Digital Design Lab

ECE 315

Lab/Project #1A

Sequence Generator

Group #7

Karysse Hay, Nikeem

Dunkelly-Allen and Dibyanshi Mishra

Zeinab Ramezani, TA

University of Miami

2/6/2022

Overview

A sequence generator is a type of digital logic circuit that generates a set of output values. It can be used as a code generator, a counter, a random bit generator and a sequence. This lab focuses on the creation of a simple sequence generator that can generate a sequence which has 10 or more different states. The sequence must contain both the numbers 0 and 15 and must ensure that all invalid states of the machine clear out and set to zero. The sequence generator can be created using multiplexers (MUXs) and flip flops (FFs). A multiplexer is a combinational logic circuit designed to receive multiple data inputs and combine them into a single output by line selection. A flip flop is an electronic circuit component that has two stable states and can be used to store different state information. When done correctly, the sequence generator outputs numbers in the sequence it is designed to follow and in this case, that sequence was: 0, 11, 14, 5, 4, 15, 12, 9, 2, 13. Our sequence generator was not able to generate a sequence of 10 or more different states despite our efforts to build the circuit correctly and to use the correct electronic components. There are several reasons that could have caused faults in the circuit, including a faulty breadboard or faulty MUXs and FFs.

Equipment

Many different pieces of equipment were needed to create a sequence generator. Two D Flip flops with a chip number 7474 were used along with four multiplexers with chip number 74151 Furthermore, a Digital Training kit, breadboard, and about 50-60 wires were used to complete the design.

Description

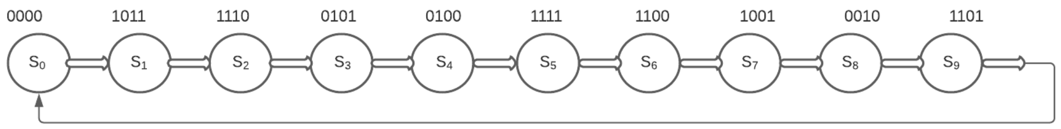
A sequence generator was designed to implement a sequence which has 10 or more different states, including 0 and 15. The design was created using D flip flops and multiplexers, where mux were used to derive the input of each flip flops. K-maps were used to the output equations of the 10 sequences, which were then used as input for the multiplexers. The D flip flops 7474 were connected to the multiplexers 74151 on breadboard via circuit connections. Furthermore, a flip flop clock connection was made to connect to one of the pulse switches from the digital training kit. Once power and other pre-defined pins were connected, Boolean equations were implemented on the breadboard and flip flops were connected to LED connections on the digital training kit. Lastly, mux inputs were connected with outputs based on the sequence generator design.

Design Synthesis

STEPS

1. The sequence was given from the lab assignment: 0, 11, 14, 5, 4, 15, 12, 9, 2, 13
2. The state diagram for the sequence was created as can be seen below.
3. From the truth table given, four K-maps were created for each multiplexer and flip-flop. The D flip-flop synthesis table was used as reference to complete the DCBA K-maps.
4. After the K-maps were created, the next state inputs for the multiplexers were determined and recorded.
5. CBA was used as the address lines for the multiplexers.

