGA IMPLEMENTATION OF LOW POWER 32 BIT RISC PROCESSOR

Niveathasaro v[1], Vijayasaro V[2]
[1] niveathasaro@gmail.com, [2] viji.saro17@gmail.com
[1] PG scholar, ECE Department, JJ college of engineering and Technology, Thiruchirapalli.
[2] PHD Scholar, Prist University, Thanjavur.

ABSTRACT

In this paper concerned with the design and implementation of a low power 32bit Reduced Instruction Set Computer (RISC) processor on a Field Programmable Gate Arrays (FPGAs). The processor has been designed with Verilog HDL, synthesized using Xilinx ISE 9.1i Web pack, simulated using ModelSim simulator and some components have been implemented and tested on Xilinx FPGA. The test bench waveforms for the different parts of the RISC processor are presented in it.

INDEX TERMS: RISC Processor, Modelsim, Xilinix, FPGA.

I INTRODUCTION

Computer Engineering and Computer Design are very much concerned with the cost and performance of components in the implementation domain. Reduced Instruction Set Computer (RISC) focuses on reducing the number and complexity of instructions in the machine. The RISC concept first originated in the early 1970's when an IBM research team proved that 20% of instruction did 80% of the work. The RISC architecture follows the philosophy that one instruction should be performed every clock cycle.

There is also another kind of processor architecture called CISC (Complex Instruction set Computing), which is having many instructions (nearly 100) and used to perform more complex operations using less instructions than the RISC processor. To reduce the number of accesses to main memory, designers added instruction and data cache to the processors. A cache is a special type of high speed RAM where data and the address of the data is stored. Whenever the processor tries to read data from main memory, the cache is examined first. Cache is commonly ten times faster than main memory, so you can see the advantage of getting data in 10 nanoseconds instead of 60 nanoseconds. Only when we miss (i.e., do not find the required data in the cache), does it take the full access time of 60 nanoseconds. But this can only happen once. Since a copy of the new data is written into the cache after a miss. The data will be there the next time we need it. Instruction cache is used to store frequently used instructions. Implementing fewer instructions and addressing modes on silicon reduces the complexity of the instruction decoder, the addressing logic, and the execution unit. This allows the machine to be clocked at a faster speed, since less work needs to be done each clock period.

RISC typically has large set of registers. The number of registers available in a processor can affect performance the same way a memory access does. A complex calculation may require the use of several data values. If the data values all reside in memory during the calculations, many memory accesses must be used to utilize them. If the data values are stored in the internal registers of the processor instead, their access during calculations will be much faster. It is good then to have lot of internal registers.

Architectural design of a 32-bit floating point RISC processor from Specifications.
1.Behavioral modeling of Design blocks.
2.Design of stimulus modules to test the functionality of Design blocks.
3. Synthesize design to extract Gate level net list.

This RISC Processor is designed to incorporate 20 basic instructions involving Arithmetic, Logical, Data Transfer, Control instructions and Floating Point arithmetic operations. To implement these instructions the design incorporates various design blocks like Control Logic Unit (CLU), Arithmetic Logic Unit (ALU), Multiplexer, Accumulator, Program Counter (PC), Instruction Register (IR), Memory, Clock Generator, Resetter and additional glue logic like buffer, OR gate.The Instruction format contains first five MSB's as OPCODE and remaining 27 bits as ADDRESS BUS. The paper involves design of a simple RISC Processor and simulating it. A Reduced Instruction Set Computer (RISC) is a microprocessor that has been designed to perform a small set of instructions, with the aim of increasing the overall speed of the processor and consumes low power of the processor.

II. LITERATURE SURVEY

1. Michacl slater "Microprocessor-Based Design (a comprehensive guide to effective hardware design)" 2001 pp.55

National semi-conductor's 32000 family was the first to be designed from the start as a 32 bit Microprocessor and is popular in high performance application.The Intel and Motorola families dominate and are available from several alternate source.These 16 and 32 bit microprocessor include multiply and divide instruction and a variety of addressing mode and datatype.Address bus with typically is 20-24 bit,Yielding on effective addressing range of 1-16 Mega Byte(MB).

