BHADRAK ENGINEERING SCHOOL & TECHNOLOGY (BEST),
ASURALI, BHADRAK

DIGITAL ELECTRONICS

(Th- 03)

(As per the 2020-21 syllabus of the SCTE&VT, Bhubaneswar, Odisha)

Third Semester

Electronics & Tele-comm. Engg.

Prepared By: *Er K M Jena*

DIGITAL ELECTRONICS

CHAPTER WISE DISTRIBUTION OF PERIODS & MARKS

Sl.No.	Chapter	Topic	Periods as per syllabus	Actually, periods required	Expected marks
01	01	**Basics of Digital Electronics**	12	17	20
02	02	**Combinational Circuits**	12	13	15
03	03	**Sequential Circuits**	12	12	10
04	04	**Registers, Memories & PLD**	08	09	10
05	05	**A/D & D/A converters**	07	06	15
06	06	**Logic Families**	09	07	05
		Total	**60**	**64**	**110**

UNIT 1

1.1. Number System-Binary,Octal,Decimal,Hexa decimal-conversion from one no system to another no system.

1.2. Arithmetic operation-Adition, subtraction, Multiplication, Division,1,s & 2's Complements of binary nos & subtraction using complements methods.

1.3. Digital code & its application & distinguish between weighted & non-weighted code. Binary code,Gray code.

1.4. Logic Gates- AND,OR,NOT,NAND,NOR, Ex-OR,EX-NOR-Symbols,Function, expression, truth Table& timing Diagram.

1.5. Universal Gates & its realization.

1.6. Boolean algebra,Boolean expression,De-Morgan;s Theorem.

1.7. Represent Logic expression:SOP,POS. forms.

1.8. Karnaugh Map(3 & 4 variables) & Minimization of logical expression, don't care conditions.

UNIT 1
BASICS OF DIGITAL ELECTRONICS

INTRODUCTION:-

- The term digital refers to a process that is achieved by using discrete unit.
- In number system there are different symbols and each symbol has an absolute value and also has place value.

RADIX OR BASE:-

The radix or base of a number system is defined as the number of different digits which can occur in each position in the number system.

1.1 NUMBER SYSTEM:-

TYPES OF NUMBER SYSTEM:-

There are four types of number systems. They are

1. Decimal number system
2. Binary number system
3. Octal number system
4. Hexadecimal number system

DECIMAL NUMBER SYSTEM:-

- The decimal number system contain ten unique symbols 0,1,2,3,4,5,6,7,8 and 9.
- In decimal system 10 symbols are involved, so the base or radix is 10.
- It is a positional weighted system.
- The value attached to the symbol depends on its location with respect to the decimal point. In

general,

$d_n \; d_{n-1} \; d_{n-2} \; \ldots\ldots\ldots\ldots \; d_0 \; . \; d_{-1} \; d_{-2} \; \ldots\ldots\ldots\ldots d_{-m}$

is given by

$(d_n \times 10^n) + (d_{n-1} \times 10^{n-1}) + (d_{n-2} \times 10^{n-2}) + \ldots + (d_0 \times 10^0) + (d_{-1} \times 10^{-1}) + (d_{-2} \times 10^{-2}) + \ldots + (d_{-m} \times 10^{-m})$ For

example:-

$9256.26 = 9 \times 1000 + 2 \times 100 + 5 \times 10 + 6 \times 1 + 2 \times (1/10) + 6 \times (1/100)$

$= 9 \times 10^3 + 2 \times 10^2 + 5 \times 10^1 + 6 \times 10^0 + 2 \times 10^{-1} + 6 \times 10^{-2}$

BINARYNUMBER SYSTEM:-

- The binary number system is a positional weighted system.
- The base or radix of this number system is 2.
- It has two independent symbols.
- The symbols used are 0 and 1.
- A binary digit is called a bit.
- The binary point separates the integer and fraction parts.

In general,

$d_n \; d_{n-1} \; d_{n-2} \; \ldots\ldots\ldots\ldots \; d_0 \; . \; d_{-1} \; d_{-2} \; \ldots\ldots\ldots\ldots d_{-k}$

is given by

$(d_n \times 2^n) + (d_{n-1} \times 2^{n-1}) + (d_{n-2} \times 2^{n-2}) + \ldots.+ (d_0 \times 2^0) + (d_{-1} \times 2^{-1}) + (d_{-2} \times 2^{-2}) + \ldots.+(d_{-k} \times 2^{-k})$

OCTAL NUMBER SYSTEM:-

- It is also a positional weighted system.

Its base or radix is 8.

- It has 8 independent symbols 0,1,2,3,4,5,6 and 7.
- Its base $8 = 2^3$, every 3- bit group of binary can be represented by an octal digit.

HEXADECIMALNUMBER SYSTEM:-

- The hexadecimal number system is a positional weighted system.

The base or radix of this number system is 16.

- The symbols used are 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E and F
- The base 16 = 24 , every 4 – bit group of binary can be represented by an hexadecimal digit.

CONVERSIONFROM ONENUMBER SYSTEM TOANOTHER :-

1. BINARYNUMBERSYSTEM:-

(a) Binary todecimal conversion:-

In this method, each binary digit of the number is multiplied by its positional weight and the product terms are added to obtain decimal number.

For example:

(i) Convert $(10101)_2$ to decimal.

Solution :

(Positional weight) $2^4 \; 2^3 \; 2^2 \; 2^1 \; 2^0$

Binary number 10101

$= (1 \times 2^4) + (0 \times 2^3) + (1x 2^2) + (0 \times 2^1) + (1 \times 2^0)$

$= 16 + 0+ 4+ 0+ 1$

$= (21)_{10}$

(ii) Convert $(111.101)_2$ to decimal.

Solution:

$(111.101)_2 = (1 \times 2^2) + (1 \times 2^1) + (1x 2^0) + (1 \times 2^{-1}) + (0 \times 2^{-2}) + (1 \times 2^{-3})$

$= 4+ 2+ 1 + 0.5 + 0 + 0.125$

$= (7.625)_{10}$

(b)Binary to Octal Conversion:-

For conversion binary to octal the binary numbers are divided into groups of 3 bits each, starting at the binary point and proceeding towards left and right.

Octal	Binary	Octal	Binary
0	000	4	100
1	001	5	101
2	010	6	110
3	011	7	111

For example:

(i) Convert $(101111010110.110110011)_2$ into octal.

Solution :

Group of 3 bits are	101	111	010	110	.	110	110	011
Convert each group into octal =	5	7	2	6	.	6	6	3

The result is $(5726.663)_8$

(ii) Convert $(10101111001.0111)_2$ into octal.

Solution :

Binary number	10	101	111	001	.	011	1
Group of 3 bits are =	010	101	111	001	.	011	100
Convert each group into octal =	2	5	7	1	.	3	4

The result is $(2571.34)_8$

(c) Binary to Hexadecimal conversion:-

For conversion binary to hexadecimal number the binary numbers starting from the binary point, groups are made of 4 bits each, on either side of the binary point.

Hexadecimal	Binary	Hexadecimal	Binary
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

For example:
(i) Convert $(1011011011)_2$ into hexadecimal.

Solution:

Given Binary number 10 1101 1011

Group of 4 bits are 0010 1101 1011

Convert each group into hex = 2 D B The result is $(2DB)_{16}$

(ii) Convert $(01011111011.011111)_2$ into hexadecimal.

Solution:

Given Binary number 010 1111 1011 . 0111 11

Group of 3 bits are = 0010 1111 1011 . 0111 1100

Convert each group into octal = 2 F B . 7 C The result is $(2FB.7C)_{16}$

2. DECIMAL NUMBER SYSTEM:-

(a)Decimal to binary Conversion:-

In the conversion the integer number are converted to the desired base using successive division by the base or radix.
For example:
(i) Convert $(52)_{10}$ into binary.

Solution:

Divide the given decimal number successively by 2 read the integer part remainder upwards to get equivalent binary number. Multiply the fraction part by 2. Keep the integer in the product as it is and multiply the new fraction in the product by 2. The process is continued and the integer are read in the products from top to bottom.

2 | 52
2 | 26 — 0
2 | 13 — 0
2 | 6 — 1
2 | 3 — 0
2 | 1 — 1
0 — 1

Result of $(52)_{10}$ is $(110100)_2$

(ii) Convert $(105.15)_{10}$ into binary.

Solution:

Integer part		Fraction part
2 I105		0.15 x 2 = 0.30
2 l52	— 1	0.30 x 2 = 0.60
2 l26	— 0	0.60 x 2 = 1.20
2 l13	— 0	0.20 x 2 = 0.40
2 l6	— 1	0.40 x 2 = 0.80
2 l3	— 0	0.80 x 2 = 1.60
2 l1	— 1	
0	— 1	

Result of $(105.15)_{10}$ is $(1101001.001001)_2$

(b)Decimal tooctalconversion:-

To convert the given decimal integer number to octal, successively divide the given number by 8 till the quotient is 0. To convert the given decimal fractions to octal successively multiply the decimal fraction and the subsequent decimal fractions by 8 till the product is 0 or till the required accuracy is obtained.

For example:
(i) Convert $(378.93)_{10}$ into octal.

Solution:

8 I378		0.93 x 8 = 7.44
8 l47	— 2	0.44 x 8 = 3.52
8 l5	— 7	0.52 x 8 = 4.16
0	— 5	0.16 x 8 = 1.28

Result of $(378.93)_{10}$ is $(572.7341)_8$

(c)Decimal to hexadecimal Conversion:-

The decimal to hexadecimal conversion is same as octal.

For example:
(i) Convert $(2598.675)_{10}$ into hexadecimal.

Solution:

	Remainder			
	Decimal	Hex		Hex
16 I2598			0.675 x 16 = 10.8	A
16 l162	— 6	6	0.800 x 16 = 12.8	C
16 l10	— 2	2	0.800 x 16 = 12.8	C
0	— 10	A	0.800 x 16 = 12.8	C

Result of $(2598.675)_{10}$ is $(A26.ACCC)_{16}$

3. OCTAL NUMBERSYSTEM:-

(a)Octal tobinary conversion:-

To convert a given a octal number to binary, replace each octal digit by its 3- bit binary equivalent.

For example:

Convert $(367.52)_8$ into binary.

Solution:

Given Octal number is 3 6 7 . 5 2

Convert each group octal to binary = 011 110 111 . 101 010

Result of $(367.52)_8$ is $(011110111.101010)_2$

(b)Octal todecimal conversion:-

For conversion octal to decimal number, multiply each digit in the octal number by the weight of its position and add all the product terms

For example: -

Convert $(4057.06)_8$ to decimal

Solution:

$(4057.06)_8 = 4 \times 8^3 + 0 \times 8^2 + 5 \times 8^1 + 7 \times 8^0 + 0 \times 8^{-1} + 6 \times 8^{-2}$

$= 2048 + 0 + 40 + 7 + 0 + 0.0937$

$= (2095.0937)_{10}$

Result is $(2095.0937)_{10}$

(c) Octal tohexadecimal conversion:-

For conversion of octal to Hexadecimal, first convert the given octal number to binary and then binary number to hexadecimal.

For example :-

Convert $(756.603)_8$ to hexadecimal.

Solution :-

Given octal no.		7	5	6	.	6	0	3
Convert each octal digit to binary	=	111	101	110	.	110	000	011
Group of 4bits are	=	0001	1110	1110	.	1100	0001	1000
Convert 4 bits group to hex.	=	1	E	E	.	C	1	8

Result is $(1EE.C18)_{16}$

(4)HEXADECIMALNUMBERSYSTEM :-

(a)Hexadecimaltobinary conversion:-

For conversion of hexadecimal to binary, replace hexadecimal digit by its 4 bit binary group. For example:

Convert $(3A9E.B0D)_{16}$ into binary.

Solution:

Given Hexadecimal number is 3 A 9 E . B 0 D

Convert each hexadecimal digit to 4 bit binary = 0011 1010 1001 1110 . 1011 0000 1101

Result of $(3A9E.B0D)_8$ is $(0011101010011110.101100001101)_2$

(b)Hexadecimaltodecimal conversion:-

For conversion of hexadecimal to decimal, multiply each digit in the hexadecimal number by its position weight and add all those product terms.

For example: -
Convert $(A0F9.0EB)_{16}$ to decimal
Solution:

$$(A0F9.0EB)_{16} = (10 \times 16^{3}) + (0 \times 16^{2}) + (15 \times 16^{1}) + (9 \times 16^{0}) + (0 \times 16^{-1}) + (14 \times 16^{-2}) + (11 \times 16^{-3})$$

$$= 40960 + 0 + 240 + 9 + 0 + 0.0546 + 0.0026$$
$$= (41209.0572)_{10}$$

Result is $(41209.0572)_{10}$

(c) Hexadecimal toOctal conversion:-

For conversion of hexadecimal to octal, first convert the given hexadecimal number to binary and then binary number to octal.

For example :-
Convert $(B9F.AE)_{16}$ to octal.

Solution :-

Given hexadecimal no.is		B		9		F	.	A		E
Convert each hex. digit to binary	=	1011		1001		1111	.	1010		1110
Group of 3 bits are	=	101	110	011	111		.	101	011	100
Convert 3 bits group to octal.	=	5	6	3	7		.	5	3	4

Result is $(5637.534)_{8}$

1.2 ARITHEMATIC OPERATION :-

1. BINARY ADDITION:-

The binary addition rules are as follows
0 + 0 = 0 ; 0 + 1 = 1 ; 1 + 0 = 1 ; 1 + 1 = 10 , i.e 0 with a carry of 1

For example :-
Add $(100101)_2$ and $(1101111)_2$.
Solution :-

```
    1 0 0 1 0 1
+   1101111
  ----------
    10010100
```

Result is $(10010100)_2$

2. BINARYSUBTRACTION:-

The binary subtraction rules are as follows
0 - 0 = 0 ; 1 - 1 = 0 ; 1 - 0 = 1 ; 0 - 1 = 1 , with a borrow of 1

For example :-

Substract $(111.111)_2$ from $(1010.01)_2$.

Solution :-

```
    1 0 1 0 . 0 1 0
-     111 . 111
    ---------------
      0010 .011
```

Result is $(0010.011)_2$

3. BINARY MULTIPLICATION:-

The binary multiplication rules are as follows

0 x 0 = 0 ; 1 x 1 = 1 ; 1 x 0 = 0 ; 0 x 1 = 0

For example :-

Multiply $(1101)_2$ by $(110)_2$.

Solution :-

```
      1 1 0 1
  x     1 1 0
  -------------
      0 0 0 0
    1 1 0 1
+ 1 1 0 1
  -------------
  1 0 0 1 1 1 0
```

Result is $(1001110)_2$

4. BINARY DIVISION:-

The binary division is very simple and similar to decimal number system. The division by '0' is meaningless. So we have only 2 rules

0 ÷ 1 = 0

1 ÷ 1 = 1

For example :-

Divide $(10110)_2$ by $(110)_2$.

Solution :-

```
110 ) 101101 ( 111.1
  - 110
    ----
    1010
     110
    ----
    1001
     110
    ----
     110
     110
    ----
     000
    ----
```

Result is $(111.1)_2$

1's COMPLEMENT REPRESENTATION :-

The 1's complement of a binary number is obtained by changing each 0 to 1 and each 1 to 0.

For example :-

Find $(1100)_2$ 1's complement.

Solution :-

Given 1 1 0 0

1's complement is 0 0 1 1

Result is $(0011)_2$

2's COMPLEMENT REPRESENTATION :-

The 2's complement of a binary number is a binary number which is obtained by adding 1 to the 1's complement of a number i.e.

2's complement = 1's complement + 1

For example :-

Find $(1010)_2$ 2's complement.

Solution :-

Given 1 0 1 0

1's complement is 0 1 0 1

\+ 1

2's complement 0 1 1 0

Result is $(0110)_2$

SUBSTRACTION USING COMPLEMENT METHOD :-

1's COMPLEMENT:-

In 1's complement subtraction, add the 1's complement of subtrahend to the minuend. If there is a carry out, then the carry is added to the LSB. This is called end around carry. If the MSB is 0, the result is positive. If the MSB is 1, the result is negative and is in its 1's complement form. Then take its 1's complement to get the magnitude in binary.

Subtract $(10000)_2$ from $(11010)_2$ using 1's complement.

Solution:-

1 1 0 1 0 = 26

\- 1 0 0 0 0 => + 01111 (1's complement) = - 16

Carry → 1 0 1 0 0 1 + 10

\+ 1

01010 = +10

2'sCOMPLEMENT:-

In 2's complement subtraction, add the 2's complement of subtrahend to the minuend. If there is a carry out, ignore it. If the MSB is 0, the result is positive. If the MSB is 1, the result is negative and is in its 2's complement form. Then take its 2's complement to get the magnitude in binary.

For example:-

Subtract $(1010100)_2$ from $(1010100)_2$ using 2's complement.

Solution:-

1 0 1 0 1 0 0 = 84

\- 1 0 1 0 1 0 0 => + 0101100 (2's complement) = - 84

1 0 0 0 0 0 0 0 (Ignore the carry) 0

= 0 (result = 0

Hence MSB is 0. The answer is positive. So it is +0000000 =0

1.3 DIGITAL CODES:-

In practice the digital electronics requires to handle data which may be numeric, alphabets and special characters. This requires the conversion of the incoming data into binary format before it can be processed. There is various possible ways of doing this and this process is called encoding. To achieve the reverse of it, we use decoders.

WEIGHTEDANDNON-WEIGHTEDCODES:-

There are two types of binary codes

1) Weighted binary codes
2) Non- weighted binary codes

In weighted codes, for each position (or bit) ,there is specific weight attached.
For example, in binary number, each bit is assigned particular weight 2n where 'n' is the bit number for n = 0,1,2,3,4 the weights are 1,2,4,8,16 respectively.
Example :- BCD

Non-weighted codes are codes which are not assigned with any weight to each digit position, i.e., each digit position within the number is not assigned fixed value.
Example:- Excess – 3 (XS -3) code and Gray codes

EXCESS THREE(XS-3) CODE:-

The Excess-3 code, also called XS-3, is a non- weighted BCD code. This derives it name from the fact that each binary code word is the corresponding 8421 code word plus 0011(3). It is a sequential code. It is a self complementing code.

GRAYCODE:-

The gray code is a non-weighted code. It is not a BCD code. It is cyclic code because successive words in this differ in one bit position only i.e it is a unit distance code.

Gray code is used in instrumentation and data acquisition systems where linear or angular displacement is measured. They are also used in shaft encoders, I/O devices, A/D converters and other peripheral equipment.

BINARY-TO–GRAYCONVERSION:-

If an n-bit binary number is represented by B_n B_{n-1} ----- B_1 and its gray code equivalent by G_n G_{n-1} ----- G_1, where B_n and G_n are the MSBs , then gray code bits are obtained from the binary code as follows

$G_n = B_n$

$G_{n-1} = B_n \ominus B_{n-1}$

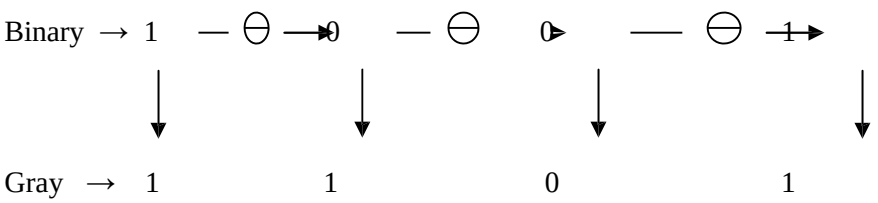
.

$G_1 = B_2 \ominus B_1$

Where the symbol $\ominus$ stands for Exclusive OR (X-OR)
For example :-

Convert the binary 1001 to the Gray code.

Solution :-`



The gray code is 1101

GRAY-TO-BINARY CONVERSION:-

If an n-bit gray number is represented by G_n G_{n-1} ------- G_1 and its binary equivalent by B_n B_{n-1} - - - - - B_1, then binary bits are obtained from Gray bits as follows :

$B_n = G_n$

$B_{n-1} = B_n \ominus G_{n-1}$

.

$B_1 = B_2 \quad G_1 \ominus$

For example :-

Convert the Gray code 1101 to the binary.

Solution :-

Gray → 1 1 0 1

Binary→ 1 0 0 1

The binary code is 1001

1.4 LOGIC GATES:-

- Logic gates are the fundamental building blocks of digital systems.

There are 3 basic types of gates AND, OR and NOT.

- Logic gates are electronic circuits because they are made up of a number of electronic devices and components.
- Inputs and outputs of logic gates can occur only in 2 levels. These two levels are termed HIGH and LOW, or TRUE and FALSE, or ON and OFF or simply 1 and 0.
- The table which lists all the possible combinations of input variables and the corresponding outputs is called a truth table.

DIFFERENT TYPES OF LOGIC GATES:-

NOTGATE (INVERTER):-

- A NOT gate, also called and inverter, has only one input and one output. It

is a device whose output is always the complement of its input.

- The output of a NOT gate is the logic 1 state when its input is in logic 0 state and the logic 0 state when its inputs is in logic 1 state.

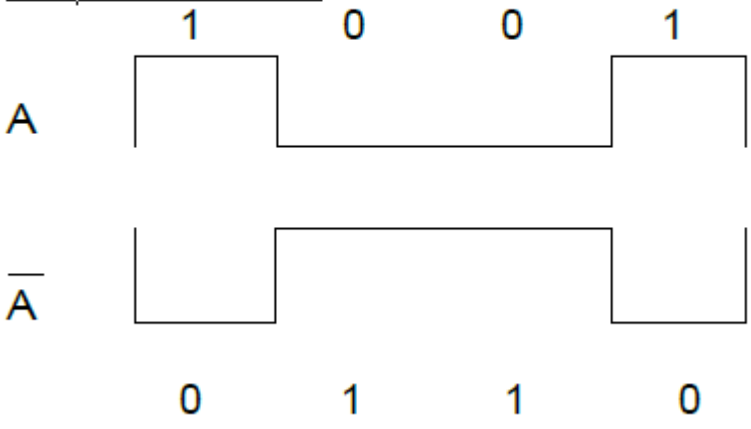
IC No. :- 7404

Logic Symbol

Truth table

INPUT A	OUTPUT $\overline{A}$
0	1
1	0

Timing Diagram



ANDGATE:-

- An AND gate has two or more inputs but only one output.
- The output is logic 1 state only when each one of its inputs is at logic 1 state.

The output is logic 0 state even if one of its inputs is at logic 0 state.

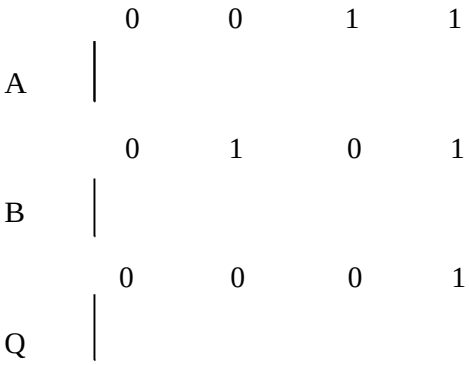
IC No.:- 7408

Logic Symbol

Truth Table

		OUTPUT
A	B	Q=A . B
0	0	0
0	1	0
1	0	0
1	1	1

Timing Diagram



ORGATE:-

- An OR gate may have two or more inputs but only one output.
- The output is logic 1 state, even if one of its input is in logic 1 state.
- The output is logic 0 state, only when each one of its inputs is in logic state. IC

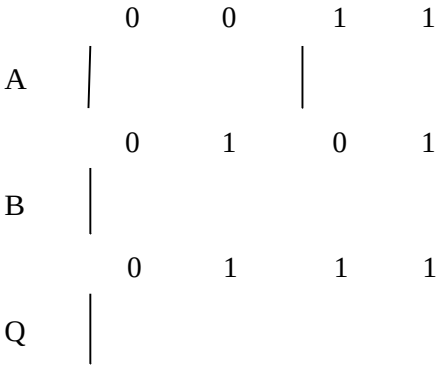
No.:- 7432

Logic Symbol

Truth Table

INPUT		OUTPUT
A	B	Q=A + B
0	0	0
0	1	1
1	0	1
1	1	1

Timing Diagram



NANDGATE:-

- NAND gate is a combination of an AND gate and a NOT gate.
- The output is logic 0 when each of the input is logic 1 and for any other combination of inputs, the output is logic 1.

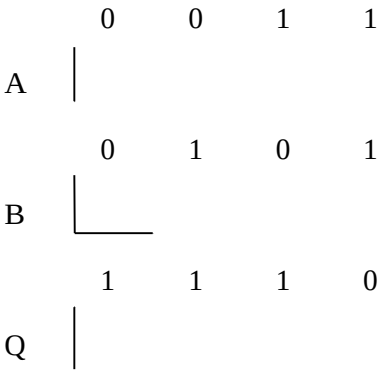
IC No.:- 7400 two input NAND gate
7410 three input NAND gate
7420 four input NAND gate
7430 eight input NAND gate

Logic Symbol

Truth Table

INPUT		OUTPUT
A	B	Q= A . B
0	0	1
0	1	1
1	0	1
1	1	0

Timing Diagram



NORGATE:-

- NOR gate is a combination of an OR gate and a NOT gate.
- The output is logic 1, only when each one of its input is logic 0 and for any other combination of inputs, the output is a logic 0 level.

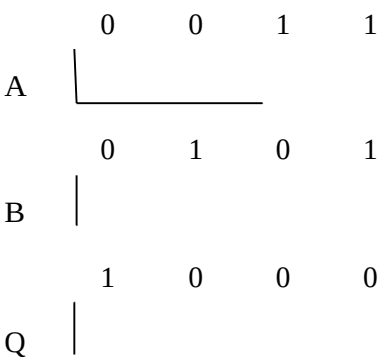
IC No.:- 7402 two input NOR gate
7427 three input NOR gate
7425 four input NOR gate

Logic Symbol

Truth Table

INPUT		OUTPUT
A	B	Q= $\overline{A + B}$
0	0	1
0	1	0
1	0	0
1	1	0

Timing Diagram



EXCLUSIVE – OR (X-OR) GATE:-

- An X-OR gate is a two input, one output logic circuit.
- The output is logic 1 when one and only one of its two inputs is logic 1. When both the inputs is logic 0 or when both the inputs is logic 1, the output is logic 0.

IC No.:- 7486

Logic Symbol

Truth Table

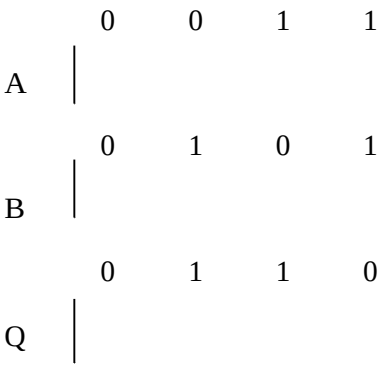
INPUT		OUTPUT Q
A	B	= A ⊖ B
0	0	0
0	1	1
1	0	1
1	1	0

INPUTS are A and B

OUTPUT is Q = A ⊖ B

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

Timing Diagram



EXCLUSIVE – NOR (X-NOR) GATE:-

- An X-NOR gate is the combination of an X-OR gate and a NOT gate.

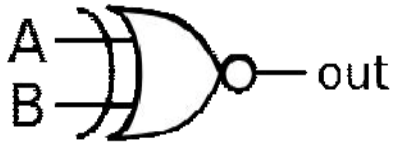
An X-NOR gate is a two input, one output logic circuit.

- The output is logic 1 only when both the inputs are logic 0 or when both the inputs is 1.

The output is logic 0 when one of the inputs is logic 0 and other is 1.

IC No.:- 74266

Logic Symbol

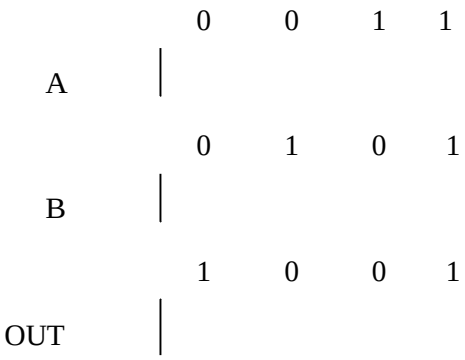


INPUT		OUTPUT
A	B	OUT =A XNOR B
0	0	1
0	1	0
1	0	0
1	1	1

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

= A XNOR B

Timing Diagram



1.5 UNIVERSAL GATES & ITS REALISATION:-

There are 3 basic gates AND, OR and NOT, there are two universal gates NAND and NOR, each of which can realize logic circuits single handedly. The NAND and NOR gates are called universal building blocks. Both NAND and NOR gates can perform all logic functions i.e. AND, OR, NOT, EXOR and EXNOR.

