CS2110 Fall 2015
Homework 1

Author: Marissa Wall

This assignment is due by:

Day: Check T-square for due date
Time: 11:54:59pm

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

Rules and Regulations

Submission Guidelines

1. You are responsible for turning in assignments on time. This includes allowing for unforeseen circumstances. If you have an emergency let us know ***IN ADVANCE*** of the due time supplying documentation (i.e. note from the dean, doctor's note, etc). Extensions will only be granted to those who contact us in advance of the deadline and no extensions will be made after the due date.

2. You are also responsible for ensuring that what you turned in is what you meant to turn in. After submitting you should be sure to download your submission into a brand new folder and test if it works. No excuses if you submit the wrong files, what you turn in is what we grade. In addition, your assignment must be turned in via T-Square. When you submit the assignment you should get an email from T-Square telling you that you submitted the assignment. If you do not get this email that means that you did not complete the submission process correctly. Under no circumstances whatsoever we will accept any email submission of an assignment. Note: if you were granted an extension you will still turn in the assignment over T-Square.

3. There is a 6-hour grace period added to all assignments. You may submit your assignment without penalty up until 11:55PM, or with 25% penalty up until 5:55AM. *So what you should take from this is not to start assignments on the last day and plan to submit right at 11:54AM.* You alone are responsible for submitting your homework before the grace period begins or ends; neither T-Square, nor your flaky internet are to blame if you are unable to submit because you banked on your computer working up until 11:54PM. The penalty for submitting during the grace period (25%) or after (no credit) is non-negotiable.

General Rules

1. Starting with the assembly homeworks, Any code you write (if any) must be clearly commented and the comments must be meaningful. You should comment your code in terms of the algorithm you are implementing we all know what the line of code does.
2. Although you may ask TAs for clarification, you are ultimately responsible for what you submit. This means that (in the case of demos) you should come prepared to explain to the TA how any piece of code you submitted works, even if you copied it from the book or read about it on the internet.
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 it would be greatly appreciated if you reported 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. 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.
2. When preparing your submission you may either submit the files individually to T-Square or you may submit an archive (zip or tar.gz only please) of the files (preferred). You can create an archive by right clicking on files and selecting the appropriate compress option on your system.
3. If you choose to submit an archive please don't zip up a folder with the files, only submit an archive of the files we want. (See **Deliverables**).
4. Do not submit compiled files that is .class files for Java code and .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

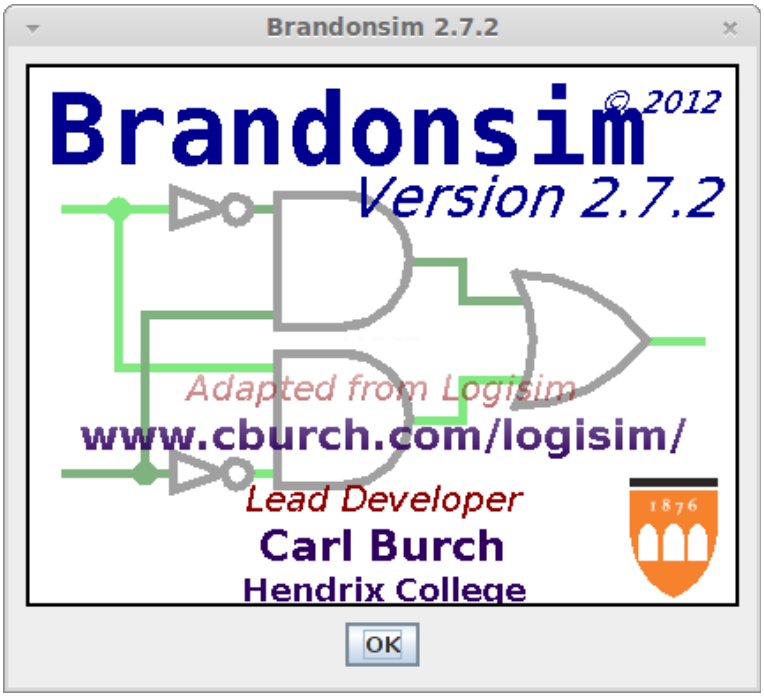
1. To get a feel for Logisim and learn how to do the most basic of tasks.
2. To learn how the logic gates work.
3. To learn the basic I/O components of Logisim.

