Code: 20CS3301, 20IT3301

PVP 20

II B.Tech- I Semester- Regular / Supplementary Examinations

DECEMBER 2023

FUNDAMENTALS OF DIGITAL LOGIC DESIGN

(COMMON for CSE, IT)

(SCHEME OF VALUATION)

Duration: 3 Hours **Max. Marks : 70**

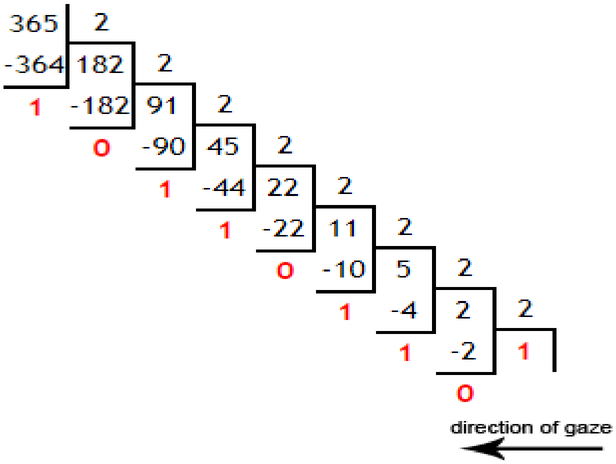
UNIT-I

1) a) Convert the following base conversions using Number System.

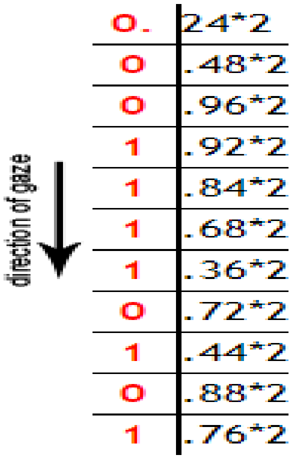
i) $(365.24)_{10}$ to $(?)_2$

Ans:

The Integer part of the number is divided by the base of the new number system:



The Fractional part of the number is multiplied by the base of the new number system:



Final Answer: $(365.24)_{10} = (101101101.0011110101)_2$

ii) $(333.45)_8$ to (? $)_2$

Ans:

Translation from octal to binary like this:

$(333.45)_8$ =3 3 3. 4 5 = 3(=011) 3(=011) 3(=011). 4(=100) 5(=101) = $(011011011.100101)_2$

Final Answer: $(333.45)_8$ = $(011011011.100101)_2$

iii) $(AB7.D)_{16}$ to (? $)_8$

Ans:

Translation from hexadecimal to binary like this:

$(AB7.D)_{16}$ = A B 7. D = A(=1010) B(=1011) 7(=0111). D(=1101) = $(101010110111.1101)_2$

$(AB7.D)_{16}$ = $(101010110111.1101)_2$

Fill in the number with missing zeros on the right

let's make a direct translation from binary to post-binary like this:

$(101010110111.110100)_2$ = 101 010 110 111. 110 100 = 101(=5) 010(=2) 110(=6) 111(=7). 110(=6) 100(=4) = $(5267.64)_8$

Final Answer: $(AB7.D)_{16}$ = $(5267.64)_8$

1) b) i) Convert the $(1110001.10001)_2$ binary number to decimal, hexadecimal and octal numbers.

Ans:

1(b) Converting binary to decimal

(i) $(\underset{2^6}{1}\underset{2^5}{1}\underset{2^4}{1}\underset{2^3}{0}\underset{2^2}{0}\underset{2^1}{0}\underset{2^0}{1}\cdot\underset{2^{-1}}{1}\underset{2^{-2}}{0}\underset{2^{-3}}{0}\underset{2^{-4}}{0}\underset{2^{-5}}{1})_2 = (\quad)_{10}.$

$\Rightarrow 1\times2^6+1\times2^5+1\times2^4+0\times2^3+0\times2^2+0\times2^1+1\times2^0+$
$1\times2^{-1}+0\times2^{-2}+0\times2^{-3}+0\times2^{-4}+1\times2^{-5}$

$\Rightarrow 1\times64+1\times32+1\times16+0\times8+0\times4+0\times2+1\times1+1\times0\cdot5$
$+0\times0\cdot25+0\times0\cdot125+0\times0\cdot0625+1\times0\cdot03125$

$\Rightarrow 64+32+16+0+0+0+1+0\cdot5+0+0+0+0\cdot03125$

$\Rightarrow (113\cdot53125)_{10}.$

$$\boxed{\therefore\ (1110001\cdot10001)_2=(113\cdot53125)_{10}.}$$

Converting binary to Hexadecimal :

$(1110001\cdot10001)_2 = (\,.\,)_{16}.$

$\Rightarrow \underbrace{\underline{0}111}_{7}|\underbrace{0001}_{1}\cdot\underbrace{1000}_{8}|\underbrace{1\underline{0}\,\underline{0}\,\underline{0}}_{8}$

$$\Rightarrow \boxed{(1110001\cdot10001)_2=(71\cdot88)_{16}.}$$

Converting binary to octal :

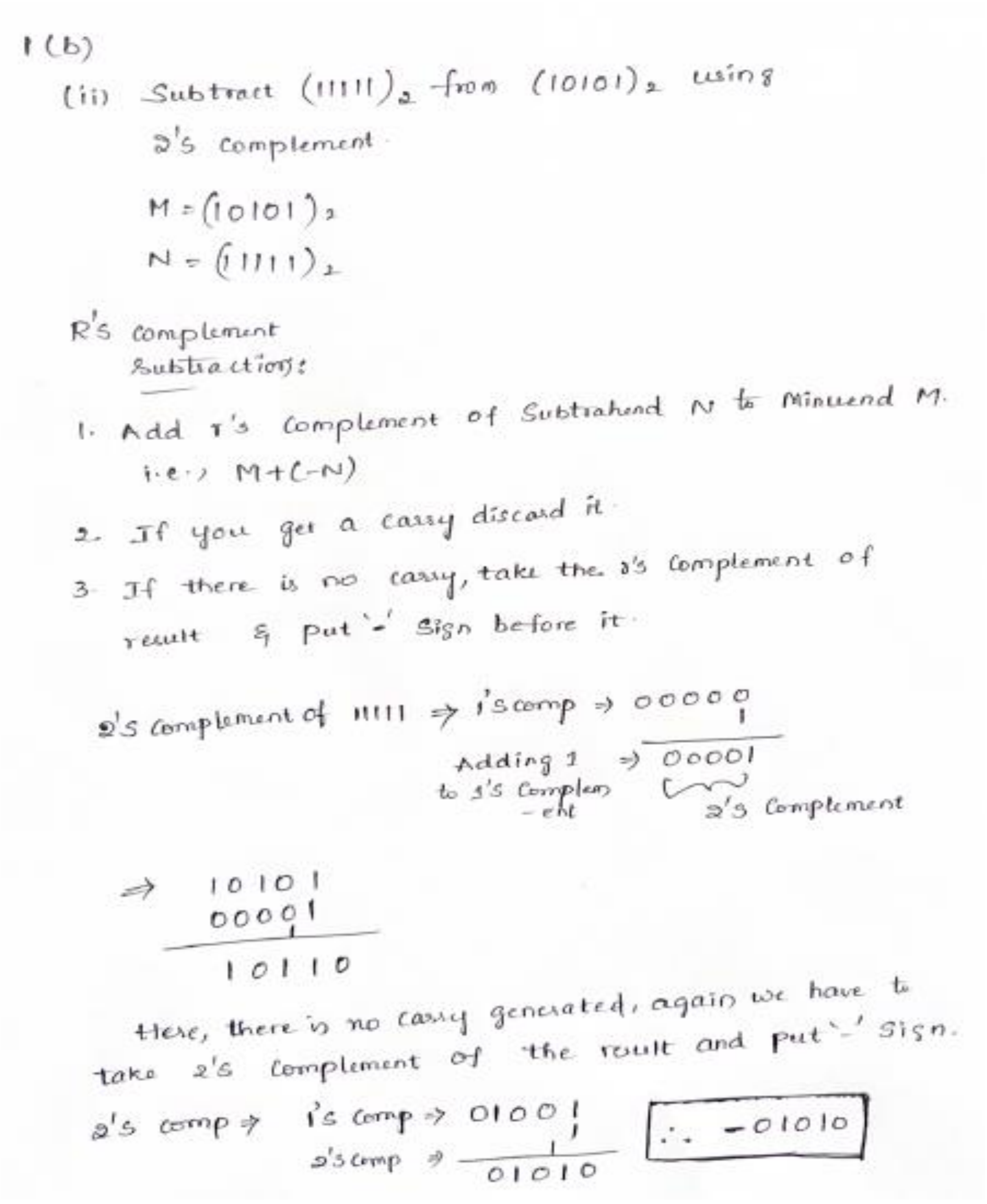
$(1110001\cdot10001)_2 = (\ )_8.$

$\Rightarrow \underbrace{\underline{0}\,\underline{0}1}_{1}|\underbrace{110}_{6}|\underbrace{001}_{1}\cdot\underbrace{100}_{4}|\underbrace{01\underline{0}}_{2}$

$$\Rightarrow \boxed{(1110001\cdot10001)_2=(161\cdot42)_8}$$

ii) Subtract $(11111)_2$ from $(10101)_2$ using 2's complement method.

Ans:



1 (b)

(ii) Subtract $(11111)_2$ from $(10101)_2$ using 2's complement

$M = (10101)_2$

$N = (11111)_2$

R's complement Subtraction:

1. Add r's complement of Subtrahend N to Minuend M. i.e., M+(-N)
2. If you get a carry discard it.
3. If there is no carry, take the 2's complement of result & put '-' sign before it.

2's complement of 11111 ⇒ 1's comp ⇒ 00000

```
  00000
      1
  -----
  00001
```

Adding 1 to 1's Complement ⇒ 00001 — 2's Complement

```
⇒  10101
   00001
   -----
   10110
```

Here, there is no carry generated, again we have to take 2's complement of the result and put '-' sign.

