09/05/2022 # LOGIC & SWITCHING THEORY

* **Digital:** works with electricity, numbers, Eg. Gadgets
  Series of digits; works with zeroes & ones
* **Analog:** physical appearance; some mechanism, continuous wave, signals

⇒ Use of ~~analog~~ digital over analog:

digital: discretion of signal

- quality of signal: accurate
- more cable / circuit design is simple.
- time/speed (processing)
- loss of data is less ⇒ problem is solved by changing it to digits
- Processing is easy

* **Logic:** Reasoning & Connecting the given things effectively

UNIT-1: Boolean Algebra

UNIT-1 & 2: Designing effectively / Minimization

UNIT-2: Components / Logic Gates

UNIT-3: Combinational Design

UNIT-4: Sequential Design (Memory element involved)

UNIT-5: Programmable Logic Device

12/05/2022 * **Purpose:** To design digital circuits.

* **Analog:** Continuous data

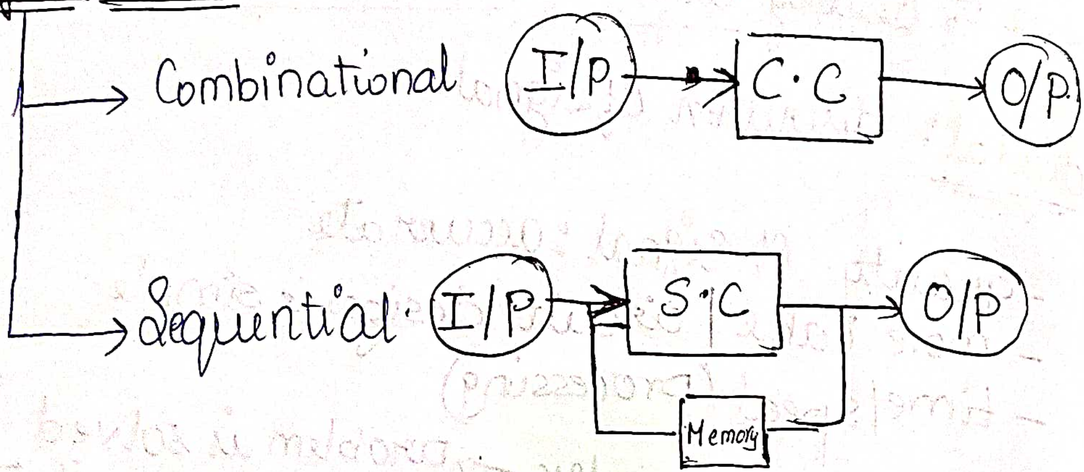
* **Digital:** Discrete data

* Most of digital circuits take 5V (or) 12V.

for 5V $\quad 0-2V \rightarrow 0 \rightarrow$ logic 0

$\quad\quad\quad > 2-5V \rightarrow 1 \rightarrow$ logic 1

* **Digital Circuits:**



1) $(63)_{10} = (111111)_2$

$$\begin{array}{l} 2 \,|\, 63 \\ 2 \,|\, 31 - 1 \\ 2 \,|\, 15 - 1 \\ 2 \,|\, 7 - 1 \\ 2 \,|\, 3 - 1 \\ \quad 1 - 1 \end{array}$$

$2 \,|\, 7$

2) $(111)_2 = 4+2+1 = (7)_{10}$

$\quad 2^2 2^1 2^0$

3) $(63)_{10} = (77)_8$

$\quad\quad\quad\quad (3F)_{16}$

$8 \,|\, 63 - 7$

$\quad 7$

$16 \,|\, 63 - 15$

$\quad\quad 3$

$\begin{array}{r} 63 \\ 48 \\ \hline 15 \end{array}$

$(77)_8 = 56 + 7 = 63.$

$8^1 8^0$

$(3F)_{16} = 48 + 15 = 63$

$16^1 16^0$

4) $(1F4)_{16} - (500)_{10}$

$16^2 16^1 16^0$

$256 + 240 + 4 = 500.$

5) $(193)_{10}$ = ~~(301)~~$_{16}$ $(C1)_{16}$.

$(301)_8$

$$8\,|\,193 - 1$$
$$8\,|\,24 - 0$$
$$3$$

24

$$16\,|\,193 - 1$$
$$12$$

12

13/05/2022

1) $(1B2)_{16} = (\quad)_8$
2) $(765)_8 = (\quad)_{16}$.

2) $(768)_8 = (\quad)_{16}$

Not possible.

1) $(1B2)_{16} = (434)_{10}$.

$16^2\ 16^1\ 16^0$

$256+176+2$.

$(434)_{10} = (662)_8$.

$$16\,|\,434 - 2$$
$$16\,|\,27$$
$$1$$

$$\begin{array}{r} 434 \\ 32 \\ \hline 114 \end{array}$$

$$\begin{array}{r} 16 \\ \times 7 \\ \hline 112 \end{array}$$

$$8\,|\,434 - 2$$
$$8\,|\,54 - 6$$
$$6$$

2) $(765)_8 = (501)_{10}$

$8^2\,8^1\,8^0$

$(501)_{10} = (1F5)_{16}$.

$$16\,|\,501 - 5$$
$$16\,|\,31 - 15$$
$$1$$

3) $(110\cdot011)_2 = (6\cdot375)_{10}$

$2^2\,2^1\,2^0\cdot 2^{-1}\,2^{-2}\,2^{-3}$

$4+2 \quad 0+0\cdot25+0\cdot125$

30/05/2022

BOOLEAN ALGEBRA:

Properties:

$\boxed{x, y, z \in B}$

1) $B = \{0, 1\}$

x, y is closed under : $+$ (or)

: $(\cdot)$ (and).

2) Commutative Law :

$$\boxed{\begin{aligned} x+y &= y+x \\ x \cdot y &= y \cdot x \end{aligned}}$$

3) Distributive Law: $x(y+z) = xy + xz$

$x + yz = (x+y)(x+z)$.

4) $x \in B$

$\exists\, x' \in B$ } Unary Operator

5) $x + x' = 1 \in B$

$x \cdot x' = 0 \in B$

6) $x, y \in B$; then $x \neq y$

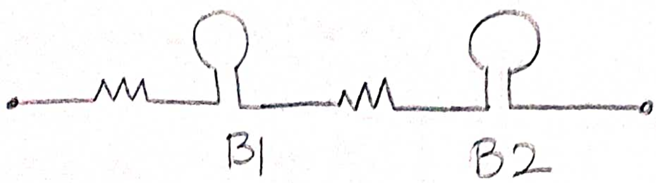
*

→ $\wedge$ – AND

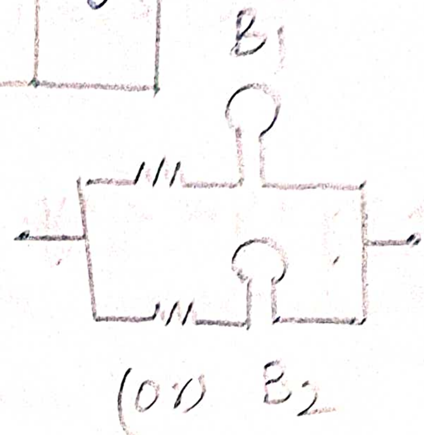
→ $\vee$ – OR

→ $-$ / $'$ – NOT

x	y	$x \cdot y$	$x+y$	x'	y'
0	0	0	0	1	ϕ
1	0	0	1	0	1
0	1	0	1	1	0
1	1	1	1	0	0



(and)



(or)

x	y	$x+y$	$y+x$	$x \cdot y$	$y \cdot x$	$x(y+z)$	$x \cdot y + x \cdot z$
0	0	0	0	0	0		
1	0	1	1	0	0		
0	1	1	1	0	0		
1	1	1	1	1	1		

x	y	z	$y+z$	$x(y+z)$	$x \cdot y$	$y \cdot z$	$xy+yz$
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	1	0	0	0	0
ϕ	0	0	0	0	0	0	0
1	0	1	1	1	0	0	0
1	1	0	1	1	1	0	1

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

* **Associative law:**

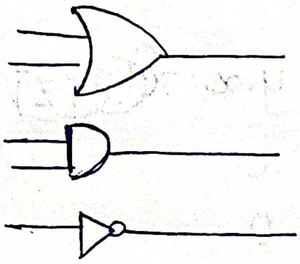
$$x + (y + z) = (x + y) + z$$
$$x \cdot (y \cdot z) = (x \cdot y) \cdot z$$

1) $x + x = x$
2) $x \cdot x = x$
3) $x + xy = x$
4) $x \cdot 1 = x$
5) $xy + yx = xy$. $(\because x + x = x$

* $xy + yx = xy$.

