Lab 12 Report

Kiran Nadkarni

Elizabeth

11/22/19

Sequence Detector

Objectives:

The objective of this lab is to use all of the knowledge of flip flops, shift registers, and state diagrams that we have learned about in previous modules and labs, and apply them to create a physical circuit on our protoboard that can be connected to a buzzer to detect a sequence of three 0's followed by a 1 or a sequence of three 1's followed by a 0. The basic steps that we want to take to complete this assignment is to create a State diagram then a truth table of the flip flops, inputs, future flip flop values and output that we will need, complete K-maps and minimize all of the outputs, and then model the circuit in LogicWorks to then finally build it on the protoboard.

Introduction:

The background to this lab is a situation where we have TAs grading exams, but they want to make sure that too many of the answers are not either true or false in a true false section. So we want to detect a sequence of 4 answers that are either true, true, true and false, or false, false, false and true.

As mentioned before, we are going to create a state diagram of all of the different states that the circuit will be able to sense in our sequence. We can then create a truth table of all possible input and output value combinations for this circuit, next we will create Karnaugh maps of those values, and minimize them to find the input combinations for Qa, Qb, and Qc, the three flip-flops we will be using. We will also be using D-flip flops because they will make our equations simpler when looking for input combinations for the Q outputs.

The state diagram is the potentially most confusing part, because you have to understand that the circuit cannot technically read previous numbers, but it can jump to a different pre-established state based upon those numbers and the current input value. Because we are essentially tracking a set of 3 digits and an input digit, we will only need six states. We will need a state for only seeing a 0, a state for seeing two 0's, a state for seeing three 0's, the three equivalent states for those same number of 1's, and that is it, because we do not need a separate state for the fourth digit, since that will just be our output condition. This leads us to have a total of six states, and since we are using three digits, there are 2^3 or 8 possible combinations of numbers, so two of these extra potential number combinations are "Don't care" states, which will be included in our truth table but not our state diagram.

The specific gate numbers we will be using are two 7408 2-input AND chips, the two 7474 double flip flop chips, a 7432 2-input OR chip, a 7421 4-input AND chip, a 7420 NAND chip for the debouncer, and finally the 7404 inverter chip to invert the A input and the 7405 inverter chip that we normally use for our LEDs.

Procedure:

Even though there are a lot of wires to this circuit, the procedure for building it can be seen as somewhat straightforward.

1- The first thing we did to create this circuit was to design a state diagram and assign a value for Qa, Qb, and Qc for each state. I did this somewhat unconventionally after testing a few different state representations to see which would give me the smallest minimized equations. There might be one that gives even smaller minimized equations but this one was doable with the amount of chips that we had. My states were assigned as 000 for one 0, 100 for two 0's, 110 for three 0's, 001 for one 1, 011 for two 1's, and 111 for three 1's. Because there are three digits in these representations, there is a possibility for 010 and 101 states, which we are going to mark as our "don't care" states on our truth table.
2- The next step is to create a truth table with the inputs of A, Qa, Qb, and Qc. This truth table is below, but it is not very difficult to create once we know our state diagram, which is also linked below. The truth table should have 16 rows because of the 4 inputs mentioned above, and only two of these rows will give an output B as 1, while the other 14, including the don't care rows, give an output 0.
3- The next step is to create our K-maps and minimize the equations, these are also linked below, and they are fairly straightforward once you have the table.
4- After the K-maps have been minimized by grouping the 1's and looking for the x's to help use the least boxes, we can then look to design our circuit in LogicWorks. The LogicWorks circuit will only use 3 flip flops as mentioned above, and the minimized equations will be the inputs to the flip flops and the binary switch that leads to output B.
5- We can now start to build our circuit, which will follow the working LogicWorks diagram exactly, with the 4 LEDs representing the binary probes from the diagram.