2's comp ⇒ 1's comp ⇒ 01001

```
   01001
       1
   -----
   01010
```

2's comp ⇒ 01010

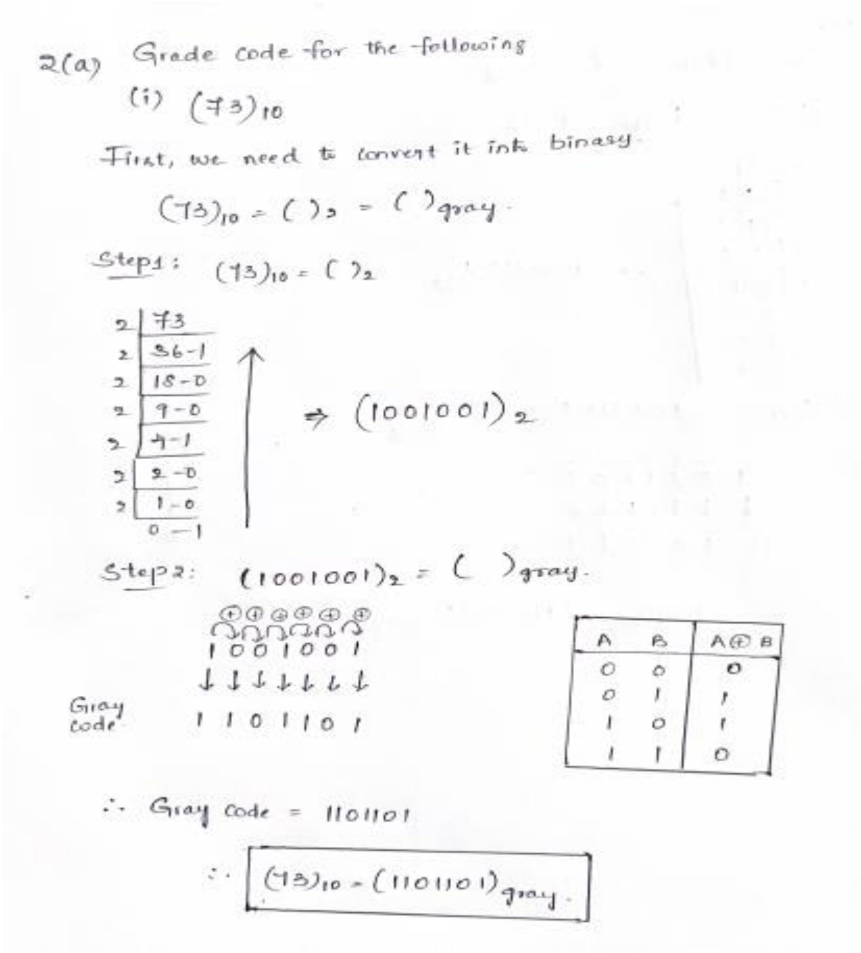
$\boxed{\therefore\ -01010}$

(OR)

2) a) Show the Gray code for the following decimal numbers.

i) $(73)_{10}$

Ans:



2(a) Grade code for the following

(i) $(73)_{10}$

First, we need to convert it into binary.

$(73)_{10} = (\ )_2 = (\ )_{gray}$.

Step1 : $(73)_{10} = (\ )_2$

```
2 | 73
2 | 36-1
2 | 18-0
2 | 9-0
2 | 4-1
2 | 2-0
2 | 1-0
    0-1
```

$\Rightarrow (1001001)_2$

Step2: $(1001001)_2 = (\ )_{gray}$.

```
       ⊕ ⊕ ⊕ ⊕ ⊕ ⊕
       1 0 0 1 0 0 1
       ↓ ↓ ↓ ↓ ↓ ↓ ↓
Gray
code   1 1 0 1 1 0 1
```

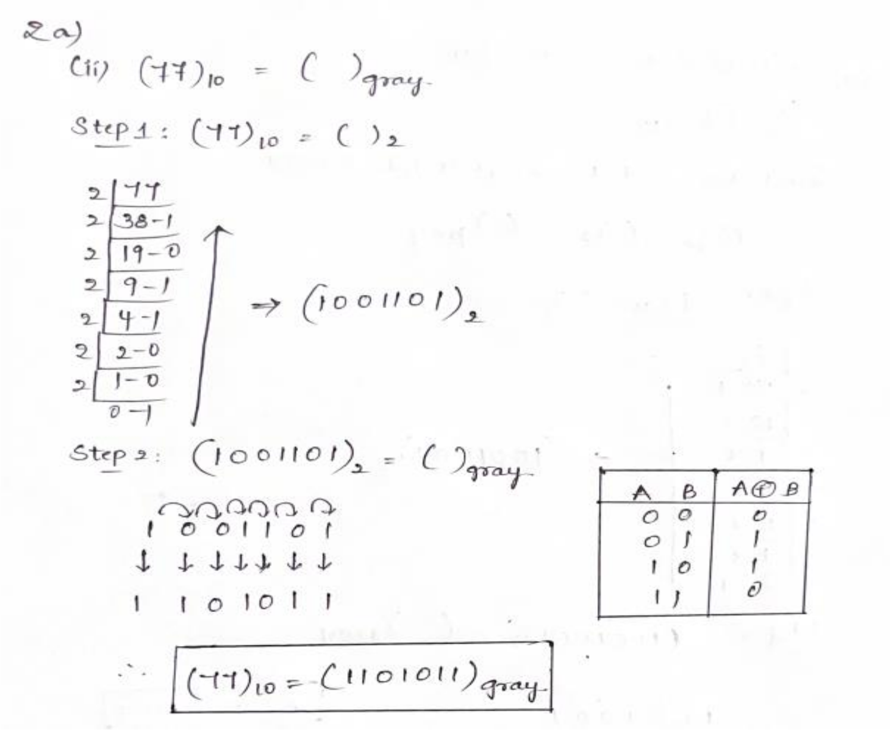
A	B	A ⊕ B
0	0	0
0	1	1
1	0	1
1	1	0

∴ Gray code = 1101101

∴ $\boxed{(73)_{10} = (1101101)_{gray}}$

ii) $(77)_{10}$

Ans:



2a)

(ii) $(77)_{10} = (\ )_{gray}$

Step 1 : $(77)_{10} = (\ )_2$

```
2 | 77
2 | 38-1
2 | 19-0
2 | 9-1
2 | 4-1
2 | 2-0
2 | 1-0
    0-1
```

$\Rightarrow (1001101)_2$

Step 2: $(1001101)_2 = (\ )_{gray}$

```
1 0 0 1 1 0 1
↓ ↓ ↓ ↓ ↓ ↓ ↓
1 1 0 1 0 1 1
```

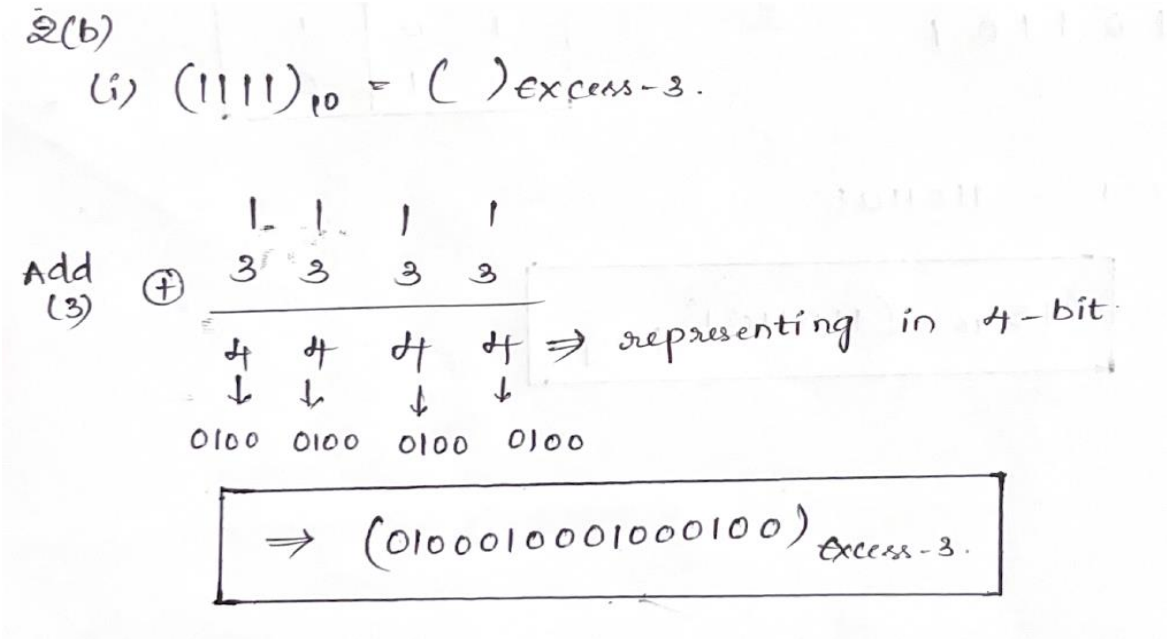
A	B	A ⊕ B
0	0	0
0	1	1
1	0	1
1	1	0

∴ $\boxed{(77)_{10} = (1101011)_{gray}}$

2) b) Show the Excess 3 code for the following decimal numbers.

i) $(1111)_{10}$

Ans:



2(b)

(i) $(1111)_{10} = (\ )_{Excess-3}$.

