Unit - I

Number Systems

Number System :

In digital electronics, the number system is used for representing the information. The number system has different bases and the most common of them are the decimal, binary, Octal and Hexadecimal. The base or Radix of the number System is the total number of the digit used in the number system. Suppose if the number system representing the digit from 0-9 then the base of the System is 10.

Types of Number Systems :

Some of the important types of Number Systems are

- Decimal Number System.
- Binary Number System.
- Octal Number System.
- Hexadecimal Number System.

Decimal Number Systems : The number System is having digit 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, this number known as a decimal Number System because total ten digits are involved The base of the decimal Number System is 10.

Binary Number Systems : It is a positional weighted System. The base of binary number System is 2.

because it has only two digit 0 and 1. The digital electronic equipments are work on the binary number system & hence the decimal Number System is Converted into binary system.

Octal Number System : The Octal Number System is also positional weighted System. The Octal System has the base of eight as it uses digit 0, 1, 2, 3, 4, 5, 6, 7.

Hexadecimal Number : The hexddecimal Number System has a base of 16 & uses 16 symbols named 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. The symbols A, B, C, D, E, F represent the 10, 11, 12, 13, 14, and 15 respectively. The least significant position has a weight of 16^0 i.e 1 the higher significant position are given weights in the ascending powers of i.e. 16^1, 16^2, 16^3 ... 16^n.

Conversion fr

I[st] Decimal

(53.625)

2 | 53
2 | 2
2 | 1
2 |
2 |

(821)

The integer

(53)$_{10}$

0.625

0.25

0.50

0.00

Multiplica

Conversion from One base to another

I[a] Decimal to Binary

$(53.625)_{10} - (\quad)_2$

2 | 53		LSB
2 | 26	→ 1	
2 | 13	→ 0	
2 | 6	→ 1	
2 | 3	→ 0	
1	→ 1	
MSB		

The integer number is 53. The fraction is 0.625

$(53)_{10} - (110101)_2$

∴ If the D.N is a fraction its binary equivalent is obtained by multiplying the Number continuously by 2,

$0.625 \times 2 = 1.25 \rightarrow 1$ MSB

$0.25 \times 2 = 0.50 \rightarrow 0$

$0.500 \times 2 = 1.00 \rightarrow 1$

$0.00 \times 2 = 0.00 - 0$

Multiplication by 2 is not Possible since the product is 0.

$(0.625)_{10} - (101)_2$

$(53.625)_{10} - (110101.101)_2$

(b) Decimal to Octal

$(444.96)_{10}$ — $(674.75341)_8$

$(444)_{10}$ - $(674)_8$

8	444	
8	55	→ 4 ↑
8	6	→ 7 ↑

Fractional part

$0.96 \times 8 = 7.68 \leftarrow 7$ MSB

$0.68 \times 8 = 5.44 \leftarrow 5$

$0.44 \times 8 = 3.52 \leftarrow 3$

$0.52 \times 8 = 4.16 \leftarrow 4$

$0.16 \times 8 = 1.28 \leftarrow 1$ (LSB)

This process will continue further, so may take the result upto 5 places

$(0.96)_{10}$ - $(75341)_8$

(c) Decimal to Hexadecimal

$(2586)_{10}$ — $(A1A)_{16}$

16	2586	
16	161	- 10-A ↑
	10	- 1
	A	

$(7864)_{10}$ — $(EB8)_{16}$

16	7864	
16	491	- 8
16	30	- 11-B
	1	- 14-E

2^n - - - -

II (a) Binary

$(101.01)_2$

$1 \times 2^2 +$

$4 +$

(b) Binar

The binar number

(11

(c) Bi

The bin number

(11

2^n ..... 2^4 2^3 2^2 2^1 2^0
16 8 4 2 1

II (a) Binary to Decimal

$(101.01)_2 = (5.25)_{10}$ - A Binary number can be Converted into decimal Number by Multiplying the binary number 1 or 0 by their weight

$1\times2^2 + 0\times2^1 + 1\times2^0 + 0\times2^{-1} + 1\times2^{-2}$

$4 + 0 + 1 + 0 + 0.25$

(b) Binary to Octal :

The binary Number can be Converted in to octal number by making group of 3 bits.

$(110101.101010)_2 = (65.52)_8$

4 2 1 | 4 2 1 . 4 2 1 | 4 2 1
1 1 0 | 1 0 1 . 1 0 1 | 0 1 0
↓ ↓ ↓ ↓
6 5 5 2

(c) Binary to Hexadecimal

The binary number can be Converted into hexadecimal number by making group of 4 bits

$(1101\ 0010\ 1011\ 0101)_2 = (34B75)_{16}$

8 4 2 1 | 8 4 2 1 | 8 4 2 1 | 8 4 2 1 | 8 4 2 1
0 0 1 1 | 0 1 0 0 | 1 0 1 1 | 0 1 1 1 | 0 1 0 1
↓ ↓ ↓ ↓ ↓
3 4 B 7 5

III (a) Octal to Binary

For obtaining the binary equivalent of an Octal number each significant digit in the given number is replaced by 3-bit binary equivalent.

