Number systems :-

The different types of number systems are

1) Decimal number system

also known as base 10 number system. base represents the number of different symbols used for representing whole numbers or fractions.

∴ Base 10 → 10 symbols,

0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

2) Binary number system :- Base 2

∴ 2 symbols — 0 and 1

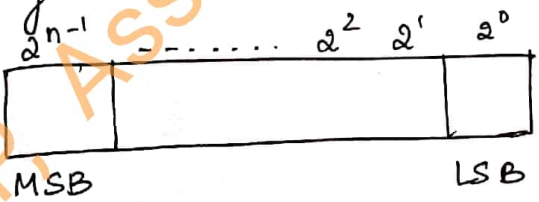
The two symbols 0 and 1 are known as bits

A binary number with 4 bits — NIBBLE

A binary number with 8 bits — BYTE

A binary number with 16 bits — WORD

The binary number can be represented as



where MSB → most significant bit

LSB → least significant bit

Eg: 1) $(1011)_2$ → its decimal equivalent is obtained as follows.

$\rightarrow 1 \times 2^3 + 0 \times 2^2 + 1 \times 2 + 1 \times 1$

$= 8 + 0 + 2 + 1 = 11_{10}$.

2) $10101.011 \rightarrow 1\times2^4 + 0\times2^3 + 1\times2^2 + 0\times2 + 1\times1 \cdot \frac{0}{2} + \frac{1}{4} + \frac{1}{8}$

$= 16 + 0 + 4 + 0 + 1 \cdot 0.5 + 0.25 + 0.125$

$= (21.375)_{10}$.

3) Octal Number system :- Base 8

hence there are 8 symbols.

0, 1, 2, 3, 4, 5, 6, 7

used extensively by early minicomputers.

4) Hexadecimal number system :- Base 16.

It is commonly used in computers.

It has 16 possible symbols.

0 - 1 - 2 - 3 - 4 - 5 - 6 - 7 - 8 - 9 - A - B - C - D - E - F

letters from A to F are used to represent digits larger than 9.

Table showing relationship b/w different number system

Decimal	Hexadecimal	Octal	Binary
0	0	0	0000
1	1	1	0001
2	2	2	0010
3	3	3	0011
4	4	4	0100
5	5	5	0101
6	6	6	0110
7	7	7	0111
8	8	10	1000
9	9	11	1001
10	A	12	1010
11	B	13	1011
12	C	14	1100
13	D	15	1101
14	E	16	1110
15	F	17	1111

Binary Coded Decimal [BCD]

BCD is a digital representation where each digit of a decimal number is represented by its binary equivalent.

largest decimal digit = 9 → 4 bits are required to code each digit.

BCD is inefficient as only 10 of the 16 possible states of 4 bit numbers are used.

It does not use the numbers 1010, 1011, 1100, 1101, 1110 and 1111

Eg: 123 → BCD equivalent code → 0001 0010 0011

hence 12 bits are needed to code 123 in BCD format.

$(123)_D = (1111011)_B$

hence 8 bits are needed for coding in binary format.

2 | 123
2 | 61 — 1
2 | 30 — 0
2 | 15 — 1
2 | 7 — 1
2 | 3 — 1
1 — 1

One's complement:

Each and every bit in the given binary number is inverted to obtain its one's complement

Eg: One's complement of 1) 1011 is 0100.
2) 1110 is 0001

2's complement: It is a 2 step procedure.

1) each bit in the given binary number is inverted
2) 1 is added to the LSB of the inverted data

Eg: 1) 1011 → 1st complement → 0100
+ 1
0101 → 2's complement

2) 1110 → 1s complement → 0001
+ 1
0010 → 2s complement

Logic gates :-

They are devices that convert binary inputs into binary outputs based on rules of mathematical logic operations.

They are also known as GATES as they control the flow of signals from i/p's to a single o/p.

These are

1) 3 basic gates – AND gate, OR gate, NOT gate
2) 2 universal gates – NAND gate, NOR gate
3) 2 special gates – X-OR gate, X-NOR gate.

1) OR gate / operation

It is symbolically represented as.

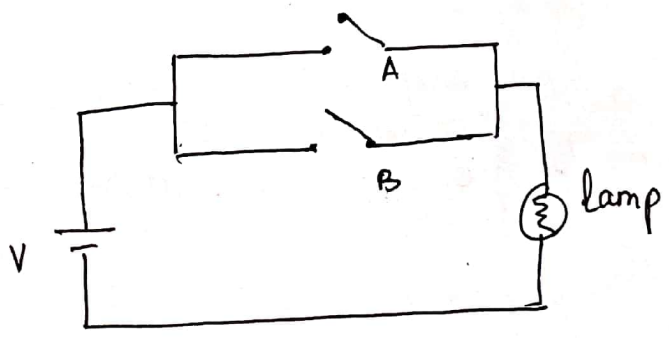
logical expression $y = A + B$ → + stands for OR operation.

Truth table.

A	B	o/p
0	0	0
0	1	1
1	0	1
1	1	1

variables
2 i/p
$\therefore 2^2 \rightarrow$ 4 i/p combinations

Electrical equivalent ckt representing OR operation,



The lamp glows when either of the switch is closed
ie., A = 1, B = 0 → lamp glows
A = 0, B = 1

It glows when both switches are closed.
A = 1, B = 1 → Lamp glows

The lamp remains off when both A and B are open ie., A = 0, B = 0 → Lamp is off

Hence this operation is equivalent to the OR gate operation as described in its truth table.

2) AND gate / operation

symbol of 2 i/p AND gate

logical expression

$y = A \cdot B$

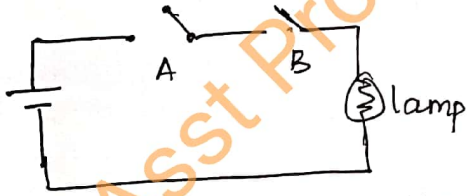
where $\cdot$ represents AND operation.

The truth table of an AND gate is given by

A	B	o/p
0	0	0
0	1	0
1	0	0
1	1	1

The AND gate has a logic high output only when both the i/ps are 1, for all other conditions the o/p of AND gate = 0.

The electrical equivalent circuit of an AND gate



The lamp glows only when both switches A and B are closed ie., when A = B = 1.

for all other conditions the lamp remains off.

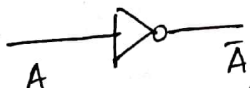
Hence the circuit operation is similar to AND gate operation as described in its truth table.

3) NOT gate / operation

symbol of NOT gate.

logical expression.

$y = \bar{A}$

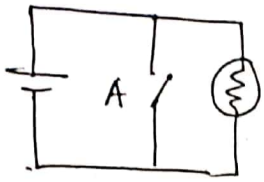


NOT gate is also known as an inverter.

TRUTH TABLE

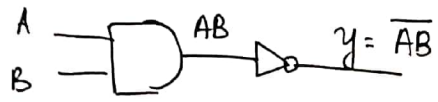
A	y
0	1
1	0

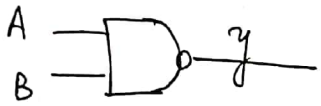
The electrical circuit for NOT operation is given by,



Universal Gates :

1) NAND gate / operation :





logical expression of NAND gate is,

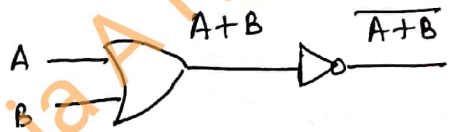
$$y = \overline{AB}$$

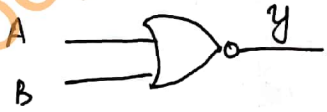
Truth table

A	B	y
0	0	1
0	1	1
1	0	1
1	1	0

output of a NAND gate = 0 only when both or all the inputs are high.
for all other i/p conditions the output of a NAND gate = 1.

2) NOR gate / operation :





The logical expression for a NOR gate is given by

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

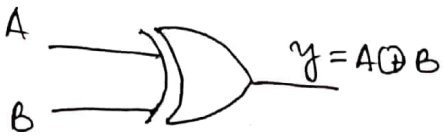
Truth table

A	B	y
0	0	1
0	1	0
1	0	0
1	1	0

The output of NOR gate = 1 when both or all the inputs are zero.
For all other input combinations the output of NOR gate = 0.

Special gates.

1) X-OR gate / operation



symbol of 2 i/p X-OR gate.

logical expression of a X-OR gate is given by

$y = A \oplus B$.

Truth table.

A	B	y
0	0	0
0	1	1
1	0	1
1	1	0

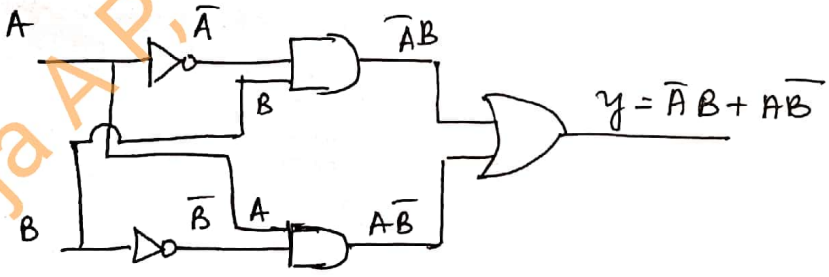
The output of the X-OR gate = 1 when odd number of input bits was high.

The o/p of an X-OR gate is equal to zero when even number of input bits are high or low.

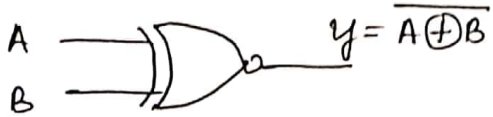
The boolean expression of X-OR gate can be obtained from the truth table.

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

The implementation of X-OR gate using basic gates.



2) X-NOR gate / operation



symbol.

logical expression of X-NOR gate is given by

$$y = \overline{A \oplus B}$$

Truth table.

A	B	y
0	0	1
0	1	0
1	0	0
1	1	1

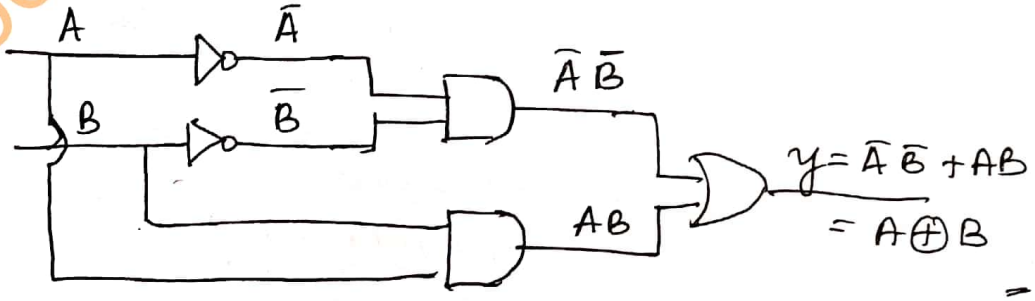
The output of the X-NOR gate = 1 when even number of input bits are either high or low.

The output of the X-NOR gate = 0 when odd number of input bits are high.

The boolean expression for the X-NOR gate can be obtained from the truth table.

$$y = AB + \bar{A}\bar{B} = \overline{A \oplus B}$$

The implementation of X-NOR gate using basic gates -



Boolean Laws:-

1) Complementary law :-

The rules for complementary law are.

a) $\overline{0} = 1$

b) $\overline{1} = 0$

c) $\overline{\overline{A}} = A$

If $\overline{A} = 0$ then $\overline{A} = 1$, then if $A = 1$; $\overline{A} = 0$.

bar over the letter stands for NOT operation

2) AND law :-

The rules of AND law are.

a) $A \cdot 0 = 0$ where $A = 0$ or 1

b) $A \cdot 1 = A$

c) $A \cdot A = A$

d) $A \cdot \overline{A} = 0$

3) OR laws :-

The rules of OR laws are

a) $A + 1 = 1 \longrightarrow$ $A = 0$ then $0 + 1 = 1$
$A = 1$ then $1 + 1 = 1$

b) $A + 0 = A$

c) $A + A = A$

d) $A + \overline{A} = 1 \longrightarrow$ $A = 0$ then $A + \overline{A} = 1 + 0 = 1$
$A = 0$ then $A + \overline{A} = 0 + 1 = 1$

4) Commutative law :-

a) In a two input OR gate, the input signals can be transposed without changing the o/p.

$$A+B = B+A .$$

ie., adding A with B produces same result as adding B with A.

b) In two input AND gate, the input signals can be transposed without changing input.

$$AB = BA .$$

5) Associative law :-

OR gate → $A+(B+C) = (A+B)+C$

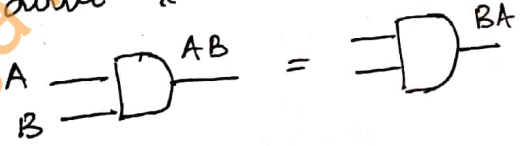
AND gate → $A\cdot(B.C) = (A.B.)C .$

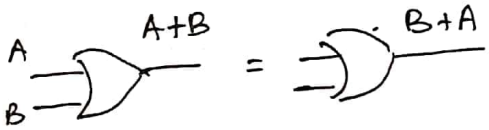
3) Distributive law :-

$$A(B+C) = AB + AC .$$

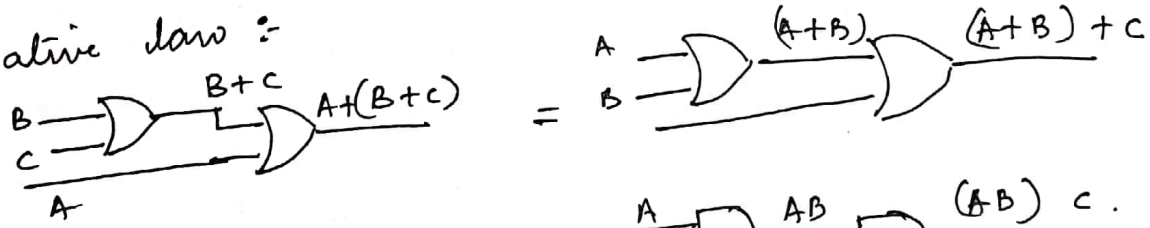
Gate representations :-

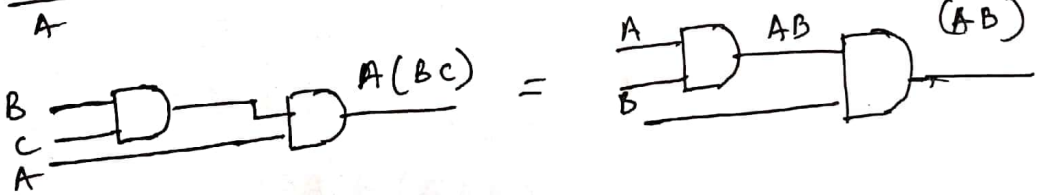
1) Commutative law :-

a) 

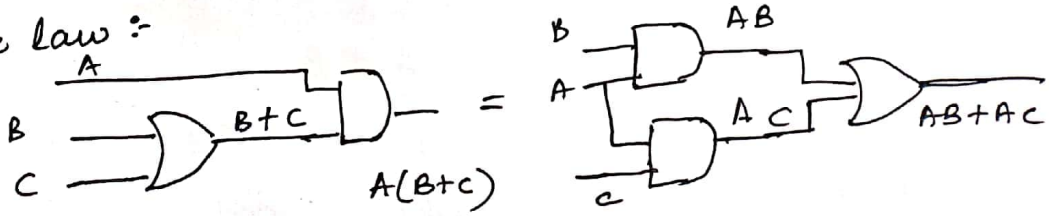
b) 

2) Associative law :-





3) Distributive law :-



Duality Theorem :-

Several boolean relations are derived from various boolean laws mentioned above by.