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 CS1331/1332 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.

Please note that the version of Logisim distributed with the assignment is different from the version hosted on Logisim's main website; we use a bugfix version by Brandon Whitehead. 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://www.cburch.com/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*

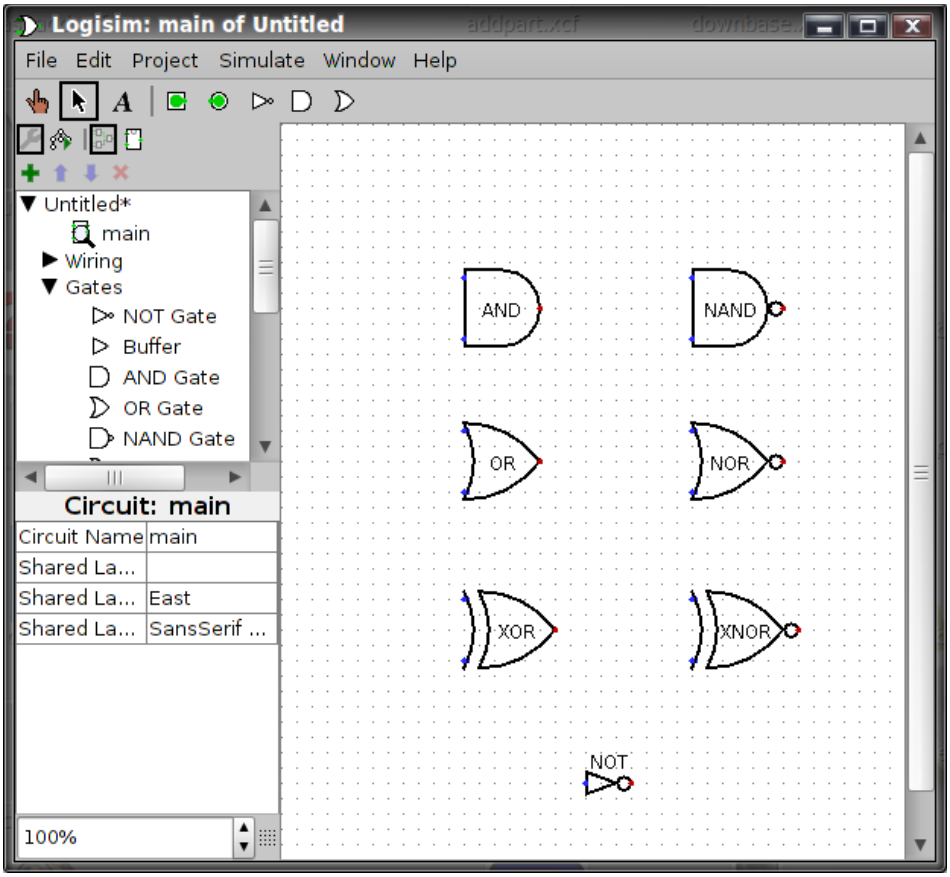
2. Save the circuit you just created as xor.circ

3. Read sections “Libraries and Attributes” and “Wire Bundles”

Part 2

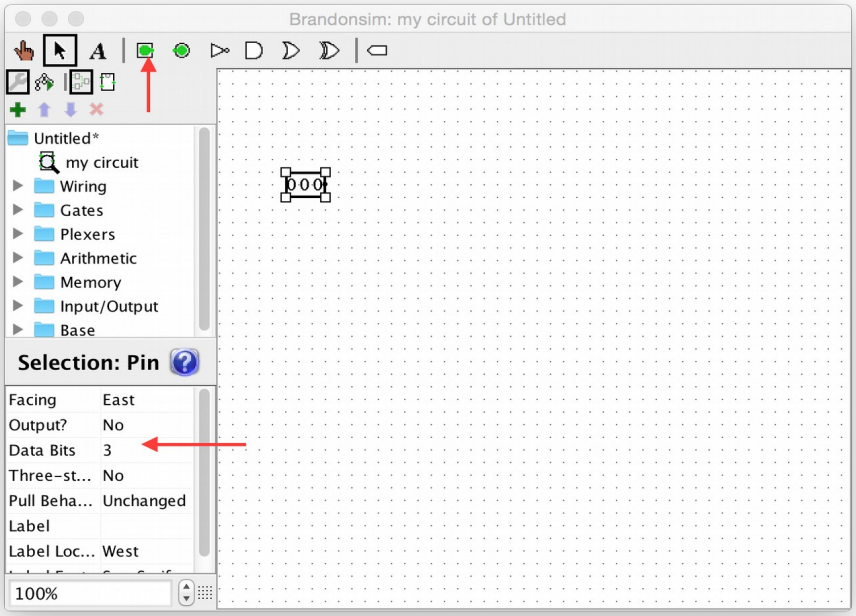
(Note: Screenshots may not match up with your version, but the steps are EXACTLY the same. The screenshots are only a guide)