$(376)_8$ - 3 7 6
011 111 110

$(376)_8 - (011111110)_2$

(b) Octal to Decimal.

$(237)_8 - (\quad 159 \quad)$

$2\times8^2 + 3\times8^1 + 7\times8^0$

$2\times64 + 3\times8 + 7 = 128 + 24 + 7$

$= 159$

(c) Hexadecimal to Octal $-(46)_{16} - (\quad)_8$

Octal Hexadecimal

Convert the Hexadecimal in to Binary

$(46)_{16} - (100110)_2$

Form group of 3-bits - 010 011 110

$= (176)_8$

8 4 2 1 8 4 2 1
0 1 0 0 1 1 1 0

7

8 4 2 1 8 4 2 1
0 0 1 0 0 1 1 0

2 6

IV (a) Hex

$(2AC9)_{16}$

$2\times16^3 +$

(b) Hexad

For obtai
Number
is the

$(2A5)_{16}$

(2A

(c) He

Conver

$(47)_{16}$ -

Form

$(345)_8$

$(345)_8 =$

IV (a) Hexadecimal to Decimal

$(2AC9)_{16} = (10,953)_{10}$

$2\times16^3 + 10\times16^2 + 12\times16^1 + 9\times16^0$

$= 10,953$

(b) Hexadecimal to Binary

For obtaining the binary equivalent of hexadecimal Number each significant digit in the given number is replaced by 4-bit digit Number

$(2A5)_{16}$ = 2 A 5

0010 1101 0101

$(2A5)_{16} = (001011010101)_2$

(c) Hexadecimal to Octal - $(47)_{16} = (107)_8$

Convert the hexadecimal in to Binary

$(47)_{16} = (01000111)_2$

Form group of 3-bits 001 000 111

1 0 7

8 | 46

5 | 6

56

$4\times8^1 + 6\times8^0$

$32+6$ 38

$2\times8^1 + 6\times8^0$

$16+6$ 22

8 | 46

Binary Coded Decimal (BCD) :

The BCD uses the binary number system to specify the decimal numbers 0 to 9. It has four bits. The weights are assigned according to the position occupied by these digits. The weight of the 1st position is 2^0(1), the second (2)' 2, the third $(2)^2$ 4, and the fourth $(2)^3$ 8. Reading from left to Right the weights are 8-4-2-1 & hence it is called 8421 code.

The binary equivalent of 7 is $[111]_2$, but the same number is represented in 4-bit form as 0111. Also the numbers from 0 to 9 are represented in the same way as the binary system, but after 9 the representation in BCD are different. For ex

12 in the binary system is $[1100]_2$ but the same number is represented as [0001 0010] in BCD

Ex- Give the BCD code for the decimal Number 874

Decimal Number - 874
BCD code - 1000 0111 0100
$(874)_{10}$ - (1000 01110100)BCD

Excess - 3 C

The excess - 3
form, as a
Code is
Number.

[64

Decimal
add 3 to
Sum

Converting

Excess -

Gray Cod

The Gr
minimu
in th
One
non-We
digit
this
only

Excess - 3 Code :

The excess - 3 represents a decimal number, in binary form, as a number greater than 3. An excess - 3 Code is obtained by adding 3 to a decimal Number.

$[643]_{10}$ in to excess - 3 code

Decimal number	6	4	3
Add 3 to each bit	+3	+3	+3
Sum	9	7	6

Converting the above Sum in to BCD code

Sum → 9 7 6

1001 0111 0110

Excess - 3 code for $[643]_{10}$ - 100101110110

Gray Codes :

The Gray Codes belongs to a class of codes called minimum - change codes, in which only one bit in the Code group changes when moving from One Step to the next. The Gray Code is a non-weighted code. The Gray Code is a reflective digit Code which has a special property of that adjacent Code numbers that differ by only 1-bit. Therefore, it is also called a Unit - distance Code.

Ex:- $[10110]_2$ to Gray Code

Step 1 The first bit MSB of the Gray code is the same as the first bit of the binary no.

1 0 1 1 0 Binary

↓

1 Gray

Step 2

Add the 1st bit of the binary to 2nd bit of the binary digit no.