1) changing each OR to AND
2) changing each AND to OR
3) complementing each 0 to 1 and 1 to 0.

The boolean expression obtained by applying the above rules is said to be the dual of the original expression.

The dual of the expression does not alter the meaning / value of the original expression.

Eg:- dual of $A+0=A$ is $A.1=A$ =..

Eg:- Duality theorem can be used by considering an example.

Consider the boolean expression defining the distributive law,

$A(B+C) = AB + AC$.

applying duality theorem,

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

$= A.A + AC + AB + BC$

$= A + AC + AB + BC$

$= AC(1+C) + AB + BC$

$= A \cdot AB + BC$

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

$A+BC = A+BC$

LHS = RHS $\Rightarrow$ hence proved.

as LHS = RHS duality theorem does not alter the original expression

Some of the boolean expressions and their duals are.

Boolean Expression	Dual
1) $A+B = B+A$	$AB \neq BA$
2) $A+(B+C) = (A+B)+C$	$A.(BC) = (AB).C$
3) $A(B+C) = AB+BC$	$A+BC = (A+B)(B+C)$
4) $A+0 = A$	$A.1 = A$
5) $A+1 = 1$	$A.0 = 0$
6) $A+A = A$	$A.A = A$
7) $A+\bar{A} = 1$	$A.\bar{A} = 0$
8) $\bar{\bar{A}} = A$	$\bar{\bar{A}} = A$
9) $A+\bar{A}B = A+B$	$A(A+B) = A$
10) $A+AB = A$	$A(\bar{A}+B) = AB$

Demorgan's Theorem :-

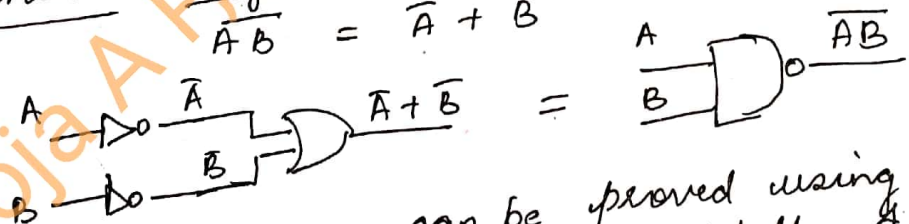
They are useful in simplifying boolean expressions.

1) Demorgan's 1st theorem → $\overline{A+B} = \bar{A}\bar{B}$

2) Demorgan's 2nd theorem. → $\overline{AB} = \bar{A}+\bar{B}$

Consider demorgan's second theorem.

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



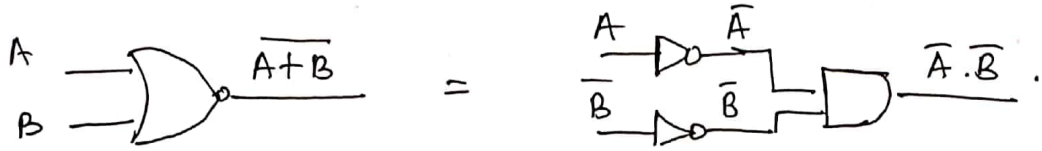
The above theorem can be proved using truth table, There are 2 i/p variable, hence totally 4 i/p combinations

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

as LHS = RHS, the demorgan's theorem is proved.

Consider Demorgan's 1st theorem.

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



The above theorem can be proved by using truth table, as there are 2 input variable, hence total no. of input combinations = 4.

		LHS		RHS		
A	B	A + B	$\overline{A+B}$	$\bar{A}$	$\bar{B}$	$\bar{A}\bar{B}$
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

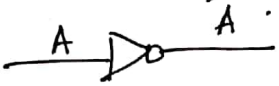
as LHS = RHS hence demorgan's theorem is proved.

Universality of NAND gate and NOR gate

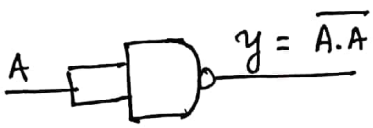
NAND and NOR gates are known as universal gates as they can be used to realize / implement the basic gates and special gates [AND, OR, NOT, X-OR, X-NOR]

Universality of NAND gates:

1) NOT gate using NAND gate.

not gate 

A	y
0	1
1	0



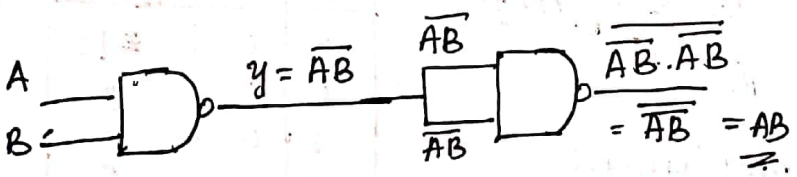
A	$y = \bar{A}$
0	1
1	0

2) AND gate using NAND gate.

A	B	$y = AB$
0	0	0
0	1	0
1	0	0
1	1	1

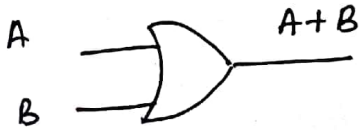
from boolean laws $\bar{\bar{A}} = A$,

$\therefore \overline{\bar{A}\bar{B}} = AB \rightarrow$ the following logic can be used to implement AND gate using NAND only logic.



A	B	$y = \overline{\overline{AB}}$
0	0	0
0	1	0
1	0	0
1	1	1

3) OR gate using NAND gate :



A	B	$y = A+B$
0	0	0
0	1	1
1	0	1
1	1	1

OR gate cannot be implemented directly using NAND gate, it can be implemented by first taking the double complement of the expression, following by applying demorgan's theorem.

∴ o/p of OR gate is

$$y = A + B$$

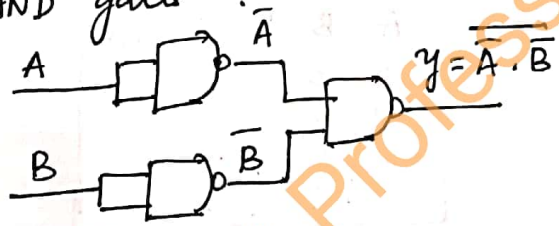
taking double complement.

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

applying demorgan's theorem $\overline{A+B} = \overline{A}.\overline{B}$.

$$\therefore y = \overline{\overline{A}.\overline{B}}$$

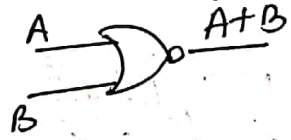
the above equation can be implemented using NAND gate.



A	B	$\overline{A}$	$\overline{B}$	$\overline{A}\,\overline{B}$	$y = \overline{\overline{A}\,\overline{B}}$
0	0	1	1	1	0
0	1	1	0	0	1
1	0	0	1	0	1
1	1	0	0	0	1

4) NOR gate using NAND gate.

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

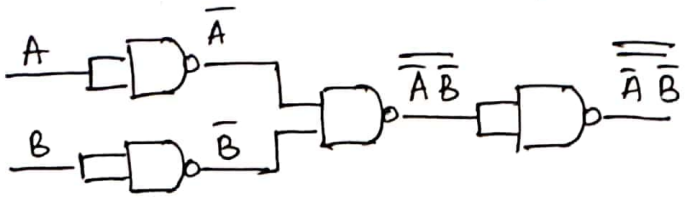


A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

taking double complement

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

$$y = \overline{\overline{A}.\overline{B}}$$

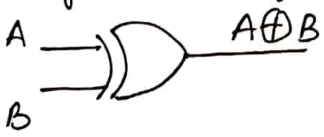


A	B	$\bar{A}$	$\bar{B}$	$\bar{A}\,\bar{B}$	$\overline{\bar{A}\,\bar{B}}$
0	0	1	1	1	1
0	1	1	0	0	0
1	0	0	1	0	0
1	1	0	0	0	0

5) X-OR gate using NAND gate.

$y = A \oplus B$

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



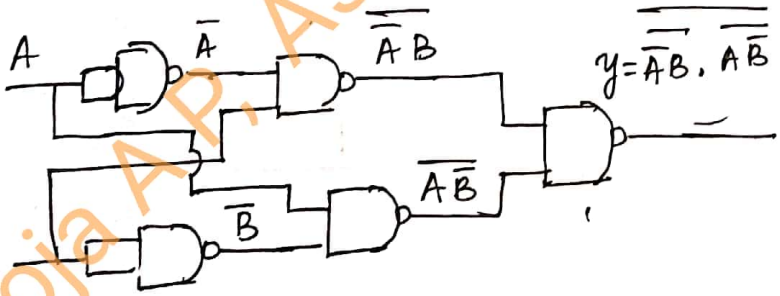
A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

taking double complement.

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

applying demorgan's theorem,

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

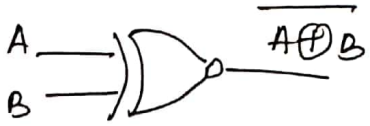


A	B	$\bar{A}$	$\bar{B}$	$\bar{A}B$	$A\bar{B}$	$\overline{\bar{A}B}$	$\overline{A\bar{B}}$	$y = \overline{\bar{A}B} \cdot \overline{A\bar{B}}$
0	0	1	1	0	0	1	1	0
0	1	1	0	0	0	0	1	1
1	0	0	1	0	0	1	0	1
1	1	0	0	0	0	1	1	0

6) X-NOR gate using NAND gate.

$y = \overline{A \oplus B}$

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



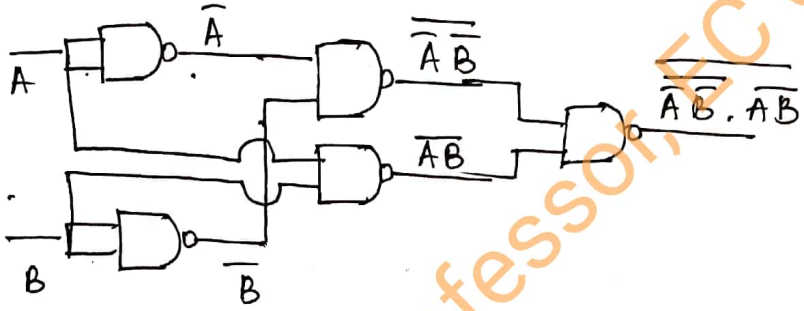
A	B	y
0	0	1
0	1	0
1	0	0
1	1	1

taking double complement.

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

applying demorgan's theorem,

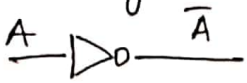
$y = \overline{\overline{\bar{A}\bar{B}} \cdot \overline{AB}}$



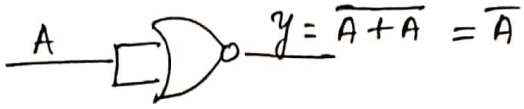
A	B	AB	$\overline{AB}$	$\bar{A}$	$\bar{B}$	$\overline{\bar{A}\bar{B}}$	$\overline{\bar{A}\bar{B}}$	$\overline{\overline{\bar{A}\bar{B}} \cdot \overline{AB}}$
0	0	0	1	1	1	1	0	1
0	1	0	1	1	0	0	1	0
1	0	0	1	0	1	0	1	0
1	1	1	0	0	0	0	1	1

NOR gate as Universal gate.

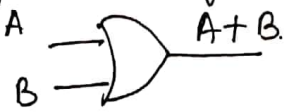
1) NOT gate using NOR gate.



A	y
0	1
1	0.



2) OR gate using NOR gate.



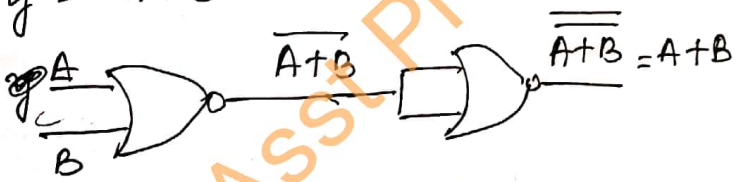
A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

$y = A+B$.

taking double complement does not alter the function because $\bar{\bar{A}} = A$

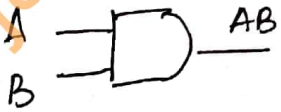
∴ taking double complement

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



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

3) AND gate using NOR gate :-



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

AND gate cannot be implemented directly using NOR gate, it can be implemented by taking double complemented following by applying demorgan's theorem.

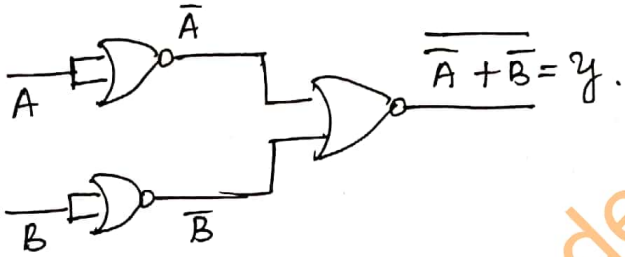
$y = AB$

taking double complement

$y = \overline{\overline{AB}}$

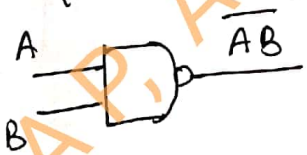
applying demorgan's theorem,

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



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

4) NAND gate using NOR gate.



