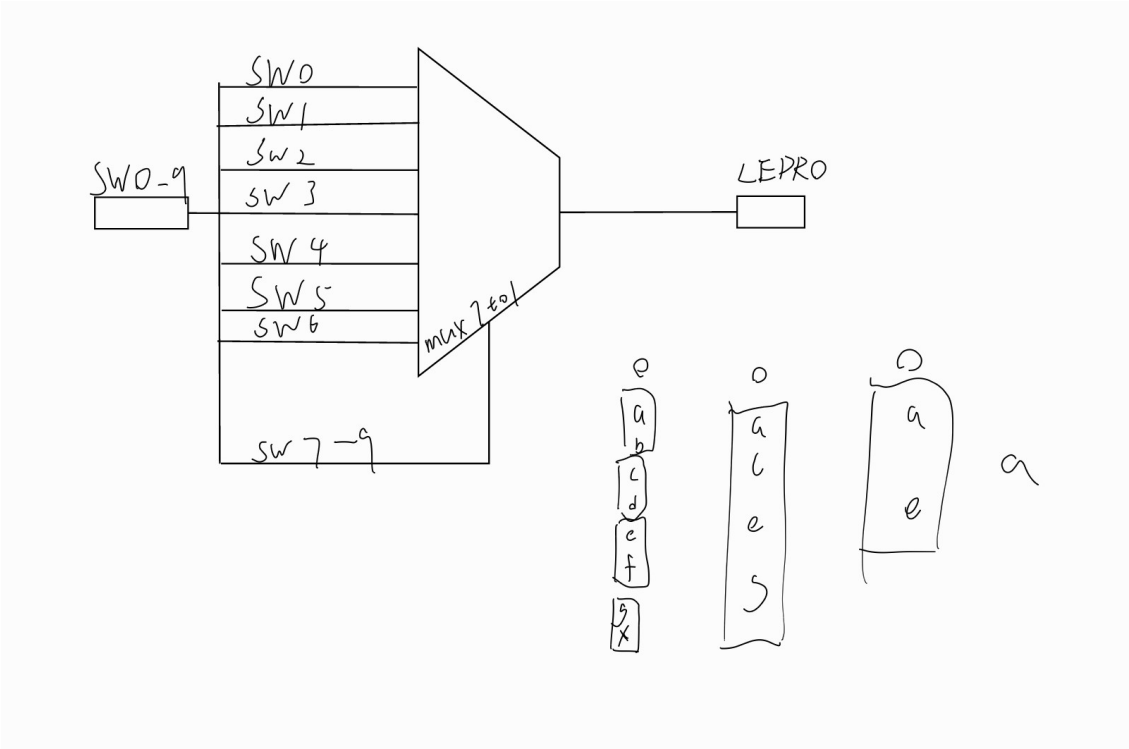
CSC258 Lab3 Report

Tianle Wang 1006337028

Part I

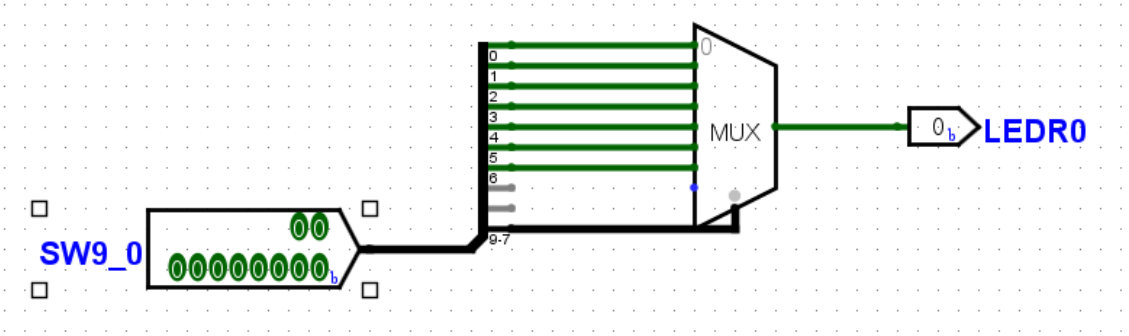
1. Before implementing this on Logisim, draw a schematic that shows how all of the inputs to a 7-to-1 multiplexer can be provided by a single multi-bit input source. Show how your overall design breaks down into interconnected modules and label all inputs, outputs and wires between modules. Use the following points to guide your design, and be prepared to explain your design approach to the TA as part of your preparation.



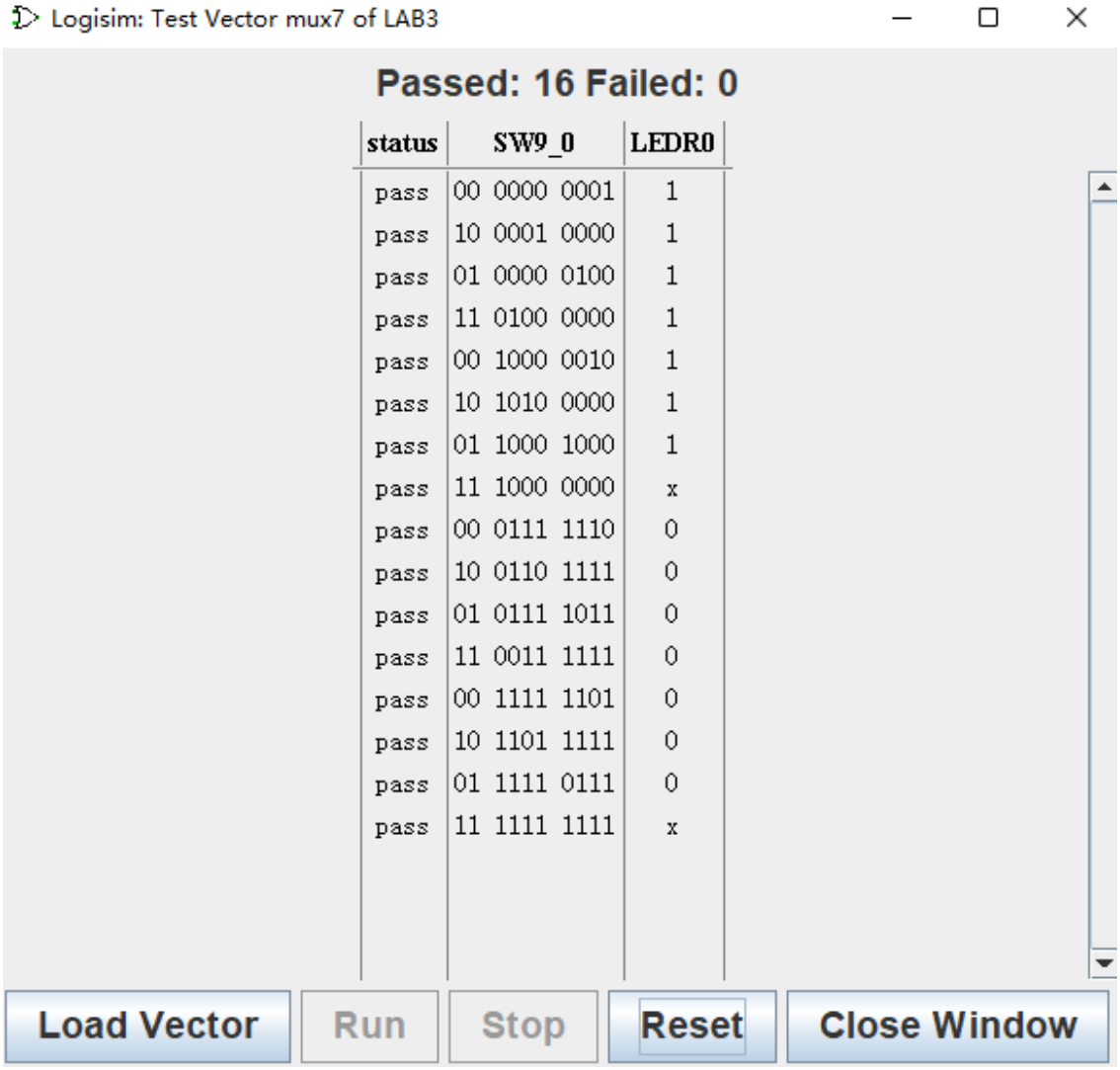
    1. Assume that at the highest level, the multi-bit input is coming from the DE1-SOC switches, with the first data bit coming from SW[0]. The output of the multiplexer will be displayed on LEDR0. This is for labeling purposes only, not because we expect you to upload your design onto the DE1-SOC board.
    2. How big does the multi-bit input need to be to provide all the inputs to the 7-to-1 multiplexer? Provide your answer in your pre-lab report.