Results:

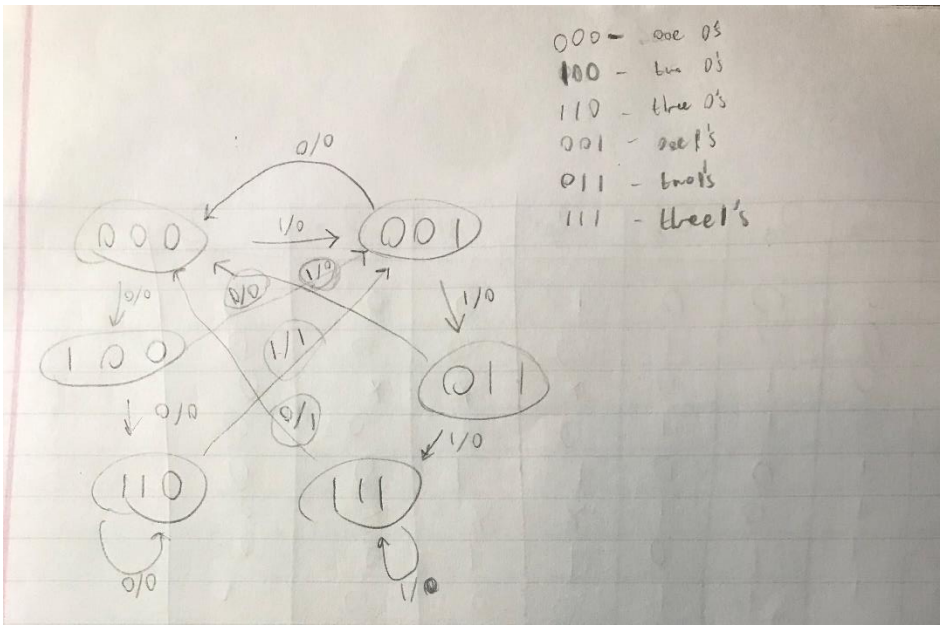
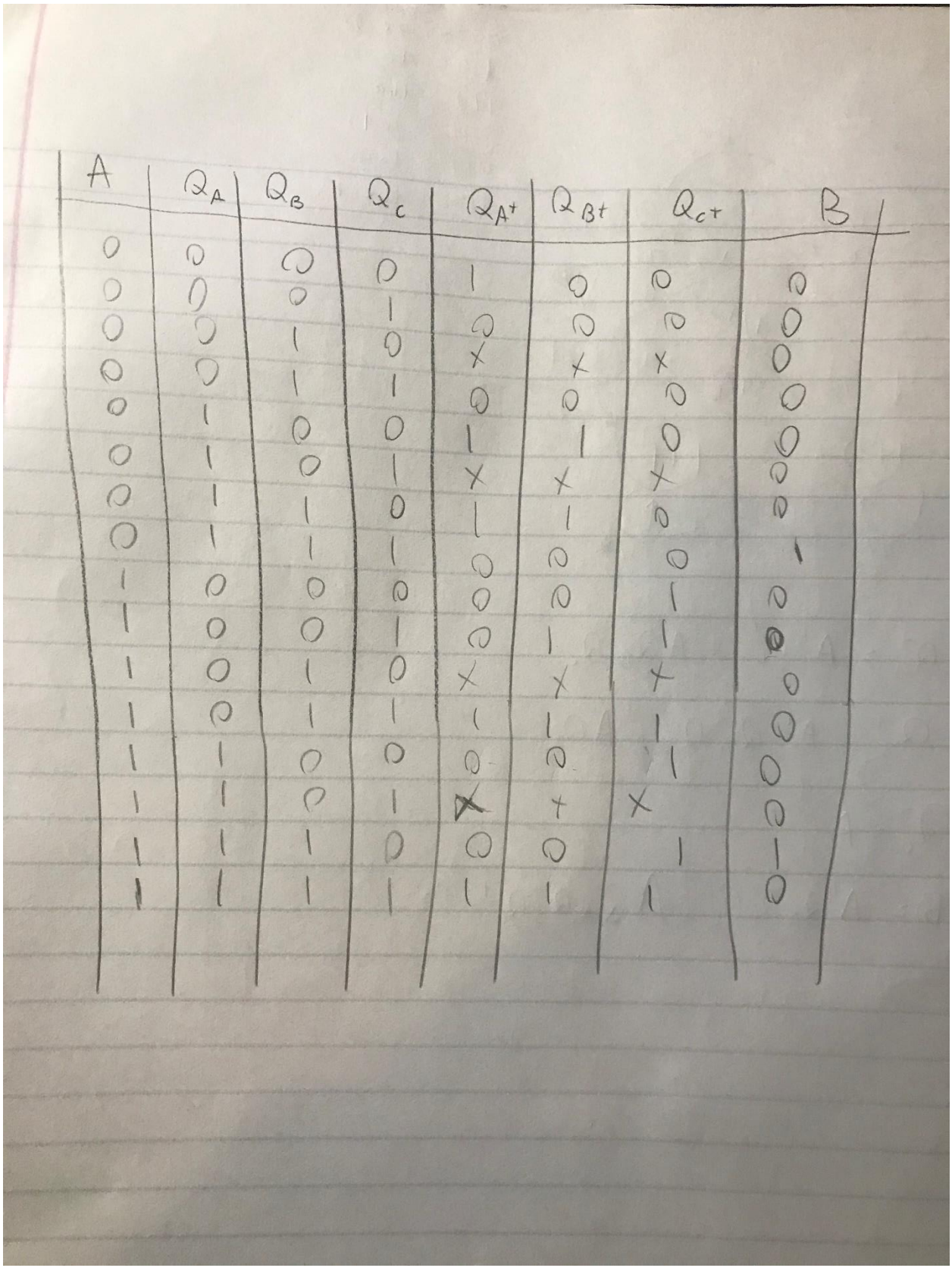