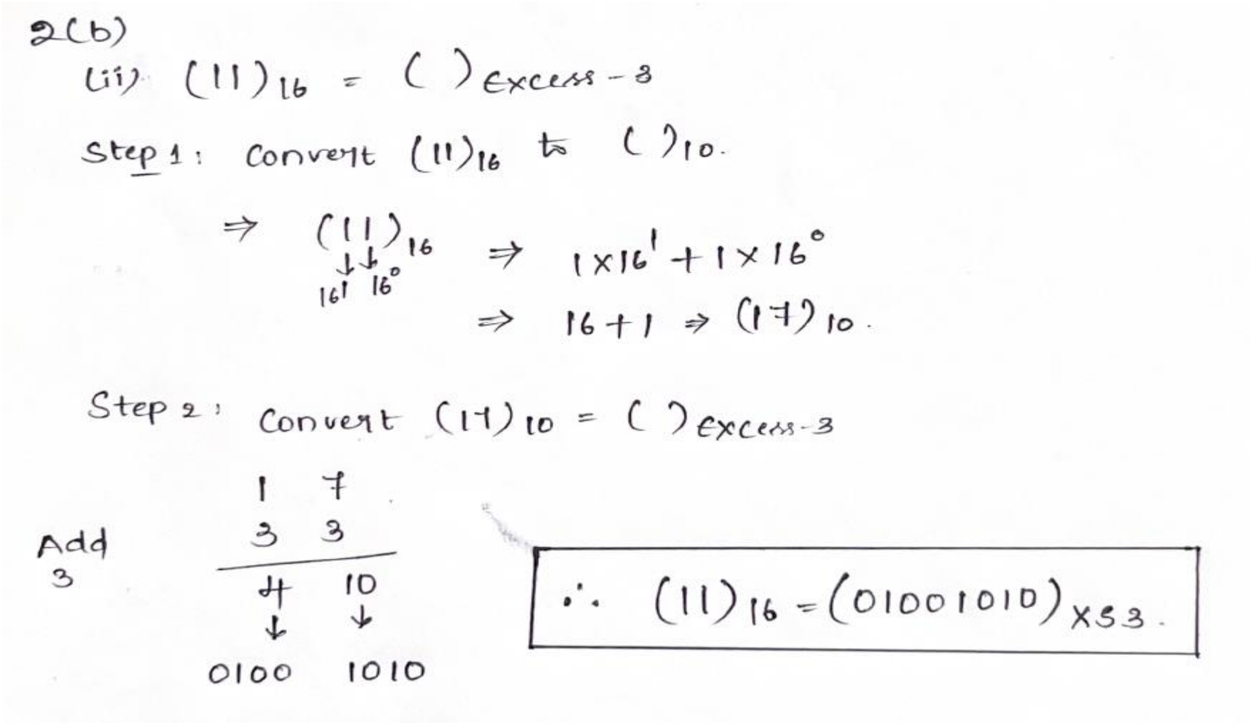
$$\begin{array}{lcccc} & 1 & 1 & 1 & 1 \\ \text{Add (3)} \ \oplus & 3 & 3 & 3 & 3 \\ \hline & 4 & 4 & 4 & 4 \\ & \downarrow & \downarrow & \downarrow & \downarrow \\ & 0100 & 0100 & 0100 & 0100 \end{array}$$

⇒ representing in 4-bit

$$\boxed{\Rightarrow (0100010001000100)_{Excess-3}.}$$

ii) $(11)_{16}$

Ans:



2(b)

(ii) $(11)_{16} = (\ )_{Excess-3}$

Step 1: Convert $(11)_{16}$ to $(\ )_{10}$.

$\Rightarrow (11)_{16}$ (16^1 16^0) $\Rightarrow 1\times16^1 + 1\times16^0$

$\Rightarrow 16+1 \Rightarrow (17)_{10}$.

Step 2: Convert $(17)_{10} = (\ )_{Excess-3}$

$$\begin{array}{lcc} & 1 & 7 \\ \text{Add 3} & 3 & 3 \\ \hline & 4 & 10 \\ & \downarrow & \downarrow \\ & 0100 & 1010 \end{array}$$

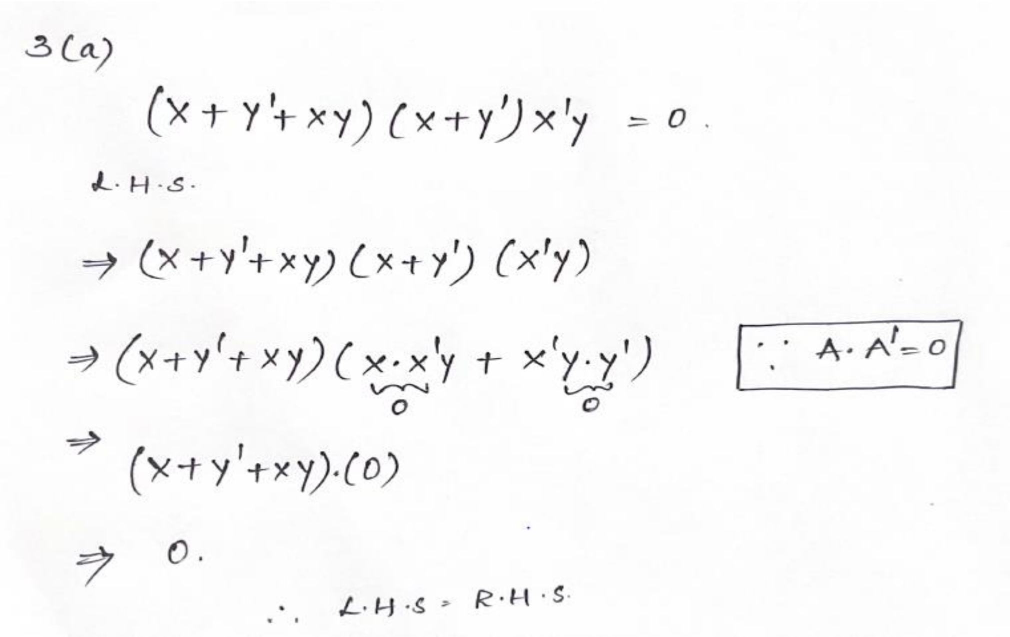
$$\boxed{\therefore (11)_{16} = (01001010)_{XS3}.}$$

UNIT-II

3) a) Using Boolean laws verify the following equation.

(X+Y’+XY)(X+Y’)X’Y = 0

Ans:



3(a)

$$(x+y'+xy)(x+y')x'y = 0.$$

L.H.S.

$\Rightarrow (x+y'+xy)(x+y')(x'y)$

$\Rightarrow (x+y'+xy)(\underbrace{x\cdot x'}_{0}y + x'\underbrace{y\cdot y'}_{0})$ $\boxed{\because A\cdot A'=0}$

$\Rightarrow (x+y'+xy)\cdot(0)$

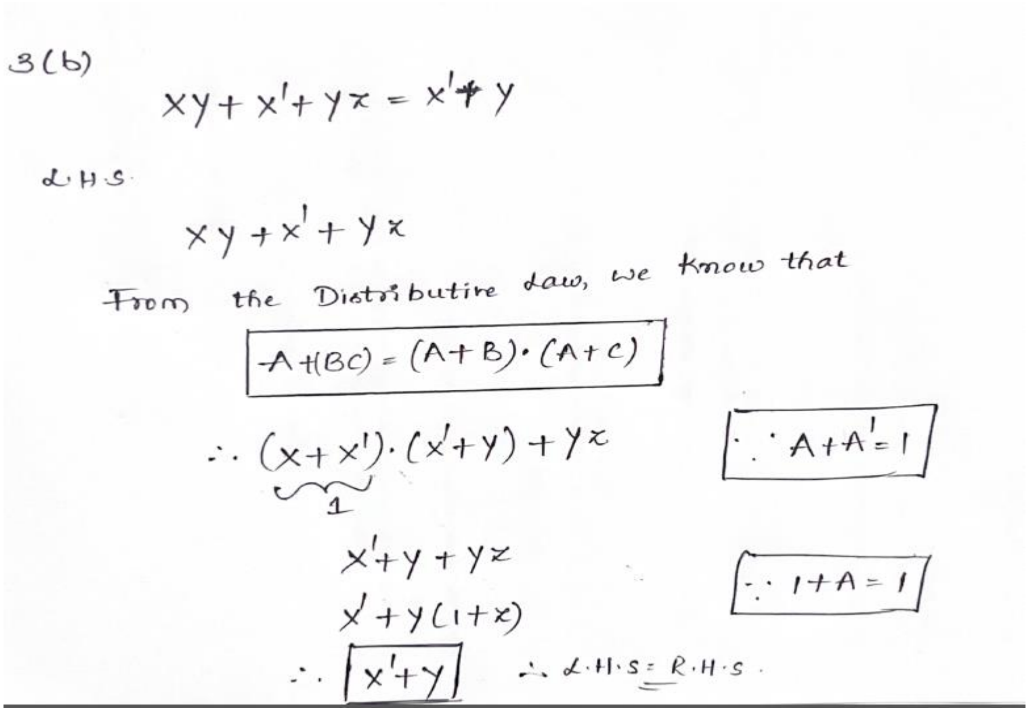
$\Rightarrow 0.$

$\therefore$ L.H.S = R.H.S.

