CS 2110 Fall 2013 Homework 1

Author: Alexander Ikonomidis

This assignment is due by:

Day: Tuesday, August 27th
Time: 11:55:00pm

Please read the following sections. They apply to all homework assignments you will submit this semester. This will be at the top of each homework as a reminder.

Academic Misconduct

1. Academic misconduct is taken very seriously in this class. You may not copy code from your peers, someone who has already taken this course, or from the Internet. You may work with others who are enrolled in the course, but each student should be turning in their own version of the assignment. Be very careful when supplying your work to a classmate that promises to "just look at it". If they turn it in as their own you will both be penalized.
2. We will be using automated code analysis and comparison tools to enforce these rules. If you are caught you will receive a zero and be reported to Dean of Students.

Submission Guidelines

1. You are responsible for turning in assignments on time. The onus is on you to compensate for unforeseen circumstances. If you have an emergency, let us know in advance of the due time and supply appropriate documentation. Extensions will only be granted to those who contact us in **advance** of the deadline.
2. You are also responsible for ensuring that what you turned in is what you meant to turn in. After submitting you should download your submission into a new folder and test if it works. We will not accept excuses if you submit the wrong files: what you turn in is what we grade.
3. Your assignment must be turned in via T-Square. After submitting an assignment you should get an email confirming that it was successfully submitted. If you do not get this email that means that you did not complete the submission process correctly. We will not accept email submission of assignments. If granted an extension, you will still turn the assignment in over T-Square.

General Rules

1. Starting with the assembly homework, any code you write must be clearly, meaningfully commented.
2. Although you may ask TAs for clarification, you are ultimately responsible for what you submit.
3. Please read the assignment in its entirety before asking questions.
4. Please start assignments early, and ask for help early. Do not email us the night the assignment is due with questions.
5. If you find any problems with the assignment please report them to the author (which can be found at the top of the assignment). Announcements will be posted if the assignment changes.

Submission Conventions

1. Failure to follow these may result in a max of 5 points taken off
2. All files you submit for assignments in this course should have **your name at the top of the file as a comment** for any source code file, and somewhere in the file, near the top, for other files unless otherwise noted.
3. Please submit your deliverables as an archive (zip or tar.gz only please).
4. Do not submit compiled files (.class files for Java code or .o files for C code). Only submit the files we ask for in the assignment.
5. Do not submit links to files. We will not grade assignments submitted this way as it is easy to change the files after the submission period ends.