    Answer: Since it asks 7-to-1 multiplexer, we can think of 4+2+1 where we need 3 more variables to control. Therefore totally 10-bits are needed.

2. Build your circuit in Logisim, using a 7-to-1 multiplexer and the splitter components. Make sure your implementation reflects your design in the previous step, including the labels.



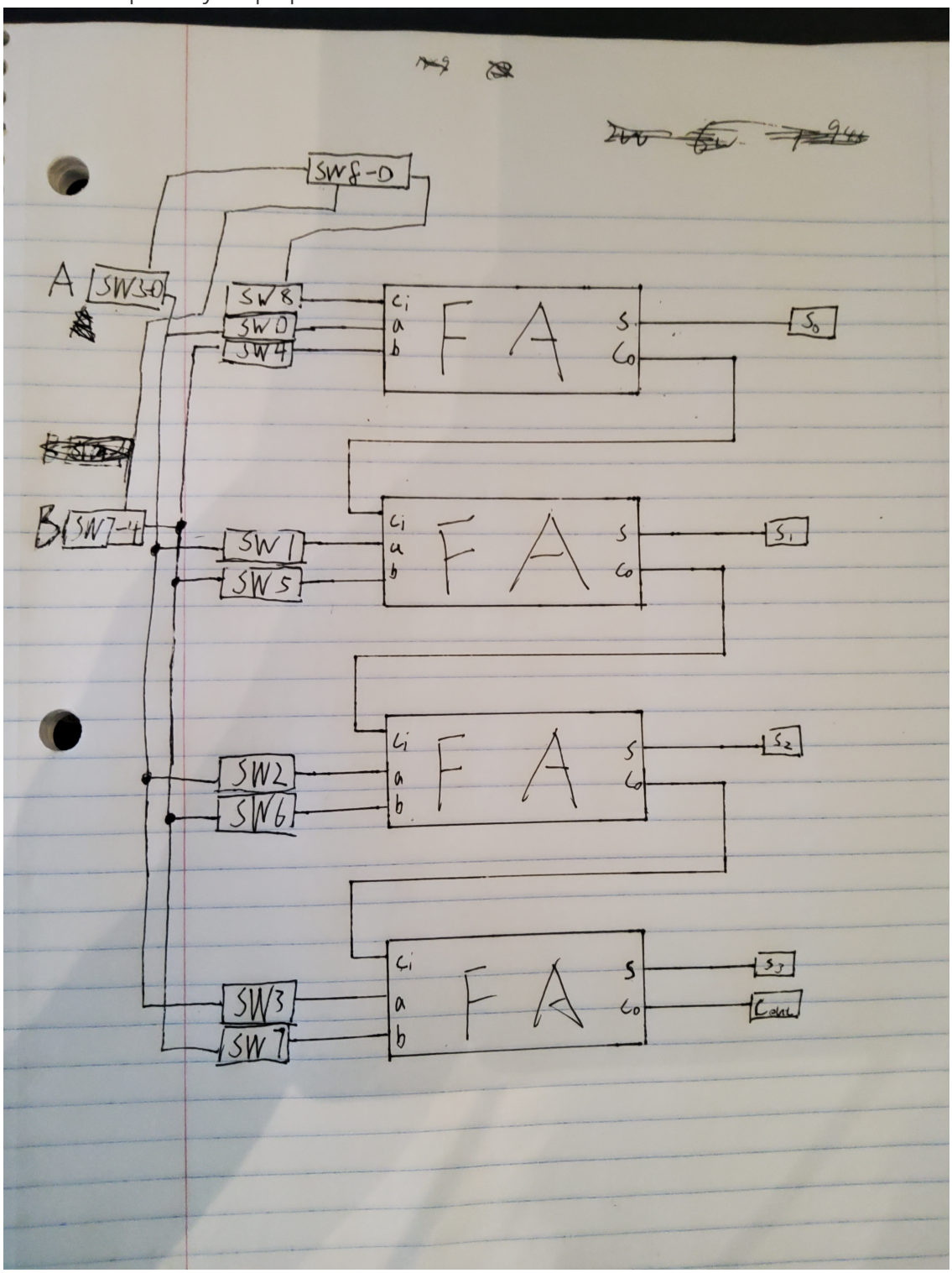
3. Simulate your circuit with test vectors in Simulate > Test Vector…. Include a screenshot of the simulation output as part of your prelab report.



