Purpose

In this lab you are going to demonstrate and validate your understanding of the topics you learned in Chapter 4.

You are asked to design and simulate following logic circuits using Logisim tool.

- Adders
  - Half adder
  - Full adder
  - Half- and Full- Adders Subcircuits
  - 3-bit adder
- Implementing Boolean functions using a Multiplexer

There is a 20 mark, all, or nothing extra credit.

Before you begin

For this lab work, we expect you have completed Lab 0 and Lab 1 entirely and your lab environment is ready to start this lab assignment. If you haven't, please refer to the Lab 0 instructions. Again, you are allowed to use any working lab environment and in fact, you may store your latest work in a secondary storage (like a thumb drive) or cloud system so that you can work from the latest copy from any environment at your convenience.

Also, you may create a new folder for Lab 4 to store all your circuits there. If you are using Mac, please place that folder in your home directory. You can use Logisim Evolution, but you need to clearly state that you're using it in your submission pdfs.

1. Part 1: *Adders*

For this part of the lab, you will be creating one single project named `Lab4Part1.circ` *and you will be creating multiple canvas and circuits on it.*

- Open a new project and save it as `Lab4Part1.circ`

1.1 Half Adder (10 points)

1. Select the main canvas.
2. Add a label with "**Half_Adder**" on it.
3. Add multiple labels stacked line by line to draw a half-adder truth table.
   - 2 inputs: input X and Y. How many rows must be in the truth table?
4. Create your half-adder logic diagram on the Canvas.
   - Name the upper input "X" and the lower input "Y."
   - The outputs should be named "SUM" and "CARRY."
   - **IMPORTANT: Use XOR** gate to implement a half-adder
5. Simulate your design for all combinations of inputs and make sure all the output works as expected. You may use 4 data bits.

1.2 Full Adder (10 points)

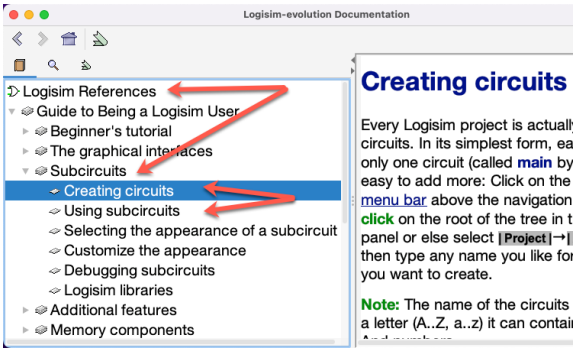
1. In the same project, and in the main canvas, add a label with "**Full_Adder**" on it.
   - Add some space to separate from part 1.1.
2. Add multiple labels stacked line by line to draw a full adder truth table.
   - 3 inputs: input X, Y, and Carry-In. How many rows must be in the truth table?
3. Create your full-adder logic diagram on the Canvas.
   - Name the upper input "X" and the lower input "Y."
   - The outputs should be named "SUM" and "CARRY."
4. **IMPORTANT:**
   - **You must use two half_adders** (design implemented in Part 1.1 to implement a full adder.
   - Wire the SUM output of half adder "A" to the input of the other half adder "B"
   - The Carry-In input will be the input the other input to half adder B.
   - The SUM output of half-adder B will be the "SUM" output of the full adder.
   - The "CARRY" outputs from both half adders should be `OR` 'ed and its result will be the "CARRY" output of the full adder.

1.3 Half-Adder and Full-Adder subcircuits (25 points)

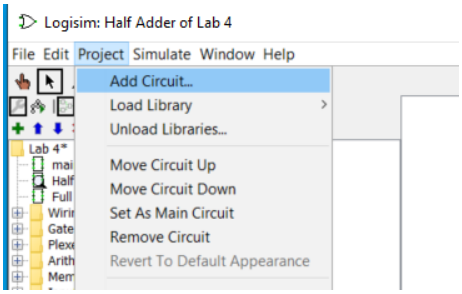
In this exercise, you are going to create a Half-Adder subcircuit which is very much like a function (a.k.a. subroutine) in a programming language. The purpose of creating a subcircuit is just like those of creating function in a programming language.

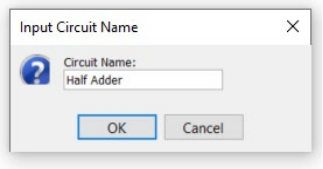
- Modularized approach for efficient circuit design
- Reusability of circuits
- Maintainability of circuits
- Readability of circuits

1. First read the "Subcircuits" section in Logisim Reference.
   - Select Help→Tutorial from the menu bar
   - Expand Subcircuits
   - Read all subsections but at least the first two sections, "*creating circuits"* and "*using subcircuits"*.



2. Now, let's create a subcircuits for ***Half_Adder***
   - Click Project→Add Circuit menu item from the menu bar
   - Provide "***Half_Adder***" in the Circuit Name textbox then click on OK
   - This will add a new circuit (***Half_Adder***) In the navigation panel.
   - Then double-click to the main Canvas.
   - Select the entire half adder circuit on the canvas.
   - Copy & paste the selected half adder circuit in the Half Adder subcircuit window.
   - Make sure the data bits of all gates and inputs/outputs are set to 1.
   - That's it. You just created a "***Half_Adder***" subcircuit.