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

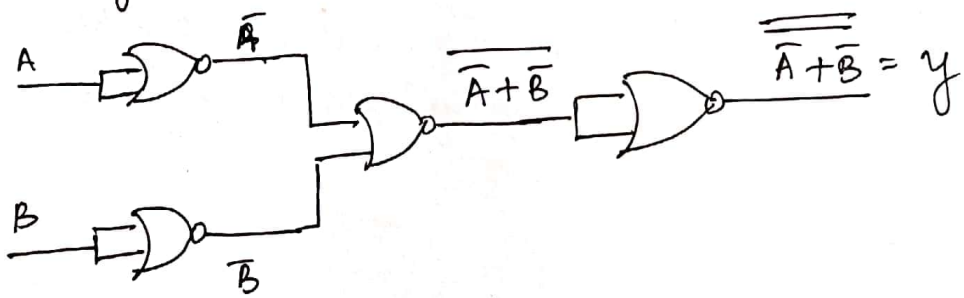
$y = \overline{AB}$

Taking double complement.

$y = \overline{\overline{AB}}$

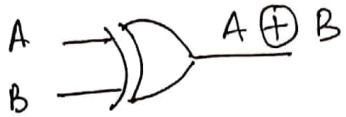
applying demorgan's theorem.

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



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

5) X-OR gate using NOR gate :-



A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

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

taking double complement

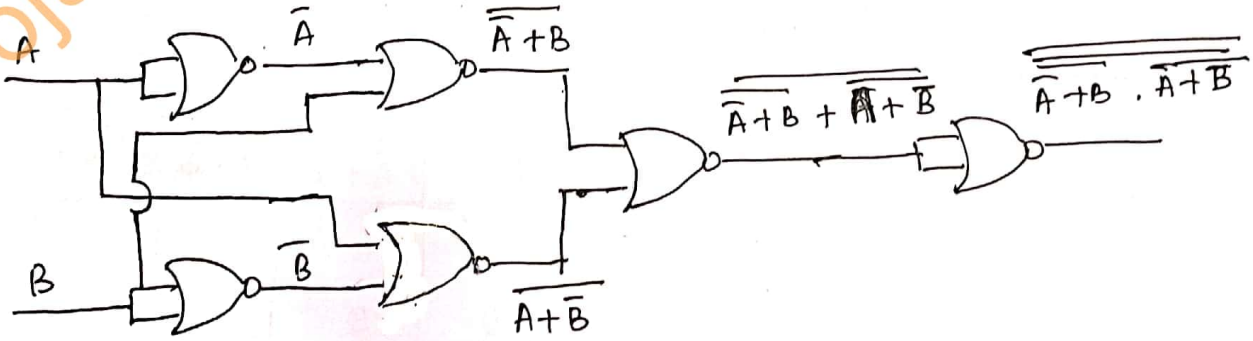
$$y = \overline{\bar{A}B + A\bar{B}}$$

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

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

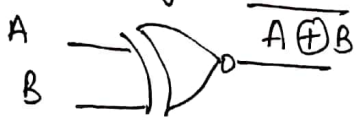
$$y = (\bar{\bar{A}} + \bar{B}) + (\bar{A} + B)$$

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



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

6) X-NOR gate using NOR gate.



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

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

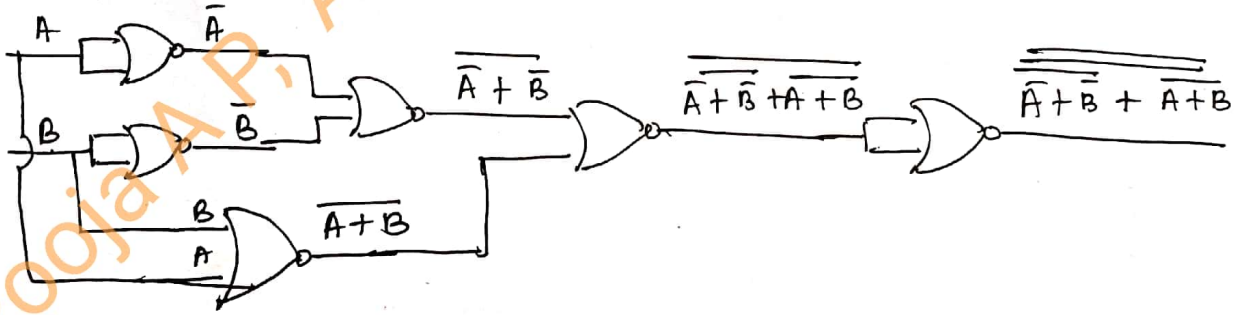
taking double complement

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

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

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

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



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

POS and SOP

POS → Product of Sums

SOP → sum of products.

Consider a 4 variable boolean function.

$$f(w,x,y,z) = \bar{x} + w\bar{y} + \bar{w}\,\bar{y}z \quad -(1)$$

Each occurance of a variable [complement or uncomplement] in the boolean expression - LITERAL.

∴ Eqn (1) has 6 literal.

SOP In Eqn (1) a literal or a product of literals is known as product term.

A boolean formula this is written as a single product term or as a sum of product terms is ~~used~~ said to be in SOP form [Disjunctive normal form].

POS :-

Consider a boolean function,

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

→ 6 literals

A sum or sum of literals is known as SUM TERM

In the eqn there are 3 sum terms → z

→ $x+\bar{y}$

→ $w+\bar{x}+\bar{y}$

A boolean formula that is written as a single sum term or as a product of sum terms is said to be in SOP forms [CONJUNCTIVE NORMAL FORM]

Canonical form

Truth table is used to represent the boolean function. It is used to tabulate the output obtained for every possible input combinations.

Two types of expressions can be obtained from the truth table

1) Minterm canonical form
2) Maxterm canonical form

{ Typical cases of SOP and POS }

1) Minterm Canonical form.

Consider the truth table.

x	y	z	f(x,y,z)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

the function evaluates to 1 at

1) $\bar{x}\bar{y}\bar{z}$ ← P

ie., if $x, y, z = 0, 0, 1$

then $\bar{x}\bar{y}z = 1.1.1 = \underline{1}$ o/p.

2) $\bar{x}yz$

ie., if $xyz = 011$

then $\bar{x}yz = 1.1.1 = 1 = o/p$.

3) $x\bar{y}\bar{z}$

ie., if $(x, y, z) = 100$

then $x\bar{y}\bar{z} = 1.1.1 = 1 = o/p$.

Hence the function evaluates to 1 only for these three product terms.

The above ~~equation~~ results can be combined to obtain the boolean expression.

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

$$= m_1 + m_3 + m_4.$$

$$f(x, y, z) = \sum_{(1,3,4)} m \rightarrow \text{MINTERM CANONICA FORM OR STANDARD SOP FORM}$$

2) Maxterm canonical form.

From the previous truth table here rows corresponding to functional value 0 is considered.

i.e. ⇒ $x + y + z$.

if $x\ y\ z = 0\ 0\ 0$ then.

$x + y + z = 0 + 0 + 0 = 0 \rightarrow$ o/p.

2) $\bar{x} + \bar{y} + z$.

if $x\ y\ z = 0\ 1\ 0$ then.

$x + \bar{y} + z = 0 + 0 + 0 = 0 \rightarrow$ o/p.

3) $\bar{x} + y + \bar{z}$

if $x\ y\ z = 1\ 0\ 1$ then.

$\bar{x} + \bar{y} + \bar{z} = 0 + 0 + 0 = 0 =$ o/p.

4) $\bar{x} + \bar{y} + z$ then,

if $x\ y\ z = 1\ 1\ 0$.

$\bar{x} + \bar{y} + \bar{z} = 0 + 0 + 0 = 0 \rightarrow$ o/p.

5) $\bar{x} + \bar{y} + \bar{z}$

if $x\ y\ z = 1\ 1\ 1$

then, $\bar{x} + \bar{y} + \bar{z} = 0 + 0 + 0 = 0 \rightarrow$ o/p.

The above sum terms can be combined to form the boolean expression

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

$$= M_0 . M_2 M_5 M_6 M_7.$$

$$f(x, y, z) = \Pi M (0\ 2\ 5\ 6\ 7).$$

Sum of Products

Simplify the following boolean expressions.

1) $y = ABCD + ABD$

$= ABD(C + 1) = \underline{ABD}$ ..

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

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

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

$= XY + \bar{X}YZ$

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

$= \underline{XY + YZ}$.

$YZ(\cdot X + \bar{X})$

$+ XY(\bar{Z}+1)$

$= YZ + XY$

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

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

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

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

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

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

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

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

$= \bar{C}(\bar{B}+B) = \underline{\bar{C}}$.

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

$= (A\bar{B} + \bar{A}C)[ABC.BC + ABC.B\bar{C}]$

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

$= \underline{\underline{0}}$ ..

6. Simplify and realize using basic gates :

1) $y = \overline{(C + DE)(CE + DF)}$

applying de morgan's theorem $\quad \overline{AB} = \bar{A} + \bar{B}$

$y = \overline{(C + DE)} + \overline{(CE + DF)}$

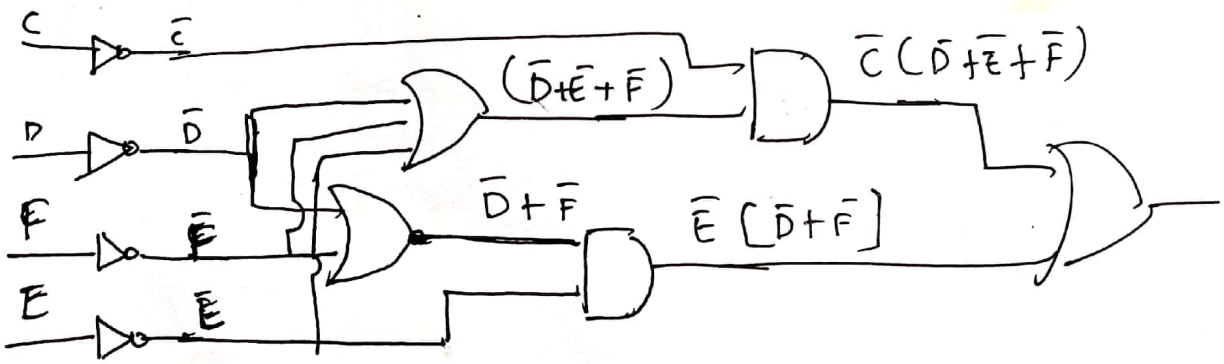
$y = (\bar{C}.\overline{DE}) + (\overline{CE}\ \overline{DF})$

$y = \bar{C}(\bar{D} + \bar{E}) + [(\bar{C} + \bar{E})(\bar{D} + \bar{F})]$

$y = \bar{C}\bar{D} + \bar{C}\bar{E} + \bar{C}D + \bar{C}\bar{F} + \bar{D}\bar{E} + D\bar{E}\bar{F}$

$y = \bar{C}\bar{D} + \bar{C}\bar{E} + \bar{C}\bar{F} + \bar{D}\bar{E} + \bar{E}\bar{F}$

$y = \bar{C}[\bar{D} + \bar{E} + \bar{F}] + E[\bar{D} + \bar{F}]$



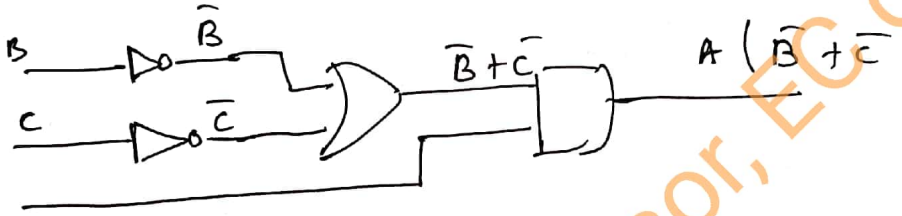
2) $y = A(\overline{ABC} + A\bar{B}C)$

$y = A[\bar{A} + \bar{B} + \bar{C} + A\bar{B}C]$

$= [A\bar{A} + A\bar{B} + A\bar{C} + A\bar{B}C]$

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

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



3) $y = A[B + C(\overline{AB + AC})]$

$y = AB + AC[\bar{A}\bar{B} \cdot \bar{A}\bar{C}]$

$y = AB + AC[(\bar{A} + \bar{B})(\bar{A} + \bar{C})]$

$= AB + AC[\bar{A}\bar{A} + \bar{A}\bar{B} + \bar{A}\bar{C} + \bar{B}\bar{C}]$

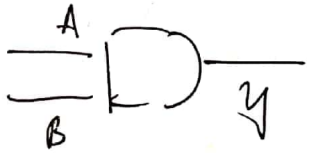
$= AB + AC[\bar{A} + \bar{A}\bar{B} + \bar{A}\bar{C} + \bar{B}\bar{C}]$

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

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

$= AB + AC[\bar{A} + \bar{B}\bar{C}]$

$y = AB$



4) $y = A \oplus B + C$

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

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

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

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

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

$= [A\bar{A} + \bar{A}\bar{B} + AB + B\bar{B}]C + \bar{A}B\bar{C} + A\bar{B}\bar{C}$ ($B\bar{B} = 0$)

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

5) $y = \overline{\overline{XY + XYZ} + X(Y + X\bar{Y})}$

$y = (\overline{\overline{XY + XYZ}})\,\overline{(X(Y + X\bar{Y}))}$

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

$= XY[\bar{X} + \bar{Y}(\overline{X\bar{Y}})]$

$= XY[\bar{X} + \bar{Y}(\bar{X} + \bar{\bar{Y}})]$

$= XY[\bar{X} + \bar{X}\bar{Y} + \bar{Y}Y]$ ($\bar{Y}Y = 0$)

$F = 0.$

c) $y = \overline{\overline{A}BC\,D}$

$y = \overline{\overline{A}BC} + \overline{D}$

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

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

Simplify and realize using NAND only.