2. Douglas V.HALL "Micrprocessor and Interfacing (Programming and Hardware) 2005 pp.547

The Intel 80386 has a 32 bit ALU so it can operate on 32 bit data words, program can have as many as 16,384 segment the virtual address space then is 16,384 segment X 4 Giga Byte(GB) are about 64 bytes Tera Byte(TB).A 32 bit address bus allows an 8036 to address upto Giga Byte(GB) of physical memory.The 80386 has Intel 8086 mode.

. The 80386 processor is available in two different version the 386DX and the 386SX has a 32 bit address bus and data bus. It is package in the 132 pin ceromic pin grid array package.The 386SX, which is package in the 100pin flat pack has same internal architecture as the 386DX.But has the only a 24 bit address bus and a 16 bit data bus the lower cost package of the case of interfacing to 8 and 16 bit memory and peripherals mode. The 386SX sutiable for using lower cost system.

3. R.Radhakrishnan "Microprocessor and Microcontroller architecture,program and application"2007 pp.397

In the Pentium processor architecture has a some of major characteristics it is a super scalar and the pipelined are also the super pipelined architecture and the enhanced speed.

In the Pentium chip floating point access unit and performance wise also it is high. It generates 13w power of the output and it can separates the 2 cache memories.

4. HOSHI P.Mistry "Microprocessor-1: Way Through 8085 " 2003 pp.143-147

From this book we referred the ALU, program counter, Accumulator, instruction register and also some main parts of the Architecture pp.8-11.In also the data transfer instruction are the move,store,load pp.44-45

5. Luker, Jarrod D., Prasad, Vinod B., "RISC system design in an FPGA", 2001 pp.186-188

In this paper, we implement the 32-Bit low power RISC processor . which has been fabricated as part of a processor, consists of a typical RISC-CPU, Two simple special-purpose processing registers are to performing the RISC processor. This paper has to employing a large number of simple general purpose processors, or in any other embedded system or verilog.

6. Nikos A. Nikolaou, Dionisios N. Pnevmatikatos Article: "Microprocessors and Microsystems", 2001 pp.123-125

In order to facilitate the implementation of most instruction as register-to register operations, a sufficient amount of CPU general purpose registers has to be provided. A sufficiently large register set will permit temporary storage of intermediate results, needed as operands in subsequent operations, in the CPU register file.

III. BLOCK DIAGRAM

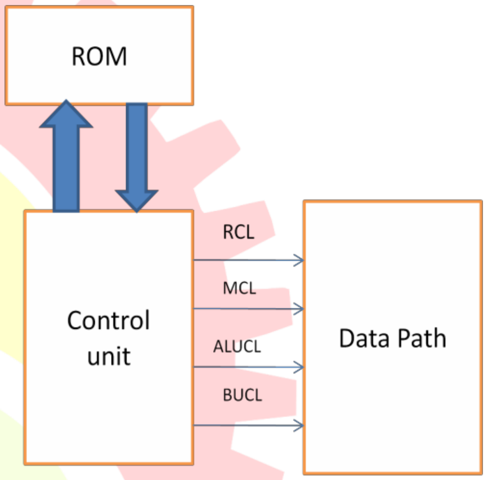

Figure 3.1 Block diagram of RISC processor

System Architecture:

The RISC processor presented in this paper consists of three components as shown in Figure, these components are, the Control Unit (CU), the Data Path, and the ROM. The Central Processing Unit (CPU) has 17 instructions. In the following sections we will describe the design of the three main components of the processor.

3.1.1 Design of the Control Unit:

The control unit design is based on using FSM (Finite State Machine) and we designed it in a way that allows each state to run at one clock cycle, the first state is the reset which is initializes the CPU internal registers and variables. The machine goes to the reset state by enabling the reset signal for a certain number of clocks. Following the reset state would be the instruction fetching and decoding states which will enable the appropriate signals for reading instruction data from the ROM then decoding the parts of the instruction. The decoding state will also select the next state depending on the instruction, since every instruction has its own set of states, the control unit will jump to the correct state based on the instruction given. After all states of a running instruction are finished, the last one will return to the fetch state which will allow us to process the next instruction in the program. Figure .2 shows the state diagram for the control unit.

