GOVT. POLYTECHNIC BALASORE

[LECTURE NOTES]

DIGITAL ELECTRONICS
[TH3]

DIPLOMA
IN
3RD SEMESTER, E&TC ENGINEERING

PREPAREDBY:-ER. YOGASAKTI YOGAMAYA

(LECT. IN E & TC)

DEPT. OF ELECTRONICS & TELECOMMUNICATION ENGINEERING

Codes

Codes: codes are the representation of information in a particular forme.

→ In general, the numbers in a digital system or computer are used in coded form to achieve

i) To represent numeric or alphanumeric or special characters in only binary digits i.e 0 or 1.

ii) To check whether a character transmitted in the coded form is correctly received if not then to correct it.

→ There are two types of binary code in use i.e

i) Weighted code

ii) Non-weighted code.

→ In weighted code, for each position (or bit), there is specific weight attached.

BCD (Binary coded decimal):

- A code that represents each digit of a decimal number represented by a binary value.

8421 BCD Code:

- In 8421 code each decimal digit is expressed by its 4 bit binary equivalent.

8421	Binary	decimal
0000	0000	0
0011	0011	3
0100	0100	4
0101	0101	5
1001	1001	9
0001 0000	1010	10
0001 0101	1111	15
0001 0110	10000	16
1001 1000	1100010	98
0001 0000 0000	1100100	100

The six code combinations that are not used in BCD are 1010, 1011, 1100, 1101, 1110 & 1111.

BCD Addition:

1. Add two BCD numbers.
2. If a 4 bit sum is equal to less than 9 (1001), it is a valid BCD number.
3. If a 4 bit sum is greater than 9, or if a carry out of the 4 bit group is generated, it is an invalid result.
4. Add 6 (0110) to the 4 bit sum in order to skip the six invalid states & return the codes to 8421.

```
  647          0110  0100  0111
+ 482          0100  1000  0010
-----          ---------------
 1129          1010  1100  1001
               0110  0110
               ----------------
             1 0001  0001  0010  1001
               ----------------
               1     1     2     9
```

Decimal	7421	6311	5421
0 —	0000	0000	0000
1 —	0001	0001	0001
2 —	0010	0011	0010
3 —	0011	0100	0011
4 —	0100	0101	0100
5 —	0101	0111	0101
6 —	0110	1000	1001
7 —	1000	1001	1010
8 —	1001	1011	1011
9 —	1010	1100	1100
latur consu to —	1011		

bit

Non weighted code:

- Non-weighted codes are codes that are not positional weighted.
- This means that each position within a binary number is not assigned a fixed value.

Ex: Excess 3 code, Gray code.

Ex-3 Code:

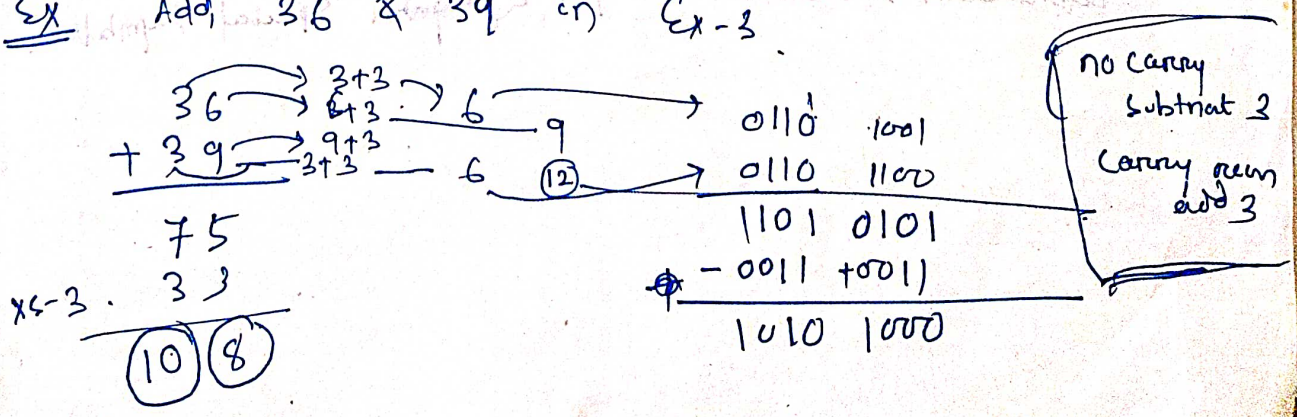
- An excess-3 code is obtained by adding 3 to a decimal number.

Decimal		Ex-3
0	——	0011
1	——	0100
2	——	0101
3	——	0110
4	——	0111
5	——	1000
6	——	1001
7	——	1010
8	——	1011
9	——	1100
10	10+33	0100 0011
11	11+33	0100 0100

Excess-3 code

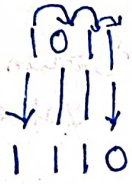
1. Add excess-3 codes
2. If above addition produces a carry ① 1, adds 3 (0011) to the sum of digits & carry is 0 subtract 3 (0011) from the sum

Ex Add 36 & 39 in Ex-3.

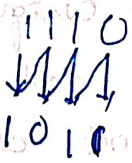


Gray code : Non weighted code.

Convert B - G



G - B



Alphanumeric code:

A code used to represent letters of the alphabet & numerical characters.

ASCII code

- Stands for American standard code for information interchange.
- It is a ~~8 bit 7 bit code~~ used extensively for printers, typewriters & terminals of small computer system.
- It is a 7 bit code but 8th bit is usually added which is either set 0 & 1 as a parity bit.
- The code includes decimal numbers 0-9, capital alphabets, lowercase alphabet & some special symbols.

EBCDIC code: (Pronounced eb-si-dik)

- Stands for Extended binary Coded decimal interchange ~~code~~ code.
- ~~Mostly used in~~ It is used in most of large computers for comm^n.

→ It is 8 bit code. & used BCD.

- This code also includes Capital alphabets, lowercase alphabets, numbers 0-9 & ~~Symbol~~ Special symbols.

Boolean Algebra

Boolean algebra: Boolean algebra is an algebra that may be defined with set of elements, a set of operations & a number of unproved axioms & postulates.

(axioms are set of logical expression that we accept without proof).

Laws:

1. $\bar{0} = 1$
2. $\bar{1} = 0$
3. $\bar{\bar{x}} = x$
4. $x \cdot 0 = 0$
5. $x \cdot 1 = x$
6. $x \cdot x = x$
7. $x \cdot \bar{x} = 0$
8. $x + 0 = x$
9. $x + 1 = 1$
10. $x + x = x$
11. $x + \bar{x} = 1$
12. $x + y = y + x$
13. $x \cdot y = y \cdot x$
14. $x + (y+z) = (x+y) + z$] Associative Law
15. $x \cdot (y \cdot z) = (x \cdot y) \cdot z$]
16. $x(y+z) = (x \cdot y) + (x \cdot z)$] Distributive Law
17. $x + yz = (x+y)(x+z)$]
18. $x + \bar{x}y = x + y$
19. $\overline{x+y} = \bar{x} \cdot \bar{y}$] Demorgan's Theorem
20. $\overline{xy} = \bar{x} + \bar{y}$]
21. $x + xy = x$ } Absorption Law
22. $x(x+y) = x$ }
23. $x(\bar{x}+y) = x \cdot y$

① DeMorga's Theorem:

⇒ Demorgan's 1st theorem states that the complement of sum of digital signal is equal to the product of its complement.

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

Proof

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

From above table its prove that $\overline{A+B} = \bar{A}\cdot\bar{B}$

ii) Demorgan's 2nd theorem states that the complement of a product of digital signal is equal to sum of its complement.

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

Proof:

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

From able table its prove that

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

Prove

1. $X + XY = X$

L.H.S $X + XY$

$= X(1+Y)$

$= X \cdot 1$

$= X$

2. $X(X+Y) = X$

L.H.S $= X(X+Y)$

$= X \cdot X + X \cdot Y$

$= X + XY$

$= X(1+Y)$

$= X$

3. $X(\bar{X}+Y) = XY$

L.H.S $X(\bar{X}+Y)$

$= X \cdot \bar{X} + XY$

$= 0 + XY$

$= XY$

4. $X + \bar{X}Y = X+Y$

LHS $= X + \bar{X}Y$

$= X \cdot 1 + \bar{X}Y$

$= X(1+Y) + \bar{X}Y$

$= X + XY + \bar{X}Y$

$= X + \bar{X}Y + XY$

$= X + Y(X+\bar{X})$

$= X + Y$ $(\because X+\bar{X} = 1)$

5. $X + YZ = (X+Y)(X+Z)$

LHS : $X + YZ$

$= X \cdot 1 + YZ$

$= X(1+Y) + YZ$

$= X + XY + YZ$

$= X \cdot 1 + XY + YZ$

$= X(1+Z) + XY + YZ$

$= X + XZ + XY + YZ$

$= X + XY + XZ + YZ$

$= X \cdot X + XY + XZ + YZ$

$= X(X+Y) + Z(X+Y)$

$= (X+Y)(X+Z)$ Pw

6. $XY + \bar{X}Z + YZ = X \cdot Y + \bar{X}Z$

LHS : $XY + \bar{X}Z + YZ$

$= XY + \bar{X}Z + YZ(X+\bar{X})$

$= XY + \bar{X}Z + XYZ + \bar{X}YZ$

$= XY + XYZ + \bar{X}Z + \bar{X}YZ$

$= XY(1+Z) + \bar{X}Z(1+Y)$

$= XY + \bar{X}Z$ Pw

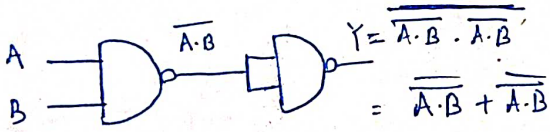
Universal Gates:

* NAND & NOR gates are called universal gates because both can be used to implement any gate like AND, OR, NOT, EX-OR & EX-NOR.

NAND:

NOT:

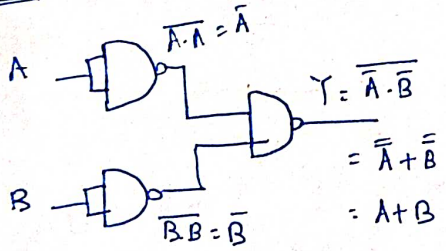
AND:



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

$= A \cdot B$

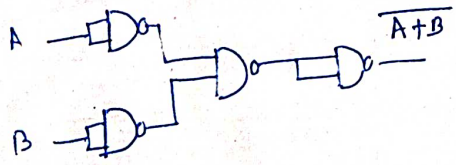
OR:



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

$= A + B$

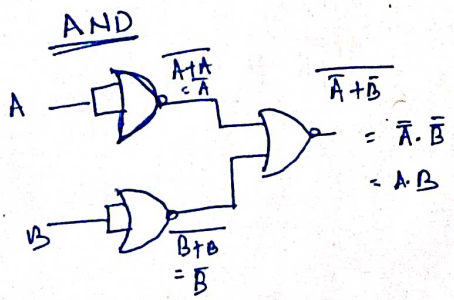
NOR



Using NOR:

NOT:

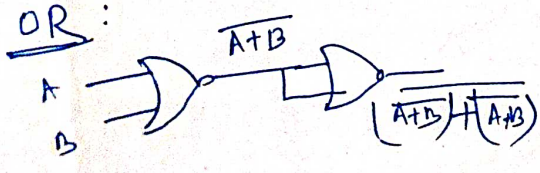
AND



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

$= A \cdot B$

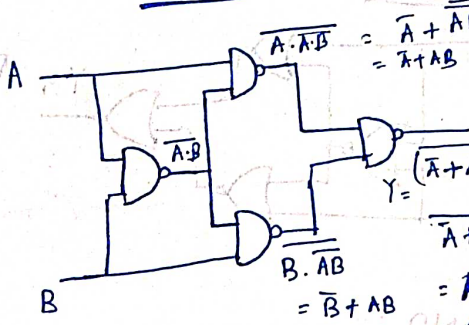
OR:



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

$= (A+B)(A+B) = A+B$

EX-OR



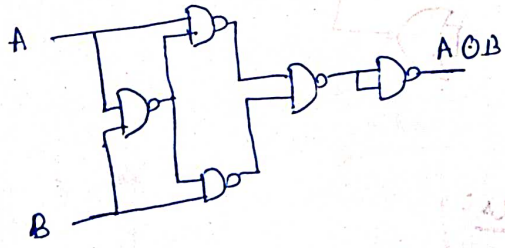
$\overline{\bar{A}+AB} + \overline{\bar{B}+AB}$

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

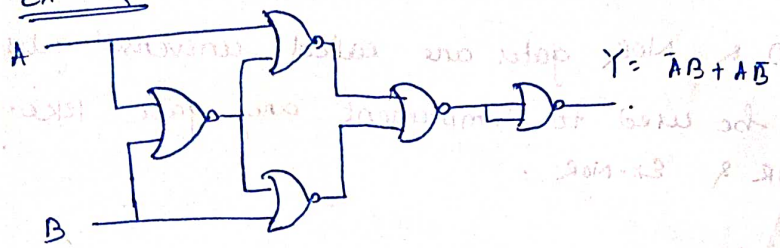
$= A(\bar{A}+\bar{B}) + B(\bar{A}+\bar{B})$

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

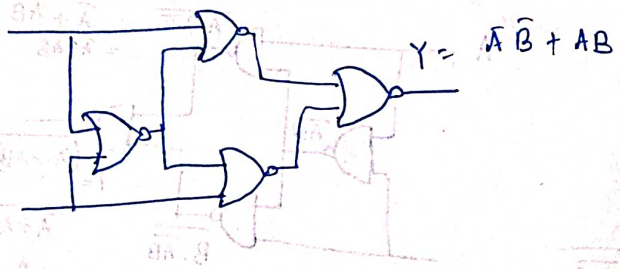
EX-NOR



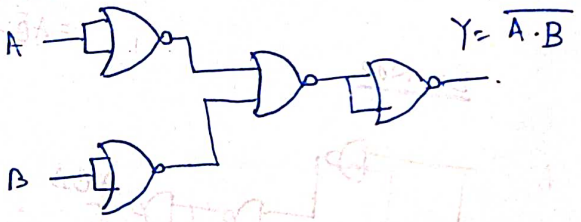
Ex-OR :



Ex-NOR :



NAND :



ICs :

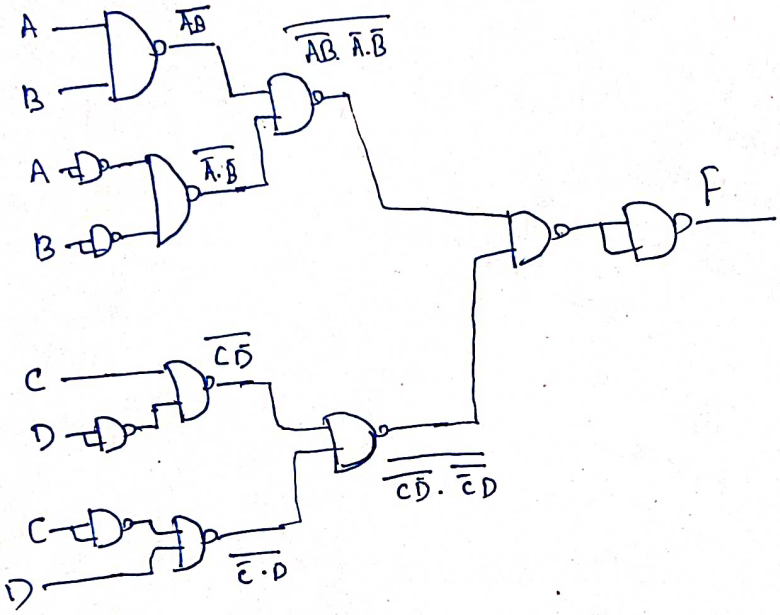
NOT — 7404
NAND — 7400
NOR — 7402
AND — 7408
EX-OR — 7486
NOR —
OR —

Design the logic using NAND gate only.

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

Ans $$F = \overline{\overline{(AB + \bar{A}\bar{B})}\ \overline{(C\bar{D} + \bar{C}D)}}$$

$$= \overline{\overline{AB} \cdot \overline{\bar{A}\bar{B}}}\ \overline{\overline{C\bar{D}} \cdot \overline{\bar{C}D}}$$



Q. $A\bar{B} + \bar{A}B$

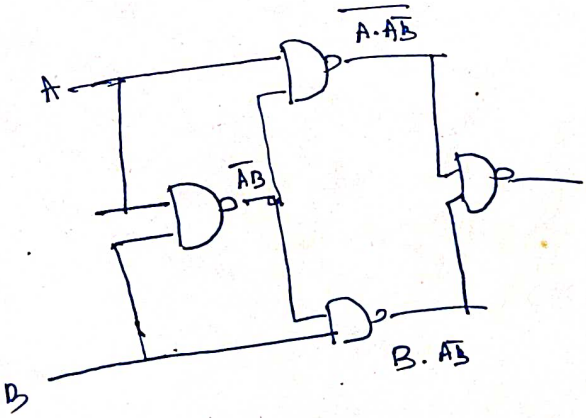
$= A.\bar{A} + A\bar{B} + B.\bar{B} + \bar{A}B$

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

$= \cancel{A+B}\ A.\overline{AB} + B.\overline{AB}$

$= \overline{A.\overline{AB} + B.\overline{AB}}$

$= \overline{A.\overline{AB}} \cdot \overline{B.\overline{AB}}$



① $xy + \bar{x}z + yz = xy + \bar{x}z$

Representation Logic Expression:

i) Sum of Product (SOP): A type of Boolean expression where several product terms are summed (ORed) together.

Ex: $Y = AB + BC + ABCD$

ii) Product of Sum (POS): A type of Boolean expression where several sum terms are multiplied (ANDed) together.

Ex: $Y = (A+B)(C+D)$

Canonical & Standard forms:

i) Minterm: It is a Product of Boolean expression where all possible variables appears once in complement or uncomplement variable.

Ex: $Y = AB + A\bar{B}$

Find min term of $Y = A + BC$

$Y = A + BC$

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

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

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

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

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

$\Rightarrow Y = m_7 + m_6 + m_5 + m_4 + m_3$

So the canonical form of SOP

$Y = \sum m(7,6,5,4,3)$

Minterms are represented by $m_0, m_1, m_2 \ldots$ Here the suffix indicates the decimal code corresponding to the minterm comb^n.

Ex

A B C D	Minterm	Designation
0 0 0 0	$\bar{A}\bar{B}\bar{C}\bar{D}$	m_0
0 0 0 1	$\bar{A}\bar{B}\bar{C}D$	m_1
0 0 1 0	$\bar{A}\bar{B}C\bar{D}$	m_2
0 0 1 0	$\bar{A}\bar{B}CD$	m_3
1 1 1 1	$ABCD$	m_{16}

ii) Maxterm: A sum term containing all n variables of the function in either complemented or uncomplemented form is called Maxterm.

- It is designated with (capital alphabetic letter) M & the suffix indicates the decimal code corresponding to the maxterm combination.

A B C	Maxterm	Designation
0 0 0	$A+B+C$	M_0
0 0 1	$A+B+\bar{C}$	M_1
0 1 0	$A+\bar{B}+C$	M_2
0 1 1	$A+\bar{B}+\bar{C}$	M_3
⋮	⋮	⋮
1 1 1	$\bar{A}+\bar{B}+\bar{C}$	M_7

Ex Express $Y = AB + \bar{A}C$ in Max term.

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

Canonical form of POS $= M_0 \; M_1 \; M_2 \; M_3$

$$\boxed{Y = \Pi M(0,1,2,3)}$$

Maxterm is complement of Minterm.

List Don't care condition: An o/p condition that can be regarded as either high or low.

- We can't group only don't care, that group having at least one 1 (SOP) or 0 (POS)

∴ In some digital system, certain input cond^n never occurs during normal operation, therefore corresponding o/p never occurs. That is don't care cond^n.

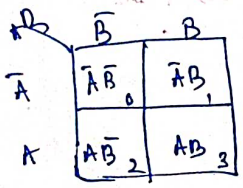
0	0	0	1

Karnaugh Map (K-map):

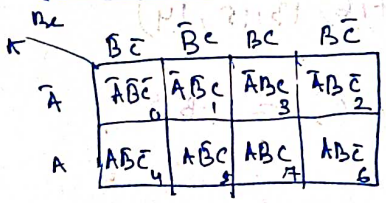
→ K map is a method of simplifying Boolean functions in a systematic mathematical way.

