VEER KUNWAR INSTITUTE OF TECHNOLOGY, BIJNOR

SWITCHING THEORY & LOGIC DESIGN

(NEC-304)

Lalit Kumar Rathi (E.C Deptt.)

Syllabus

NEC-304 Switching Theory and Logic Design

UNIT-Ist

Digital system and Binary Numbers: Signed binary numbers, binary codes. Gate-level minimization: The map method up to four variables, don't care conditions, POS simplification, NAND and NOR implementation, Quine Mc-Clusky method (Tabular method).

UNIT-IInd

Combination logic: Combinational circuits, analysis procedure, design Procedure, binary-adder-subtractor, decimal adder, binary multiplier, magnitude comparator, decoders, encoders, multiplexers.

UNIT-IIIrd

Synchronous Sequential logic: Sequential circuits, Storage elements: latch, flip flops, Analysis of clocked sequential circuits, State reduction and assignments, design procedure.

Asynchronous Sequential logic: Analysis procedure, circuit with latch design procedure, reduction of state and flow table, race free state assignment, Hazards.

UNIT-IVth

Registers and counters: Shift registers, ripple counter, synchronous counter, other counters.

Memory and programmable logic: RAM, ROM, PLA, PAL.

References

1. Digital design → J.S. Katre
2. Digital logic design → Rashmi Sharma
3. Switching theory and Logic design → Dr. Sanjay Sharma
4.

UNIT-Ist Digital Systems and Binary Numbers

① **Signals** :- We can define "signal" as a physical quality, which contains some information and which is a function of one or more independent variables.

Types of Signals :- The signals can be of two types

i) Analog signals ii) Digital Signals

ii) **Digital Signals** :- A Digital signal is defined as the signal which has only a finite number of distinct values.

Digital signals are not continious signal. They are discrete signals as shown in fig. below.

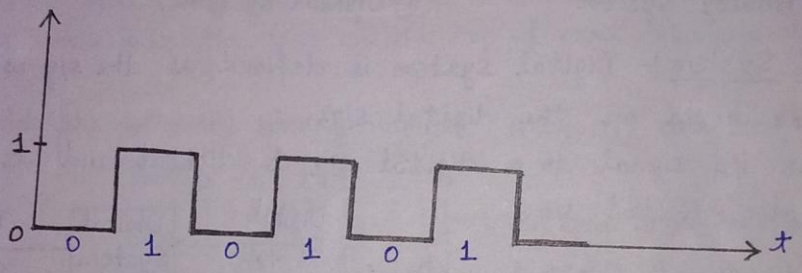


Fig. → Digital signal

Types of Digital s/gs	No. of distinct values
Binary	2
Octal	8
Hexadecimal	16

⇒ **Sources of Digital Signal** :-

* The digital signal can be obtained directly from the computer. All the data used by the computer is digital.

* We can also use an A to D converter so as to convert analog signals into digital signals.

⇒ Advantages of Digital Signals:-

* Digital signals can be processed and transmitted more efficiently and reliably than analog signals.
* It is possible to store the digital data.
* The effect of "noise" (unwanted fluctuations) is less. So digital data does not get corrupted.

② System or Circuit :- A system (s/y) is defined as the physical device or group of devices which performs the required operations of the signal (s/g) applied at its input (i/p).

Systems can be of two types.

i) Analog System ii) Digital System

ii) Digital System :- Digital system is defined as the s/y which processes or works on the digital s/g's.

The i/p signal to a digital s/y is digital and its o/p signal is also digital s/g.

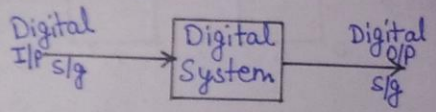


Fig. → Digital System

⇒ Advantages of Digital systems:-

* They have higher accuracy.
* They are less affected by variation in temperature.
* Less affected by Noise.
* Design is much easier.
* They are cost efficient.
* Communication and data transmission b/w the digital s/ys is much easier.

⇒ Examples of digital systems:-

* In Comm. System
* Traffic control
* Weather monitoring
* Digital TV and disks.
* Digital cameras.
* Digital computers

③ Binary Arithmetic :-

⇒ **Binary Addition :-** The four basic rules for binary addition in term of sum and carry are as follows.

Rule	A	B	Sum	Carry
1.	0	0	0	0
2.	0	1	1	0
3.	1	0	1	0
4.	1	1	0	1

Binary (2), Octal (8), Decimal (10), Hexa-decimal (16) } Weighted codes

Excess-3, Gray Code } Non-Weighted codes

Steps for Binary Addition :-

Step I → Add the least significant bits (LSB)
Step II → Write the carry digit on top of the next column of bits.
Step III → Add carry with the digits in the second column to get the sum.
Step IV → Write the carry on the top of next column and continue.

Q. Add the following binary numbers. $A = (10111)_2$ and $B = (11001)_2$.

Solu.

Carry	1	1	1	1		
A =	1	0	1	1	1	→ LSBs (Least significant bits)
+ B =	1	1	0	0	1	
A+B =	1	1	0	0	0	0

$(10111)_2 + (11001)_2 = (110000)_2$

10111 → 23	Simple Addition Method
11001 → 25	
110000 → 48	

Q Add the following binary numbers. 011 and 101.

Solu.

Carry	1	1	
A =	0	1	1
+ B =	1	0	1
A+B =	1	0	0	0

$(011)_2 + (101)_2 = (1000)$

011 → 3
101 → 5
1000 → 8

⇒ **Binary Subtraction**:- The four basic rules for binary subtraction in terms of subtraction and borrow are as follows.

Case	A	B	Subtraction	Borrow
1	0	0	0	0
2	1	0	1	0
3	1	1	0	0
4	0	1	1	1

Steps for Binary Substraction:-

Step I → Subtract column by column

Step II → Always start from the column of LSBs.

Step III → If necessary, barrow from the next higher column.

Q. Subtract the following binary numbers : $A=(11011)_2$ & $B=(10110)_2$

Solu.

Borrow -1

A = 1 1 0 1 1

− B = 1 0 1 1 0

A − B = 0 0 1 0 1

$(11011)_2 = 27$

$(10110)_2 = 22$

$(00101)_2 = 5$

Q. Subtract the decimal number $(38)_{10}$ and $(29)_{10}$ by converting them into binary.

Solu. $(38)_{10} = (100110)_2$ $(29)_{10} = (11101)_2$

Borrow 1 1 -1

A = 1 0 0 1 1 0

− B = 0 1 1 1 0 1

A − B = 0 0 1 0 0 1

38

−29

9 = $(1001)_2$

Q. Subtract the following numbers : $A=(110101)_2$ & $B=(010111)_2$

Borrow 1 1 1 1

A = 1 1 0 1 0 1

B = 0 1 0 1 1 1

A − B 0 1 1 1 1 0

④ Complements :- Complements are used in the digital computers in order to simplify the subtraction operation and for the logical manipulations.

⇒ 1's Complement :- The 1's complement of a binary number is the number that results when we complement each bit. The process of obtaining 1's complement has been illustrated in fig. below.

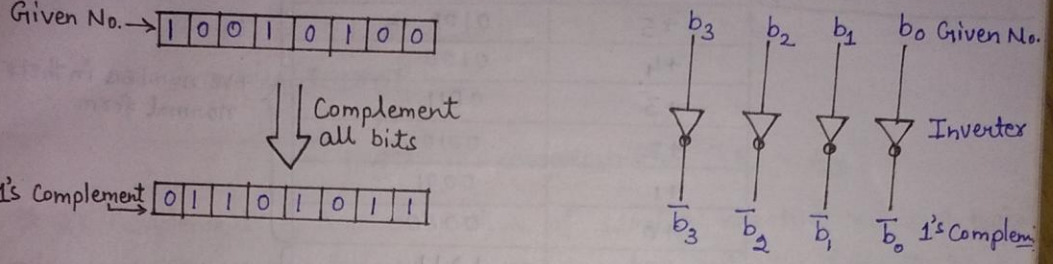


Fig → Obtaining 1's complement

Note : This system is called as 1's complement because the number can be subtracted from 1111 1111 to obtain the result. The answer will be same as if all the 1's were changed to 0's and all the 0's were changed to 1's.

Q. Obtain the 1's complement of the following numbers: $(1010)_2$ & $(11010101)_2$

Solu.

Given Number	1's complement
1010	0101
11010101	00101010

And

```
  1111
- 1010 → Given No.
-------
  0101 → 1's complement
```

```
  11111111
- 11010101 → Given Number
----------
  00101010  1's complement
```

⇒ Representation of positive and Negative Numbers using 1's complement :-

* Positive number in 1's complement form are represented in the same way as the positive sign magnitude number. That means $(+5)_{10}$ can be represented as $(0101)_2$.
* But the -ve numbers are the 1's complement of the corresponding +ve

numbers. For example $(-5)_{10}$ is represented as:

$(+5)_{10} \rightarrow$ [0] 101 (Sign; Magnitude)

$(-5)_{10} \rightarrow 1010$ 1's complement

Max. +ve no. $= 2^{n-1} - 1$

Max. -ve no. $= -(2^{n-1} - 1)$

Decimal Number	1's Complement
+7	0111
+6	0110
+5	0101
+4	0100
+3	0011
+2	0010
+1	0001
+0	0000
-0	1111
-1	1110
-2	1101
-3	1100
-4	1011
-5	1010
-6	1001
-7	1000
-8	

+ve numbers in their normal form.

-ve numbers in their 1's complement form

⇒ 2's complement :- The 2's complement of a binary number is obtained by adding 1 to the LSB of 1's complement of that number

i.e 2's complement = 1's complement +1

Q. Obtain the 2's complement of $(10110010)_2$.

Solu:

Given number: 10110010

1's complement : 01001101

+1

2's complement : 01001110 Ans

⇒ Representation of +ve and -ve Numbers using 2's complement: (4)

* Positive numbers in 2's complement form are represented the same way as in the sign-magnitude and 1's complement form.
* For example $(+6)_{10}$ is represented as 0110 in the 2's complement form.
* But the 2's complement of -ve number is obtain by add 1 in the 1's complement of given binary number.
* For example $(-6)_{10}$ is represented in the 2's complement form as follows;

$(+6)_{10} = 0110$

1's complement = 1001

$+1$

1010 2's complement of $(-6)_{10}$. Ans

* 2's complement is another method of storing negative values. A typical microcomputer represents +ve and -ve numbers with the help of 2's complement.

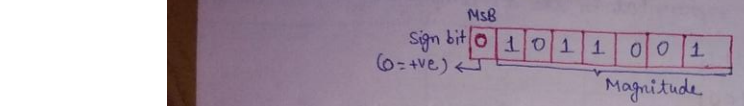
+ve integers: $2^{n-1}-1$

-ve integers: 2^n

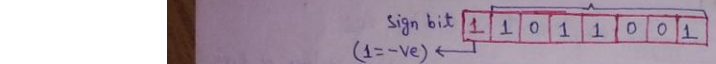
Decimal	1's Complement	2's Complement
+7	0111	0111
+6	0110	0110
+5	0101	0101
+4	0100	0100
+3	0011	0011
+2	0010	0010
+1	0001	0001
0	0000	0000
-1	1110	1111
-2	1101	1110
-3	1100	1101
-4	1011	1100
-5	1010	1011
-6	1001	1010
-7	1000	1001
-8	10111	11000

⑤ **Sign-Magnitude Numbers:-** If the data has +ve as well as -ve no. then the signed binary numbers should be used. The +ve or -ve signs are also represented in the binary form i.e by using 0 or 1. So a 0 is used to represent the +ve sign and 1 is used to represent the (-) sign.

* The MSB of a binary number is used to represent the sign and the remaining bits are used for representing the magnitude. 8-bit signed binary numbers are shown below.



$\Rightarrow (89)_{10}$



$\Rightarrow (-89)_{10}$

Note: The numbers shows above contains a sign bit followed by magnitude bits. Numbers in this form are called as sign-magnitude no. or sign-numbers.

Advantages of sign-magnitude numbers:-

* The main advantage is their simplicity.
* We can easily find the magnitude by deleting the sign bit.

Disadvantages:- The sign-magnitude numbers have a limited used because they require complicated circuits. These no.s are used in A to D convertor.

⇒ **Addition of Signed Numbers:-**

⇒ **Case 1:- Addition of both +ve Numbers:-**

A = +7 0000 0111
B = +12 + 0000 1100
0 0001 0011
Sign bit | Magnitude

⇒ **Case 2:- Smaller number is negative:-** A=-7 , B=+12

A=-7 ⇒ +7 = 0000 0111
-7 = 1111 1001 ← 2's complement of $(+7)_{10}$

A=-7 1111 1111 1001
B=+12 0000 0111
0000

A=-7 1111 1111 1001
B=+12 0000 1100
Discard first carry 1 0000 0101

And the correct answer is $\underbrace{0}_{\text{Sign bit}}\underbrace{000\ 0101}_{\text{Magnitude}} = (+5)_{10}$

⇒ Case 3: Bigger number negative:- A = +7 and B = -12

+12 = 0000 1100 Binary of +12

1111 0011 1's complement

+1

B = -12 = 1111 0100 2's complement

A = +7 = 0000 0111

1 1111 011

Sign bit Magnitude

Here the sign bit is 1 means it answer is negative and in the form of 2's complement, thus the original answer is given as,

1 1111 011

−1

1 1111 010

↓

1 0000 101 → $(-5)_{10}$

⇒ Case 4:- Both numbers negative:

+7 = 0000 0111

↓ Invert all bits

1111 1000 1's complement

+1

$(-7)_{10}$ = 1111 1001 2's complement

$(-12)_{10}$ = 1111 0100

1 1 1110 1101

Discard carry

Here sign bit is 1 but 1 carry is remaining thus the answer is -ve and have in its real form.

(−7)

+ (−12)

−19 = $(11101101)_2$

⑥ Binary Subtraction Using 1's and 2's complements :-

⇒ **Subtraction using 1's complement** : The steps for subtraction $(A)_2 - (B)_2$ using 1's complement are as follows.

* **Step I** :- Convert number to be subtracted $(B)_2$ to its 1's complement.

* **Step II** :- Add $(A)_2$ and 1's complement of $(B)_2$ using the rules of binary addition.

* **Step III** :- If final carry is 1, then add it to the result of addition obtained in step II to get the final result of $(A)_2 - (B)_2$. Note that if the final carry is 1 then the subtraction is +ve and in its true form.

* **Step IV** :- If the final carry produced in step 2 is 0 then the result obtained in step 2 is -ve and in the 1's complement form. So convert it into true form.

There are four possible cases depending on the magnitude and sign of the numbers.

Case 1 : No. A & B, both positive and A > B.
Case 2 : A & B, both positive & A < B.
Case 3 : Both nos are -ve.
Case 4 : A = B.

Case 1 :- Q. Perform $(9)_{10} - (4)_{10}$ using 1's complement method.

Solu. Step I → Convert $(4)_{10}$ into 1's complement,

$(4)_{10} = (0100)_2$

1's complement = $(1011)_2$

Step II → Add $(9)_{10}$ & 1's complement of $(4)_{10}$.

$(9)_{10} = 1001$

1's complement of $(4)_{10} = 1011$

$\boxed{1}\ 0100$

Step III → Here final carry is 1 so add final carry into the result of step II,

```
  1001
  1011
 [1]0100
   └──+1
 ------
  0101
```

Ans Answer is positive and its true form

Case 2: Q. Subtract $(9)_{10}$ from $(4)_{10}$ using 1's complement method.

Solu. Obtain 1's complement of $(9)_{10}$ as,

$(9)_{10} = (1001)_2$

1's complement of $(9)_{10} = 0110$

Now Add $(4)_{10}$ & 1's complement of $(9)_{10}$

$(4)_{10} = (100)_2$

```
  0100
  0110
 ------
  1010   → final carry is zero thus answer in -ve & 1's complement form.
```

So convert answer into its true form i.e,

```
1010
  ↓ Invert all bits.
0101
```

Hence $(4)_{10} - (9)_{10} = (0101) = (-5)_{10}$ Ans

Case 3: Q. Perform $(-4)_{10} - (-8)_{10}$ using 1's complement method.

Solu. $(-4)_{10} - (-8)_{10} = (-4)_{10} + (8)_{10} = (8)_{10} - (4)_{10}$

$(+4)_{10} = (0100)_2$

1's compt. of $(4)_{10} = (1011)_2$

$(8)_{10} = 1000$

1's compt. of $(4)_{10} = 1011$

```
  1 0011
    +  1
 -------
  0 1 0 0   Answer is +ve and its true form
```

$(-4)_{10} - (-8)_{10} = (0100)_2$ Ans

Case 4: Q. Subtract $(5)_{10}$ from $(5)_{10}$ using 1's complement.

Solu. $(5)_{10} = (0101)_2$

1's compt. of $(5)_{10} = (1010)_2$

$(5)_{10} = 0101$

1's compt. of $(5)_{10} = 1010$

```
  1111   final carry is 0 so answer is in 1's complement form.
```

1 1 1 1

↓ Invert all bits

0 0 0 0

Thus $(5)_{10} - (5)_{10} = (0000)_2$ Ans

⇒ Binary Subtraction using 2's complement Method :- Steps for subtraction $(A)_2 - (B)_2$ using 2's complement.

* Step I :- Add $(A)_2$ to the 2's complement of $(B)_2$.

* Step II :- If the carry is generated then the result is +ve and in its true form.

* Step III :- If the carry is not produced then the result is -ve and in its 2's complement form.

There are four possible cases depending on the magnitude and sign of the number.

Case 1 :- Both A & B are +ve with A>B.

Case 2 :- Both A&B are +ve with A<B.

Case 3 :- Both A&B are -ve.

Case 4 :- Both A&B are equal.

Case 1 :- Q. Perform $(9)_{10} - (5)_{10}$ using 2's complement method.

Solu. Obtain 2's complement of $(5)_{10}$

$(5)_{10} \longrightarrow (0101)_2 \longrightarrow$ 1010 (1's compt.)

+1

1011 (2's compt.)

$(9)_{10}$ = 1001

2's compt of $(5)_{10}$ = 1011

1 0100

Final carry shows, the answer is +ve and in its true form

Thus $(9)_{10} - (5)_{10} = (0100)_2$ Ans

Case 2 :- Q. Perform $(5)_{10} - (9)_{10}$ using the 2's complement Method.

Solu. $(9)_{10} \longrightarrow (1001)_2 \longrightarrow$ 0110 (1's compt.)

+1

0111 (2's compt.)

$(5)_{10}$ = 0101

2's compt of $(9)_{10}$ = 0111

0 1100

It indicates carry=0 & in its 2's complement

Thus Answer

$$\begin{array}{r} 1100 \\ -\ 1 \\ \hline 1011 \\ \downarrow \text{ Invert bits} \\ \hline 0100 \end{array}$$

$(5)_{10} - (9)_{10} = (-4)_{10} = (0100)_2$ Ans

Case 3 :- Q. Perform $(-4)_{10} - (-6)_{10}$ using the 2's complement method.

Solu. $(-4)_{10} - (-6)_{10} = (-4)_{10} + (6)_{10} = (6)_{10} - (4)_{10}$

$(4)_{10} \longrightarrow (0100)_2 \longrightarrow (1011)$ 1's compt.

$+1$

1100 2's compt.

$(6)_{10} = 0110$

2's compt of $(4)_{10} = 1100$

$\boxed{1}\ 0010$

Discard carry

Hence, $(-4)_{10} - (-6)_{10} = (0010)_2$ Ans

Case 4 :- Q. Perform $(5)_{10} - (5)_{10}$ using the 2's complement method.

Solu. $(5)_{10} \longrightarrow (0101)_2 \longrightarrow 1010$ 1's compt.

$+1$

1011 2's compt.

$(5)_{10} = 0101$

2's compt of $(5)_{10} = 1011$

$\boxed{1}\ 0000$

Discard final carry

$(5)_{10} - (5)_{10} = (0000)_2$ Ans

Q. Perform the subtraction using: a) 1's compt b) 2's compt.

if $A = \underset{26}{\underset{16\,8\,4\,2\,1}{11010}}$ & $B = \underset{16}{\underset{16\,8\,4\,2\,1}{10000}}$

a) $(01010)_2$ (10) b) $(01010)_2$ (10)

Q. Perform the following subtraction using 1's complement method.

$(0011.1001)_2 - (0001.1110)_2$

Solu. $(0001.1110)_2 \longrightarrow 1110.0001$ 1's complement

0011.1001

$\boxed{1}\ 0001.1010$

Final carry $\longrightarrow +1$

0001.1011 Ans

Q. Perform following subtraction using 1's and 2's complement.

$(1010)_2 - (1011)_2$ $\quad -(0001)_2$ $\quad -(0001)_2$

Q. Perform following binary operation using 2's complement method.

a) $(1010)_2 - (0101)_2$ $\quad$ b) $(1001)_2 - (1101)_2$

$(0101)_2$ $\quad (0100)_2$

Q. Perform the operation of subtraction with the following binary numbers using 2's complement.

a) 10010 − 10011 $\quad$ b) 100 − 110000 $\quad$ c) 11010 − 10000.

$-(00001)_2$ $\quad -(101100)_2$ $\quad (01010)_2$

⑦ Binary Coded Decimal (BCD) code :-

* In this code each decimal digit is represented by a 4-bit binary number. Thus BCD is a way to express each of the decimal digits with a binary code.

Decimal	0	1	2	3	4	5	6	7	8	9
BCD	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

← BCD nos are same as 4 bit binary nos →

⇒ Invalid BCD codes :- with four bits we can represent sixteen numbers (0000 to 1111). But in BCD code only the first ten of these are used (0000 to 1001).

The remaining six code combinations i.e 1010 to 1111 are invalid in BCD.

⇒ Conversion of big decimal numbers to BCD :- To express any decimal number to BCD, simply express each decimal digit with its equivalent 4-bit BCD code as shown in below.

Decimal Number: 9 8 $\quad$ 1 6 4

BCD equivalent: 1001 1000 $\quad$ 0001 0110 0100

space b/w two adjacent groups

⇒ Advantages of BCD codes :-

* It is very similar to decimal number.
* We need to remember binary equivalents of decimal numbers 0 to 9 only.

⇒ Disadvantages of BCD codes :-

* The addition and subtraction of BCD have different rules.
* The BCD arithmetic is little more complicated.
* BCD needs more number of bits than binary to represent the same decimal number. So BCD is less efficient than Binary.

Q. Convert decimal 8620 into BCD.

Solu.

BCD → 8 6 2 0 → 1000 0110 0010 0000

(8) BCD Arithmetic :-

⇒ BCD Addition :- There are four possible cases for BCD Addition.

* Case 1 :- Sum is equal to or less than 9 and carry 0.

Q. Obtain addition of $(2)_{10}$ & $(6)_{10}$ in BCD.

Solu.

	Decimal	BCD
	$(2)_{10}$	0010
	$(6)_{10}$	0110
Answer	$(8)_{10}$	0 1000

Sum is a valid BCD Number.

Final carry is 0

$(2)_{10} + (6)_{10} = (1000)_2$ Ans

* Case 2 :- Sum is grater than 9 and carry is 0.

Q. Obtain addition of $(7)_{10}$ and $(6)_{10}$ in BCD.

Solu.

	Decimal	BCD
	$(7)_{10}$	0111
	$(6)_{10}$	0110
Answer	$(13)_{10}$	0 1101

This is a invalid BCD number.

Final carry

If we get invalid BCD no. then add (6)$_{10}$ or (0110) to obtain answer.

Thus,

Invalid BCD ⟶ 1101

Add $(6)_{10}$ ⟶ + 0110

Final carry → 1 | 0011

0001 | 0011

1 | 3

Final answer:- $(7)_{10} + (6)_{10} = (13)_{10} = (0001\ 0011)_2$ Ans

* Case 3:- Sum less than or equal to 9 but carry 1

Q. Obtain $(9)_{10} + (8)_{10}$ in BCD.

Solu

Decimal		BCD
$(9)_{10}$	⟶	1001
$(8)_{10}$	⟶ +	1000
Answer = $(17)_{10}$		1 | 0001

0001 0001 Incorrect result (BCD)

Now add $(6)_{10}$ in incorrect BCD result.

0001 0001

\+ 0000 0110

0001 0111

1 | 7

$(9)_{10} + (8)_{10} = (17)_{10} = (0001\ 0111)$ Ans

* Case 4 :- Sum is grater than 9 but carry 1.

Q. Add $(57)_{10}$ and $(26)_{10}$ in BCD.

Solu

Decimal		BCD
$(57)_{10}$	⟶	0101 0111
$(26)_{10}$	⟶	0010 0110
Result = $(83)_{10}$		0111 | 1101 Incorrect Answer
		Valid BCD Invalid BCD

Now add $(6)_{10}$ in invalid BCD i.e

0111 1101

\+ 0000 0110

1000 0011

8 | 3

$\therefore (57)_{10} + (26)_{10} = (83)_{10} = (1000\ 001)$ Ans

Q. Add $(569)_{10}$ and $(687)_{10}$ in BCD.

Solu.

Decimal	BCD
	1 1 111
569	0101 0110 1001
+ 687	0110 1000 0111
1256	1011 1111 0000 Incorrect Answer
	Invalid BCD, Invalid BCD, Valid BCD with carry 1

Thus,

	1011	1111	0000	
+	0110	0110	0110	Add $(6)_{10}$
Final Carry →	1	0001	0101	0110
	↓	↓	↓	↓
	0001	2	5	6
	1			

$(569)_{10} + (687)_{10} = (1256)_{10} = (0001\ 0010\ 0101\ 0110)$ Ans

⇒ BCD subtraction :- The subtraction of two BCD nos A & B can be performed using one of the following methods.

* 9's complement * 10's complement.

* 9's complement :- The 9's complement of a BCD number can be obtained by subtracting it from 9.

For example 9's complement of 1 is 8.

It is performed as following (A-B).

* Step I :- Obtain the 9's complement of number B.

* Step II :- Add A and 9's complement of B.

* Step III :- If a carry is generated in step II then add it to the sum to obtain the final result. The carry is called as end around carry.

* Step IV :- If carry is not produced then the result is -ve hance take 9's complement of the result.

Q. Subtract $(3)_{10}$ from $(7)_{10}$ in BCD

Solu. Obtain 9's complement of $(3)_{10}$

⇒ $9 - 3 = (6)_{10}$

Now add $(7)_{10}$ and 9's complement of $(3)_{10}$.

	7	1 3
+	6	↳1
Final carry ←	1 3	4

Thus, $(7)_{10} - (3)_{10} = (4)_{10}$

Q. Perform the subtraction $(4)_{10} - (7)_{10}$ using 9's complement.

Solu. 9's complement of $(7)_{10} = 9 - 7 = (2)_{10}$

Now add $(4)_{10}$ & 9's complement of $(7)_{10}$,

$$\begin{array}{r} 4 \\ +\ 2 \\ \hline 6 \end{array} \qquad \begin{array}{r} 0100 \\ +\ 0010 \\ \hline 0110 \end{array}$$

Final carry ← Thus result is -ve & in 9's complement form.

So take 9's complement of result,

$$\begin{array}{r} 9 \\ -\ 6 \\ \hline 3 \end{array} \qquad \begin{array}{r} 1001 \\ 0110 \\ \hline 0011 \end{array}$$

$\therefore (4)_{10} - (7)_{10} = (-3)_{10} = -(0011)_2$ Ans

Q. Perform $(83)_{10} - (21)_{10}$ using 9's complement method.

$(83)_{10} - (21)_{10} = (62)_{10}$ Ans

Solu. Obtain 9's complement of $(21)_{10}$,

$$\begin{array}{r} 99 \\ -\ 21 \\ \hline 78 \end{array}$$

Now add $(83)_{10}$ & 9's complement of $(21)_{10}$ i.e

$$\begin{array}{r} 83 \\ 78 \\ \hline 161 \\ \hookrightarrow +1 \\ \hline 62 \end{array} \qquad \begin{array}{r} 1000\ 0011 \\ 0111\ 1000 \\ \hline 1111\ 1011 \end{array}$$

Invalid BCD, Invalid BCD

$$\begin{array}{r} 1111\ \ 1011 \\ 0110\ \ 0110 \\ \hline 1111\ \ 11 \\ \hline 1\ 0110\ \ 10001 \\ \hookrightarrow \qquad\quad +1 \\ \hline 0110\ \ 0010 \\ 6 \qquad 2 \end{array}$$

Final carry is one & the result is +ve & its true form

$\therefore (83)_{10} - (21)_{10} = (62)_{10} = (0110\ 0010)$ Ans

Q. Perform $(52)_{10} - (89)_{10}$ using 9's complement.

Solu. 9's complement of $(89)_{10} = 99 - 89 = (10)_{10}$

$(52)_{10} = 0101 \quad 0010$

$(10)_{10} = + \; 0001 \quad 0000$

$0110 \quad 0010$

Here final carry is 0, so the answer is negative and in the form of 9's complement. So take the 9's complement of answer as,

$1001 \quad 1001$

$- \; 0110 \quad 0010$

$0011 \quad 0111$ Final answer

$\downarrow \quad \downarrow$

$3 \quad 7$

Thus $(52)_{10} - (89)_{10} = -(37)_{10} = -(0011\ 0111)$ Ans

* Subtraction using 10's complement :- The 10's complement is obtain by adding 1 to the 9's complement. The 10's complement can be used to perform the BCD subtraction as follows.

* Step I :- Obtain the 10's complement of number to be subtracted.

* Step II :- Add the minuend to the 10's complement of subtrahend.

* Step III :- Discard carry. If carry is 1 then the answer is +ve and in its true form.

* Step IV :- If carry is not produced then the answer is -ve. So take 10's complement to get the answer.

Q. Perform the subtraction $(9)_{10} - (4)_{10}$ in BCD using the 10's complement.

Solu. Obtain 10's complement of $(4)_{10} = (9 - 4) + 1 = (6)_{10}$

Now add, $(9)_{10} = 1001$

$(6)_{10} = + \; 0110$

1111 Invalid BCD

So add $(6)_{10}$ in invalid BCD number as,

1111

0110

Final carry $\leftarrow$ (1) $0101 \rightarrow$ This answer is +ve and in true form

Thus, $(9)_{10} - (4)_{10} = (5)_{10} = (0101)$ Ans

Q. Perform $(3)_{10} - (8)_{10}$ in BCD using 10's complement.

Solu. 10's complement of $(8)_{10} = (9-8)+1 = 2$

Add $(3)_{10}$ & 10's complement of $(8)_{10}$ i.e

$$\begin{array}{rl} (3) & = 0011 \\ (2)_{10} & \overset{+}{=} 0010 \\ \hline & 0101 \end{array}$$

Here final carry is 0 thus result is negative & in 10's complement

Thus,

$$\begin{array}{rl} & 1001 \\ - & 0101 \\ \hline & 0100 \quad \text{9's complement} \\ & +1 \\ \hline & 0101 \quad \text{10's complement \& final Result} \end{array}$$

Hence $(3)_{10} - (8)_{10} = (-5)_{10} = (0101)$ Ans

Q. Perform $(54)_{10} - (22)_{10}$ in BCD using 10's complement.

Q. Perform $(22)_{10} - (54)_{10}$ in BCD using 10's complement.

Q. Convert the decimal number +21 & -21 into 10's & 9's complement.

Solu. 9's complement of +21 = 021 Ans

10's complement of +21 = 021 + 1 = 022 Ans

9's complement of -21 i.e $99 - 21 = 78$ Ans

10's complement of -21 i.e

$$\begin{array}{r} 78 \\ +1 \\ \hline 79 \end{array}$$ Ans

Q. Write 9's and 10's complement of the following numbers.

+9090, -3578, +136.8, -136.28

Solu. 9's complement of +9090 : 0000 1001 0000 1001 0000

9's complement of -3578 → 93578

Sign Bit

$$\begin{array}{r} 9999 \\ -\ 93578 \\ \hline 96421 \end{array}$$

Sign Bit — Magnitude.

9's complement of +136.8 → 0000 0001 0110 . 1000

9's complement of −136.28 →

$$\begin{array}{r} 999.99 \\ -\ 136.28 \\ \hline 9863.71 \end{array}$$

Sign Bit — Magnitude

10's complements

+9090 ⟶ 09091
−3578 ⟶ 96422
+136.8 ⟶ 0137.8
−136.28 ⟶ 9864.71 Ans

⑨ **Excess-3 code :-**

* Excess-3 (XS-3) code is a non-weighted code used to express decimal numbers as shown below.

Decimal No. → 8421 BCD $\xrightarrow[0011]{\text{Add}}$ XS-3 code

Decimal	BCD 8 4 2 1	Excess-3 BCD + 0011
0	0 0 0 0	0 0 1 1
1	0 0 0 1	0 1 0 0
2	0 0 1 0	0 1 0 1
3	0 0 1 1	0 1 1 0
4	0 1 0 0	0 1 1 1
5	0 1 0 1	1 0 0 0
6	0 1 1 0	1 0 0 1
7	0 1 1 1	1 0 1 0
8	1 0 0 0	1 0 1 1
9	1 0 0 1	1 1 0 0

⑩ **Gray Code :-** Gray code is another non-weighted code. It has a very special feature that only one bit will change

⇒ **Gray to Binary Conversion :-** For gray to binary conversion, follow the steps given below.

