Computer Architecture 2020 Fall Homework 4

ID: r09922136

1. Modules Explanation

1) Control

Control module reads Opcode (instruction[6-0]) from the current instruction as input, and outputs the control signals ALUOp, ALUSrc and RegWrite to control the other modules. By default, This module set its internal register "RegWrite_o" as true, and the other two internal registers "ALUOp_o", "ALUSrc_o" will be toggled from "Op_i", which telling us what instruction type we are executing now.

More information of the control signals :
ALUOp_o: the input for ALU control
ALUSrc_o: decide which second source should ALU execute from
RegWrite_o: set true if we need to write back the Registers.

2) ALU control

ALU control module reads funct5 (instruction[31-25]) + funct3 (instruction[14-12]) from the current instruction as input, and outputs the control signal ALUCtrl to decide which arithmetic execution should ALU do, and I set the signals from following values by corresponding executions.

R-type		**I-type**	
and	3'b001	addi	3'b100
xor	3'b010	sari	3'b111
sll	3'b011		
add	3'b100		
sub	3'b101		
mul	3'b110		

3) Sign extend

Sign extend module reads 12-bit immediate value (instruction[31-20]) from the current instruction as input, and outputs the 32-bit extended immediate value. This module is designed to increase the size of a 12-bit data item by replicating the high-order sign bit of the original 12-bit data item in the high-order bits of the larger, destination 32-bit data item.

4) Adder

Adder module reads constant value 4 and the PC of current cycle as input, while outputs the next cycle PC. This module will add PC of current cycle and constant value 4 to be the output.

5) MUX_ALUSrc

MUX_ALUSrc module reads one control signal named “ALUSrc”, “data1_i” (Register Read Data 2) and “data2_i” (Sign Extend output) as input, while outputs data which is send to the ALU’s input port named “data2_i”. This module will select which source to be executed in ALU.

6) ALU

ALU module reads control signal named “ALUCtrl_i”, “data1_i” (Register Read Data 1) and “data2_i” (MUX_ALUSrc output) as input, while outputs the data which named “data_o” and “Zero_o”. This module will set the outputs according to the control signal from doing the corresponding arithmetic operation on two input fields. “Zero_o” is set to true if the branch is taken (“data1_i” == “data2_i”)

7) CPU

CPU module links all other modules together to simulated one-cycle CPU. We required slight modification from the provided file, with declaration of wires, we just link all modules with the data path from the following Figure.

modules in your CPU.v.

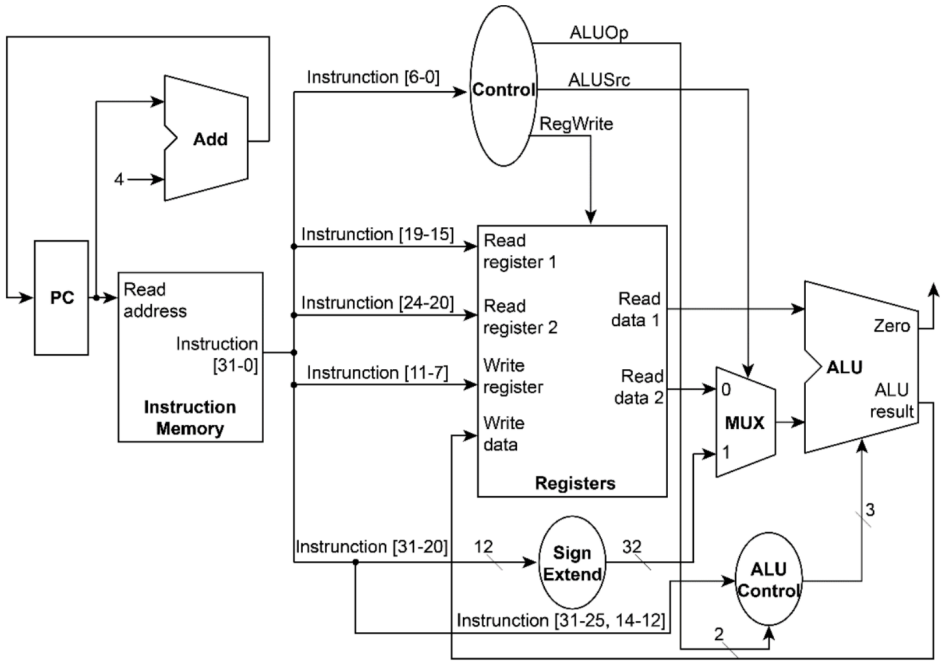

Figure 1 Data path of the CPU in this homework

8) PC, Instruction memory, Registers, testbench

These modules are provided by TA.

2. Development Environment

- OS : MacOS
- Compiler : iverilog