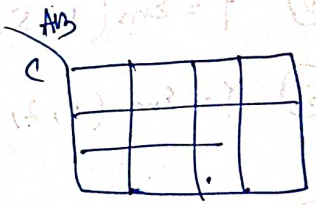
⇒ A K map is consists of cells based upon 2^n. where n = no of variables.

Two variable:

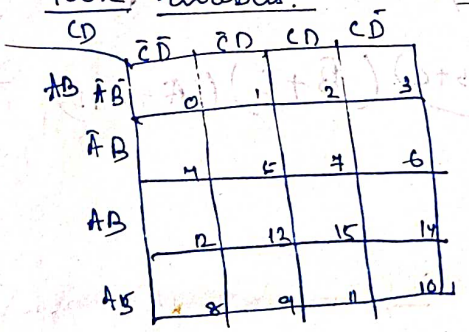


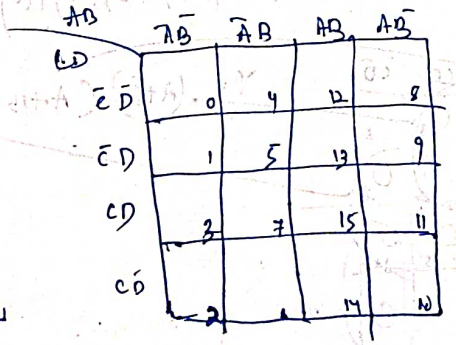
Three variable





Four variables:



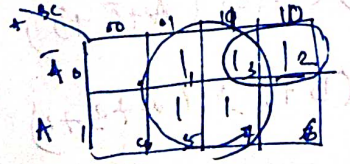


- In K-map minterms are marked by 1 & max terms are marked by 0.
- Grouping is performed by combining the adjacent logic cells which contain 1's ~~or 0's~~ (in case of SOP) & 0's (for POS).
- We can group 1, 2, 4, 8, 16, 32 no of 1's or 0's.

Solve by K map:

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

⇒ $F = \Sigma m(7, 5, 3, 2, 1)$



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

$+ \bar{B}C$

(2) $Y = m_1 + m_3 + m_5 + m_7 + m_9 + m_{12} + m_{13}$

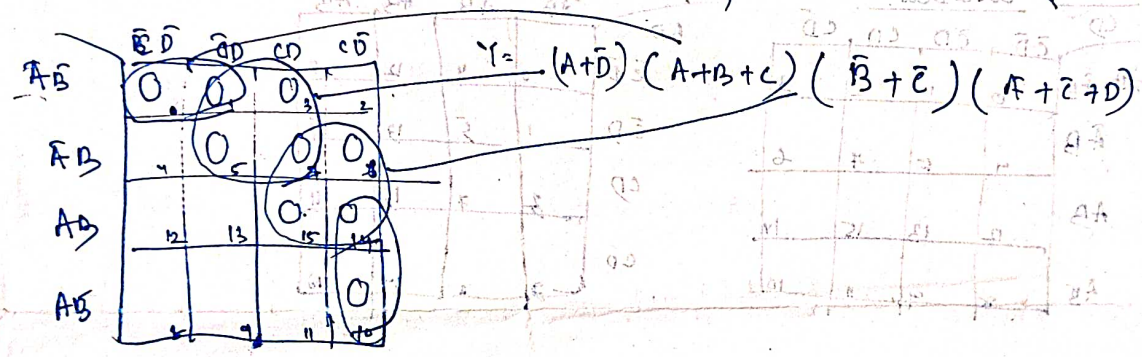
$Y = \bar{A}D + A\bar{B}C + \bar{C}D$

(3) $Y = \Sigma m(8, 10, 11, 12, 13, 14, 15) = A\bar{B} + A\bar{D} + A\bar{C}$

(4) $Y = \Sigma m(4, 5, 7, 6, 13, 15, 14) = \bar{A}B + B\bar{D} + B\bar{C}$

(5) $Y = \Sigma m(0, 2, 8, 10) = B\bar{D}$

(6) $Y = \Pi M(0, 1, 3, 5, 6, 7, 10, 14, 15)$



(7) $Y = \Pi(0, 1, 4, 5, 6, 8, 9, 12, 13, 14) = \bar{C}(\bar{B}+D)$

(8) $Y = \Sigma m(0, 1, 4, 5, 3, 2, 11, 10) = \bar{A}\bar{C} + B\bar{C}$

$Y = \Sigma m(2, 3, 6, 7, 10, 11, 14, 15) = C$

$F(A, B, C, D) = \Sigma m(0, 4, 7, 8, 9, 12, 13, 15) = \bar{C}\bar{D} + A\bar{C} + BCD$

(10) $F(A, B, C, D) = \Sigma m(0, 1, 4, 5, 12, 13, 8, 9, 2, 6, 14) = \bar{C} + \bar{A}\bar{D} + B\bar{D}$

(11) $F(A, B, C, D) = \Sigma m(3, 5, 6, 7, 8, 11, 12, 13, 14, 15) = A\bar{C}\bar{D} + B\bar{C} + CD + B\bar{D}$

(12) $F = \Pi M(0, 3, 7, 8, 9, 12, 13) = (A+B+C+D)(\bar{A}+C)(A+\bar{C}+\bar{D})$

(13) $F = \Pi M(4, 5, 7, 10, 11, 12, 13, 15) = (\bar{B}+C)(\bar{B}+\bar{D})(\bar{A}+B+C)$

(14) $F = \Sigma m(2, 3, 5, 7, 9, 11, 12, 13, 14, 15) = \bar{A}\bar{B}C + BD + AB + A\bar{D}$

→ see don't care in previous page.

(15) $F = \Sigma m(2, 3, 4, 10, 12, 13) + d(11, 14, 15) = AB + \bar{B}C + B\bar{C}$

(16) $F = \Sigma m(0, 2, 3, 5, 6, 7, 8, 9) + d(10, 11, 12, 13, 14, 15) = B\bar{D} + C + A + BD$

(17) $f = \Sigma m(1, 3, 7, 11, 15) + \Sigma d(0, 2, 5) = CD + \bar{A}\bar{B}$

(18) $F = \Sigma m(5, 7, 8, 10, 13, 15) + \Sigma d(0, 1, 2, 3) = BD + B\bar{D}$ → As we can't take only don't care as a group.

(19) $F = \Sigma m(0, 1, 2, 3, 6, 7, 13, 14) + d(9, 5, 10, 12)$, $= \bar{A}\bar{B} + A\bar{C} + CD + \bar{A}\bar{D}$

Combinational Switching Logic

Combinational Logic circuit design;

→ A digital device may be consists of two types of ckts
- combinational logic circuit
- Sequencial logic ckt.

→ In a combinational logic ckt we have a number of inputs & may have one or more output.

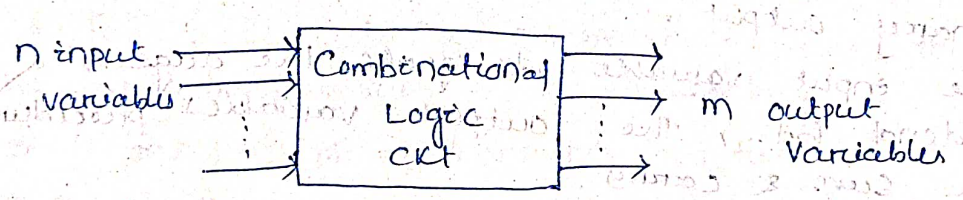
→ The o/p at any time depends only upon the inputs at that very time & is independent of the input values which might may been inputted earlier.

→ A combinational ckt consists of input variables & logic gates & output variables.

→ On receiving input signal the logic gates generate the outputs.

→ This process transforms binary information from the given input data to the required output data.

→ Both i/p & o/p data are represented by binary signals i.e they exist in two possible values, one representing logic 1 & other logic 0.



→ The n input binary variables come from an external source, m o/p variables go to an external destination.

→ for n variables there are 2^n possible combinations of binary i/p values. For each possible input combination, there is one & only one possible o/p combination.

Procedure →

1- Problem is stated.

2→ The number of available input variables & required output variable is determined.

3→ The input & output variables are assigned letter symbol.

Truth table →

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

→ The 8 rows under the i/p variables designate all possible combinations of 1's & 0's that these variables may have.

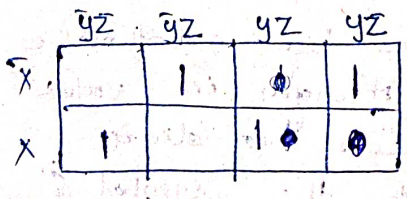
→ When all the input bits are 0's, the output is 0's.

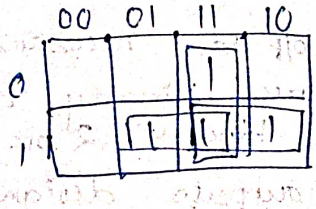
→ The 'S' output is equal to 1 when one i/p is equal to 1 or when all three inputs are equal to 1.

→ The 'C' output has a carry of 1 if two or three inputs are equal to 1.

→ The input-output logical relationship of full-adder ckt may be expressed in 2 Boolean functions, one for each output variables i.e it requires unique map for its simplifications.

Maps of Full Adder →





$$S = \bar{x}\bar{y}z + \bar{x}y\bar{z} + x\bar{y}\bar{z} + xyz$$
$$= \bar{x}(\bar{y}z + y\bar{z}) + x(\bar{y}\bar{z} + yz)$$
$$= \bar{x}(y \oplus z) + x(y \odot z)$$

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

$C = \Sigma m(3,5,6,7)$

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

put $P = y \oplus z$

$\bar{P} = y \odot z$

$S = \bar{x}P + x\bar{P}$

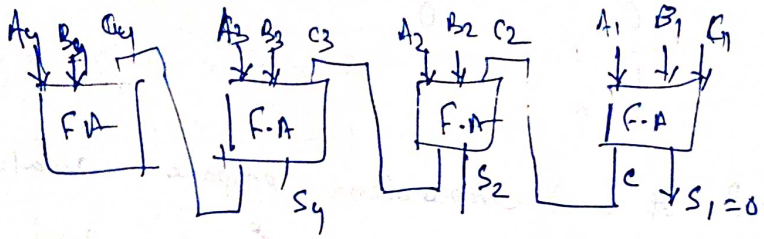
$= x \oplus P$

$= x \oplus y \oplus z$

$\Rightarrow C = xy + yz + zx$

Binary Parallel Adder:

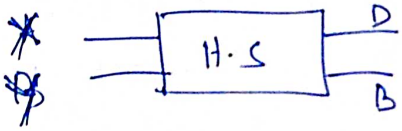
→ By using a number of full adder we can built binary adder of any length.



(4bit binary adder)

Subtractor:

Half Subtractor:

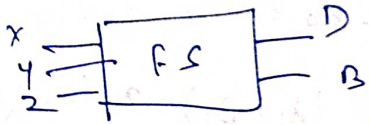


x	y	D	B
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

$D = \bar{x}y + x\bar{y}$

$B = \bar{x}y$

Full Subtractor:



$x-(y+z)$

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

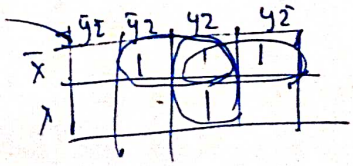
$D = \bar{x}\bar{y}z + \bar{x}y\bar{z} + x\bar{y}\bar{z} + xyz$

$= \bar{x}(\bar{y}z + y\bar{z}) + x(\bar{y}\bar{z} + yz)$

$= \bar{x}(y \oplus z) + x(y \odot z)$

$= x \oplus y \oplus z.$

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



$B = yz + \bar{x}z + \bar{x}y$

Comparators:

It compare ni.

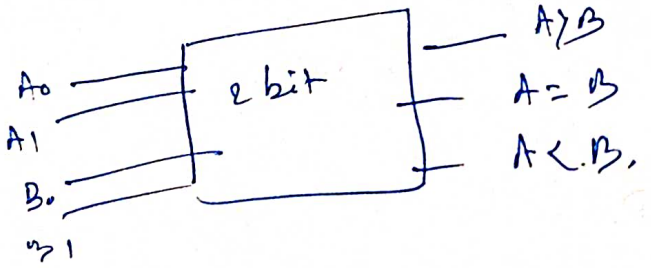
A	B	A > B	A = B	A < B
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0

$A > B \Rightarrow A\bar{B}$

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

$A < B \Rightarrow \bar{A}B$.

The fun of a comparator compare whether the binary ni > or = or < then from a 2nd binary ni.



1) The two nis are equal if

$(A_1 = B_1)$ and $(A_0 = B_0)$

$= (A_1 \odot B_1)(A_0 \odot B_0)$

2) If A is greater than B.

$(A_1 > B_1)$ or $(A_1 = B_1)$ and $(A_0 > B_0)$

$= A_1\bar{B_1} + (A_1 \odot B_1) \& A_0 \cdot \bar{B_0}$

3) If $A < B \Rightarrow (A_1 < B_1)$ or $(A_1 = B_1)$ & $(A_0 < B_0)$

$= \bar{A_1}B_1 + (A_1 \odot B_1)\bar{A_0}B_0$.

2 bit Magnitude Comparator:

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

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

$A > B = \bar{A}_1 A_0 \bar{B}_1 \bar{B}_0 + A_1 \bar{A}_0 \bar{B}_1 \bar{B}_0 + A_1 \bar{A}_0 B_1 \bar{B}_0 + A_1 A_0 B_1 \bar{B}_0 + A_1 A_0 \bar{B}_1 B_0 + A_1 A_0 B_1 \bar{B}_0$

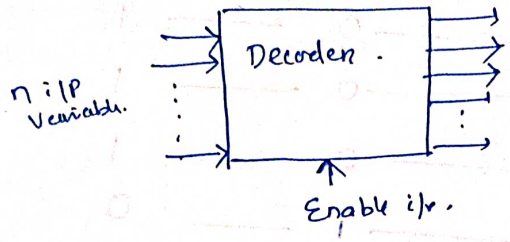
K map. 

$= \bar{A}_1 \bar{A}_0 B_0 + \bar{A}_1 B_1 + \bar{A}_0 B_1 B_0$

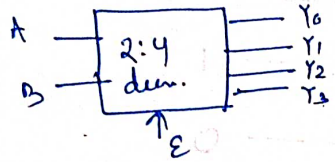
$A < B = \bar{A}_1 \bar{A}_0 \bar{B}_1 B_0 + \bar{A}_1 \bar{A}_0 B_1 \bar{B}_0 + \bar{A}_1 \bar{A}_0 B_1 B_0 + \bar{A}_1 A_0 B_1 \bar{B}_0 + \bar{A}_1 A_0 B_1 B_0 + A_1 \bar{A}_0 B_1 B_0$

By K map $\Rightarrow A_0 \bar{B}_1 \bar{B}_0 + A_1 A_0 \bar{B}_0 + \bar{A}_1 \bar{B}_1$

Decoder: It is a combinational logic ckt that converts binary information from n input lines to max^m 2^n unique o/p lines.

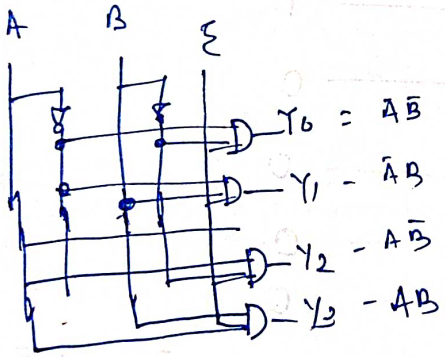


2:4 Binary decoder:



E	A	B	Y_3	Y_2	Y_1	Y_0
0	X	X	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

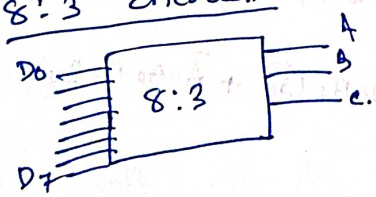
- If Y_0 is active when AB = 00 if $E = 1$
If $E = 0$ then whatever may be the comb^n of AB, all o/p = 0



Encoder: A ckt that generates a binary code at its o/p in response to one or more active input line.

- It's functional reverse of decoder fun^n.
- It has 2^n i/p & n o/p.

8:3 Encoder (Octal to Binary Encoder)

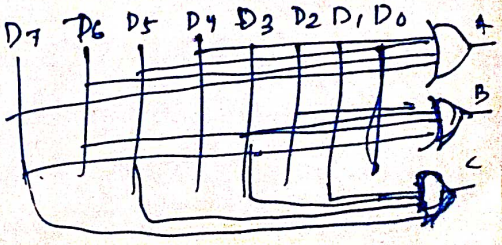


D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	A	B	C
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

$A = D_4 + D_5 + D_6 + D_7$

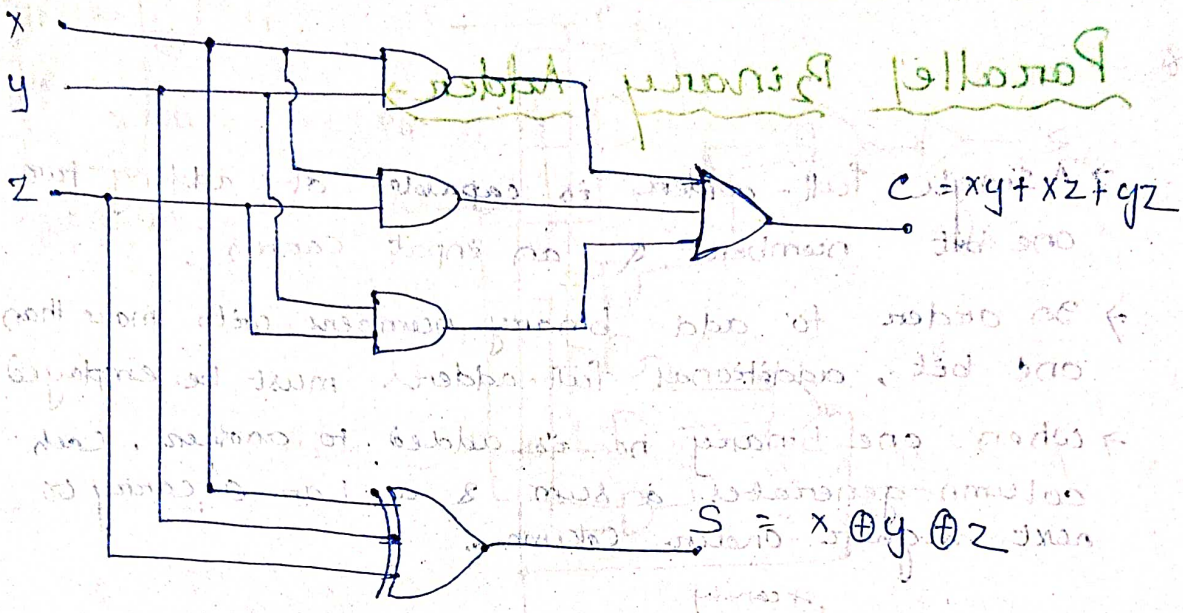
$B = D_2 + D_3 + D_6 + D_7$

$C = D_1 + D_3 + D_5 + D_7$.



$= X \oplus Y \oplus Z$

Logic Gates →



(full adder by Half adder) → class Note!

SUBTractors →

→ A subtraction of two binary numbers may be accompalished by taking the complement of subtrahend & adding it to the minuend.

→ By this method, the subtraction operation becomes an addition operation requiring full adders for its mechine implementation.

→ In subtraction if minuend bit is smaller than the subtrahend bit, a 1 is borrowed from the next significant position.

→ In circuitry a borrow is generated if subtrahend bit is greater than the corresponding minuend bit & this borrow is carried as carry to the next stage as its input & taken care in the subtraction stage of the next pairs of bits.

→ A subtraction may be dealt as full.

Half Subtractor →

* Parallel Binary Adder →

→ A single full-adder is capable of adding two one bit numbers & an input carry.

→ In order to add binary numbers with more than one bit, additional full-adders must be employed.

→ When one binary no is added to another, each column generates a sum & a 1 or 0 carry to next higher order column.

