# CS2110 Spring 2014
# Homework 1

**This assignment is due by:**

Day:
Time: 11:54:59pm

# Objectives

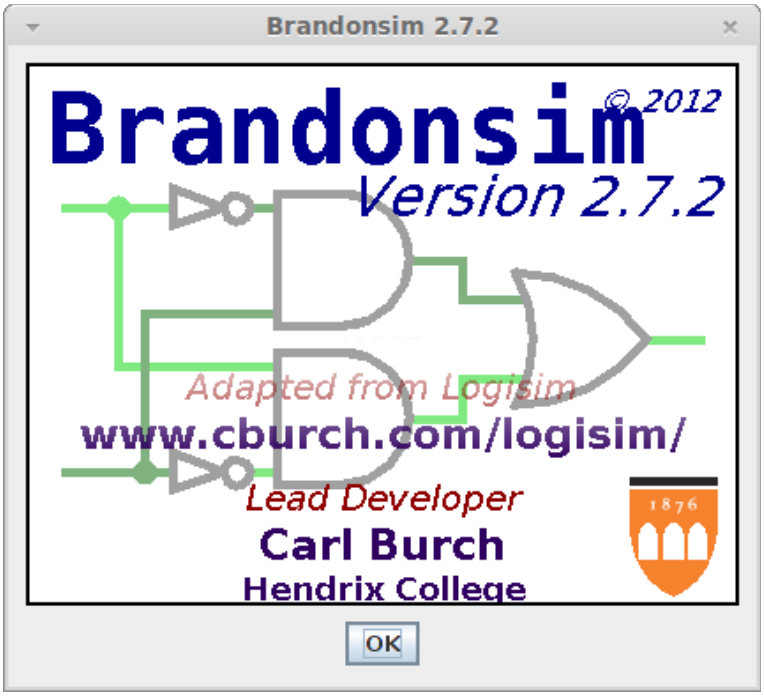
1. To get a feel for Logisim and learn how to do the most basic of tasks.
2. To learn how the logic gates work.
3. To learn the basic I/O components of Logisim.

# Overview

Logisim is an interactive circuit simulation package. We will be using this program for the next couple of homework assignments. Before you start please ensure that you have at least Java 5 (JDK) installed on your computer. This should not be a problem if you are coming from classes such as CS1331/32 which require that you code in Java. Logisim will only run on machines with at least Java 5 installed. Your next homework will involve programming in Java so make sure you have the JDK installed.

This semester we will be using a newer version of logisim that I have wrote. It is attached to this assignment. If for some reason you are retaking this course or have dropped you are now not allowed to use the version of logisim you got that semester. **You are required to update to my version of Logisim.** The TAs will be grading your assignments using this version so you risk major point deductions if we find problems with your circuit on our computer. If you submit the assignment using a previous version of logisim you risk getting major points off (if not a zero). In addition we will be checking to see if you have the correct version of Logisim so make sure you are using it!.

When opening Logisim you should get this screen.




Logisim is a powerful simulation tool designed for educational use. This gives it the advantage of being a little more forgiving than some of the more commercial simulators. However, it still requires some time and effort to be able to use the program efficiently. With this in mind, we present you with the following assignment:

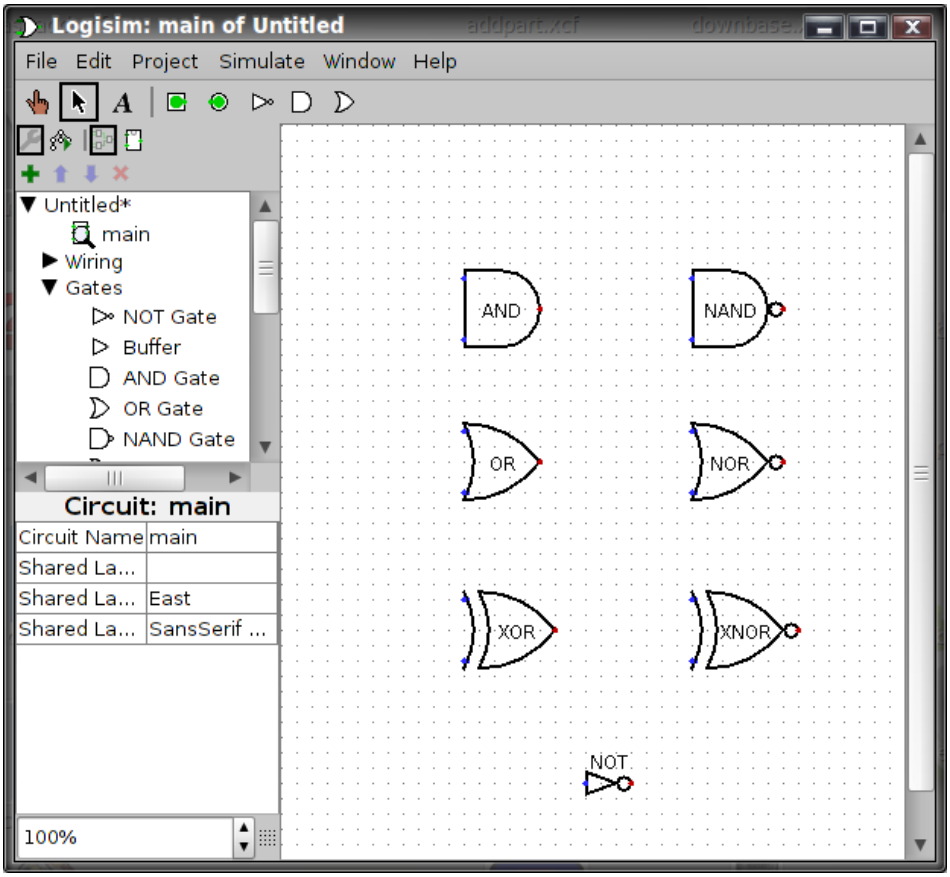
# Part 1

1. Please follow the Beginners Tutorial at http://ozark.hendrix.edu/~burch/logisim/docs/2.6.0/en/guide/tutorial/index.html *Note: yes I am well aware that it says 2.6.0 in the URL it is still applicable for the version of logisim distributed with the assignment in addition the url for the current version does not exist*
2. Save the circuit you just created as xor.circ
3. Read sections “Libraries and Attributes” and “Wire Bundles”

# Part 2

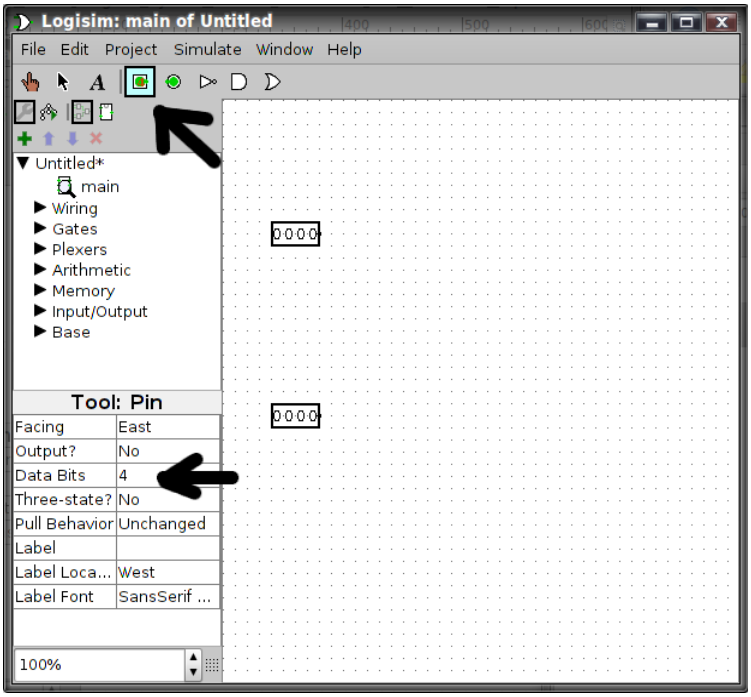
## *(Note: Screenshots may not match up with your version, but the steps are EXACTLY the same. The screenshots are only a guide)*

Please follow along below I will cover some of the most common components you will be using. Here is a list of all of the basic gates available to you. Notice how the gates on the right side have a bubble where the output is. As you will learn in class these gates are just the complement of the gates on the left side. You can use these gates to make more complex circuits, for example, in Part 1 you saw that a XOR gate can just be made from AND's OR's and NOT's.