3.1.2 Design of the Data Path:

The Data Path consists of subunits that are necessary for performing all of arithmetic and logic operations. A Data path is a hardware that performs data processing operations [8, 9, 10, and 11]. It is one of two types of modules used to represent a digital system, the other being a control unit. The Data path model we designed consists of the units necessary to perform all the operations on the data selected by the control unit. The components include a Register File, Arithmetic Logic Unit, Memory Interface and Branching Unit as shown in figure. The register file holds the table of the 32 general purpose registers available to the CPU, it has two output ports (output1, output2) and one input port, also it has a 16 bit bus connected directly to the Control Unit to pass immediate data. The ALU design consists of two input ports and one output port which mainly performs operations on two operands. It has a design similar to the control unit which selects an operation based on a code given by the ALUCL. The memory interface was designed to accommodate simple load/store operations with the 16x32 memory. The effective address is calculated by adding the content of the address register and the immediate data. The Branch Unit calculates a given condition by the control unit and raises a branch flag whether the condition is met or not, and if the flag is raised, it sends the branch address back to the control unit in order to replace the program counter. The control lines coming from the control unit operate all the units in the data path. The path starts from the register file that has two output ports which are connected to all the other units, after that the processing is done by one of the other units then finally returned back to the register files input port using the multiplexer. The signals used in the data path are forwarded from the control unit to each subcomponent as needed.

3.2 ARICHTECTURE OF 32- BIT RISC PROCESSOR:

This is the RISC processor architecture. It consists of the accumulator, buffer, Program counter, alu, resetter, control and decode, clock generator, instruction Register, mux, memory.

3.2.1 Arithmetic and Logic Unit:

The Arithmetic logic unit performs all the operations specified in the specifications of the processor. It performs around 20 instructions. The logic for all the instructions is mentioned in this module.

The floating point instructions in the processor are written in this module as functions. They don't involve the clock or the reset signal. They are just combinational circuits that are implemented by internal registers, counters and other small combinational circuits. The arithmetic logic unit consists of a temporary register that is used for operations between the Accumulator and the data on the data bus. This register is later assigned to the Accumulator.The ALU as in this module can be programmed to work in any fashion desired. Any application or control desired can be obtained by writing a function of the application and running it in the ALU module with the appropriate control signals from the Control Logic Decoder.

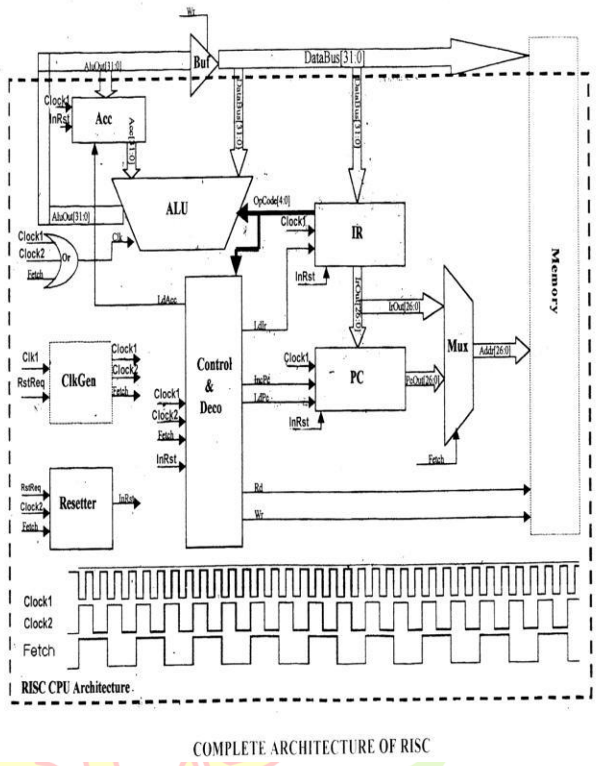

Figure 3.2 RISC Processor Architecture

3.2.2 Resetter:

The Resetter is a component that generates the internal reset signal. InRst taking the external reset request from the external world. The output from this module supplies all the components with the reset signal that initializes all the hardware components in the system.