Objectives

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

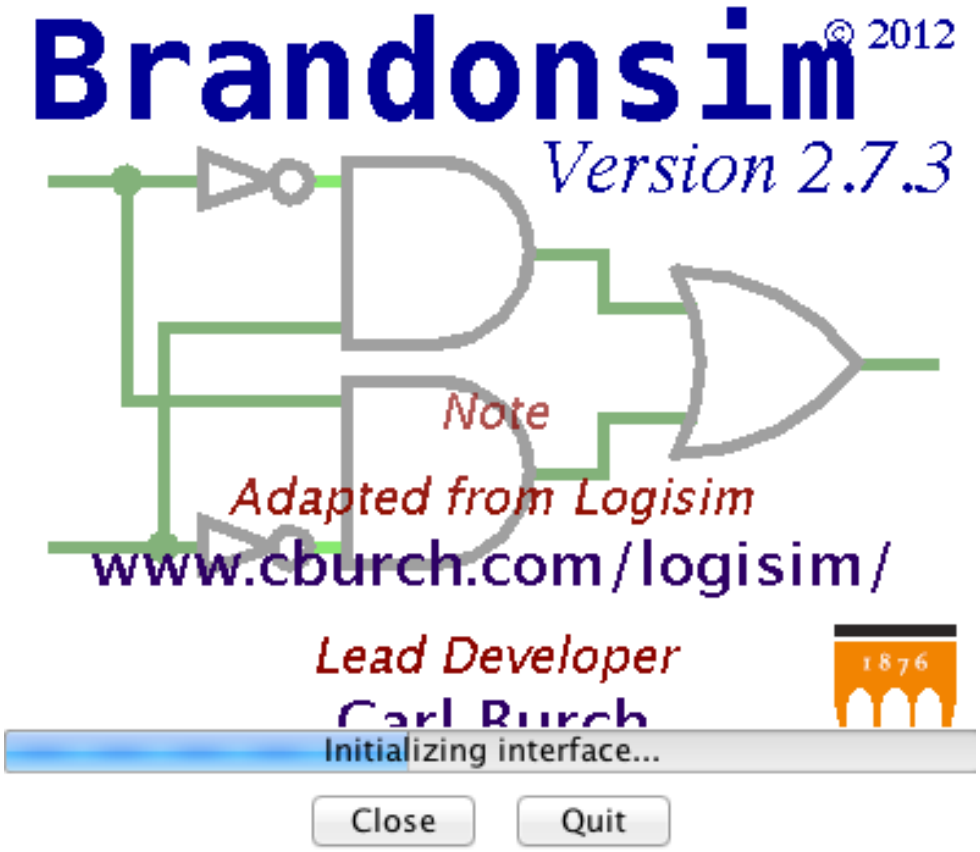
Overview

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 version of Logisim that a previous head TA wrote. It is attached to this assignment. If for some reason you are retaking this course or have dropped you are not allowed to use the version of Logisim you got that semester. You

are required to update to the new 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 losing some of your points, as functionality in older versions may not be the same as the version we will use to grade.

When opening Logisim you should get the screen below:



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. Despite that, 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.

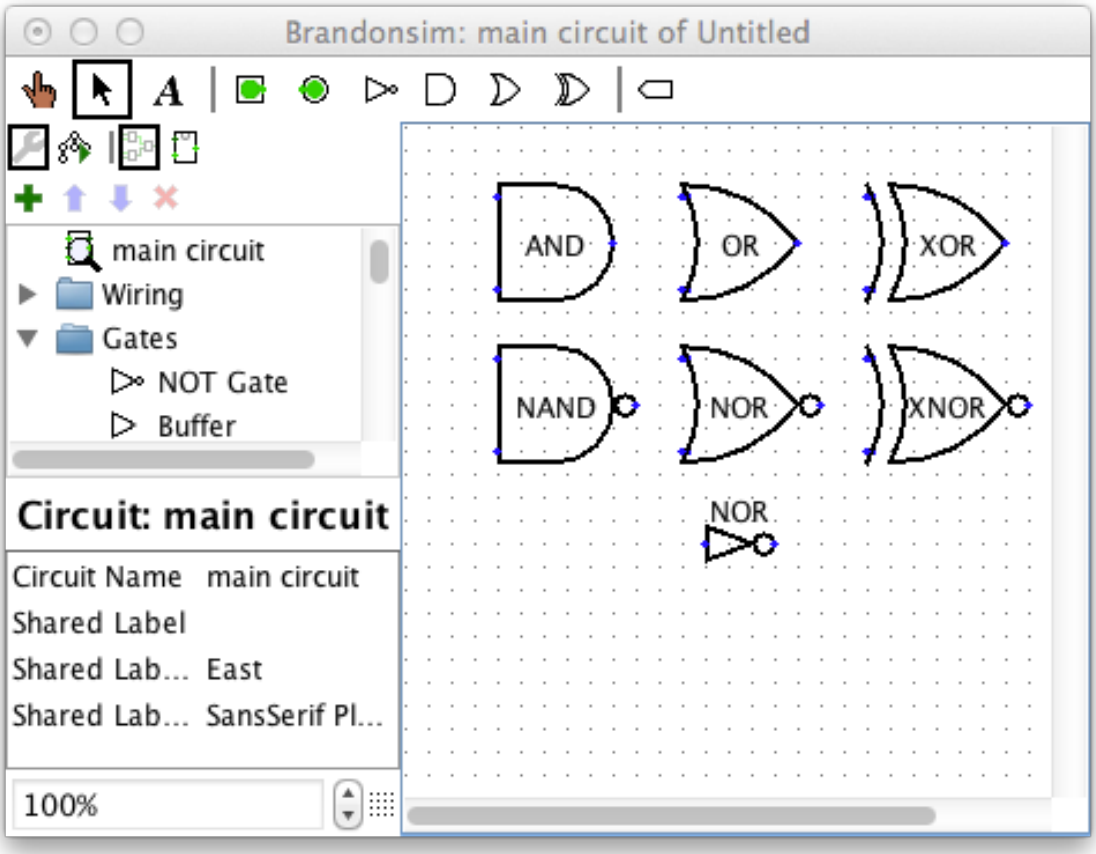
Part 1

1. Please follow the Beginners Tutorial at http://ozark.hendrix.edu/~burch/logisim/docs/2.6.0/en/guide/tutorial/index.html Note: Although it says 2.6.0 in the URL, the tutorial is still applicable for the version of Logisim distributed with the assignment.
2. Save the circuit you just created as xor.circ
3. Read sections "Libraries and Attributes" and "Wire Bundles."

