ISLAM JIHAD 1191375

Exp 1

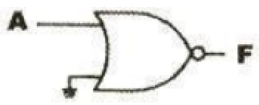


Fig. 1.2 NOR gate used as NOT gate

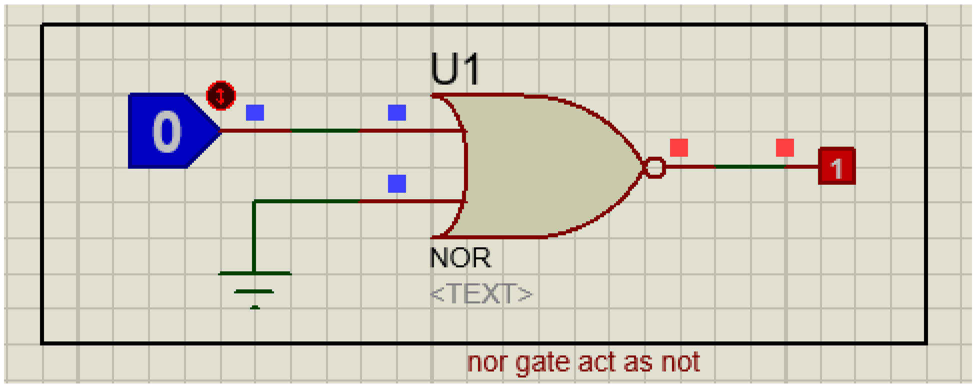
2. Connect inputs A, B to Data Switches SW0, SW1 and output F1 to Logic Indicator L1. Set SW0 to "0", observe states of F1 at SW1= "0" and SW1= "1".

When SW1="0". F1= ____1____

When SW1="1", F1= ____0____

Does the circuit act as a NOT gate? (as shown in Fig. 1.2)

Yes it does



Here it depends on the input only because it's an OR gate with not on the end that make the negative of it.

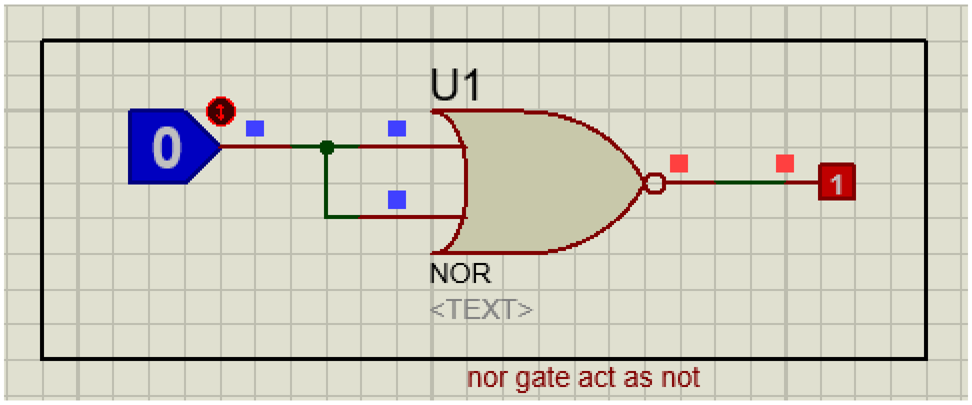
3. Insert a connection clip between A and B as shown in Fig. 1.3 below. Connect A to SW0 and F1 to L1. What is the state of F1 when SW0=0 and SW0=1?

When SW0="0", F1= **1**

When SW0="1", F1= 0

Does the circuit act as a NOT gate?

Yes



When SW0= "0", F3 = ___0___
When SW0= "1", F3 = ___1___

Does the circuit act as a buffer?

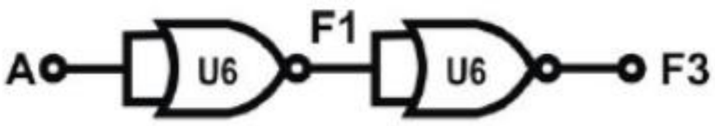
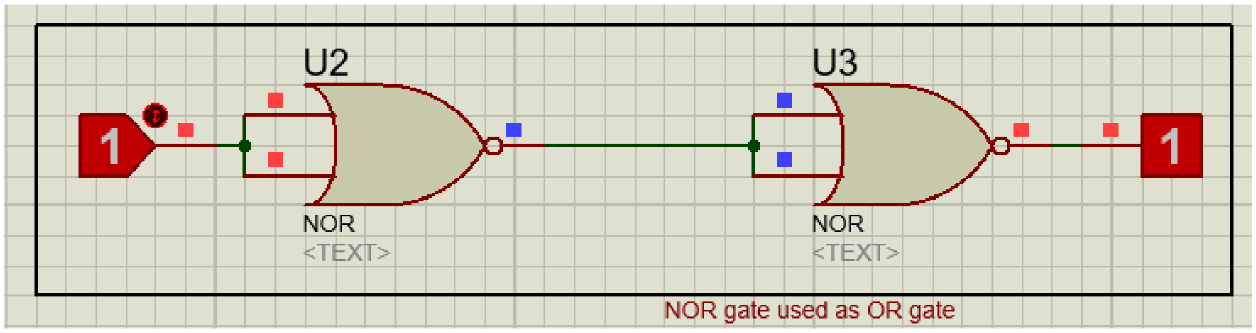
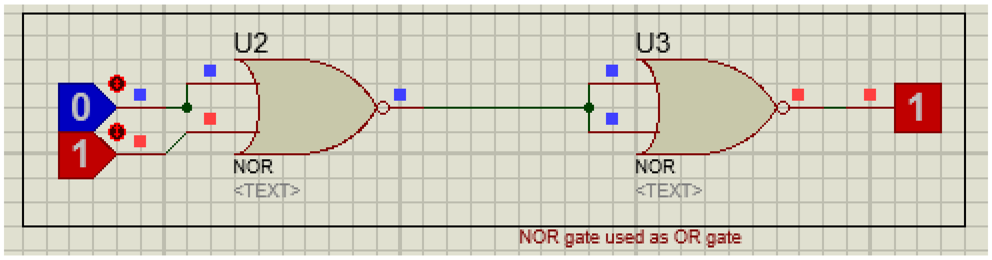