1) $y = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$

$y = AC(\overline{B} + B) + \overline{A}BC + AB\overline{C}$

$= AC + \overline{A}BC + AB\overline{C}$

$= C[A + \overline{A}B] + AB\overline{C}$

$= C[A + B] + AB\overline{C}$

$= AC + ABC + AB\overline{C}$

$= AC + AB(\overline{C} + 1)$

$= A(B + C)$

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

$\overline{A}BC + A\overline{B}C + \cancel{AB}AB\overline{C} + ABC$

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

$= \overline{A}BC + AB\overline{C} + AC$

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

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

$= \overline{A}BC + AC + AB.$

$= C[\overline{A}B + A] + AB$

$= C[A + B] + AB = AB + AC + BC.$

$y = \overline{\overline{AB} + AC + BC}$

$y = \overline{\overline{AB} \cdot \overline{AC} \cdot \overline{BC}}$

9) $y = A(\overline{ABC} + A\bar{B}C)$

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

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

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

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

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

Simplify the following using NOR only,

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

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

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

$y = C[A + \bar{A}B] + AB\bar{C}$

$y = C[A + B] + AB\bar{C}$

$y = AC + AB + AB\bar{C}$

$y = A[C + B\bar{C}] + AB$

$y = A[B + C] + AB$

$y = \overline{\overline{AB + Ac + AB}}$

$y = \overline{\bar{A}B} \cdot \overline{A\bar{C}} \cdot \overline{A\bar{B}}$

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

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

$y = \overline{\overline{\bar{A} + \bar{B}} + \overline{\bar{A} + \bar{C}} + \overline{\bar{A} + \bar{B}}}$.

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

taking double complement

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

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

Gate structure must be drawn for all problems.

Karnaugh Maps [K-Maps]

Diagrammatic representation of boolean functions.

1) One variable K-map.

$n = 1$ ∴ no. of cells = $2^1 = 2$ cells.

Truth table

x	f(x)
0	f(0)
1	f(1)

x = 0	x = 1
f(0) (0)	f(1) (1)

2) Two variable K-map.

$n = 2$ ∴ no. of cells = 4 cells and 4 different input combinations.

Truth table

x	y	f(x,y)
0	0	f(0,0)
0	1	f(0,1)
1	0	f(1,0)
1	1	f(1,1)

K-map

x \ y	0	1
0	f(0,0) (0)	f(0,1) (1)
1	f(1,0) (2)	f(1,1) (3)

Simplification methods of grouping of cells,

fig (1)

x \ y	0	1
0	1	1
1		

fig (2)

x \ y	0	1
0	1	
1	1	

(3)

x \ y	0	1
0	1	1
1	1	1

The boolean expression for the given K-map consists of the o/p variable that are common to the grouped cells.

(1) & (2) represents 2 cells grouping

fig 1) $f(x,y) = \bar{x}$ [y changes from 0 to 1 hence not included]

fig 2) $f(x,y) = \bar{y}$ [x changes from 0 to 1 hence not included]

fig (3) → both x & y changes

∴ $f(x,y) = 1$

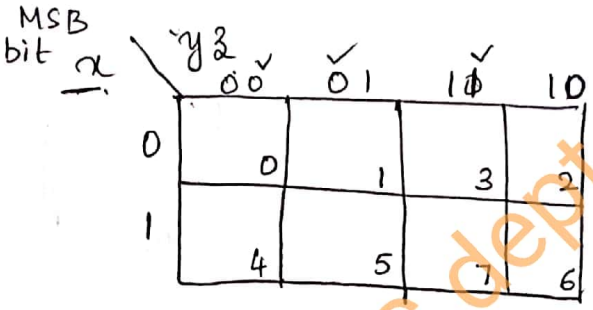
Three variable K-map.

$n = 3$ ∴ no of cells $= 2^3 = 8$ cells ; 8 i/p combinations

Truth table

x	y	z	f(x,y,z)
0	0	0	f(0,0,0)
0	0	1	f(0,0,1)
0	1	0	f(0,1,0)
0	1	1	f(0,1,1)
1	0	0	f(1,0,0)
1	0	1	f(1,0,1)
1	1	0	f(1,1,0)
1	1	1	f(1,1,1)

K - map

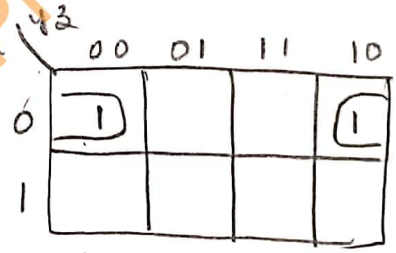


the cell nos are arranged such that there is only one bit change b/w adjacent positions

Grouping of cells.

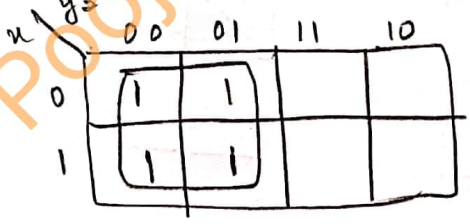
2 cell grouping.

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

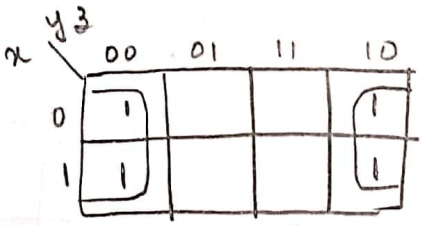


$f(x,y,z) = \bar{x}\bar{z}$

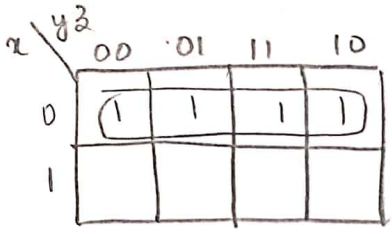
3 cell grouping



$f(x,y,z) = \bar{y}$

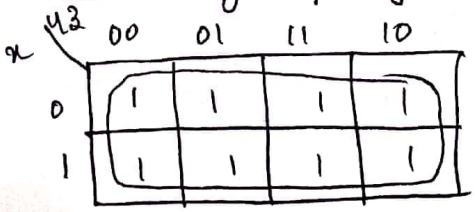


$f(x,y,z) = \bar{z}$



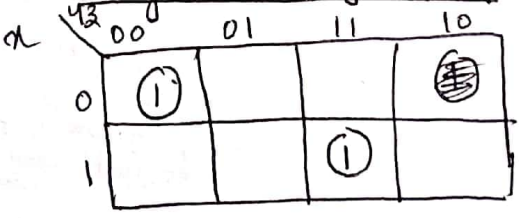
$f(x,y,z) = \bar{x}$

8 cell grouping



$f(x\,y\,z) = 1$

Single cell grouping



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

Simplify using K-maps.

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

Truth table

A	B	C	f(A,B,C)	minterms
0	0	0	0	
0	0	1	0	
0	1	0	1	→ m_2
0	1	1	0	
1	0	0	0	
1	0	1	0	
1	1	0	1	→ m_6
1	1	1	1	→ m_7

$\therefore f(A,B,C) = m_2 + m_6 + m_7$

$= \sum m (2,6,7)$

A \ BC	00	01	11	10
0	0	1	3	1 (2)
1	4	5	7	1 (6)

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

2) $f(x,y,z) = \sum m (0,1,3,4,5,7)$

Truth table

x	y	z	f(x,y,z)	minterms
0	0	0	1	→ m_0
0	0	1	1	→ m_1
0	1	0	0	
0	1	1	1	→ m_3
1	0	0	1	→ m_4
1	0	1	1	→ m_5
1	1	0	0	→ m_7
1	1	1	1	

x \ yz	00	01	11	10
0	1 (0)	1 (1)	1 (3)	2
1	1 (4)	1 (5)	1 (7)	6

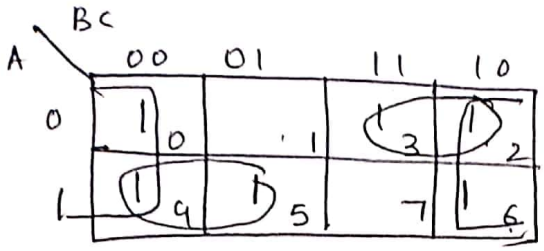
$f(x,y,z) = \bar{y} + z$

3) $y(A,B,C) = \sum M (0,1,3,5)$

A \ BC	00	01	11	10
0	1 (0)	1 (1)	1 (3)	2
1	4	1 (5)	7	6

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

4) $f(ABC) = \sum m(0, 2, 3, 4, 5, 6)$



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

5) $f = \bar{A}B + \bar{A}C + BC + A\bar{B}C$

as the above expression is in normal form, it should be converted to canonical form before applying K-MAP to the above boolean function

wkt $A + \bar{A} = 1$ and $1.A = A$

$f = \bar{A}1$

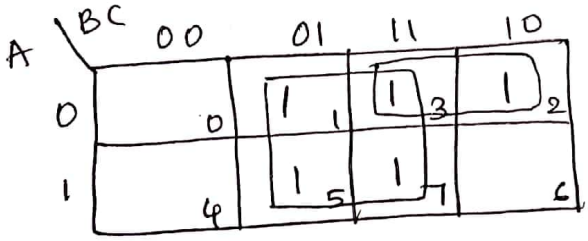
using the above relationships in the given function,

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

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

$= m_2 + m_3 + m_1 + m_7 + m_5$

$= \sum m(1, 2, 3, 5, 7)$

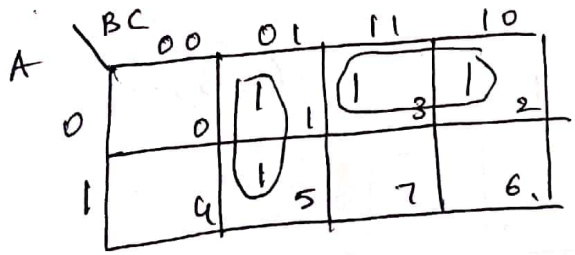


$f = C + \bar{A}B$

6) $f = \bar{A}B\bar{C} + \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}C$

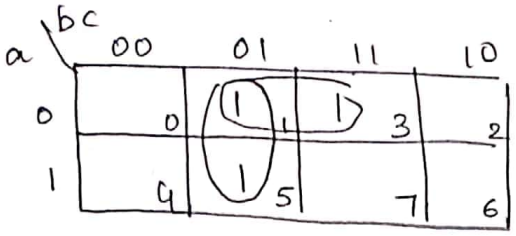
$= m_2 + m_1 + m_3 + m_5$

$= \sum m(2, 1, 3, 5)$



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

7) $f(a,b,c) = \sum m(1,3,5)$



$f = \bar{a}c + \bar{b}c$

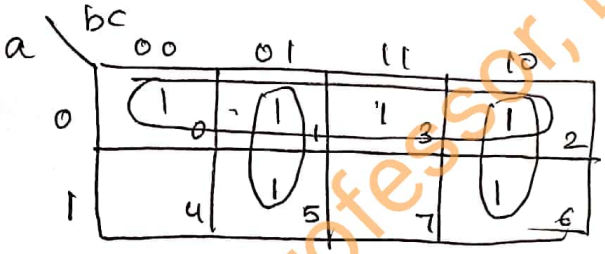
8) $f(a,b,c) = ab\bar{c} + \bar{b}c + \bar{a}$

$= ab\bar{c} + (a+\bar{a})\bar{b}c + \bar{a}(b+\bar{b})(c+\bar{c})$

$= ab\bar{c} + a\bar{b}c + \bar{a}\bar{b}c + \bar{a}bc + \bar{a}\bar{b}c + \bar{a}\bar{b}\bar{c} + \bar{a}\bar{b}\bar{c}$

$= m_6 + m_5 + m_3 + m_0 + m_2 + m_1$

$= \sum m$
$(6,5,3,0,2,1)$.



$f = \bar{a} + \bar{b}c + b\bar{c}$

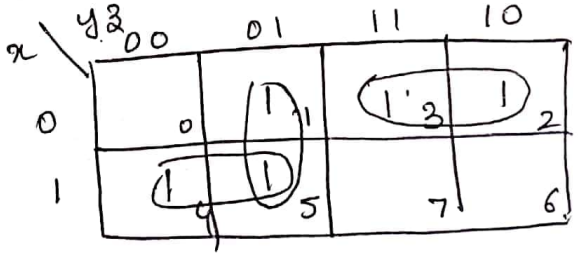
9) $f(x,y,z) = x\bar{y} + \bar{x}y + \bar{y}z$.

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

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

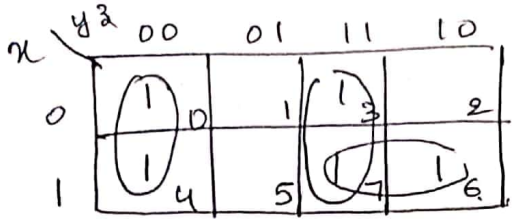
$= m_5 + m_4 + m_3 + m_2 + m_1$

$= \sum m$
$(1,2,3,4,5)$



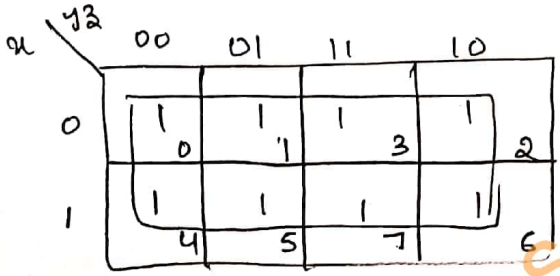
$f(x,y,z) = x\bar{y} + \bar{y}z + \bar{x}y$

10) $f(x, y, z) = \sum m(0, 4, 3, 7, 6)$.



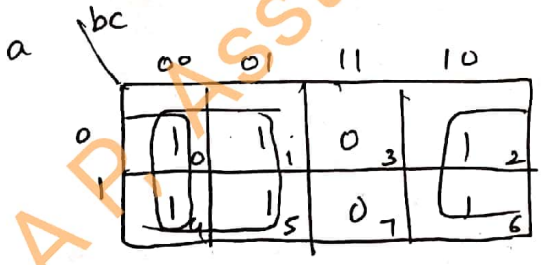
$f(x, y, z) = \bar{y}\bar{z} + xy + yz$

11) $f(x, y, z) = \sum m(0, 1, 2, 3, 4, 5, 6, 7)$



$f(x, y, z) = 1$

12) $f(a, b, c) = \sum m(0, 1, 2, 4, 5, 6)$



$f(a, b, c) = \bar{c} + \bar{b}$.