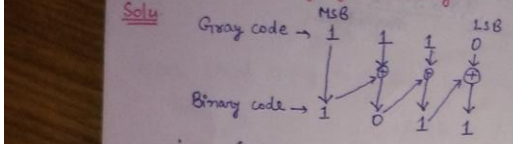
Step I :- The MSB of Gray and binary are same. So write it directly.

Step II :- Add binary MSB to the next bit of Gray code. Record the result and ignore the carries.

Step III :- Continue this process until the LSB is reached.

Q. Convert 1110 gray to binary.

Solu.



$\therefore (1110)_{Gray} = (1011)_2$ Ans

Rules for Addition in Gray to binary conversion

$0 \oplus 0$	$= 0$
$0 \oplus 1$	$= 1$
$1 \oplus 0$	$= 1$
$1 \oplus 1$	$= 0$

⇒ Binary to Gray Conversion :- For binary to gray conversion follow the following steps given below.

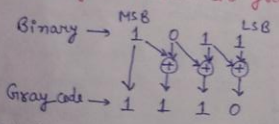
Step I :- Record the MSB as it is.

Step II :- Add this bit to the next position, recording the sum & neglecting any carry.

Step III :- Continue this process until the last digit is reached.

Q. Convert binary code 1011 to gray code.

Solu.

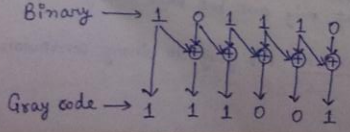


$\therefore (1011)_{Binary} = (1110)_{Gray}$ Ans

Q. Encode the decimal number 46 to Gray code.

Solu.

$(46)_{10} = (101110)_2$



$(46)_{10} = (101110)_2 = (111001)_{gray}$ Ans

(11) Error Detection and Correction codes :- When transmission of digital signals takes place b/w two systems then the noise can introduce an error in the binary bits travelling from one system to other system. This means a 0 may change to 1 or 1 may change to 0. These error can become a serious threat to the accuracy of the system. Therefore it is necessary to detect and correct the errors.

*How detect and correct error? :- For the detection or correction of these errors, one or more than one extra bits are added to the data bits at the time of transmitting. These extra bits are called as parity bits.

Duality theorem :- Starting with a Boolean relation, we can derive another Boolean relation, called its dual by following steps:

i) Changing each OR sign to an AND sign.

ii) Changing each AND sign to an OR sign.

iii) Complementing all 0's and 1's.

E.g. dual of $A \cdot 0 = 0$ is written as, $A + 1 = 1$.

And Applying duality theorem to $A \cdot (B + C) = A \cdot B + A \cdot C$

then $A + (B \cdot C) = (A + B) \cdot (A + C)$

Boolean Algebric Theorems/Laws/Expressions :-

S.N	Theorems	
1.	$A \cdot (B + C) = AB + AC$	
2.	$A + (BC) = (A + B)(A + C)$	
3.	$A + AB = A$	$1 + A = 1$
4.	$A \cdot (A + B) = A$	$A \cdot A = A$
5.	$A + \bar{A}B = A + B$	
6.	$A \cdot (\bar{A} + B) = A \cdot B$	$A\bar{A} = 0$
7.	$AB + A\bar{B} = A$	$A + \bar{A} = 1$
8.	$(A + B)(A + \bar{B}) = A$	
9.	$AB + \bar{A} \cdot C = (A + C) \cdot (\bar{A} + B)$	
10.	$(A + B) \cdot (\bar{A} + C) = (A \cdot C) + (\bar{A} \cdot B)$	
11.	$A \cdot B + \bar{A} \cdot C + B \cdot C = A \cdot B + \bar{A} \cdot C$	
12.	$(A + B) \cdot (\bar{A} + C) \cdot (B + C) = (A + B) \cdot (\bar{A} + C)$	

Boolean Algebra and Logic Gates

Boolean Laws:-

① a) An identity element with respect to (+), designated by 0.

$$A+0 = 0+A = A$$

b) An identity element w.r.t (·), designated by 1.

$$A \cdot 1 = 1 \cdot A = A$$

② a) Commutative w.r.t (+).

$$A+B = B+A$$

b) Commutative w.r.t (·).

$$A \cdot B = B \cdot A$$

③ a) (·) is distributed over (+).

$$A \cdot (B+C) = (A \cdot B) + (A \cdot C)$$

b) (+) is distributed over (·).

$$A+(B \cdot C) = (A+B) \cdot (A+C)$$

④ For every binary element A, there exists an element called complement of A (it is denoted by $\bar{A}$) such that,

$$A+\bar{A} = 1 \quad \text{and} \quad A \cdot \bar{A} = 0$$

Note that if $A=0$ then $\bar{A}=1$ and if $A=1$ then $\bar{A}=0$.

Inputs A	Inputs B	Output A·B
0	0	0
0	1	0
1	0	0
1	1	1

AND operator

Inputs A	Inputs B	Output A+B
0	0	0
0	1	1
1	0	1
1	1	1

OR operator

Input A	O/P $\bar{A}$
0	1
1	0

NOT operator

Commutative Law

Inputs A	Inputs B	A·B	B·A	A+B	B+A
0	0	0	0	0	0
0	1	0	0	1	1
1	0	0	0	1	1
1	1	1	1	1	1

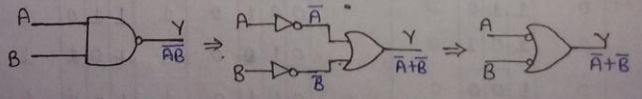
Distributed law

Inputs A	B	C	B+C	A·(B+C)	AB	AC	AB+AC
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	1	1	1	0	0	0	0
1	0	0	0	0	0	0	0
1	0	1	1	1	0	1	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

S.N.	Law Name	Statement of the law
1.	Commutative Law	$A \cdot B = B \cdot A$ $A + B = B + A$
2.	Associative law	$(A \cdot B) \cdot C = A \cdot (B \cdot C)$ $(A + B) + C = A + (B + C)$
3.	Distributive Law	$A \cdot (B + C) = (A \cdot B) + (A \cdot C)$
4.	AND Law	$A \cdot 0 = 0$ $A \cdot 1 = A$ $A \cdot A = A$ $A \cdot \bar{A} = 0$
5.	OR Law	$A + 0 = A$ $A + 1 = 1$ $A + A = A$ $A + \bar{A} = 1$
6.	Inversion Law	$\bar{\bar{A}} = A$
7.	Other Important Law	$A + BC = (A + B)(A + C)$ $\bar{A} + AB = \bar{A} + B$ $\bar{A} + A\bar{B} = \bar{A} + \bar{B}$ $A + AB = A$ $A + \bar{A}B = A + B$

De-Morgan's theorems:- The two theorems suggested by De-Morgan and which are extremely useful in Boolean algebra are as follows.

⇒ Theorem 1: $\overline{AB} = \bar{A} + \bar{B}$: NAND = Bubbled OR :- This theorem states that the complement of a product is equal to the addition of the complements. This theorem is illustrated in fig. below.



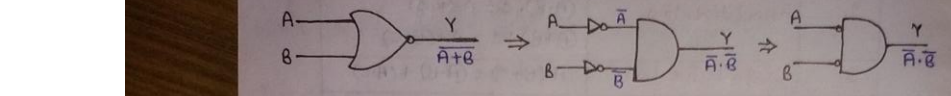
Verification of this theorem (Truth table)

A	B	AB	$\overline{AB}$	$\bar{A}$	$\bar{B}$	$\bar{A}+\bar{B}$
0	0	0	1	1	1	1
0	1	0	1	1	0	1
1	0	0	1	0	1	1
1	1	1	0	0	0	0

⇒ Theorem 2: $\overline{A+B} = \overline{A} \cdot \overline{B}$; NOR = Bubbled AND :- This theorem states that the complement of sum is equal to product of the complements. i.e

$$\overline{A+B} = \overline{A} \cdot \overline{B}$$

This theorem is illustrated in fig. below.



A	B	A+B	$\overline{A+B}$	$\overline{A}$	$\overline{B}$	$\overline{A} \cdot \overline{B}$
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

↑ Same ↑

Q. Demonstrate by means of truth tables the validity of the following theorems of Boolean algebra i) De Morgan's theorem for 3-variables.
ii) The distribution law of + over.

Solu: i) De Morgan's theorem for 3-variables.

Ya $\overline{A+B+C} = \overline{A} \cdot \overline{B} \cdot \overline{C}$ & $\overline{ABC} = \overline{A}+\overline{B}+\overline{C}$

A	B	C	A+B+C	$\overline{A+B+C}$	$\overline{A}$	$\overline{B}$	$\overline{C}$	$\overline{A} \cdot \overline{B} \cdot \overline{C}$
0	0	0	0	1	1	1	1	1
0	0	1	1	0	1	1	0	0
0	1	0	1	0	1	0	1	0
0	1	1	1	0	1	0	0	0
1	0	0	1	0	0	1	1	0
1	0	1	1	0	0	1	0	0
1	1	0	1	0	0	0	1	0
1	1	1	1	0	0	0	0	0

↑ Same ↑

A	B	C	ABC	$\overline{ABC}$	$\overline{A}$	$\overline{B}$	$\overline{C}$	$\overline{A}+\overline{B}+\overline{C}$
0	0	0	0	1	1	1	1	1
0	0	1	0	1	1	1	0	1
0	1	0	0	1	1	0	1	1
0	1	1	0	1	1	0	0	1
1	0	0	0	1	0	1	1	1
1	0	1	0	1	0	1	0	1
1	1	0	0	1	0	0	1	1
1	1	1	1	0	0	0	0	0

↑ Same ↑

ii) Distributive law of + over .

$A+B\cdot C = (A+B)\cdot(A+C)$

A	B	C	B·C	A+B·C	A+B	A+C	(A+B)(A+C)
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1

↑ same ↑

Q. Prove the following Boolean expressions :-

i) $A+AB=A$ ii) $A+\bar{A}B=A+B$ iii) $(A+B)(A+C)=A+BC$

iv) $(A+\bar{B}+AB)(A+\bar{B})(\bar{A}B)=0$ v) $ABCD+A\bar{B}CD=ACD$

Solu. i) $L.H.S = A+AB = A(1+B) = A = R.H.S$ ∵ $1+A=1$ (OR Law)

ii) $L.H.S = A+\bar{A}B = A(1+B)+\bar{A}B = A+AB+\bar{A}B$ ∵ $1+B=1$ (OR Law)

$= A+B(A+\bar{A})$ & $A+\bar{A}=1$ (OR Law)

$= A+B = R.H.S$

iii) $L.H.S = (A+B)(A+C) = AA+AC+AB+BC$ ∵ $A\cdot A=A$ (AND La

$= A+AC+AB+BC$

$= A(1+C)+AB+BC$ $1+C=1$ (OR Law

$= A+AB+BC$

$= A(1+B)+BC$ $1+B=1$ (OR Law)

$= A+BC = R.H.S$

iv) $L.H.S = (A+\bar{B}+AB)(A+\bar{B})(\bar{A}B)$

$= [A(1+B)+\bar{B}](A+\bar{B})(\bar{A}B)$

$= (A+\bar{B})(A+\bar{B})(\bar{A}B)$ ∵ $A\cdot A=A$

$= (A+\bar{B})(\bar{A}B)$ ∴ $(A+\bar{B})(A+\bar{B})=A+\bar{B}$

$= A\bar{A}B+\bar{A}\bar{B}B = 0 = R.H.S$ & $A\bar{A}=0$ & $B\bar{B}=0$

v) L.H.S = $ABCD + A\bar{B}CD = ACD(B+\bar{B})$
$= ACD$ $\because B+\bar{B}=1$ (OR Law)

Q. Simplify the following expression: $Y = \overline{(A\bar{B} + \bar{A} + AB)}$.

Solu. $Y = \overline{(A\bar{B} + \bar{A} + AB)}$ $\because \overline{A\bar{B}} = \bar{A}+\bar{B}$ De Morgan's theorem
$= \overline{(\bar{A}+\bar{B}+\bar{A}+AB)}$
$= \overline{(\bar{A}+\bar{A}+\bar{B}+AB)}$ $\because \bar{A}+\bar{A} = \bar{A}$ (OR Law)
$= \overline{(\bar{A}+\bar{B}+AB)}$ $\overline{A+B+C} = \bar{A}\cdot\bar{B}\cdot\bar{C}$ De Morgan's
$= \bar{\bar{A}}\cdot\bar{\bar{B}}\cdot\overline{AB}$
$= A\cdot B\cdot(\bar{A}+\bar{B})$
$= AB\bar{A} + AB\bar{B}$
$= A\bar{A}B + AB\bar{B}$ $\because A\bar{A} = 0$ (AND Law)
$= 0\cdot B + A\cdot 0 = 0$ Ans

Q. Simplify the following Boolean expressions:
i) $A\bar{B} + A\bar{B} + AB + \bar{A}\bar{B}$ ii) $A\bar{B}C + \bar{A}BC + ABC$

Solu. i) $Y = A\bar{B} + A\bar{B} + AB + \bar{A}\bar{B}$
$= A\bar{B} + AB + A\bar{B} + \bar{A}\bar{B}$
$= A(\bar{B}+B) + \bar{B}(A+\bar{A})$
$= A + \bar{B}$

$\overbrace{A+A=A}$
$Y = A\bar{B} + A\bar{B} + AB + \bar{A}\bar{B}$
$Y = A + AB + \bar{A}\bar{B}$
$Y = A\underbrace{(1+B)}_{1+A=1} + \bar{A}\bar{B}$
$Y = \underbrace{A + \bar{A}\bar{B}}_{A+\bar{A}B = A+B} = A+\bar{B}$

ii) $Y = A\bar{B}C + \bar{A}BC + ABC$
$= A\bar{B}C + BC(\bar{A}+A)$
$= A\bar{B}C + BC = C(A\bar{B}+B) = C(A+B)$ Ans

$\because A+\bar{A}B = A+B$
$\therefore A\bar{B}+B = A+B$

Q. Simplify $Y = (AB+C)(AB+D)$

Solu. $Y = (AB+C)(AB+D)$
$= AB\cdot AB + AB\cdot D + AB\cdot C + DC$
$= AB + ABD + ABC + DC$
$= AB(1+D+C) + DC = AB + CD$ Ans

$\because 1+C+D = 1$
& $A\cdot A = A$

Q. For the given function $F = x\bar{y} + x\bar{y}$, find the complement of F.

Solu. $F = x\bar{y} + x\bar{y} = x\bar{y}$ $\because A + A = A$ (OR Law)

$\bar{F} = \overline{x\bar{y}} = \bar{x} + \bar{\bar{y}} = \bar{x} + y$ Ans

Logic Gates :- Logic gates are the devices used as basic building blocks of all the digital circuits. The basic logic gates are NOT, AND & OR along with NOR, NAND, EX-OR etc. We have to use different laws, rules and theorems to analyze the digital circuits. A logic gate is an electronic circuit having one or more than one inputs and only one output.

Logic Gates

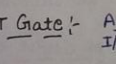
Basic Gates	Universal Gates	Special purpose Gates
* NOT	* NAND	* EX-OR
* AND	* NOR	* EX-NOR
* OR		

Positive and Negative Logic :-

Positive logic : Logic 0 (Low) = 0V
Logic 1 (High) = +5V

Negative logic : Logic 0 (Low) = +5V
Logic 1 (High) = 0V

NOT Gate :-

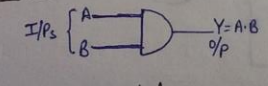


symbol

Input A	output Y
0	1
1	0

Truth table

AND Gate :-

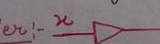


symbol

Inputs A	Inputs B	Output Y = A·B
0	0	0
0	1	0
1	0	0
1	1	1

Truth table

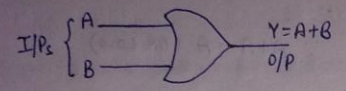
Buffer :-



x	f
0	0
1	1

Truth table

OR Gate :-

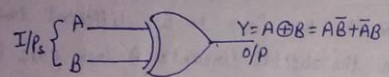


Symbol

I/Ps A	I/Ps B	O/P Y=A+B
0	0	0
0	1	1
1	0	1
1	1	1

Truth table

EX-OR Gate :-

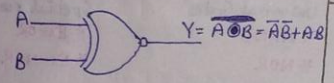


symbal

I/Ps A	I/Ps B	O/P Y=A⊕B
0	0	0
0	1	1
1	0	1
1	1	0

Truth table

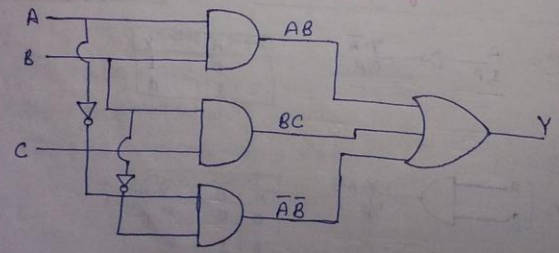
EX-NOR Gate :-



I/Ps A	I/Ps B	O/P Y=A⊙B
0	0	1
0	1	0
1	0	0
1	1	1

Q. Draw the combinational circuit using the basic gates to obtain the following output. $Y = AB + BC + \bar{A}\,\bar{B}$

Solu.



Q: Sketch the following equation. Using one AND gate and one OR gate only. $Y = (A+B)(A+C)$

Solu. $Y = (A+B)(A+C)$

$Y = AA + AC + AB + BC$

$Y = A + AC + AB + BC$

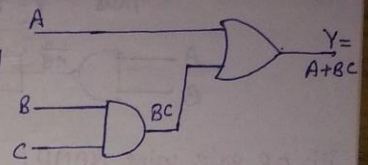
$Y = A + AB + BC$

$Y = A + BC$

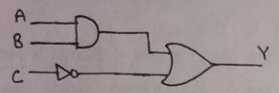
$\because AA = A$

$1 + C = 1$

$1 + B = 1$

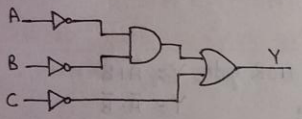


Q: Obtain the Boolean equation for the logic circuit shown in fig. below.



Solu. $Y = A \cdot B + \bar{C}$ Ans

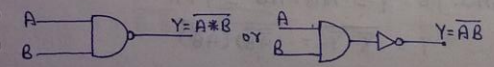
Q: Write the Boolean expression for the logic circuit shown in fig. below.



Solu. $Y = \bar{A} \cdot \bar{B} + \bar{C}$ Ans

UNIVERSAL Gates :- The NAND and NOR gates are called as Universal gates because it is possible to implement any boolean expression with the help of only NAND or only NOR gates. Hence a user can build any combinational circuit with the help of only NAND gates or only NOR.

⇒ NAND gate :-



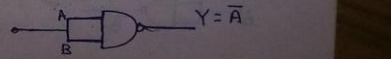
I/Ps		O/Ps
A	B	$Y = \overline{AB}$
0	0	1
0	1	1
1	0	1
1	1	0

⇒ Universality of NAND gate :-

i) NOT gate using NAND gate :-

$Y = \overline{A \cdot B} = \overline{A \cdot A} = \bar{A}$

Input A=B=A



ii) AND gate using NAND :- Expression for AND gate, $Y = A \cdot B$

Take double inversion of R.H.S i.e $Y = \overline{\overline{A \cdot B}}$

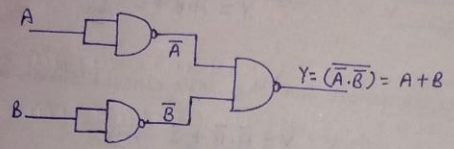
Thus $Y = A \cdot B = \overline{\overline{A \cdot B}}$

iii) OR gate using NAND :- Expression for OR gate, $Y = A + B$

Take double inversion i.e, $Y = \overline{\overline{A+B}}$

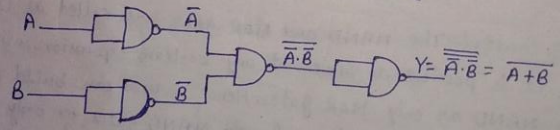
$Y = \overline{(\overline{A+B})} = \overline{(\overline{A} \cdot \overline{B})}$

Thus, $Y = A + B = \overline{(\overline{A} \cdot \overline{B})}$



iv) NOR gate using NAND :- Expression for NOR gate, $Y = \overline{A+B}$

Take double inversion of R.H.S i.e $Y = \overline{\overline{\overline{A} \cdot \overline{B}}}$ $Y = \overline{A} \cdot \overline{B}$



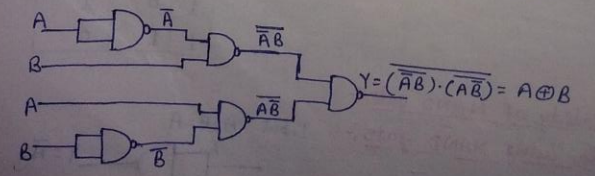
v) Ex-OR gate using NAND :- Expression for Ex-OR gate, $Y = A \oplus B = A\overline{B} + \overline{A}B$

Take double inversion of R.H.S i.e $Y = \overline{\overline{A\overline{B} + \overline{A}B}} = \overline{\overline{Y + X}}$

Let $X = \overline{A}B$

$Y = A\overline{B}$

$Y = \overline{\overline{X} \cdot \overline{Y}} = \overline{(\overline{A}B) \cdot (A\overline{B})}$

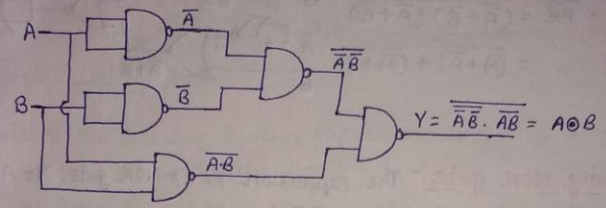


vi) Ex-NOR gate using NAND :- Expression for Ex-NOR gate

$$Y = A \odot B = \overline{A}\overline{B} + AB = X + Y$$

Take double inversion of R.H.S i.e

$$Y = \overline{\overline{X+Y}} = \overline{\overline{X} \cdot \overline{Y}} = \overline{\overline{A}\overline{B} \cdot \overline{AB}}$$



⇒ NOR gate :- Here we are going to illustrate the implementation of every logic operation using only NOR gates. So the Boolean expression of the given logic circuit must be first converted into the NOR format which is $Y = \overline{A+B}$

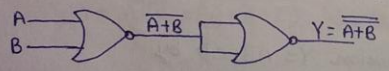
* NOT gate using NOR gate :- Input A = B = A

∵ A+A=A $\quad Y = \overline{A+A} = \overline{A}$

* OR gate using NOR gate :- The expression for OR gate, Y = A+B

Now take double inversion of R.H.S i.e,

$$Y = \overline{\overline{A+B}} = \overline{\overline{A} \cdot \overline{B}} = \overline{A} + \overline{B} = A+B$$

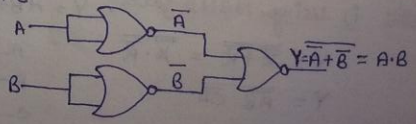


$\overline{A}B$

* AND gate using NOR gate :- The expression for AND gate, Y = A·B

Take double inversion of R.H.S i.e

$$Y = \overline{\overline{A \cdot B}} = \overline{\overline{A} + \overline{B}}$$

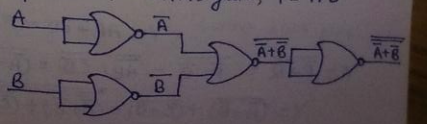


* NAND gate using NOR gate :- The expression for NAND gate, $Y = \overline{A \cdot B}$

$$Y = \overline{A} + \overline{B}$$

Take double inversion, $Y = \overline{(\overline{A}+\overline{B})}$

$$Y = \overline{(\overline{A} \cdot \overline{B})} = \overline{A \cdot B}$$



* **EX-OR using NOR gate:-** The expression for EX-OR gate, $Y = A \oplus B$

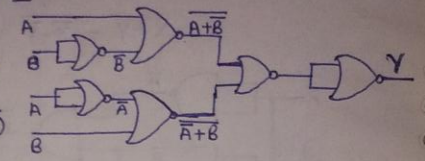
$Y = \bar{A}B + \bar{B}A = X + Y$ Let $X = \bar{A}B$ & $Y = A\bar{B}$

Now take double inversion on R.H.S

$$Y = \overline{\overline{X+Y}} = \overline{\bar{X} \cdot \bar{Y}}$$

$$Y = \overline{\overline{\bar{A}B} \cdot \overline{A\bar{B}}} = \overline{(\bar{\bar{A}} + \bar{B})(\bar{A} + \bar{\bar{B}})}$$

$$= \overline{(A+\bar{B})} + \overline{(\bar{A}+B)}$$



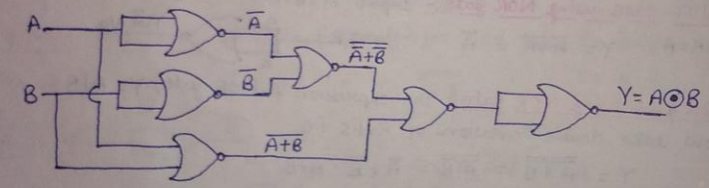
* **EX-NOR using NOR gate:-** The expression for EX-NOR gate, $Y = A \odot B$

$Y = \bar{A}\bar{B} + AB = X + Y$ Let $X = \bar{A}\bar{B}$ & $Y = AB$

Now take double inversion on R.H.S,

$$Y = \overline{\overline{\bar{A}\bar{B} + AB}} = \overline{(\bar{A}\bar{B}) \cdot (\overline{AB})}$$

$$= \overline{(A+B)(\bar{A}+\bar{B})} = \overline{(A+B)} + \overline{(\bar{A}+\bar{B})} = \overline{\overline{(A+B)} + \overline{(\bar{A}+\bar{B})}}$$



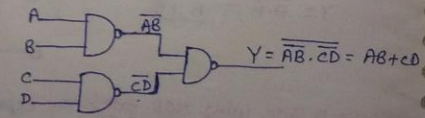
Q. With sketch realize the expression $Y = AB + CD$ by

i) NAND gates only ii) NOR gates only.

Solu: i) Using NAND gates, $Y = AB + CD = X + R$

$$Y = \overline{X+R} = \overline{\bar{X} \cdot \bar{R}}$$

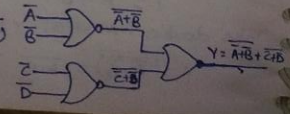
$$Y = \overline{\overline{AB} \cdot \overline{CD}}$$

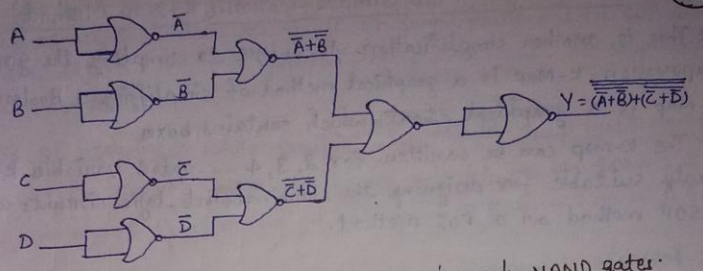


ii) Using NOR gates:- , $Y = AB + CD = P + Q$

$$Y = \overline{P+Q} = \overline{\bar{P} \cdot \bar{Q}} = \overline{\overline{AB} \cdot \overline{CD}} = \overline{(\bar{A}+\bar{B}) \cdot (\bar{C}+\bar{D})}$$

$$Y = \overline{(\bar{A}+\bar{B})} + \overline{(\bar{C}+\bar{D})} = \overline{\overline{(\bar{A}+\bar{B})} + \overline{(\bar{C}+\bar{D})}}$$



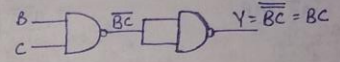


Q. Realize the following Boolean expression using only NAND gates.

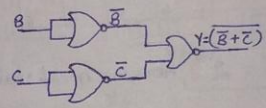
$$Y = (AB+BC)\cdot C$$

Solu. $Y = (AB+BC)\cdot C = ABC + BC\,C = ABC + BC = (A+1)BC = BC$ ($\because C\,C = C$)

$Y = BC \Rightarrow Y = \overline{\overline{BC}}$



Using NOR, $Y = \overline{\overline{BC}} = \overline{(\overline{B}+\overline{C})}$

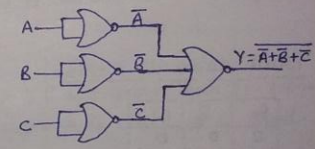


Q. Realize the following expression using only NOR gates: $Y = (ABC+\overline{B}\overline{C})C$

Solu. $Y = (ABC+\overline{B}\overline{C})\cdot C = ABC\,C + \overline{B}\overline{C}\,C$

$Y = ABC + \overline{B}\cdot 0 = ABC$ [$\because C\,C = C$ & $\overline{C}\,C = 0$]

$Y = \overline{\overline{ABC}} = \overline{(\overline{A}+\overline{B}+\overline{C})}$



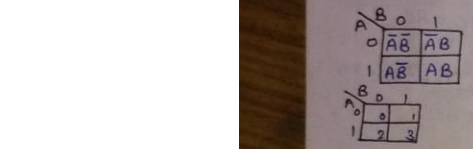
Karnaugh - Map Simplification (K-Map Method)

* This is another simplification technique to simplify the given boolean expression. K-Map is a graphical method of simplifying a Boolean equation. K-map is a graphical chart which contains boxes.

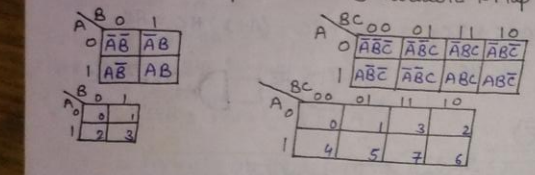
* The K-map can be written for 2, 3, 4 ---upto6 variables. K-Map is idealy suitable for designing the combinational logic circuits using either a SOP method or a POS method.

K-Map structures :-

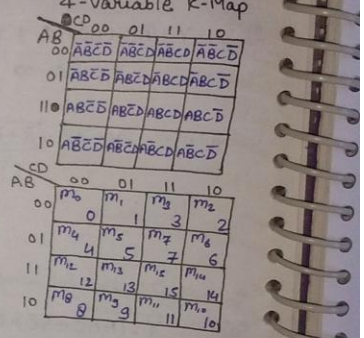
2-Variable K-Map



3-Variable K-Map



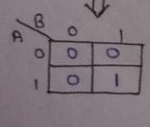
4-Variable K-Map



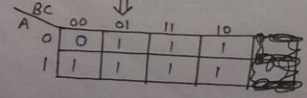
No. of Boxes = 2^n Here, n = no. of variables

Truth table to K-Map :-

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1



A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1



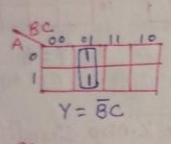
How does Simplification takes place :- Once we plot the logic fun. or truth table on a K-Map, we have to use the grouping technique for simply simplifying the logic function. Grouping means combining the terms in the adjacent cells. The grouping of either adjacent 1's or 0's result in the

Simplification ~~is in the POS form~~ of boolean expression. If we group the adjacent 1's then the result of simplification is in SOP form. And if the adjacent 0's are grouped, then the result of simplification is in the POS form.

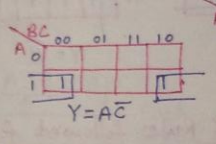
Way of Grouping :- while grouping, we should group most no. of 1's (or 0's). The grouping follows the binary rule i.e we can group 1, 2, 4, 8 ---- number of 1's or 0's. we can not group 3, 5, 6 ---- no. of 1's & 0's.

⇒ Grouping Two adjacent One's (Pairs) :- If we group two adjacent 1's on a K-map, to form a pair then the resulting term will have one less literal (variable) than the original term. That means by pairing two adjacent 1's we can eliminate one variable.

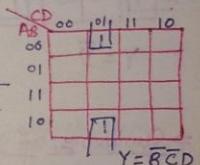
$Y = \bar{A}B$



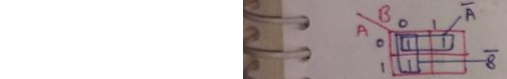
$Y = \bar{B}C$



$Y = A\bar{C}$



$Y = \bar{B}\bar{C}D$

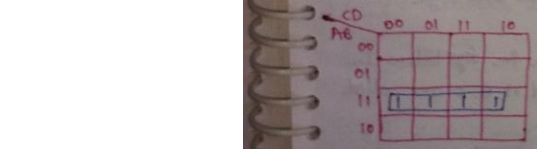


$Y = \bar{A} + \bar{B}$

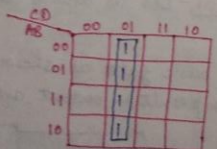
$Y = AC + \bar{A}B$

(This pair is not required because all the 1's are used for making group)

⇒ Grouping Four Adjacent Ones (Quad) :- If we group four 1's from the adjacent cells of a K-map then the group is called as a Quad.