Part 2

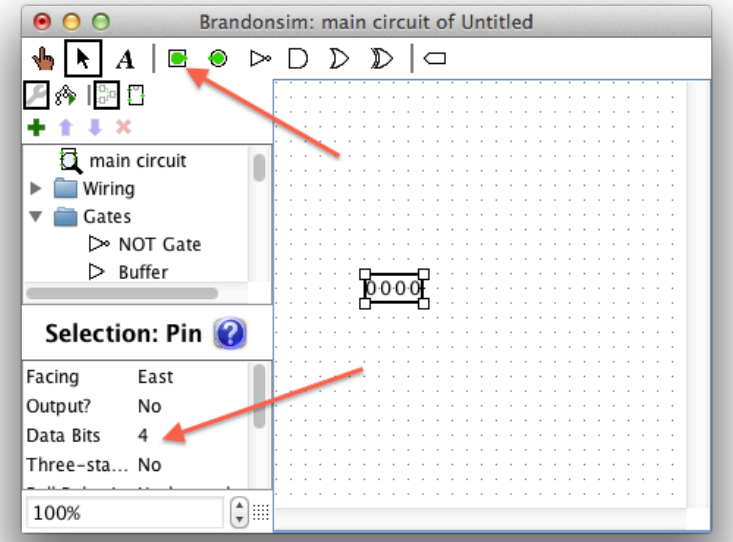
(Note: Screenshots may not match up with your version, but the steps are EXACTLY the same. The screenshots are only a guide)

Here is a list of some of the basic gates available to you. Notice how the gates on the bottom have a bubble where the output is. As you will learn in class, these gates are just the complement of the gates on top. 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, OR, and NOT gates.

