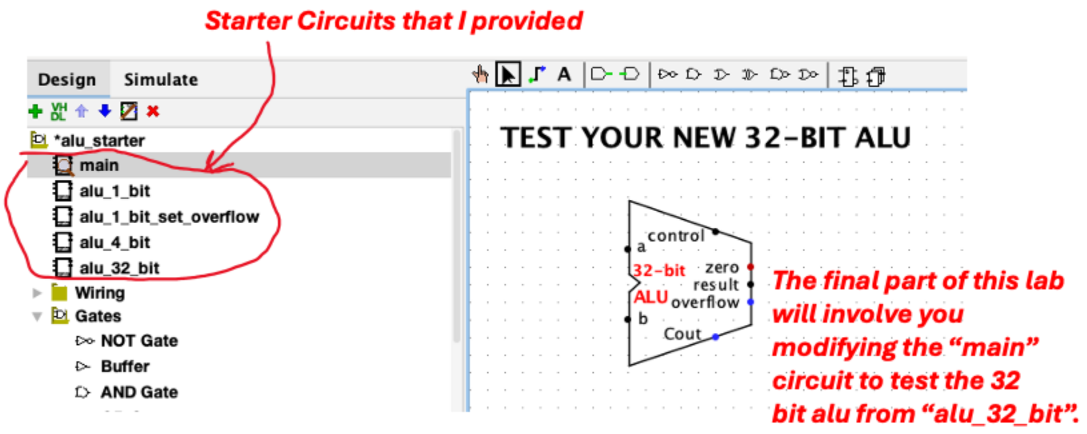
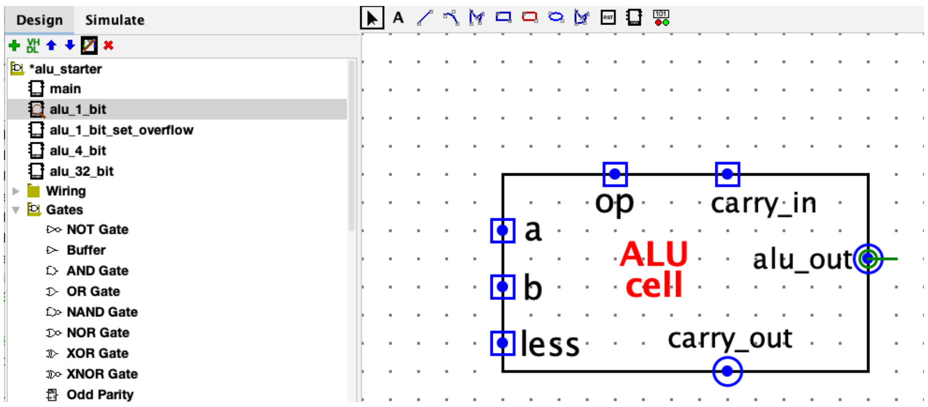
Lab6 – 32 Bit ALU Construction

Introduction
For this lab you will be using Logisim Evolution to build a 32 bit ALU from a starter file that I am providing.

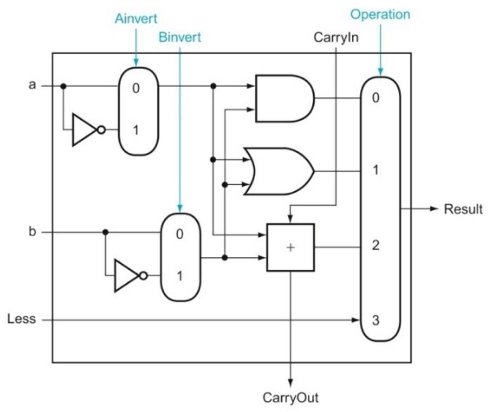
PART 1: Getting Familiar with the Starter File
The starter circuit file that I provided, "alu_starter" contains a main circuit and 4 sub-circuits. For this lab I created the inputs and outputs in each sub_ciruit, you will be implementing the details. The main circuit is used as a test bed, after completing each sub circuit you can test it in the main circuit.



