SEQUENCE DISPLAY IN A SEVEN SEGMENT

Member Names

1. Md. Shahriar Al Hasan **ID# 1511372642**

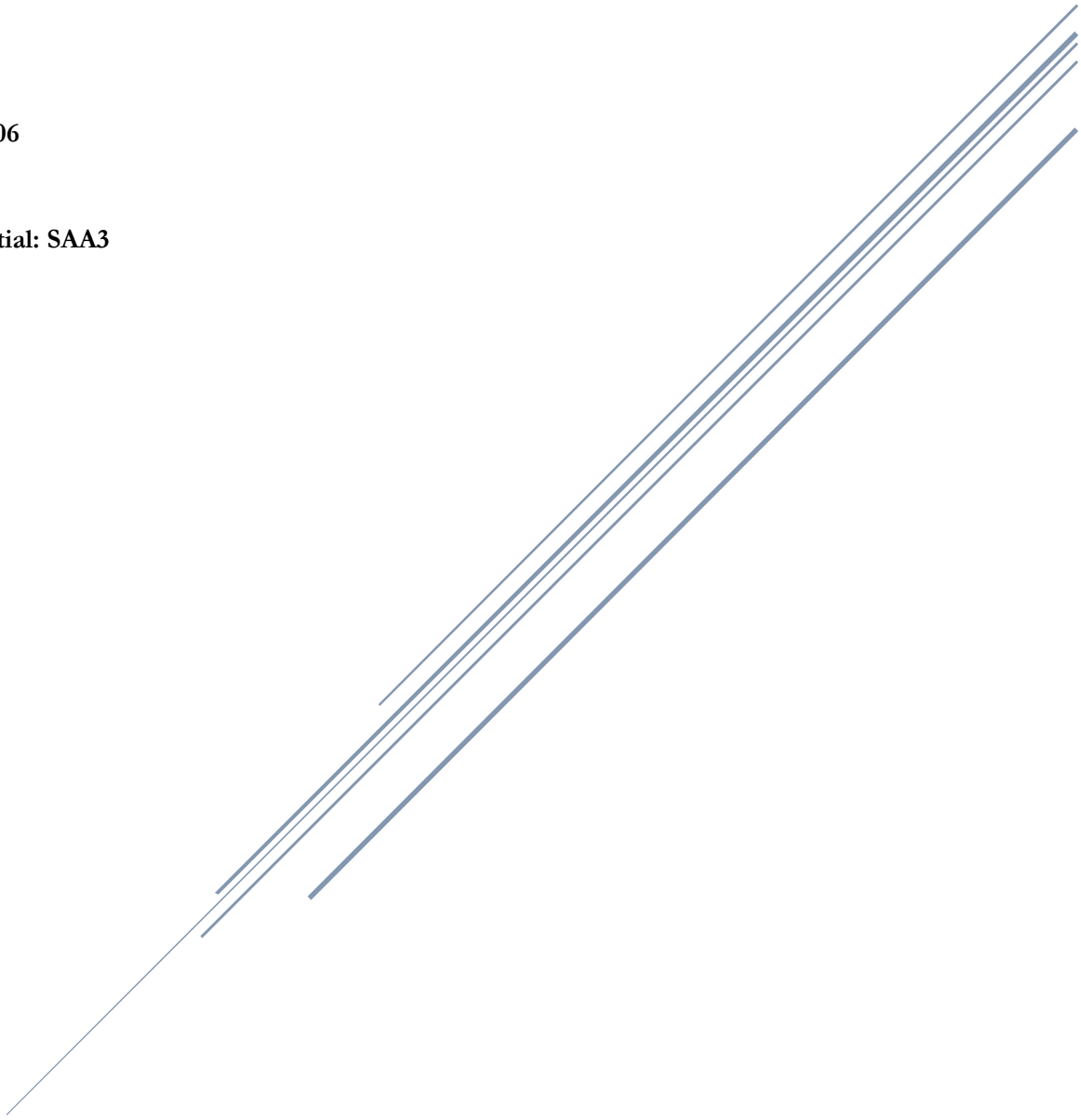
2. Jannatul Ferdaous **ID# 1731733642**

3. Md. Mahir Absar Bin Mohsin **ID# 1812777642**

Group no: 06

Faculty Initial: SAA3

Section: 10



North South University

CSE 231 – Digital Logic Design

Abstract

We will show the sequence "SAA3-DLD-231" in this project. We will implement the combinational part of our circuit via MUX & the sequential part via D Flip-flop. The whole project is divided into two parts, combinational part and sequential part. We are going to solve the combinational part of our project by using basic gates (SOP & POS), universal gates, decoders, multiplexers.

Table of Contents

Introduction

In our seven segment display project, we will be showing a prescribed sequence, "SAA3-DLD-231". Each letter will be displayed for a time period of 2-3 seconds on the seven segment. To do this we had to prepare a combinational part & a sequential part for the circuit. We will also present a demo as well as a software implementation of our project.

Background

As we can see, we have used Multiplexers, Flip-Flops, Seven Segment Display and a timer IC to implement the circuit. We could have used Decoders also instead of Multiplexers.