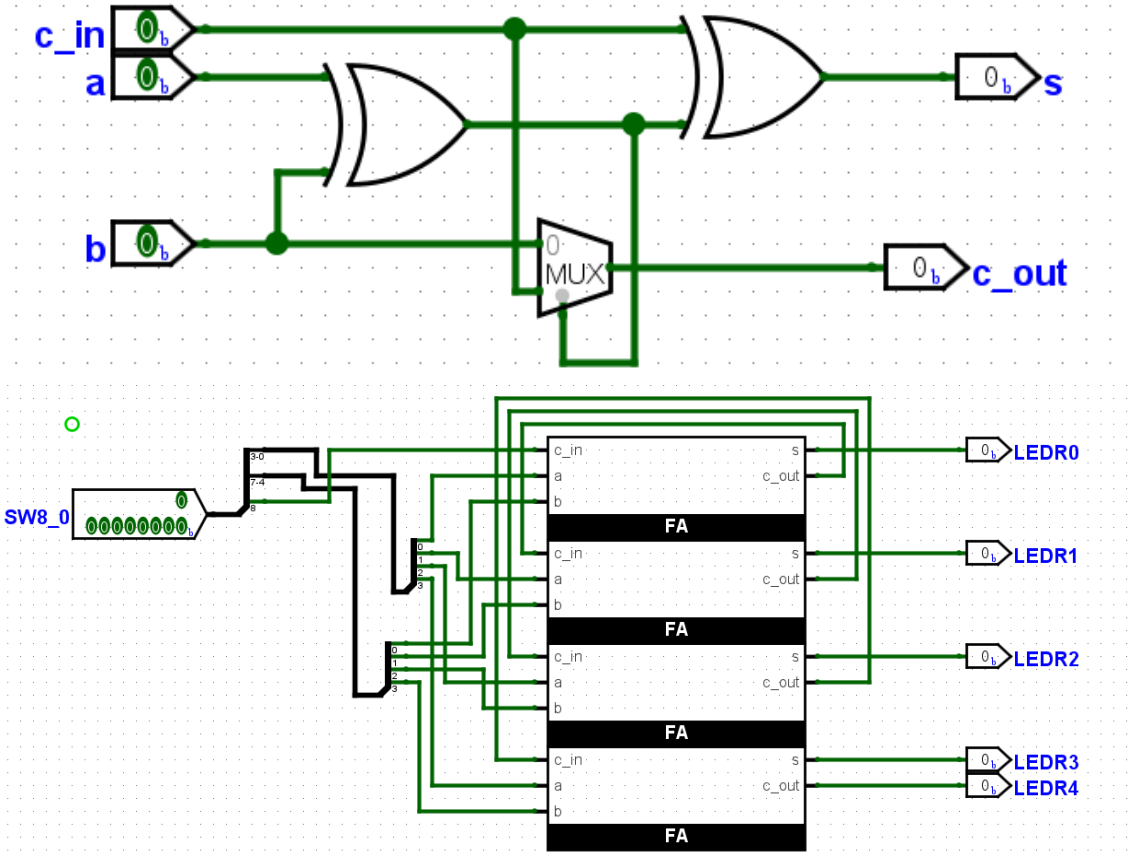
Part II

1. Draw a schematic showing your code structure with all wires, inputs and outputs labeled. Your schematic should resemble Figure 1(d), though it should also contain module and signal labels, and shows external connections to switches and LEDs (use SW3-0 for A, SW7-4 for B and SW8 for Cin. Use LEDR4-0 for the outputs). Be prepared to explain your design approach

to the TA as part of your preparation.

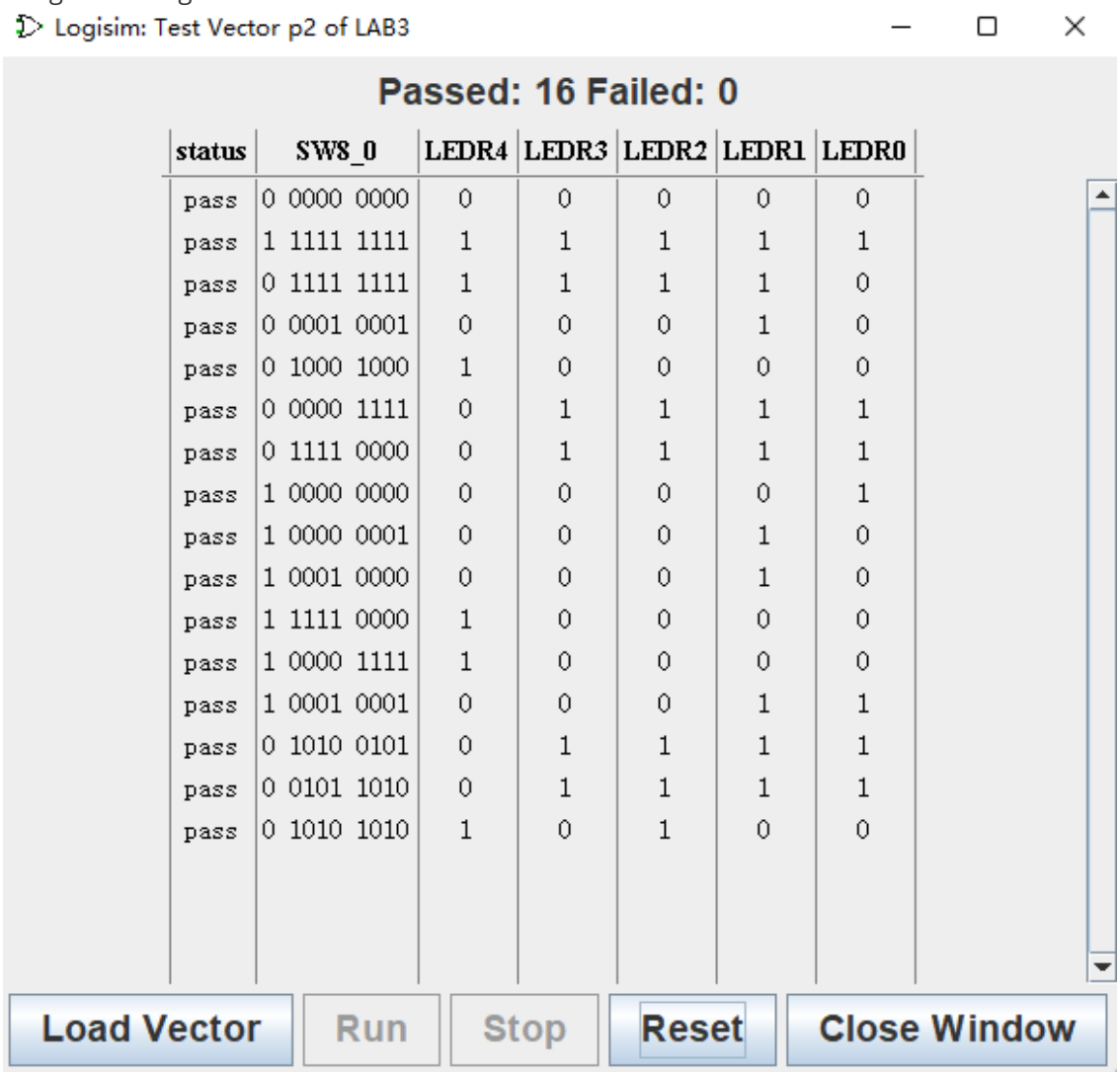


2. In Logisim, build the module for the full adder subcircuit. Once this subcircuit is complete, implement the larger module (the one resembling Figure 1(d)), which instantiates the four instances of this full adder. Name the input ports A, B and c_{in}, and the output ports S and c_{out}. Note: You should NOT use Logisim's built-in arithmetic addition component in your full-adder. Doing so will earn you 0 marks for this part.



3. Simulate your 4-bit ripple-carry adder with test vectors for intelligently chosen values of A, B and c_{in}. Include a screenshot of your test vector results in your prelab report. Note that as circuits get more complicated, it is not practical to simulate or test every single input case (in this case, 9 input bits would result in 2^9 test cases). This means that you must select an intelligently chosen subset of input values for your test vector. Here intelligently chosen means to find particular corner cases that exercise key aspects of the circuit. An example would be a pattern that shows that the carry signals are working, or how the circuit behaves with maximum and minimum values of A and B. Be prepared to explain why your test cases

are good enough.