Fig. 1.4 NOR gate used as Buffer

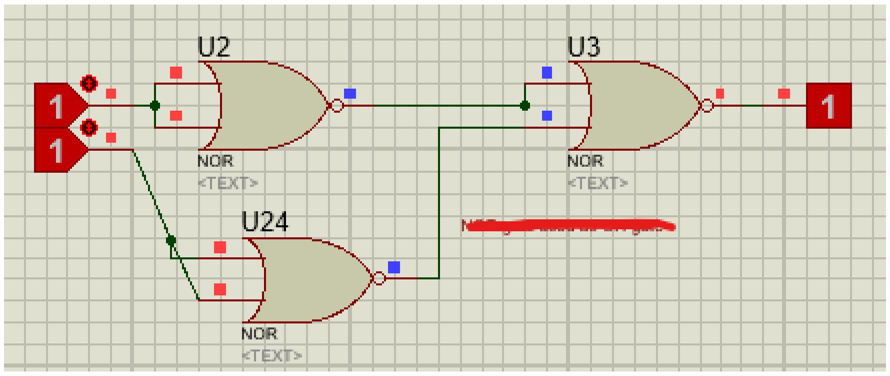
Yes it does





NOR gate used as OR gate, with one input and two

SW1(B)	SW0(A)	F3
0	0	0
0	1	1
1	0	1
1	1	1



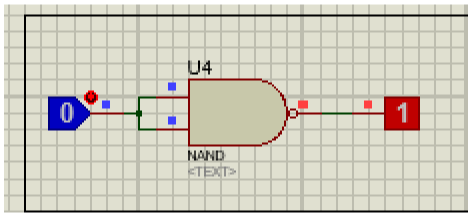
SW1(D)	SW0(A)	F3
0	0	0
0	1	0
1	0	0
1	1	1

Table 1.2

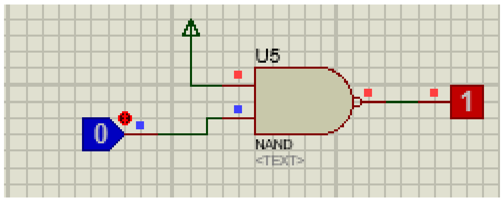
2. Connect input A to A1 and Data Switch SW1 and output F2 to Logic Indicator L1. Observe the output states.
   When SW1= “0”, F2=__1__
   When SW1= “1”, F2=__0__

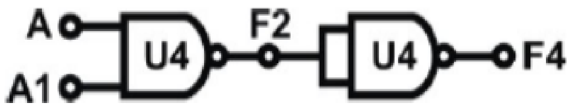
   Does the circuit act as a NOT gate?

yes



3. Connect input A to +5V("1") and remove the connection clip between A and A1 to create the NOT gate shown on the right side of Fig. 1.2.2 (b). Connections remain the same. Observe the output states.
 When SW1= "0", F2=___1___
When SW1= "1", F2=___0___





(b) Equivalent to an AND gate

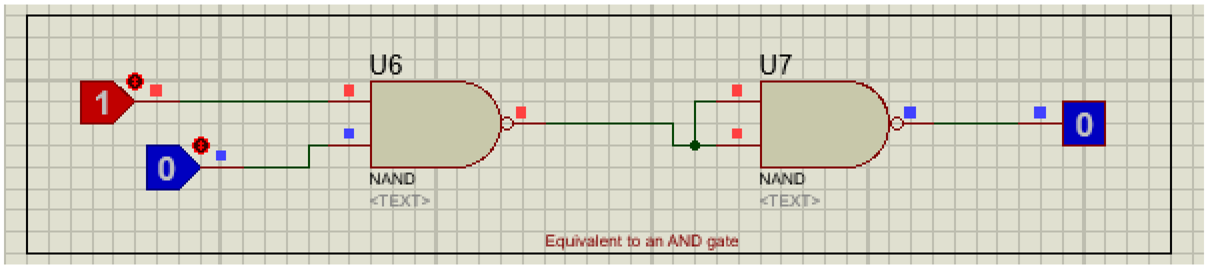
Fig. 1.9 AND gate constructed with NAND gates

5. Follow the input sequences given below and record the outputs in Table 1.3.Does the circuit act as a NOT gate?

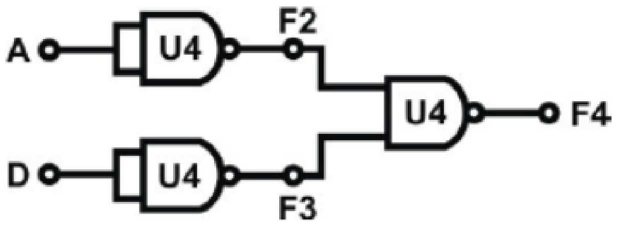
SW2(A1)	SW1(A)	F4
0	**0**	0
0	**1**	0
1	**0**	0
1	**1**	1

Table 1.3

Yes it does



Here I made and gate using NAND gates, and it works the same as demorgan low (B')'= B



(b) Equivalent to an OR gate

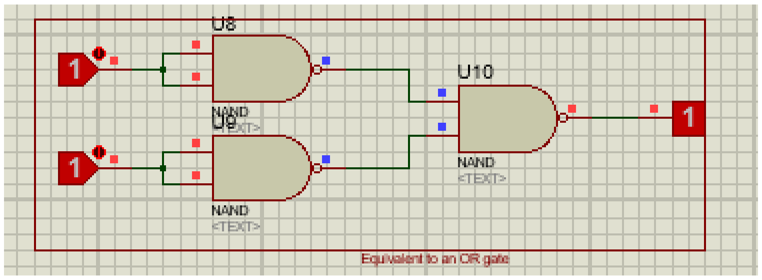
Fig 1.10 OR gate constructed with NAND gates.

