



**NORTH SOUTH UNIVERSITY**  
**Department of Electrical and Computer**  
**Engineering**

**Digital Logic Design (CSE 231)**

Faculty – Dr Mohammad Monirujjaman Khan(KMM)

Section: 06

Group: 04

**Final Project**

<b>Name</b>	<b>ID</b>
Towsif Muhtadi Khan	1911576642
Khalid Bin Shafiq	1911342642
Rafidul Islam	1912152642
Rashiqur Rahman Rifat	1911445642

## CONTRIBUTION

Work done By	Topic
<b>Towsif Muhtadi Khan</b> <b>(Coordinator)</b>	1.BCD to 7 Segment decoder 2. Truth Table ( <b>CSE-231</b> )
<b>Rafidul Islam</b>	1..K Map ( <b>Using SOP</b> ) 2.Combination Circuit ( <b>SOP</b> )
<b>Rasiquir Rahman Rifat</b>	1..K Map ( <b>Using POS</b> ) 2.Combinationa Circuit( <b>POS</b> )
<b>Khalid Bin Shafiq</b>	1.Circuit diagram ( <b>NAND Gate</b> )
<b>Towsif Muhtadi Khan</b>	1.Circuit diagram ( <b>NOR Gate</b> )
<b>Rasiquir Rahman Rifat</b>	1.Sequential Circuit
<b>Towsif Muhtadi khan</b>	1.State Diagram 2.State table
<b>Rafidel Islam</b>	1.Kamp
<b>Towsif Muhtadi Khan</b> <b>Khalid Bin Shafiq</b> <b>Rafidul Islam</b> <b>Rashiquir Rahman Rifat</b>	Sequential Circuit
<b>Khalid Bin Shafiq</b>	555 Timer

## BCD to 7 Segment Decoder:

A Digital Decoder IC, is a device which converts one digital format into another and one of the most commonly used devices for doing this is called the Binary Coded Decimal (BCD) to 7-Segment Display Decoder.

We have to show CSE-231 using BCD to 7 segment display. As we need to show 7 words so we took 3 input because  $2^3 = 8$  and through it we can easily show the given word.

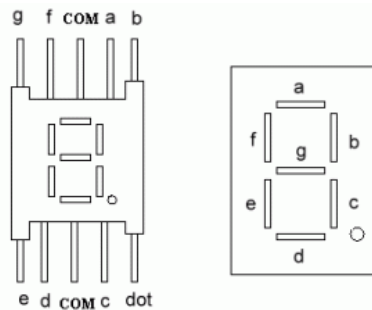


Figure : 7 Segment display

For showing the CSE-231 we need a truth table and set the value of a, b, c, d, e, f and g to show the words properly. So the truth table is given below-

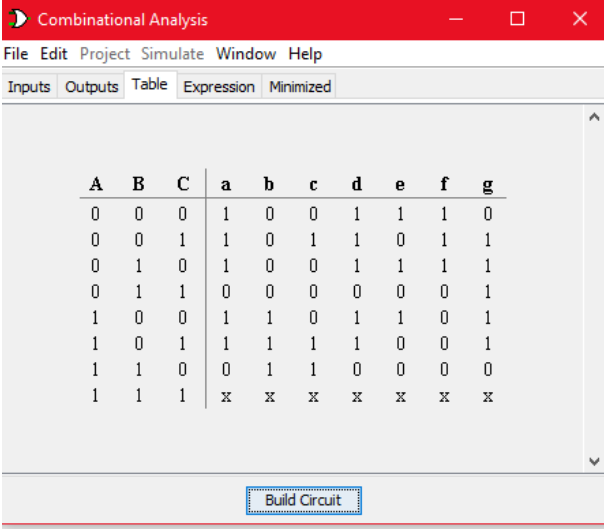
### Truth Table for “CSE-231”:

Truth Table:

word	Input			Output						
	A	B	C	a	b	c	d	e	f	g
C	0	0	0	1	0	0	1	1	1	0
S	0	0	1	1	0	1	1	0	1	1
E	0	1	0	1	0	0	1	1	1	1
-	0	1	1	0	0	0	0	0	0	1
2	1	0	0	1	1	0	1	1	0	1
3	1	0	1	1	1	1	1	0	0	1
1	1	1	0	0	1	1	0	0	0	0
	1	1	1	X	X	X	X	X	X	X

Table : Truth table for  
"CSE-231"

If we give the given input in Logisim the table will be like:



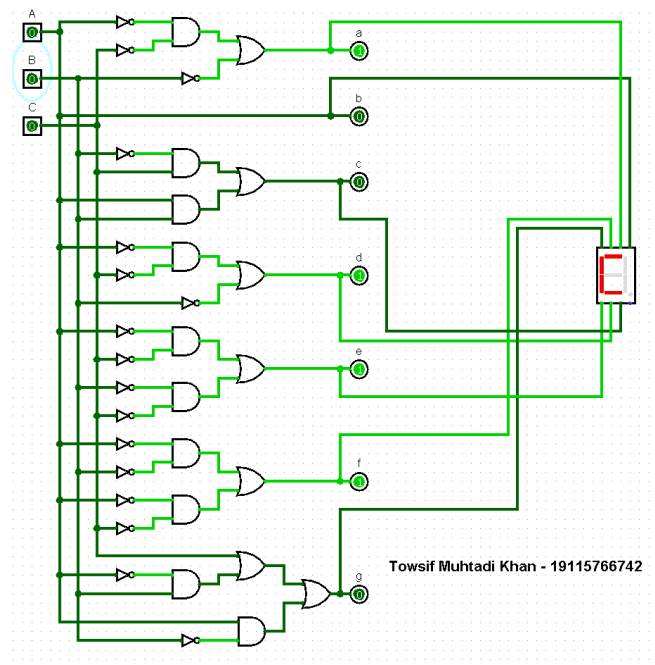
The screenshot shows the 'Combinational Analysis' window in Logisim. It has a menu bar (File, Edit, Project, Simulate, Window, Help) and tabs (Inputs, Outputs, Table, Expression, Minimized). The 'Table' tab is selected, displaying a truth table with 8 rows and 10 columns. The columns are labeled A, B, C, a, b, c, d, e, f, and g. The first three columns (A, B, C) represent the inputs, and the last seven columns (a, b, c, d, e, f, g) represent the outputs. The last row of the table contains 'x' values for the output columns, indicating 'don't care' conditions.

A	B	C	a	b	c	d	e	f	g
0	0	0	1	0	0	1	1	1	0
0	0	1	1	0	1	1	0	1	1
0	1	0	1	0	0	1	1	1	1
0	1	1	0	0	0	0	0	0	1
1	0	0	1	1	0	1	1	0	1
1	0	1	1	1	1	1	0	0	1
1	1	0	0	1	1	0	0	0	0
1	1	1	x	x	x	x	x	x	x

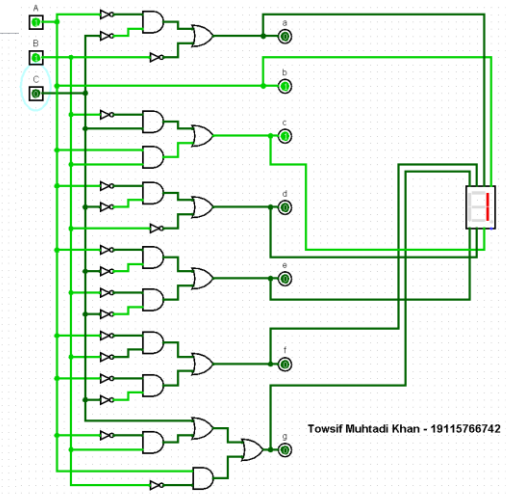
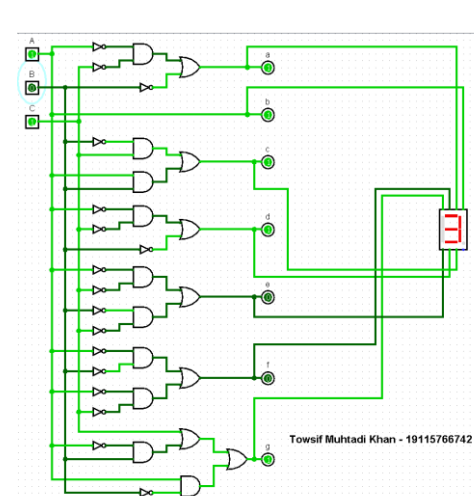
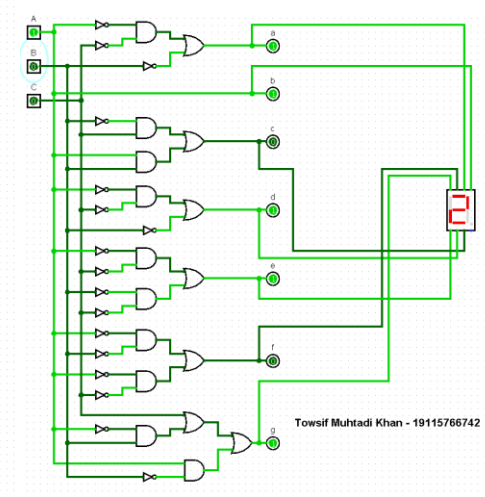
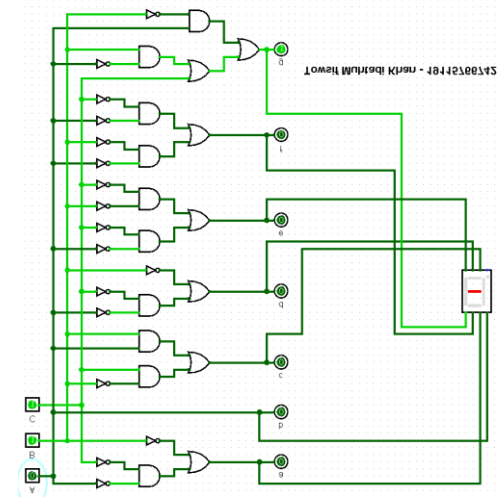
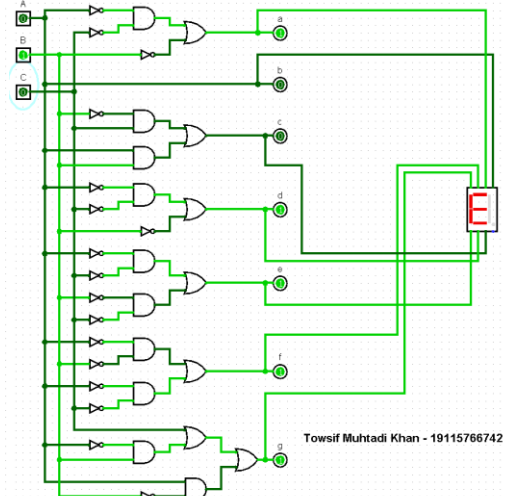
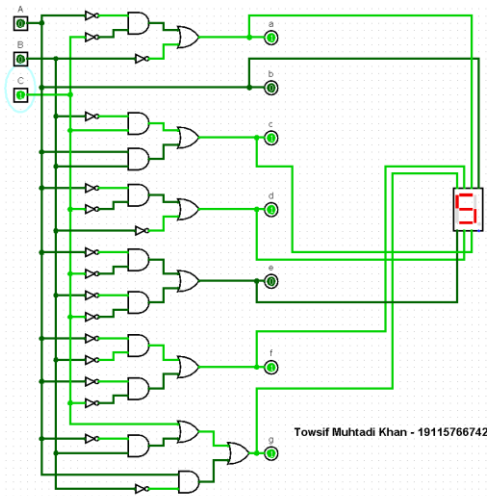
At the bottom of the window, there is a 'Build Circuit' button.