Proving - Postulates
- Using table

→ $+$ →
→ $\cdot$ →
→ $'$ →



BASIC LOGIC GATES

03/06/2022

$x = 1 \quad x' = 0$
$y = 0$

$x + x'y = x + y = 1$

$x = 0 ; y = 1 ; x' = 1$

$x + x'y = 1 = x + y$

x	y	x'	$x'y$	$x + x'y$	$x + y$
0	0	1	0	0	0
0	1	1	1	1	1
1	0	0	0	1	1
1	1	0	0	1	1

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

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

$= x + xy + x'y$

$= x + y(1)$

$= \underline{\underline{x+y}}$

* $x'y'z + yz + xz$

$x'y'z + yz\cdot 1 + xz\cdot 1$

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

$z[x'y' + x + y] = z[x'(y') + x(1+y) + y(1+x)]$

$= z[x'y' + x + xy + y + xy]$

* $[x'y' + x + y]z$

$z[x'y' + x + y] = z$

x	y	z	x'	y'	$x'y'$	$x+y$	$x'y'+x+y$	z
0	0	0	1	1	1	0	1	0
0	0	1	1	1	1	0	1	1
0	1	0	1	0	0	1	1	0
1	0	0	0	1	0	1	1	0
1	1	1	0	0	0	1	1	1
1	1	0	0	0	0	1	1	0
1	0	1	0	1	0	1	1	1
0	1	1	1	0	0	1	1	1

$z[x'y' + x + y]$

$= z[x + y' + y]$

$= z[x + 1] = z$

* $(x+y)[x'(y'+z')]' + x'y + x'z$

$(x+y)[x'y' + x'z']' + x'(y+z)$

* <u>De-Morgan's Theorem</u>:

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

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

$(x+y)(x'y')' \cdot (x'z')' + x'(y+z)$

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

$(x+y)^2(x+z) + x'(y+z)$

$x(x+y)^2 + z(x+y)^2 + x'(y+z)$

$(x+y)^2(x + x'z) + x'y + x'z$

* $(x+y)[x'(yz)']' + x'(y+z)$

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

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

$= \cancel{x} + xy(1) + x'(y+z) + yz$

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

$= x + y + x'z$ $\quad = x + x'y + x'z + yz$

$= \cancel{x + x'z} = \cancel{x'z}$ $\quad = \cancel{x + x'(y+z)}$

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

$= x + x'y + zy + z$

$= x + y + yz + z$

$= x + y(1) + z$

$= \underline{\underline{x + y + z}}$

* $xyz + x(y' + z')$.

$x(yz + y') + xz'$

$= x(y' + z) + xz'$

$= xy' + x$

$= x(y' + 1)$

$= \underline{\underline{x}}$

*

$w'x(z' + y'z) + x(w + w'yz)$

$\cancel{w'(x)(z' + y') + xw(1 + yz)}$

$\cancel{w'(x)(z' + y') + xwyz}$

$w'x(z' + y') + x(w + yz)$

$\cancel{w'xz' + w}$

$x[w'z' + w'y' + w + yz]$

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

* $w'x(z'+y'z) + x(w+w'yz)$.

$w'(x)(z'+y') + x(w+yz)$.

$x[w'z' + w'y' + w + yz]$

$= x[w'(yz)' + w + yz]$

$= x[(w+yz)' + (w+yz)] = x[1] = x$

* $xy + x'z + yz = xy + x'z$

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

$\cancel{= xy + z(xy)'}$

$= xy + x'z + zy$

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

x	x'	y	z	$x+z$	$y(x+z)$	$x'z$	Sum (LHS)	xy (~~RHS~~)	$xy+x'z$ (RHS)
0	1	0	0	0	0	0	0	0	0
0	1	0	1	1	0	1	1	0	1
0	1	1	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0	0
1	0	1	1	1	1	0	1	1	1
1	0	0	1	1	0	0	0	0	0
1	0	1	0	1	1	0	1	1	1
0	1	1	1	1	1	1	1	0	1

$\boxed{LHS = RHS}$

* $xy+x'z+yz = xy+x'z$

$\cancel{xy+x'z}$

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

$y(x+x'\cdot z)+x'z$

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

$= xy+x'z$

1. $xy+yz+x'z \Rightarrow$ ~~CONSENSUS THEOREM~~

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

$xy+xyz+yz+x'z$

$xy+x'z+yz$

* CONSENSUS THEOREM : $xy+x'z+yz = xy+x'z$

$\therefore xy+x'z+yz+1$

A function contains variables $x, y, z \in B$ and operators (and, or, not)

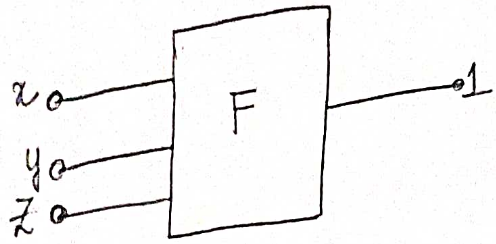
x	y	z	xy	$x'z$	yz	$xy+x'z+yz$ (F)
0	0	0	0	0	0	0
0	0	1	0	1	0	1
0	1	0	0	0	0	0
1	0	0	0	0	0	0
0	1	1	0	0	1	1
1	0	1	0	0	0	0
1	1	0	1	0	0	1
1	1	1	1	0	1	1

$$\boxed{F = \Sigma m(1,3,6,7)} \rightarrow \text{minterms: } \boxed{F = 1}$$

$$F = \underbrace{x'y'z}_{\text{term}} + x'yz + xyz' + xyz$$

↳ variable/literal

For these four terms: function evaluates to



$$= xy + x'z + yz$$

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

$$= xyz + xyz' + x'yz + x'y'z + xyz + x'yz$$

$$= xyz + xyz' + x'y'z + x'yz \cancel{+ (x'+x)yz}$$

$$= xyz + xyz' + x'y'z + x'yz$$

Minterm: Combination of all the variables which are unknown in a given expression.

Function 'F' is referred to as Sum of Products form. (SOP)

$$F = x'y'z + x'yz + xyz' + xyz \Rightarrow \text{Sum of Minterms}$$

$$\boxed{F' = \Sigma m(0,2,4,5)} \quad \boxed{F = 0}$$

$x=0,\ y=0;\ z=0;\ F'=0$

$\underbrace{\hphantom{x=0,\ y=0;\ z=0}}$

maxterms $\Rightarrow$ to represent F'

$x=1;\ y=1;\ z=1;\ F=1$

$\underbrace{\hphantom{x=1;\ y=1;\ z=1}}$

min terms $\Rightarrow$ to represent F

$F = \Pi M(0,2,4,5)$ $\quad F' = \Pi M(1,3,6,7)$.

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

$\Rightarrow$ Product of Sums (POS form) = Product of Max terms

* $F = xy + x'z$

(a) POS

(b) complement of F

(c) SOP

(d) Canonical of POS.

$F = xy + x'z$

$F = xyz + x'yz$

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

(c) $F = xy + x'z$

(b) $F = (xy + x'z)'$

$= (xy)' \cdot (x'z)'$

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

$F' = (x'+y')\cdot(x+z')$

If we have SOP we get POS as compliment

(or)

If we have POS we get SOP as compliment

(d) POS: $xy + x'z$

$= xyz + xyz' + x'yz + x'y'z$

a) POS

$a\cdot(b+c) = a\cdot b + a\cdot c.$

$\underset{a}{xy} + \underset{b}{x'}\underset{c}{z} = (xy + x')(xy+z)$ $\quad a + bc = (a+b)(a+c)$

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

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

$= (x'+y)(xy+z)$ $\quad \Pi$

$= (x'+y)(x+z)(y+z) \longrightarrow$ POS

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

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

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

$F = \Pi M(4, 5, \cancel{0}, 2).$

$F = \Sigma m(1, 3, 6, 7)$

) Canonical SOP:

$= xy + x'z.$

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

$= xyz + xyz' + x'yz + x'y'z$

* $F = yz + x' + xz'$ 06/06/2022.

a) Canonical SOP
b) Canonical POS
c) Minimal POS
d) F'
e) Truth Table.

x	y	z	F
0	0	0	1
0	0	1	1
0	1	0	1
1	0	0	1
1	1	1	1
1	1	0	1
1	0	1	0
0	1	1	1

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

1 1 0
1 0 0

POS =
$(x + y' + z)$

Canonical SOP $= \Sigma m(0, 1, 2, 4, 7, 3, 6)$

Canonical POS $= \Pi M(5)$.

