

INTRODUCTION TO DIGITAL SYSTEMS

SYSC2310 (Fall 2020)

Dr. Mostafa Taha

Lab 1: Introduction to Logisim and Basic Logic Gates

Objectives

- Get familiarized with the basic features of Logisim.
- Introduce, design and test basic logic gates.
- Build a 1-bit half adder and 1-bit full adder.
- Introduce the concept of hierarchy by building an 8-bit adder, using the 1-bit full adder.

Relation to course outcomes

The work in this lab is related to the two following course outcomes:

- Understand the basic building blocks of digital systems.
- Apply analysis skills to correctly describe the behavior of given combinational logic circuits.

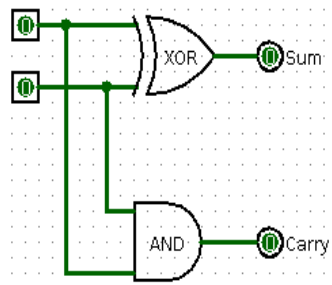
Preparation

- Read the Logisim Guide posted on cuLearn attentively prior to the lab.
 - Take the time to read this document carefully to understand the essence of the work to be performed.
1. Download Logisim from cuLearn according to your operating system. You may have to install legacy java 6 using the link in cuLearn.
 2. Create a new folder with the title 'sysc2310lab'.
 3. Inside the 'sysc2310lab' folder, create five folders: 'lab1', 'lab2', 'lab3', 'lab4' and 'lab5'.
 4. Inside the 'lab1' folder, create three folders 'Ex1', 'Ex2' and 'Ex3'.

By these steps, you are required to be organized, and start every experiment in its independent folder. If any exercise depends on a previous one, copy the required file(s) from the first folder to the new one, and modify only the new copy. By the end of the term, you are required to have the files for all the exercises in every lab. TAs and lab technicians will not be able to help recovering any deleted file, or any file that you have overwritten by mistake.



Exercise 1: Build a 1-bit half adder

Build the following circuit, which represents a 1-bit half adder:



1- Bring an AND gate and an XOR Gate from the library of gates in Explorer pane to the Canvas (the drawing area). Change the 'Number of Inputs' of each gate to 2.

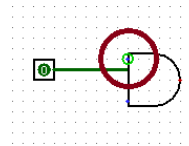
Common Mistake: Make sure you changed the 'Number of Inputs' of each gate to 2.

2- Bring 2 input square pins () from the toolbar (or press Ctrl-4) and one output circle pin () , again from the toolbar (or press Ctrl-5). You can use the pin from the wiring library in the Explorer pane, however make sure to change the following attributes: the input pin has 'no' in 'Three-state?', and the output pin has 'Yes' 'Output?', and Facing 'West'.


Common Mistake: If you are using "pin" from the "wiring library" make sure to change the attributes.

3- Connect the 2 input pins to the inputs of the XOR and AND gates, and connect the output pins to their outputs as shown in the figure above.

Common Mistake: Make sure you know where the inputs of the gate are. Hover over the input side of the gate and watch for the green circle. You may be connecting the input pin to somewhere near the gate inputs, not the actual inputs as shown in the figure below, and your circuit won't simulate properly. Do the same check in the output.



4- Now, we can simulate the circuit and log the results in a file. In order to enable logging, from the menu, click on 'Simulate', then 'Logging'. In the Selection tab, add the inputs, the sum then the carry. Each added signal will have a '-2' following it, to represent the Radix. The default is Radix-2, but you can change it. In the file tab, click select, navigate to the folder \sysc2310lab\lab1\Ex1, then write 'Lab1_Ex1_log.txt' in the selection. Keep the logging window active on the Table tab, next to the main window.

5- Use the poke tool () to change the values of the input bits. Notice the sum and carry outputs.

Common Mistake: In Logisim, there are two modes, editing and simulation. In editing mode, you should have the cursor selected. Simulation is running in this mode, but the inputs are restricted to zeros. In the simulation mode, you should have the poke tool selected. In this mode, you can change the input values, but you can no longer edit the circuit. Before updating the circuit, you have to return to the editing mode by selecting the cursor.



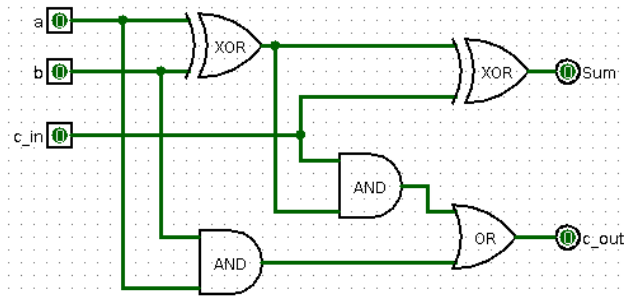
6- Save the circuit as 'Ex1.circ'.

A half adder is a full functioning adder that generates the sum and the carry, but it does not accept an input carry. I.e., It can be used to add the LSB of two binary numbers where there is no carry_in. Next, we will build a full adder, which accepts two inputs and a carry-in.