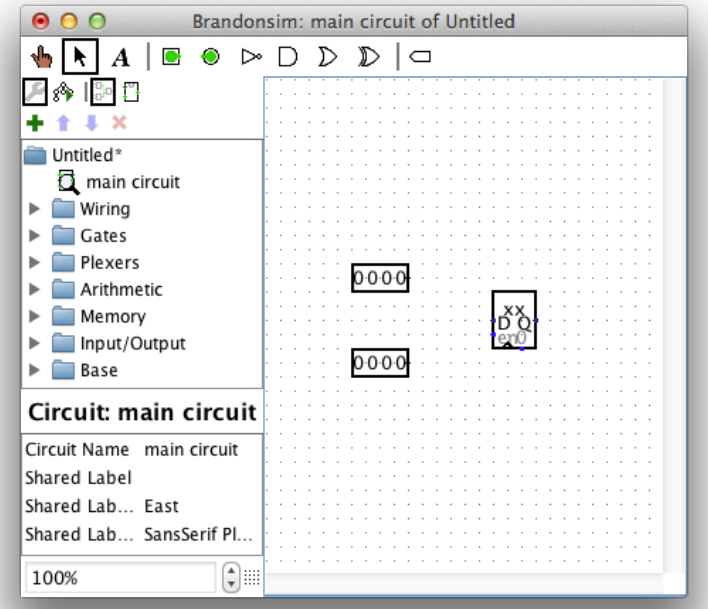


1. Create a new file and save it as part2.circ

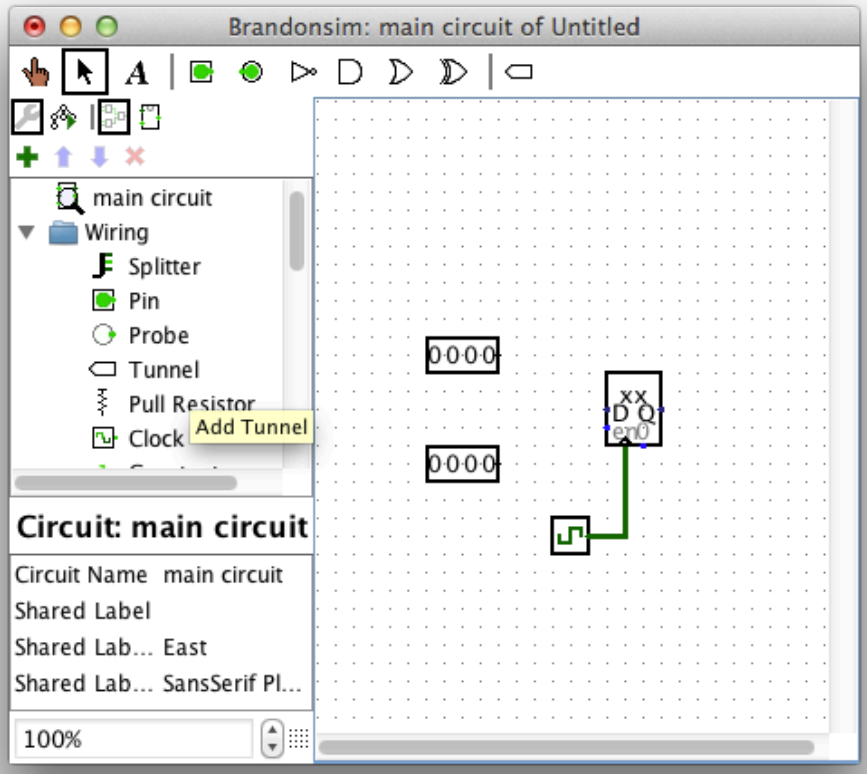
2. We will start by adding in two input pins to our circuit. The input pin will allow us to specify that a value is coming in from somewhere. We also want this input pin to be 4 bits, so modify the "Data Bits" attribute of the input pin. Set the Output? attribute to 'No'. The image below shows you where to find the input pin and the Data Bits attribute.



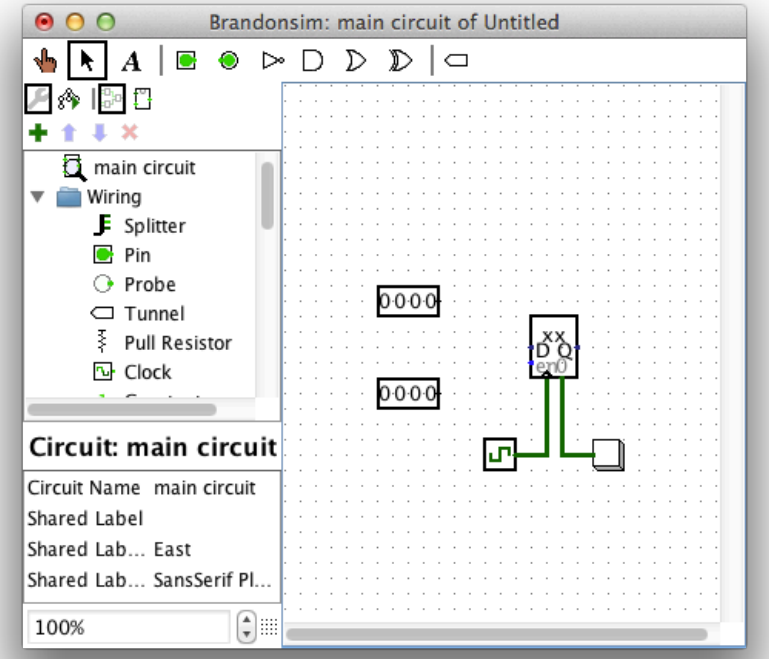
3. Next we are going to add a register to our circuit. You'll learn about registers later, but for now just know that registers store a value. The register can be found in the "Memory" library. Use its default settings, but make sure the Data Bits attribute is set to 8.



