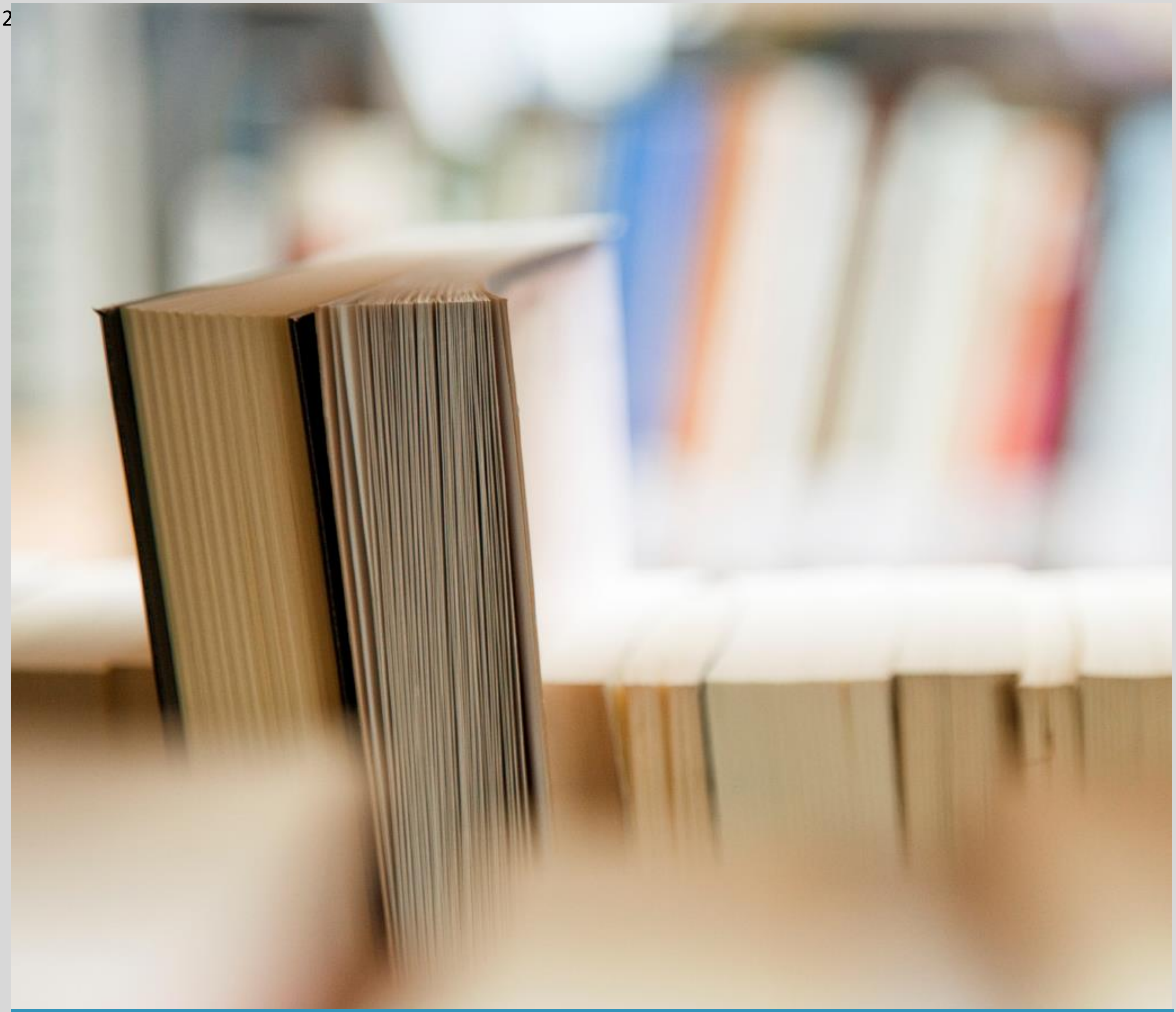
2



BOOLEAN SIMPLIFICATION AND CODE CONVERSION

Experiment 3 & 4

Bimal Parajuli(20BDS0405) 7/2/21 CSE - 1003 (LAB)

SIMPLIFICATION OF BOOLEAN FUNCTIONS USING K-MAP TECHNIQUES

OBJECTIVE:

- To develop the truth table for a combinational logic problem
- To use Karnaugh map to simplify Boolean expressions.
- To draw and simplify sum of products expressions.
- To draw logic diagrams using NAND gates.

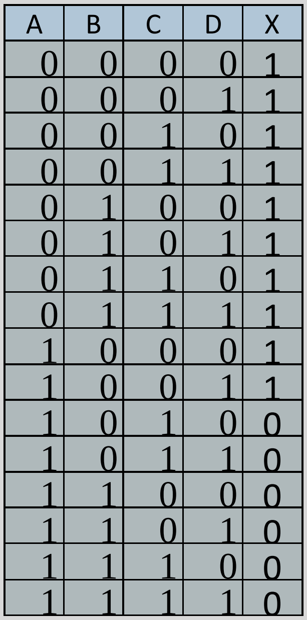
APPARATUS:

- PB-503
- 7400 Quadruple 2 input NAND gates.
- 7404 Hex inverters
- 7410 Triple 3-input NAND gates
- 7420 Dual 4-input NAND gates
- 7432 Dual 2-input OR gates
- 7408 Dual 2-input AND gates

Part 1: BCD invalid code detector

BCD is a 4-bit binary code representing the decimal numbers 0 through 9. The binary numbers 1010 through 1111 are not used in BCD.

a) Construct a truth table containing all possible inputs and desired output. Assume that the desired output for a valid code is a 1, and for an invalid code is 0. Complete the truth table as shown in Table 1. A is the most significant bit, and D is the least significant bit.



A	B	C	D	X
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

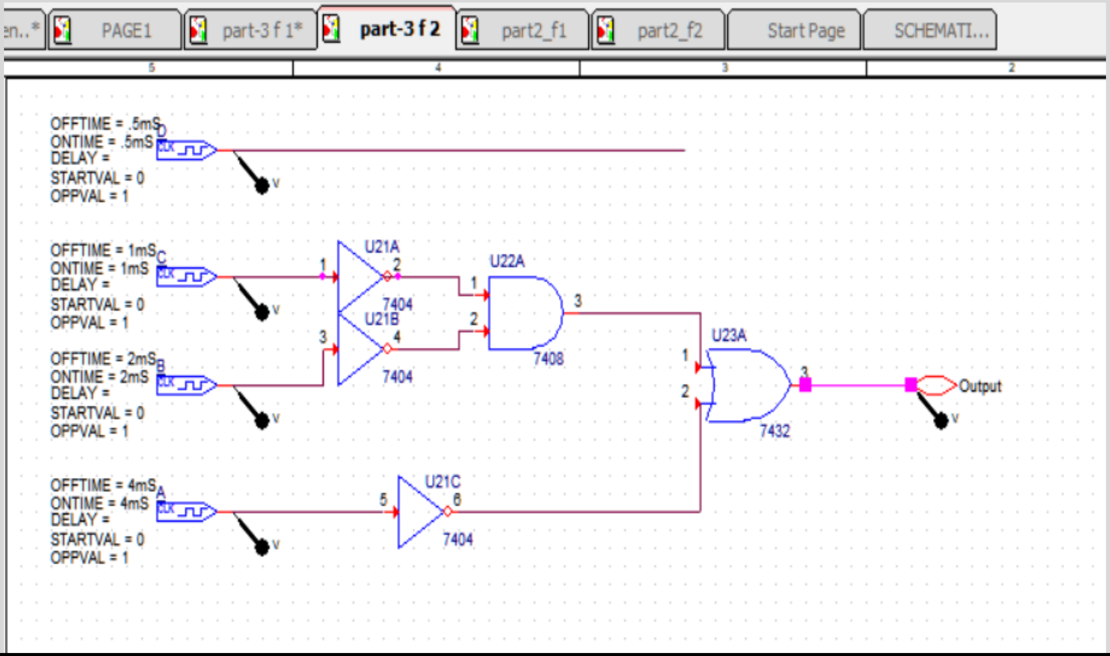
Table: Truth Table of validity of BCD.

b) Draw the Karnaugh map, and write the simplified Boolean expression for the valid codes as sum of products.

AB \ CD	00	01	11	10
00	1	1	1	1
01	1	1	1	1
11	0	0	0	0
10	1	1	0	0

F = A’ + B’ C’

c) Draw the circuit for the above simplified Boolean expression.

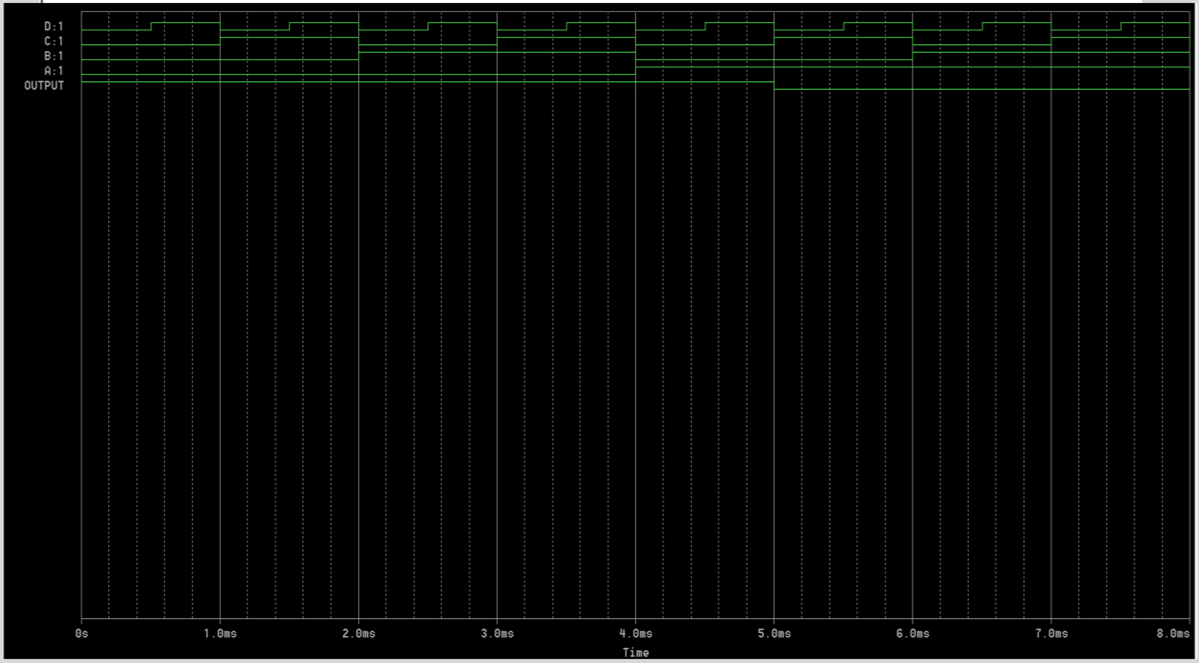


OFFTIME = .5mS
ONTIME = .5mS
DELAY =
STARTVAL = 0
OPPVAL = 1

OFFTIME = 1mS
ONTIME = 1mS
DELAY =
STARTVAL = 0
OPPVAL = 1

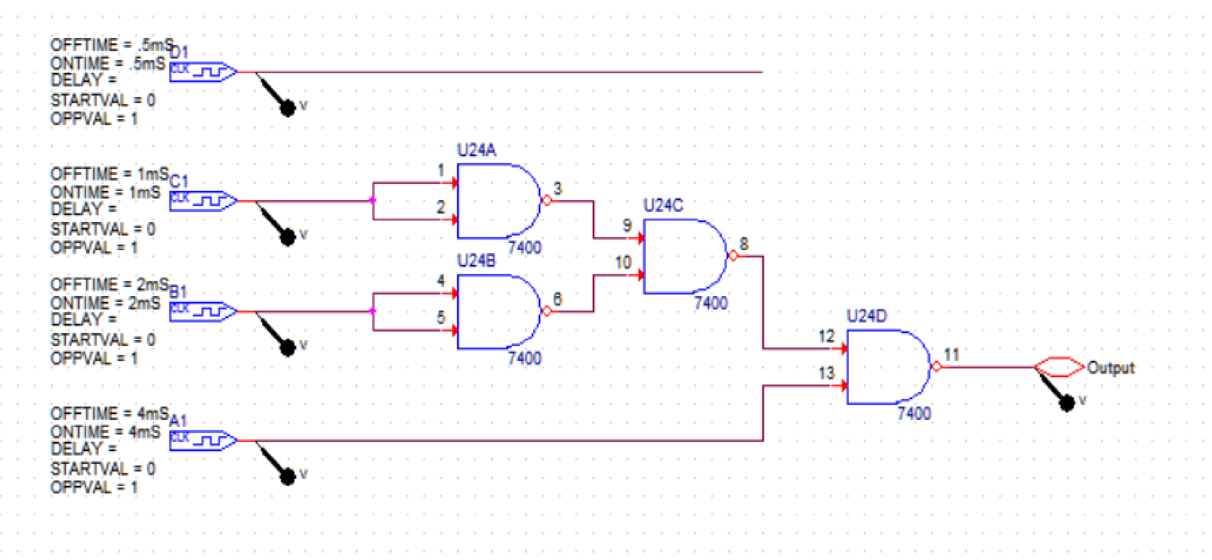
OFFTIME = 2mS
ONTIME = 2mS
DELAY =
STARTVAL = 0
OPPVAL = 1

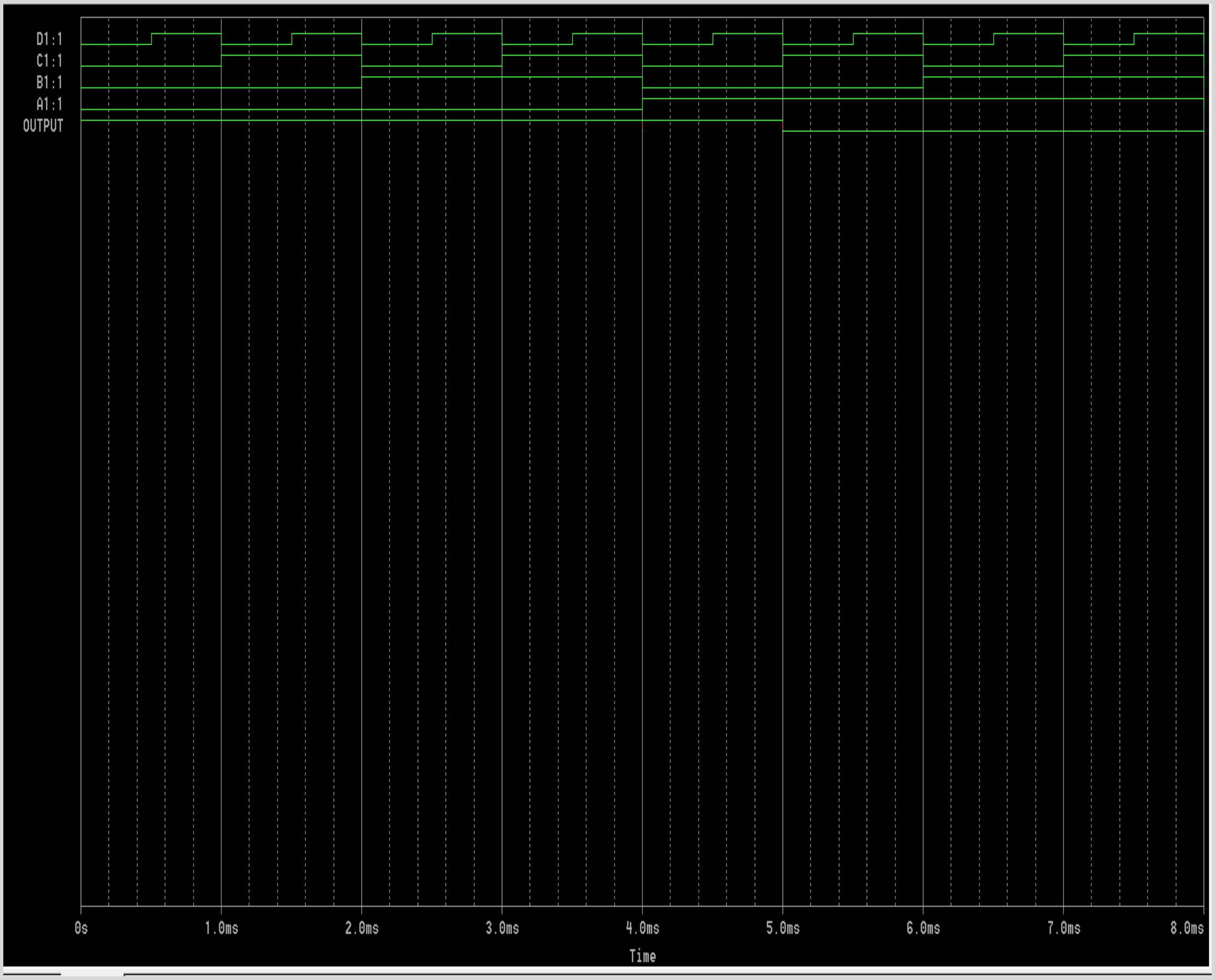
OFFTIME = 4mS
ONTIME = 4mS
DELAY =
STARTVAL = 0
OPPVAL = 1



d) Using the universal property of the NAND gate connect an equivalent circuit for these codes that uses only NAND gates.