3) b) Using Boolean laws prove that

xy + x’ +yz = x’ + y

Ans:



3(b)

$$xy+x'+yz = x'+y$$

L.H.S

$xy+x'+yz$

From the Distributive law, we know that

$\boxed{A+(BC) = (A+B)\cdot(A+C)}$

$\therefore \underbrace{(x+x')}_{1}\cdot(x'+y)+yz$ $\boxed{\because A+A'=1}$

$x'+y+yz$ $\boxed{\because 1+A=1}$

$x'+y(1+z)$

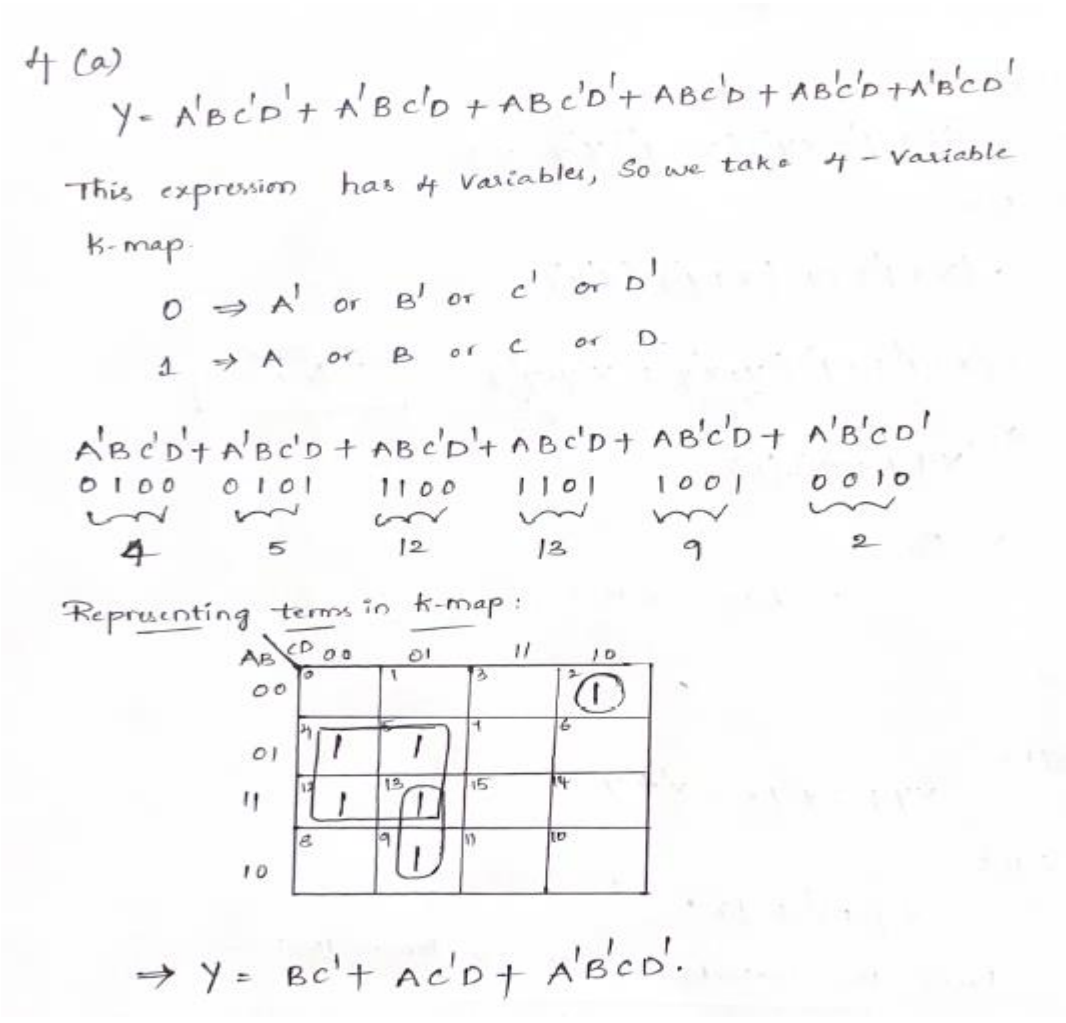
$\therefore \boxed{x'+y}$ $\therefore$ L.H.S = R.H.S.

(OR)

4) a) Minimize the expression using K-map.

Y = A’BC’D’ + A’BC’D + ABC’D’ + ABC’D +AB’C’D +A’B’CD’

Ans:



4 (a)

$Y = A'BC'D' + A'BC'D + ABC'D' + ABC'D + AB'C'D + A'B'CD'$

This expression has 4 variables, So we take 4 - Variable K-map.

$0 \Rightarrow A'$ or B' or C' or D'

$1 \Rightarrow A$ or B or C or D

$$\underbrace{A'BC'D'}_{\substack{0100\\4}} + \underbrace{A'BC'D}_{\substack{0101\\5}} + \underbrace{ABC'D'}_{\substack{1100\\12}} + \underbrace{ABC'D}_{\substack{1101\\13}} + \underbrace{AB'C'D}_{\substack{1001\\9}} + \underbrace{A'B'CD'}_{\substack{0010\\2}}$$

Representing terms in K-map:

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

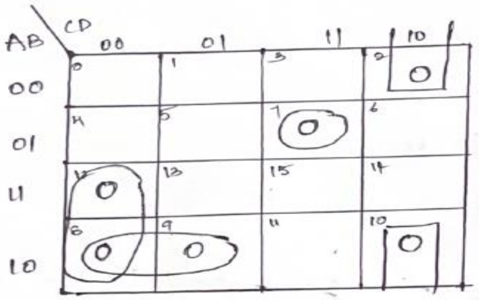
$\Rightarrow Y = BC' + AC'D + A'B'CD'$.

4) b) Obtain the simplified expression in product of sums using K-map and draw the circuit diagram.

f (A,B,C,D) = π(2, 7, 8, 9, 10, 12)

4(b) $f(A,B,C,D) = \pi(2,7,8,9,10,12)$.

4-variables, So we have to use 4-variable k-map



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

⇒ Expression is

$$\boxed{\Rightarrow (B+C'+D)\cdot(A'+B+C)\cdot(A'+C+D)\cdot(A+B'+C'+D')}$$

UNIT-III

5) a) Design a BCD code to excess – 3 code converter and draw the logic diagram.

Ans :

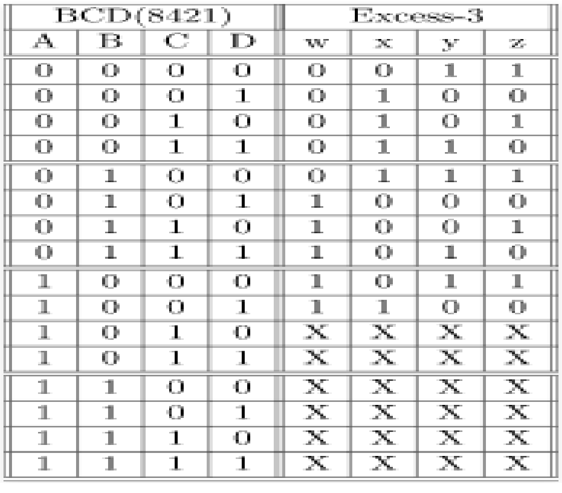
BCD digit can be converted to its corresponding Excess-3 code by simply adding 3 to it. Since we have only 10 digits (0 to 9) in decimal, we don't care about the rest and marked them with a cross (X).

Let A, B, C, and D be the bits representing the binary numbers, where D is the LSB and A is the MSB, and

Let w, x, y, and z be the bits representing the gray code of the binary numbers, where z is the LSB and w is the MSB.

The truth table for the conversion is given below. The X's mark is don't care condition.

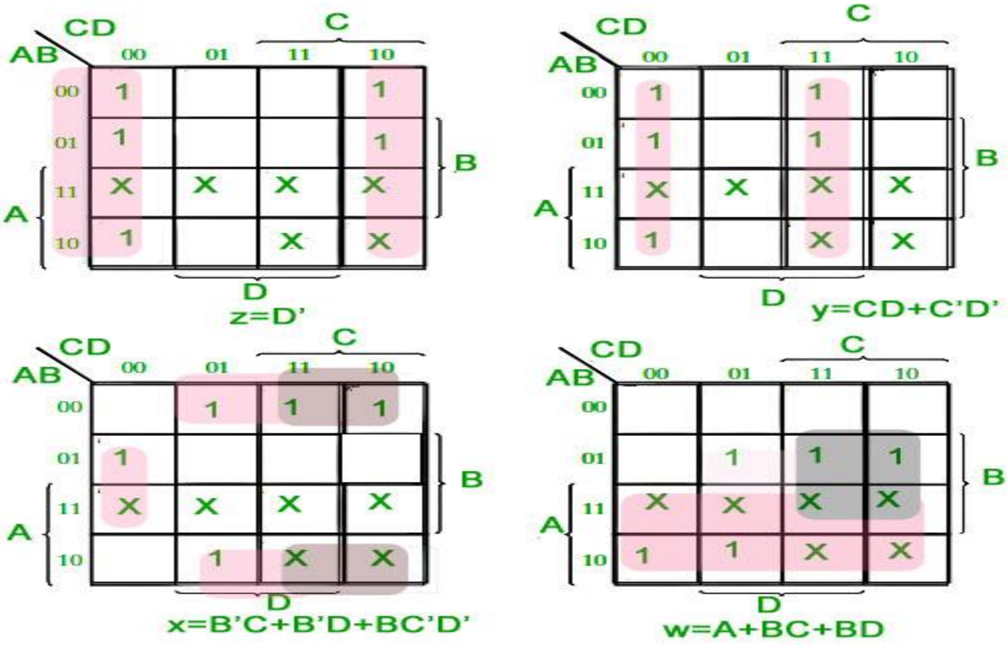
Truth Table :



BCD(8421)				Excess-3			
A	B	C	D	w	x	y	z
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0
1	0	1	0	X	X	X	X
1	0	1	1	X	X	X	X
1	1	0	0	X	X	X	X
1	1	0	1	X	X	X	X
1	1	1	0	X	X	X	X
1	1	1	1	X	X	X	X

K-map Simplification:

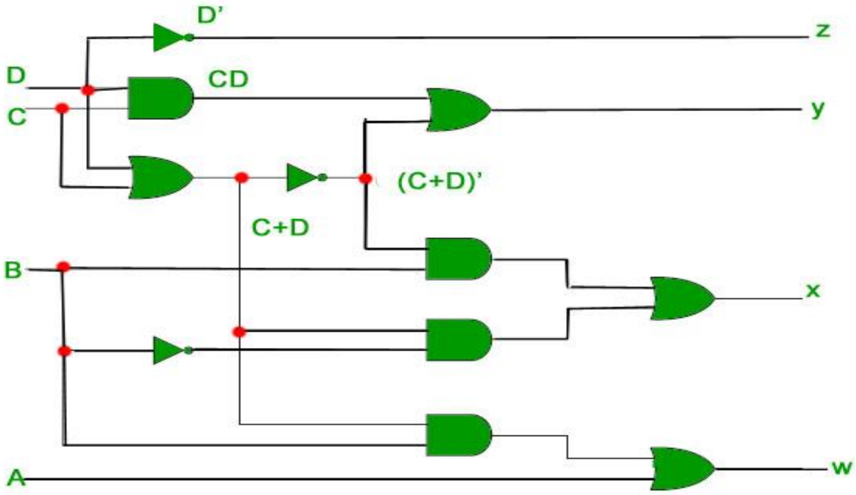
To find the corresponding digital circuit, we will use the K-Map technique for each of the Excess-3 code bits as output with all of the bits of the BCD number as input.