And if we implement the table and build the circuit using the logisim, we get the following output

For the input ABC=000, got the expected output a=1, b=0, c=0, d=1, e=1, f=1, g=0 and 'C' written in the 7-segment display.



If we change the value for inputs (A,B,C ) then we get the following output



So we can see that we are getting our expected output properly.

## KMap Using SOP:

### SOP:

Sum of Product is the abbreviated form of SOP. Sum of product form is a form of expression in Boolean algebra in which different product terms of inputs are being summed together. We know that in SOP, it evaluates with 1 for each combination of x and y. '1' means the variable is 'Not complemented' and '0' means the variable is 'Complemented'. Following the method of SOP form we drew the kmaps for our project and got the following equations:

The 3-variable Karnaugh map is an array of eight cells. For our project Kmap Using SOP is given below:

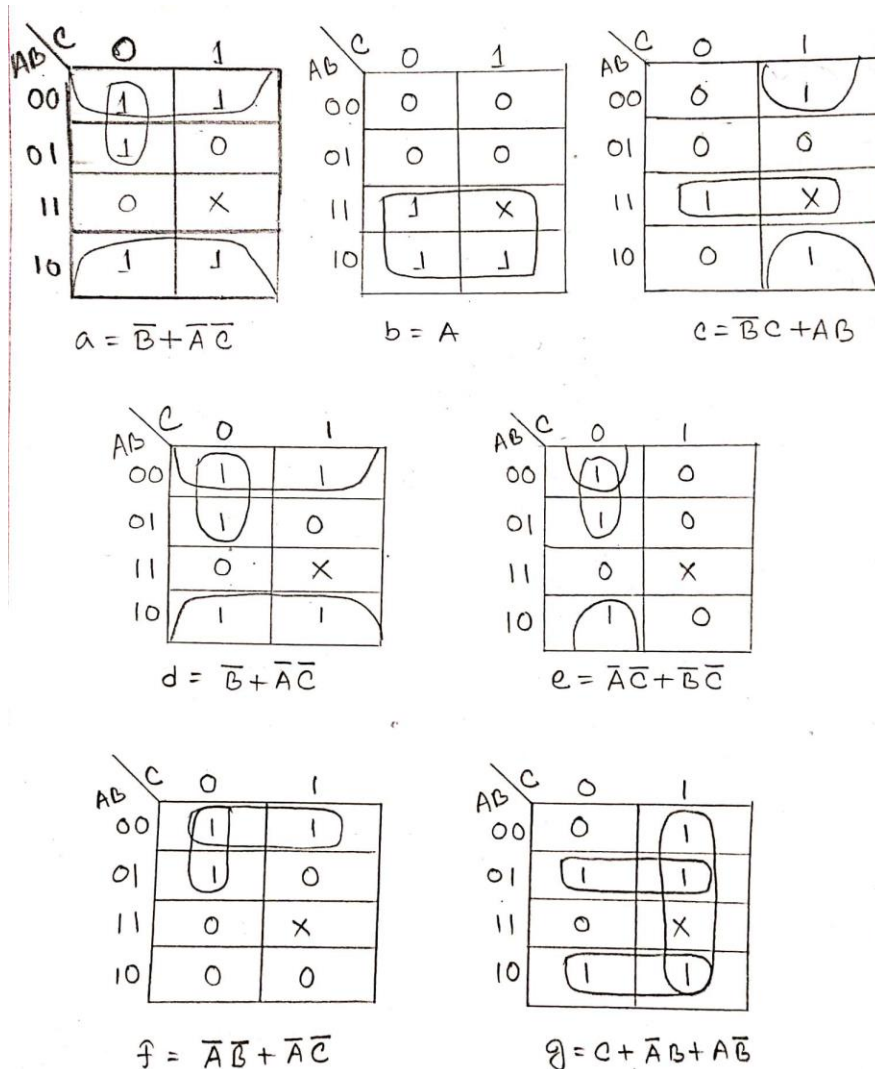


Figure: KMap Using SOP

## Circuit diagram (SOP)

Then we drew the combinational circuit using the equations we got from the kmap:

The equations are :  $a = \bar{B} + \bar{A}\bar{C}$ ,  $b = A$ ,  $c = \bar{B}C + AB$ ,  
 $d = \bar{B} + \bar{A}\bar{C}$ ,  $e = \bar{A}\bar{C} + \bar{B}\bar{C}$ ,  $f = \bar{A}\bar{B} + \bar{A}\bar{C}$ ,  $g = C + \bar{A}B + A\bar{B}$

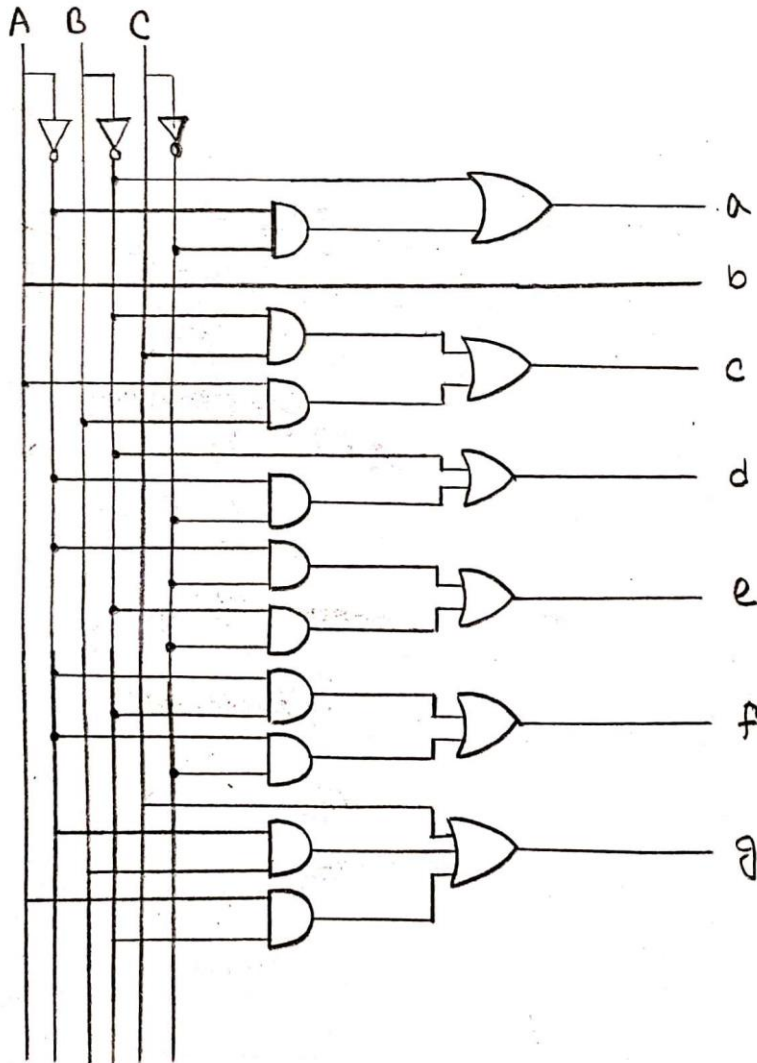
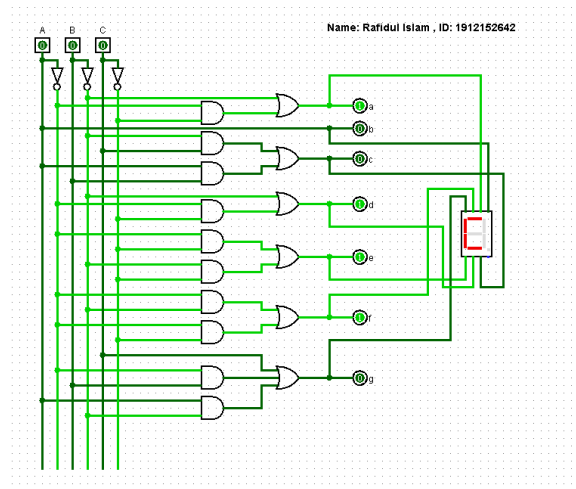


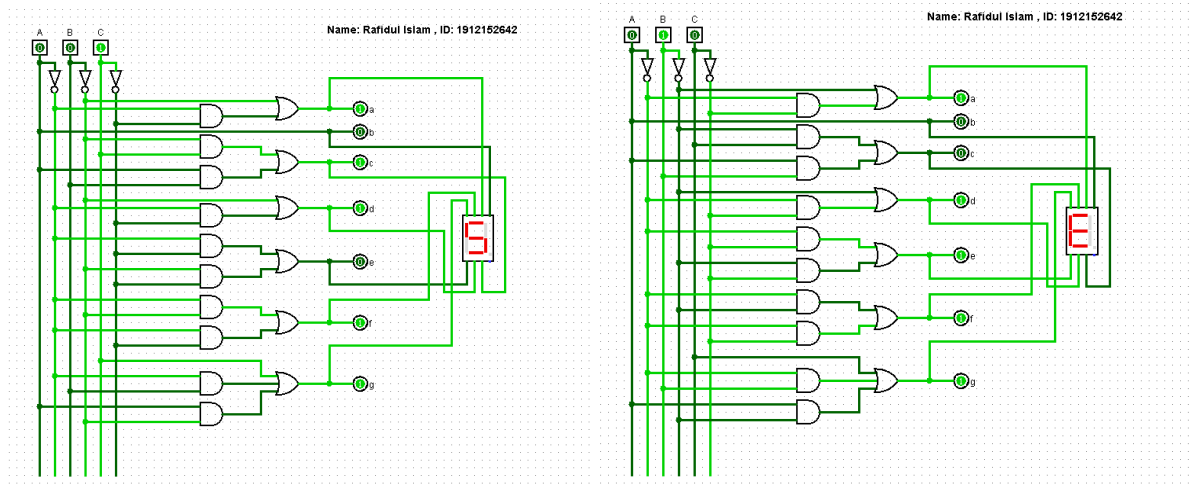
Figure : Circuit Diagram (SOP)

After building the combinational circuit in logisim using SOP we added the seven segment display and after that when we give the inputs we got our expected values:

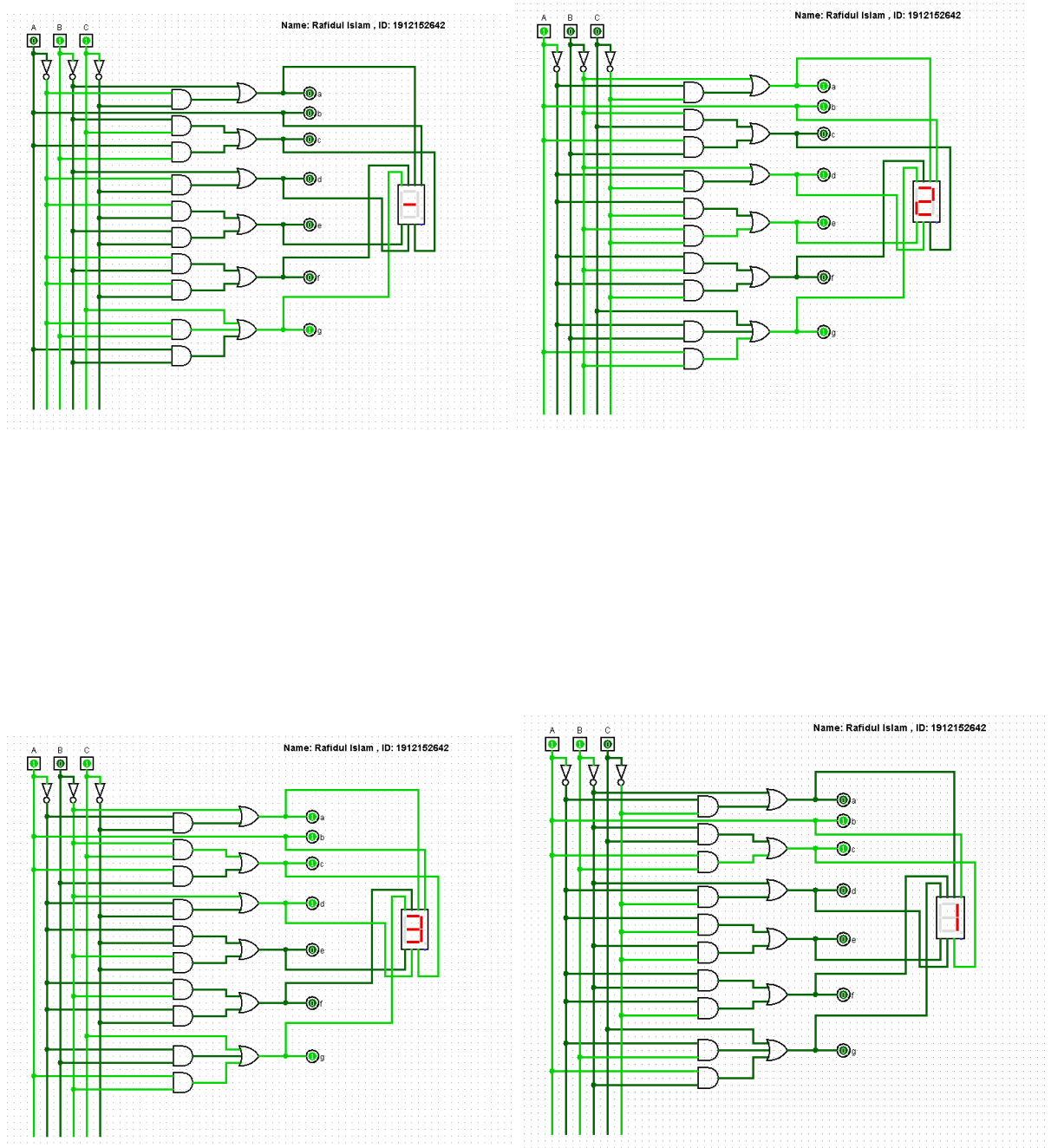
For the input ABC=000, got the expected output a=1, b=0, c=0, d=1, e=1, f=1, g=0 and 'C' written in the 7-segment display.



Then we change the values of the input and got our expected results:





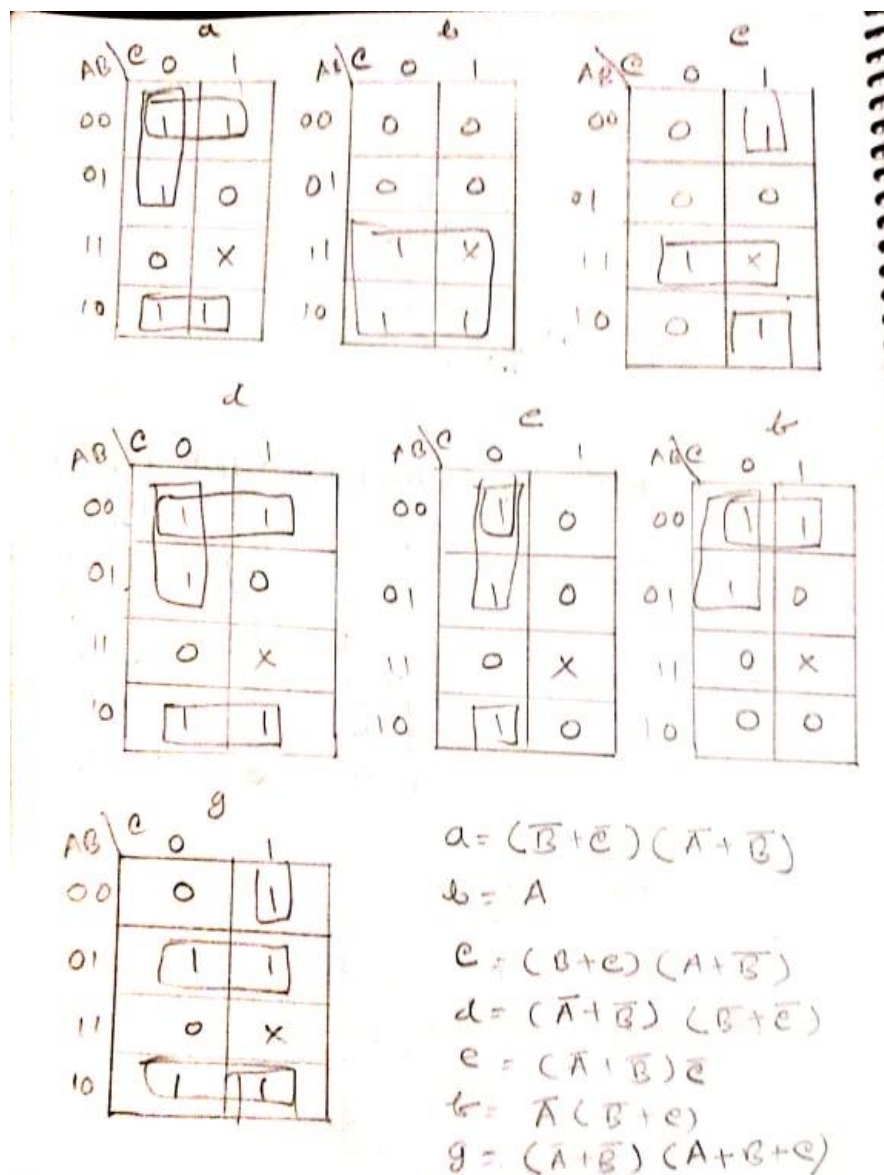


**Figure:** Circuit diagram (SOP)

## KMap Using POS:

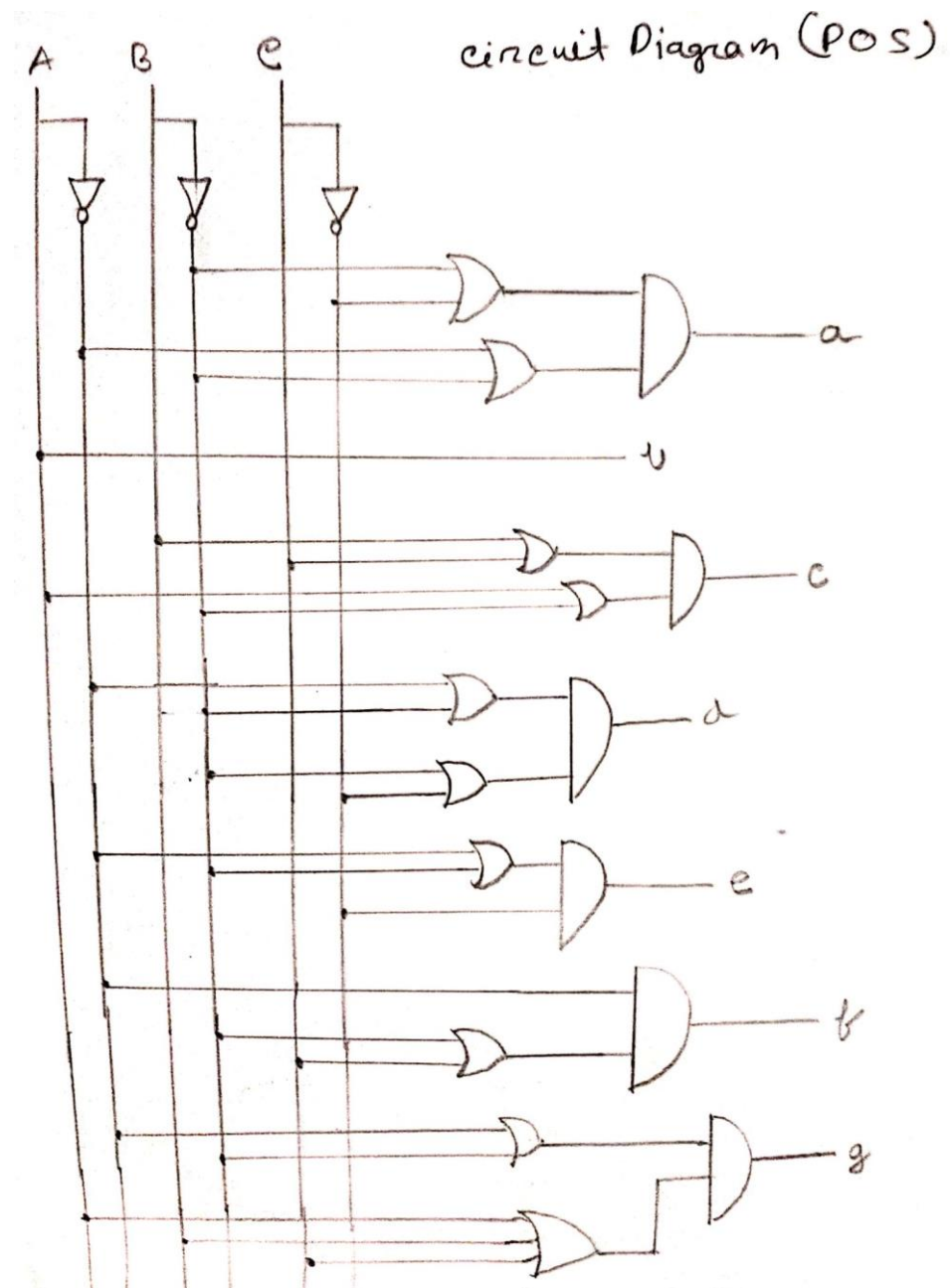
The product of sums form is a method (or form) of simplifying the Boolean expressions of logic gates. All these sum terms are multiplied together to get the product-of-sum form. This form is exactly opposite to the SOP form. Here the sum terms are defined by using the OR operation and the product term is defined by using AND operation. When two or more sum terms are multiplied by a Boolean OR operation, the resultant output expression will be in the form of product-of-sums form or POS form.