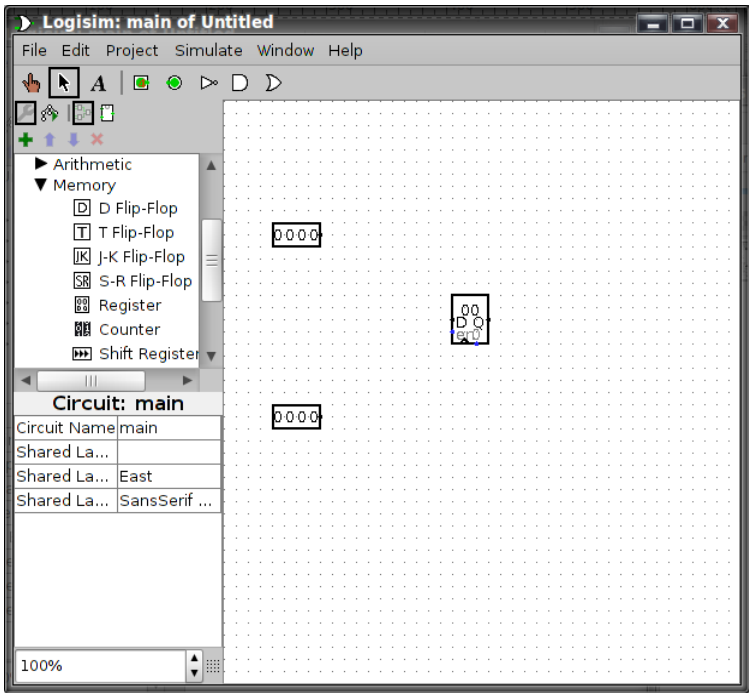


1. First create a new file and save it as part2.circ

2. We will start by adding in two input pins to our circuit. The input pin will allow us to specify that a value is coming in from somewhere. We also want this input pin to be 4 bits so we modify the “Data Bits” attribute of the input pin. Also make sure that the Output? Attribute is No. The image below shows you where the input pin is (Alternatively you can look under the wiring library for pin) and it also shows the Data Bits attribute.

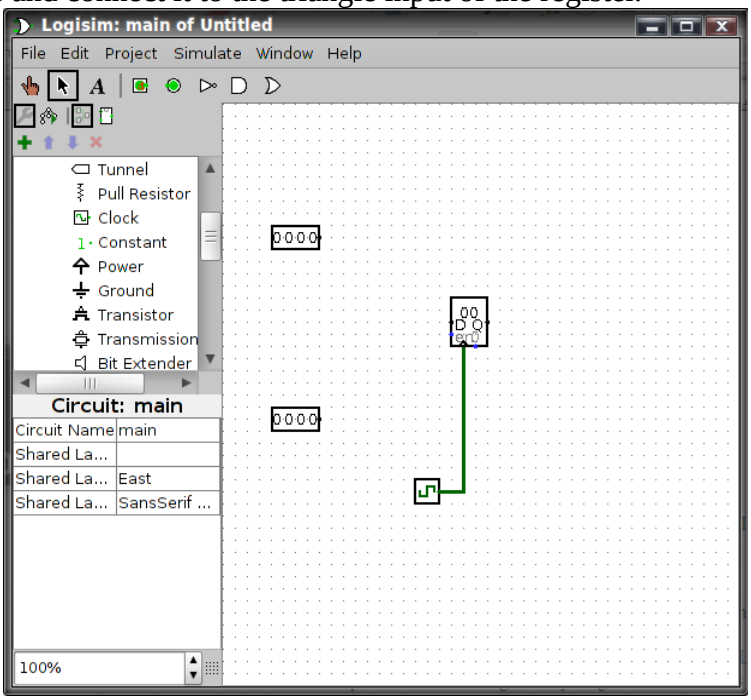


3. Next we are going to add a register to our circuit. Don't worry about the details on what it does now but a register just stores a value. The register can be found in the “Memory” library leave all settings alone (The Data Bits attribute should be 8).