Timer Circuit:

The designed circuit is a synchronous sequential circuit which means that the memory components of the sequential logic design depends on the rising edge of the clock signal to change their state. To generate a regular clock pulse, the NE555 timer IC was used in its stable mode. In the stable mode the NE555 generated regular timer square wave pulse which is suitable to be used as the clock pulse of the sequential circuit. The formula used to calculate the frequency is provided below-

$$f = (1/T)$$

$$= (1.44)/(R1+2R2)*C$$

Multiplexer:

A Multiplexer is a combinational circuit that selects binary information from one of many input lines & directs it to a single output line. Selection of input lines is controlled by a set of selection lines. A truth table for a 4:1 MUX is given below:

S1	S2	Y
0	0	I0
0	1	I1
1	0	I2
1	1	I3

Decoder:

A combinational circuit that converts binary information from n input lines to a maximum of 2^n unique output line is a decoder. Its purpose is to generate the 2^n (or fewer) minterms of n input variables. Each combination of inputs will assert a unique output.

Inputs			Outputs							
X	Y	Z	D0	D1	D2	D3	D4	D5	D6	D7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0

1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Flip-Flop

A Flip Flop is a binary storage device that has two stable states and is capable of storing one bit binary information. The output can only change when a clock pulse is supplied. We will be using D Flip-Flop in our circuit as it is easier to work with.

The D Flip-Flop captures the value of the D- input at a definite portion of the clock cycle. That captured value becomes the Q output.

Clock	D	Q_{n+1}
0	X	Q
1	0	0
1	1	1

Design Methodology

Design of Combinational Circuit:

In the combinational part, we had to identify 7 segment outputs **a, b, c, d, e, f, g** for the 4 inputs **W, X, Y, Z** to represent the sequence **"SAA3-DLD-CSE231"**. We accomplished this by generating a truth table that shows the 7 segment outputs for all the possible values of inputs. After that we had to feed the inputs to the 7 segment display, so that we get our desired sequence. To do this we used several methods.

Truth Table:

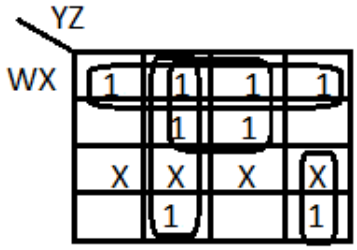
Index	Characters	Inputs				Seven Segment Outputs						
		W	X	Y	Z	a	b	c	d	e	f	g
0	S	0	0	0	0	1	0	1	1	0	1	1
1	A	0	0	0	1	1	1	1	0	1	1	1
2	A	0	0	1	0	1	1	1	0	1	1	1
3	3	0	0	1	1	1	1	1	1	0	0	1
4	-	0	1	0	0	0	0	0	0	0	0	1
5	D	0	1	0	1	1	1	1	1	1	1	0
6	L	0	1	1	0	0	0	0	1	1	1	0
7	D	0	1	1	1	1	1	1	1	1	1	0
8	-	1	0	0	0	0	0	0	0	0	0	1
9	2	1	0	0	1	1	1	0	1	1	0	1
10	3	1	0	1	0	1	1	1	1	0	0	1
11	1	1	0	1	1	0	1	1	0	0	0	0

Combinational circuit using basic gates:

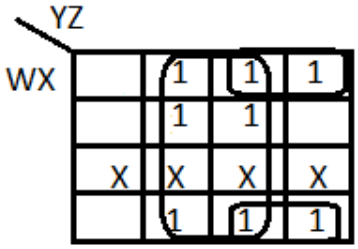
At first we tried to use basic gates to display the sequence. For this we generated a function for each output of the segment display by the help of K-maps.

By combining the 1's in the K-map we got the first canonical form (i.e. sum of products) in term of the inputs **W, X, Y, Z**. Whereas by combining the 0's in the K-maps we got the second canonical form (i.e. product of sums) in term of the inputs **W, X, Y, Z**.

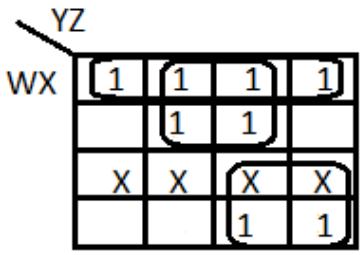
K-maps (Sum of Products):



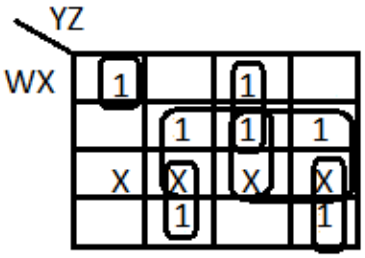
a= W'X'+W'Z+Y'Z+X'YZ'



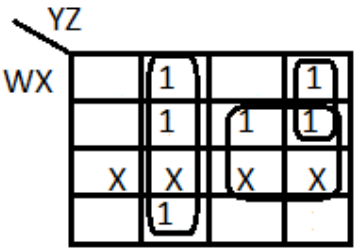
b= X'Y+Z



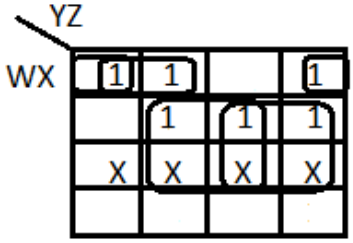
c= W'X'+W'Z+WY



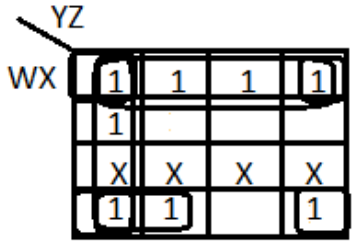
d= W'X'Y'Z' + W'YZ+
XY+ XZ+ WY'Z+ WYZ'



e= W'YZ'+XY+Y'Z



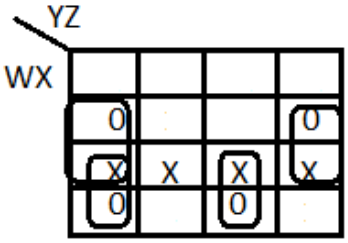
f= XZ+XY+W'X'Y'+W'X'Z'