Add 1+0 = 1. The result is the 2nd bit of the Gray

1 ⊕ 0 1 1 0 Binary

1 1 Gray

Step 3

Add 2nd bit 0 to the third bit 1 of the binary

1 0 ⊕ 1 1 0

↓

Step 4 1 1 1

Add 3rd bit 1 to the 4th bit 1 & omit the Carry

1 0 1 ⊕ 1 0

↓

Step 5 1 1 1 0

Add 4th bit to the last bit

1 0 1 1 ⊕ 0

↓

1 1 1 0 1

$(10110)_2 \rightarrow (11101)_{Gray}$

$(10110111)_2$

$(11111101)_{Gray}$

Conversion from G

1. The first binary the first Gray

2. If the second bit is the Sa

If the second bit is the

3. Step 2 c

$(110101)_{Gr}$

write

Step-1 1 1

↓

1

Step. 2 The

Therefore the 2

inverse of the

↓

1

Step 3 The

the third

that of the

$(10101101)_2 - (\quad)_G$

$(11111011)_{10}$

Conversion from gray Code to Binary

1. The first binary bit (MSB) is same as that of the first Gray Code bit.

2. If the second Gray bit is 0, the second binary bit is the same as that of the first binary.

If the second gray bit is 1, the second binary bit is the inverse of its first binary bit.

3. Step 2 is repeated for each successive bit.

$(110101)_{Gray} - (\quad)_2$

Step-1 write the first binary bit 1 which is the MSB of the Gray code

1 1 0 1 0 1 — Gray

↓

1 — Binary

Step. 2 The second bit of Gray Code is 1 therefore the 2nd bit of the binary is 0. inverse of the 1st binary bit '1'

1 1 0 1 0 1 Gray

↓ ↓

1 0 Binary

Step 3 The 3rd of the Gray code is 0, therefore the third bit of the binary is same as that of the second binary bit 0

1 1 0 1 0 1

1 0 → 0

Step 4 1 1 0 1 0 1
↓
1 0 0 → 1

Step 5 1 1 0 1 0 1
↓
1 0 0 1 → 1

Step 6 1 1 0 1 1 0 1

1 0 0 1 . 1 0

$(110101)_G = (1.00110)_2$

ASCII Code : A standurised code that has been widely accepted by the industry, the ASCII (pronounced "ass-kee") code - American Standards code for information Interchange, used in most Computers by its manufactuies. The ASCII Code represents a character with seven bits, which can be stored as one byte with one bit unused. The format of ASCII Code is $X_6 X_5 X_4 X_3 X_2 X_1 X_0$ where each bit is a 0 or a '1'

BCDIC
EBCDIC Code : The full form of EBCDIC code is Extended Binary Coded Decimal Information Code. It differs from ASCII only in its code grouping for the different alphanumeric characters. It uses eight bits for each character & a nineth bit for parity

Arithmetic Op

Binary Arith

The arithmet
Multiplication
below
Addition

0 + 0 = 0
0 + 1 = 0
1 + 0 = 0
1 + 1 = 10

Binary Additi

Binary Subtr

Binary Mul

Arithmetic Operation

Binary Arithmetic

The arithmetic rules for Addition, Subtraction, Multiplication and Division of Binary numbers are given below

Addition	Subtraction	Multiplication	Division
0 + 0 = 0	0 - 0 = 0	0 x 0 = 0	0 x 1 = 0
0 + 1 = 0	1 - 0 = 1	0 x 1 = 0	1 x 1 = 1
1 + 0 = 0	1 - 1 = 0	1 x 0 = 0	0 x 0 = Not allowed
1 + 1 = 10	10 - 1 = 1	1 x 1 = 1	1 x 0 = Not allowed

Binary Addition

```
  1 1 1 1          15
  1 0 1 0          10
-------------    ----
1 1 0 0 1        25
```

Binary Subtraction

```
  1 1 0 1          13
- 1 0 0 1        -  9
-------------    ----
  0 1 0 0          4
```

Binary Multiplication 1011 x 1101

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

Binary Division $1001 \div 101$

$$
\begin{array}{r|l}
 & 101 \\
\hline
101 & 1001 \\
 & 101 \\
\hline
 & 0101 \\
 & 101 \\
\hline
 & 0000
\end{array}
$$

1's Complement

Subtraction of a number from another can be accomplished by adding the Complement of the Subtrahend to the Minuend

Ex Subtract $(1010)_2$ from $(1111)_2$ using the 1's Complement Method

$$
\begin{array}{rl}
 & 1111 \\
 & 0101 \leftarrow \text{1's Complement} \\
\text{Carry} & 1\ 0100 \\
\text{Add Carry} & +\ 1 \\
\hline
 & 0101
\end{array}
$$

