

Design of Instruction Fetch Unit and ALU for Pipelined RISC Processor

Sarika U. Kadam¹, Prof. Samadhan D. Mali²

¹ME Student (ENTC),
Sinhgad College of Engineering,
Pune, Maharashtra, India
kadamsarikau@gmail.com

²Assistant Professor, Dept. of ENTC,
Sinhgad College of Engineering,
Pune, Maharashtra, India
Sdmali.scoe@sinhgad.edu

Abstract: *Pipelining is a concept which improves the performance of processor. A five stage pipelined RISC processor has stages as instruction fetch, decode, execute, memory, write back. RISC has a simpler and faster instruction set architecture. The aim of paper is to design instruction fetch unit and ALU which are part of RISC processor architecture. Instruction fetch is designed to read the instructions present in memory. ALU is in the execution stage of pipelining which performs all computations i.e. arithmetic and logical operations. Xilinx 8.Ii is used to simulate the design using VHDL language.*

Keywords: Instruction, Pipeline, RISC, VHDL.

1. Introduction

Different computers in world are distinguished by the type of instructions it holds. There are two different ways of designing instruction set of computers that are CISC and RISC. Processors designed with RISC instruction set has simpler and faster instruction set. In RISC load-store architecture is used and instruction executes in one machine cycle. As compared to RISC, CISC has complex, larger instruction set and an instruction takes longer to execute. CISC doesn't constitute load-store architecture.

Performance or speed of processor depends on many factors. It can be improved by using faster circuit technologies or rearranged hardware to perform more than one functions at a time, such process increases throughput of processor. Throughput is number of instructions executed per unit time. In pipelining instruction execution is divided into number of stages. Pipelining is the concept which improves the performance of processor. The five pipelining stages are instruction fetching, decode, execute, memory, write back. While first instruction is being executed the second instruction is fetched from memory in pipelining.

Instruction fetch unit is a very first step in pipelining. It reads instructions one by one according to the address provided. Output of this stage is provided to decoder. ALU in the stage of execution performs necessary computation on data applied to its input. Instruction fetch and ALU are designed and used as a part of pipelined processor.

2. Processor Architecture

Figure 1 shows the data path for five stage pipelined RISC processor. Different blocks are present in each stage of pipelining as shown in Figure 1. The five stages of processor are:

- a) Fetch,
- b) Decode,
- c) Execute,
- d) Memory and
- e) Write back of which fetch and ALU in execute stage are designed.

RISC processor has load/store architecture. RISC is having reduced, simpler and faster instruction set. Instruction in RISC executes in one machine cycle.

Data flow in Figure 1 is typically from left to right. There are some exceptions for this, in write back stage data or result is stored in register file and selecting next value of PC between incremented and branch address in these two cases flow of data is from right to left. As compared to CISC, RISC is less complex and simpler. Instruction is fetched from memory by using PC as an address to memory, the instruction fetched from memory is given to the next stage in pipelining i.e. decode.

In decode stage the instruction is decoded and the operands and Opcode of the instruction are identified. According to the Opcode the corresponding control signals are provided to the execution block with operands to perform that specific operation given to that Opcode. After performing operation on data the results are stored either in data memory or register file. For load/store instructions only the data memory comes into picture. For load instruction register is loaded with value from data memory and for store instruction register's value is stored in the data memory. Write back is a stage which stores results either from ALU or data memory to register file.