Rules of Binary addition [mod 2 addition]

It results in only 4 possible cases,

$0 + 0 \rightarrow$ sum = 0, carry = 0

$0 + 1 \rightarrow$ sum = 1, carry = 0

$1 + 1 \rightarrow$ sum = 0, carry = 1

$1 + 1 + 1 \rightarrow$ sum = 1, carry = 1

Eg:-

```
   1 <- C
   1  1  0  0
   1  1  1  0
------------------
1  1  0  1  0
------------------
```

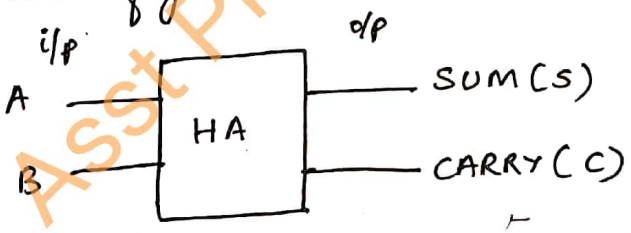
$(1100)_2 \rightarrow 12$ d.

$+ (1110)_2 \rightarrow +14$ d

$(11010)_2 \rightarrow 26$ d.

Half adder :-

The block diagram of a half adder is as shown in the figure below,



→ It perform 2 single bit addition

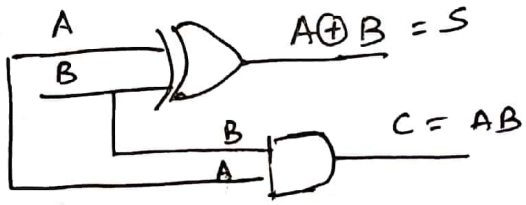
→ It does not take into consideration, the carry generated from previous addition

TRUTH TABLE

i/p		o/p	
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

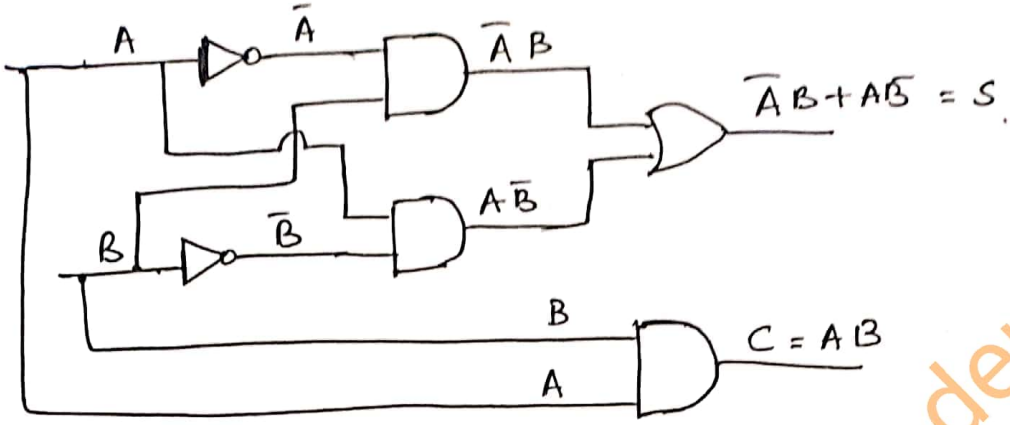
S (sum) $= \bar{A}B + A\bar{B} = A \oplus B$

C (carry) $= AB$



Half adder using basic gates :-

$S = \bar{A}B + A\bar{B}$ $\quad C = AB$



Half adder using NAND only logic :-

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

taking double complement

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

applying demorgan's theorem, $\overline{A+B} = \bar{A} \cdot \bar{B}$

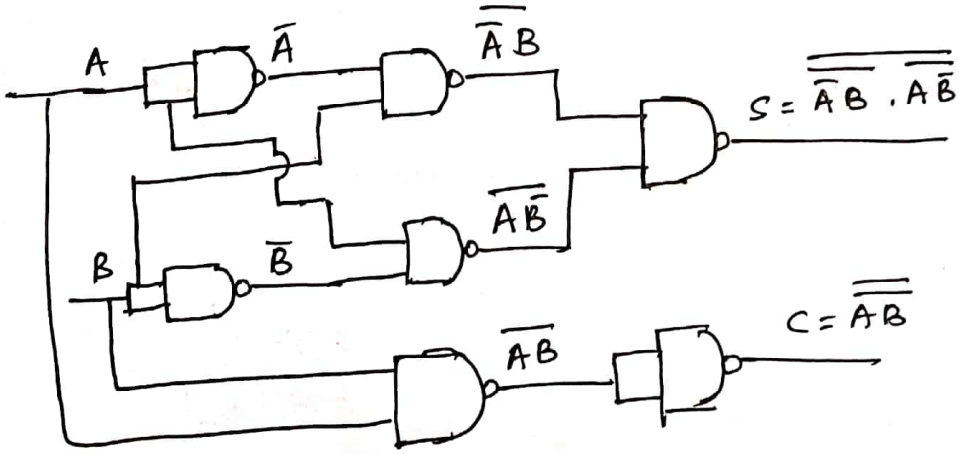
$$S = \overline{\overline{\bar{A}B} \cdot \overline{A\bar{B}}}$$

$C = AB$

taking double complement

$$C = \overline{\overline{AB}}$$

7 NAND gates are needed.



Half adder using minimum NAND gates [5 NAND gates]

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

let x be a complement function ie., $X = \bar{A}$ or $X = \bar{B}$

$\therefore$, $X = \bar{A}$ or $\bar{B} = \bar{A} + \bar{B}$

$\Rightarrow$ $S = XB + AX$

$X = \bar{A}$ $\quad$ $X = \bar{B}$

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

applying demorgan's theorem,

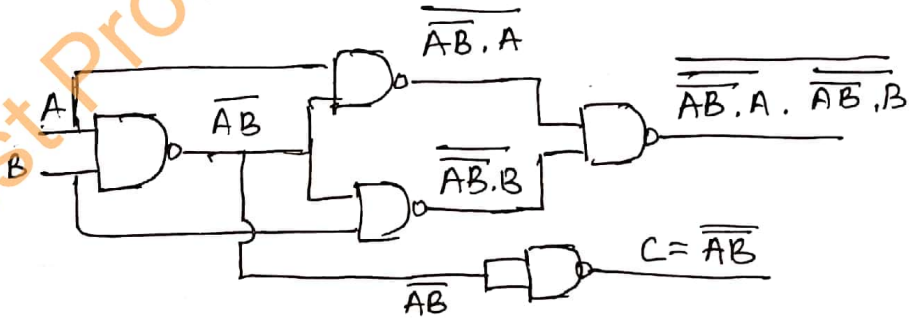
$S = \overline{AB}.B + A\overline{AB}$

taking double complement & applying demorgan's theorem,

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

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

$C = AB = \overline{\overline{AB}}$



the equation can also be verified from output end,

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

applying demorgan's theorem,

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

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

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

$S = \cancel{A\bar{A}}^{0} + A\bar{B} + \bar{A}B + \cancel{B\bar{B}}^{0}$

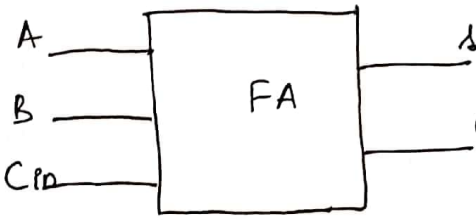
$S = \bar{A}B + A\bar{B}$ = sum eqn of half adder

Full adder :-

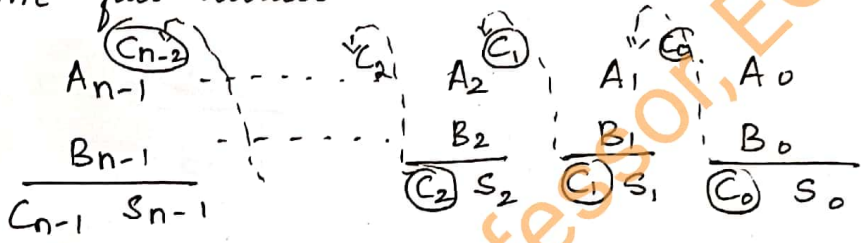
Performs addition of 3 binary bits. Full adder also considers the carry generated from the previous sum to perform addition.

Thus the third bit here is the carry from previous addition (Cin).

The block diagram of a full adder is as shown in the figure below,



The full adder addition can be expressed as,



∴ the final sum expression is $S = C_{n-1} S_{n-1} \ldots\ldots S_2 S_1 S_0$.

Truth table :-

A	B	Cin	S	Co
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

using K-maps

Sum :

A \ BCin	00	01	11	10
0	0 (0)	1 (1)	0 (3)	1 (2)
1	1 (4)	0 (5)	1 (7)	0 (6)

→ Checkerboard pattern.

∴ $S = A \oplus B \oplus C_{in}$.

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

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

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

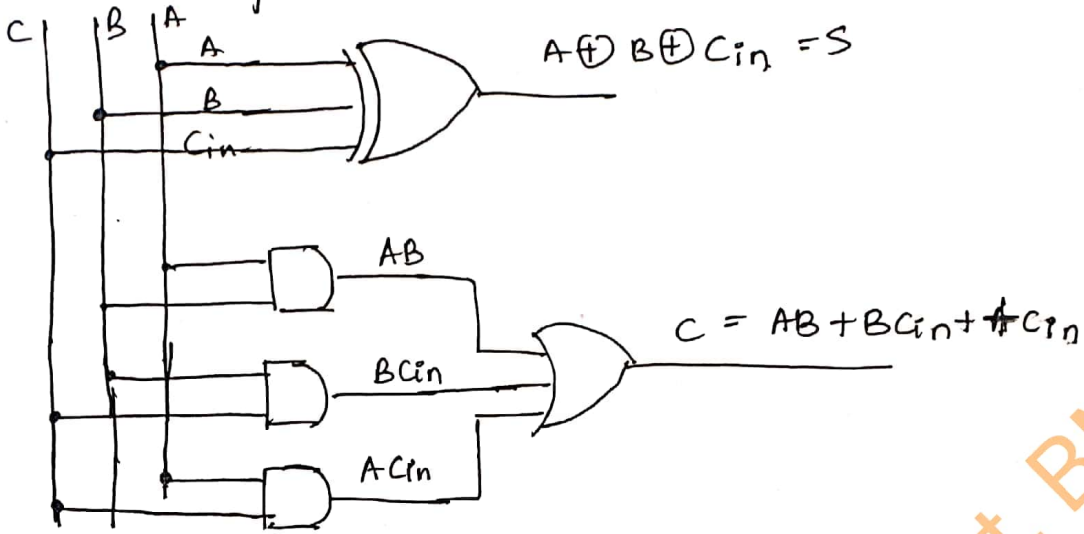
$S = A \oplus B \oplus C$.

Carry :

A \ BCin	00	01	11	10
0	0	1	1 (3)	2
1	0 (4)	1 (5)	1 (7)	1 (6)

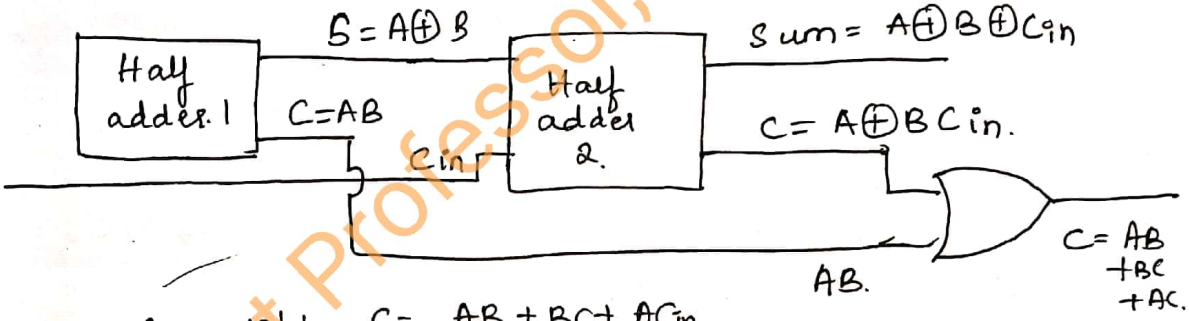
$C = BC_i + AC_i + AB$.

Gate level representation



Implementation of full adder using 2 half adder.

Half adder does not use the carry generated from previous addition.



∴ Ckt C = AB + BCin + ACin

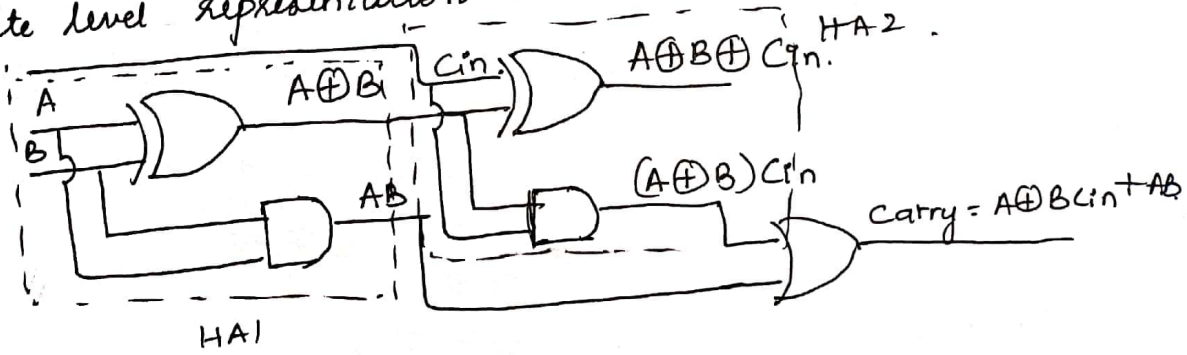
carry from HA1 is AB.

The carry from half adder 2 is product of the inputs applied to HA2

i.e., $C_2 = (A \oplus B)C_{in}$

∴ the total carry = $AB + (A \oplus B)C_{in}$.

∴ the gate level representation is



Implementation of full adder using minimal NAND gates.