Corresponding minimized Boolean expressions for Excess-3 code bits –

$$\begin{aligned} w &= A + BC + BD \\ x &= B'C + B'D + BC'D' \\ y &= CD + C'D' \\ z &= D' \end{aligned}$$

Circuit Diagram :

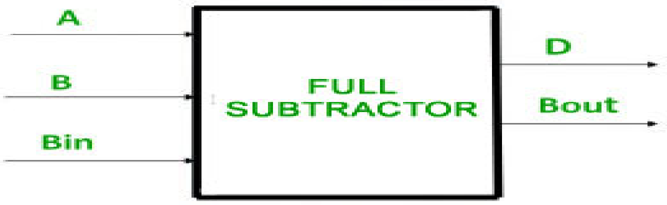


5) b) Design a full subtractor with two half-subtractors and an OR gate.

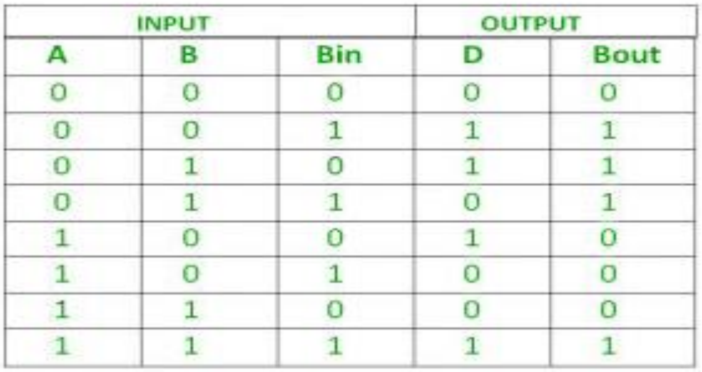
Ans:

- In multi-bit subtraction, we have to subtract bit along with the borrow of the previous digit subtraction. Such subtraction requires subtraction between three bits, which is not possible with half-subtractor.
- The disadvantage of a half subtractor is rectified by full subtractor.
- The full subtractor is a combinational circuit with three inputs A, B, Bin and two outputs Difference(D) and Borrow(Bout). A is the ‘minuend’, B is ‘subtrahend’, Bin is the ‘borrow’ produced by the previous stage, D is the difference output and Bout is the borrow output.

Block Diagram :



Truth Table :



INPUT			OUTPUT	
A	B	Bin	D	Bout
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Expressions & Circuit Diagram :

Logical expression for Difference :

D = ∑(1,2,4,7)

=A’B’Bin + A’BBin’ + AB’Bin’ + ABBin

= Bin(A’B’ + AB) + Bin’(AB’ + A’B)

= Bin(A XNOR B) + Bin’(A XOR B)

= Bin (A XOR B)’ + Bin’(A XOR B)

= Bin XOR (A XOR B)

= (A XOR B) XOR Bin

Logical expression for Borrow :

Bout = ∑(1,2,3,7)

=A’B’Bin + A’BBin’ + A’BBin + ABBin

= A’B’Bin +A’BBin’ + A’BBin + A’BBin + A’BBin + ABBin

= A’B’Bin +A’BBin’ + A’BBin + A’BBin + A’BBin + ABBin

= A’Bin(B + B’) + A’B(Bin + Bin’) + BBin(A + A’)

= A’Bin + A’B + BBin

OR

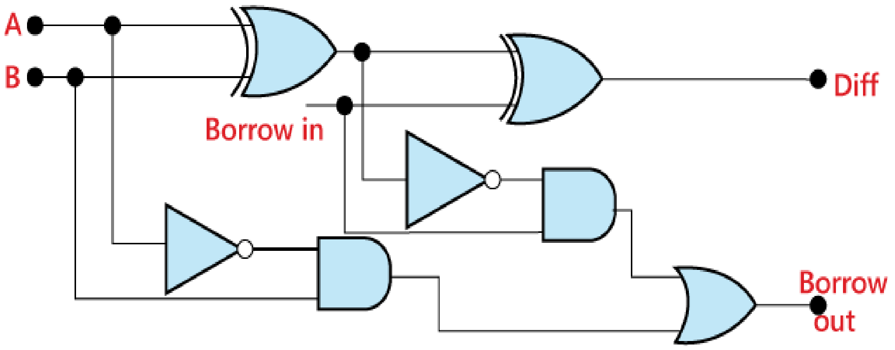
Bout = A’B’Bin + A’BBin’ + A’BBin + ABBin

= Bin(AB + A’B’) + A’B(Bin + Bin’)

= Bin(A XNOR B) + A’B

= Bin (A XOR B)’ + A’B

Logic/Circuit Diagram of Full Subtractor :



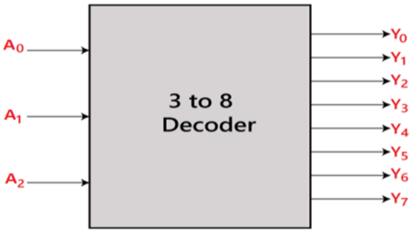
Full-Subtractor Circuit

(OR)

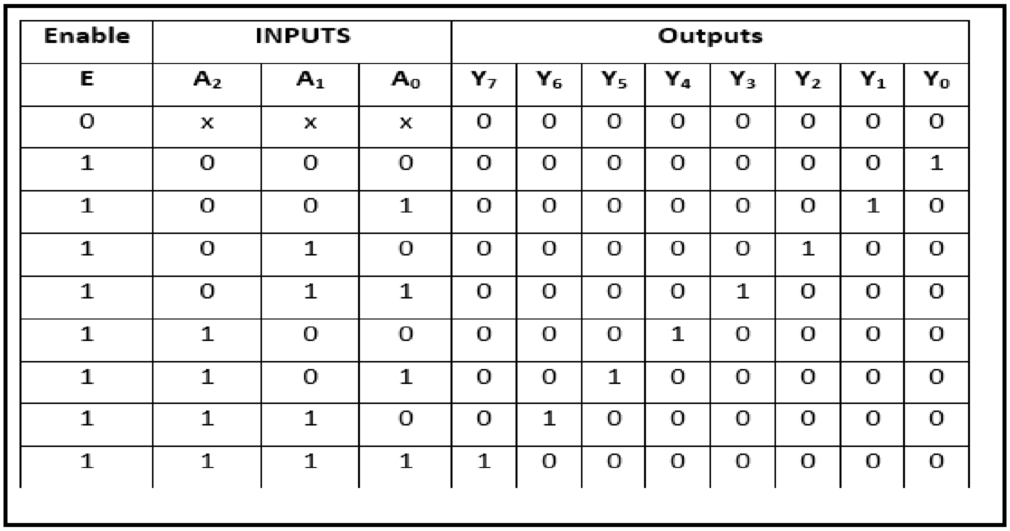
6) a) Draw the truth table and the circuit for 3 : 8 decoder and explain.

Ans:

Block Diagram: The 3 to 8 line decoder is also known as binary to Octal Decoder. In a 3 to 8 line decoder, there is a total of eight outputs, i.e., Y0, Y1, Y2, Y3, Y4, Y5, Y6, and Y7 and three outputs, i.e., A0, A1, and A2. This circuit has an enable input 'E'. when enable 'E' is set to 1, one of these four outputs will be 1. The block diagram and the truth table of the 3 to 8 line encoder are given below.



Truth Table:



Enable	INPUTS			Outputs							
E	A_2	A_1	A_0	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
0	x	x	x	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	0	1	0	0
1	0	1	1	0	0	0	0	1	0	0	0
1	1	0	0	0	0	0	1	0	0	0	0
1	1	0	1	0	0	1	0	0	0	0	0
1	1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0	0

Logical Expression: The logical expression of the term Y0, Y1, Y2, Y3, Y4, Y5, Y6, and Y7 is as follows:

$Y_0=A_0'.A_1'.A_2'$

$Y_1=A_0.A_1'.A_2'$

$Y_2=A_0'.A_1.A_2'$

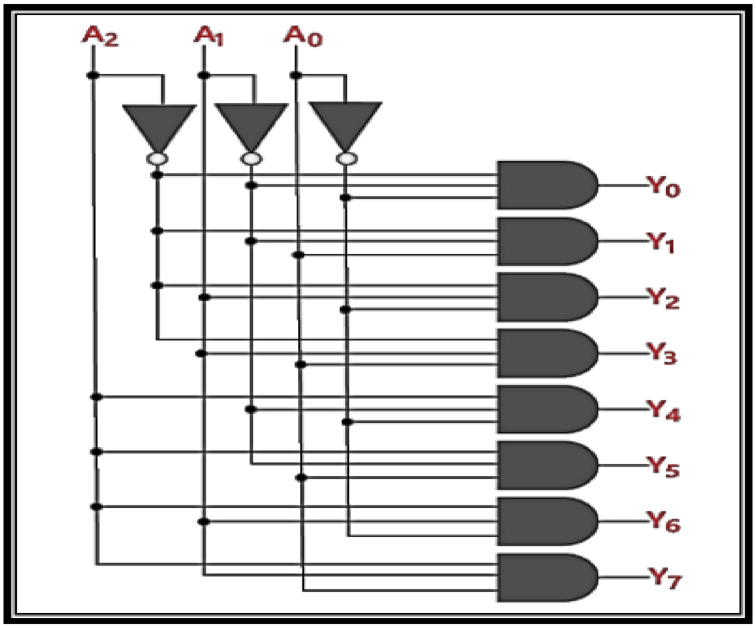
$Y_3=A_0.A_1.A_2'$

$Y_4=A_0'.A_1'.A_2$

$Y_5=A_0.A_1'.A_2$

$Y_6=A_0'.A_1.A_2$

$Y_7=A_0.A_1.A_2$



6) b) Show the truth table of 4 – bit priority encoder and design the logical circuit of the 4-bit priority encoder.