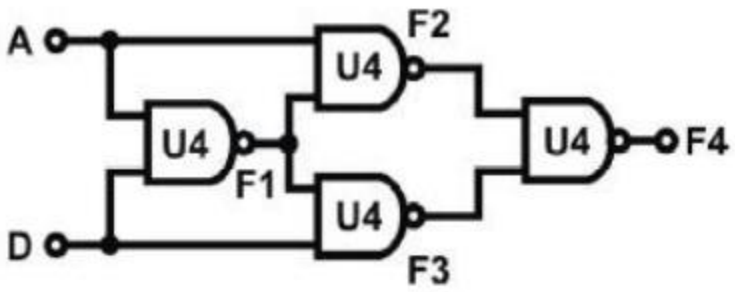
7. Follow the input sequences in Table 1.4 and record the outputs. Does the circuit act as an OR gate (F=A+B)?.

SW2(D)	SW1(A)	F4
0	**0**	0
0	**1**	1
1	**0**	1
1	**1**	1

Table 1.4.

Yes it does





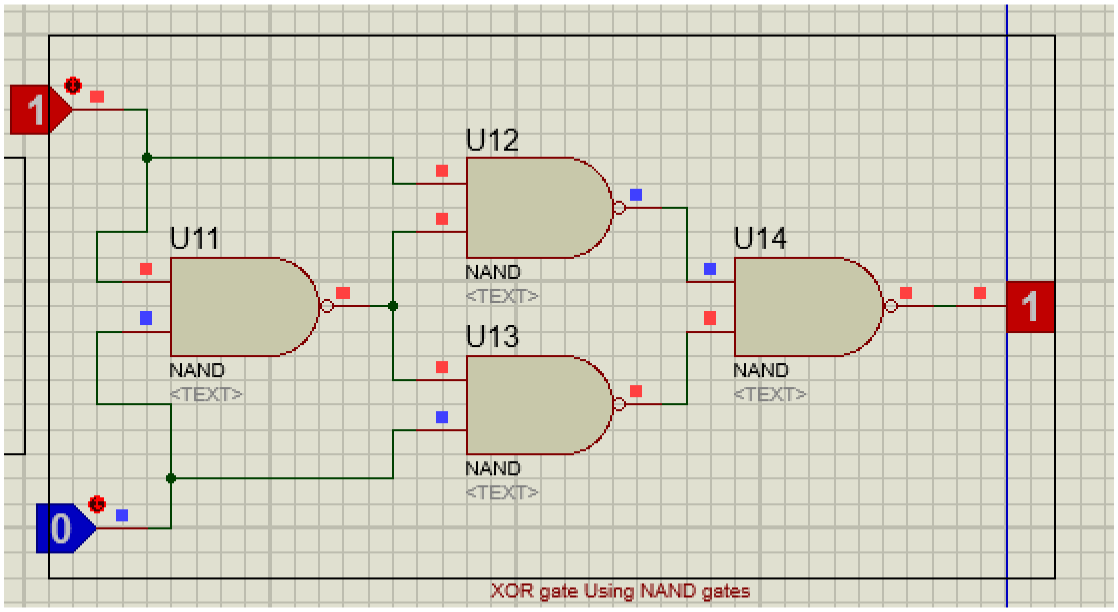
(b) Equivalent Circuit

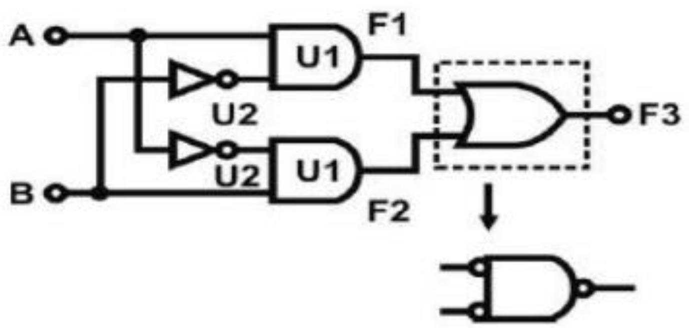
Fig. 1.13 XOR gate Using NAND gates

2. Follow the input sequences for A and D in Table 1.5 and record the outputs.

INPUT		OUTPUT			
D	A	F1	F2	F3	F4
0	0	1	1	1	0
0	1	1	1	0	1
1	0	1	0	1	1
1	1	0	1	1	0

Table 1.5





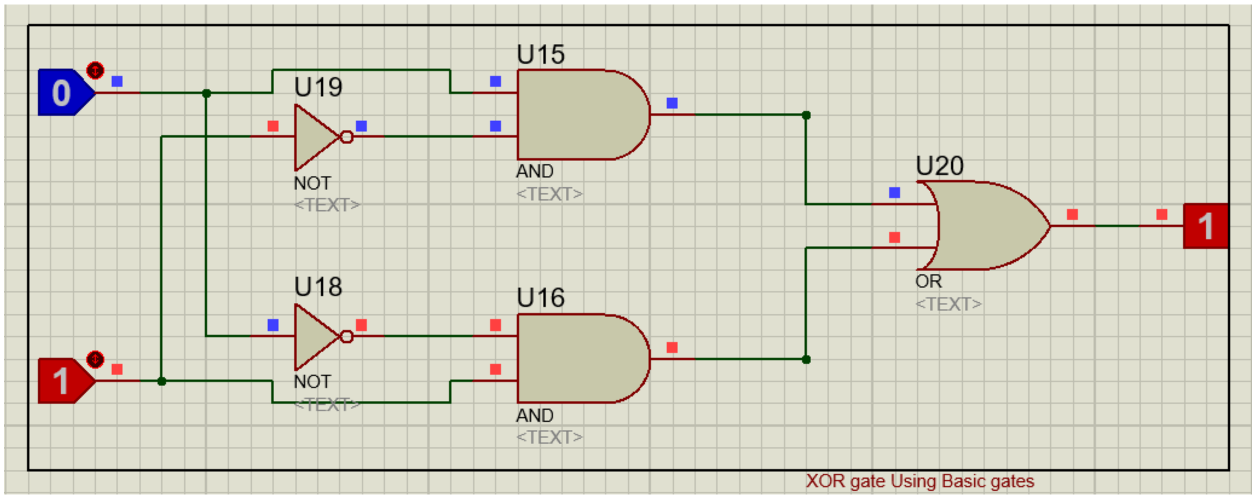
(b) Equivalent Circuit

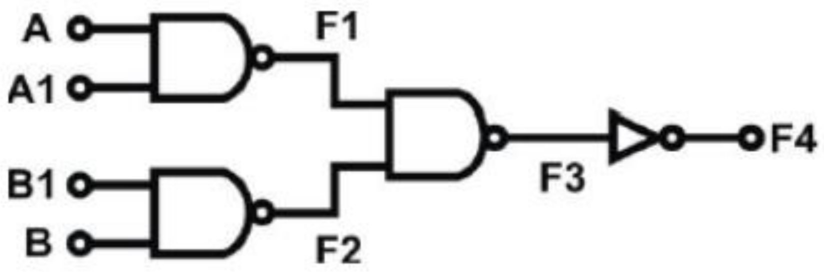
Fig. 1.14 XOR gate Using Basic gates

3. Follow the input sequences for A and B in Table 1.6 and record the outputs.

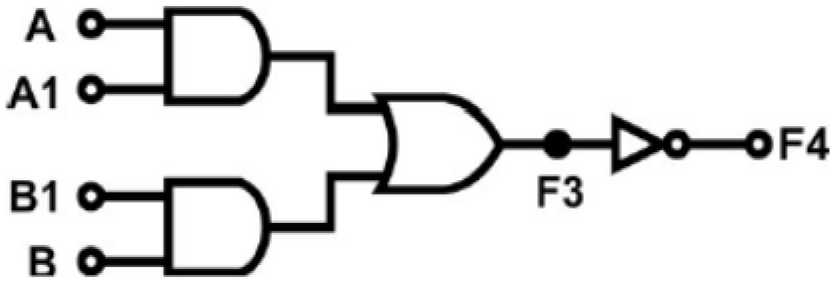
INPUT		OUTPUT		
SW2(B)	SW1(A)	F1	F2	F3
0	0	0	0	0
0	1	0	1	1
1	0	1	0	1
1	1	0	0	0

Table 1.6





(b) Actual circuit



(c)Equivalent circuit

Fig. 1.16: AOI circuit.

2. Connect inputs A, A1, B, B1 to Date Switches SW0, SW1, SW2 and SW3 respectively. Connect outputs F3, F4 to Logic Indicators L1 and L2.