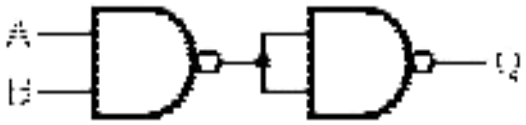
NANDGATE:-

a) Inverter from NAND gate

Input = A
Output Q = $\overline{A}$

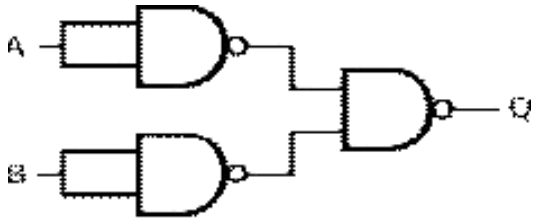
b) AND gate from NAND gate

Input s are A and B
Output Q = A.B



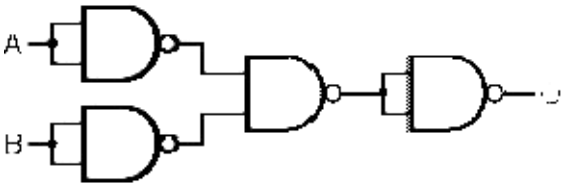
c) OR gate from NAND gate

Inputs are A and B
Output Q = A+B



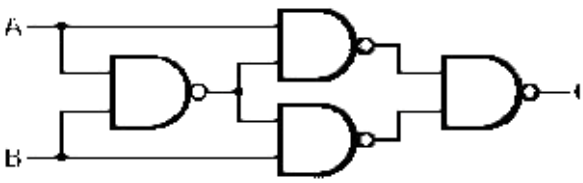
d) NOR gate from NAND gate

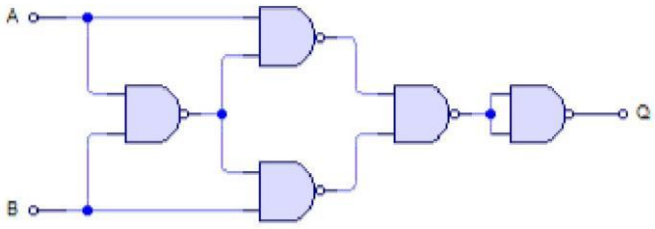
Inputs are A and B
Output Q = $\overline{A+B}$



e) EX-OR gate from NAND gate

Inputs are A and B
Output Q = $A\overline{B} + \overline{A}B$





NORGATE:-

a) Inverter from NOR gate

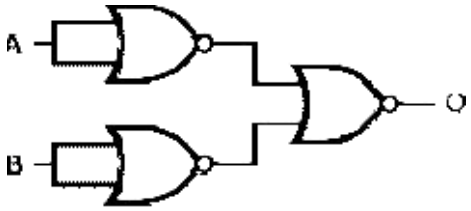
Input = $\overline{A}$

Output Q = $\overline{A}$

b) AND gate from NOR gate

Input s are A and B

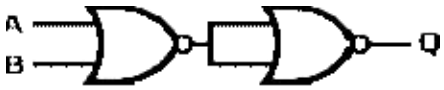
Output Q = A.B



c) OR gate from NOR gate

Inputs are A and B

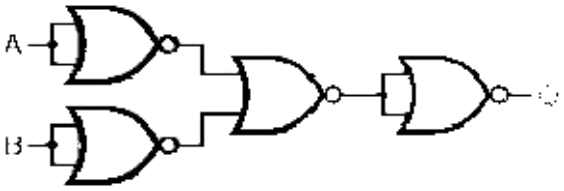
Output Q = A+B

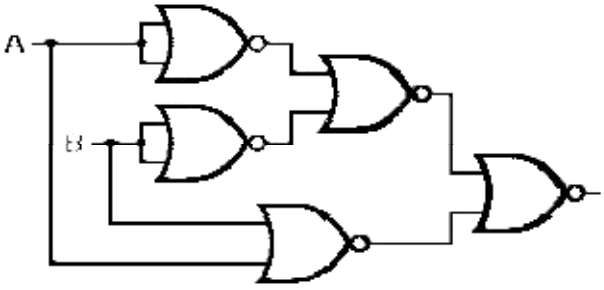


d) NAND gate from NOR gate

Inputs are A and B

Output Q = $\overline{A.B}$





f) EX-NOR gate From NOR gate

Inputs are A and B Output Q = A B + A B

1.6 BOOLEAN ALGEBRA. BOOLEAN EXPRESSION, DEMORGAN'S THEOREMS

INTRODUCTION:-

- Switching circuits are also called logic circuits, gates circuits and digital circuits.

Switching algebra is also called Boolean algebra.

- Boolean algebra is a system of mathematical logic. It is an algebraic system consisting of the set of elements (0,1), two binary operators called OR and AND and unary operator called NOT. It is

the basic mathematical tool in the analysis and synthesis of switching circuits.

- It is a way to express logic functions algebraically.
- Any complex logic can be expressed by a Boolean function.
- The Boolean algebra is governed by certain well developed rules and laws.

AXIOMS ANDLAWSOFBOOLEAN ALGEBRA:-

Axioms or postulates of Boolean algebra are set of logical expressions that are accepted without proof and upon which we can build a set of useful theorems. Actually, axioms are nothing more than the definitions of the three basic logic operations AND, OR and INVERTER. Each axiom can be interpreted as the outcome of an operation performed by a logic gate.

AND operation	OR operation	NOT operation
Axiom 1: 0 . 0 = 0	Axiom 5: 0 + 0 = 0	Axiom 9: $\bar{1}$ = 0
Axiom 2: 0 . 1 = 0	Axiom 6: 0 + 1 = 1	Axiom 10:0 = 1
Axiom 3: 1 . 0 = 0	Axiom 7: 1 + 0 = 1	
Axiom 2: 1 . 1 = 1	Axiom 8: 1 + 1 = 1	

1. Complementation Laws:-

The term complement simply means to invert, i.e. to changes 0s to 1s and 1s to 0s. The five laws of complementation are as follows:

Law 1: $\bar{0} = 1$

Law 2: $\bar{1} = 0$

Law 3: if A = 0, then $\bar{A} = 1$

Law 4: if $\bar{A} = 1$,then$A = 0$

Law 5: $\bar{\bar{A}} = 0$ (double complementation law)

2. OR Laws:-

The four OR laws are as follows Law

1: A + 0 = A(Null law) Law 2:

A + 1 = 1(Identity law) Law 3: A

+ A = A

Law 4: A +$\bar{A} = 1$

3. AND Laws:-

The four AND laws are as follows Law

1: A . 0 = 0(Null law) Law 2:

A . 1 = 1(Identity law) Law 3:

A . A = A

Law 4: A .$\bar{A} = 0$

4. Commutative Laws:-

Commutative laws allow change in position of AND or OR variables. There are two commutative laws.

Law 1: A + B = B + A

Proof

A	B	A + B
0	0	0
0	1	1
1	0	1
1	1	1

=

B	A	B+ A
0	0	0
0	1	1
1	0	1
1	1	1

Law 2: A . B = B . A

Proof

A	B	A . B
0	0	0
0	1	0
1	0	0
1	1	1

=

B	A	B. A
0	0	0
0	1	0
1	0	0
1	1	1

This law can be extended to any number of variables. For example
A.B. C = B. C. A = C. A. B = B. A. C

5. Associative Laws:-

The associative laws allow grouping of variables. There are 2 associative laws.

Law 1: (A + B) + C = A + (B + C)

Proof

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

=

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

Law 2: (A .B) C = A (B .C)

Proof

A	B	C	AB	(AB)C
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	0
1	0	0	0	0
1	0	1	0	0
1	1	0	1	0
1	1	1	1	1

=

A	B	C	B.C	A(B.C)
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	1	0
1	0	0	0	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

This law can be extended to any number of variables. For example

A(BCD) = (ABC)D = (AB) (CD)

6. Distributive Laws:-

The distributive laws allow factoring or multiplying out of expressions. There are two distributive laws.

Law 1: A (B + C) = AB + AC

Proof

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

=

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

Law 2: A + BC = (A+B) (A+C)

Proof RHS = (A+B) (A+C)

= AA + AC + BA + BC

= A + AC + AB + BC

= A (1+ C + B) + BC

= A. 1 + BC (1 +C + B = 1 + B = 1)

= A + BC

= LHS

7. Redundant Literal_Rule (RLR):-

Law 1: A + $\bar{A}$B = A + B

Proof

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

$= 1.(A + B)$

$= A + B$

Law 2: $A(\bar{A} + B) = AB$

Proof

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

$= 0 + AB$

$= AB$

8. Idempotence Laws:-

Idempotence means same value.

Law 1: $A.A = A$

Proof

If $A = 0$, then $A.A = 0.0 = 0 = A$ If

$A = 1$, then $A.A = 1.1 = 1 = A$

This law states that AND of a variable with itself is equal to that variable only.

Law 2: $A + A = A$

Proof

If $A = 0$, then $A + A = 0 + 0 = 0 = A$

If $A = 1$, then $A + A = 1 + 1 = 1 = A$

This law states that OR of a variable with itself is equal to that variable only.

9. Absorption Laws:-

There are two laws:

Law 1: $A + A \cdot B = A$

Proof

$A + A \cdot B = A(1 + B) = A \cdot 1 = A$

A	B	AB	A+AB
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

Law 2: $A(A + B) = A$

Proof

$A(A + B) = A \cdot A + A \cdot B = A + AB = A(1 + B) = A \cdot 1 = A$

A	B	A+B	A(A+B)
0	0	0	0
0	1	1	0
1	0	1	1
1	1	1	1

10. Consensus Theorem (Included Factor Theorem):-

Theorem 1:

$AB + \bar{A}C + BC = AB + A\bar{C}$

Proof

$LHS = AB + \bar{A}C + BC$

$= AB + AC + BC\,(A+A)$

$= AB + \bar{A}C + BCA + BCA$

$= AB\,(1 + C) + \bar{A}C\,(1+ B)$

$= AB\,(1) + \bar{A}C\,(1)$

$= AB + \bar{A}C$

$= RHS$

Theorem 2:

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

Proof

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

$= (A\bar{A} + AC + B\bar{A} + BC)\,(B + C)$

$= (AC + BC + A\bar{B})\,(B + C)$

$= ABC + BC + A\bar{B} + AC + BC + ABC$

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

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

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

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

$= LHS$

11. Transposition Theorem:-

Theorem:

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

Proof

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

$= A\bar{A} + C\bar{A} + AB + CB$

$= 0 + \bar{A}C + AB + BC$

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

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

$= AB + \bar{A}C$

$= LHS$

12. De Morgan's Theorem:-

De Morgan's theorem represents two laws in Boolean algebra. Law

1: $\overline{A + B} = \bar{A} \cdot \bar{B}$

Proof

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

=

A	B	A	$\bar{B}$	$A\,\bar{B}$
0	0	1	1	1
0	1	1	0	0
1	0	0	1	0
1	1	0	0	0

This law states that the complement of a sum of variables is equal to the product of their individual complements.

Law 2: $\overline{A \cdot B} = \overline{A} + \overline{B}$

Proof

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

=

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

This law states that the complement of a product of variables is equal to the sum of their individual complements.

DUALITY:-

The implication of the duality concept is that once a theorem or statement is proved, the dual also thus stand proved. This is called the principle of duality.

$[f(A, B, C, \ldots.., 0, 1, +, \cdot)]_d = f(A, B, C, \ldots., 1, 0, \cdot, +)$

Relations between complement and dual

$f_c(A, B, C, \ldots..) = \overline{f(A, B, C, \ldots..)} = f_d(A, B, C, \ldots)$

$f_d(A, B, C, \ldots..) = \overline{f(\overline{A}, \overline{B}, \overline{C}, \ldots)} = f_c(\overline{A}, \overline{B}, \overline{C}, \ldots..)$

The first relation states that the complement of a function f(A, B, C, …) can be obtained by complementing all the variables in the dual function $f_d(A, B, C, \ldots..)$.

The second relation states that the dual can be obtained by complementing all the literals in $\overline{f(A, B, C, \ldots.)}$.

DUALS:-

	Given expression	Dual
1.	$\overline{0} = 1$	$1 = 0$
2.	$0 \cdot 1 = 0$	$1 + 0 = 1$
3.	$0 \cdot 0 = 0$	$1 + 1 = 1$
4.	$1 \cdot 1 = 1$	$0 + 0 = 0$
5.	$A \cdot 0 = 0$	$A + 1 = 1$
6.	$A \cdot 1 = A$	$A + 0 = A$
7.	$A \cdot A = A$	$A + A = A$
8.	$A \cdot A = 0$	$A + A = 1$
9.	$A \cdot B = B \cdot A$	$A + B = B + A$
10.	$A \cdot (B \cdot C) = (A \cdot B) \cdot C$	$A + (B + C) = (A + B) + C$
11.	$A \cdot (B + C) = AB + AC$	$A + BC = (A + B)(A + C)$
12.	$A(A + B) = A$	$A + AB = A$
13.	$\overline{A} \cdot (A \cdot B) = A \cdot B$	$\overline{A + A} + \overline{B} = A + B$
14.	$AB = A + B$	$A + B = \overline{A}\,\overline{B}$
15.	$(A + B)(A + C)(B + C) = (A + B)(A + C)$	$AB + AC + BC = AB + AC$
16.	$A + BC = (A + B)(A + C)$	$A(B + C) = AB + AC$
17.	$(A+C)(A+B) = AB+AC$	$AC+AB=(A+B)(A+C)$
18.	$(A+B)(C+D) = AC + AD + BC + BD$	$(AB+CD) = (A+C)(A+D)(B+C)(B+D)$
19.	$A+B=AB+$ $AB + AB$	$AB=(A+B)(A+B)$ $(A+B)$
20.	$\overline{AB} + A + AB = 0$	$\overline{A + B} \cdot A \cdot (A + B) = 1$

1.7 REPRESENT LOGIC EXPRESSION: SUM-OF-PRODUCTS & PRODUCT-OF-SUM FORM:-

- This is also called disjunctive Canonical Form (DCF) or Expanded Sum of Products Form or Canonical Sum of Products Form.
- In this form, the function is the sum of a number of products terms where each product term contains all variables of the function either in complemented or uncomplemented form.
- This can also be derived from the truth table by finding the sum of all the terms that corresponds to those combinations for which 'f ' assumes the value 1.

  For example

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

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

  $= \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}C + ABC$
- The product term which contains all the variables of the functions either in complemented or uncomplemented form is called a minterm.
- The minterm is denoted as mo, m1, m2 … .
- An 'n' variable function can have 2n minterms.
- Another way of representing the function in canonical SOP form is the showing the sum of minterms for which the function equals to 1.

  For example

  $f(A, B, C) = m_1 + m_2 + m_3 + m_5$

  or

  $f(A, B, C) = \sum m(1, 2, 3, 5)$

  where $\sum m$ represents the sum of all the minterms whose decimal codes are given the parenthesis.

PRODUCT- OF-SUMSFORM:-

- This form is also called as Conjunctive Canonical Form (CCF) or Expanded Product - of – Sums Form or Canonical Product Of Sums Form.
- This is by considering the combinations for which f = 0
- Each term is a sum of all the variables.
- The function $f(A, B, C) = (A + B + C \cdot C) \cdot (A + B + C \cdot C)$

  $= (\bar{A} + B + C)(A + \bar{B} + C)(A + B + C)(A + B + \bar{C})$
- The sum term which contains each of the 'n' variables in either complemented or uncomplemented form is called a maxterm.
- Maxterm is represented as $M_0, M_1, M_2, \ldots\ldots$

  Thus CCF of 'f' may be written as

  $f(A, B, C) = M_0 \cdot M_4 \cdot M_6 \cdot M_7$

  or

  $f(A, B, C) = (0, 4, 6, 7)$

  Where represented the product of all maxterms.

CONVERSIONBETWEEN CANONICAL FORM:-

The complement of a function expressed as the sum of minterms equals the sum of minterms missing from the original function.

Example:-

$f(A, B, C) = \sum m(0,2,4,6,7)$

This has a complement that can be expressed as

$\overline{f(A, B, C)} = \sum m(1, 3, 5) = m_1 + m_3 + m_5$

If we complement f by De- Morgan's theorem we obtain 'f' in a form.

$f = \overline{(m_1 + m_3 + m_5)} = \bar{m}_1 \cdot \bar{m}_3 \cdot \bar{m}_5$

$= M_1 M_3 M_5 = \prod M(1, 3, 5)$

Example:-

Expand $A (A + B) (A + B + C)$ to maxterms and minterms. Solution:-

In POS form

$A(\overline{A} + B)(A + B + \overline{C})$

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

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

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

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

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

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

Therefore

$A(\overline{A} + B)(\overline{A} + B + \overline{C})$

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

$= (000)(001)(010)(011)(100)(101)$

$= M_0 \cdot M_1 \cdot M_2 \cdot M_3 \cdot M_4 \cdot M_5$

$= \prod M(0, 1, 2, 3, 4, 5)$

The maxterms M_6 and M_7 are missing in the POS form. So,

the SOP form will contain the minterms 6 and 7

1.8. KARNAUGH MAP OR K-MAP & MINIMISATION OF LOGICAL EXPRESSIONS, DON'T CARE CONDITIONS: -

- The K- map is a chart or a graph, composed of an arrangement of adjacent cells, each representing a particular combination of variables in sum or product form.
- The K- map is systematic method of simplifying the Boolean expression.

TWO VARIABLE K-MAP:-

A two variable expression can have $2^2 = 4$ possible combinations of the input variables A and B.

Mapping of SOP Expression:-

- The 2 variable K-map has $2^2 = 4$ squares. These squares are called cells.
- A '1' is placed in any square indicates that corresponding minterm is included in the output expression, and a 0 or no entry in any square indicates that the corresponding minterm does not appear in the expression for output.

A \ B	0	1
0	$\overline{A}\,\overline{B}$	$\overline{A}B$
1	$A\overline{B}$	AB

Example:-

Map expression $f = A\overline{B} + A\overline{B}$

Solution:-

The expression minterms is

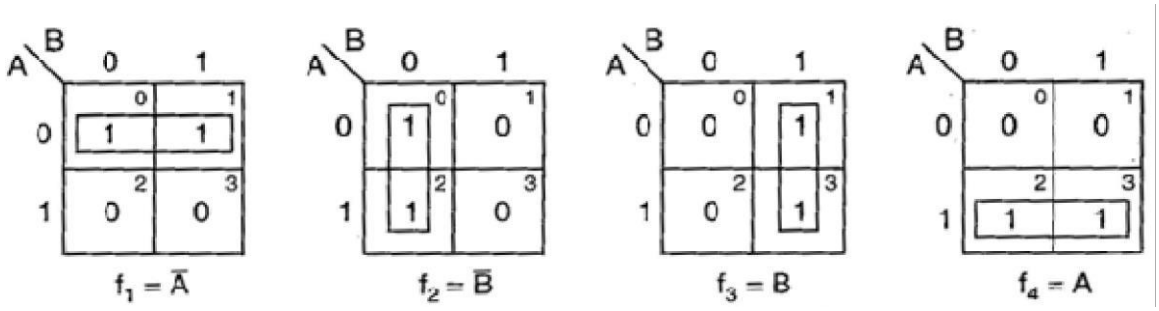
$F = m_1 + m_2 = m(1, 2)$

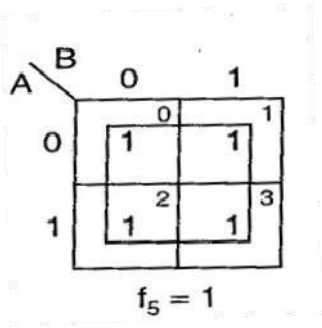
A \ B	0	1
0	0 (0)	1 (1)
1	1 (2)	0 (3)

Minimization of SOP Expression:-

To minimize a Boolean expression given in the SOP form by using K- map, the adjacent squares having 1s, that is minterms adjacent to each other are combined to form larger squares to eliminate some variables.

The possible minterm grouping in a two variable K- map are shown below





- Two minterms, which are adjacent to each other, can be combined to form a bigger square called 2 – square or a pair. This eliminates one variable that is not common to both the minterms.
- Two 2-squares adjacent to each other can be combined to form a 4- square. A 4- square eliminates 2 variables. A 4-square is called a quad.
- Consider only those variables which remain constant throughout the square, and ignore the variables which are varying. The non-complemented variable is the variable remaining constant as 1.The complemented variable is the variable remaining constant as a 0 and the variables are written as a product term..

Example:-

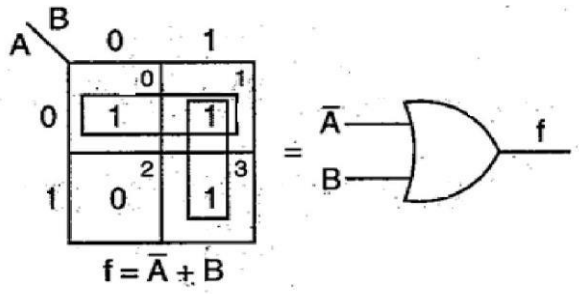
Reduce the expression f= $A\bar{B} + \bar{A}\bar{B} + AB$ using mapping.

Solution:-

Expressed in terms of minterms, the given expression is $f = m_0$

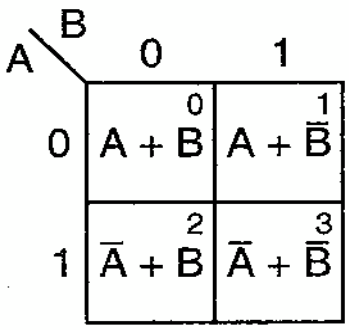
$+ m_1 + m_3 = \sum m\ (0, 1, 3)$

$F = \bar{A} + B$

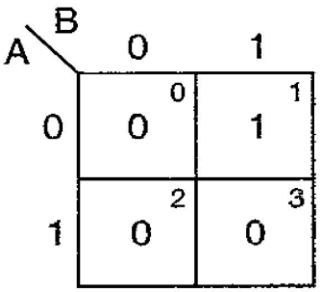


Mapping of POS Expression:-

Each sum term in the standard POS expression is called a Maxterm. A function in two variables (A,B) has 4 possible maxterms, A + B, A + B, A + B and $\overline{A}$ + B . They are represented as M_0, M_1, M_2 and M_3 respectively.



The maxterm of a two variable K-map



Example:-

Plot the expression f= (A + B)(A + $\overline{B}$)(A +B) $\overline{\ }$

Solution:-

Expression interms of maxterms is f = πM (0, 2, 3)

Example:-

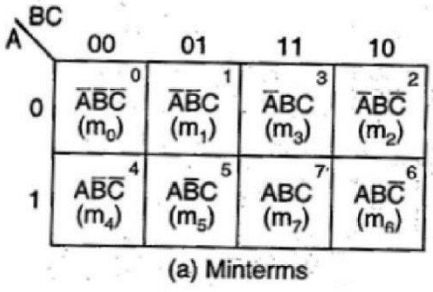
Reduce the expression f = (A + B)($\overline{A}$ + B)($\overline{A}$ +B) using mapping

Solution:-

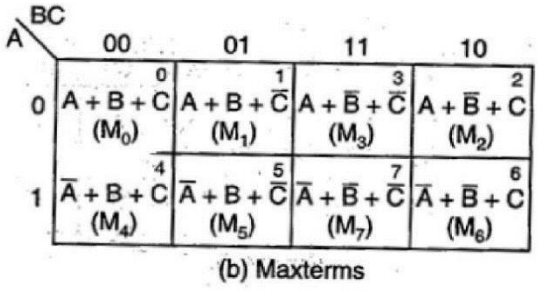
The given expression in terms of maxterms is f = πM (0, 1, 3)

THREE VARIABLE K-MAP:-

A function in three variables (A, B, C) can be expressed in SOP and POS form having eight possible combination. A three variable K- map have 8 squares or cells and each square minterm on the map represents a or maxterm is shown in the figure below.



(a) Minterms



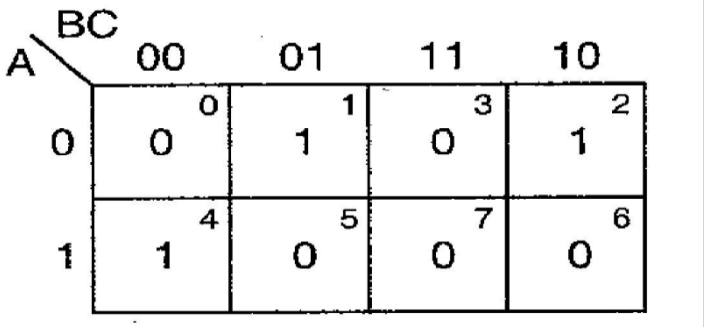
(b) Maxterms

Example:-

Map the expression f = (A + B + C) (A + B+C) (A + B + C) (A + B + C) (A + B + C)

Solution:-

So in the POS form the expression is f = π M (0, 5, 7, 3, 6)



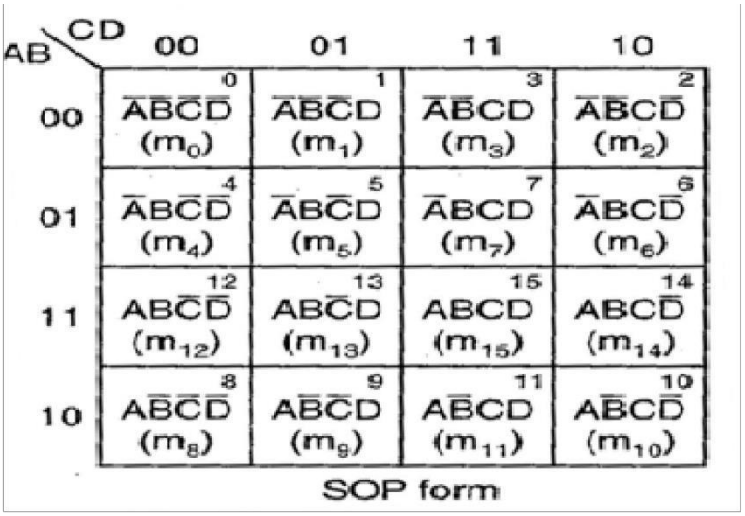
Minimization of SOP and POS Expressions:-

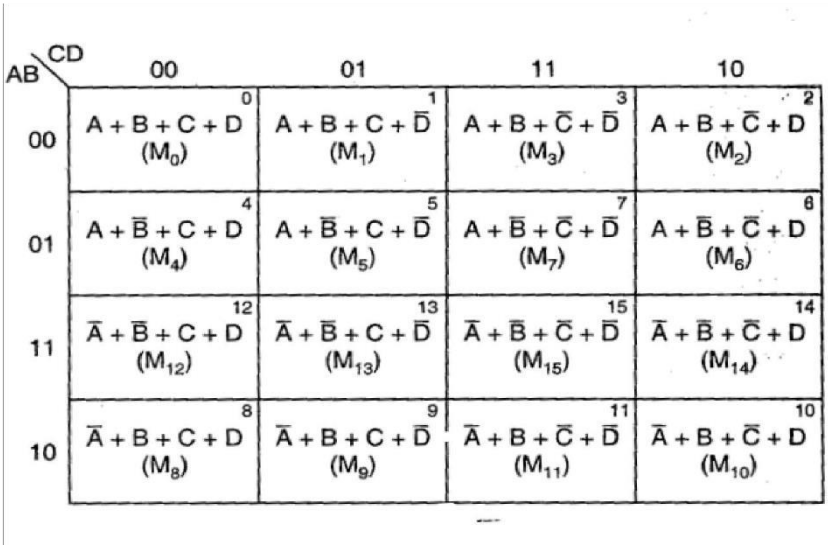
For reducing the Boolean expressions in SOP (POS) form the following steps are given below

- Draw the K-map and place 1s (0s) corresponding to the minterms (maxterms) of the SOP (POS) expression.
- In the map 1s (0s) which are not adjacent to any other 1(0) are the isolated minterms (maxterms). They are to be read as they are because they cannot be combined even into a 2-square.
- For those 1s (0s) which are adjacent to only one other 1(0) make them pairs (2 squares).

FOUR VARIABLE K-MAP:-

A four variable (A, B, C, D) expression can have $2^4 = 16$ possible combinations of input variables. A four variable K-map has $2^4 = 16$ squares or cells and each square on the map represents either a minterm or a maxterm as shown in the figure below. The binary number designations of the rows and columns are in the gray code. The binary numbers along the top of the map indicate the conditions of C and D along any column and binary numbers along left side indicate the conditions of A and B along any row. The numbers in the top right corners of the squares indicate the minterm or maxterm desginations.





POS FORM

Minimization of SOP and POS Expressions:-

For reducing the Boolean expressions in SOP (POS) form the following steps are given below

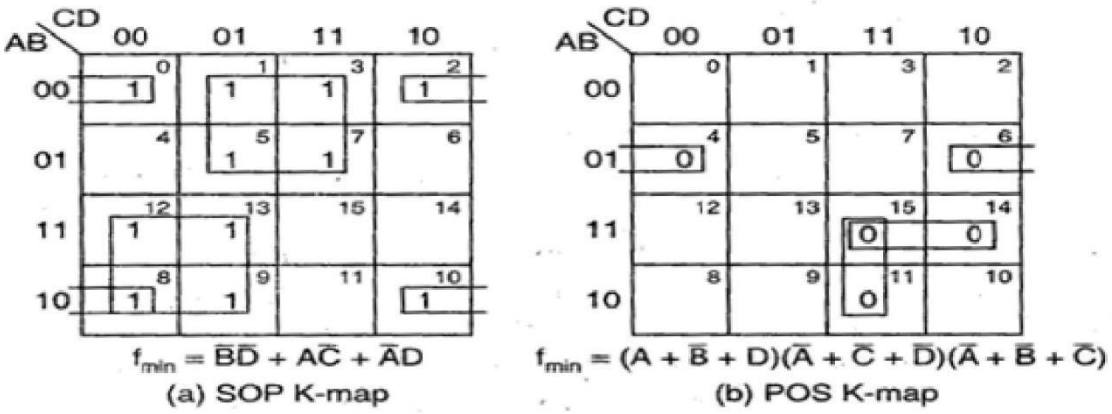
- Draw the K-map and place 1s (0s) corresponding to the minterms (maxterms) of the SOP (POS) expression.
- In the map 1s (0s) which are not adjacent to any other 1(0) are the isolated minterms (maxterms). They are to be read as they are because they cannot be combined even into a 2-square.
- For those 1s (0s) which are adjacent to only one other 1(0) make them pairs (2 squares).
- For quads (4- squares) and octet (8 squares) of adjacent 1s (0s) even if they contain some 1s (0s) which have already been combined. They must geometrically form a square or a rectangle.
- For any 1s (0s) that have not been combined yet then combine them into bigger squares if possible. Form the minimal expression by summing (multiplying) the product (sum) terms of all the groups.