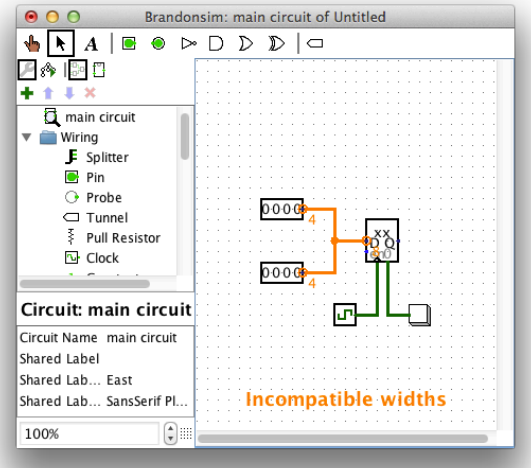
4. As you can see the register has many inputs and a single output on the right hand side. If you want to know what each port on a component does just mouse over it or refer to the documentation included within Logisim.
5. Notice the input on the bottom of the register that has a funny looking triangle shape. Whenever you see this on a port it means to connect a clock to it. The clock is a component used to synchronize components. A clock's output is toggled on regular intervals based on the simulation speed. Let's add a clock! You can find it in the wiring library. Use the default attributes and connect it to the triangle input of the register.

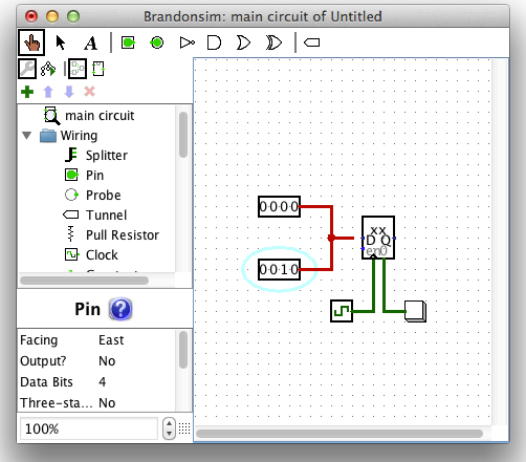


6. Next let's connect a button to the "0" input of the register. Like the input pin, when you press the button the value on its output is 1. However, unlike the input pin, when you release the button the value on its output goes back to 0. You can find the button in the Input/Output library.



7. Lastly we will connect our inputs to our register. You cannot directly connect the input pins to the "D" input to the register for 2 reasons. The bit depth of the input pins is 4 while the bit depth of the register's D input is 8: this does not match. Also, connecting two inputs together causes a conflict.The images below show you what not to do in Logisim.