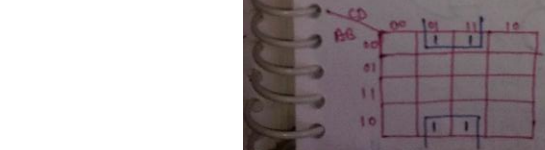


$Y = AB$



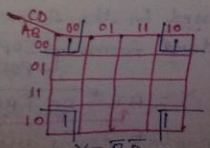
$Y = \bar{C}D$

$Y = BD$

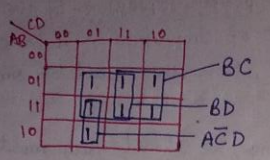


$Y = \bar{B}D$

$Y = B\bar{D}$

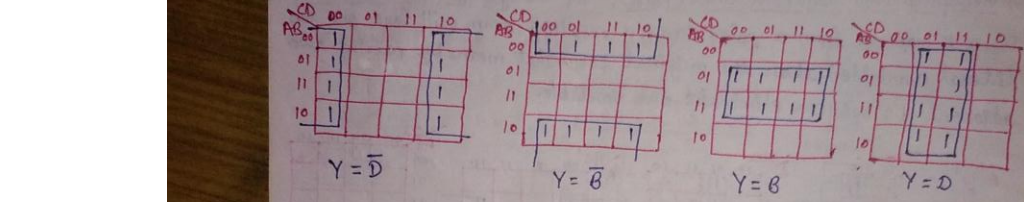


$Y = \bar{B}\bar{D}$



$Y = BC + BD + A\bar{C}D$

⇒ **Grouping Eight Adjacent Ones (Octet):-** It is possible to form a group of eight adjacent ones. Such a group is called as octet.



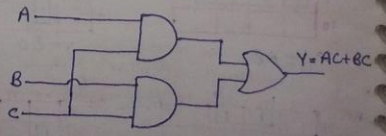
⇒ **Summary of Rules followed for K-Map Simplification:-**

* No zeros allowed
* No diagonals
* Only power of 2 number of cells in each group.
* Group should be as large as possible.
* Overlapping allowed.
* Wrap around allowed

SOP and POS Representation

⇒ **Sum-of-Products (SOP) form:-** Let us assume that the logic expression is, $Y = AC + BC$

Then it can be realized using basic gates as shown in fig. below. In this Boolean expression, Y is the result or output and A, B, C are called as input or literals.



Any logic expression may be expressed in the following two standard forms. → SOP form, → POS form.

If $Y = AB + AC + BC$ (Sum, Product terms)

Here, product terms are sum to each other so this form is called

Sum of product (SOP) form. other examples of SOP form are

$Y = ABC + BCD + C\bar{D}$

$A = X\bar{Y} + XYZ + X\bar{Z}$

$Y = P\bar{Q} + \bar{P}QR + P\bar{R}$ etc.

⇒ Product of Sum (POS) form :- If $Y = (\bar{A}+B)\cdot(B+C)\cdot(A+\bar{C})$ — Sum terms; Product

Here, sum terms are product to each other so this form is called as product of sum (POS) form. other examples of POS form are,

$Y = (A+B+C)\cdot(B+C+D)\cdot(C+\bar{D})$

$A = (\bar{X}+Y)\cdot(X+\bar{Y}+Z)\cdot(X+\bar{Z})$ etc.

⇒ Standard or Canonical SOP and POS form :- Let us consider the SOP and POS expressions as following.

Standard SOP form: $Y = \underline{ABC} + \underline{A\bar{B}\bar{C}} + \underline{\bar{A}BC}$ — Each product term consist of all the i/Ps in the complement or uncompl: form.

Standard POS form: $Y = \underline{(A+B+C)}\cdot\underline{(A+\bar{B}+\bar{C})}\cdot\underline{(\bar{A}+B+C)}$ — Each sum term consist of all the i/Ps in the complement or uncompl: form

Example. Convert the expression $Y = AB + A\bar{C} + BC$ into standard SOP form.

Solu Step I st :- $Y = AB + A\bar{C} + BC$ (C, B, A → Missing terms (i/Ps or literal))

Step II nd :- AND each term with (Missing literal + its complement) :-

$Y = AB\cdot(C+\bar{C}) + A\bar{C}(B+\bar{B}) + BC(A+\bar{A})$

$= \underset{7}{ABC} + \underset{6}{AB\bar{C}} + \underset{6}{AB\bar{C}} + \underset{4}{A\bar{C}\bar{B}} + \underset{7}{ABC} + \underset{3}{\bar{A}BC}$

$= (ABC + ABC) + (AB\bar{C} + A\bar{B}\bar{C}) + \bar{A}BC + A\bar{B}\bar{C}$

$\boxed{Y = ABC + AB\bar{C} + \bar{A}BC + A\bar{B}\bar{C}}$ Here each term contains all the i/Ps or literal so, its into canonical SOP form

Example: Convert the expression $Y=(A+B)(A+C)(B+\bar{C})$ into canonical form (POS).

Solu: **Step I :-** $Y=(A+B)(A+C)(B+\bar{C})$

$\downarrow \quad \downarrow \quad \downarrow$

$C \quad B \quad A \longrightarrow$ Missing inputs

Step II :- OR each term with (Missing literal & its complement)

$$Y=(A+B+C\bar{C})(A+B\bar{B}+C)(A\bar{A}+B+\bar{C})$$

$\because A+BC=(A+B)(A+C)$

$$\therefore \quad Y=(A+B+C)(A+B+\bar{C})(A+B+C)(A+\bar{B}+C)(A+B+\bar{C})(\bar{A}+B+\bar{C})$$

$$\boxed{Y=(A+B+C)(A+B+\bar{C})(A+\bar{B}+C)(\bar{A}+B+\bar{C})} \leftarrow \text{canonical POS form}$$

Each term contains all the literals

⇒ **Concept of Minterm and Maxterm :-** Each individual term in the canonical SOP form is called as minterm. While each individual term in the canonical POS form is called as Maxterm.

Canonical SOP → $Y=ABC+A\bar{B}\bar{C}+\bar{A}B\bar{C}$ — Each individual term is called minterm

Canonical POS → $Y=(A+B+C)\cdot(A+\bar{B}+\bar{C})\cdot(\bar{A}+B+\bar{C})$ — Each individual term is called Maxterm

Note: The no. of minterm and Maxterm $= 2^n$; n = no. of literal or inputs

Variable			Minterms	Maxterms
A	B	C	$m_{i=0,1,2,\ldots,n-1}$	$M_{i=0,1,\ldots,n-1}$
0	0	0	$\bar{A}\bar{B}\bar{C}=m_0$	$A+B+C=M_0$
0	0	1	$\bar{A}\bar{B}C=m_1$	$A+B+\bar{C}=M_1$
0	1	0	$\bar{A}B\bar{C}=m_2$	$A+\bar{B}+C=M_2$
0	1	1	$\bar{A}BC=m_3$	$A+\bar{B}+\bar{C}=M_3$
1	0	0	$A\bar{B}\bar{C}=m_4$	$\bar{A}+B+C=M_4$
1	0	1	$A\bar{B}C=m_5$	$\bar{A}+B+C=M_5$
1	1	0	$AB\bar{C}=m_6$	$\bar{A}+\bar{B}+C=M_6$
1	1	1	$ABC=m_7$	$\bar{A}+\bar{B}+\bar{C}=M_7$

Q. Express following function in product of maxterms (POS):

$F(x,y,z) = (xy+z)(y+xz)$.

Solu. $F(x,y,z) = (xy+z)(y+xz) = xyy + xyxz + yz + xzz$

$= xy + xyz + yz + xz$

$F(x,y,z) = xy(z+\bar{z}) + xyz + yz(x+\bar{x}) + xz(y+\bar{y})$

$= xyz + xy\bar{z} + xyz + xyz + \bar{x}yz + xyz + x\bar{y}z$

$F(x,y,z) = x\bar{y}z + xy\bar{z} + \bar{x}yz + xyz = m_7 + m_6 + m_3 + m_5$

$F(x,y,z) = \sum m(3,5,6,7) \longrightarrow$ canonical SOP form.

Note: The canonical SOP and canonical POS form have a complementry relationship. Thus, $F(x,y,z) = \Pi M(0,1,2,4)$

$F(x,y,z) = (x+y+z)(x+y+\bar{z})(x+\bar{y}+z)(\bar{x}+y+z)$ Ans

⇒ Writing SOP and POS forms for a given Truth table:-

* For SOP form:-

Step I:- Consider only those combinations of inputs which correspond to Y = 1.

Step II:- Write down a product term for each combination

Step III:- OR all these product terms to get the canonical SOP.

* For POS form:-

Step I:- Consider only those combinations of inputs which correspond to Y = 0.

Step II:- Write the maxterms only for such combinations.

Step III:- AND these maxterms to obtain the expression in canonical POS form

Q. For the truth table given below write the logical expression in the canonical SOP and POS form:

A	B	C	Y	
0	0	0	0	→ $\bar{A}+\bar{B}+\bar{C}$
0	0	1	1	→ $\bar{A}\bar{B}C$
0	1	0	0	→ $A+\bar{B}+C$
0	1	1	0	→ $A+\bar{B}+\bar{C}$
1	0	0	1	→ $A\bar{B}\bar{C}$
1	0	1	0	→ $\bar{A}+B+\bar{C}$
1	1	0	1	→ $AB\bar{C}$
1	1	1	1	→ ABC

Solu. So, $Y = \bar{A}\bar{B}C + A\bar{B}\bar{C} + AB\bar{C} + ABC$ (m_1, m_4, m_6, m_7)

$Y = \sum m(1,4,6,7) \longrightarrow$ SOP form

$Y = (A+B+\bar{C})(A+\bar{B}+\bar{C})(A+\bar{B}+C)(\bar{A}+B+\bar{C})$ (M_0, M_2, M_3, M_5)

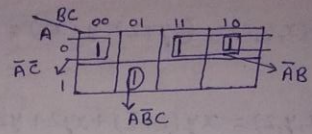
$Y = \Pi M(0,2,3,5) \longrightarrow$ POS form

Q: A logical expression in the canonical SOP form is as follows:

$Y = \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + \bar{A}BC + A\bar{B}C$ Minimize it using K-Map Method.

Solu. $Y = \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + \bar{A}BC + A\bar{B}C$

$Y = \Sigma m(0,2,3,5)$



Thus, minimized expression is,

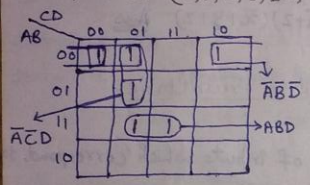
$\boxed{Y = A\bar{B}C + \bar{A}B + \bar{A}\bar{C}}$ Ans

Q: For logical expressions given below draw the K-Map and obtain the simplified logical expression i) $Y = \Sigma m(0,1,2,5,13,15)$

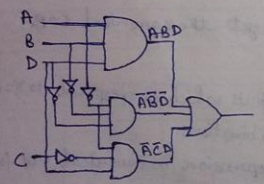
ii) $Y = \Sigma m(1,5,7,9,11,13,15)$

And Realization with basic gates.

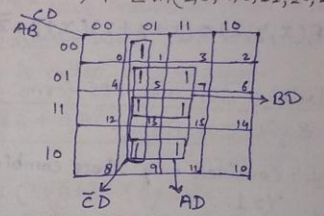
Solu: i) $Y = \Sigma m(0,1,2,5,13,15)$



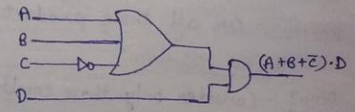
$Y = ABD + \bar{A}\bar{B}\bar{D} + \bar{A}\bar{C}D$



ii) $Y = \Sigma m(1,5,7,9,11,13,15)$

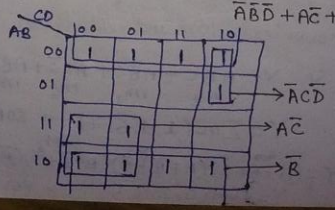


$Y = AD + BD + \bar{C}D = (A+B+\bar{C})\cdot D$



Q: Minimize the logic equation given below, $Y = \bar{A}\bar{B}CD + \bar{A}BC\bar{D} + \bar{A}B\bar{C} + \bar{A}\bar{B}\bar{D} + A\bar{C} + \bar{B}$

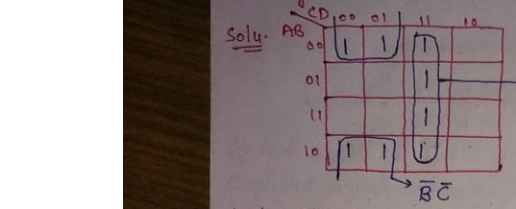
Solu: Thus, $Y = \bar{B} + A\bar{C} + \bar{A}C\bar{D}$ Ans



Quine-McCluskey Minimization Method (Tabular Method) :-

Q: Simplify the following Boolean expression using K-Map and verify it using the Quine-McCluskey method. $Y(A,B,C,D) = \sum m(0,1,3,7,8,9,11,15)$

Solu.



$Y = \bar{B}\bar{C} + CD$

Using Tabular Method :-

Step I :-

Decimal	Binary	Decimal	Binary
0	0000	1	0001
3	0011	7	0111
8	1000	9	1001
11	1011	15	1111

Step II :- Now arrange all the terms according to the no. of 1's

Group	Minterm	A	B	C	D
1	m_0	0	0	0	0
2	m_1	0	0	0	1
	m_8	1	0	0	0
3	m_3	0	0	1	1
	m_9	1	0	0	1
4	m_7	0	1	1	1
	m_{11}	1	0	1	1
5	m_{15}	1	1	1	1

Step III :- Combination of minterms into group of two.

Group	Minterm	A	B	C	D
1	$m_0 - m_1$	0	0	0	–
	$m_0 - m_8$	–	0	0	0
2.	$m_1 - m_3$	0	0	–	1

	$m_1 - m_9$	–	0	0	1
	$m_8 - m_9$	1	0	0	–
3.	$m_3 - m_7$	0	–	1	1
	$m_3 - m_{11}$	–	0	1	1
	$m_9 - m_{11}$	1	0	–	1
4.	$m_7 - m_{15}$	–	1	1	1
	$m_{11} - m_{15}$	1	–	1	1

Step IV :- Combination of minterm into group four.

Group	Minterm	A	B	C	D
1.	$m_0 - m_1 - m_8 - m_9$	–	0	0	–
	$m_0 - m_8 - m_1 - m_9$	–	0	0	–
2.	$m_1 - m_3 - m_9 - m_{11}$	–	0	–	1
	$m_1 - m_9 - m_3 - m_{11}$	–	0	–	1
3.	$m_3 - m_7 - m_{11} - m_{15}$	–	–	1	1
	$m_3 - m_{11} - m_7 - m_{15}$	–	–	1	1

Step V :- Prime Implicant table :-

Prime Implicant	0	1	3	7	8	9	11	15
0,1,8,9 ($\bar{B}\bar{C}$)	(X)	X			(X)	X		
1,3,9,11 ($\bar{B}D$)		X	X			X	X	
3,7,11,15 (CD)			X	(X)			X	(X)

$Y = \bar{B}\bar{C} + CD$ Ans

Q: Simplify the following Boolean expression using Quine-McCluskey Method. $Y(A,B,C,D) = \Sigma m(1,5,6,12,13,14) + d(2,4)$

UNIT-IInd Combinational Logic

① **Combinational Circuits:-** The output of combinational logic circuit at any instant of time, depends only on the levels present at input terminals. The combinational circuits do not use any memory. Hence the previous state of input does not have any effect on the present state of the circuit.

Thus a combinational circuit is a logic circuit the output of which depends only on the combination of the inputs. The circuit does not depend on the past value of inputs or outputs. Hence combinational circuits do not require any memory.

The block diagram of a combinational circuit is shown in fig. below.

Examples:-
i) Adders, Subtractors
ii) Comparator
iii) Code converters
iv) Encoder, decoders
v) Multiplexers, demultiplexers

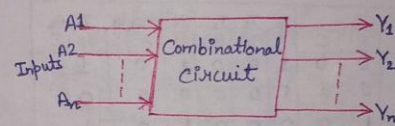
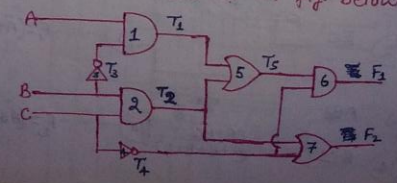


Fig.→Block diagram of combinational circuit

② **Analysis of Combinational circuit:-** In analysis of combinational circuit, the logic diagram of combinational circuit is given to us. We have to obtain the boolean expression for its output or write a truth table or explain the operation of circuit.

Procedure:- i) Write down the Boolean function for each input gate
ii) Obtain the Boolean expression at all other gates outputs.
iii) Write down the Boolean expression for the final output in terms of i/p

Q. Analyze the combinational circuit shown in fig. below.



Solu: Step I → Boolean expression for input gates outputs :-

$T_1 = A\bar{B}$, $T_2 = BC$, $T_3 = \bar{B}$, $T_4 = \bar{C}$

Step II → Boolean expression for remaining gates :-

$T_5 = T_1 + T_2 = A\bar{B} + BC$, $T_6 = T_5 \cdot T_4 = (A\bar{B} + BC)\cdot\bar{C} = A\bar{B}\bar{C} + BC\bar{C}$

$= A\bar{B}\bar{C}$

$T_7 = T_2 + T_4 = BC + \bar{C}$

Step III :- Boolean expression for final outputs :-

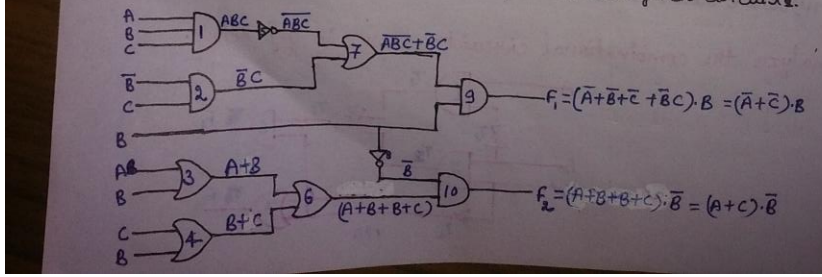
$F_1 = T_6 = A\bar{B}\bar{C}$

$F_2 = T_7 = BC + \bar{C}$

Step IV :- Write the truth table :-

Inputs			T_3	T_1	T_2	Outputs T_4	$T_5=T_6$	$T_7=F_2$
A	B	C	$\bar{B}$	$A\bar{B}$	BC	$\bar{C}$	$A\bar{B}\bar{C}$	$BC+\bar{C}$
0	0	0	1	0	0	1	0	1
0	0	1	1	0	0	0	0	0
0	1	0	0	0	0	1	0	1
0	1	1	0	0	1	0	0	1
1	0	0	1	1	0	1	1	1
1	0	1	1	1	0	0	0	0
1	1	0	0	0	0	1	0	1
1	1	1	0	0	1	0	0	1

Q: Obtain the boolean functions for outputs of the given circuits.



③ Design of Combinational logic :-

Step I :- You will be given a problem.

Step II :- Determine the number of inputs and outputs and assign letter symbols to inputs and outputs variables.

Step III :- Write a truth table relating this input and outputs.

Step IV :- Write a K-Map for each output and obtain the simplified boolean expression for each output.

Step V :- Draw the logic diagram.

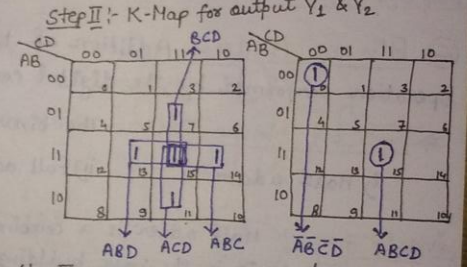
Q) A circuit has four inputs and two outputs. One of the outputs is high when majority of inputs are high. The second output is high only when all inputs are of same type. Design the combinational circuit.

Solu.

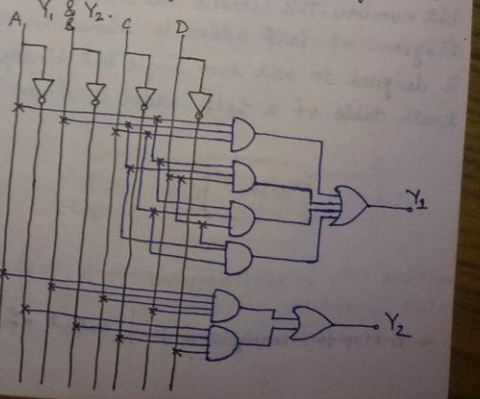
Step I :- Truth table :-

Inputs				Outputs	
A	B	C	D	Y_1	Y_2
0	0	0	0	0	1
0	0	0	1	0	0
0	0	1	0	0	0
0	0	1	1	0	0
0	1	0	0	0	0
0	1	0	1	0	0
0	1	1	0	0	0
0	1	1	1	1	0
1	0	0	0	0	0
1	0	0	1	0	0
1	0	1	0	0	0
1	0	1	1	1	0
1	1	0	0	0	0
1	1	0	1	1	0
1	1	1	0	1	0
1	1	1	1	1	1

Step II :- K-Map for output Y_1 & Y_2



Step III :- Design combinational logic CKT for Y_1 & Y_2.



Q: A Boolean function has three input variables and one output variable. The output for first four input conditions is logic 1. For rest of input condition output is logic 0.

i) Draw the truth table ii) Simplify using K-Map.

Q: Design a combinational logic circuit that will allow input signal A to pass through the output only when the control inputs B and C are same otherwise the output is high.

Q: The input to combinational logic circuit is a 4-bit binary number. Design the logic circuit with two output (Y_1 & Y_2) for the following conditions

i) $Y_1 = 1$ if the input binary is 5 or less than 5.

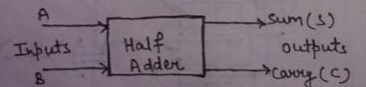
ii) $Y_2 = 0$ if input binary number is 9 or more than 9.

④ Binary Adders:- Addition of two binary digits is most basic operation performed by the digital computers.

⇒ Types of binary adders:- The binary adders are of two types.

i) Half adder ii) Full adder.

i) Half Adder:- Half adder is a combinational logic circuit with two inputs and two outputs. It is the basic building block for addition of two single bit numbers. This circuit has two outputs namely carry and sum. The block diagram of half adder is shown in fig. below. The half adder circuit is designed to add two single bit binary numbers A and B. Hence the truth table of a half adder is shown in fig. below.

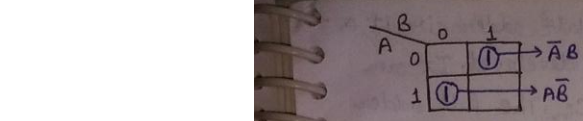


i) Block diagram

Inputs		Outputs	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

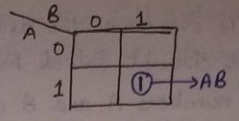
ii) Truth-Table

* K-Map for carry and sum output of half adder are shown as,



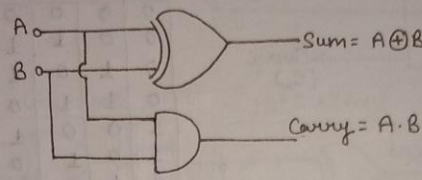
$Sum = \bar{A}B + A\bar{B}$

$= A \oplus B$



$Carry = AB$

Thus the circuit diagram for half adder is expressed as,

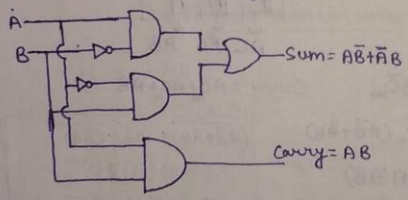


Half adder circuit diagram

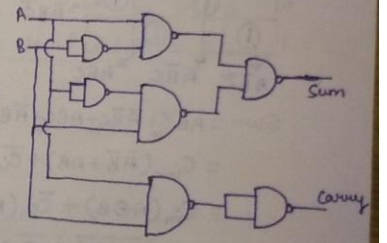
⇒ Half adder using basic gates:-

$Sum = \bar{A}B + A\bar{B}$

$Carry = AB$



⇒ Half adder using only NAND gates:



⇒ Disadvantage of Half Adder:- For adding two 2-bits numbers A and B is shown in above fig. Let $A = A_1 A_0$ & $B = B_1 B_0$

Then sum,

$A_1 \quad A_0$

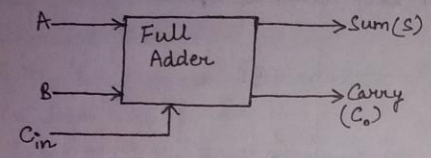
$+ \quad B_1 \quad B_0$

C_0 ← Carry generated from addition of A_0 & B_0

$C_1 \quad S_1 \quad S_0$

* A half adder can add A_0 and B_0 to produce S_0 and C_0. But the addition of next bits requires the addition of A_1, B_1, and C_0. The addition of three bits is not possible to perform by using a half adder. Hence we can not use a half adder.

ii) **Full Adder**:- To overcome the drawback of Half adder circuit, a 3 single bit adder circuit called Full adder is developed. It can add two one-bit numbers A and B and carry C_{in}. The full adder is a three input and two output combinational circuit. The block diagram of a full adder is shown in fig. below.

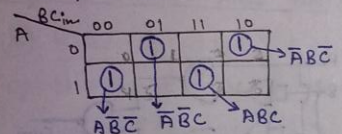


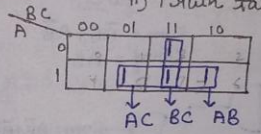
i) Block diagram

Inputs			Outputs	
A	B	C_{in}	S	C_o
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

ii) Truth table

* K-Map for outputs,





$$\text{Sum} = A\bar{B}\bar{C}_{in} + \bar{A}\bar{B}C_{in} + ABC_{in} + \bar{A}B\bar{C}_{in}$$

$$= C_{in}(\bar{A}\bar{B} + AB) + \bar{C}_{in}(A\bar{B} + \bar{A}B)$$

$$= C_{in}(A \odot B) + \bar{C}_{in}(A \oplus B)$$

$$= C_{in}\overline{(A \oplus B)} + \overline{C_{in}}(A \oplus B)$$

$$= C_{in} \oplus A \oplus B$$

$$\text{Carry} = AC_{in} + BC_{in} + AB$$

$$\overline{(A\bar{B} + \bar{A}B)} = \overline{(A\bar{B})} \cdot \overline{(\bar{A}B)}$$

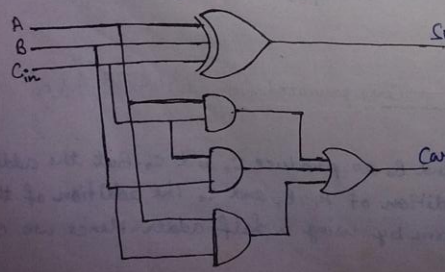
$$= (\bar{A} + B)(A + \bar{B})$$

$$= \bar{A}A + \bar{A}\bar{B} + AB + B\bar{B}$$

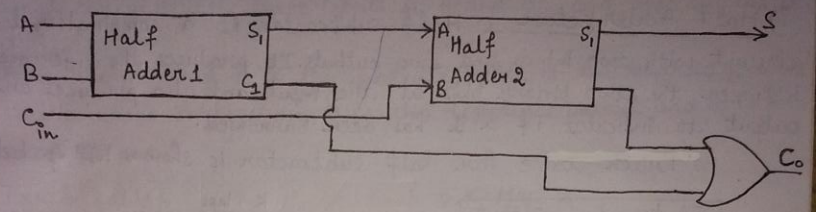
$$= 0 + \bar{A}\bar{B} + AB + 0$$

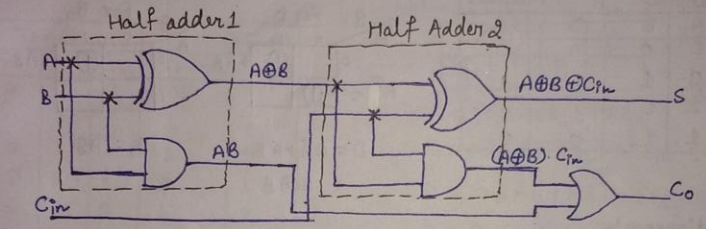
$$= \bar{A}\bar{B} + AB = A \odot B$$

* The circuit diagram of Full adder is shown in fig. below.



⇒ Full adder using Half adder :- The full adder circuit can be constructed using two half adders as shown in fig. below.





■ Full adder using Half adders

$$\boxed{S = A \oplus B \oplus C_{in}}$$

$$C_o = (A \oplus B) \cdot C_{in} + AB$$

$$= (A\bar{B} + \bar{A}B) \cdot C_{in} + AB(C_{in} + 1)$$

$$= A\bar{B}C_{in} + \bar{A}BC_{in} + ABC_{in} + AB \qquad 1 + A = 1$$

$$= A\bar{B}C_{in} + BC_{in}(\bar{A} + A) + AB$$

$$= A\bar{B}C_{in} + BC_{in} + AB(1 + C_{in})$$

$$= A\bar{B}C_{in} + BC_{in} + AB + ABC_{in}$$

$$= AC_{in}(\bar{B} + B) + BC_{in} + AB$$

$$\boxed{C_o = AC_{in} + BC_{in} + AB}$$ Proved

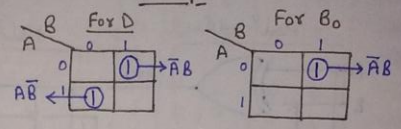
⑤ Binary Subtractors :- The binary subtractors are two types
i) Half ~~adder~~ subtractor ii) Full subtractor

i) Half Subtractor :- Half subtractor is a combinational circuit with two inputs and two outputs. It produces the difference between the two binary bits at the input and also produces an output to indicates if a 1 has been borrowed.

Truth table for half subtractor is shown below.

Inputs		Outputs	
A	B	D	B_o
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

K-Maps

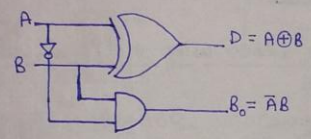


$D = A\bar{B} + \bar{A}B$

$= A \oplus B$

$B_o = \bar{A}B$

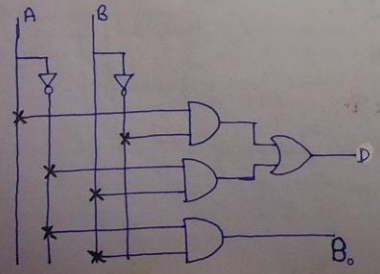
Logic diagram :-



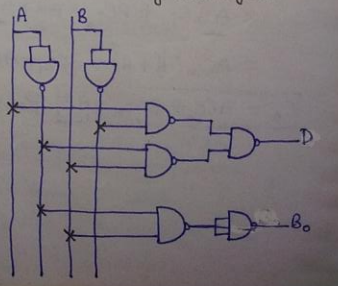
Half subtractor

⇒ Disadvantage of Half ~~a~~subtractor :- Half subtractor can only perform the subtraction of two binary bits. But while performing the subtraction it does not take into account the borrow of the lower significant stage.

⇒ Half subtractor using basic gates



⇒ Half adder using NAND gates



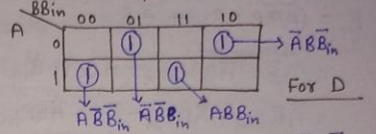
ii) **Full subtractor** :- The disadvantages of a half subtractor is overcome by the full subtractor The full subtractor is a combin-ational circuit with three inputs A, B and B_{in} and two outputs D and B_o.

* A is the minuend, B is the subtrahend, B_{in} is the borrow produce by the previous subtraction, D is the difference output and B_o is the borrow output.

⇒ **Truth table** :-

Inputs			Outputs	
A	B	B_{in}	$D = A-B-B_{in}$	B_o
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

⇒ **K-Maps** :-



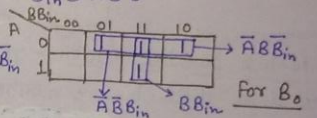
$D = A\bar{B}\bar{B}_{in} + \bar{A}\bar{B}B_{in} + ABB_{in} + \bar{A}BB_{in}$

$D = B_{in}(\bar{A}\bar{B} + AB) + \bar{B}_{in}(A\bar{B} + \bar{A}B)$

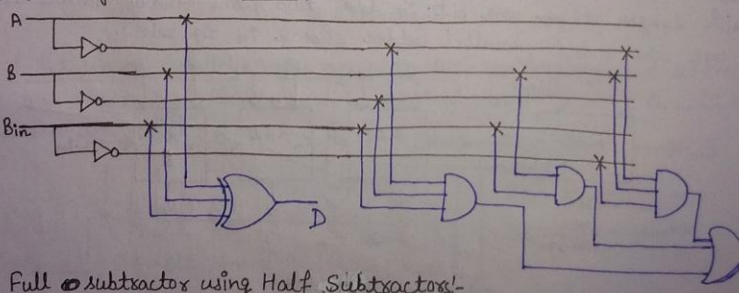
$D = B_{in}(A \odot B) + \bar{B}_{in}(A \oplus B)$

$D = B_{in} \oplus A \oplus B$

$B_o = \bar{A}\bar{B}B_{in} + BB_{in} + \bar{A}B\bar{B}_{in}$

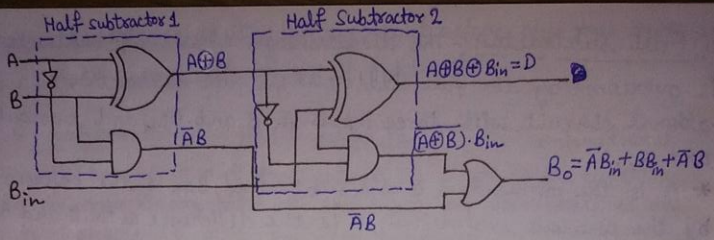


⇒ **Logic diagram for full subtractor** :-



⇒ **Full subtractor using Half Subtractors** :-

* Fig. below shows the implementation of a full subtractor using two half subtractors.