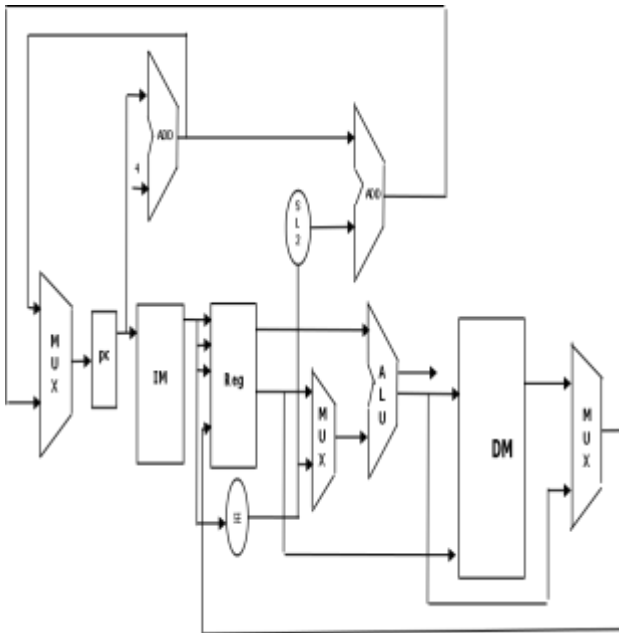


Figure 1: Data path of processor

3. Instruction Fetch unit

Instruction fetching is the first step of pipelined processor. Figure 2 shown below has different blocks named add, PC, instruction memory. Instruction memory is nothing but a ROM in which all instructions are stored. ROM is a read only memory, it has a single address line and one output port. It doesn't consist of write port as it is read only memory. Once the instructions are in memory it can't be changed. When provided with address instruction memory gives output as instruction stored in that particular address.

Add is a block which increments PC (program counter value) by one. On every clock cycle PC is incremented by and new instruction is fetched out. PC holds the address of instruction to be fetched, it points to memory where instruction is stored and at output port instruction is read. This instruction has various fields which shows Opcode, source register, destination, target register address. There are three different types of instruction R-type, I-type, J-type. Each instruction type has different format.

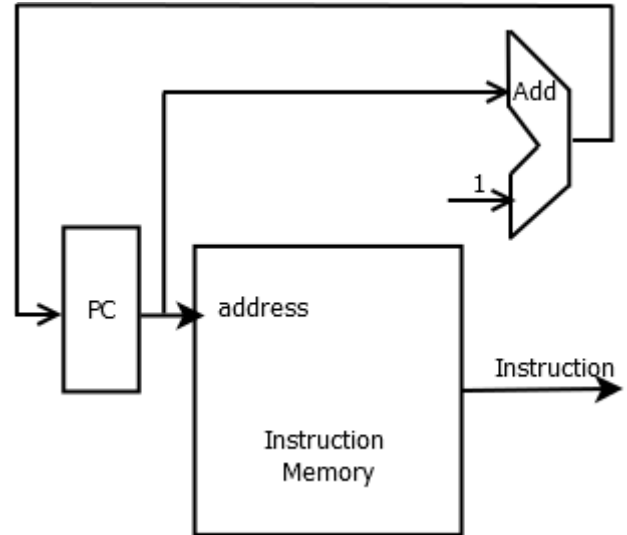


Figure 2: Instruction Fetch Unit

Table 1: Instruction code

Address/PC	Instruction code
0	0
1	134217728
2	268435456
3	402653184
4	536870912
5	671088640
6	805306368
7	939524096

4. ALU

ALU is used in execution stage of processor. It performs all necessary computations. It operates on data taken from previous stage i.e. decode.

As shown in Figure 3 ALU has two inputs called as operands each of 32-bit. A 5 bit 'Operation' is used as select line for ALU operation. 'Result' is an output which is also 32-bit. ALU designed performs functions such as MOV, ADD, SUB, AND, OR, NOT, XOR. Table 1 shows functions for different Opcodes.

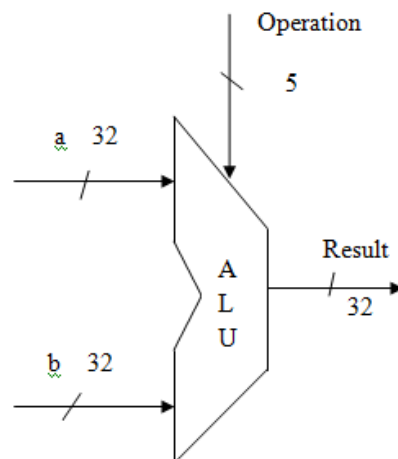


Figure 3: Block Diagram of ALU

Table 2: ALU operation

Opcode	Operation
00000	MOV
00001	ADD
00010	SUB
00011	AND
00100	OR
00101	NOT
00110	XOR