Using NAND gates only:





Part 2: Boolean Functions (1)

1. Simplify the following two Boolean functions by means of Karnaugh maps.

2. Draw the logic diagrams for outputs F_1 and F_2 in terms of the inputs A, B, C, and D.

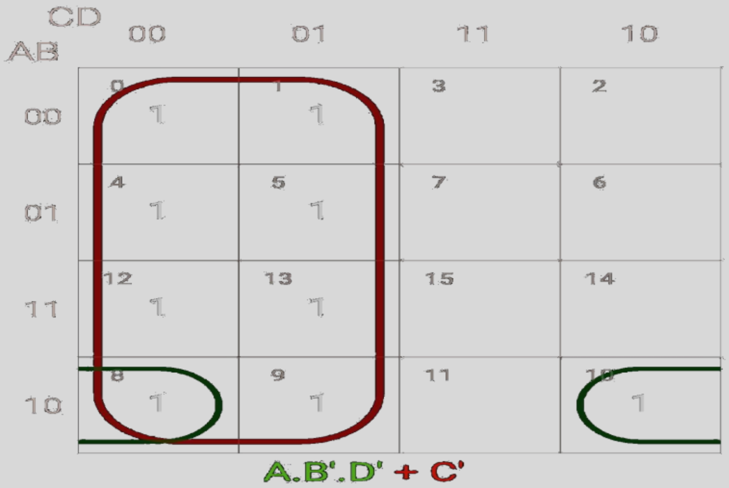
3. Implement and draw the two functions F_1 and F_2 together by using minimum number of NAND gates.

4. Connect the circuit and verify it's operation by preparing a truth table for F_1 and F_2 similar to Table 1.

For Boolean Function F1:

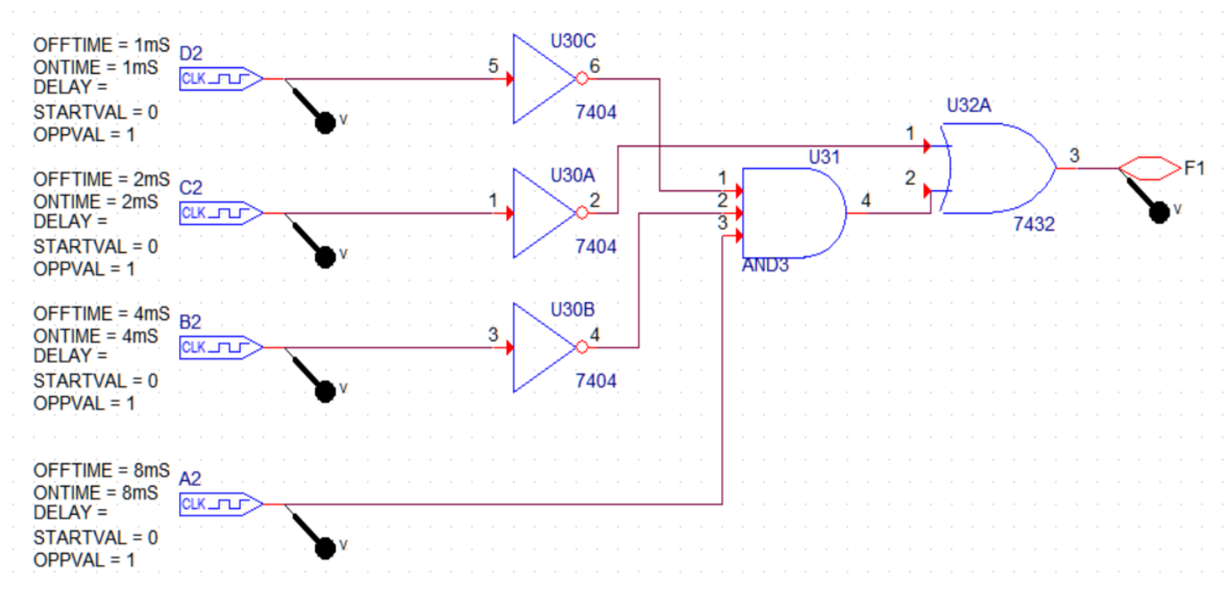
$$F_1 (A, B, C, D) = \Sigma m\ (0,1,4,5,8,9,10,12,13)$$

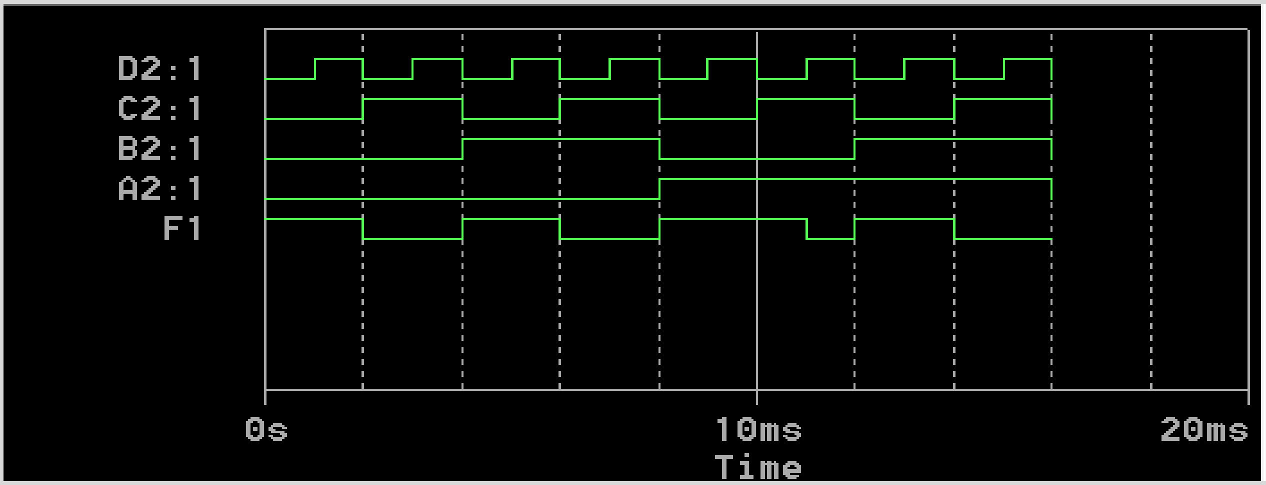
K- Map of F_1:



20BDS0405

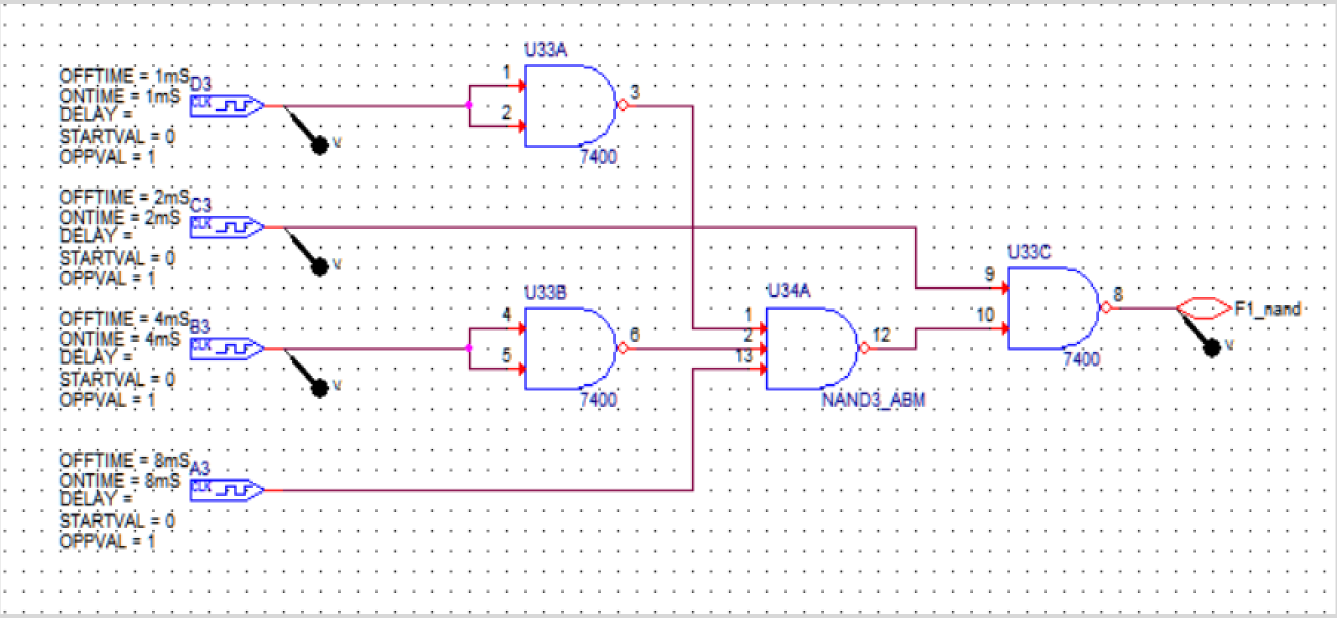
Logic Circuit of F1:

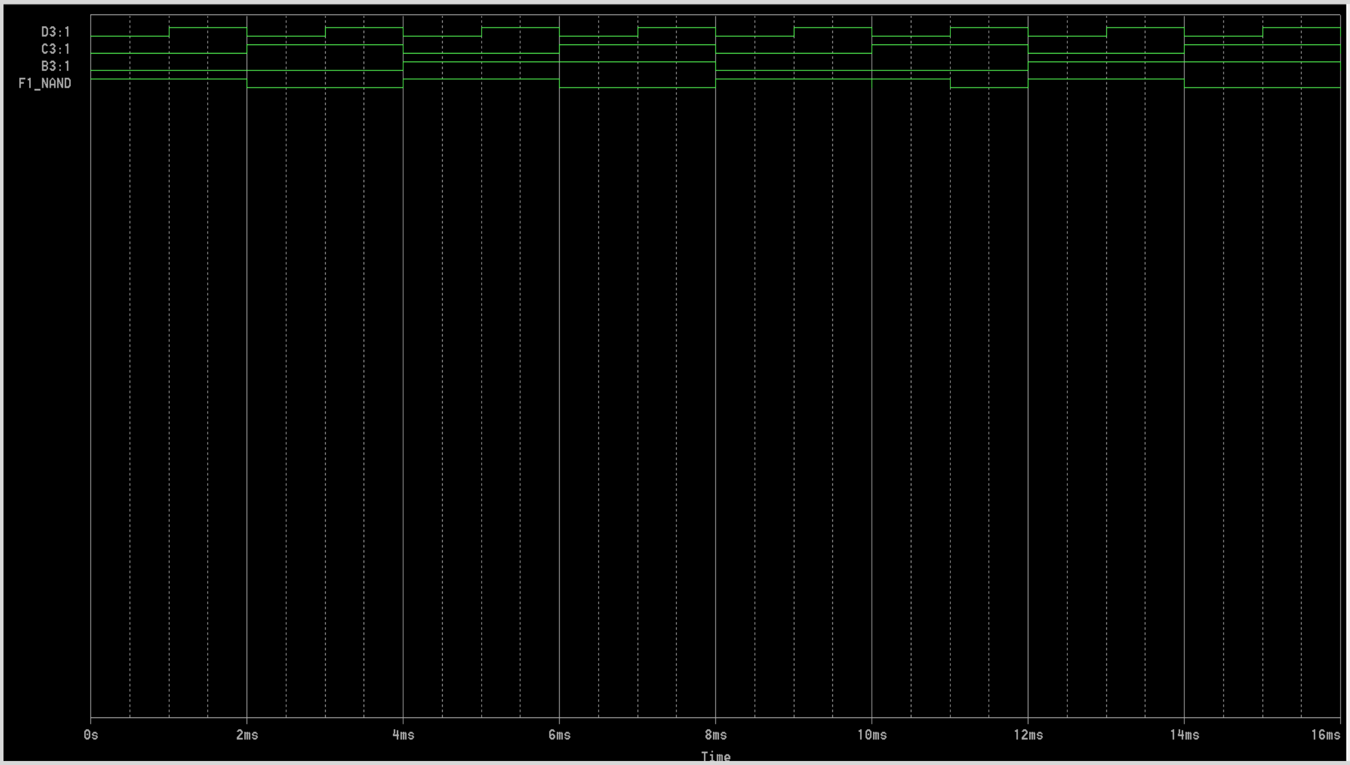




20BDS0405

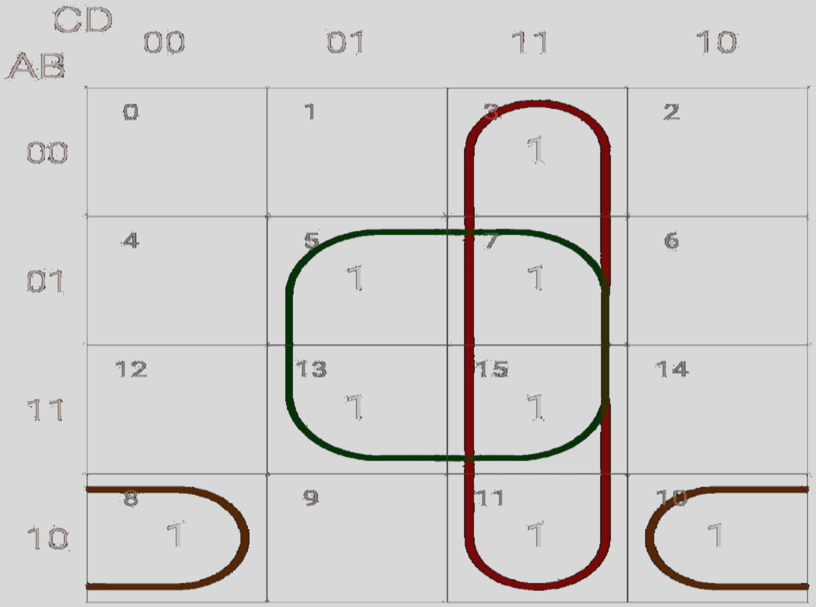
Logic Circuit of F1 using Only NAND gates:





$$F_2\,(A,B,C,D) = \Sigma m\,(3,5,7,8,10,11,13,15)$$

K- Map of F_2:

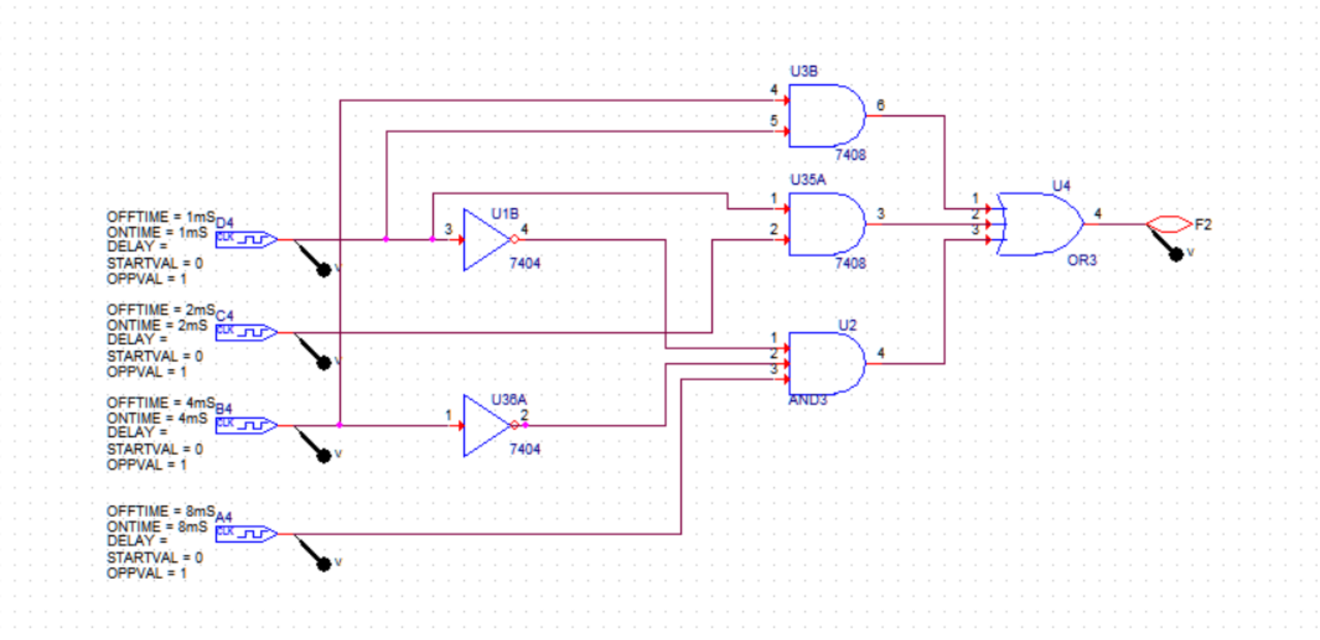


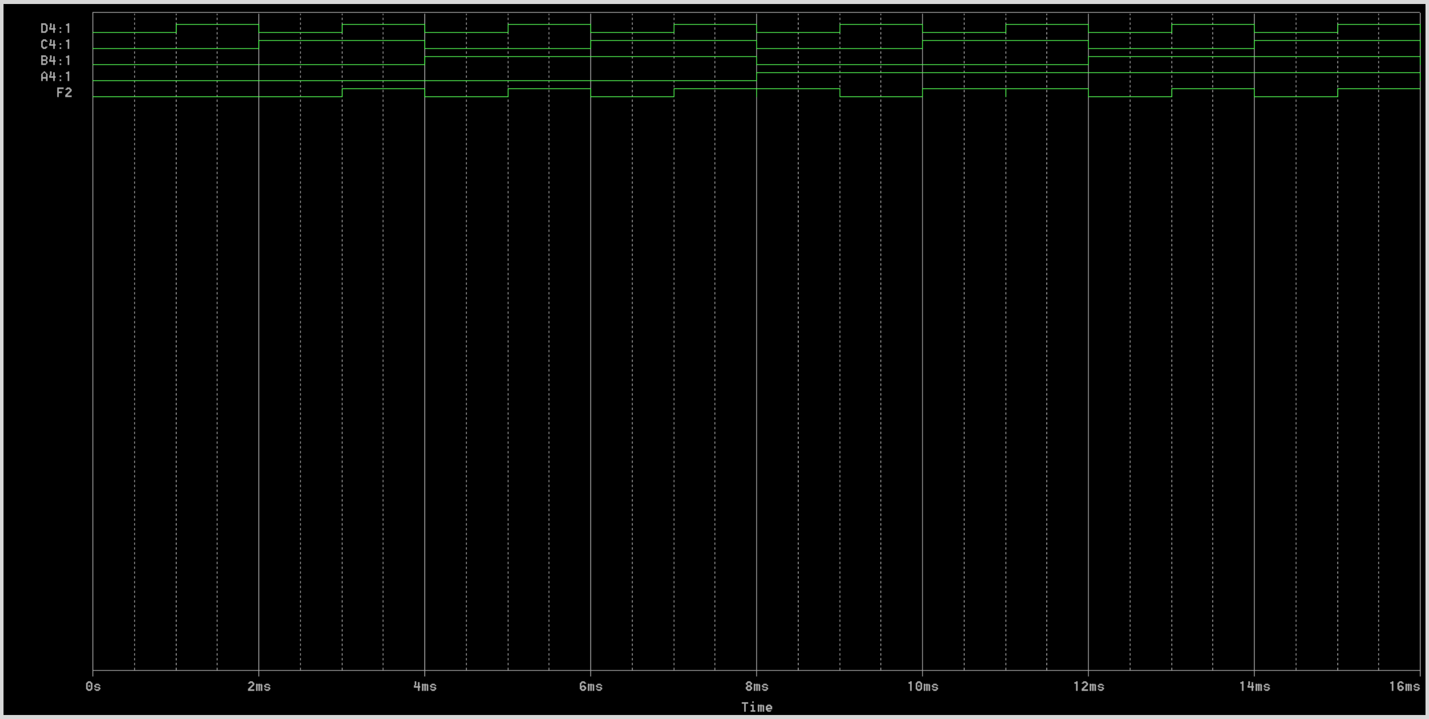
CD AB	00	01	11	10
00	0	1	3 1	2
01	4	5 1	7 1	6
11	12	13 1	15 1	14
10	8 1	9	11 1	10 1

A.B'.D' + C.D + B.D

20BDS0405

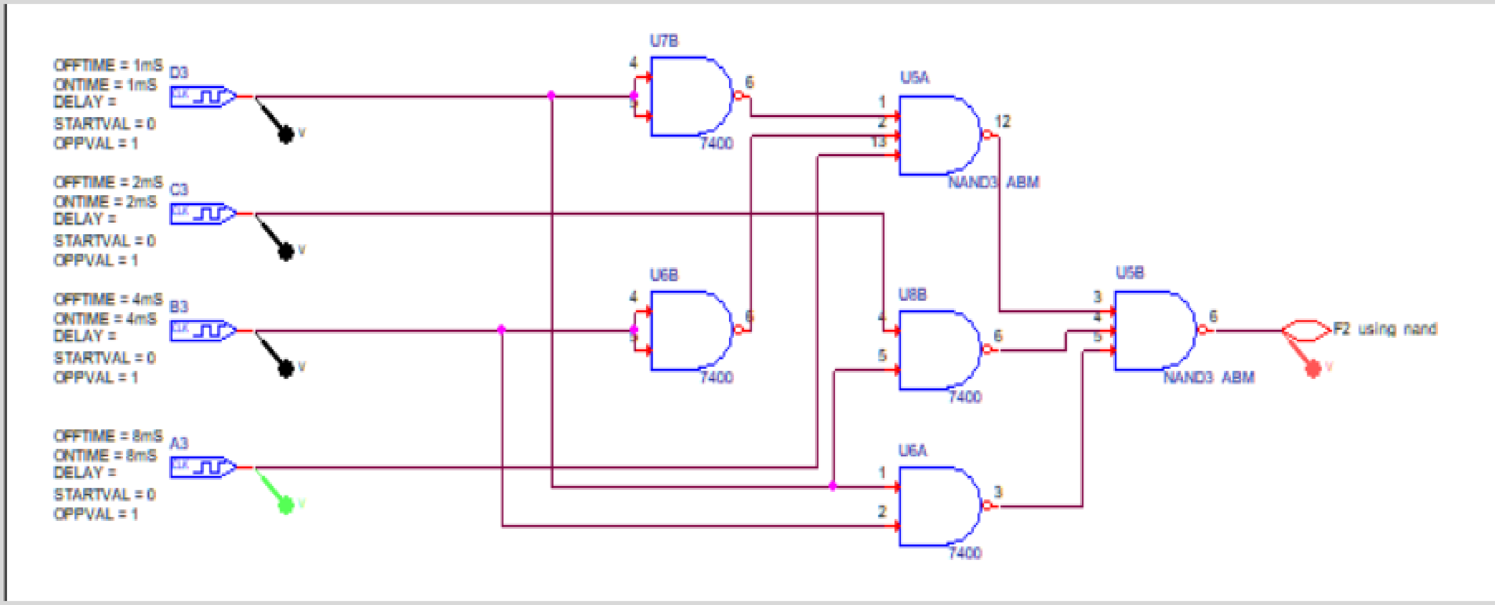
Logic Circuit of F2:

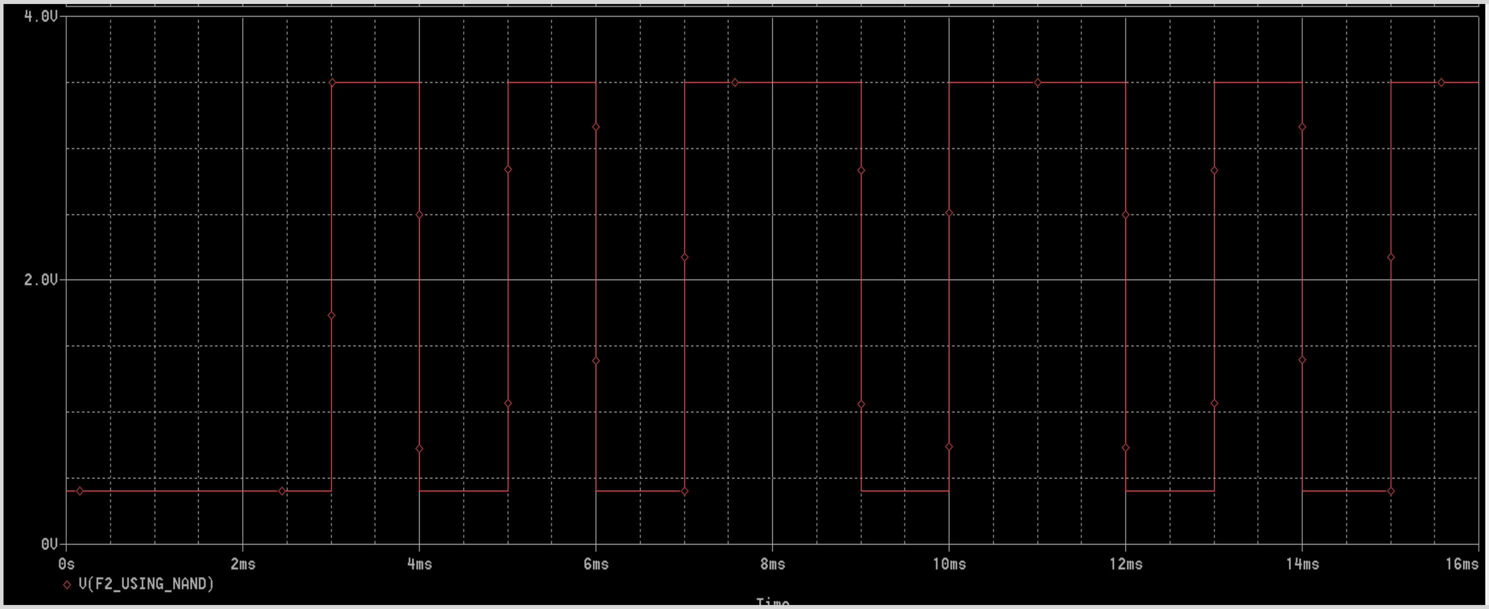




20BDS0405

Logic Circuit of F2 using Only NAND gates:

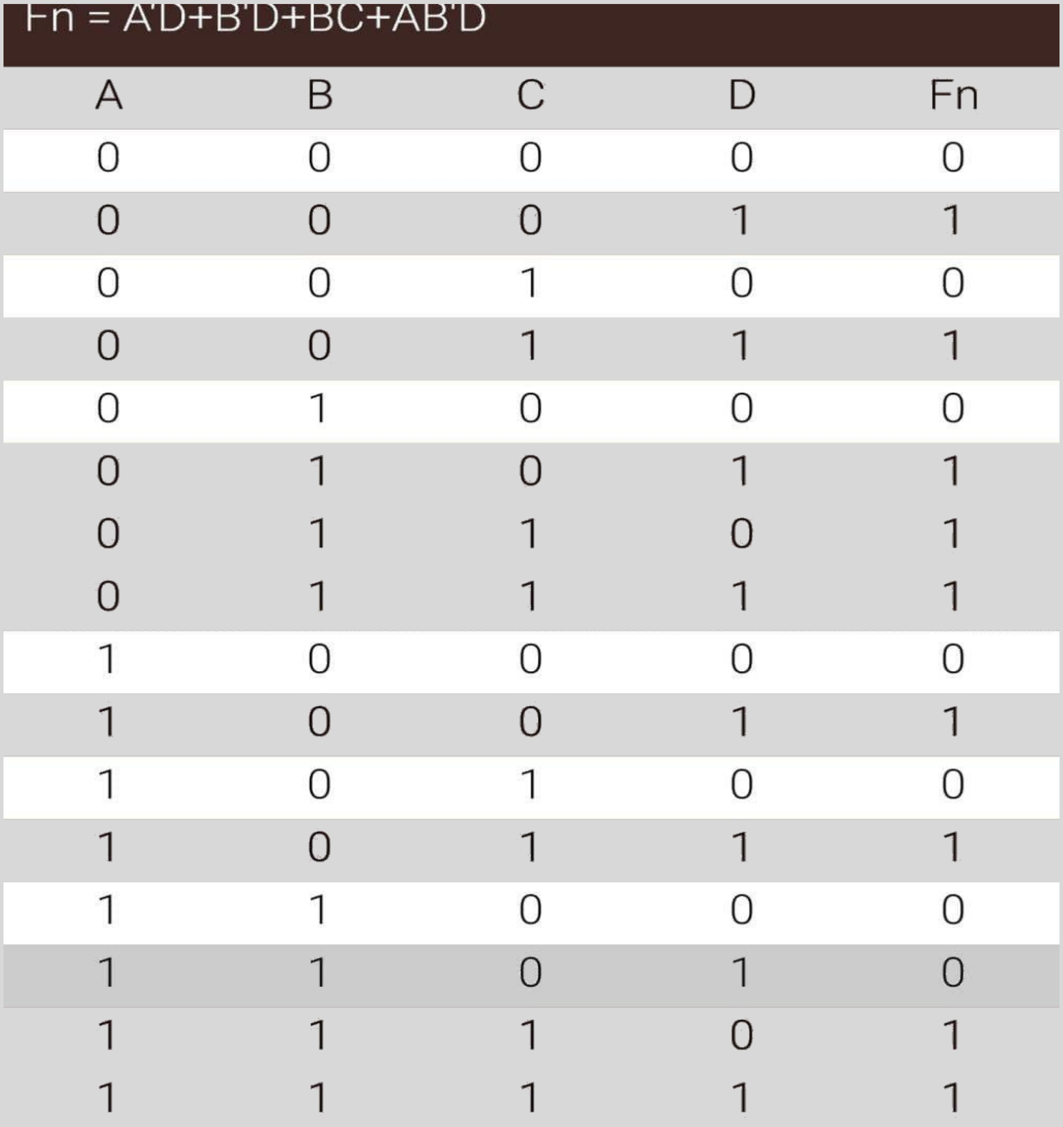




Part 3: Boolean Functions (2)