Here $\boxed{D = A \oplus B \oplus B_{in}}$

$$B_o = \overline{(A \oplus B)} \cdot B_{in} + \bar{A}B$$

$$= (\overline{A\bar{B} + \bar{A}B}) \cdot B_{in} + \bar{A}B(1 + B_{in})$$

$$= \bar{A}\bar{B}B_{in} + \bar{A}BB_{in} + \bar{A}B + \bar{A}BB_{in}$$

$$= \bar{A}B_{in}(\bar{B} + B) + ABB_{in} + \bar{A}B(1 + B_{in})$$

$$= \bar{A}B_{in} + ABB_{in} + \bar{A}BB_{in} + \bar{A}B$$

$$= \bar{A}B_{in} + (A + \bar{A}) \cdot BB_{in} + \bar{A}B \Rightarrow B_o = \boxed{\bar{A}B_{in} + BB_{in} + \bar{A}B}$$

⇒ N-bit parallel Adder :- The full adder is capable of adding only two single digit binary numbers along with a carry input.

* But in practice we need to add binary numbers which are much longer than one bit. To add two n-bit binary no. we need to use the n-bit parallel adder shown in fig. below.

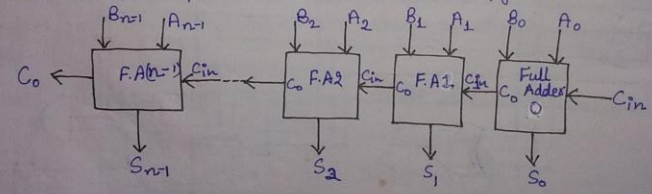


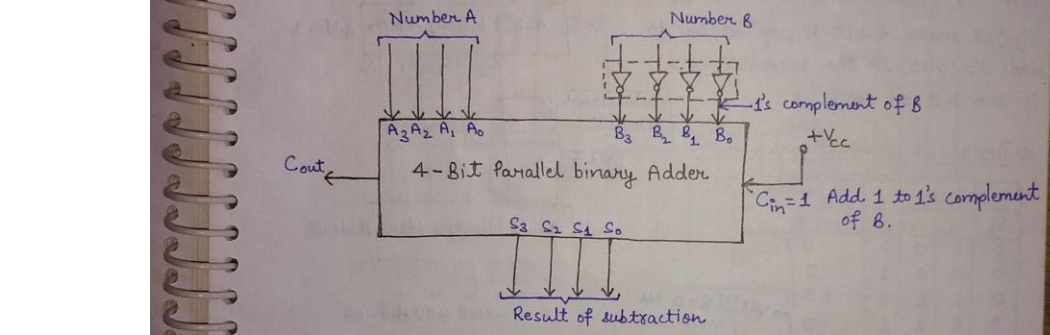
Fig → N-bit parallel adder

⇒ N-bit Parallel Subtractor :- The subtraction can be carried out by taking the 1's and 2's complement of the number to be subtracted. For example we can ~~support~~ perform the subtraction (A-B) by adding either

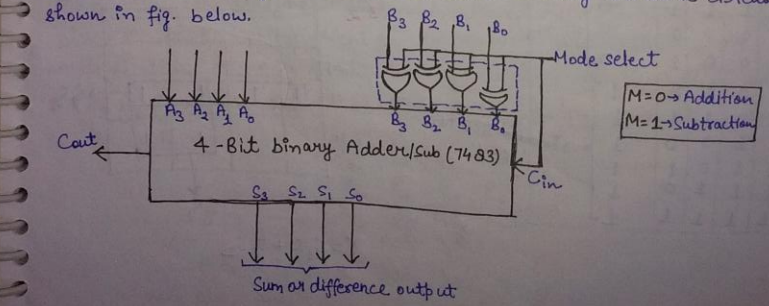
1's or 2's complement of B to A. That means we can use a binary adder to perform the binary subtraction.

⇒ Four bit Parallel subtractor using 2's complement :- A four bit parallel subtractor using a 4-bit parallel adder is shown in fig. below.

* The number to be subtracted (B) is first passed through inverters to obtain its 1's complemen. Then 1 is added to 1's complement of B, by making $C_{in}=1$. Thus we obtain the 2's complement of B.
* The 4-bit adder than adds A and 2's complement of B to produce the subtraction. $S_3S_2S_1S_0$ represent the result of binary subtraction (A-B) and carry output C_{out} represents the polarity of the result.
* If A>B then $C_{out}=0$ and the result is in true binary form but if A<B then $C_{out}=1$ and the result is in the 2's complement form.



⇒ 4-Bit binary Parallel Adder or subtractor :- The addition and subtraction of two 4-bit binary numbers can be obtained using the same circuit as shown in fig. below.



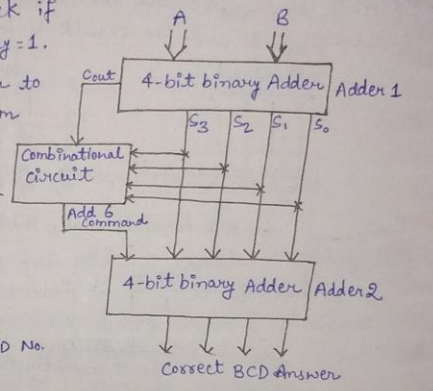
⑥ **BCD Adder**:- A BCD adder adds two BCD digits and produces a BCD digit. A BCD cannot be grater than 9. The two given BCD numbers are to be added using the rules of binary addition. If sum is less than or equal to 9 and carry = 0 then no correction is necessary. The sum is correct and in the true BCD form. But if sum is invalid BCD or carry 1 then the result is wrong and needs correction. For correction adding $(6)_{10}$ to it.

⇒ **Block Diagram of BCD adder**:- From above points stated above, we understand that the 4-bit, BCD adder should consist of the following

i) A 4-bit binary adder to add the given numbers A and B.

ii) A combinational CKT to check if sum is grater than 9 or carry = 1.

iii) One more 4-bit binary adder to add six (0110) to the incorrect sum if sum > 9 or carry = 1.



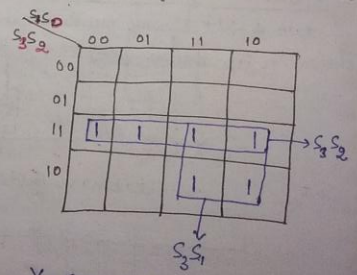
⇒ **Design of combinational Circuit**:-

Inputs				Output
S_3	S_2	S_1	S_0	Y
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	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
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	1

→ Valid BCD No. ∴ Y = 0

→ Invalid BCD No. ∴ Y = 1

Block diagram of BCD Adder



$Y = S_3S_1 + S_3S_2$

The output of combinational CKT should be 1 if C_{out} of adder-I is high. Therefore Y is ORed with C_{out} of adder-1 as shown in fig. below. And the output of combinational circuit is connected to B_1B_2 inputs of adder-2 and $B_3 = B_0 = 0$ as they are connected to ground.

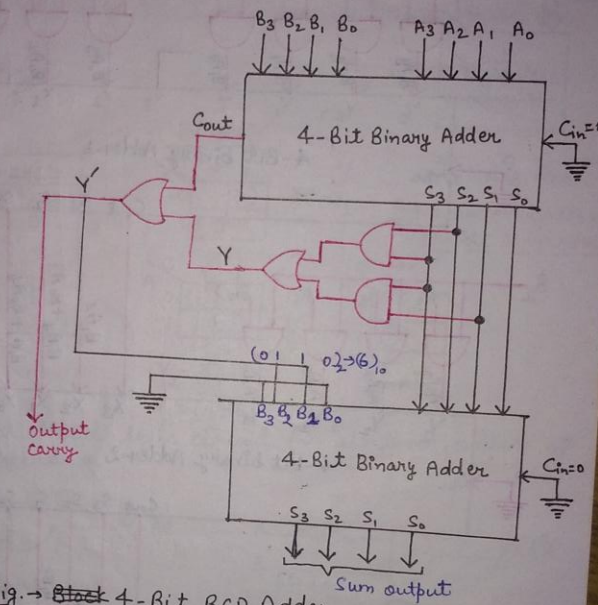
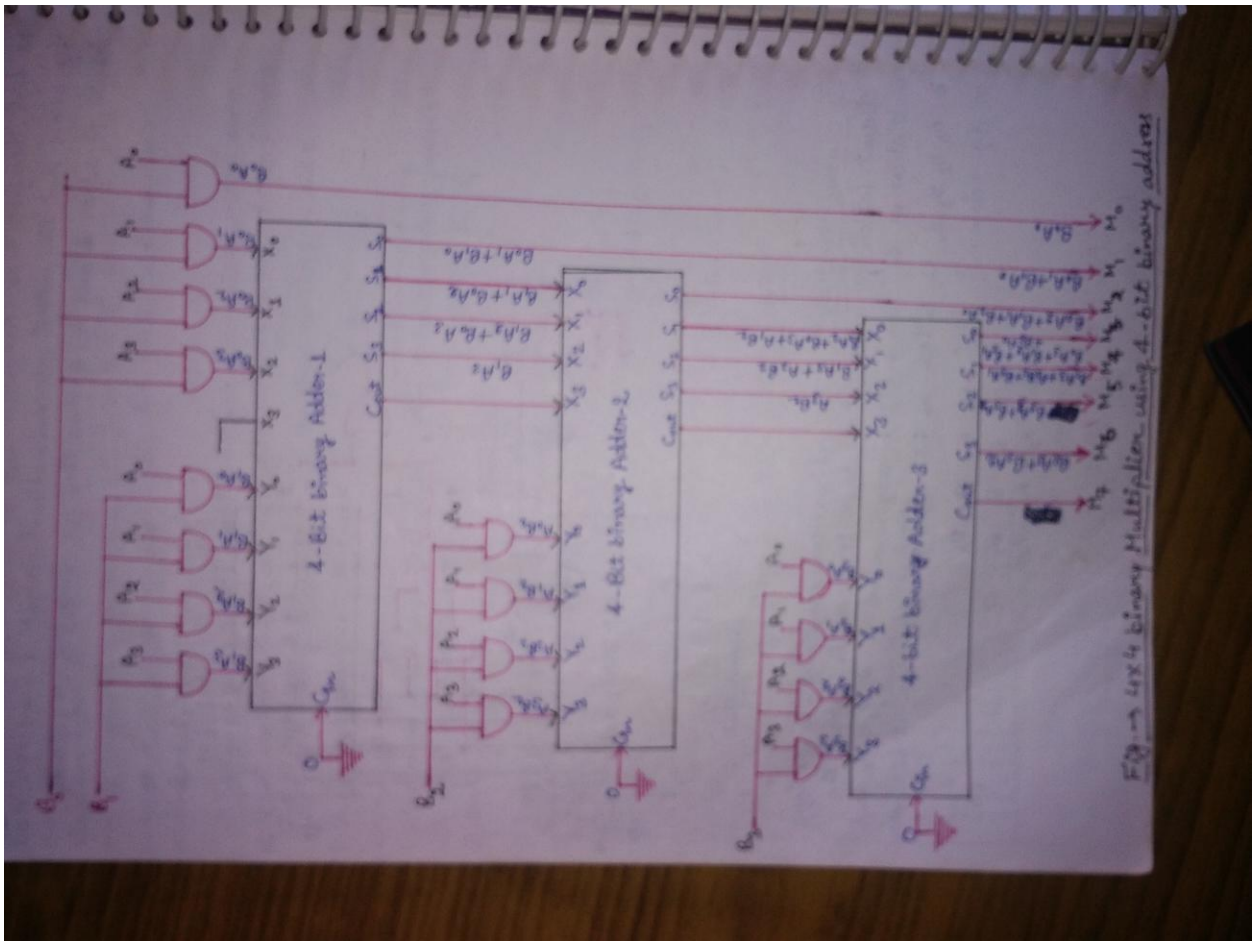


Fig.→ ~~Block~~ 4-Bit BCD Adder

⑦ Binary Multipliers:- A combinational multiplier is the logic circuit which is implemented to perform multiplication. The multiplication of two 4-bit binary numbers $A = A_3A_2A_1A_0$ and $B = B_3B_2B_1B_0$ is called as 4×4 multiplication and it is shown in fig. below.

		A_3	A_2	A_1	A_0		
	X	B_3	B_2	B_1	B_0		
		A_3B_0	A_2B_0	A_1B_0	A_0B_0		
	A_3B_1	A_2B_1	A_1B_1	A_0B_1	X		
A_3B_2	A_2B_2	A_1B_2	A_0B_2	X	X		
A_3B_3	A_2B_3	A_1B_3	A_0B_3	X	X	X	
M_7	M_6	M_5	M_4	M_3	M_2	M_1	


4-Bit binary Adder-1
4-Bit binary Adder-2
4-Bit binary Adder-3
Cout
Cin

⑧ Magnitude Comparators :-

The block diagram of an n-bit digital comparator is shown in fig. below.

Digital comparator is a combinational ckt, designed to compare the two n-bit binary words applied at its input.

The comparator has three outputs namely A>B, A=B and A<B.

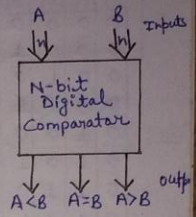


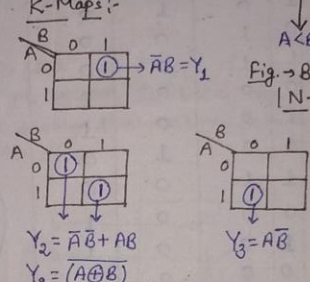
Fig. → Block diagram of N-bit comparator

1-Bit comparator :-

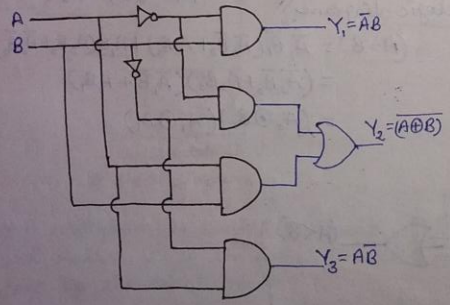
Truth table :-

Inputs		Outputs		
A	B	Y_1 A<B	Y_2 A=B	Y_3 A>B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

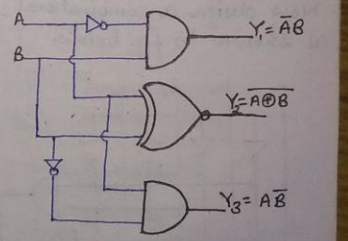
K-Maps :-



Now design a combinational logic circuit for all three outputs of the magnitude comparators.



a) Using basic gates



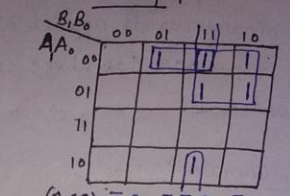
b) Using AND & EX-NOR gates

⇒ 2-Bit Comparator :-

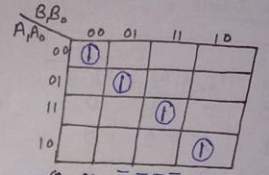
Truth Table

I/Ps				O/Ps		
A_1	A_0	B_1	B_0	A<B	A=B	A>B
0	0	0	0	0	1	0
0	0	0	1	1	0	0
0	0	1	0	1	0	0
0	0	1	1	1	0	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
0	1	1	1	1	0	0
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	0	1	0
1	0	1	1	1	0	0
1	1	0	0	0	0	1
1	1	0	1	0	0	1
1	1	1	0	0	0	1
1	1	1	1	0	1	0

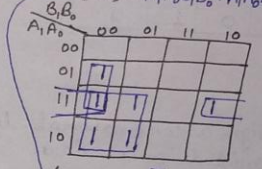
K-Maps :-



$(A<B) = \bar{A}_1B_1 + \bar{A}_1\bar{A}_0B_0 + \bar{A}_0B_1B_0$



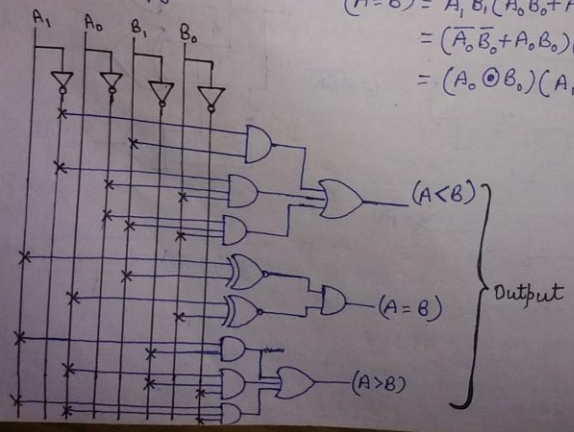
$(A=B) = \bar{A}_1\bar{A}_0\bar{B}_1\bar{B}_0 + \bar{A}_1A_0\bar{B}_1B_0 + A_1A_0B_1B_0 + A_1\bar{A}_0B_1\bar{B}_0$

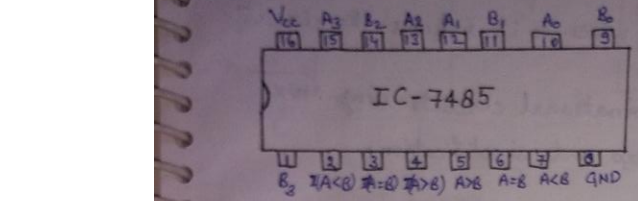


$(A>B) = A_1\bar{B}_1 + A_0\bar{B}_1\bar{B}_0 + A_1A_0\bar{B}_0$

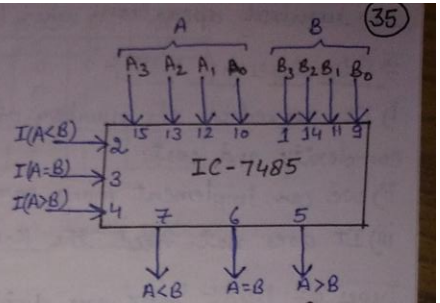
Now design a combinational logic diagram as shown in fig. below.

$$(A=B) = \bar{A}_1\bar{B}_1(\bar{A}_0\bar{B}_0 + A_0B_0) + A_1B_1(A_0B_0 + \bar{A}_0\bar{B}_0)$$
$$= (\bar{A}_0\bar{B}_0 + A_0B_0)(\bar{A}_1\bar{B}_1 + A_1B_1)$$
$$= (A_0 \odot B_0)(A_1 \odot B_1)$$



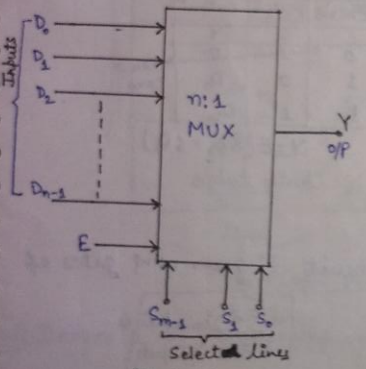


a) PIN diagram of IC-7485

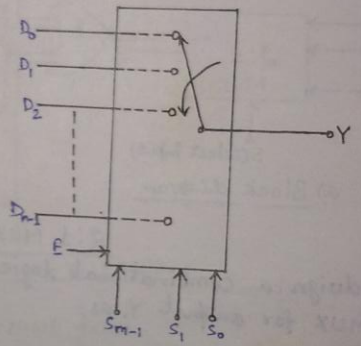


b) Logic diagram of IC-7485

(9) **Multiplexer**:- A multiplexer is a special type of combinational circuit. The block diagram of an n to 1 multiplexer has been shown in fig. below and its equivalent circuit also. As shown there are n-data inputs, one output and m select inputs with $2^m = n$.



a) Block diagram



b) Equivalent circuit

* A multiplexer is a digital circuit which selects one of the n inputs and routes it to the output. The selection of one of the n inputs is done by the select inputs.
* To select n inputs we need m select lines such that $2^m = n$, depending on the digital code applied at the select inputs, one out of n data sources is selected and transmitted to the single output Y.
* E is called as a strobe or enable input which is useful for cascading. It is generally an active low terminal, that means it will perform

the required operation when it is low.

⇒ Advantages of MUX :-

i) It reduces the number of wires. So it reduces the circuit complexity and cost.

ii) We can implement many combinational circuits using MUX.

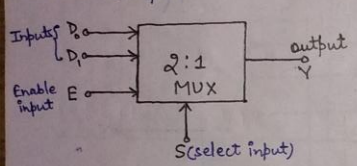
iii) It does not need the K-Map and simplification.

Types of MUX :- MUX are divided into following types

i) 2:1 MUX ii) 4:1 MUX iii) 8:1 MUX iv) 16:1 MUX

v) 32:1 MUX

i) 2:1 MUX :- Block diagram of 2:1 MUX is shown in fig below. It has two data inputs D_0 and D_1, one select input S, an enable input and one output.



a) Block diagram

Enable	Select input	Output
E	S	Y
0	X	0
1	0	D_0
1	1	D_1

Don't care

$Y = E(\bar{S}D_0 + SD_1)$

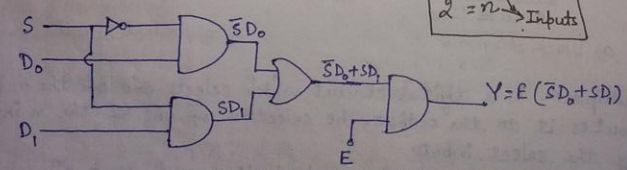
b) Truth table

2:1 MUX

Now design a combinational logic circuit diagram using gates of 2:1 MUX for output Y. as,

$2^1 = 2$ → Select lines

$2^m = n$ → Inputs



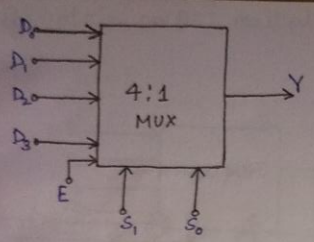
2:1 MUX using gates

ii) 4:1 MUX :- Block diagram of 4:1 MUX is shown in fig. below where have four input terminals, 2 select lines, 1 enable input and one output terminal.

$\because 2^2 = 4 \Rightarrow 2^m = n$

m = select line

n = no. of inputs.



a)→ Block diagram

Enable	Select I/Ps		O/P
E	S_1	S_0	Y
1	0	0	D_0
1	0	1	D_1
1	1	0	D_2
1	1	1	D_3

b)→ Truth table

4:1 MUX

And the combinational logic circuit of 4:1 MUX is shown in fig below using logic gate.

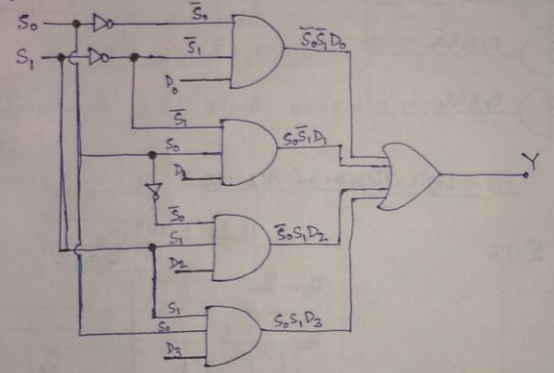
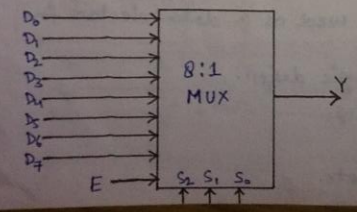


Fig.→ Logic diagram of 4:1 MUX

Q. Design a 4:1 MUX using only NAND gate.

iii) 8:1 MUX :- ~~The~~ Block diagram and truth table of 8:1 MUX is given below, which consist by three select line, 8 input and one o/p & enable bit terminals. $2^3 = 8 \Rightarrow 2^m = n$



Enable	Select line			output
E	S_2	S_1	S_0	Y
0	X	X	X	0
1	0	0	0	D_0
1	0	0	1	D_1
1	0	1	0	D_2
1	0	1	1	D_3
1	1	0	0	D_4
1	1	0	1	D_5
1	1	1	0	D_6
1	1	1	1	D_7

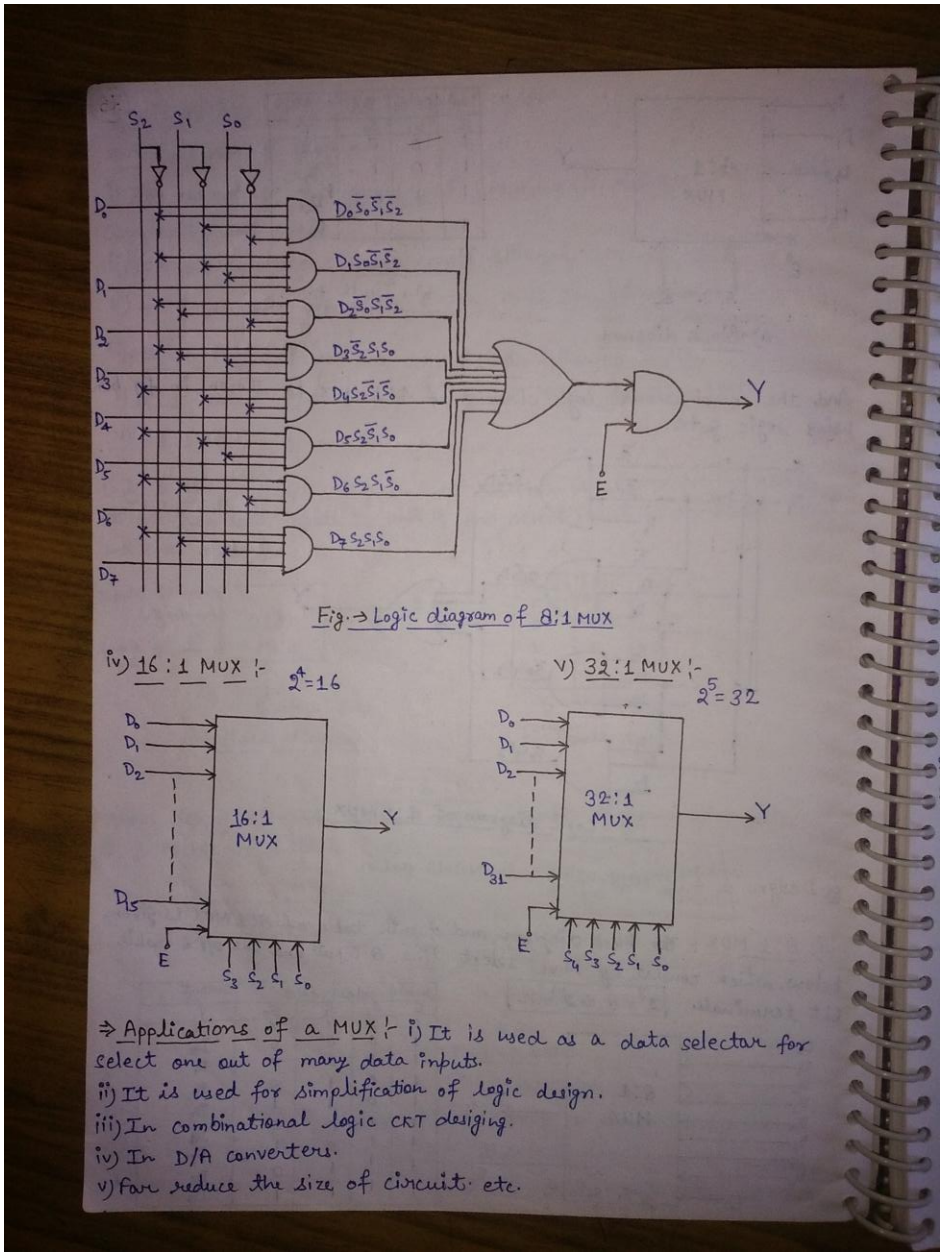


Fig. → Logic diagram of 8:1 MUX

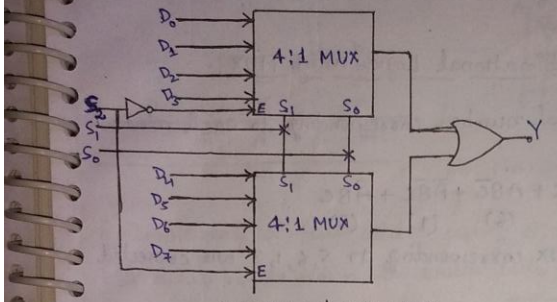
iv) 16 : 1 MUX :- $2^4 = 16$

v) 32 : 1 MUX :- $2^5 = 32$

⇒ Applications of a MUX :- i) It is used as a data selector for select one out of many data inputs.
ii) It is used for simplification of logic design.
iii) In combinational logic CKT desiging.
iv) In D/A converters.
v) For reduce the size of circuit etc.

Q. Obtain an 8:1 multiplexer using two 4:1 MUX.

Solu.



Select Input			Output	
S_2	S_1	S_0	Y	
0	0	0	D_0	4:1 MUX 1
0	0	1	D_1	
0	1	0	D_2	
0	1	1	D_3	
1	0	0	D_4	4:1 MUX 2
1	0	1	D_5	
1	1	0	D_6	
1	1	1	D_7	

Truth table

8:1 MUX using 4:1 MUX

Q. Implement a 16:1 MUX using 4:1 MUX.

Solu.

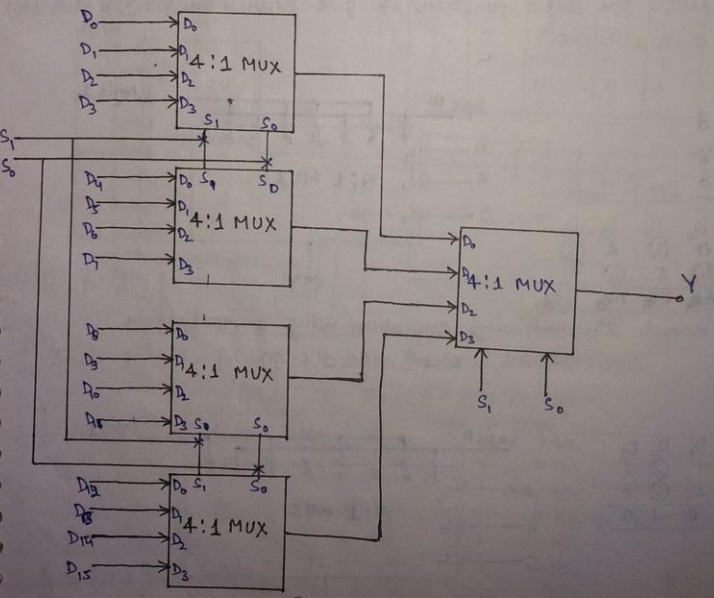


Fig.→ 16:1 MUX using 4:1 MUX

Q. Implement 16:1 MUX using 8:1 MUX.

Q. Implement 4:1 MUX using 2:1 MUX.

Q. Implement 32:1 MUX using 16:1 MUX.

⇒ Implementation of Combinational Logic using MUX :-

Step I :- Identify the decimal number corresponding to each minterm in the given expression as

$$Y = A\bar{B}C + AB\bar{C} + \bar{A}\bar{B}C + \bar{A}BC$$
$$\quad\ (5) \quad\ (6) \quad\ (1) \quad\ (3)$$

Step II :- The i/Ps line of MUX corresponding to 5, 6, 1, 3 are connected to logic 1 level.

Step III :- All other i/Ps line are connected to logic 0.

Step IV :- And the i/Ps A, B, C are connected to select inputs.

Q. Realize the logic function of give expression $Y = \Sigma m(1,3,4,6)$ using a multiplexer.

Solu.

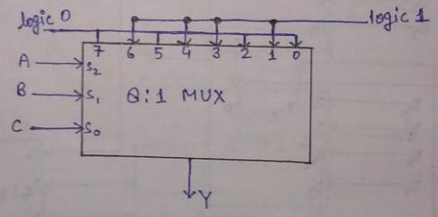
$2^3 = 8$

$m = 3$

$n = 8$

Design Table

	D_0	D_1	D_2	D_3
$\bar{A}$	0	(1)	2	(3)
A	(4)	5	(6)	7
	$A\bar{B}\bar{C}$	$\bar{A}\bar{B}C$	$AB\bar{C}$	$\bar{A}BC$

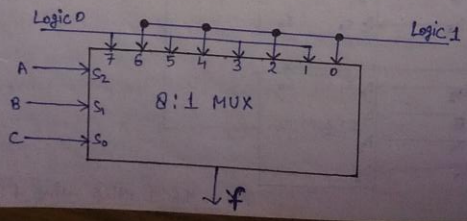


Q. Implement the following expression using a Multiplexer.

$$f(A,B,C,D) = \Sigma m(0,2,4,6)$$

Solu.