3. Set B×B1to "0", follow the input sequences for A, A1 in Table 1.7 and record the outputs.

A1	A	F3	F4
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Does F3 act as an AND gate between A and A1?

Yes it does

4. WhenB×B1 is "0", does F3 act as an AND gate between A and A1? (F3=A×A1)

Yes it does

5. When$A1\times A$ is "0", follow the input sequences for B, B1 in Table 1.8 and record the outputs.

B1	B	F3	F4
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Table 1.8

Does F3 act as an AND gate between B and B1?

Yes

6. When $A\times A1$ is "0", does F3 act as an AND gate between B and B1?

yes

Does F3 equal to$A\times A1+B\times B1$?

Yes

1. Given the logic system is shown in Fig. 1.17; output of the system F is in"1"state in each of the following condition
   C and D are in the "1" state.
   A, B and D are in the "1" state, C is in "0" state.
   B and D are in the "1" state A and C are in "0" state.
   C and B are in the "1" state A and D are in "0" state.
   C is in the "1" state A, B and D are in the "0" state. Complete the truth table of the above logic system.

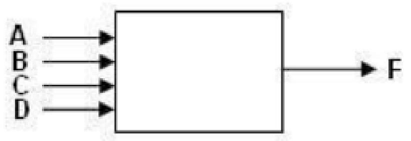
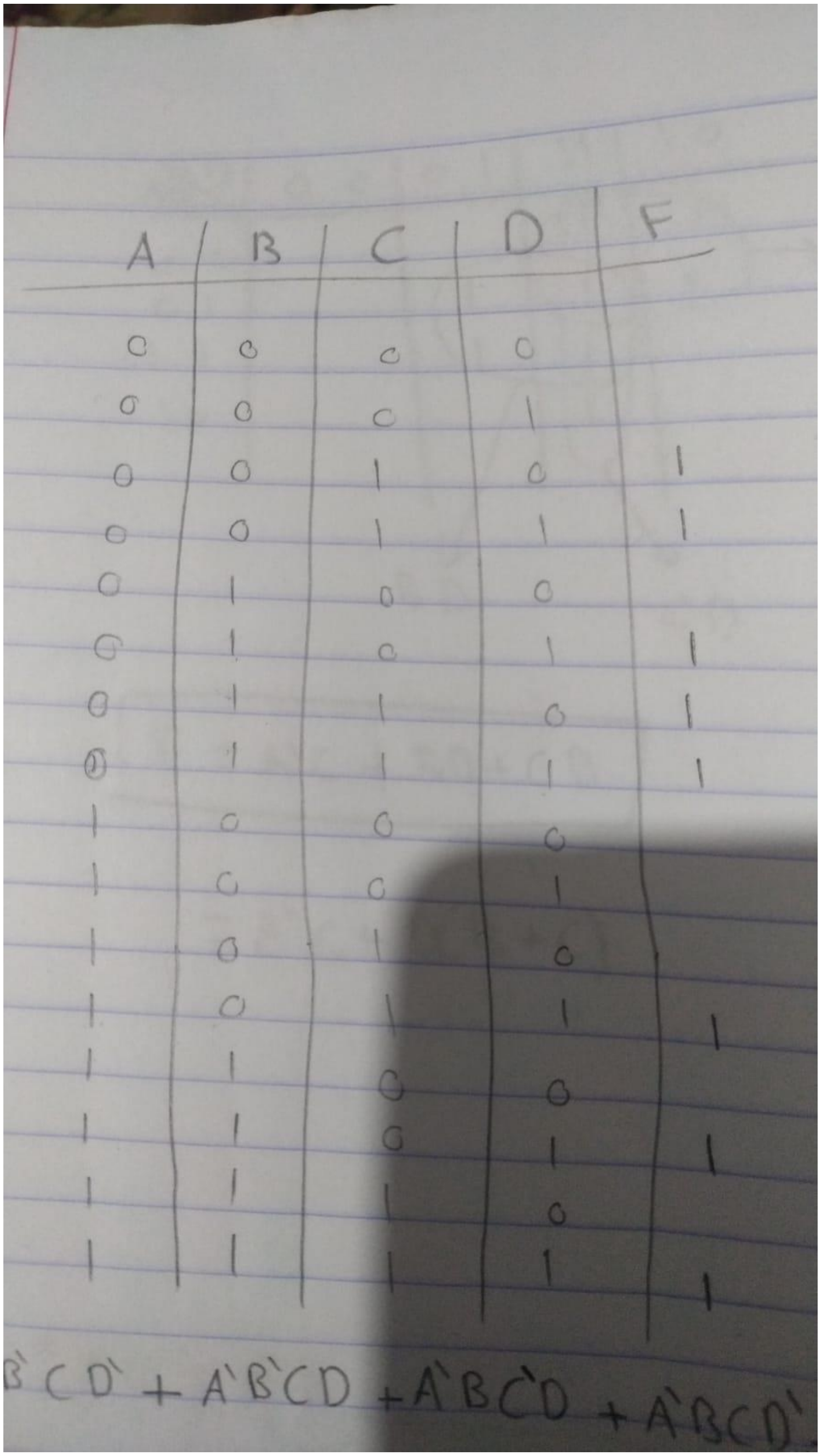