Example:-

Reduce using mapping the expression f = ∑ m (0, 1, 2, 3, 5, 7, 8, 9, 10, 12, 13)

Solution:-

The given expression in POS form is f = π M (4, 6, 11, 14, 15) and in SOP form f = ∑ m (0, 1, 2, 3, 5, 7, 8, 9, 10, 12, 13)



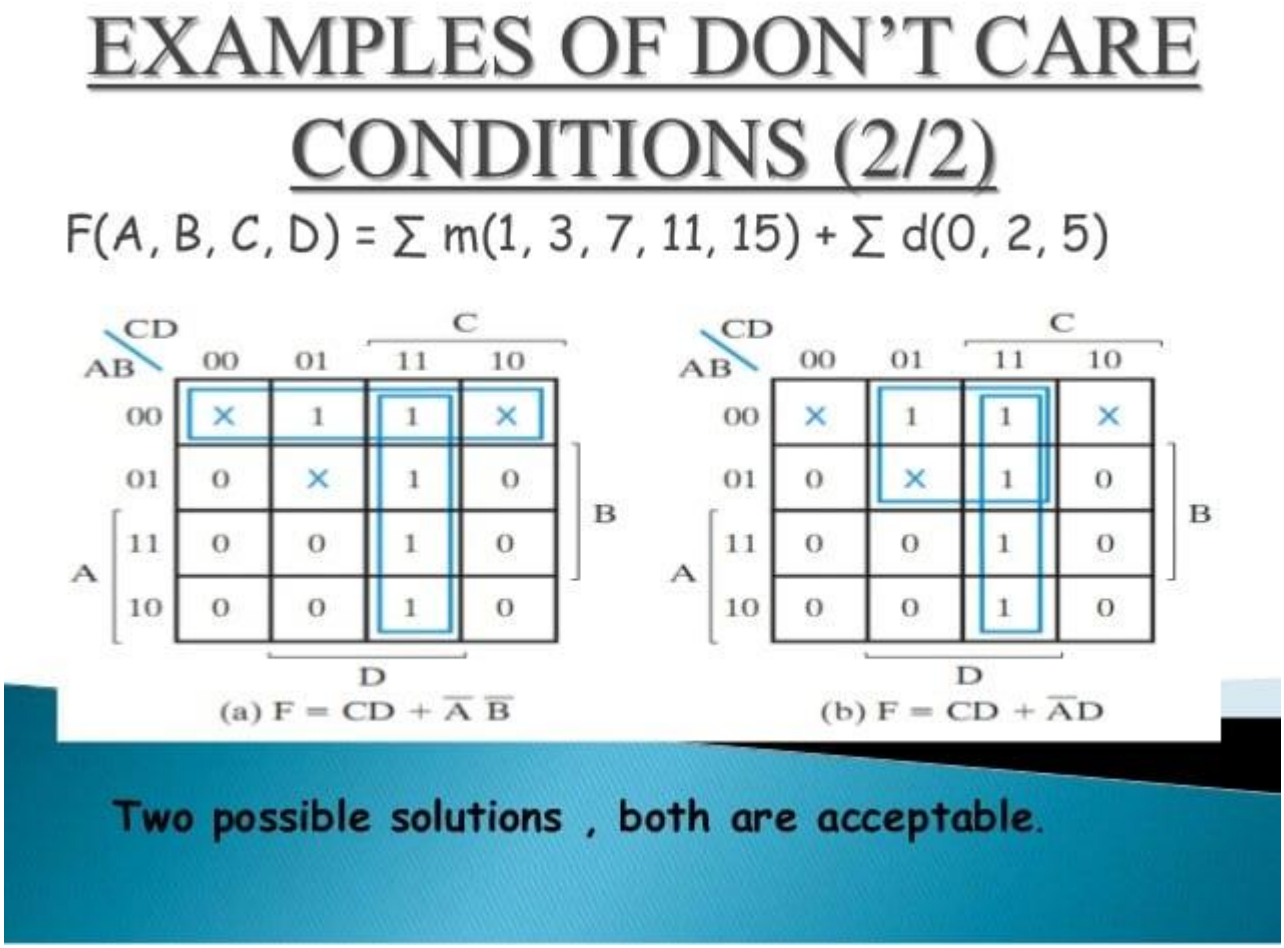
The minimal SOP expression is $f_{min} = B\overline{D} + \overline{A}C + \overline{A}\overline{D}$

The minimal POS expression is $f_{min} = (A + B + \overline{D})(A + \overline{C} + \overline{D})(A + B + C)$

DON'T CARE COMBINATIONS:-

The combinations for which the values of the expression are not specified are called don't care combinations or optional combinations and such expression stand incompletely specified. The output is a don't care for these invalid combinations. The don't care terms are denoted by d or X. During the process of designing using SOP maps, each don't care is treated as 1 to reduce the map otherwise it is treated as 0 and left alone. During the process of designing using POS maps, each don't care is treated as 0 to reduce the map otherwise it is treated as 1 and left alone.

A standard SOP expression with don't cares can be converted into standard POS form by keeping the don't cares as they are, and the missing minterms of the SOP form are written as the maxterms of the POS form. Similarly, to convert a standard POS expression with don't cares can be converted into standard SOP form by keeping the don't cares as they are, and the missing maxterms of the POS form are written as the minterms of the SOP form.



Possible Short Question with Answer & Long question

1. Define Radix /Base. (2009-S,2010-S, 2019-S)
   Ans- It specifies the no of symbols used for corresponding number system. In decimal no system base is 10. In Binary no system base is 2 .
2. Convert $(111.101)_2$ to decimal. (S-2009)
   Solution:
3. $(111.101)_2 = (1 \times 2^2) + (1 \times 2^1) + (1 \times 2^0) + (1 \times 2^{-1}) + (0 \times 2^{-2}) + (1 \times 2^{-3})$
   $= 4 + 2 + 1 + 0.5 + 0 + 0.125$
   $= (7.625)_{10}$
4. Convert $(01011111011.011111)_2$ into hexadecimal.

Solution:

Given Binary number 010 1111 1011 . 0111 11

Group of 3 bits are = 0010 1111 1011 . 0111 1100

Convert each group into octal = 2 F B 7 7 C

The result is $(2FB.7C)_{16}$

5. Convert $(378.93)_{10}$ into octal.

Solution:

8 | 378		0.93 x 8 = 7.44
8 | 47	— 2	0.44 x 8 = 3.52
8 | 5	— 7	0.52 x 8 = 4.16
0	— 5	0.16 x 8 = 1.28

Result of $(378.93)_{10}$ is $(572.7341)_8$

6. Subtract $(10000)_2$ from $(11010)_2$ using 1's complement.

Solution:-

1 1 0 1 0 1 1 0 1 0 = 26

- 1 0 0 0 0 => + 0 1 1 1 1 (1's complement) = - 16

Carry → 1 0 1 0 0 1 + 10

\+ 1

01010 = +10

Result is +10

7. Write the difference between between weighted code & Non weighted code. (S-2016)

Ans.-In weighted codes, for each position (or bit) ,there is specific weight attached.

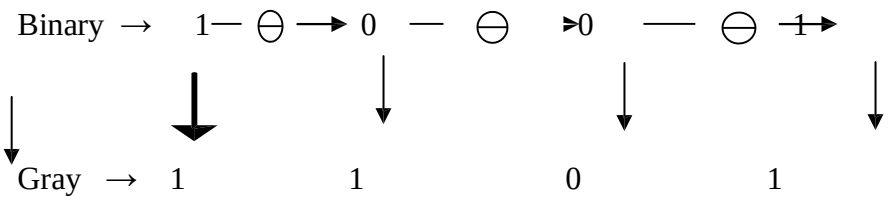
For example, in binary number, each bit is assigned particular weight 2^n where 'n' is the bit number for n = 0,1,2,3,4 the weights are 1,2,4,8,16 respectively. Example :- BCD

Non-weighted codes are codes which are not assigned with any weight to each digit position, i.e., each digit position within the number is not assigned fixed value.

Example:- Excess – 3 (XS -3) code and Gray codes

8. Convert the binary 1001 to the Gray code. (W-2020)

Solution :-`



The gray code is 1101

9. Which gates are called universal gates & why.? (S-2008,09)
   Ans.The NAND and NOR gates are called universal building blocks. Both NAND and NOR gates can perform all logic functions i.e. AND, OR, NOT, EX OR and EX NOR. As all other gates & any other functions are derived from these gates ,hence these are called universal gates.

10. State De Morgan's Theorems (S-2006,14,19)

Ans- It has two laws

1st law- This law states that the complement of a sum of variables is equal to the product of their individual complements.

2nd law - This law states that the complement of a product of variables is equal to the sum of their individual complements.

Possible Long questions.

1. Subtract 13 from 29 by using 2' complement method. (S-2007)
2. Subtract 16 from 13 by using 2's complement method. (S-2019)
3. Explain Ex-3 code. Find the equivalent Ex-3 code for 29. (S-2010-2016)
4. Convert $(100110)_2$ to Gray Code & verify the result.
5. Implement Ex-or gate by using NAND gate only.
6. Implement Ex_OR gate by using NOR gate only. S-2014,15,16,19) W-2020.
7. Minimize Mi.M3.M5 .M7 .M4 by using K-Map (S-2019,20)
8. Minimize m_{1+} m_{2+} m_{4+} m_{5+} m_{7+} m_{9+} m_{11+} m_{13+} m_{14+} m_{15} + d(3,4) by using K-Map. (S-2016,17.18.19.20,21)
9. Implement A+B'C by using NAND gates only. (S-2019,20)

UNIT 2: COMBINATIONAL LOGIC CIRCUITS

Learning Objectives:

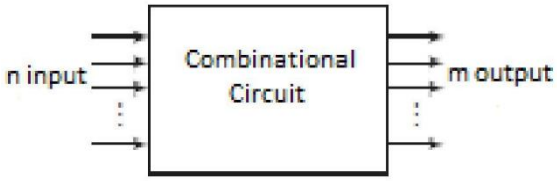
2.1 Half adder, Full adder, Half Subtractor, Full Subtractor, Serial & Paraller Binary 4 bit adder.

2.2 Multiplexure (4:1), De-Multiplexure(1:4), Decoder, Encoder, Digital Comparator(3 bit)

2.3 Seven Segment Decoder.

(Definition, relevance, Gate Level of Logic circuit, Truth Table, Application of above)

- A combinational circuit consists of logic gates whose outputs at any time are determined from only the present combination of inputs.
- A block diagram of a combinational circuit is shown in the below.
- The n input binary variables come from an external source; the m output variables are produced by the internal combinational logic circuit and go to an external destination.
- Each input and output variable to be a binary signal that represents logic 1and logic 0.



2.1 HALF ADDER, FULL ADDER, HALF SUBTRACTOR, FULL SUBTRACTOR, SERIAL & PARALLEL BINARY FOUR BIT ADDER: -

HALF ADDER-

- This circuit needs two binary inputs and two binary outputs.
- The input variables designate the augend and addend bits; the output variables produce the sum and carry. Symbols x and y are assigned to the two inputs and S (for sum) and C (for carry) to the outputs.
- The truth table for the half adder is listed in the below table.
- The C output is 1 only when both inputs are 1. The S output represents the least significant bit of the sum.

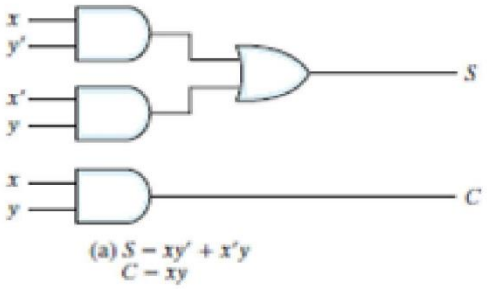
x	y	D	B
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Truth Table

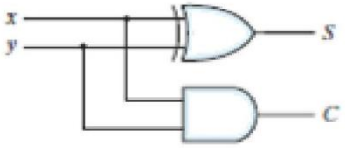
- The simplified sum-of-products expressions are

$$S = x'y + xy' \quad C = xy$$

- The logic diagram of the half adder implemented in sum of products is shown in the below figure. It can be also implemented with an exclusive-OR and an AND gate.



(a) $S = xy' + x'y$
$C = xy$



(b) $S = x \oplus y$
$C = xy$

FULL ADDER:-

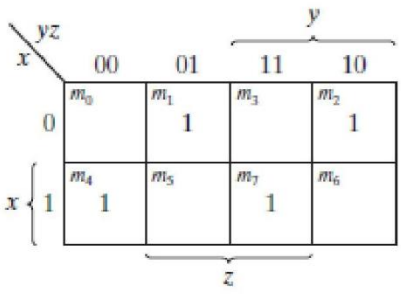
- A full adder is a combinational circuit that forms the arithmetic sum of three bits.
- It consists of three inputs and two outputs. Two of the input variables, denoted by x and y , represent the two significant bits to be added. The third input, z , represents the carry from the previous lower significant position.

x	y	z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

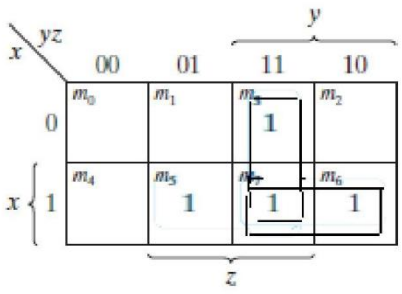
Truth Table

- The two outputs are designated by the symbols S

for sum and C for carry.



(a) $S = x'y'z + x'yz' + xy'z' + xyz$



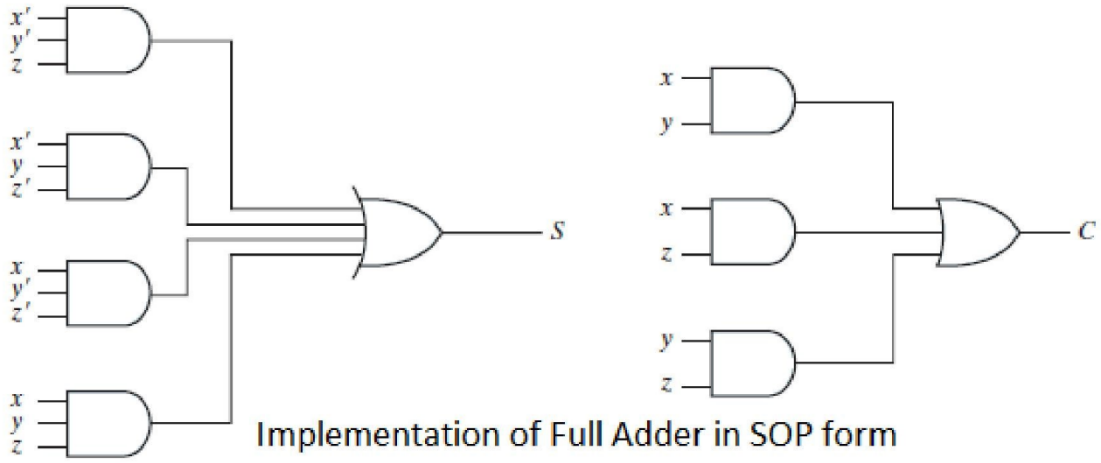
(b) $C = xy + xz + yz$

K-Map for full adder

- The binary variable S gives the value of the least significant bit of the sum. The binary variable C gives the output carry formed by adding the input carry.
- From Truth Table it is seen that when all input bits are 0, the output is 0.
- The S output is equal to 1 when only one input is equal to 1 or when all three inputs are equal to1
- The C output has a carry of 1 if two or three inputs are equal to 1.
- The simplified expressions are

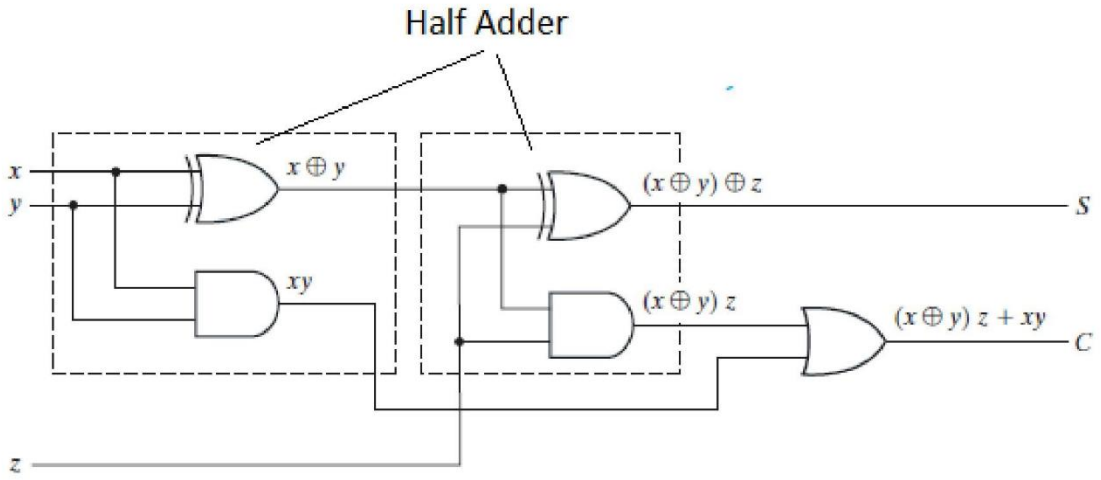
S = x'y'z + x'yz' + xy'z' + xyz
C = xy + xz + yz

- The logic diagram for the full adder implemented in sum-of-products form is shown in figure.



Implementation of Full Adder in SOP form

- It can also be implemented with two half adders and one OR gate as shown in the figure.



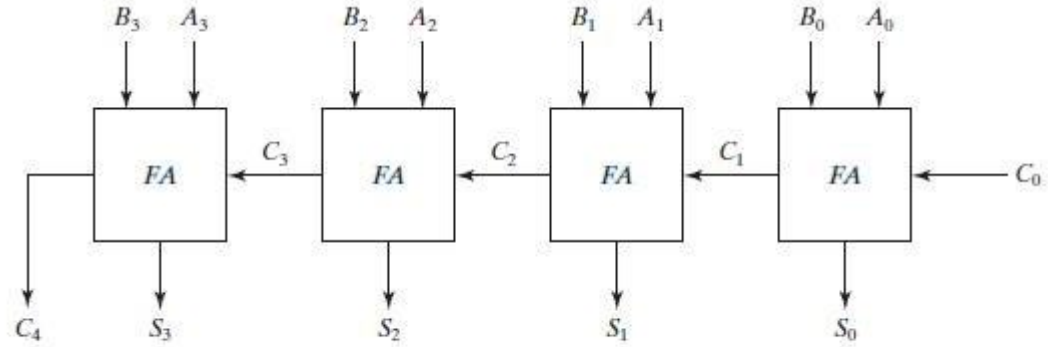
Implementation of Full Adder using Two Half Adders and an OR gate

- A full adder is a combinational circuit that forms the arithmetic sum of three bits.

BINARY ADDER:-

- A binary adder is a digital circuit that produces the arithmetic sum of two binary numbers.
- It can be constructed with full adders connected in cascade, with the output carry from each full adder connected to the input carry of the next full adder in the chain.
- Addition of n-bit numbers requires a chain of n full adders or a chain of one-half adder and (n-1) full adders.
- The interconnection of four full-adder (FA) circuits to provide a four-bit binary ripple carry adder is shown in the figure.
- The augend bits of A and the addend bits of B are designated by subscript numbers from right to left, with subscript 0 denoting the least significant bit.
- The carries are connected in a chain through the full adders.
- An n -bit adder requires n full adders, with each output carry connected to the input carry of the next higher order full adder.

- The bits are added with full adders, starting from the least significant position (subscript 0), to form the sum bit and carry bit. The input carry C_0 in the least significant position must be 0.
- The value of C_{i+1} in a given significant position is the output carry of the full adder. This value is transferred into the input carry of the full adder that adds the bits one higher significant position to the left.
- The sum bits are thus generated starting from the rightmost position and are available as soon as the corresponding previous carry bit is generated. All the carries must be generated for the correct sum bits to appear at the outputs.



Four Bit Binary Adder

HALF SUBTRACTOR: -

- This circuit needs two binary inputs and two binary outputs.
- Symbols x and y are assigned to the two inputs and D (for difference) and B (for borrow) to the outputs.
- The truth table for the half subtractor is listed in the below table.

x	y	D	B
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

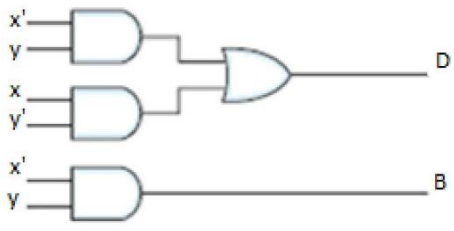
Truth Table

- The B output is 1 only when the inputs are 0 and 1. The D output represents the least significant bit of the subtraction. The subtraction operation is done by using the following rules as

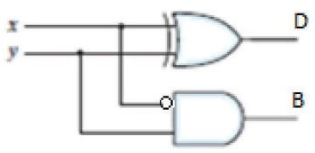
0-0=0;
0-1=1 with borrow 1;
1-0=1;
1-1=0.

- The simplified Boolean functions for the two outputs can be obtained directly from the truth table. The simplified sum-of-products expressions are

D = x'y + xy' and B =x'y



D = x'y+xy'
B = x'y



D = x ⊕ y
B=x'y

- The logic diagram of the half adder implemented in sum of products is shown in the figure. It can be also implemented with an exclusive-OR and an AND gate with one inverted input.

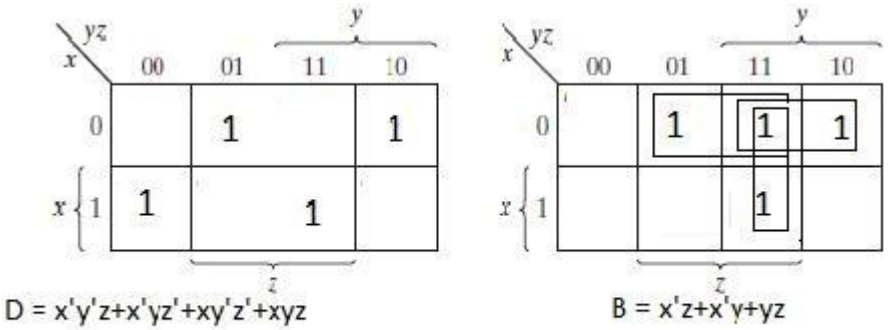
FULL SUBTRACTOR:-

- A full subtractor is a combinational circuit that forms the arithmetic subtraction operation of three bits.
- It consists of three inputs and two outputs. Two of the input variables, denoted by x and y , represent the two significant bits to be subtracted. The third input, z, is subtracted from the result 0f the first subtraction.

x	y	z	D	B
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

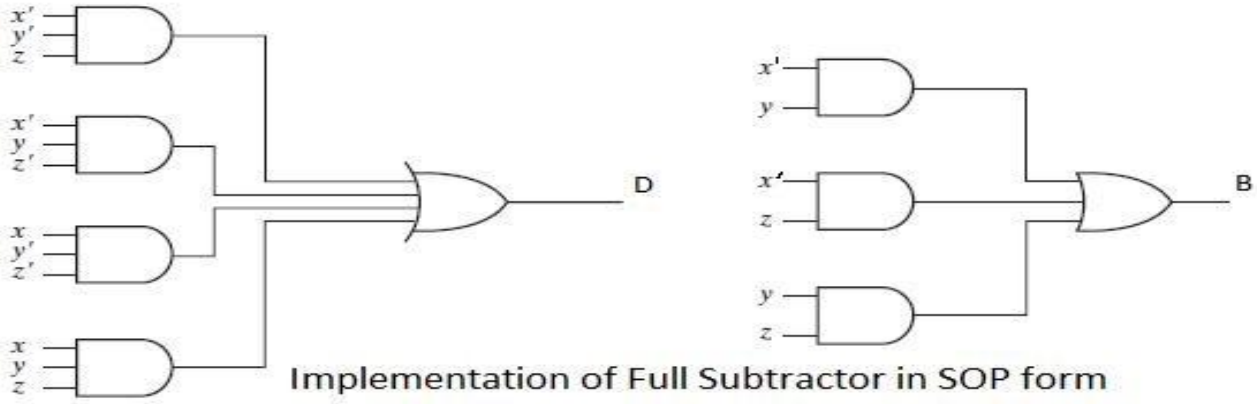
Truth Table

- The two outputs are designated by the symbols D for difference and B for borrow.
- The binary variable D gives the value of the least significant bit of the difference. The binary variable B gives the output borrow formed during the subtraction process.



K-Map for full Subtractor

- The simplified expressions are
  D = x'y'z + x'yz' + xy'z' + xyz
  B = x'z + x'y + yz
- The logic diagram for the full adder implemented in sum-of-products form is shown in figure.



Implementation of Full Subtractor in SOP form

MAGNITUDE COMPARATOR: -

- A magnitude comparator is a combinational circuit that compares two numbers A and B and determines their relative magnitudes.
- The following description is about a 2-bit magnitude comparator circuit.
- The outcome of the comparison is specified by three binary variables that indicate whether A < B, A = B, or A > B.
- Consider two numbers, A and B, with two digits each. Now writing the coefficients of the numbers in descending order of significance:

$$A = A_1 A_0$$
$$B = B_1 B_0$$

- The two numbers are equal if all pairs of significant digits are equal i.e. if and only if A1 = B1, and A0 = B0.
- When the numbers are binary, the digits are either 1 or 0, and the equality of each pair of bits can be expressed logically with an exclusive-NOR function as

$$x_1 = A_1 B_1 + A_1' B_1'$$

And

$$x_0 = A_0 B_0 + A_0' B_0'$$

- The equality of the two numbers A and B is displayed in a combinational circuit by an output binary variable that we designate by the symbol (A = B).
- This binary variable is equal to 1 if the input numbers, A and B , are equal, and is equal to 0 otherwise.
- For equality to exist, all xi variables must be equal to 1, a condition that dictates an AND operation of all variables:

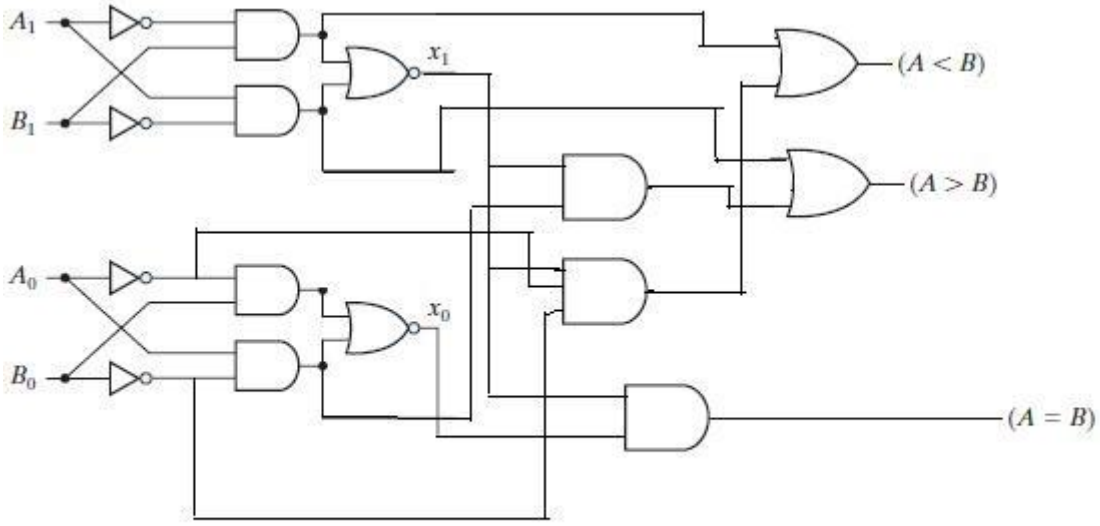
$$(A = B) = x_1 x_0$$

- The binary variable (A = B) is equal to 1 only if all pairs of digits of the two numbers are equal.
- To determine whether A is greater or less than B, we inspect the relative magnitudes of pairs of significant digits, starting from the most significant position. If the two digits of a pair are equal, we compare the next lower significant pair of digits. If the corresponding digit of A is 1 and that of B is 0, we conclude that A > B. If the corresponding digit of A is 0 and that of B is 1, we have A < B. The sequential comparison can be expressed logically by the two Boolean functions

$$(A>B) = A_1 B_1' + x_1 A_0 B'0$$
$$(A<B) = A_1' B_1 + x_1 A_0' B_0'$$

A_1	A_0	B_1	B_0	A>B	A<B	A=B
0	0	0	0	0	0	1
0	0	0	1	0	1	0
0	0	1	0	0	1	0
0	0	1	1	0	1	0
0	1	0	0	1	0	0
0	1	0	1	0	0	1
0	1	1	0	0	1	0
0	1	1	1	0	1	0
1	0	0	0	1	0	0
1	0	0	1	1	0	0
1	0	1	0	0	0	1
1	0	1	1	0	1	0
1	1	0	0	1	0	0
1	1	0	1	1	0	0
1	1	1	0	1	0	0
1	1	1	1	0	0	1

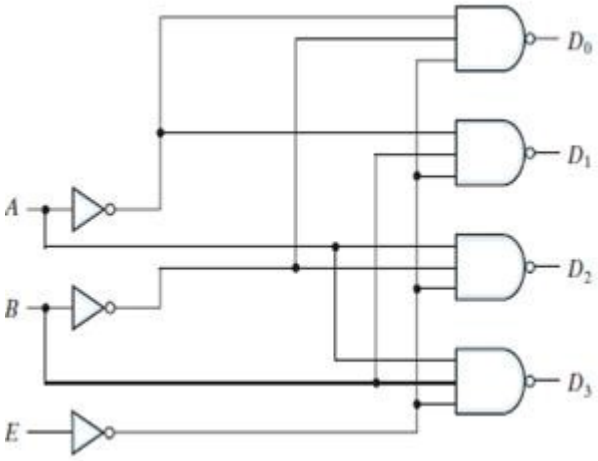
Truth Table



Logic Diagram of 2-bit Magnitude Comparator

2.2 MULTIPLEXER (4:1),DEMULTIPLEXER(1:4),DECODER, ENCODER, DIGITALCOMPARATOR (3BIT):

DECODER



E	A	B	D_0	D_1	D_2	D_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	1

- A decoder is a combinational circuit that converts binary information from n input lines to a maximum of 2^n unique output lines.
- Consider the three-to-eight-line decoder circuit of three inputs are decoded into eight outputs, each representing one of the minterms of the three input variables.
- The input variables represent a binary number, and the outputs represent the eight digits of a number in the octal number system.
- However, a three-to-eight-line decoder can be used for decoding any three-bit code to provide eight outputs.
- A two-to-four-line decoder with an enable input constructed with NAND gates is shown in Fig.
- The circuit operates with complemented outputs and a complement enable input. **The decoder is enabled when E is equal to 0 (i.e., active-low enable)**. As indicated by the truth table, only one output can be equal to 0 at any given time; all other outputs are equal to 1.
- The output whose value is equal to 0 represents the minterm selected by inputs A and B.
- **The circuit is disabled when E is equal to 1**, regardless of the values of the other two inputs.
- When the circuit is disabled, none of the outputs are equal to 0 and none of the minterms are selected.
- In general, a decoder may operate with complemented or un-complemented outputs.
- The enable input may be activated with a 0 or with a 1 signal.
- A decoder with enable input can function as a demultiplexer— a circuit that receives information from a single line and directs it to one of 2^n possible output lines.
- The selection of a specific output is controlled by the bit combination of n selection lines.
- The decoder of Fig. can function as a one-to-four-line demultiplexer when E is taken as a data input line and A and B are taken as the selection inputs.
- If the selection lines AB = 10, output D_2 will be the same as the input value E, while all other outputs are maintained at 1.
- Since decoder and demultiplexer operations are obtained from the same circuit, a decoder with an enable input is referred to as a decoder – demultiplexer.
- **A application of this decoder is binary-to-octal conversion**.