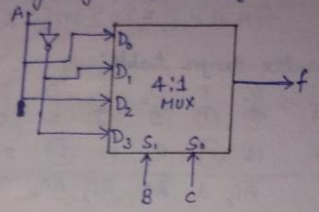
	D_0	D_1	D_2	D_3
$\bar{A}$	(0)	1	(2)	3
A	(4)	5	(6)	7
	1	0	1	0



Q. Implement the logic function $f(A,B,C) = \Sigma m(1,3,4,6)$ using a 4:1 MUX.

Solu. Draw the design table for the given fun" using 4:1 MUX;

	D_0	D_1	D_2	D_3
$\bar{A}$	0	(1)	2	(3)
A	(4)	5	(6)	7
	A	$\bar{A}$	A	$\bar{A}$



Q. Implement a full adder using 8:1 MUX.

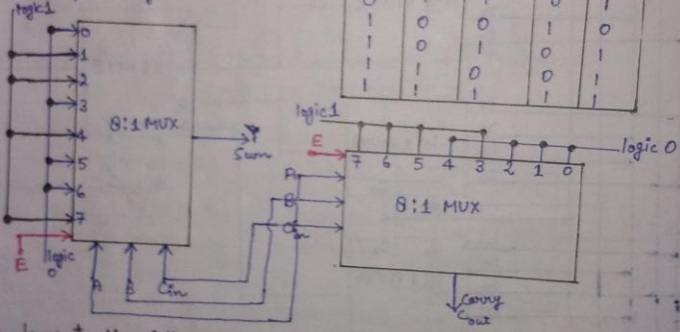
Solu. First write down the truth table ~~for~~ of full adder.

$S = \Sigma m(1,2,4,7) \rightarrow$ SOP

$C_{out} = \Sigma m(3,5,6,7) \rightarrow$ SOP

Now realize both outputs through 8:1 MUX as following.

Inputs			Outputs	
A	B	C_{in}	S	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

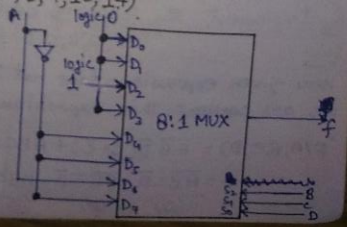


Q. Implement the following Boolean function using 8:1 MUX.

$f(A,B,C,D) = \Sigma m(2,4,5,7,10,14)$

Sdu. Here write design table first, as following.

	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
$\bar{A}$	0	1	(2)	3	(4)	(5)	6	(7)
A	8	9	(10)	11	12	13	(14)	15
	0	0	1	0	$\bar{A}$	$\bar{A}$	A	$\bar{A}$



Q. Implement the following function using 16:1 MUX.

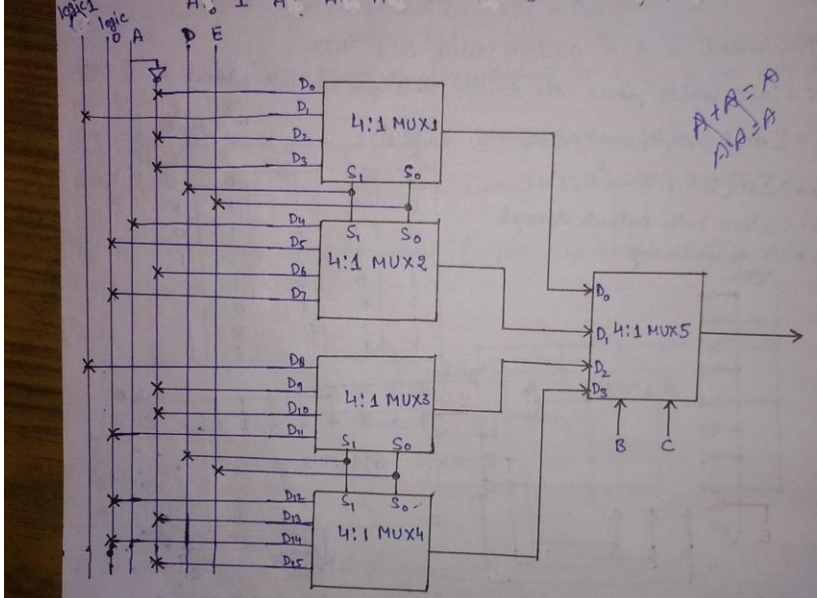
$F(A,B,C,D,E) = \Sigma m(2,4,6,7,10,14,15,16,17,25,26,30,31)$

Q. Implement the following Boolean function using all 4:1 MUX.

$f(A,B,C,D,E) = \Sigma m(0,1,2,3,6,8,9,10,13,15,17,20,24)$.

Solu. Write design table;

	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_9	D_{10}	D_{11}	D_{12}	D_{13}	D_{14}	D_{15}
$\bar{A}$	(0)	(1)	(2)	(3)	4	5	(6)	7	(8)	(9)	(10)	11	12	(13)	14	(15)
A	16	(17)	18	19	(20)	21	22	23	(24)	25	26	27	28	29	30	31
	$\bar{A}$	1	$\bar{A}$	$\bar{A}$	A	0	$\bar{A}$	0	1	$\bar{A}$	$\bar{A}$	0	0	$\bar{A}$	0	$\bar{A}$



Q. Implement the following Boolean expression using 8:1 MUX.

Solu. $F(A,B,C,D) = \bar{A}\bar{B}\bar{D} + A\bar{B}C + \bar{B}CD + \bar{A}C\bar{D}$

Here given expression have not in standard/canonical SOP form so first we are convert this expression into standard SOP form as,

$F(A,B,C,D) = \bar{A}\bar{B}\bar{D}(C+\bar{C}) + A\bar{B}C(D+\bar{D}) + \bar{B}CD(A+\bar{A}) + \bar{A}CD(B+\bar{B})$

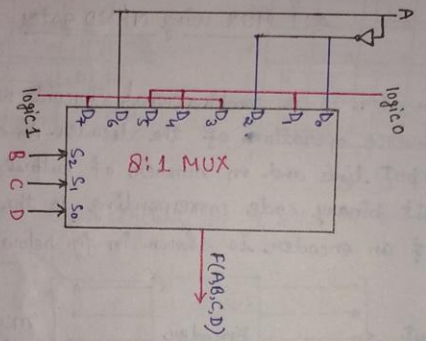
$= \bar{A}\bar{B}C\bar{D} + \bar{A}\bar{B}\bar{C}\bar{D} + A\bar{B}CD + A\bar{B}C\bar{D} + A\bar{B}CD + \bar{A}\bar{B}CD + \bar{A}B\bar{C}D + \bar{A}\bar{B}CD$

(2) (0) (15) (14) (11) (3) (7) (3)

i.e $F(A,B,C,D) = \Sigma m(0,2,3,7,11,14,15)$

Here have 16 inputs but in 8:1 MUX use only 8 inputs so first draw the design table as following

	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
$\bar{A}$	(0)	1	(2)	(3)	4	5	6	(7)
A	8	9	10	(11)	12	13	(14)	(15)
	$\bar{A}$	0	$\bar{A}$	1	0	0	A	1



4 2 0
A B C → Pos

Q. Implement the following logic function using 4:1 Multiplexer.

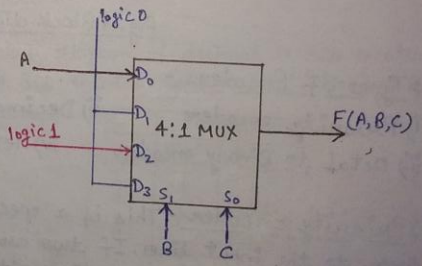
$F(A,B,C) = \Pi M(0,1,3,5,7)$

Solu: $F(A,B,C) = \Pi M(0,1,3,5,7)$

$= \Sigma m(2,4,6)$

Now draw design table,

	D_0	D_1	D_2	D_3
$\bar{A}$	0	1	(2)	3
A	(4)	5	(6)	7
	$\bar{A}$	0	1	0



Q. Implement the following function using an 8:1 MUX.

Q. Implement a full adder circuit using two 4:1 MUX.

Q. Design a 4:1 MUX using only NAND gates.

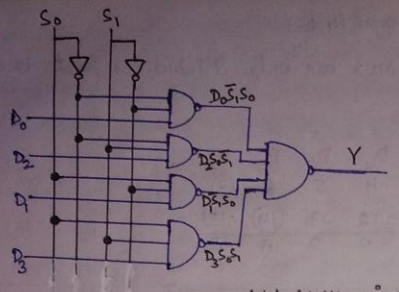
Solu: Truth table for 4:1 MUX

Select line		O/P
S_1	S_0	Y
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

$\therefore Y = A = \bar{\bar{A}}$

$Y = \overline{(\bar{S}_1\bar{S}_0D_0 + \bar{S}_1S_0D_1 + S_1\bar{S}_0D_2 + S_1S_0D_3)}$

$Y = \overline{(\bar{S}_1\bar{S}_0D_0)}\,\overline{(\bar{S}_1S_0D_1)}\,\overline{(S_1\bar{S}_0D_2)}\,\overline{(S_1S_0D_3)}$



4:1 MUX using NAND gates

(10) Encoders :- Encoder is a combinational circuit which is designed to perform the inverse operation of the decoder. An encoder has 'n' number of of input lines and m number of output lines. An encoder produces an m bit binary code corresponding to the digital input no. Block diagram of an encoder is shown in fig. below.

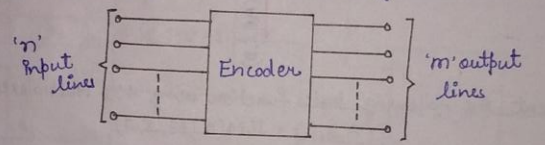
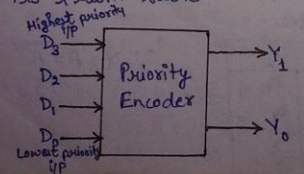


Fig. → Block diagram of an encoder

⇒ Types of Encoders :-

i) Priority encoders
ii) Decimal to BCD encoder
iii) Octal to Binary encoder
iv) Hexadecimal to binary encoder

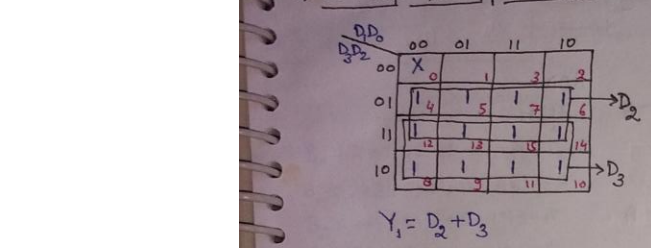
i) Priority Encoder :- This is a special type of encoder. Priorities are given to the input lines. If two or more input lines are "1" at the same time then the input line with highest priority will be considered. The block diagram of priority encoder is shown in fig. below. and also its truth table.



Highest	Inputs		Lowest	Outputs	
D_3	D_2	D_1	D_0	Y_1	Y_0
0	0	0	0	X	X
0	0	0	1	0	0
0	0	1	X	0	1
0	1	X	X	1	0
1	X	X	X	1	1

*There are four inputs D_0 to D_3 and two outputs Y_1 & Y_0. Out of the four inputs D_3 has the highest ~~pow~~priority and D_0 has the lowest priority. That means if $D_3 = 1$ then $Y_1 Y_0 = 1\ 1$ respectively of the other inputs. Similarly $D_3 = 0$ and $D_2 = 1$ then $Y_1 Y_0 = 1\ 0$ irrespective of the other inputs.

⇒ K-Map for Y_1 and Y_0 :-



Now implement the logic circuit for Y_1 & Y_0 as following,

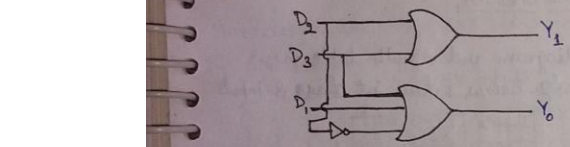


Fig.→ Priority Encoder

ii) Decimal to BCD Encoder :- The block diagram of decimal to BCD encoder is shown in fig. below. Decimal to BCD encoder has 10 inputs terminals & 4 outputs terminals.

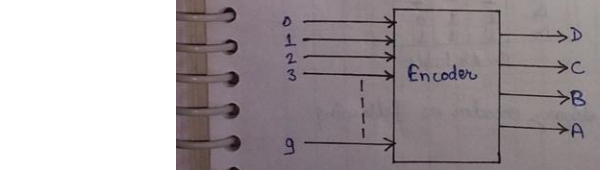


Fig.→ Block diagram of decimal to BCD Encoder

Input	Outputs			
	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

Truth table

outputs are,

$D = 8 + 9$

$C = 4 + 5 + 6 + 7$

$B = 2 + 3 + 6 + 7$

$A = 1 + 3 + 5 + 7 + 9$

Now design the combinational logic CKT diagram for decimal to BCD encoder as shown in fig. below.

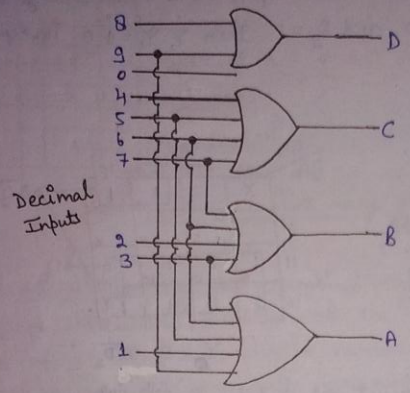
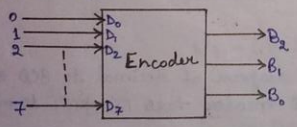


Fig. → Decimal to BCD Encoder

iii) Octal to Binary Encoder :- Block diagram and truth table for octal to Binary Encoder is shown ~~in fig.~~ below, where it has 8-inputs and three outputs terminals.



Block diagram

Inputs	Outputs		
	B_2	B_1	B_0
D_0	0	0	0
D_1	0	0	1
D_2	0	1	0
D_3	0	1	1
D_4	1	0	0
D_5	1	0	1
D_6	1	1	0
D_7	1	1	1

Truth Table

$B_0 = D_1 + D_3 + D_5 + D_7$

$B_1 = D_2 + D_3 + D_6 + D_7$

$B_2 = D_4 + D_5 + D_6 + D_7$

Now design the logic diagram for Octal to Binary encoder as following

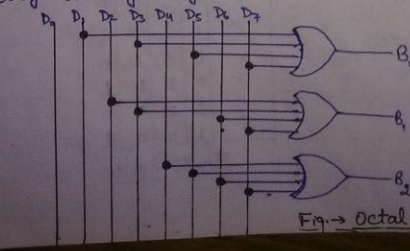
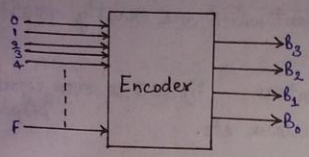


Fig. → Octal to Binary Encoder

iv) Hexadecimal to Binary Encoder :- The block diagram and truth table of Hex to Binary encoder is shown below, where it has 16-input terminals and 4-output terminals.



Block Diagram

Input	Outputs			
	B_3	B_2	B_1	B_0
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
A	1	0	1	0
B	1	0	1	1
C	1	1	0	0
D	1	1	0	1
E	1	1	1	0
F	1	1	1	1

Truth table

$B_0 = D_1 + D_3 + D_5 + D_7 + D_9 + D_B + D_D + D_F$

$B_1 = 2+3+6+7+A+B+E+F$

$B_2 = 4+5+6+7+C+D+E+F$

$B_3 = 8+9+A+B+C+D+E+F$

Now design the logic CKT diagram of HEX to Binary Encoder as shown in fig. below.

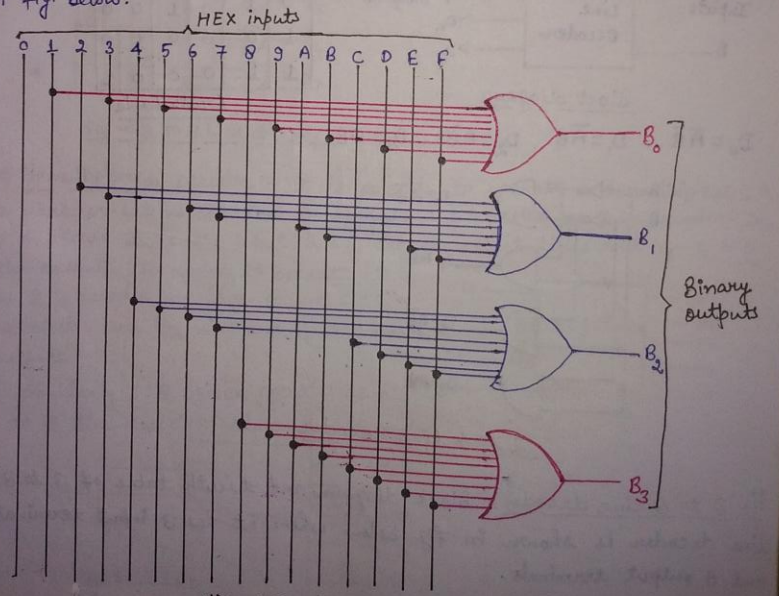
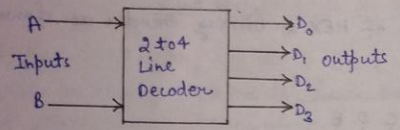


Fig. → HEX to Binary Encoder

(11) DECODER :- A decoder is a combinational circuit. Fig. below shows the block diagram of a decoder. It has n inputs and to a maximum 2^n outputs. Decoder is identical to a DEMUX without any data input. It performs operations ~~without~~ which are exactly opposite to those ~~of~~ an encoder.

⇒ Applications of Decoder :- i) code converters ii) BCD to seven segment Display iii) Relay actuators etc.

i) 2 to 4 Line Decoder :- The block diagram of a 2 to 4 line decoder is shown in fig. below. Here A and B are the two inputs whereas D_0 to D_3 are four outputs. And the table shows truth table which explain the operation of the decoder. It shows that each output is 1 for only a specific combination of inputs.

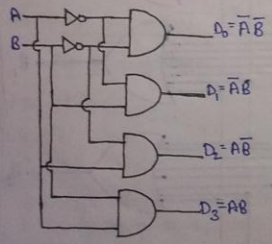


Block diagram

Inputs		Outputs			
A	B	D_0	D_1	D_2	D_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

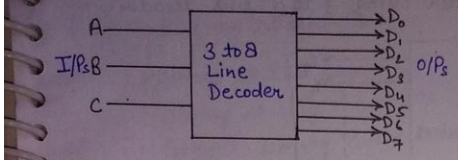
Truth Table

$D_0 = \overline{A}\,\overline{B}$, $D_1 = \overline{A}B$, $D_2 = A\overline{B}$, $D_3 = AB$



2 to 4 line Decoder

ii) 3 to 8 Line decoder :- Block diagram and truth table of 3 to 8 line decoder is shown in fig. below where it has 3 input terminals and 8 output terminals.



I/Ps			O/Ps							
A	B	C	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
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

$D_0 = \bar{A}\bar{B}\bar{C}$, $D_1 = \bar{A}\bar{B}C$, $D_2 = \bar{A}B\bar{C}$
$D_3 = \bar{A}BC$, $D_4 = A\bar{B}\bar{C}$, $D_5 = \bar{A}\bar{B}C$
$D_6 = AB\bar{C}$, $D_7 = ABC$

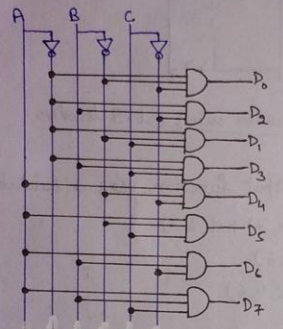
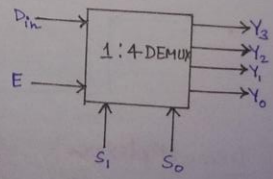
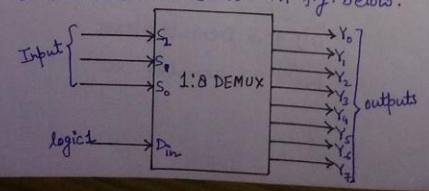


Fig.→ 3 to 8 Line decoder

⇒ Demultiplexer as Decoder :- It is possible to operate a demultiplexer as a decoder. Let us see how to operate a 1:4 deMUX as 2:4 decoder. In 1:4 DEMUX D_{in} is the input, S_1 & S_0 are the select lines and $Y_3 - Y_0$ are the outputs. In order to operate it as 2:4 decoder, we have to use S_1 & S_0 as inputs, feed D_{in} open and Y_0 to Y_3 as outputs.



Similarly 1:8 DEMUX operates as a 3 to 8 line decoder shown in fig. below.



Q. Design a 4 to 16 line decoder using 3 to 8 line decoder.

Solu.

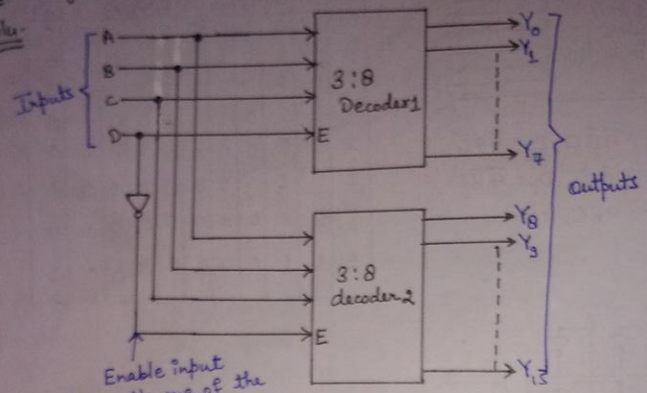


Fig. → 4 to 16 decoder using 3 to 8 decoder

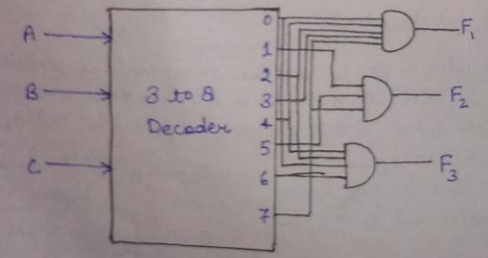
Q. Implement the following multiple output function using a suitable ~~detect~~ decoder.

$f_1(A,B,C) = \Sigma m(0,4,7) + d(2,3)$

$f_2(A,B,C) = \Sigma m(1,5,6)$

$f_3(A,B,C) = \Sigma m(0,2,4,6)$

Solu.



(12) Demultiplexers :-

i) 1:4 Demultiplexer

ii) 1:16 Demultiplexer

iii) 1:8 Demultiplexer

iv) 1:32 Demultiplexer

(13) Code Conversion:-

i) Binary to BCD converter
ii) Binary to Gray Converter
iii) Gray to Binary converter
iv) BCD to Excess-3 Converter
v) Ex-3 to BCD converter
vi) BCD to Gray code converter

Flip flops

① **Sequential Circuits** :- In the sequential circuit, the timing parameter comes into picture. The output of a sequential circuit depends on the present time inputs, the previous output and the sequence in which the inputs are applied. In order to provide a previous input or output a memory element is required to be used. Thus a sequential circuit needs a memory element. Fig. below shows the block diagram of a sequential circuit which includes the memory element in the feedback path.

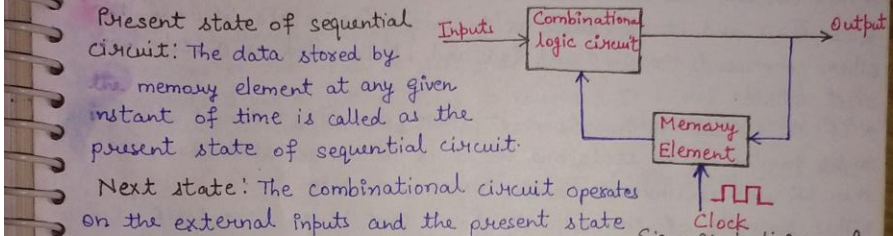


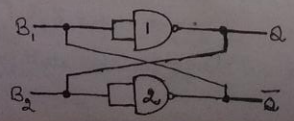
Fig → Block diagram of Sequential circuit

Present state of sequential circuit: The data stored by the memory element at any given instant of time is called as the present state of sequential circuit.

Next state: The combinational circuit operates on the external inputs and the present state to produce new outputs. Some of these new outputs are stored in the memory element and called as the next state of the sequential circuit.

The most important part of the sequential circuit seems to be the memory element. The memory element is known as flip flop (FF). It is the basic memory element.

② **1-Bit Memory Cell (Basic Bistable Element)** :- Flip flop is also known as the basic digital memory circuit. It has two stable states namely logic 1 state and logic 0 state. We can design it either using NOR gates or NAND gates. A flip flop can be designed by using the fundamental circuit shown in fig. below. Here NAND gates 1 and 2 are basically acting as inverters. Hence this circuit is called as a cross coupled inverter.



D Flip Flop → Data Flip Flop
RS F/F → RESET & SET F/F
JK F/F → Jack Kilby F/F

In above fig. the output of gate-1 is connected to the input of gate-2 and output of gate-2 is connected to input of gate-1.

Operation :- Assume that output of gate-1 i.e $Q = 1$ thus $B_2 = 1$.

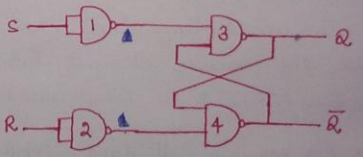
* As $B_2 = 1$, output of gate-2 i.e $\bar{Q} = 0$. This makes $B_1 = 0$, thus $Q = 1$.
* Similar process for $Q = 0$ and $\bar{Q} = 1$.

Conclusion :- We conclude that the outputs of the circuit (Q and $\bar{Q}$) will always be complementary. That means if $Q = 0$ then $\bar{Q} = 1$ and vice-versa. They will Never be equal.

* This circuit has two stable states. One of them corresponds to $Q = 1$, $\bar{Q} = 0$ and it is called as "1-state" or "set state". Whereas the other corresponds to $Q = 0$ and $\bar{Q} = 1$ and it is called as 0 state or reset state.
* If the circuit is in the reset state ($Q = 0$, $\bar{Q} = 1$) then it will continue to be in the reset state and if it is in the set state ($Q = 1$, $\bar{Q} = 0$) then it will continue to remain in the set state.
* This property of the circuit shows that it can store 1 bit of digital information. Therefore it is called as a 1-bit memory cell.

(3) Latch :- The cross coupled inverter of above fig. is capable of locking or latching the information. Hence this circuit is also called as a latch.

* The disadvantage of this circuit is that we can not enter the desired digital data into it.
* This disadvantage can be overcome by modifying the circuit as shown in fig. below. This modification will allow us to enter the digital data into the circuit.



Operation :- Case 1 :- $S = R = 0$

* The outputs of gates 1 & 2 will become 1.
* Let $Q = 0$ and $\bar{Q} = 1$ initially. Hence both the inputs to gate 3 are 1 and the inputs to gate-4 are (01).
* Hence gate-3 output i.e $Q = 0$ and gate-4 output $\bar{Q} = 1$.
* Thus with $S = R = 0$, there is no change in the state of outputs.

Case 2: S = 1 , R = 0

* Since S=1 and R=0, one of the inputs of gate-3 will be 0. This will force Q output to 1.
* Hence both the inputs to gate 4 will be 1. This forces $\bar{Q}$ to 0.
* Hence for S=1, R=0, the outputs are Q=1 and $\bar{Q}$=0. This is set state.

Case 3: S = 0 , R = 1

* If S=0, R=1 then one of the inputs to gate-4 will be 0. This will force the $\bar{Q}$ output to 1.
* Hence both the inputs to gate-3 will be 1. This forces Q to 0.
* Thus for S=0, R=1 then the outputs are Q=0 , $\bar{Q}$=1. This is reset state.

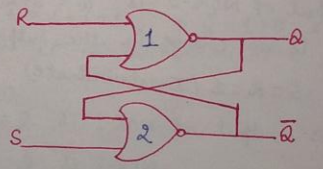
Case 4: S = R = 1

* If S=R=1 then outputs of gates 1 and 2 will be zero.
* Hence one of the inputs to gates 3 and 4 will be zero.
* So outputs Q and $\bar{Q}$ both will try to become 1. It is not allowed as Q and $\bar{Q}$ should be complementry. Hence S=R=1 condition is prohibited.

(4) Types of Flip-Flops :- There are many types of flip flops which are given below.

i) RS Flip flop ii) D Flip flop iii) JK Flip flop iv) T flip flop.

i) RS Flip flop :- In fig. below shows the RS flip flop using NOR gates. Latch is a sequential logic circuit which checks all the input continiously as soon as change its o/p. The output of the NOR gate is 0 when the input is 1 & the o/p of NOR gate is 1 when the i/p is 0. In fig. the output of NOR gate-1 connected to the i/p terminal of gate-2 and the o/p of NOR gate-2 connected to the i/p terminal of gate-1.



RS Flip flop using NOR gates

Thus the general expression for the RS flip flop is,

NOR-1 output $Q = \overline{R+\bar{Q}}$

NOR-2 output $\bar{Q} = \overline{S+Q}$

Case 1 :- R = 0 & S = 1 (Set Condition)

* Initially we assume Q = 0 ~~and $\bar{Q}$ = 1~~ thus, $\bar{Q} = \overline{S+Q} = \overline{1+0} = \bar{1} = 0$

So, the output of gate-1, $Q = \overline{R+\bar{Q}} = \overline{0+0} = \bar{0} = 1$

And, the output of gate-2, $\bar{Q} = \overline{S+Q} = \overline{1+1} = \bar{1} = 0$

Case 2 :- R = 1 & S = 0 (Reset Condition)

* Initially we assume Q = 0 then

→ inputs of NOR-2 is 0 and 0 thus

output of NOR-2, $\bar{Q} = \overline{S+Q} = \overline{0+0} = \bar{0} = 1$

→ inputs of NOR-1 is 1 and 1 thus

output of NOR-1, $Q = \overline{R+\bar{Q}} = \overline{1+1} = \bar{1} = 0$

$Q = \overline{R+\bar{Q}} = \bar{R}\cdot\bar{\bar{Q}} = 0\cdot Q$

$\Rightarrow \boxed{Q = 0}$

$\bar{Q} = \overline{S+Q} = \bar{S}\cdot\bar{Q} = 1\cdot\bar{Q}$

$\boxed{\bar{Q} = \bar{Q} = 1}$

* Initially if we assume Q = 1 then

→ inputs of NOR-1 is 1 and 0 thus

output of NOR-1, $\bar{Q} = \overline{R+\bar{Q}} = \overline{1+0} = \bar{1} = 0$

output of NOR-2 $\bar{Q} = \overline{S+Q} = \overline{0+0} = \bar{0} = 1$

Case 3 :- S = R = 0 (No change)

* Since output of NOR-1 is, $Q = \overline{R+\bar{Q}} = \bar{R}\cdot\bar{\bar{Q}} = \bar{0}\cdot Q = 1\cdot Q = Q$

output of NOR-2 is, $\bar{Q} = \overline{S+Q} = \bar{S}\cdot\bar{Q} = \bar{0}\cdot\bar{Q} = 1\cdot\bar{Q} = \bar{Q}$

Thus there is no change in the output of NOR gate 1 & 2.

Case-4 :- S = R = 1 (Indeterminate)

* Since output of NOR-1 is, $Q = \overline{R+\bar{Q}} = \bar{R}\cdot\bar{\bar{Q}} = 0\cdot Q = 0$

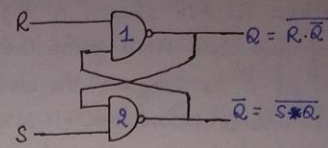
NOR-2 is $\bar{Q} = \overline{S+Q} = \bar{S}\cdot\bar{Q} = 0\cdot\bar{Q} = 0$

* Here both outputs Q and $\bar{Q}$ will force to 0 hence this is an ~~indet~~ indeterminated state which should be avoid by latch because Q and $\bar{Q}$ should be complements of each other.

Truth table :-

Inputs		Outputs		Comment
R	S	Q	$\bar{Q}$	
0	1	1	0	SET
1	0	0	1	RESET
0	0	0	0	NC
1	1	X	X	Indeterminate

b) RS Flip flop using NAND gate :- In fig. below shows the RS FF using NAND gates.



