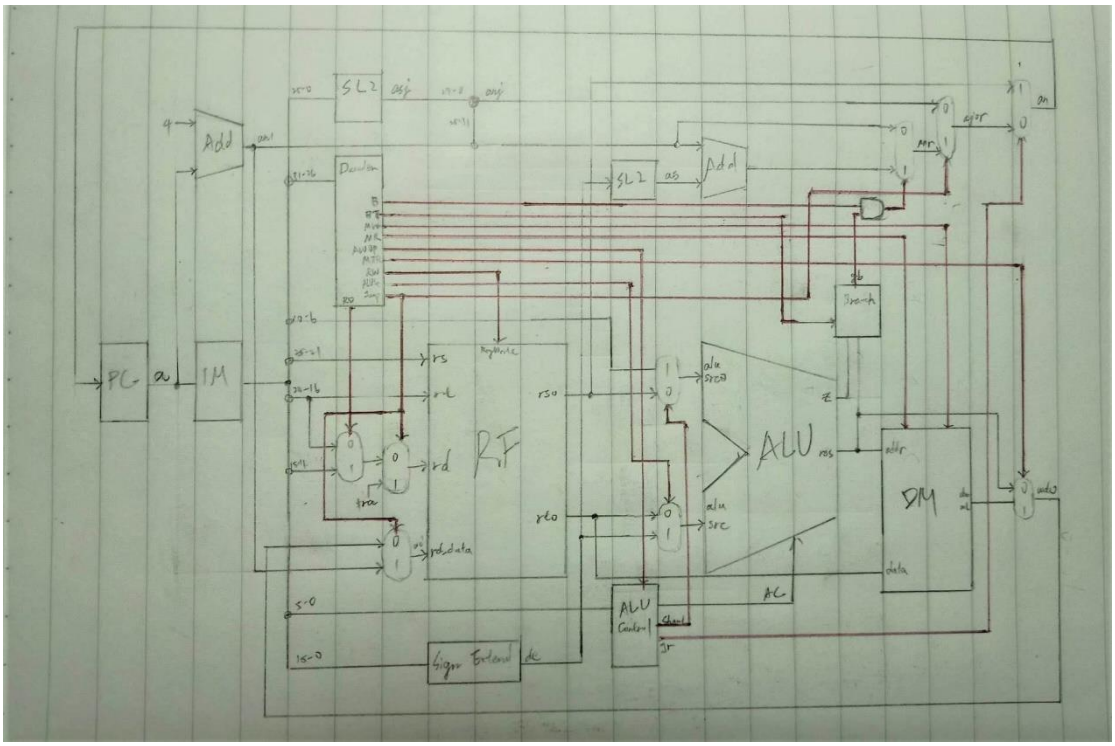
Computer Organization Lab 3: Single-cycle CPU With Memory Access And Jump

Student ID: 0716085 Name: 賴品樺

Teammate ID: 0716316 Name: 洪珩均

(1) Architecture diagram



(2) Implementation details

ALU.v:

Solve the negative number problem of ***srav*** and ***sra*** in Lab2.

Add the function of shift left and multiplication.

ALU_Ctrl.v:

Add the corresponding ALU control code to ***sll***, ***mul***, ***jr***.

Add 2 new output signals, Shamt and JumpReg, first is to determine one of the src of ALU comes from output of rs or shamt, second is to determine next PC comes from data of $ra or not.

Data_Memory:

New module for ***lw*** and ***sw***.

Decoder.v:

Add some new output signals for multiplexers to choose the input signals or for some systems to control the behavior.

Sign_Extend.v:
Add the function of zero extension to solve the failure of ***ori*** in Lab2.
Simple_Single_CPU.v:
Updating total structure to satisfy all the behaviors of the Architecture diagram.

(3) Questions

1.
PC=10[j Jump] 00001000000000000000000000011001
PC=38[bgtz $at, no_swap] 00011100001000000000000000000011
PC=54[bgtz $at, next_turn] 00011100001000000000000000000001
PC=58[j inner] 00001000000000000000000000001011
PC=5C[bne $t1, $0, outer] 00010101001000001111111111110000
PC=60[j End] 00001000000000000000000000011101
PC=6C[beq $t1, $t2, Loop] 00010001001010101111111111111110
PC=70[j bubble] 00001000000000000000000000000101

2.
The main purpose is function call.
example:

```
main:
    …
    jal mul
    …
mul:
    …
    jr $ra
```

In the main scope, when jal is executed, $ra register will store next instruction's PC and the program will jump into mul scope. At the end of mul scope, program will execute jr instruction which makes next PC become the value in $ra register. Next PC is the value in $ra means that the program will back to main scope and execute the instructions after the jal instruction.

3.
slt $t0, $rs, $rt
beq $t0, $zero, offset

4.
In Lab3, blez and bgtz can be combined by other instructions. The advantage is that we can save some opcode to design some more different instruction. The disadvantage is that one instruction needs one cycle, it needs more time to finish. And I think the fatal is in pipeline. In my

answer of previous question, input of beq instruction is output of slt instruction, which means the data hazard will occur. To solve this problem, the beq instruction can't go after slt instruction at the next cycle, which means more than 2 cycles is necessary.

(4) Contribution

Modify ALU, ALU_Ctrl, Sign_Extend, Simple_Single_CPU to satisfy Lab3's tasks.

Design the control signals in the architecture.

(5) Discussions

This Lab is very useful for me to understand to total structure of single cycle cpu and every instructions of MIPS. And this is my personal opinion, I think that the time period of Lab3 can be prolonged and combine all tasks in Lab2 into Lab3 In fact, after finishing Lab3, I can't understand the necessity of an independent Lab2.