3.2.3 Clock Generator:

The clock generator is a combinational circuit that generates clock signals that are responsible for the function of the entire processor. The clock signal gets its input from a crystal oscillator that is connected from an external source. This is not shown in this module. The connection is understood to be implicit.The three clocks are generated as per the following logic. The first clock

Clock1 synchronizes with the external crystal oscillator. While Clock2 changes at the negative edge of Clock1. So it has a period twice that of Clock1 and changes for the negative edge of Clock1. The third clock Fetch changes at the positive edge of Clock2. So it has a period twice that of Clock2, or effectively it has a period 4 times that of Clock1, and changes only for every other negative edge of the external crystal oscillator.This sort of pattern is generated so that, eight distinct control signals can be generated by taking different combinations of these clocks.

3.2.4 Program Counter:

The program counter is one of the essential parts of the RISC processor. It keeps on incrementing the address of the next address location. It is nothing but a simple counter that keeps on incrementing when the control signal INCPC appears at the input. When the LDPC signal is given, the program counter is loaded with the value or lROUT. This takes place at the negative edge of the Fetch signal.

3.2.5 Buffer:

The Buffer is used between the ALUOUT and the Data Bus. Under normal circumstances the ALUOUT is connected to the Accumulator. But when WR is 1 then, ALUOUT is amplified and put on the Data Bus. Otherwise the ALUOUT is connected to the Accumulator.

3.2.6 Accumulator:

The accumulator is a 32 bit register that is used to store one of the operands or the result of an operation. It is connected to both the ALU and the Data bus through a buffer. All Accumulator operations are synchronized with the Clock 1 and InRst. When the LDACC is one with Clock 1 going through its positive edge, the Accumulator is loaded from the ALUOUT or the data from the Accumulator is put on the Data Bus.

3.2.7 Instruction Register:

The instruction register is a 32-bit register that is used to store the instruction address after it is fetched from the memory. This is used to go to the address location in the memory that contains the Opcode and the operand. This synchronizes with the clock 1 and InRst. For the positive edge of the Clock 1 when LDIR is one, the instruction is loaded into the instruction register.

3.2.8 Multiplexer:

The Multiplexer is used to multiplex the outputs of the instruction register and program counter i.e. IROUT and PCOUT respectively. This is done as, during the same clock cycle (Fetch) the memory is accessed by both these registers. As a result we need to multiplex both of them.

3.2.9 Memory:

The Memory in the RISC processor is located inside the processor. This decreases the memory access times and improves the speed of the system. There are only two instructions that are used to access the memory, the load and Store accumulator instructions. This is basically a RAM that is present inside the memory.

3.2.10 Control and logic decoder:

The Control Logic Decoder is the most important part of the RISC processor. It is central nervous system of the entire processor as it outputs the critical control signals that are responsible for the function of the processor. It has the three clock signals and the 5-bit Opcode as inputs.Depending on the Opcode generated an individual set of control signals are generated that makes the RISC processor work like it does. The end user specifies this pattern of control signals that decides how the processor should function.The control logic is nothing but a decoder that generates different combinations based on different inputs given.

IV. RESULTS AND DISCUSSIONS

This chapter describes the design and implementation of RSIC Processor on FPGA. It describes, in brief, about simulation had been done in XILINX Software using Verilog Language and implementation done using ALTERA FPGA BOARD. System development is done in incremental steps. At each successive step, test cases are developed and simulation is done to verify the correct behavior. At any step, if any violation from the expected behavior is found, the design entry is modified to rectify the violation and the process is repeated until all design expectations are met. Initially, after completing the design entry, simulation is done using several test benches.