Figure 1: This is my state diagram, which shows the 6 states and what each of these 6 states represents.



A	Q_A	Q_B	Q_C	Q_{A^+}	Q_{B^+}	Q_{C^+}	B
0	0	0	0	1	0	0	0
0	0	0	1	0	0	x	0
0	0	1	0	0	1	0	0
0	0	1	1	0	x	x	0
0	1	0	0	1	x	0	0
0	1	0	1	0	0	1	0
0	1	1	0	0	0	x	0
0	1	1	1	0	0	0	1
1	0	0	0	0	0	1	0
1	0	0	1	0	1	x	0
1	0	1	0	0	x	1	0
1	0	1	1	0	0	0	0
1	1	0	0	0	0	1	0
1	1	0	1	0	1	x	0
1	1	1	0	0	x	1	0
1	1	1	1	0	0	0	0

Figure 2: This is the Transition table for the 4 inputs and next state and output.

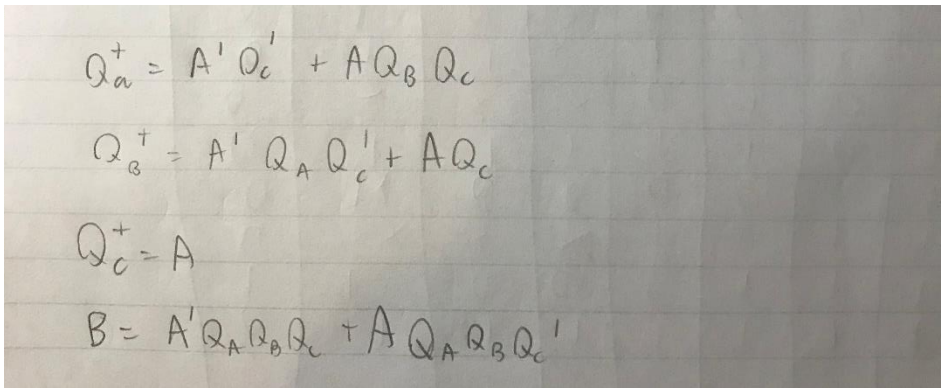


Figure 3: These are my minimized equations for the future values of the flip flops and B, the output.