The 3-variable Karnaugh map is an array of eight cells. For our project Kmap Using POS is given below:



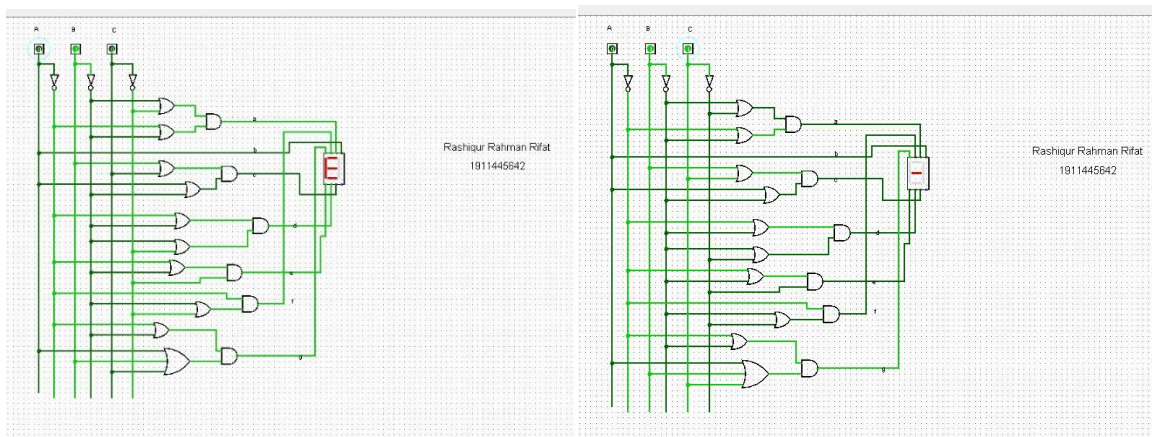
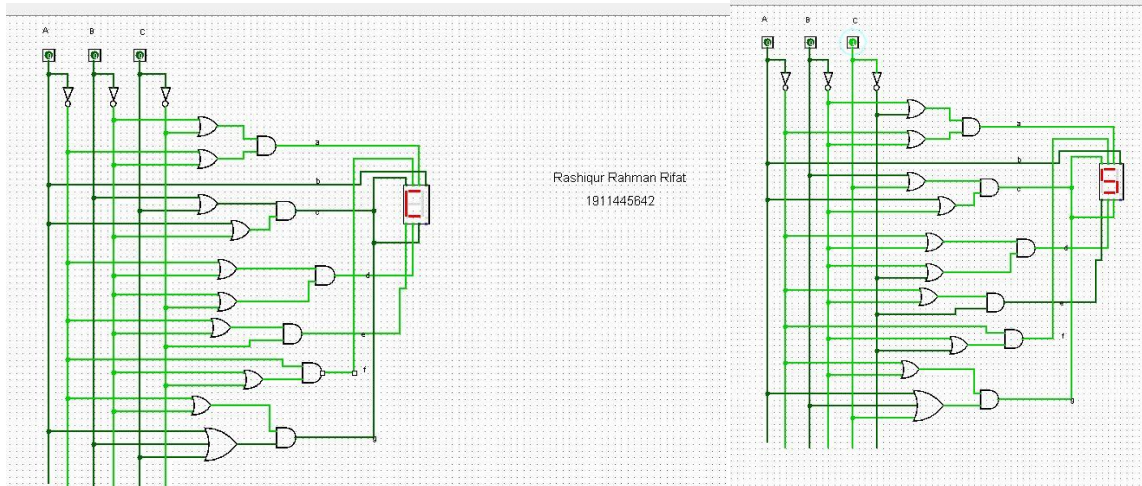
**Figure: KMap Using POS**

# Circuit diagram (POS)

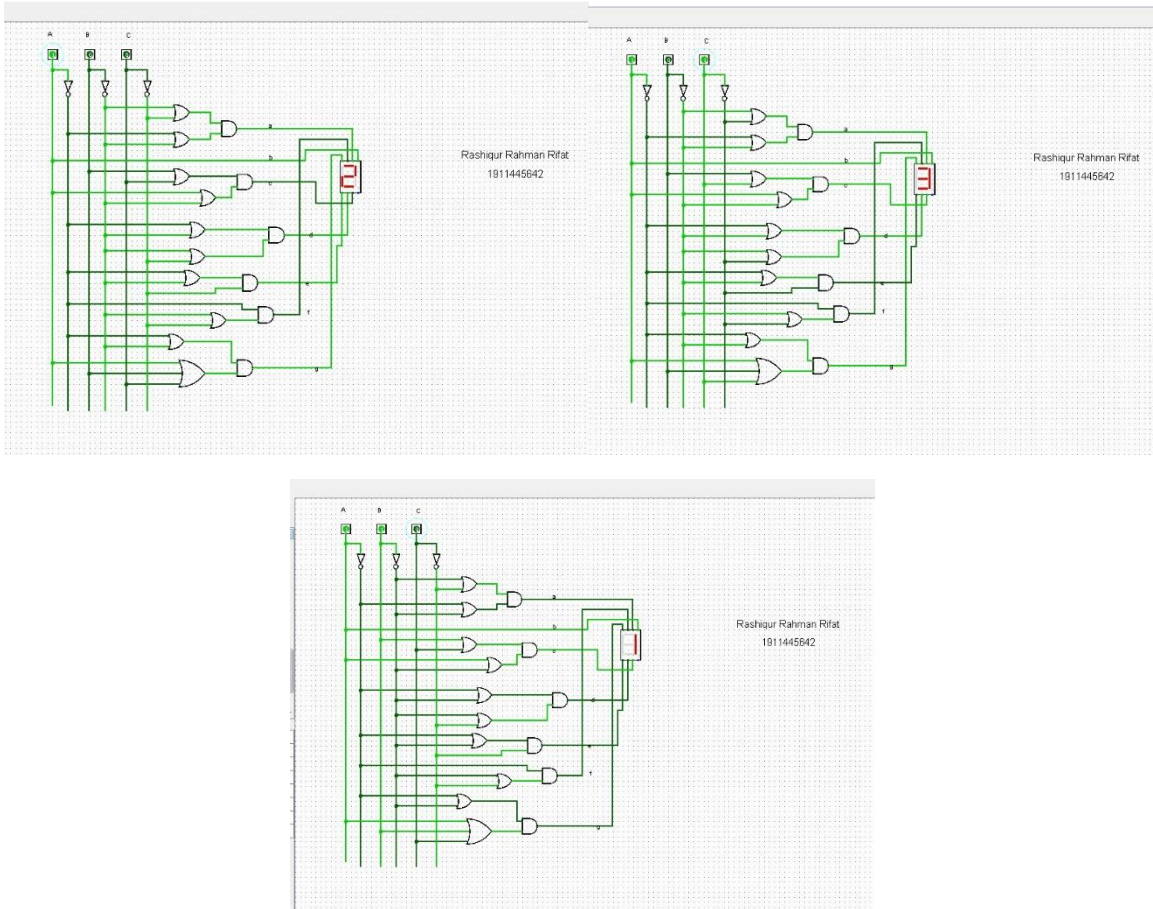


**Figure:** Circuit diagram (POS)

If we draw the circuit using the equation which we get by solving the kmap the circuit will be like this:







**Figure: Circuit diagram (POS)**

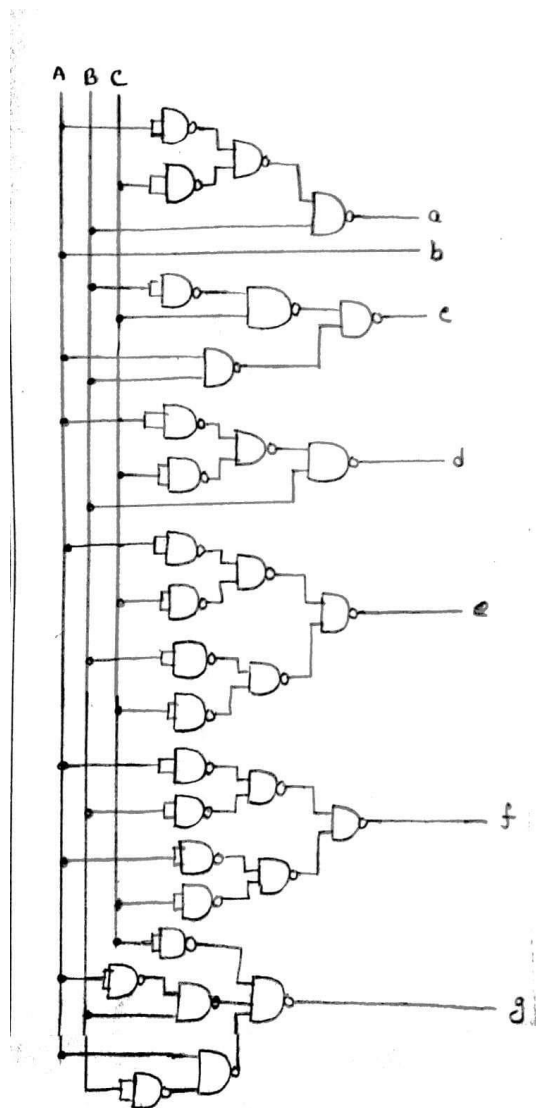
So we can see that we are getting our expected output properly.

## Circuit diagram (NAND Gate)

### Nand gate:

An operator which gives the value zero if and only if all the operands have a value of one, and otherwise has a value of one (equivalent to NOT AND).