Direct Subtraction

$$
\begin{array}{r}
1111 \\
1010 \\
\hline
0101
\end{array}
$$

2's Complement

Subtract (1

Complement

1010

1's Compleme

2's Complement

1111

+ 0111

Carry → 0 1 0

Direct Subt

9's Compl

Find the

(a) 19

(a) 99

− 19

80

(b) 99

14

85

2's Complement

Subtract (1010) from $(1111)_2$ using 2's Complement Method

$$\begin{array}{r} 1010 \\ \text{1's Complement} \quad 0101 \end{array}$$

$$\begin{array}{r} 0101 \\ +\ 1 \\ \hline 0110 \end{array}$$

2's Complement

$$\begin{array}{r} 1111 \\ +\ 0110 \\ \hline 0101 \end{array}$$

Carry → 1

The Carry is discarded. Thus the answer is $(0101)_2$

Direct Subtraction

$$\begin{array}{r} 1111 \\ 1010 \\ \hline 0101 \end{array}$$

9's Complement

Find the 9's Complement

(a) 19 (b) 146 (c) 469

(a) $\begin{array}{r} 99 \\ -19 \\ \hline 80 \end{array}$ → 9's Complement of 19

(b) $\begin{array}{r} 999 \\ 146 \\ \hline 853 \end{array}$ → 9's Complement 146

Ex Subtract 18-06 using 9's Complement

Regular Subtraction

```
  18
- 06
----
  12
```

```
  99
- 06
----
  93  ← 9's Complement of 06
```

```
  1
  18
+ 93  ← 9's Complement of 6
-----
 (1)11
    +1   Add Carry to Result
-----
  12
```

10's Complement

The 10's Complement of a decimal Number is equal to its 9's Complement +1.

(a) 9

```
  9
- 9
---
  0  ← 9's Complement of 9
+ 1
---
  1  ← 10's Complement of 9.
```

(b) 46

```
  99
- 46
----
  53  ← 9's Complement of 46
+ 1
----
  54  ← 10's Complement 46
```

Subtract

9 - 4

```
  9
- 4
---
  5
```

Logic G

A Logic g
makes lo

OR - Gate

Truth Table

Input	
A	B
0	0
0	1
1	0
1	1

AND - G

I/P	
A	B
0	0
0	1
1	0
1	1

Subtract 9-4 using 10's Complement

9 - 4

$$\begin{array}{r} 9 \\ -\,4 \\ \hline 5 \end{array} \qquad \begin{array}{r} 9 \\ +\,6 \\ \hline (1)\ 5 \end{array} \leftarrow \text{10's Complement of 4.}$$

Drop Carry

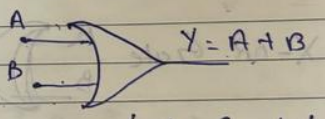
Logic Gates

A Logic gate is an electronic circuit which makes logical decision

OR - Gate : The OR-gate performs logical addition

Truth Table

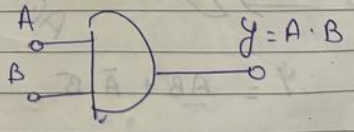
Input		O/P Y = A + B
A	B	
0	0	0
0	1	1
1	0	1
1	1	1

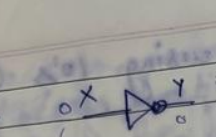


Logic Symbol

AND - Gate : The AND-gate performs logical Multiplication.

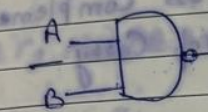
I/P		O/P
A	B	Y = A·B
0	0	0
0	1	0
1	0	0
1	1	1



NOT Gates – 

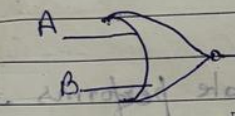
$Y = A'$

i/p	o/p
A	Y
0	1
1	0

NAND Gate 

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

i/p		o/p
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

NOR Gate 

$Y = \overline{A + B}$

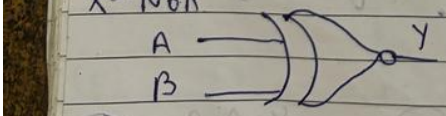
i/p		o/p
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