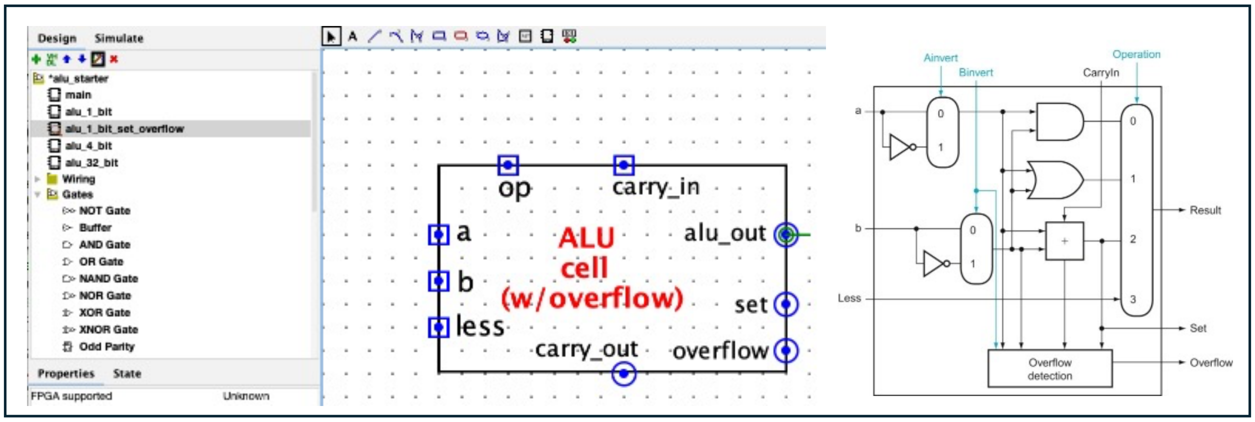
PART 2: The 1-Bit ALU Cell
After you become familiar with the layout of the starter file, start by attacking the 1-bit ALU cell. I have already created the layout consisting of the inputs and outputs, you need to do the implementation.



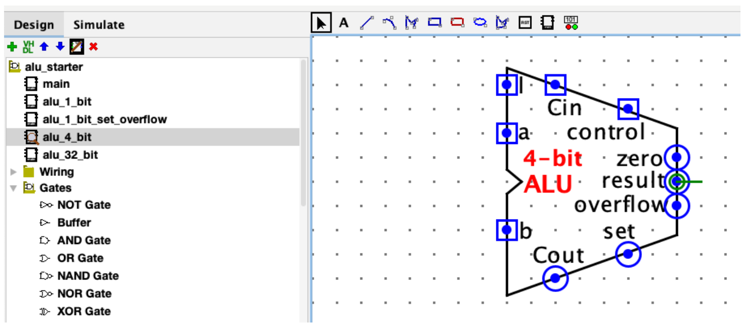
Recall that the implementation of the basic 1-bit ALU cell from class lectures, this is what you will be implementing in Logisim:

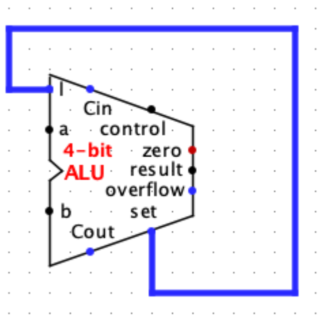


PART 3: The 1-Bit ALU Cell with SET and OVERFLOW
For this part, implement the “alu_1_bit_set_overflow” circuit. You can copy the implementation from part 2 “alu_1_bit” as a starter. From there wire up the set and overflow logic that I provided in the starter file