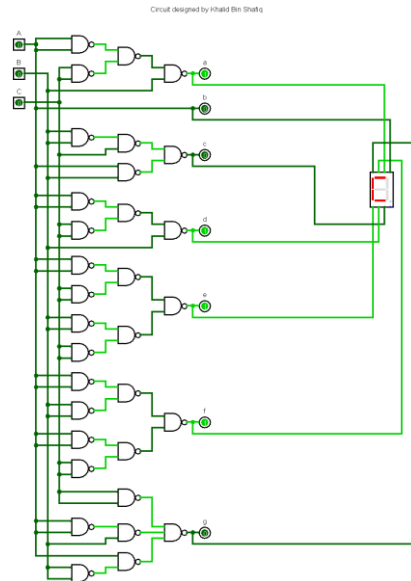
For using NAND gate, we replaced all the basic gates with NAND gates. We constructed NOT gate, OR gate and AND gates using NAND gates. And after that we got the following circuit:



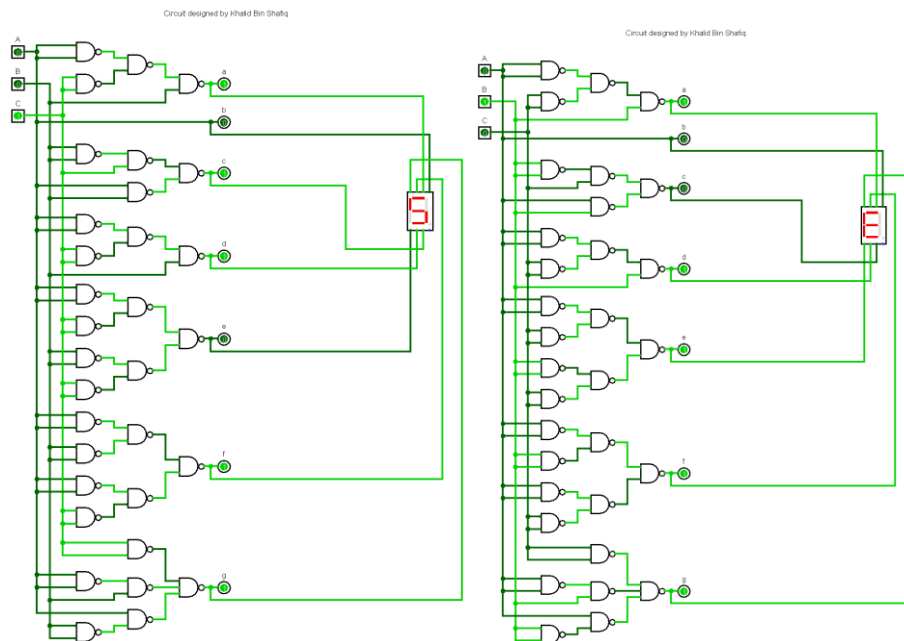
**Figure :** Circuit Diagram (Using Nand Gates)

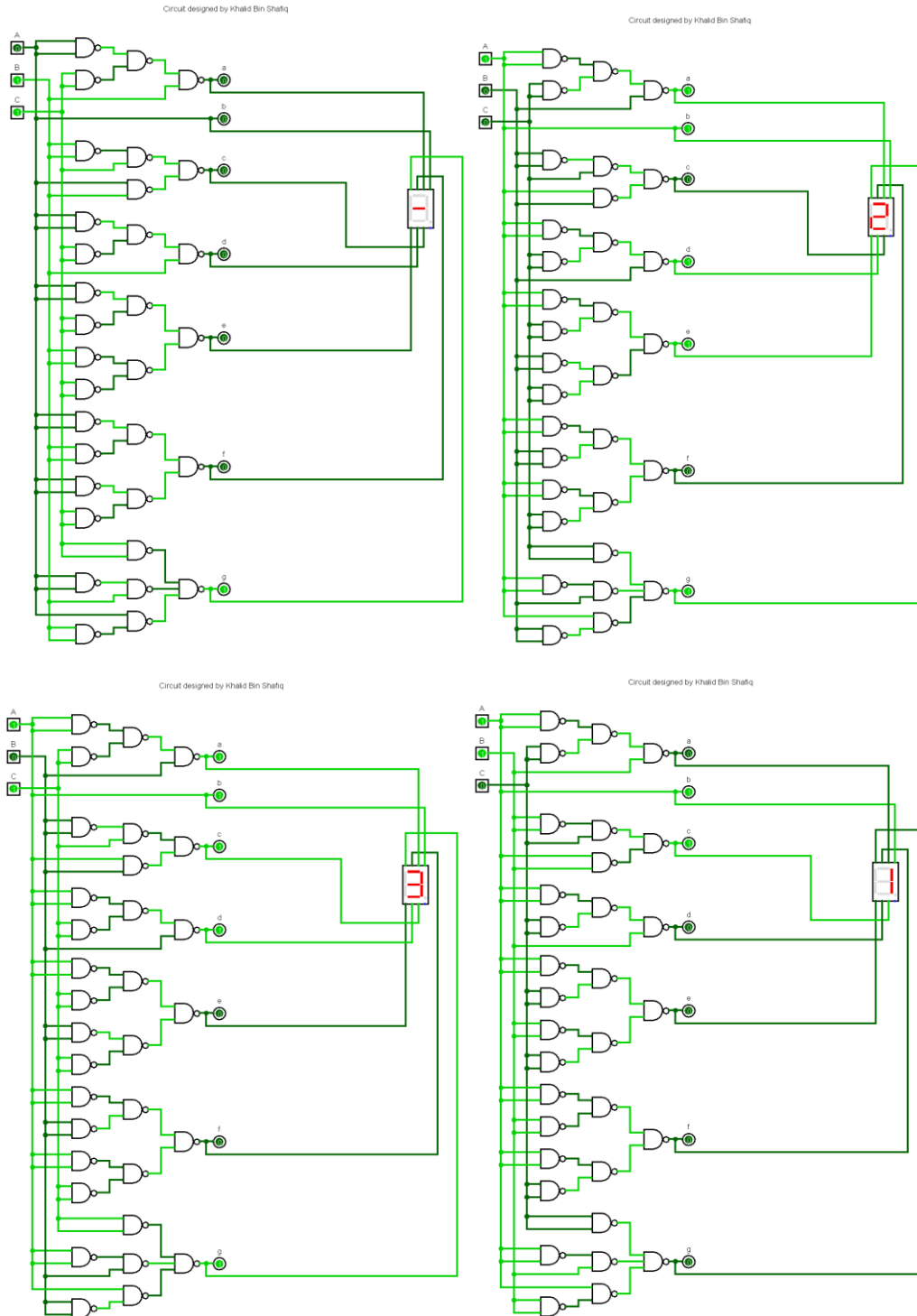
After building the combinational circuit using Nand gate we added the seven segment display and after that when we give the inputs we got our expected values:

For the input ABC=000, got the expected output a=1, b=0, c=0, d=1, e=1, f=1, g=0 and 'C' written in the 7-segment display.



Then we change the values of the input and got our expected results:





**Figure: Circuit diagram (NAND GATE)**

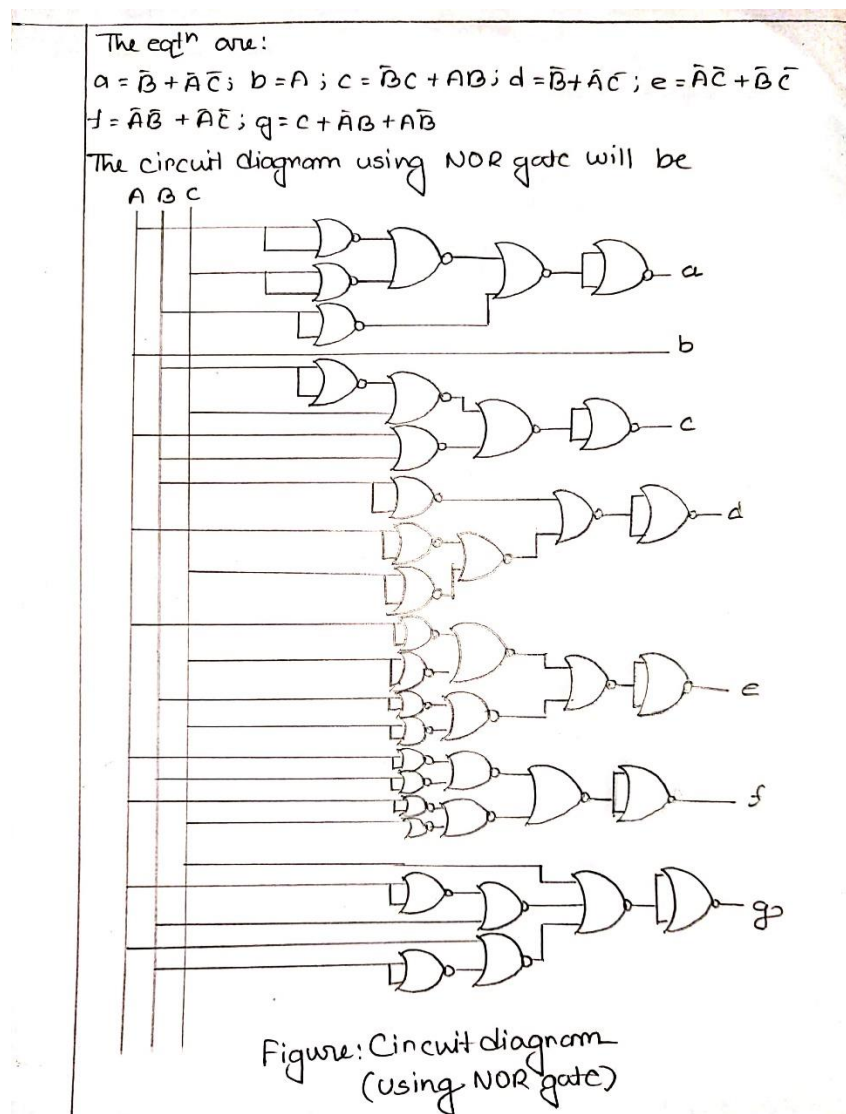


## Circuit diagram (NOR Gate)

### Nor Gate:

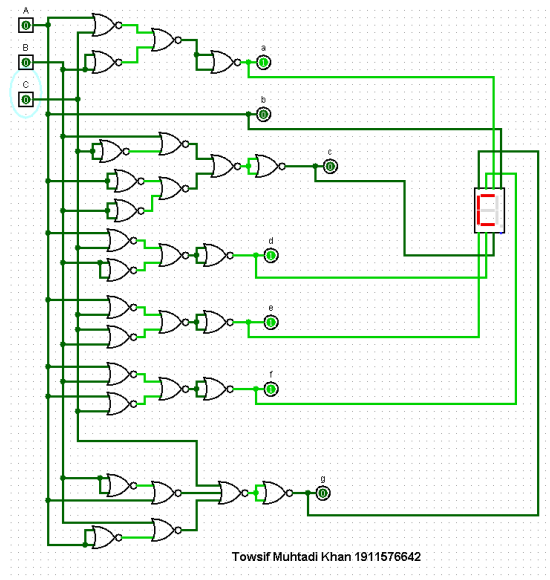
A Boolean operator which gives the value one if and only if all operands have a value of zero and otherwise has a value of zero.