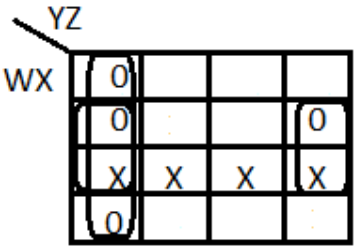
g= W'X'+X'Y'+X'Z'+Y'Z'

K-Maps for Seven Segment functions SOP

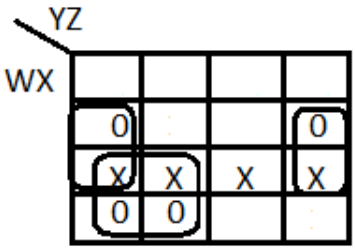
K-maps (Product of Sum):



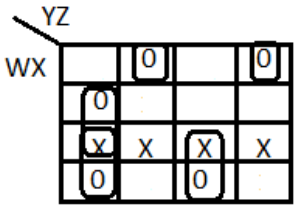
a= (X'+Z)(W'+Y+Z)
(W'+Y'+Z')



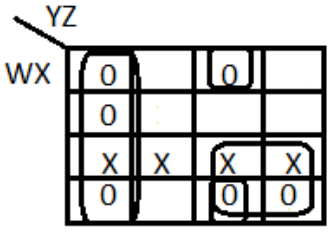
b= (Y+Z)(X'+Z)



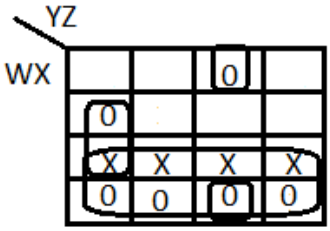
c= (W'+Y)(X'+Z)



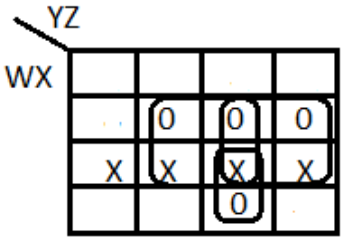
d= (W+X+Y+Z')
(W+X+Y'+Z)(X'+Y+Z)
(W'+Y+Z)(W'Y'Z')



e= (Y+Z)(W'+Y')(X+Y'+Z')



f= W'(X+Y'+Z')(X'+Y+Z)



g= (X'+Z')(X'+Y')
(W'+Y'+Z')

K-maps for Seven Segment functions POS

Designing the combinational circuit with decoder:

Alternatively we used decoders to produce the desired sequence. We have four inputs, we used a 4*16 decoder. A decoder activates (i.e. set to 1) one output rail. For each combination of input, while the rest remains zero. That is the decoder generates a minterm for each combination of input. As a result we combined the outputs of the decoder along with external OR-gates to produce a function for each output of the Seven Segment display. The output from the OR-gates is then routed into the corresponding pins of the Seven Segment display.

Index	Characters	Inputs				Decoder Outputs												Seven Segment Outputs						
		W	X	Y	Z	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	a	b	c	d	e	f	g
0	S	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	1	1
1	A	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	1	1	1	0	1	1	1
2	A	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	1	0	1	1	1
3	3	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1
4	-	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
5	D	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	1	1	1	1	1	1	0
6	L	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	1	0
7	D	0	1	1	1	0	0	0	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	0
8	-	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1
9	2	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	1	1	0	1
10	3	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	0	0	1
11	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0

Truth-Table for Decoder

Designing the combinational circuit using MUX:

Finally we used 8:1 MUX to implement our circuit. As the Seven Segment display has seven different outputs we had to use seven 8:1 MUX. An 8:1 MUX has 3 selector bits & 8 inputs. We used W, X, Y namely the most significant bits as our selector bits. The value of this selector bits determine which input is activated at the output. The output of the MUX is fed into the corresponding pin of the 7 segment display to produce the desired sequence.

Index	Characters	Inputs				Seven Segment Outputs							Multiplexers Inputs						
		W	X	Y	Z	a	b	c	d	e	f	g	Ma	Mb	Mc	Md	Me	Mf	Mg
0	S	0	0	0	0	1	0	1	1	0	1	1	I0 =1	I0 = Z	I0 =1	I0 = Z'	I0 = Z	I0 =1	I0 =1
1	A	0	0	0	1	1	1	1	0	1	1	1							
2	A	0	0	1	0	1	1	1	0	1	1	1	I1 = 1	I1 =1	I1 =1	I1 = Z	I1 = Z'	I1 = Z'	I1 =1
3	3	0	0	1	1	1	1	1	1	0	0	1							
4	-	0	1	0	0	0	0	0	0	0	0	1	I2 = Z	I2 = Z	I2 = Z	I2 = Z	I2 = Z	I2 = Z	I2 =Z'
5	D	0	1	0	1	1	1	1	1	1	1	0							
6	L	0	1	1	0	0	0	0	1	1	1	0	I3 = Z	I3 = Z	I3 = Z	I3 =1	I3 =1	I3 =1	I3 =0
7	D	0	1	1	1	1	1	1	1	1	1	0							
8	-	1	0	0	0	0	0	0	0	0	0	1	I4 = Z	I4 = Z	I4 =0	I4 = Z	I4 = Z	I4 =0	I4 =1
9	2	1	0	0	1	1	1	0	1	1	0	1							
10	3	1	0	1	0	1	1	1	1	0	0	1	I5 =Z'	I5 =1	I5 =1	I5 = Z'	I5 =0	I5 =0	I5 =Z'
11	1	1	0	1	1	0	1	1	0	0	0	0							

