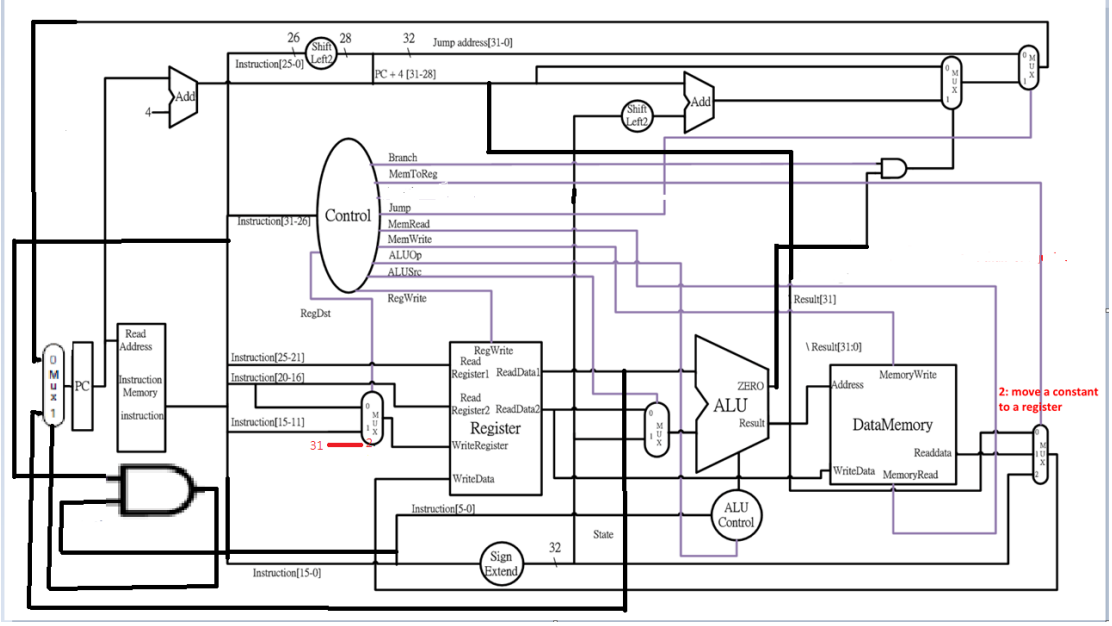
Computer Organization Lab3

Name: 邱弘竣

ID: 109550136

Architecture diagrams:



Hardware module analysis:

- Adder：we have two adder
  - one is for PC + 4, and one is for imm
- ALU
  - Do the different things based on defferent input
- ALU_Ctrl
  - control signal of ALU
- Instr_Memory
  - read instruction
- ProgramCounter
  - let us know the current PC
- Shift_Left_Two_32
  - shift left two bits and add zero to the empty bit
- Sign_Extend
  - Extend the leftmost bit
- MUX_2to1：we have four MUX_2to1

    - The first one is to check whether to perform jr
    - The second one is to check R-format or I format
    - The third one is to check whether to perform jump
    - The fourth one is to check whether to perform branch
- MUX_3to1：we have two MUX_3to1
    - The first one is to determine the value of Write Register
    - The second one is to determine the value of Write Data

Finished part:

Data1

```
PC =        128
Data Memory =          1,         2,         0,         0,         0,         0,         0,         0
Data Memory =          0,         0,         0,         0,         0,         0,         0,         0
Data Memory =          0,         0,         0,         0,         0,         0,         0,         0
Data Memory =          0,         0,         0,         0,         0,         0,         0,         0
Registers
R0 =          0, R1 =          1, R2 =          2, R3 =          3, R4 =          4, R5 =          5, R6 =          1, R7 =          2
R8 =          4, R9 =          2, R10 =         0, R11 =         0, R12 =         0, R13 =         0, R14 =         0, R15 =         0
R16 =         0, R17 =         0, R18 =         0, R19 =         0, R20 =         0, R21 =         0, R22 =         0, R23 =         0
R24 =         0, R25 =         0, R26 =         0, R27 =         0, R28 =         0, R29 =       128, R30 =         0, R31 =         0
```

Data2

```
PC =        128
Data Memory =          0,         0,         0,         0,         0,         0,         0,         0
Data Memory =          0,         0,         0,         0,         0,         0,         0,         0
Data Memory =          0,         0,         0,         0,        68,         2,         1,        68
Data Memory =          2,         1,        68,         4,         3,        16,         0,         0
Registers
R0 =          0, R1 =          0, R2 =          5, R3 =          0, R4 =          0, R5 =          0, R6 =          0, R7 =          0
R8 =          0, R9 =          1, R10 =         0, R11 =         0, R12 =         0, R13 =         0, R14 =         0, R15 =         0
R16 =         0, R17 =         0, R18 =         0, R19 =         0, R20 =         0, R21 =         0, R22 =         0, R23 =         0
R24 =         0, R25 =         0, R26 =         0, R27 =         0, R28 =         0, R29 =       128, R30 =         0, R31 =        16
```

Problems you met and solutions:

In the beginning, I didn't notice that I have to design the diagram by myself. I felt a little bit strange for the given diagram to implement the function we have to finish, but I chose to finish it by following the diagram. It was failed, of course. Then I noticed that I have to make a change to MUX to finish this lab. I spent a lot of time thinking of how to finish this lab, looking for the reference about our lesson, and finally, I finished my own version.

It's hard for me to debug in Verilog, too. There are a lot of wires there. I don't even know which unit leads to error. It takes me a lot of time to debug.

Summary:

After this lab, I realize the importance of "control". All the MUX are controlled by this unit. If it has a little error, then our project must be failed.