X-OR Gate 

$A \oplus B$

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

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

X-NOR 

$A \odot B$

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

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

Basics Laws

1. Commutative

2. Associativ

3. Distributi

4. Absorption

AND law

OR-law –

Basics Laws of Boolean Algebra

1. Commutative property - $A+B = B+A$
   $A \cdot B = B \cdot A$

2. Associative property - $A+(B+C) = (A+B)+C$
   $A \cdot (B \cdot C) = (A \cdot B) \cdot C$

3. Distributive property : $A+BC = (A+B)(A+C)$
   $A \cdot (B+C) = A \cdot B + A \cdot C$

4. Absorptions Law = $A + AB = A$
   $A \cdot (A+B) = A$
   $A + \bar{A}B = A + B$
   $A(\bar{A}+B) = AB$

AND Law
$0 \cdot 0 = 0$
$0 \cdot 1 = 0$
$1 \cdot 0 = 1$
$1 \cdot 1 = 1$

OR-Law -
$0+0 = 0$
$0+1 = 1$
$1+0 = 1$
$1+1 = 1$

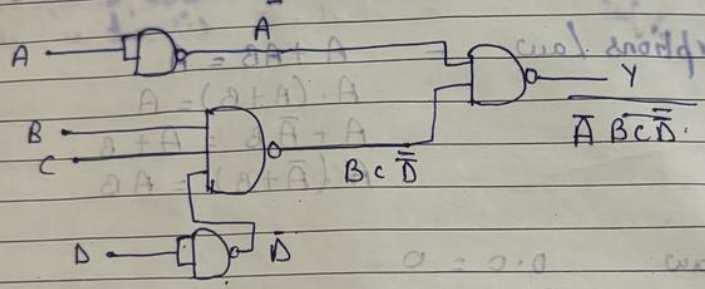
$B + \bar{B} = 1$
$A + \bar{A} = 1$
$A\bar{A} = 0$
$A + \bar{A}B = A + B$
$A \cdot B \neq A$
$A \cdot 0 = 0$
$A \cdot A = A$
$A + 1 = 1$

Implementation of Logic functions using gates, NAND - NOR

Ex. $Y = A + BC\bar{D}$ using NAND Gate &

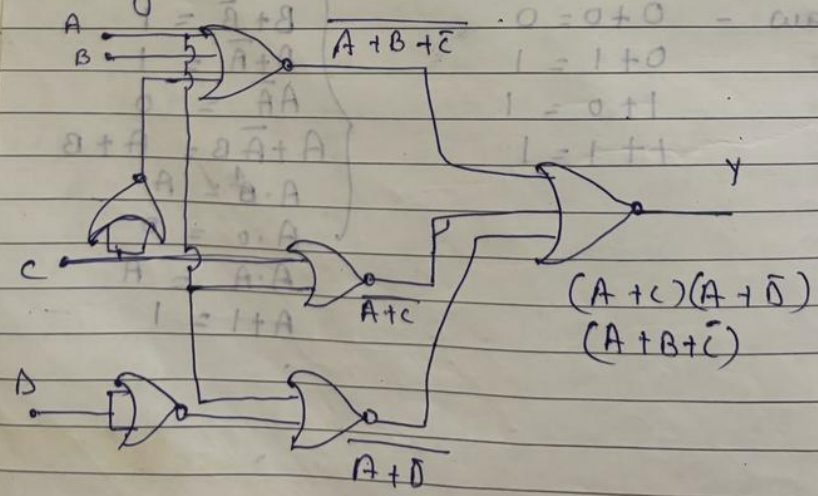
$Y = (A+C)(A+\bar{D})(A+B+\bar{C})$ using NOR gates

(a) Using NAND Gates

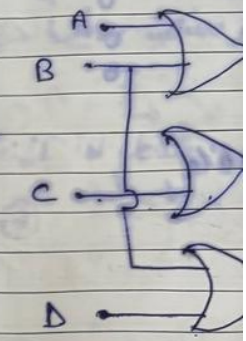


$= A + BC\bar{D}$

(b) Using NOR Gates



Ex- Realise the
using basic ga

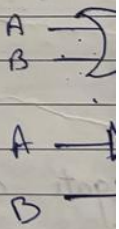


De-Morgans

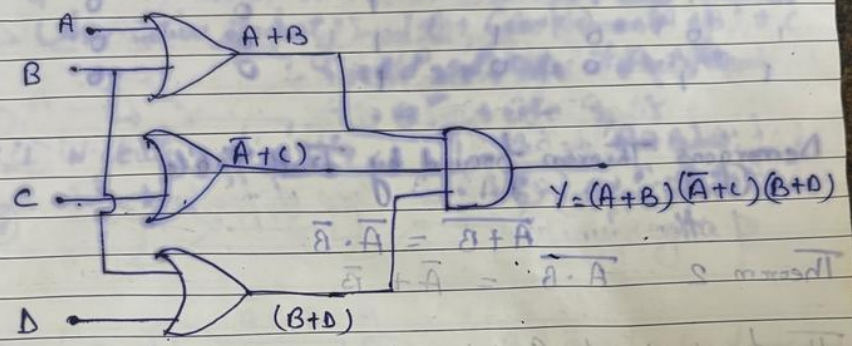
Theorem 1

The Complem
Complement

Prof



Ex→ Realise the logic expression $Y=(A+B)(\bar{A}+c)(B+D)$ using basic gates.

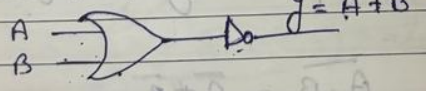


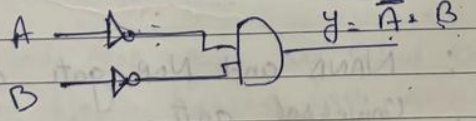
De-Morgans Theorem :

Theorem I $\quad \overline{A+B} = \overline{A}\cdot\overline{B}$

The Complement of Sum equal to the product of Complement

Proof





Truth Table

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

OR

Demorgans Theorem proved by Truth Table

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

Theorem 2 $\overline{A\cdot B} = \bar{A}+\bar{B}$

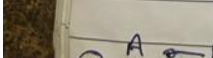
The product of Complement equal to sum of the Complement.