STATE DIAGRAM



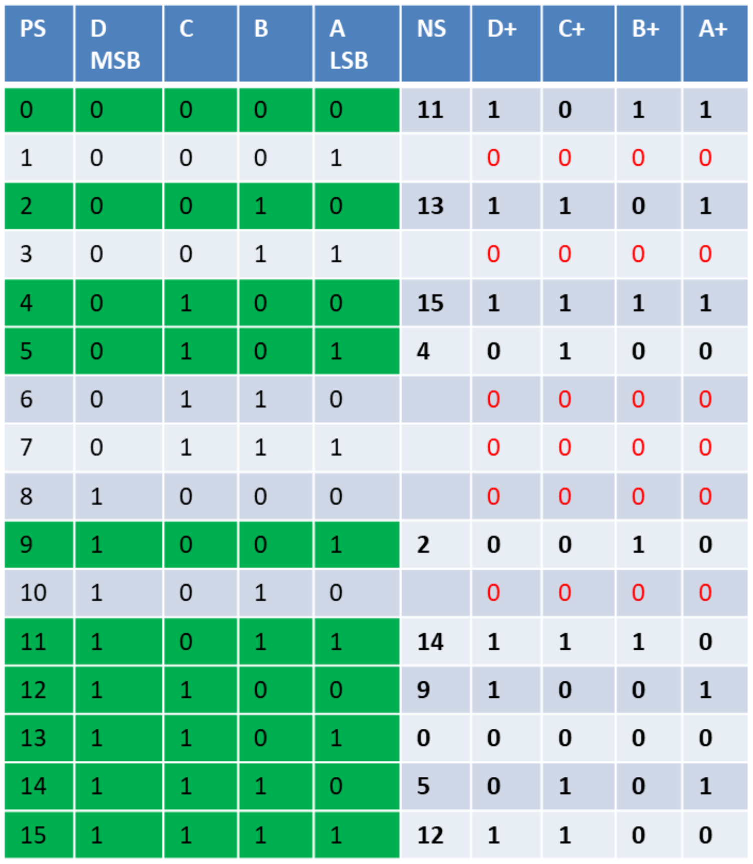
State Diagram for Sequence Given

TRUTH TABLES

Q	Q’	D
0	0	0
0	1	1
1	0	0
1	1	1

Synthesis Table for D- flip flop

D = Q’



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

Truth Table for Sequence Generator

MULTIPLEXER K-MAPS

1.

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
A+	D’	0	D’	0	1	0	D	0

2.

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
B+	D’	D	0	D	D’	0	0	0

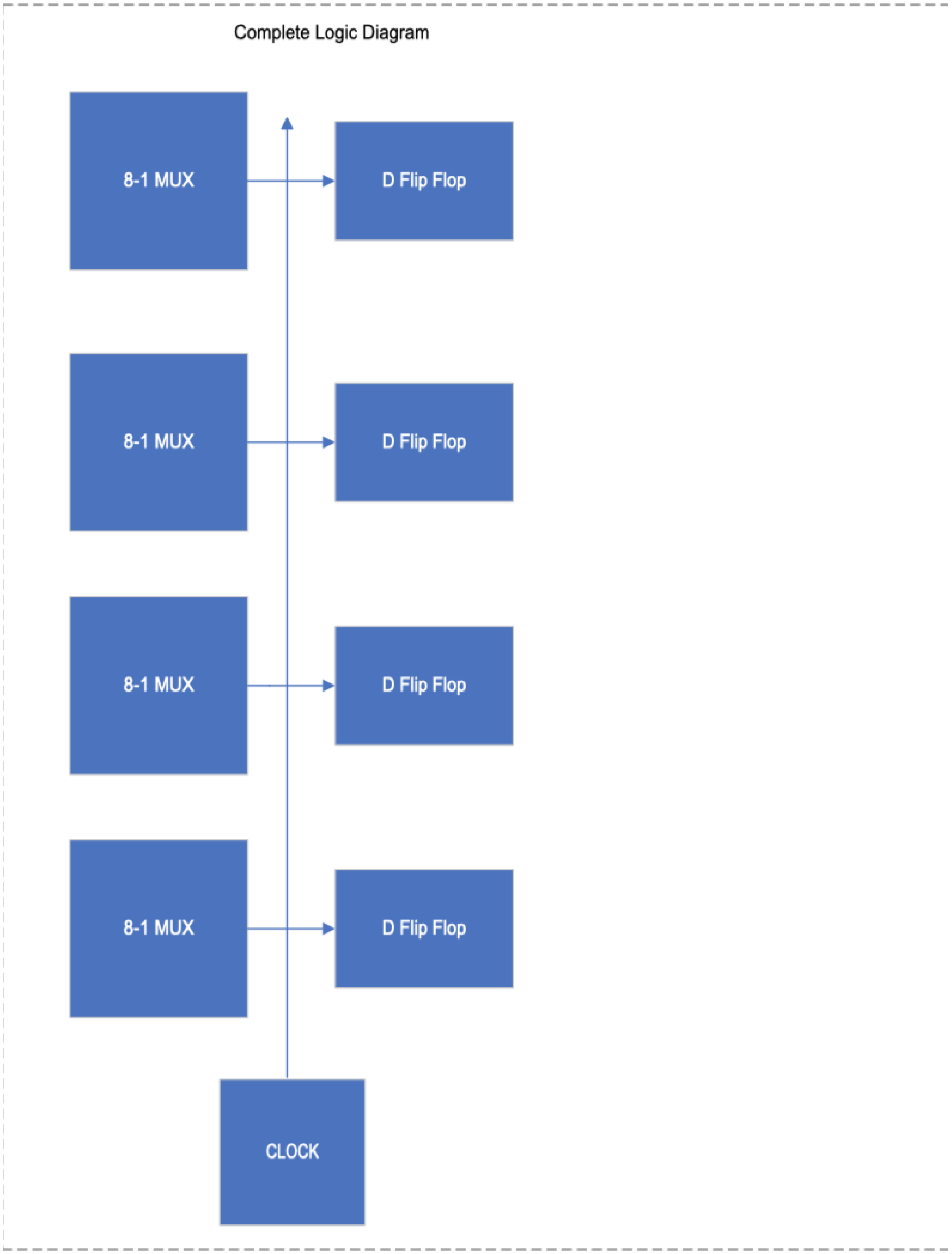
3.