1. Derive a truth table for the following Boolean Functions.

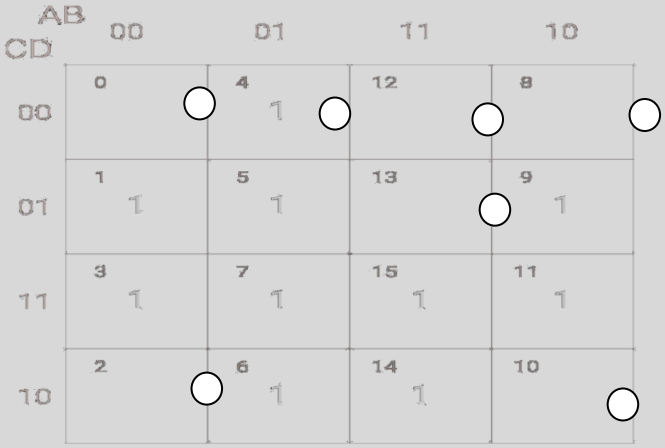
$$F=A'D+B'D+BC+AB'D$$



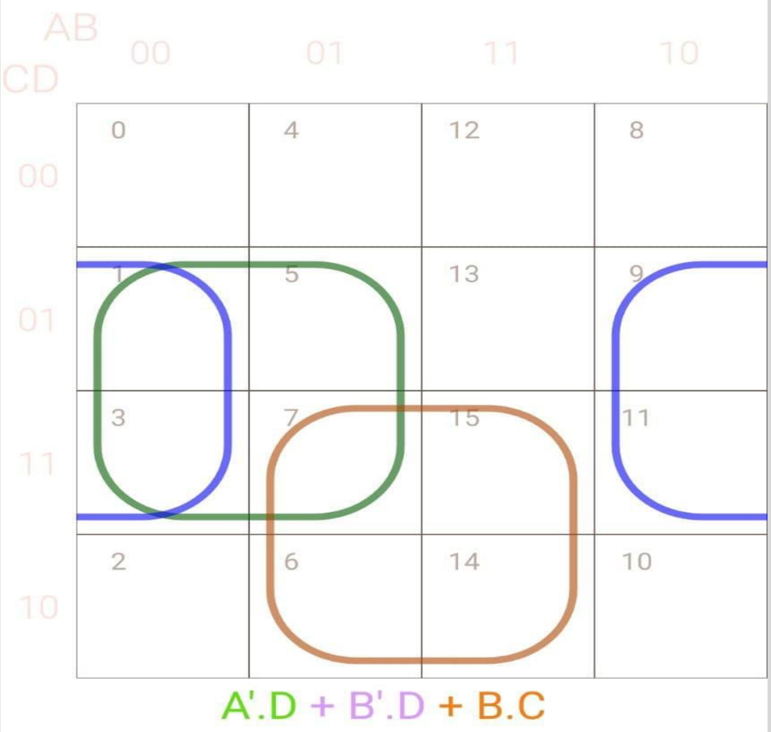
Fn = A'D+B'D+BC+AB'D

A	B	C	D	Fn
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

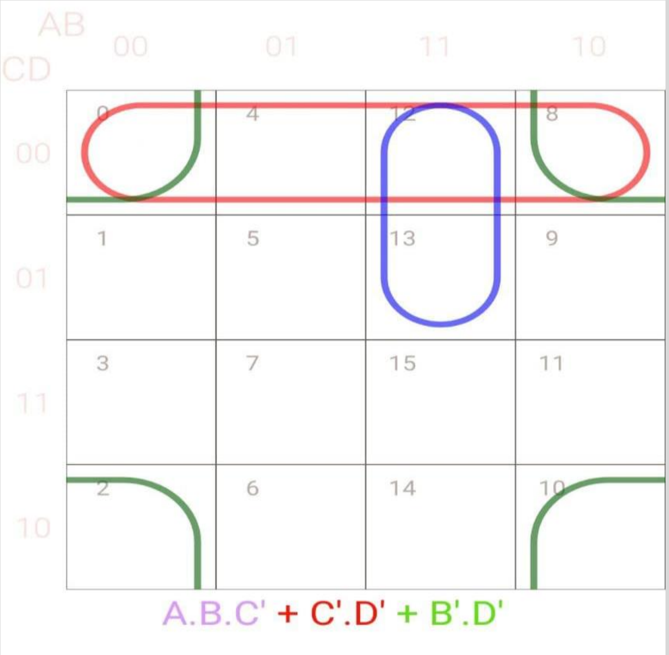
2. Draw a Karnaugh map.



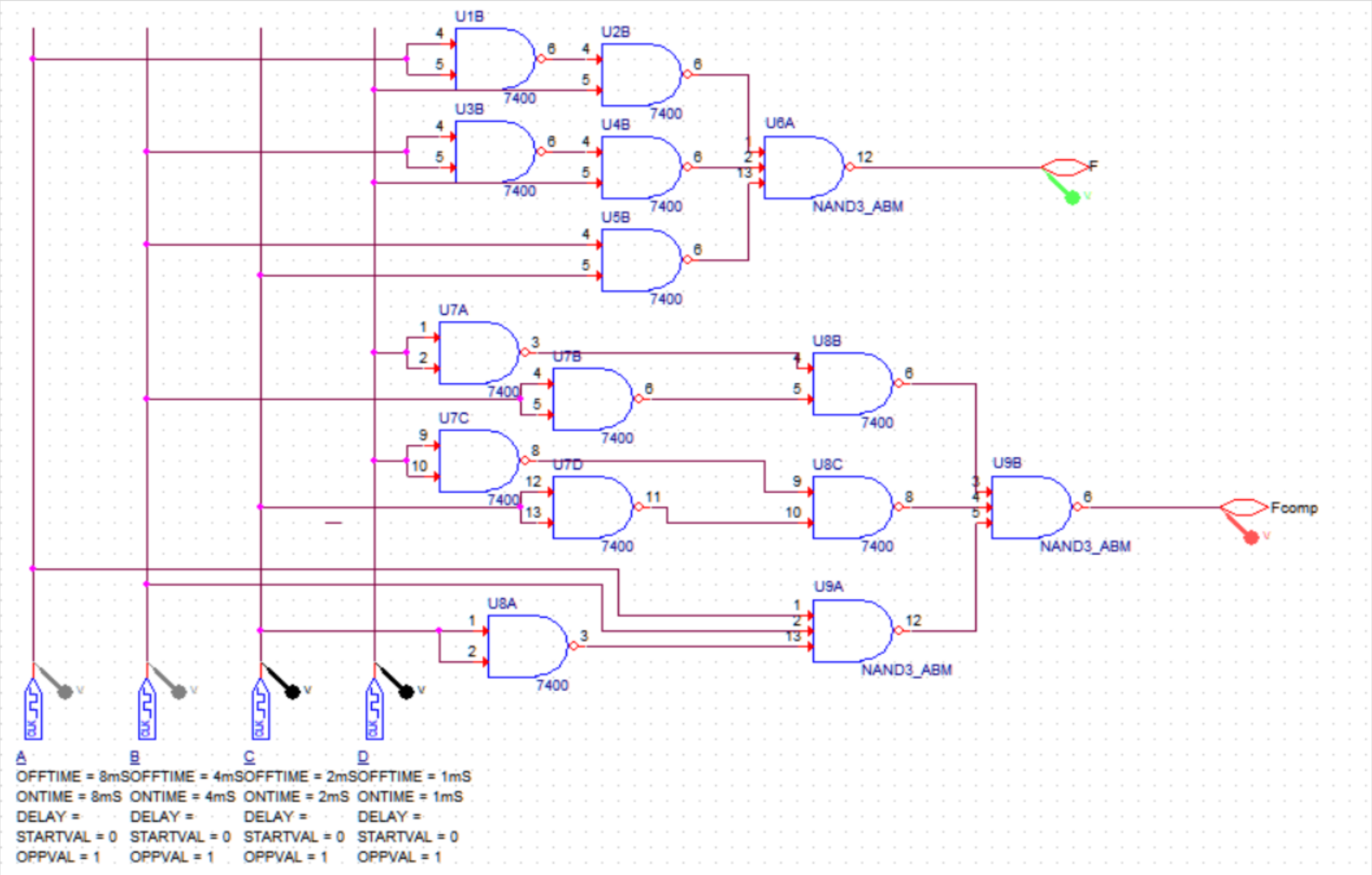
3. Combine all the 1’s to obtain the simplified function for *F*.



4. Combine all the 0’s to obtain the simplified function for *F’*.

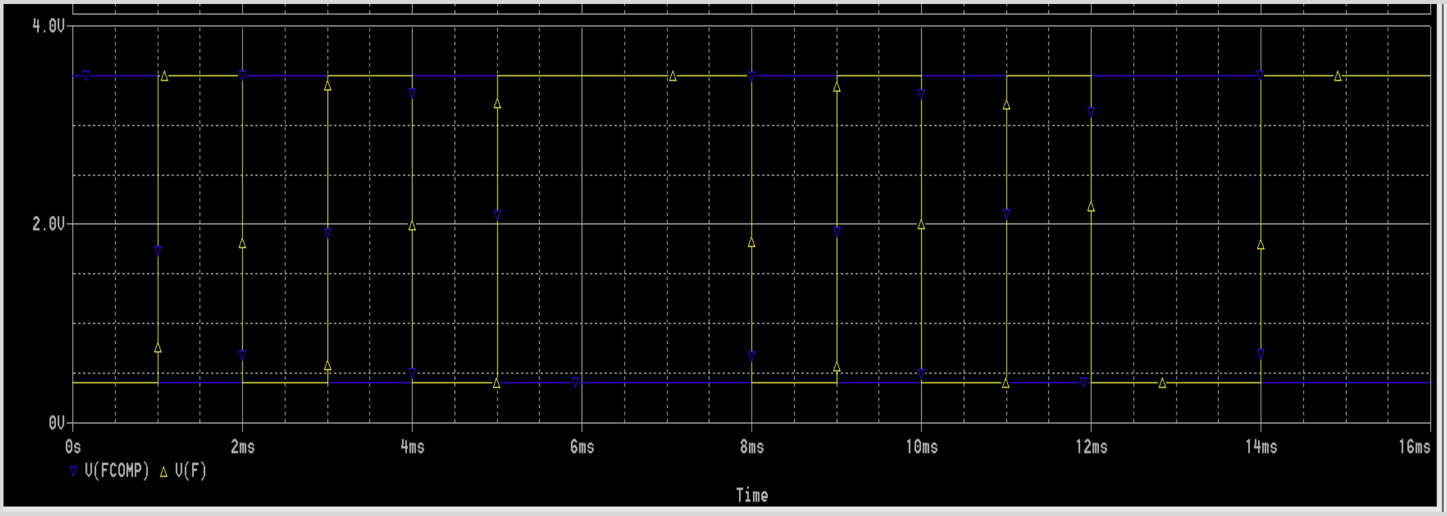


5. Draw both circuits.


U1B
U2B
U3B
U4B
U5B
U6A
NAND3_ABM
F
U7A
U7B
U7C
U7D
U8A
U8B
U8C
U9A
U9B
Fcomp
7400
A
B
C
D
OFFTIME = 8mS ONTIME = 8mS DELAY = STARTVAL = 0 OPPVAL = 1
OFFTIME = 4mS ONTIME = 4mS DELAY = STARTVAL = 0 OPPVAL = 1
OFFTIME = 2mS ONTIME = 2mS DELAY = STARTVAL = 0 OPPVAL = 1
OFFTIME = 1mS ONTIME = 1mS DELAY = STARTVAL = 0 OPPVAL = 1

6. Using OrCAD, implement both F and F’ using NAND gates and connect two circuits to the same input switches but to separate output LED’s. Prove that both circuits are complement of each other. In the lab implement and verify the operations of the circuit.

Logical Circuit and Output of Both above circuits are:



Part 4: A Majority

A nine member legislative committee requires a 2/3 vote to spend a billion dollars. The vote is tabulated and converted to BCD code. If 2/3 of the committee is in favor, the vote will be the BCD representation of 6, 7, 8, or 9.

1. Derive a truth table for the problem, Table 2.

A	B	C	D	X
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

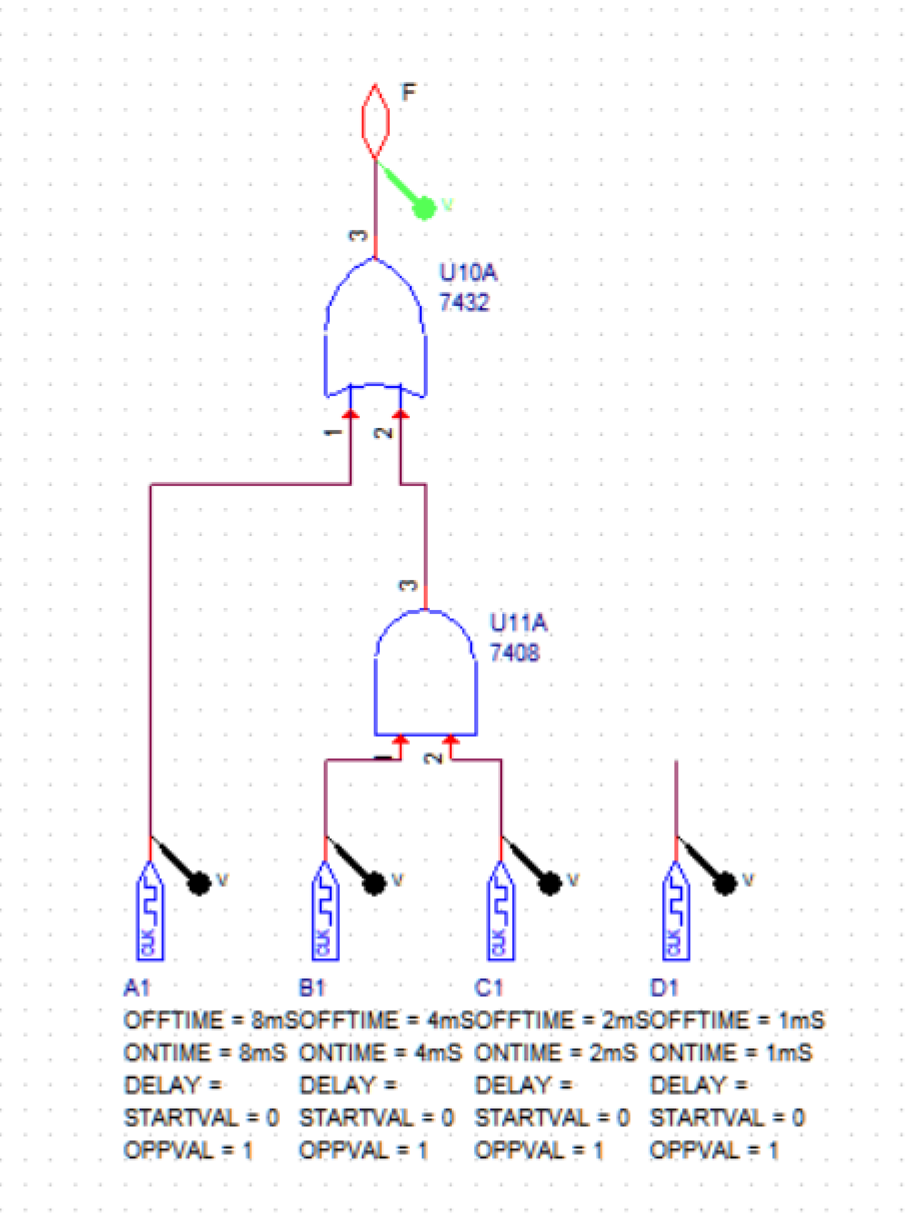
2. Derive a minimum sum of products expression from the map. {Enter the invalid BCD codes on the map as don’t cares (x)}.

AB\CD	00	01	11	10
00	0	0	0	0
01	0	0	1	1
11	x	x	x	x
10	1	1	x	X

Hence, from the above Kmap, the minimal SOP form is:

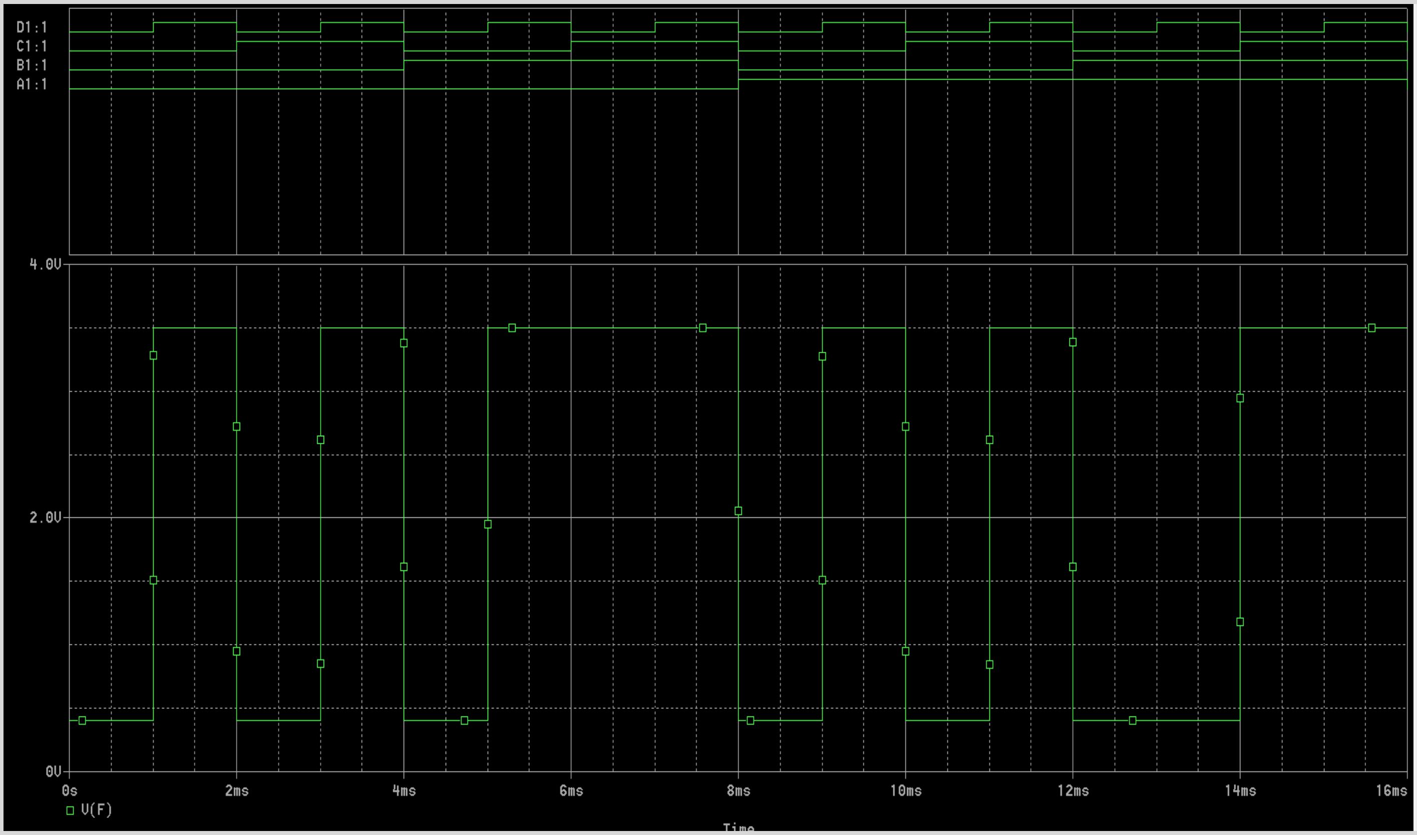
F= A + BC

3. Using LogicWorks, design a circuit that lights an LED if a majority has voted in favor of spending the billion dollars. Implement this circuit and verify its operation in the lab using hardware.



