Digital Design Lab

ECE 315

Lab/Project 1

Sequence Generator

Group # 4

Alex Smith, Gunter Weisenber, Brandon Rubio

Zeinab Ramezani, TA

University of Miami

2/7/22

Overview

The purpose of this lab is to design and implement a sequence generator. A sequence generator is a sequential circuit that creates a chosen sequence determined by the input sequence. Its outputs work simultaneously with the CLK. Even though various implementation errors and logical issues were discovered in the initial design, the final build successfully created a sequence generator capable of cycling through 10 different states.

Equipment

Tool	Quantity
7474 Dual Positive Edge Triggered D-FF- 2 FF	2
74151 8:1 Mux	4
Breadboard	1
Ets - 7000A Digital - Analog Training system	1
Probe	1

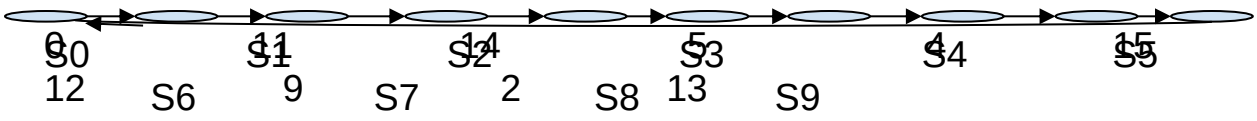
Description

First, we had to obtain the output equations for each of the four flip flops. This was done by constructing the k-maps for each flip flop and deriving the mux inputs for variables A, B, C, D. We did this by using the information provided in the lab description creating the next state equations: A+, B+, C+, D+. Once this was achieved, we set up the circuit accordingly: Connecting the 7474 flip flops inputs according to the equations derived by the k-maps and the outputs to the 4 74151 muxes correspondingly. For the 7474 flip flops making sure VCC and ground are connected, the CLRs are connected to VCC, CKs, while PRs go to ground. We also connected a pulse switch to our CLK in order to cycle through the sequence. At first, we were unable to get the sequence to work. After careful review of our K-maps we discovered that one of the inputs to our MUX was wrong so we re-evaluated the logic and corrected it. However, thi still didn’t result in the correct sequence. Then after debugging and checking all of the

connections we connected all 0 data inputs (from K-map equation) to ground which resulted in the circuit generating the correct sequence.

Design Synthesis

Step 1: Draw state diagram for sequence: 0, 11, 14, 5, 4, 15, 12, 9, 2, 13.



Step 2: Use transition table with D-flip flop to find corresponding present and next state

PS	D MSB	C	B	A LSB	NS	D+	C+	B+	A+
0	0	0	0	0	11	1	0	1	1
1	0	0	0	1		0	0	0	0
2	0	0	1	0	13	1	1	0	1
3	0	0	1	1		0	0	0	0
4	0	1	0	0	15	1	1	1	1
5	0	1	0	1	4	0	1	0	0
6	0	1	1	0		0	0	0	0
7	0	1	1	1		0	0	0	0
8	1	0	0	0		0	0	0	0
9	1	0	0	1	2	0	0	1	0
10	1	0	1	0		0	0	0	0
11	1	0	1	1	14	1	1	1	0
12	1	1	0	0	9	1	0	0	1
13	1	1	0	1	0	0	0	0	0
14	1	1	1	0	5	0	1	0	1
15	1	1	1	1	12	1	1	0	0

Step 3. Create k-map (using ABC for columns and D for rows) for all four Muxes: A B C D

(D’ = D compliment)