Ex

```
  ,→carry
   1 1
   0 1
 -------
   1 0 0
```

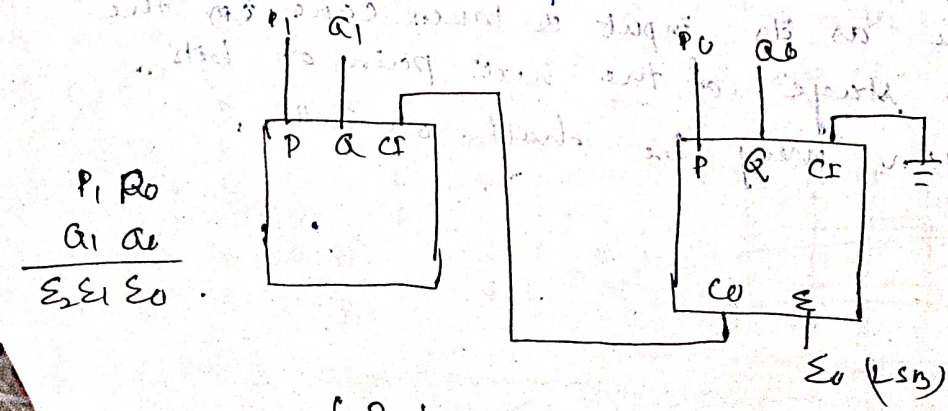
carry from 2nd column become sum bit.

→ To implement the addition of binary numbers with logic ckt, a full-adder is required for each column.

→ So for two-bit nos, two adders are needed for 3 bit no, 3 adders are used.

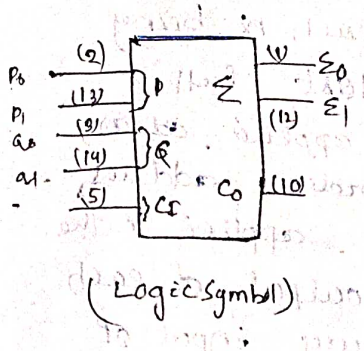
→ The carry output of each adder is connected to the carry input of the next higher-order adder.

→ Either a half adder can be used for the least significant position, or the carry input of a full adder is made 0, b'coz there is no carry into LSB position.

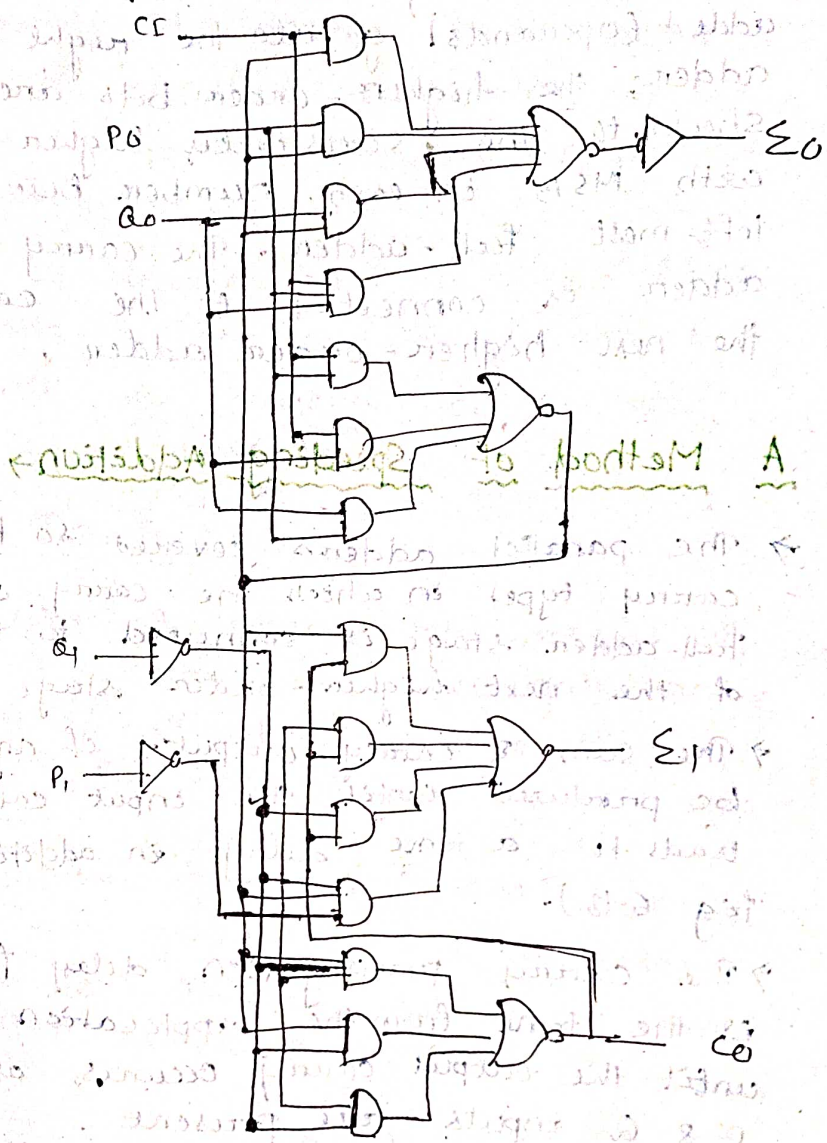


P1 P0
Q1 Q0
Σ2 Σ1 Σ0

(2 bit parallel adder)



(Logic Symbol)



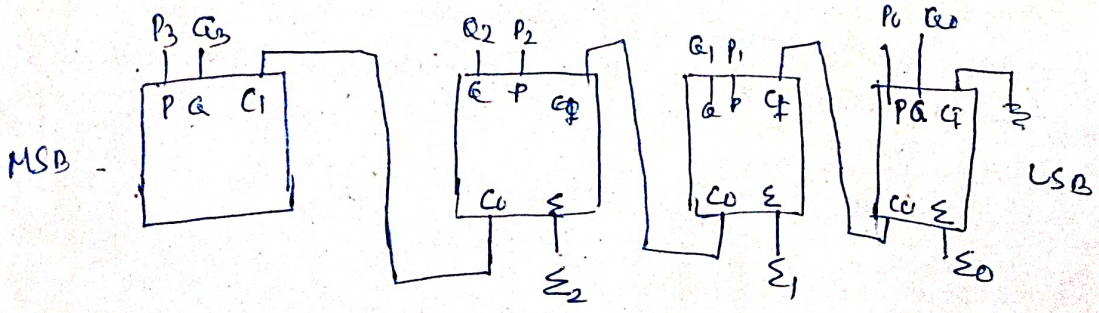
[Q] Verify that the two bit parallel adder properly performs the addition

$$\begin{array}{r} 11 \\ 10 \\ \hline 101 \end{array}$$

Ans: $P_3\ P_2\ P_1\ P_0$

$Q_3\ Q_2\ Q_1\ Q_0$

$\Sigma_3\ \Sigma_2\ \Sigma_1\ \Sigma_0$



(4 bit binary Adder)

A 4 bit binary adder LSB in each nr being added (operands) go into the right MOST full adder; the higher-order bits are applied as shown to the successively higher order adders, with MSB in each number being applied to the left-most full-adder. The carry output of each adder is connected to the carry input of the next higher-order adder.

* A Method of Speeding Addition →

→ The parallel adders covered so far are ripple carry types in which the carry output of each full-adder stage is connected to the carry input of the next higher-order stage.

→ The sum & carry outputs of any stage cannot be produced until the input carry occurs, this leads to a time delay in addition process (as shown in fig 6.13).

→ The carry propagation delay for each full adder is the time from the application of input carry until the output carry occurs, assuming that the P & Q inputs are present.

(Xerox).

* Magnitude comparator → Xerox

Decoder

Introduction →

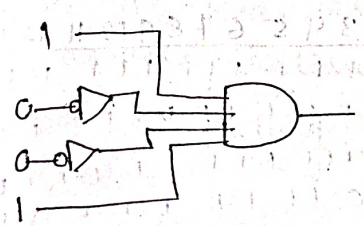
→ The basic function of decoder is to detect the presence of specified combination of bits on its inputs & to indicate that the presence by specified output level.

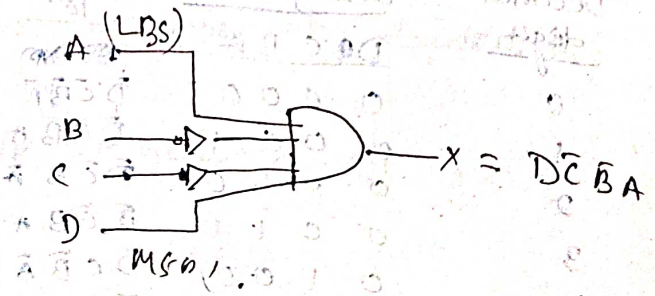
→ In general a decoder has n input lines to handle n bits & from one to 2^n output lines to indicate the presence of one or more n-bit combinations.

Basic Binary Decoder :-

→ To determine when a binary 1001 occurs on the inputs of a digital ckt, an AND gate can be used as the basic decoding element b/coz it produce a high o/p only when all of its i/ps are high.

→ Therefore, we must make sure that all of the inputs to the AND gate are high when the binary number 1001 occurs: this can be done by inverting the two middle bits (the 0s), as shown below:-



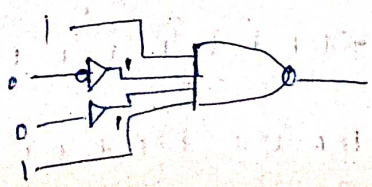


→ The logic eqn for decoder → The output fun^n o/p is 1 only when A=1, B=0, C=0 & D=1 are applied to the inputs.

→ A is LSB & D is the MSB.

→ LSB is always the right most bit in horizontal arrangement & top-most bit in a vertical arrangement.

By NAND Gate

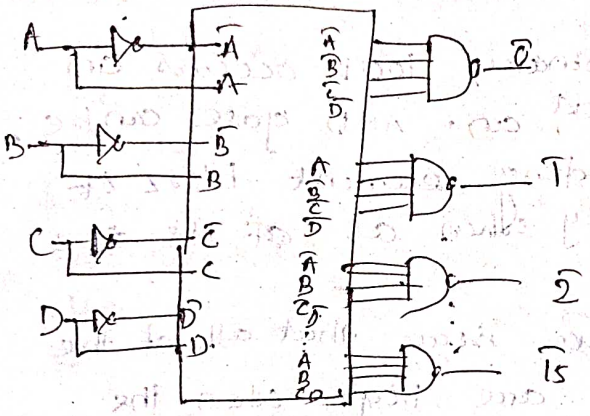


The Four-Bit Binary Decoder →

→ In order to decode all possible combinations of four bits, sixteen decoding gates are required ($2^4 = 16$).

→ This type of decoder is commonly called a 4 line to 16 line decoder b'coz there are 4 i/ps & 16 outputs.

→ If an active-Low o/p is desired for each decoded number, the entire decoder can be implemented with NAND gates & inverters

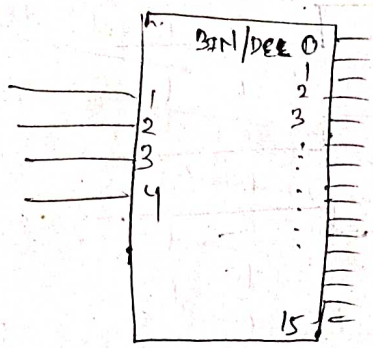


Truth table →

Decimal digit	Binary inputs D C B A	Logic fn	Outputs 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
0	0 0 0 0	$\bar{D}\bar{C}\bar{B}\bar{A}$	0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1	0 0 0 1	$\bar{D}\bar{C}\bar{B}A$	1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1
2	0 0 1 0	$\bar{D}\bar{C}B\bar{A}$	1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1
3	0 0 1 1	$\bar{D}\bar{C}BA$	1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1
4	0 1 0 0	$\bar{D}C\bar{B}\bar{A}$	1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1
5	0 1 0 1	$\bar{D}C\bar{B}A$	1 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1
6	0 1 1 0	$\bar{D}CB\bar{A}$	1 1 1 1 1 1 0 1 1 1 1 1 1 1 1 1
7	0 1 1 1	$\bar{D}CBA$	1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 1
8	1 0 0 0	$D\bar{C}\bar{B}\bar{A}$	1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1
9	1 0 0 1	$D\bar{C}\bar{B}A$	1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1
10	1 0 1 0	$D\bar{C}B\bar{A}$	1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1
11	1 0 1 1	$D\bar{C}BA$	1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1
12	1 1 0 0	$DC\bar{B}\bar{A}$	1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1
13	1 1 0 1	$DC\bar{B}A$	1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1
14	1 1 1 0	$DCB\bar{A}$	1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1
15	1 1 1 1	$DCBA$	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0

→ In order to decode each of the 16 binary code words, 16 NAND gates are required (AND gates can be used to produce active-High o/ps).

→ A simple logic symbol for 4-line to 16 line decoder is as



The BCD-to-Decimal Decoder →

→ The BCD to decimal decoder converts each BCD code word (8421 code) into one of ten possible decimal digit indication.

→ It is frequently referred to as 4 line to 10 line decoder.

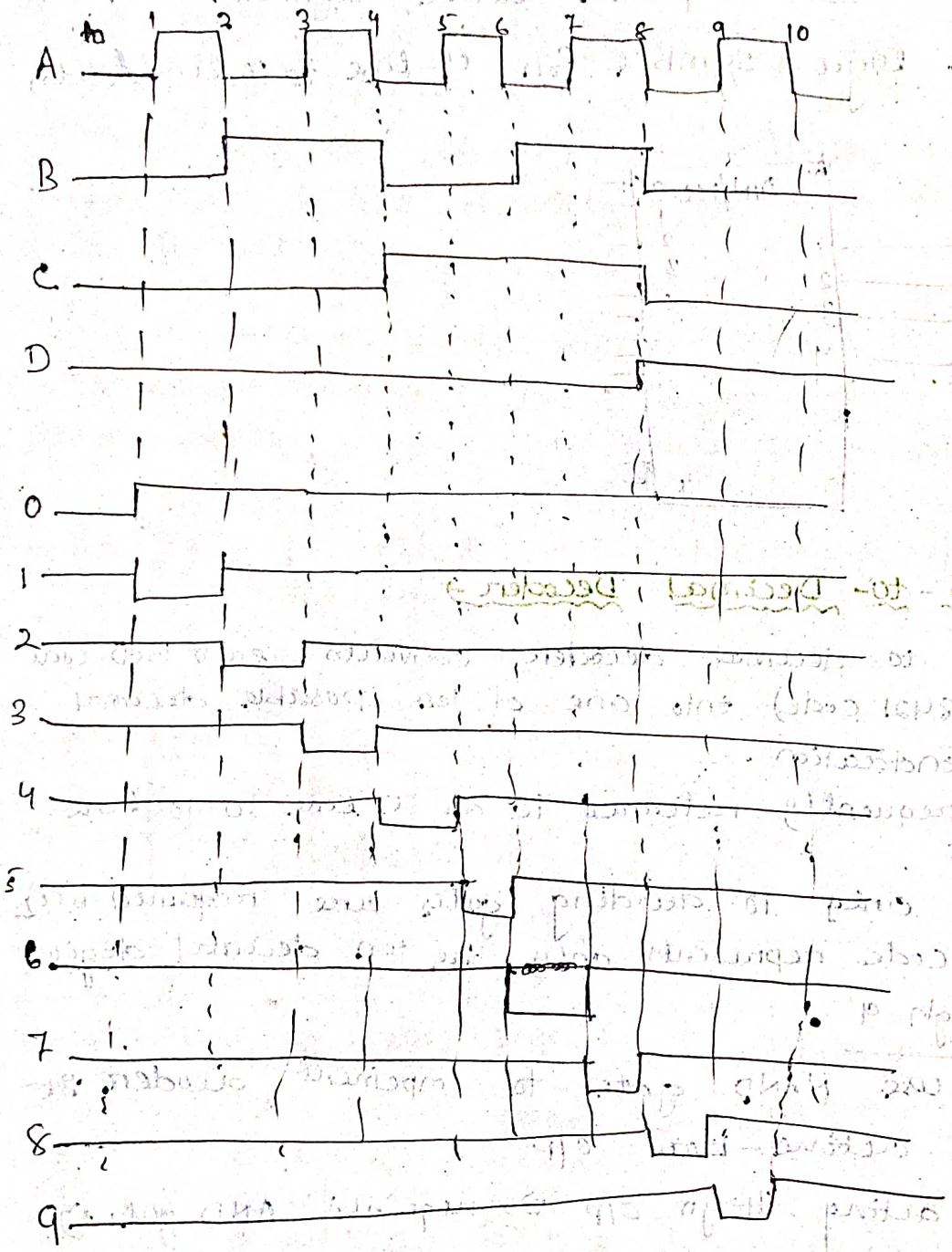
→ In this only 10 decoding gates are required b'cz the BCD code represents only the ten decimal digits 0 through 9.

→ If we use NAND gate to implement decoder, it provide active-Low o/p.

→ If an acting High o/p is required AND gate is used.

Truth table →

Decimal digit	BCD code B	C	B	A	Logic Fun
0	0	0	0	0	$\bar{D}\bar{C}\bar{B}\bar{A}$
1	0	0	0	1	$\bar{D}\bar{C}\bar{B}A$
2	0	0	1	0	$\bar{D}\bar{C}B\bar{A}$
3	0	0	1	1	$\bar{D}\bar{C}BA$
4	0	1	0	0	$\bar{D}C\bar{B}\bar{A}$
5	0	1	0	1	$\bar{D}C\bar{B}A$
6	0	1	1	0	$\bar{D}CB\bar{A}$
7	0	1	1	1	$\bar{D}CBA$
8	1	0	0	0	$D\bar{C}\bar{B}\bar{A}$
9	1	0	0	1	$D\bar{C}\bar{B}A$



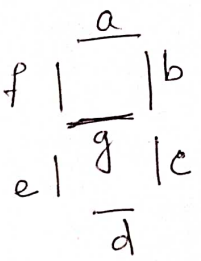
* BCD to 7 segment Decoder/Driver

Xerox :

Seven Segment Decoder:

- A BCD to Seven Segment decoder driver ICs are used to convert the BCD signal into form suitable for driving these display.

- The binary 1 is activated the LED for corresponding digit



Truth table:

Digit	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	1	0	1
10	1	0	1	0	d	d	d	d	d	d	d

$a = \Sigma_m(0,2,3,5,6,7,8,9)$

$b = \Sigma_m(0,1,2,3,4,7,8,9)$

$c = \Sigma_m(0,1,3,4,5,6,7,8,9)$

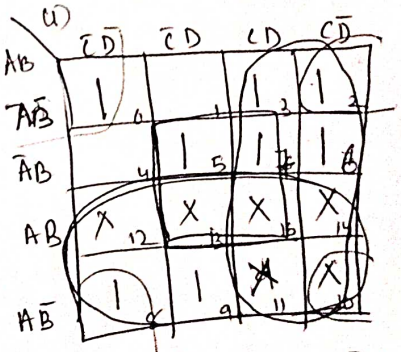
$d = \Sigma_m(1,2,3,5,6,8,9)$

$e = \Sigma_m(0,2,6,8,)$

$f = \Sigma_m(0,4,5,6,8,9)$

$g = \Sigma_m(2,3,4,5,6,8,9)$

(a)



$a = B\bar{D} + e + BD + A$

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

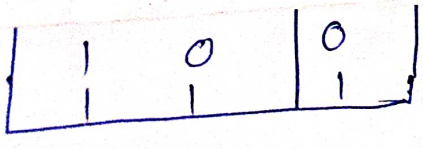
$e = B + \bar{e} + D$

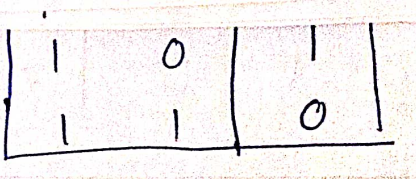
$d = \bar{B}\bar{D} + C\bar{D} + B\bar{C}D + \bar{B}C$

$e = B\bar{D} + C\bar{D}$

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

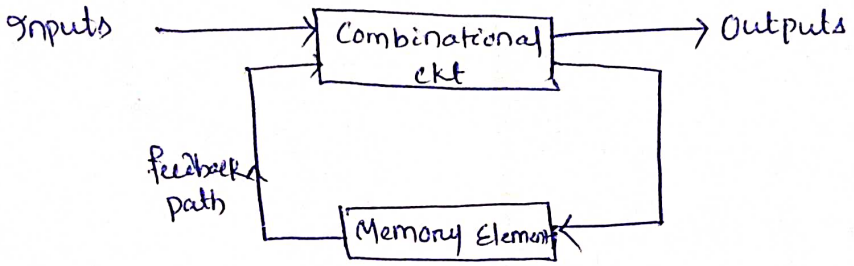
$g = A + B\bar{C} + \bar{B}e + CD$





Sequential Circuit:

- A ckt whose output depends not only on the present input, but also on the past inputs.
- It consists of a combinational ckt to which storage elements are connected to form a feedback path.



Types: The sequential ckts can be classified in two ways depending on the timing of their signal.

