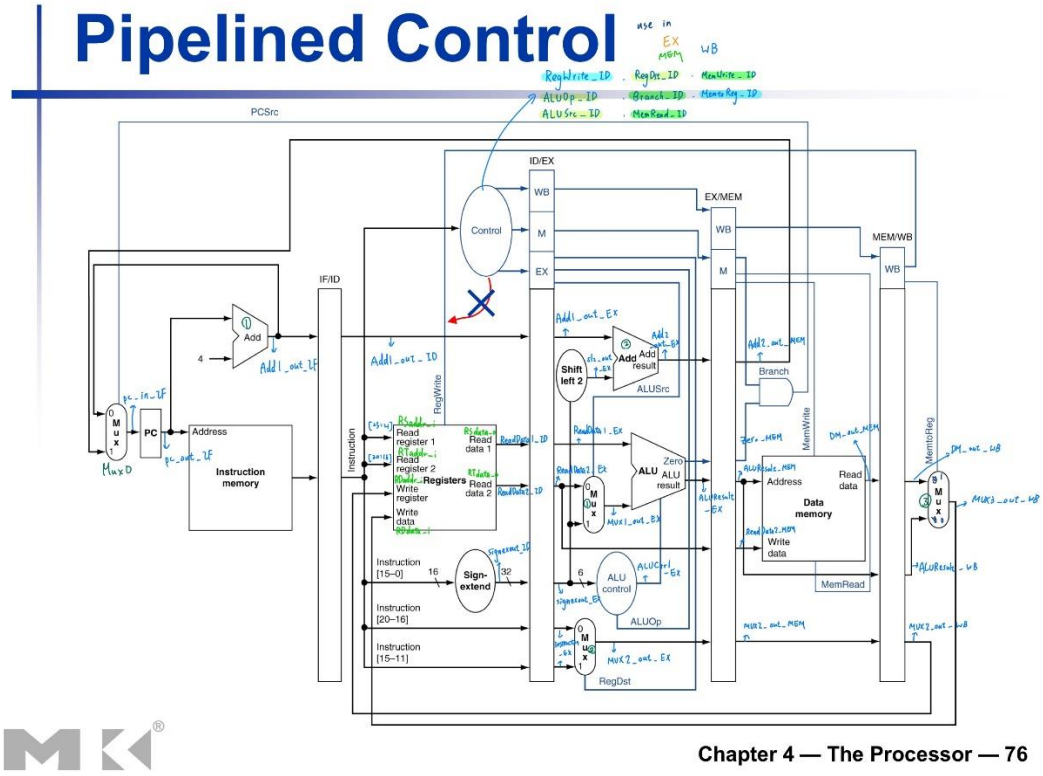
Computer Organization Lab4

ID:110550065 Name:尤茂為

Architecture diagrams:



Hardware module analysis:

(Explain how you design work and what are the difference between this lab and Lab3)

In Lab4 ,we don’t need to handle the jal and jr instruction, so we can use MUX 2to1 instead of MUX 3to1.Also,The input of MUX3(MemtoReg) is different from this lab. In order to deal with XOR and MULT instructions, I add some code for these instructions.

ALUCtrl:

```
always@(ALUOp_i,funct_i) begin
    if(ALUOp_i==3'b000)begin//R-type
        case(funct_i)
            6'b100000:ALUCtrl_o<=4'b0010;//add
            6'b100010:ALUCtrl_o<=4'b0110;//sub
            6'b100100:ALUCtrl_o<=4'b0000;//and
            6'b100101:ALUCtrl_o<=4'b0001;//or
            6'b100111:ALUCtrl_o<=4'b1100;//nor
            6'b101010:ALUCtrl_o<=4'b0111;//slt
            //new
            6'b011000:ALUCtrl_o<=4'b1101;//mult
            6'b100110:ALUCtrl_o<=4'b1110;//xor
            endcase
        end
```

ALU:

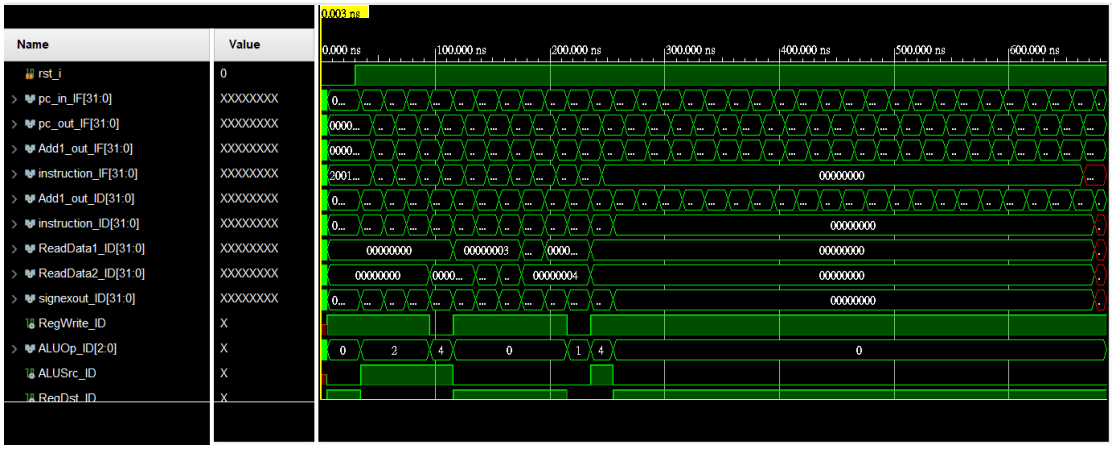
```
57          4'b1101:begin//mult
58              result_o<=src1_i*src2_
59          end
60          4'b1110:begin//xor
61              result_o<=src1_i^src2_
62          end
63          default:result_o<=0;
64      endcase
```

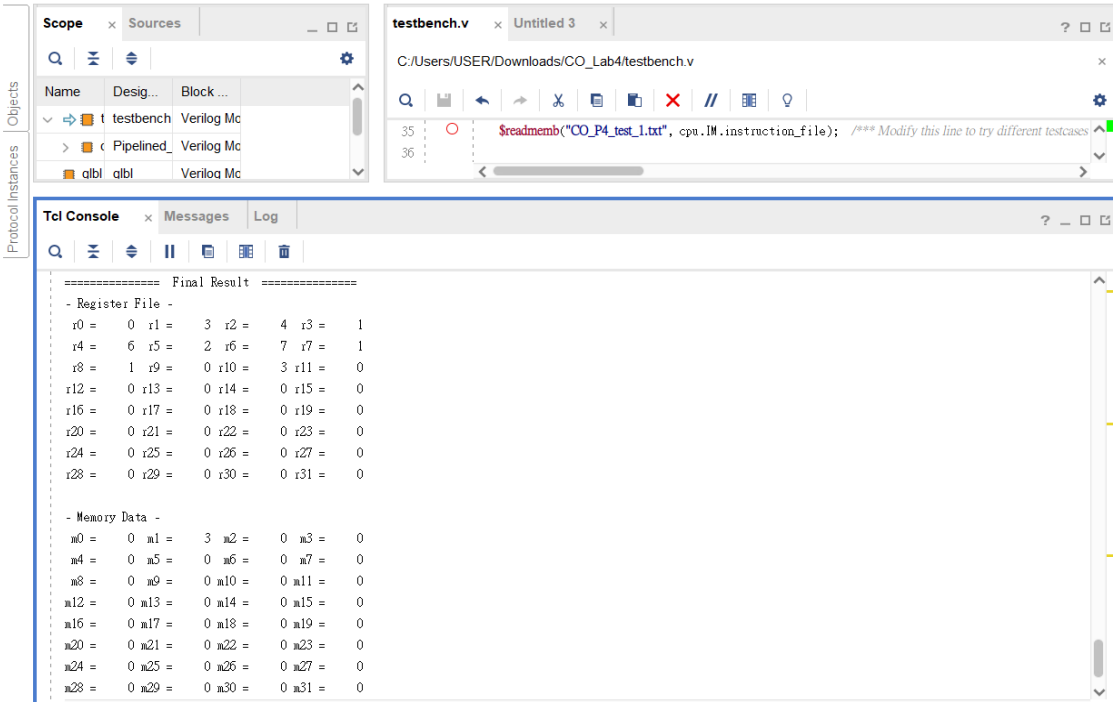
The main difference between lab4 and lab3 is we need numbers of wires to store the data in different stage since different stages has different instructions in the same time.