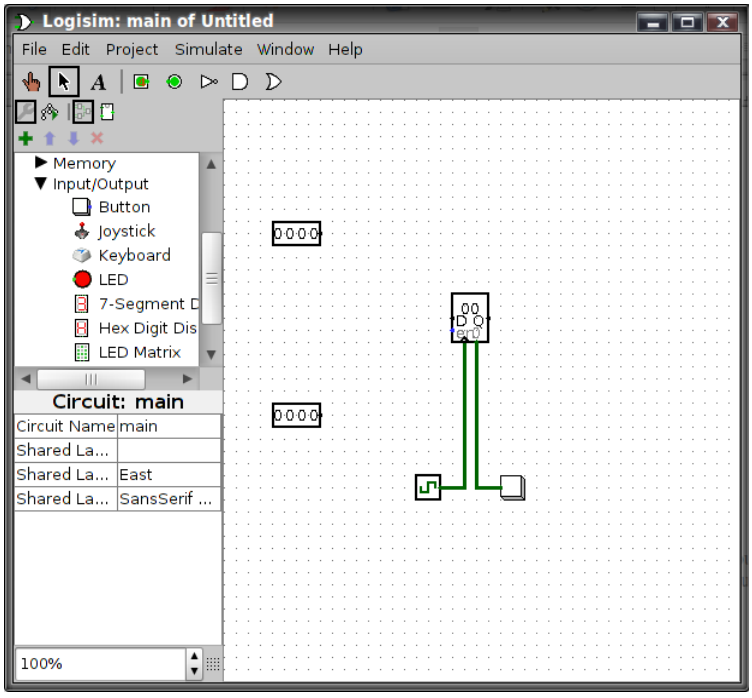



4. As you can see the register has many inputs and a single output on the right hand side. If you want to know what each port on a component does just mouse over it or refer to the documentation included within logisim.

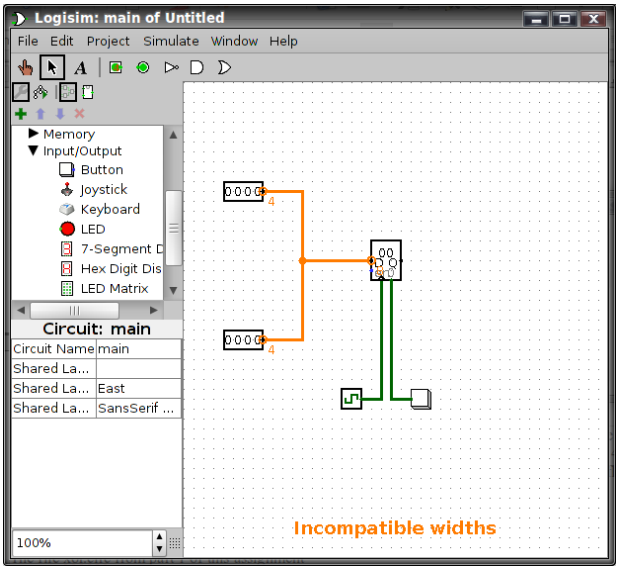
5. Notice the input on the bottom of the register that has a funny looking triangle shape. Whenever you see this on a port it means to connect a clock to it. The clock is a component used to synchronize other components. The output is toggled on regular intervals based on the simulation speed. Lets add a clock you may find it in the wiring library. Again leave all of its attributes alone and connect it to the triangle input of the register.

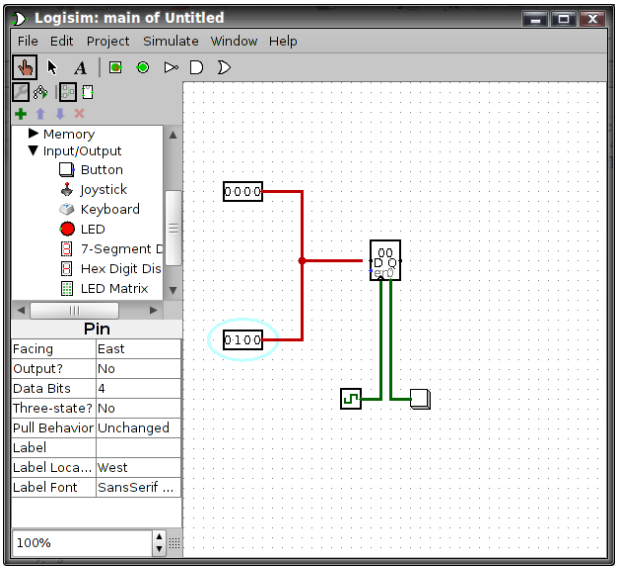


6. Next lets connect a button to the “0” input of the register. Like the input pin when you press the button the value on its output is 1. However, unlike the input pin when you release the button the value on its output goes back to 0 again. You may find the button in the Input/Output library.