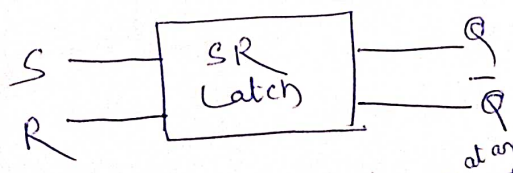
1. Synchronous Sequential ckt.
2. Asynchronous Sequential ckt.

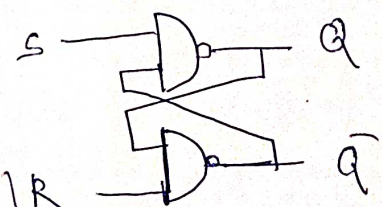
Latches:

* Two cross couple NAND or NOR is known as latches.
- These are basic ckts from which all flip flops are constructed.
- Lathes are useful for storing binary information.

SR Latch Using NAND

→ Two cross couple NAND gate is called Latch.
→ It has two inputs S & R and two output Q & $\bar{Q}$.



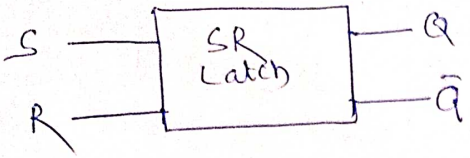


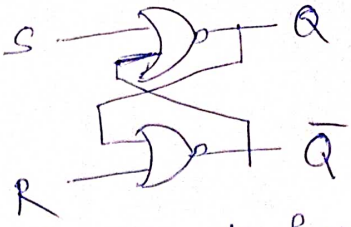
→ We know that a logic 0 at any i/p of a NAND gate force its output to a logic 1 immediately.

Truth table:

S	R	Q	$\bar{Q}$	Comment
0	0 →	–	–	Invalid
0	1 →	1	0	Set
1	0 →	0	1	Reset
1	1 —	Q0	$\bar{Q_0}$	No change

SR Latch Using NOR Gate: (2)





We know that logic 1 at any input of NOR gate force its output to a logic 0 immediately.

Truth table:

S	R	Q	$\bar{Q}$	comment
0	0	Qn	$\bar{Q}n$	No change.
0	1	1	0	Set
1	0	0	1	Reset
1	1	—	—	Invalid

When both inputs are 1, they force the o/p of both NOR gate to logic 0 i.e $Q = 0$, $\bar{Q} = 0$. As Q & $\bar{Q}$ complement of each other, it contradict so it is a invalid case.

Flip flops: Flip Flop is a circuit that that has two stable state & can be used to store ~~information~~ a single bit.

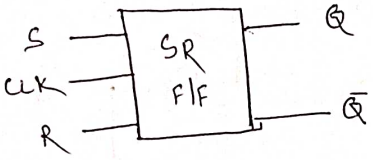
Types of Flip flop: There are 4 types of Flip flop.

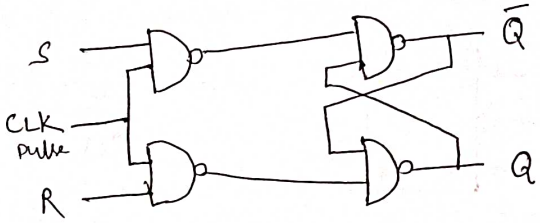
1. SR Flip flop
2. JK Flip flop
3. D Flip flop
4. T Flip flop.

Clocked SR F/F:

A SR flip flop can be converted into a clocked SR F/F by using additional NAND or AND gate & providing a clock input.

- A clocked SR F/F operates in step with the clock i.e operates synchronously.



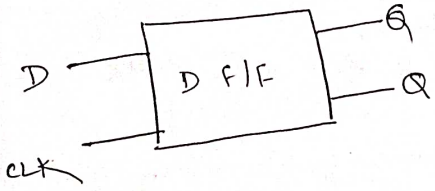


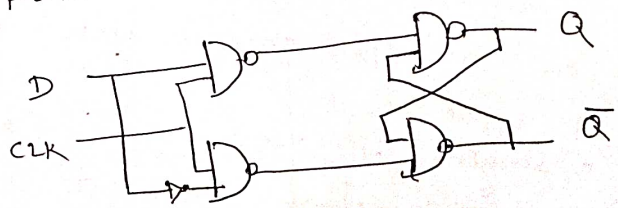
Truth table:

Input			Output		Mode of operation
CLK	S	R	Q	$\bar{Q}$	
1	0	0	Q$_{n-1}$	$\bar{Q}_{n-1}$	No change or Hold
1	0	1	0	1	Reset
1	1	0	1	0	Set
1	1	1	1	1	Invalid

The D F/F:

- A modified form of clocked SR F/F is D flip flop.
- A D F/F has only one data input.
- D F/F allows the value of D to reach the output only when a clock pulse occurs & prevent the value of D to reach output when there is no clock pulse.





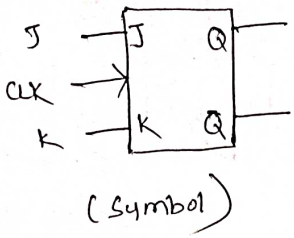
Truth table:

Input		output		comment
CLK	D	Q	$\bar{Q}$	
0	X	—	—	Last latch
1	0	0	1	Reset
1	1	1	0	Set.

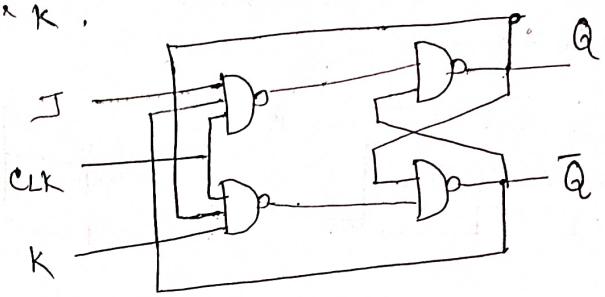
JK FlF :

→ JK FlF is the improved version of SR FlF so that when both inputs are 1 then also outputs Q & $\bar{Q}$ are complement of each other.

- JK FlF has two input J & K.



(Symbol)



Truth table:

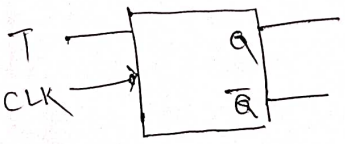
input		output		comment
J	K	Q.	$\bar{Q}$	~~No change~~
0	0	Q_0	$\bar{Q}_0$	No change.
0	1	0	1	Reset
1	0	1	0	Set
1	1	$\bar{Q}_0$	Q_0	Toggle

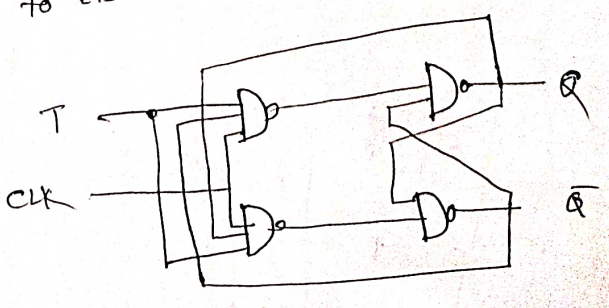
→ when J & K both are high, the output will change to its complement. i.e 0 will become 1 & 1 will become 0.

T FlF

→ A toggle on T FlF changes state each time a pulse occurs on its T input.
→ It is a single i/p version of JK FlF.
→ If i/p of T is high on 1 on arrival of clock pulse the output is changed to its complement.

Symbol





Truth table.

T	Q $\bar{Q}$	Comme
0	Q $\bar{Q}$	No C
1	$\bar{Q}$ Q	Toggle

~~Master slave FF~~

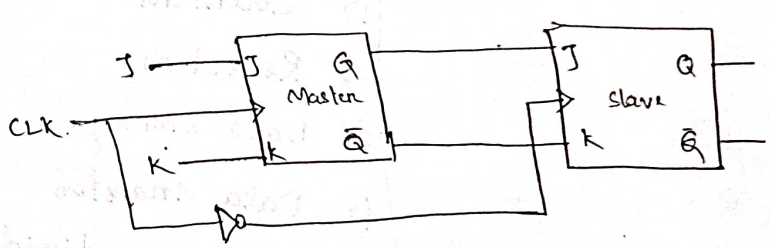
Race- Around Condition / Racing condition:

→ A race condition is said to exist when two or more binary state variables change value in response to a change in an input variables.

Ex: When S & R inputs of an SR FF is at logic 1 & then i/p is changed to any other condition, then the o/p becomes unpredictable & this is called race condition.

JK Master Slave FF: →

→ It is constructed from two FFs. One FF serves as master & other as slave.



→ This FF requires two JK FFs one is called master & other is called slave.

→ The master is positively clocked. Due to presence of inverter, the slave is negatively clocked.

→ When the clock is high, the master is active & the slave is inactive.

- when CLK is low, the master is inactive & the slave is active.

→ When J=1 & K=0, the master sets as soon as positive edge of clock signal occurs forcing the output to be high.

→ The high o/p (Q=1) of the master drives the J input of the slave.

→ Hence when a -ve edge of clock signal occurs, the slave sets doing the same as master did or copies the master.

* If J=0 & K=1, the master resets on the occurance of the positive edge of the CLK ~~pulse~~ signal which forces $\bar{Q}$ to be 1 or Q=0.

→ The high $\bar{Q}$ output of the master drives the K input of slave. Hence when a -ve edge of the clk signal occurs it forces the slave to reset. Also slave copies the master.

- If J=1 & K=1, both inputs are high for the master, it toggles on arrival of positive clk edge & the slave toggles on occurance of -ve clock edge.

Truth table:

CLK	J	K		Q.	comment
1	0	0	—	~~Q~~ Q	N.C
1	0	1	—	0	Reset
1	1	0	—	1	Set
1	1	1	—	$\bar{Q}$	toggle.

Application of flip flop

1) Counters
2) Registers
3) Data storage
4) Data transfer
5) Frequency dividers.

Here:

Counters:

→ Counter is an sequential ckt capable of counting the number of clock pulses arriving at its clock input.

→ A counter is a set of flipflops whose state change in response to pulse applied at the input to the counter.

→ It can be used as a frequency divider.

- A counter that follows the binary sequence is called a binary counter.
- An n bit counter consists of n flip flop & can count in binary from 0 to $2^n - 1$.
- It is also called modulus counter or MOD-N counter where N = number of states.

Types of Counter:

Counters are classified into two types.

i) Ripple or Asynchronous counters.

ii) Synchronous counter.

→ In synchronous counter, the common clock input is connected to all the flip flops & thus they are clocked simultaneously.

→ In asynchronous counters each flf is triggered by the previous flf & the 1st flf is triggered by External clk pulse.

i) Asynchronous Counter:

→ To design the ripple counter, the number of flfs required depends on the number of states.

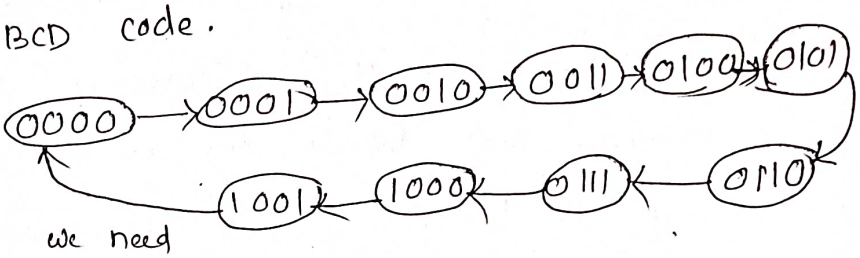
→ The max^m number of states of a counter is 2^n, where n is the number of flf in the counter.

Mod-10 Ripple Counter / Decade counter / BCD Counter:

→ A decimal counter follows a sequence of ten states & returns to 0 after the count of 9.

* A counter with a count sequence 0(0000) through 9(1001) is called a BCD counter because its ten states sequence is the BCD code.

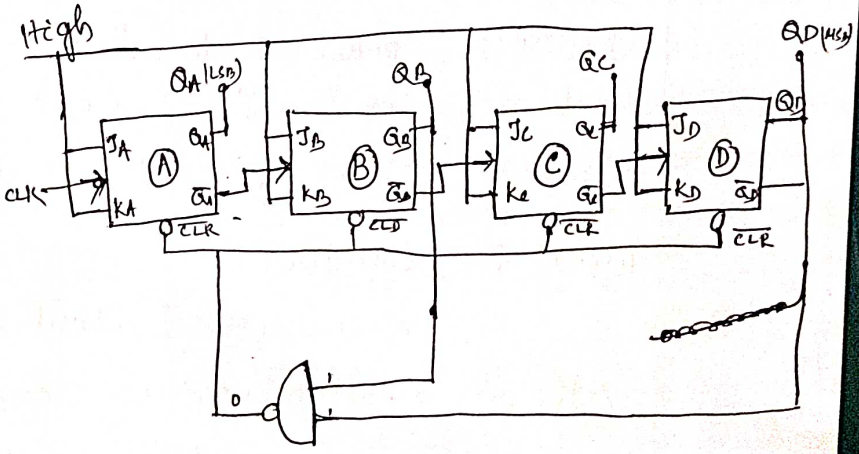
State diagram:



→ To design a BCD counter we need 4 FFs.

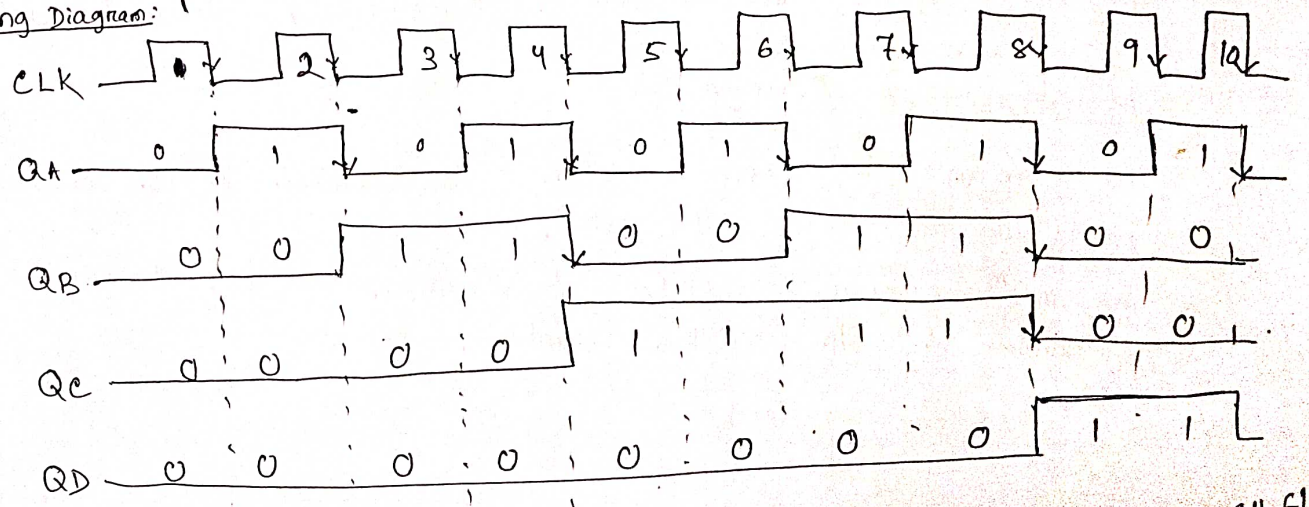
Truth table:

CLK	Q_D	Q_C	Q_B	Q_A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1



* The CLK input is connected to the 1st FF only. The 2nd, 3rd & 4th FF are triggered by $\overline{Q_A}$, $\overline{Q_B}$ & $\overline{Q_C}$ Respectively.

Timing Diagram:



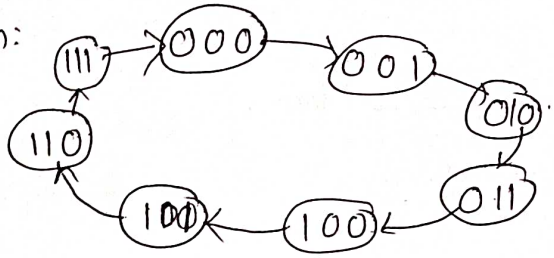
* When the counter goes to count 1010, the NAND o/p goes low & all FFs are in clear condition. So the counter o/p $Q_D Q_C Q_B Q_A$ = 0000 instead of 1010.

3 bit Asynchronous Counter | Mod 8

Asynchronous counter

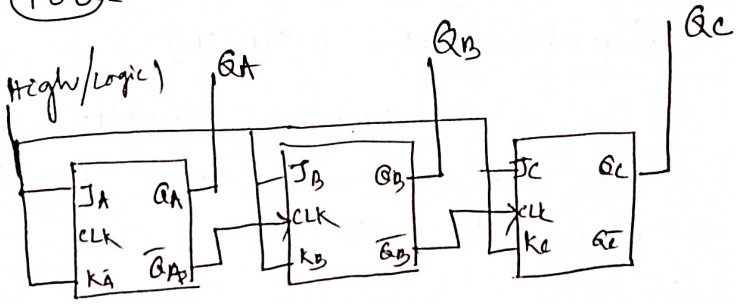
→ 3 bit asynchronous counter requires 3 FFs & count sequence starts from 000 to 111.

State diagram:



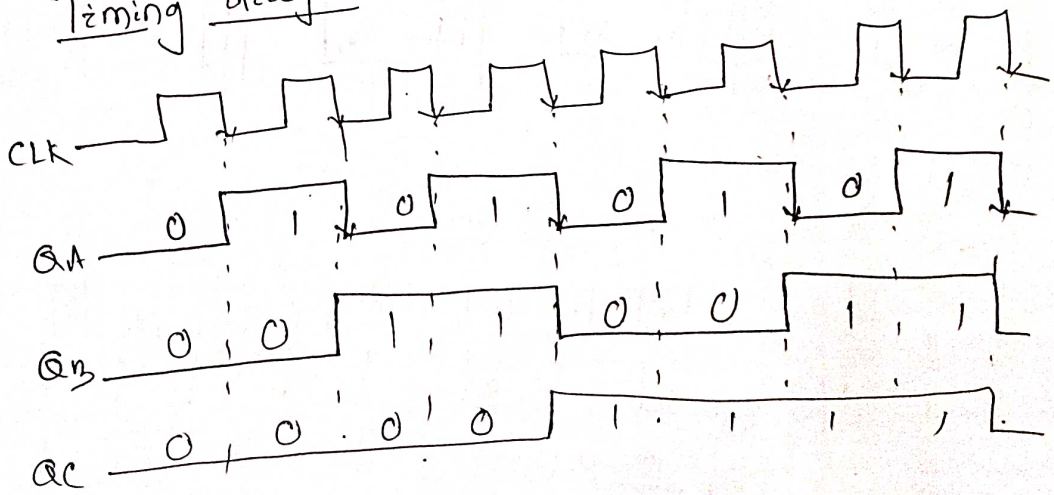
Truth table

CLK	Q_C	Q_B	Q_A
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1



- Here 1st FF is triggered by the CLK which is holding LSB.
- The 2nd FF is triggered by $\overline{Q_A}$ & 3rd FF is triggered by $\overline{Q_B}$.

Timing diagram:



Advantages of ASynchronous Counter:

i) It is very simple & straight forward operation & construction.
ii) Require fewer components.
iii) Very easy to design.

Disadvantages:

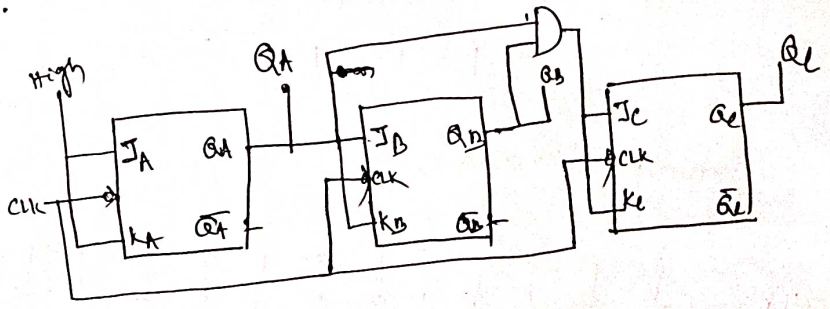
i) Speed of operation is ~~very~~ low.
ii) The propagation delay.

Synchronous Counter:

→ In a synchronous counter, all the FFs ~~are~~ change their state simultaneously, the operation of each stage being initiated by CLK.