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

let $A \oplus B = y$, then,

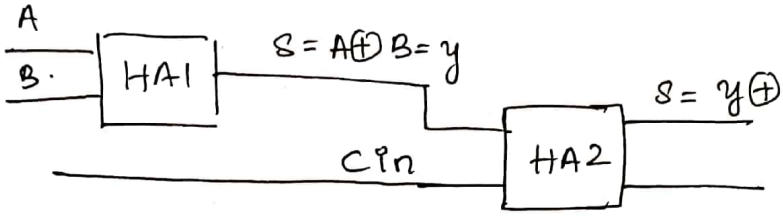
Sum = $y \oplus C_{in}$

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

let $A \oplus B = y$

carry = $AB + C_{in}y$

The sum equation can be written as,

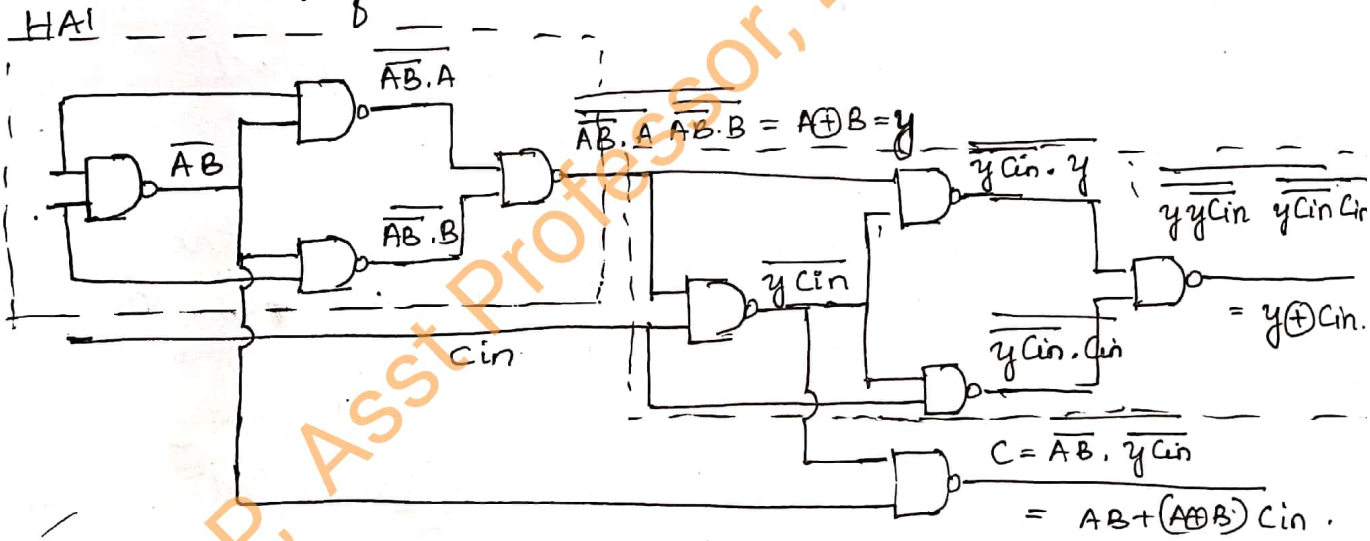


The sum equation represents X-OR operation, XOR gate can be represented using 4 NAND gates.

In stage 1 [HA1] → A and B are added ⇒ S = A⊕B = y

stage 2 [HA2] ⇒ A⊕B is added with Cin.

∴ final sum = $A \oplus B \oplus C_{in} = y \oplus C_{in}$.



Consider sum eqn of HA1.

$S = \overline{A\overline{AB}}\ \overline{B\overline{AB}}$

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

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

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

$S = \overline{A}B + A\overline{B} = y$

Consider sum of HA2

$S = y\overline{y}C_{in}\ \overline{y}C_{in}C_{in}$

$S = y\overline{y}C_{in} + \overline{y}C_{in}C_{in}$

$S = y\overline{y}C_{in} + \overline{y}C_{in}C_{in}$

$S = y(\overline{y}+\overline{C_{in}}) + C_{in}(\overline{y}+\overline{C_{in}})$

$S = y\overline{C_{in}} + \overline{y}C_{in} = y \oplus C_{in}$

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

carry = $\overline{AB}.\overline{y}C_{in}$

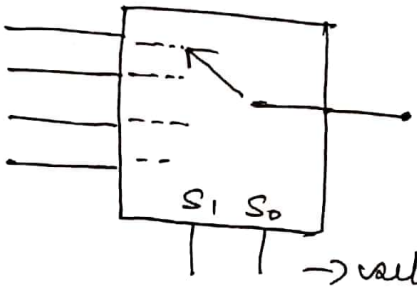
$= \overline{AB} + \overline{y}C_{in}$

$= AB + yC_{in}$ = $AB + (A \oplus B)C_{in}$ = Carry

Multiplexers :-

It is a combinational circuit that selects binary information from one of the many input lines and directs it to the output line.

Eg:- simple data selector



→ select line or selector variable

the selector lines S_1, S_0 are used to select or choose which input line is to be sent to the output.

Advantages → reduces number of wires
→ reduces circuit complexity & cost.

The different types of multiplexers are

1) 2 : 1 mux
2) 4 : 1 mux
3) 8 : 1 mux
⋮

The multiplexer is also known as MUX. It always has a single output line.

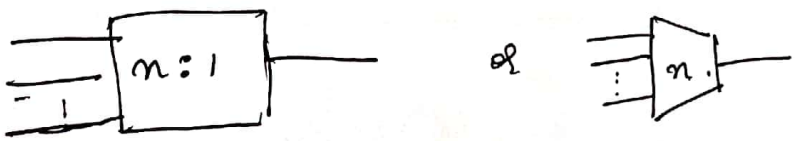
The number of input combinations $= 2^m$. → [m select variable]

ie, 2 combinations $[I_1, I_0] = 2^1$ → [1 select variable]

8 combinations $[I_0 \dots I_7] = 2^3$ and so on [3 select variable]

here m gives the no. of select lines to be used for the given mux

The mux can be represented as

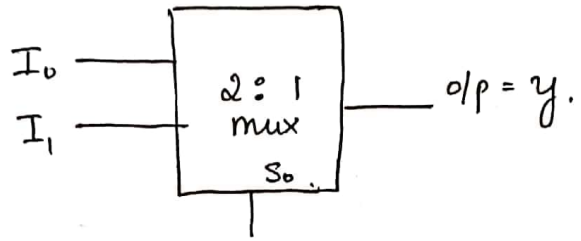


1) 2:1 mux :-

no. of input combinations 2.

∴ no. of select variables is $2^1 = 1$.

A 2:1 mux is represented as follows



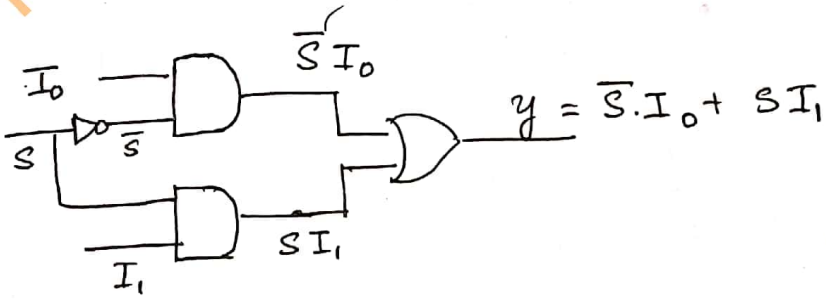
The select variable S_0 selects which input variable must be sent to the output y.

Truth table

I_0	I_1	S_0	y
		0	I_0
		1	I_1

$y = I_0\bar{S_0} + I_1S_1$

∴ the gate level implementation is given by.

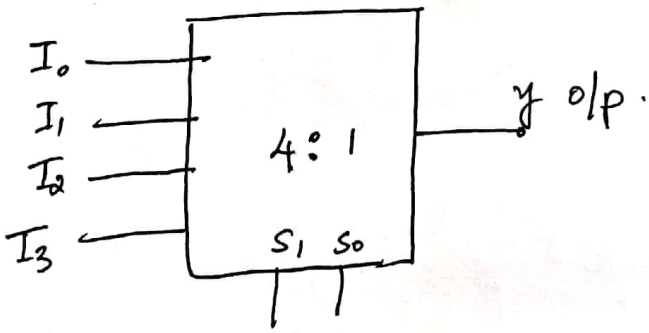


2) 4:1 multiplexer :-

no. of input combinations = 4.

∴ no. of select variables is $2^2 = 4 \Rightarrow$ 2 select lines.

A 4:1 mux is represented as follows,



The select variable $S_1 S_0$ decides which input line must be connected to the output.

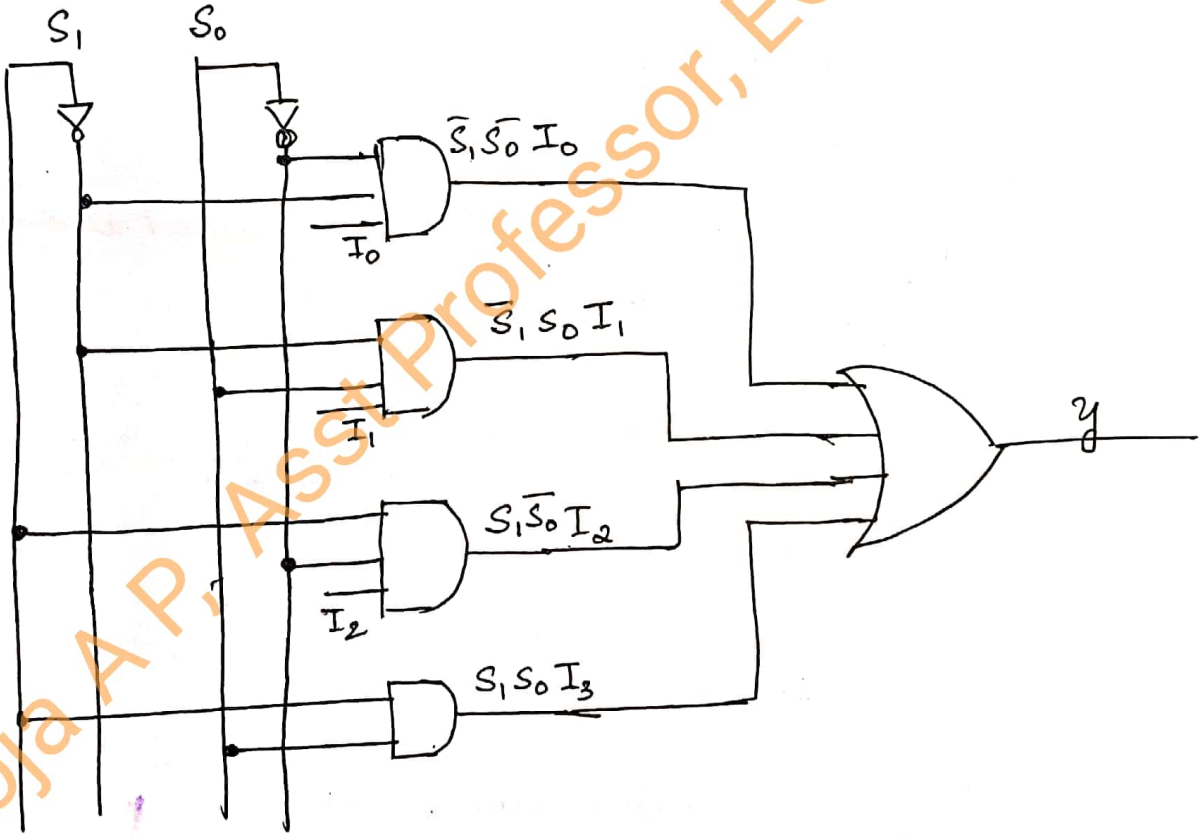
Truth table

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

$\therefore y = \bar{S_1}\bar{S_0} I_0 + \bar{S_1} S_0 I_1 + S_1 \bar{S_0} I_2 + S_1 S_0 I_3$

$\therefore$ the ar

$\therefore$ the gate level implementation is given by,



Implementation of 4:1 mux using 2:1 mux.

For a 4:1 mux., the no. of input combination is 4.

∴ no. of 2:1 mux needed to implement 4:1 mux is.

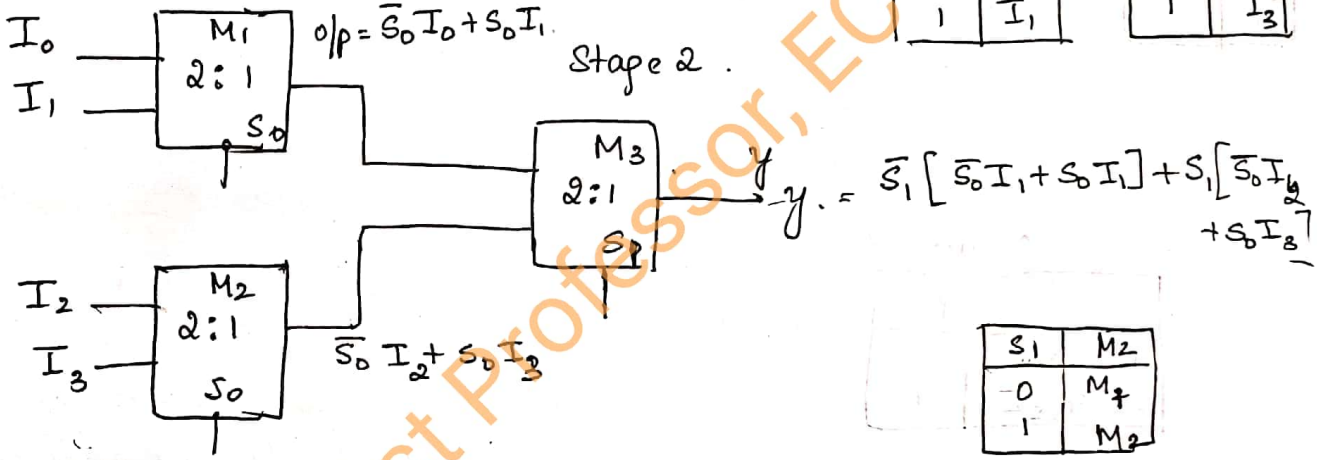
$$\frac{4}{2} = 2.$$

$$\frac{2}{1} = 1$$

3 → we need 3 2:1 mux to implement a 4:1 mux.