Ans :

- A 4 to 2 priority encoder has four inputs Y3, Y2, Y1 & Y0 and two outputs A1 & A0. Here, the input, Y3 has the highest priority, whereas the input, Y0 has the lowest priority.

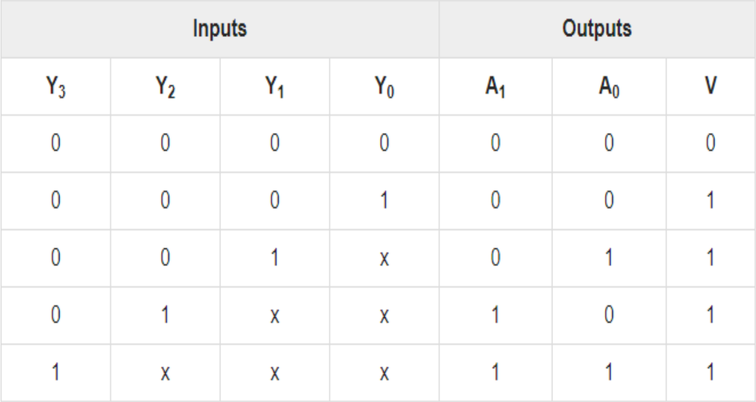
- In this case, even if more than one input is '1' at the same time, the output will be the binary code corresponding to the input, which is having higher priority.

- We considered one more output, V in order to know, whether the code available at outputs is valid or not.

- If at least one input of the encoder is '1', then the code available at outputs is a valid one. In this case, the output, V will be equal to 1.

- If all the inputs of encoder are '0', then the code available at outputs is not a valid one. In this case, the output, V will be equal to 0.

Truth Table :



Inputs				Outputs		
Y_3	Y_2	Y_1	Y_0	A_1	A_0	V
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

K-map : Use 4 variable K-maps for getting simplified expressions for each output.

The simplified Boolean functions are:

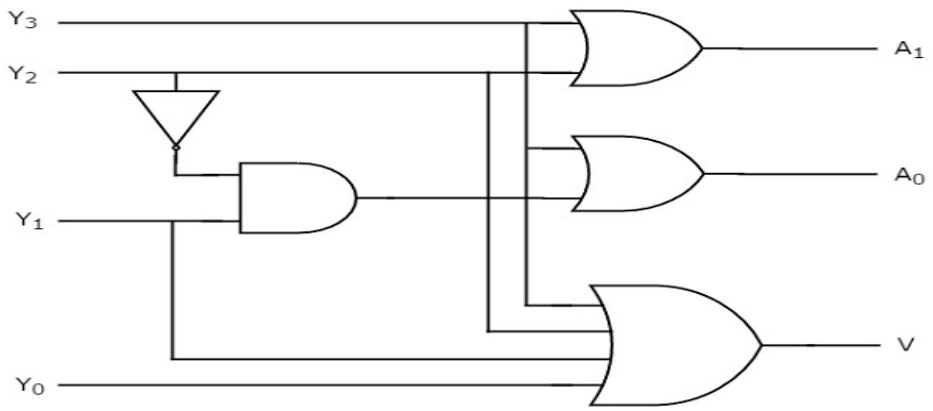
$$A_1 = Y_3 + Y_2$$

$$A_0 = Y_3 + Y_2'Y_1$$

The Boolean function of output, V as:

$$V = Y_3 + Y_2 + Y_1 + Y_0$$

Circuit Diagram :



The above circuit diagram contains two 2-input OR gates, one 4-input OR gate, one 2input AND gate & an inverter. Here AND gate & inverter combination are used for producing a valid code at the outputs, even when multiple inputs are equal to '1' at the same time. Hence, this circuit encodes the four inputs with two bits based on the priority assigned to each input.

UNIT-IV

7 a) Design the conversion logic to convert SR flip-flop into T flip-flop.

Ans:

Rules for conversion:

Step-1:

Find the characteristics table of required flip-flop and the excitation table of the existing (given) flip-flop.

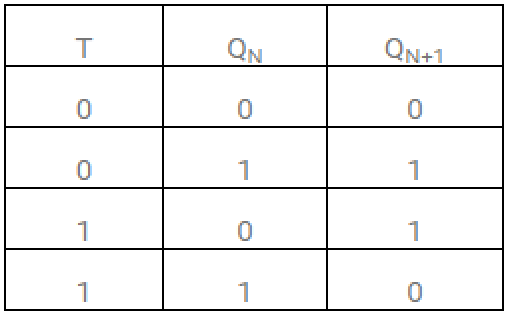
Step-2:

Find the expression of given flip-flop in terms of required flip-flop using K-map.

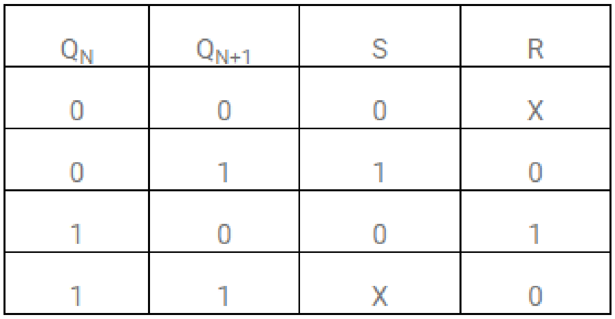
Step-3:

Find the circuit diagram of required flip-flop.

Characteristic Table for T Flip Flop & Excitation Table for SR Flip Flop:

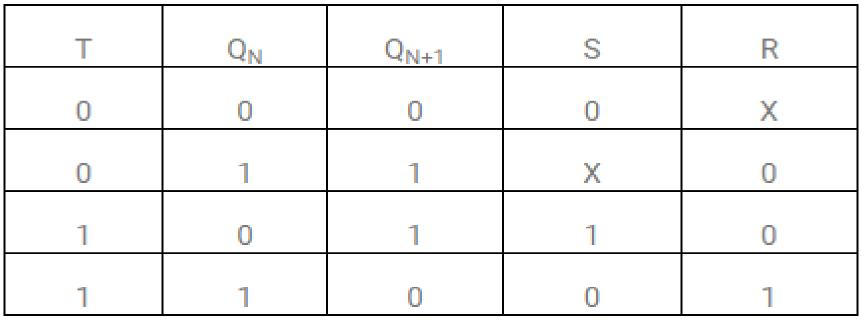


T	Q_N	Q_{N+1}
0	0	0
0	1	1
1	0	1
1	1	0



Q_N	Q_{N+1}	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

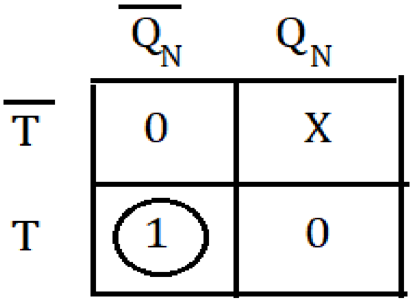
Conversion Table & Boolean Expression:



T	Q_N	Q_{N+1}	S	R
0	0	0	0	X
0	1	1	X	0
1	0	1	1	0
1	1	0	0	1

In this case the given flip-flop is SR. Therefore, write the Boolean expressions for S and R from the conversion table using K-Maps.

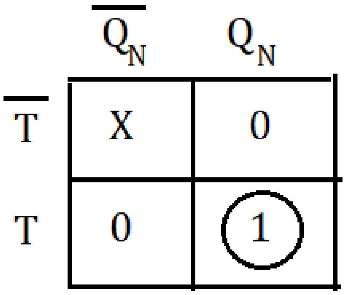
K-Map for S:



	$\overline{Q_N}$	Q_N
$\overline{T}$	0	X
T	1	0

S = T.QN’

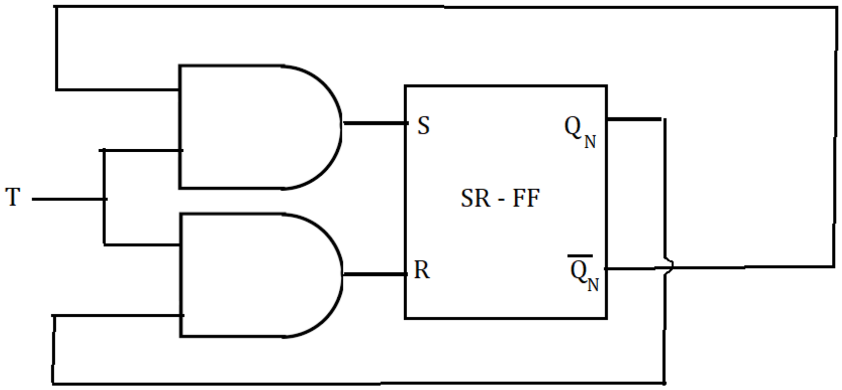
K-Map for R:



	$\overline{Q_N}$	Q_N
$\overline{T}$	X	0
T	0	1

R = T.QN

Circuit Diagram :



7 b) Compare combinational and Sequential circuits.

Combinational Circuit	Sequential Circuit
1. In this output depends only upon present input.	**1.** In this output depends upon present as well as past input.
2. Speed is fast.	**2.** Speed is slow.
3. It is designed easy.	**3.** It is designed tough as compared to combinational circuits.
4. There is no feedback between input and output.	**4.** There exists a feedback path between input and output.
5. This is time independent.	**5.** This is time dependent.
6. Elementary building blocks: Logic gates	**6.** Elementary building blocks: Flip-flops
7. Used for arithmetic as well as boolean operations.	**7.** Mainly used for storing data.
8. Combinational circuits don’t have capability to store any state.	**8.** Sequential circuits have capability to store any state or to retain earlier state.
9. As combinational circuits don’t have clock, they don’t require triggering.	**9.** As sequential circuits are clock dependent they need triggering.
10. These circuits do not have any memory element.	**10.** These circuits have memory element.
11. It is easy to use and handle.	**11.** It is not easy to use and handle.
12. In this output depends only upon present input.	**12.** In this output depends upon present as well as past input.
13. Speed is fast.	**13.** Speed is slow.
14. It is designed easy.	**14.** It is designed tough as compared to combinational circuits.