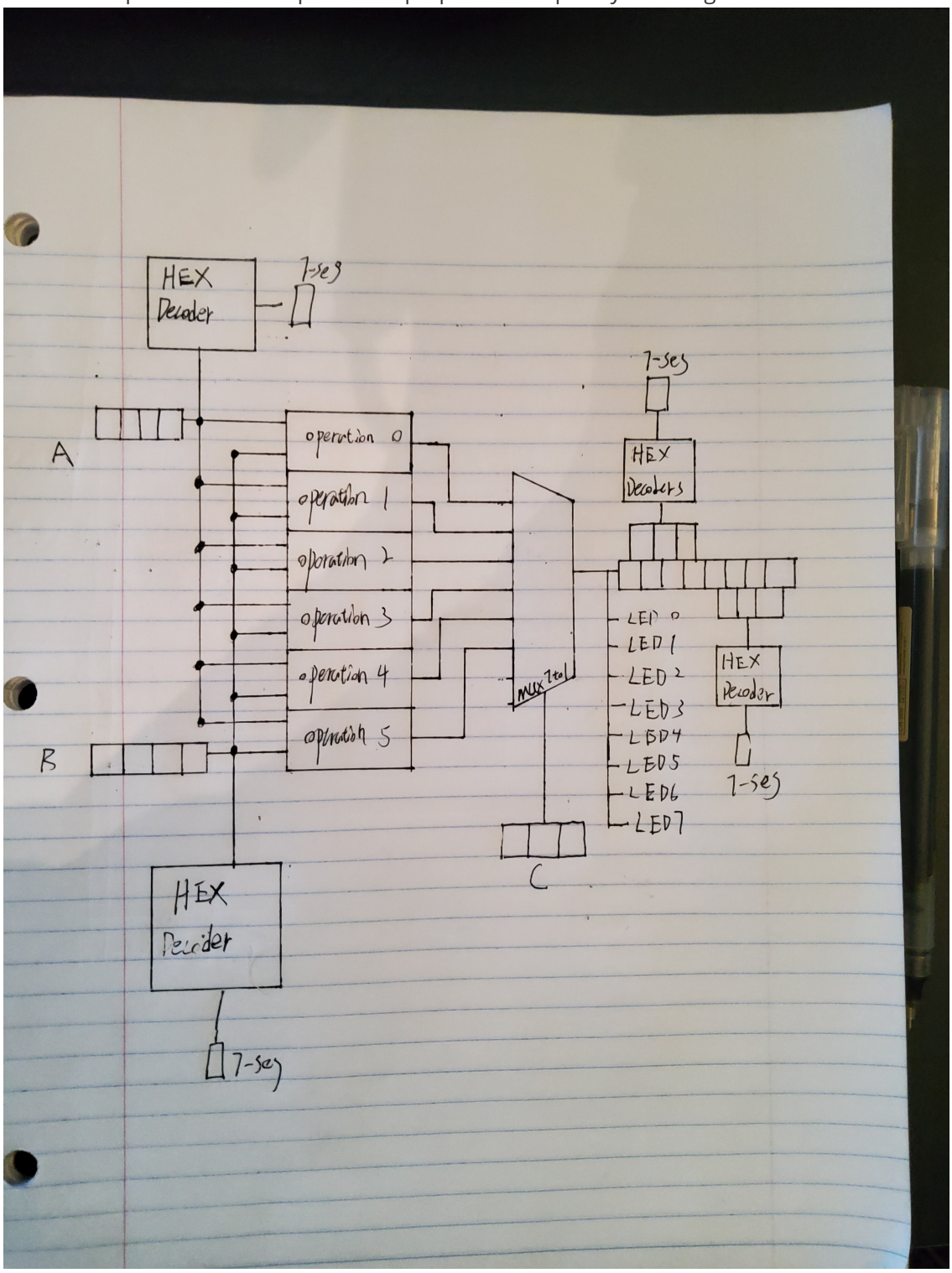


status	SW8_0	LEDR4	LEDR3	LEDR2	LEDR1	LEDR0
pass	0 0000 0000	0	0	0	0	0
pass	1 1111 1111	1	1	1	1	1
pass	0 1111 1111	1	1	1	1	0
pass	0 0001 0001	0	0	0	1	0
pass	0 1000 1000	1	0	0	0	0
pass	0 0000 1111	0	1	1	1	1
pass	0 1111 0000	0	1	1	1	1
pass	1 0000 0000	0	0	0	0	1
pass	1 0000 0001	0	0	0	1	0
pass	1 0001 0000	0	0	0	1	0
pass	1 1111 0000	1	0	0	0	0
pass	1 0000 1111	1	0	0	0	0
pass	1 0001 0001	0	0	0	1	1
pass	0 1010 0101	0	1	1	1	1
pass	0 0101 1010	0	1	1	1	1
pass	0 1010 1010	1	0	1	0	0