4.1 Simulation Results

4.1.1 Output of addition operation:

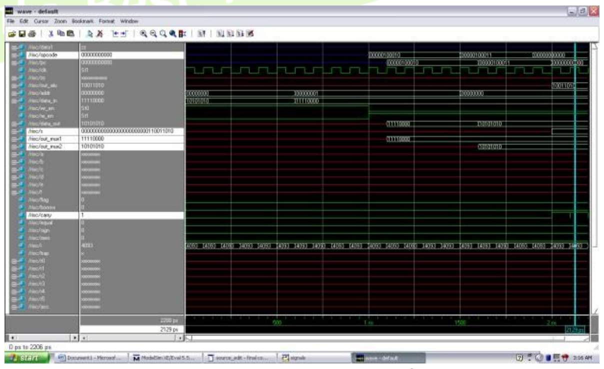

Figure 4.1: Simulation Result of addition

4.1.2 Output for Subtraction Operation:

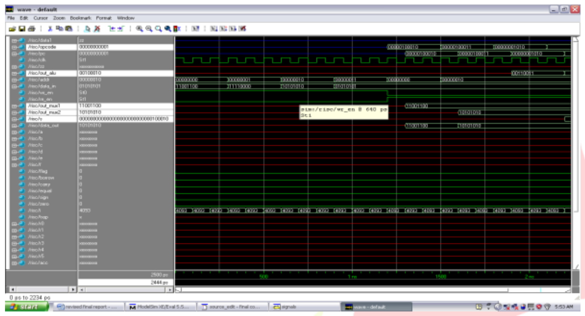
Figure 4.2: Simulation Result of subtraction

4.1.3 Output for OR operation:

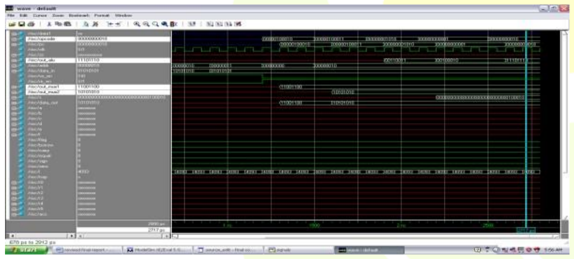
Figure 4.3: Simulation Result of OR

4.1.4 Output for AND operation:

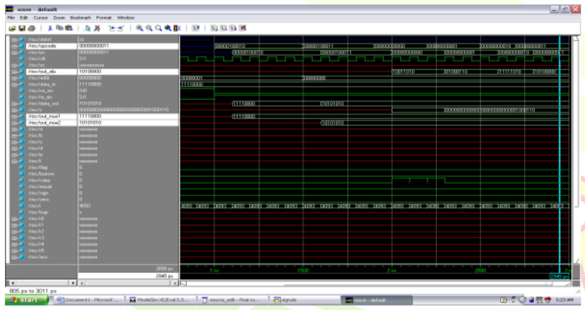
Figure 4.4: Simulation Result of AND

4.1.5 Output for XOR operation:

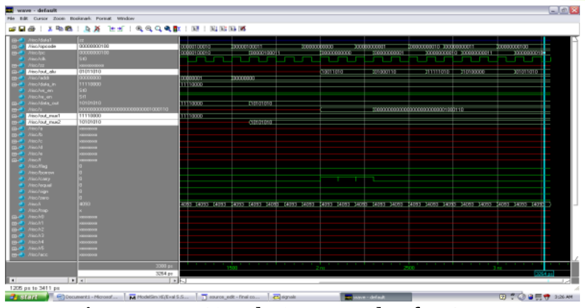
Figure 4.5: Simulation Result of XOR

4.1.6 Output for XNOR operation:

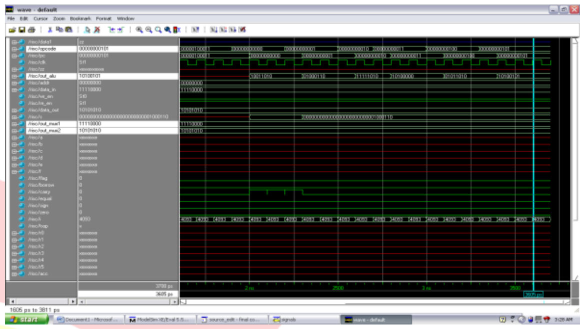
Figure 4.6: Simulation Result of XNOR

4.1.7 Output For Left Shift:

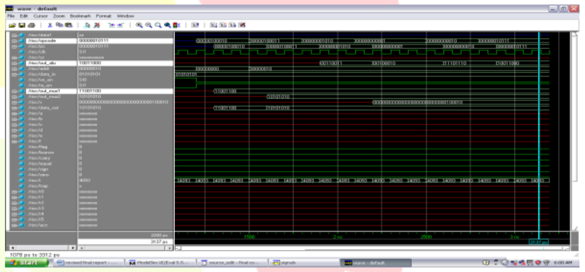
Figure4.7: Simulation Result of left shift