A1	B1	C1	D1
OFFTIME = 8mS	OFFTIME = 4mS	OFFTIME = 2mS	OFFTIME = 1mS
ONTIME = 8mS	ONTIME = 4mS	ONTIME = 2mS	ONTIME = 1mS
DELAY =	DELAY =	DELAY =	DELAY =
STARTVAL = 0	STARTVAL = 0	STARTVAL = 0	STARTVAL = 0
OPPVAL = 1	OPPVAL = 1	OPPVAL = 1	OPPVAL = 1

20BDS0405


D1:1
C1:1
B1:1
A1:1
4.0V
2.0V
0V
0s
2ms
4ms
6ms
8ms
10ms
12ms
14ms
16ms
□ V(F)
Time

EXPERIMENT #4
DESIGN OF CODE CONVERTERS

OBJECTIVE:

1. Design and build gray code to binary converter.
2. Design and build BCD-to-7 segment converter.

APPARATUS:

- Seven segment display.
- SN 7400 quad 2-input NAND gates (1)
- SN 7410 triple 3-input NAND gates (4)
- SN 7420 dual 4-input NAND gates (4)
- SN 7404 HEX inverter (1)
- SN 7446 BCD-to-seven segment decoder.

THEORY:

The conversion from one code to another is common in digital systems. Sometimes the output of a system is used as the input to the other system. A conversion circuit is necessary between 2 systems if each system uses different codes for the same information.

In this experiment we will design and construct 3-combinational circuit converters:

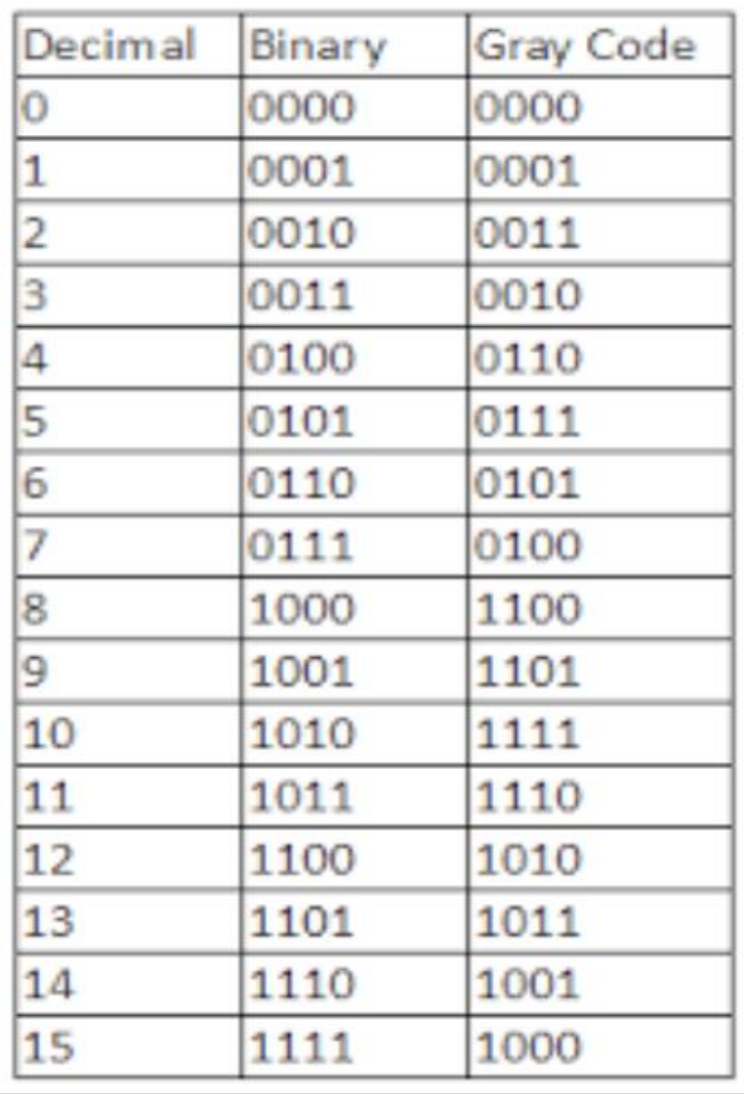
See section 4-5 in your book for further information.

Procedure:

1. *Gray code to Binary converter:*

Gray code is one of the codes used in digital systems. It has the advantage over binary numbers that only one bit in the code word changes when going from one number to the next.

The following table illustrates the corresponding code representations of Decimal numbers in Binary Number System (Base 2 number system) and Gray Code Convention (Non-weighted)



Decimal	Binary	Gray Code
0	0000	0000
1	0001	0001
2	0010	0011
3	0011	0010
4	0100	0110
5	0101	0111
6	0110	0101
7	0111	0100
8	1000	1100
9	1001	1101
10	1010	1111
11	1011	1110
12	1100	1010
13	1101	1011
14	1110	1001
15	1111	1000

Design a combinational circuit with 4 inputs and 4 outputs that converts a four- bit gray code number into an equivalent four-bit Binary number.

Let, the 4 Bit Gray Code number be ABCD and Binary number be WXYZ.

	Gray Code (ABCD)					Binary Code (WXYZ)			
ecimal	A	B	C	D		W	X	Y	Z
0	0	0	0	0		0	0	0	0
1	0	0	0	1		0	0	0	1
2	0	0	1	1		0	0	1	0
3	0	0	1	0		0	0	1	1
4	0	1	1	0		0	1	0	0
5	0	1	1	1		0	1	0	1
6	0	1	0	1		0	1	1	0
7	0	1	0	0		0	1	1	1
8	1	1	0	0		1	0	0	0
9	1	1	0	1		1	0	0	1
10	1	1	1	1		1	0	1	0
11	1	1	1	0		1	0	1	1
12	1	0	1	0		1	1	0	0
13	1	0	1	1		1	1	0	1
14	1	0	0	1		1	1	1	0
15	1	0	0	0		1	1	1	1

Table 1: Gray code with corresponding Binary and Decimal Equivalent values

- Use Karnaugh map technique for simplification.
- Use LogicWorks OR OrCAD for pre-lab demonstrations.
- Select the library "7400dev.clf ' in the Parts Palette.
- Then select the XOR chip 74-86. This would give you a set of 4 XOR's as shown in Fig. 1, just like the hardware chip 74-86. You could use as many as needed from these XOR gates in your design.
- Get back to ALL LIBRARIES and select switches for the inputs and Binary Probes as indicators of the outputs.
- Verify your design in the pre-Lab. During the Lab construct the circuit and verify its operation

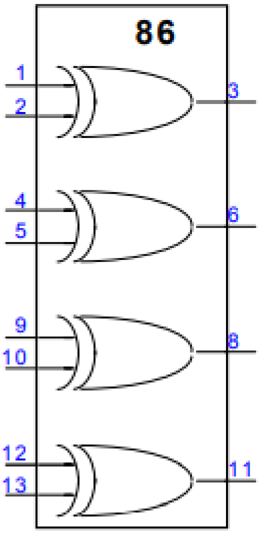
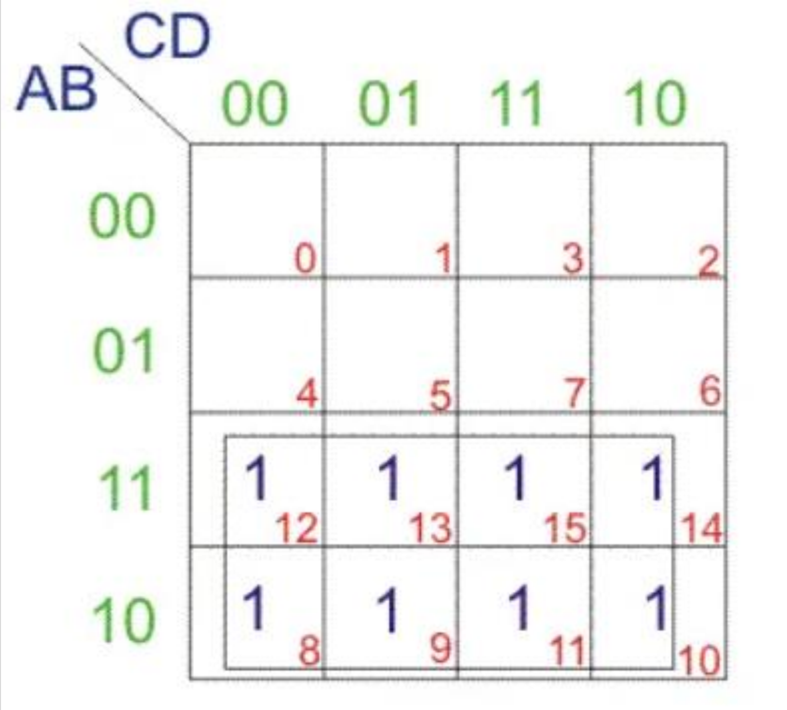


Figure. 1 XOR chip74-86

20BDS0405

Following is the K Map simplification for Gray code to Binary Conversion. Each digits of the Binary number form a separate Karnaugh Map where each digit is a direct function of all the gray code digits.

For W,

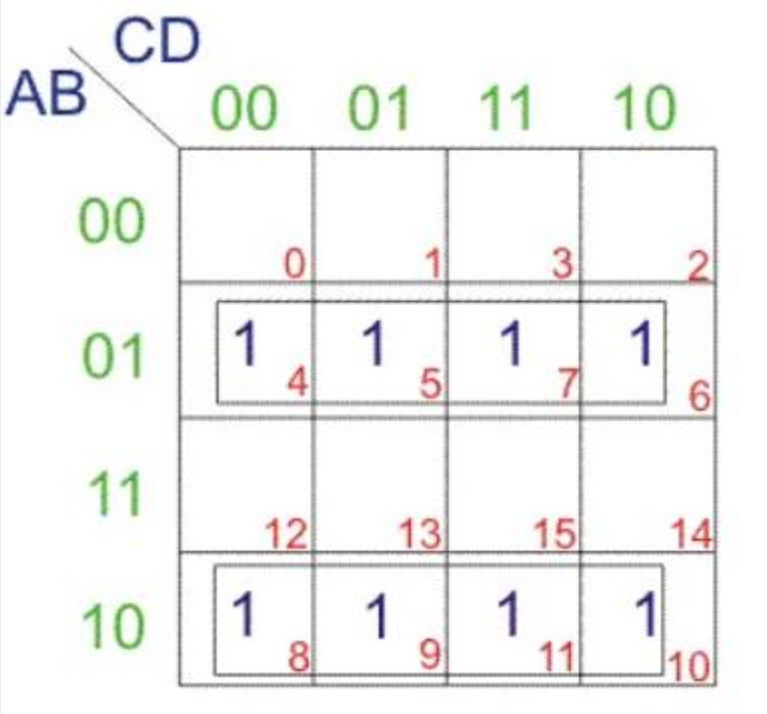


AB \ CD	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	1 12	1 13	1 15	1 14
10	1 8	1 9	1 11	1 10

W=A

20BDS0405

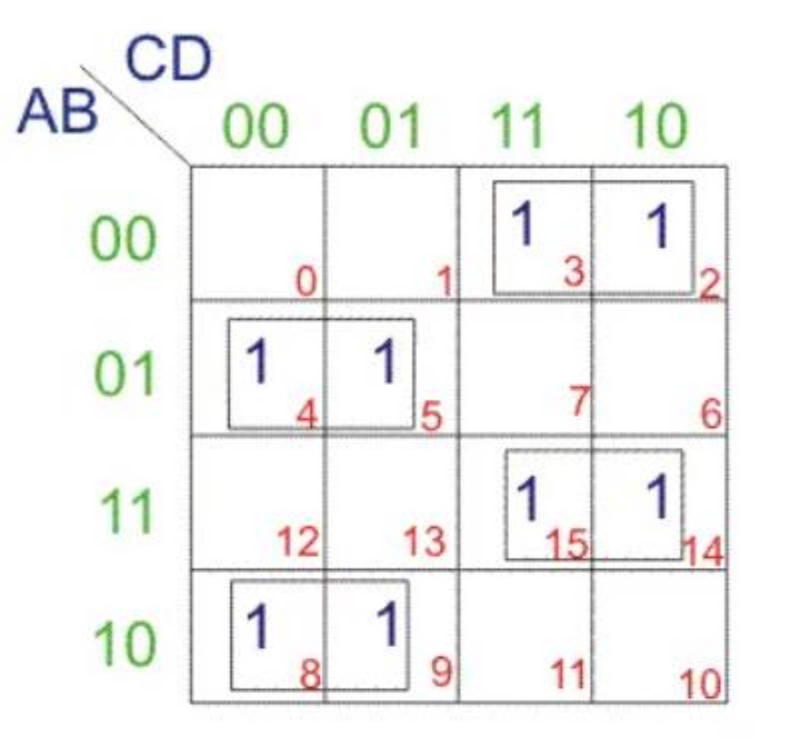
For X,



AB \ CD	00	01	11	10
00	0	1	3	2
01	1 4	1 5	1 7	1 6
11	12	13	15	14
10	1 8	1 9	1 11	1 10

$$X = A'B + AB'$$
$$= A \oplus B$$

For Y,

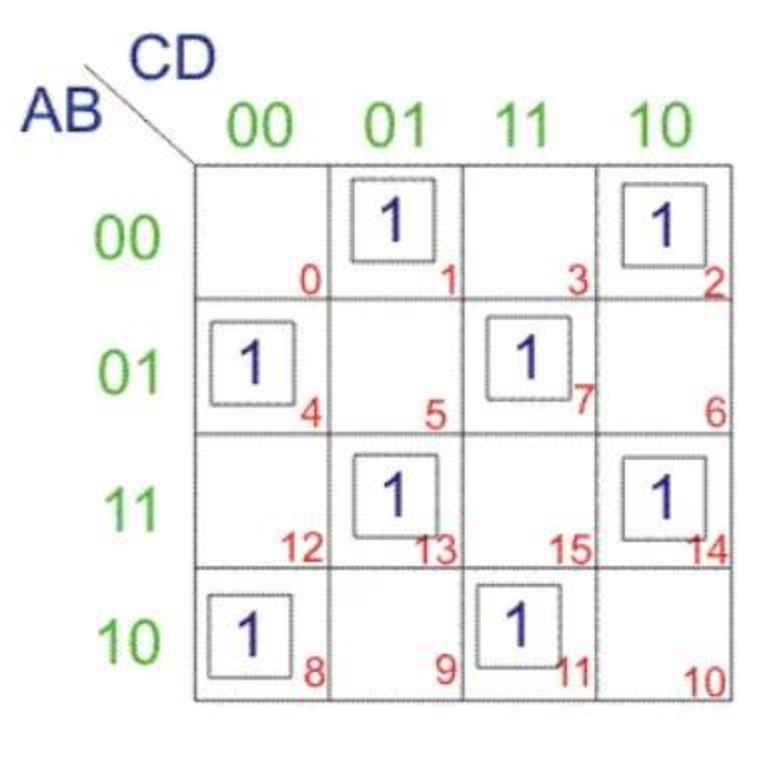


AB \ CD	00	01	11	10
00	0	1	1 3	1 2
01	1 4	1 5	7	6
11	12	13	1 15	1 14
10	1 8	1 9	11	10

$$
\begin{aligned}
Y &= A'BC' + AB'C' + A'B'C + ABC \\
&= A\,(B'C'+BC) + A'\,(BC'+B'C) \\
&= A\,(BC'+B'C)' + A'\,(BC'+B'C) \\
&= A\,(B \oplus C)' + A'\,(B \oplus C) \\
&= A \oplus B \oplus C
\end{aligned}
$$