For using NOR gate, we replaced all the basic gates with NOR gates. We constructed NOT gate, OR gate and AND gates using NOR gates. And after that we got the following circuit:

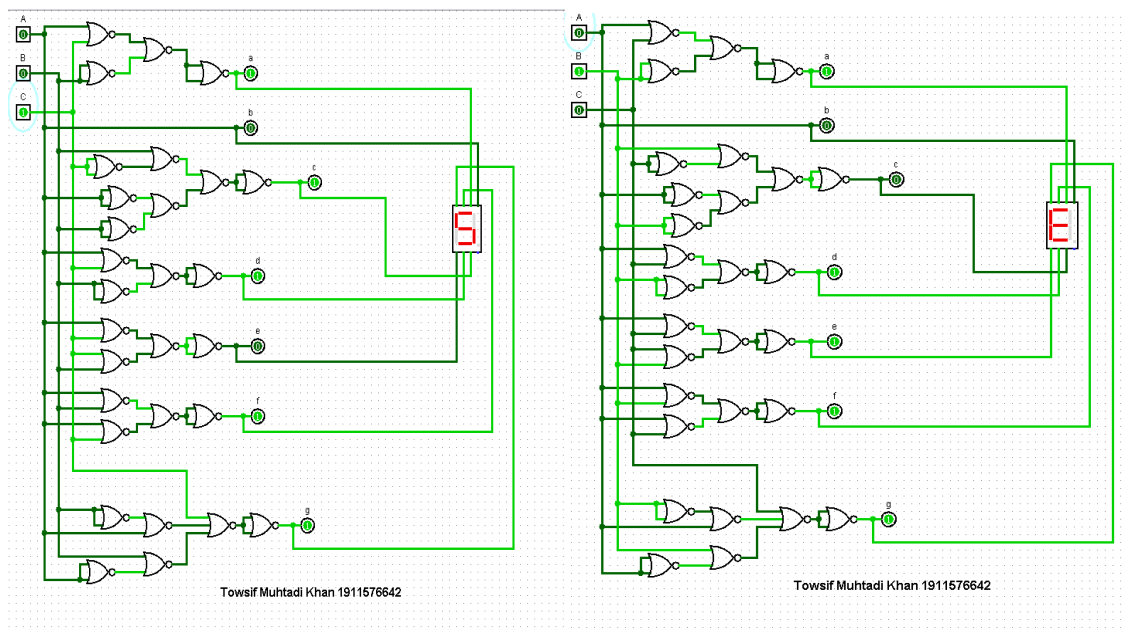


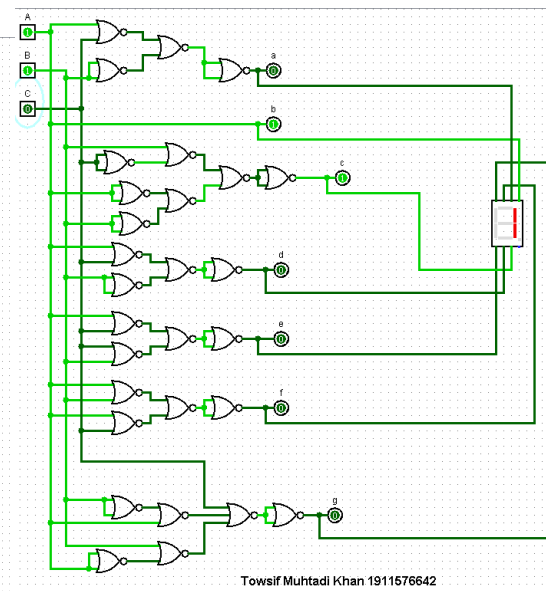
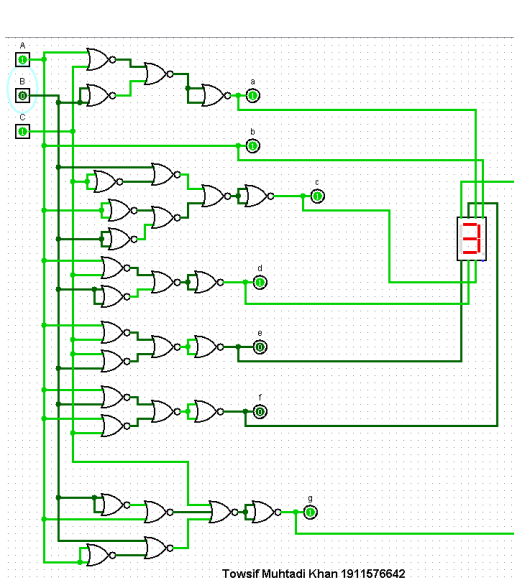
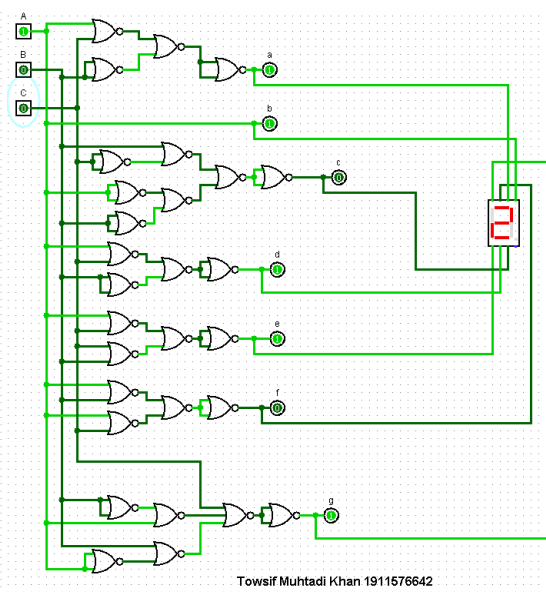
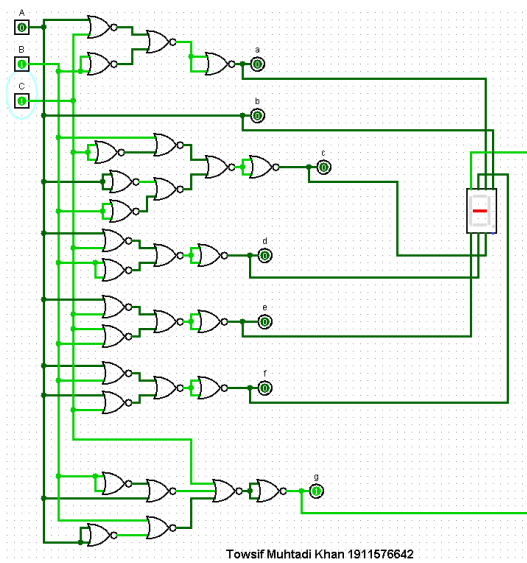
**Figure: Circuit Diagram (Using NOR Gates)**

For the input ABC=000, got the expected output a=1, b=0, c=0, d=1, e=1, f=1, g=0 and 'C' written in the 7-segment display.



Then we change the values of the input and got our expected results:





**Figure:** Circuit Diagram (Using NOR Gates)

## Sequential Circuits:

Sequential Circuit is made of combinational circuits and memory storage where circuits conduct the operation of present output based on present inputs and past outputs stored in memory storage. In a sequential circuit, the values of the outputs depend on the past behavior of the circuit, as well as the present values of its inputs. A sequential circuit has states, which in conjunction with the present values of inputs determine its behavior. Sequential circuits can be Synchronous where flip-flops are used to implement the states, and a clock signal is used to control the operation.

### JK Flip flop:

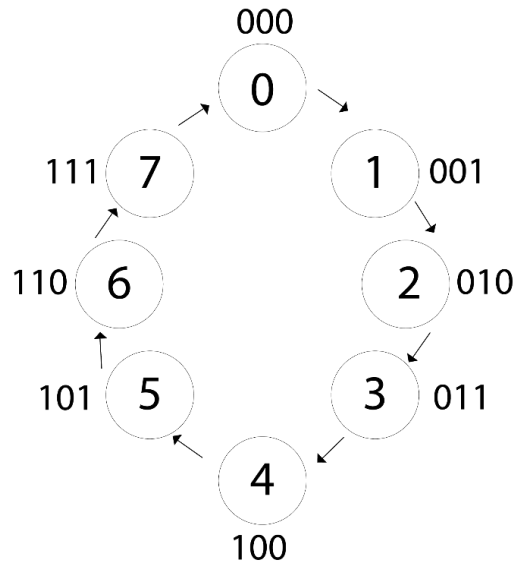
A JK flip flop is basically a gated SR flip-flop with the addition of a clock input circuitry that prevents the illegal or invalid output condition that can occur when both inputs S and R are equal to logic level “1”.

Q	Q <sub>next</sub>	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

*JK flip-flop: Excitation Table*

## The state diagram for our project (CSE-231) is:

For our project we design the following state diagram:



Using the state diagram as well as the Excitation Table of JK flip flop we get the following state table for our project:

### State Table:

Present state Input			Next state Output			Flip-flop input functions					
A	B	C	A	B	C	J <sub>A</sub>	K <sub>A</sub>	J <sub>B</sub>	K <sub>B</sub>	J <sub>C</sub>	K <sub>C</sub>
0	0	0	0	0	1	0	X	0	X	1	X
0	0	1	0	1	0	0	X	1	X	X	1
0	1	0	0	1	1	0	X	X	0	1	X
0	1	1	1	0	0	1	X	X	1	X	1
1	0	0	1	0	1	X	0	0	X	1	X
1	0	1	1	1	0	X	0	1	X	X	1
1	1	0	1	1	1	X	0	X	0	1	X
1	1	1	0	0	0	X	1	X	1	X	1

## Kmap:

AB \ C	0	1
00	0	0
01	0	1
11	x	x
10	x	x

$$J_A = BC$$

AB \ C	0	1
00	x	x
01	x	x
11	0	1
10	0	0

$$K_A = BC$$

AB \ C	0	1
00	0	1
01	x	x
11	x	x
10	0	1

$$J_B = C$$

AB \ C	0	1
00	x	x
01	0	1
11	0	1
10	x	x

$$K_B = C$$

AB \ C	0	1
00	1	x
01	1	x
11	1	x
10	1	x

$$J_C = 1$$

AB \ C	0	1
00	x	1
01	x	1
11	x	1
10	x	1

$$K_C = 1$$

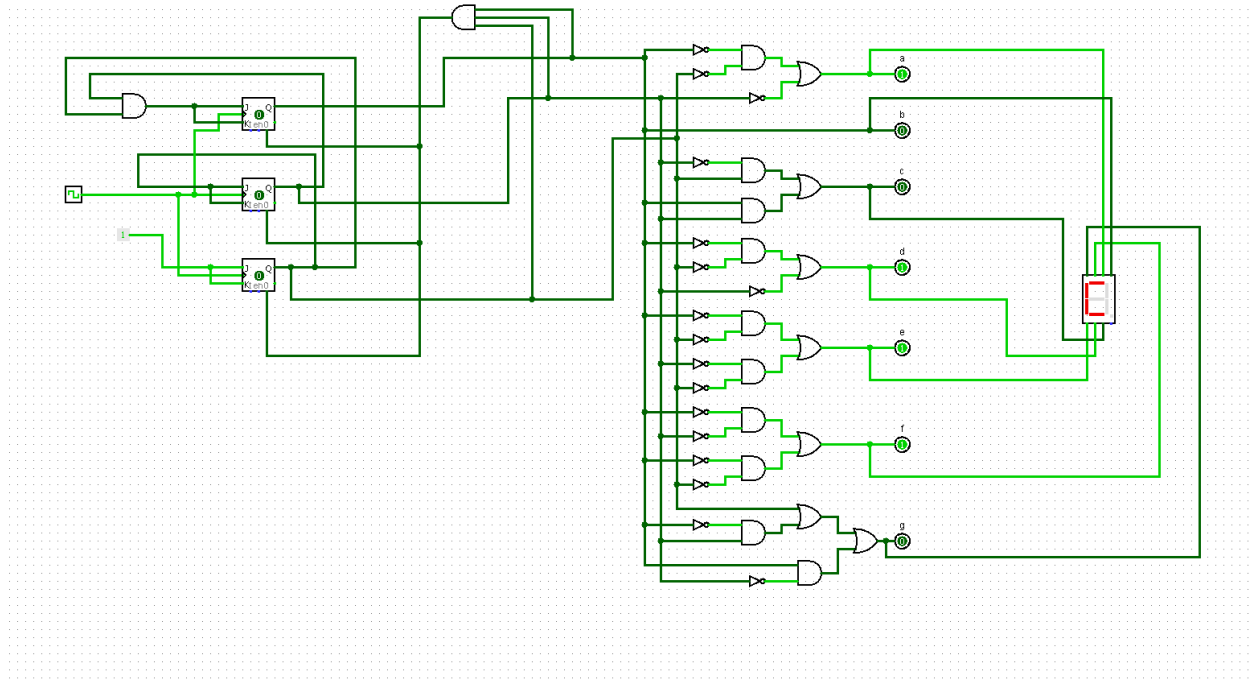
From the state table we got values for  $J_A, K_A, J_B, K_B, J_C, K_C$  and after that we use K map to get the following equations:

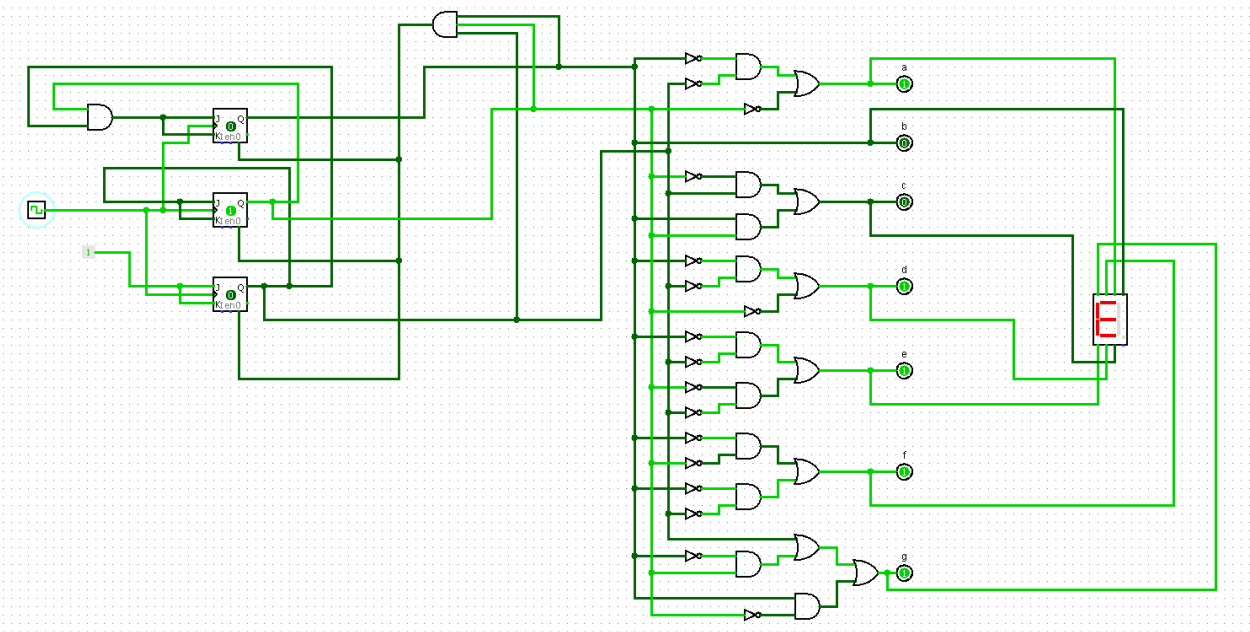
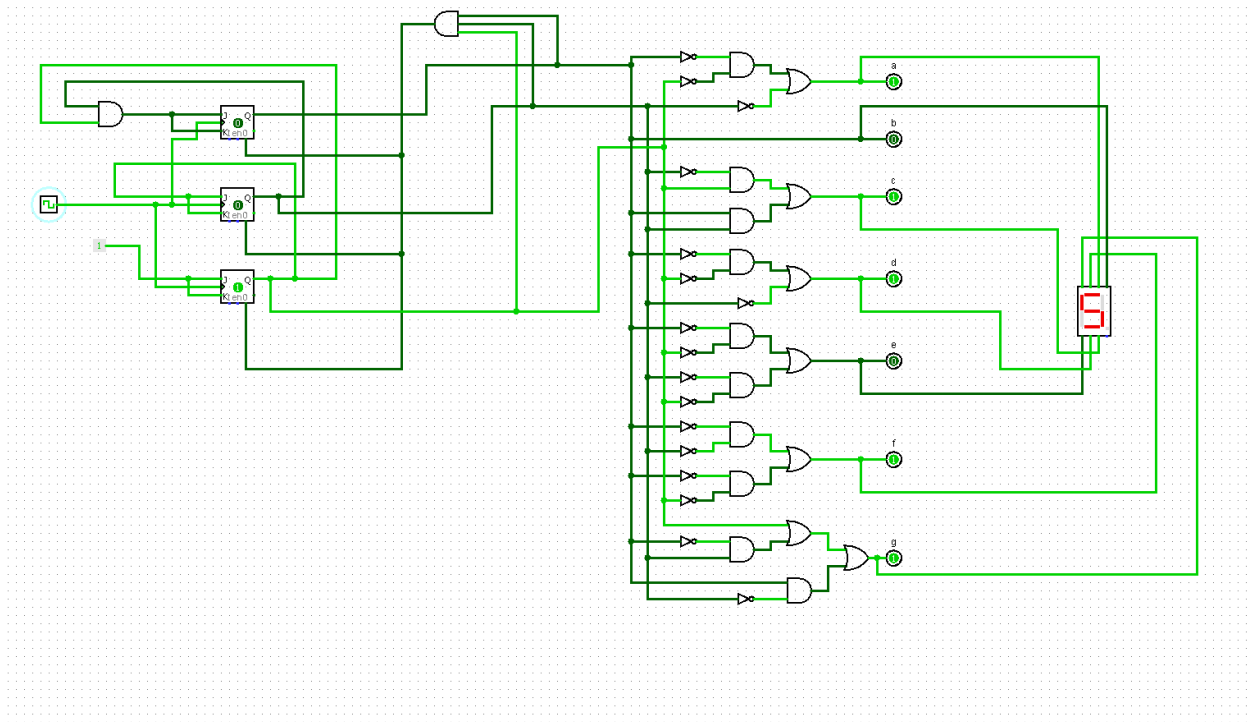
$$J_A = BC \quad K_A = BC \quad J_B = C \quad K_B = C \quad J_C = 1 \quad K_C = 1$$

After this we started our main work basically the sequential part of our project. Using all the information we design the JK flip flop and after that we added the JK flip-flop with our Combinational circuit. But as in the state diagram we took an extra value 111 which is not required for our project. To solve this problem, we clear the value so that for 111 we don't get any kind of unexpected output. As we cannot use 555 timer in the Logisim so we set the frequency and Enabled the "Ticks Enabled" then we can easily see the CSE- 231 in our 7 segment display. Through this process we completed the final part of our project.

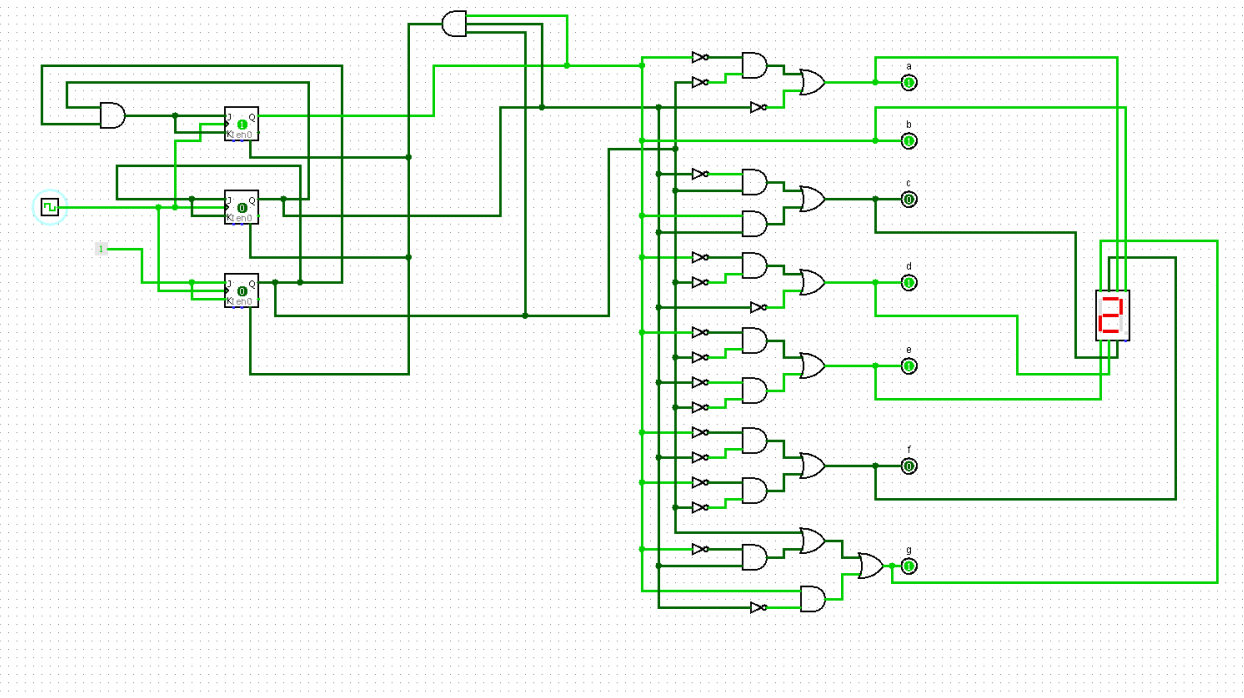
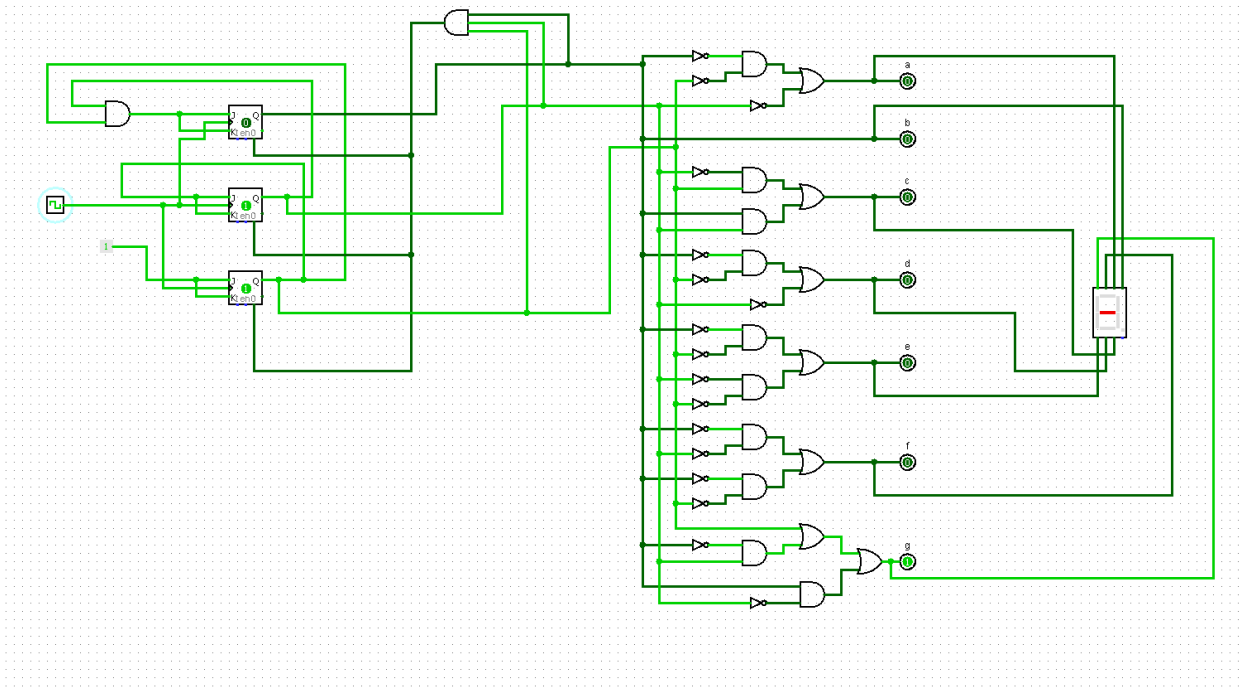
### **Circuit Diagram for Sequential Part:**