Exercise 2: Build a 1-bit full adder

1- Copy the circuit 'Ex1.circ' from 'Ex1' folder to the 'Ex2' folder. Rename the file to 'Ex2.circ'. From Logisim, File, Open, 'Ex2.circ'

2- Modify the circuit to be as follows, which represents a 1-bit full adder:



3- Enable logging to a file name 'Lab1_Ex2_log.txt'

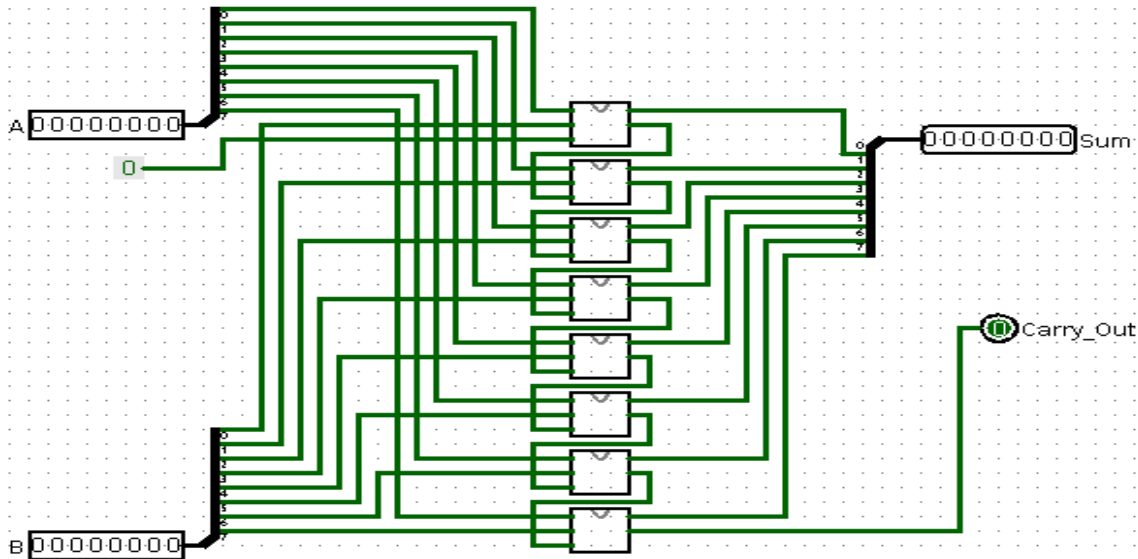
4- Simulate the circuit by using the poke tool, to test all the possible inputs.

5- Save the circuit as 'Ex2.circ'.

Exercise 3: Build an 8-bit full adder by using 8 1-bit full adders.

1- Again, copy the 'Ex2.circ' from the 'Ex2' folder to the 'Ex3' folder. Rename it to 'Ex3.circ'.

Note that, we need to use the previous 1-bit full adder as a module inside a bigger circuit. Here, we will try to build the following circuit:



2- From the Explorer Pane, select the main circuit to show its attributes. Change the 'Circuit Name' to '1b_full_adder'.

3- From the Menu bar, click 'Project', 'Add Circuit', and name it 'main'. Now, we have an empty 'main' circuit and a module that is called '1b_full_adder'.

4- Add 8 copies of the module '1b_full_adder'.

5- Add 2 input Pins (Ctrl-4), and change their 'Data Bits' attribute to 8 bits, name them A and B.

6- Add a 'Constant' from the wiring library. Change its value to 0x0. This should work as a carry in to the LSB.

7- Also, add 1 output Pin (Ctrl-5), and change its 'Data Bits' attribute to 8 bits, name is Sum.

8- Add 1 output Pin (Ctrl-5), and name is 'Carry_Out'.

9- Add 2 Splitters (Input) from the wiring library. Change its attributes as follows: 'Bit Width in': '8' and Fan Out to '8'.

- 10- Add 1 Splitters (output) from the wiring library. Change its attributes as follows: 'Bit Width in': '8' and Fan Out to '8', and facing 'West'.
- 11- Connect all the wires. Essentially, we connect bit 0 from input A (a0) and bit 0 from input B (b0) the inputs of the first 1-bit full adder module (the top one). a1 and b1 to the second module, and so on. The carry-in of the first module is the constant 0. The carry_in for all the other modules is the carry_out of the previous module. Carry_out of the last module is the overall carry out. Connect also the output sum. Make sure that all the wires are dark green and there is no green dots (🚫), which indicates two wires are cross-connected to each other.
- 12- Enable logging to a file name 'Lab1_Ex3_log.txt', and add 'A', 'B', 'Sum' and the 'Carry_out' change the radix of all of them to 10, to make it easy for the TA to review your work.
- 13- Simulate the circuit by using the poke tool, to test some of the possible inputs.
- 14- Save the circuit as 'Ex3.circ'.

Congratulations. You have just designed a fully functioning ADD instruction that could be used in a simple 8-bit CPU.

Demonstration:

- In order to get full credit, complete all exercises and be prepared to answer questions asked by the TA during the one-on-one meeting.
- Before the end of the scheduled lab time, even if you could not complete the lab, make sure you attend the one-on-one meeting with the TA.
- If you could not complete the lab during the scheduled lab time, finish up the lab on your own time. In this case, the demonstration points will be reduced in accordance to the percentage of work completed. Make sure you take permission from your TA.
- If a student did not show up in the one-on-one meeting, 0 points are attributed for this lab.