CS 301K
Boolean Logic Design Project
Due Friday, Sept. 26

This project will give you some experience in building circuitry implementing logic. Rather than work with the physical gates, you will be using a simulator program named Logisim. The project comes in three parts that are fairly close to the three problems I described in class on Friday, Sept. 5. In preparation for working on this project, you might want to review the material I presented then. You may find it at:
http://www.cs.utexas.edu/users/ear/cs301k/LogisimExample.pptx

Lastly, if the project intrigues you, please feel free to extend it for extra credit. As an example, you might expand the third part from being just a circuit to light a single segment of a 7-segment display to a circuit that lights all seven segments. Feel free to discuss with us any ideas you have regarding extensions.

0. First you should download and install the Logisim simulator on your computer. Here are the steps:

a. Go to http://www.cburch.com/logisim/
b. Click Download Logisim
c. Click Logisim's SourceForge.net page
d. Click Download in the green box.
e. The code should install itself.
f. Consider doing the short tutorial found under Help.
g. Discover how to save your design as circ file using the Save as … command under File.

1. Build a circuit to implement the logical expression:

$$(r \vee \neg s) \vee ((p \wedge q) \wedge \neg(p \vee \neg s)).$$

You will have four inputs (i.e., p, q, r, and s) and the one output. Test it on all 16 possibilities for the set of inputs and discover for what sets the output is false (i.e., 0). Use the text capability to specify these sets. Save your circuit as a circ file. Name it xxeidxx1.circ, where xxeidxx is your student eid.
.

2. Build a circuit to implement the logical expression:

$$\neg(p \wedge q) \equiv (\neg p \vee \neg q).$$

This is one of the DeMorgan Laws. Since there is no equivalence gate or even an implies gate, you must go through two preliminary steps to obtain an expression using only $\wedge$s, $\vee$s, and $\neg$s:

a. First, rewrite the expression replacing its original form

$$exp1 \equiv exp2$$

with

$$(exp1 \rightarrow exp2) \wedge (exp2 \rightarrow exp1).$$

b. Then, use the Conditional Disjunction rule to replace the two $\rightarrow$s with $\wedge$s, $\vee$s, and $\neg$s.

You will have two inputs (i.e., p and q) and the one output. Test it on all 4 possibilities for the set of inputs and insure that the output is always true. (It should be if you accept the DeMorgan Laws.). Save your circuit as a circ file. Name it xxeidxx2.circ.

3. Using the third problem in my slides as an example, build a circuit to light the **e** segment (the one at the lower left of the array) of the 7-segement display for numbers between 0 and 9. (Recall my example lit the **f** segment for numbers between 0 and 7.)

a. Determine which numbers should light the **e** segment.
b. There will be four input lines in this case to accommodate the binary representations of 0 through 9 (i.e., 0000 through 1001).
c. Calling the leading bit p, the next bit q, the next bit r, and the last bit s, write a logical expression whose truth value is true for exactly the binary representations of the numbers you determined in part a. (Hint: You will likely have a disjunction of several conjunctions, just as I had in my example.)
d. Build the circuit using the four inputs and a single output: the **e** connection on the 7-sement display. (You get the 7-sement display in the group labeled Input/Output.)

Test it on all 10 possibilities for the set of inputs and insure that the **e** segment is lit exactly when you have the binary versions of the appropriate numbers. Save your circuit as a circ file. Name it xxeidxx3.circ.

4. Submit the three circ files using Canvas.