Computer Organization

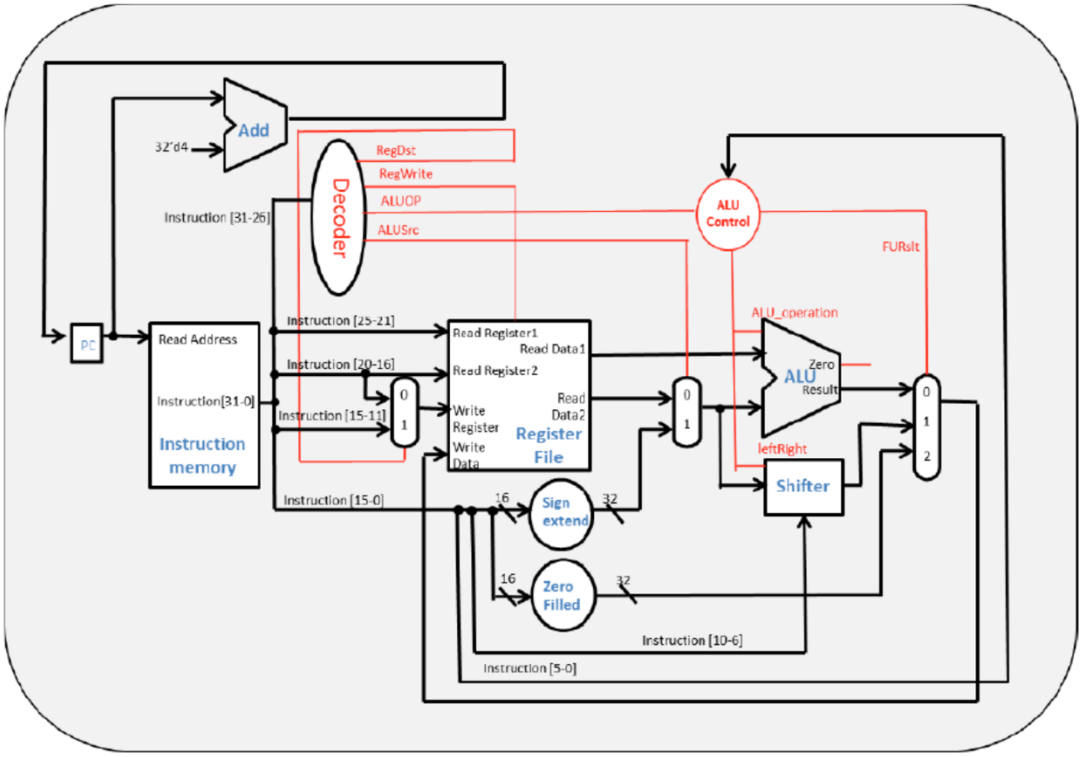
Architecture diagrams:

I connected the lines the same way as in the spec.

For the ALU_operation from ALU Control:
or = 0000, and = 0001, add = 0010, sub = 0110, nor = 1101, slt = 0111.

For the FURslt from ALU Ctonrol:
ALU_result = 00, Shifter_result = 01, Zero_filled = 10.
*Since this project doesn't use Zero_filled, I set FURslt[1] = 1'b0.

For the ALUOp from Decoder:
sw/lw = 000, beq = 001, R-format = 010, addi = 100.
*Since this project doesn't use sw/lw/beq, their values don't matter.



Hardware module analysis:

Adder:
Using assign, sum = src1 + src2.

ALU_Ctrl:
Using the ALUOp and funct input, determine the ALU_operation output value. Also using ALUOp and funct input, determine the FURslt output value. Since there is no Zero_filled used in this project, set FURslt[1] = 1'b0.

Decoder:
Using the instruction operation input, determine the ALUOp output for sw/lw, beq, R-format, and addi. Also using the instruction operation input, determine the mux input for ALUSrc, RegWrite, and RegDst.

Mux2to1:
Simply output data0 if select is 0 and output data1 if select is 1.

Mux3to1:
Simply output data0 if select is 00, output data1 if select is 01, and output data2 if select is 10. Else, output 0.

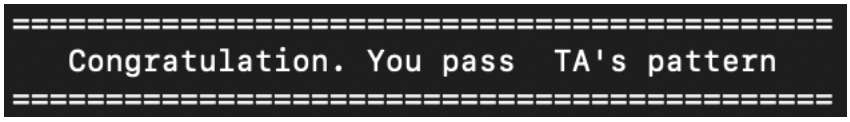
Sign_Extend:
Using assign, output = 16*input[15] + input[15:0].

Zero_Filled:
Using assign, output = 32 zeros.

Simple_Single_CPU:
Connect all the wires required to run this CPU properly.

Finished part:

I finished all the modules including ALU_Ctrl, Decoder, Muxes, Sign_Extend, Simple_Single_CPU, and Zero_Filled. Moreover, I passed all three test cases given by the TA's.



Problems you met and solutions:

One of the first problem I met is that the testbench given by the TA's doesn't let me run the line *for(;count != `END_COUNT;)* in iverilog. I think it's because iverilog requires you to put values on all three slots of the for loop. There is two ways that my friends and I found that solves this minor problem: (1) turn this 'fake' for loop into a while loop. Well, this for loop actually functions like a while loop so just change it to what it was supposed to be. (2) put *count=count* on the other two slots. This doesn't change the habit of the loop but satisfies the requirements of filling all three slots. I chose solution (1) as my way of solving this problem.

Another obstacle I met is correctly connecting all the wires in Simple_Single_CPU. You really have to know which line goes where and keep track of all the wires. It took me some time to figure out all the right places to connect for each wire. But at the end, it works.

Summary:

Upon completion of this project, I attained a more comprehensive comprehension of the inner workings of a basic single CPU. I gained an insightful understanding that each individual module within the CPU possesses a crucial role to perform, and collectively, they create a robust system that enables us to tackle more intricate and demanding tasks. The interconnection of these modules is a fundamental aspect that contributes to the CPU's enhanced functionality and efficiency, making it a formidable tool for addressing complex problems with relative ease. In summary, through this project, I was able to deepen my knowledge and appreciation for the intricacies of a single CPU and its ability to handle diverse computing requirements.