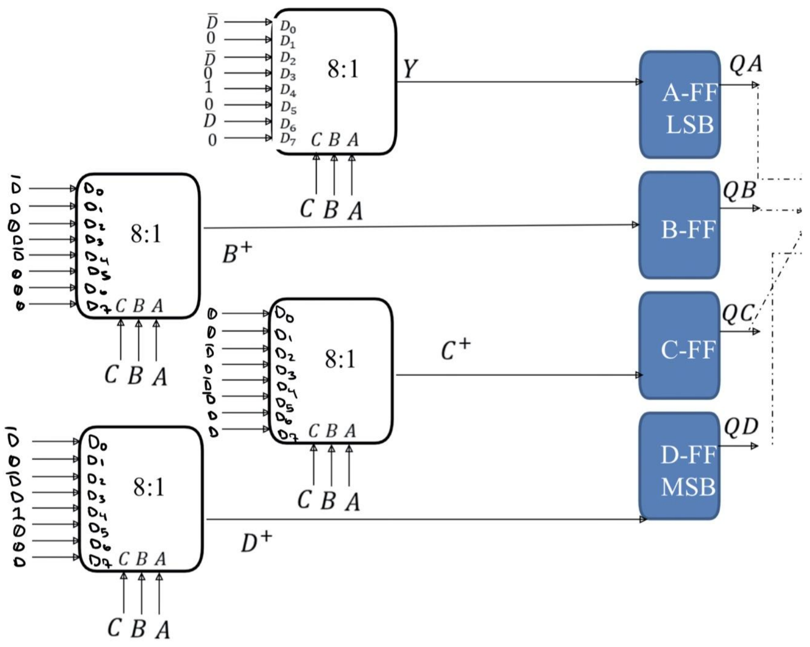
D | CBA	000	001	010	011	100	101	110	111
0	0	0	1	0	1	1	0	0
1	0	0	0	1	0	0	1	1
C+	0	0	D’	D	D’	D’	D	D

4.

