COS 116
The Computational Universe
Laboratory 6: Digital Logic I

In this lab you'll construct simple combinational circuits in software, using a simulator. You'll use these circuits to learn about digital logic.

If you get stuck at any point, feel free to discuss the problem with another student or a TA. However, you are not allowed to copy another student's answers.

Hand in your lab report at the beginning of lecture on Tuesday, April 1. Make sure you answer the questions and complete the deliverables printed in bold. (Number them by Part and Step.) Include a picture of the Majority-of-3 and Vending-Machine-Coin-Counter circuit that you design in Logisim. Also include the boolean expression for Vending-Machine-Coin-Counter. Otherwise you will not get credit for the lab. This lab is due on Tuesday, March 30, 2010.

Part 0: Midterm Review

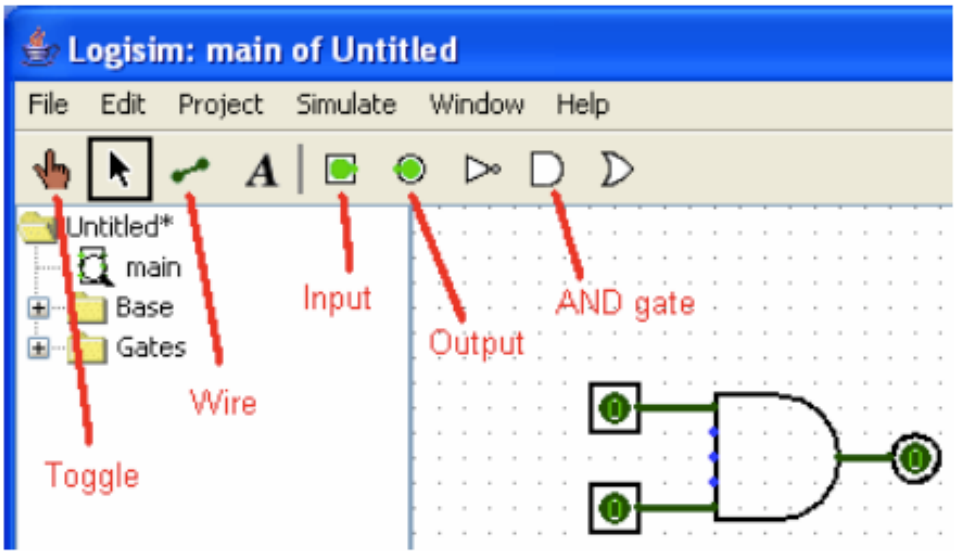
Part 1: Understanding Logisim

Logisim ("logic simulator") is a software program for designing combinational circuits. We will use Logism in this lab and the next lab to help us design complex circuits. The purpose of this section is to help you become comfortable with Logisim. **You do not have to submit your notes for this section as part of your lab report.**

1. Install Logisim by clicking this link:

http://www.cs.princeton.edu/courses/archive/spring08/cos116/lab6_files/logisim.exe

When prompted, save the file to the Desktop. After the download completes, double-click logisim.exe. (If you are warned that this program may be unsafe, click "Run" and "Ok" as needed.)



2. Select the AND gate from the toolbar (see figure above), and place the gate anywhere in the dotted field.

3. Select the Input tool, and place two inputs to the left of the AND gate.

4. Select the Output tool, and place one output to the right of the AND gate.

5. Select the Wire tool, and draw a wire connecting the two inputs to the input pins of the AND gate. Also connect the output to the output pin of the AND gate. Your diagram should look similar to the circuit in the figure above.

6. Use the Toggle tool to change the values of the inputs while observing the output. Confirm that this circuit behaves as it should.

Part 2: Boolean Formulas and Truth Tables

As you saw in the lecture, every Boolean function has three equivalent representations: a Boolean formula, a Boolean circuit, and a truth table. One simple example of a Boolean function is the majority function. Given *n* inputs, the majority function is true if more than half the inputs are true; otherwise it is false. Here is the Boolean formula for the three-input version of the majority function:

$$(A \cdot B) + (B \cdot C) + (A \cdot C)$$

Recall that the "·" symbol means AND, and the "+" symbol means OR. The parentheses are used to specify the order of operations, just like in arithmetic.

Confirm that the Boolean formula given above is correct by completing the truth table below. Make sure that the final column matches the behavior of the majority function.

A	B	C	**(A · B)**	**(B · C)**	**(A · C)**	**A · B) + (B · C) + (A · C)**
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

Part 3: Building Combinational Circuit: Majority-of-3

1. **Construct the majority-of-three circuit in Logisim. Your circuit should have three inputs and one output. Use the Boolean formula above to help you.** If you are having trouble getting started, first determine how many gates your circuit will need.
Important: The gates on your chips have only two inputs. However, in Logisim, gates can have several inputs. Since you will implement your circuit design on the breadboard in Part 6, make sure that all the gates in your Logisim circuit use only two of their input pins.

2. Use the Toggle tool to compare the behavior of your circuit to the truth table you completed in Part 4. Confirm that your circuit correctly computes the majority function.

3. Save a picture of your circuit to hand in with your lab report. There are a few ways to do this. Here is a simple method:

a. Click anywhere in the window of the Logisim program, and press Alt+PrintScreen. This takes a "snapshot" of the active window.
b. Open a new Word document, and press Ctrl+V. This copies your snapshot into the document.

4. Show your circuit to the TA before proceeding to Part 6. This will ensure that any mistakes are caught early, and will save you time.

Part 4: Building Vending-Machine-Coin-Counter

Suppose you are to design a coin counter for a vending machine. Suppose it only accepts quarters that can be placed in 3 slots. Soda can X costs 50 cents and soda can Y costs 75 cents. A person puts some coins and presses X or Y, and the vending machine has to return change and a soda can. Your program should take the following on the input:
Q1, Q2, Q3, quarter slots, each variable has value 1 if coin was placed and 0 if person did not put any coins in a slot.
X, soda selection, has value 1 if a person selected soda X and 0 if a person selected soda Y.

Your circuit should output the following variables:
R, 'release soda', should have value 1 is soda is to be released, 0 otherwise.
C, 'change', should be 1 if quarter is to be returned to the person, 0 otherwise.

1. Write truth table for R as a function of Q1, Q2, Q3 and X

2. Write truth table for C as a function of Q1, Q2, Q3, and X

3. Implement this circuit in Logism.

Further Questions to answer in your report

1. Write the truth table for the majority-of-4 function and represent it with a Boolean expression.

2. Determine the Boolean function that is represented by the following breadboard circuit. Denote each of the inputs and the output of the circuit by a variable, and write the Boolean expression of the function being computed.

3. You verified the correctness of the majority-of-3 circuit by checking that its behavior matches the truth table. This involved checking $2^3 = 8$ combinations of values by hand. How long would this technique take if you were checking a circuit with 100 inputs? (State any assumptions you make.) How long would it take you if you were as fast as the Pentium 4 processor and could check three billion combinations each second?