UNIVERSITY OF CALIFORNIA IRVINE

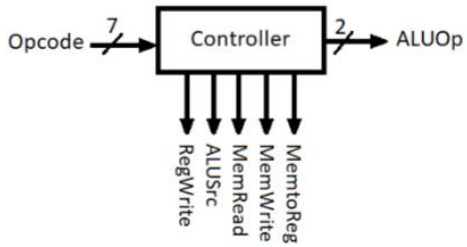
EECS 31L: Introduction to Digital Logic Laboratory (Spring 2024)

Lab 5: Single-Cycle RISC-V Processor

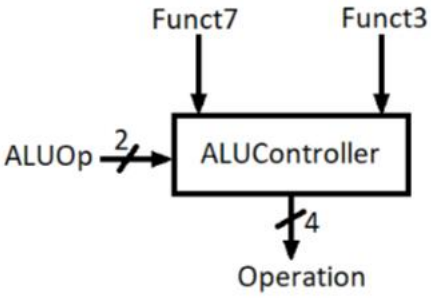
Kevin Li (LiKY4@uci.edu)

1. Overview

Lab 5 expands on Lab 4's Datapath by adding controllers. A controller and an ALU controller are added to this lab to further expand on the Datapath. These controllers provide input for the datapath, and also take in output from the datapath as their input. Combined, the controllers and the datapath make a full single-cycle RISC-V Processor.



The controller provides the RegWrite, ALUSrc, MemRead, MemWrite, and MemtoReg (all 1-bit) for the datapath. For the controller itself, these are outputs. Furthermore, the controller takes in the Opcode (7-bit) from the datapath as an input. The value of opcode determines the values for its output. In addition, the controller provides the ALUOp (2-bit) for the ALU controller.



The ALU controller provides the 4-bit ALU_CC for the datapath. This is its output. As for its input, the ALU controller takes in Funct3 (3-bit), Funct7 (7-bit), and ALUOp (2-bit) from the controller for its input. Depending on the value of these inputs, a different Operation (ALU_CC for the datapath) will be generated.

The full processor is the combination of the ALU controller, controller, and the Datapath built in lab4.

2. Hardware Design

Table 2 : Control Signals.

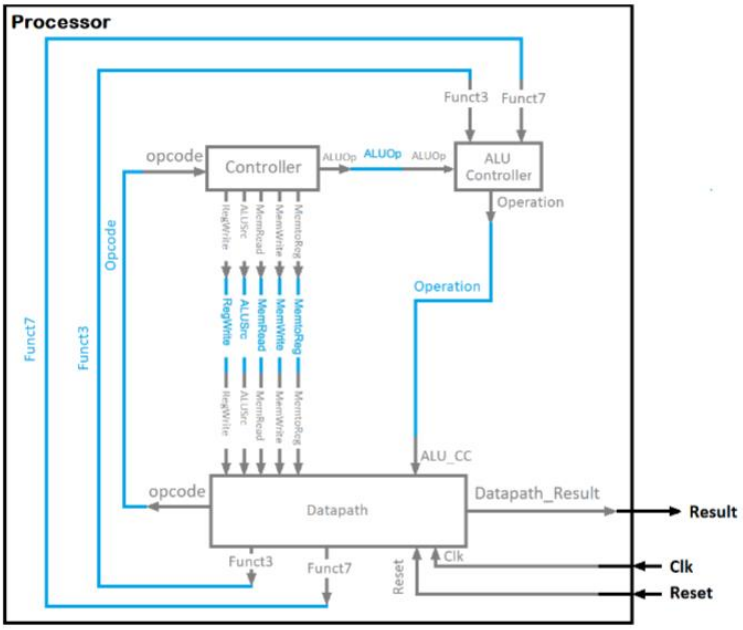
		Opcode			
		0110011	0010011	0000011	0100011
		AND, OR, ADD, SUB, SLT, NOR	ANDI, ORI, ADDI, SLTI, NORI	LW	SW
Control Signals	MemtoReg	0	0	1	0
	MemWrite	0	0	0	1
	MemRead	0	0	1	0
	ALUSrc	0	1	1	1
	Regwrite	1	1	1	0
	ALUOp	10	00	01	01

For designing the controller, I followed a truth table of values that shows the relationship of Opcode to MemtoReg, MemWrite, MemRead, ALUSrc, Regrite, and ALUOp. If the Opcode is 010011, then this relates to all the R-Type Operations. If the Opcode is 0010011, then this relates to the I-Type Operations. Finally, if the Opcode is 0000011 and 0100011, this relates to the Load and Store Operations respectively. I used a behavioral description to model this table with case statements.