ENCODER:-

- An encoder is a digital circuit that performs the inverse operation of a decoder.
- An encoder has 2^n (or fewer) input lines and n output lines.
- The output lines generate the binary code corresponding to the input value.

Inputs								Outputs		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	x	y	z
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

- The above Encoder has eight inputs (one for each of the octal digits) and three outputs that generate the corresponding binary number.
- It is assumed that only one input has a value of 1 at any given time.
- Output z is equal to 1 when the input octal digit is 1, 3, 5, or 7.
- Output y is 1 for octal digits 2, 3, 6, or 7, and output x is 1 for digits 4, 5, 6, or 7.
- These conditions can be expressed by the following Boolean output functions:

$$z = D_1 + D_3 + D_5 + D_7$$
$$y = D_2 + D_3 + D_6 + D_7$$
$$x = D_4 + D_5 + D_6 + D_7$$

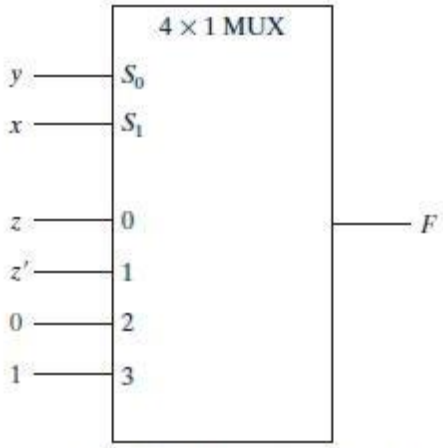
- The encoder can be implemented with three OR gates.
- The encoder defined above has the limitation that only one input can be active at any given time.

MULTIPLEXER:-

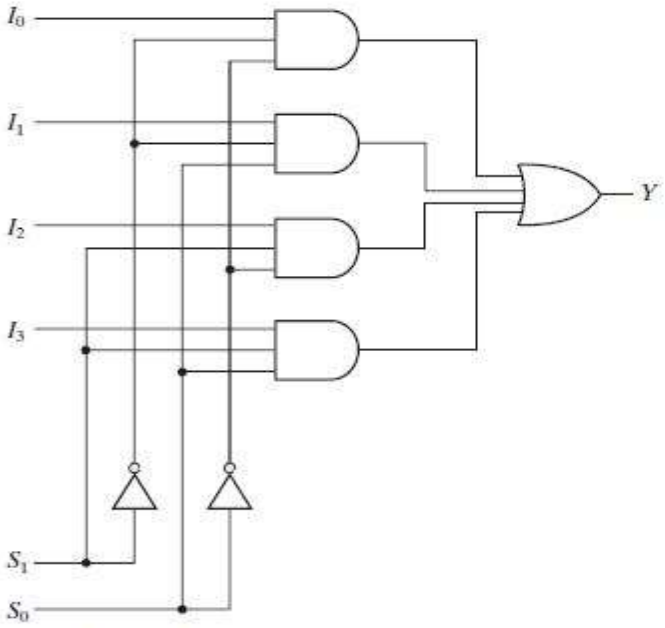
- A multiplexer is a combinational circuit that selects binary information from one of many input lines and directs it to a single output line.
- The selection of a particular input line is controlled by a set of selection lines.
- Normally, there are 2^n input lines and n selection lines whose bit combinations determine which input is selected.
- A four-to-one-line multiplexer is shown in the below figure. Each of the four inputs, I_0 through I_3, is applied to one input of an AND gate.
- Selection lines S_1 and S_0 are decoded to select a particular AND gate. The outputs of the AND gates are applied to a single OR gate that provides the one-line output.
- The function table lists the input that is passed to the output for each combination of the binary selection values.
- To demonstrate the operation of the circuit, consider the case when

$S_1S_0 = 10$.

- The AND gate associated with input I_2 has two of its inputs equal to 1 and the third input connected to I_2.
- A multiplexer is also called a data selector, since it selects one of many inputs and steers the binary information to the output line.



(b) Multiplexer implementation



Logic diagram

S_1	S_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

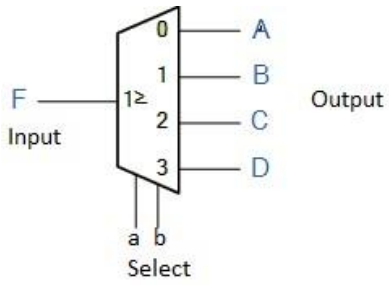
Truth table

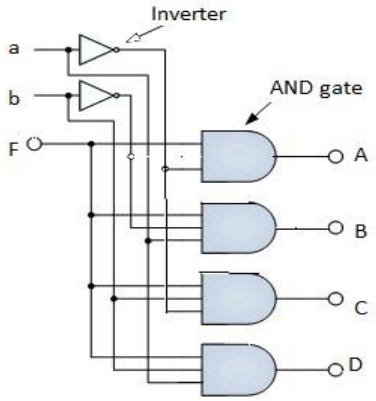
DE MULTIPLEXER:-

- The data distributor, known more commonly as a Demultiplexer or "Demux" for short, is the exact opposite of the Multiplexer.
- The demultiplexer takes one single input data line and then switches it to any one of a number of individual output lines one at a time. The demultiplexer converts a serial data signal at the input to a parallel data at its output lines as shown below.
- The Boolean expression for this 1-to-4 demultiplexer above with outputs A to D and data select lines a, b is given as:

F a'b' =A
F a'b =B
F ab' =C
F ab =D

- The function of the demultiplexer is to switch one common data input line to any one of the 4 output data lines A to D in our example above. As with the multiplexer the individual solid state switches are selected by the binary input address code on the output select pins "a" and "b" as shown.





Logic Diagram

- Demultiplexer which convert multiple lines to a single data line, there are devices available which convert data to and from multiple lines.

Output Select		Data output Selected
b	a	
0	0	A
0	1	B
1	0	C
1	1	D

Truth Table

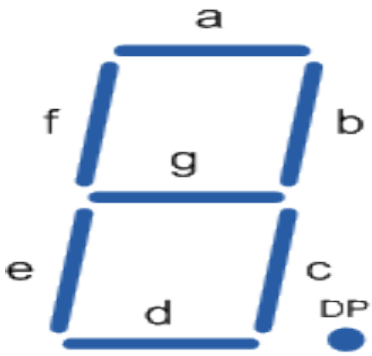
2.3 Seven segment decoder:

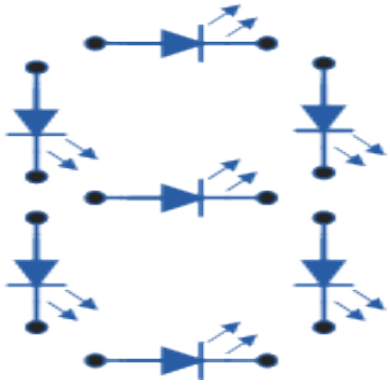
- This BCD to seven segment decoder has four input lines (**A, B, C and D**) and 7 output lines (a, b, c, d, e, f and g), this output is given to seven segment LED display which displays the decimal number depending upon inputs.
- A seven-segment display is a form of electronic display device for displaying decimal numerals that is an alternative to the more complex dot matrix displays. Seven-segment displays are widely used in digital clocks, electronic meters, basic calculators, and other electronic devices that display

numerical information.

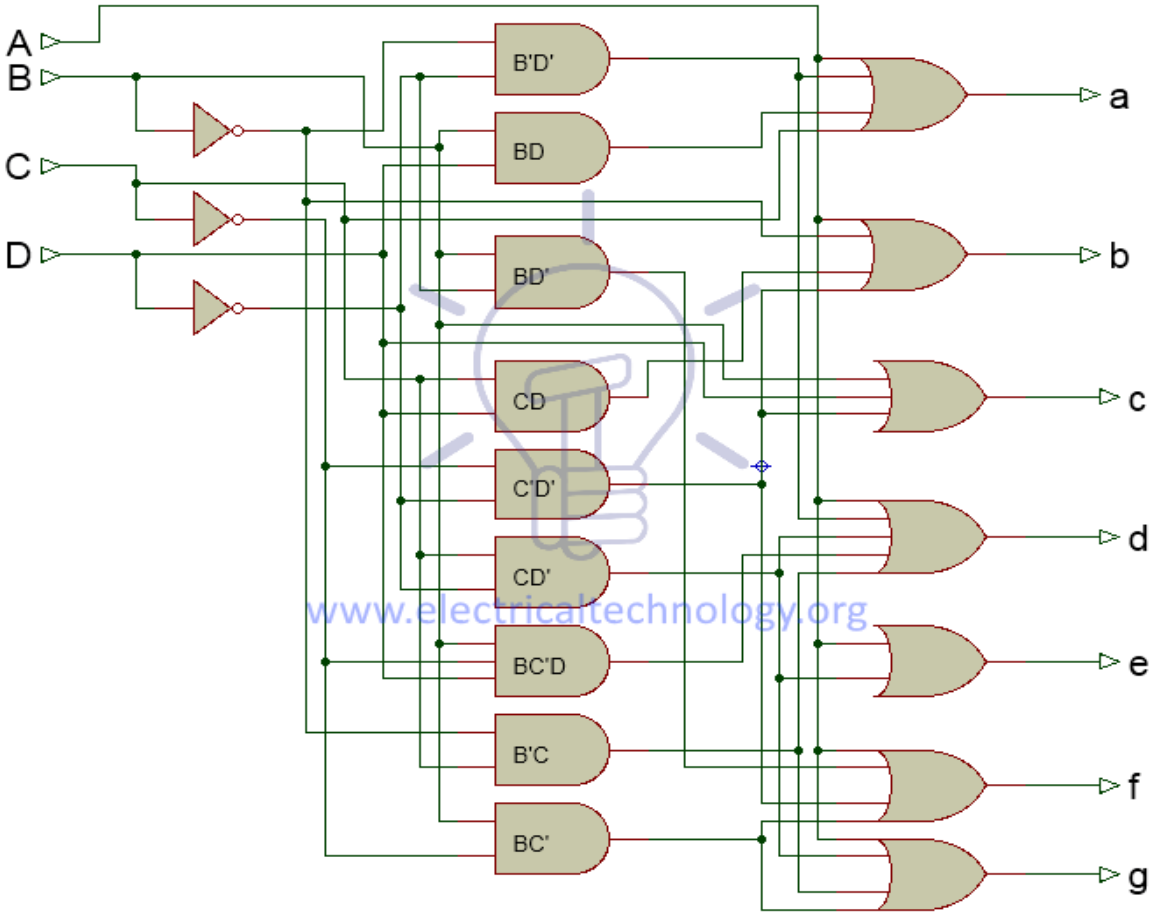
- The binary digit 1 is activate the LED for corresponding digit in common cathode & binary digit 0 is activate the LED for common anode seven segment display.

7-Segment Display Format





Decimal Digit	Input lines				Output lines							Display pattern
	A	B	C	D	a	b	c	d	e	f	g	
0	0	0	0	0	1	1	1	1	1	1	0	
1	0	0	0	1	0	1	1	0	0	0	0	
2	0	0	1	0	1	1	0	1	1	0	1	
3	0	0	1	1	1	1	1	1	0	0	1	
4	0	1	0	0	0	1	1	0	0	1	1	
5	0	1	0	1	1	0	1	1	0	1	1	
6	0	1	1	0	1	0	1	1	1	1	1	
7	0	1	1	1	1	1	1	0	0	0	0	
8	1	0	0	0	1	1	1	1	1	1	1	
9	1	0	0	1	1	1	1	1	0	1	1	



Schematic of BCD to 7-Segment Decoder

Possible Short Type Question with Answer.

1. **Define De-Mux.**

Ans- The data distributor, known more commonly as a Demultiplexer or "Demux" for short, is the exact opposite of the Multiplexer.

- The demultiplexer takes one single input data line and then switches it to any one of a number of individual output lines one at a time. The demultiplexer converts a serial data signal at the input to a parallel data at its output lines

2. **Define Mux. (W-2019) or Why it is called Data Selector? W-2020,S-2019**

Ans- A multiplexer is a combinational circuit that selects binary information from one of many input lines and directs it to a single output line. Hence it is called as Data Selector.The selection of a particular input line is controlled by a set of selection lines.Normally, there are 2^n input lines and n selection lines whose bit combinations determine which input is selected.

3. **Define Comparator .**

The basic function of a comparator is to compare the magnitude of two binary quantities to determine the relationship of those quantities . The Ex-Or gate can be used as a basic comparator.

4. **Define Decoder.**

Ans-A decoder is a combinational circuit that converts binary information from n input lines to a maximum of 2^n unique output lines

Possible Long Question

1. **With neat circuit diagram explain 4:1 line MUX circuit S-2015/2016/2019(s/W)**
2. **Design & explain about Full Subtractor. Give its logic expression & truth table. Implement logic ckt with any one universal gate. 2014(s),2016 (s)**
3. **With neat ckt diagram explain the function of 1:4 De Mux Ckt. 2016 (s). W-2020.**
4. **Design 4:1 encoder with details.**
5. **Design & explain about Full Adder. Give its logic expression & truth table. Implement logic ckt with any one universal gate.**

UNIT-3
SEQUENTIAL LOGIC CIRCUITS

Learning Objectives:

3.1- PRINCIPLE OF F/F OPERATION, ITS TYPES

3.2-SR F/F USING NAND, NOR LATCH (UN CLOCKED)

3.3- Clocked SR, D, JK, T, MSJK Flip-Flop Symbol, Logic Circuit, Truth Table, and Applications

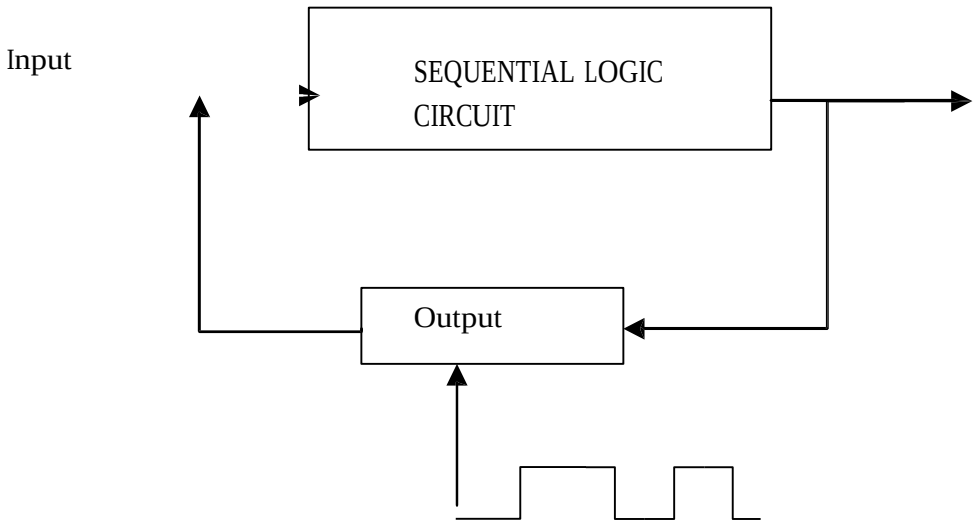
3.4-Concept of RACING CONDITION & how it can be avoided

SEQUENTIALCIRCUIT:-

- It is a circuit whose output depends upon the present input, previous output and the sequence in which the inputs are applied.

DIFFERENCE BETWEEN SLC AND CLC:-

- In combinational circuit output depends upon present input at any instant of time and do not use memory. Hence previous input does not have any effect on the circuit. But sequential circuit has memory and depends upon present input and previous output.
- Sequential circuits are slower than combinational circuits and these sequential circuits are harder to design. SLC consists of CLC to which storage elements are connected to form a feedback path.



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

3.1;- PRINCIPLE OF F/F OPERATION & ,ITS TYPES:-

Sequential logic circuits (SLC) are classified as

(i)
Synchronous SLC
(ii)
Asynchronous SLC

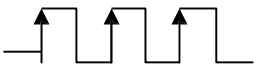
- The SLC that are controlled by clock are called synchronous SLC and those which are not controlled by a clock are asynchronous SLC.
- Clock:- A recurring pulse is called a clock.

FLIP-FLOP AND LATCH:-

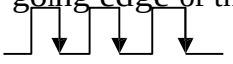
- A flip-flop or latch is a circuit that has two stable states and can be used to store information.
- It is a binary storage device capable of storing one bit of information. In a stable state, the output of a flip-flop is either 0 or 1.
- Latch is a non-clocked flip-flop and it is the building block for the flip-flop.
- Storage element that operate with signal level are called latches and those operate with clock transition are called as flip-flops lip-flop is a bistable element, its output remains in either of the stable states until an external event (known as a trigger) is applied.
- A flip-flop is also called a bi-stable multi-vibrator as it has two stable states. The input signals which command the flip-flop to change state are called excitations.
- Flip-flops using the clock signal are called clocked flip-flops.

Flip flop= Latch + Clock

- Clock-signals may be positive-edge triggered or negative-edge triggered.
- Positive-edge triggered flip-flops are those in which state transitions take place only at positive- going edge of the clock pulse.



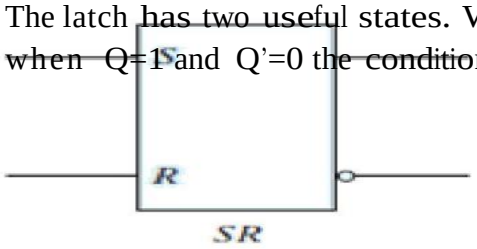
- Negative-edge triggered flip-flops are those in which state transition take place only at negative- going edge of the clock pulse.



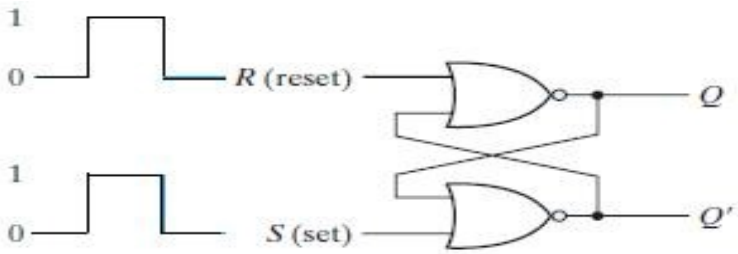
- Some common type of flip-flops includes
  a) SR (set-set) F-F
  b) D (data or delay) F-F
  c) T (toggle) F-F and
  d) JK F-F

3.2;-SR F/F USING NAND, NOR LATCH (UN CLOCKED)

- It has two outputs labeled Q and Q'. Two inputs are labeled S for set and R for reset.
- The latch has two useful states. When Q=0 and Q'=1 the condition is called reset state and when Q=1 and Q'=0 the condition is called set state.



Symbol for SR latch



- The figure represents a SR latch with two cross-coupled NOR gates. We know if any one of the input for a NOR gate is HIGH then its output will be LOW and if both the inputs are LOW then only the output will be HIGH. So input I is dominating over other input 0.

Case 1: R = 0 and S = 0

- In the first case, the inputs of both the NOR gates are Logic '0'. As neither of them are dominating inputs, they have no effect on the output. So, the output retains their previous states i.e., there is no change in the output. This condition is called as Hold Condition or No Change Condition.

Case 2: R = 0 and S = 1

- In this case, the 'S' input is 1, which means the output of the lower NOR Gate will become 0. As a result, both the inputs of upper NOR Gate become 0 and hence the output of the upper NOR Gate and thus the value of Q is 1 (HIGH). As '1' at input S makes the output to switch to one of its stable states and sets it to '1', the S input is known as SET input.

Case 3: R = 1 and S = 0

- In this case, the 'R' input is 1, which means the output of the upper NOR Gate will become 0 i.e., Q is 0 (LOW). As a result, both the inputs of lower NOR Gate become 0 and hence the output of the NOR Gate is 1 (HIGH). As '1' at input R makes the output to switch to one of its stable states and resets it to '0', the R input is known as RESET input.

Case 4: R = 1 and S = 1

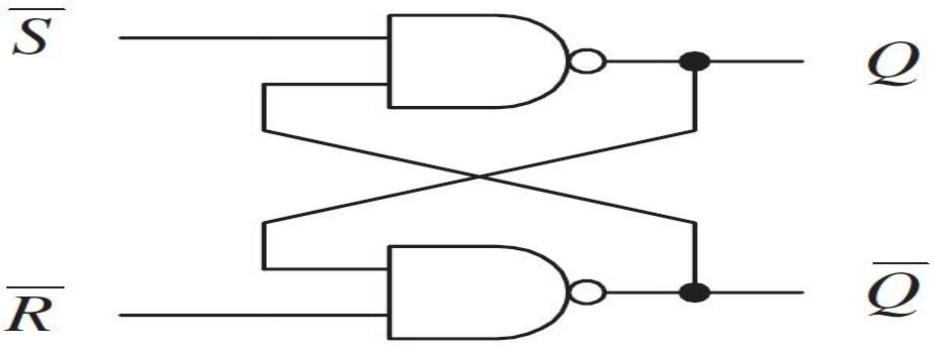
- This input condition is forbidden as it forces outputs of both NOR Gates to become 0, which is a violation of complementary outputs. Even if this input condition is applied, if the next inputs become R = 0 and S = 0 (hold condition), then it causes a 'race condition' between the NOR Gates, which causes an unstable or unpredictable state at the output. In normal operation, this condition is avoided by making sure that 1's are not applied to both inputs simultaneously.

Truth table for SR latch designed with NOR gates is shown below.

R	S	Q	State
0	0	Last State	No Change
0	1	1	Set
1	0	0	Reset
1	1	Not Applied (?)	Forbidden

SR latch using NAND gate

- An SR flip flop can also be designed by cross coupling of two NAND gates, but the Hold and Forbidden states are reversed. It is an active low input SR flip – flop and hence let us call it RS Flip-Flop. The circuit of SR flip – flop using NAND gates is shown in below figure.



- An important point about NAND gate is that its dominating input is 0 i.e., if any of its input is Logic '0', the output is Logic '1', irrespective of the other input. The output is 0, only if all the inputs are 1. With this in mind, let us see the working of a NAND based RS Flip-Flop.

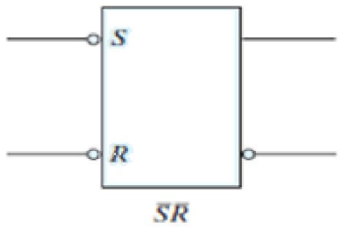
- Case 1: R = 1 and S = 1 When both the S and R inputs are HIGH, the output remains in previous state i.e., it holds the previous data.
- Case 2: R = 1 and S = 0
- When R input is HIGH and S input is LOW, the flip flop will be in SET state. As R is HIGH, the output of NAND gate B i.e., Q becomes LOW. This causes both the inputs of NAND gate A to become LOW and hence, the output of NAND gate A i.e., Q becomes HIGH.
- Case 3: R = 0 and S = 1

- When R input is LOW and S input is HIGH, the flip flop will be in RESET state. As S is HIGH, the output of NAND gate A i.e., Q becomes LOW. This causes both the inputs of NAND gate B to become LOW and hence, the output of NAND gate A i.e., Q becomes HIGH.
- Case 3: R = 0 and S = 0
- When both the R and S inputs are LOW, the flip flop will be in undefined state. Because the low inputs of S and R, violates the rule of flip – flop that the outputs should complement to each other. So, the flip flop is in undefined state (or forbidden state).

Truth table for SR latch designed with NAND gates is shown below.

R	S	Q	State
1	1	Last State	No Change
1	0	1	Set
0	1	0	Reset
0	0	Not Applied (?)	Forbidden

- The RS Flip-Flop using NAND gates can be converted to have a same truth table as a regular SR Flip-Flip by inverting the inputs. Instead of using inverters, we can use NAND gates with common input as shown in the following figure.
- In comparing the NAND with the NOR latch, note that the input signals for the NAND require the complement of those values used for the NOR latch. Because the NAND latch requires a 0 signal to change its state, it is sometimes referred to as an S'R' latch.

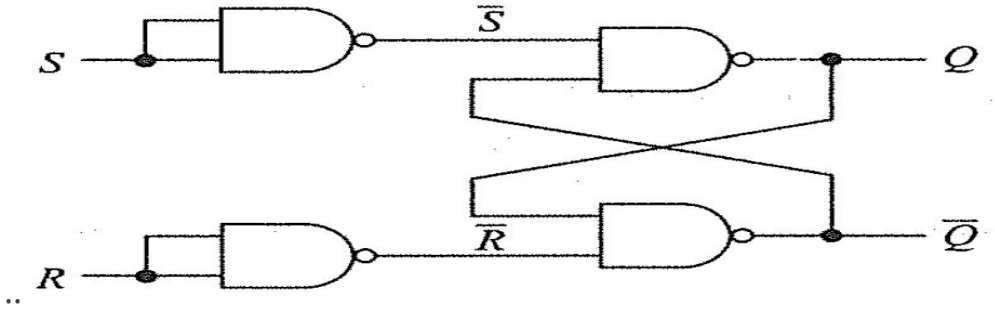


- The above represents the symbol for inverted SR latch or SR latch using NAND gate.

Truth table for SR latch using NAND gate or Inverted SR latch

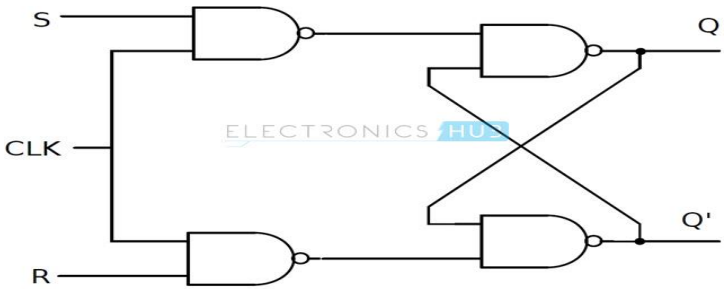
S	R	Q_{next}	Q'_{next}
0	0	Race	Race
0	1	0	1 (Reset)
1	0	1	0 (Set)
1	1	Q (No change)	Q' (No change)

3.3 Clocked SR, D, JK, T, MSJK Flip-Flop Symbol, Logic Circuit, Truth Table, and Applications



1. Clocked SR Flip – Flops

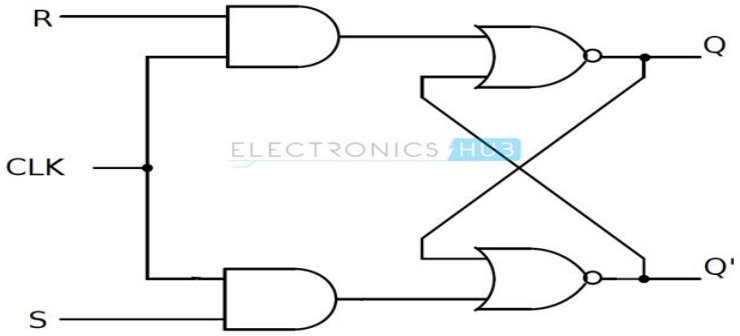
- Two types of clocked SR flip – flops are possible: based on NAND and based on NOR. The circuit of clocked SR flip – flop using NAND gates is shown below



- This circuit is formed by adding two NAND gates to NAND based SR flip – flop. The inputs are active high as the extra NAND gate inverts the inputs. A clock pulse is given as input to both the extra NAND gates. Hence the transition of the clock pulse is a key factor in functioning in this device. Assuming it is a positive edge triggered device, the truth table for this flip – flop is shown below.