Case 1 :- R = 0 , S = 1 (SET)

* Since $Q = \overline{R \cdot \bar{Q}}$ (Gate-1 o/p)

$\bar{Q} = \overline{S \cdot Q}$ (Gate-2 o/p)

thus, in this case,

$Q = \overline{0 \cdot \bar{Q}} = \bar{0} + \bar{\bar{Q}} = 1 + Q$ ⇒ if Q = 0, $Q = 1+0 = 1$ ⇒ if Q = 1, $Q = 1+1 = 1$

and $\bar{Q} = \overline{S \cdot Q} = \bar{S} + \bar{Q} = \bar{1} + \bar{Q} = 0 + \bar{Q}$ ⇒ $\bar{Q} = 0+0 = 0$ ⇒ $\bar{Q} = 0+0 = 0$

Case 2 :- R = 1 , S = 0 (RESET)

* ∵ $Q = \overline{R \cdot \bar{Q}}$ & $\bar{Q} = \overline{S \cdot Q}$

In this case,

$Q = \bar{R} + \bar{\bar{Q}} = \bar{1} + Q = 0 + Q$ ⇒ if Q = 0, $Q = 0+0 = 0$ ⇒ if Q = 1, $Q = 0+1 = 1 \to 0$

and $\bar{Q} = \bar{S} + \bar{Q} = \bar{0} + \bar{Q} = 1 + \bar{Q}$ ⇒ $\bar{Q} = 1+1 = 1$ ⇒ $\bar{Q} = 1+0 = 1$

Case-3 :- R = S = 0 (Indeterminative)

* ∵ $Q = \overline{R \cdot \bar{Q}}$ & $\bar{Q} = \overline{S \cdot Q}$

in this case,

$Q = \bar{R} + \bar{\bar{Q}} = \bar{0} + Q = 1 + Q$ ⇒ if Q = 0, $Q = 1+0 = 1$ ⇒ if Q = 1, $Q = 1+1 = 1$

$\bar{Q} = \bar{S} + \bar{Q} = \bar{0} + \bar{Q} = 1 + \bar{Q}$ ⇒ $\bar{Q} = 1+1 = 1$ ⇒ $\bar{Q} = 1+0 = 1$

Here both O/Ps are same thus it is indeterminative state

Case 4 :- S = R = 1 (No change)

* ∵ $Q = \overline{R \cdot \bar{Q}}$ & $\bar{Q} = \overline{S \cdot Q}$

in this case,

$Q = \bar{R} + \bar{\bar{Q}} = \bar{1} + Q = 0 + Q$ ⇒ if Q = 0, $Q = 0+0 = 0$ ⇒ if Q = 1, $Q = 0+1 = 1$

$\bar{Q} = \bar{S} + \bar{Q} = \bar{1} + \bar{Q} = 0 + \bar{Q}$ ⇒ $\bar{Q} = 0+1 = 1$ ⇒ $\bar{Q} = 0+0 = 0$

Here there is no change in both outputs of NAND-1 & 2 gates.

Truth table :-

Inputs		outputs		Condition
R	S	Q	$\bar{Q}$	
0	0	X	X	Indeterminated
0	1	1	0	SET
1	0	0	1	RESET
1	1	NC	NC	No change

Clocked RS Flip Flop :- Gated RS F/F :- Level Triggered RS F/F :-

The clocked RS F/F shown in fig. below consist of a basic NOR flip flop and two AND gates. The outputs of the two AND gates remain at 0 as long as the clock pulse is 0, regardless of the S and R input values. When the clock pulse goes to 1, information from the S and R inputs allowed to reach the basic flip flop. The set state is reached with S=1, R=0 and CP=1. To change to the reset state, the inputs must be S=0, R=1 and CP=1. With both S=1 and R=1, the occurance of a clock pulse causes both outputs to go to 0. When the pulse is removed, the state of the flip flop is ~~indt~~ indeterminate i.e either state may result, depending on whether the set or the reset input of the basic flip flop remains 1 longer before the transition to 0 at the end of the pulse.

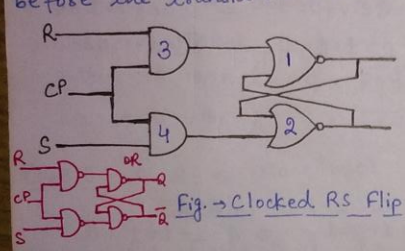
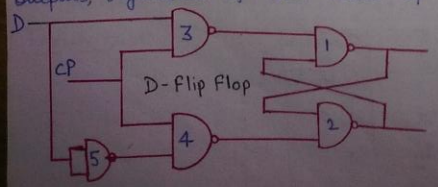


Fig. → Clocked RS Flip flop

CP	Inputs R	S	outputs Q	$\bar{Q}$	
1	0	0	No	NC	No change
1	0	1	1	0	SET
1	1	0	0	1	RESET
1	1	1	X	X	Indeterminate
0	X	X	Q	$\bar{Q}$	No change

Truth Table

ii) D Flip flop :- The D flip flop is shown in fig. below which is a modification of the clocked RS flip flop. NAND gates 1 and 2 form a basic F/F and gate 3 and 4 modify it into a clocked RS F/F. The D input goes directly to the S input and its complement through gate. 5 is applied to the R input. As long as the clock pulse input is at 0, gates 3 and 4 have a 1 in their outputs, regardless of the value of the other inputs.



Input		Output	
CP	D	Q_n	Q_{n+1}
1	0	1	0
1	1	0	1

Truth table

In D-flip flop there are only two input conditions

Case 1 :- When D=0 then S=0 and R=1

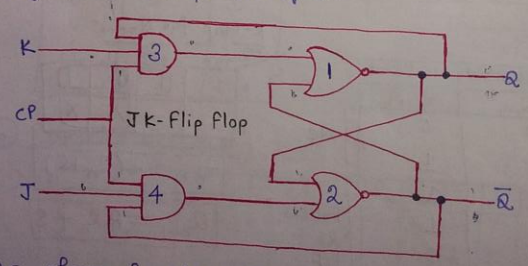
Case 2 :- When D=1 then S=1 and R=0.

During the occurance of clock pulses if D=1 then output Q is SET and if D=0 then output Q is RESET. Here Q_n is the present state and Q_{n+1} next state.

The D FF receives the designation from its ability to transfer data into a FF. It is basically an RS FF with an inverter in the R input. The added inverter reduces the number of inputs from two to one. This type of FF is sometimes called a gated D-latch.

iii) JK Flip Flop :- A JK FF is a refinement of the RS FF in that the ~~inter~~ indeterminate state of the RS type is defined in the JK type. Inputs J and K behave like inputs S and R to set and clear the FF (note that in a JK FF, the letter J is for set and letter K is for reset) When inputs are applied to both J and K simultaneously, the flip flop switches to its complement state that is if Q=1, it switches to Q=0 and vice versa.

A clocked JK FF is shown in fig. below. Output Q is ANDed with K and CP inputs so that the FF is cleared during a CP only if Q was previously 1. Similarly, output $\bar{Q}$ is ANDed with J and CP inputs so that the FF is set with a clock pulse only if $\bar{Q}$ was previously 1.



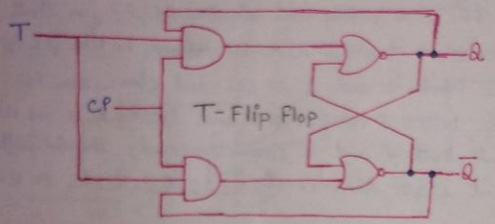
Q	J	K	Q_{n+1}
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Table

As shown in above fig. the JK FF behaves like an RS FF except when both J and K are equal to 1. When both J and K are 1, the clock pulse is transmitted through one AND gate only - the one whose input is connected to the FF

output which is presently equal to 1. Thus if Q = 1, the output of the upper AND gate becomes 1 upon application of a clock pulse and the flip flop is cleared (reset). If $\bar{Q} = 1$, the output of the lower AND gate becomes a 1 and the flip flop is set. In either cases, the output state of the flip flop is complemen

iv) T Flip flop :- It is known as Toggle Flip Flop. T flip flop is a modified/modification of JK Flip flop. It is obtained from JK Flip flop by connecting both inputs J & K together as shown in fig. below.



Present State	FF I/P	Next State
Q_n	T	Q_{n+1}
0	0	0
0	1	1
1	0	1
1	1	0

In this type of Flip flop, when T is zero both AND gates are disabled hence there is no change in the ~~present~~ previous output. But when T is 1 (J=K=1) then the output is Toggles.

5) Characteristic Equation for Flip flops :-

i) For RS Flip flop :-

Input		Present state	Next State	
R	S	Q_n	Q_{n+1}	
0	0	0	0	NC
0	0	1	1	
0	1	0	1	Follow RS F/F
0	1	1	1	
1	0	0	0	
1	0	1	0	
1	1	0	X	Indeterminate
1	1	1	X	

K-Map for Q_{n+1} :-

R \ SQ_n	00	01	11	10
0	1	1	1	1
1			X	X

$Q_{n+1} = \bar{R}(Q_n + S)$

$Q_{n+1} = \bar{R}Q_n + S$

ii) For D Flip flop :-

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

K-Map :-

Q_n / D	0	1
0		
1		

$Q_{n+1} = D$

If D = 0 then O/P $Q_{n+1} = 0$
D = 1 then O/P $Q_{n+1} = 1$

iii) For JK Flip flop :-

Input		Present	Next
J	K	Q_n	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

K-Map :-

J / KQ_n	00	01	11	10
0		1		
1	1	1		1

$Q_{n+1} = J\bar{Q}_n + \bar{K}Q_n$

iv) For T Flip Flop :-

Input	Present	Next
T	Q_n	Q_{n+1}
0	0	0
0	1	1
1	0	1
1	1	0

K-Map :-

T / Q_n	0	1
0		1
1	1	

$Q_{n+1} = T\bar{Q}_n + \bar{T}Q_n$

⑥ Flip Flop Excitation table :-

i)

Q_n	Q_{n+1}	R	S
0	0	X	0
0	1	0	1
1	0	1	0
1	1	0	X

RS F/F

ii)

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

JK F/F

iii)

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

D F/F

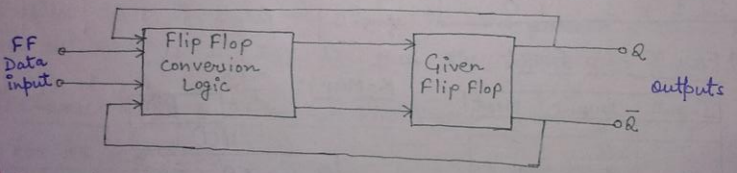
iv)

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

T F/F

⑦ Flip Flop Conversion :- The conversion from one type of flip flop to the other, needs a systematic approach using the excitation tables and K-Map simplifications. Fig. below shows a generalized model for conversion from one FF to the other. As shown in fig. the required FF is actually a combination of given FF and a combinational logic circuit using gates.

The conversion logic is designed by combining the excitation tables of both the FF. The truth table of the conversion logic has data inputs and Q and $\bar{Q}$ outputs of the given FF as inputs and the inputs of the given FF are the outputs of the truth tables. Then we draw the K-Map for each output and obtain the simplified expression. The conversion logic is then implemented using gates.



i) Conversion from SR F/F to D F/F :- The truth table is prepared by combining the excitation tables of D F/F and SR F/F.

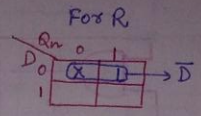
Inputs			Outputs	
D	Present state Q_n	Next Q_{n+1}	S	R
0	0	0	0	X
1	0	1	1	0
0	1	0	0	1
1	1	1	X	0

← Excitation table of D F/F →
← Excitation table of SR F/F →

K-Map :- For S

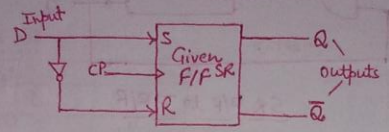
$S = D$

For R



$R = \bar{D}$

Logic Diagram :-



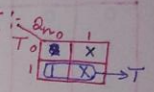
SR F/F to D F/F

ii) Conversion of JK F/F to T F/F :-

Inputs			outputs	
T	Present state Q_n	Next Q_{n+1}	J	K
0	0	0	0	X
1	0	1	1	X
1	1	0	X	1
0	1	1	X	0

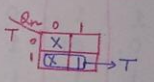
K-Map :-

For J :-



$J = T$

For K :-



$K = T$

Logic diagram :-

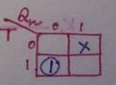
JK F/F to T F/F

iii) Conversion of SR F/F to T F/F :-

Inputs			outputs	
T	Present state Q_n	Next state Q_{n+1}	S	R
0	0	0	0	X
1	0	1	1	0
1	1	0	0	1
0	1	1	X	0

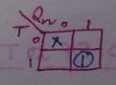
K-Map :-

For S :-



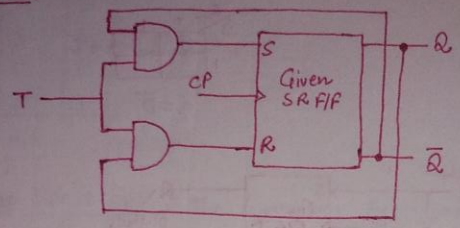
$S = T\bar{Q}_n$

For R :-



$R = TQ_n$

Logic diagram :-



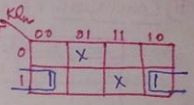
SR F/F to T F/F

iv) S-R F/F to JK F/F :-

Inputs				Outputs	
J	K	Q_n	Q_{n+1}	S	R
0	0	0	0	0	X
0	1	0	0	0	X
1	0	0	1	1	0
1	1	0	1	1	0
0	1	1	0	0	1
1	1	1	0	0	1
0	0	1	1	X	0
1	1	1	1	X	0

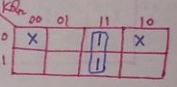
K-Map :-

For S :-



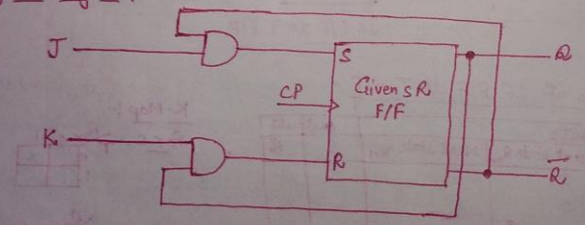
$S = J\bar{Q}_n$

For R :-



$R = KQ_n$

Logic diagram :-

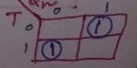


v) D F/F to T F/F :-

Inputs			Output
T	Q_n	Q_{n+1}	D
0	0	0	0
1	0	1	1
1	1	0	0
0	1	1	1

K-Map :-

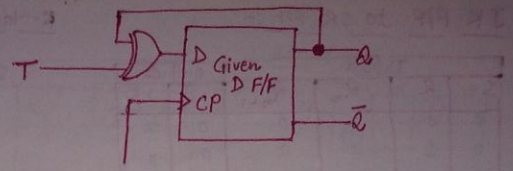
For D :-



$D = T\bar{Q}_n + \bar{T}Q_n$

$D = (T \oplus Q_n)$

Logic diagram :-



vi) T F/F to D F/F :-

Inputs			Output
D	Q_n	Q_{n+1}	T
0	0	0	0
1	0	1	1
0	1	0	1
1	1	1	0

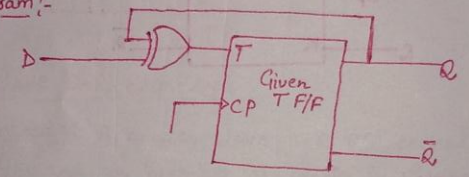
K-Map :-

D \ Q_n	0	1
0	1	
1		1

$T = D\bar{Q}_n + \bar{D}Q_n$

$T = D \oplus Q_n$

Logic diagram :-



vii) JK F/F to D F/F :-

Inputs			Outputs	
D	Q_n	Q_{n+1}	J	K
0	0	0	0	X
1	0	1	1	X
0	1	0	X	1
1	1	1	X	0

K-Map :-

For J

D \ Q_n	0	1
0		X
1	1	X

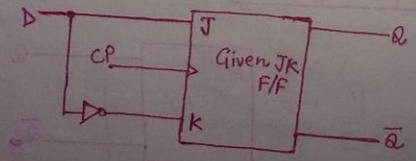
$J = D$

For K

D \ Q_n	0	1
0	X	1
1	X	

$K = \bar{D}$

Logic diagram :-



viii) JK F/F to SR F/F :-

Inputs				outputs	
S	R	Q_n	Q_{n+1}	J	K
0	0	0	0	0	X
0	1	0	0	0	X
1	0	0	1	1	X
1	0	0	1	1	X
0	1	1	0	X	1
0	1	1	0	X	1
0	0	1	1	X	0
1	0	1	1	X	0

K-Map :- For J :-

S \ RQ_n	00	01	11	10
0		X	X	
1	1	X	X	X

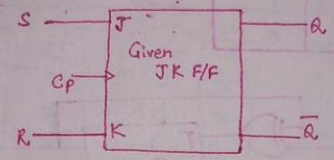
$J = S$

For K :-

S \ RQ_n	00	01	11	10
0	X	X	1	X
1	X	X	X	X

$K = R$

Logic diagram :-



ix) D F/F to SR F/F :-

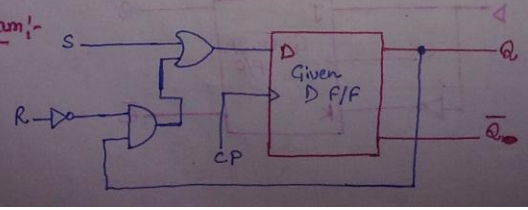
Inputs				output
S	R	Q_n	Q_{n+1}	D
0	0	0	0	0
0	1	0	0	0
1	0	0	1	1
1	0	0	1	1
0	1	1	0	0
0	1	1	0	0
0	0	1	1	1
1	0	1	1	1

K-Map :-

S \ RQ_n	00	01	11	10
0		1		
1	1	1	X	X

$D = S + \bar{R}Q_n$

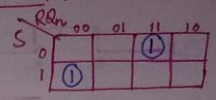
Logic diagram :-



x) T F/F to SR F/F :-

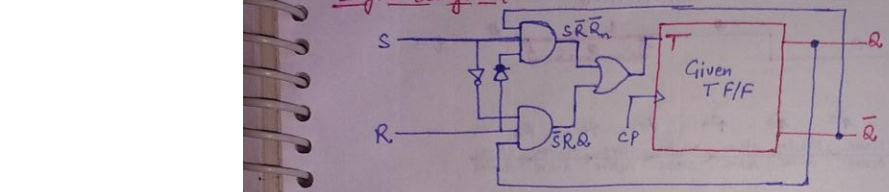
Inputs				Output
S	R	Q_n	Q_{n+1}	T
0	X	0	0	0
1	0	0	1	1
0	1	1	0	1
X	0	1	1	0

K-Map :-



$T = S\bar{R}\bar{Q}_n + \bar{S}RQ_n$

Logic diagram :-



(8) Master Slave S-R Flip Flop :-

**** Basic Concepts :-** A master slave S-R F/F consists of two level triggered F/Fs and an inverter as shown in fig. below. The first F/F is called as master. It receives the S-R inputs directly. The clock signal is applied directly to the master. Outputs (Q & $\bar{Q}$) of the master are applied to the S-R inputs of the slave. The clock is inverted and applied to the slave F/F. This will force the slave f/f to respond to the low level of the clock.

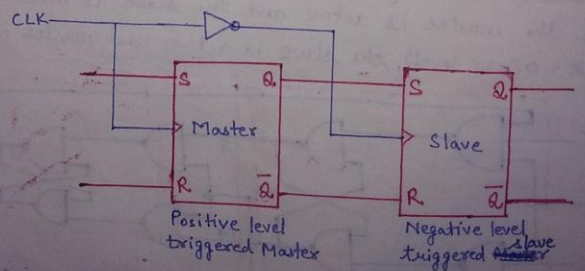


Fig.- Master slave S-R F/F

Waveform :-

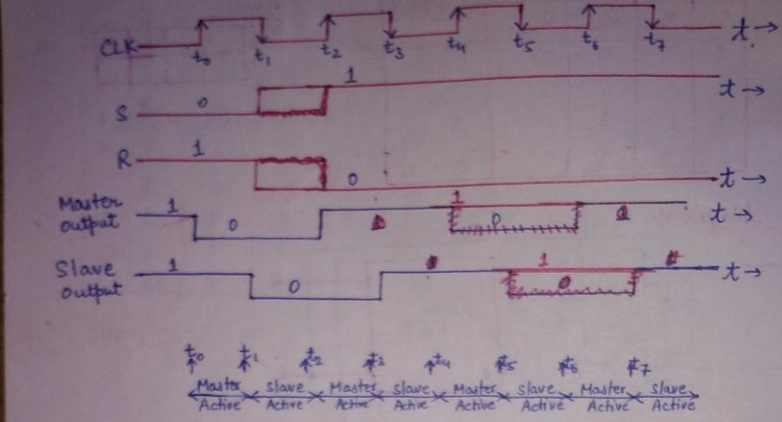
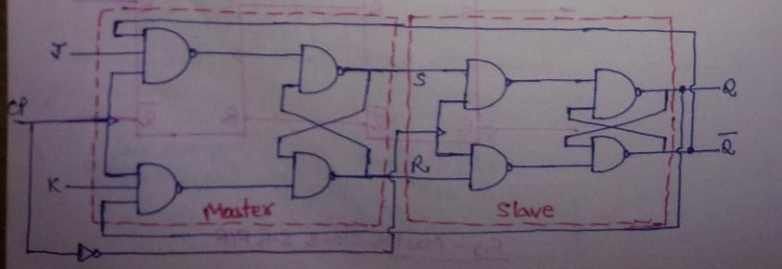


Fig. → Waveform of Master-slave SR F/F

⑨ Master Slave J K F/F :-

✱→ **Basic Concept** :- Fig. below shows the Master slave JK F/F. It is a combination of a clocked JK latch and clocked S-R latch. The clocked JK latch acts like as the master and clocked SR latch acts as the slave. Master is positive level triggered. ~~and~~ But due to the presence of the inverter in the clock ~~time~~ line, the slave will be respond to the negative level. Hence when the clock = 1 (+ve level) then the master is active and the slave is inactive. where as when clock = 0 (-ve level), the slave is active and master is inactive.



(10) **Classification of Sequential circuits**:- A sequential circuit is classified into two categories as follows,

* Synchronous sequential circuits
* Asynchronous sequential circuits.

Note 1→ In Synchronous sequential CKT, the contents of memory elements can be changed only at the rising or falling edge of clock signal.

2→ In asynchronous sequential CKT, the contents of memory elements can be changed at any instant of time.

S.N	Synchronous Sequential CKT	Asychronous Sequential CKT
1.	These CKTs are easy to design.	These CKTs are difficult to design.
2.	A clocked F/F acts as a memory element	An unclocked F/F or time delay element is used as memory element.
3.	Slower, because the delays correspond to those of the memory element.	Faster as the clock is not present.
4.	The status of memory element is affected only at the active edge of clk if the i/p is changed.	The status of memory element will change any time as soon as the i/p is changed.

(11) **Analysis of Clocked Sequential Circuits**:- The behaviour of a clocked sequential circuit is dependent on the i/ps, the outputs and the state of its flip flop. The outputs as well as the next state both are function of i/ps and present state. The analysis of the given clocked sequential circuit includes writing the state table and drawing the state ~~table~~ diagram for the given circuit.

i) **State Table**:- The basic element of a sequential CKT is a F/F. The o/ps Q_A, Q_B--- etc of F/Fs will be used state inputs to a sequential circuit. They are also called as state variable. Here X represents an external i/p and Y represents the o/p of the sequential CKT. Y will be dependent on the state variables and external i/p X.

Present state		Next state				Output Y	
		X=0		X=1			
Q_A	Q_B	Q_{A+}	Q_{B+}	Q_{A+}	Q_{B+}	X=0	X=1
0	0	0	0	0	1	0	0
0	1	1	1	0	1	0	0
1	0	1	0	0	0	0	1
1	1	1	0	1	1	1	0

ii) State diagram :- The information available in the state table is represented graphically using the state diagram which is shown in fig. below.

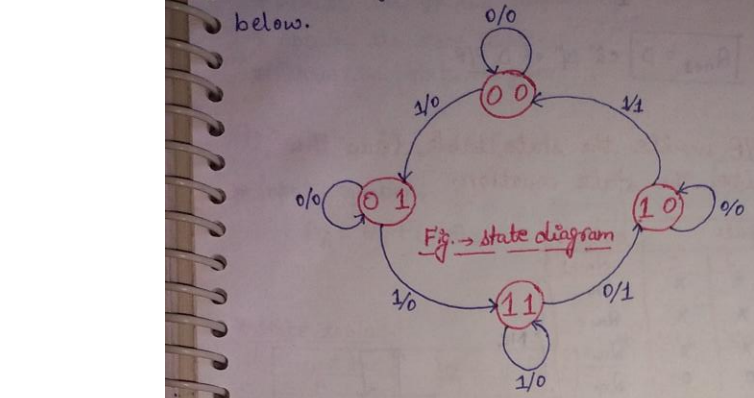
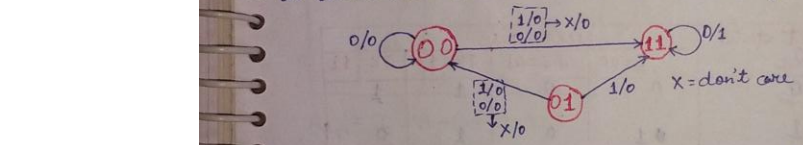


Fig. → state diagram

Don't care condition in state diagram :- some-times the same next state is reached for more than one present states. This is called as don't care condition in the state diagram.



iii) State equation :- State equation is an algebraic equation. The left side of this equation represents the next state of the flip flops. And the right hand side of this equation specifies the present state condition which make the next state.

State equation also called as application equation.

Q. For a clocked D F/F write the state table, draw the state diagram and write the state equation.

Solu: **i) Truth table :-**

Inputs		outputs	
CLK	D	Q_{n+1}	$\bar{Q}_{n+1}$
0	X	Q_n	$\bar{Q}_n$
1	X	Q_n	$\bar{Q}_n$
↓	X	Q_n	$\bar{Q}_n$
↑	0	0	1
↑	1	1	0

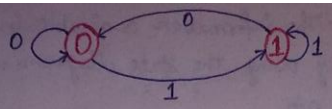
NC

Follow D F/F

ii) State table :-

Present state Q_n	Next state D=0	Next state D=1
0	0	1
1	0	1

iii) State diagram :-



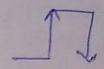
iv) State equation :- $Q_{n+1} = D$ Ch^r. eq^n. of D F/F.

Q For a clocked JK F/F write the state table, draw the state diagram and write the state equation.

Solu:

i) Truth table :-

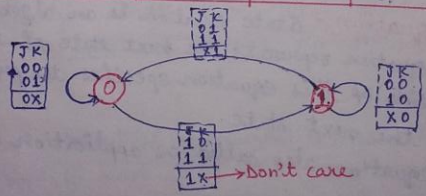
Inputs			Outputs	
CLK	J	K	Q_{n+1}	
0	X	X	Q_n	NC
1	X	X	Q_n	
↓	X	X	Q_n	
↑	0	0	Q_n	
↑	0	1	0	→ RESET
↑	1	0	1	→ SET
↑	1	1	$\bar{Q}_n$	→ Toggle



ii) State table :-

Present state Q_n	Next state			
	JK=00	JK=01	JK=10	JK=11
0	0	0	1	1
1	1	0	1	0

iii) State diagram :-



iv) State equation :- $Q_{n+1} = J\bar{Q}_n + \bar{K}Q_n$

Q. For a toggle FF write the state table, draw the state diagram and write the state equation.

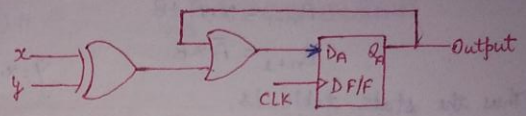
(12) Analysis Process :-

* Write the i/p and output eqn from the given circuit.
* Obtain the state table.
* Draw the state diagram.

i) Analysis with D F/Fs :-

* Input equations,

$$D_A = Q_A + (x \oplus y)$$

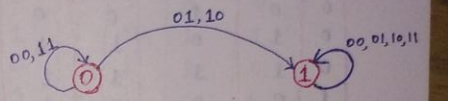


Logic diagram

* State table :-

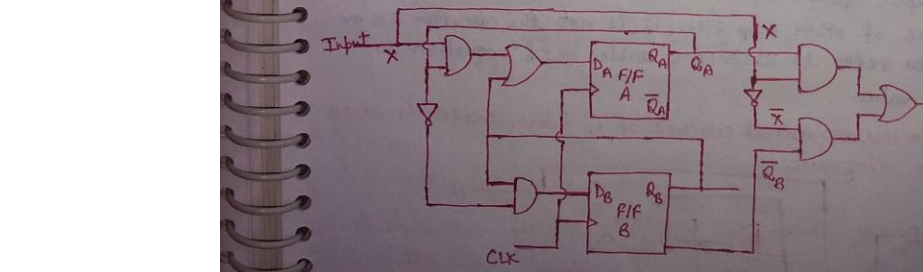
Present state	Inputs		Next state
A	x	y	A_{n+1}
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

* State diagram :-



$$A_{n+1} = D_A = A + (x \oplus y)$$

Q. Analyze the sequential CKT of fig. below. Write the input, output equations, obtain the state table and draw the state diagram.



Solu From given circuit, $D_A = X \cdot Q_A + Q_B$

$D_B = \overline{Q}_A * Q_B$

$Y = X Q_A + \overline{X}\,\overline{Q}_B$

The next state is obtained from the state equation. For a D F/F the state equation is same as input equation.

$A_{n+1} = X \cdot A + B$

$B_{n+1} = \overline{A} \times B$

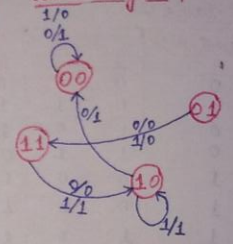
$Y = X \cdot A + \overline{X} \cdot \overline{B}$

Thus the state table is

Present A	Present B	Input X	Next A_{n+1}	Next B_{n+1}	Output Y
0	0	0	0	0	1
0	0	1	0	0	0
0	1	0	1	1	0
0	1	1	1	1	0
1	0	0	0	0	1
1	0	1	1	0	1
1	1	0	1	0	0
1	1	1	1	0	1

state table

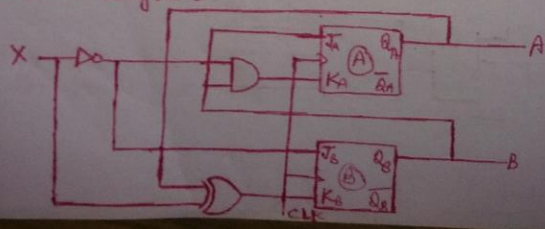
state diagram:-



ii) Analysis using JK Flip Flops:- In case of D flip flop, the state equation which is obtained the next state value to the state table is same as the input equations.

But in case of other flip flops, it is not the case. For JK or T F/F, we have to refer to their C-S tables or Ch^r. equation to obtain the next value.

Q: Analyze the clocked sequential circuit of fig below. Write the state table and state diagram.



Solu: From given circuit, $J_A = B$, $K_A = \bar{X}\cdot B$, $J_B = \bar{X}$, $K_B = X\bar{A} + \bar{X}A$

Now write the next state eqn by using char. eqn of JK flip flop,

$J_{A+1} = J_A \cdot \bar{A} + \bar{K}_A \cdot A$ ~~$J_{B+1} = J_B \cdot \bar{B} + \bar{K}_B \cdot B$~~ $\overline{X\bar{A}} * \overline{\bar{X}A}$ → $(\bar{X}+A)(X+\bar{A})$

$\Downarrow$

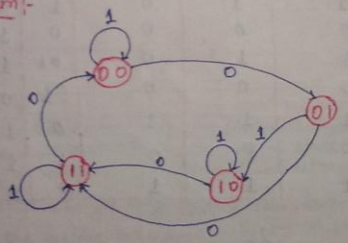
$A_{n+1} = B\cdot\bar{A} + \overline{(\bar{X}B)}\cdot A$ $\qquad B_{n+1} = \bar{X}\bar{B} + \overline{(X\bar{A}+\bar{X}A)}\cdot B$

$A_{n+1} = B\bar{A} + (X+\bar{B})A$ $\qquad$ ~~$B_{n+1} = \bar{X}\bar{B} + X\bar{A}B + \bar{X}AB$~~

State table:-

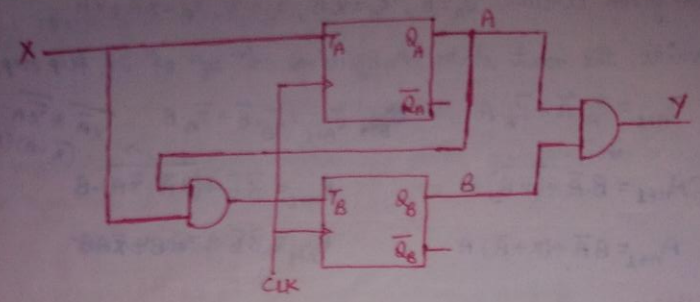
Present state		Input	F/F Input				Next State	
A	B	X	J_A	K_A	J_B	K_B	A_{n+1}	B_{n+1}
0	0	0	0	0	1	0	0	1
0	0	1	0	0	0	1	0	0
0	1	0	1	1	1	0	1	1
0	1	1	1	0	0	1	1	0
1	0	0	0	0	1	1	1	1
1	0	1	0	0	0	0	1	0
1	1	0	1	1	1	1	0	0
1	1	1	1	0	0	0	1	1

State diagram:-



iii) Analysis using T Flip Flop:-

Q. Analyze the circuit of fig below. Obtain the state table and draw the state diagram.



Sol:- From the circuit, $T_A = X$, $T_B = X \cdot A$

$$Y = A \cdot B$$

And the eqn for next state written from c-s eqn of T F/F.