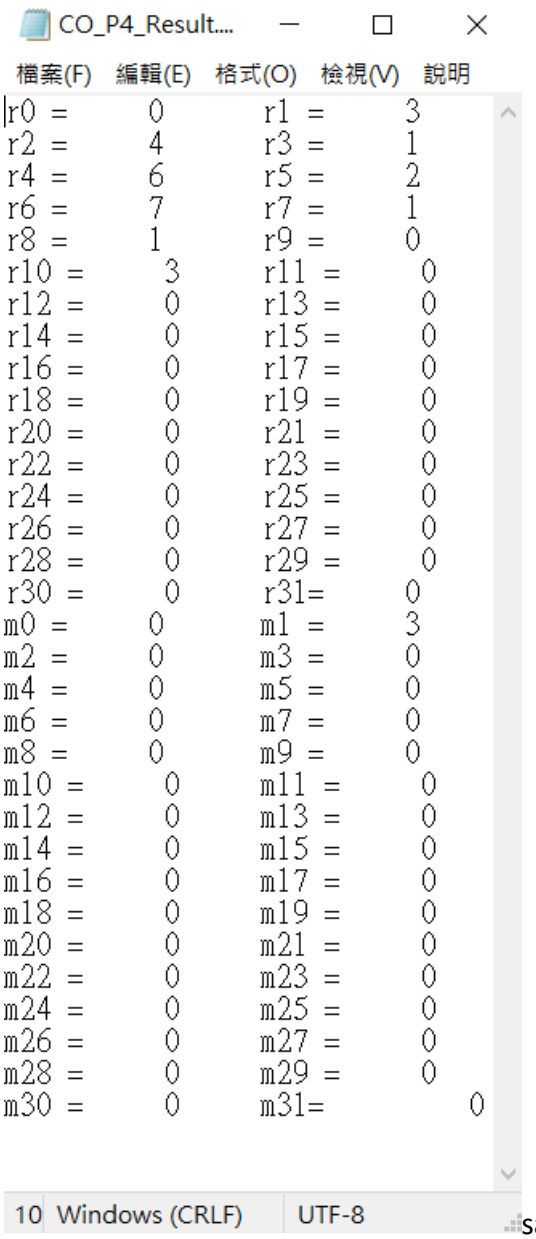
Simulation results:

(Show the screenshot of the simulation results and waveform and explain them)

Test1:

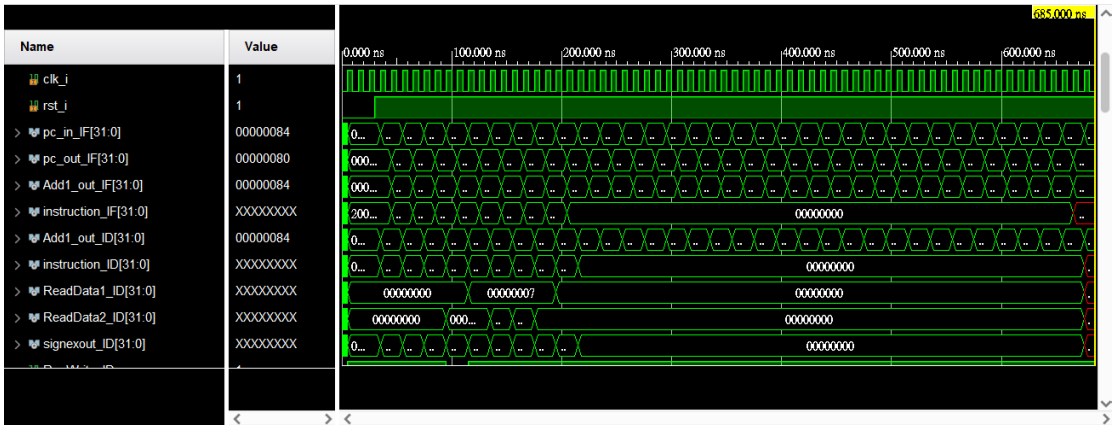


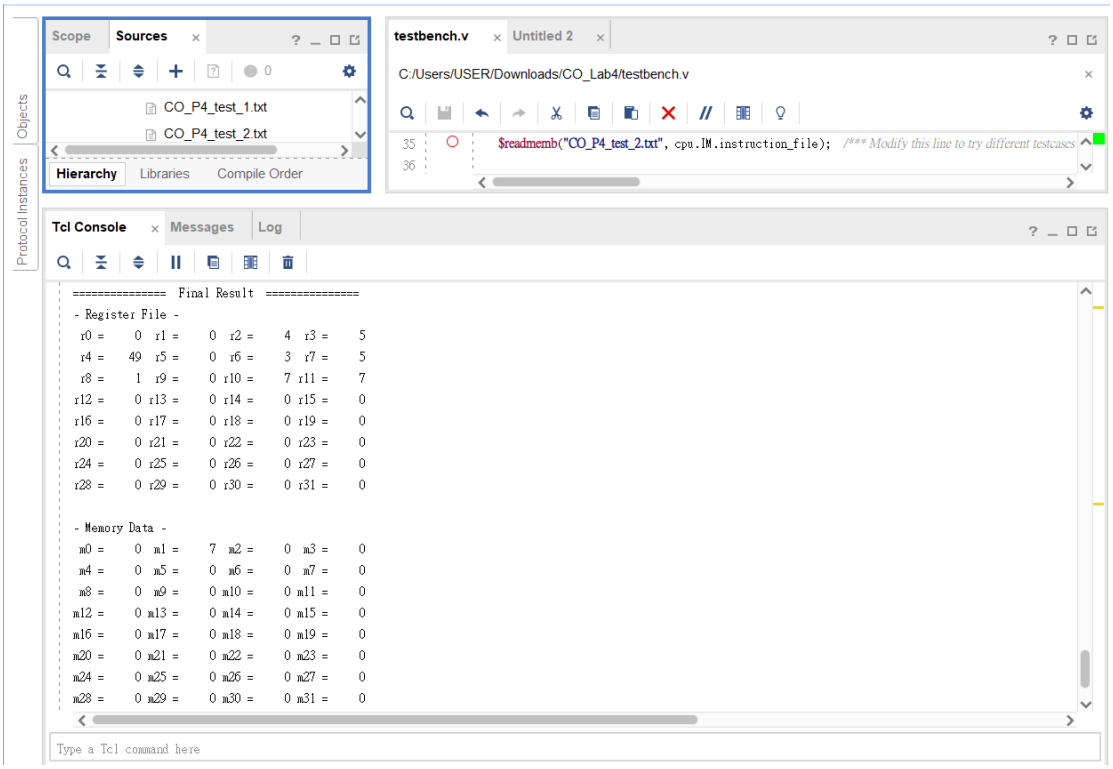


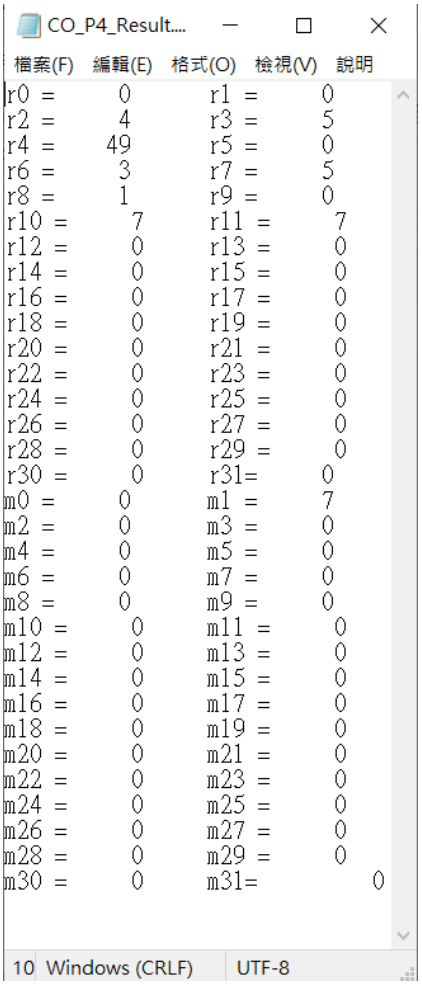


same as the CO_P4_test_1_result

Test2:

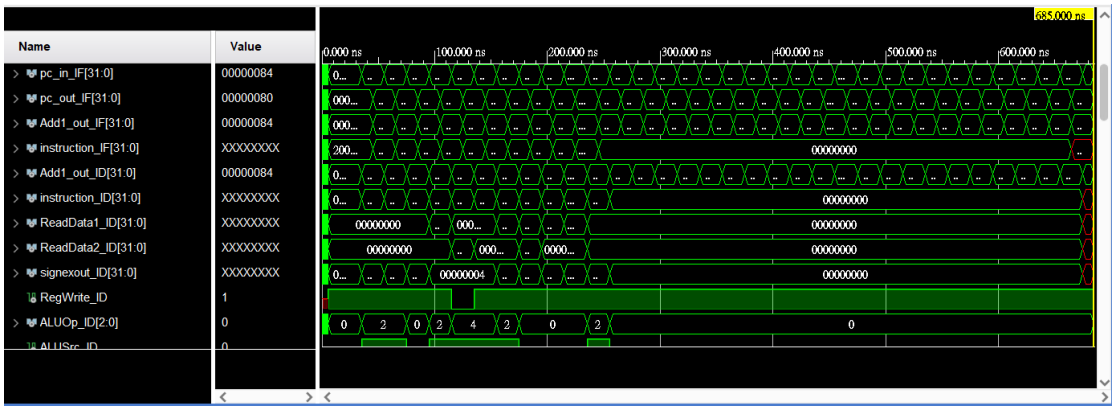


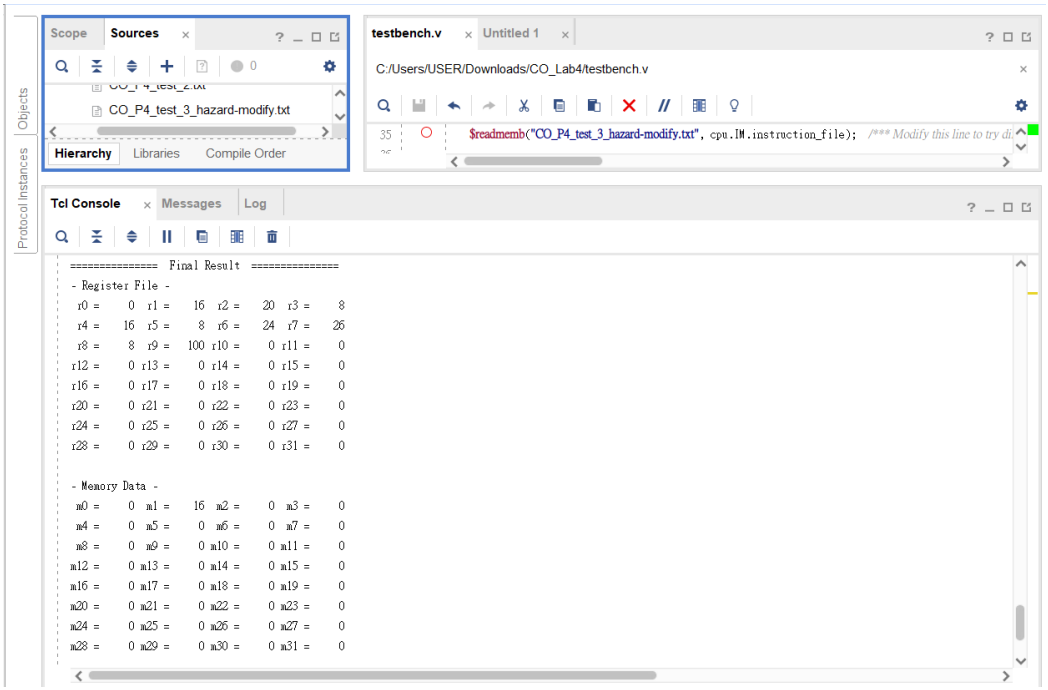


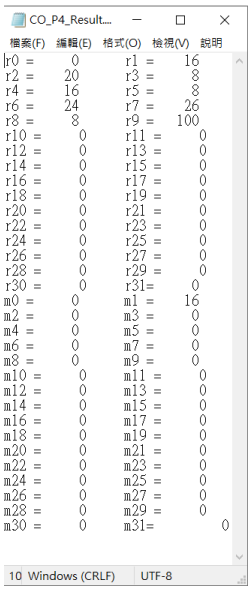


same as the CO_P4_test_2_result

Bonus Test3:

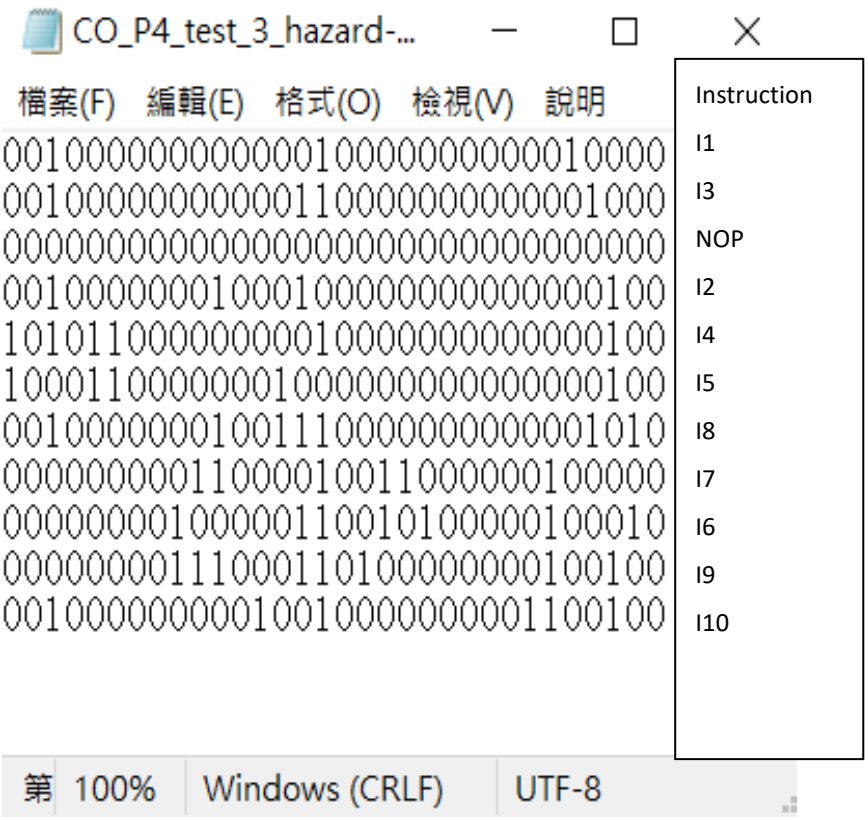






same as the CO_P4_test_3_result

Co_P4_test_3_hazard.txt after modify:



Solve the hazard between I1/I2:

I reorder I2 and I3 and insert a NOP between these two instruction.

Solve the hazard between I5/I6,I8/I9:

I reorder I5, I6, I7, I8.

Problems you met and solutions:

Too many variables makes me confused.

=>add stage name after almost each variables.

In Lab3, I used MUX 3 to 1 to complete my work, but in this lab, it only needs to use MUX 2 to 1.

=>used the MUX 2 to 1 of lab2.

Summary:

Compared to the previous labs, this time only need to focus on a few .v file. It's easier to find errors when debugging.

After understanding how the pipeline CPU works, it was easy to finish this lab if the submodule made before was correct.