$F = yz(x + x') + x'(y + y')(z + z') + x(y + y')z'$

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

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

a) POS

$$a\cdot(b+c) = a\cdot b + a\cdot c$$

$$\underline{xy} + \underline{x'}\,\underline{z} = (xy+x')(xy+z) \qquad a+bc = (a+b)(a+c)$$

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

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

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

$$= (x'+y)(x+z)(y+z) \longrightarrow \underline{\underline{POS}}$$

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

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

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

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

$$F = \Pi M(4, 5, 0, 2)$$

$$F = \Sigma m(1, 3, 6, 7)$$

e) Canonical SOP:

$$= xy + x'z$$

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

$$= xyz + xyz' + x'yz + x'y'z$$

06/06/2022

* $F = yz + x' + xz'$

a) Canonical SOP
b) Canonical POS
c) Minimal POS
d) F'
e) Truth Table.

Max: $x = 0$
$x' = 1$

Min: $x = 1$
$x' = 0$

x	y	z	F
0	0	0	1
0	0	1	1
0	1	0	1
1	0	0	1
1	1	1	1
1	1	0	1
1	0	1	0
0	1	1	1

POS =
$(x + y' + z)$

Canonical SOP $= \Sigma m(0, 1, 2, 4, 7, 3, 6)$

Canonical POS $= \Pi M(5)$.

$F = x' + y$

$F = yz(x + x') + x'(y + y')(z + z') + x(y + y')z'$

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

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

09/06/2022

* $F = (x+y)\cdot(x'+y')$

$F = x\cdot x' + xy' + y\cdot x' + yy'$

$\boxed{F = xy' + x'y}$ → Canonical SOP

$F' = \left[(x+y)\cdot(x'+y')\right]'$

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

$(F')' = (x'y' + xy)'$ $\Rightarrow \underline{x'y + xy'}$

$= \cancel{(x'y')'\cdot(xy)'}$

$= \cancel{(x+y)\cdot(x'+y')}$

* $F = (x+y+z')(x'+y'+z)(x+y')$

$= (xx' + xy' + xz + yx' + yy' + yz + z'x'$
$+ z'y' + z'z)(x+y')$

$= (xy' + xz + x'y + yz + x'z' + y'z')(x+y')$

$= xy' + xy' + xz + xy'z + (0) + 0 + xyz + 0$
$+ 0 + x'z'y' + xy'z' + y'z'$

$F = xy' + xz + xzy' + xyz + x'y'z' + xy'z'$
$+ y'z'$

$F = xy' + xz + y'z'$

x	y	z	F
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

$\Rightarrow$ Maxterm form

$F = \prod M(1,2,3,6) \Rightarrow$ POS

$= \Sigma m(0,4,5,7) \Rightarrow$ SOP.

$F' = \Sigma m(1,2,3,6)$

$F' = \prod M(0,4,5,7)$

$F = \Sigma m(0,4,5,7)$

$\Rightarrow$ gives canonical SOP

$\Rightarrow$ we need to simplify to get minimal SOP.

10/06/2022

* KARNAUGH MAP: [K-Map]

$F_{(x,y,z)} = \Sigma m(0,4,5,7)$.

The number of cells in a map depends on the no. of combinations (no. of variables) $= 2^n$.

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

x → most signifi bit

y → middle.

z → least signifi bit

1) Take a map with 2^n cells (n = no. of cells).
2) Map 1's on to the cells based on function.
3) Grouping 1's. (Grouping can be done only in horizontal/vertical.
4) Consider grouping of max. possible 1's.

5) Group must be of powers of 2 (2, 4, 8, ... $2^0, 2^1, 2^2, 2^3, \ldots$)

$\Rightarrow \boxed{\cancel{y'z' + xz}}$; $\boxed{y'z' + xz + xy'}$ / $y'z' + xz$

This method is used only when it is in canonical

$$\left.\begin{matrix} 00 \\ 01 \\ 11 \\ 10 \end{matrix}\right\}$$ Gorey Code: which only one bit changes at a time.

$\Rightarrow xy' + xz + y'z'$

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

x	y	z	x'	y'	z'	xy'	xz	$y'z'$	$y'z$	xz
0	0	0	1	1	1	0	0	1	1	0
0	0	1	1	1	0	0	0	0	0	0
0	1	0	1	0	1	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0	0
1	0	0	0	1	1	1	0	1	1	0
1	0	1	0	1	0	1	0	0	0	1
1	1	0	0	0	1	0	0	0	0	0
1	1	1	0	0	0	0	1	0	0	1

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

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

* $F = \Sigma m(0,1,2,3)$

x \ yz	00	01	11	10
0	1	1	1	1
1				

$x'y' + x'y = x'$

$= x'$

* $F = \Sigma m(0,1,6,7)$

x \ yz	00	01	11	10
0	1	1		
1			1	1

$\boxed{x'y' + xy}$

* $F = \Sigma m(0,1,2,5,7)$

x \ yz	00	01	11	10
0	1	1		1
1		1	1	

$x'y' + xz + x'yz'$

* $\Sigma m(0,1,3,6,7)$

x \ yz	00	01	11	10
0	1	1	1	
1			1	1

$\boxed{x'y' + x'z + xy}$

*

x \ yz	00	01	11	10
0		1	1	
1		1	1	1

$x'y' + x'z + yz$

$= x'z + xz + xy$

$= z + xy.$

*

x \ yz	00	01	11	10
0	1		1	1
1	1			1

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

$= z' + x'y$

*

$x \backslash yz$	00	01	11	10
0	1			1
1			1	

$x'z' + xyz$

* **Implicant**: 1 mapped in K-map.

* **Prime Implicant**: Group of 1's (Largest | Maximum)

$x \backslash yz$				
	1	1	1	
			1	1

5 implicant
3 Prime implicants

* **Essential Prime Implicant**:

Atleast one 1 not covered by any other group.

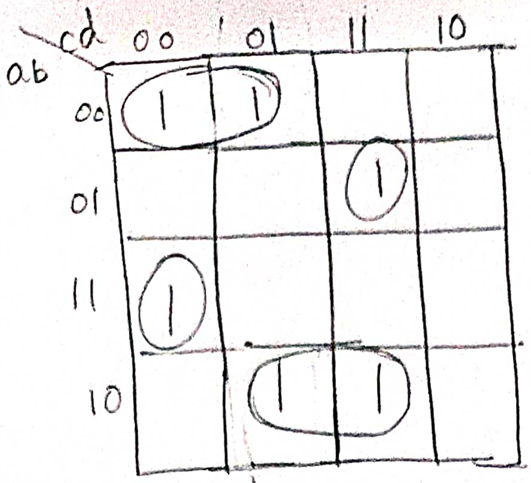
Ex: $F = x'y' + x'z + xy$
$F = x'y' + yz + xy$

$\therefore$ $x'y'$ and xy are essential prime implicant

* **K-Map (Karnaugh Map)**:

4 Variable K-map: $F = \Sigma m(0, 1, 7, 9, 11, 12)$

$ab \backslash cd$	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10	8	9	11	10

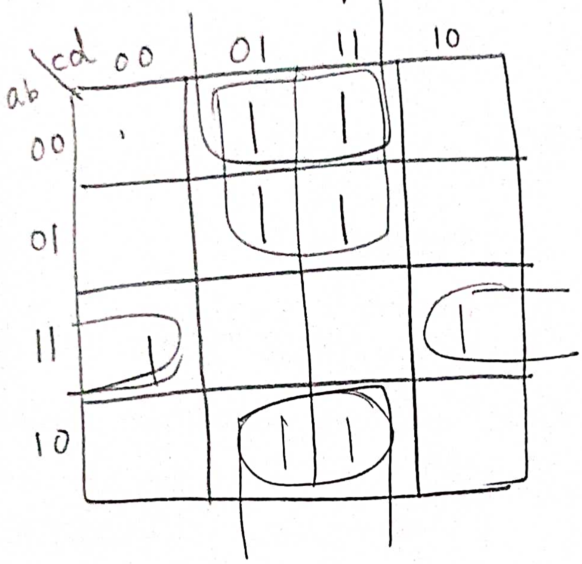


$a'b'c' + a'bcd + abc'd' + ab'd$

Implicants = 6

PI = 4

EPI = 4



$abd' + a'd + ab'd$

Implicants = 8

P.I = 3

EPI = 2

ab \ cd	00	01	11	10
00	1	1		
01			1	
11	1			
10		1	1	

$a'b'c' + a'bcd + abc'd' + ab'd$ ~~+ ...~~

Implicants = 6

PI = 4

EPI = 4

ab \ cd	00	01	11	10
00		1	1	
01		1	1	
11	1			1
10		1	1	

$abd' + a'd + ab'd$

Implicants = 8

P.I = 3

EPI = 3

13/06/2022

4 var K-Map

$F = \Sigma(0,1,2,3,5,9,10,11,13,15)$.

ab \ cd	00	01	11	10
00	1 [0]	1 [1]	1 [3]	1 [2]
01	[4]	1 [5]	[7]	[6]
11	[12]	1 [13]	1 [15]	[14]
10	[8]	1 [9]	1 [11]	1 [10]

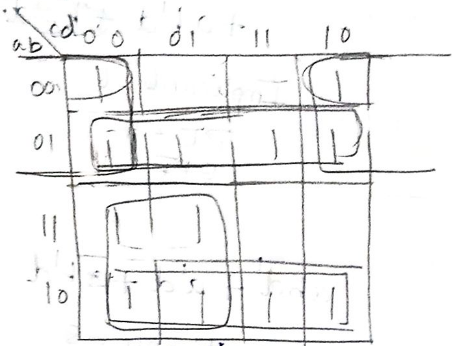
Implicants = ~~8~~ 10

PI = 4

EPI = 4

$a'b' + c'd + b'c + ad$.