Please follow along below; it covers 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.

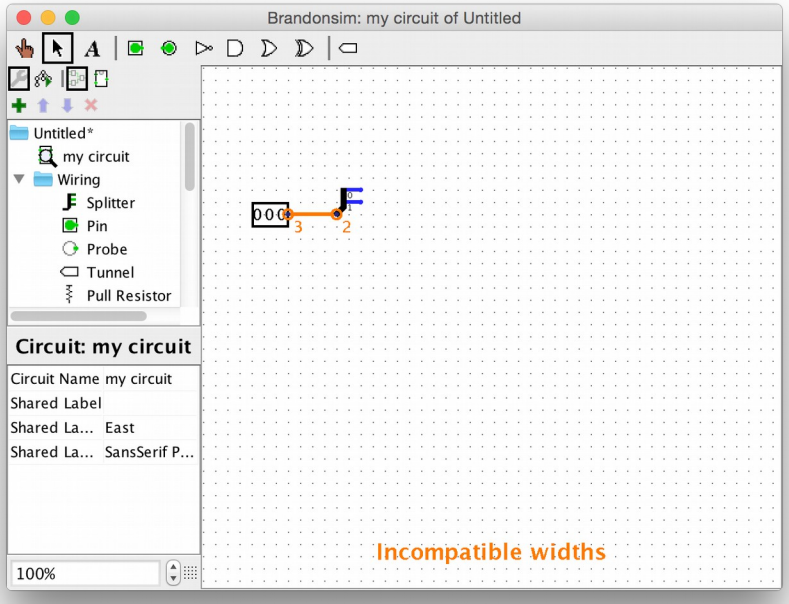


1. First create a new file and save it as part2.circ. In this part, we will create a circuit that will tell you whether you should bring an umbrella with you when you go out. To determine this we'll follow the following formula: you should bring an umbrella with you if it's cloudy and it's windy or if the weatherman tells you to bring an umbrella.

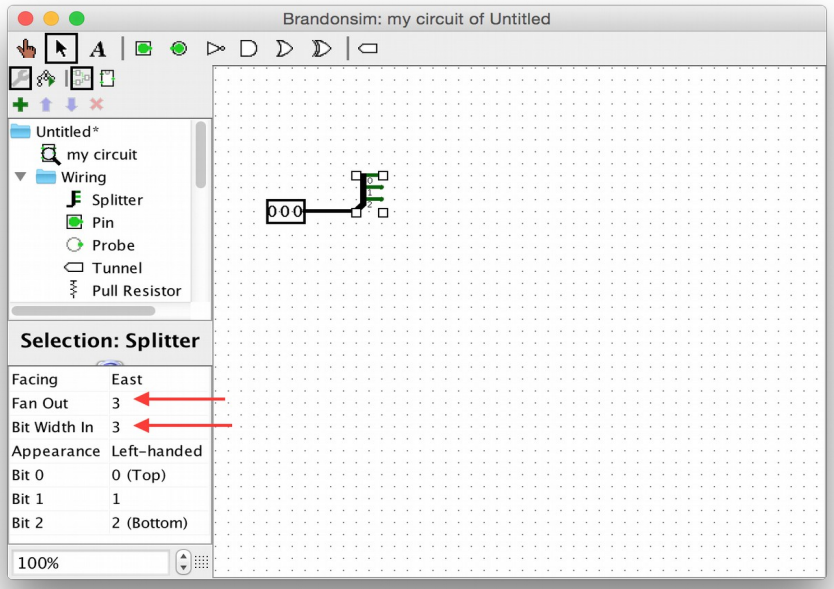
2. This circuit will have a single 3-bit input that will handle all of the conditions. Add an input to the circuit and change it to be three bits:



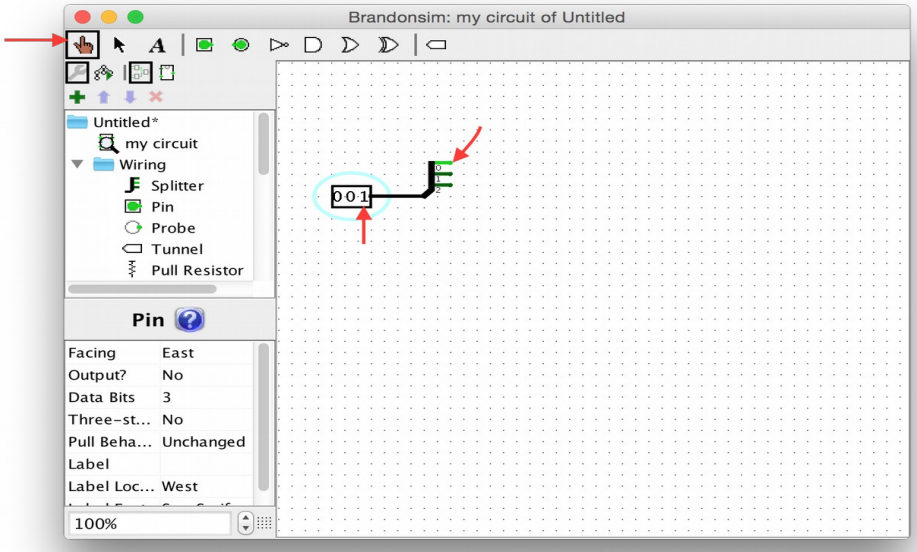
3. Now that we have the input placed, we need to access each of the individual bits. The 3-bit input we have placed combines three basic 1-bit inputs into one circuit element. However, it is often useful to access each of the bits from the input individually. To do this, we must use a splitter which can be found under the wiring folder. Go ahead and add one to the circuit and connect it to the input.



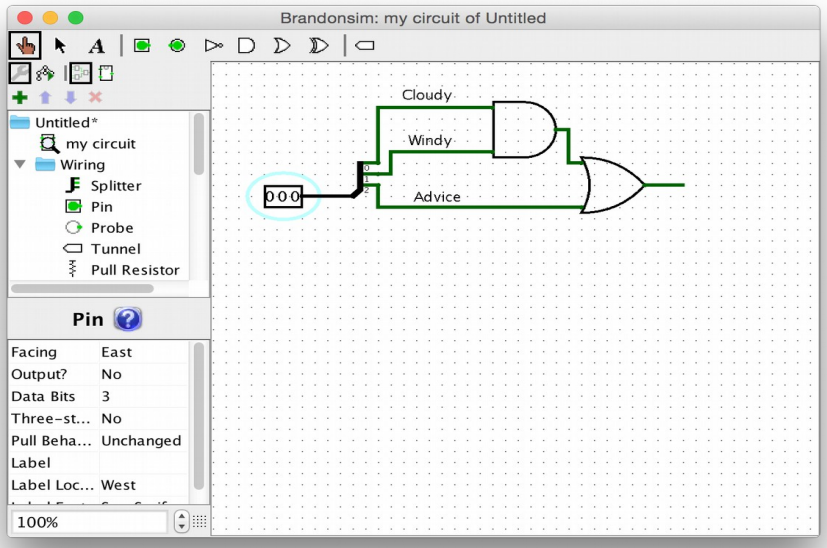
4. You will notice off the bat that the splitter is not set up properly. Our splitter should take in a single 3 bit wire and split it into three 1-bit wires. However, by default it takes in a single 2-bit wire and splits it into two 1-bit wires. We can fix this by changing the Bit Width In and Fan Out properties:



5. You can test the splitter to see how it works by changing the input with the poking tool (the one that looks like a finger). After choosing the poking tool, simply click on the bits in the input to change their value. Notice how the wires coming from the splitter change their colors based on the values you set. It is also worth noting that the wire labeled zero corresponds to the least significant bit of the input (the right most bit).