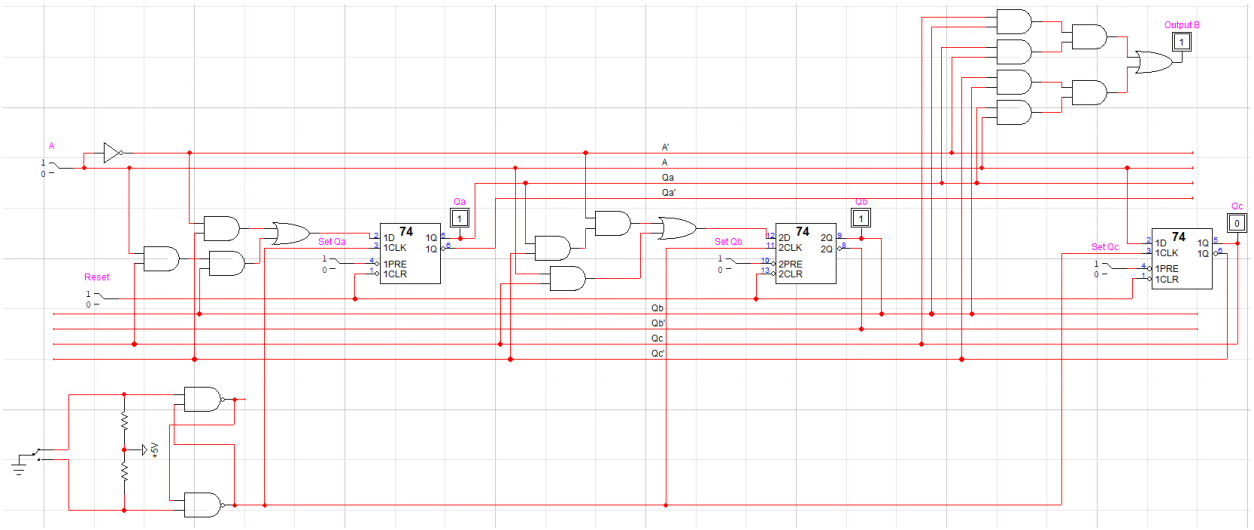


Figure 4: This is my circuit diagram. Notice the debouncer circuit from the last lab, the input A, the output B with the accompanying gates, and the binary probes used to display the output values of the flip flops.