3 bit Synchronous Counter:

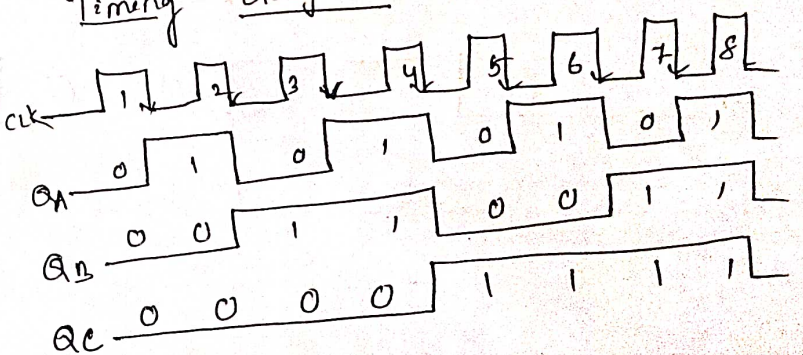
→ 3 bit synchronous counter has 3 J-K FFs.
→ Here CLK signal is connected in parallel to the CLK inputs of all FFs.



State Sequence:

CLK	Q_C	Q_B	Q_A
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Timing diagram:



→ FF C has to change its state only when Q_A & Q_B both are at logic 1. This condition is detected by AND gate applied to J_C & K_C i/p of FF C.

Up and Down Counter:

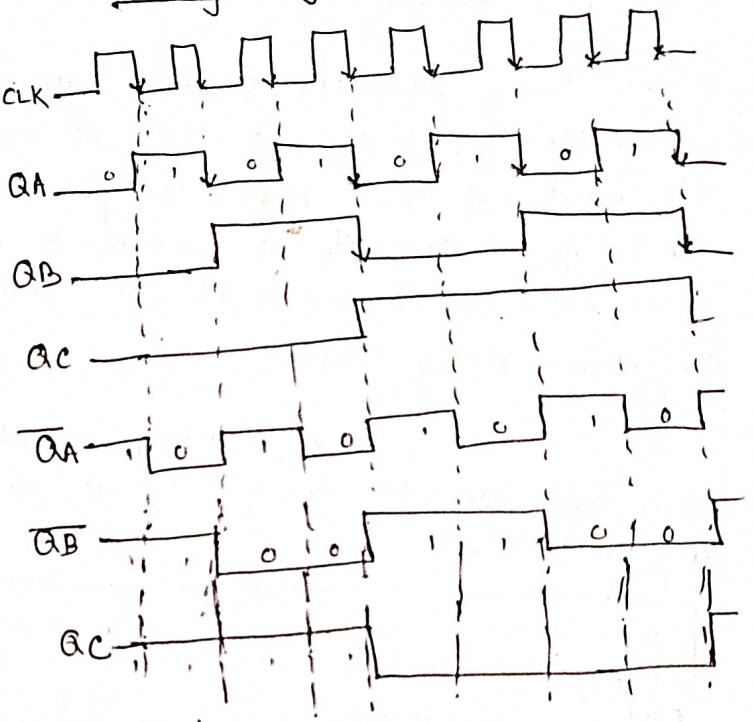
→ A counter which can count in upward or downward sequence is called an up down counter.

Ripple Mod 8 counter:

Truth table:

Count	F/F output						Count
	Q_C	Q_B	Q_A	$\bar{Q}_C$	$\bar{Q}_B$	$\bar{Q}_A$	
0	0	0	0	1	1	1	7
1	0	0	1	1	1	0	6
2	0	1	0	1	0	1	5
3	0	1	1	1	0	0	4
4	1	0	0	0	1	1	3
5	1	0	1	0	1	0	2
6	1	1	0	0	0	1	1
7	1	1	1	0	0	0	0

Timing diagram:

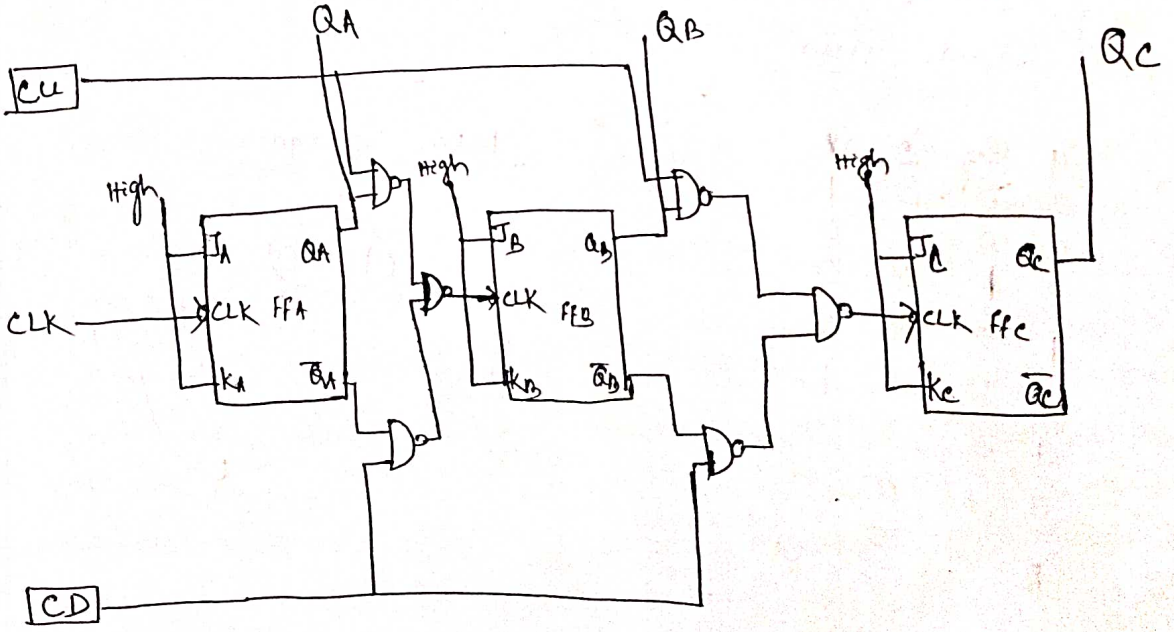


→ A ripple counter can be used as up or down counter by using two additional signal count up (CU) & count down (CD)

→ When CU is high or 1, the count sequence is up & when CD is high or 1, the count sequence is down

→ So CU & CD are complement ~~with~~ each other.

→ When CU = 1, the Q output fed as CLK pulse for next stage of F/F & CD = 1, $\bar{Q}$ is fed as the CLK pulse to next F/F.

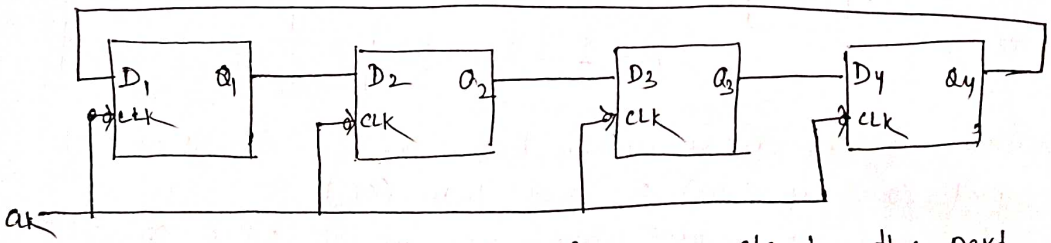


Ring Counter:

→ A shift register can also be used as counter.
→ It can obtained from serial in, serial out shift register by providing feedback from the o/p of last flf to the i/p of 1st flf.
→ The most widely used shift register counters are i) Ring counter ii) Johnson counter.

→ In Ring counter f/fs are arranged as in normal shift register i.e the Q o/p of each stage is connected to the D input of the next stage.
→ The Q o/p of last flf is connected back to D i/p of the 1st flf such that the array of flf is arranged in a ring & therefore the name ring counter.

* A ring counter is a circular shift register with only one flf being set at any particular time & all others are cleared.

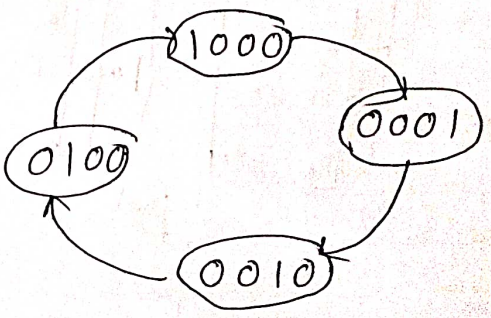


→ The single bit is shifted from one flf to the next to produce the sequence of timing signals.
- The initial value of the register is 1000.
→ The single bit is shifted right with every CLK pulse & circulated back from Q4 to Q1.

Sequence table:

CLK	Q4	Q3	Q2	Q1
0	0	0	0	1
1	0	0	1	0
2	0	0	0	0
3	1	0	0	0
4	0	0	0	1
5	0	0	1	0
6	0	1	0	0
7	1	0	0	0

State diagram:



i/p of flfs.

Shift Register:

→ A register is a group of flip flops used to store binary bits.

→ An n register consists of n flfs & is capable of storing information of n bits.

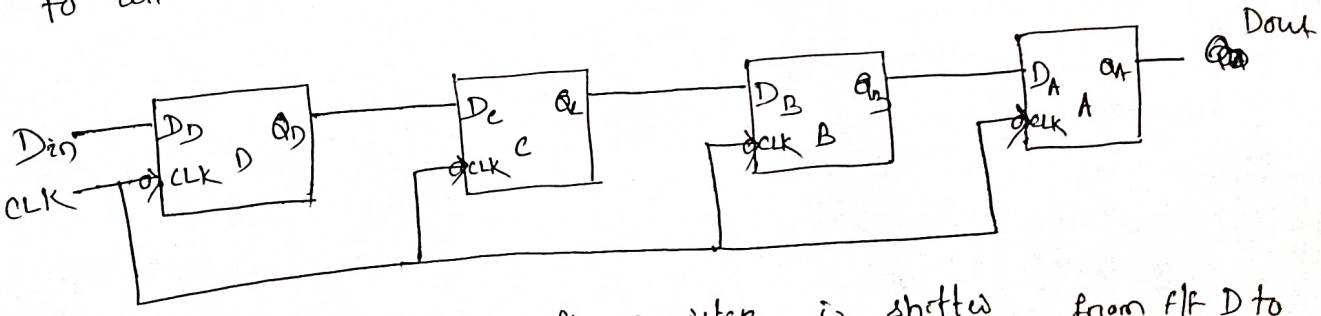
→ A synchronous sequential ckt that will store & move nbit data either serially or in parallel in n flfs is called shift Register.

→ There are 5 types of shift Register.

i) Serial in - Serial out shift Register (SISO)
ii) Serial in Parallel out " " (SIPO)
iii) Parallel in Serial out " " (PISO)
(iv) Parallel in Parallel out " " (PIPO)
(v) Bidirectional shift Register.

i) Serial in - Serial out shift Register (SISO):

→ It consists of 4 D flf. CLK pulse applied simultaneously to all the flfs.



→ The data stored in shift register is shifted from flf D to C to B and finally through F/F A to the output as a stream of data bits.

- So the data output from the shift register is obtained serially.
- So this type of shift register is known as SISO Register.

Solution

(5) Flip Flop Conversion (1)

Flip flop characteristic table:

SR Flip flop:

S	R	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	–

JK FIF

J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	$\bar{Q}_n$

D FIF

D	Q_{n+1}
0	0
1	1

or

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

T FIF

T	Q_{n+1}
0	Q_n
1	$\bar{Q}_n$

or

T	Q_n	Q_{n+1}
0	X	Q_n
1	X	$\bar{Q}_n$

Here $Q_n \rightarrow$ Present State
$Q_{n+1} \rightarrow$ Next State.

Flip flop excitation table:

SR FIF

Q_n	Q_{n+1}	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

JK FIF

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

D FIF

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

T FIF

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

Steps for conversion:

1) Consider truth table of destination flip flop & Extend it as excitation table of source flip flop.
2) Combine truth table & excitation table to form conversion table.
3) Draw k map for source flip flop input variables, it will provide Expression for conversion.
4) Draw the ckt according to k map Expression.

Conversion of SR FlF to D FlF:

Here destination FlF = D FlF
Source.. FlF = SR FlF

Step-1

Truth Table of D FlF

D	Qn	Qn+1
0	0	0
0	1	0
1	0	1
1	1	1

or

D	Qn	Qn+1
0	X	0
1	X	1

Excitation table of SR FlF

Qn	Qn+1	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

Step-2

Conversion table:

Truth table of D

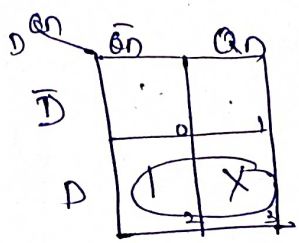
D	Q_n	Q_{n+1}	S	R
0	0	0	0	X
0	1	0	0	1
1	0	1	1	0
1	1	1	X	0

Excitation of SR

* Draw K map of S & R. Here input is D & Q_n. Always Q_{n+1} is the output.

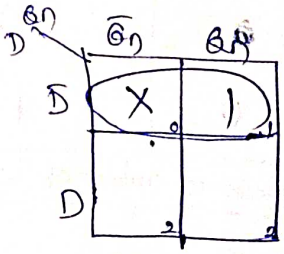
Step-3

K map of S



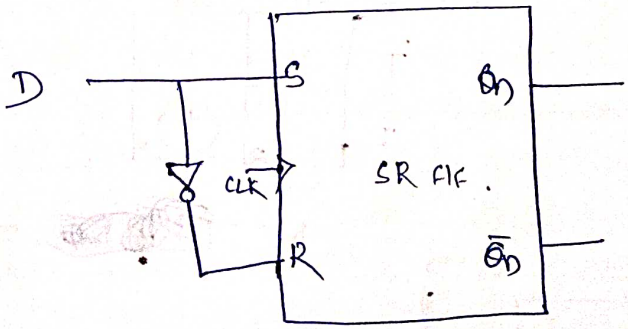
$S = D$

K map for R



$R = \bar{D}$

Step-4



SR FlF to T FlF

Step-1

Truth table of T FlF.

T	Qn	Qn+1
0	0	0
0	1	1
1	0	1
1	1	0

Excitation table of SR FlF

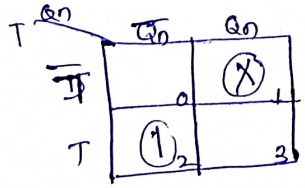
Qn	Qn+1	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

Step-2 Conversion table

Truth table of T				
T	Qn	Qn+1	S	R
0	0	0	0	X
0	1	1	X	0
1	0	1	1	0
1	1	0	0	1
	Excitation table of SR			

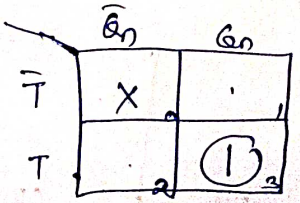
Step-3

K map for S



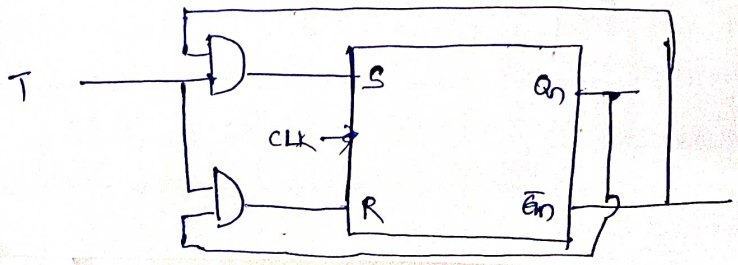
$S = T\bar{Q}_n$

K map for R



$R = TQ_n$

Step-4



S.R FlF to JK FlF:

Step-1

Truth table of JK

J	K	Qn	Qn+1
0	0	X	Qn
0	1	X	0
1	0	X	1
1	1	X	$\bar{Q}_n$

Excitation table of SR

Qn	Qn+1	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

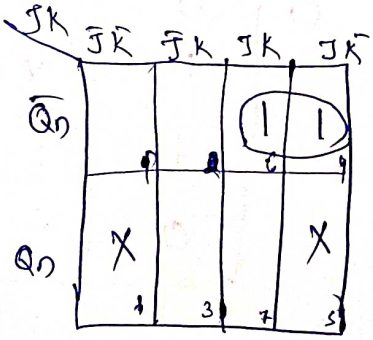
Step-2

Conversion table:

Truth table of JK					
J	K	Qn	Qn+1	S	R
0	0	0	0	0	X
0	0	1	1	X	0
0	1	0	0	0	X
0	1	1	0	0	1
1	0	0	1	1	0
1	0	1	1	X	0
1	1	0	1	1	0
1	1	1	0	0	1
Excitation table of SR					

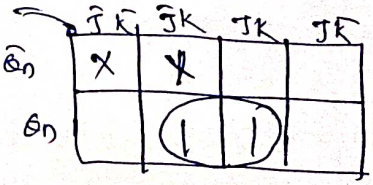
Step-3

K map for S

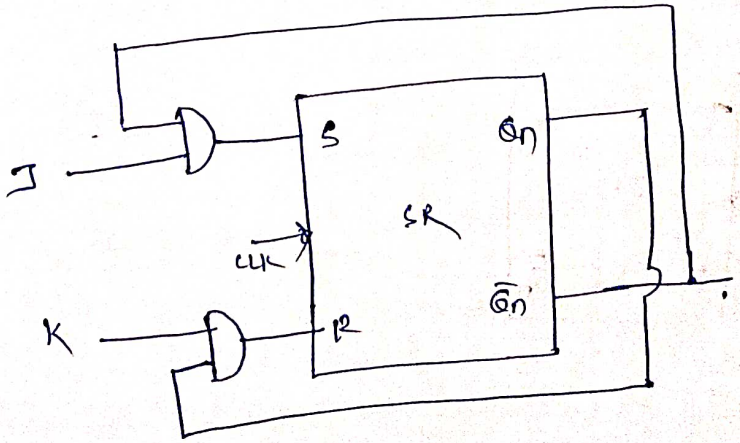


$S = \bar{Q}_n J$

K map for R



$R = Q_n K$



JK FlF to T FlF:

Step-1

Truth table of T

T	Qn	Qn+1
0	X	Qn
1	X	$\bar{Q_n}$

Excitation table of JK FF

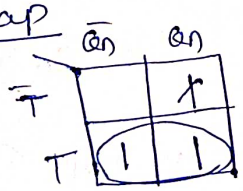
Qn	Qn+1	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Step-2

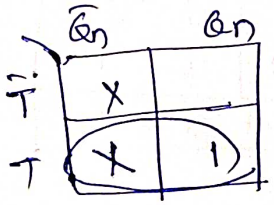
Conversion table

T	Qn	Qn+1	J	K
0	0	0	0	X
0	1	1	X	0
1	0	1	1	X
1	1	0	X	1

Step-3 K map

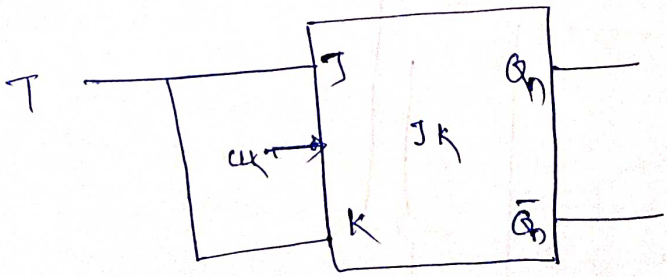


J = T



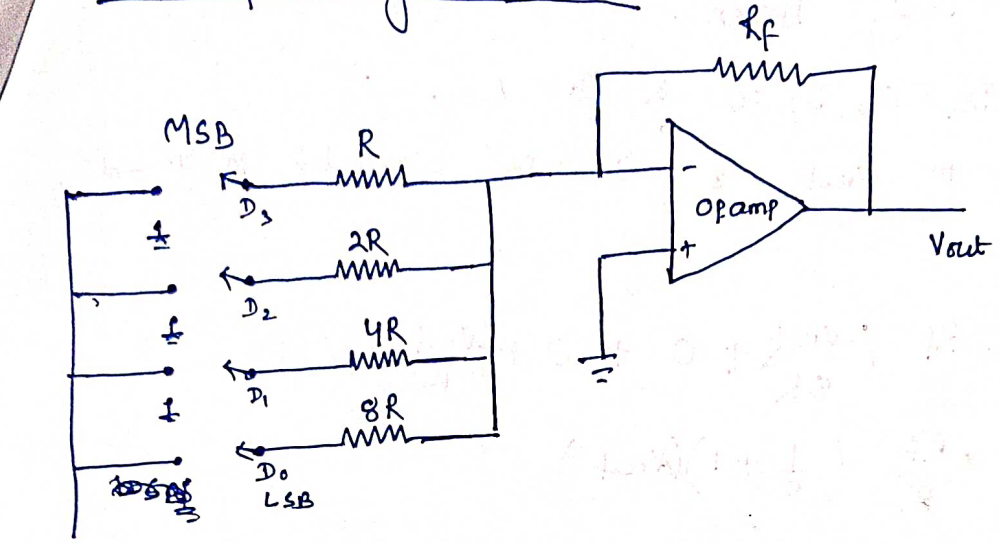
K = T

Step-4



DAC

1) Binary Weighted Resistor:



- Basically, DAC is a digital code that represents digital value converted to analog voltage or current.
- Consider a 4bit DAC binary code, the digital input terminal D_3, D_2, D_1, D_0.
- The resistor network is the main structure of DAC ckt.
- Binary weights depends on the no of bits we want to convert. Ex - if we want to convert 4 bit data then ~~we have 8R registers.~~ we use upto 2^3R resistor. If we convert N bit data then we use $2^{N-1}R$ resistor.
- Digital data is given. if digital data is 1 then switch is connected to -Vref & if digital data is 0 then switch is connected to ground terminal.
- The output of an op-amp

$$V_{out} = -\frac{R_f}{R_{in}} V_{in}$$

Here R_{in} ⇒ Input Resistance, R_f = feedback Resistance, V_{in} = input voltage, V_o = output voltage.

