Page No.
Date: / /

SYLLABUS - UNIT 1

- Logic Families : CMOS Logic, CMOS Dynamic Electrical behaviour
Bipolar Logic : Diode Logic, Transistor Logic Inverter, TTL Logic, NMOS, CMOS/TTL Interface, ECL
- Minimization Techniques and Logic gates :
[Minimization Techniques]: [Boolean postulates and Laws] - [De Morgan Theorem] - Principle of Duality - Boolean expressions - [Minimization of Boolean Expressions] - [Minterms, Maxterms] [Sum of Products (SOP)], [Product of SUMs (POS)], Karnaugh map minimization. Don't care conditions -, Quine, Mc Cluskey method of minimization, [Number System : Representation of -ve numbers] & [1's Complement], [10's Complement], [Arithmetic Using 2's Complement]

- CONVERSIONS :

- Decimal to Binary - divide the num successively by 2 and record the remainder until 0 as quotient. Reverse the remainders.

for fractional part : multiply successively the fractional part by 2, Integral part each time 'd be digits of binary.

ex - 4.47

2	4	0
2	2	0
	1	

$0.47 \times 2 = 0 + 0.94$ 0
$0.94 \times 2 = 1 + 0.88$ 1
$0.88 \times 2 = 1 + 0.75$ 1

$\Rightarrow$ 100.011

- **Binary to Decimal**:

ex- 100.011

$$1\times2^{2}+0\times2^{1}+0\times2^{0}+0\times2^{-1}+1\times2^{-2}+1\times2^{-3}$$

$$4 \quad + \quad 0.25 + 0.125$$

$$4.375$$

- **Decimal to Octal**: same repeated division method as in Decimal to Binary.

ex- $(266)_{10} = (412)_8$

$$\begin{array}{r|ll} 8 & 266 & \\ 8 & 33 & 2 \\ 8 & 4 & 1 \\ & 0 & 4 \end{array} \uparrow$$

- **Octal to Decimal**:

ex- $(372)_8 = 3\times8^2 + 7\times8 + 2\times8^0 = (250)_{10}$

- **Octal to Binary**: 3 bit equivalent of each digit is written.

ex- $(472)_8 = (100111010)_2$

4 7 2 ← binary of individual digits
↓ ↓ ↓ (using 3 bits of each octal digit)
100 111 010

$$\begin{array}{r|ll} 2 & 4 & \\ 2 & 2 & 0 \\ 2 & 1 & 0 \\ & 0 & 1 \end{array}\uparrow \qquad \begin{array}{r|ll} 2 & 7 & \\ 2 & 3 & 1 \\ & 1 & 1 \end{array} \qquad \begin{array}{r|ll} 2 & 2 & \\ 2 & 1 & 0 \\ & 0 & 1 \end{array}\uparrow$$

Page No.
Date: / /

Couple 3 bits n convert to make each digit

Base 2 to Base 8 ⇓ 2^3

directly writting 3 bit equivalent

write 4 bit equivalent.

Base 2 to Base 16 ⇓ 2^4

directly coupling 4 bits n converting to write each 1 digits for 4 binary.

We can directly convert bases with power. just the same way.

Base 4 ⟶ Base 16   Base 3 ⟶ Base 9

- Binary to Octal: bits of binary grouped into 3 bits from Left to Right

ex- $(011\ 010\ 110)_2 = (326)_8$

0+2+1 0+2+0 2+2+0

* Left Right ka chakkar mi h bs jaha 0 add kr skte h krlo jisse 3-3 ki grouping hojae.

- Decimal to Hex: repeated division by 16.
remainders greater than 9 are represented by letters A to F.

A	B	C	D	E	F
10	11	12	13	14	15

ex- $(423)_{10} = (1A7)_{16}$

16	423	
16	26	7
16	1	10
	0	1

- Hex to Decimal:

ex- $(356)_{16} = (3 \times 16^2 + 5 \times 16^1 + 6 \times 16^0) = (854)_{10}$

$(2AF)_{16} = (2 \times 16^2 + 10 + 16^1 + 15 \times 16^0) = (687)_{10}$

Page No.
Date: / /

- **Binary to Hex**: bits grouped into 4.

ex- $(1010111010)_2 = (2BA)_{16}$

$\underbrace{10}_{2}\ \underbrace{1011}_{11 = B}\ \underbrace{1010}_{10 = A}$

ex- $(10101110.010111)_2 = (AE.5C)_{16}$

$\underbrace{1010}_{A}\ \underbrace{1110}_{E}\ .\ \underbrace{0101}_{5}\ \underbrace{11}_{C}$

- **Hex to binary**: write 4 bit binary equivalent of each digit

ex- $(9F2)_{16} = (100111110010)_2$

2 | 9		
2 | 4	1	
2 | 2	0	
1	0	

1001

2 | 15		
2 | 7	1	
2 | 3	1	
2 | 1	1	

1111

2 | 2		
2 | 1	0	
0	1	

0010

- Base 3 to Base 7
  ↳ Base 10 ↗

- Base 4 to Base 8
  ↳ Base 10 ↗
  ↳ Base 2 ↗

ex- $(3123)_4 \longrightarrow (333)_8$

11 10 10 11

$\frac{110}{3}$ $\frac{100}{3}$

Page No.
Date: / /

- COMPLEMENT :
  - 1's complement — 2's complement.

