Three-Out-of-Four

By: Erik O. Kriz

Introduction/Theory:

The theory behind this exercise relies on your understanding of how to determine whether a sequence has occurred in a string of binary bits. Many communication systems use start and stop codes to delineate the data and to be able to detect these codes is very important. The theory behind this bit counting relies on storing a state in a series of flip flops which tell the system where you are in the processing of the data. For example, if you are looking to find a 010 string then a state must be set that recognizes the 01 input and then shift to either the state that is determined by a subsequent 0 or 1.

Procedure:

Step 1)

- The first step is to create your state maps.
- This is done by deciding how many states you are going to need
- Then you use those states to create a transition diagram of how you want your states to flow. In this case, use an A/B notation for each transition, where A is your input, and B is the resulting output of that input.

Step 2)

- Using your transition diagram, draw up some karnaugh maps in order to start to get some logical equations on your inputs/output system
- In the K-maps, the variables you should be using are the three starting bits Q0,Q1,Q2, and the input bit I.
- Repeat this process for each of the output bits D2,D1,D0. This will give you the logic to create between each of flip flops.
- Drawing the circuit with appropriate logic between flip flops in logicworks helps with the circuit building.

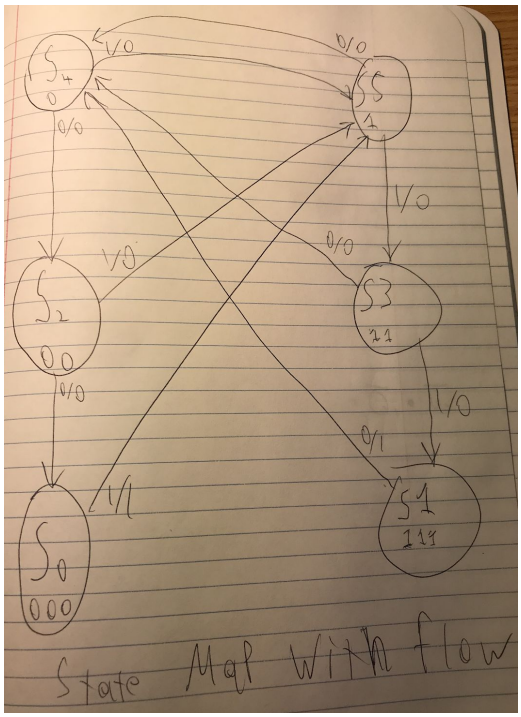
Step 3)

- Next is to physically build the circuit

- Your equations for the D input bits have to be physically implemented here. So it helps if you equations use a low number of total gates and a low number of different kinds of gates.
- For the purposes of this lab, the preset option is not necessary and clearing each flip flop individually is not necessary. So, we hook up all preset slots to power and all of the clear slots to the same switch

Results:

- Frist is my state diagram with flow



- Next is a chart of each initial state, its respective next state, and the resulting output of going from the first state to the second with each respective input.

Q2,Q1,Q0	I	D2, D1, D0	Output
000	0	101	1
000	1	000	0
001	0	001	0
001	1	100	1
010	0	101	0
010	1	000	0
011	0	001	0
011	1	100	0

100	0	101	0
100	1	010	0
101	0	011	0
101	1	100	0
110	0	xxx	x
110	1	xxx	x
111	0	xxx	x
111	1	xxx	x

- Next I have my Karnaugh maps for all of my output bits, D2, D1, D0, and the final output

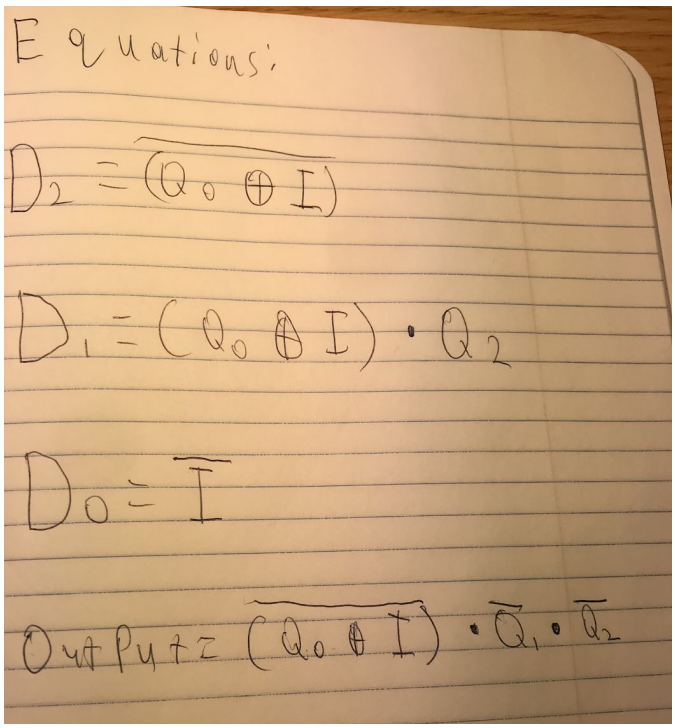
d2	Q0/I			
Q2/Q1	0/0	0/1	1/1	1/0
0/0	1	0	1	0
0/1	1	0	1	0
1/1	x	x	x	x
1/0	1	0	1	0