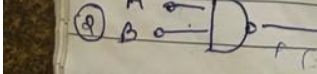
A	B	$\bar{A}$	$\bar{B}$	$A\cdot B$	$\overline{A\cdot B}$	$\bar{A}+\bar{B}$
0	0	1	1	0	1	1
0	1	1	0	0	1	1
1	0	0	1	0	1	1
1	1	0	0	1	0	0

proof by Truth table $\overline{A\cdot B} = \bar{A}+\bar{B}$

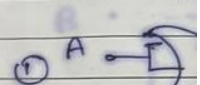
Universal Gate : NAND and NOR gate are Universal gate.

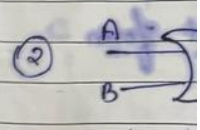
NAND as a Universal gate logic ckt can be build by NAND gate

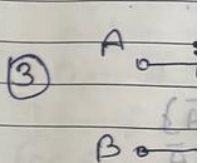
①  $y = \bar{A}$ — NOT Gate

②  $y = \overline{A\cdot B} = A\cdot B$. AND gate

NOR Ga

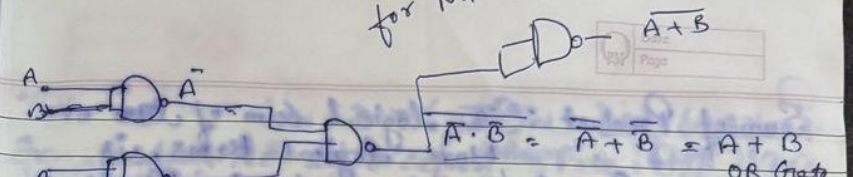
① 

② 

③ 

NAND and
building b
any gate
Combination
by NAND
gate

for NOR Gate



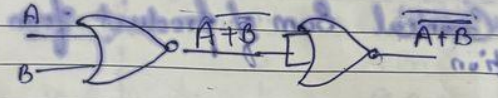
by Demorgans Theorem

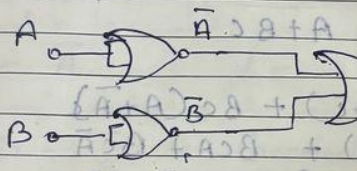
$\overline{A \cdot B} = \overline{A} + \overline{B}$

NOR Gate as a Universal gate

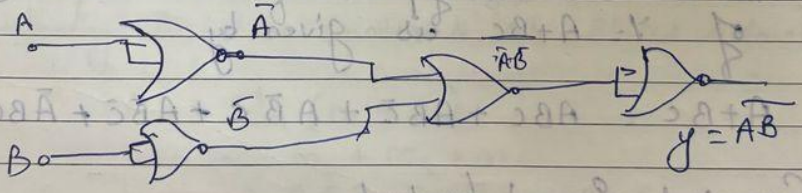
① 

— NOT gate

② 

③ 

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



NAND and NOR gates are called Universal gates or Universal building blocks because both can be used implement any gate like AND, OR, and NOT gates or any combination of these basic gates

~~For~~ a NAND gate can be used to realise various logic gate

Sum of Product : The logical sum of two or more logical terms, is called a Sum of products expression. It is basically an OR operation of AND operated Variables such as :

$Y = AB + BC + AC$

$Y = AB + \bar{A}c + BC$

Ex- Obtain the Canonical Sum of product form of the function

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

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

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

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

$\{\because ABC + ABC = ABC\}$

∴ the Canonical Sum of product form of $Y = A + BC$ is given by

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

Canonical Sum of product

It is defined as the logical sum of all the minterms derived from the rows of a truth table, for which the value of the function is 1. It is also called a Minterm & Canonical form.

The minterm
by no. m_1
M 2
A B
0 0
0 0
0 1
0 1
1 0
1 0
1 1
0 ÷ $\bar{A}$

For Ex. If
of a 3- Va
minterms
be expres
Corresponding

The minterm of a 3-Variables func^n can be represented by $m_0, m_1, m_2, m_3, m_4, m_5, m_6, m_7$

A	B	C	Minterm	
0	0	0	$\bar{A}\bar{B}\bar{C}$	m_0
0	0	1	$A\bar{B}C$	m_1
0	1	0	$\bar{A}B\bar{C}$	m_2
0	1	1	$\bar{A}BC$	m_3
1	0	0	$A\bar{B}\bar{C}$	m_4
1	0	1	$A\bar{B}C$	m_5
1	1	0	$AB\bar{C}$	m_6

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

For Ex - If the Canonical Sum of product form of a 3-Variables logic function Y has three minterms $\bar{A}\bar{B}\bar{C}$, $A\bar{B}\bar{C}$, $AB\bar{C}$, this can be expressed as the Sum of the decimal Code Corresponding to these minterms as Stated below:

$$Y = \sum m (0, 5, 6)$$
$$= m_0 + m_5 + m_6$$
$$= \bar{A}\bar{B}\bar{C} + A\bar{B}C + AB\bar{C}$$

Product of Sums (POS): A product of Sums expression is a logical product of two or more logical sum terms. It is basically an AND operation of OR operated variables

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

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

Canonical product of Sum expression

This is defined as the logical product of all the maxterms derived from the rows of a truth table for which the value of function is 0. It is also known as the Maxterm Canonical form.