(OR)

8 a) Design the conversion logic to convert JK flip-flop into D flip-flop.

Ans:
Rules for conversion:
Step-1:

Find the characteristics table of required flip-flop and the excitation table of the existing (given) flip-flop.

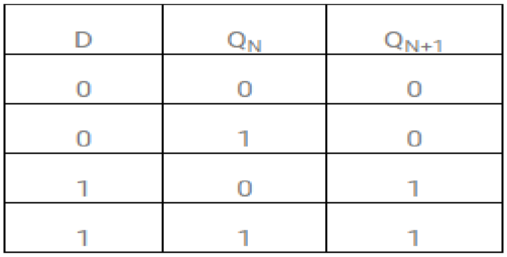
Step-2:

Find the expression of given flip-flop in terms of required flip-flop using K-map.

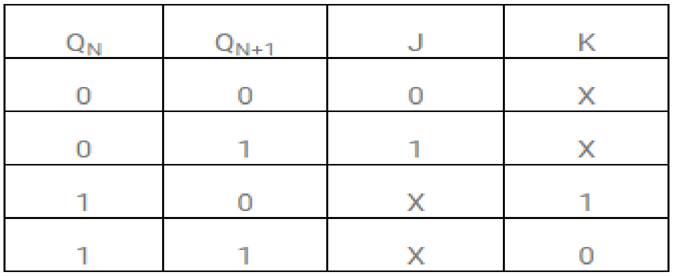
Step-3:

Find the circuit diagram of required flip-flop.

Characteristic Table for D Flip Flop & Excitation Table for JK Flip Flop:

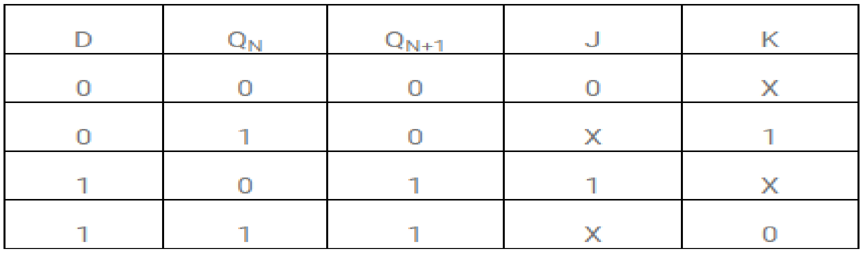


D	Q_N	Q_{N+1}
0	0	0
0	1	0
1	0	1
1	1	1



Q_N	Q_{N+1}	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

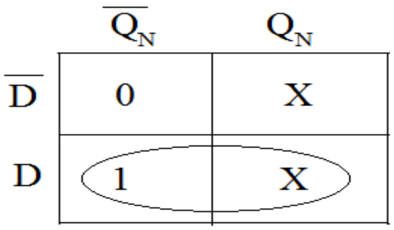
Conversion Table & Boolean Expression:



D	Q_N	Q_{N+1}	J	K
0	0	0	0	X
0	1	0	X	1
1	0	1	1	X
1	1	1	X	0

Therefore, write the Boolean expressions for J and K from the conversion table using K-Maps.

K-Map for J:

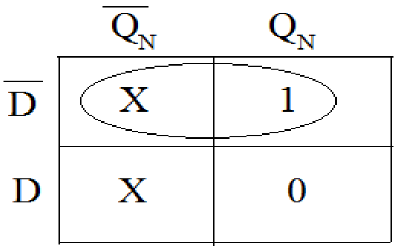


	$\overline{Q_N}$	Q_N
$\overline{D}$	0	X
D	1	X

Expression for J would be

J = D

K-Map for K:

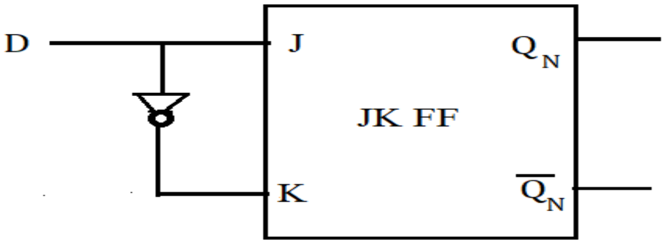


	$\overline{Q_N}$	Q_N
$\overline{D}$	X	1
D	X	0

Expression for K would be

K = D’

Circuit Diagram :



8 b) Show the excitation table of the following flip-flops.

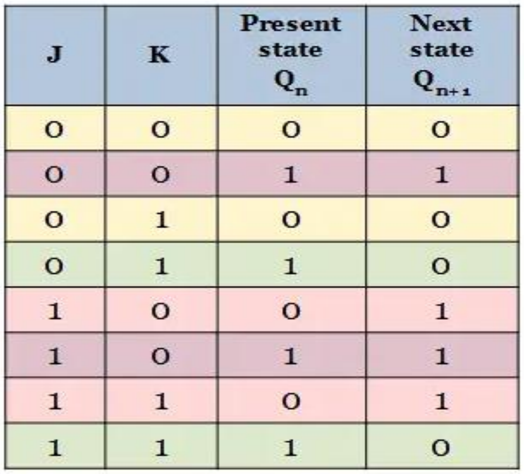
i) JK flip-flop ii) D flip-flop

Ans : JK Flip-Flop:

For the JK flip flop, the excitation table is derived from the truth table, for the present state and next state values Qn = 0 and Qn+1 = 0 the inputs are J = 0 and K = 0 or 1.

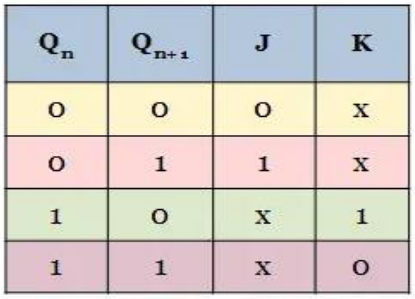
Since K input has two values, it is considered as a don't care condition(x).

Thus the state transition from Qn = 0 to Qn+1 = 0 takes place when J = 0, K = x. It is filled in the first row of the excitation table.



J	K	Present state Q_n	Next state Q_{n+1}
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Truth table of JK flip flop



Q_n	Q_{n+1}	J	K
0	0	0	x
0	1	1	x
1	0	x	1
1	1	x	0

Excitation table of JK flip flop

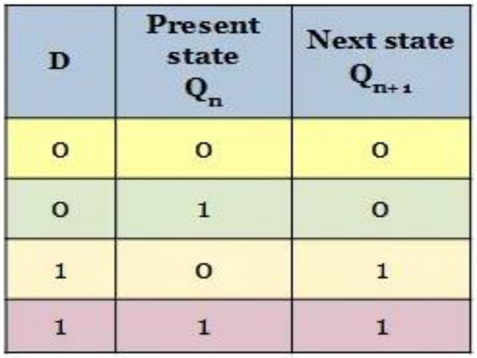
The state transition from present state Qn = 0 to the next state Qn+1 = 1 occur, when the inputs are either J = 1, K = 0 or J = 1, K = 1Thus the excitation table is filled with datas Qn = 0, Qn+1 = 1, J = 1 and K = x.

Similarly, for the transition of the state from 1 to 0, the inputs are J = 0, K = 1 or J = 1, K = 1. So for this transition, the required inputs are J = x and K =1, as the value of J can be either 0 or 1.

For the state transition from Qn = 1 to Qn+1 = 1, the J input can be 0 or 1 but the K input remains at 0. For this transition to occur, the excitation inputs are J = x and K = 0.

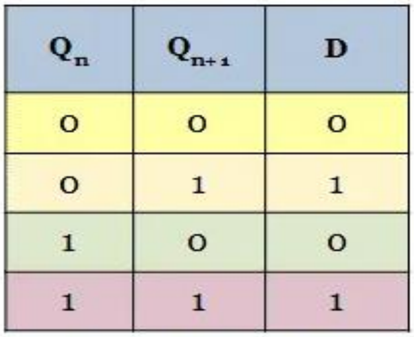
D flip-flop:

The excitation table of the D flip-flop is derived from its truth table. the next state output is equal to the D input. So it is very simple to construct the excitation table.



D	Present state Q_n	Next state Q_{n+1}
0	0	0
0	1	0
1	0	1
1	1	1

Truth table of D flip flop



Q_n	Q_{n+1}	D
0	0	0
0	1	1
1	0	0
1	1	1

Excitation table of D flip flop

For the state transition from Qn = 0 to Qn+1 = 0, the required excitation input is D = 0, regardless of Qn value. For transition of states from Qn = 0 to Qn+1 = 1, the input required to excite is D = 1.

The state transit from Qn = 1 to Qn+1 = 0 for the input D = 0. For the input D = 1, the state transition takes place from Qn = 1 to Qn+1 = 1.

