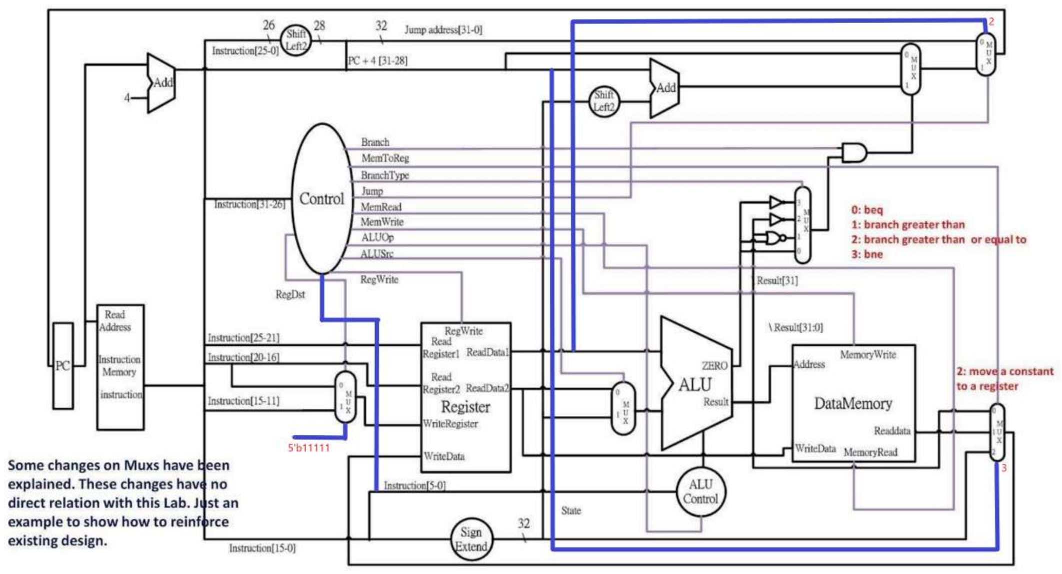
Computer Organization Lab3

Name: 房天越

ID: 110550085

Architecture diagrams:



This diagram is slightly modified based on the diagram given in the spec.
I used two extra wires to make it easier to realize jal and jr.
Also, I connected the function code to the decoder to determine whether it's jr when it's an R-format code.

Hardware module analysis:

Adder.v:

Output the sum of the two given numbers.

ALU_control.v:

Output ALUCtrl to control the ALU.v based on the function code and the ALUOp code.

ALU.v:

Do commands given by the ALUCtrl.v, including, and, or, add, sub, and slt.

MUX_2to1.v:

Choose to which data of the two given data to output based on the select input.

MUX_3to1.v:

Choose to which data among the three given data to output based on the select input.

MUX_4to1.v:

Choose to which data among the four given data to output based on the select input.

Shift_Left_Two_32.v:

Shift the given data left by two bits and set the last two bits to 00.

Sign_Extend.v:

Extend the given 16-bit data to 32 bit by filling the above 16 bits with the sign bit.

Decoder.v

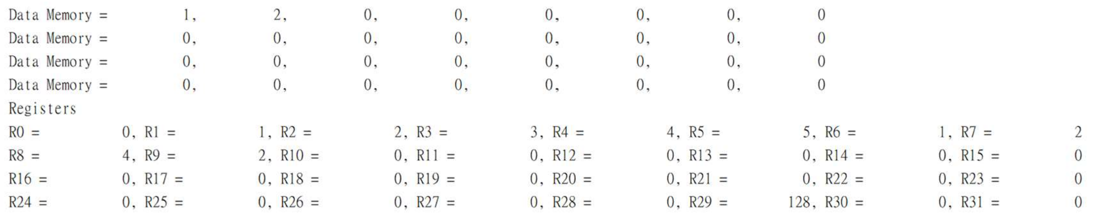
Output the ten signals in the above diagram based on the instruction, some signals that are don't care are given the value 0.

Simple_Single_CPU.v:

Realize the CPU with the diagram and modules.

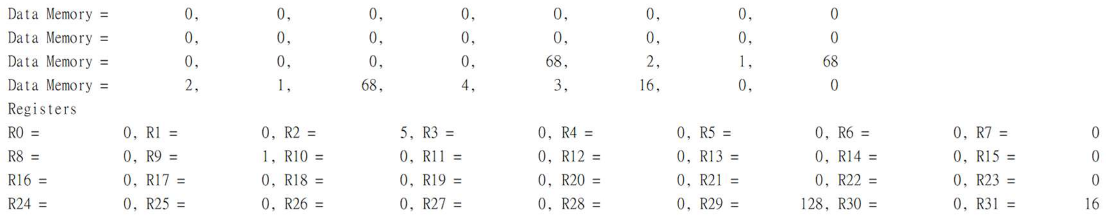
Finished part:

Data1:



```
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:



```
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
```

This result is the part of the part that are commented in the testbench file. I chose to use this because I found it more clear to see the current state of the memory and register.

Problems you met and solutions:

The diagram in this lab is much more complicated than the previous two labs, because there are now more commands to realize. Because of that,

I found debugging a nightmare and spent more time debugging than writing it. At last, I checked over the two most complicated modules, which are the decoder and the CPU file for many hours, and ended up finding error in the ALU file, I had a typo of “2” to be “1”.
After spending half a day debugging, I finally made the result to be the same as what I expected to.

Summary:

After this Lab, I have now a better understanding to the CPU and what role all the signals and modules play in it, and understand better about the commands that are added in this Lab compared to the previous Lab. Also, I found that I really have to stay focused when writing the Verilog code, otherwise I would have a bad time debugging and end up finding the error is only due to a small typo. I hope that I would do better in the future.