M_1	
S_0	o/p
0	I_0
1	I_1

M_2	
S_0	o/p
0	I_2
1	I_3



$$y = \bar{S}_1[\bar{S}_0 I_1 + S_0 I_1] + S_1[\bar{S}_0 I_2 + S_0 I_3]$$

S_1	M_2
0	M_1
1	M_2

The truth table of a 4:1 mux is given by

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

∴ the 4:1 mux has 2 select lines but here we have 3 2:1 mux hence there are 3 select lines. S_0 one of the select line is made common to 2 2:1 mux.

Hence let select line S_0 be used in stage 1 and S_1 in stage 2.

∴ When $S_1 S_0 = 00$. output of $m_1 = I_0$

output of $m_2 = I_2$.

I_0, I_2 are i/p to stage 2 mux [M_3].

as $S_1 = 0$ ∴ the I_0 is sent to the output.

Full Adder :- ~~performs~~ 3 bit ~~binary~~ c

case 2 :- when $S_1 S_0 = 01$

when $S_0 = 01$ output of mux 1 = I_1

output of mux 2 = I_3.

I_1 and I_3 are i/p to stage 2.

as $S_1 = 0 \Rightarrow$ o/p of mux 3 = I_1

case 3 : when $S_1 S_0 = 10$.

$S_0 = 0 \rightarrow$ o/p of mux 1 = I_0, o/p of mux 2 = I_2.

$S_1 = 1 \rightarrow$ o/p of mux 3 = I_2.

case 4 :- when $S_1 S_0 = 11$

$S_0 = 1 \rightarrow$ o/p of mux 1 = I_1

o/p of mux 2 = I_3

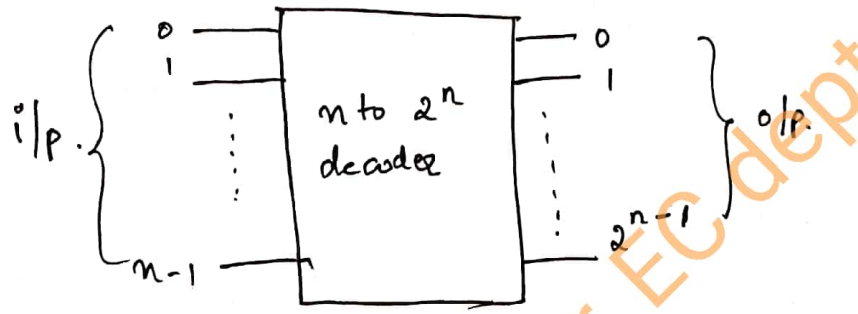
as $S_1 = 1 \rightarrow$ o/p of mux 3 = I_3

Hence from the above analysis it is clear that three 2:1 mux can be used to implement a 4:1 mux successfully.

Decoders :-

A binary decoder is a combinational logic circuit that converts binary information from n coded inputs to a maximum of 2^n unique outputs.

A general n i/p 2^n o/p decoder is as shown in the figure below,



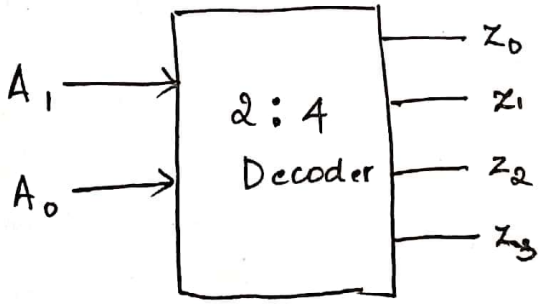
Here only one of the 2^n o/p lines responds with a logic 1 to the given input combination of values on its n i/p lines.

1) 2 : 4 Decoder :-

A 2:4 decoder has two inputs and 4 outputs ie., $n = 2 \rightarrow$ i/p

$2^n = 2^2 = 4 \rightarrow$ o/p.

The block diagram is shown in figure below,



here one of these four outputs will be 1 for each combination of inputs :

Truth table

inputs		outputs			
A_1	A_0	Z_0	Z_1	Z_2	Z_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

∴ the boolean expression for o/p are

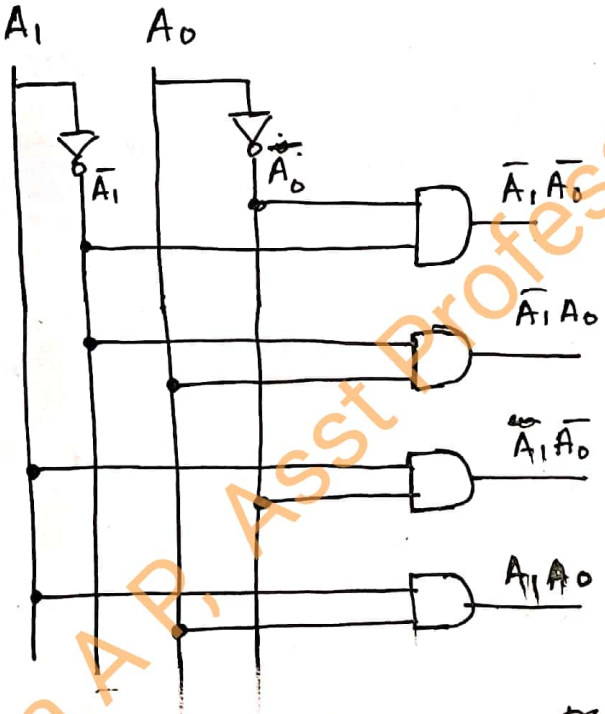
$Z_0 = \bar{A}_1 \bar{A}_0$

$Z_1 = \bar{A}_1 A_0$

$Z_2 = A_1 \bar{A}_0$

$Z_3 = A_1 A_0$

Hence the logic diagram of a 2:4 decoder is as shown below.



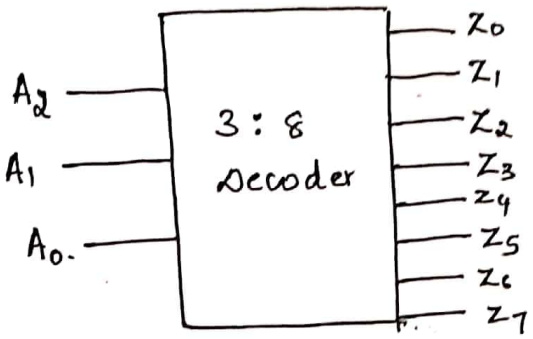
This each o/p has one product term. hence there are four product terms in total. The four product terms are implemented using AND and NOT gates.

2) 3 : 8 Decoder :-

A 3 : 8 decoder has three inputs and 8 outputs. ie., $n = 3 \rightarrow$ I/P

$2^n = 2^3 = 8 \rightarrow$ O/P.

The block diagram is as shown in figure below.

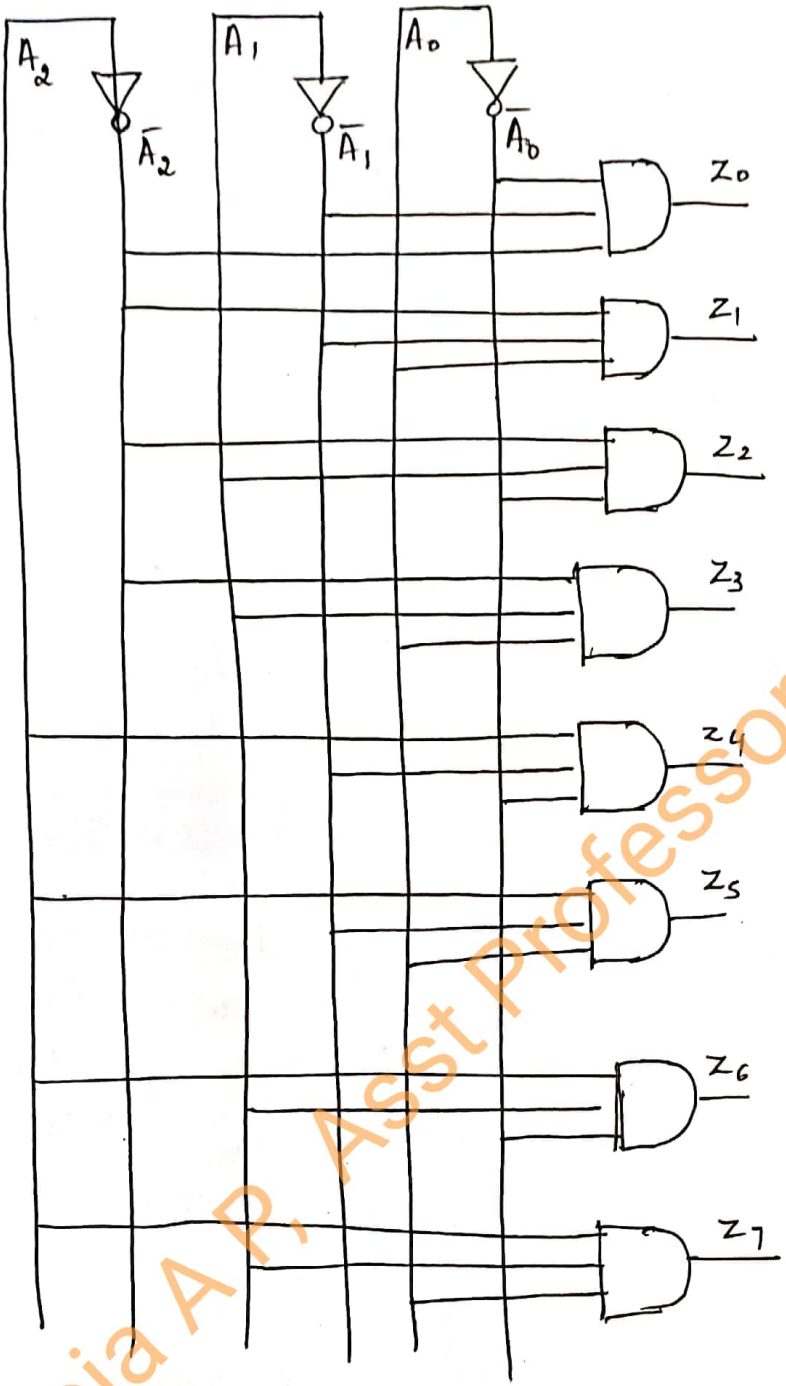


here one of the 8 ~~deep~~ outputs will be high for each combination of inputs.

Truth table

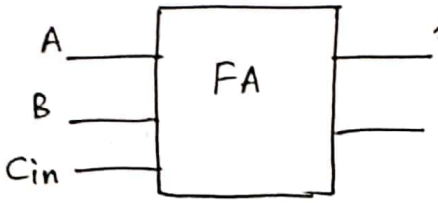
I/P			O/P.								
A_2	A_1	A_0	Z_0	Z_1	Z_2	Z_3	Z_4	Z_5	Z_6	Z_7	Boolean exp
0	0	0	1	0	0	0	0	0	0	0	$Z_0 = \bar{A}_2 \bar{A}_1 \bar{A}_0$
0	0	1	0	1	0	0	0	0	0	0	$Z_1 = \bar{A}_2 \bar{A}_1 A_0$
0	1	0	0	0	1	0	0	0	0	0	$Z_2 = \bar{A}_2 A_1 \bar{A}_0$
0	1	1	0	0	0	1	0	0	0	0	$Z_3 = \bar{A}_2 A_1 A_0$
1	0	0	0	0	0	0	1	0	0	0	$Z_4 = A_2 \bar{A}_1 \bar{A}_0$
1	0	1	0	0	0	0	0	1	0	0	$Z_5 = A_2 \bar{A}_1 A_0$
1	1	0	0	0	0	0	0	0	1	0	$Z_6 = A_2 A_1 \bar{A}_0$
1	1	1	0	0	0	0	0	0	0	1	$Z_7 = A_2 A_1 A_0$

The logic diagram of a 3 : 8 decoder is as shown below.



Implementation of full adder using 3 : 8 decoder :-

The block diagram of a full adder is given by



The truth table and the expression for sum and carry output is .

A	B	Cin	(S) sum	(C) Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$sum = \bar{A}\bar{B}C_{in} + \bar{A}B\bar{C}in + A\bar{B}\bar{C}in + ABCin$.

$= m_1 + m_2 + m_4 + m_7$

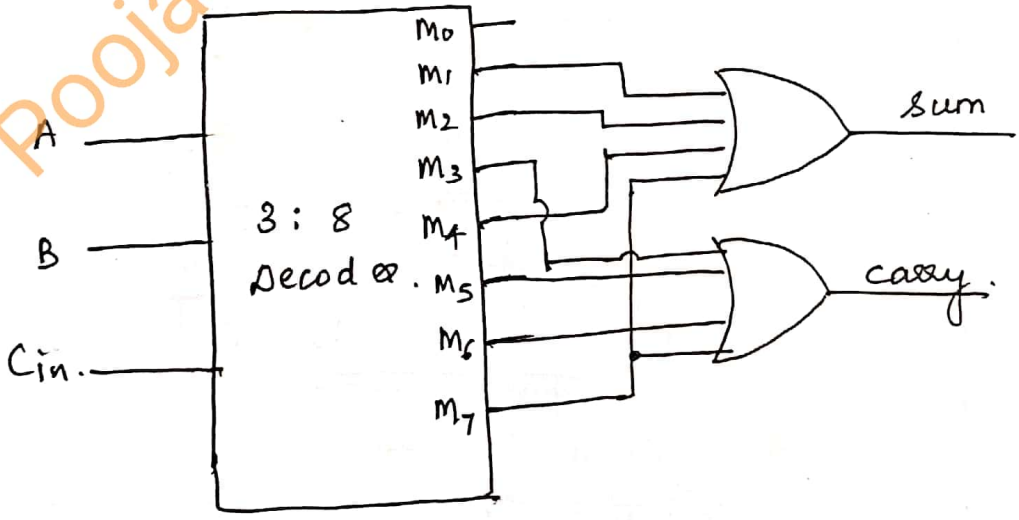
$sum = \sum m (1,2,4,7)$.