Table 4 : The truth table for the 4 operation code.

	Funct7	Funct3	ALUop	Operation			
AND	0000000	111	10	0	0	0	0
OR	0000000	110	10	0	0	0	1
NOR	0000000	100	10	1	1	0	0
SLT	0000000	010	10	0	1	1	1
ADD	0000000	000	10	0	0	1	0
SUB	0100000	000	10	0	1	1	0
ANDI	-	111	00	0	0	0	0
ORI	-	110	00	0	0	0	1
NORI	-	100	00	1	1	0	0
SLTI	-	010	00	0	1	1	1
ADDI	-	000	00	0	0	1	0
LW	-	010	01	0	0	1	0
SW	-	010	01	0	0	1	0

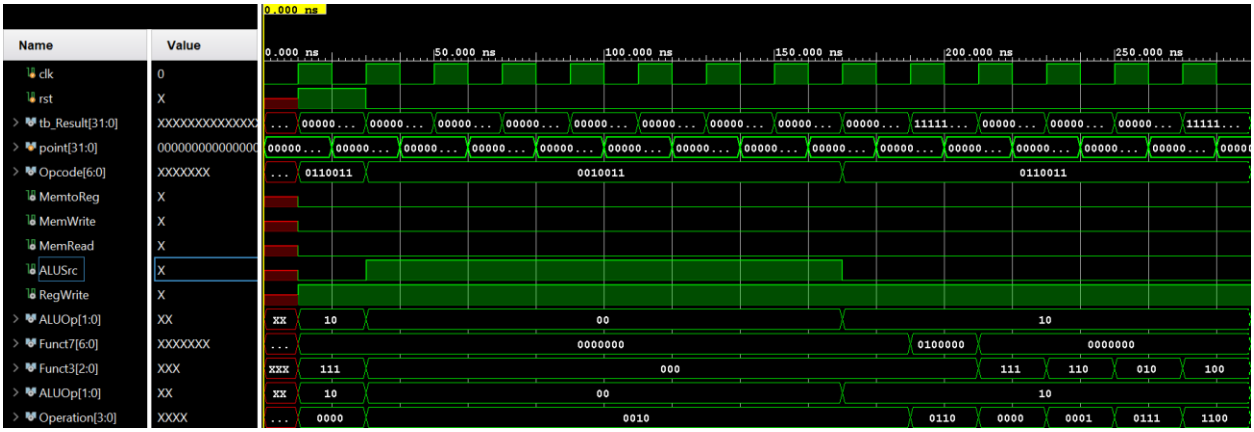
When designing the ALU controller, I also followed a truth table of values that showed the relationship between the inputs Funct7, Funct3, and ALUOp and the output Operation. Because there were many more bits and cases for the ALU controller than the controller, I used a dataflow description and assigned each bit of output Operation to a different Boolean expression. I found the Boolean expression by using minterm representation for each bit of Operation. For example, bit 2 in Operation can be expressed as AB'C'DE' + A'BC'DE' + AB'C'D'E' +A'BC'D'E' where ABC are the bits in Funct3 and DE are the bits in ALUOp. This can be simplified down to AB'C'E' + A'BC'E'.



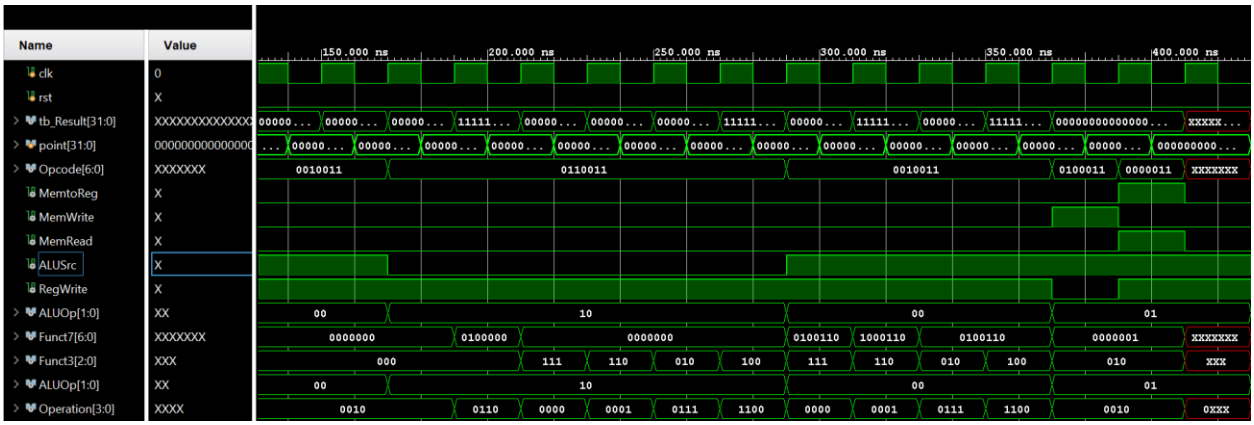
When designing the processor, I modeled it closely after this diagram. The blue lines were made into wires in my design module and connected to the appropriate ports of the controller, ALU controller, and datapath. The only inputs and outputs that the processor had was reset, Clk, and result (32-bit). To design this, I just instantiated my controller, alu controller and datapath modules and wired everything accordingly.