7. Lastly we will connect our inputs to our register however we do not directly connect the input pins to the “D” input to the register. For 2 reasons 1) The bit depth of the input pins are 4 while the bit depth of the register's D input is 8 this does not match. 2) We could cause a conflict if we connect two inputs together. The image below shows you what not to do in logisim

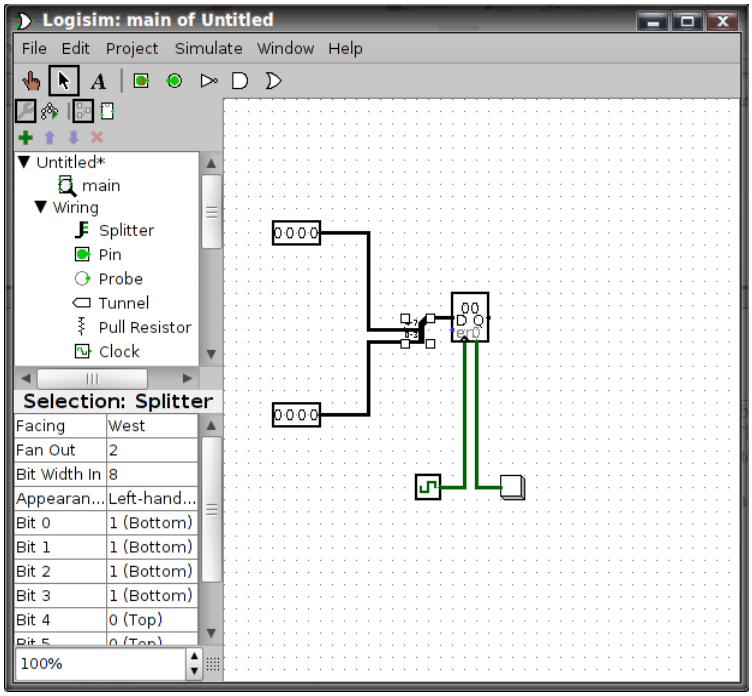


8. As you can see with the image on the left Logisim gives you an error message Incompatible widths, and on the conflicting wires it will tell you the number of bits the wire is composed of. To fix this you need to ensure that whatever is connected is the same number of bits or else Logisim will not do anything. The image on the right hand side is a more serious mistake and Logisim will indicate this by coloring the wire red. The right hand side illustrates a conflict or a short circuit – you are telling the wire to be two different values at once.

9. To fix this problem we do not connect the input pins directly to each other nor do we connect them into the input of the register. We will first combine the two 4 bit outputs from both input pins to create an 8 bit wire suitable for the "D" input to the register. We will do this by using a splitter; a splitter either allows us to split a multi-bit wire into several wires or it can join wires together to form a wire of greater bit width. Here, we are joining the two 4 bit wires into an 8 bit wire.

10. You may find the splitter in the Wiring library. Next we will need to configure the attributes to get the splitter to do what we want. In this case first we set the Fan out attribute to 2. The fan out is 2 because we want to combine two wires. Next we set the Bit Width In attribute to 8 since when we combine the 2 4 bit wires we will get an 8 bit wire. Lastly we want Bits 7-4 to correspond to the top input pin and 3-0 to correspond to the bottom one so we change the Bit X attributes to correspond to this.

