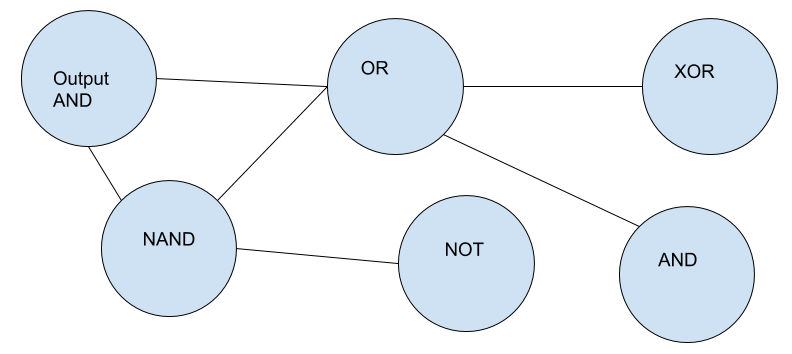
Wilson Hon

Project 1 – Combinational and Sequential Logic Circuit Simulation Report

For this project, I want to report from a top down approach. So, I first started out with the Component Class. The Component Class represent the logic gates for the circuit. The Component is a base class for the derived classes such as AND, OR, NOT, XOR, etc. For the derived class it, each has their own execution function where they perform their actions. However, for the D Flip Flop I had to create a small circuit since it has states and consist of logic gates. At the same time, I had to create another smaller circuit, the Set Reset Latch which the D Flip Flop holds to perform its task.

For the connecting of the logic gates, I created a Node class which would contain a unique derived class of Component (AND, OR, NOT, etc). The Node class has a vector which holds all the other Node pointers. A great visual representation of this is a graph, which is what this is. All Nodes points to other Nodes that they depend on in order to get inputs.



As for the execution, I have a pointer that points to the output node where I would call its execution function. The execution function goes through every edge and executes those Nodes. Executing the Nodes is a Depth First Search graph traversal. When it reaches to the end of the Node where it does not contain any dependents it will execute the Component and return to the previous Node. In the end, the Nodes will have the correct inputs before executing their Components.

What happens when a Node is dependent on the output Node or Nodes that point to the same Node and running that Node’s execution function. Both of those situations will create an infinite loop. So, I set up a condition where it checks to see if the function is currently executing at the moment. If it is currently executing, then don’t call its execution function and request its output. As for D Flip Flops, I was having trouble getting the right inputs if there were multiple Nodes pointing to the D Flip Flops. The problem was, since we are traversing the graph in a Depth First Search traversal. The D Flip Flop will be executed and its state will be updated. So, when a different Node request the output for the D Flip Flop, it will get the incorrect output since the D Flip Flop updated its state. The solution was to iterate through the Node’s dependents vector and save a copy of the D Flip Flop dependents state before executing.

To create the circuit itself, I created the class Circuit which handles the menu and creating and deleting Nodes. The Circuit class also manages all the Nodes and inputs in order to create a functioning circuit. I haven’t really use all that much algorithm besides the graph and DFS for the Nodes. I use a Hash Map for my Set Reset Latch component to store my Nodes, but that seems to be it.