20BDS0405

For Z,



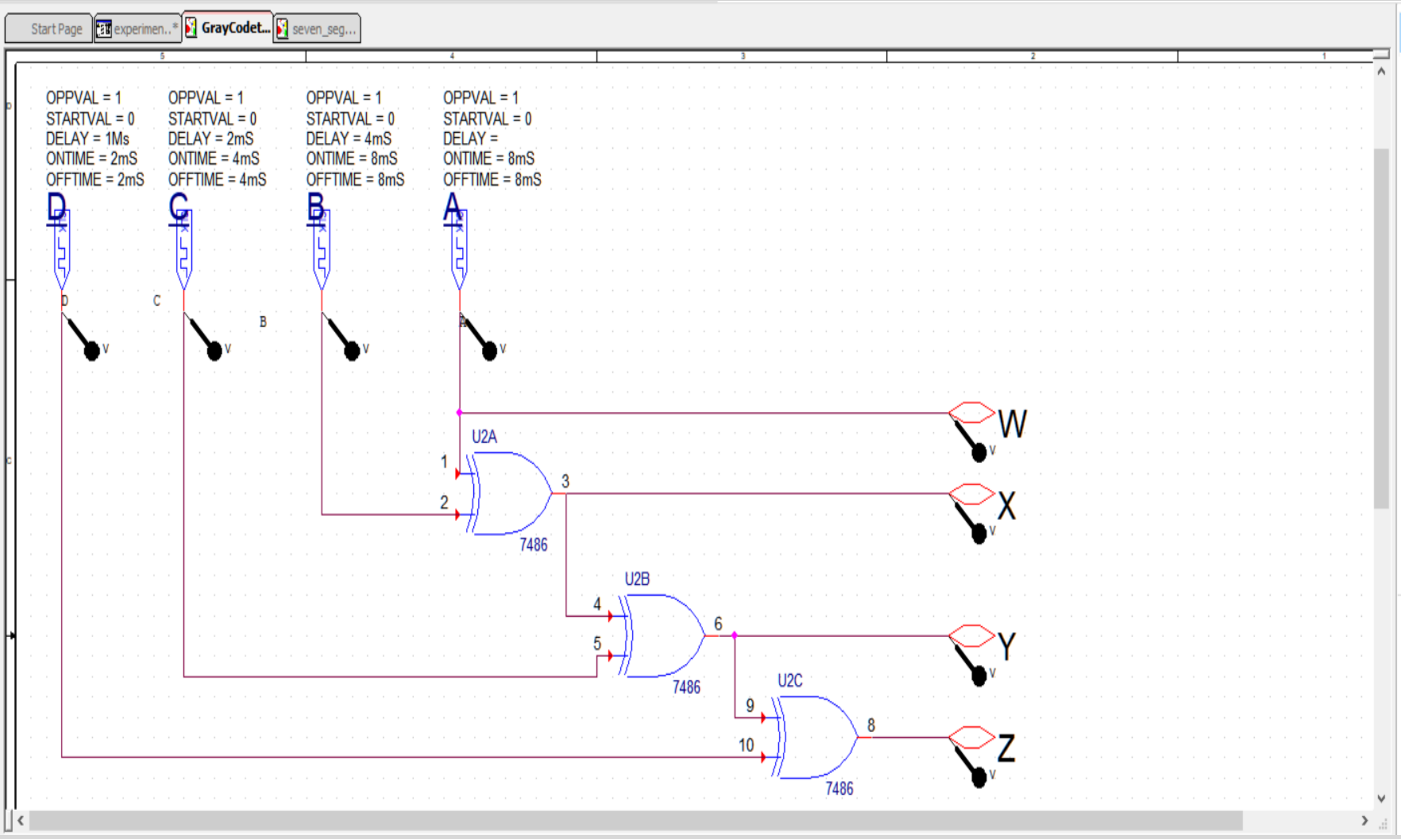
AB \ CD	00	01	11	10
00	0	1 1	3	1 2
01	1 4	5	1 7	6
11	12	1 13	15	1 14
10	1 8	9	1 11	10

Z=A’B’C’D + A’B’CD’ + A’BC’D’ + AB’C’D’…
…+ A’BCD + AB’CD + ABC’D + ABCD’

$Z= A \oplus B \oplus C \oplus D$

20BDS0405

Construction and simulation of Logic diagram to Convert Gray code Input to Binary Number.:



OPPVAL = 1
STARTVAL = 0
DELAY = 1Ms
ONTIME = 2mS
OFFTIME = 2mS

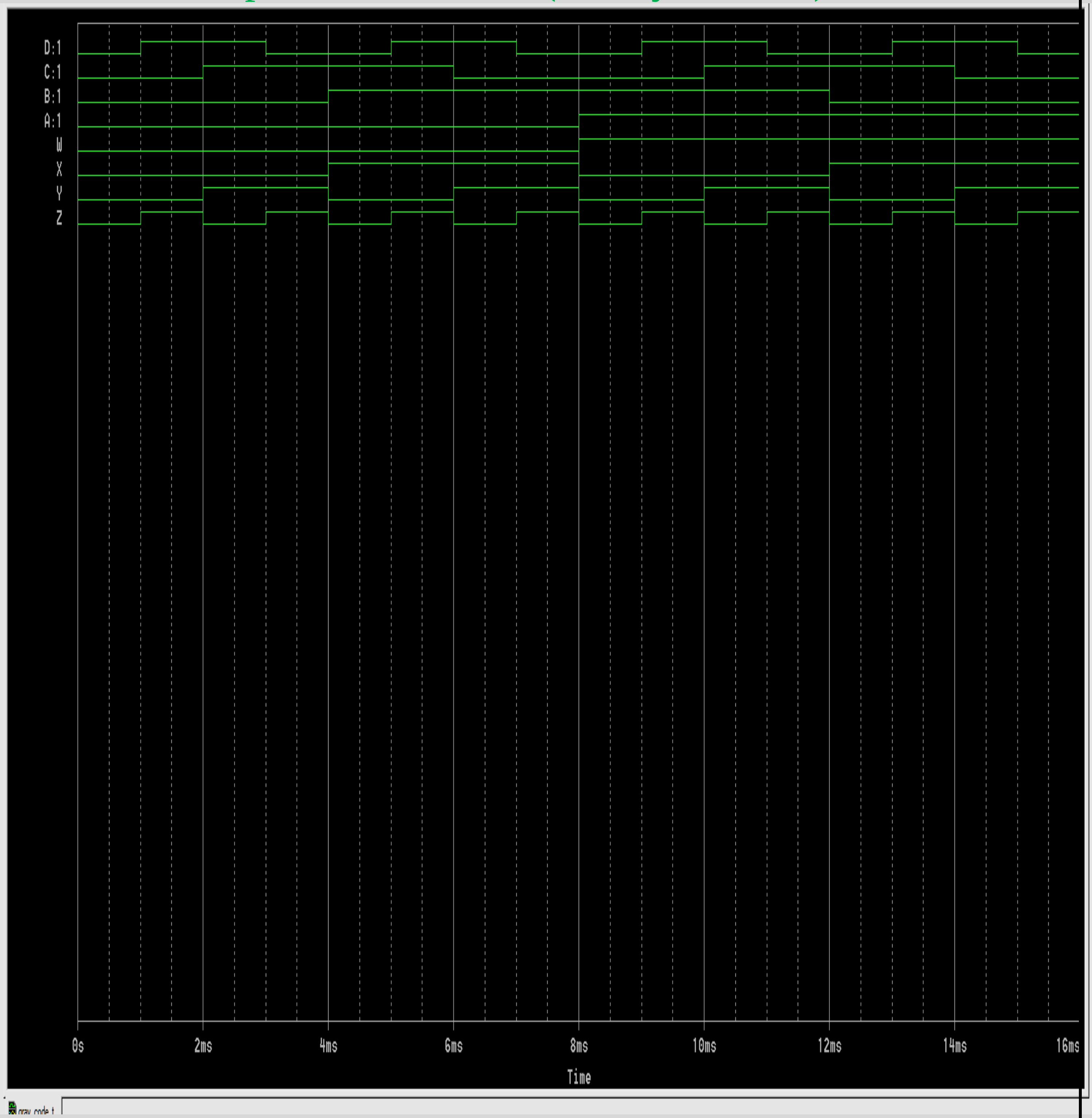
OPPVAL = 1
STARTVAL = 0
DELAY = 2mS
ONTIME = 4mS
OFFTIME = 4mS

OPPVAL = 1
STARTVAL = 0
DELAY = 4mS
ONTIME = 8mS
OFFTIME = 8mS

OPPVAL = 1
STARTVAL = 0
DELAY =
ONTIME = 8mS
OFFTIME = 8mS

Inputs: A, B, C, D (Gray Code)

Outputs: W, X, Y, Z (Binary number)



2. BCD-to-seven Segment converter:

A light emitting Diode (LED) is a PN junction diode. When the diode is forward biased, a current flows through the junction and the light is emitted.

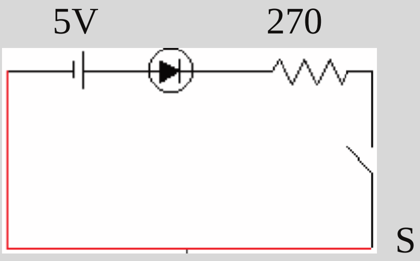


Figure.2

A seven segment LED display contains 7 LEDs. Each LED is called a segment and they are identified as (a, b, c, d, e, f, g) segments. Figure 3.

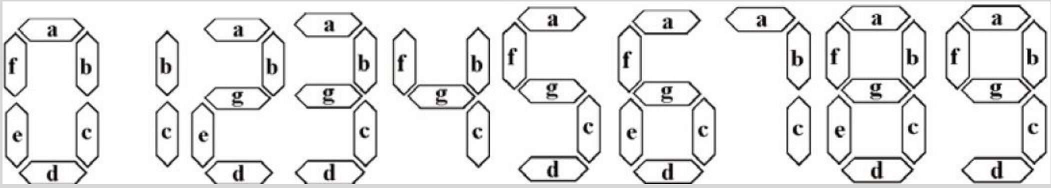


Figure 3. Digits represented by the 7 segments

The display has 7 inputs each connected to an LED segment. All anodes of LEDs are tied together and joined to 5 volts (this type is called common anode type). A limiting resistance network must be used at the inputs to protect the 7-segment from overloading.

BCD inputs are converted into 7 segment inputs (a, b, c, d, e, f, g) by using a decoder, as shown in Fig.4.

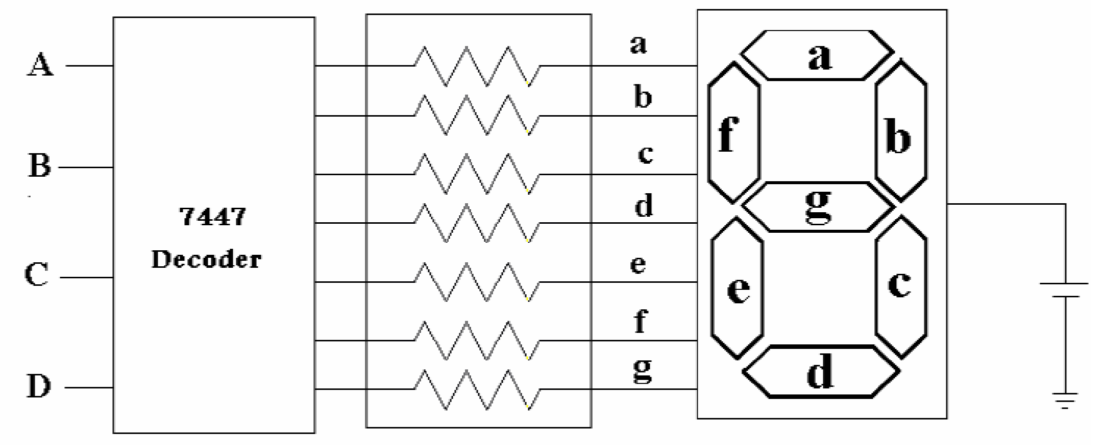
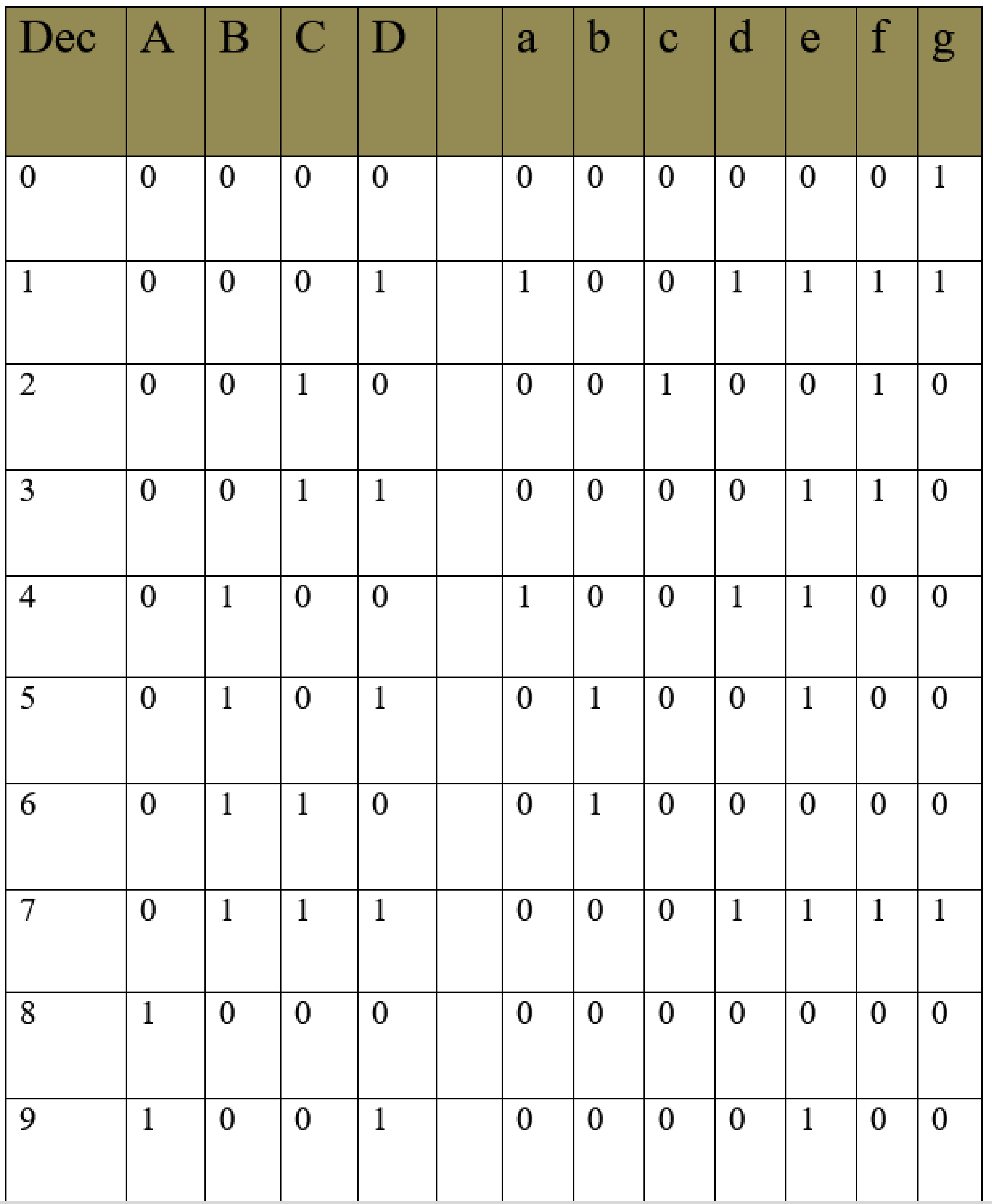


Figure. 4

A decoder is a combinational circuit that converts binary information from n input lines to a maximum of 2n output lines. The input to the decoder is a BCD code and the outputs of the systems are the seven segments a, b, c, d, e, f, and g. For further information and pin connections, consult the specification sheet for decoder and 7-segment units.