(b) Generally $\boxed{R_f = R}$

Let us consider the 4 bit data is 1001,

$V_{ref} = 16V$.

Here $D_3 = 1$, $D_2 = 0$, $D_1 = 0$ & $D_0 = 1$.

So D_3, D_0 are connected to V_{ref} & D_2 & D_1 connected to ground.

So

$$V_{out} = -R_f \left(\frac{-V_{ref}}{8R} + 0 + 0 + \left(\frac{-V_{ref}}{R}\right)\right)$$

$$= +\frac{R_f}{R}\left(\frac{1}{8} + 1\right)(V_{ref})$$

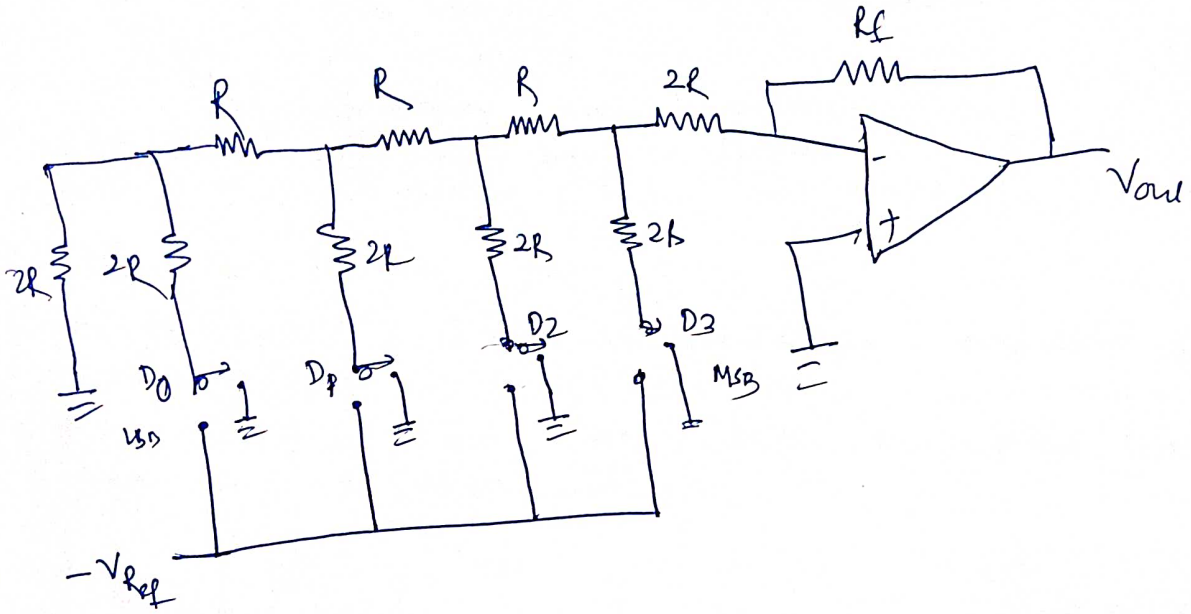
$$= +\frac{9}{8} V_{ref}$$

$$= \frac{9}{8} \times 16 = 18\,V.$$

Sairam

Converters

R - 2R Ladder DAC



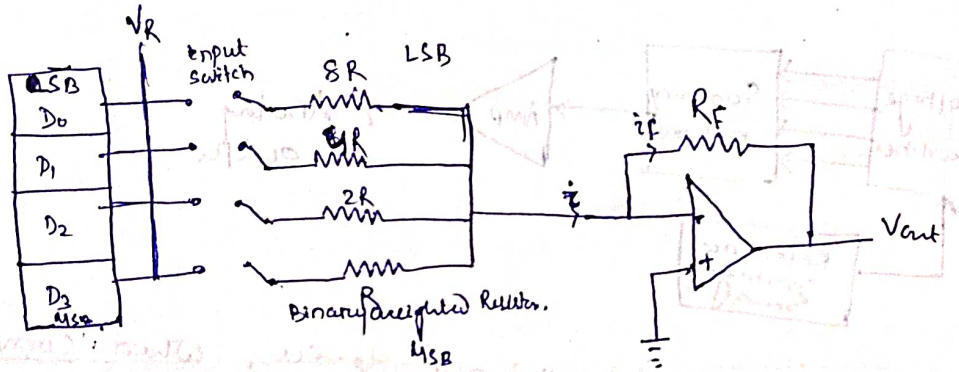
Types of DAC:

The conversion of digital to analog can be achieved by using several techniques such as

i) Binary weighted network
ii) Binary weighted ladder
iii) Binary weighted quad groups

i) Weighted Resistor DAC:

- The most simple & straight forward method of converting digital to analog involves the use of resistor network.
- The resistor used in this network weight the current passing through them which are then summed up.



- It is a 4 bit DAC uses 4 inputs.
- The resistances used has the value that represent the binary weights of the input bits of the digital code.
- The switch symbol represents transistor switches for inputting each of the four bits.
- R_f is the feedback resistance.
- The operational amplifier offers a very high i/p impedance & its inverting input looks like ~~virtual~~ virtual ground.
- As a result the current through the feedback resistance is sum of the input currents & the output is proportional to the current through R_f.
- The high binary-weighted input or MSB input (2^3) corresponds to lowest value ~~resistor~~.
- The other resistors are multiples of R i.e 2R, 4R, 8R corresponds to the binary weight 2^0, 2^1, 2^2, 2^3 respectively.
- If V_R is the reference voltage, then the current in the feedback resistor when a state 1000 is at digital input.

$$I = \frac{V_R}{R}$$

- If the digital input is 1111, then current through V_R.

$$I = V_R \left[\frac{1}{R} + \frac{1}{2R} + \frac{1}{4R} + \frac{1}{8R} \right]$$

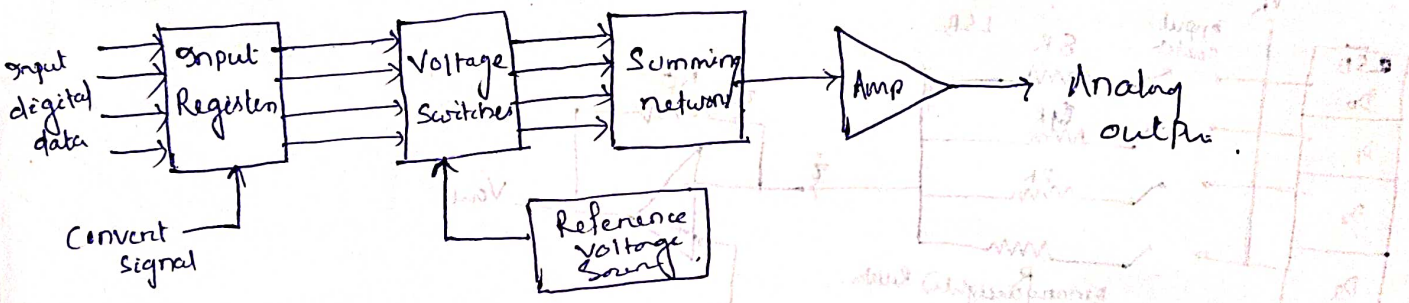
$$= \frac{V_R}{R} \left[1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} \right] = 1.875 \frac{V_R}{R}$$

The output voltage V_0 is given by.

$$V_0 = -V_R \left[\frac{R_f}{R} A_{N-1} + \frac{R_f}{2^1 R} A_{N-2} + \frac{R_f}{2^2 R} A_{N-3} + \cdots \frac{R_f}{2^{N-1} R} A_0 \right]$$

where A_N = binary digit & can have a value 1 or 0.

Block diagram:



- Input resistor is a parallel in - parallel out device, when convert signal is high, the binary data is clocked into the resistors
- The voltage switches are connect or disconnect V_R (Reference voltage or ground to the resistor network.
- The function of resistive summing network is to weight & sum the binary currents.
- The op-amp performs two functions i) convert input current to voltage ii) To scale the voltage.

Dig- advantages:

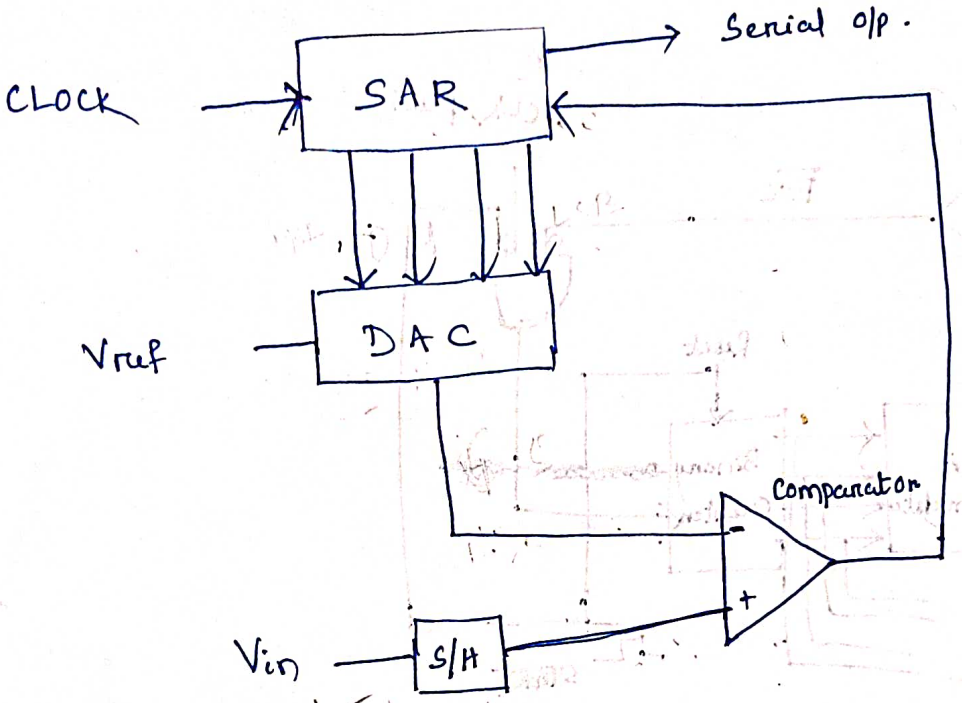
- It requires too many values of resistors.

Ex: 8 bit converter requires 8 resistors i.e R, 2R, 4R .... 128R.

* To overcome this ladder network is used.

R/2R ladder

Successive Approximation A/D convention:

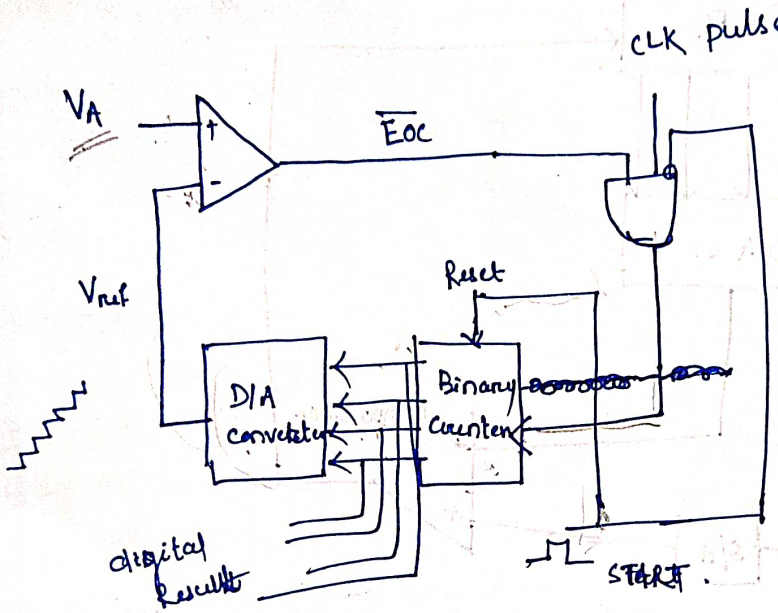


SAR - Successive Approximation Register

- The SAR is initialised so that the MSB is equal to a digital 1.
- This code is fed into the DAC, which then supplies the analog equivalent of this digital code into comparator for comparison with the sampled input voltage.
- If this analog voltage exceeds Vin the comparator causes the SAR to reset bit, otherwise, the bit is left a 1.
- Then the next bit is set 1 and the same procedure is done until every bit in SAR has been tested.
- The resulting code is outputed at SAR.

ADC

Counter Method / Staircase - Ramp A/D Converter :/ Digital Ramp :



- A START pulse is applied to reset the counter to zero. The High at START also inhibits clock pulse from passing through the AND gate into the counter
- With all 0's at its input, the DAC's output will be $V_{ref} = 0V$.
- Since $V_A > V_{ref}$, the comparator output $\overline{EOC}$, will be high.
- When START returns low, the AND Gate enables & the CLK pulses get through to the counter.
- As the counter advances, the DAC output, V_{ref} increases one step at a time.
- This continues until V_{ref} reaches a step that exceeds V_A.
- At this point, $\overline{EOC}$ will go Low & inhibit the flow of pulses into the counter & the counter stop counting.
- The conversion process is now complete & the content of counter are the digital representation of V_A.
- The counter will hold the digital value until the next START pulse initiates a new conversion.

Transistor Transistor Logic (TTL) :

→ TTL logic family implemented with bipolar process technology.
→ The NAND gate is the basic building block of this logic family.
→
→ TTL gate in all versions come in 3 different types of output configuration.

i) Open collector output configuration
ii) Totem pole Output configuration
iii) Tristate or three state Output configuration

i) TTL with Open collector output configuration :

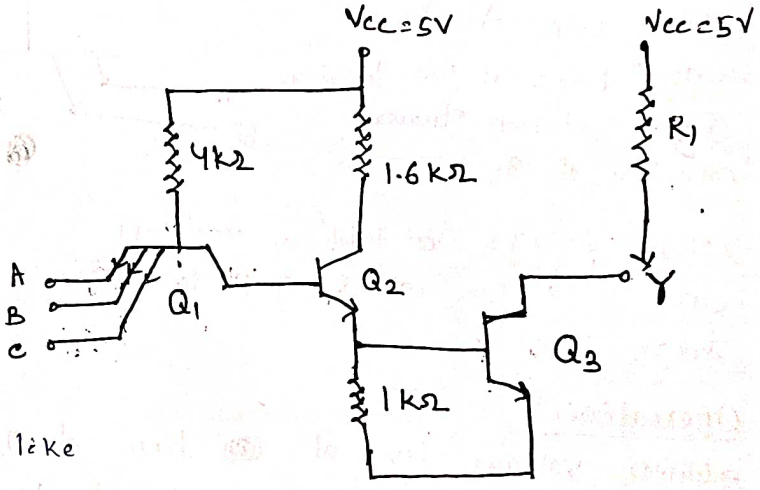
Construction :

→ The basic TTL NAND gate is shown in figure.

→ Q_1 is a multiple emitters transistor & logic inputs are applied to the emitter of Q_1.

→ These emitters are behave like a input diode.

→ The Output of TTL gate is taken from the open collector of Q_3.



Operation :

→ When all the inputs are high, no emitter base junction of Q_1 is forward bias & voltage at the base of Q_1 is higher than 2.1 V.
→ Hence transistor Q_2 is driven saturation as well as Q_3, provided it has current through the collector.
→ The collector current may be available from the external pull up resistance or from the connected load at output.
→ The output Y is VCE (saturation) i.e 0.2 V.
→ Thus gate operation conforms the NAND operation as when any input is low output is high & when all inputs are high, out is low.

Application:

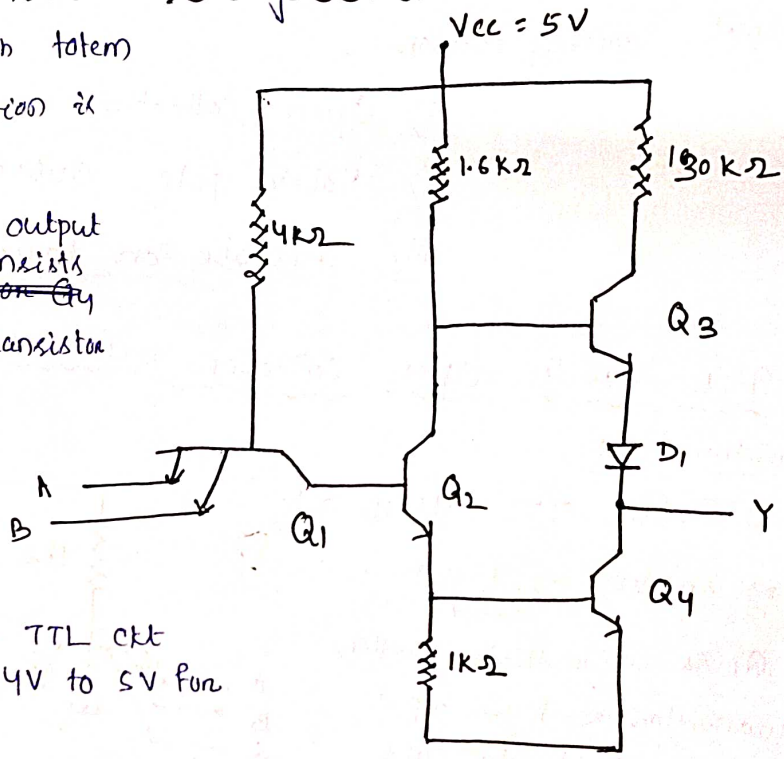
→ TTL integrated ckt were widely used in applications such as computer, industrial control, test equipments & instrumentation, consumer electronics etc.

ii) TTL with Totem Pole Configuration:

→ The TTL NAND gate with totem pole output configuration is shown.

→ It is called totem pole output configuration, as it consists of pull up & pull down transistor along with diode.

→ The base of the transistor Q_3 is driven from collector of Q_2.



→ Two voltage levels of TTL ckt are 0.2V for low & 2.4V to 5V for high level.

Operation:

→ When voltage level at base of Q_1 is low, then the o/p voltage of Q_1 is not sufficient to drive Q_2 & Q_4. They are cut off condition.

→ When Q_2 & Q_4 are off, high base current available for Q_3 to operate & the output Y is high.

→ When all the inputs are high, no base emitter junction is forward bias & voltage at the base of Q_1 is higher than 2.1V.

→ Hence transistor Q_2 & Q_4 are driven saturation & Q_3 is in cutoff condition.

→ Thus the gate operation confirms NAND function, as when any of the inputs is low, the output Y is high & if all inputs are high, output is low.

Complementary Metal Oxide Semiconductor (CMOS):

→ In CMOS ckt both n-channel & P-channel device can be fabricated on the same substrate.

→ In CMOS ckt both types of MOS devices are interconnected to form a logic function.

→ The power consumption extreamly low, enhance noise immunity, high fan out capability & simpler interfacing with other ckt.