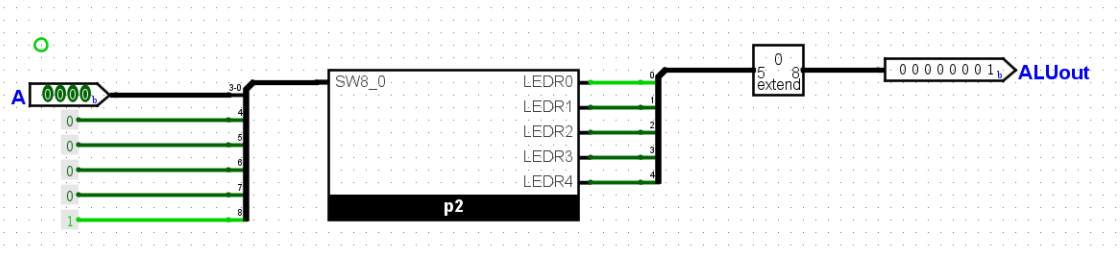
Part III

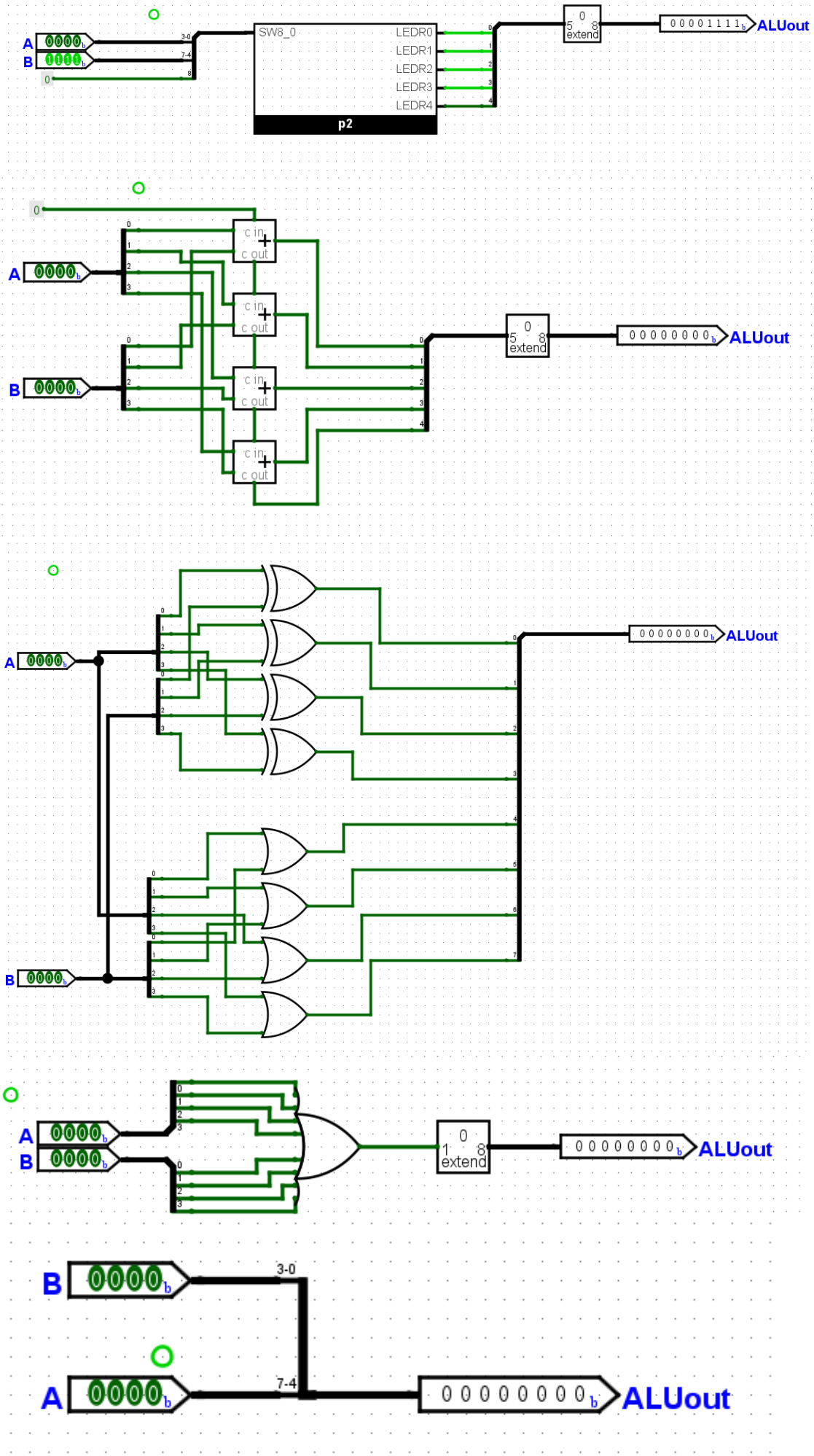
1. Draw a schematic showing your module structure as a block diagram with all wires, inputs and outputs labeled. Feel free to include multiple schematics if you wish to illustrate multiple levels in your design hierarchy. In particular, clearly highlight the multiplexer for your ALU as

well as all inputs to this multiplexer. Be prepared to explain your design choices to the TA