Fig. 1.17 Block diagram



A	B	C	D	F
0	0	0	0	
0	0	0	1	
0	0	1	0	1
0	0	1	1	1
0	1	0	0	
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	1
1	1	0	0	
1	1	0	1	1
1	1	1	0	
1	1	1	1	1

$B'CD' + A'B'CD + A'BC'D + A'BCD'$

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

$A'B'CD' + A'B'CD + A'BC'D + A'BCD' + A'BCD + AB'CD + ABC'D + ABCD$

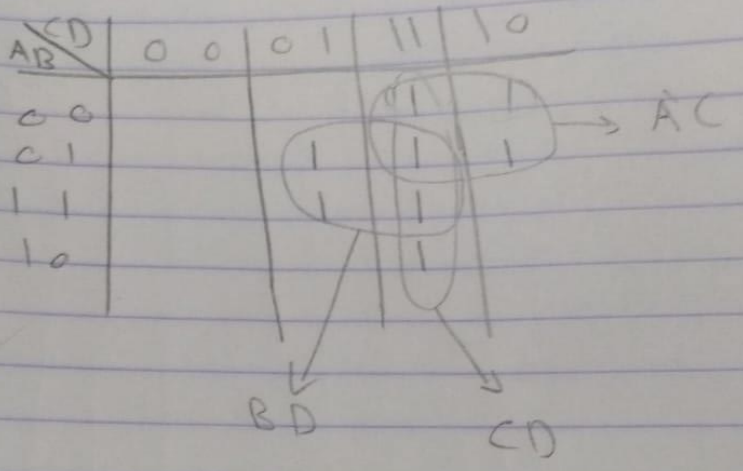
$A'B'CD' + A'BCD' + A'B'CD + AB'CD + A'BC'D + A'BCD + ABC'D + ABCD$