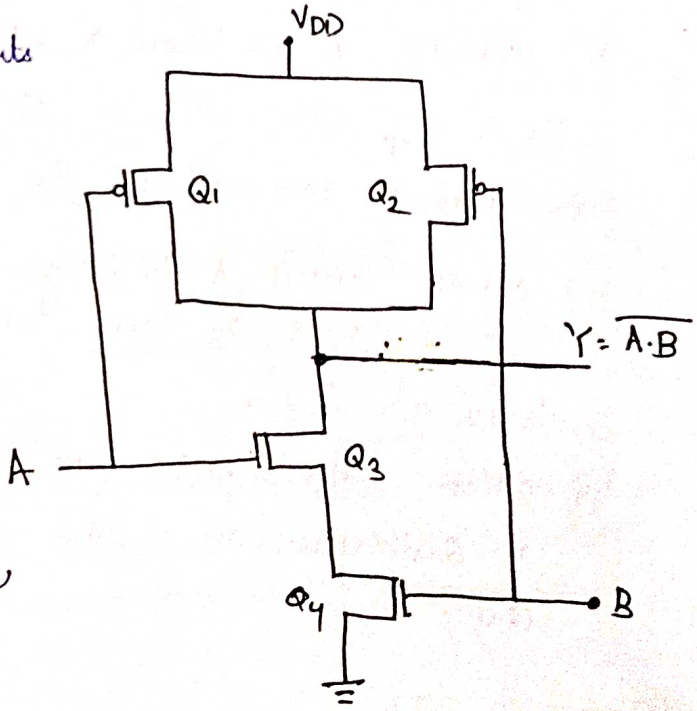
i) CMOS NAND Gate:

→ CMOS NAND gate with two inputs & arrangement of complementary pairs (PMOS & NMOS).

→ The operations of CMOS NAND gate are:

i) When both inputs are low, Q_1 & Q_2 are ON & Q_3 & Q_4 are off. So the o/p is high.

ii) When input A is low & input B is high, then Q_1 & Q_4 are ON, Q_2 & Q_3 are off. The o/p is high by Q_1.



iii) When input A is high & input B is low, then Q_2 & Q_3 are ON, Q_1 & Q_4 are off. The o/p is high by Q_2.

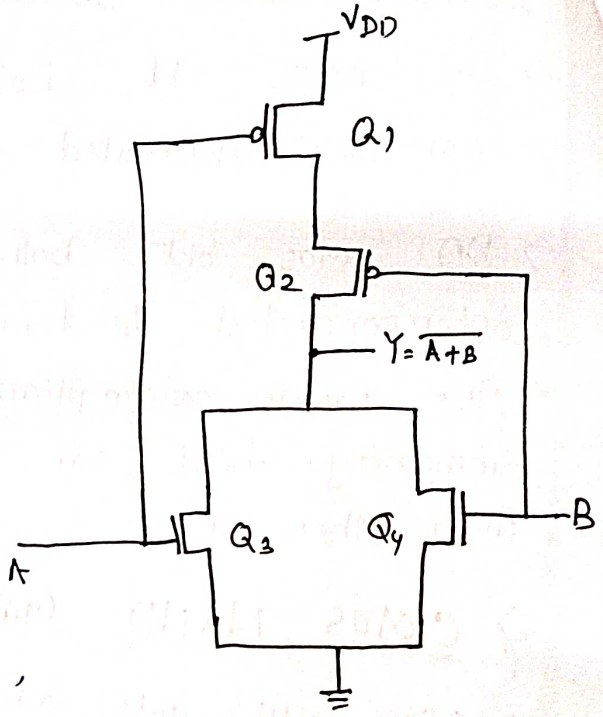
iv) When both inputs are high, Q_1 & Q_2 off, Q_3 & Q_4 are ON. The output is pulled Low by Q_3 & Q_4.

Truth Table:

Input		MOSFET				Output
A	B	Q_1	Q_2	Q_3	Q_4	Y
L	L	ON	ON	OFF	OFF	H
L	H	ON	OFF	OFF	ON	H
H	L	OFF	ON	ON	OFF	H
H	H	OFF	OFF	ON	ON	L

ii) CMOS NOR Gate:

→ Fig. shows a CMOS NOR gate with two inputs A & B & one output Y.



→ The operation of NOR gate is as follows:

i) When both inputs are Low, Q_1 & Q_2 are ON & Q_3 & Q_4 are off. The o/p is pulled high by Q_1 & Q_2.

ii) When A is Low & B is high, Q_1 & Q_4 are ON, Q_2 & Q_3 are off. The output is pulled low by Q_4.

iii) When input A is high & input B is low, Q_1 & Q_4 are off & Q_2 & Q_3 are ON. The output is pulled Low through Q_3.

iv) When both inputs are High, Q_1 & Q_2 are off & Q_3 & Q_4 are ON. The output is pulled low by Q_3 & Q_4.

Truth table:

Input		MOSFET				Output
A	B	Q_1	Q_2	Q_3	Q_4	Y
0	0	ON	ON	OFF	OFF	1
0	1	ON	OFF	OFF	ON	0
1	0	OFF	ON	ON	OFF	0
1	1	OFF	OFF	ON	ON	0

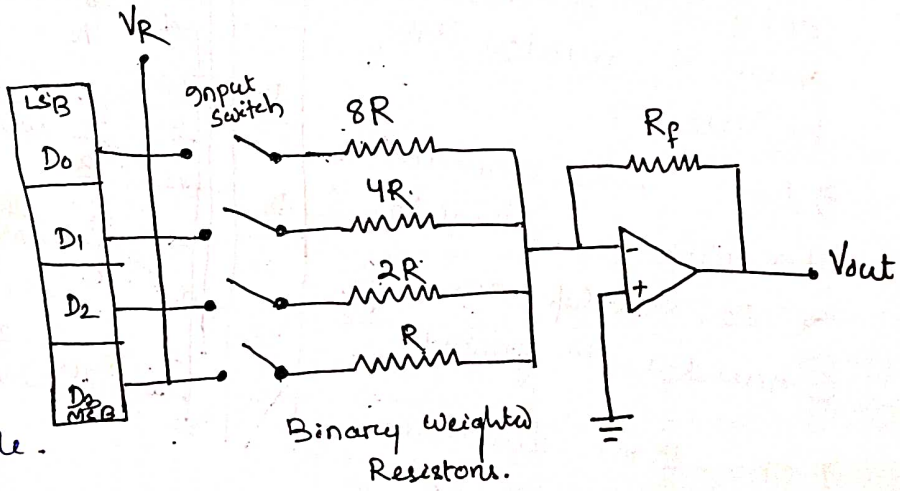
Digital to Analog Converter:

→ A digital to analog conversion (D/A) is an important interface process for I/O operation.

→ The coversion of digital signal to analog signal can be achieved by
  i) Weighted - Resistor method
  ii) R-2R ladder network.

i) Weighted Resistor digital to Analog converter:

→ A basic type of resistor network D/A converter is shown in the fig.



→ It is a 4bit DAC uses 4 inputs.

→ The resistances used has the values that represent the binary weights of the input bits of the digital code.

→ The switch symbols represent transistor switch for inputting each of 4 bits.

→ Rf is the feedback resistance.

→ The Op-amp offers a very high i/p impedance. As a result current through the feedback resistance is sum of input currents & output is proportional to current through Rf.

* The high binary weighted input or the MSB input (2^3) corresponds to the lowest value of resistor

→ VR is the reference voltage.

$$I = \frac{V_R}{R}$$ as no current passes through 2R, 4R & 8R

of digital input is 1111 then

$$I = V_R\left(\frac{1}{R} + \frac{1}{2R} + \frac{1}{4R} + \frac{1}{8R}\right)$$

$$= \frac{V_R}{R}\left[1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8}\right)$$

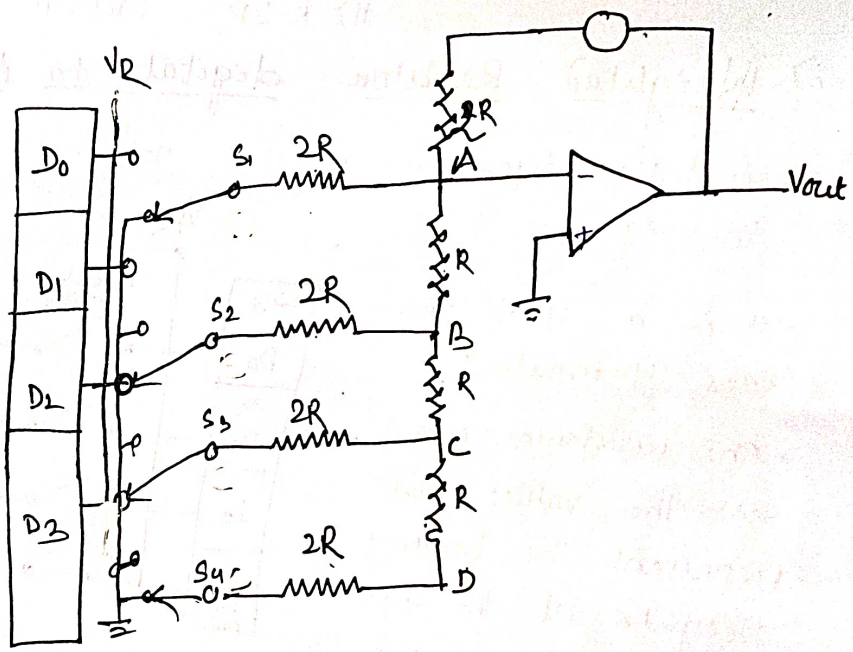
$$= \frac{V_R}{R}(1.875)$$

$$V_{out} = \frac{R_f}{2^{N-1} R} \left[2^{N-1} V_{N-1} + 2^{N-2} \cdot V_{N-1} + \ldots 2^1 V_1 + 2^0 V_0 \right]$$

ii) Ladder type DAC / R-2R ladder.

→ The advantages of this type of network is that it requires only two different values of resistor to be used.

→ Let S_1 is connected to reference voltage V_R & S_2, S_3, S_4 are connected to ground.

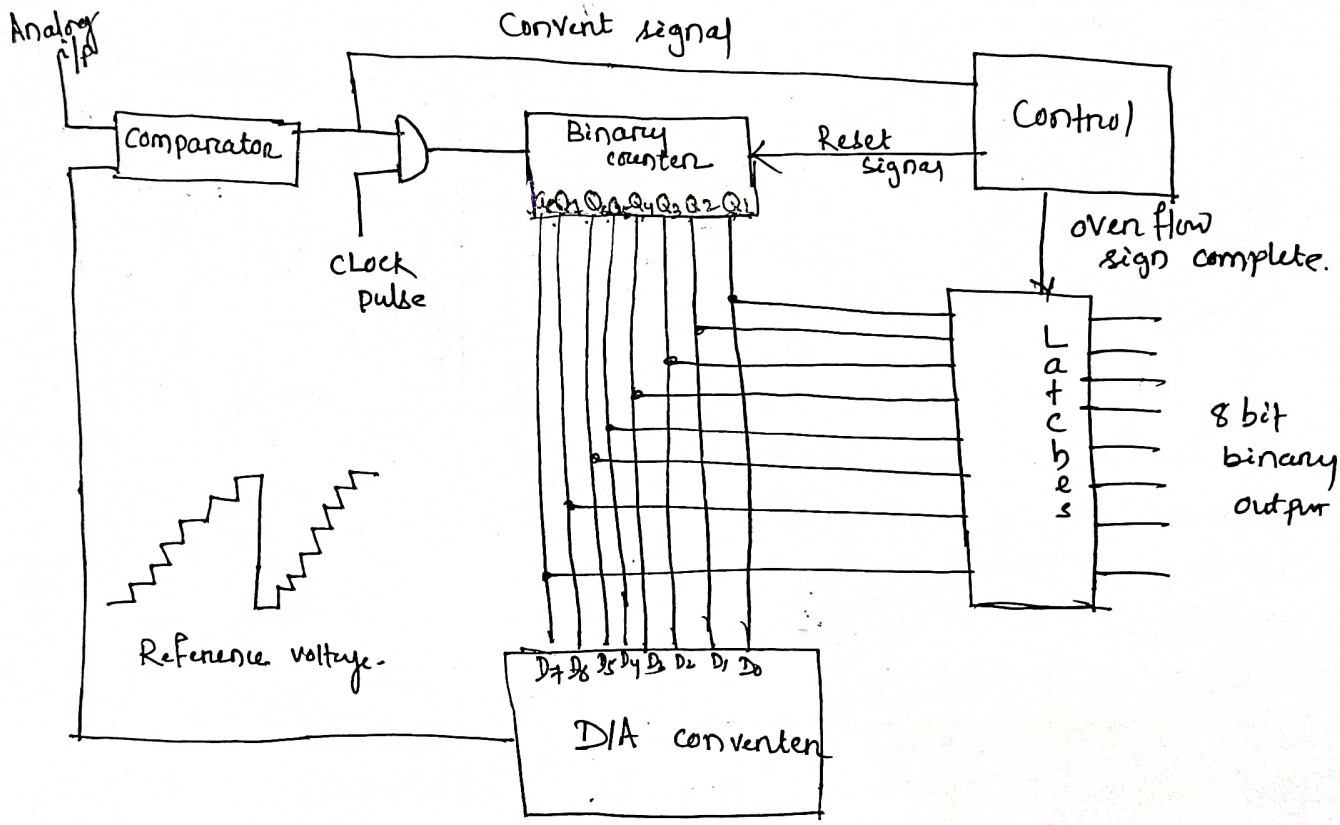


→ the total resistance at point A is 3R & total current is $I = \frac{V_R}{3R}$

Analog to Digital Conversion:

→ The process of converting analog voltage into equivalent digital signal is called analog to digital conversion & the device which perform this is called ADC.

i) Counter Method or Staircase - Ramp A/D converter:



→ The analog input voltage to be converted into digital is fed to the comparator & a CONVERT signal is used.

→ The control resets the counter to an all zero state, & then supplies clock pulse to the counter.

→ The binary output from the counter is fed to DAC which converts this input to analog voltage & output the digital voltage corresponding to ~~this~~ analog input.

→ This analog is fed to the comparator which compares the DAC output with analog input.

→ The moment DAC o/p exceeds the analog input voltage it is made on or its output becomes a high, under

a constant Vol

this situation comparator sends a signal to the control ckt & it stops the counter.

→ The binary number stored in the counter represents the digital voltage corresponding to the input analog voltage.

→ The binary number corresponding to the input analog voltage is loaded into the latches.

Saifan

Converters

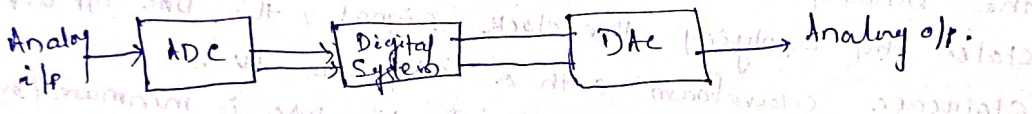
Introduction:

Real world processes produce analog signals that carry information.

- An analog signal is defined over a continuous period of time & its amplitude may assume a continuous range of values.
- It's difficult to store, compare, calculate & manipulate this information with good accuracy using purely analog technology.

* Digital signal processing technology has a number of advantages over analog signal processing.

- As real data available in analog form so →

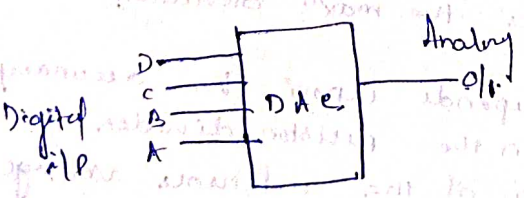


Digital to Analog Converter (DAC):

→ The digital to analog conversion involves translating digital information into its equivalent analog information.

→ OR It is the process of converting a value represented in digital code such as binary or BCD into a voltage or current which is proportional to the digital value.

Ex:



(4 bit DAC)

- Each of the digital inputs A, B, C & D can assume a value 0 or 1, therefore $2^4 = 16$ possible combinations of inputs.

→ Each i/p number 0000, 0001, 0010 ... 1111, the DAC outputs a unique value of voltage.

- The analog output voltage Vout is proportional to the input binary number.

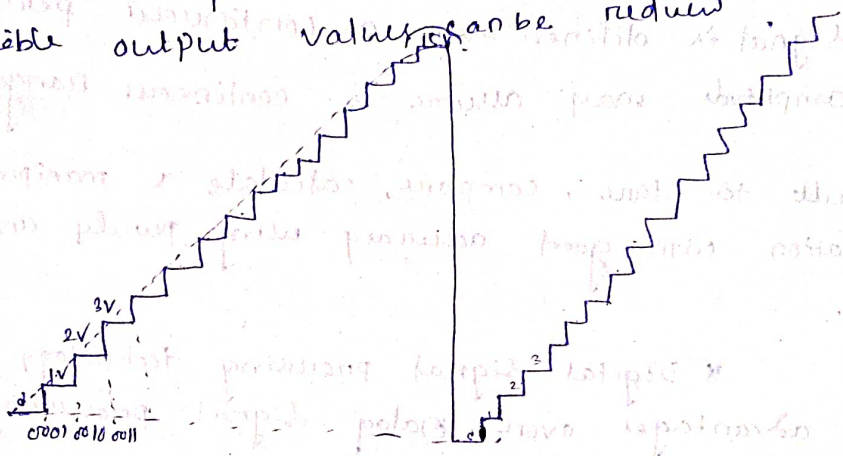
$$\text{Analog output} = K \times \text{digital input}$$

where K is proportionality factor & is a constant value for a given DAC.

→ Actually the o/p of DAC is not a true analog quantity, because it can take on only specific values. In that sense, it is actually digital.

→ Thus the output of a DAC is a pseudo analog quantity.

→ By increasing the number of input bits, the number of possible output values can be reduced.



→ When the binary counter is continuously recycled through its 16 states by applying the clock signal, the DAC o/p will be a staircase waveform with a step size 1V.

→ When the counter is 0000, the o/p of the DAC is minimum (0V) & when the counter is 1111, the o/p of DAC is max^m (15V). This is the full scale output.

Parameters of DAC:

Accuracy:

* - It is a measure of how close the actual voltage is to the theoritical output value.
  - Absolute accuracy defines the max^m deviation of the output from the ideal value.
  - The accuracy of DAC depends upon the accuracy of the precision resistor used in the resistor divider or ladder network & the precision of the reference voltage.
  - It is specified as a percentage of full-scale or max^m o/p voltage.
* - Accuracy specifies the max^m error that can occur in a output voltage.

Ex: Theoritical o/p voltage of 10V for a full scale digital i/p & accuracy is ±5%. It means that the output voltage of DAC for the same digital i/p lies between 9.5V to 10.5V.

Resolution:

* It defines the smallest possible change in the output analog voltage due to the change in digital input
→ The resolution is always equal to the weight of LSB
& it is also known as step size.
– It is a function of the number of bits in digital input

* On a 4 bit DAC using a ladder, if the full scale voltage 16 V, then the resolution is $16/2^4 = 1V$.
Thus the output voltage changes in a step of 1V.

$$\text{Percentage Resolution} = \frac{\text{step size}}{\text{full scale}} \times 100.$$

Since, full scale = number steps × step size, resolution can be expressed as

$$\text{Percentage resolution} = \frac{1}{\text{total no of steps}} \times 100$$

To get a good resolution, the no of input bits (n) of DAC should maximum.

Time:

Analog to Digital Conversion:

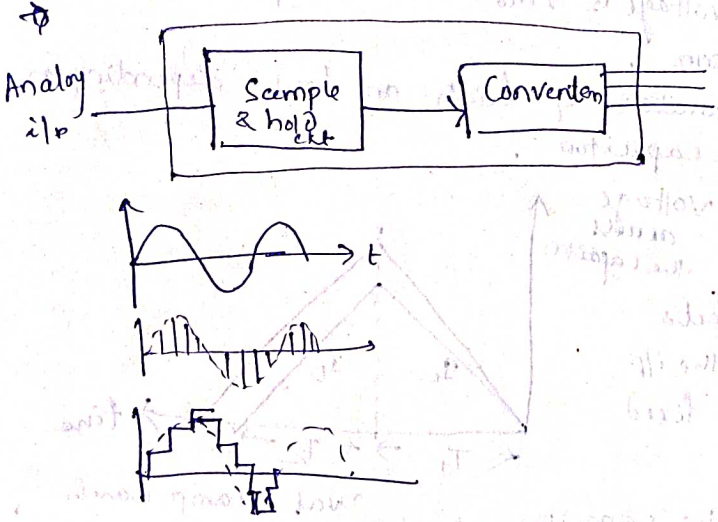
→ An ADC does the inverse function of DAC.

→ In ADC, the input analog voltage can have any value in a range & the digital o/p can have only 2^N discrete value of an N bit ADC.