Clock	R	S	Q	State
↓ or 0 or 1	X	X	Last State	No Change (Hold)
↑	0	0	Last State	No Change (Hold)
↑	0	1	1	Set
↑	1	0	0	Reset
↑	1	1	Not Applied (?)	Forbidden

- The same can be achieved by using NOR gates. The circuit of clocked SR flip – flop using NOR gates is shown below.



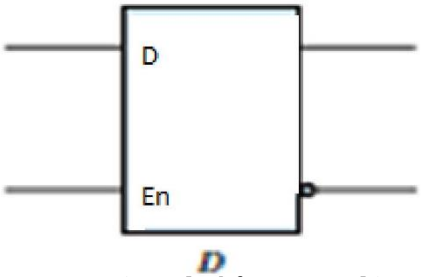
- The figure suggests a structure of RS flip – flop (as R is associated to the output Q), the functionality of SET and RESET remain the same i.e., when S is high, Q is set to 1 and when R is high, Q is reset to 0.

Applications

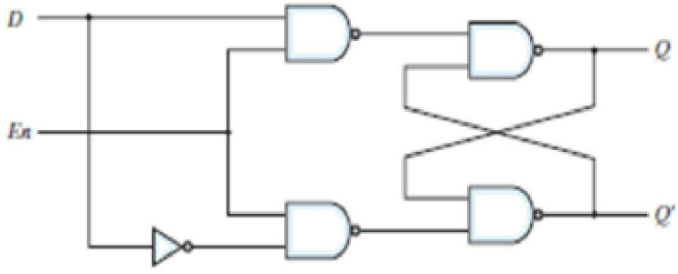
- SR flip – flops are very simple circuits but are not widely used in practical circuits because of their illegal state, where both S and R are high (S = R = 1). But they are used in switching circuits as they provide simple switching function (between Set and Reset).
- One such application is a Switch de-bounce circuit. The SR flip-flops are used to eliminate mechanical bounce of switches in digital circuits.

D-LATCH: -

- One way to eliminate the undesirable condition of the indeterminate state in the SR latch is to ensure that inputs S and R are never equal to 1 at the same time. this is done in the D latch. This latch has only two inputs: D (data) and En (enable).
- The D input goes directly to the S input, and its complement is applied to the R input.



(Symbol for D-Latch)



(a) Logic diagram

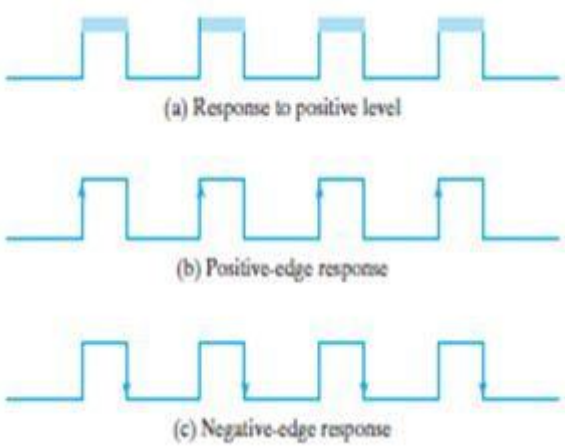
- As long as the enable input is at 0, the cross-coupled SR latch has both inputs at the 1 level and the circuit can't change state regardless of the value of D.
- The below represents the truth table for the D-latch.

En	D	Next State of Q
0	X	No change
1	0	Q=0;Reset State
1	1	Q=1;Set State

- The D input is sampled when En = 1. If D = 1, the Q output goes to 1, placing the circuit in the set state.
- If D = 0, output Q goes to 0, placing the circuit in the reset state. This situation provides a path from input D to the output, and for this reason, the circuit is often called a TRANSPARENT latch. It is used in data transfer.

TRIGGERING METHODS:-

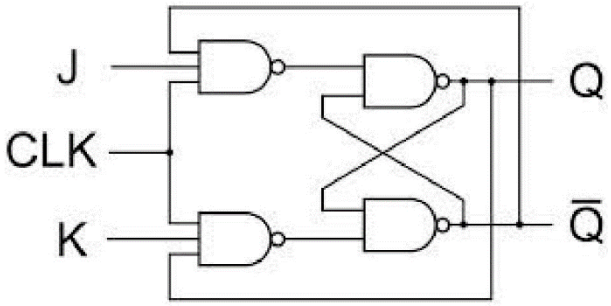
- The state of a latch or flip-flop is switched by a change in the control input. This momentary change is called a trigger, and the transition it causes is said to trigger the flip-flop.
- Flip-flop circuits are constructed in such a way as to make them operate properly when they are part of a sequential circuit that employs a common clock.
- This can be accomplished by eliminating the feedback path that is inherent in the operation of the sequential circuit using latches. A clock pulse goes through two transitions: from 0 to 1 and the return from 1 to 0.
- A ways that a latch can be modified to form a flip-flop is to produce a flip-flop that triggers only during a signal transition (from 0 to 1 or from 1 to 0) of the synchronizing signal (clock) and is disabled .



JK FLIP-FLOP:-

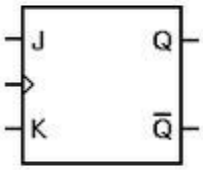
- The JK flip-flop can be constructed by using basic SR latch and a clock. In this case the outputs Q and Q' are returned back and connected to the inputs of NAND gates.
- This simple JK flip Flop is the most widely used of all the flip-flop designs and is considered to be a universal flip-flop circuit.
- The sequential operation of the JK flip flop is exactly the same as for the previous SR flip-flop with the same "Set" and "Reset" inputs.
- The "JK flip flop" has no invalid or forbidden input states of the SR Latch even when S and R are both at logic "1".

(The below diagram shows the circuit diagram of a JK flip-flop)



- Due to this additional clocked input, a JK flip-flop has four possible input combinations, "logic 1", "logic 0", "no change" and "toggle".
- The symbol for a JK flip flop is similar to that of an SR bistable latch except the clock input.

(Symbol of JK f/f)



- Both the S and R inputs of the SR is replaced by two inputs called the J & K inputs. , respectively after its inventor Jack and Kilby. Then this equates to: J = S and K = R.
- The two 2-input NAND gates of the gated SR have been replaced by two 3-input NAND gates with the third input of each gate connected to the outputs at Q and Q'.
- This cross coupling of the SR flip-flop allows the previously invalid condition of S = "1" and R = "1"
  state to be used to produce a "toggle action" as the two inputs are now interlocked.

(Truth table for JK flip- flop)

Input		Output		Comment
J	K	Q	Q_n	
0	0	0	0	No change
0	0	1	1	
0	1	0	0	Reset
0	1	1	0	
1	0	0	1	Set
1	0	1	1	
1	1	0	1	Toggle
1	1	1	0	

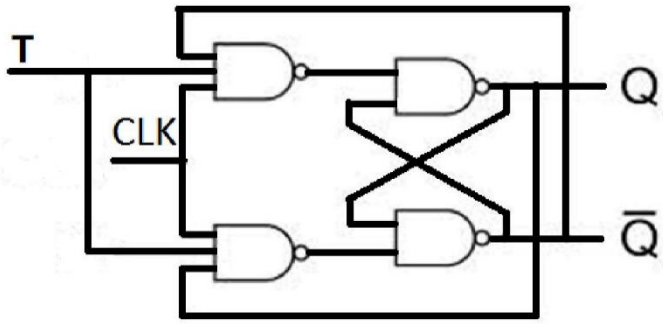
- When both inputs J and K are equal to logic "1", the JK flip flop toggles.

T FLIP-FLOP:-

- Toggle flip-flop or commonly known as T flip-flop.
- This flip-flop has the similar operation as that of the JK flip-flop with both the inputs J and K are shorted i.e. both are given the common input.

- Hence its truth table is same as that of JK flip-flop when J=K= 0 and J=K=1.So its truth table is as follows. It is used in counting nos.

T	Q	Q_{next}	Comment
0	0	0	No change
	1	1	
1	0	1	Toggles
	1	0	



CHARACTERISTIC TABLE:-

- A characteristic table defines the logical properties of a flip-flop by describing its operation in tabular form.
- The next state is defined as a function of the inputs and the present state.
- Q (t) refers to the present state and Q (t + 1) is the next.
- Thus, Q (t) denotes the state of the flip-flop immediately before the clock edge, and Q(t + 1) denotes the state that results from the clock transition.
- The characteristic table for the JK flip-flop shows that the next state is equal to the present state when inputs J and K are both equal to 0. This condition can be expressed as Q (t + 1) = Q (t), indicating that the clock produces no change of state.

CharacteristicTable Of JK Flip-Flop

J	K	Q(t+1)
0	0	Q(t) No change
0	1	0 Reset
1	0	1 Set
1	1	Q'(t) Complement

- When K = 1 and J = 0, the clock resets the flip-flop and Q(t + 1) = 0. With J = 1 and K = 0, the flip-flop sets and Q(t + 1) = 1. When both J and K are equal to 1, the next state changes to the complement of the present state, a transition that can be expressed as Q(t + 1) = Q'(t).
- The characteristic equation for JK flip-flop is represented as

$$Q(t+1) = JQ' + K'Q$$

CharacteristicTable of D Flip-Flop

D	Q(t+1)
0	0
1	1

- The next state of a D flip-flop is dependent only on the D input and is independent of the present state.
- This can be expressed as Q (t + 1) = D. It means that the next-state value is equal to the value of D.
- Note that the D flip-flop does not have a "no-change" condition and its characteristic equation is written as Q(t+1)=D.

CharacteristicTable of TFlip-Flop

T	Q(t+1)
0	Q(t) No change
1	Q'(t) Complement

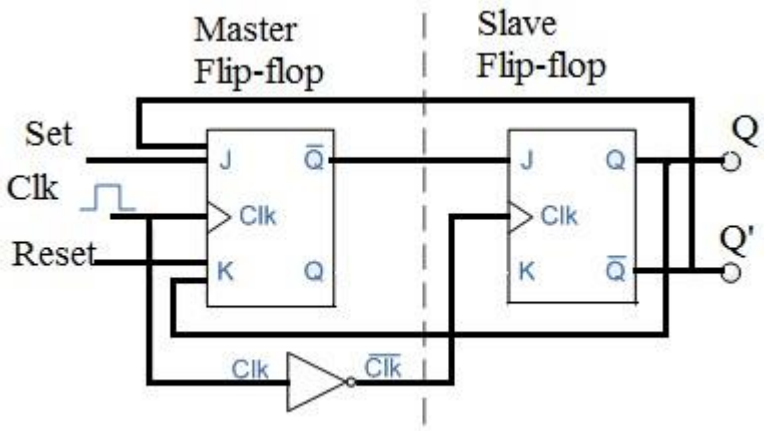
- The characteristic table of T flip-flop has only two conditions: When T = 0, the clock edge does not change the state; when T = 1, the clock edge complements the state of the flip-flop and the characteristic equation is

$$Q(t+1) = T \oplus Q = T'Q + TQ'$$

MASTER-SLAVE JK FLIP-FLOP:-

- The Master-Slave Flip-Flop is basically two gated SR flip-flops connected together in a series configuration with the slave having an inverted clock pulse.
- The outputs from Q and Q' from the "Slave" flip-flop are fed back to the inputs of the "Master" with the outputs of the "Master" flip flop being connected to the two inputs of the "Slave" flip flop.
- This feedback configuration from the slave's output to the master's input gives the characteristic toggle of the JK flip flop as shown below.

The Master-Slave JK Flip Flop



- The input signals J and K are connected to the gated "master" SR flip flop which "locks" the input condition while the clock (Clk) input is "HIGH" at logic level "1".
- As the clock input of the "slave" flip flop is the inverse (complement) of the "master" clock input, the "slave" SR flip flop does not toggle.When the clock is "LOW", the outputs from the "master" flip flop are latched and any additional changes to its inputs are ignored.
- The gated "slave" flip flop now responds to the state of its inputs passed over by the "master" section.
- Then on the "Low-to-High" transition of the clock pulse the inputs of the "master" flip flop are fed through to the gated inputs of the "slave" flip flop and on the "High-to-Low" transition the same inputs are reflected on the output of the "slave" making this type of flip flop edge or pulse-triggered.
- Then, the circuit accepts input data when the clock signal is "HIGH", and passes the data to the output on the falling-edge of the clock signal.
- In other words, the Master-Slave JK Flip flop is a "Synchronous" device as it only passes data with the timing of the clock signal.

3.4;-RACINGCONDITION;-

- In JK F/F when J=K=1, and clock =1 for a longer period of time, then Q output will toggle as long as CLK=1 HIGH, which makes the output of the f/f unstable or uncertain. This problem is called race around condition. In can be avoided by using MSJK F/F. So if tp $\leq \Delta t$. Lumped delay delay lines can be used in series with the feed back connection to avoid this. In MS JK f/f one is master & another is slave are connected in series to avoid race around condition.

Pre Set & Clear: -

- Pre Set & Clear:-In F/F ,when power is switched on, the state of circuit is not known. It may come to set (Q=1) or Reset (Q'=0) set.In many cases it is desired to initially set or reset the F/F is assigned. The preset(Pr) & clear (Cr) input may be applied at any time between clock pulse. If Pr=Cr=1 the ckt operates in accordance with T/T of SR f/f.

Probable short question with answer

1. Write down some application of clocked SR f/F
   Ans- Clocked SR f/F used in calculators, computers. It is widely used in morden electronic products.
2. Write down some application of D f/F
   Ans- It is used in temporary memory device. D flip flops are wired together to form shift registers& storage. Registers, which are commonly used in digital systems.
3. Define propagation delay time.
   Ans-The propagation delay time of the flip flop is defined as the time interval between the trigger edge & the stabilization of the output to a new state.
4. Give some basic flip flop application.
   Ans. The applications of flip flop are
   a) Parallel data storage
   b) Data transfer
   c) Frequency division &
   d) Counting & shift register
   e) Temporary memory storage device (D-f/f)
5. What is racing condition? (2014-S,2015-S,2016-S ,2019-S, 2020-W)
   Ans- A race condition or race hazard is the behavior of an electronic of software system where the output is dependent on the sequence or timing of other uncontrollable events. The name originates with the idea of two signals racing each other to influence the output of first.
   It can be avoided by using Master Slave Flip Flop.

Probable long question.

1. Explain the working of clocked RS F/F with truth table & timing diagram (2009-S, 2010-S2016-S,2020-W)

2. Distinguish between CLC & SLC (2007-S,2019-S&W)

3 Draw the diagram of D F/F & explain its working with function table (2019-S)

4 Draw the circuit of Master Slave JK flip flop. Explain it with functional table. (2019-S&W

5. What is racing? How it can be eliminated in a MSJK FF? Explain with suitable diagram. (2008-S,2007-S)

UNIT-4

REGISTERS,MEMORIES AND PLD

4.1;-SHIFT REGISTERS-SERIAL IN SERIAL OUT,SERIAL IN PARALLEL OUT,PISO (Parallel In Serial Out),PIPO (Parallel In Parallel Out)

4.2;-UNIVERSAL SHIFT REGISTERS- APPLICATIONS

4.3;-TYPES OF COUNTERS AND APPLICATIONS

4.4;- Binary counters Asynchronous ripple counter(UP/DOWN), Decade Counter, synchronous counter, & application.

4.5. Concept of Memories RAM.ROM, static RAM, dynamic RAM,PS RAM

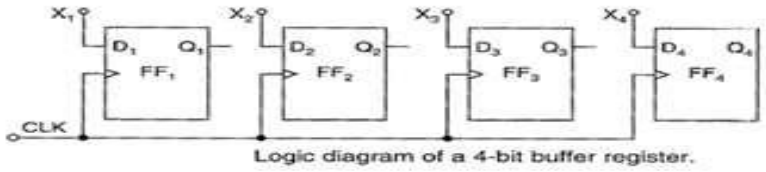
4.6: Basic Concepts of Programmable Logic Devices PLDs & application

4.1;-SHIFT REGISTERS-SERIAL IN SERIAL OUT,SERIAL IN PARALLEL OUT,PISO,PIPO ;-

- Registers are used for storage and transfer of binary information in a digital system.
- A register is mostly used for the purpose of storing and shifting binary data entered into
- It from an external source and has no characteristics internal sequence of states.
- The storage capacity of a register is defined as the number of bits of digital data, it can store or retain.

BUFFER REGISTER:-

- These are the simplest registers and are used for simply storing a binary word.
- These may be controlled by Controlled Buffer Register.
- D flip – flops are used for constructing a buffer register .
- The figure shown below is a 4- bit buffer register.



Logic diagram of a 4-bit buffer register.

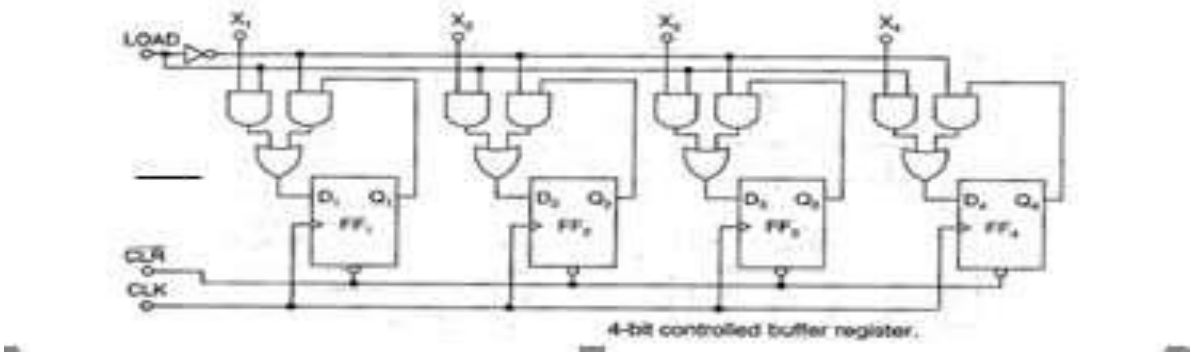
- The binary word to be stored is applied to the data terminals. When the clock pulse is applied, the output word becomes the same as that of input terminals, i.e. the input word is loaded into the register by the application of clk. When the positive clock edge arrives, the stored word becomes:

Q4 Q3 Q2 Q1= X4 X3 X2 X1 .

or

Q = X

.



4-bit controlled buffer register.

- If CLR goes LOW, all the flip-flops are RESET and the output becomes, Q = 0000.
- When CLR is HIGH, the register is ready for action. LOAD is control input.
- When LOAD is HIGH, the data bits X can reach the D inputs of FFs.
- At the positive going edge of the next clock pulse, the register is loaded, i.e.
- When LOAD is LOW, the X bits cannot reach the FFs. At the same time the inverted signal LOAD is HIGH. This forces each flip-flop output to feedback to its data input.
- Therefore data is circulated or retained as each clock pulse arrives.
- In other words, the content register remains unchanged in spite of the clock pulses.

CONTROLLED BUFFER REGISTER:-

- A number of FFs connected together such that data may be shifted into and shifted out of them is called a shift register.
- Data may be shifted into or out of the register either in serial form or in parallel form.
- There are four basic types of shift registers

1. Serial in, serial out
2. Serial in, parallel out
3. Parallel in, serial out
4. Parallel in , parallel out

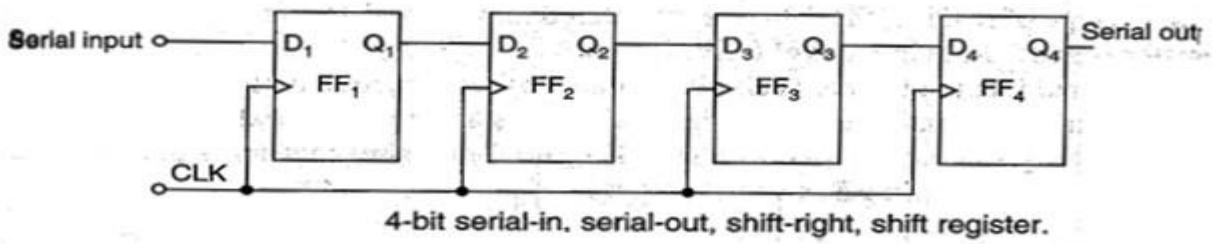
SERIAL IN, SERIAL OUTSHIFT REGISTER (SISO):-

- This type of shift register accepts data serially, i.e., one bit at a time and also outputs data serially.
- The logic diagram of a four bit serial in, serial out shift register is shown in below figure:
- In 4 stages i.e. with 4 FFs, the register can store upto 4 bits of data.
- Serial data is applied at the D input of the first FF. The Q output of the first FF is connected to the D input of the second FF, the output of the second FF is connected to the D input of the third FF and the Q output of the third FF is connected to the D input of the fourth FF. The data is outputted from the Q terminal of the last FF.

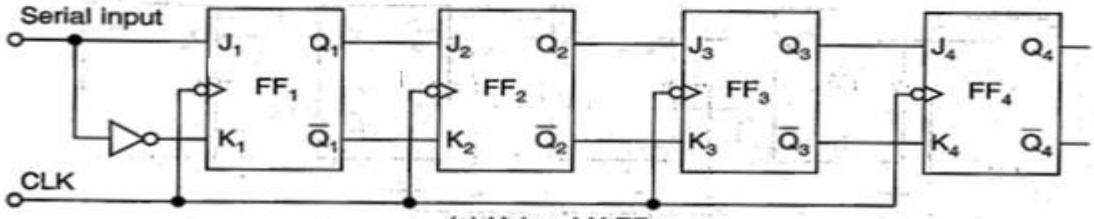
- When a serial data is transferred to a register, each new bit is clocked into the first FF at the positive going edge of each clock pulse.
- The bit that is previously stored by the first FF is transferred to the second FF.
- The bit that is stored by the second FF is transferred to the third FF, and so on.
- The bit that was stored by the last FF is shifted out.
- A shift register can also be constructed using J-K FFs or S-R FFs as shown in the figure below.
- Let data input bits are 1010. So after application of Clock pulser the output table will be

CP	Input	Q1	Q2	Q3	Q4
0	**0**	**0**	**0**	**0**	**0**
1	**0**	**0**	**0**	**0**	**0**
2	**1**	**1**	**0**	**0**	**0**
3	**0**	**0**	**1**	**0**	**0**
4	**1**	**1**	**0**	**1**	0
5	0	0	**1**	**0**	1
6	0	0	0	**1**	0
7	0	0	0	0	1

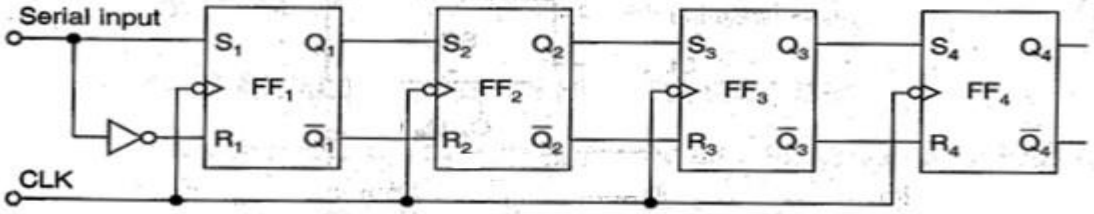
- After0 4 clock pulses data 1010 stored in respective FFs. But application of further 03 CPs bit are outs at FF Q4 . So total CPs will be N+(N-1) = (2N-1). If single CP timing will be t millisecond. Then total time required for N bit transmitted is (2N-1) t Msec.
- In case of sipo, total time will be nt millisecond. I.e. after n pulses all datas are loaded/outputed in ffs.



4-bit serial-in, serial-out, shift-right, shift register.



(a) Using J-K FFs



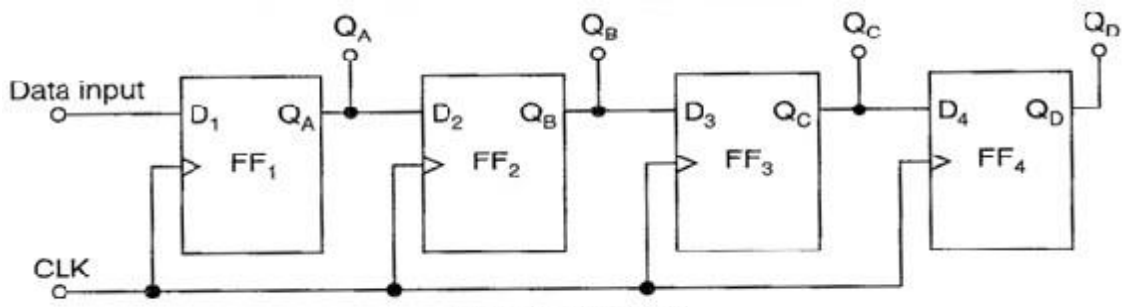
(b) Using S-R FFs

A 4-bit serial-in, serial-out, shift register.

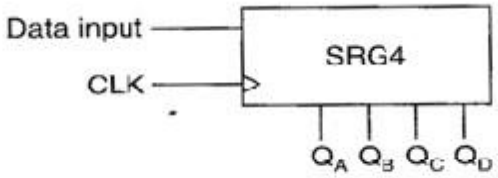
SERIAL IN,PARALLEL OUT SHIFT REGISTER (SIPO):-

- In this type of register, the data bits are entered & stored in the register serially but the data stored in the register is shifted out in the parallel form.
- When the data bits are stored once, each bits appears on its respective output line and all bits are available simultaneously, rather than bit – by – bit basis as in the serial output.
- The serial in, parallel out shift register can be used as a serial in, serial out shift register if the output is taken from the Q terminal of the last FF.
- The logic diagram and logic symbol of a 4 bit serial in, parallel out shift register is given below.

(A 4- bit serial in, parallel out shift register)



(a) Logic diagram

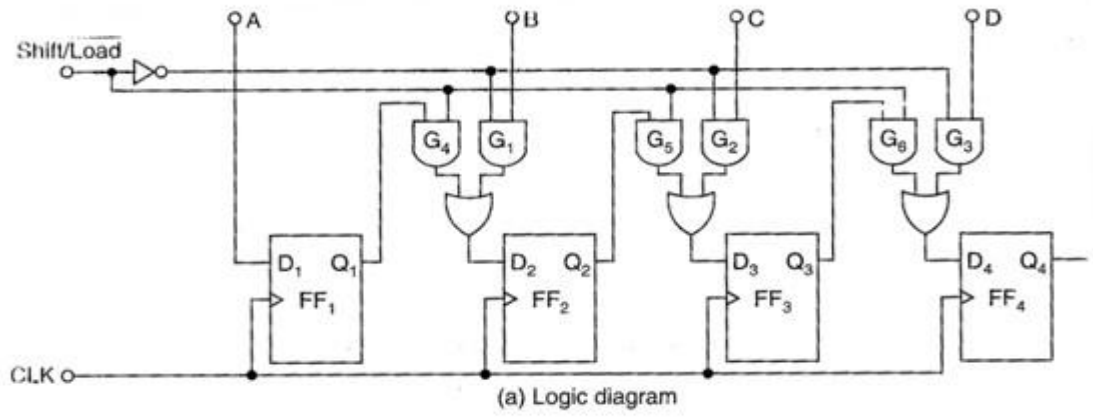


(b) Logic symbol

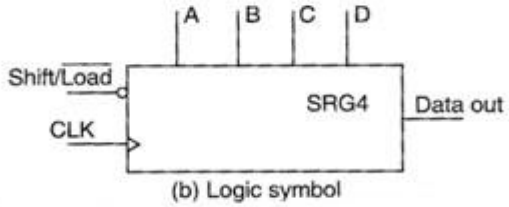
PARALLEL IN SERIAL OUTSHIFT REGISTER (PISO):-

- For parallel in, serial out shift register the data bits are entered simultaneously into their respective stages on parallel lines, but the data bits are transferred out of the register serially, i.e., on a bit by bit basis over a single line.
- The logic diagram and logic symbol of 4 bit parallel in, serial out shift register using D FFs is shown below.
- There are four data lines A, B, C and D through which the data is entered into the register in parallel form.
- The signal Shift /LOAD allows
  1. The data to be entered in parallel form into the register and
  2. The data to be shifted out serially from terminal Q_4.
- When Shift /LOAD line is HIGH, gates G1, G2, and G3 are disabled, but gates G4, G5 and G6 are enabled allowing the data bits to shift right from one stage to next.
- When Shift /LOAD line is LOW, gates G4, G5 and G6 are disabled, whereas gates G1, G2 and G3 are enabled allowing the data input to appear at the D inputs of the respective FFs.
- When clock pulse is applied, these data bits are shifted to the Q output terminals of the FFs and therefore the data is inputted in one step.
- The OR gate allows either the normal shifting operation or the parallel data entry depending on which AND gates are enabled by the level on the Shift /LOAD input.