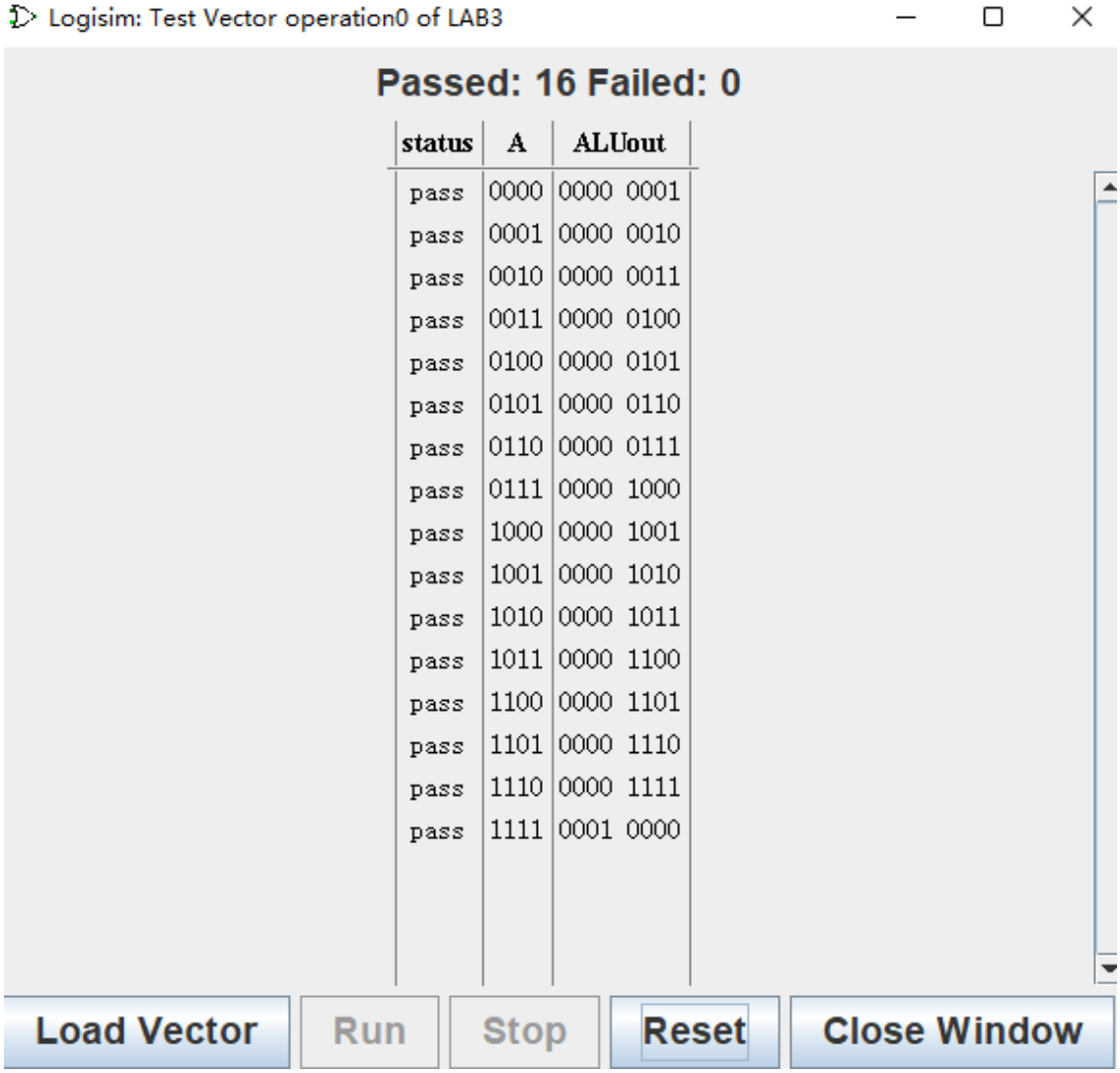


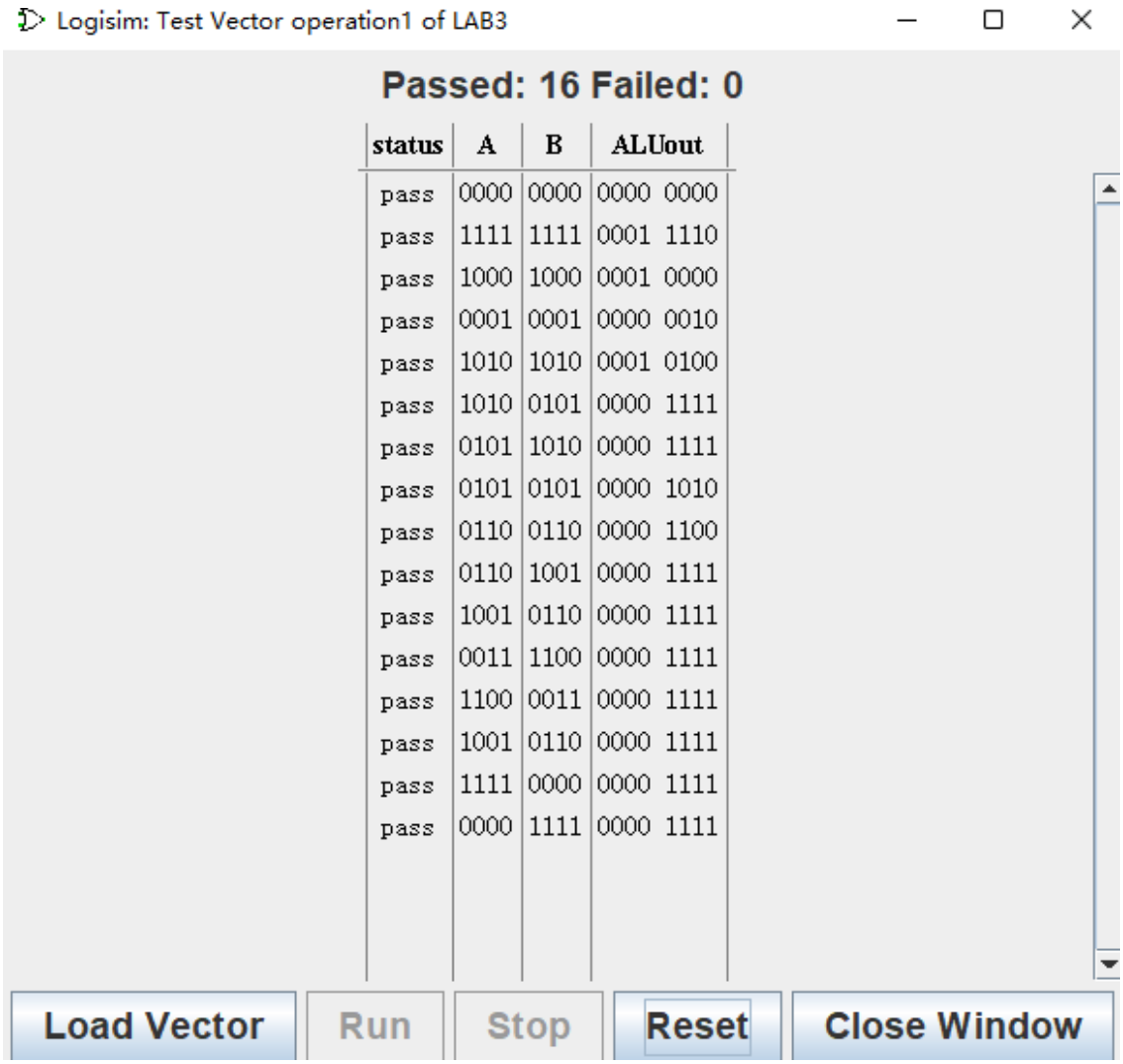
2. Build the Logisim module for the ALU including all high-level inputs and outputs.