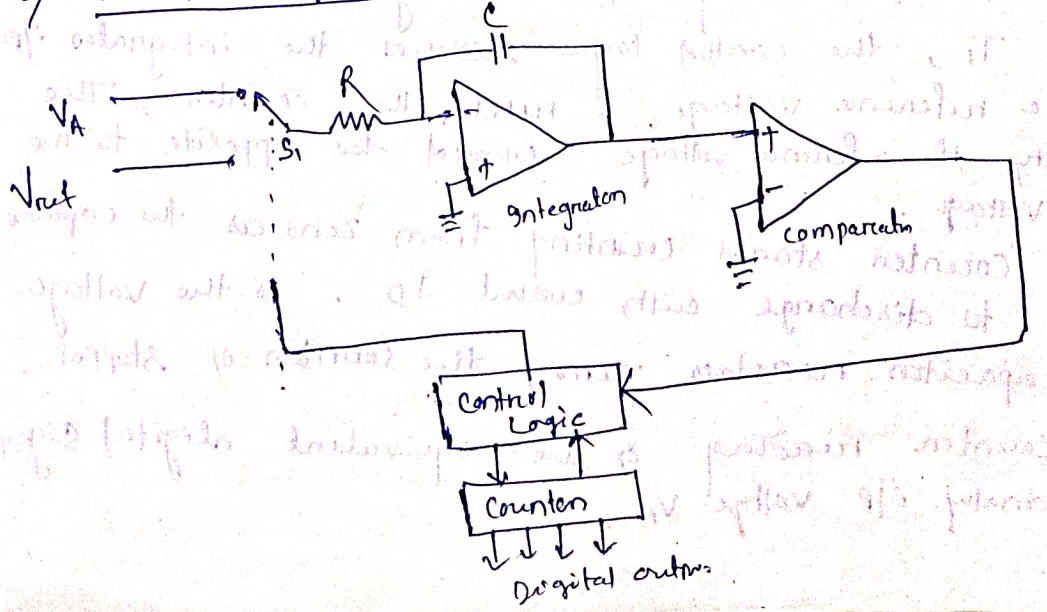
→ In DAC, the possible number of digital input is fixed, Ex: in 4 bit DAC, there are 16 possible i/p's.

① → The process of converting analog voltage into an equivalent digital signal is known as analog to digital conversion & the device which performs this is called analog to digital converter.

→ The ADC process includes sampling of input analog signal & then, each sample is converted into its binary equivalent.



i) Dual Slope ADC:



→ In dual slope ADC, the integrator generates two different ramps: one with an unknown analog input voltage V_A as input, & another with a known reference voltage V_{ref} as the input. Hence it is known as dual slope ADC.

→ The basic principle of a dual slope ADC is that, an analog signal is converted into a proportional time period, which is then measured using digital counter.

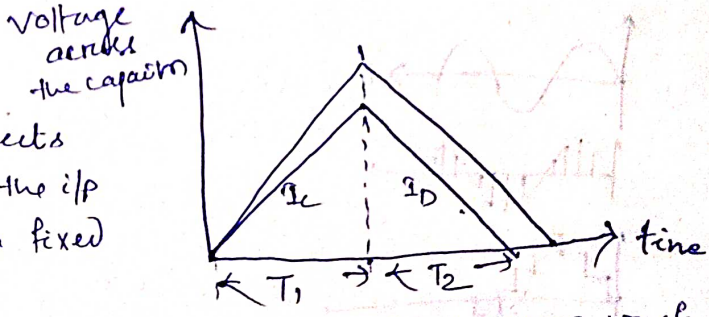
→ A dual slope ADC includes an integrator, comparator, counter, control logic & a reference voltage.

Operation:

→ When V_A (unknown analog input voltage) is connected to integrator, the capacitor is charged linearly. The voltage across the capacitor depends on the amplitude of V_A & duration T.

→ The comparator compares the output of integrator with reference input. The reference voltage is zero & the comparator acts as a zero crossing detector.

→ The output of the comparator is high or low depending on the voltage across the capacitor.



Dual ramp waveform

start → The switch S_1 connects the analog voltage V_A as the i/p to the integrator for a fixed period T_1.

→ During this period, the capacitor of integrator is charged up to a certain voltage with charging current I_C, which is the function of analog input voltage.

→ After T_1, the control logic switches the integrator i/p to the reference voltage & resets the counter. The polarity of reference voltage would be opposite to the input voltage.

→ The counter starts counting from zero as the capacitor starts to discharge with current I_D. As the voltage of capacitor reaches zero, the counter is stopped.

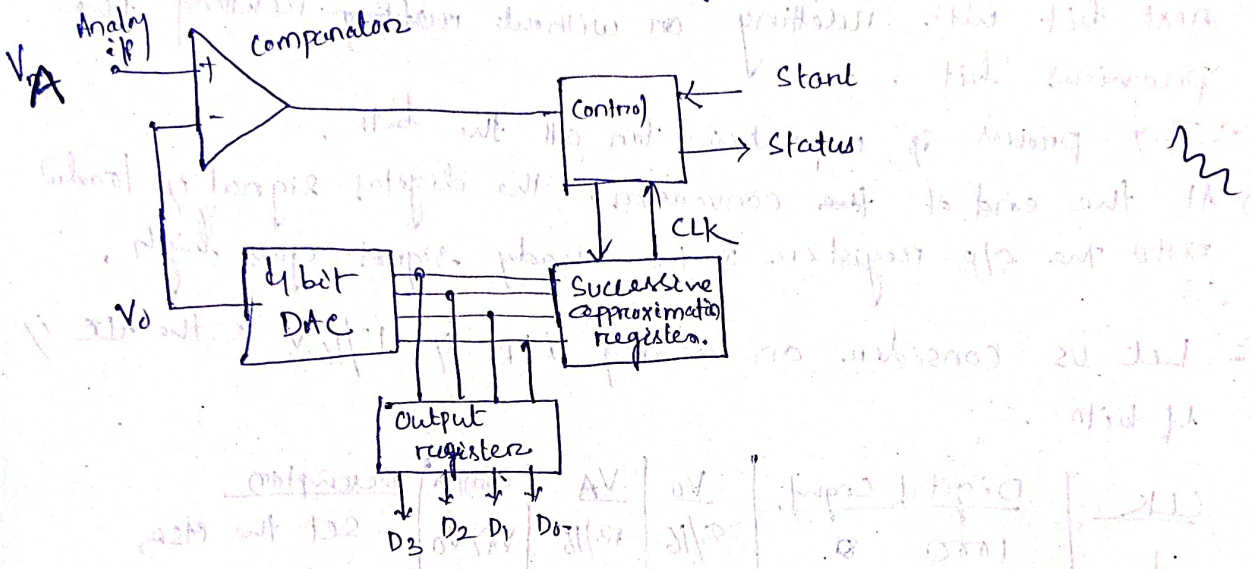
→ The counter reading is the equivalent digital signal for analog i/p voltage V_A.

Successive - approximation ADC :

→ It is a high - resolution ADC that uses one comparator with variable reference voltage.

→ The variable reference voltage can be obtained by a sequence on binary counter & DAC.

→ The basic principle of this ADC is that the unknown analog i/p voltage is approximate against an n bit digital value by trying one bit at time, beginning with the MSB.



→ It consists of a DAC, a comparator, a successive app. register & control ckt & an o/p register.

→ The DAC converts the digital data into analog signal, which is given as one of the i/p to the comparator.

→ The comparator compares the analog input that has to be converted into digital with analog o/p.

→ The result of the comparator may be $+V_{sat}$ or $-V_{sat}$ (i.e 1 or 0) which controls the control ckt.

→ The control ckt changes the data of successive-approx register as per the o/p signal of the comparator.

* When the start signal is provided by a user, the MSB of successive - approx register is set (1) & other bits are reset (0) with clock pulse.

→ The DAC converts this digital data into analog voltage V_0, the comparator compares this analog voltage with the analog input voltage V_A.

→ When $V_A > V_0$, the o/p of comparator is high, the control ckt sets the 2nd MSB without resetting the MSB.

→ When $V_A < V_0$, the o/p of comparator is low, the control ckt sets the 2nd MSB with resetting the MSB

→ Again the DAC converts this digital data into analog voltage V_0, the comparator compares this analog voltage with the analog i/p voltage V_A.

→ As per the results of comparator, the control ckt sets the next bit with resetting or without ~~resetting~~ resetting the previous bit.

→ This process is repeated for all the bits.

→ At the end of ~~the~~ conversion, the digital signal is loaded into the o/p register & the ready signal goes high.

Ex:- Let us consider an analog i/p of $13/16$ V & the ADC of 4 bits.

CLK	Digital signal	V_0	V_A	compn	Description
1	1000	8/16	13/16	$V_A > V_0$	Set the MSB
2	1100	12/16	13/16	$V_A > V_0$	Set the 2nd MSB without resetting the MSB
3	1110	14/16	13/16	$V_A < V_0$	Set the next bit without resetting the previous bit
4	1101	13/16	13/16	$V_A = V_0$	Set the next bit with resetting the previous bit

—o—

Parameters of ADC:

i) Accuracy: It measures how closed the actual output voltage is to the theoritical output voltage. It depends on the quantization level (size of ADC). The accuracy of higher bit is more as compared to lower bits.

ii) Resolution: It is defined as the voltage i/p changes necessary for 1 bit change in output. Resolution in terms of voltage is the full-scale i/p voltage divided by the total number of levels. The resolution of an n bit ADC is given as

$$\text{Resolution} = \frac{V}{(2^n - 1)}$$

iii) Conversion Time: It is the time required to convert the analog data into its digital equivalent. Conversion time is in milliseconds. It should be as minimum as possible.

(2) $x + \bar{x}y = x + y$

L.H.S : $x + \bar{x}y$

$= x(1+y) + \bar{x}y$

$= x \cdot 1 + xy + \bar{x}y$

$= x + xy + \bar{x}y$

$= x + xy + \bar{x}y$

$= x + y(x + \bar{x})$

$= x + y(1)$

$= x + y$ R.H.S

(3) $x + xy = x$

L.H.S : $x + xy$

$= x \cdot 1 + x \cdot y$

$= x(1+y)$

$= x \cdot 1 = x$

(4) $x(x+y) = x$

L.H.S : $x(x+y)$

$= x \cdot x + xy$

$= x + xy$

$= x(1+y)$

$= x$

(5) $x(\bar{x}+y) = xy$

L.H.S : $x(\bar{x}+y)$

$= x\bar{x} + xy$

$= 0 + xy = xy$ R.H.S

(6) $x \cdot y + \bar{x}z + y \cdot z = xy + \bar{x}z$

L.H.S : $xy + \bar{x}z + yz$

$= xy + \bar{x}z + yz(x + \bar{x})$

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

$= xy(1+z) + \bar{x}z(1+y)$

$= xy + \bar{x}z$

(7) $(x+y)(\bar{x}+z)(y+z) = (x+y)(\bar{x}+z)$

L.H.S : $(x+y)(\bar{x}+z)(y+z)$

$= (x \cdot \bar{x} + xz + \bar{x}y + yz)(y+z)$

$= (xz + \bar{x}y + yz)(y+z)$

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

$= xyz(x+\bar{x}) + \bar{x} + z + x + y$

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

$= y(z+1) + 1 + z$

$= 1 + y + z$

Examples →

(1) $x \cdot y \cdot y \cdot z \cdot z \cdot z$

$= x(y \cdot y) \cdot z(z \cdot z)$

$= x \cdot y \cdot z \cdot z$

$= xyz$

(2) $x + x + y + y + y$

$= (x+x) + y + (y+y)$

$= x + y + y$

$= x + (y+y) = x + y$

(3) $x \cdot x \cdot y + x \cdot y \cdot y + y \cdot \bar{y} \cdot x \cdot x$

$= (x \cdot x)y + x(y \cdot y) + (y \cdot \bar{y})(x \cdot x)$

$= xy + xy + 0 \cdot x$

$= xy$

(4) $x \cdot x \cdot \bar{y} + \bar{x}y \cdot z \cdot z$

$= (x \cdot x)(y \cdot \bar{y}) + \bar{x}y(z \cdot z)$

$= xy + \bar{x}yz$

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

$= y(x+z)$

: Boolean Algebra :-

Rules & Laws of Boolean Algebra →

1. $X \cdot \bar{X} = 0$
2. $X + \bar{X} = 1$
3. $X \cdot 0 = 0$
4. $X \cdot 1 = X$
5. $X \cdot X = X$
6. $X \cdot \bar{X} = 0$
7. $X + 0 = X$
8. $X + 1 = 1$
9. $X + X = X$
10. $X + \bar{X} = 1$
11. $X(Y+Z) = XY + XZ$
12. $X + (YZ) = (X+Y)(X+Z)$
13. $X + (\bar{X} \cdot Y) = X + Y$
14. $X + XY = X$
15. $X(X+Y) = X$
16. $X(\bar{X}+Y) = X \cdot Y$
17. $XY + \bar{X}Z + YZ = XY + \bar{X}Z$
18. $(X+Y)(\bar{X}+Z)(Y+Z) = (X+Y)(\bar{X}+Z)$
19. $\bar{X}Y + \bar{X}Z = (X+Z)(\bar{X}+Y)$
20. $(X+Y)(\bar{X}+Z) = XZ + \bar{X} \cdot Y$
21. $X\bar{Y} + \bar{X}Y = (X+Y)(\bar{X}+\bar{Y})$
22. $XY + \bar{X}\bar{Y} = (X+\bar{Y})(\bar{X}+Y)$

DeMorgan's Theorem →

23. $\overline{X+Y} = \bar{X} \cdot \bar{Y}$
24. $\overline{X \cdot Y} = \bar{X} + \bar{Y}$

Verification →

$X + (YZ) = (X+Y)(X+Z)$

$= X + Y \cdot Z$

$= X \cdot 1 + Y \cdot Z$

$= X(Y+1) + Y \cdot Z \quad (\because Y+1 = 1)$

$= XY + X + YZ$

$= XY + X(1+Z) + YZ \quad (Z+1 = 1)$

$= XY + X + XZ + YZ$

$= XY + XX + XZ + YZ \quad (\because X \cdot X = X)$

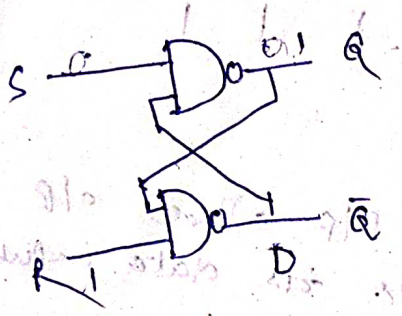
$= X(X+Z) + XY + YZ$

$= X(X+Z) + Y(X+Z)$

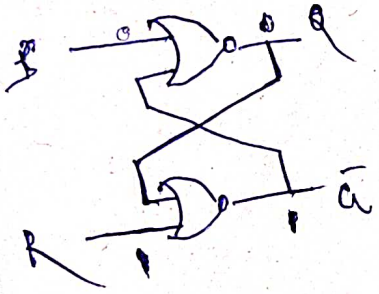
$= (X+Z)(X+Y)$ Proved

$=$

Latches: Two cross coupled NAND or NOR Gate is known as Latches.
It is a bistable device.



S	R	Q	$\bar{Q}$	
0	0	-	-	Invalid / Indeterm. / Forbidden
0	1	1	0	Reset
1	0	1	0	Set
1	1	Q_0	$\bar{Q}_0$	No change.



S	R	Q	$\bar{Q}$	
0	0	Q_0	$\bar{Q}_0$	N.C
0	1	1	0	Set
1	0	0	1	Rest
1	1	-	-	Indeterminate.

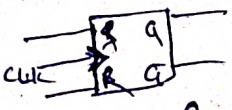
F/F

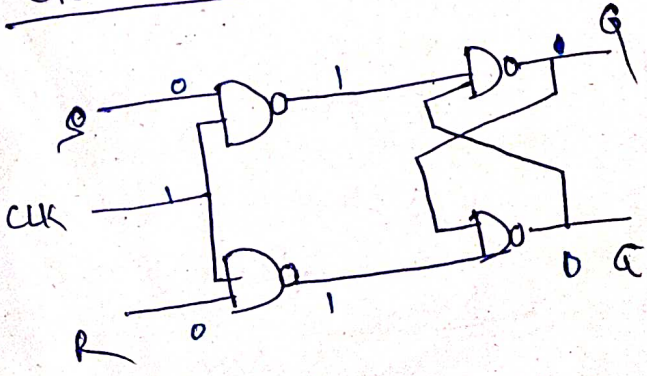
⇒ F/F are synchronous bistable devices.

- The term synchronous means that the output changes state only at a specific point on a triggering input called clock i.e changes in the output occurs in synchronization with the clock.

- There are 4 types of F/F...
  i) SR ii) JK iii) D iv) T

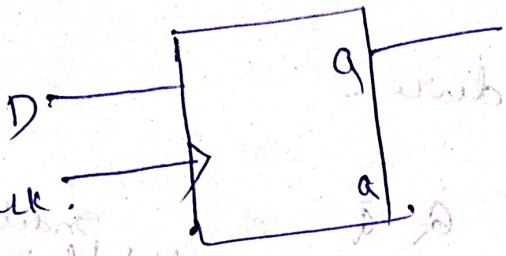
Clocked SR F/F:



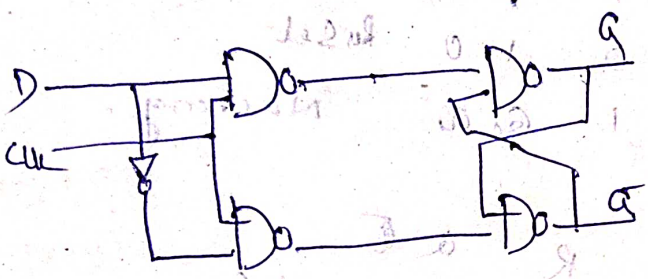


CLK	S	R	Q	$\bar{Q}$	Comm
0	X	X	Q_0	$\bar{Q}_0$	N.C
↑	0	0	Q_0	$\bar{Q}_0$	NC
↑	0	1	0	1	Reset
↑	1	0	1	0	Set
↑	1	1	-	-	Invalid.

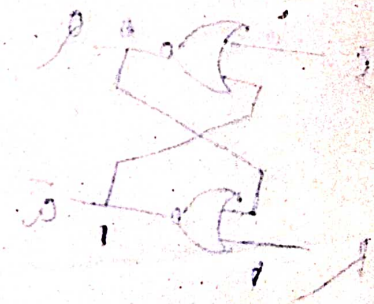
D F/F



CLK	D	Q_n	
0	X	Q_n	N.C
1	0	0	
1	1	1	



A F/F whose o/p follows its data input



JK FIC

Sailon

4th Electrical — Digital Electronics

~~1. The radix of a octal number~~

1. The octal number system has a radix of
   a) 16 b) 8 c) 2 d) 10

2. What is equivalent of -35_{10} in 8 bit 2's complement ?
   a) 01011101 b) 11011111 c) 11011101 d) 00101101

3. Decimal equivalent of $(567)_8$ is $5\times8^2+6\times8^1+7\times8^0 = 5\times64+48+7 =$
   a) 567 b) 487 c) 375 d) 501

4) How many parity bits are required for transmitting a data of 4 bit.
   a) 1 b) 2 c) 3, d) 4

5) What is the gray code for decimal number 12 ?
   a) 1010 b) 1100 c) 1110 d) 0111

6) Which of the following is non-weighted code.
   a) 8421 BCD b) XS-3 c) 5421 BCD d) BCD 4221

7) Under what conditions the output of a two input AND gate is one ?
   a) Both the inputs are 0 b) Any one input is 0 c) Both the inputs are 1 d) none of these

8) The unique output of a NAND logic gate is 0
   a) when all inputs are 0
   b) when all inputs are 1
   c) when any one input is 0
   d) when any one input is 1

9) The minterm designator for the term $AB\bar{C}\bar{D}E$ is
   a) 9 b) 11 c) 13 d) 25

10) The Boolean expression $F = A\bar{B}CD + A\bar{B}C\bar{D} + \bar{A}BCD$ can be written as
   a) $F = \Sigma m(7,8,9)$
   b) $F = \Sigma m(7,9,10)$
   c) $F = \Sigma m(7,8,10)$
   d) $F = \Sigma m(6,9,10)$

11) Which gates are known as universal logic gates ?
   a) AND, OR b) EX-OR - Ex-NOR c) NAND, NOR d) NOT - AND

12) Solve $x + (\bar{x}y)$

a) $x+y$ b) xy c) $2xy$ d) none of these

13) Which one of the De-morgan's theorem ?

a) $X \cdot \bar{X} = 0$ b) $X + \bar{X} = 1$ c) $x(\bar{x} \cdot y) = x + y$

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

14) For a function $\Sigma m(2, 3, 6, 7, 10, 11, 14, 15)$ of (A, B,
the minimal expression obtained using K map will be

a) A b) B c) C d) $\bar{A} + B + \bar{C}$

15) Using Boolean algebra the reduced expression for functi
$A\bar{B}C + ABC$ can be realized by using how many number of gates ?

a) ~~7~~ b) 3 c) 2 d) 1