4.1.8 Output For Load Operation:

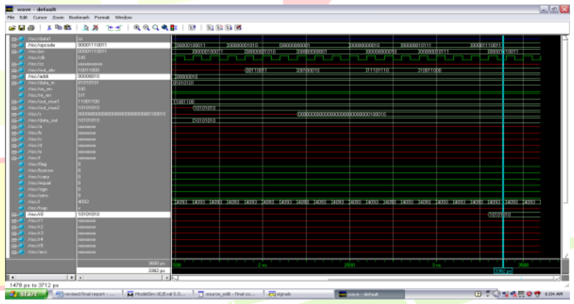
Figure 4.8: Simulation Result of load operation

4.1.9 Output For Store Operation:

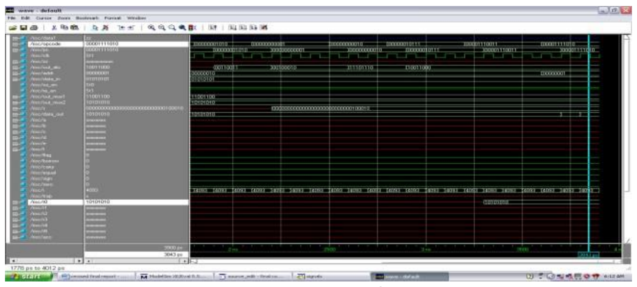
Figure 4.9: Simulation Result of store operation

V. CONCLUSION AND FUTURE WORK

The 32-bit RISC processor has been implemented using Xilinx FPGA. The simulation result is obtained and verified using ModelSim Simulator. The programming language used is Verilog HDL. We are using the hardwired control and decode unit, which is more efficient than the micro-programmed control.. The proposed RISC processor performs the arithmetic, logical, relational, load, store etc.

In future implementation of 64 bit RISC processor can be attempted with more pipelining stages. Also there are other high performance microprocessor architectures like VLIW (very long Instruction word), and EPIC. We have planned to introduce low power techniques like clock gating, and power gating etc., to reduce the power consumption in the system.

REFERENCES

1.Wayne Wolf, FPGA Based System Design, Prentice Hall, 2005.
2.Luker, Jarrod D., Prasad, Vinod B., "RISC system design in an FPGA",MWSCAS 2001, v2,2001,pp.532536.
3.John L. Hennessy, and David A. Patterson, "Computer Architecture A Quantitative Approach", 4th Edition; 2006.
4.Vincent P. Heuring, and Harry F. Jordan, "Computer Systems Design and Architecture", 2nd Edition, 2003.
5.Computer Organization: Patterson & Hennessy.
6.Rainer Ohlendorf, Thomas Wild, Michael Meitinger, Holm Rauchfuss, Andreas Herkersdorf, "Simulated and measured performance evaluation of RISC based SoC platforms in network processing applications", Journal of Systems Architecture 53 (2007) 703–718.
7.Computer Architecture: Hennessy & Patterson.
8.Jiang, Hongtu; "FPGA implementation of controller data path pair in custom image processor design"; IEEE International Symposium on Circuits and Systems Proceedings; 2004, p V141V144.
9.Jiang Hongtu, Owall Viktor, "FPGA implementation of controller datapath pair in custom image processor design", IEEE International Symposium on Circuits and Systems, Proceedings v 5, p V141 V144.
10.Michacl slater "Microprocessor-Based Design (a comprehensive guide to effective hardware design)" pp.55
11. Douglas V.HALL "Micrprocessor and Interfacing (Programming and Hardware)pp.547