* $\Sigma(0,2,4,5,6,7,8,9,10,11,12,13)$.

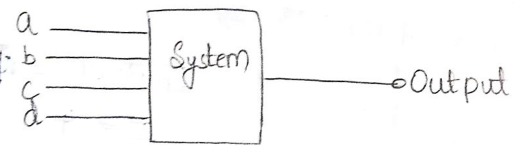


$F = a'd + \cancel{bc'}\; ac' + a'b + ab'$

Implicants = 12 ; P.I = 4 ; EPI = 4

* Don't cares / condition:

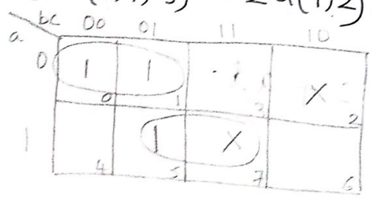
*→ For a particular input; we get an output which is not necessary.



10, 11, 12, 13, 14, 15 } Eroneous Input
1010 . – – – – – . 1111

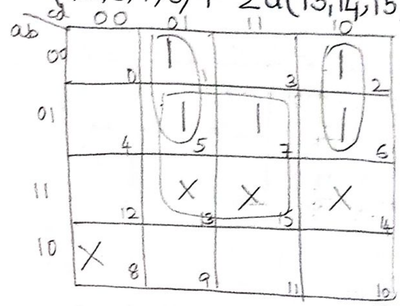
16/06/2022

* $F = \Sigma m(0,1,5) + \Sigma d(7,2)$.



$a'b' + ac$

* $G = \Sigma m(1,2,5,7,6) + \Sigma d(13,14,15,8)$.



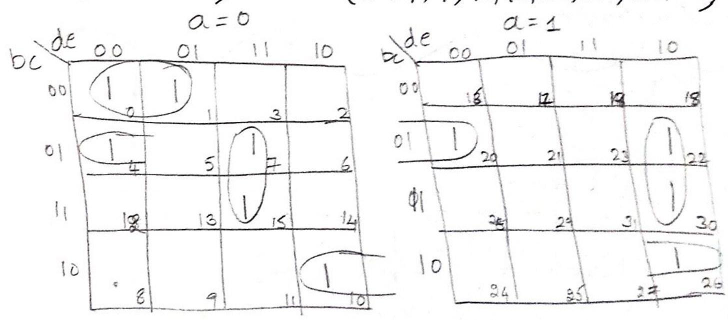
$a'c'd + bd + a'cd'$

* 5-Variable K-Map:

$2^5 = 32$ combinations.

We use 2 geilds containing 16 cells each.

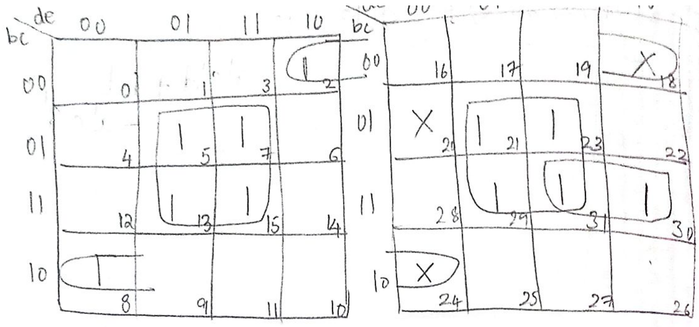
$F(a,b,c,d,e) = \Sigma m(0,1,4,7,10,15,20,22,26,30)$



$F = a'b'c'd' + a'cde + b'cd'e' + bc'de'$

* $F(a,b,c,d,e) = \Sigma m(2,5,7,13,15,8,21,23,29,30,31) + \emptyset(18,20,24)$

$+ acde'$



$a=0$ $\qquad$ $a=1$

$F = b'c'de' + bc'd'e' + a'ce + ace + abcd$

$= b'c'de' + bc'd'e' + ce + abcd.$

$I = 14 \qquad P.I = 4 \qquad EPI = 4$

23/06/2022

UNIT-2

TABULATION METHOD:

Purpose: K-Map → Simplify Boolean functions to minimum no. of literals

$F = \Sigma m(0,1,3,6,7)$

Step-1: Group according to no. of 1's

✓0 − 0000
✓1 − 0001) one bit change

✓3 − 0011
6 − 0110

✓7 − 0111

Step-2: Group adjacent elements (only one bit change group)

0,1 : 00_
1,3 : 0_1
6,7 : 11_
3,7 : _11

Step-3: Cannot group further → Stop

Step-4: Prime Implicant Chart

		0	1	3	6	7
*	a'b' (01)	X	X			
	a'c (13)		X	X		
	bc (37)			X		X
*	ab (67)				X	X

column with only 1 X is called P.I.

$F = a'b' + a'c + ab$

(or) $= a'b' + bc + ab$

$E.I = 2(a'b', ab)$

2) $F = \Sigma m(0,1,5,6,9,11,12,13,15)$.

Step-1

✓0 - 0000
✓1 - 0001
✓5 - 0101
6 - 0110
✓9 - 1001
✓12 - 1100
✓~~12~~ - 1001
✓13 - 1101
✓15 - 1111

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

2|11-1
2|5-1
2|2-0
1

2|13-1
2|6-0
2|3-1
1

Step-2: underscore should match

0,1 : 000_
✓1,5 : 0_01
✓1,9 : _001
✓9,11 : 10_1
✓5,13 : _101
✓9,13 : 1_01
12,13 : 110_
✓11,15 : 1_11
✓13,15 : 11_1

Step-3

1,5,9,13 : __01
1,9,5,13 : __01
9,13,11,15 : 1__1
9,11,13,15 : 1__1

Step-4 P.I. chart

	0	1	5	6	9	11	12	13	15
*a'bcd'				X					
*a'b'c'	X	X							
*abc'							X	X	
*c'd		X	X		X			X	
*ad					X	X		X	X

$F = a'bcd' + a'b'c' + abc' + c'd + ad$

24/06/2022

1) $\Sigma m(2,6,8,9,10,11,14,15)$

2 - 0010
6 - 0110
8 - 1000
9 - 1001
10 - 1010
11 - 1011
14 - 1110
15 - 1111

Step-I

✓2 - 0010
✓8 - 1000
✓6 - 0110
✓9 - 1001
✓10 - 1010
✓11 - 1011
✓14 - 1110
✓15 - 1111

Step-II

✓2,6 : 0_10
✓2,10 : _010
✓8,9 : 100_
✓8,10 : 10_0
✓6,14 : _110
✓9,11 : 10_1
✓10,11 : 101_
✓10,14 : 1_10
✓11,15 : 1_11
✓14,15 : 111_

Step-III

a b c d

2,6,10,14 : __10
2,10,6,14 : __10
8,9,10,11 : 10__
8,10,9,11 : 10__
10,14,11,15 : 1_1_
10,11,14,15 : 1_1_

	2	6	8	9	10	11	14	15
*cd'	X	X			X		X	
*ab'			X	X	X	X		
*ac					X	X	X	X

$\boxed{F = cd' + ab' + ac}$

3) $G = \Sigma m(0,4,5,7,12,15) + d(8,11)$

0	0000
4	0100
5	0101
7	0111
12	1100
15	1111
8	1000
11	1011

I

✓0 – 0000

✓4 – 0100
✓8 – 1000

✓5 – 0101
✓12 – 1100

✓7 – 0111
✓11 – 1011

✓15 – 1111

Step–II

✓0,4 : 0–00
✓0,8 : –000

4,5 : 010–
✓4,12 : –100
✓8,12 : 1–00

5,7 : 01–1
11,15 : 1–11
7,15 : –111

Step–III:

0,4,8,12 : – – 00
0,8,4,12 : – – 00

$\boxed{\text{P.I} = 5,\ \text{EPI} = 1}$

	0 ✓	4 ✓	5	7	12 ✓	15	8	11
a'bc'		X	X					
a'bd			X	X				
acd						X		X
bcd				X		X		
* c'd'	X	X			X		X	

$\boxed{\cancel{F = acd}}$ $\boxed{\cancel{F = c'd'}}$

$F = c'd' + a'bd + acd$

$= c'd' + bcd + a'bc'$

$= c'd' + a'bd + acd$

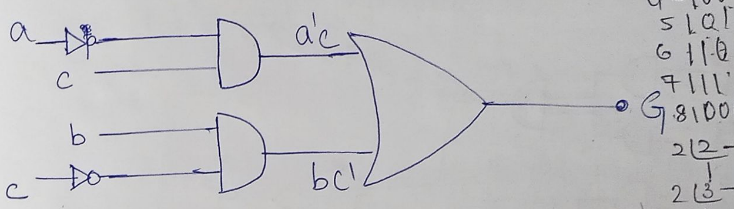
27/06/2022

$G = \Sigma m(1,3,6) + d(2,5)$: Design a AND-OR circuit.