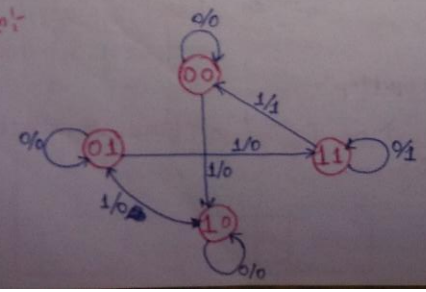
$$A_{n+1} = T_A \cdot \overline{A} + \overline{T_A} \cdot A \qquad B_{n+1} = T_B \cdot \overline{B} + \overline{T_B} \cdot B$$

$$A_{n+1} = X\overline{A} + \overline{X} \cdot A \qquad B_{n+1} = X \cdot A \cdot \overline{B} + (\overline{X} + \overline{A}) \cdot B$$

State table :-

Present state		Input	f/f inputs		Next state		Output
A	B	X	T_A	T_B	A_{n+1}	B_{n+1}	Y
0	0	0	0	0	0	0	0
0	0	1	1	0	1	0	0
0	1	0	0	0	0	1	0
0	1	1	1	0	1	1	0
1	0	0	0	0	1	0	0
1	0	1	1	1	0	1	0
1	1	0	0	0	1	1	1
1	1	1	1	1	0	0	1

State diagram :-



(13) Models of Synchronous Sequential Circuits :-

There are two models developed for representing the synchronous sequential circuit.

i) Moore circuit ii) Melay circuit

→ The synchronous sequential circuit is called as a Moore circuit if the output depends only on the present state of flip flops.

→ The circuit is called as a Melay circuit if the output is dependent on the present state of flip flops and the external inputs.

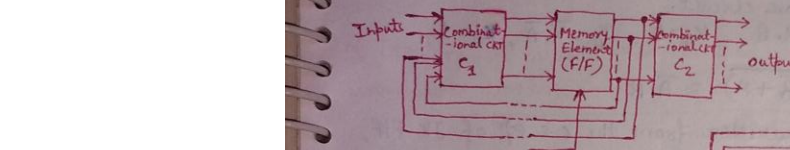


Fig → General form of Moore CKT

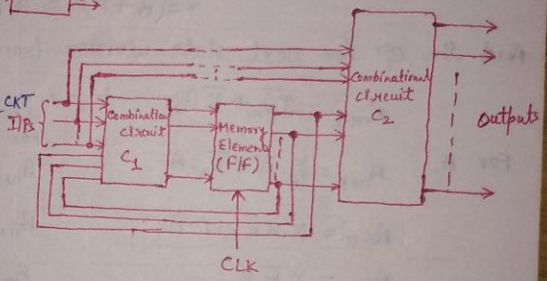
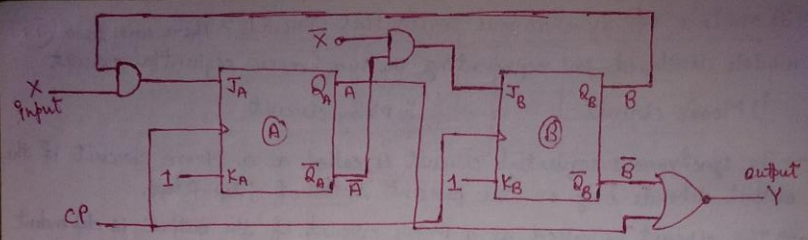


Fig. → General form of Melay CKT

* Comparison b/w Moore and Melay :-

S.N.	Moore Circuit	Melay circuit
1.	The final output depends only on the present state of memory elements.	The final output depends on the present state of memory element and external inputs.
2.	The output changes only after the active CLK edge.	Output can change in b/w the CLK edge if the external inputs change.
3.	The implementation of a logic fu^n needs more number of states than Melay circuit.	Implementation of the ~~CKT~~ same logic fu^n requires less number of states than Moore CKT.

Q. Identify the type of circuit given in fig. below, write the state table and draw the state diagram.



Solu. Since output $Y = \overline{A+B}$, it is dependent only on the present state of memory elements. Hence this is a moore circuit.

From given circuit, $J_A = X \cdot B$, $K_A = 1$, $J_B = \overline{X} \cdot \overline{A}$, $K_B = 1$

$$Y = \overline{(\overline{B} + A)} = \overline{A} \cdot B$$

And the eqn for next state written from the c-s eqn of JK FlF,

$$Q_{n+1} = J\overline{Q}_n + \overline{K} Q_n$$

For A,

$$A_{n+1} = J_A \cdot \overline{A} + \overline{K}_A \cdot A$$

$$A_{n+1} = X \cdot B\,\overline{A} + 0 \cdot A$$

$$A_{n+1} = X\,\overline{A}\,B$$

For B,

$$B_{n+1} = J_B \cdot \overline{B} + \overline{K}_B \cdot B$$

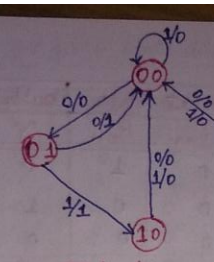
$$B_{n+1} = \overline{X}\,\overline{A} \cdot \overline{B} + 0 \cdot B$$

$$B_{n+1} = \overline{X}\,\overline{A}\,\overline{B}$$

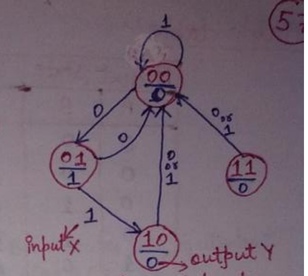
State table :-

Present state		I/P	Next state		Output
A	B	X	A_{n+1}	B_{n+1}	Y
0	0	0	0	1	0
0	0	1	0	0	0
0	1	0	0	0	1
0	1	1	1	0	1
1	0	0	0	0	0
1	0	1	0	0	0
1	1	0	0	0	0
1	1	1	0	0	0

State diagram :-

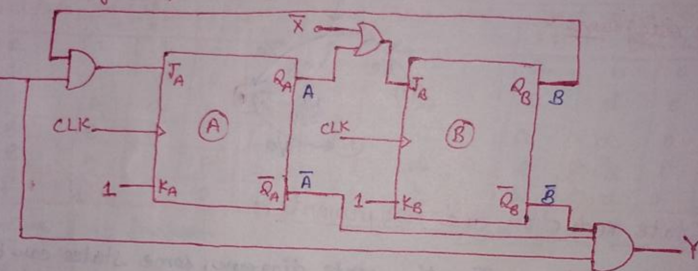


State diagram



Modified state diagram

Q. Identify the circuit given in fig. below. Write the state table and draw state diagram for the same.



Soln Since output, $Y = \overline{B} \cdot \overline{A} \cdot X$, it is dependent on the present state as well as the X input. Thus this is a Melay circuit.

From the given circuit,

$J_A = X \cdot B$, $K_A = 1$, $J_B = A + \overline{X}$, $K_B = 1$ and $Y = \overline{A}\,\overline{B}X$

And the next state eqns,

$A_{n+1} = J_A \cdot \overline{A} + \overline{K_A} A$

$= X\overline{A}B + 0$

$A_{n+1} = X\overline{A}B$

$B_{n+1} = J_B \cdot \overline{B} + \overline{K_B} \cdot B$

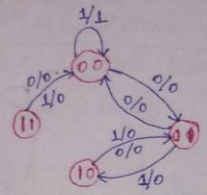
$B_{n+1} = (A + \overline{X})\overline{B} + 0$

$B_{n+1} = A\overline{B} + \overline{X}\overline{B}$

Stable table :-

Present state		I/P	Next state		output
A	B	X	A_{n+1}	B_{n+1}	Y
0	0	0	0	1	0
0	0	1	0	0	1
0	1	0	0	0	0
0	1	1	1	0	0
1	0	0	0	1	0
1	0	1	0	1	0
1	1	0	0	0	0
1	1	1	0	0	0

State diagram :-



(14) State reduction and Assignments :-

i) State reduction :- In the state diagram, some states can be redundant states. The redundant states are avoided by the technique called state reduction technique. The state reduction results in elimination of redundant states which ultimately results in reducing the no. of F/Fs and gates to be used in a sequential CKT.

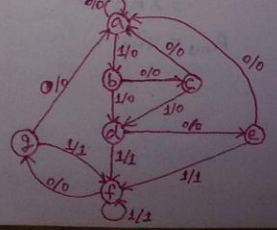
Steps :- i) write the state table from the given state diagram.

ii) Identify the equivalent states.

iii) Eliminate one of the two equivalent states and write new state table

iv) In the new state table, check for equivalent states and till all the redundant states are eliminated.

Ex:



Write state table for given state diagram

Present state	Next state X=0	Next state X=1	output Y X=0	output Y X=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	g	f	0	1
g	a	f	0	1

→ Both states are equivalent states (e & g)

Modified state table :-

Present state	Next state X=0	Next state X=1	output X=0	output X=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	f	0	1
e	a	f	0	1
f	(g) e	f	0	1

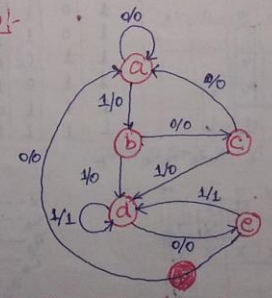
Replacing g by e

Reduced state table :-

Present state	Next state X=0	Next state X=1	output X=0	output X=1
a	a	b	0	0
b	c	d	0	0
c	a	d	0	0
d	e	(f) d	0	1
e	a	(f) d	0	1

Replacing f by d

Reduced state diagram :-



Q. For a given state table do the state reduction and realize using F/F.

PS	X=0 NS	X=0 O/P	X=1 N.S	X=1 O/P
a	a	0	b	0
b	c	1	d	0
c	a	1	—	—
d	e	1	f	c
e	g	1	—	—
f	a	e	—	—
g	e	1	—	—

Solu: Draw the state table in original form:-

PS	NS		Output	
	X=0	X=1	X=0	X=1
a	a	b	0	0
b	c	d	1	0
c	a	–	1	–
d	e	f	1	0
e	g	–	1	–
f	a	–	1	–
g	e	–	1	–

→ Both states are equivalent so replace f by c

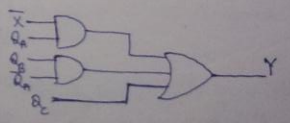
Now draw the reduced state table:-

PS	NS		Output	
	X=0	X=1	X=0	X=1
a	a	b	0	0
b	c	d	1	0
c	a	–	1	–
d	e	c	1	0
e	g	–	1	–
g	e	–	1	–

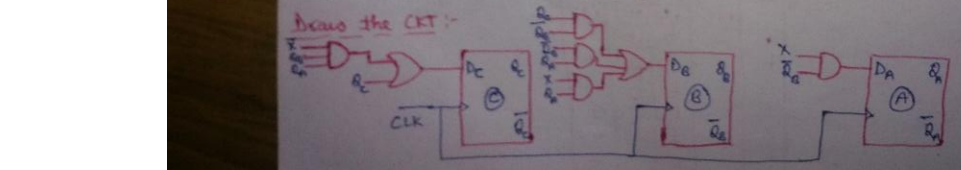
Draw the CKT excitation table

Input	PS			NS			F/F I/P			Output
X	Q_C	Q_B	Q_A	Q_{C+1}	Q_{B+1}	Q_{A+1}	D_C	D_B	D_A	Y
0	0	0	0a	0	0	0a	0	0	0	0
0	0	0	1b	0	1	0c	0	1	0	1
0	0	1	0c	0	0	0a	0	0	0	1
0	0	1	1d	1	0	0e	1	0	0	1
0	1	0	0e	1	1	0g	1	1	0	1
0	1	0	1f	x	x	x	x	x	x	x
0	1	1	0g	1	x	0e	1	x	0	1
0	1	1	1h	x	x	x	x	x	x	x
1	0	0	0a	a	0	1h	0	0	1	0
1	0	0	1b	0	1	1d	0	1	1	0
1	0	1	0c	x	x	x	x	x	x	x
1	0	1	1d	0	1	0c	0	1	1	0
1	1	0	0e	x	x	x	x	x	x	x
1	1	0	1f	x	x	x	x	x	x	x
1	1	1	0g	x	x	x	x	x	x	x
1	1	1	1h	x	x	x	x	x	x	x

Now draw the K-Map:-



Draw the CKT:-



(15) Design Procedure for clocked Sequential CKTs :-

Step I :- A state diagram or timing diagram or some other information is given which describe the behaviour of the CKT that to be designed.

Step II :- Obtain the state table.

Step III :- The no. of states can be reduced by state reduction methods if the sequential CKT can be characterized by I/o relationships, independent no. of stat

Step IV :- Assign binary values to each state if the state table obtained in Step II & III contain letter symbols.

Step V :- Determine the no. of F/Fs required and assign a letter symbole to each.

Step VI :- Decide the type of F/F to be used.

Step VII :- Derive the CKT excitation table and o/p table from the state table.

Step VIII :- Obtain the expressions for the CKT o/p and F/F i/ps.

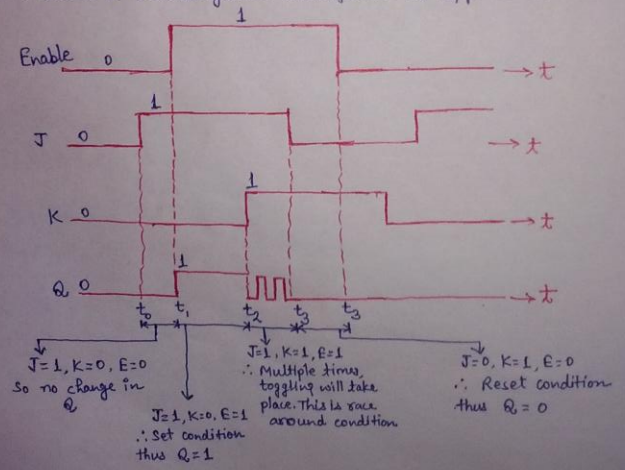
Step IX :- Draw the logic diagram.

⇒Difference between Latch and Flip flops :-

* Latches and Flip flops both are basically the bistable elements.
* A latch has got an enable input. As long as it is active, the latch output will keep changing according to the changes in its input. In other words latch is a level triggered Flip Flop.
* But flip flop is a sequential circuit which generally samples its inputs and changes its outputs only at particular instants of time and not continiously
* The Flip flops are therefore said to be edge sensitive or edge triggered rather than being level triggered latches.

⇒Race Around condition in JK Latch :-

The "Race around condition" that we are going to explain, this condition occurs when the latch is in toggle mode. Fig. below shows the waveforms for the various modes, when a rectangular waveform is applied to the "Enable" input.



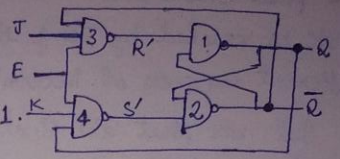
Interval t_2-t_3 : Race Around.

* At instant J=1=K & E=1. Hence JK latch is in the toggle mode and Q becomes low and $\overline{Q} = 1$.

* These changes outputs get applied at the i/Ps of NAND gates-3 & 4 of the JK latch. Thus the new i/Ps to Gate 3 & 4 are, (2)

NAND-3 $\bar{J}=1$, $R=1$, $\bar{Q}=1$

NAND-4 $K=1$, $E=1$ $Q=0$



Hence R' will becomes 0 & S' will become 1.

* Therefore after a time period corresponding to the propogation delay, the Q and $\bar{Q}$ outputs will change to, $Q=1$ and $\bar{Q}=0$. These changed o/p again get applied to the inputs of NAND-3 & 4 and the o/Ps will toggle again.

This multiple toggling in the JK-latch is called as Race Arround condition. It must be avoided.

⇒ How avoid race arround condition:- The race arround condition in JK latch/Flip Flop can be avoided by

i) Using edge triggered JK F/F ii) Using master slave JK F/F.

⇒ Race:- ~~both~~ we have already discussed that the race condition in F/Fs. There both the outputs try to changes their state in response to the change in input.

In an asynchronous machine a race condition is said to be have occured if two or more states varies changes their values when their is a state transition. There are two categories of races;

i) Non-critical race ii) Critical race.

i) Non-critical race :- A race is said to be non-critical when a correct stable next state is eventually reached, during the state transition. Due to unequal delays the state variables may change in an unpredictable manner if race condition occurs.

The final stable state attained by the circuit may or may not depend on the order in which the state variables changes due to race condition. If the final stable state reached by the CKT is not dependent on the order in which the state variables changes, then the race condition is called as a Non-critical Race and it is not harmful.

Fig. below shows the concept of the non-critical race.

Fig. shows that the initial state is 11 and the final stable state is 10. This stable state is reached by going through three different sequences of intermediate states.

But note that the final stable state reached in all the three cases is same i.e 10. Hence we conclude that the final state to which the asynchronous circuit reaches does not depend on the order in which the state variables change. Hence this is a non-critical race.

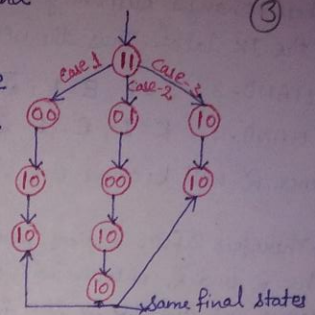


Fig. → Non-critical race

UNIT-4th Registers and Counters

Introduction:- F/F is a 1 bit memory cell which can be used for storing the digital data. To increase the storage capacity in terms of no. of bits, we have to use a group of F/F. Such a group of F/Fs is known as a register. Thus register is a group of F/Fs. The n-bit register will consist of 'n' number of F/Fs and it is capable of storing an 'n-bit' word.

① Shift Registers:- As a matter of fact, the binary data in a register can be moved within the register from one F/F to the other or outside it with application of clock pulses. Those registers which allows such data transfers are known as shift register.

Mode of operation of a shift Register:- The various modes in which a shift register can operate may be listed as,

i) Serial input Serial output (SISO)
ii) Serial input parallel output (SIPO)
iii) Parallel in serial out (PISO)
iv) Parallel in parallel out (PIPO)

i) SISO (shift left Mode):- Fig. below shows the serial input serial o/p type register ~~whi~~ with shift left mode.

Let us consider that all the F/Fs initially are in the reset condition. This means that i.e $Q_3 = Q_2 = Q_1 = Q_0 = 0$. Let us illustrate the entry of a four bit binary no. 1 1 1 1 into the register. When this is to be done, this number must be applied to D_{in} bit by bit with the MSB bit applied first.

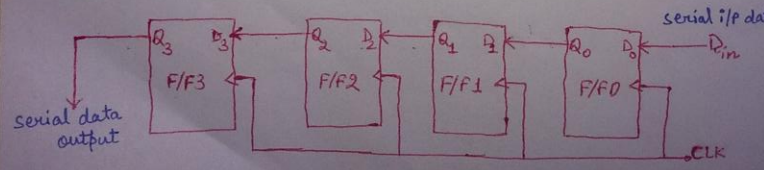
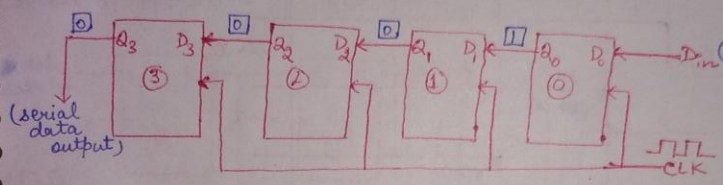


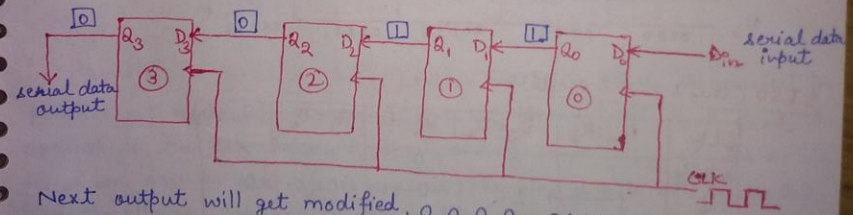
Fig. → SISO (shift left Mode)

This means that the D i/p of F/F-0 i.e D_0 is connected to serial data input D_{in}. Also, output of F/F-0 i.e Q_0 is connected to the i/p of the next F/F i.e D_1 and so on.

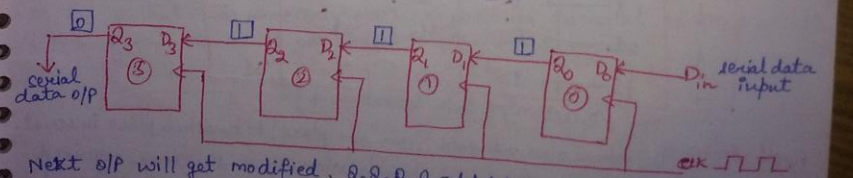
Working Operation:- Before application of CLK pulse, let us consider that $Q_3Q_2Q_1Q_0 = 0000$ and we apply MSB bit of the no. to be entered to D_{in}. Therefore $D_{in} = D_0 = 1$. Now we apply the clock. On the first falling edge of CLK, the F/F-0 is set and the stored word in the register will be $Q_3Q_2Q_1Q_0 = 0001$



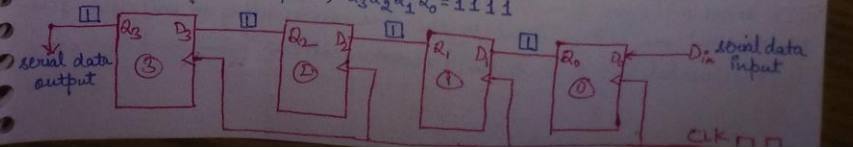
Then we apply the next bit to D_{in}. Hence, $D_{in} = 1$ As soon as the next -ve edge of the CLK hits, F/F-1 will set and the stored word will changes to, $Q_3Q_2Q_1Q_0 = 0011$



Next output will get modified, $Q_3Q_2Q_1Q_0 = 0111$.



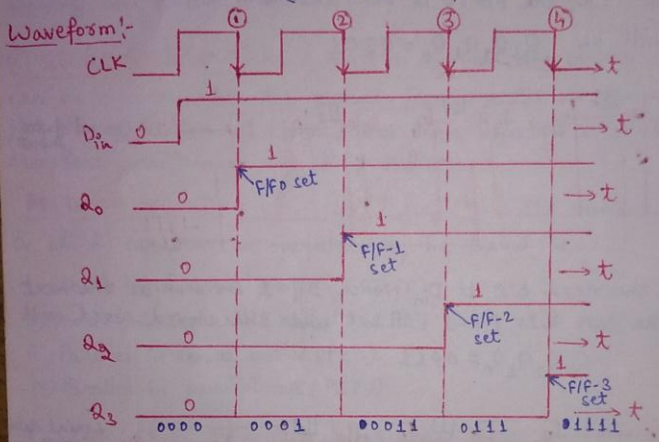
Next o/p will get modified, $Q_3Q_2Q_1Q_0 = 1111$



CLK	Q_3	$Q_2 = D_3$	$Q_1 = D_2$	$Q_0 = D_1$	Serial I/P $D_0 = D_{in}$
Initially	0	0	0	0	
↓	0	0	0	1	1
↓	0	0	1	1	1
↓	0	1	1	1	1
↓	1	1	1	1	1

← Direction of data travel

Waveform:-



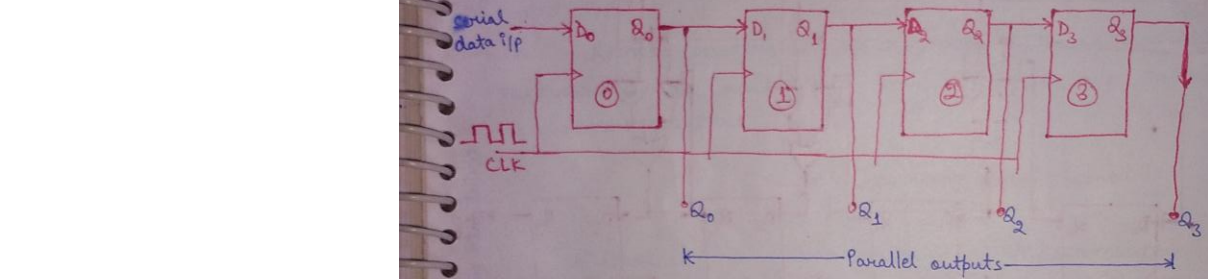
SISO (Shift Right Mode):-

CLK	$D_{in} = Q_3$	$Q_3 = D_2$	$Q_2 = D_1$	$Q_1 = D_0$	$Q_0 = D_1$
Initially		0	0	0	0
↓	1 →	1	0	0	0
↓	1 →	1	1	0	0
↓	1 →	1	1	1	0
↓	1 →	1	1	1	1

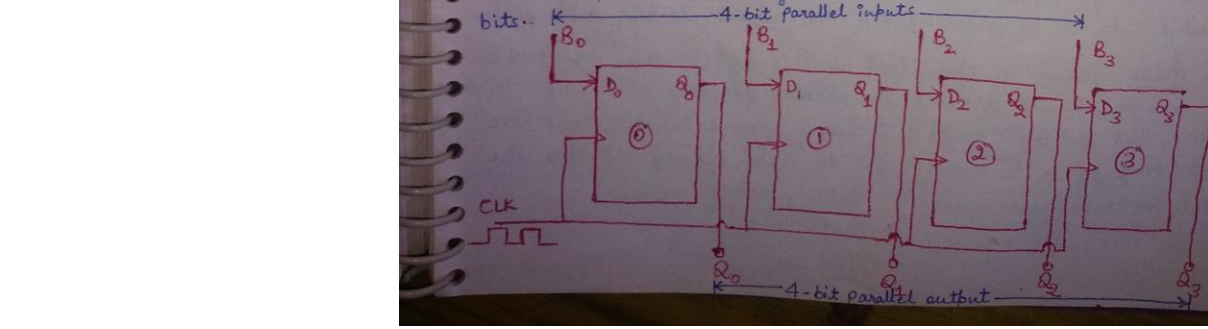
Direction of data travel →

Application:- i) Transmission of data from one place to another place in serial.
ii) It takes a longer time for serial transmission, since the time required to transmit one bit is equal to the time corresponding to one clock cycle.
iii) It is useful for long distance communication.

ii) Serial in Parallel out (SIPO) :- In this particular operation, the data is entered serially and then taken out in parallel. This means that first the data is loaded bit by bit. The o/Ps are disabled as long as the loading is taking place. As soon as the loading is complete and all the F/Fs consist of their required data, the o/Ps are enabled so that all the loaded data is made available over all the o/p lines simultaneously. Also no. of CLK cycles required to load a four bit word is 4. Therefore, the speed of operation of SIPO mode will remain ~~not~~ same as SISO.



iii) Parallel in Parallel out (PIPO) :- Fig. below shows the parallel in parallel out mode of operation. The 4-bit binary input B_0, B_1, B_2, B_3 is applied to the data inputs D_0, D_1, D_2 and D_3 respectively of the 4-F/Fs. As shown as -ve CLK edge is applied, the binary bits will be loaded into the F/Fs simultaneously. The loaded bits will appear simultaneously to the output side. ~~One~~Only one CLK pulse is essential to load all the bits.



iv) Parallel in serial out (PISO) :- In this type of register, the bits are entered in parallel i.e shown in fig. below. This CKT shows a 4-bit parallel input serial output register. Here the o/p of the previous F/F is connected to the input of the next one via a combinational CKT. The binary input word B_0, B_1, B_2 & B_3 is applied through the same combinational CKT. There are two modes in which this CKT can work namely shift mode or load mode.

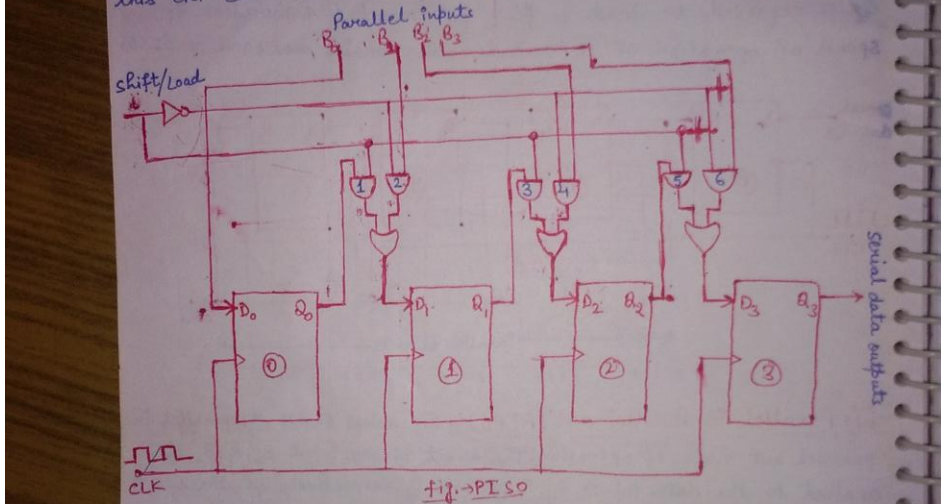


fig. → PISO

Load Mode :- When the shift/load line is low(0), the AND gate 2, 4 & 6 becomes active. They will pass B_1, B_2, B_3 bits to the corresponding F/Fs.
→ On the low going edge of clock, the binary inputs B_0, B_1, B_2, B_3 will get loaded into the corresponding F/Fs. Thus parallel loading takes place.

Shift Mode :- When the shift/load line is high(1) then AND gate 2, 4 and 6 becomes inactive. Hence the parallel loading of the data becomes impossible. But the AND gates 1, 3 and 5 becomes inactive. Therefore the shifting of data from left to right bit by bit. Thus the parallel in serial out operation takes place.

(2) Counters :-

The digital CKT used for counting pulses is known as counter. It is a sequential ckT. Counters count the number of clock pulses. Hence with some modifications it can be used for measuring fr. or time period.

* Types of counters :-

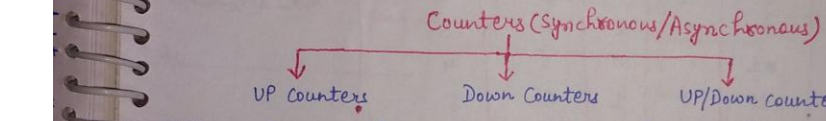
Counters are basically of two types.

i) Asynchronous Counter or ripple counters.

ii) Synchronous counters.

⇒ In Asynchronous or ripple counters the external clock signal is applied to one flip flop and then o/p of preceding F/F is connected to the CLK of next F/F.

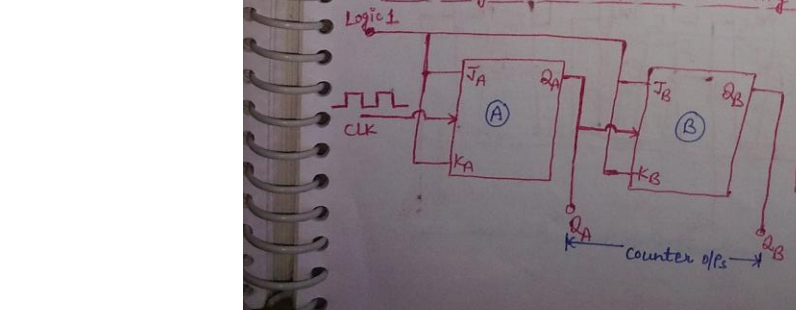
⇒ In Synchronous counters all the F/Fs receive the external clock pulse simultaneously. Ring Counter and Johnson counter are the exp examples of synchronous counters.