For Ex: if the Canonical product of Sum of a 3-variable logic function Y, has four Maxterms $(A+B+C)$, $(A+\bar{B}+C)$, $(\bar{A}+B+C)$ and $(\bar{A}+\bar{B}+\bar{C})$, then it can be expressed as the product of decimal Codes as given below:

$Y = \prod(0,2,4,7)$

$M_0 . M_2 . M_4 . M_7$

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

A
0
0
0
0
1
1
1
1

0 - A

Ex → Obtai
Y(ABC) =

Sol → In th
ein the first
the Variable

Y = CAB

by using

Y = (A+B̄
(A+B

Y = (A+B̄
(A

	A	B	C		
	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	0	0	$\bar{A}+B+C$	M_4
	1	0	1	$\bar{A}+B+\bar{C}$	M_5
	1	1	0	$\bar{A}+\bar{B}+C$	M_6
	1	1	1	$\bar{A}+\bar{B}+\bar{C}$	M_7

$0 - A \qquad 1 - \bar{A}$

Ex → Obtain the Canonical product of Sum expression

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

Sol → In the given expression, the variable C is missing in the first term, the variable A in the second term, the variable B in the third term. ∴ $C\bar{C}$, $A\bar{A}$ & $B\bar{B}$

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

$(A+\bar{B}+0)(B+C+0)(A+\bar{C}+0)$

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

by using distributive property.

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

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

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

K-Map (Karnaugh Map)

A K-Map is a representation of a truth table of a logical function

A K map can be used to obtain a "minimal" expression of a function in the SOP or in the POS

For Ex. 2 Variable K-Map

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

C \ AB	0	1	0	X
0	0	1	0	X
1	1	0	1	0

A K-map is the same as the truth table.

Consider the following function in (a) SOP form (b) POS form

$F(A,B,C,D) = \Sigma_m (0,1,2,5,8,9,10)$

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

CD \ AB

00

01

11

10

Y =

Pos Form

(Ā

Don't Care

In Certain

never occu

because th

never to

don't ca

Simplify

F(A,B,C

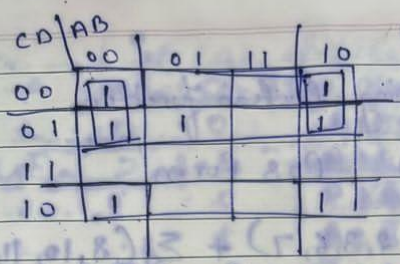
CD

00

01

11

10



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

Pos Form

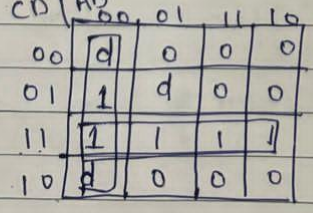
$(\bar{A}+\bar{B})(\bar{C}+\bar{D})(\bar{B}+D)$

Don't Care Combinations :

In Certain digital Systems, Some input Combinations never occur during the process of a normal operation because those input Conditions are geranted never to occur. Such input Combinations are don't care Combinations

Simplify the Boolean function

$F(A,B,C,D) = \Sigma_m(1,3,7,11,15) + \Sigma_d(0,2,5$



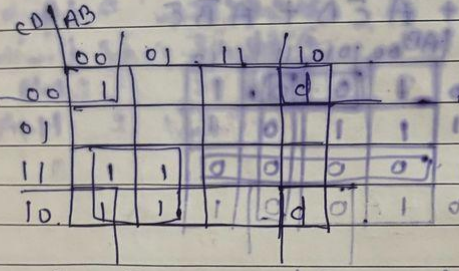
$Y = \bar{A}\bar{B} + CD$

Simplify the following Boolean function.