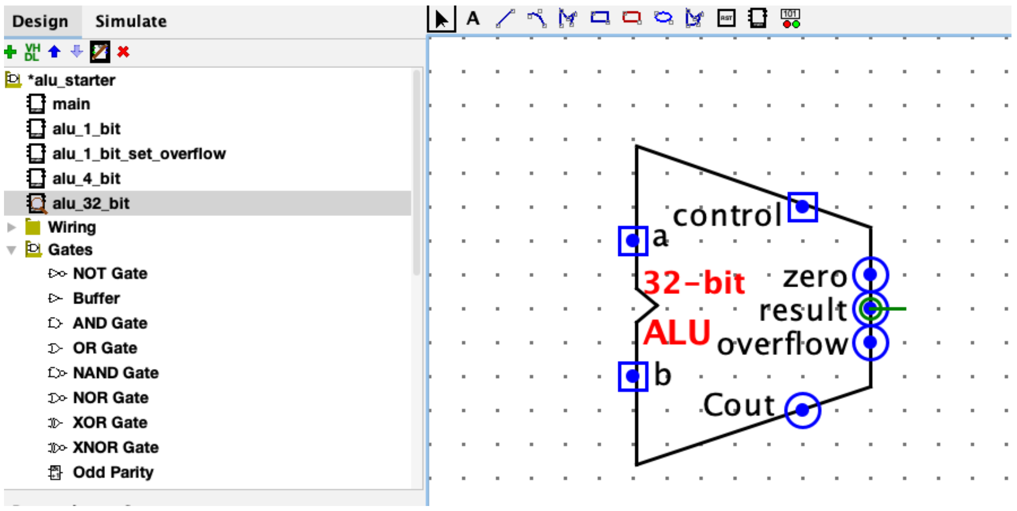
PART 4: The 4-Bit ALU Cell
Now that we created the 1-bit building blocks, implement the 4-bit ALU cell from the starter circuit. Note that bits 0,1, and 2 use the basic cell (from part 2), and bit 3 will use the extended cell (from part 3). Also note that because we want to use this cell as a building block for the 32-bit ALU we need to expose the less input that goes into bit 0 from this version of the ALU. You can make this 4-bit ALU work for SLT by wiring up the “set” output to the “less” input as shown below on the right. After you are done building this, make sure you test this circuit because it will be used to implement the full 32-bit ALU in Part 5.





PART 5: The 32-Bit ALU Cell

Now that we created a 4-bit ALU in part 4, we can use 8 of these to implement a full 32-bit ALU. The starter for this circuit is in “alu_32_bit”. Implement the full 32 bit ALU and test it