$A'CD'(B'+B) + B'CD(A'+A) + A'BD(C'+C) + ABD(C'+C)$

$A'CD' + B'CD + A'BD + ABD$

$A'CD' + B'CD + BD(A'+A)$

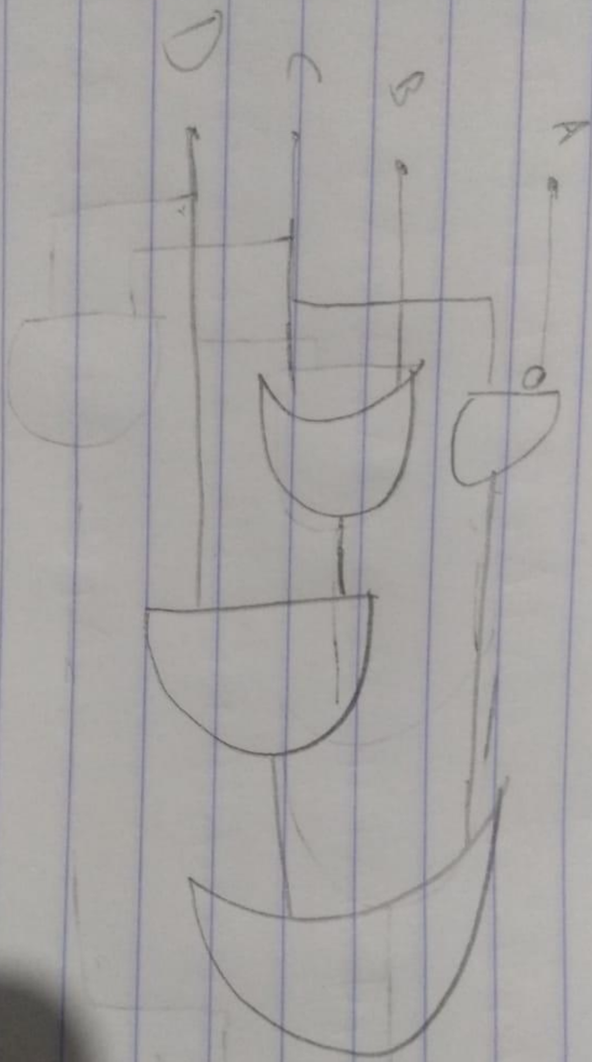
$A'CD' + B'CD + BD$

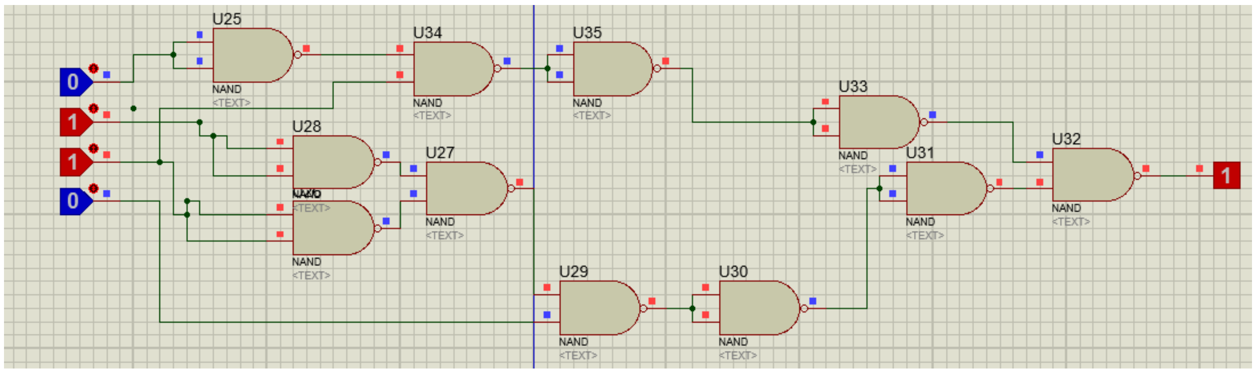


AB \ CD	00	01	11	10
00			1	1
01		1	1	1
11		1	1	
10			1	

$$\boxed{F = A'C + BD + CD}$$

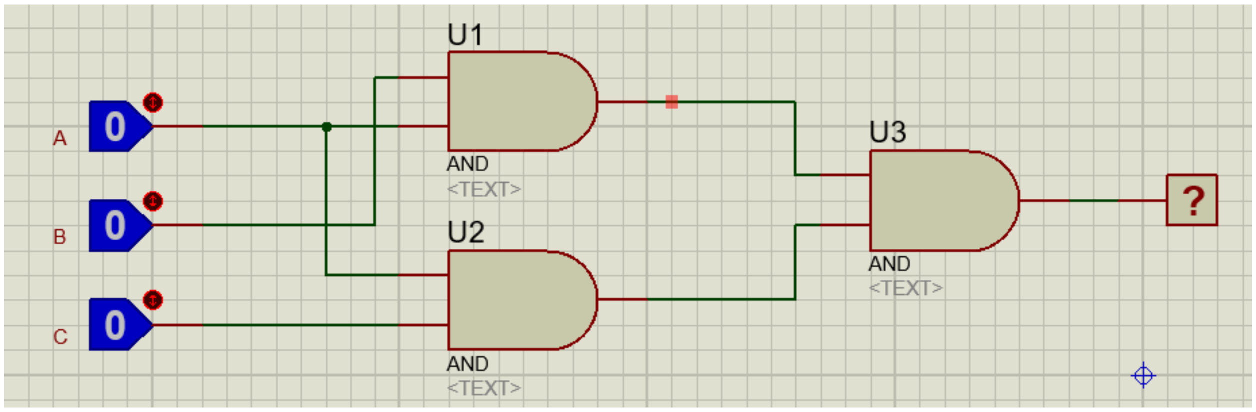
$$= A'C + D(B + C)$$


D
C
B
A



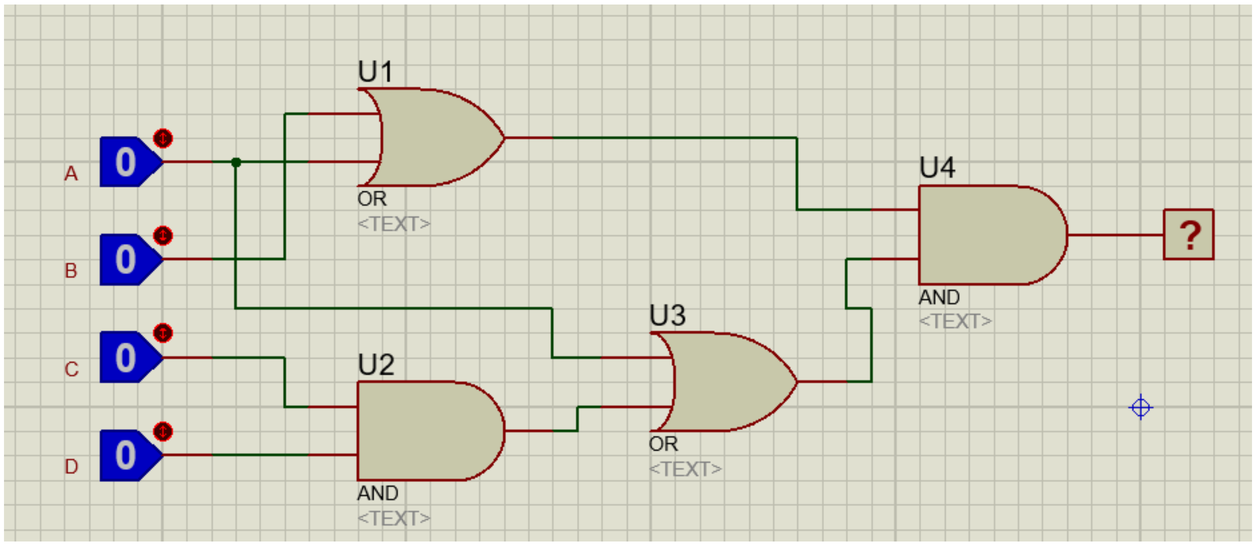
1.6 Post-lab:

1. Draw the logic diagram showing the implementation of the following Boolean equation using "AND" gates F = AB (CA).

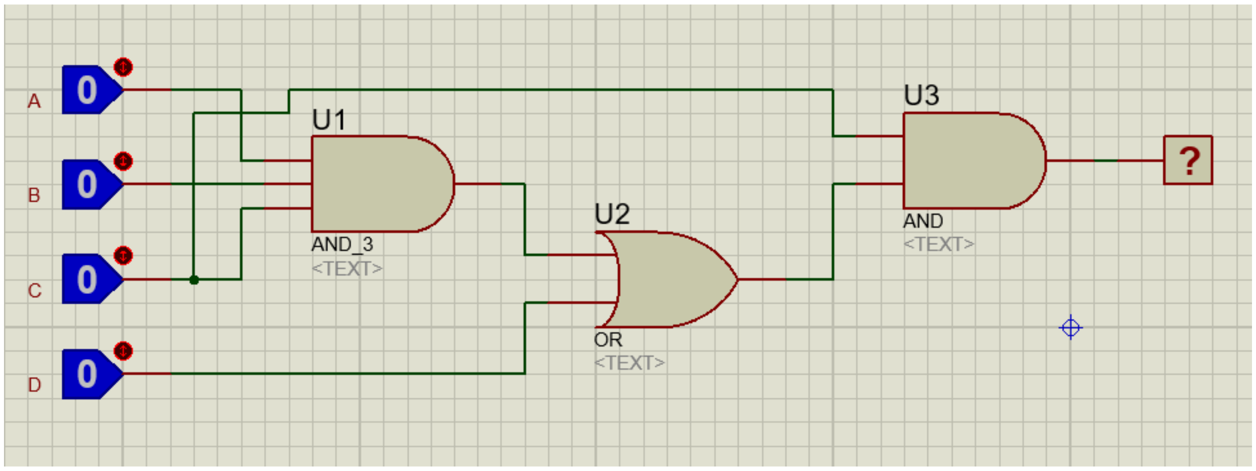


2. Draw the logic diagram of the following Boolean equations

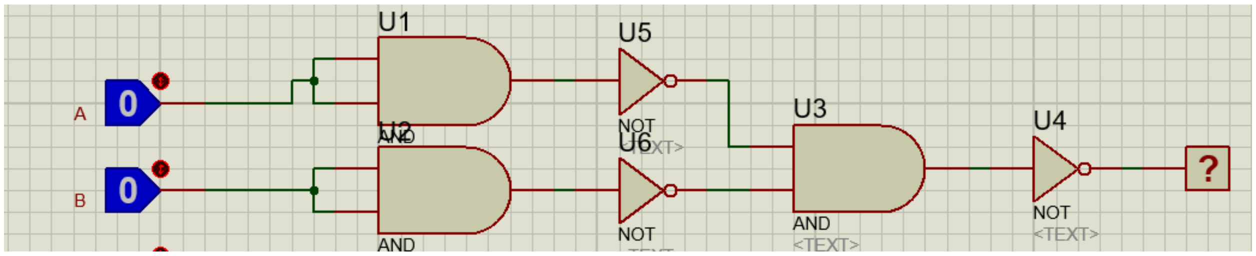
a. F2= (A+B) (CD+A)



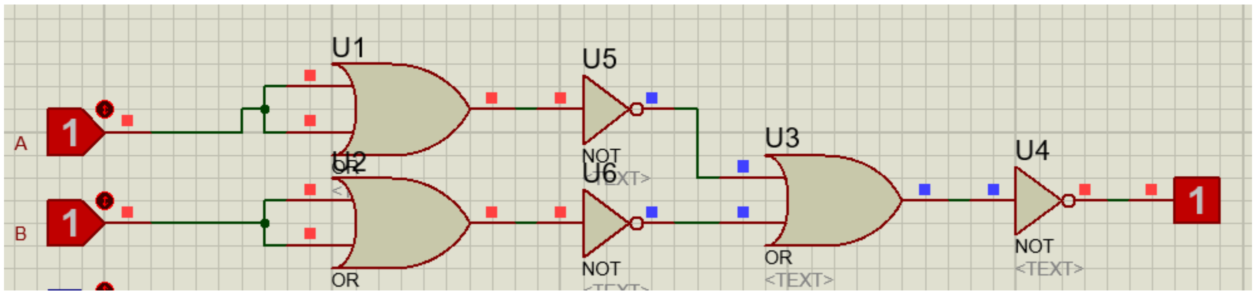
b. F3= (ABC+D) C



3. Implement the OR operation using AND, NOT gate.



4. Implement the AND gate using OR, NOT gate. Draw the logic diagram used in both cases and write Boolean equation.



5. Prove that the equality operation Fl =AB+A’B’ is the inverse of exclusive OR operation F2=AB’+A’B (use Demorgarn’s theorem).

$F_1 = AB + A'B'$

$F_2 = AB' + A'B$

$(AB + A'B') \overset{?}{=} (AB' + A'B)'$

$= (A'+B).(A+B')$

$= A'A + AB + B'B + B'A'$

$= 0 + AB + 0 + A'B'$

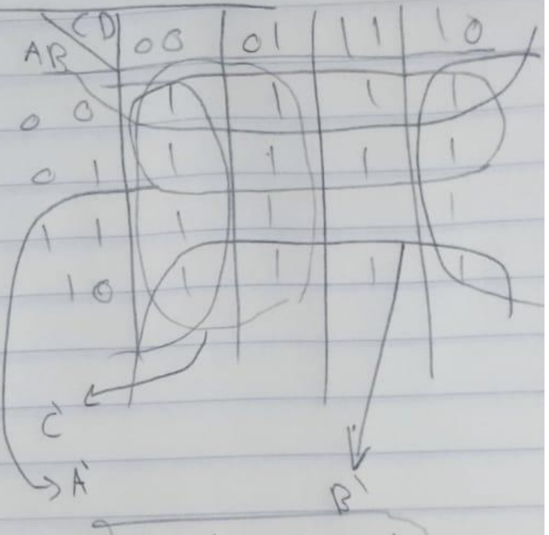
$= AB + A'B'$

[illegible]

6. Suggest a logic diagram which will give a NAND gate with four inputs using two input NAND gates, Implement suggested network.