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+	D’	0	D’	D	1	0	0	D

Complete Logic Diagram




8:1
Y
A-FF LSB
QA
B-FF
QB
C-FF
QC
D-FF MSB
QD
B+
C+
D+
C B A

Results and Simulations

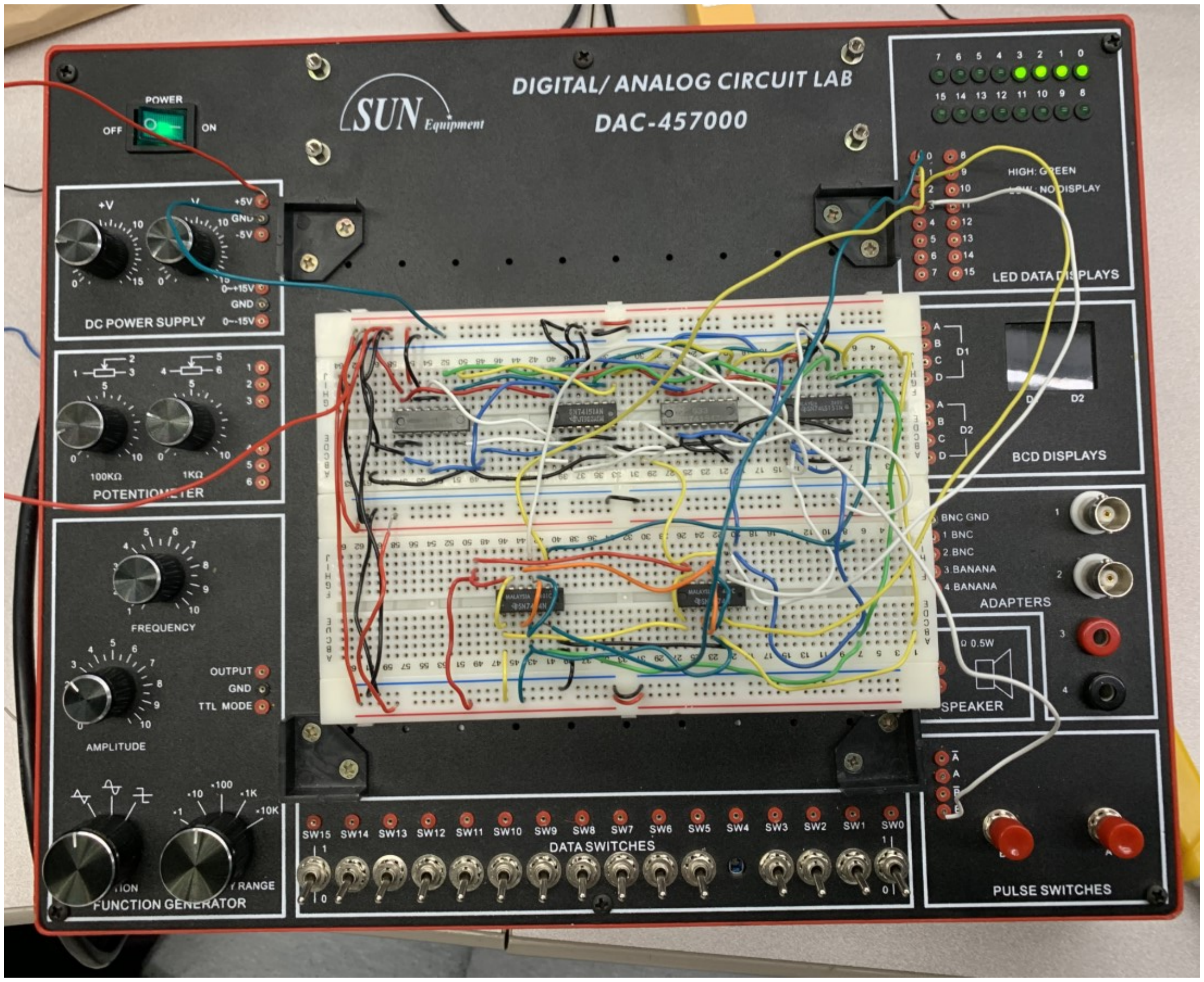


Figure 1: Sequence Generator Circuit

The sequence generator designed did not work and was not able to produce a sequence of 10 or more different states. All the LEDs turned on when the circuit connected to the digital/analog circuit lab, but they did not change with the pulse switch. Several efforts were made to find the fault in the circuit and to create a successful circuit. Firstly, the entire circuit was designed two times with different wires to ensure that no faulty wires were used in the circuit. A current meter was also used to check the connections between all MUXs and FFs to determine if the current was flowing through the circuit. All the MUXs and FFs were changed in case the fault lay in the electrical components and not in the design of the circuit. Despite several efforts, the sequence generator did not work properly. It could have been that the breadboard had broken pins inside or MUXs and FFs used for the second time were faulty.

Conclusion

This lab required to design a sequence generator from the equipment given. The sequence generator was set to contain 10 or more different states. The invalid stages of the machine were set to zero and it was essential that the sequence contained both 0 and 15 in it.

Before we completed the design, all pre-lab work including truth tables, boolean equations etc. were determined and recorded. This gave insight on what the outcomes would be for the flip flops and multiplexer. This information also helped when debugging as we knew what outcomes to look for.

The breadboard was put together by our group. We color coded the wires as much as possible to avoid further problems as it got more complicated (red- power line, black/grey - ground etc.)

However, despite our best efforts, our group faced a huge issue. This issue was the quality of the equipment provided. We were given a faulty wire to use, which we did not know at the time. This caused our sequence generator to skip certain numbers. However, we did not know the wires were faulty until it was too late. This led to our group having to obtain a new breadboard and designing the entire thing a second time. Nevertheless, our sequence generator did not work for a second time as well. Several efforts were made, including restarting the circuit design, replacing all MUXs and FFs and replacing all faulty wires with new ones. Current flow was also checked using a probe to ensure that all current was flowing through all the connections. There could have been several reasons due to which the sequence generator did not work. One of the main reasons could be the electrical components, such as MUXs, FFs and breadboards used. The quality of these components could have caused an error in the sequence generator, causing the LEDs to not change the sequence with pulse switch.

We learnt that sequence generators are important as they can be used to generate primary key values and to generate numeric sequence values like 1, 2, 3, 4, 5 etc.