* Complements are to represent -ve number

for 1's complement, "reverse each bit".

ex- 100101 $\xrightarrow{\text{1's complement}}$ 011010

2's complement → 1's complement + 1

trick ex- 100101 $\xrightarrow{\text{2's complmt}}$ 011011

reverse each bit after 1st 1 bit starting from L to R

```
  011010
  +    1
  ------
  011011
```

ex- 00010100 $\xrightarrow{\text{2's complmt}}$ 11101100

reversing after this 1

```
  11101011
  +      1
  --------
  11101100
```

- Substraction using 2's Complement

[A - B = A + 2's complement of B]

Towards the end —
- If carry's there, ⇒ Ans is +ve
  for result, ignore the carry.
- If carry not there ⇒ Ans is -ve
  for result, obtain the 2's complement of the Ans, that'd be the final result.

ex- A= 11001 B = 10110 find A-B.

```
  11001
  01010
  ------
 100011
```

→ carry

hence Ans → 00011

Page No.
Date: / /

In every base, We can define 2 complements.

Base r ⟶ r's complement [Radix complement]

↳ $(r-1)$'s complement [Diminished Radix complement]

$(r-1)$s complement → $(r^n - 1) - N$ — n is no. of digits in given question

(r)s complement → $r^n - N$ — N is no. given in the question.

Subtraction doesn't exist in computer.

ex- $7-3$ can be written as $7-3+10 = \cancel{1}4$ ignored

ex- 124 [9's complement = $(10^3-1) - 124 = 875$] [10's → $10^3 - 124 = 876$]

- BOOLEAN ALGEBRA :

basic operations → OR AND NOT

↳ Closure property. (answers should be in set)

- Boolean postulates & Theorem:

- $x + 0 = x$
- $x . 1 = x$
- $x + 1 = 1$
- $x . 0 = 0$
- $x + \bar{x} = 1$
- $x . \bar{x} = 0$

* $x + xy = x(1+y) = x$
* $A + \bar{A}B = (A+\bar{A})(A+B) = A + B$
- $A.(B+C) = A.B + A.C$
* $A + (B.C) = (A+B).(A+C)$

- De Morgan's Law:

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

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

Page No.
Date: / /

- DUAL of an expression:

$$A + \bar{A} = 1 \iff A.\bar{A} = 0$$

$+ \longrightarrow .$

$. \longrightarrow +$ ex- $A + \bar{A}B = A + B$

$1 \longrightarrow 0$ $A.(\bar{A}+B) = A.B$

$0 \longrightarrow 1$ $A.B = A.B$

* n bit variable - 2^n combinations

A	B	A+B	A.B
0	0	0	0
0	1	1	0
1	1	10	1
1	0	1	0

carry

Design:

1. Identify the number of inputs and outputs
2. Write O/P for each combination of I/P.
(Make the Truth table).

	A	B	S	SOP	POS
0→	0	0	0	$\bar{A}\bar{B}$	$A+B$
1→	0	1	1	$\bar{A}B$	$A+\bar{B}$
3→	1	1	0	AB	$\bar{A}+\bar{B}$
2→	1	0	1	$A\bar{B}$	$\bar{A}+B$

$SOP = \Sigma(1,2)$
$= \bar{A}B + A\bar{B}$

$POS = \Pi(0,3)$
$= (A+B).(\bar{A}+\bar{B})$

sequencing through Gray Code

$0 \to \bar{A}, \bar{B}$
$1 \to A, B$
minterms
[Collect all 1s]

$0 \to A, B$
$1 \to \bar{A}, \bar{B}$
maxterms
[Collect all 0s]

Page No.
Date: / /

Binary	Gray Code
0	0
1	1
2	3
3	2

* Gray code is used so that ek bit se next bit me jaane se sirf 1 bit ka change ae. Bade circuits me isse time difference aata h.

Each successive member differs just by 1 bit...

- CANONICAL FORM - STANDARD FORM of an expression.

ex- $Y = AB + B\bar{C}$ (Standard form) [SOP Case]

$AB.1$ $\qquad$ $B\bar{C}(A+\bar{A})$

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

$\underbrace{ABC}_{2^2+2+1=7} + \underbrace{AB\bar{C}}_{2^2+2=6}$ $\qquad$ $\underbrace{AB\bar{C}}_{2^2+2+0=6} + \underbrace{\bar{A}B\bar{C}}_{2=2}$

$\underbrace{Y = \Sigma(2,6,7)}$
(Canonical form)

ex- $(A+B)(\bar{B}+C)$ [POS case]

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

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

$\underbrace{0\ 0\ 0}_{0}$ $\underbrace{0\ 0\ 1}_{1}$ $\qquad$ $\underbrace{0\ 1\ 0}_{2}$ $\underbrace{1\ 1\ 0}_{6}$

$Y = \Pi(0,1,2,6)$

* We can convert SOP to POS and vice versa by writing canonical form of the expression first, then change Σ to Π or Π to Σ as per question, then we can write the Standard form using the converted canonical form.