First design a combinational circuit which would simulate the decoder function for only the segment "a", of the display. This can be done in the following steps:

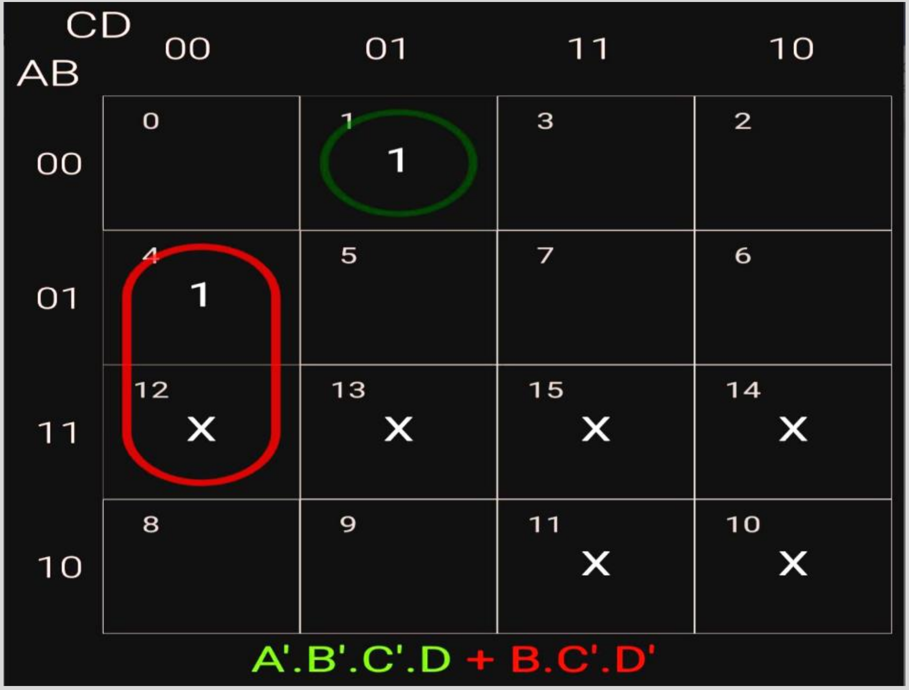
a) Write down the truth table with 4 inputs and 7 outputs (Table 2)



Dec	A	B	C	D		a	b	c	d	e	f	g
0	0	0	0	0		0	0	0	0	0	0	1
1	0	0	0	1		1	0	0	1	1	1	1
2	0	0	1	0		0	0	1	0	0	1	0
3	0	0	1	1		0	0	0	0	1	1	0
4	0	1	0	0		1	0	0	1	1	0	0
5	0	1	0	1		0	1	0	0	1	0	0
6	0	1	1	0		0	1	0	0	0	0	0
7	0	1	1	1		0	0	0	1	1	1	1
8	1	0	0	0		0	0	0	0	0	0	0
9	1	0	0	1		0	0	0	0	1	0	0

b) For only the output "a", obtain a minimum logic function. Realize this function using NAND gates and inverters only. For example if decimal 9 is to be displayed a, b, c, d, f, g must be 0 and the others must be 1 (For common anode type display units), if decimal 5 is to be displayed then a, f, g, c, d must be 0 and the others must be 1.

K Map for a



CD AB	00	01	11	10
00	0	1 1	3	2
01	4 1	5	7	6
11	12 X	13 X	15 X	14 X
10	8	9	11 X	10 X

$A'.B'.C'.D + B.C'.D'$

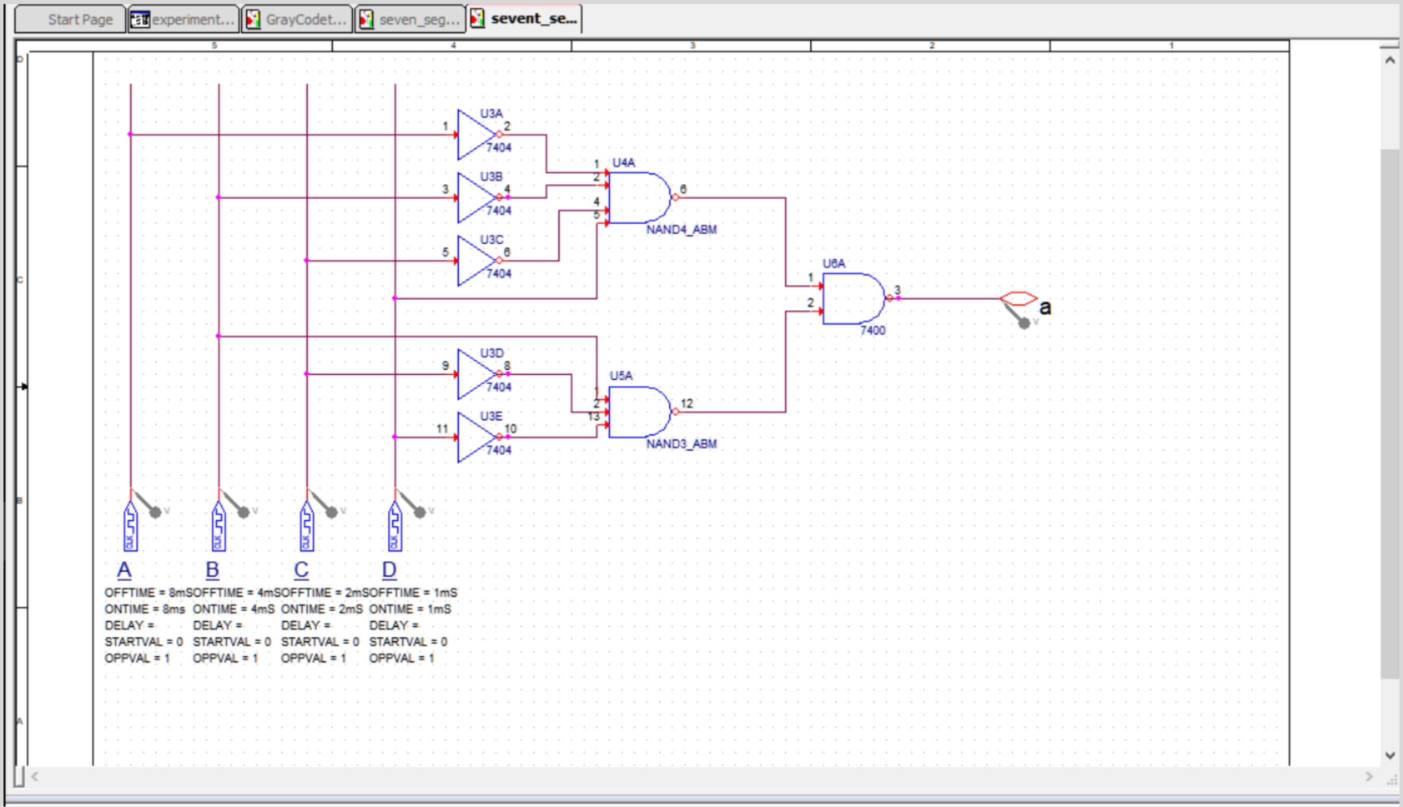
By Using De Morgan’s Law,

$a = A' B'C'D + BC'D'$

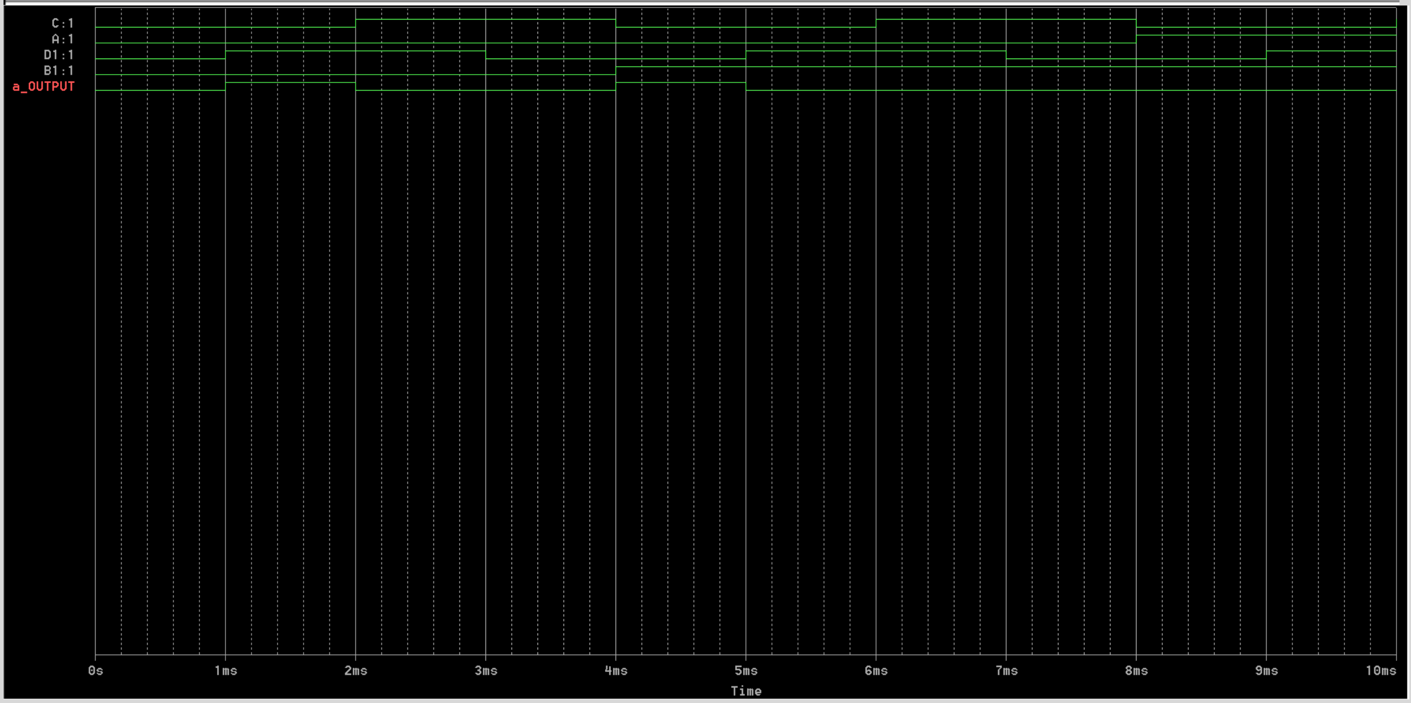
$-= ((A'B'C'D)'. (BC'D')')'$

20BDS0405

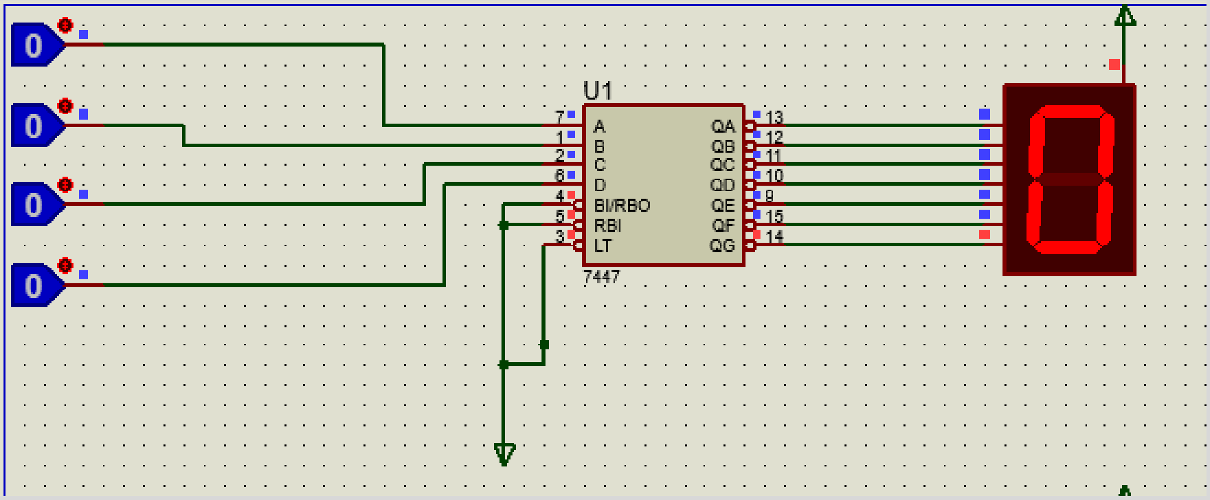
c) Connect the output "a" of your circuit to appropriate input of 7-segment display unit. By applying BCD codes verify the displayed decimal digits for that segment for "a" of the display.

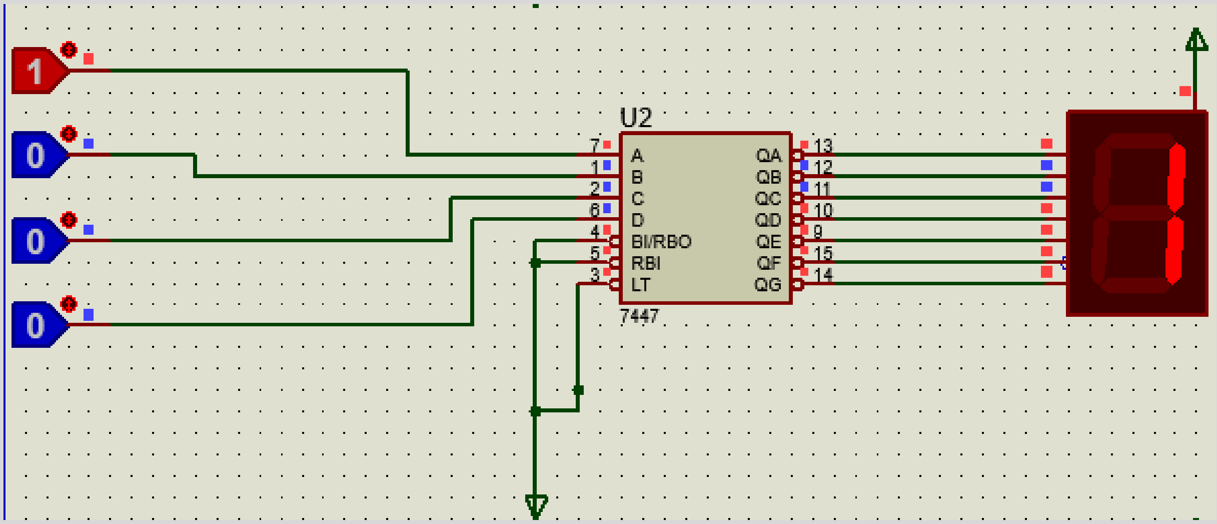


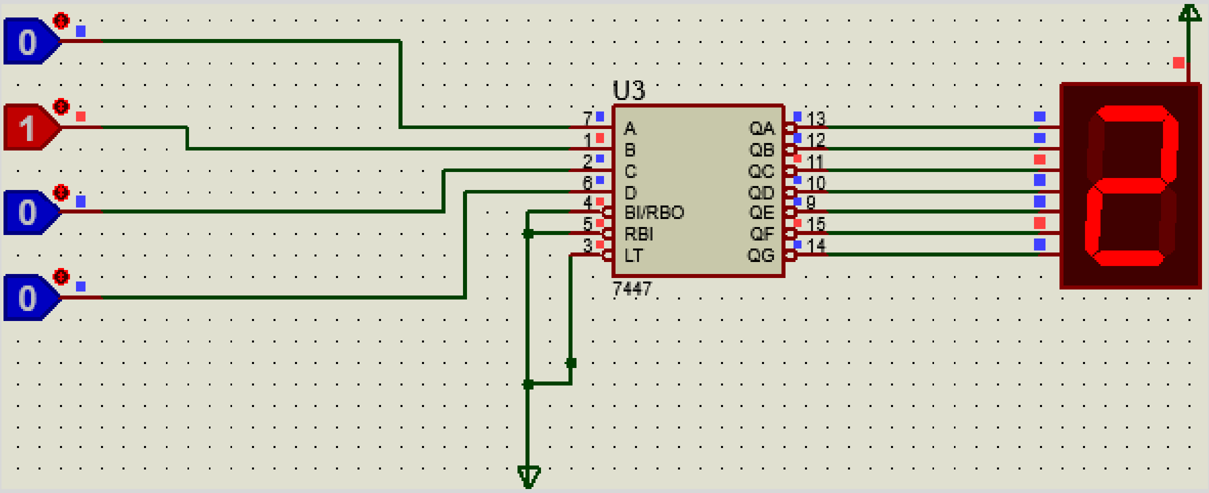
```
OFFTIME = 8mSOFFTIME = 4mSOFFTIME = 2mSOFFTIME = 1mS
ONTIME = 8ms  ONTIME = 4mS  ONTIME = 2mS  ONTIME = 1mS
DELAY =       DELAY =       DELAY =       DELAY =
STARTVAL = 0  STARTVAL = 0  STARTVAL = 0  STARTVAL = 0
OPPVAL = 1    OPPVAL = 1    OPPVAL = 1    OPPVAL = 1
```



