Computer Organization 105-2

Lab 3: Single Cycle CPU

Due: 2016/5/9 23:59:59

1. Goal

- Based on lab2, adding a memory unit(named the module DM in Simple_Single_CPU). Implement a complete single cycle CPU that can run R-type, I-type, branch, and jump instructions.

2. Requirement (70%)

- Please use ModleSim or Xilinx as your HDL simulator.
- It's a team assignment (max: 2 people). Please attach your names and student IDs as comments in the top of each file. The assignment you upload on e3 must have the form of "studentID_Name_studentID_Name_LAB3.zip".
- Reg_File[29] represents stack point. Please give an initial value to Reg_File[29] as 128, others 0.
- Decoder may add the following control signal:
  - BranchType_o
  - Jump_o
  - MemRead_o
  - MemWrite_o
  - MemtoReg_o
- Please name the Data_Memory module DM in Simple_Single_CPU.
- Basic instruction (50%)
  - Lab2 instruction + lw、sw、mul、jump
  - LW
    - Reg[rt] ← Mem[rs+imm]
  - SW
    - Mem[rs+imm] ← Reg[rt]
  - MUL
    - Reg[rd] ← Reg[rs]*Reg[rt]
  - JUMP
    - PC={PC[31:28]，address<<2}

Instruction	op[31:26]	rs	rt	rd	shamt	func
LW	6'b100011	rs[25:21]	rt[20:16]	imm[15:0]		
SW	6'b101011	rs[25:21]	rt[20:16]	imm[15:0]		
MUL	6'b000000	rs[25:21]	rt[20:16]	rd[15:11]	5'b00000	6'b011000
JUMP	6'b000010	Address[25:0]				

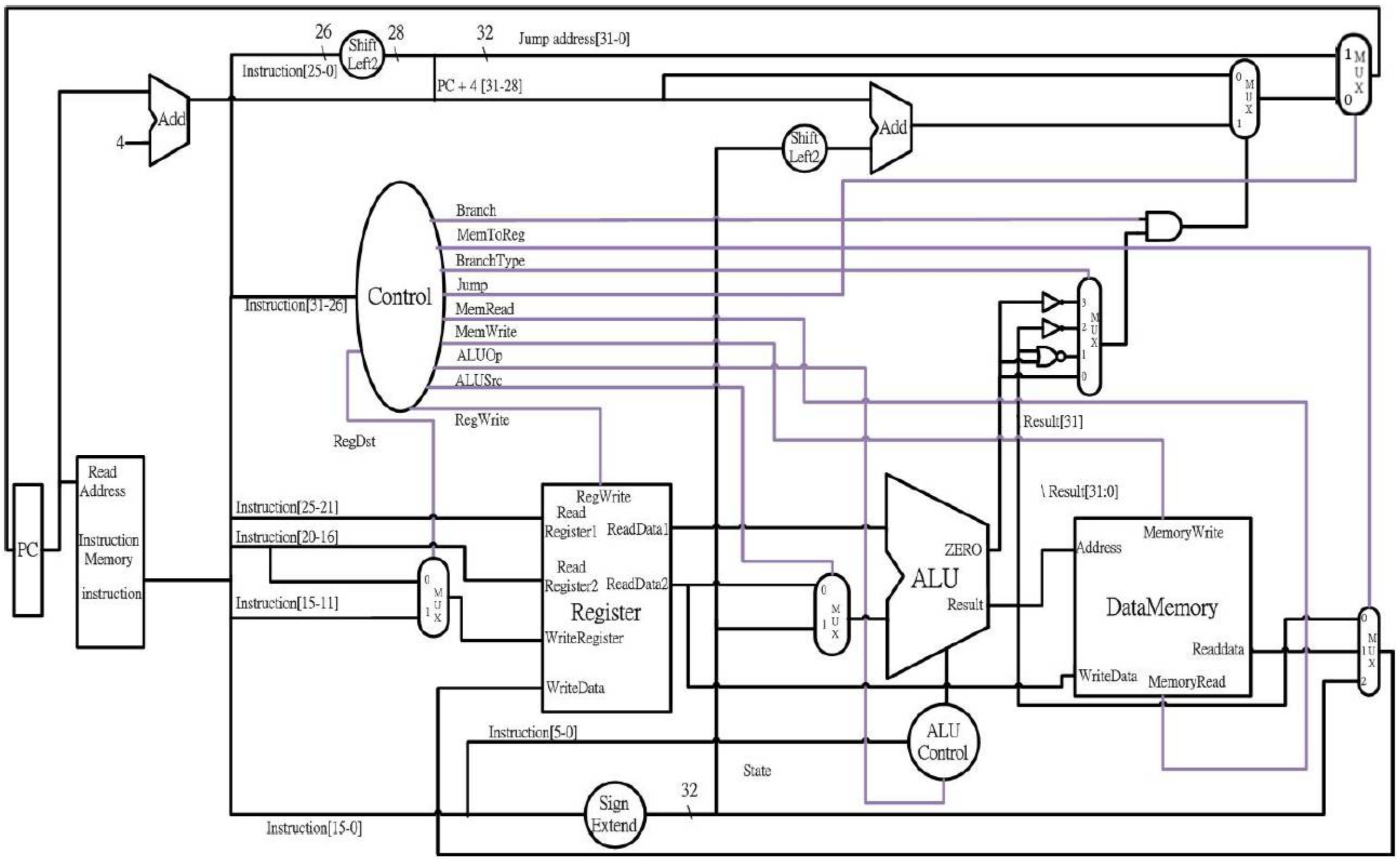
- Advanced instruction (20%)

Instruction	op[31:26]	rs	rt	imm
BGT	6’b000111	rs[25:21]	rt[20:16]	imm[15:0]
BNEZ	6’b000101	rs[25:21]	5’b00000	imm[15:0]
BGEZ	6’b000001	rs[25:21]	5’b00001	imm[15:0]

  - BGT (branch greater than)
    - If (rs > rt) then PC = PC + 4 + (sign_imm<<2)
  - BNEZ (branch non equal zero)
    - If (rs != 0) then PC = PC + 4 + (sign_imm<<2)
  - BGEZ (branch greater equal zero)
    - If (rs >= 0) then PC = PC + 4 + (sign_imm<<2)

3. Architecture Diagram

If you need to use extra control signal, please draw your design to the architecture and descript the implement flow on the report.



4. Bonus(10%)

Instruction	op[31:26]	rs	rt	rd	shamt	func
JAL	6’b000011	Address[25:0]				
JR	6’b000000	rs	0	0	0	6’b001000

5. Test

- Basic instruction
  - TA offers CO_LAB3_test_data1.txt to let you test your design. Refer to test_data1_result.txt to check your result.
- Advanced instruction
  - CO_LAB3_test_data2.txt extend the previous part and add more instruction to let your test your design. Refer to test_data2_result to check your result.
- Bonus
  - CO_LAB3_test_data3.txt, it is a Fibonacci function. When it done r2 is the answer we want. Refer to test_data3_result to check your result.
- Please change the line in the file "Instr_Memory" whenever you want to use a different test pattern file.
  - $readmemb("CO_LAB3_test_data1.txt", Instr_Mem);

6. Grading Policy

- Total source: 110pts
  - Basic score: 50 pts, Advance instructions: 20 pts, Bonus: 10%, Q&A: 20pts, Report: 10 pts.

  ※ **Any Plagiarism will be punished with a null score!**
- **Delay: 10%off/day**

7. Hand in your assignment

Please upload the assignment to the E3.
Put all of .v source files and report into same compressed file. (Use "studentID_Name_studentID_Name_LAB3.zip" to be the name of your compressed file)