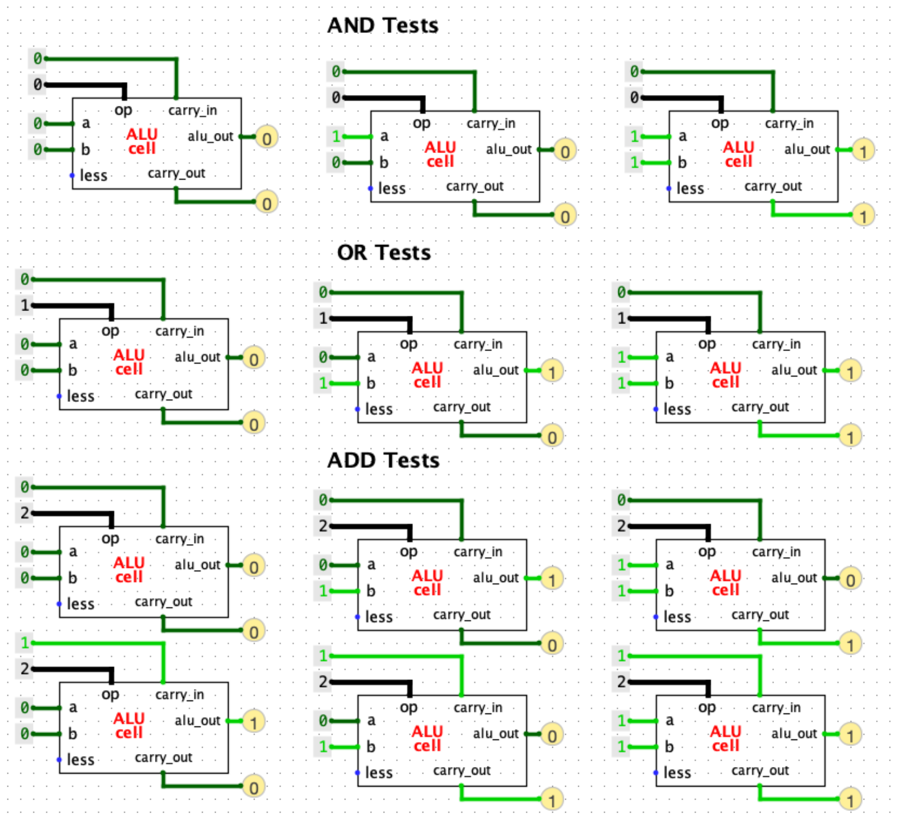


PART 6: Testing and Validation

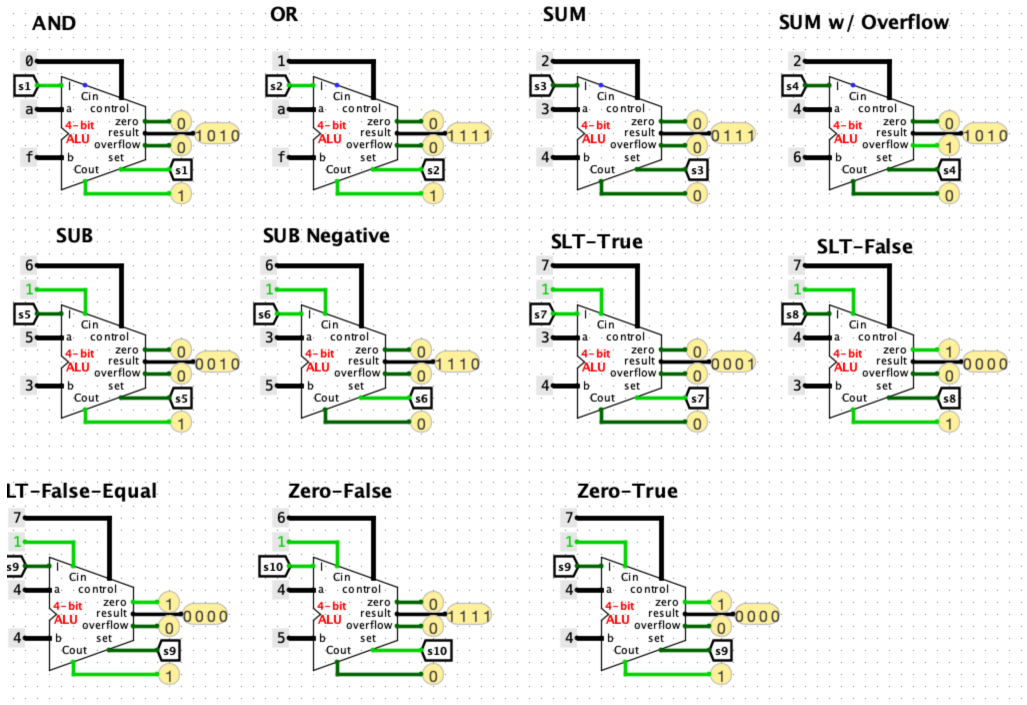
Notice in the starter circuit I provided there are 3 test circuits – test_1_bit, test_4_bit, test_32_bit. I provided implementations for the test_1_bit and test_4_bit_circuits. After implanting your circuits your output for these should look like the pictures below.

I did not create a test_32_bit circuit. Please create one using the test_4_bit circuit as a model. Try larger inputs to drive a full 32 bit alu to show its operating correctly.

test_1_bit:



test_4_bit:



Please hand in using blackbord 4 things:

1. **Your Logisim .circ file after you have implemented all of the circuits described in Parts 2-5. This should also have the 3 test circuits described in Part 6.**
2. **Along with handing in your Logisim file, please provide screen shots of the following circuits:**
   a. **test_1_bit**
   b. **test_4_bit**
   c. **test_32_bit**