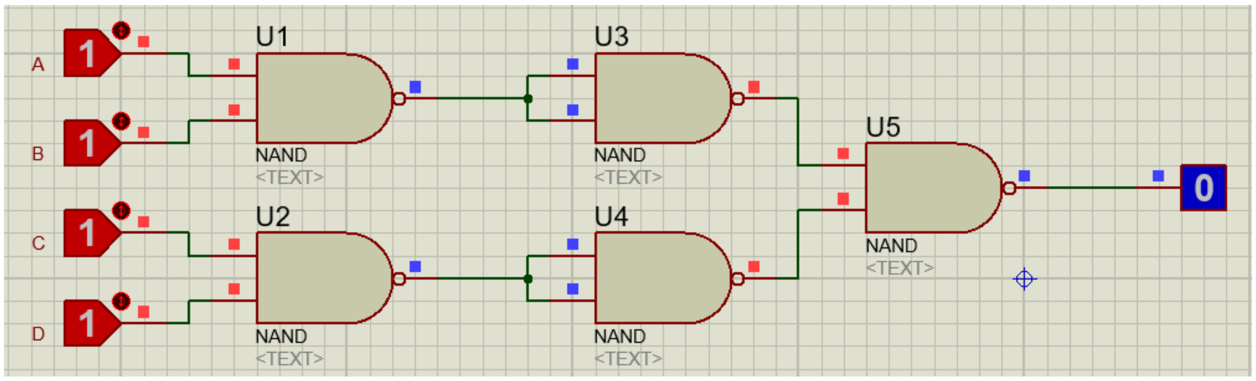
= AB + A B

A	B	C	D	F
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	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0



$\boxed{A' + B' + C' + D'}$

$(AB)' + (CD)'$



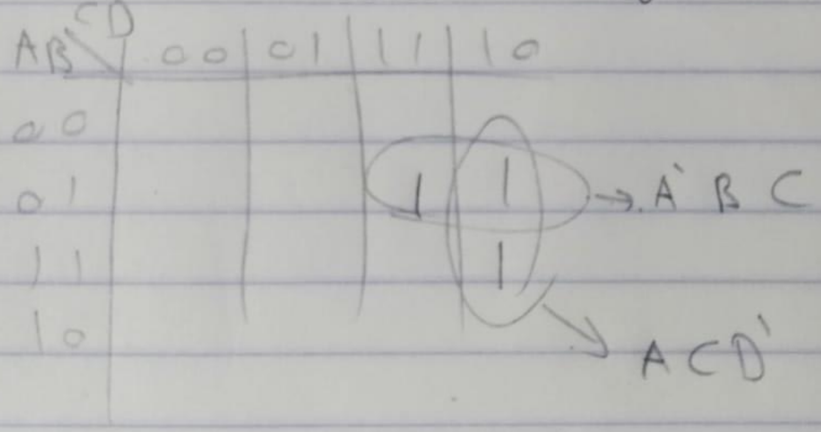
7. Show how is it possible to reduce Boolean expressions by means of karnaugh map

F1 = A'BCD + ABCD' + A'BCD' + ABCD'

$F_1 = A'BCD + ABCD' + A'BCD' + ABCD'$

0111 1110 0110 1110

7 14 6 14

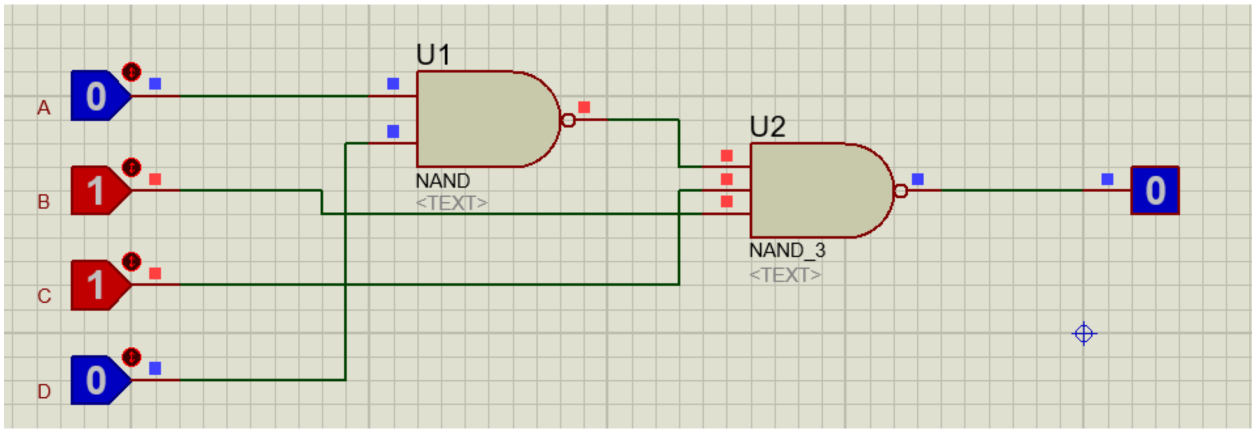


$AB \backslash CD$	00	01	11	10
00				
01			1	1
11				1
10				

$F_1 = A'BC + BCD'$

$BC(A' + D')$

$BC(A \cdot D)'$



F2 = A'B'C'D' + AB'CD' + A'B'CD' + A'BC'D'

Implement the minimal expressions using NAND gates.

$F_2 = A'B'C'D' + AB'CD' + A'B'CD' + A'BC'D'$

$0000 \quad 1010 \quad 0010 \quad 0100$

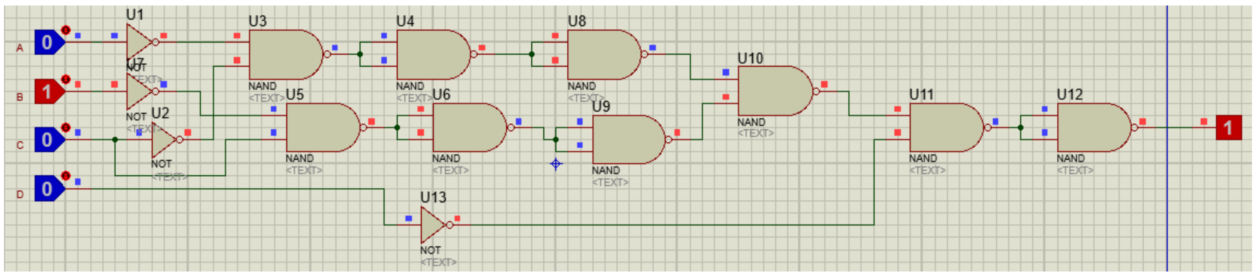
$AB \backslash CD$	00	01	11	10
00	1			1
01	1			
11				
10				1

$A'C'D' \leftarrow$ 01

$B'CD'$

$F_2 = A'C'D' + B'CD'$

$D'(A'C' + CB')$


A
B
C
D
0
1
0
0
U1
U7
U2
U3
U4
U5
U6
U8
U9
U10
U11
U12
U13
NOT
NAND
1