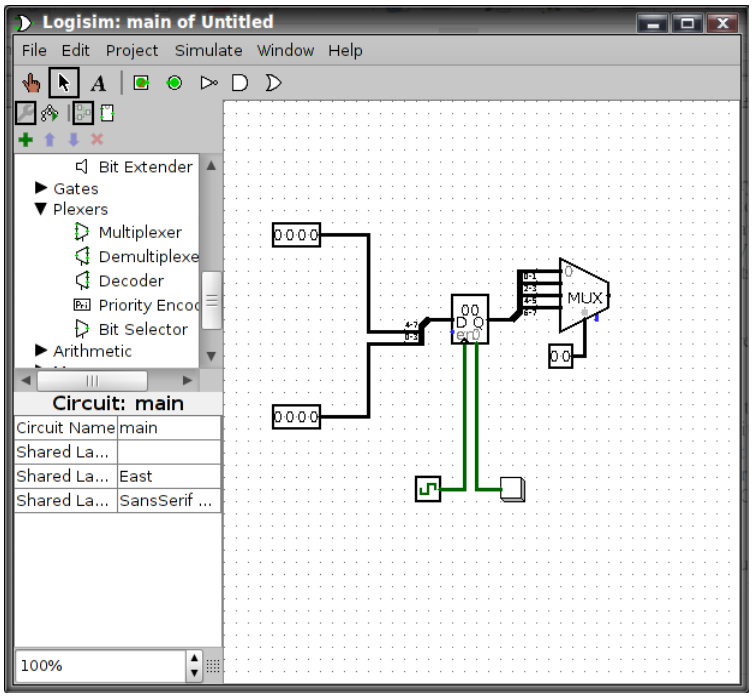



11. Notice how we didn't get a Incompatible widths error message from Logisim when everything was connected. This is because with the splitter logisim already knows what bit width to expect from the left hand side. Had we set the fan out for the splitter to say 3 logisim would have created 3 wires in which the first and second expect 3 bits and the last expects 2 bits. When the fan out does not evenly divide into the bit width In Logisim will try to distribute the bits as evenly as possible.

12. Next add to the output of the register, that is, the “Q” pin a splitter that will split the wire into 4 2 bit wires.

13. Now I will introduce another component, a multiplexer. Again you don't need to know the details of how these work as of now but think of it as the equivalent in circuitry as a switch/case statement in coding. You may find the multiplexer in the Plexers library. For the select bits select 2. Since we have 4 things we must have 2 bits in order to select among them. For data bits we will also select 2 as each input is a 2 bit wire. Leave the other attributes alone. Connect each individual output from the splitter into the corresponding input to the multiplexer as shown below.

14. Next we will add another 2 bit input pin to act as our selector you will connect this input pin to the “Select” pin of the multiplexer (The leftmost pin again you may mouse over pins to figure out what they do).




15. Lastly to prevent massive clutter in your circuit and to get on your grading TA's good side (HINTHINT) you should use what is called a tunnel. A Tunnel allows you to not have wires going everywhere in your circuit and making it hard for others to view. Two tunnels with the same label are considered to be connected. So lets now make use of a tunnel.

16. The tunnel may be found in the Wiring library. Be sure to set this tunnels data bits attribute to 2 and its Label to some name of your choosing. Connect the output from the multiplexer to the input of the Tunnel. Lastly create a tunnel and place it somewhere else in the circuit and connect it to an output pin. Note that you can also select the tunnel from the Project Toolbar or you can automatically create a tunnel anywhere by left clicking while holding down the CTRL key. You are done with building this circuit.

17. Now on to testing the circuit. You may use the poke tool and poke the 2 four bit input pins and the 2 bit select input pin from the multiplexer and notice the change on the output and register. However if you do this right now you will not notice any changes. This is because we have not enabled the simulation yet. The registers value only changes the instant the clock goes from 0 to 1 since simulation has not been enabled the clock just stays at 0.

18. To enable the circuit simulation we go under the Simulation menu and select Ticks Enabled. This will make the clock tick on and of at a regular interval. To make the clock speed up or slow down under the Simulation menu you can select Tick Frequency and select a new value (higher Hertz means the clock will tick faster).

19. When you are happy with your work save it as part2.circ this file will be included with your submission for this assignment.

20. Remember to put your name in your circuit. Make sure you have read the rules at the top of the assignment.

# Part 3

Create a new text file named part3.txt answer the following questions. To answer questions 1-2 be sure ticks are enabled. To answer question 3 make sure ticks are disabled.

1) Give the topmost 4 bit input pin a value of 1011, give the other 4 bit input pin a value of 0001, and give the 2 bit input pin a value of 10
    a) What value is displayed on the register?
    b) What value is showing on the output pin?
2) What happens when you press down the button that was created in step 6?
3) When ticks are disabled, what happens when you poke the register and then type 55 and press enter?
4) What situations will cause a red wire in logisim?
5) What situations will cause a blue wire in logisim?
6) (True or False) Can a splitter be used to join wires together?

# Evaluation

Your submission will be evaluated based on how well you followed the directions and your answers to the questions for part 3.  Please ensure you have done all of the steps correctly.

If you are caught using an incorrect version of logisim you will be heavily penalized.

# Deliverables

The file xor.circ from part 1 of this assignment
The file part2.circ from part 2 of this assignment
The file part3.txt from part 3 of this assignment

Please only submit the above files or an archive (zip and tar.gz only) of ONLY those files and not those files in a folder.

Submit your assignment by the deadline on T-Square.