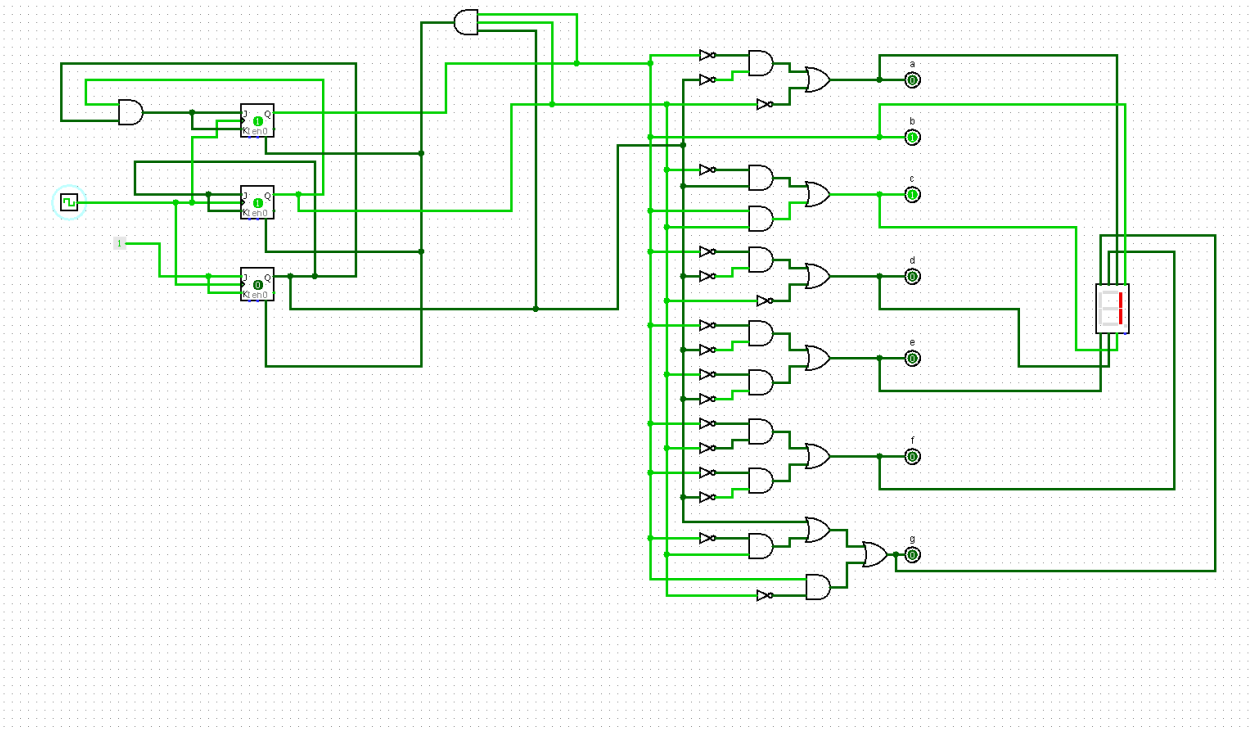
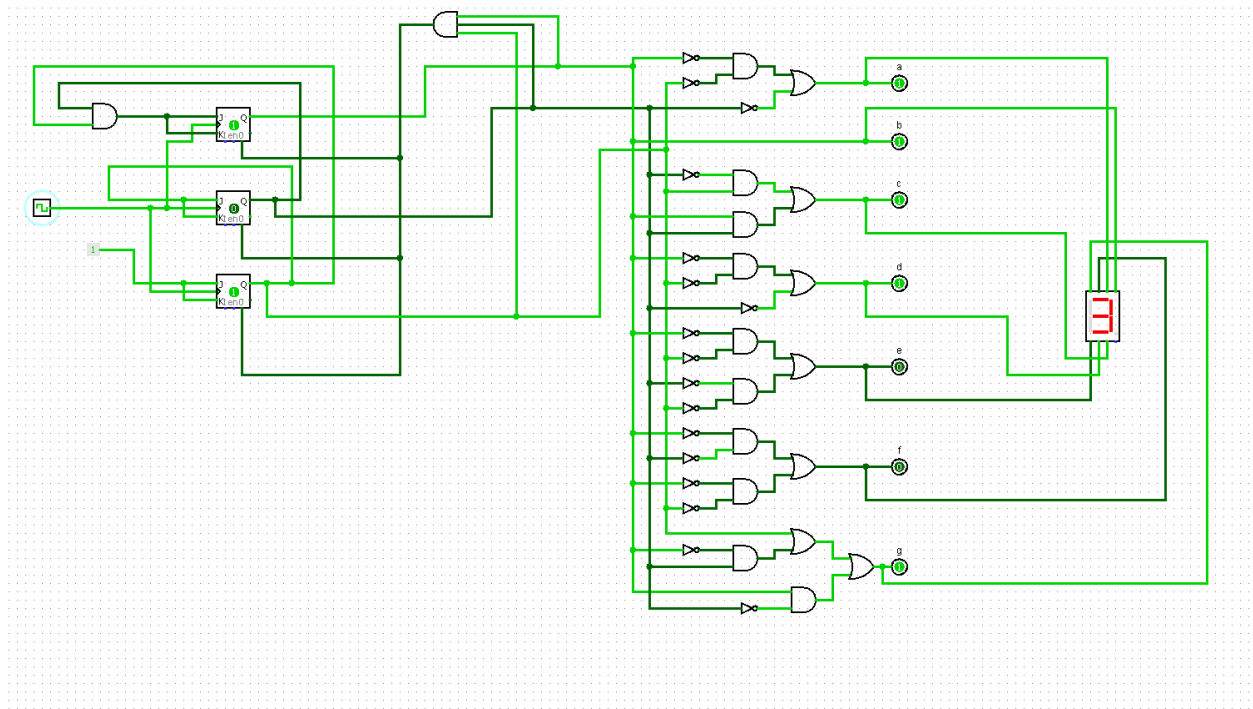
If we give the clock pulse, then we will get the following output:







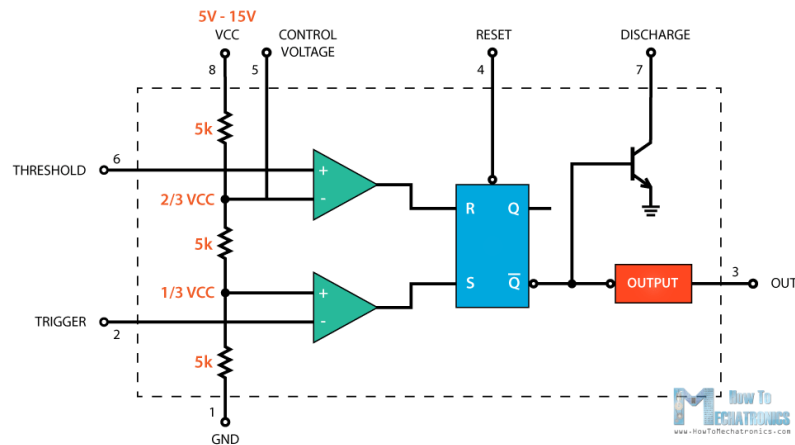




We attached the video part where we Enabled the “Ticks Enable”

## 555 Timer:

555 timers are integrated timing circuits which are used commonly as a source of clock pulses to drive subsequent timer circuits. They are analogue devices which can produce an oscillating and digital output. The IC can be configured to give an astable, period output or a monostable, single triggered output.



**Figure: 555 Timer**

This is the basic structure of a 555 timer represented with a block diagram it consists of 2 comparators, a flip-flop, a voltage divider, a discharge transistor and an output stage

The voltage divider consists of three identical 5k resistors which create two reference voltages at 1/3 and 2/3 of the supplied voltage, which can range from 5 to 15V.

Next are the two comparators. A comparator is a circuit element that compares two analog input voltages at its positive (non-inverting) and negative (inverting) input terminal. If the input voltage at the positive terminal is higher than the input voltage at the negative terminal the comparator will output 1. Vice versa, if the voltage at the negative input terminal is higher than the voltage at the positive terminal, the comparator will output 0.

The first comparator negative input terminal is connected to the 2/3 reference voltage at the voltage divider and the external “control” pin, while the positive input terminal to the external “Threshold” pin.

On the other hand, the second comparator negative input terminal is connected to the “Trigger” pin, while the positive input terminal to the 1/3 reference voltage at the voltage divider.

So using the three pins, Trigger, Threshold and Control, we can control the output of the two comparators which are then fed to the R and S inputs of the flip-flop. The flip-flop will output 1 when R is 0 and S is 1, and vice versa, it will output 0 when R is 1 and S is 0. Additionally, the flip-flop can be reset via the external pin called “Reset” which can override the two inputs, thus reset the entire timer at any time.

The Q-bar output of the flip-flop goes to the output stage or the output drivers which can either source or sink a current of 200mA to the load. The output of the flip-flop is also connected to a transistor that connects the “Discharge” pin to ground.

In our project we are showing “CSE – 231” sequentially through BCD to 7 segment display. For the sequential part we are using a JK Flip-flop. As we are using Logisim so we are using “Clock” to give a clock pulse. But if we use a 555 Timer then we can set the frequency and then “CSE-231” will be displayed sequentially in the display after a fixed time automatically.