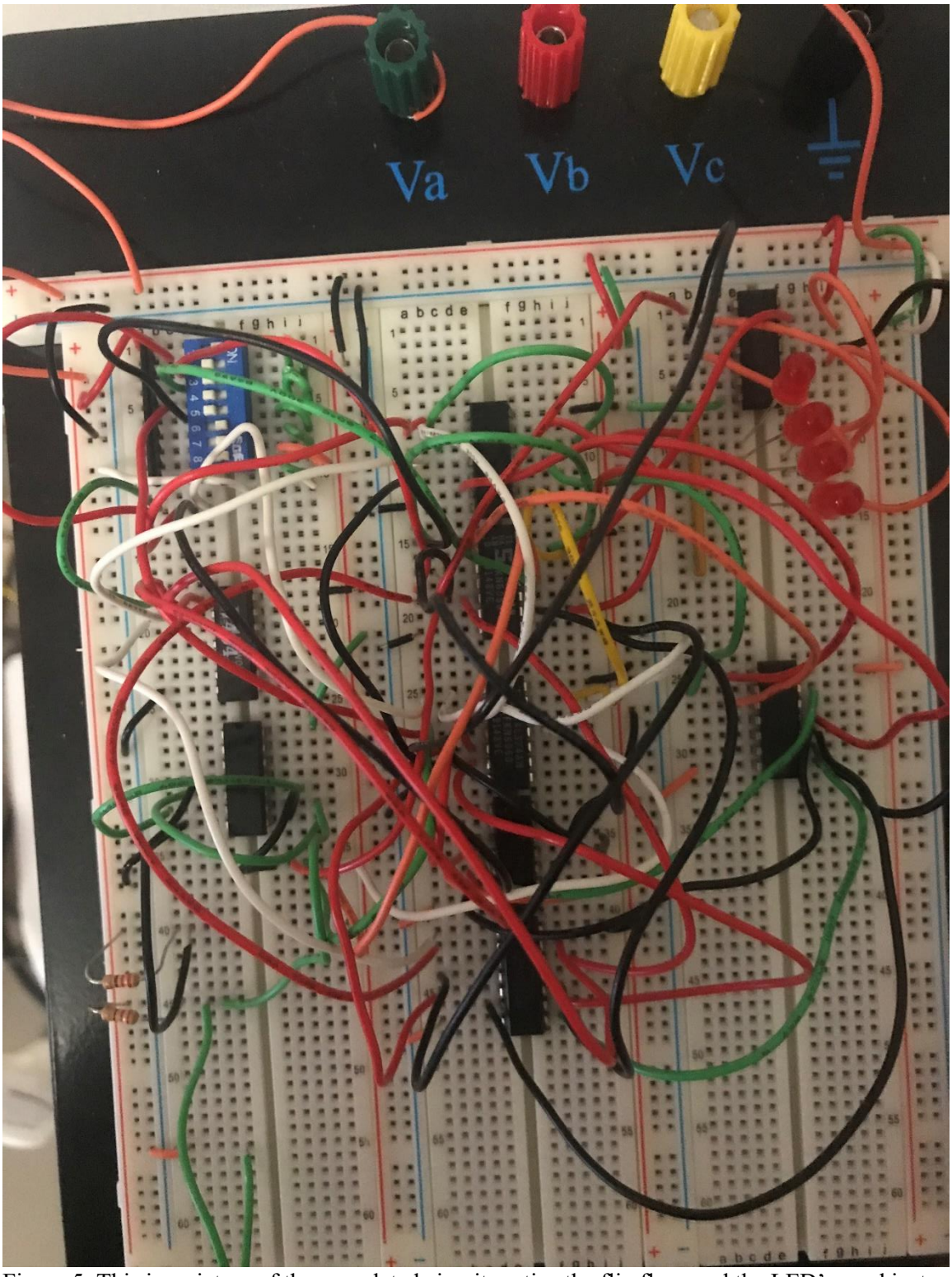


Figure 5: This is a picture of the completed circuit, notice the flip flops and the LED’s used instead of binary probes.

Discussion:

The results I obtained were accurate, since they are what is to be expected for the possible situations that might occur at different input sequences. I did not encounter any specific problems during the creation of my circuit outside of just the normal debugging of misplaced wires. I also did learn about how the numbering system for my state diagram does not have to be in a specific order, the order is arbitrary and the only important factor is that the numbers used for the states are different. In fact, I tried a different state diagram design and realized that the minimized equations for that respective truth table were going to require more gates than my current design, which I tried afterwards to see if I could lower the number of gates used. After understanding the concepts of flip-flops I think this lab is not confusing, but the construction process of the circuit after designing it in LogicWorks was the hardest part because it the largest circuit that I have built for this class.

Conclusion:

I was able to achieve the goal that was set at the beginning of the lab, to create a circuit that could detect the sequences of 0001 and 1110. Being able to utilize my knowledge of state diagrams, flip-flops, and K-maps to bring an idea like this to hardware is something that seems fairly daunting and something that, before reaching this point in the modules and labs, I would not have believed I was capable of doing. The number of steps that were required to create the final product made the arduous process even more rewarding.