(A 4- bit parallel in, serial out shift register)



(a) Logic diagram

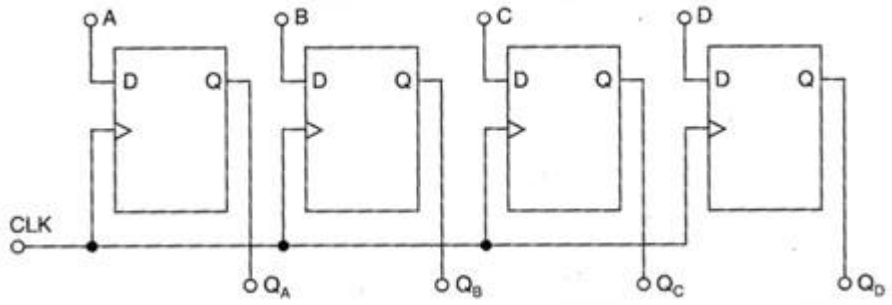


(b) Logic symbol

\\\\\\

PARALLELIN,PARALLELOUTSHIFT REGISTER (PIPO):-

- In a parallel in, parallel out shift register, the data entered into the register in parallel form and also the data taken out of the register in parallel form. Immediately following the simultaneous entry of all data bits appear on the parallel outputs.
- The figure shown below is a 4 bit parallel in parallel out shift register using D FFs.
- Data applied to the D input terminals of the FFs.
- When a clock pulse is applied at the positive edge of that pulse, the D inputs are shifted into the Q outputs of the FFs.
- The register now stores the data.
- The stored data is available instantaneously for shifting out in parallel form.



Logic diagram of a 4 – bit parallel in, parallel out shift register

4.2;-UNIVERSAL SHIFT REGISTERS- APPLICATIONS:-

- The register which has both shifts and parallel load capabilities, it is referred as a universal shift register. So, universal shift register is a bidirectional register, whose input can be either in serial form or in parallel form and whose output also can be either in serial form or parallel form.
- The universal shift register can be realized using multiplexers.
- The figure shows the logic diagram of a 4 bit universal shift register that has all the capabilities of a general shift register.

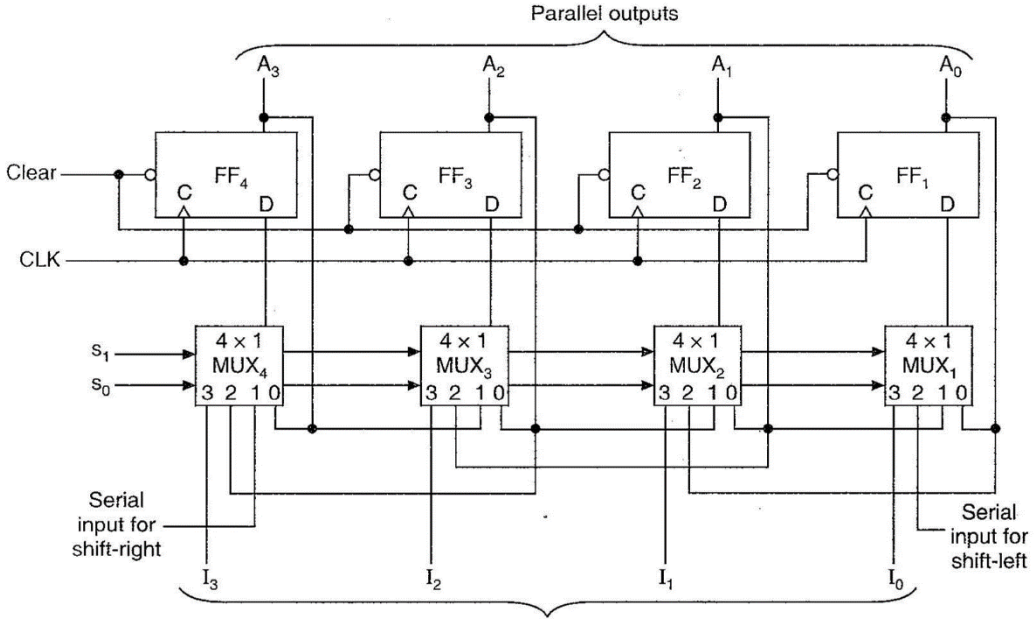


Fig- (a) 4 bit universal shift register

- It consists of four D flip- flops and four multiplexers.
- The four multiplexers have two common selection inputs S_1 and S_0.
- Input 0 in each multiplexer is selected when $S_1S_0 = 00$, input 1 is selected when $S_1S_0 = 01$, and input 2 is selected when $S_1S_0 = 10$ and input 3 is selected when $S_1S_0 = 11$.
- The selection inputs control the mode of operation of the register is according to the function entries shown in the table.
- When $S_1S_0 = 00$ the present value of the register is applied to the D inputs of flip-flops. This condition forms a path from the output of each FF into the input of the same FF.
- The next clock edge transfers into each FF the binary value it held previously, and no change of state occurs.
- When $S_1S_0 = 01$, terminal 1 of the multiplexer inputs have a path of the D inputs of the flip- flops.
- This causes a shift right operation, with serial input transferred into FF_4.
- When $S_1S_0 = 10$ a shift left operation results with the other serial input going into the FF_1.

 Finally when $S_1S_0 = 11$, the binary information on the parallel input lines is transferred into the register simultaneously during the next clock edge.

Functional table for the register of fig – a:

Mode control		
S_1	S_0	Register operation
0	0	No change
0	1	Shift right
1	0	Shift left
1	1	Parallel load

APPLICATIONS OF SHIFT REGISTERS:-

1. Shift registers are used for converting serial data to parallel form, so that a serial input can be processed by a parallel system and for converting parallel data to serial form, so that parallel data can be transmitted serially.
2. A serial in, parallel out shift register can be used to perform serial-to parallel conversion, and a parallel in, serial out shift register can be used to perform parallel- to –serial conversion.
3. A universal shift register can be used to perform both the serial- to – parallel and parallel-to- serial data conversion.
4. A bidirectional shift register can be used to reverse the order of data.
5. It is allso used in delay line.

4.3;-TYPES OF COUNTERS AND APPLICATIONS;-

- A counter is a device which stores (and sometimes displays) the number of times a particular event or process has occurred.
- There are different types of counters, viz.

  - Binary Counter
  - Asynchronous ripple counter (UP/DOWN)
  - Decade counter
  - Synchronous counter
  - Ring counter
  - Johnson counter
  - Cascaded counter

- Modulus counter.

4.4;- Binary counters Asynchronous ripple counter(UP/DOWN), Decade Counter, synchronous counter, & application.

- A binary counter is a hardware circuit that is made out of a series of flip-flops. The output of one flip-flop is sent to the input of the next flip-flop in the series.
- There are two **types of counters** based on the flip-flops that are connected in synchronous or not.

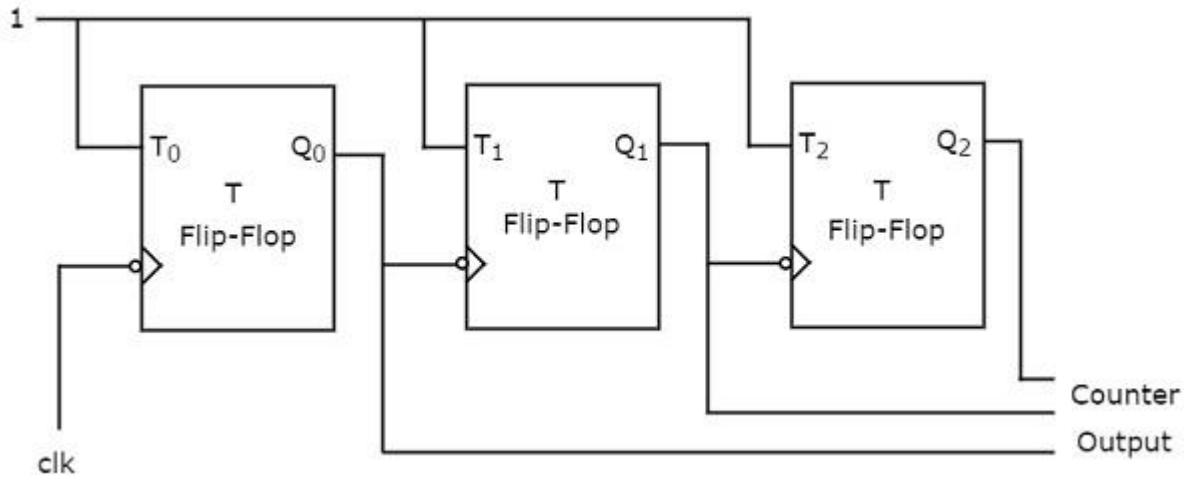
- Asynchronous counters
- Synchronous counters
- If the flip-flops do not receive the same clock signal, then that counter is called as **Asynchronous counter**. The output of system clock is applied as clock signal only to first flip-flop. The remaining flip-flops receive the clock signal from output of its previous stage flip-flop. Hence, the outputs of all flip-flops do not change affect at the same time
- But in the **Synchronous Counter**, the external clock signal is connected to the clock input of every individual flip-flop within the counter so that all of the flip-flops are clocked together simultaneously (in parallel) at the same time giving a fixed time relationship. In other words, changes in the output occur in "synchronisation" with the clock signal.
- **Ripple Counter:** Ripple counter is an Asynchronous counter. It got its name because the clock pulse ripples through the circuit.
- Asynchronous Counters

. Now, let us discuss the following two counters one by one.

- Asynchronous Binary up counter
- Asynchronous Binary down counter

Asynchronous Binary Up Counter

- An 'N' bit Asynchronous binary up counter consists of 'N' T flip-flops. It counts from 0 to $2^N - 1$. The **block diagram** of 3-bit Asynchronous binary up counter is shown in the following figure.



- The 3-bit Asynchronous binary up counter contains three T flip-flops and the T-input of all the flip-flops are connected to '1'. All these flip-flops are negative edge triggered but the outputs change asynchronously. The clock signal is directly applied to the first T flip-flop. So, the output of first T flip-flop **toggles** for every negative edge of clock signal.

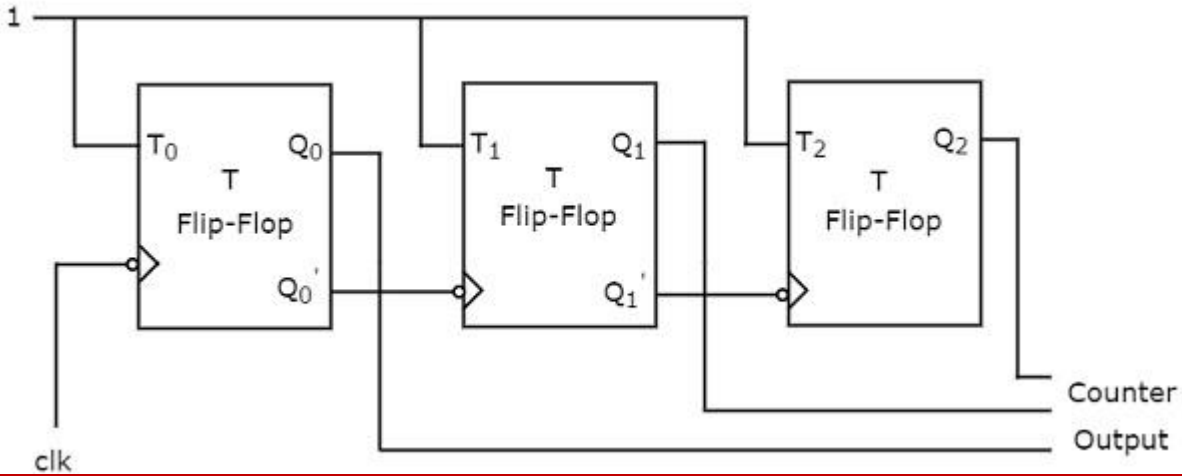
- The output of first T flip-flop is applied as clock signal for second T flip-flop. So, the output of second T flip-flop toggles for every negative edge of output of first T flip-flop. Similarly, the output of third T flip-flop toggles for every negative edge of output of second T flip-flop, since the output of second T flip-flop acts as the clock signal for third T flip-flop.
- Assume the initial status of T flip-flops from rightmost to leftmost is $Q_2Q_1Q_0=000$. Here, Q_2 & Q_0 are MSB & LSB respectively. We can understand the **working** of 3-bit asynchronous binary counter from the following table.

No of negative edge of Clock	Q_0 LSB	Q_1	Q_2 MSB
0	0	0	0
1	1	0	0
2	0	1	0
3	1	1	0
4	0	0	1
5	1	0	1
6	0	1	1
7	1	1	1

- Here Q_0 toggled for every negative edge of clock signal. Q_1 toggled for every Q_0 that goes from 1 to 0, otherwise remained in the previous state. Similarly, Q_2 toggled for every Q_1 that goes from 1 to 0, otherwise remained in the previous state.
- The initial status of the T flip-flops in the absence of clock signal is $Q_2Q_1Q_0=000$. This is incremented by one for every negative edge of clock signal and reached to maximum value at 7^{th} negative edge of clock signal. This pattern repeats when further negative edges of clock signal are applied.

Asynchronous Binary Down Counter

- An 'N' bit Asynchronous binary down counter consists of 'N' T flip-flops. It counts from $2^N - 1$ to 0. The **block diagram** of 3-bit Asynchronous binary down counter is shown in the following figure.



- The block diagram of 3-bit Asynchronous binary down counter is similar to the block diagram of 3-bit Asynchronous binary up counter. But, the only difference is that instead of connecting the normal outputs of one stage flip-flop as clock signal for next stage flip-flop, connect the **complemented outputs** of one stage flip-flop as clock signal for next stage flip-flop. Complemented output goes from 1 to 0 is same as the normal output goes from 0 to 1.
- Assume the initial status of T flip-flops from rightmost to leftmost is $Q_2Q_1Q_0=000$. Here, Q_2 & Q_0 are MSB & LSB respectively. We can understand the **working** of 3-bit asynchronous binary down counter from the following table.

No of negative edge of Clock	**Q_0**LSB	**Q_1**	**Q_2**MSB
0	0	0	0
1	1	1	1
2	0	1	1
3	1	0	1
4	0	0	1
5	1	1	0
6	0	1	0
7	1	0	0

- Here Q_0 toggled for every negative edge of clock signal. Q_1 toggled for every Q_0 that goes from 0 to 1, otherwise remained in the previous state. Similarly, Q_2 toggled for every Q_1 that goes from 0 to 1, otherwise remained in the previous state.
- The initial status of the T flip-flops in the absence of clock signal is $Q_2Q_1Q_0=000$. This is decremented by one for every negative edge of clock signal and reaches to the same value at 8^{th} negative edge of clock signal. This pattern repeats when further negative edges of clock signal are applied.

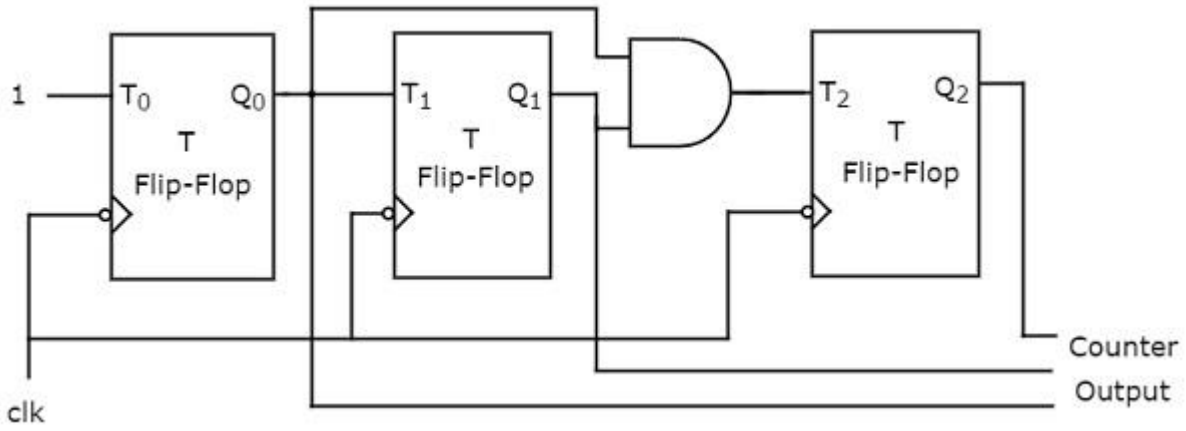
Synchronous Counters

- If all the flip-flops receive the same clock signal, then that counter is called as **Synchronous counter**. Hence, the outputs of all flip-flops change affect at the same time.
- Now, let us discuss the following two counters one by one.

- Synchronous Binary up counter
- Synchronous Binary down counter

Synchronous Binary Up Counter

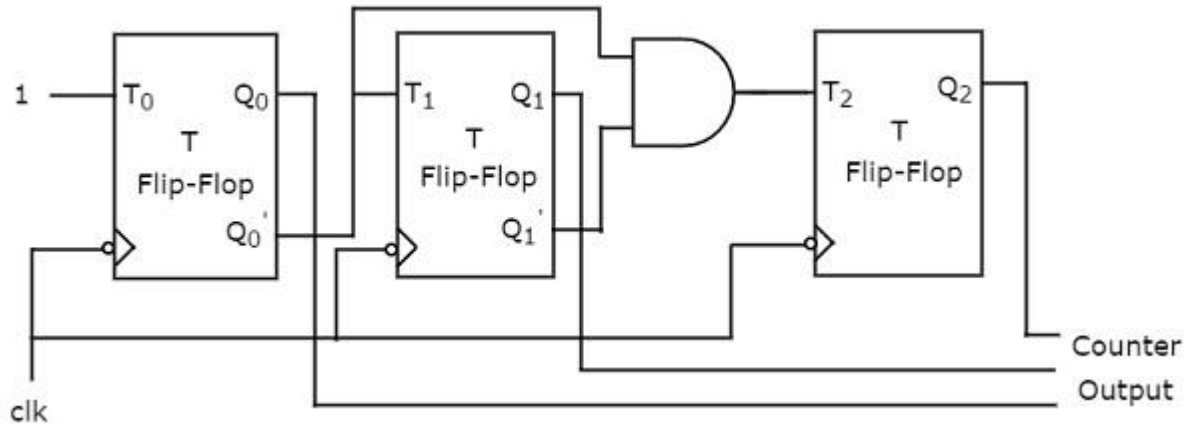
- An 'N' bit Synchronous binary up counter consists of 'N' T flip-flops. It counts from 0 to $2^N - 1$. The **block diagram** of 3-bit Synchronous binary up counter is shown in the following figure.



- The 3-bit Synchronous binary up counter contains three T flip-flops & one 2-input AND gate. All these flip-flops are negative edge triggered and the outputs of flip-flops change affect synchronously. The T inputs of first, second and third flip-flops are 1, Q_0 & Q_1Q_0 respectively.
- The output of first T flip-flop **toggles** for every negative edge of clock signal. The output of second T flip-flop toggles for every negative edge of clock signal if Q_0 is 1. The output of third T flip-flop toggles for every negative edge of clock signal if both Q_0 & Q_1 are 1.

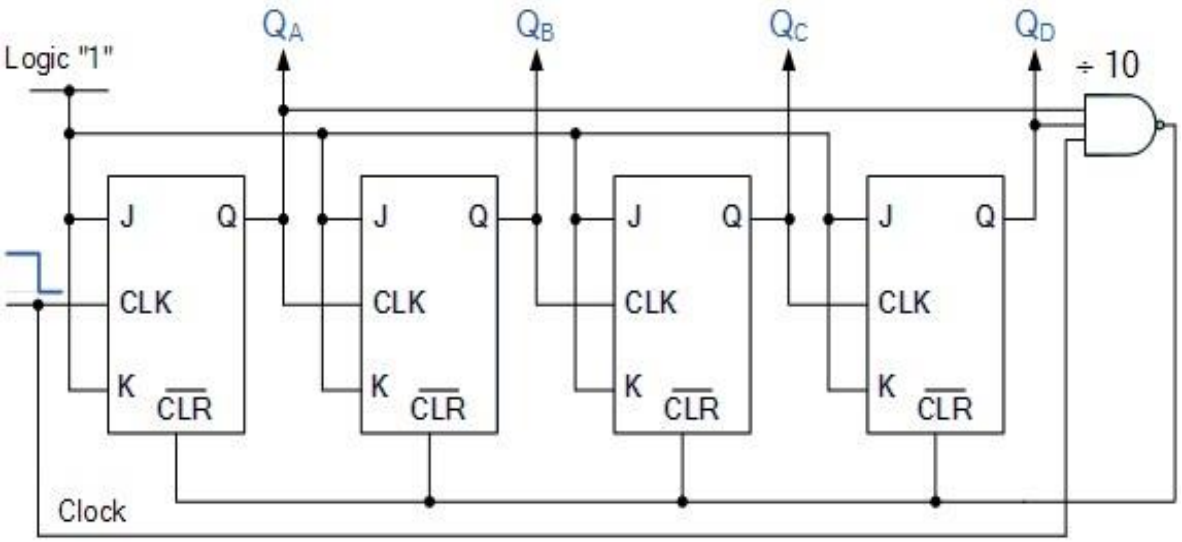
Synchronous Binary Down Counter

- An 'N' bit Synchronous binary down counter consists of 'N' T flip-flops. It counts from $2^N - 1$ to 0. The **block diagram** of 3-bit Synchronous binary down counter is shown in the following figure.



- The 3-bit Synchronous binary down counter contains three T flip-flops & one 2-input AND gate. All these flip-flops are negative edge triggered and the outputs of flip-flops change affect synchronously. The T inputs of first, second and third flip-flops are 1, $Q_{0'}$ &' $Q_{1'}Q_{0'}$ respectively.
- The output of first T flip-flop **toggles** for every negative edge of clock signal. The output of second T flip-flop toggles for every negative edge of clock signal if $Q_{0'}$ is 1. The output of third T flip-flop toggles for every negative edge of clock signal if both $Q_{1'}$ & $Q_{0'}$ are 1.

Asynchronous DecadeCounter



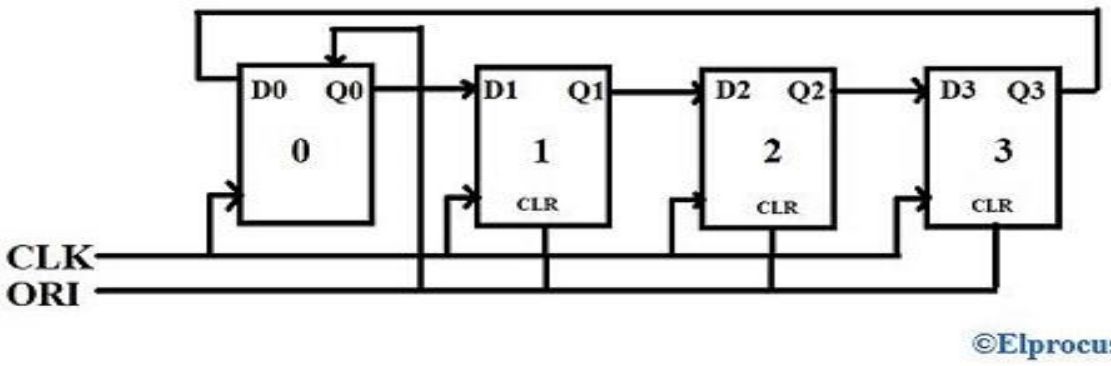
- A decade counter can count from BCD "0" to BCD "9".
- A decade counter requires resetting to zero when the output count reaches the decimal value of 10, ie. when DCBA = 1010 and this condition is fed back to the reset input.
- A counter with a count sequence from binary "0000" (BCD = "0") through to "1001" (BCD = "9") is generally referred to as a BCD binary-coded-decimal counter because its ten state sequence is that of a BCD code but binary decade counters are more common.
- This type of asynchronous counter counts upwards on each leading edge of the input clock signal starting from 0000 until it reaches an output 1001 (decimal 9).
- Both outputs Q_A and Q_D are now equal to logic "1" and the output from the NAND gate changes state from logic "1" to a logic "0" level and whose output is also connected to the CLEAR (CLR) inputs of all the J-K Flip-flops.
- This signal causes all of the Q outputs to be reset back to binary 0000 on the count of 10. Once QA and QD are both equal to logic "0" the output of the NAND gate returns back to a logic level "1" and the counter restarts again from 0000. We now have a decade or Modulo-10 counter.

Decade Counter Truth Table

Clock Count	Output bit Pattern				Decimal Value
	QD	QC	QB	QA	
1	0	0	0	0	0
2	0	0	0	1	1
3	0	0	1	0	2
4	0	0	1	1	3
5	0	1	0	0	4
6	0	1	0	1	5
7	0	1	1	0	6
8	0	1	1	1	7
9	1	0	0	0	8
10	1	0	0	1	9
11	Counter Resets its Outputs back to Zero				

Ring Counter

- A ring counter is also known as SISO shift register counter, where the output of the last flip flop is connected to the input of the1st flip flop which acts as a ring counter. The designing of the ring counter can be done by using four D-Flip Flops with a common clock signal and overriding input can be connected to pre-set and clear.



(Block-diagram-of-ring-counter)

- Pre-set or Clear: The main function of this is if the input clock signal changes, then the output value is also changed.
- One input is connected to the first flip-flop ff0-Q0, & another input is connected to CLR of the other three flip flops like ff1, ff2, ff3.
- For example, let us take a condition where pre-set = '0000' then the outputs obtained at each flip flop is as follows. For FF0, the output at Q0 is '1', whereas in other flipflops like ff, ff2, ff3 (which are connected to clear where CLR = 0) the outputs obtained at Q1 = Q2 = Q3 ='0′. This can be understood by following the truth table and its output waveforms obtained when executed using Verilog HDL code .

Truth Table

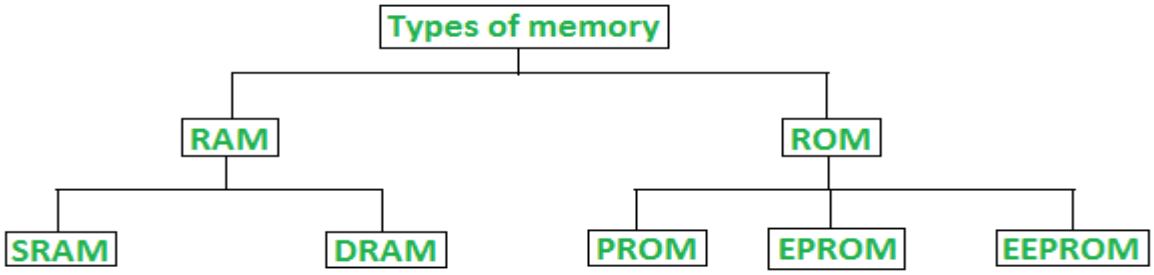
ORI	CLK	Q0	Q1	Q2	Q3
Low Pulse	X	1	0	0	0

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

- Where Inputs = ORI and CLK & X = Clock can be either a positive or a negative edge. Outputs = Q0,Q1, Q2, Q3.

4.5. Concept of Memories - RAM.ROM, stactic RAM,dynamic AM,PS RAM

- A memory unit is a collection of cells capable of storing a large quantity of binary information. Binary information received from an input device is stored in memory & information transferred to an output device is taken from memory. Classification of memory is as under-



Classification of computer memory

- **Random-access memory** (**RAM**) is a form of computer memory that can be read and changed in any order, typically used to store working data and machine code. A random-access memory device allows data items to be read or written in almost the same amount of time irrespective of the physical location of data inside the memory, in contrast with other direct-access data storage media (such as hard disks, CD-RWs, DVD-RWs and the older magnetic tapes and drum memory),
- RAM contains multiplexing and demultiplexing circuitry, to connect the data lines to the addressed storage for reading or writing the entry.]
- The two main types of volatile random-access semiconductor memory are static random-access memory (SRAM) and dynamic random-access memory (DRAM). .

DRAM	SRAM
1. Constructed of tiny capacitors that leak electricity.	1.Constructed of circuits similar to D flip-flops.
2.Requires a recharge every few milliseconds to maintain its data.	2.Holds its contents as long as power is available.
3.Inexpensive.	3.Expensive.
4. Slower than SRAM.	4. Faster than DRAM.
5. Can store many bits per chip.	5. Can not store many bits per chip.
6. Uses less power.	6.Uses more power.
7.Generates less heat.	7.Generates more heat.
8. Used for main memory.	8. Used for cache.

Difference between SRAM and DRAM

(PS RAM)pseudostatic (random-access) memory

- A combinational form of a dynamic RAM that incorporates various refresh and control circuits on-chip (e.g., refresh address counter and multiplexer, interval timer, arbiter). These circuits allow the PSRAM operating characteristics to closely resemble those of an SRAM.
- A random-access memory whose internal structure is a dynamic memory with refresh control signals generated internally, in the standby mode, so that it can mimic the function of a static memory.
- **Read-only memory** (**ROM**) is a type of non-volatile memory used in computers and other electronic devices. Data stored in ROM cannot be electronically modified after the manufacture of the memory device. Read-only memory is useful for storing software that is rarely changed during the life of the system,
- *Read-only memory* strictly refers to memory that is hard-wired, such as diode matrix or a mask ROM integrated circuit (IC), which cannot be electronically changed after manufacture. ICs cannot. Correction of errors, or updates to the software, require new devices to be manufactured and to replace the installed device.
- Floating-gate ROM semiconductor memory in the form of erasable programmable read-only memory (EPROM), electrically erasable programmable read-only memory (EEPROM) and flash memory can be erased and re-programmed. But usually, this can only be done at relatively slow speeds, may require special equipment to achieve, and is typically only possible a certain number of times.