5. Results

For Opcode “00001” performs addition operation with ‘1’ and ‘0’ as operands, it gives result ‘1’. Similarly for Opcodes “00010”, “00011”, “00100”, “000101”, “00110” it performs subtraction, AND, OR, NOT, XOR respectively. Code for ALU is written in VHDL and simulated in Xilinx 8.1i.

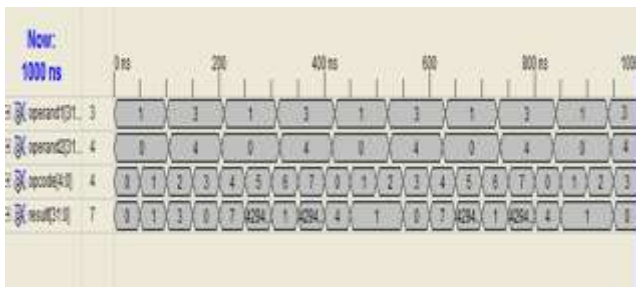


Figure 4: Simulation of ALU

Design summary of ALU in Figure 5 shows no errors and warnings and device utilization.

ABC Project Status			
Project File:	abc.ae	Current State:	Synthesized
Module Name:	alu_0	• Errors:	No Errors
Target Device:	xc3a1200e-3625	• Warnings:	No Warnings
Product Version:	ISE 8.1i	• Updated:	Thu Dec 31 22:31:05 2015

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	100	8672	1%
Number of 4-input LUTs	194	17044	1%
Number of bonded CIBs	101	190	53%

Figure 5: Design Summary of ALU

PC is a register which holds the address of instruction in

memory. On every clock cycle ‘PC’ is incremented by ‘1’ and it points to next instruction in memory. Instruction is read out at instruction output port. Fetched instruction is given to decoder unit where it is decoded to know registers or memory addresses which contains operands.

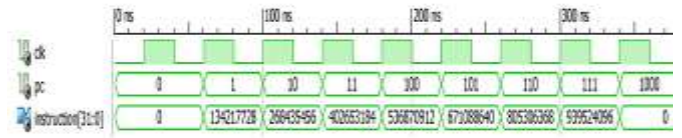


Figure 6: Simulation of Instruction Fetch Unit

6. Conclusion

This paper proposes ALU which performs operations such as addition, subtraction, AND, OR, NOT, XOR etc successfully. ALU provides correct results according to Opcodes and operands provided. ALU designed in this paper is used in execution stage of pipelined processor.

Instruction fetch unit works correctly when provided with address it fetches correct instruction from memory. It is used to read instruction from memory which is the first step of pipelined processor. The designed fetch unit and ALU are used in pipelined RISC processor.

References

- [1] Pravin S. Mane, Indra Gupta, M. K. Vasantha, “Implementation of RISC Processor on FPGA”, Electrical Engineering Department, Indian Institute of Technology Roorkee, 2006 IEEE.
- [2] Kui YI, Yue-Hua DING, “32-bit RISC CPU Based on MIPS”, International Joint Conference on Artificial Intelligence 2009.
- [3] Mrs. Rupali S. Balpande, Mrs. Rashmi S. Keote, “Design of FPGA based Instruction Fetch and Decode Module of 32-bit RISC (MIPS) Processor”, International Conference on Communication Systems and Network Technologies, 2011.
- [4] Mr. S. P. Ritpurkar, Prof. M. N. Thakare, Prof. G. D. Korde, “Synthesis and Simulation of a 32Bit MIPS RISC Processor using VHDL”, IEEE International Conference on Advances in Engineering & Technology Research, 2014.
- [5] David A. Patterson, John L. Hennessy, “Computer Organization and Design”, the Hardware and Software Interface, Third Edition.