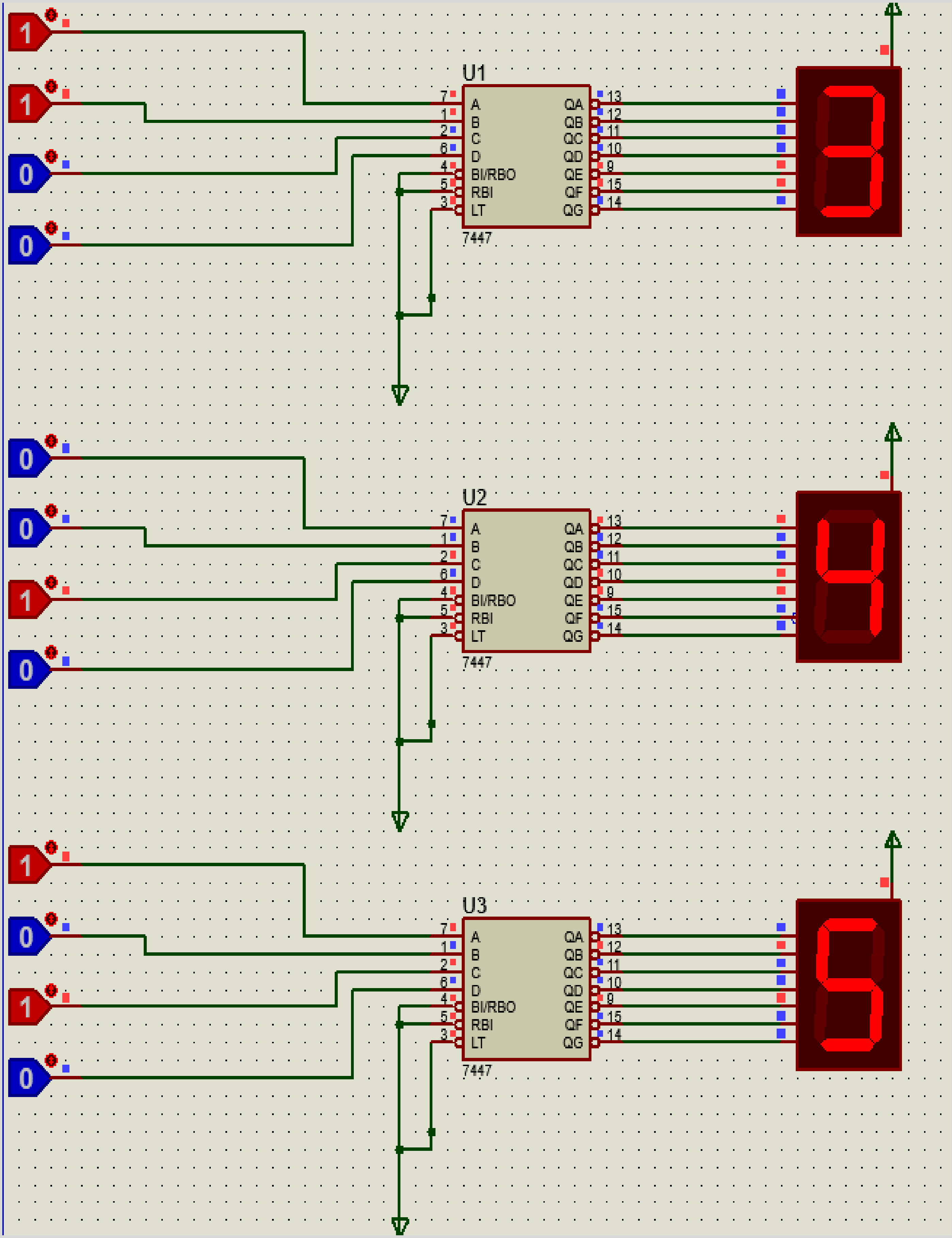
d) Replace your circuit by a decoder IC 7447 for all of the seven segments. Observe the display and record the segments that will light up for invalid inputs sequence.



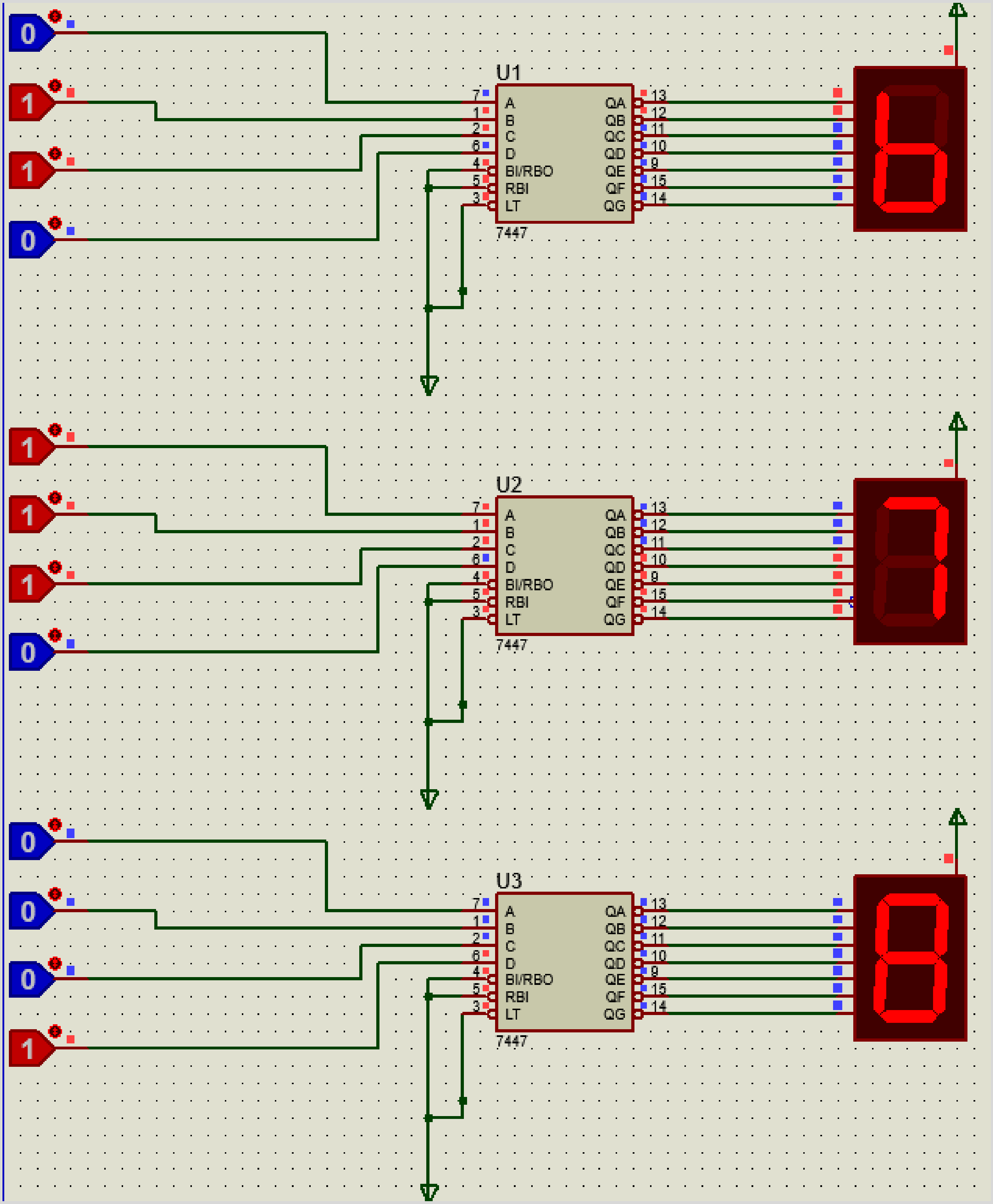


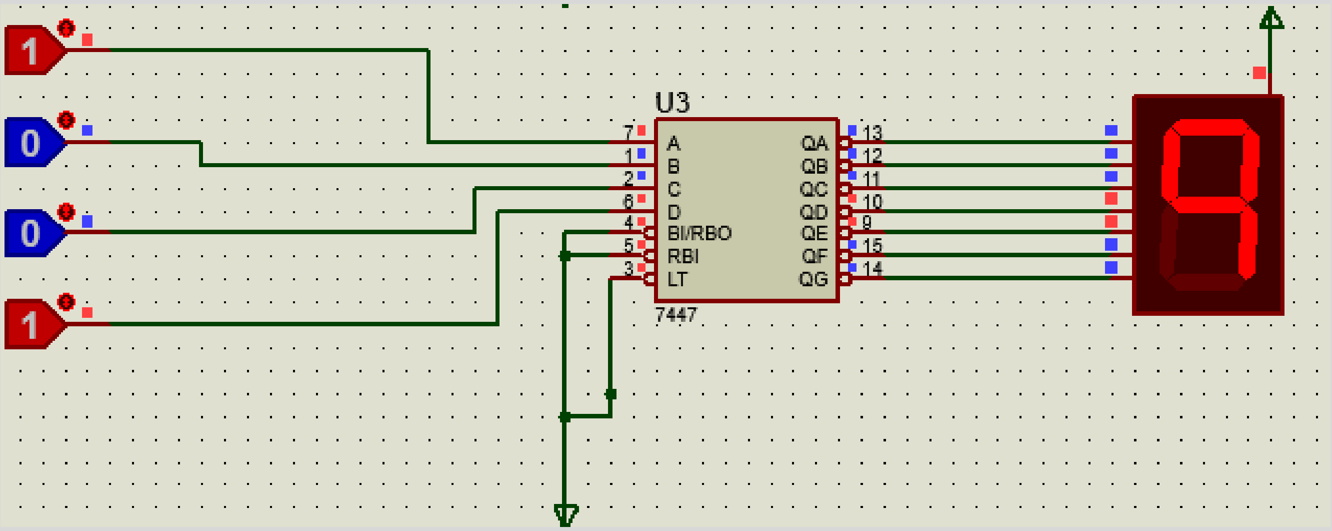


20BDS0405


1
1
0
0
U1
7
1
2
6
4
5
3
A
B
C
D
BI/RBO
RBI
LT
QA
QB
QC
QD
QE
QF
QG
13
12
11
10
9
15
14
7447
0
0
1
0
U2
7447
1
0
1
0
U3
7447

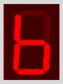
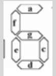
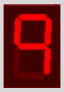
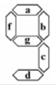
20BDS0405


0
1
1
0
U1
7
1
2
6
4
5
3
A
B
C
D
BI/RBO
RBI
LT
QA
QB
QC
QD
QE
QF
QG
13
12
11
10
9
15
14
7447
1
1
1
0
U2
7
1
2
6
4
5
3
A
B
C
D
BI/RBO
RBI
LT
QA
QB
QC
QD
QE
QF
QG
13
12
11
10
9
15
14
7447
0
0
0
1
U3
7
1
2
6
4
5
3
A
B
C
D
BI/RBO
RBI
LT
QA
QB
QC
QD
QE
QF
QG
13
12
11
10
9
15
14
7447



NOTE:

As observed from the simulation, the actual behavior of the 7447 IC circuit is slightly different from the expected ones.

In case of 6, which shows  instead of  and in case of 9, which shows  instead of .

However, still they are sufficient to distinguish from the remaining numbers

e) Comment on the design if you don't want to see any digit for invalid input sequence.

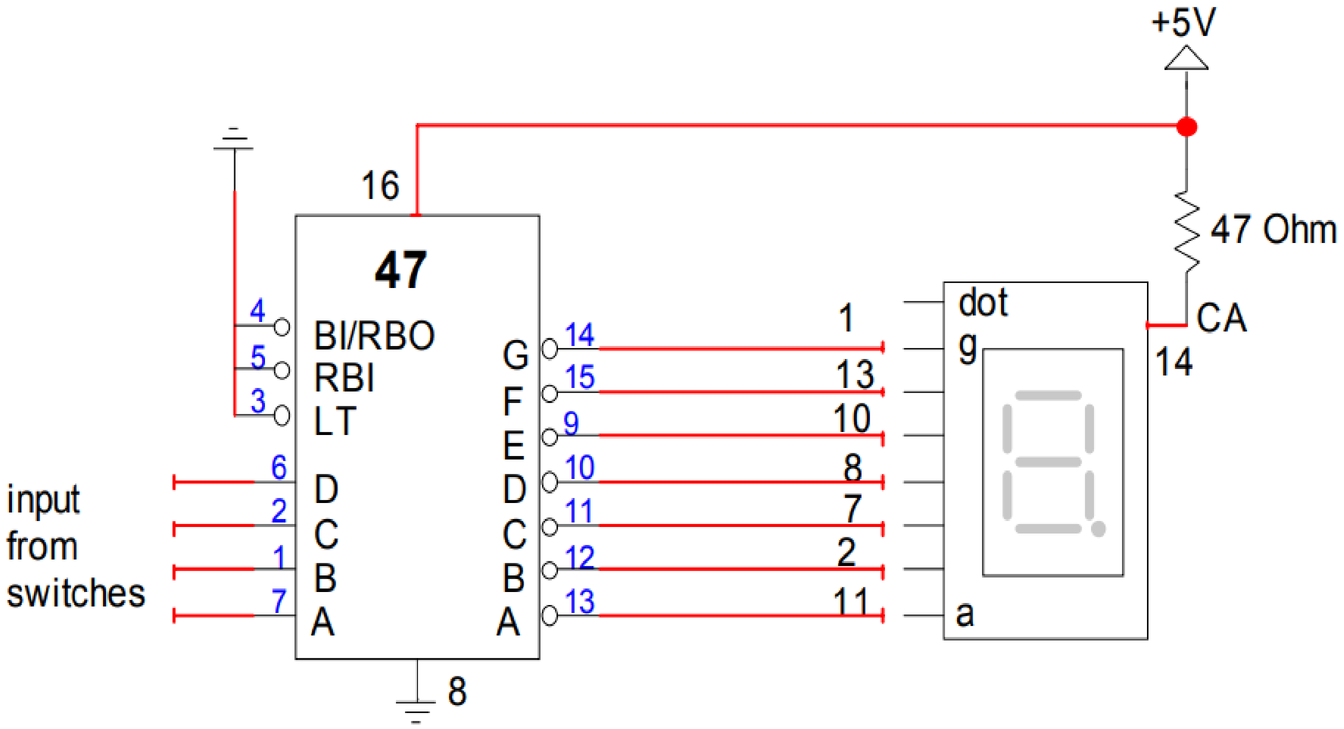
While implementing the 7 segment decoder circuit, the value above 9 are considered as don’t care and hence can have really random output that we don’t care. However, if we want to get rid of that output, we can have the following ways:

I. Limit the simulation time to only 10 units of time(0-9), instead of 16 units of time(0-16)

II. Add a combinational block before the decoder circuit such that it sets all the outputs from the clock to 0.

Table 2

Dec.	BCD				Outputs						
	A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	1	1	0	0	1	1	1	1
2	0	0	0	0	0	0	1	0	0	1	0
3	0	0	1	1	0	0	0	0	1	1	0
4	0	1	0	0	1	0	0	1	1	0	0
5	0	1	0	1	0	1	0	0	1	0	0
6	0	1	1	0	0	1	0	0	0	0	0
7	0	1	1	1	0	0	0	1	1	1	1
8	1	0	0	0	0	0	0	0	0	0	0
9	1	0	0	1	0	0	0	0	1	0	0
10	1	0	1	0	X	X	X	X	X	X	X
11	1	0	1	1	X	X	X	X	X	X	X
12	1	1	0	0	X	X	X	X	X	X	X
13	1	1	0	1	X	X	X	X	X	X	X
14	1	1	1	0	X	X	X	X	X	X	X
15	1	1	1	1	X	X	X	X	X	X	X



BCD-to-Seven Segment Decoder and 7-segment display

Note: In an actual 7-segment display the dot is on the left