Report of CO\_LAB03 109550027紀竺均

● Detailed description of the implementation

1. PC + 4 Adder (Adder.v)

Simply add src1 and src2 together. I’ll assign src2 = 4 (in decimal).

2. ALU (alu.v)

Here, I modified my LAB2 ALU. First, remain ADD, SUB, AND, OR and SLT operations. Then I add three additional operations: XOR, SLL and SRA. I simply use operator ‘ ^ ’, ‘ << ‘ ,and ‘ >>> ‘ to implement. I defined the ALU\_control of them by myself and details will be record in ALU\_Ctrl part.

3. ALU control (ALU\_Ctrl.v)

The input 4 bit instr is consist of {instr[30],instr[14:12]}, where instr[14:12] = fun3. I define the relation between instr and output ALU\_Ctrl\_o below.

|  |  |  |
| --- | --- | --- |
| operation | 4-bits-instr | ALU\_Ctrl\_o |
| xor | 0100 | 1000 |
| sll | 0001 | 1001 |
| sra | 1101 | 1011 |

4. Decoder (Decoder.v)

Because all the operations in this lab is R-type, so I simply assign them with the same R-type value.

assign ALUSrc = 1'b0;

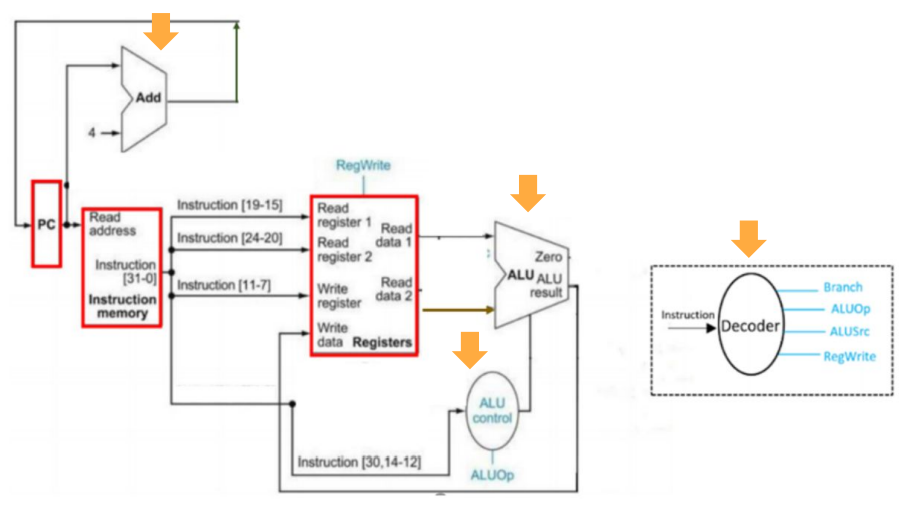
assign RegWrite = 1'b1;

assign Branch = 1'b0;

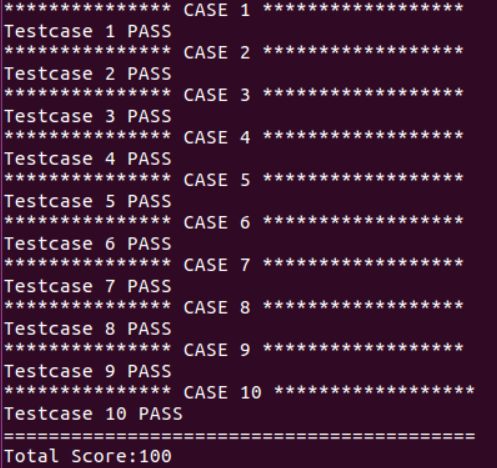
assign ALUOp = 2'b10;

5. Simple\_Single\_CPU (Simple\_Single\_CPU.v)

Just combine all the modules together.

.

● Implementation results



● Problems encountered and solutions

主要是環境的問題，一開始很高興下載好Ubuntu以後就跑跑看，怎麼跑結果都是0分，後來發現忘記下載iverilog…超笨的！

還有除到一個錯是Simple\_Single\_CPU裡面，一開始把instr[19:15]、instr[24:20]丟到src1、src2裡面，只有得到10分，後來才想到要丟data進去而不是address，最終順利解決！