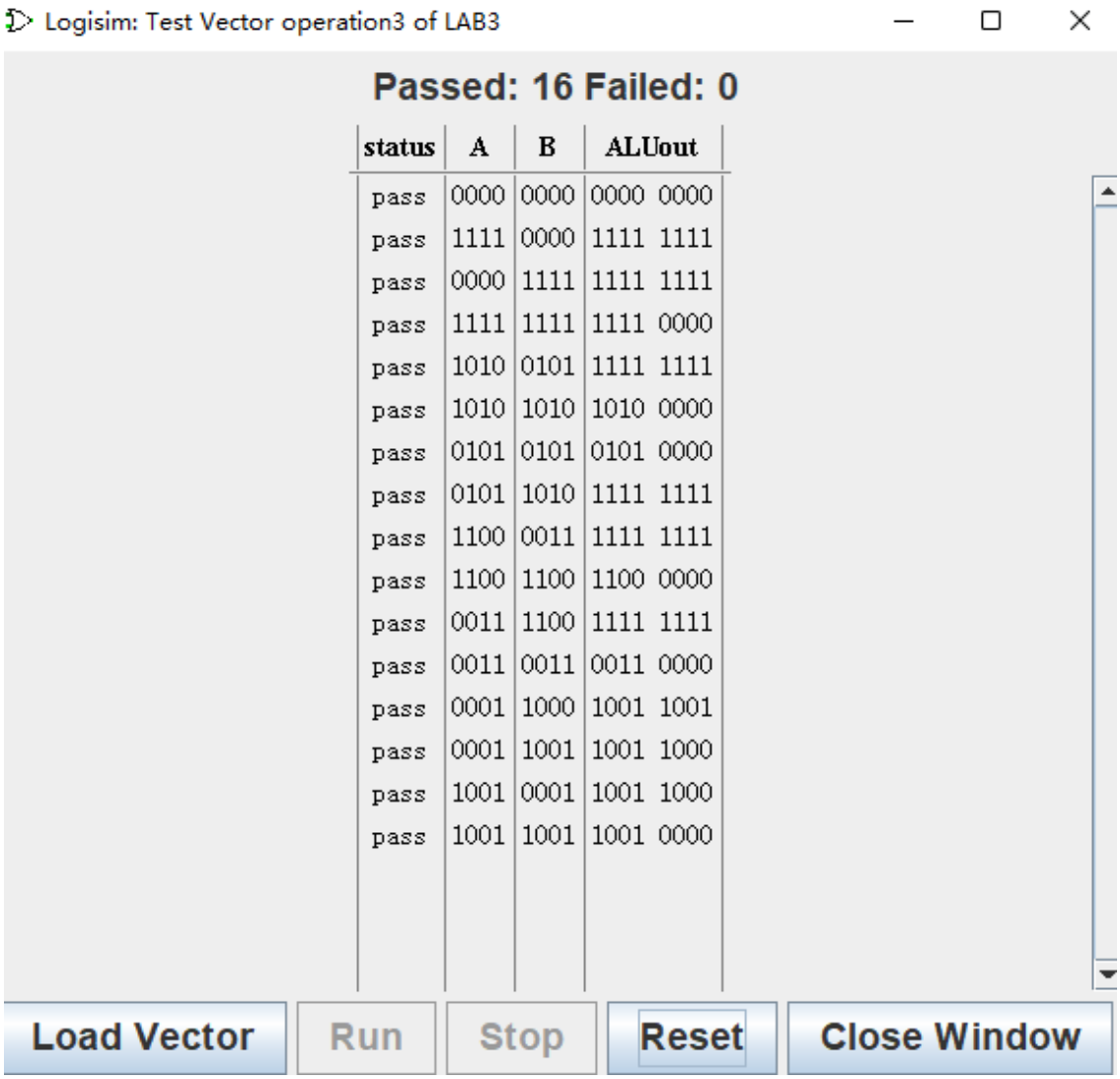

A
B
SW8_0
LEDR0
LEDR1
LEDR2
LEDR3
LEDR4
p2
extend
ALUout
c in
c out
3-0
7-4

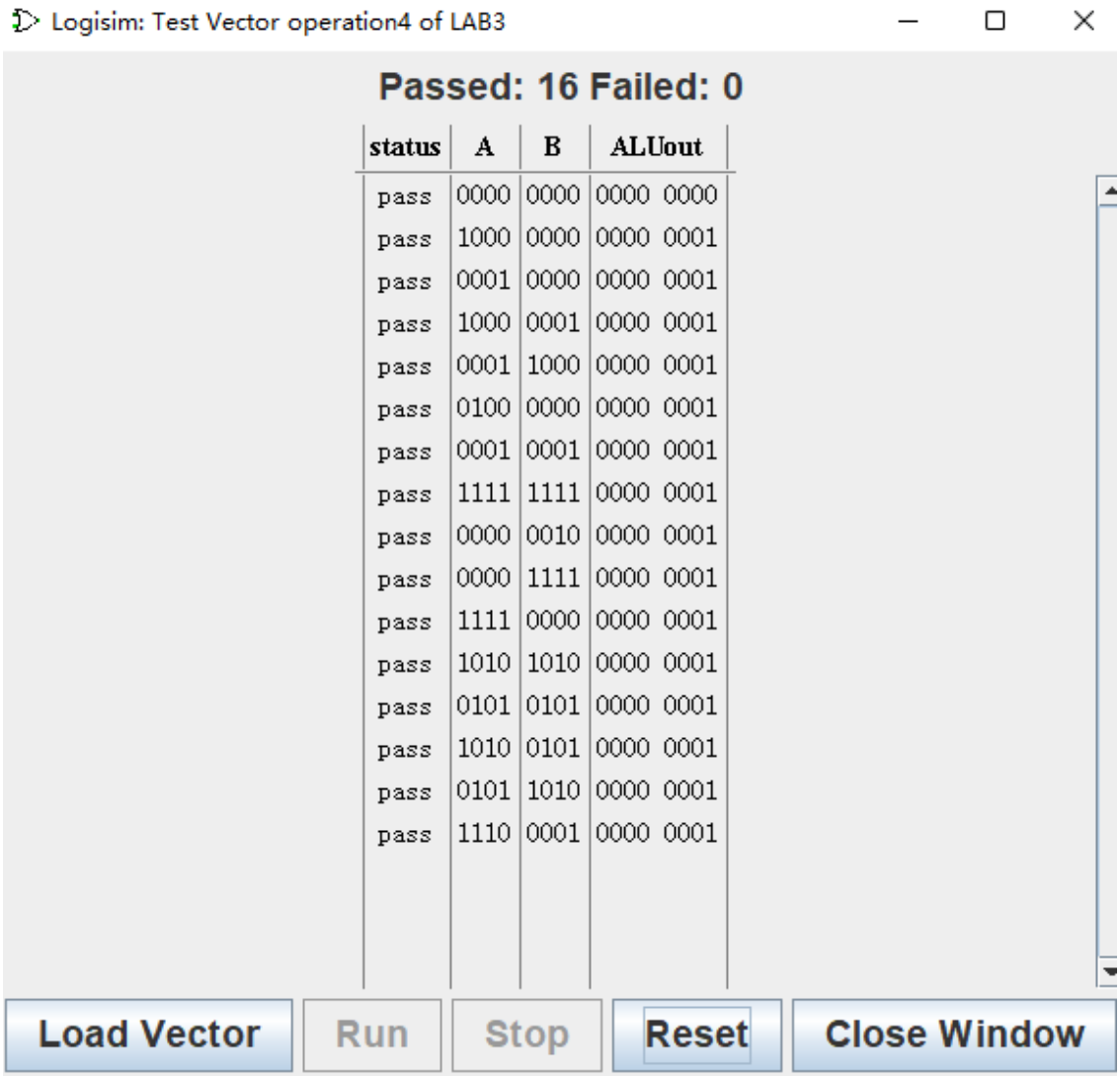
3. Simulate your circuit with test vectors for a variety of input settings, ensuring the all the tests are passing. Include screenshots of your successful test output as part of your prelab.

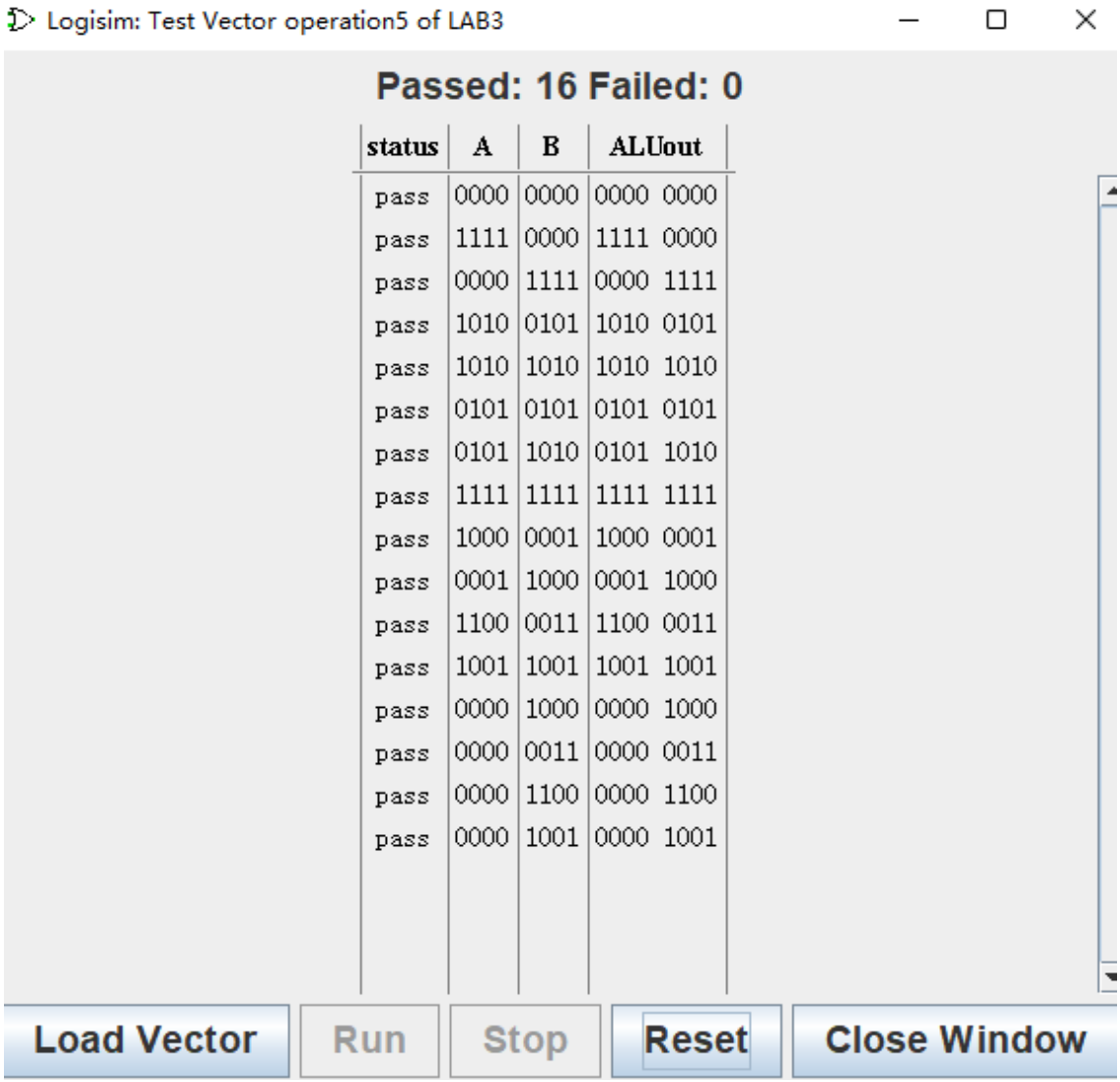



Logisim: Test Vector operation1 of LAB3
Passed: 16 Failed: 0
status A B ALUout
pass 0000 0000 0000 0000
pass 1111 1111 0001 1110
pass 1000 1000 0001 0000
pass 0001 0001 0000 0010
pass 1010 1010 0001 0100
pass 1010 0101 0000 1111
pass 0101 1010 0000 1111
pass 0101 0101 0000 1010
pass 0110 0110 0000 1100
pass 0110 1001 0000 1111
pass 1001 0110 0000 1111
pass 0011 1100 0000 1111
pass 1100 0011 0000 1111
pass 1001 0110 0000 1111
pass 1111 0000 0000 1111
pass 0000 1111 0000 1111
Load Vector
Run
Stop
Reset
Close Window

