Computer Organization Lab04 Report

Author: 109652039 林立倫, 109550003 陳茂祥

1 Implementation

1.1 Decoder

With the 7-bit input opcode, the decoder will output the corresponding control signals. We implement the decoder using case statement with opcode as the switch condition. For don't care signals, we set them to 0.

1.2 Immediate Generator

With 32-bit instruction as input, we can generate the immediate value using concatenation and replication operations as follows:

opcode	immediate value	Note
0110011	`32'b0`	R-type
0010011	`{ { 21{ sign } }, instr_i[30:20] }`	ADDI
0000011	`{ { 21{ sign } }, instr_i[30:20] }`	Load
0100011	`{ { 21{ sign } }, instr_i[30:25], instr_i[11:7] }`	Store
1100011	`{ { 20{ sign } }, instr_i[7], instr_i[30:25], instr_i[11:8], 1'b0 }`	Branch
1101111	`{ { 12{ sign } }, instr_i[19:12], instr_i[20], instr_i[30:21], 1'b0 }`	JAL
1100111	`{ { 21{ sign } }, instr_i[30:20] }`	JALR

where `sign` is the 31st bit of the instruction.

1.3 ALU Control

The ALU control is defined as follows:

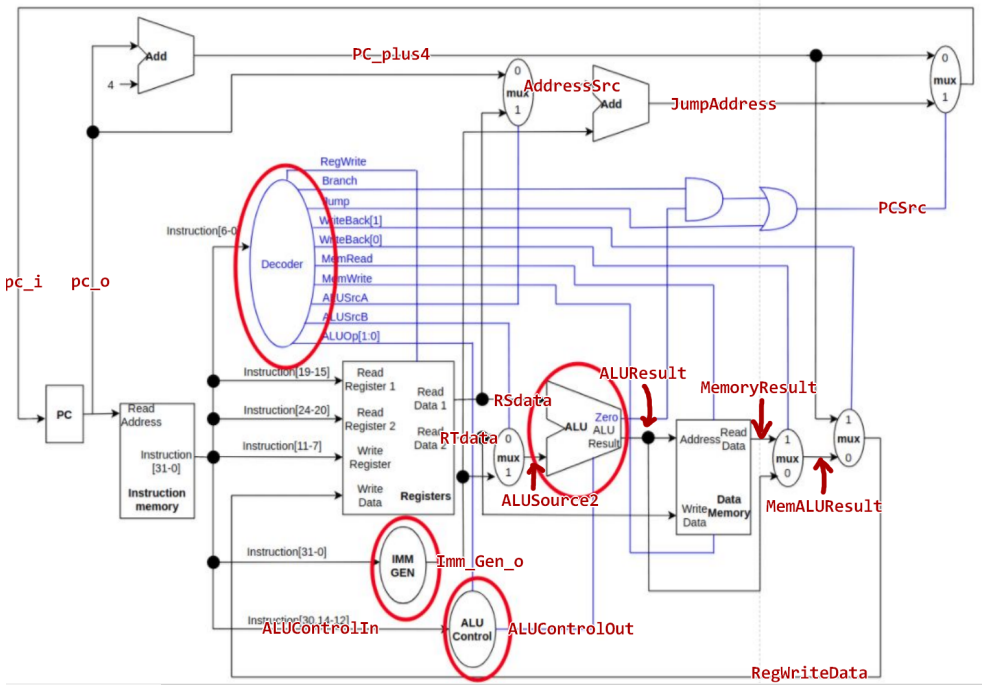
ALUOp	instr (I_{30}+func3)	ALU Control	Note
00	-	0010	Load / Store
01	-	0110	Branch
10	0000	0010	Addition
	1000	0110	Subtraction
	0111	0000	Bitwise AND
	0110	0001	Bitwise OR
	0010	0111	Set on Less Than

1.4 ALU

The 32-bit ALU is implemented using builtin operators. Note that SLT instruction depends on the result of addition, we choose the result to be `32'b0` or `32'b1` correspond to the 31st bit of the addition result. In order to determine whether the input `src1` and `src2` need to be inverted or not, we use tenary conditional operator to decide the actual operands of the operations.

1.5 Simple Single CPU

Simply connect the wire to the corresponding input and output. We defined the additional wires as follows:



2 Result

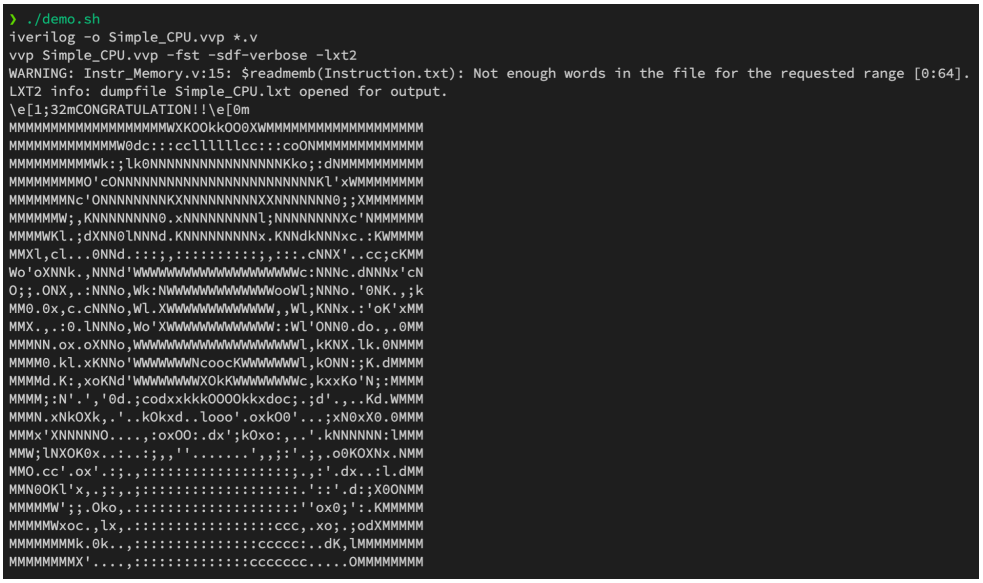


Figure 1: Simulation Result

3 Difficulties and Solutions

- The simulation is composed of lots of instructions, therefore it's hard to find bugs on specific instructions.

  Solution: `make` the testbench ourselves, and use the `diff` tool to find the differences in output which contains the information of what the instruction is, so that we can add `$display()` to show result in desired code fragment.