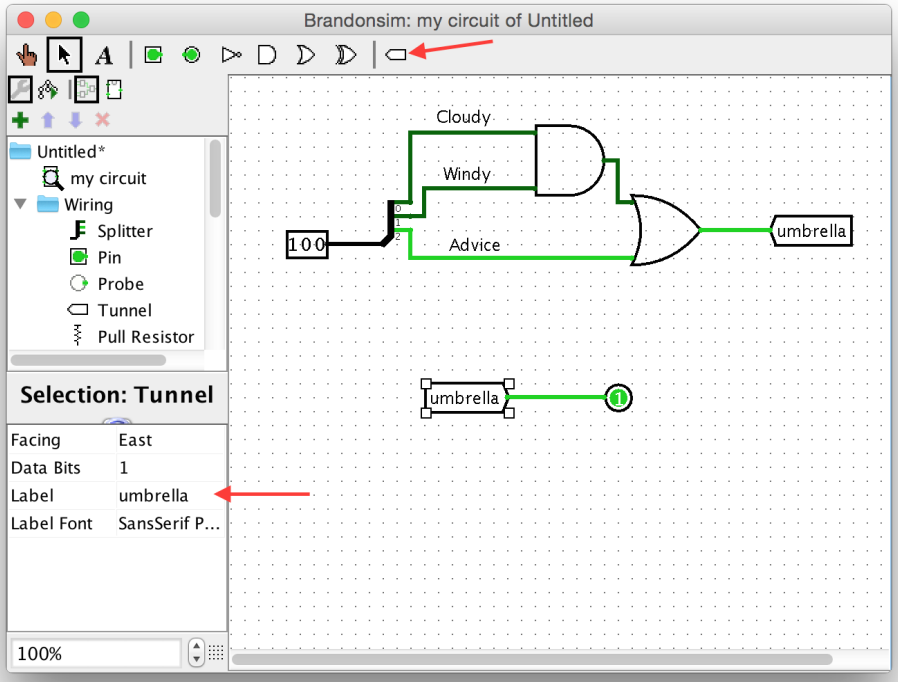


6. Now lets actually wire up the circuit. If we look back to the formula we want to use, we can partition it and rewrite it to make the circuit obvious: IF ((Cloudy AND Windy) OR Weatherman Advice) THEN Take Umbrella. Can you see what gates you need to use? How would you wire it? It may be useful to label the wires so you don't get them confused. Create your circuit so that Wire 0 from the splitter is the Cloudy wire, Wire 1 is the Windy wire, and Wire 2 is the Weatherman's Advice wire. Your circuit should look something like this when you are finished:



7. We are close to being done with our circuit! The last thing we must do is connect the OR gate to the output of the circuit. Instead of directly connecting the gate to an output, we're going to use tunnels. Tunnels are a very useful organizational tool to use when making circuits. They can be thought of as invisible wires; two wires with matching labels will act as if they are connected by a wire. They are not something that exist in real life but rather can be used to tidy up your circuits (which make the TAs VERY HAPPY).

Start by adding an output and two tunnels (found under the wiring folder or by pressing the tunnel button on the very right of the item bar) to the circuit. In the properties of the tunnels, set the label property of both tunnels to the same label (such as “output” or “umbrella”). Connect one of the labels to the OR gate and the other to the output. You should have something like this when you are done. Notice how the value on the wire is carried through the tunnel:



8. We're done with the circuit! Create a label in your circuit with your name and gtid. Lastly, you should test circuit. Try different inputs and make sure that the output is correct!

Part 3

Create a new text file named part3.txt answer the following questions.

1) What is the purpose of a splitter? Can it be used to join wires?

2) Why are tunnels useful?

3) What is the output of your circuit when you give it an input of 010? How about an input of 100?

Note that the correctness of these answers are dependent upon you having followed the directions correctly in part 2.

Evaluation

Your submission will be evaluated based on how well you followed the directions, and your answers to the questions for part 3.

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. **Please do not make a RAR archive.**

Submit your assignment by the deadline on T-Square.