$carry = \bar{A}BC_{in} + A\bar{B}C_{in} + AB\bar{C}in + ABCin$

$= m_3 + m_5 + m_6 + m_7$

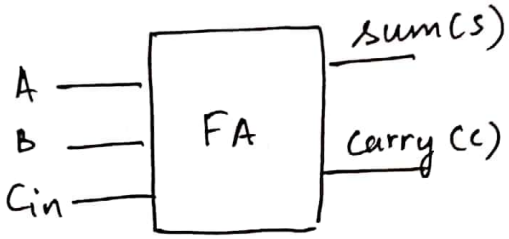
$carry = \sum m (3,5,6,7)$

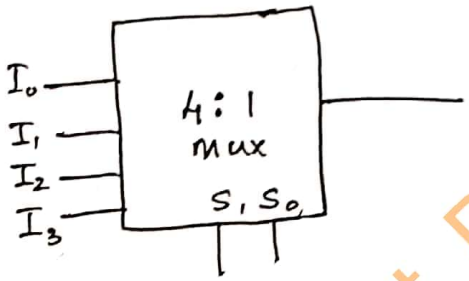
Hence the full adder can be implemented using 3:8 Decoder using OR gates as follows .



Implementation of full adder using 4:1 mux.

The block diagram of a full adder and 4:1 mux is as shown.





no. of i/p = 3

no. of o/p = 2

no of inputs = 4 = $[I_0, I_1, I_2, I_3]$

no. of select lines = 2 = $[S_1, S_0]$

∴ Truth table of 4:1 mux and ~~full adder~~ is given as,

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

∴ the output of the mux depends on the data in the select line.

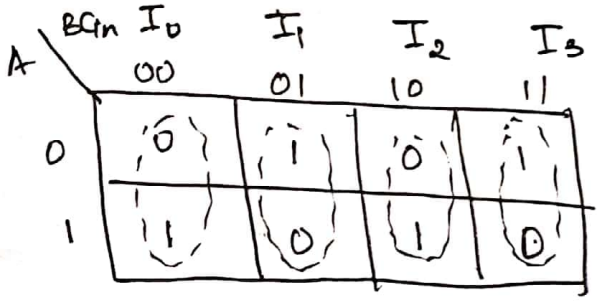
hence let B and C_{in} inputs of full adder be used as select lines.

then as per the truth table of 4×1 mux, the outputs as :- $I_0 \rightarrow BC_{in}=00$ $I_2 \rightarrow BC_{in}=10$

$I_1 \rightarrow BC_{in}=01$ $I_3 \rightarrow BC_{in}=11$.

The truth table of full adder is given by

A	B	C_{in}	sum	carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1



from K-map.

$I_0 = \bar{A}$ $I_3 = \bar{A}$

$I_2 = \bar{A}$ $I_4 = A$

for carry output,

A \ BCin	I_0	I_1	I_3	I_2
	00	01	11	10
0	0	0	1	0
1	0	1	1	1

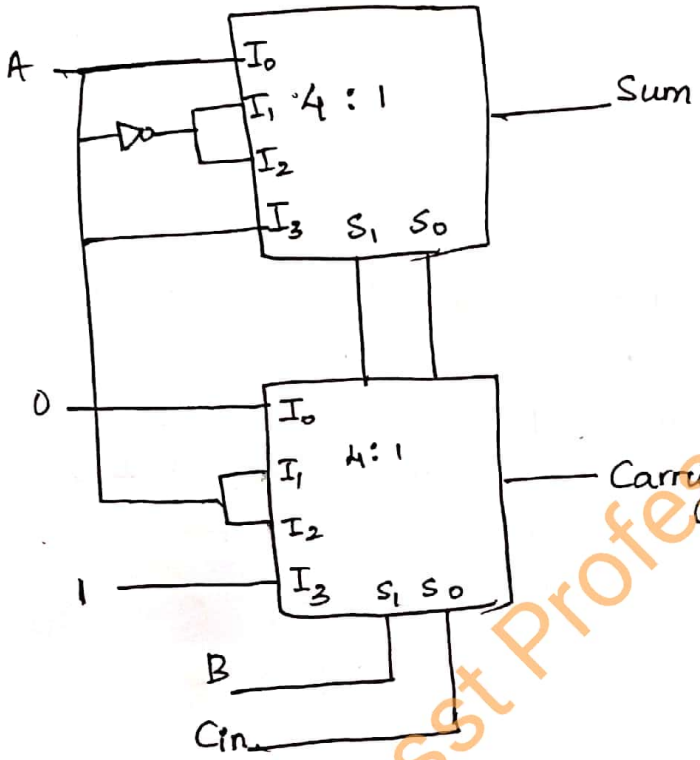
$I_0 = 0$

$I_1 = A$

$I_2 = A$

$I_3 = 1$

∴ the logical diagram for the full adder using 2 4:1 mux is



~~Flip flops :-~~

~~Digital data is stored in~~

Sequential circuits

The output of the combinational circuit depends not only on the present inputs but also on the past history of the system. These systems memorize the state in which it is in.

The circuit memorizes the state using basic memory elements called latches and flip flops.

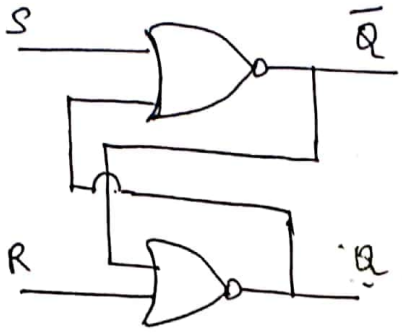
Latches :

A latch is a temporary storage device that has two stable states 0 and 1.
They are level sensitive storage element also called as bistable multivibrator.
Flip flop is a special kind of latch where the clock signal triggers the change in stored value

Different type
1) S-R Set-R type
2) D Delay (type)
3) J. K Type
4) T (toggle) type

1) S R latch
   → They are a pair of cross coupled NOR or NAND gates.
   → It consists of 2 inputs and 2 outputs. (S, R) (Q, $\bar{Q}$)
   → The device can be set to 0 or 1 by applying suitable values to S and R.

1) S-R latch using NOR gates



NOR gate

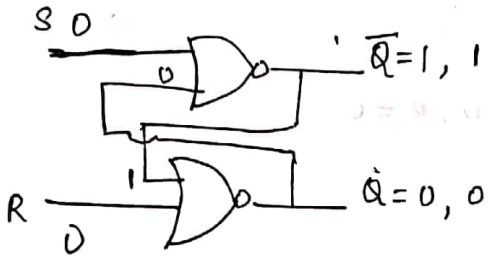
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

State Table of SR latch

S	R	Q	$\bar{Q}$
0	0	Q	$\bar{Q}$ – No change
0	1	0	1
1	0	1	0
1	1	invalid / not used	

Case 1 :- When S=0, R=0

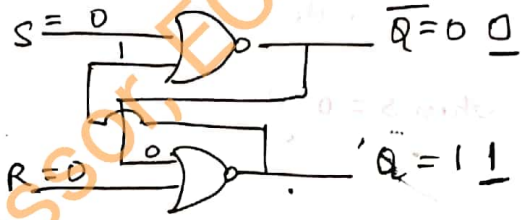
Let Q = 0, $\bar{Q}$ = 1



∴ output remains same as previous state.

When S=0, R=0

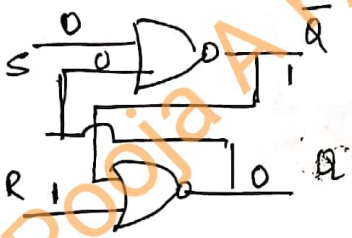
Let $\bar{Q}$ = 0, Q = 1



∴ Output remains same as previous state

hence the condition S=0, R=0 is known as MEMORY or NO CHANGE

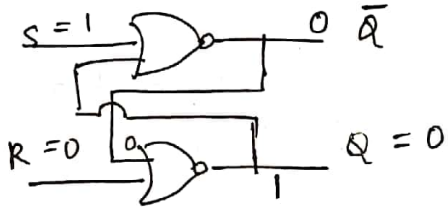
Case 2 :- When S = 0, R = 1



for NOR gate when one i/p is 1, then o/p automatically becomes 0 ∴ Q = 0

∴ $\bar{Q}$ = 1, Q = 0

Case 3 :- When S = 1, R = 0.

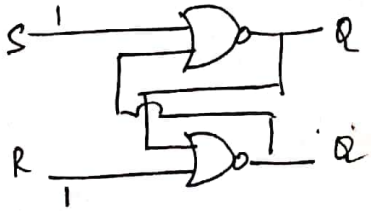


for NOR gate when one i/p is 1 then o/p = 0 ⇒ $\bar{Q}$ = 0

then from fig Q = 1

∴ o/p = $\bar{Q}$ = 0, Q = 1

Case 4 :- When S = 1, R = 1



for NOR gate when one or 2 i/p = 1

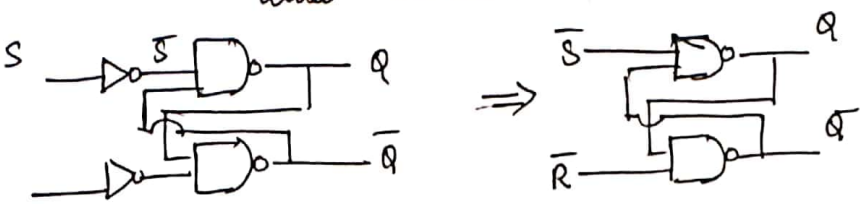
then o/p = 0 ∴ both Q, $\bar{Q}$ = 0

which is not possible

∴ S = 1, R = 1 → termed as invalid state

2) SR latch using NAND gate

also known as $\bar{S}\bar{R}$ latch.



State Table

$\bar{S}$	$\bar{R}$	Q	$\bar{Q}$	
0	0	not used		
0	1	1	0	
1	0	0	1	
1	1	Q	$\bar{Q}$	→ memory

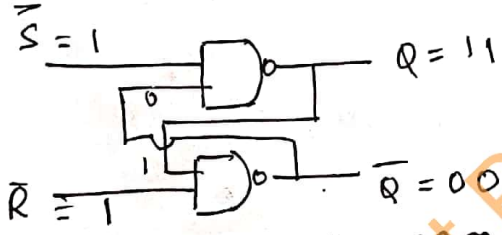
Truth table of NAND

A	B	y
0	0	1
0	1	1
1	0	1
1	1	0

Case 1 :-

$\Rightarrow \bar{S} = 1, \bar{R} = 1$;

let $Q = 1, \bar{Q} = 0$

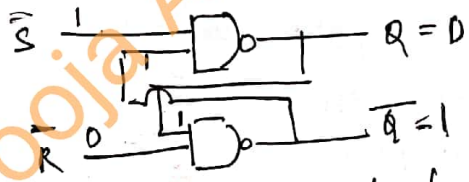


The output remains same.

The this state is known as NO CHANGE or MEMORY

case 2 :- when

$\Rightarrow \bar{S} = 1, \quad \bar{R} = 0.$



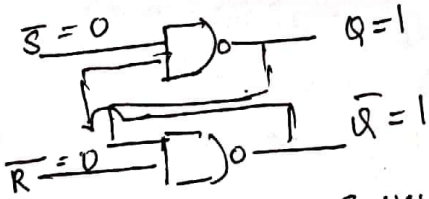
when one i/p = 0 o/p becomes = 1

$\therefore \bar{Q} = 1$ and $Q = 0$

— x —

case 4 :- when

$\Rightarrow \bar{S} = 0, \bar{R} = 0.$



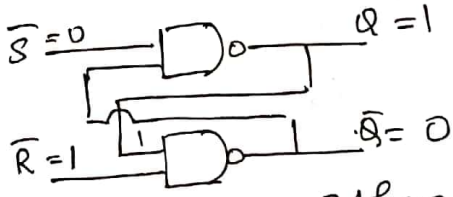
for NAND gate when i/p = 1 in both gates, then o/p's $Q = \bar{Q} = 1$ — not possible

hence it is known as invalid.

[UNPREDICTABLE BEHAVIOR HAPPENS IF INPUT RETURNS TO ONE SIMULTANEOUSLY]

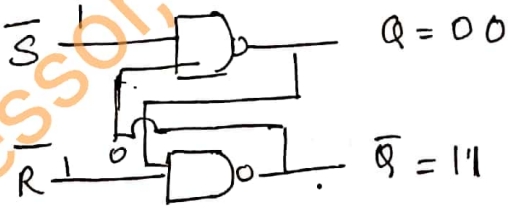
Case 3 :- when

$\Rightarrow \bar{S} = 0, \bar{R} = 1$



when one i/p = 0 then o/p becomes

$\bar{Q} = 0, \quad Q = 1$



$\therefore$ o/p remains same.

Implementation of X-OR gate using 4 NAND gates

X-OR gate truth table.

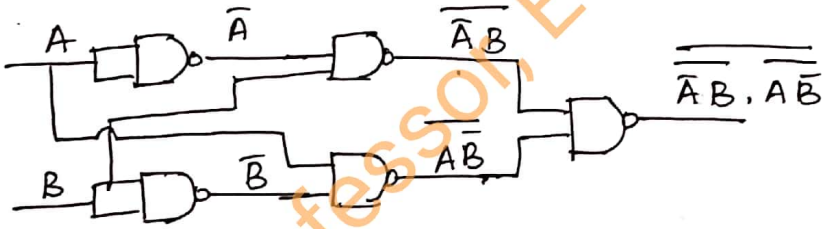
A	B	y
0	0	0
0	1	1
1	0	1
1	1	0

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

$y = A \oplus B$

using only NAND gates, (5 GATES)

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



using only 4 gates.

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

let X represent complement function.

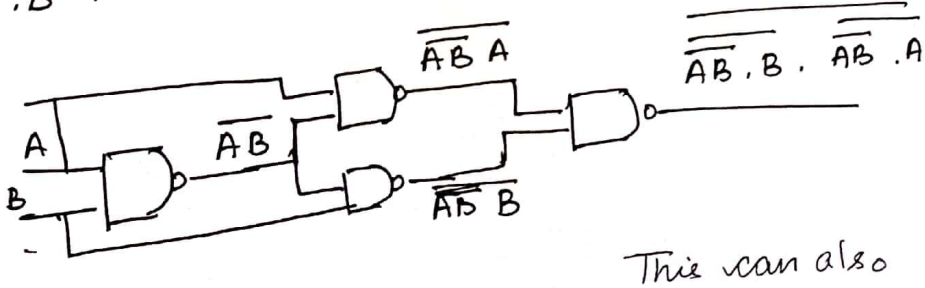
ie., $y = XB + AX$

$X = \bar{A}$ $\quad$ $X = \bar{B}$

Thus X generates $\bar{A}$ and $\bar{B}$ simultaneously. ie., 2 signal at same time

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

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



This can also be proved from o/p end refer full adder using NAND only page for steps