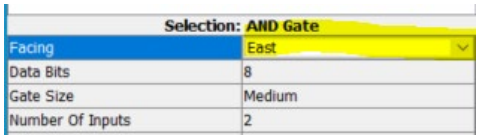


3. With that, now let's create a "***Full_Adder***" subcircuit
   - Follow the same process to create a new subcircuit named "Full Adder"
   - Double-click the "***Full_Adder***" subcircuit.
   - In the empty canvas, add 2 Half Adders by clicking and dragging the "***Half_Adder***" subcircuit into the Canvas
   - Add an `OR` gate.
   - Add inputs and outputs.
   - Now properly wire them
   - Simulate the full adder with the Poke tool.
   - Now, you just created a Full Adder subcircuit.

1.4 3-bit Binary Adder (25 points)

In this exercise, you will use the Full Adder subcircuit you just created in the section 1.3 to create a 3-bit adder. Each adder will have 3 inputs, two inputs to add and the remaining one for carry-in from the previous bit. It will produce two outputs, one for Sum and the other for Carry-out.

1. Design a 3-bit adder logic circuit on a piece of paper
2. Double-click "main" circuit in the explorer pane to go back to the main canvas
3. Create a new label then enter "3-bit Binary Adder" at the bottom of the canvas
4. Click "***Full_Adder***" subcircuit from the explorer panel and drag it to the canvas.
   a. Repeat it to create 3 full adders.
5. Add 7 inputs: 2 for each adder and one carry-in input (always set to 0)
6. Add 4 outputs: 1 for each adder Sum output and one carry-out output for the last bit.
7. Properly arrange inputs, outputs, and full adders and make sure they are facing in right direction.
   a. **NOTE**: you can change the "**Facing**" property of pins, gates, subcircuits including your own to North, South, East, or West.



8. Simulate the full adder with the Poke tool.
   a. Setting x to '`101`' and y to '`001`' should result in `110` (Sum)
9. Save the project.

2. Part 2: Implementing Boolean Functions using Multiplexers (30 points)

The multiplexer, shortened to "MUX" is a combinational logic circuit designed to switch one of several input lines through to a single common output line by the application of a control signal.

You can also use multiplexer IC's to implement Boolean Functions without requiring complex logic gates. Logisim also supports Multiplexer subcircuits.

Using a `4x1 Multiplexer`, found in "Plexers" in the Explorer pane, implement following Boolean function $F(w, x, y, z) = \sum m\ (1,4,5,7,8,11,14,15)$

NOTE: the right most bit of MUX selector bits is reserved for `'Enable'`.

Create a new Logisim project and save it as `Lab4.part2.circ`.

3. Part 3 - Extra Credit (20 points)

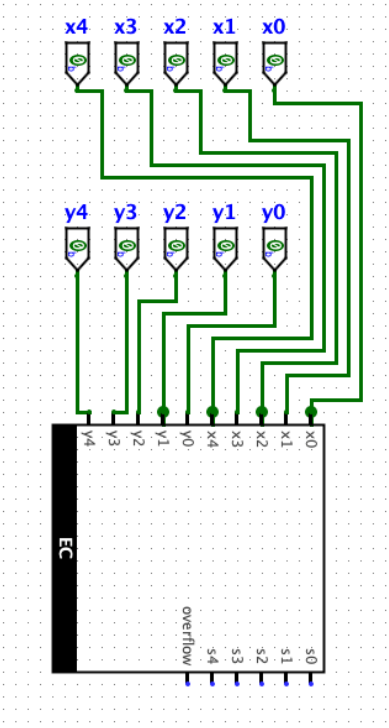
Create a Logisim project called `Lab4.part3.circ`.

3.1 Multiplexers (10 points)

Implement the same Boolean function using `8x1 MUX`: $F(w, x, y, z) = \sum m\ (1,4,5,7,8,11,14,15)$

3.2 6-bit Signed Binary Adder with Overflow Detection (10 points)

In this task, you will design a new circuit that takes two 6-bit signed binary numbers, using the 2's complement representation, and outputs the sum of these numbers. Additionally, the circuit will include an output to indicate when the result is incorrect due to overflow. Overflow detection is achieved by examining the **carry-in** and **carry-out** in the sign bit. Further information can be found in the provided resources in addition to your textbook.



Here you will find more information.

Follow these steps to complete the task:

1. Add a new canvas by adding a new circuit and name it "EC"
2. You should reuse the full adders from part 1. You can copy and paste the circuit components into your new project.

3. Add another new canvas by adding a new circuit and name it “Binary_Addition_Test”
4. Add one of your EC circuits. Arrange the two 6-bit number inputs are at the top, display the result at the bottom, and display the overflow bit on the bottom left corner.

Your final test circuit should look like the image provided on the right. Note that the image above shows 5-bit binary adder but you are asked to implement 6-bit adder in this assignment.
To evaluate your circuit, simulate it using various examples and also perform manual verification. Include screenshots of your tests and the manual verification process in your PDF report. Additionally, demonstrate the manual calculation of the sum for clarity.
Here are suggested test cases: 1 + (-4), 6 + 7, 12 + (-2), 9 + 8, -7 + (-12)

4. Deliverables

You need to submit **at least 4 files.** And a total of 6 if you do the extra credit.

Include your name in your PDF files, as a header or in the first page.

1. **Part 1:**
   - **a.** `Lab4Part1.circ` (Circuit)
   - **b.** `Lab4Part1.pdf` (Include screenshots of all your circuits)
2. **Part 2:**
   - **a.** `Lab4Part2.circ` (Circuit)
   - **b.** `Lab4Part2.pdf` (Include screenshot of all your circuits)
3. **Part 3: - Extra Credit**
   - a. `Lab4Part3.circ` (Circuit file)
   - **b.** `Lab4Part3.pdf` (Include: truth table, K-Map and Circuit screenshots)

Submit all your work to canvas. Double check your submission by downloading it and opening all downloaded files to make sure everything is correct.