1 – 0001
3 – 0011
6 – 0110
2 – 0010
5 – 0101

a \ bc	00	01	11	10
0	(0)	1 (1)	1 (3)	X (2)
1	(4)	X (5)	(7)	1 (6)

$\boxed{G = a'c + bc'}$



00
01
11
10

1 – 001
2 – 010
3 – 011
4 – 100
5 101
6 110
7 111
8 1000

$G = \Sigma m(1,2,7,9,11) + d(12,8,13)$

ab \ cd	00	01	11	10
00	(0)	1 (1)	(3)	1 (2)
01	(4)	(5)	1 (7)	(6)
11	X (12)	X (13)	(15)	(14)
10	X (8)	1 (9)	1 (11)	(10)

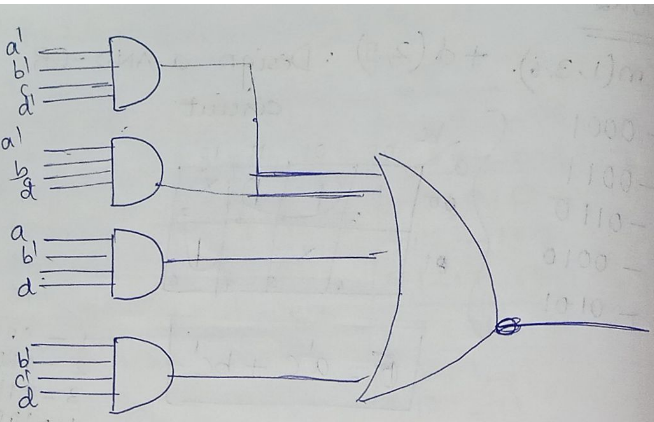
$c' = 0$
$c = 1$

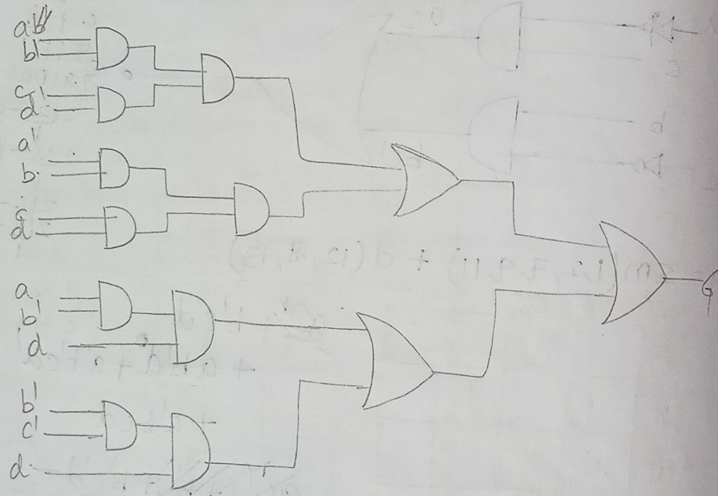
$\cancel{ac'} + b'c'd + ab'd + a'b'cd' + a'bcd$

$= \cancel{ac'(1+b') + b'c'd + a'b'cd' + a'bcd}$

$= ac' + b'c'd + a'b'cd' + \cancel{a'bcd}$

$= b'c'd + ab'd + a'b'cd' + a'bcd$





⇒ All the parallel running gates have same time delay.

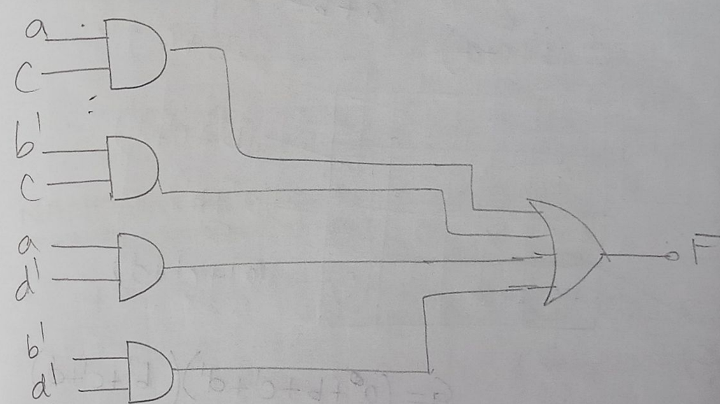
If the propagation delay of AND gate is 2msec and OR gate is 3msec. Calculate the total propagation delay.

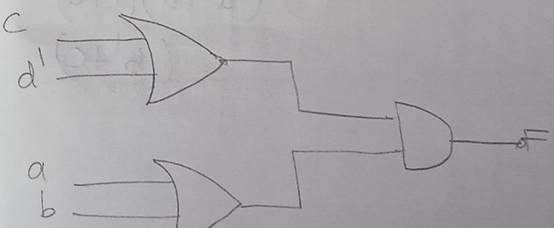
→ The amount of time taken to give the output for the given input is called propagation delay.

total time = 2+2+3+3 = 10.

$* F = (c+d')(a+b')$

$= ac + b'c + ad' + b'd'$





* OR–AND CIRCUIT:

$F = b'c'd + ab'd + a'b'cd' + a'bcd$

$\cancel{F' = (b+c) + d' + (a'+b) + d' + a+b+c'+d}$

ab \ cd	00	01	11	10
00		1		1
01			1	
11	X	X		
10	X	1	1	

$(a+b+c'+d)$

$(a+b'+c'+d')$

$(a'+b+d')$

$(b+c+d')$

ab \ cd	00	01	11	10
00	0		0	
01	0	0		0
11	X	X	0	0
10	X	X		0

$a+b+c'+d'$

$(b'+c'+d)$

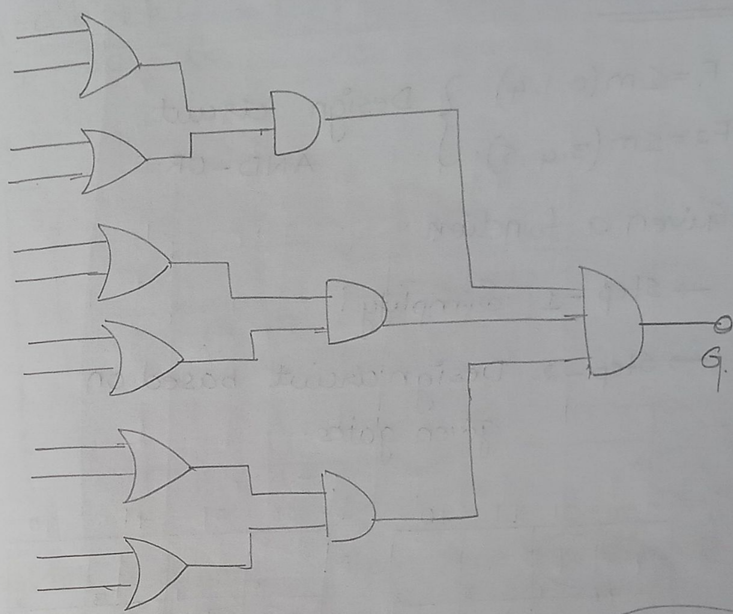
$(a'+b')$

$(a'+c'+d)$

$(b'+c)$

$c+d$

$$G = (a+b+c'+d')(b'+c'+d)(a'+b')(c+d)(b'+c)$$



* NAND and NOR Gates:

x	y	$x \cdot y$	x NAND y	x NOR Y	$x+y$
0	0	0	1	1	0
0	1	0	1	0	1
1	0	0	1	0	1
1	1	1	0	0	1

30/06/2022

* $F_1 = \Sigma m(0,1,4)$
  $F_2 = \Sigma m(3,4,5)$. } Design circuit AND-OR.

Given a function.

→ Step-1: Simplify

→ Step-2: Design circuit based on given gates.

a \ bc	00	01	11	10
0	1	1	.	
1	1	.		