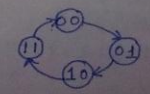
* Up counters are the counters that count from small to big count.
* Down counters are the counters that count from big to small count.
* UP/Down counter is the combination of up and down counter.

i) Asynchronous UP counters :-

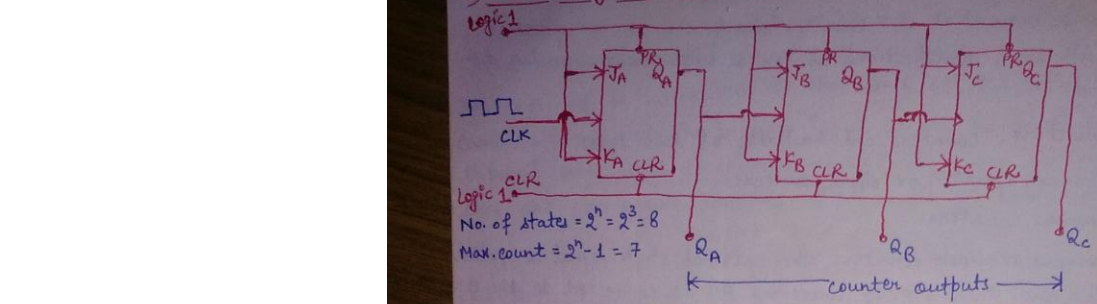
a) 2-bit Asynchronous Up counter using JK F/Fs :-



CLK	F/F o/Ps Q_B	F/F o/Ps Q_A	State Number	Decimal equivalent of counter o/p
Initially	0	0	—	0
↓	0	1	1	1
↓	1	0	2	2
↓	1	1	3	3
↓	0	0	4	0



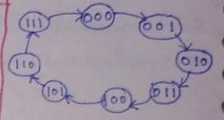
b) 3-bit Asynchronous Up Counter :-



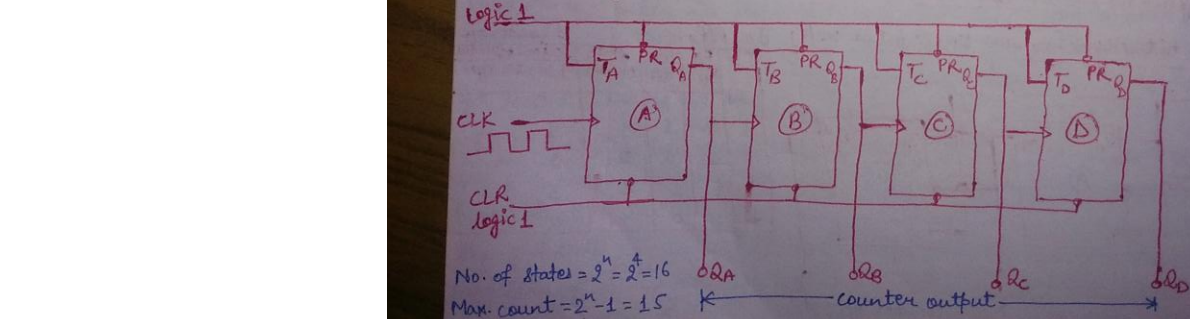
No. of states = $2^n = 2^3 = 8$

Max. count = $2^n - 1 = 7$

CLK	F/F outputs			states	Decimal Equivalent
	Q_C	Q_B	Q_A		
Initially	0	0	0	1	0
1st (↓)	0	0	1	2	1
2nd (↓)	0	1	0	3	2
3rd (↓)	0	1	1	4	3
4th (↓)	1	0	0	5	4
5th (↓)	1	0	1	6	5
6th (↓)	1	1	0	7	6
7th (↓)	1	1	1	8	7
8th (↓)	0	0	0	1	8



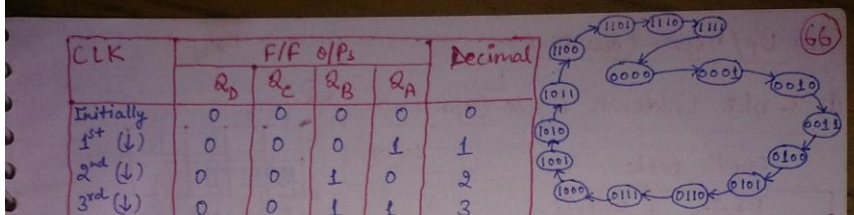
c) 4-bit Asynchronous up Counter :-

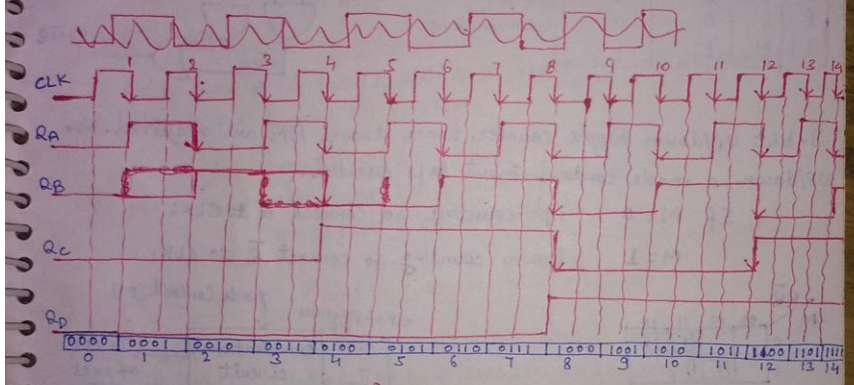


No. of states = $2^n = 2^4 = 16$

Max. count = $2^n - 1 = 15$

CLK	F/F O/Ps				Decimal
	Q_D	Q_C	Q_B	Q_A	
Initially	0	0	0	0	0
1st (↓)	0	0	0	1	1
2nd (↓)	0	0	1	0	2
3rd (↓)	0	0	1	1	3
4th (↓)	0	1	0	0	4
⋮	⋮	⋮	⋮	⋮	⋮
15th (↓)	1	1	1	1	15
16th (↓)	0	0	0	0	0

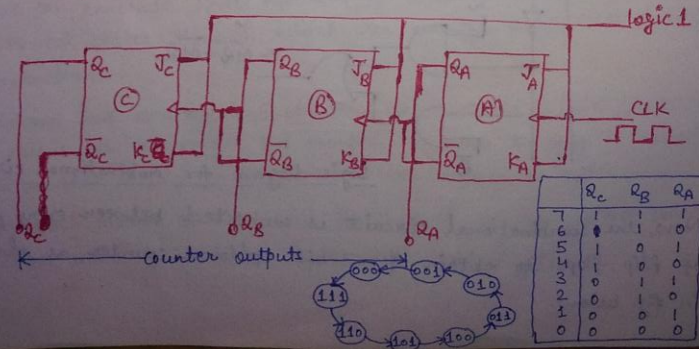




waveform

ii) Asynchronous down Counter :-

a) 3-bit Asynchronous down Counter :-



	Q_C	Q_B	Q_A
7	1	1	1
6	1	1	0
5	1	0	1
4	1	0	0
3	0	1	1
2	0	1	0
1	0	0	1
0	0	0	0

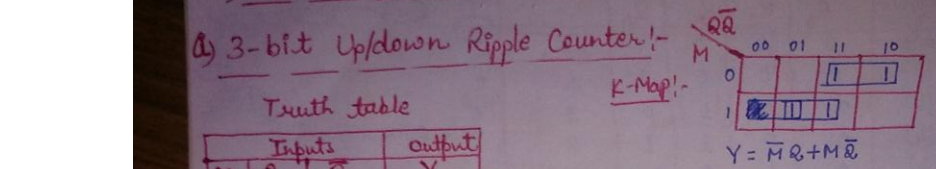
iii) Up/Down Counters :-

a) 3-bit Up/down Ripple Counter :-

Truth table

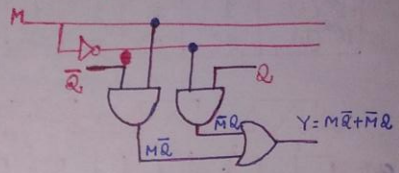
Inputs			output	
M	Q	$\bar{Q}$	Y	
0	0	0	0	Y = Q for up count
0	0	1	0	
0	1	0	1	
0	1	1	1	
1	0	0	0	Y = $\bar{Q}$ for down counting
1	0	1	1	
1	1	0	0	
1	1	1	1	

K-Map :-



$Y = \bar{M}Q + M\bar{Q}$

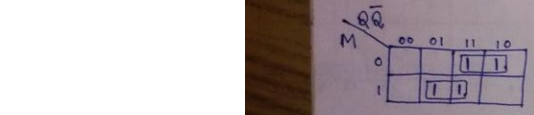
Logic diagram :-



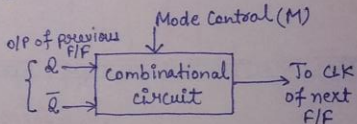
3-bit up/down Ripple counter hence three FFs are required. For up/down, a mode control input is essential.

If M = 0 up counting. So connect Q to CLK.

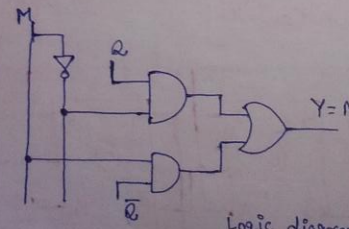
M = 1 Down counting. So connect $\bar{Q}$ to CLK.



$Y = M\bar{Q} + \bar{M}Q$



Block diagram



Logic diagram for combinational circuit

Now, the combinational circuit is connected between every pair of flip flops to obtain the 3-bit up/down counter as shown in fig. below.

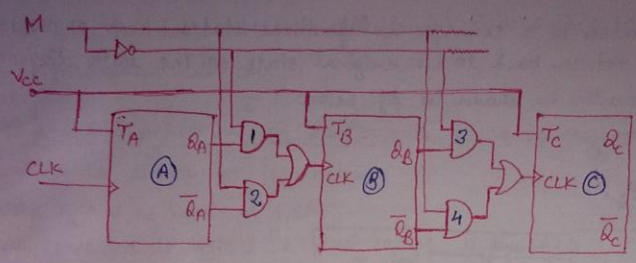


Fig. → 3 bit up/down counter

i) With M=0 (Up counting mode) :-

If M=0 then AND gates 1 and 3 in above fig. will be enabled & AND gates 2 and 4 will be disabled.

ii) With M=1 (Down counting mode) :-

If M=1 then AND gates 1 & 3 → Disabled.
AND gates 2 & 4 → Enabled.

③ Modulus of the Counter (MOD-N counter) :-

* The 2-bit ripple counter is called as MOD-4 counter and 3-bit ripple counter is called as MOD-8 counter. So in-general, an n-bit ripple counter is called as modulo-N counter where,

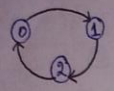
$$\text{Mod-number} = 2^n$$

So we can conclude that modulus of a counter represents the number of states through which the counter progresses during its operation.

* Can we have a modulo-5 counter using a 3-bit ripple counter? That means can we restrict the number of states to only 5 instead of 8 under normal condition? Here the answer is "Yes". We can design such modulo counters with the help of the basic ripple counter structure and a combinational logic called reset logic.

Q. Design a mod-3 asynchronous counter using a 2-bit ripple counter.

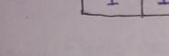
Solu: Mod-3 counter is a counter having three states i.e 00, 01, and 10. After 10 it will return back to its original state 00. The state diagram of the Mod-3 counter is shown in fig. below.



Truth Table :-

F/F outputs		o/p of Reset logic Y
Q_B	Q_A	
0	0	1
0	1	1
1	0	1
1	1	0 → Clear all the F/Fs

K-Map :-



$Y = \overline{Q}_A + \overline{Q}_B = \overline{Q_A \cdot Q_B}$

Logic diagram :-

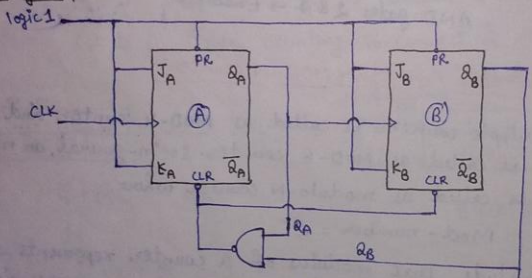
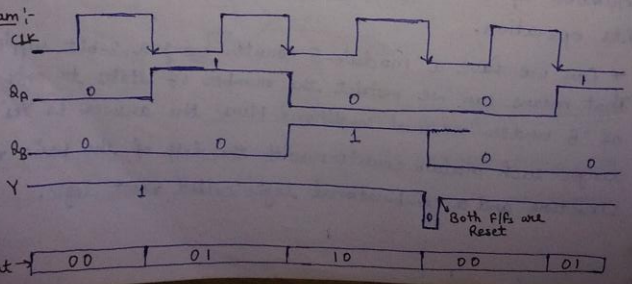


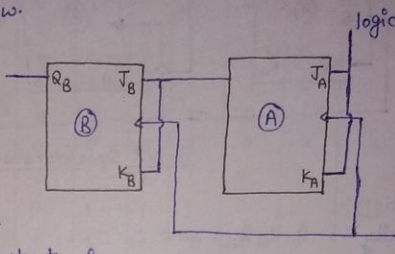
Fig. → Mod-3 counter using 2-bit ripple counter

Timing diagram :-



ii) Synchronous Counters :- If the CLK pulses are applied to all the F/Fs in a counter simultaneously, then such a counter is called as synchronous counter.

a) 2-Bit Synchronous up Counter :- A 2-bit or MOD-4 synchronous up counter is shown in fig. below.



* The J_A and K_A inputs of F/F (A) are connected to logic 1. So F/F A will work as a Toggle F/F. The J_B and K_B I/Ps are connected to Q_A. Hence F/F-B will toggle if $Q_A = 1$ and there won't to be any state change if $Q_A = 0$, at the instant of -ve CLK.

Operation :- Initially let both the F/Fs be in the reset state.

$$Q_B Q_A = 00$$

→ After 1st -ve CLK edge :- F/F-A will toggle and Q_A will change from 0 to 1. And at the instant of -ve clk edge $Q_A = 0$ ∴ $J_B = K_B = 0$ hence F/F-B will not change. i.e

$$Q_B Q_A = 01$$

→ After 2nd -ve CLK edge :- On the arrival of second -ve CLK edge, F/F-A toggles again and Q_A changes from 1 to 0. But at this instant Q_A was 1 so $J_B = K_B = 1$ and F/F-B toggle. Thus Q_B changes from 0 to 1 i.e

$$Q_B Q_A = 10$$

Similarly for third falling CLK edge, F/F-A will toggle from 0 to 1 but there is no change of state for F/F-B. i.e

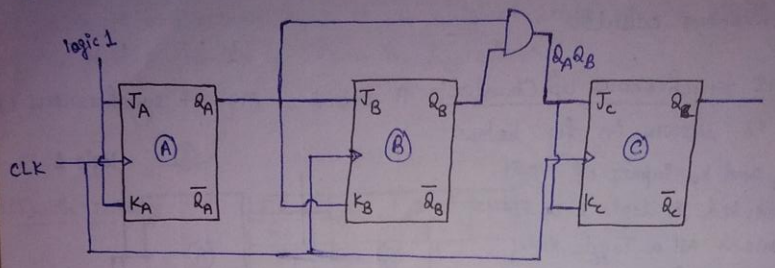
$$Q_B Q_A = 11$$

And for next -ve CLK edge F/F A will change from 1 to 0 and F/F-B will also change from 1 to 0 i.e

$$Q_B Q_A = 00$$

CLK	Counter O/Ps	
	Q_B	Q_A
Initially	0	0
↓	0	1
↓	1	0
↓	1	1
↓	0	0

b) 3-bit Synchronous Binary UP Counter :- 3-bit synchronous Binary Up counter shown in fig. below.



3-bit Synchronous binary counter

Operation :-

Initially → $Q_C Q_B Q_A = 000$

1st clock → $Q_C Q_B Q_A = 001$

2nd CLK → $Q_C Q_B Q_A = 010$

3rd CLK → $Q_C Q_B Q_A = 011$

4th CLK → $Q_C Q_B Q_A = 100$

⋮

7th CLK → $Q_C Q_B Q_A = 111$

8th CLK → $Q_C Q_B Q_A = 000$

CLK	Q_C	Q_B	Q_A
Initially	0	0	0
↓	0	0	1
↓	0	1	0
↓	0	1	1
↓	1	0	0
↓	1	0	1
↓	1	1	0
↓	1	1	1

Design of Synchronous Counter :-

Step I → Decide the number of F/Fs.

Step II → Write the excitation table of used F/F.

Step III → Write the excitation table of counter.

Step IV → From the CKT excitation table write K-maps & obtain eqns.

Step V → Draw the logic diagram.

Q. Using any type of flip flop design a 4-bit synchronous counter that up counts in gray code.

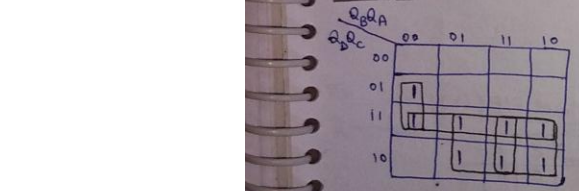
Solu.

PS				NS				F/F I/Ps			
Q_D	Q_C	Q_B	Q_A	Q_{D+1}	Q_{C+1}	Q_{B+1}	Q_{A+1}	D_D	D_C	D_B	D_A
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	1	0	0	1	1	0	0	1	1
0	0	1	1	0	0	1	0	0	0	1	0
0	0	1	0	0	1	1	0	0	1	1	0
0	1	1	0	0	1	1	1	0	1	1	1
0	1	1	1	0	1	0	1	0	1	0	1
0	1	0	1	0	1	0	0	0	1	0	0
0	1	0	0	1	1	0	0	1	1	0	0
1	1	0	0	1	1	0	1	1	1	0	1
1	1	0	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0	1	1	1	0
1	1	1	0	1	0	1	0	1	0	1	0
1	0	1	0	1	0	1	1	1	0	1	1
1	0	1	1	1	0	0	1	1	0	0	1
1	0	0	1	1	0	0	0	1	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0

These no are in gray code

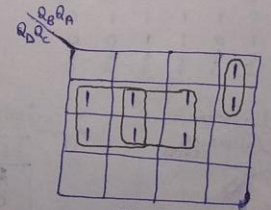
(Inputs of D F/F same as its next state values)

K-Map :-



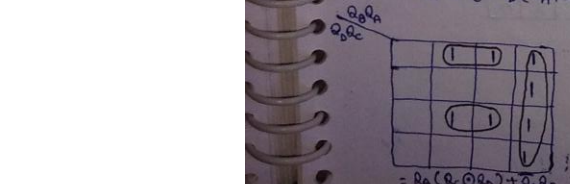
$D_D = \bar{Q}_B\bar{Q}_A Q_C + Q_D Q_C + Q_D Q_A + Q_D Q_B$

$= \bar{Q}_A\bar{Q}_B Q_C + Q_D(Q_A + Q_B + Q_C)$



$D_C = \bar{Q}_D Q_B \bar{Q}_A + Q_C\bar{Q}_B + Q_C Q_A$

$= \bar{Q}_A Q_B \bar{Q}_D + Q_C(Q_A + \bar{Q}_B)$



$= Q_A(Q_C \odot Q_D) + \bar{Q}_A Q_B$

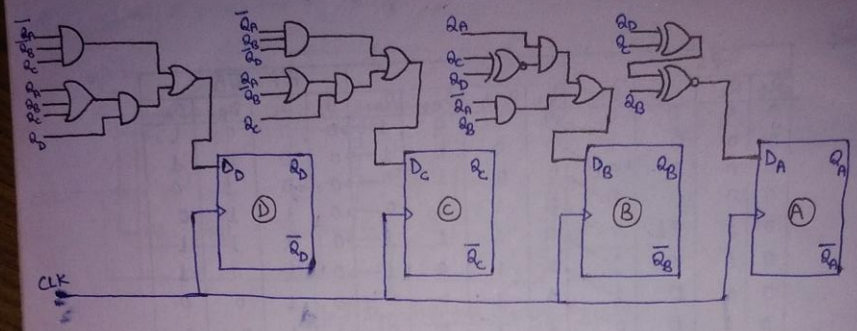
$D_B = Q_A\bar{Q}_C\bar{Q}_D + Q_A Q_C Q_D + \bar{Q}_A Q_B$

$D_A = \bar{Q}_B(\overline{Q_D \oplus Q_C}) + Q_B(Q_D \oplus Q_C)$

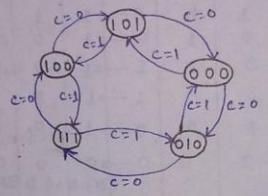
$D_A = Q_B \odot Q_D \oplus Q_C$

$D_A = \bar{Q}_B\bar{Q}_C\bar{Q}_B + Q_D Q_C\bar{Q}_B + \bar{Q}_D Q_C Q_B + Q_B\bar{Q}_C Q_B$

Logic diagram :-



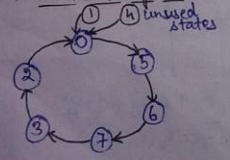
Q. Implement the state transition diagram shown in fig. below. using T F/F.



Q. Design a synchronous counter using JK F/F for the following i/p sequence.

A	0	1	1	1	0	0	0
B	0	0	1	1	1	1	0
C	0	1	0	1	1	0	0
	↓	↓	↓	↓	↓	↓	↓
	0	5	6	7	3	2	0

Solu. * State diagram :-



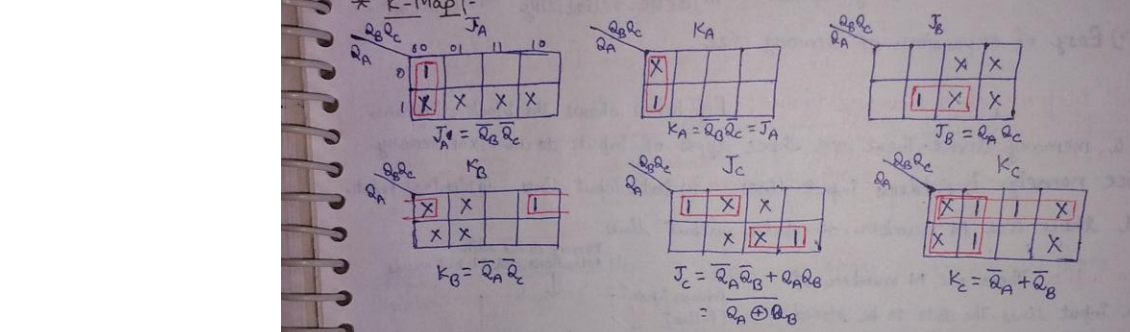
* Excitation table of used F/F (JK) :-

PS (Q_n)	NS (Q_{n+1})	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

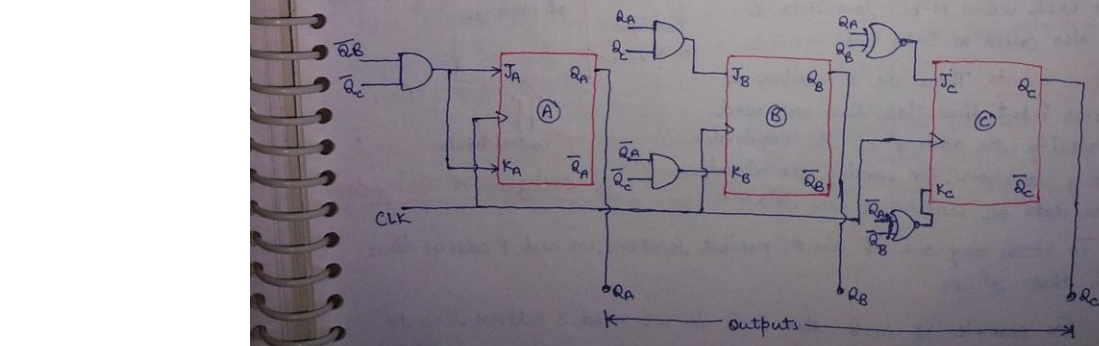
* Circuit excitation table :-

PS			NS			F/F i/Ps					
Q_A	Q_B	Q_C	Q_{A+1}	Q_{B+1}	Q_{C+1}	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	1	0	1	1	X	0	X	1	X
0	0	1	0	0	0	0	X	0	X	X	1
0	1	0	0	0	0	0	X	X	1	0	X
0	1	1	0	1	0	0	X	X	0	X	1
1	0	0	0	0	0	X	1	0	X	0	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	1	1	X	0	X	0	X	0

* K-Map :-



* Logic circuit diagram :-



Semiconductor Memories

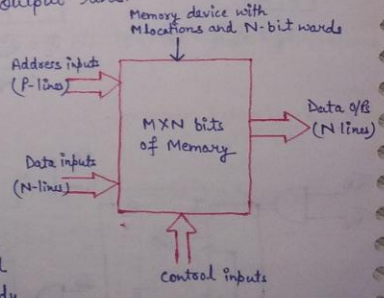
Introduction:- Any system which process digital data needs a facility for storing the unprocessed partially processed and completely processed data. A subsystem of such digital processing system which can store all the above mentioned data is called as memory. Earlier the memory used to be in the form of a magnetic type. But now a days we use semiconductor memories of various types and size.

Advantages of semiconductor Memory:- The main advantages of semiconductor memories are;

i) Small size ii) High speed iii) Better reliability iv) Low cost

v) Easy of expansion of memory size

Block Diagram of a Memory Device:- Fig. below shows the block diagram of a memory device. There are three types of inputs to an MXN memory device namely. i) Address input lines ii) Data input lines iii) Control inputs. And there are N number of data output lines.



Block diagram of Memory device

Data inputs:- There are N number of data input lines. The data to be stored is put on these lines word by word with each word N-bit long. Data i/ps are also called as Data bus.

Address inputs:- There are P number of address input lines. These lines are used to specify the address of the required memory location, for reading the already stored data or writing a new data.

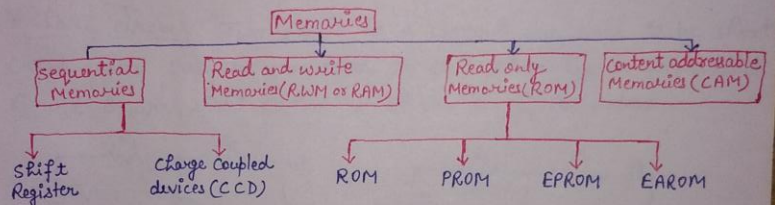
To access any one of the M possible locations, we need P address lines such that $2^P = M$

For example if M=8 then P=3. So we need 3 address lines to have 8 different combinations from 000 to 111. So address is specified in the binary form.

Data output lines :- The data available in the selected memory location can be ~~used~~ "read" on the data output lines. The number of data lines is N i.e equal to the number of bits per word. The output data lines are also called as output data bus. ~~The~~ Input and output data buses are unidirectional. That means the data flows ~~from~~ only in one direction.

Control lines :- The control lines include the read/write line and the chip select line (which acts as the enable input). The bidirectional bus is used as input data bus for some specific time when input data is to be loaded into memory (write operation). And it is used as output data bus for specific time when the stored data is to be read (read operation).

Classification based on principle of operation :-

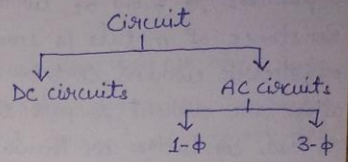


* **Sequential Memories :-** The examples of sequential memory are magnetic tape audio/video. In sequential memories the memory locations are organized in a sequence. These memories further classified into two types.
  i) shift registers ii) charge coupled devices (CCD)

* **Random Access Memory :-** Random access memory is also called as read write memory (RWM). The memory locations in this type of memory are organized in such a way that the access time required for accessin

UNIT-Ist DC circuit Analysis

⇒ **Concept of circuit** :- Circuit is an electrical configration in which various components such as resistors, capacitors, inductors etc. and voltage or current sources are electrically connected to each other.



* **Open circuit** :- Two points in a CKT are said to be open CKTed if there is no CKT element or direct connection b/w them. An open CKT exists b/w points A and B as shown in fig. (a) below.

* **Short circuit** :- Two points in a circuit are said to be short CKTed when they are connected to each other by a wire. Point A and B are short CKTed is shown in fig. (b) below.

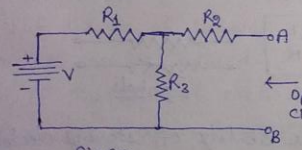


Fig (a)

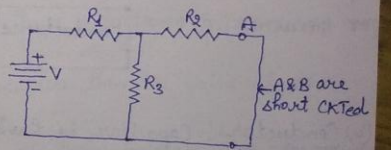
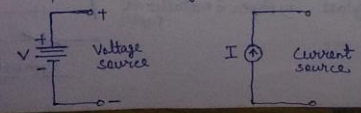


Fig (b)

⇒ **Active and Passive Elements** :- The CKT elements can be divided into two categories

i) Active elements ii) Passive Elements.

i) **Active elements** :- An independent voltage or current source which can absorb or deliver energy are known as the active elements. The independent voltage or current source is shown in fig. (c) below



Examples:-

ii) Passive Elements :- The CKT elements which can not generate energy are known as the passive elements. The ex. of passive elements are

a) Resistors b) Capacitors c) Inductors.

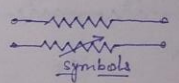
a) Resistance (R) :- Resistance of a material is defined as the opposition to flow of current. It is measured in ~~ohms~~ ohms (Ω) Resistance of metals is small that means they are good conductors of electric current. Certain materials like plastic, wood, glass do not allow the current to pass through them easily, hence they are called as bad conductors or insulators.

The mathematical expression, $\boxed{R = \rho \frac{l}{a}}$

Here,
ρ = Resistivity of a material
l = length of the conductor
a = Cross-sectional area.

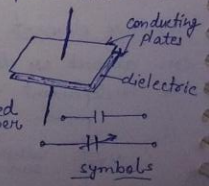
*Resistivity (ρ) :- The resistivity is also called as specific resistance and it is denoted by ρ (rho). Its unit is Ω-m.

* Conductance (G) :- The conductance is defined as the reciprocal of resistance. It is denoted by G and the unit of G is Ω^{-1} or siemens. i.e $\boxed{\text{Conductance } (G) = \frac{1}{R}}$



(b) Conductors :- Capacitor is having the ability of storing electrical energy and this ability of storing a charge is known as capacitance. All capacitors consists of two parallel conducting plates separated by an insulating material known as dielectric. Capacitance may be defined as the amount of charge required to develop a 1 volt potential difference b/w its plates and the unit of this capacitance is Farad. i.e

$\Rightarrow \boxed{C = \frac{Q}{V}}$



* Dielectric Materials :- i) Mica, glass ii) Paper & metallized paper,
iii) High permittivity ceramic etc

*Resistance of a capacitor:- A capacitor opposes to the flow of current in the ac CKTs. It is called as reactance. The reactance of a capacitor is denoted by X_c and it is expressed as,

$$X_c = \frac{1}{2\pi fc}$$

Its value decreasing with increasing in freq.

c) Inductare:- An inductor is a coil that oppose any charge in current. Inductor or coils probably vary more in design than any other component. When the current or magnetic field changes them e.m.f is induced in the CKT. This induced emf is directly proportional to

Fig. below shows the basic construction of an inductor coil in which a copper wire is ~~wor~~ wound on IRON core.

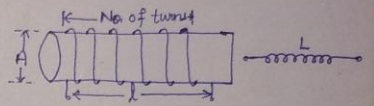


Fig. Inductar

Some important factors at which inductance of coil depends.

i) If the no. of turns increases then inductance increases bcz more voltage can be induced.

ii) If the area of each turn increases then the value of the inductance again increases.

iii) If the length of the same no. of turns increases then the value of inductance decreases.

Q1. What are the main elements of the communication system. Explain and why modulation is required for transmission of the signals.

Q2. A multiple-tone modulating signal $f(t)$, consisting of two frequency components is given by $f(t) = E_1 \cos\omega_1 t + E_2 \cos\omega_2 t$ where $\omega_2 > \omega_1$ and $E_1 > E_2$ the signal $f(t)$ modulates a carrier $e_c = E_c \cos\omega_c t$

i) ~~Draw the~~ Drive the expression for AM wave

ii) Draw a single sided spectrum and find the bandwidth of AM wave.

Q3. An AM wave is generated by ~~the~~ modulating the carrier of frequency $f_c = 800\text{KHz}$ by the message signal $m(t) = \sin 2000\pi t + 5\cos 4000\pi t$. The AM wave is $s(t) = 100[1 + m(t)] \cdot \cos 2\pi f_c t$.

i) Determine the spectrum of the AM signal.

ii) Determine the average power in the carrier and in the side bands.

Q4. Explain the working of a modulator ~~for~~ generating FM (varactor diode method). ~~AM. Drive mathematical expression also~~ using suitable diagram.

Q5. What do you mean by DSB-SC. Prove that the balance modulator produces an output consisting of sidebands only with the carrier removed.

Q6. ~~A 107.6 MHz carrier signal~~ Antenna current of an AM transmitter is 8A if only the carrier is sent, but it increase to 8.93 A if the carrier is modulated by a single sinusoidal wave. Determine the percentage modulation Also find the antenna current if the percentage of modulation changes to 0.8.

Q5 A carrier which attains a peak voltage of 5V has a f^{a} of 100MHz. This carrier is FM by a sinusoidal waveform of f^{a} 2KHz to such an extent that f^{a} deviation from carrier f^{a} is 75KHz. The modulated waveform passes through zero as is increasing at time $t = 0$. ~~What~~ Write the expression for modulated carrier waveform

Q1. Convert the following numbers into binary numbers.

i) $(26.355)_{10}$ ii) $(135.325)_{10}$ iii) $(563.329)_{10}$

Q2. Perform following binary operation using 1's and 2's complement.

i) $(1010)_2 - (0101)_2$ ii) $(1001)_2 - (1101)_2$

Q3. i) Perform $(54)_{10} - (22)_{10}$ in BCD using 10's complement method.

ii) Simplify the expression $Y = \overline{(\bar{A}B + \bar{A} + AB)}$.

Q4. i) Realize the expression using only NOR gates. $Y = (ABC + \bar{B}\bar{C}) \cdot C$

ii) Realize the expression using only NAND gates. $Y = (AB + BC) \cdot C$

Q5. Simplify the following boolean expression using K-Map and verify it using the Quine-McCluskey method.

$F(A,B,C,D) = \Sigma m(1,5,6,12,13,14) + d(2,4)$

Q6.