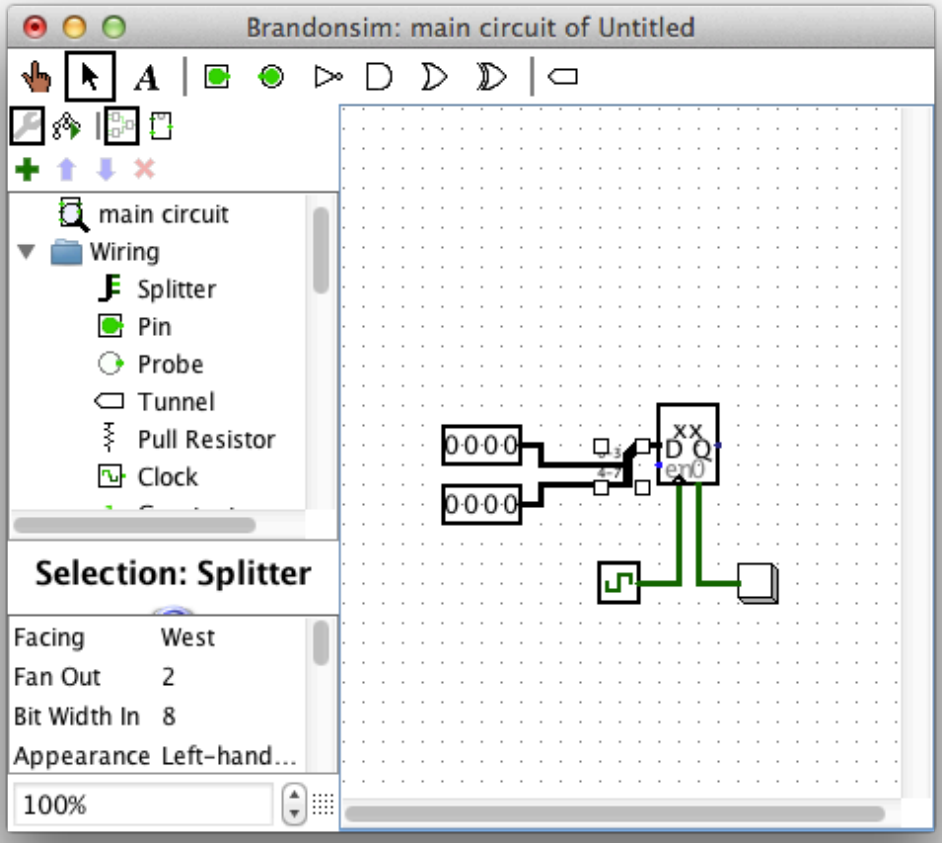




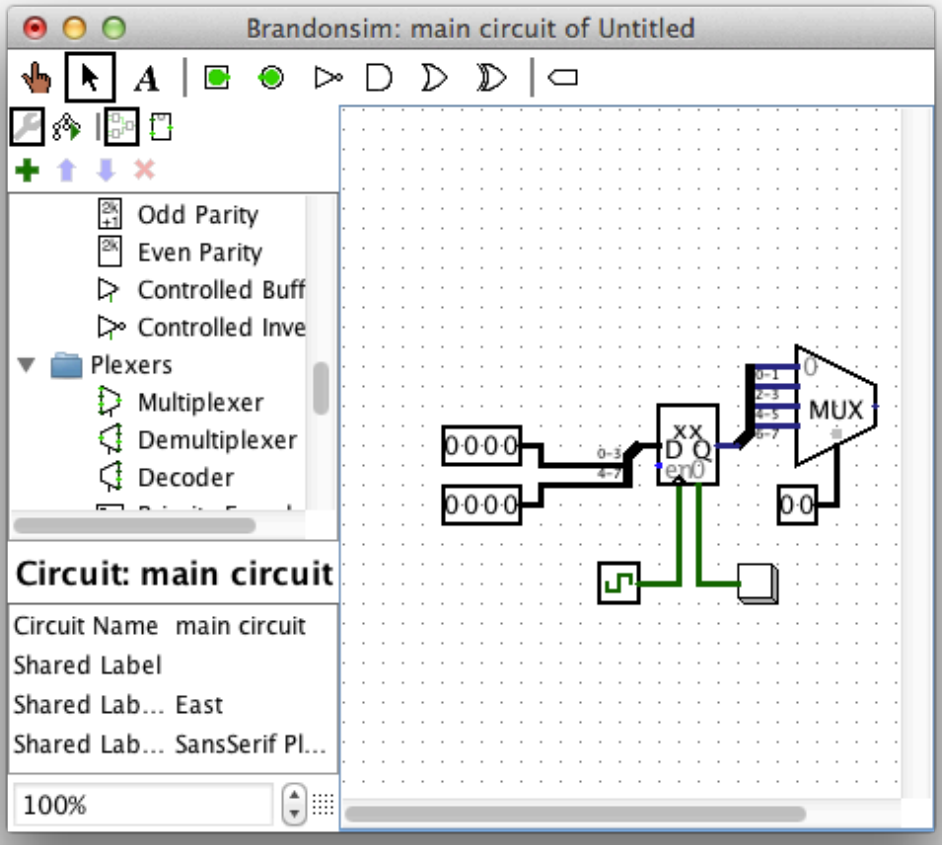
8. As you can see with the image on the left Logisim gives you an error message Incompatible widths, and on the conflicting wires it will tell you the number of

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 can 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 4-7 to correspond to the bottom input pin and 0-3 to correspond to the top one so we change the Bit X attributes to correspond to this.

