Finding the Stable States of a Sequential Feedback Circuit

Overview:

In this exercise, you are asked to write a program that identifies all stable states of a sequential feedback circuit given a set of inputs to the circuit. The circuit consists of three cross-connected, two-input gates, and each student is given a randomly selected combination of gates (chosen from AND, OR, XOR, XNOR, NAND, and NOR). The topology and gate types of your unique circuit can be found in steady_states_spec.txt, a file in your svn repository. To check out the files, please use the "svn update" command in your ECE120 svn repository, and all the files for this exercise will appear in the folder named "steady".

The learning objectives of the exercise are as follows:

- Gain skill reading and analyzing simple C code
- Be able to formulate logic expressions on single bits in C

To help you with this exercise, you are given an example program that identifies all the stable states of an S-R latch. The code walks over all possible inputs, assumes a value for Q, calculates the resulting values around the feedback loop, and checks the logic circuit for stability by comparing the assumed value of Q with the value produced by the feedback loop calculation. The code prints each stable state found. For example, it prints "1 1 0 0" for R = S = 1, and Q = $\overline{Q}$ = 0. Figure 1 is the circuit analyzed by the example code.

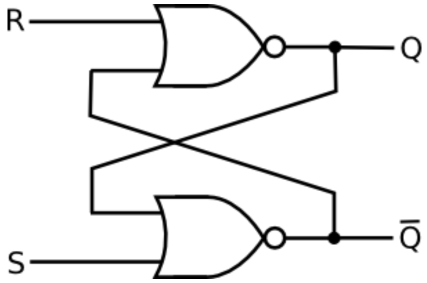
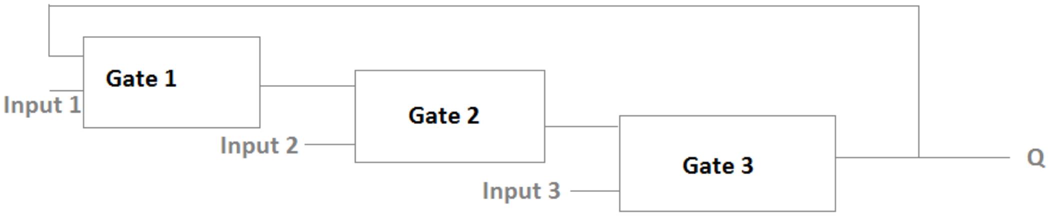
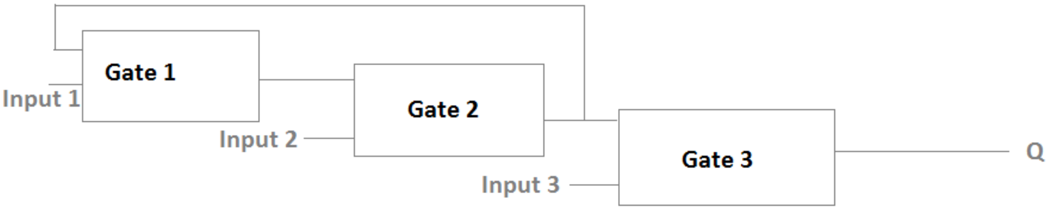


Figure 1

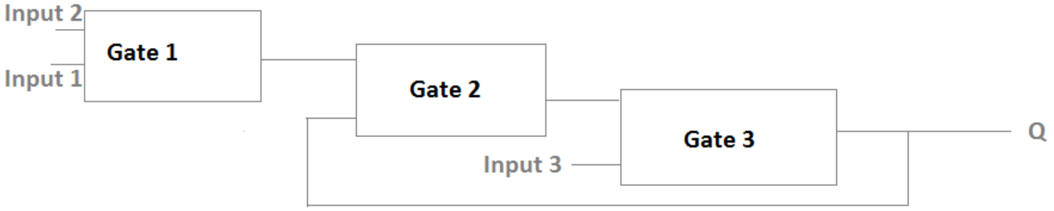
You need to understand how the example code works, and write your own code to determine the stability of your circuit on a given state. Your circuit will be one of the following three types:
Topology 1:



Topology 2:



Topology 3:



The circuit topology and gate types are specified in the file steady_states_spec.txt.

Function:

You should implement the C function main in the file "steady.c" to do the following:

1. Use printf to print a prompt: "Please enter values for input1, input2, input3.\n".
2. Use scanf to read input values for the parameters input1, input2, and input3 (in order).
3. Calculate the output of the circuit.
4. Please print the inputs and internal states (the outputs of each gate) of each stable state found with a single printf: "Stable state found at input1 input2 input3 Gate1out Gate2out Gate3out\n". For example, if only one stable state exists for inputs "1 1 0" and the outputs of the gates are all 0, print "Stable state found at 1 1 0 0 0 0\n". If the inputs cannot end up in a stable state, please print nothing.

Checkout, commit and feedback:

Checkout:

Please use “svn update” in your ECE120 svn repository to get the folder “steady” which has all the files. If you do not have your ECE120 svn, you can check out using “svn checkout https://subversion.ews.illinois.edu/svn/sp16-ece120/<netid>/”

Commit:

After your committing the code to svn (using “svn commit” command), we can detect a new version and begin to grade your code. The grading should take a couple of minutes, even if the demand is high.

Feedback:

Please use “svn update” command to get the feedback after the grading is completed. The feedback will be in separate files named “*.test”, for example, “0.test”. The feedback files are usually one or two sentences indicating your error. Test cases are also available for some errors. For this exercise, we will first test your printf formats, so please make sure that your printf follows the requirements. For any questions regarding the test cases and this exercise, please send an email to beipang2@illinois.edu.

Compilation:

You can use the command “gcc -Wall -g steady.c -o steady” to compile your code.
And use “./steady” to run your code.