Mux A

D\CBA	000	001	010	011	100	101	110	111
0	1	0	1	0	1	0	0	0
1	0	0	0	0	1	0	1	0
	D’	0	D’	0	1	0	D	0

Mux B

D\CBA	000	001	010	011	100	101	110	111
0	1	0	0	0	1	0	0	0
1	0	1	0	1	0	0	0	0
	D’	D	0	D	D’	0	0	0

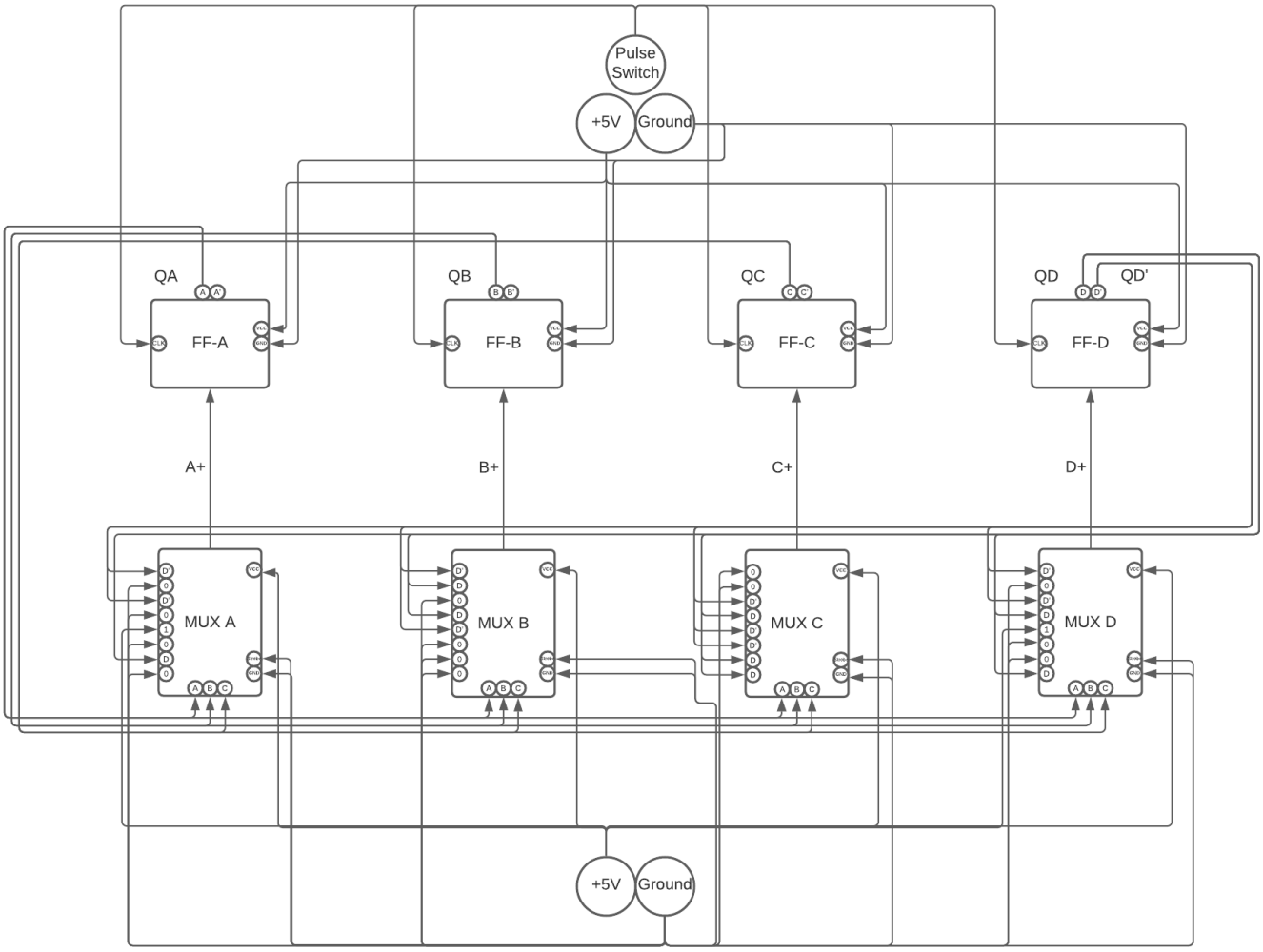
Mux C

D\CBA	000	001	010	011	100	101	110	111
0	0	0	1	0	1	1	0	0
1	0	0	0	1	0	0	1	1
	0	0	D’	D	D’	D’	D	D

Mux D

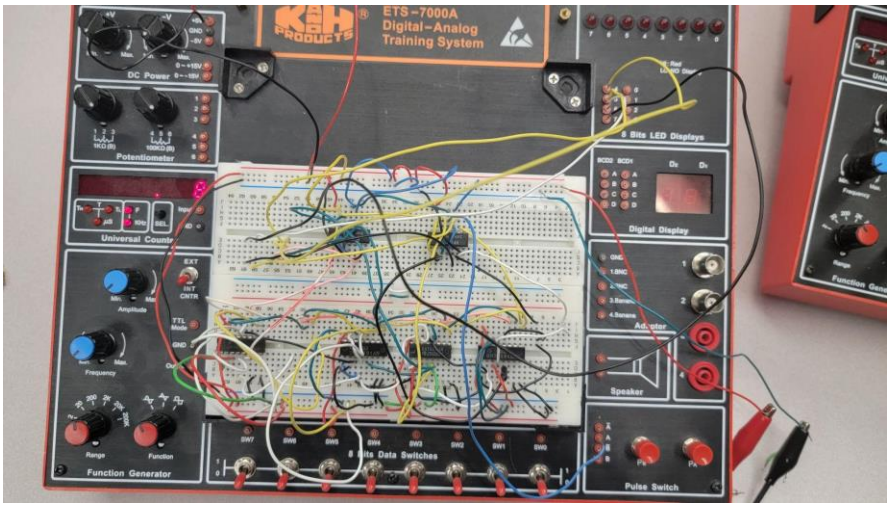
D\CBA	000	001	010	011	100	101	110	111
0	1	0	1	0	1	0	0	0
1	0	0	0	1	1	0	0	1
	D’	0	D’	D	1	0	0	D’

Complete Logic Diagram



Results and Simulations

This section includes description of your simulation and screenshot of simulation results as well as description of results on board. This should also include pictures of the board with captions describing the contents.



Final Circuit

Conclusion

In order to create this sequence generator we had to first design a state machine with our desired outputs. Doing so required a next state diagram and multiple K-maps. This step is vital as it requires exact and precise logic in order for the sequence generator to function properly. Some of our problems stemmed from this step as just one value of one of our K-maps was off thus producing an inaccurate result. Luckily this problem was relatively simple to identify and was dealt with quickly. On the other hand, our most significant problem came from our lack of experience in dealing with these processors. When implementing the circuit our connections and logic were correct however we thought that the 0s on the K-maps meant that there was no need for a wire to be connected there. This was a mistake as the processors somehow gave a feedback current across the lines in the breadboard which gave pins that should have been 0 a low voltage instead. Eventually after much time was spent debugging we figured out that each pin assigned to 0 needed to be connected to ground. All in all, this lab provided us with useful insight into circuit design and implementation as well as proper creation and utilization of sequence generators.