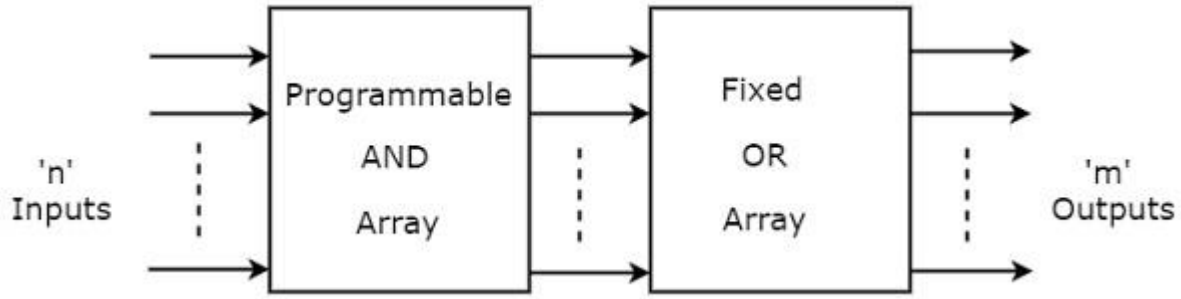
4.6: Basic Concepts of Programmable Logic Devices PLDs & application:

- PLDs are the integrated circuits. They contain an array of AND gates & another array of OR gates. There are three kinds of PLDs based on the type of arrays, which has programmable feature.

- Programmable Read Only Memory
- Programmable Array Logic
- Programmable Logic Array

- The process of entering the information into these devices is known as **programming**. Basically, users can program these devices or ICs electrically in order to implement the Boolean functions based on the requirement. Here, the term programming refers to hardware programming but not software programming.
- If the ROM has programmable feature, then it is called as **Programmable ROM** PROM. The user has the flexibility to program the binary information electrically once by using PROM programmer.
- PROM is a programmable logic device that has fixed AND array & Programmable OR array.

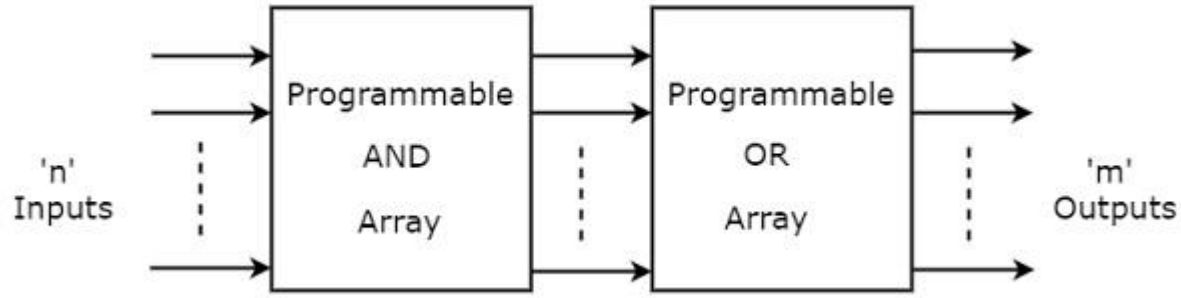
PAL is a programmable logic device that has Programmable AND array & fixed OR array. The advantage of PAL is that we can generate only the required product terms of Boolean function instead of generating all the min terms by using programmable AND gates. The **block diagram** of PAL is shown in the following figure.



- Here, the inputs of AND gates are programmable. That means each AND gate has both normal and complemented inputs of variables. So, based on the requirement, we can program any of those inputs. So, we can generate only the required **product terms** by using these AND gates.
- Here, the inputs of OR gates are not of programmable type. So, the number of inputs to each OR gate will be of fixed type. Hence, apply those required product terms to each OR gate as inputs. Therefore, the outputs of PAL will be in the form of **sum of products form**.

Programmable Logic Array PLA

PLA is a programmable logic device that has both Programmable AND array & Programmable OR array. Hence, it is the most flexible PLD. The **block diagram** of PLA is shown in the following figure.



Here, the inputs of AND gates are programmable. That means each AND gate has both normal and complemented inputs of variables. So, based on the requirement, we can program any of those inputs. So, we can generate only the required **product terms** by using these AND gates.

Here, the inputs of OR gates are also programmable. So, we can program any number of required product terms, since all the outputs of AND gates are applied as inputs to each OR gate. Therefore, the outputs of PAL will be in the form of **sum of products form**.

Probable Short questions with answers

1) List the types of Shift Register (W-2020)

Ans- SIPO (Serial In Parallel Out),

SISO (serial In Serial Out)

PIPO (Parallel In Parallel Out) &

PISO (Parallel In Serial Out

2) What is universal Register ? (W-2020)

Ans- A Register which has both shifts left & right & parallel load capabilities is referred to as a universal shift register. These are used in memory elements in computers.

3) What is modulus of a counter? (W-2019, W-2020, S-2014, 2009,2007)

Ans- The modulus or MOD of a counter is equal to the number of states that the counter goes through in each complete cycle, before it recycles back to its starting state.MOD no= 2^n where n is the no of flip flops used in counters.

4) What is the basic difference between a register & a counter ? (W-2020)

Ans- The basic difference between counter & register is that registers have no specified sequence of states but the counters have specified sequence.

Probable Long questions-

1) Explain the basic operation of PISO shift register with the help of suitable logic diagram.

(S-2014)

2) With neat sketches explain the working of PIPO & SIPO Register. (S-2016,S-2019)

3) Explain the working of SISO & PISO register with the help of suitable logic diagram.(W-2020)

4) Design the working of a 4 bit ripple counter with truth table & timing using D – flip flops.

(S-2010,S-2014 , S-2015,S-2016,W-2019)

5) Differentiate between Asynchronous & Synchronous counter. (S-2014)

UNIT-5;-
D/A and A/D Converters

Learning Objectives:

5.1;- NECESSITY OF A/D & D/A CONVERTERS

5.2-D/A CONVERSION USING WEIGHTED REGISTER METHODS

5.3;- D/A CONVERSION USING R-2 R LADDER Binary Ladder Network

5.4;- A/D/C USING COUNTER METHOD

5.5 A/D CONVERSION USING SUCCESSIVE APPROXIMATION METHOD

5.1;- NECESSITY OF A/D & D/A CONVERTERS

- In an analog signal, the quantity can vary over a range of values i.e. speaker in radio receiver, speedometer, telephone system etc. However, in digital system the quantity can take only discreet values i.e. calculators, digital watch, typewriters, digital computers etc.

- Digital signals have greater accuracy and precision, can handle by simply adding more switching circuits. But, in analog system the precision is usually limited to 3 to 4 digits because the values of voltage and current depends on circuit components.

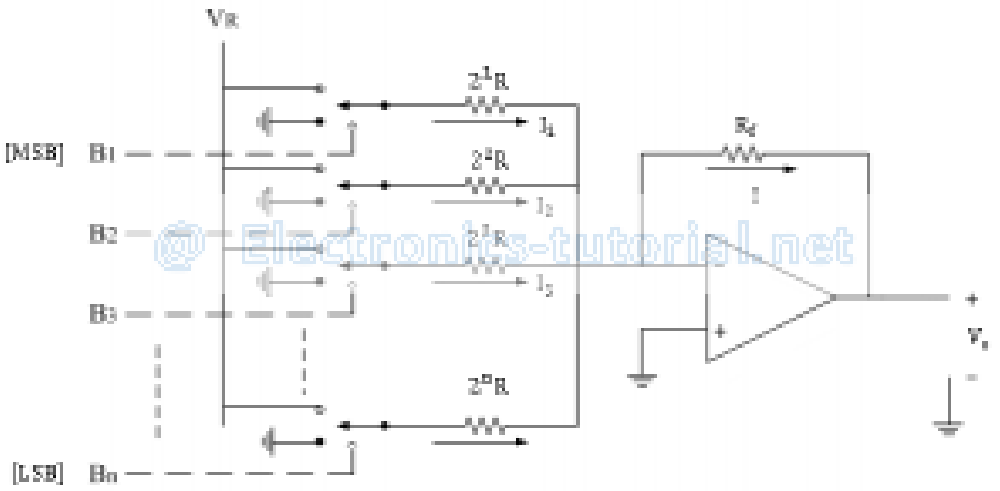
- Analog signals can be programmed but the variety and complexity is severally limited. Digital circuits are less affected by noise, and fluctuations.

- Analog's techniques are simpler and more economical but the digital techniques are complex and expensive.

- In recent times both analog and digital techniques are used in the same system to bring in advantages of both techniques. However, going forward, digital technique is expected to override analog techniques owing to the economies of scale that may come into play.

5.2-D/A CONVERSION USING WEIGHTED REGISTER METHODS

- In the weighted resistor type DAC, each digital level is converted into an equivalent analog voltage or current. The following figure shows the circuit diagram of the binary weighted resistor type DAC.



- It consists of parallel binary weighted resistor bank and a feedback resistor Rf. The switch positions decide the binary word (i.e.B1B2B3…Bn). In the circuit op-amp is used as current to voltage converter. When the switches are closed the respective currents are flowing through resistors as shown in the circuit diagram above. Since input current to the op-amp is zero, the addition current flows through feedback resistor. $\therefore I=I_1+I_2+I_3+\ldots\ldots\ldots\ldots+In$

  The inverting terminal of op-amp is virtually at ground potential.

$$\therefore V_o=-IR_f$$

$$\therefore V_o=-[I_1+I_2+I_3+\ldots\ldots\ldots\ldots+I_n]R_f$$

$$\therefore V_o=-\left[B_1\frac{V_R}{2^1R}+B_2\frac{V_R}{2^2R}+B_3\frac{V_R}{2^3R}+\ldots\ldots+B_n\frac{V_R}{2^nR}\right]R_f$$

$$V_o=-\frac{R_f}{R}V_R[B_1.2^{-1}+B_2.2^{-2}+B_3..2^{-3}+\ldots\ldots\ldots+B_n..2^{-n}]$$

$$If R_f=R$$

$$\therefore V_o=-V_R[B_1.2^{-1}+B_2.2^{-2}+B_3..2^{-3}+\ldots\ldots\ldots+B_n..2^{-n}]$$

Consider the example of 3-bit DAC. When input binary sequence is B1 B2 B3 = 001

$$\therefore V_o=-V_R[B_1.2^{-1}+B_2.2^{-2}+B_3.2^{-3}]$$

$$\therefore V_o=-V_R[0.2^{-1}+0.2^{-2}+1.2^{-3}]$$

$$\therefore V_o=-V_R[0+0+\frac{1}{8}]$$

$$\therefore V_o=-\frac{V_R}{8}$$

When input binary sequence is B1 B2 B3 = 101

$$\therefore \quad V_o = -V_R[B_1.2^{-1} + B_2.2^{-2} + B_3.2^{-3}]$$

$$\therefore \quad V_o = -V_R[1.2^{-1} + 0.2^{-2} + 1.2^{-3}]$$

$$\therefore \quad V_o = -V_R\left[\frac{1}{2} + 0 + \frac{1}{8}\right]$$

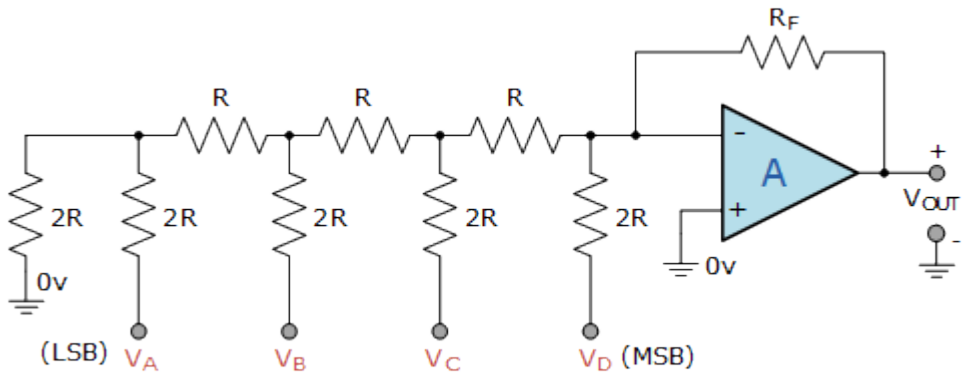
$$\therefore \quad V_o = -\frac{5V_R}{8}$$

For input 100 output will be Vo= $-4V_R/8$. For input 1001, output will be $-9V_R/16$

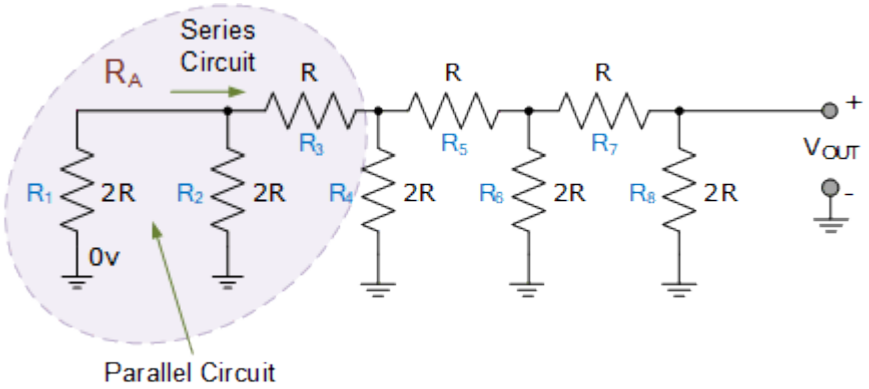
Disadvantages:

1) When number of binary input increases, it is not easy to maintain the resistance ratio.
2) Very wide ranges of different values of resistors are required. For high accuracy of conversion, the values of resistances must be accurate.
3) Different current flows through resistors, so their wattage ratings are also different.
4) Since 'R' is very large, op-amp bias currents gives a drop which offsets output.
6) Resistances of switches may be comparable with smallest resistor.

5.3;- D/A CONVERSION USING R-2 R LADDER Binary Ladder Network



- Starting from the left hand side and using the simplified equation for two parallel resistors and series resistors, we can find the equivalent resistance of the ladder network as:



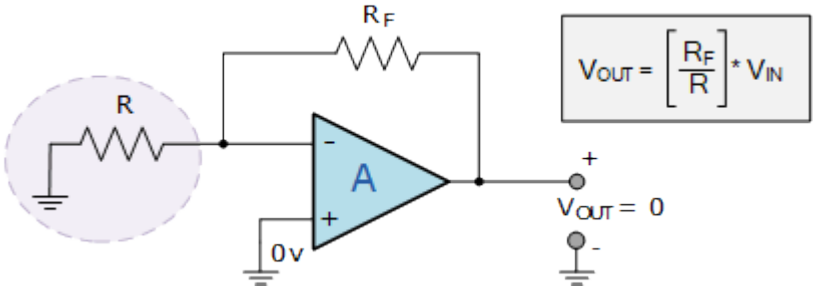
- Resistors R_1 and R_2 are in "parallel" with each other but in "series" with resistor R_3. Then we can find the equivalent resistance of these three resistors and call it R_A for simplicity (or any other form of identification you want).

$$R_A = R_3 + \frac{R_1 \times R_2}{R_1 + R_2} = R + \frac{2R \times 2R}{2R + 2R} = R + R = 2R$$

- Again we can find the equivalent resistance of this combination and call it R_B.

$$R_B = R_5 + \frac{R_A \times R_4}{R_A + R_4} = R + \frac{2R \times 2R}{2R + 2R} = R + R = 2R$$

- So R_B combination is equivalent to "2R". Hopefully we can see that this equivalent resistance R_B is in parallel with R_6 with the parallel combination in series with R_7 as shown.



- So output of an inverting OPAMP will be Vout=-(Rf/R*)Vin.

- So Generalized R-2R DAC Equation

$$V_{OUT} = \frac{V_A + 2V_B + 4V_C + 8V_D + 16V_E + 32V_F + \ldots etc}{2^n}$$

- Where: "n" represents the number of digital inputs.
- So using our equation from above, the output voltage for a binary code of 1011_2 is calculated as:

$$V_{OUT} = \frac{1V_A + 2V_B + 4V_C + 8V_D}{2^n}$$

$$V_{OUT} = \frac{1\times10 + 2\times10 + 4\times0 + 8\times10}{16}$$

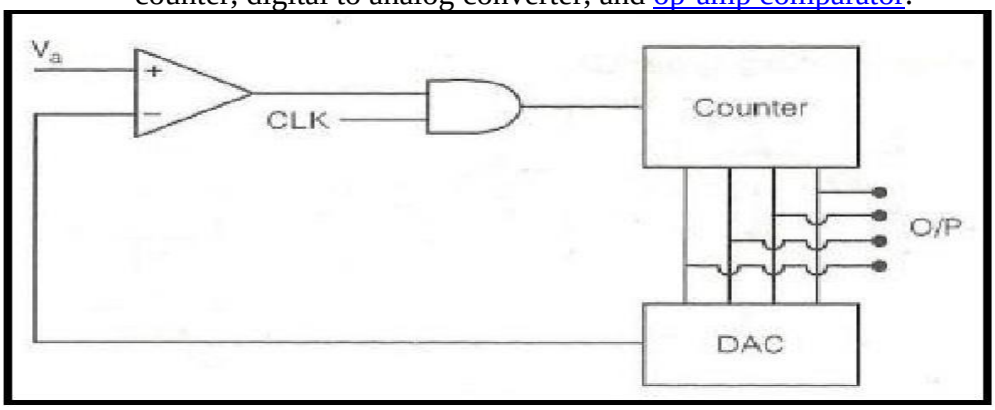
$$V_{OUT} = \frac{110}{16} = 6.875 \text{ Volts}$$

- Therefore, the analogue output voltage used to control the DC motor when the input code is 1011_2 is calculated as: -6.875 volts. Note that the output voltage is negative due to the inverting input of the operational amplifier.

- The resolution of the converter will be equal to the value of the least significant bit (LSB) which is given as: $V_{LSB} = V_{IN}/2^n$.

5.4. A/D conversion using counter Method -

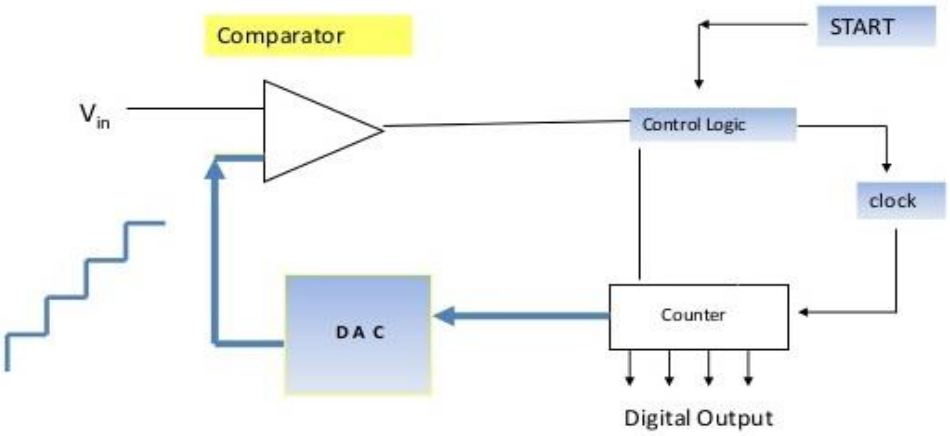
- The Counter type is the basic type of ADC, which is also known as staircase approximation ADC, or a ramp type ADC. The circuit diagram of counter type ADC is shown below. The circuit diagram of Counter Type ADC can be built with N-bit counter, digital to analog converter, and op-amp comparator.



C

Counter Type ADC

- The N-bit counter produces an n-bit digital o/p which is given as an i/p to the digital to analog circuit (DAC). The analog output equivalent to the digital i/p from DAC is contrasted with the i/p analog voltage with the help of an op-amp comparator. This Integrated Circuit evaluates the two voltages and if the produced DAC voltage is low, it gives a high pulse to the N-bit counter as a CLK pulse to raise the counter.



(Counter Type ADC)

- The similar procedure will be continued until the output of the DAC equals to the i/p analog voltage then it produces a low CLK pulse and also gives a clear signal to the counter as well as a load signal to the storage resistor to store the corresponding digital bits. These digital values are strongly matched with the analog input values with a small error.
- For each sampling interval, the output of DAC tracks a rampway so that it is named as a Digital ramp kind ADC. And this ramp seems like staircases for each sampling moment, so that it is also named as a staircase approximation kind ADC.

5.4;- A/D/C USING COUNTER METHOD

- This converter is also called digital ramp or the counter type A/D converter. Figure shows the configuration for 8 bit converter. As seen in figure it uses a D/A converter and a binary counter to produce the digital number corresponding to analog input. The main components are comparator, AND gate, D/A converter, divide by 256 counter and latches. The analog input is given to non-inverting terminal of comparator. The D/A converter provides stair step reference voltage.

- Let he counter be in reset state and output of D/A converter be zero. An analog input is given to non-inverting terminal of comparator. Since the reference input is 0, the comparator gives High output and enables the AND gate. The clock pulses cause advancing of counter through its binary states and stair step reference voltage is produced from D/A converter. As the counter keeps advancing, successively higher stair step output voltage is produced. When this stair step voltage reaches the level of analog input voltage, the comparator output goes Low and disables the AND gate. The clock pulses are cut off and counter stops. The state of counter at this point is equal to the number of steps in reference voltage at which comparison occurs. The binary number corresponding to this number of steps is the value of the analog input voltage. The control logic causes this binary number to be loaded into the latches and counter is reset.
- This converter is rather slow in action because the counter has to pass through the maximum number of states before a conversion takes place. For 8 bit device this means 256 counter states

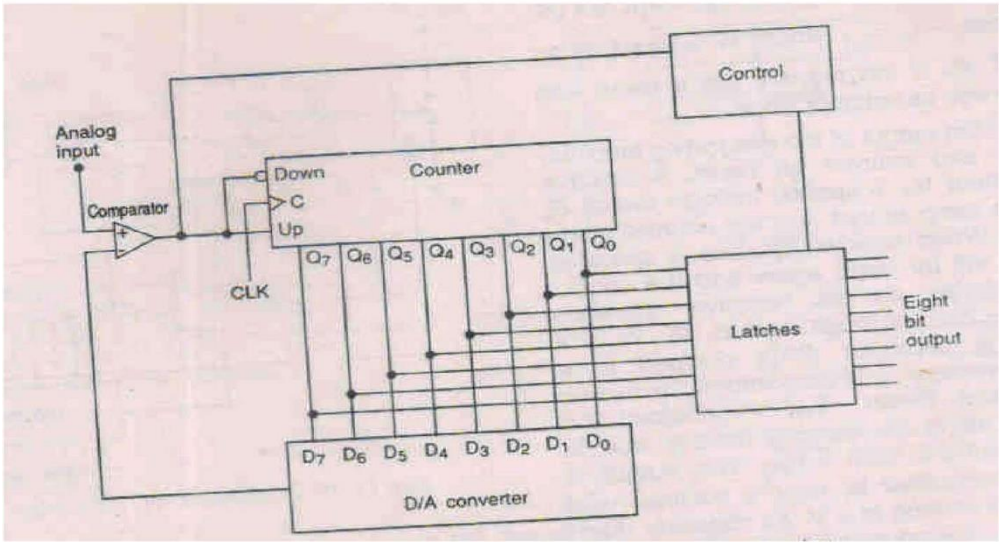
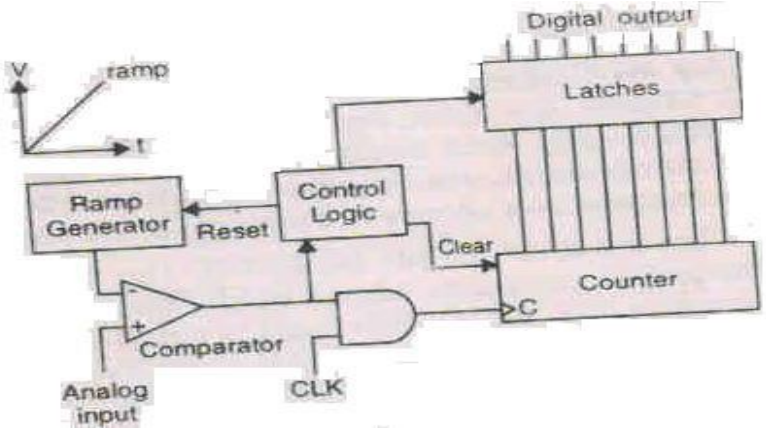


Fig (a) 8 bit up-down counter type A/D converter



Single Slope A/D converter

5.5 A/D Conversion using Successive Approximation method –

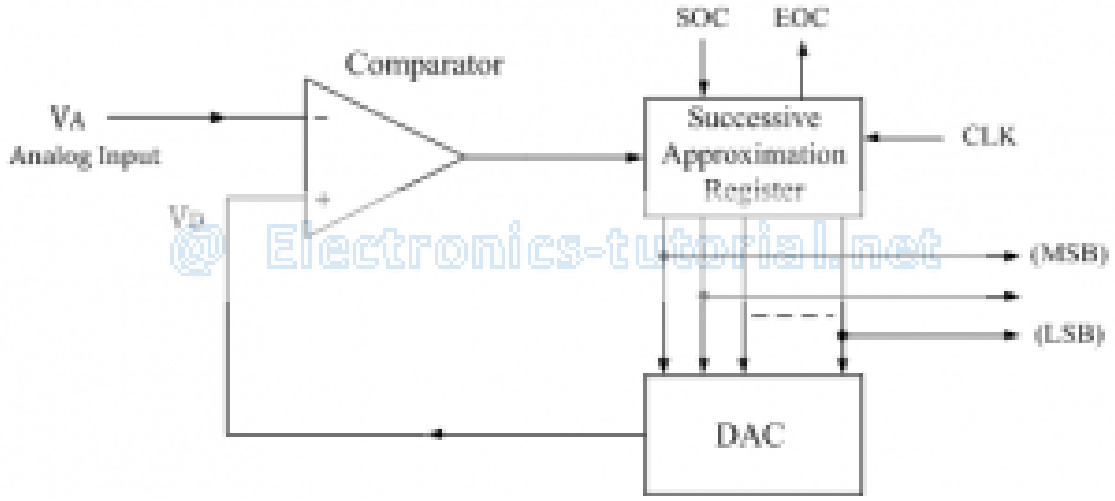
- It is the most widely used and popular ADC method. The conversion time is maintained constant in successive approximation type ADC, and is proportional to the number of bits in the digital output, unlike the counter and continuous type A/D converters. Here the unknown analog input voltage is approximated against an n-bit digital value by trying one bit at a time, beginning with the MSB. The principle of successive approximation process for a 4-bit conversion is explained here. This type of

ADC operates by successively dividing the voltage range by half, as explained in the following steps.

(1) The MSB is initially set to 1 with the remaining three bits set as 000. The digital equivalent voltage is compared with the unknown analog input voltage.
(2) If the analog input voltage is higher than the digital equivalent voltage, the MSB is retained as 1 and the second MSB is set to 1. Otherwise, the MSB is set to 0 and the second MSB is set to 1. Comparison is made as given in step (1) to decide whether to retain or reset the second MSB.

Let us assume that the 4-bit ADC is used and the analog input voltage is VA = 11 V. when the conversion starts, the MSB bit is set to1.NowVA=11V>VD=8V=[1000]2 .Since the unknown analog input voltage VA is higher than the equivalent digital voltage VD, as discussed in step (2), the MSB is retained as 1 and the next MSB bit is set to 1 as followsVD = 12V = [1100]2Now VA = 11V < VD = 12V = [1100]2
Here now, the unknown analog input voltage VA is lower than the equivalent digital voltage VD. As discussed in step (2), the second MSB is set to 0 and next MSB set to 1 asVD = 10V = [1010]2 Now again VA = 11V > VD = 10V = [1010]2 .Again as discussed in step (2) VA>VD, hence the third MSB is retained to 1 and the last bit is set to 1. The new code word isVD = 11V = [1011]2. Finally VA = VD , and the conversion stops.

The functional block diagram of successive approximation type of ADC is shown below.