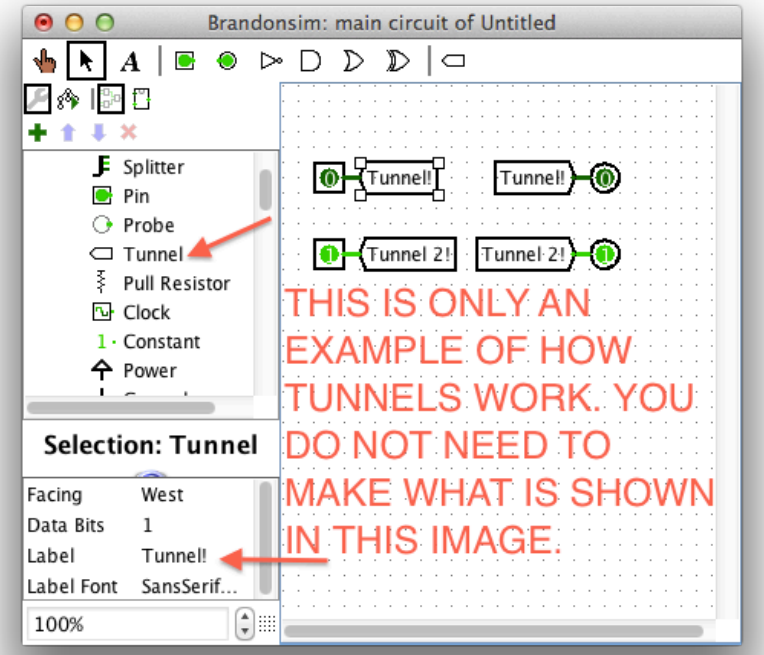


11. Notice how we didn't get a Incompatible widths error message from Logisim when everything was connected. This is because with the splitter Logisim already knows what bit width to expect from the left hand side. Had we set the fan out for the splitter to 3, Logisim would have created 3 wires. When the fan out attribute does not evenly divide into the bit width, Logisim will try to distribute the bits as evenly as possible.
12. Next add a splitter to the output of the register (the "Q" pin) 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 for 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. Connect this input pin to the "Select" pin of the multiplexer.

15. Lastly to prevent massive clutter in your circuit and to get on your grading TA's good side (HINT HINT) 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. An example of how tunnels work is shown below.



17. Now on to test your 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.

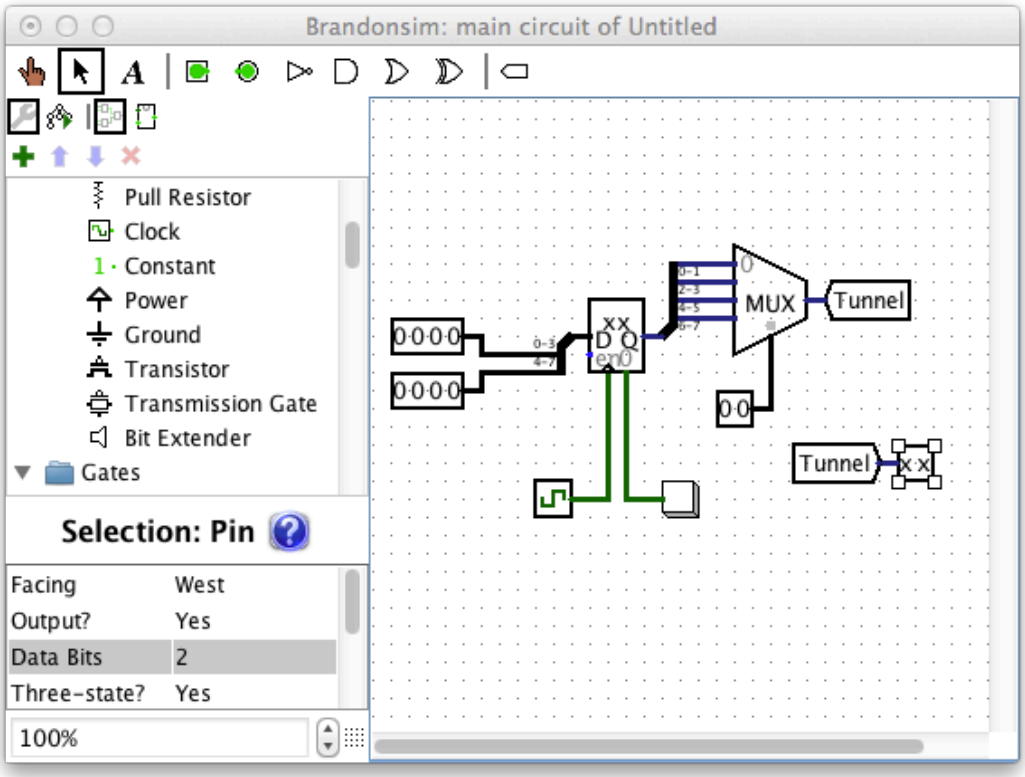
Part 3

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.



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.