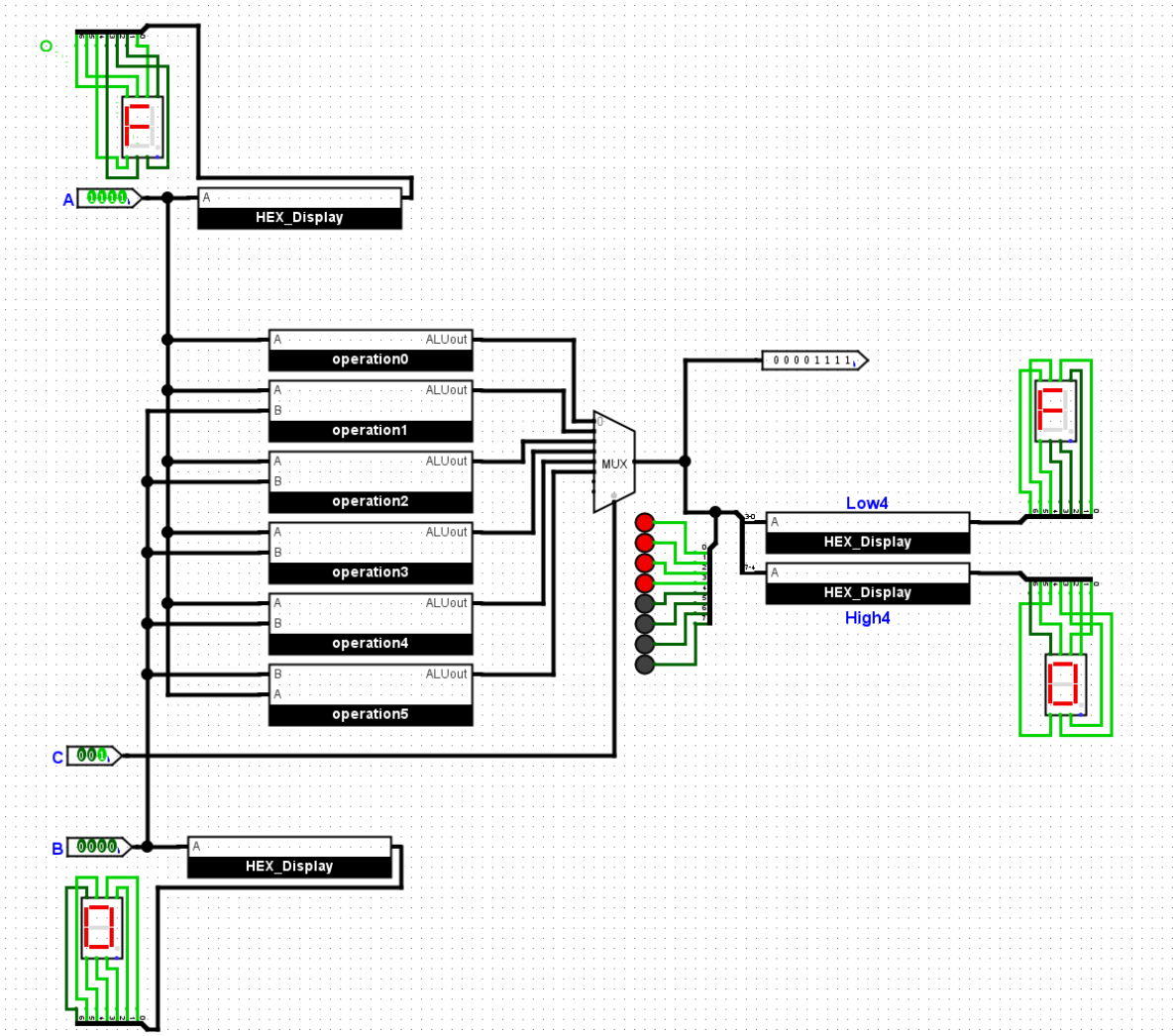

Logisim: Test Vector operation2 of LAB3
Passed: 16 Failed: 0
status A B ALUout
pass 0000 0000 0000 0000
pass 1111 1111 0001 1110
pass 1000 1000 0001 0000
pass 0001 0001 0000 0010
pass 1010 1010 0001 0100
pass 1010 0101 0000 1111
pass 0101 1010 0000 1111
pass 0101 0101 0000 1010
pass 0110 0110 0000 1100
pass 0110 1001 0000 1111
pass 1001 0110 0000 1111
pass 0011 1100 0000 1111
pass 1100 0011 0000 1111
pass 1001 0110 0000 1111
pass 1111 0000 0000 1111
pass 0000 1111 0000 1111
Load Vector
Run
Stop
Reset
Close Window


Logisim: Test Vector operation3 of LAB3
Passed: 16 Failed: 0
status A B ALUout
pass 0000 0000 0000 0000
pass 1111 0000 1111 1111
pass 0000 1111 1111 1111
pass 1111 1111 1111 0000
pass 1010 0101 1111 1111
pass 1010 1010 1010 0000
pass 0101 0101 0101 0000
pass 0101 1010 1111 1111
pass 1100 0011 1111 1111
pass 1100 1100 1100 0000
pass 0011 1100 1111 1111
pass 0011 0011 0011 0000
pass 0001 1000 1001 1001
pass 0001 1001 1001 1000
pass 1001 0001 1001 1000
pass 1001 1001 1001 0000
Load Vector
Run
Stop
Reset
Close Window


Logisim: Test Vector operation4 of LAB3
Passed: 16 Failed: 0
status A B ALUout
pass 0000 0000 0000 0000
pass 1000 0000 0000 0001
pass 0001 0000 0000 0001
pass 1000 0001 0000 0001
pass 0001 1000 0000 0001
pass 0100 0000 0000 0001
pass 0001 0001 0000 0001
pass 1111 1111 0000 0001
pass 0000 0010 0000 0001
pass 0000 1111 0000 0001
pass 1111 0000 0000 0001
pass 1010 1010 0000 0001
pass 0101 0101 0000 0001
pass 1010 0101 0000 0001
pass 0101 1010 0000 0001
pass 1110 0001 0000 0001
Load Vector
Run
Stop
Reset
Close Window



4. Prepare your design and implementation for your in-lab demo.


O
A
0000
A
HEX_Display
A
ALUout
operation0
A
B
ALUout
operation1
A
B
ALUout
operation2
A
B
ALUout
operation3
A
B
ALUout
operation4
B
A
ALUout
operation5
MUX
C
000
B
0000
A
HEX_Display
00001111
Low4
A
HEX_Display
High4
A
HEX_Display