(a) SOP (b) POS form

$Y = \Sigma_m (0, 2, 3, 6, 7) + \Sigma_d (8, 10, 11, 15)$

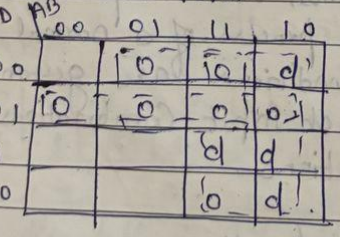
(a)



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

By Combining the 1's & d as shown in the K-map there are two quads,

(b) Minimal POS form



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

Quine –

Find the mi
expression

F(a, b, c,

0	0 0
5	0 1
8	1 0
9	1
16	1 1
11	1
14	1
15	1

Step 1

Gro

Number of
0
1
2

Quine – McCluskey Method :

Find the minimal sum of products for the Boolean expression

$F(a, b, c, d) = \Sigma(0, 5, 8, 9, 10, 11, 14, 15)$

0	0 0 0 0	
5	0 1 0 1	
8	1 0 0 0	~ 11,P
9	1 0 0 1	~ 11,01
10	1 0 1 0	~ 11,01
11	1 0 1 1	
14	1 1 1 0	
15	1 1 1 1	← Binary representation of minterms

Step 1

Group Number of 1's	Minterm	Variable A B C D
0	0 ✓	0 0 0 0
1	8 ✓	1 0 0 0
2	5	0 1 0 1
	9 ✓	1 0 0 1
	10 ✓	1 0 1 0
3	11 ✓	1 0 1 1
	14 ✓	1 1 1 0
4	15	1 1 1 1

Group of minterms for different number of 1's

Step 2 — 2-Cell Combination

Group	Matched Pair	Variables A B C D
0	0, 8	- 0 0 0
1	(8, 9) ✓	1 0 0 -
	8, 10 ✓	1 0 - 0
2	9, 11 ✓	1 0 0 - 1
	10, 11 ✓	1 1 0 0 1 -
	10, 14 ✓	1 - 1 1 0
3	11, 15 ✓	1 - 1 1 1
	14, 15 ✓	1 1 1 -

Step 3 : 4-Cell Combination

Group	Matched Pair	Variables A B C D	
0	8, 9, 10, 11	1 0 - -	$A\bar{B}$
	8, 10, 9, 11	1 0 - -	
1	10, 11, 14, 15	1 - 1 -	$A\bar{C}$
	10, 14, 11, 15	1 - 1 -	
	0, 8	- 0 0 0 -	$\bar{B}\bar{C}\bar{D}$
	5	0 1 0 1	

Step 4

P. I

8, 9, 10, 11

~~8, 10, 9, 11~~

10, 11, 14, 15

0, 8

5

~~Step 5~~

f(a, b, c, d) =

A B

Step 4 Prime Implicants Table

P.I	0	5	8	9	10	11	14	15
8, 9, 10, 11			X	(X)	X	X		
~~8, 10, 9, 11~~								
10, 11, 14, 15					X	X	(X)	(X)
0, 8	(X)		X					
5	✓	(X)		✓		✓	✓	✓

~~Step 5~~ Reduced Prime ~~Implicant Table~~

	0	5	8	9	~~10~~	~~11~~	~~14~~	~~15~~
0, 8	X		X					
5		X						

$f(a,b,c,d) =$

$$AB' + Ac + B'c'D' + A'Bc'D$$

$A\bar{B}$

Ac

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

$F(A,B,C,D) = \Sigma(4,6,9,10,11,13) + \Sigma d(2,12,15)$

Step - 1

Step 2

4	0	1	0	0
2	0	0	1	0
6	0	1	1	0
9	1	0	0	1
10	1	0	1	0
12	1	1	0	0
11	1	0	1	1
13	1	1	0	1
15	1	1	1	1

Step - 3

4, 6	0	1	-	0	P
4, 12	-	1	0	0	Q
2, 6	0	-	1	0	R
2, 10	-	0	1	0	S
9, 11	1	0	-	1	✓
9, 13	1	-	0	1	✓
10, 11	1	0	1	-	T
12, 13	1	1	0	-	U
11, 15	1	-	1	1	✓
13, 15	1	1	-	1	✓

Step - 3 4 -

9, 13, 11, 15

9, 11, 13, 15

Prime implic

P	0	1	
Q	-	1	
R	-	1	
S	-	0	
T	1	0	
U	1	1	
V		1	

	4
*P	X
Q	X
R	
S	
T*	
U	
*V	

(4, 6, 10,

P, S, T

f =

2, 12, 15)

Step - 3 4-Cell Combinations

	A	B	C	D	
9, 13, 11, 15	1	-	-	1	} - ✓
9, 11, 13, 15	1	-	-	1	

Prime implicants are

P	0	1	-	0	$A'BD'$
Q	-	1	0	0	$BC'D'$
R	-	1	0	0	$A'CD'$
S	-	0	1	0	$B'CD'$
T	1	0	1	-	$AB'C$
U	1	1	0	-	ABC'
V	1	-	-	1	AD

	4	6	9	10	11	13
*P	X	X				
Q	X					
R		X		X		
S				X		
T*				X	X	
U						X
V*	✓		(X)		X	X
	✓	✓	✓		✓	✓

(4, 6, 10) Unticked Columns

P, S, T

$F = P + T + V$

$= \bar{A}BD' + AB'C + AD$