Truth-Table for MUX

Justification of selected Design for combinational circuit:

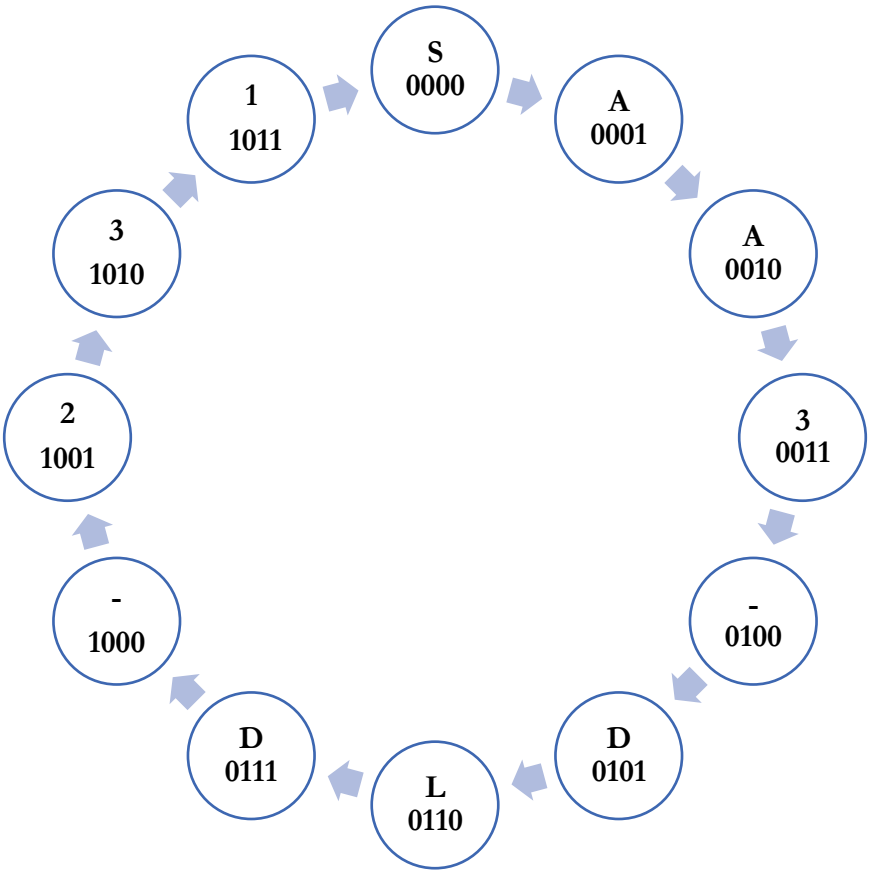
So far, we have solved the combinational part of our project by various methods, but we choose one of the many method to implement our circuit.

If we use decoder then we need to use both the basic gate IC's & the decoder IC's. This process is not only less economically beneficial but also hard to interpret because of connection between two different IC's. Using basic gates would make the circuit more complex as more IC's are required. This process is also expensive. If we use MUX, the circuit becomes easier to implement as it is more cost effective & time efficient.

Sequential Design:

We need four Flip-flops to implement the sequential circuit as there are 12 states. The combinational part of our sequential circuit consists of three 8:1 MUX whose outputs goes to the Flip-flops. The state of the first three Flip-flop (i.e. the Flip-flop holding the MSB) acts as the selector bits our MUX. The state of the Flip-flop that holds the LSB together with other values acts as the inputs of the MUX. Upon application of each clock pulse the value of the Flip-flop changes such that the state transition occurs in correct order & the desired sequence is produced.

State Diagram:



State Table:

Index	Character	Present State				Next State				Flip-Flops				Multiplexers			
		Wt	Xt	Yt	Zt	Wt+1	Xt+1	Yt+1	Zt+1	DW	DX	DY	DZ	DW	DX	DY	DZ
0	S	0	0	0	0	0	0	0	1	0	0	0	1	I0 =0	I0=0	I0 =Zt	I0 =Zt'
1	A	0	0	0	1	0	0	1	0	0	0	1	0				
2	A	0	0	1	0	0	0	1	1	0	0	1	1	I1 =0	I1=Zt	I1 =Zt'	I1 =Zt'
3	3	0	0	1	1	0	1	0	0	0	1	0	0				
4	-	0	1	0	0	0	1	0	1	0	1	0	1	I2 =0	I2=1	I2 = Zt	I2 =Zt'
5	D	0	1	0	1	0	1	1	0	0	1	1	1				
6	L	0	1	1	0	0	1	1	1	0	1	1	1	I3 =Zt	I3=Zt'	I3 =Zt'	I3 =Zt'
7	D	0	1	1	1	0	0	0	0	1	0	0	0				
8	-	1	0	0	0	1	0	0	1	1	0	0	1	I4=1	I4=0	I4=Zt	I4=Zt'
9	2	1	0	0	1	1	0	1	0	1	0	1	0				
10	3	1	0	1	0	1	0	1	1	1	0	1	1	I5= Zt'	I5=0	I5=Zt'	I5=Zt'
11	1	1	0	1	1	0	0	0	0	0	0	0	0				

Justification of Sequential Design:

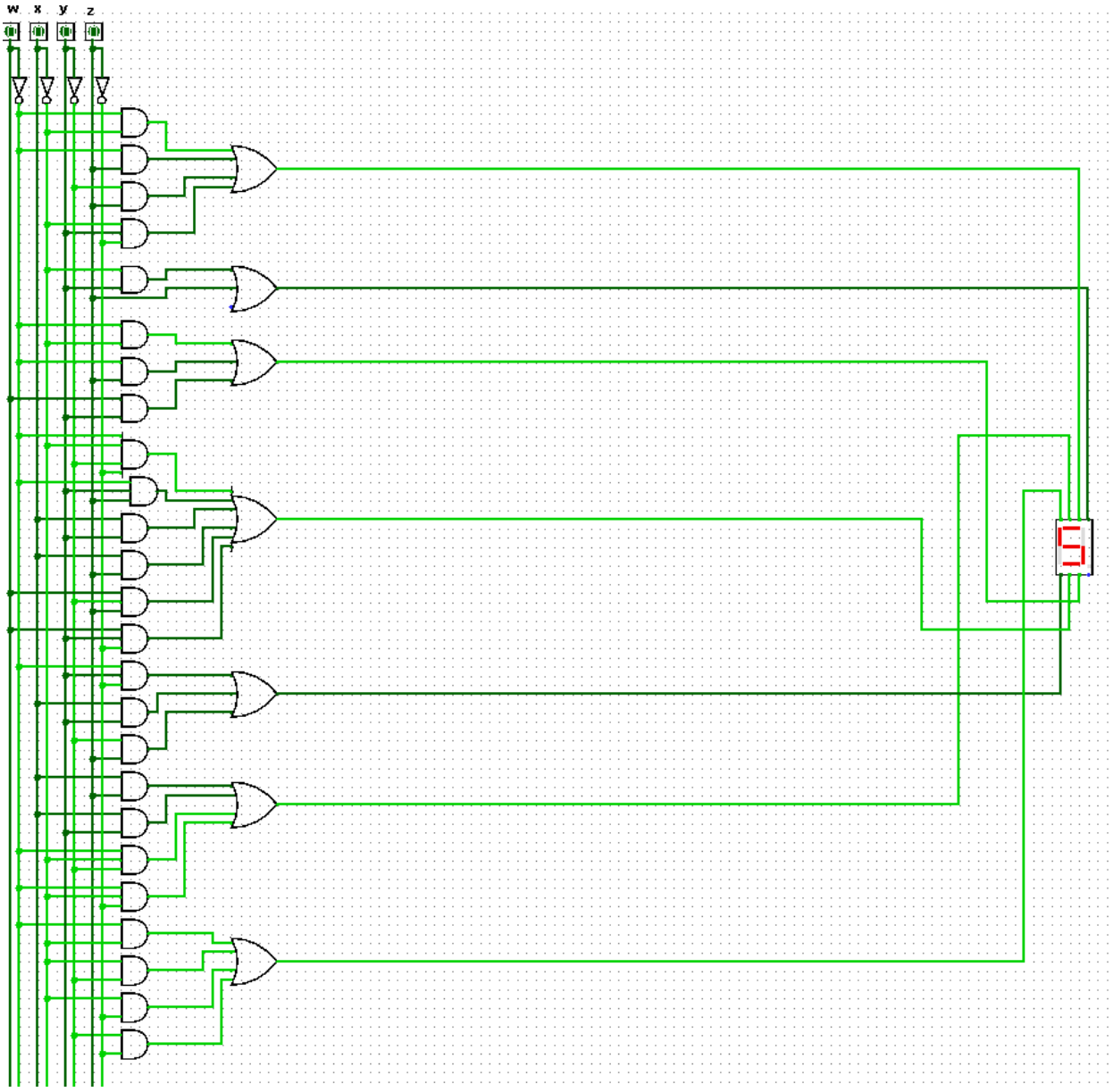
We choose D Flip-flop to implement our circuit because it is less expensive & easier to interpret if we do so. Using JK Flip-flop would require implementation of 8 functions. We also need to compute the inputs of the JK Flip-flop from the value of the previous state and other inputs so as to get a particular next state, because unlike D Flip-flop the inputs and outputs of the Flip-flop aren't the same. If we use T or D Flip-flop we need to implement four functions. We don't choose T Flip-flop as the inputs and outputs of the Flip-flops are different.

Equipments:

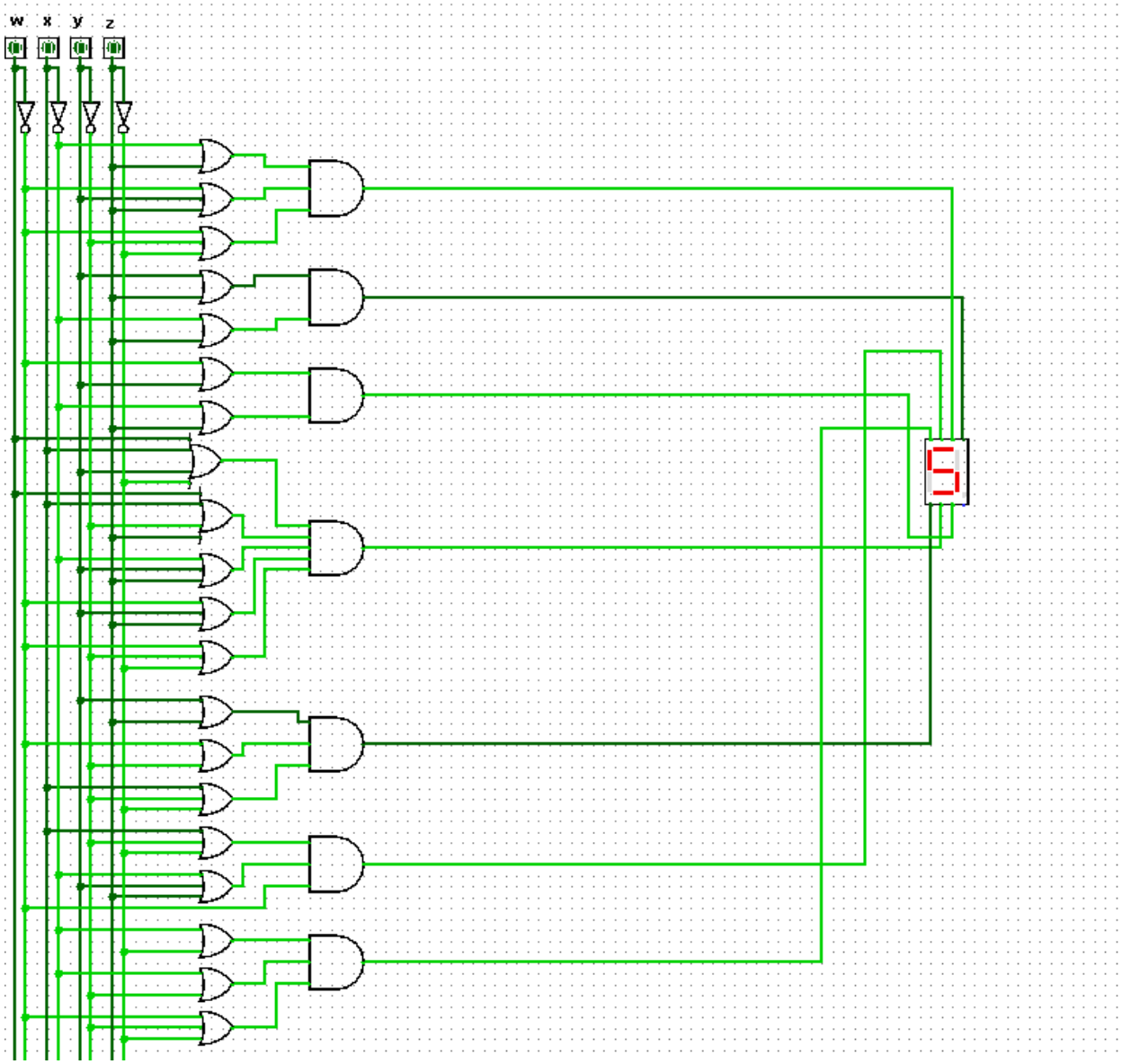
	Items	Quantity
1.	IC 7404 (NOT gate)	1
2.	IC 74151 (8:1 MUX)	11
3.	IC 7474 (Dual D Flip-flop)	2
4.	NE555 (timer IC)	1
5.	Seven Segment Display	1
6.	LM7805 voltage regulator (5V)	1
7.	9V battery	1
8.	9V battery connector	1
9.	1K ohm Resistor	2
10.	1000 micro Faraday Capacitor	1
11.	Jumper wires (M to M)	200
12.	Breadboard (big)	3
13.	Breadboard (small)	1

Results & Discussion

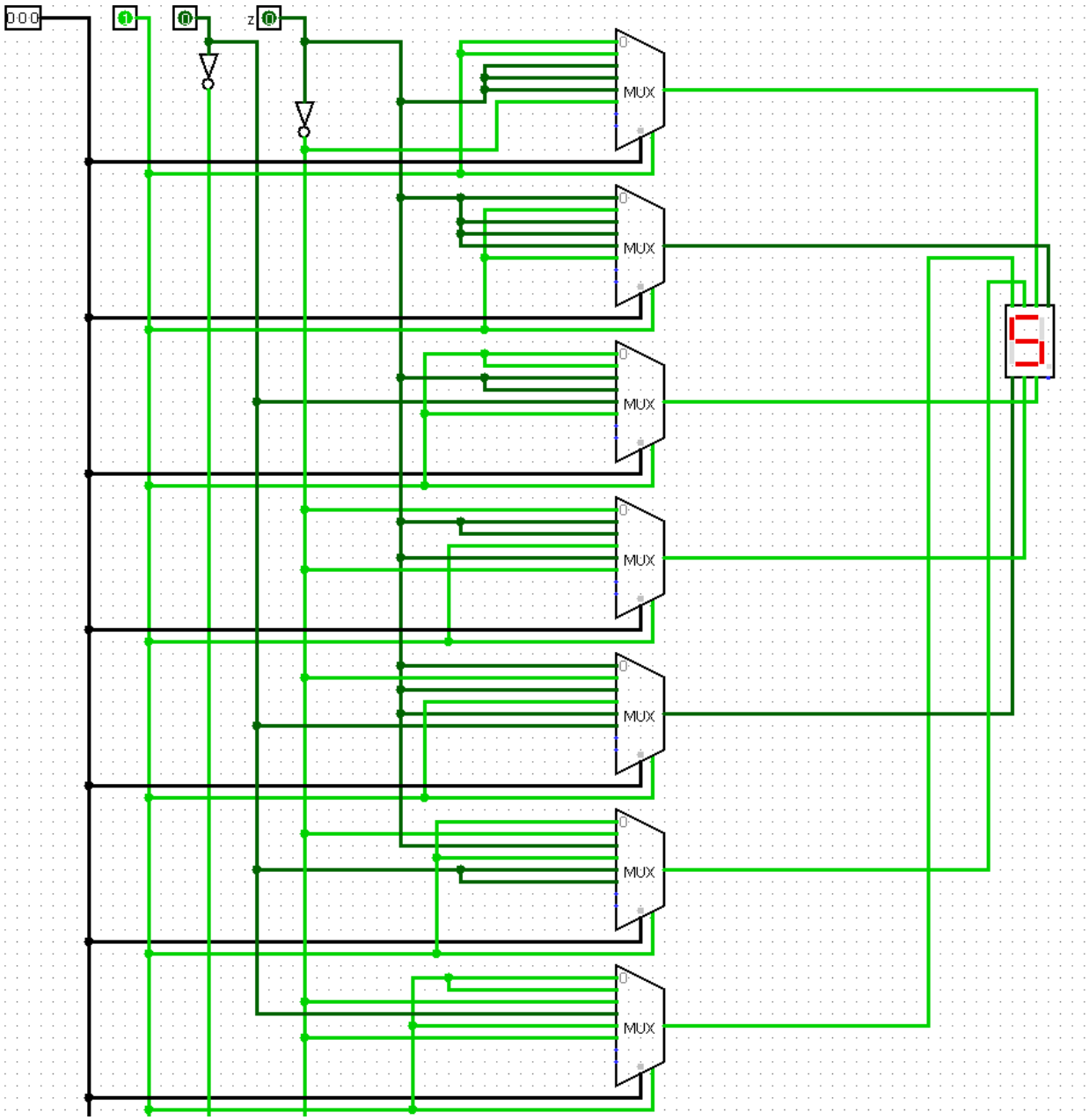
Circuit Diagram for SOP:



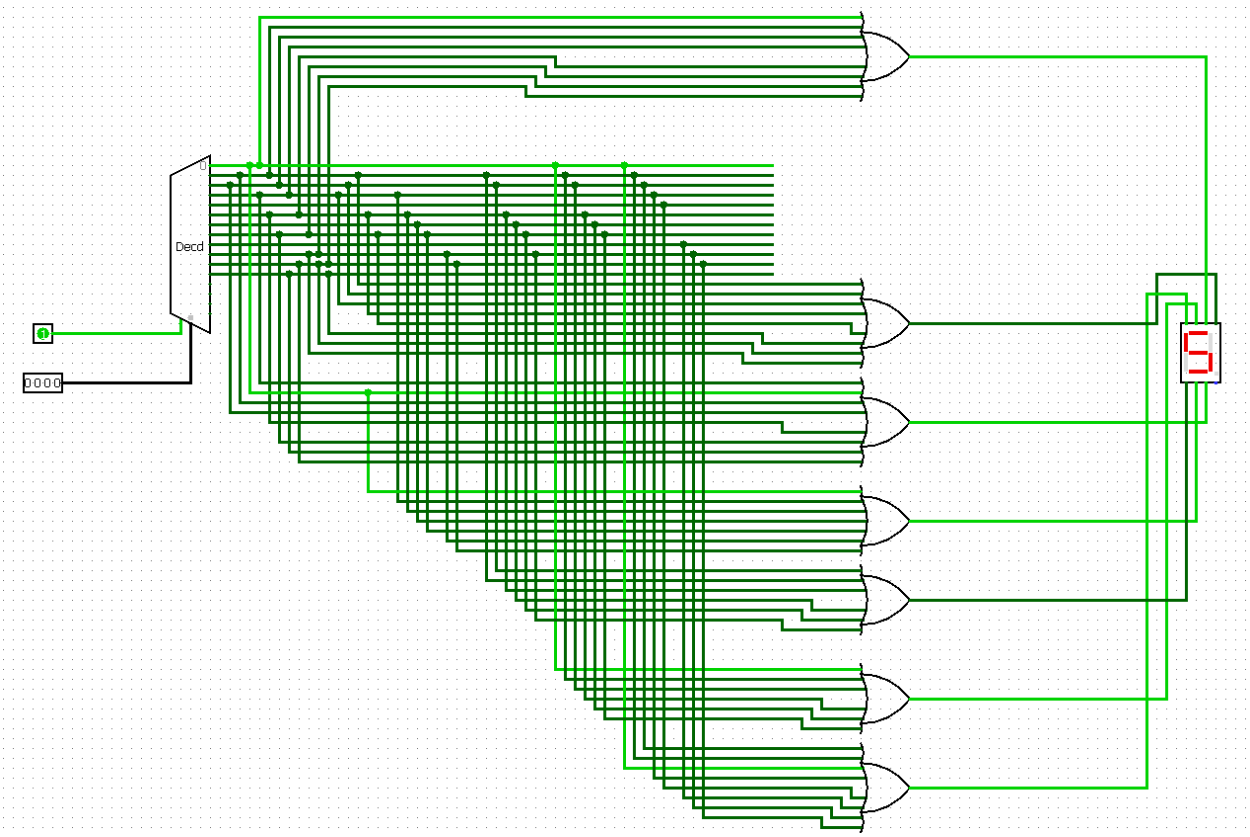
Circuit Diagram for POS:



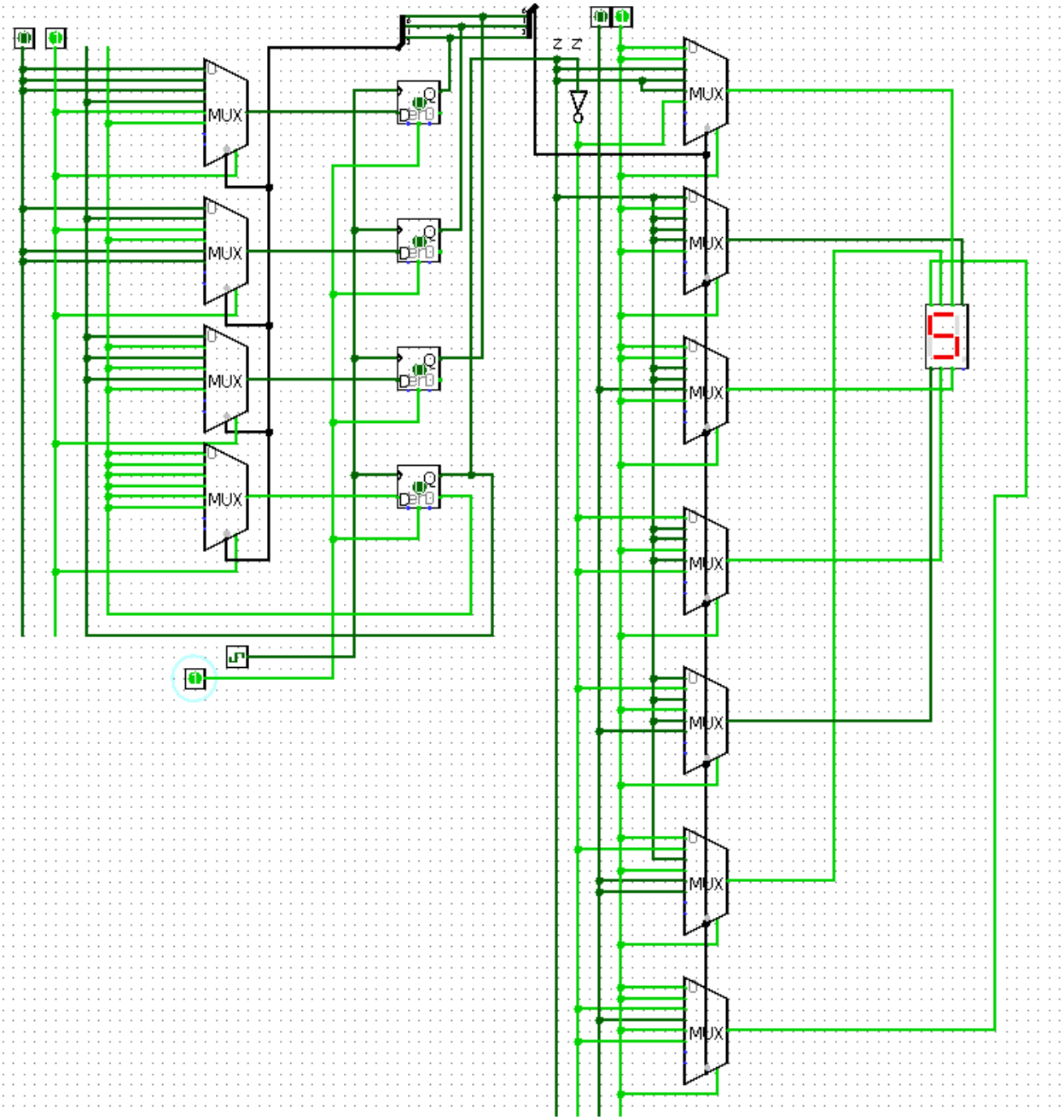
Circuit Diagram for MUX:



Circuit Diagram for Decoder:



Circuit Diagram for Sequential Circuit:



Difficulties Faced:

We faced lot of difficulties while implementing the circuit. Some of the IC's not work properly, so we had to check every IC before using it in our circuit. We also lost track of the pin numbers in the middle of this project work. As there is a lot of wiring it was very difficult for us to identify the problem after setting up the circuit. So we had to recheck the connections of each time. Moreover, our seven segment display got damaged in the beginning because of the high input voltage, so next we are using 9V to 5V converter.

Future Recommendation:

We could make the circuit more compact & user friendly by using less IC & wires. We could use wires of different sizes so that the wires didn't stick out the breadboard. That way it would be easier to understand & explain the working of the circuit.

Conclusion

In this project we showed the sequence "SAA3-DLD-231" in a seven segment display. Our project contained a combinational & sequential part. In the combinational part we used 8:1 MUX to implement our circuit, it is easily understandable & economically beneficial. The output of the MUX is routed into the input pins of the seven segment display to get the sequence. In the sequential part we used D Flip-flop to implement our circuit. The first 3 output of our Flip-flop (MSB) acts as the selector bits of the MUX's whereas the LSB acts as the MUX's. Application of each clock pulse ensures transitions of state in a correct order and therefore we get the desired sequence without any external input. We built a timer to produce the clock pulse on an internal of 2-3 sec approximately. We used a 9V battery and converted it to 5V using a voltage converter as the seven segment will get damaged upon application of high input voltage.

References

1. https://www.youtube.com/watch?v=EGmreVQ-yNM&feature=share&fbclid=IwAR3CczKa_SGtNq3izTOLmFY4uOiMd33SwxNjS-vufOsO2JFqkNwBo1NgWBw
2. https://www.youtube.com/watch?v=OYtGr2faYpw&feature=share&fbclid=IwAR1lMMJkwTtqbHwKLsklVfGiGIWL4hTXrRnqWtHVfSxEdZLZq68LO8-p6YI
3. https://www.youtube.com/watch?v=q0aVOIehsQ4&feature=share&fbclid=IwAR1EXHA2u_qrTLCN96f0MT4aQ0Rm2FFboZw8DYlpyvktd8eJark9jmLB_gs