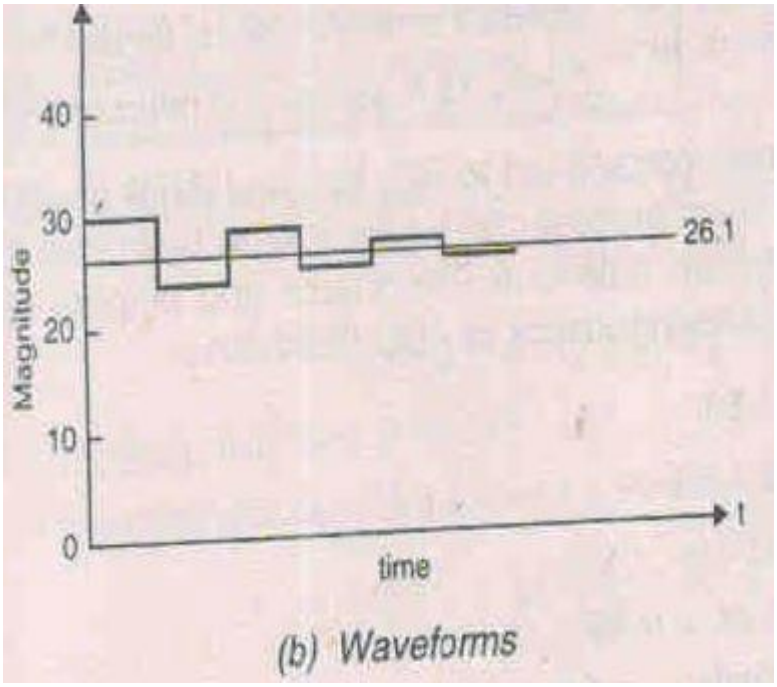
- It consists of a successive approximation register (SAR), DAC and comparator. The output of SAR is given to n-bit DAC. The equivalent analog output voltage of DAC, VD is applied to the non-inverting input of the comparator. The second input to the comparator is the unknown analog input voltage VA. The output of the comparator is used to activate the successive approximation logic of SAR.
  When the start command is applied, the SAR sets the MSB to logic 1 and other bits are made logic 0, so that the trial code becomes 1000.

Advantages:

1.Conversion time is very small, constant and independent of the amplitude of the analog input signal VA

Disadvantages:

1.Circuit is complex & the conversion time is more compared to flash type AD



(b) Waveforms

Probable short questions with answers.

Q 1) Define Resolution. [S-2015,W-2019]

Ans- It is the reciprocal of the number of discrete steps in the D/A output. Evidently resolution depends on the number of bits. The percentage resolution is [1/ (2^N -1)] * 100 where N is the number of bits. In case of 4 bits ,the resolution is $1/15^{th}$ of total voltage .

2) Define conversion time. [S-2020]

Ans- It consists of measuring the propagation delay from beginning of a conversion to the expected digital output code.

3) What is successive Approximation ADC ?

Ans.- In this conversion method,operating time is fixed & does not depend upon the value of analog input. It consists of register (SAR), a DA converter,& a comparator.

Probable Long Type Questions

Q.-1. Explain the working of ladder type DA converter. [2004.05,08,09,10]

Q-2 Explain the working of weighted resister type DAC with neat block diagram. [S-2014, W-2019,W-2020]

Q-3. Explain R-2R ladder type DAC with neat diagram [W2015]

Q-4.Explain RAMP type ADC with neat circuit diagram [S-2014,W-2019]

UNIT 6
LOGIC FAMILIES

Learning Objectives:

6.1; Various Logic families & categories according to the IC fabrication process.
6.2 Characteristics of digital ICs – Propagation Delay, Fan Out, Fan In, Power Dissipiation,Noise Margin, Power Supply requirement, & speed with reference to logic family.
6.3- Features, Circuit operation & various applications of TTL (NAND), CMOS (NAND& NOR).

Introduction -

- A circuit configuration or approach used to produce a type of digital integrated circuit is called Logic Family.
- By using logic families we can generate different logic functions, when fabricated in the form of an IC with the same approach, or in other words belonging to the same logic family, will have identical electrical characteristics.
- Some common Characteristics of the Same Logic Family include Supply voltage range, speed of response, power dissipation, input and output logic levels, current sourcing and sinking capability, fan- out, noise margin, etc.
- Choosing digital ICs from the same logic family guarantees that these ICs are compatible with respect to each other and that the system as a whole performs the intended logic function.

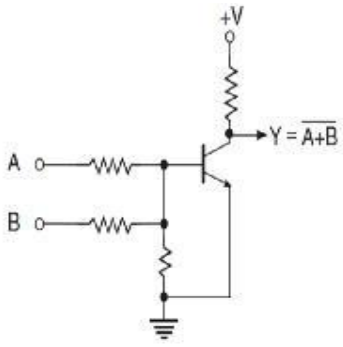
6.1; Various Logic families & categories according to the IC fabrication process:-

As per level of Integration digital ICs can be classified as under

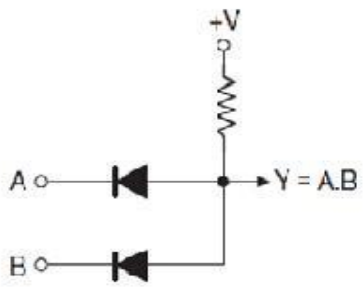
Types	No of Basic Gates on a single chip	Total no of components
SSI (Small Scale Integration)	Less than 12	Less than 100
MSI (Medium Scale Integration)	12-99	100-999
LSI(Large Scale Integration)	100-999	1000-9999
VLSI(Very Large Scale Integration)	1000-9999	10000-99999
ULSI(Ultra Large Scale Integration)	10000 & more	100000 & more

- As per operating mode of technology Integration digital ICs can be classified as under

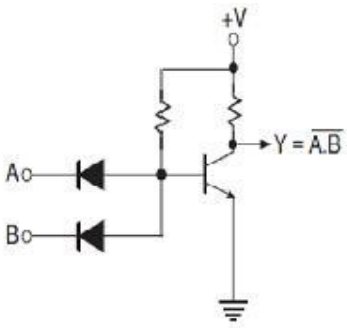
Bipolar families include: -
Diode logic (DL), Resistor-Transistor logic (RTL), Diode-transistor logic (DTL), Transistor- Transistor logic (TTL), Emitter Coupled Logic (ECL)(also known as Current Mode Logic(CML)) & Integrated Injection logic (I2L)



Resistor -Transistor logic (RTL)



Diode Logic (DL)



Diode-Transistor logic (DTL)

- Above are some examples of DL, RTL and DTL.

- MOS families include: -
  ThePMOS family (using P-channel MOSFETs) The NMOS family (using N-channel MOSFETs)
  The CMOS family (using both N- and P-channel devices)

6.2 Characteristics of digital ICs – Propagation Delay, Fan Out, Fan In, Power Dissipiation,Noise Margin, Power Supply requirement, & speed with reference to logic family

a) Propagation Delay time:-

When a signal passes (propagates) through a logic circuit, it always experiences a time delay. A change in the output level always occurs a short time, called 'propagation delay time' , later than the change in the input level that caused it.

b) Fan Out of Gates:-

When the output of a logic gate is connected to one or more inputs of other gates, a load on the driving gate is created. There is a limit to the number of load gates that a given gate can drive. This limit is called the 'Fan-Out' of the gate.

c)The fan in of agate is the no of inputs connected to the gate without any degradation in the voltage level.

d) Power Dissipation: -

It is the power or electrical energy used by a logic circuit in a specified period of time. Every Ic requires a certain amout of power to operate it. Power dissipiation is a measure of the power consumed by the logic gate when fully driven by all its inputs & it is expressed in milli volts.

A logic gate draws I_{CCH} current from the supply when the gate is in the HIGH output state, draws I_{CCL} current from the supply in the LOW output state.

- Average power is $P_D = V_{CC} I_{CC}$ where $I_{CC} = (I_{CCH} + I_{CCL}) / 2$

e) Noise Margin:-

- A measure of a circuit's noise immunity is called 'noise margin' which is expressed in volts. A measure of the ability of a logic circuit to tolerate noise.
- There are two values of noise margin specified for a given logic circuit: the HIGH (V_{NH}) and LOW (V_{NL}).

Noise margins is also defined by following equations:

$V_{NH} = V_{OH}$ (Min) - V_{IH} (Min) $V_{NL} = V_{IL}$ (Max) - V_{OL} (Max)

OR Noise Immunity-

Noise is the unwanted voltage that is induced in electrical circuits and can present a threat to the poor operation of the circuit. In order not to be adversely affected by noise, a logic circuit must have a certain amount of 'noise immunity'.

- This is the ability to tolerate a certain amount of unwanted voltage fluctuation on its inputs without changing its output state is called Noise Immunity.

f) Power Supply Requirement in Voltage: -

The nominal value of the dc supply voltage for TTL (transisitor-transistor logic) and CMOS (complementary metal-oxide semiconductor) devices is +5V. Although ommitted from logic diagrams for simplicity, this voltage is connected to Vcc or VDD pin of an IC package and ground is connected to the GND pin.

g) Speed of Operation – It is expressed in terms of propagation delay. The output & input pulses cannot change levels instantaneously of the given input. Propagation delay is defined as the timr taken for the output of agate to change after the inputs have changed.

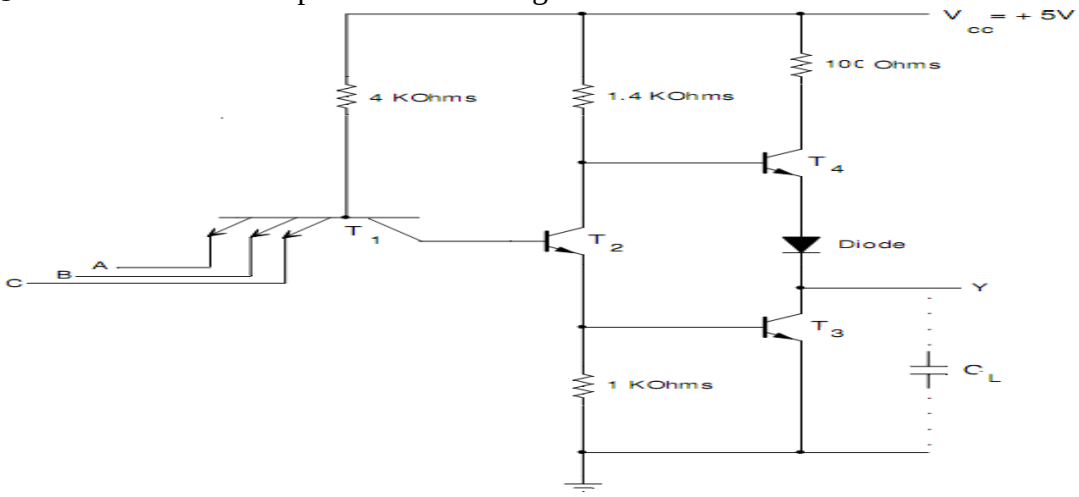
Tplh- It is the propagation delay time in going from low level (0) to high level (1).

Tphl- It is the propagation delay time in going from high level (1) to low level (0).

6.3- Features, Circuit operation & various applications of TTL (NAND), CMOS (NAND & NOR)-

Operation of TTL NAND Gate:

- Figure demonstrates a TTL NAND gate with a totem pole output. The totem pole output implies that transistor T_4 sits atop T_3 in order to give low output impedance. The low output impedance means a short time constant RC therefore the output can change rapidly from one state to the other. T_1 is a multiple type emitter transistor. Such transistor can be thought of like a combination of various transistors along with a common collector and base. Here, T_1 has 3 emitters thus there can be three inputs A, B, C. The transistor T_2 functions as a phase splitter since the emitter voltage is out of phase along with the collector voltage. The transistors T_3 and T_4 by the totem pole output, the capacitance C_L shows the stray capacitance and so on. The diode D is added to make sure that T_4 is cut off while output is low. The voltage drop of diode D remains the base-emitter junction of T_4 reverse biased therefore only T_3 conducts while output is low. The operation can be described briefly by three conditions as specified below:
- **Condition 1:** At least one input is low. Transistor T_1 saturates. Thus, the base voltage of T_2 is almost zero. T_2 is cut off and forces T_3 to cut off. T_4 functions as an emitter follower and couples a high voltage to load. Output is high
- **Condition 2:** Each input is high. The emitter base junctions of T_1 are reverse biased. The collector base junction of T_1 is forward biased. Therefore, T_1 is in reverse active mode. The collector current of T_1 flows in reverse direction. Because this current is flowing in the base of T_2, the transistors T_2 and T_3 saturate and then output Y is low.
- **Condition 3:** The circuit is operating under II while one of the inputs becomes low. The consequent emitter base junction of T_1 starts conducting and T_1 base voltage drops to a low value. Thus, T_1 is in forward active mode. The high collector current of T_1 shifts the stored charge in T_2 and T_3 and hence, T_2 and T_3 go to cut-off and T_1 saturates and then output Y returns to high.



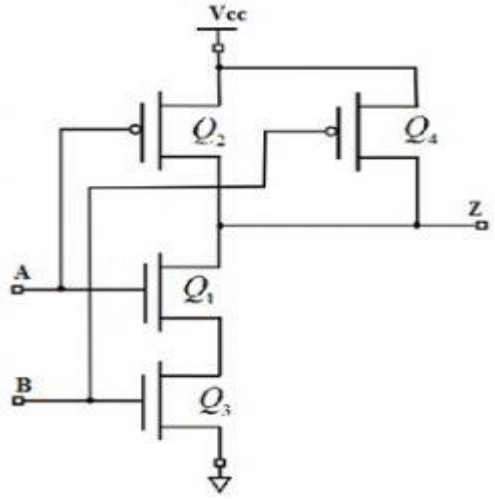
Logic Diagram of TTL NAND Gate with Totem Pole Output

CMOS : Working Principle & Its Applications

The term CMOS stands for “Complementary Metal Oxide Semiconductor”. This is one of the most popular technologies in the computer chip design industry and it is broadly used today to form integrated circuits in numerous and varied applications. It has much smaller power dissipation.
In CMOS technology, both N-type and P-type transistors are used to design logic functions. The same signal which turns ON a transistor of one type is used to turn OFF a transistor of the other type.
CMOS offers relatively high speed, low power dissipation, high noise margins in both states, and will operate over a wide range of source and input voltages.

CMOS NAND Gate

The below figure shows a 2-input Complementary MOS NAND gate. It consists of two series NMOS transistors between Y and Ground and two parallel PMOS transistors between Y and Vcc. If either input A or B is logic 0, at least one of the NMOS transistors will be OFF, breaking the path from Y to Ground. But at least one of the PMOS transistors will be ON, creating a path from Z to Vcc.



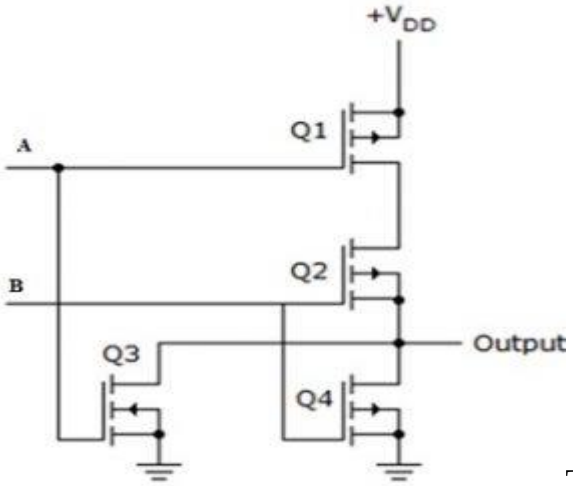
Two Input NAND Gate

Hence, the output Z will be high. If both inputs are high, both of the NMOS transistors will be ON and both of the PMOS transistors will be OFF. Hence, the output will be logic low. The truth table of the NAND logic gate given in the below table.

A	B	Pull-Down Network	Pull-up Network	OUTPUT Z
0	0	OFF	ON	1
0	1	OFF	ON	1
1	0	OFF	ON	1
1	1	ON	OFF	0

CMOS NOR Gate

A 2-input NOR gate is shown in the figure below. The NMOS transistors are in parallel to pull the output low when either input is high. The PMOS transistors are in series to pull the output high when both inputs are low, as



given in the below table. The output is never left floating. Two Input NOR Gate

The truth table of the NOR logic gate given in the below table.

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

Advantages

- The advantages of CMOS include the following.
- The main benefits of CMOS over TTL are good noise margin as well as less power consumption. This is due to no straight conducting lane from V_{cc} to GND, fall times based on the conditions of input, then the transmission of the digital signal will become easy & low cost through CMOS chips.
- CMOS is used to explain the amount of memory on the motherboard of the computer that will store in the settings of Bias. These settings mainly include the date, time, and settings of hardware
  TTL is a digital logic circuit where bipolar transistors work on DC pulses. Several transistor logic gates are normally made-up of a single IC.
- The outputs if CMOS drive actively in both ways
  - It uses a single power supply like + Vcc or V_{DD}
  - These gates are very simple, Input impedance is high, CMOS logic uses less power whenever it is held in a set state, Power dissipation is negligible, Fan out is high. It is TTL compatibility & Stability of temperature. Noise immunity is good, Designing is very well, Robust mechanically &
    Logic swing is large (V_{DD}).

Disadvantages-

- The disadvantages of CMOS include the following.
- The cost will be increased once the processing steps increases, however, it can be resolved.
- The packing density of CMOS is low as compared with NMOS.
- MOS chips should be secured from getting static charges by placing the leads shorted otherwise; the static charges obtained within leads will damage the chip. Another drawback of the CMOS inverter is that it utilizes two transistors as opposed to one NMOS to build an inverter, which means that the CMOS uses more space over the chip as compared with the NMOS. These drawbacks are small due to the progress within the CMOS technology.

CMOS Applications

- Complementary MOS processes were widely implemented and have fundamentally replaced NMOS and bipolar processes for nearly all digital logic applications. CMOS technology has been used for the following digital IC designs.
  - Computer memories, CPUs
  - Microprocessor designs
  - Flash memory chip designing
  - Used to design application-specific integrated circuits (ASICs)

The memory of the CMOS within a computer is a kind of non-volatile RAM that store BIOS settings & the information of time and date.

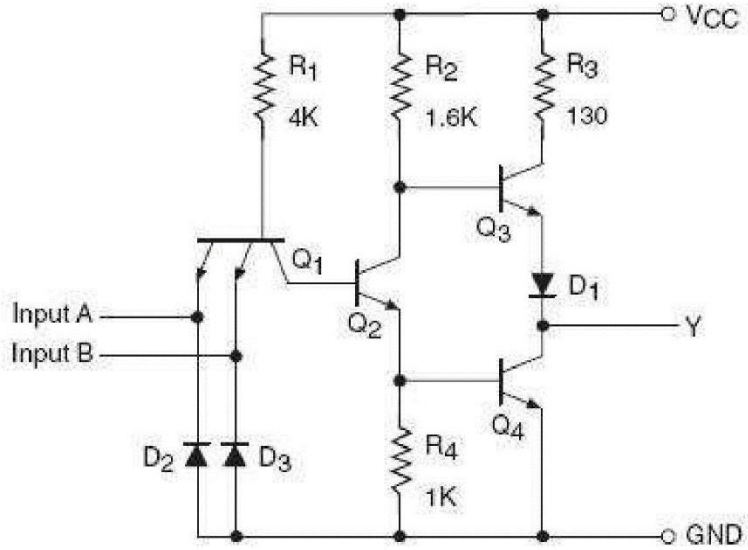
6.3;-FEATURES, CIRCUIT OPERATION AND VARIOUS APPLICATIONS OF TTL,CMOS LOGIC:-

- In Transistor-Transistor logic (TTL), logic gates are built only around transistors.
- TTL was developed in 1965. & has been improved to meet performance requirements. There are many versions or families of TTL.

- For example
  - Standard TTL
  - High Speed TTL (twice as fast, twice as much power)
  - Low Power TTL (1/10 the speed, 1/10 the power of "standard" TTL)
  - Schhottky TTL etc. (for high-frequency uses)
- All TTL logic families have three configurations for outputs
  1. Totem pole output
  2. Open collector output
  3. Tristate output

Totem pole output:-

- Addition of an active pull up circuit in the output of a gate is called totem pole.
- To increase the switching speed of the gate which is limited due to the parasitic capacitance at the output totem pole is used.
- The circuit of a totem-pole NAND gate is shown below, which has got three stages
  1. Input Stage
  2. Phase Splitter Stage
  3. Output Stage



- Transistor Q1 is a two-emitter NPN transistor, which is equivalent two NPN transistors with their base and emitter terminals tied together.
- The two emitters are the two inputs of the NAND gate

In TTL technology multiple emitter transistors are used for the input devices
Diodes D2 and D3 are protection diodes used to limit negative input voltages.

- When there is large negative voltage at input, the diode conducts and shorting it to the ground Q2 provides complementary voltages for the output transistors Q3 and Q4.
- The combination of Q3 and Q4 forms the output circuit often referred to as a totem pole arrangement (Q4 is stacked on top of Q3). In such an arrangement, either Q3 or Q4 conducts at a time depending upon the logic status of the inputs

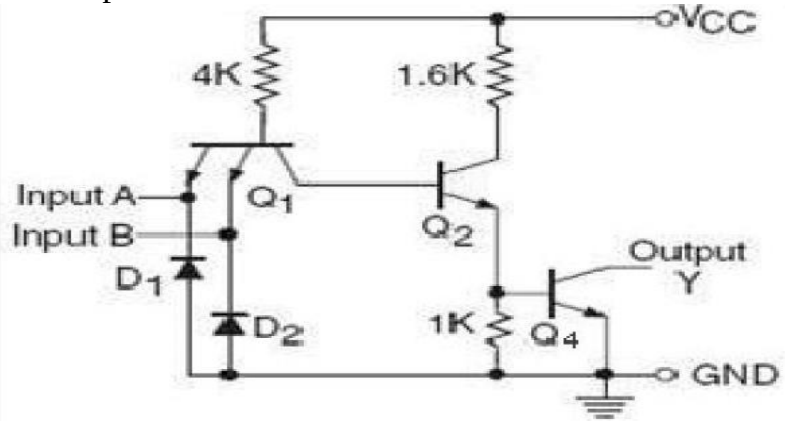
Diode D1 ensures that Q4 will turn off when Q2 is on (HIGH input) The output Y is taken from the top of Q3.

Advantagesof Totem PoleOutput:-

- The features of this arrangement are
  1. Low power consumption
  2. Fast switching
  3. Low output impedance

OPEN COLLECTOR OUTPUT:-

- Figure below shows the circuit of a typical TTL gate with open-collector output The circuit elements associated with Q3 in the totem-pole circuit are missing and the collector of Q4 is left open-circuited, hence the name open-collector.



- An open-collector output can present a logic LOW output. Since there is no internal path from the output Y to the supply voltage V_{CC} , the circuit cannot present a logic HIGH on its own.

Advantages of Open Collector Outputs:-

- Open-collector outputs can be tied directly together which results in the logical ANDing of the outputs.

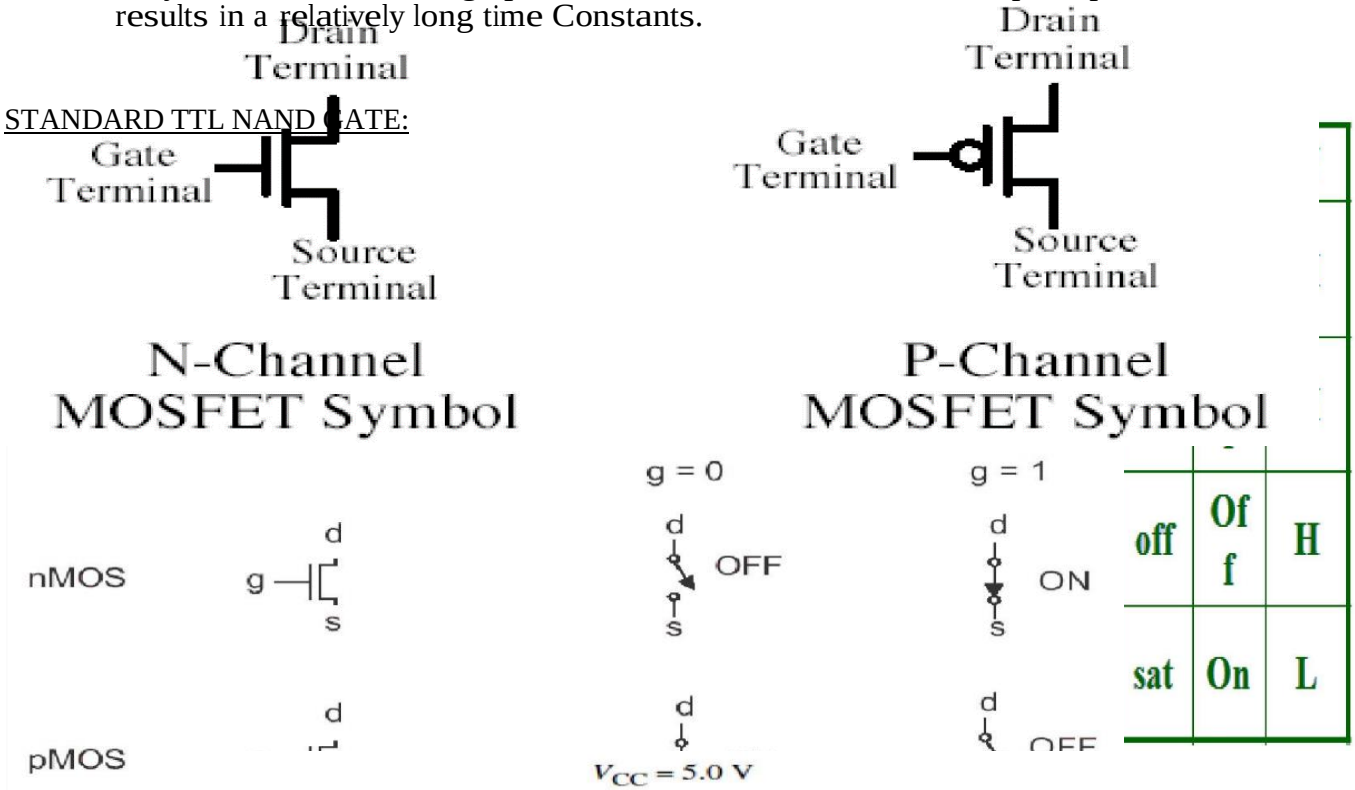
Thus the equivalent of an AND gate can be formed by simply connecting the outputs.

- Standard TTL gates with totem-pole outputs can only provide a HIGH current output of 0.4 mA and a LOW current of 1.6 mA. Many open-collector gates have increased current ratings.
- Different voltage levels - A wide variety of output HIGH voltages can be achieved using open-collector gates. This is useful in interfacing different logic families that have different voltage and current level requirements.

Disadvantage of open-collector gates:-

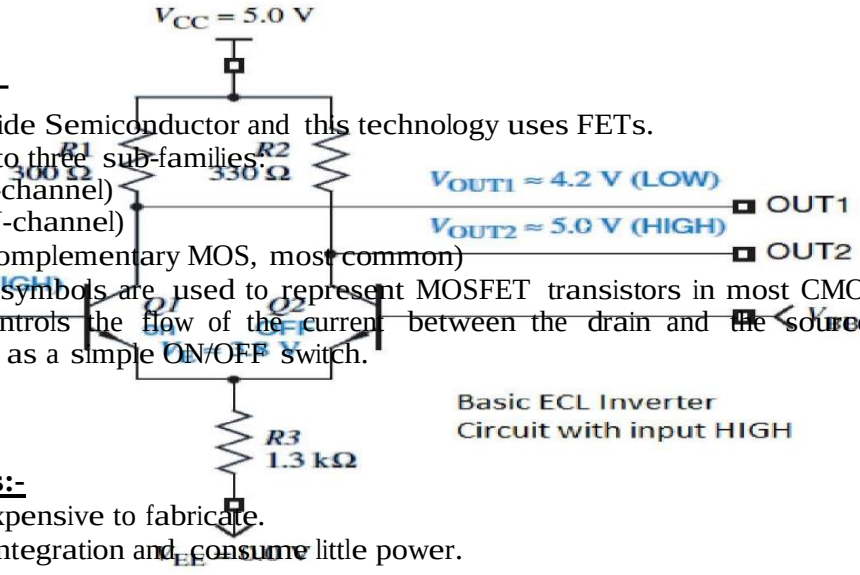
- They have slow switching speed. This is because the value of pull-up resistor is in kW, which results in a relatively long time Constants.

STANDARD TTL NAND GATE:



CMOS TECHNOLOGY:-

- MOS stands for Metal Oxide Semiconductor and this technology uses FETs.
- MOS can be classified into three sub-families:
  - PMOS (P-channel)
  - NMOS (N-channel)
  - CMOS (Complementary MOS, most common)
- The following simplified symbols are used to represent MOSFET transistors in most CMOS. The gate of a MOS transistor controls the flow of the current between the drain and the source. The MOS transistor can be viewed as a simple ON/OFF switch.



AdvantagesofMOSDigitalICs:-

- They are simple and inexpensive to fabricate.
- Can be used for Higher integration and consume little power.

DisadvantagesofMOSDigitalICs:-

- There is possibility for Static-electricity damage.
- They are slower than TTL.

PROBABLE SHORT QUESTION WITH ANSWER.

1. Define Propagation Delay. [S-2014,W-2019]

Ans- The amount of time taken by a clock pulse to propagate from input to output is called Propagation delay.

$$Tpd = (tp_H + t_{PHL})/2$$

2. Define fan In & Fan out. [S-2014,S-2015, S-2019,W-20120]

Ans- Fan In- It is defined for a logic gate as the no of inputs that the gate is designed to handle.

Fan Out- It is defined as the maximum number of standard loads that the output of the gate can drive without impairing its normal operation.

3. Define noise Margin

Ans- The noise margin represents the maximum signal that can be added to the input signal of a digital circuit without causing any undesirable change in the circuit output.

PROBABLE LONG QUESTION

1. Describe the characteristics of digital ICs. [W-2019, W-2020]

Learning Resources:			
Sl.No	**Name of Author**	**Title of the Book**	**Publisher**
1.	Ananda Kumar	Fundamental of Digital Electronics	PHI
2	B. R. Gupta & V. Singhal	Digital Electronics	S. K. Katteria
3.	P Raja	Digital Electronics	Sci Tech.