3. Simulation Results

Here are the waveform results of my simulation. The testbench was provided by the lab manual.



This above is the waveform from 0 to 280 ns.



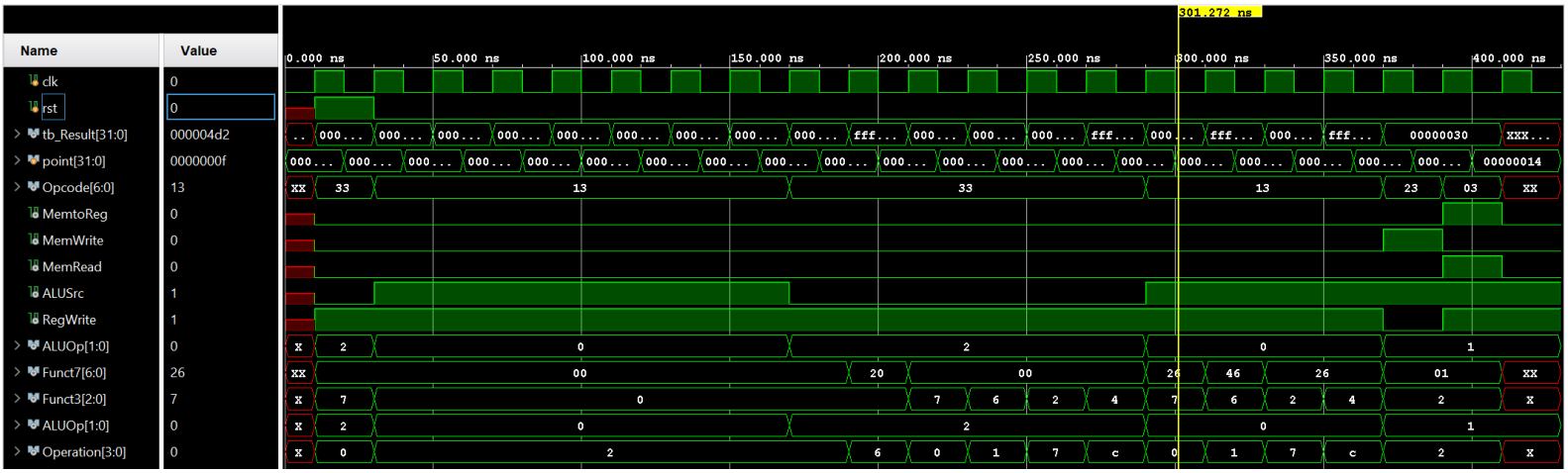
This above is the waveform from 130ns to 430 ns.

By looking at the truth tables of the controller and the alu controller, I found that Operation matched the expected value for each case depending on the value of Funct7, Funct3, and Opcode.

```
 The number of correct test cases is :         20
$finish called at time : 430 ns : File "C:/Users/kevin/UCI Stuff/Year 1 Quarter 3/EECS 31L/Lab 5 - Single-Cycle RI
INFO: [USF-XSim-96] XSim completed. Design snapshot 'tb_processor_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1000ns
launch_simulation: Time (s): cpu = 00:00:01 ; elapsed = 00:00:07 . Memory (MB): peak = 1575.664 ; gain = 0.000
```

Here is a display message I got from my waveform. I counted 20 different test cases in the testbench, and the number of correct test cases I got is also 20.

Full Waveform (Binary)



Full Waveform (Hex)