SARASWATI

Page No.
Date: / /

ex- Convert $AB+\bar{B}C$ to POS

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

$\underbrace{ABC}_{7} + \underbrace{AB\bar{C}}_{6}$ $\qquad \underbrace{A\bar{B}C}_{5} + \underbrace{\bar{A}\bar{B}C}_{1}$

$$Y = \Sigma(1,5,6,7) = \Pi(0,2,3,4)$$
$$= (A+B+C).(A+\bar{B}+C).(A+\bar{B}+\bar{C}).(\bar{A}+B+C)$$

- COMPLEMENT OF A FUNCTION:

ex- $X = AB+BC$

$$\bar{X} = (\overline{AB+BC}) = (\overline{AB}).(\overline{BC})$$
$$= (\bar{A}+\bar{B}).(\bar{B}+\bar{C})$$
$$= \bar{A}\bar{B} + \bar{B} + \bar{B}\bar{C} + \bar{A}\bar{C}$$

ex- $x = \Sigma(1,2,3)$

$$\bar{x} = \Sigma(0,4,5,6,7)$$

- CONCENSUS THEOREM:

When 3 terms, one with x one with x', third gets cancelled. Recognise the pattern.

ex- $xy + x'z + \cancel{yz} = xy + x'z$

$xy' + x'z + \cancel{y'z} = xy' + x'z$

$xy' + x'z' + \cancel{y'z'} = xy' + x'z'$

derivations of conensus:

ex - $xy + x'z + yz$

$xy + x'z + (x+x')yz$

$xy + xyz + x'z + x'yz$

$xy(1+z) + x'z(1+y) = xy + x'z$

ex- $xy' + x'z + y'z$

$xy' + x'z + (x+x')y'z$

$xy' + x'z + xy'z + x'y'z$

$xy'(1+z) + x'z(1+y')$

$= xy' + x'z$

- KARNAUGH MAP :

- 2 Variable

	$\bar{B}$	B
$\bar{A}$	0	1
A	2	3

- 3 variable

AB\C	$\bar{C}$	C
$\bar{A}\bar{B}$	0	1
$\bar{A}B$	2	3
AB	6	7
$A\bar{B}$	4	5

	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
$\bar{A}$	0	1	3	2
A	4	5	7	6

A	B	C	
0	0	0	→ 0
0	0	1	→ 1
0	1	0	→ 2
0	1	1	→ 3
1	0	0	→ 4
1	0	1	→ 5
1	1	0	→ 6
1	1	1	→ 7

- 4 variables

AB \ CD

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	1	3	2
$\bar{A}B$	4	5	7	6
AB	12	13	15	14
$A\bar{B}$	8	9	11	10

CD \ AB

0	4	12	8
1	5	13	9
3	7	15	10
2	6	14	11

- 5 variables

BC \ DE

0	1	3	2
4	5	7	6
12	13	15	14
8	9	11	10

A = 0

BC \ DE

16	17	19	18
20	21	23	22
28	29	31	30
24	25	27	26

A = 1

- GROUPING :

$\bar{A}\bar{C}D$

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

	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0	1	0	1
$\bar{A}B$	1	1	0	0
AB	0	0	1	1
$A\bar{B}$	0	0	1	1

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

AC

$$Y = \Sigma(1, 2, 4, 5, 10, 11, 14, 15)$$

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

- Prime Implicants :

Check each group ~~of~~ member if the group contains atleast 1 member ie not the part of any other group.

- 5 variable KMap.

	$\bar{D}\bar{E}$	$\bar{D}E$	DE	$D\bar{E}$
$\bar{B}\bar{C}$	0	0	0	0
$\bar{B}C$	1	0	0	1
BC	1	0	0	1
$B\bar{C}$	0	0	0	0

A=0

	$\bar{D}\bar{E}$	$\bar{D}E$	DE	$D\bar{E}$
$\bar{B}\bar{C}$	0	0	0	0
$\bar{B}C$	1	0	0	1
BC	1	0	0	1
$B\bar{C}$	0	0	0	0

A=1

$$\bar{A}C\bar{E} + AC\bar{E}$$
$$= C\bar{E}$$

- DON'T CARE CONDITION :

need ke according kuchh bhi bhar skte h

ex- $Y = \Sigma(0,2,5) + d(6,7)$

notation for don't care condition

	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
$\bar{A}$	1	0	0	1
A	0	1	x	x

putting 1

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

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

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

Diagonal elements can be implemented using XOR or XNOR