UNIT-V

9) a) Design a BCD ripple counter using JK flip-flops

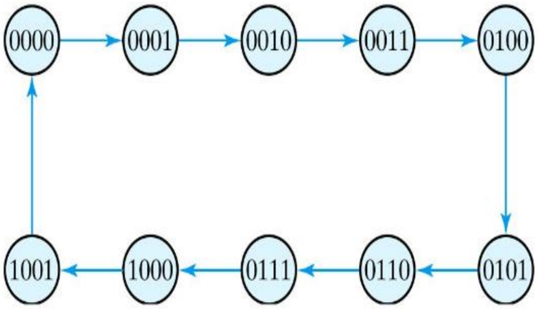
Ans :

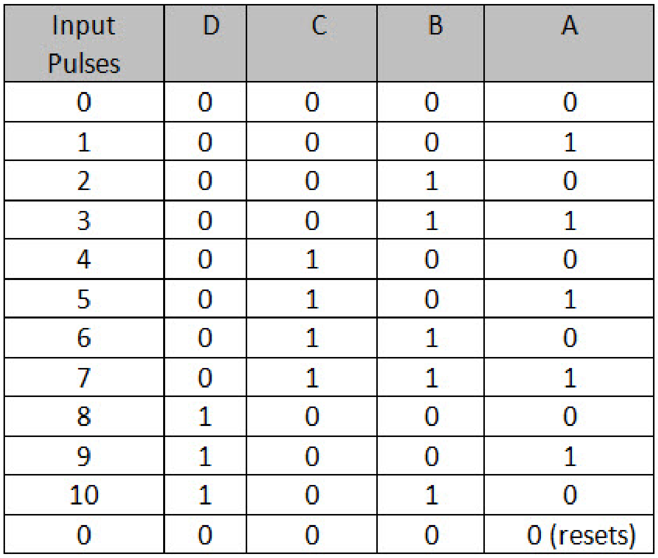
Explanation :

- When the Decade counter is at REST, the count is equal to 0000. This is first stage of the counter cycle. When we connect a clock signal input to the counter circuit, then the circuit will count the binary sequence. The first clock pulse can make the circuit to count up to 9 (1001). The next clock pulse advances to count 10 (1010).

- Then the ports X1 and X3 will be high. As we know that for high inputs, the NAND gate output will be low. The NAND gate output is connected to clear input, so it resets all the flip flop stages in decade counter. This means the pulse after count 9 will again start the count from count 0.

State Diagram :



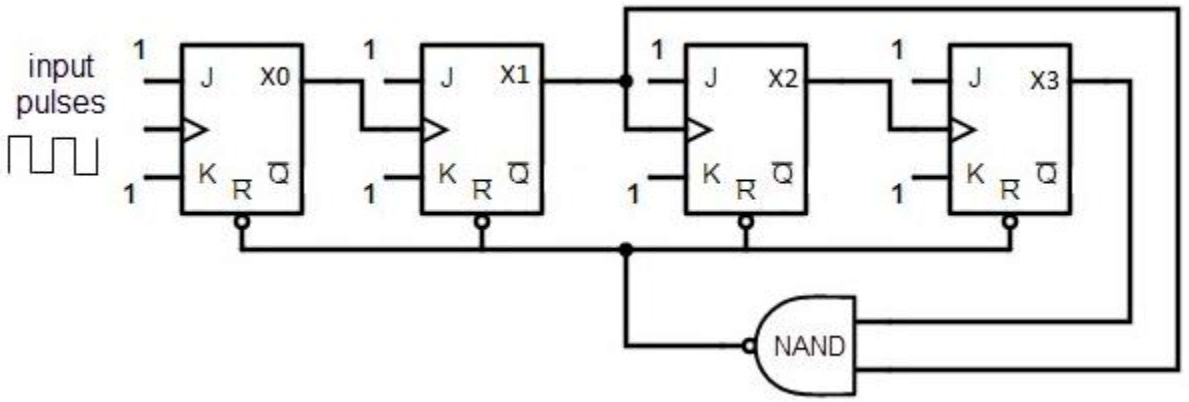


Input Pulses	D	C	B	A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
0	0	0	0	0 (resets)

It represents the count of circuit for decimal count of input pulses. The NAND gate output is zero when the count reaches 10 (1010).

The count is decoded by the inputs of NAND gate X1 and X3. After count 10, the logic gate NAND will trigger its output from 1 to 0, and it resets all flip flops.

Circuit Diagram :

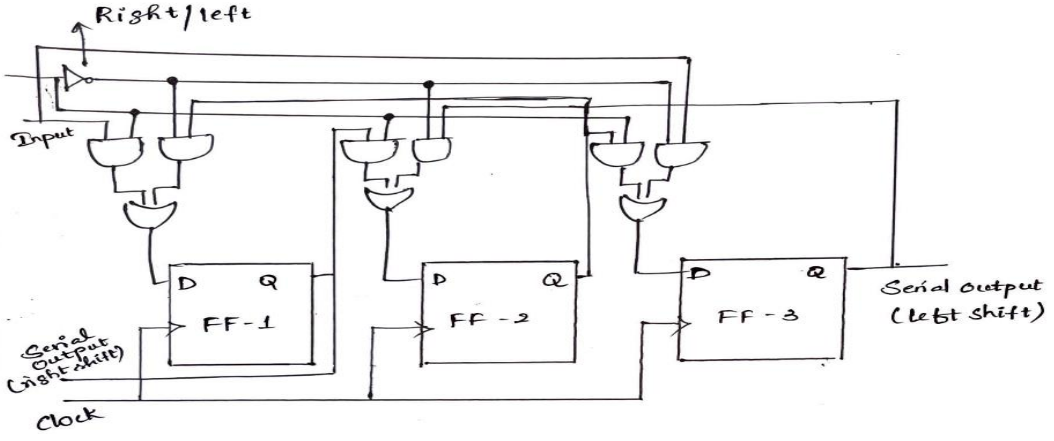


9) b) Design a three bit bi-directional shift register that shifts the bits to left when a control variable E = 0 and shifts the bits to right when E = 1 using D flip-flops.

Explanation:

- Bidirectional shift registers are the registers which are capable of shifting the data either right or left depending on the mode selected.
- If the mode selected is 1(high), the data will be shifted towards the right direction and if the mode selected is 0(low), the data will be shifted towards the left direction.
- The logic circuit given below shows a Bidirectional shift register. The circuit consists of three D flip-flops which are connected. The input data is connected at two ends of the circuit and depending on the mode selected only one and gate is in the active state.

Circuit Diagram :



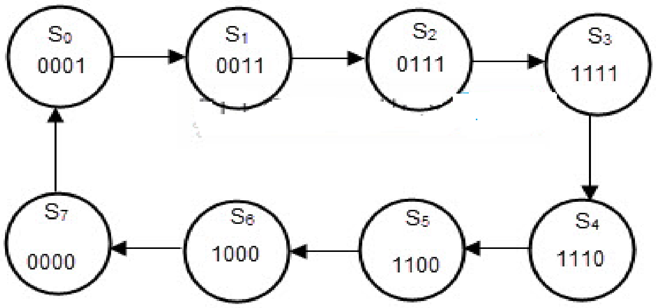
(OR)

10) a) Design a 4-bit Johnson Counter.

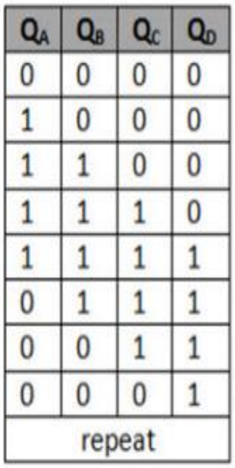
Ans:

- The Johnson counter is a modification of ring counter.
- In this the inverted output of the last stage flip flop is connected to the input of first flip flop.
- If we use n flip flops to design the Johnson counter, it is known as 2n bit Johnson counter or Mod 2n Johnson counter.
- This is an advantage of the Johnson counter that it requires only half number of flip flops that of a ring counter uses, to design the same Mod.
- The main difference between the 4 bit ring counter and the Johnson counter is that , in ring counter , we connect the output of last flip flop directly to the input of first flip flop.
- But in Johnson counter, we connect the inverted output of last stage to the first stage input.
- The Johnson counter is also known as Twisted Ring Counter, with a feedback.
- In Johnson counter the input of the first flip flop is connected from the inverted output of the last flip flop.
- The Johnson counter or switch trail ring counter.
- "n-stage" Johnson counter will circulate a single data bit giving sequence of 2n different states and can therefore be considered as a "mod-2n counter".

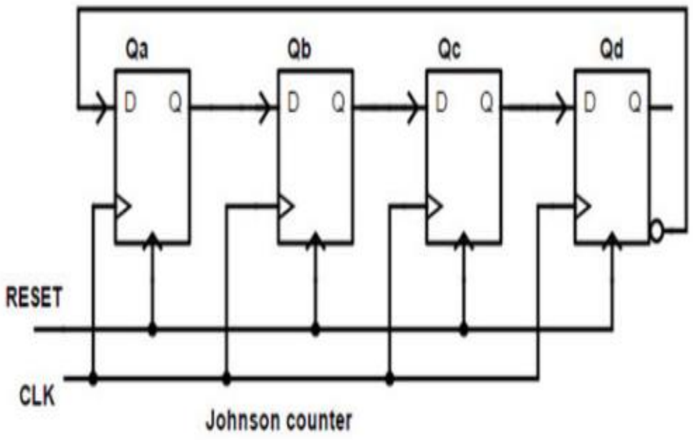
State Diagram :



Circuit Diagram :



Q_A	Q_B	Q_C	Q_D
0	0	0	0
1	0	0	0
1	1	0	0
1	1	1	0
1	1	1	1
0	1	1	1
0	0	1	1
0	0	0	1
repeat			



10 b) Compare and contrast the synchronous counters with the asynchronous counters.

S.NO	Synchronous Counter	Asynchronous Counter
1.	In synchronous counter we use a universal clock that is common to all flip flops throughout the circuit.	In asynchronous counter main clock is only applied to the first flip flop and then for rest of flip flops the output of previous flip flop is taken as a clock.
2.	Synchronous Counter is faster in operation as compared to Asynchronous Counter.	Asynchronous Counter is slower as compared to synchronous counter in operation.
3.	Synchronous Counter does not produce any decoding errors.	Asynchronous Counter produces decoding error.
4.	Synchronous Counter is also called Parallel Counter.	Asynchronous Counter is also called Serial Counter.
5.	Synchronous Counter designing as well implementation are complex due to increasing the number of states.	Asynchronous Counter designing as well as implementation is very easy.
6.	Synchronous Counter will operate in any desired count sequence.	Asynchronous Counter will operate only in fixed count sequence (UP/DOWN).
7.	Synchronous Counter examples are: Ring counter, Johnson counter.	Asynchronous Counter examples are: Ripple UP counter, Ripple DOWN counter.
8.	In synchronous counter, propagation delay is less.	In asynchronous counter, there is high propagation delay.

************ THE END**************