F1

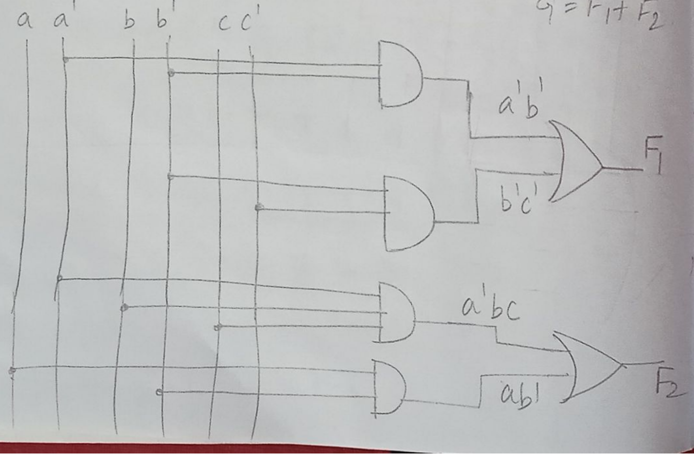
a \ bc	00	01	11	10
0	.		1	
1	1	1		

F2

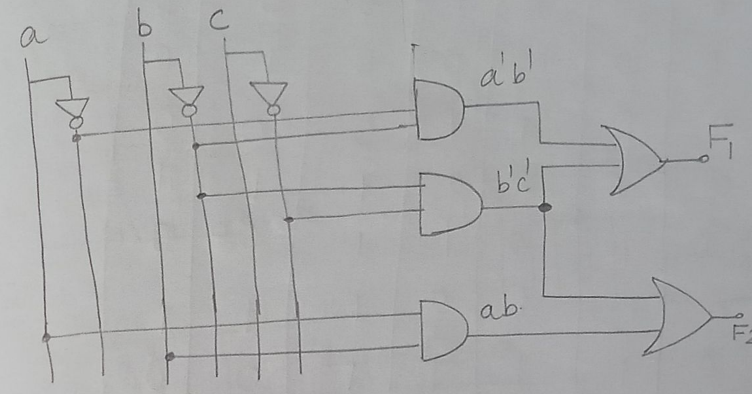
$F_1 = a'b' + b'c'$

$F_2 = a'bc + ab'$

$g = F_1 + F_2$



* $F_1 = a'b' + b'c'$
  $F_2 = b'c' + ab$.



* Consider a digital system that has ~~inputs~~ 3 outputs and takes 4 inputs

→ First O/P is assertive when b is high.

→ Second O/P is ~~assertive when~~ low when the I/P is odd.

→ Third O/P is high if both first O/P & second O/P are high.

next page

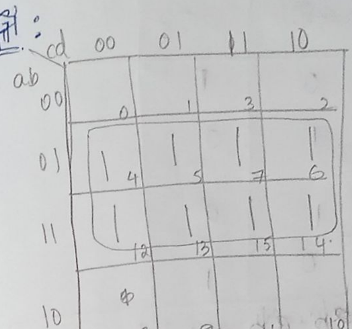
a	b	c	d	F1	F2	F3
0	0	0	0	0	1	0
0	0	0	1	0	0	0
0	0	1	0	0	1	0
0	0	1	1	0	0	0
0	1	0	0	1	1	1
0	1	0	1	1	0	0
0	1	1	0	1	1	1
0	1	1	1	1	0	0
1	0	0	0	0	1	0
1	0	0	1	0	0	0
1	0	1	0	0	1	0
1	0	1	1	0	0	0
1	1	0	0	1	1	1
1	1	0	1	1	0	0
1	1	1	0	1	1	1
1	1	1	1	1	0	0

$F1 = \Sigma m(4,5,6,7,12,13,14,15)$

$F2 = \Sigma m(0,2,4,6,8,10,12,14)$

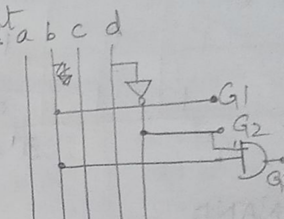
$F3 = \Sigma m(4,6,12,14)$

F1:

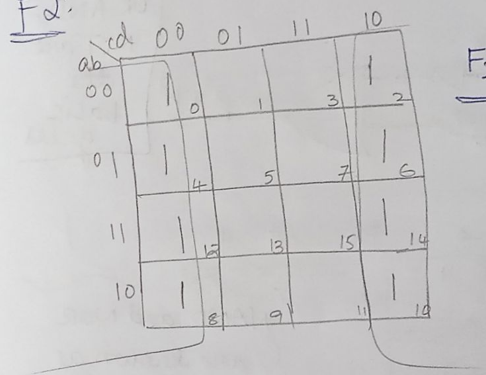


$F_1 = b$

Circuit

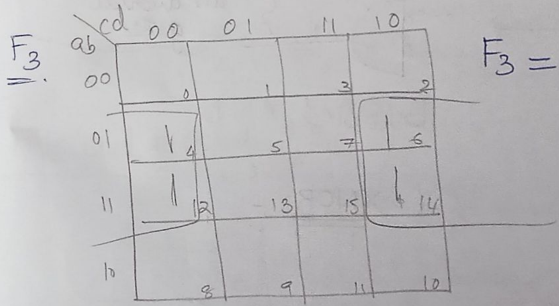


F2:



$F_2 = d'$

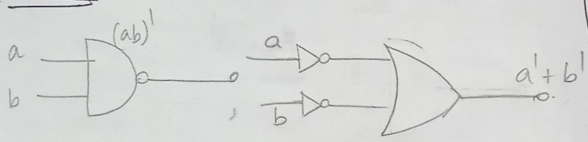
F3



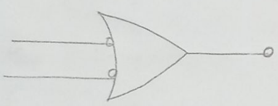
$F_3 = bd'$

a	b	NAND $(ab)'$	NOR $(a+b)'$	XOR $a \oplus b$	XNOR $a \odot b$
0	0	1	1	0	1
0	1	1	0	1	0
1	0	1	0	1	0
1	1	0	0	0	1
		$a'+b'$	$a'b'$	$ab'+a'b$	$ab+a'b'$

NAND:

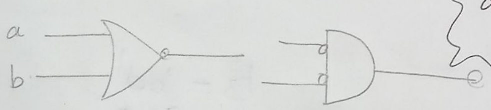


OR, AND, NOT are the basic gates



Bubbled OR

NOR:

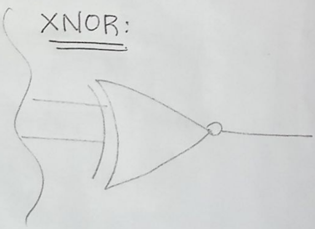


NAND and NOR are known as universal gates

Bubbled AND

XOR:

XNOR:



*

A	B	C	D	F
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	X
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	X
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

Min POS:

$F = \Sigma m(0, 2, 4, 8, 10, 13, 15) + d(6, 11)$

$F = \Pi M(1, 3, 5, 7, 9, 12, 14) + d(6, 11)$

a	b	Minterm	Maxterm
0	0	$a'b'$	$a+b$
0	1	$a'b$	$a+b'$
1	0	ab'	$a'+b$
1	1	ab	$a'+b'$

ab \ cd	00	01	11	10
00		0	0	
01		0	0	X
11	0			0
10		0	X	

$F = (a+d')(a'+b'+d)\cancel{(a'+b+d')}(b+d')$

$\boxed{F' = a'd + abd' + b'd}$

$F = (a+d')(a'+b'+d)(b+d')$

$\Updownarrow$ DUAL

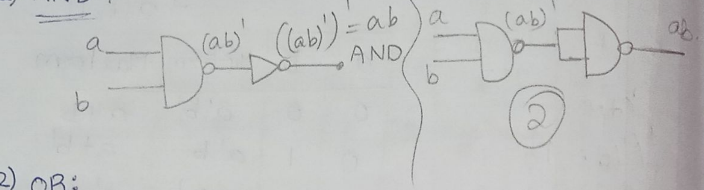
$F' = a'd + abd' + b'd$

AND $\leftrightarrow$ OR
$x \leftrightarrow x'$

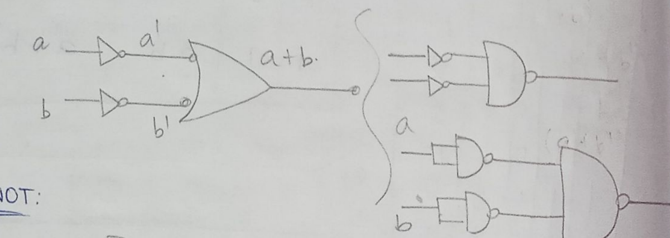
01/07/2022

* NAND Gate:

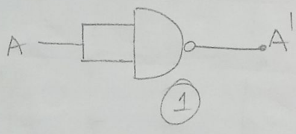
1) AND:



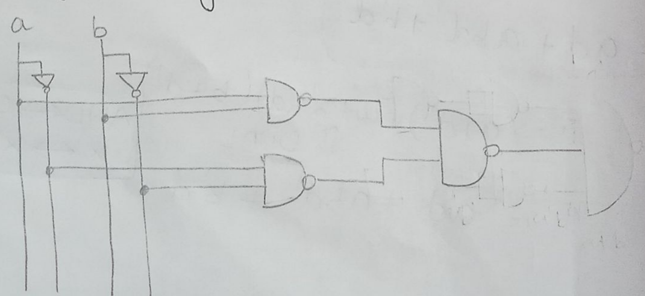
2) OR:



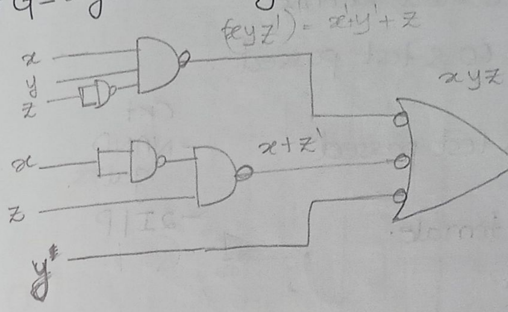
3) NOT:



* $G = ab + a'b'$ Design a circuit using NAND gates only.



* $G = xyz' + x'z + y'$



NOR ⟶ POS
NAND ⟶ SOP

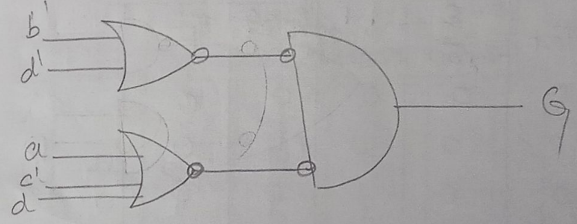
* $G = \prod M(1,3,7,9,11)$

ab \ cd	00	01	11	10
00	0	0 (1)	0 (3)	2
01	4	5	0 (7)	6
11	12	13	15	14
10	8	0 (9)	0 (11)	10

~~$G = b'd + a'bd$~~

$G = (b+d')(a+c'+d)$

$= ab + bc' + bd + ad' + c'd'$



$\Sigma m(0,2,4,5,6,8,10,12,13,14,15)$

ab \ cd	00	01	11	10
00	0	1	3	2
01	4	5	7	6
11	12	13	15	14
10				

02/07/2022

* RTA Office $\longrightarrow$ Learner's License

if age ≥ 18 & ≤ 60 & test passed.

(OR)

married & test passed

(OR)

gender = female.

(OR) and

No visual impairment

Ckt
- NAND gates
- 2I/P

a) I/P $\longrightarrow$ O/P's

b) TT

c) Simplify

d) Ckt using basic

e) Then NAND gates.

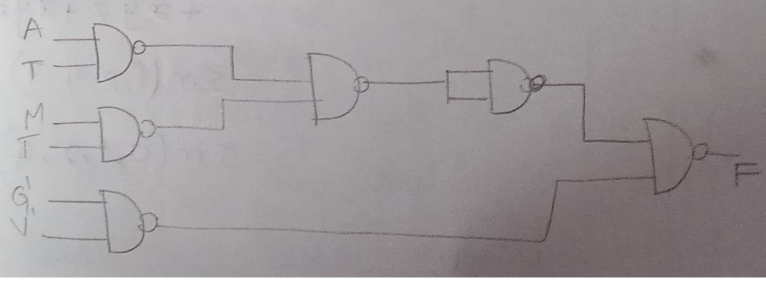
Age	Test	F=1 M=0 Gender	M=1 UM=0 Married	Yes=0 NO=1 Blind	F
0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	1	0	0
0	0	0	1	1	0
0	0	1	0	0	0
0	0	1	0	1	1
0	0	1	1	0	0
0	0	1	1	1	1
0	1	0	0	0	0
0	1	0	0	1	0

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

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

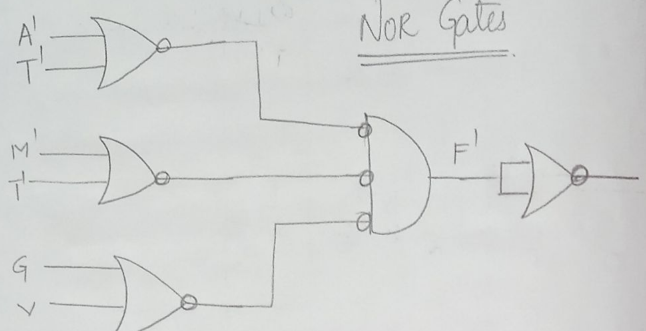
$F = A \cdot T + M \cdot T + G'V'$

Nand gates



$POS(F) = [(A'+T')(M'+T')(G+V)]'$

NOR Gates



* Design a circuit that has 2 outputs; 1^{st} output ~~if~~ is 1 iff the no. of 1's in the input are odd. 2^{nd} output is 1 iff there are atleast 2 zeroes in the input

* 3 variable input.

x	y	z	F_1	F_2
0	0	0	0	1
0	0	1	1	1
0	1	0	1	1
0	1	1	0	0
1	0	0	1	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	0

~~$F_1 = x'y'z + xyz$~~

$F_1 = x'y'z + xy'z + xy'z' + xyz$

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

$F_2 = \Sigma m(0,1,2,4)$

⇒ XOR. odd function

F_1

a \ bc	00	01	1ϕ	10
0		1		1
1	1		1	

$F_1 = x'y'z + x'yz' + xy'z' + xyz$

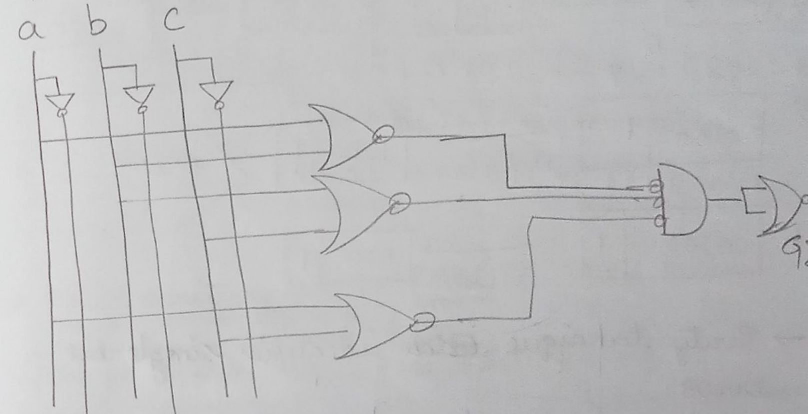
F_2

a \ bc	00	01	11	10
0	1	1		1
ϕ	1			

$F_2 = x'y' + y'z' + x'z'$

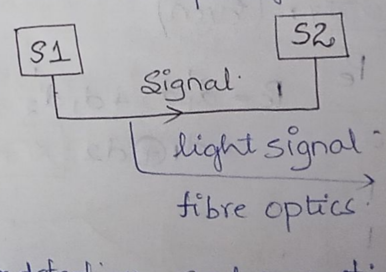
x \ yz				
	1		1	
		1		1

⇒ XNOR even function



04/07/2022

* Parity: — → Even parity
→ Odd parity



• Error detection and correction codes.

If the message sent by the sender has any errors it is sent back to the sender and do the necessary changes and re-send the message to the receiver. This is done using error detection code.

* Parity is a technique which acts as detection code which can be used only for one bit change.

1) data P E(21's) — $\boxed{0101|0} \longrightarrow$ E(21's) $\boxed{0101|0}$ | $\boxed{0101|0} \longrightarrow \boxed{1101|0}$ × error

2) data P — $\boxed{0100|1} \longrightarrow \boxed{0110|\phi}$ × error

$\boxed{0100|1} \longrightarrow \boxed{0100|1}$

→ Parity technique ~~[illegible]~~ identifies single bit error.

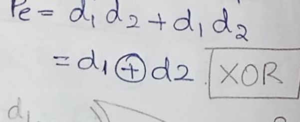
* Parity Generator: ~~Receiver side~~ Sender side
* Parity Checker: Receiver side.

⇒ 2 bit parity generator: (Even)

d_1	d_2	P_e
0	0	0
0	1	1
1	0	1
1	1	0

$P_e = d_1' d_2 + d_1 d_2'$

$= d_1 \oplus d_2$ $\boxed{\text{XOR}}$



⇒ 2 bit parity checker: (Even)

d_1	d_2	P_e	P_{ec}
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
ϕ	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

1 – true
0 – false.