d0	Q0/I			
Q2/Q1	0/0	0/1	1/1	1/0
0/0	1	0	0	1
0/1	1	0	0	1
1/1	x	x	x	x
1/0	1	0	0	1

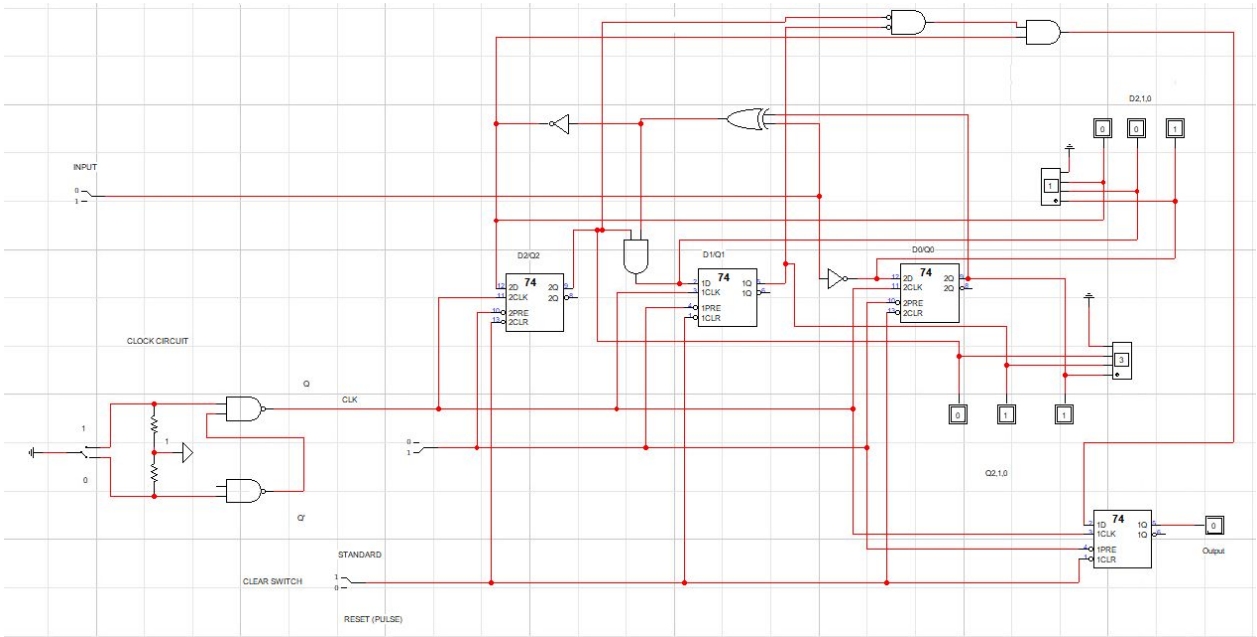
d1	Q0/I			
Q2/Q1	0/0	0/1	1/1	1/0
0/0	0	0	0	0
0/1	0	0	0	0
1/1	x	x	x	x
1/0	0	1	0	1

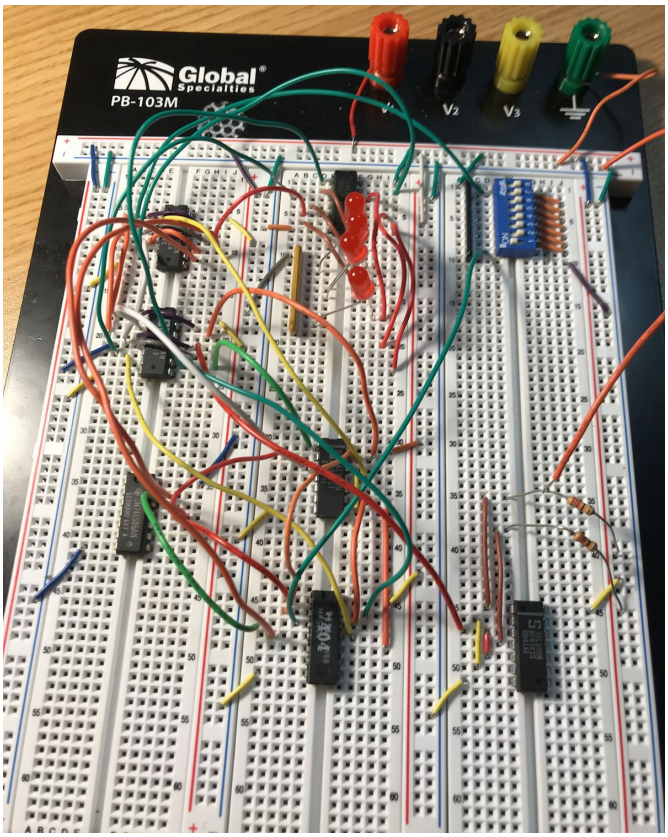
Output	Q0/I			
Q2/Q1	0/0	0/1	1/1	1/0
0/0	1	0	1	0
0/1	0	0	0	0
1/1	x	x	x	x
1/0	0	0	0	0

- Next are the equations I derived from the K-maps



- My final circuit design in logicworks and the real circuit.





- Once this circuit was completed, it followed the state diagram above to the letter and perfectly showed the state flow diagram.

Discussion:

While creating this circuit, the part that took the longest was trying to decide on which numbers to assign to each state. I redrew the karnaugh maps and made logical equations for many different combinations of state numbers in order to reduce my final equations. The name that I finally settled on were the ones that gave me the most reduced equations for all of my D bits and result bit, and I was able to accomplish these equations on my board with just an AND gate, a NOT gate, and an XOR gate (shown above). Another problem with construction was that I did not realize that the output should only output a 1 after you have gotten to a state through the correct path. What I had done at the start was I made it so that if

you were to input the currently selected input, then the output light would go on. This was an easy fix, however as I had a fourth flip flop already hooked up to the clock, clear and preset. All I had to do was rewire the output to a flip flop and then to the buzzer detector light.

Conclusion:

The major point that I think this lab was able to show me is how flip flops can be useful outside of using them just for registers. It has shown me that the ability to store a data bit and send that data bit on a set path to determine the next data bit is extremely useful, and even forms the core of creating sequential logic on real circuit boards.