$P_{ec} = d_1' d_2 d_3 + d_1' d_2' d_3'$
$+ d_1 d_2' d_3 + d_1 d_2 d_3'$

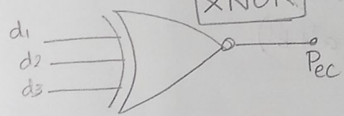
$= d_1'(d_2 d_3 + d_2' d_3')$
$+ d_1(d_2' d_3 + d_2 d_3')$

$= d_1'(d_2 \odot d_3) + d_1(d_2 \oplus d_3)$

$\Sigma m(0,3,5,6)$

a \ bc				
	1		1	
		1		1

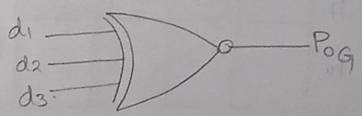
$\boxed{\text{XNOR}}$



⇒ 3 bit parity generator: (Odd)

d_1	d_2	d_3	P_{og}
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

$P_{OG} = \Sigma m(0,3,5,6)$
↳ even function



$\boxed{\text{XNOR}}$

* $F = ab + a'b'$

$F^D = \cancel{(a'+b')\cdot(a+b)} = (a+b)\cdot(a'+b')$

$F' = (ab + a'b')'$

$= (ab)' \cdot (a'b')'$

$= (a'+b')(a+b).$

F' = Complement each literal in F^D

F^D = change operator

AND $\longrightarrow$ OR

OR $\longrightarrow$ AND.

if constant $\longrightarrow$ complement.

* Find F^D of $a'bc + ab'c' + ab$

$F^D = (a'+b+c)(a+b'+c')(a+b).$

$F' = (a+b'+c')(a'+b+c)(a'+b')$

*

ab \ cd	00	01	11	10
00		0		0
01		0	0	0
11		0	0	
10				

F = SOP.

$= \Sigma m(0,3,4,8,9,10,11,12,14)$

$F' = \Sigma m(1,2,5,6,7,13,15)$

$F = \Pi M(1,2,5,6,7,13,15)$

$F' = \Pi M(0,3,4,8,9,10,11,12,14)$

for SOP: $F = 1$; $F' = 0$

for POS: $F = 0$; $F' = 1$

~~F = bd~~

$F = (b'+d)(a+c+d')(a+c$

$F = bd + a'c'd + a'cd'$

08/07/2022

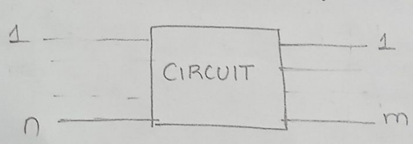
UNIT-3

COMBINATIONAL CIRCUITS:

* **Design Procedure:**

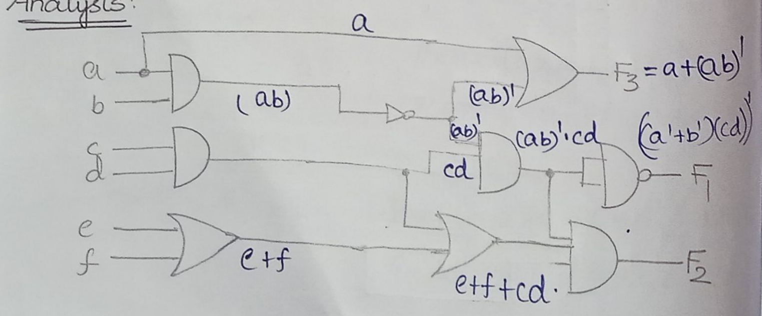
1) Identifying I/Ps and O/Ps & assign them label.
2) Draw the Truth Table.
3) Derive Boolean functions
4) Simplify (Reduce no. of literals) function.
5) Using gates provided; design the circuit diagram.

→ O/P is dependant on given I/P.



no. of inputs need not be equal no. of outputs.

Analysis:



$F_3 = a + a' + b' = b' + 1$ $\boxed{\cancel{F_3 = b'}}$ $\boxed{F_3 = 1}$

$F_1 = (a'+b')' + (cd)'$

$\boxed{F_1 = ab + (cd)'}$

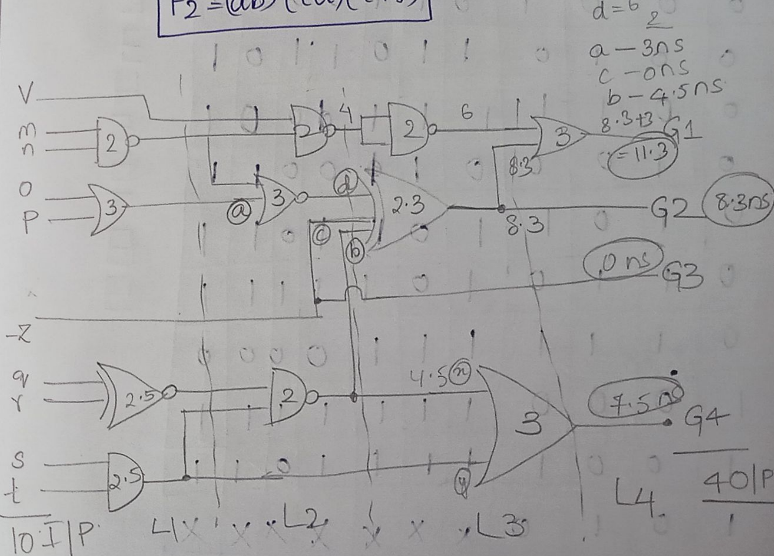
$F_2 = ((ab)' \cdot cd)(e+f+cd)$

$= \cancel{(ab)'e + (ab)'f + (ab)'cd}$

$= (ab)'cde + (ab)'cdf + (ab)'cd$

$F_2 = (ab)'cd(e+f+1) = (ab)' \cdot (cd)(e+f)$

$\boxed{F_2 = (ab)'(cd)(e+f)}$



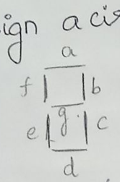
Propagation Delay

Nand – 2ns
Nor – 3ns
And – 2.5ns
OR – 3ns
EXOR = 2.3Ns

ExNOR – 2.5ns

$G_4 = [(q \cdot r)st]' + st$

* Design a circuit to lit 7-segment display.

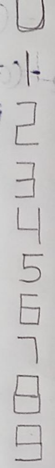


as 7-segment display 0-9 ~~bits~~

I/P size: 4 bits

let m, n, o, p → be I/Ps

m	n	o	p	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	x	x	x	x	x	x	x
1	0	1	1	x	x	x	x	x	x	x
1	1	0	0	x	x	x	x	x	x	x
1	1	0	1	x	x	x	x	x	x	x
1	1	1	0	x	x	x	x	x	x	x



1	1	1	1	x	x	x	x	x	x	x

$a = \Sigma m(0,2,3,5,6,7,8,9) + d(10,11,12,13,14,15)$

$b = \Sigma m(0,1,2,3,4,7,8,9) + d(10,11,12,13,14,15)$

$c = \Sigma m(0,1,3,4,5,6,7,8,9) + d(10,11,12,13,14,15)$

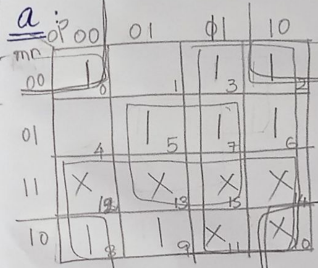
$d = \Sigma m(0,2,3,5,6,8,9) + d(10,11,12,13,14,15)$

$e = \Sigma m(0,2,6,8) + d(10,11,12,13,14,15)$

$f = \Sigma m(0,4,5,6,8,9) + d(10,11,12,13,14,15)$

$g = \Sigma m(2,3,4,5,6,8,9) + d(10,11,12,13,14,15)$

a:

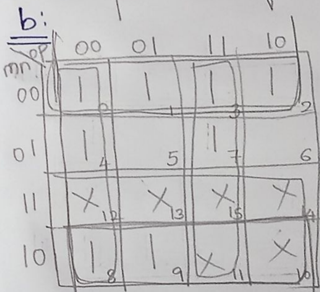


$\cancel{a = x'z + x'y'w' + xyw + x}$

$\cancel{= x + z + x'y'w' + yw}$

$\boxed{a = n'p' + m + o + np}$

b:

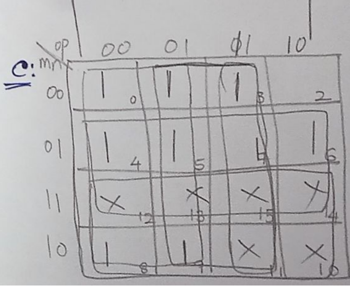


$\cancel{b = o'p'}$

$b = n' + m + \cancel{mn'} + o'p' + op$

$\boxed{b = n' + m + o'p' + op}$

c:



$c = m + n + p + op$

$\boxed{c = m + n + p + o'}$