NOR – POS
NAND – SOP

UNIT-2 (1–25)

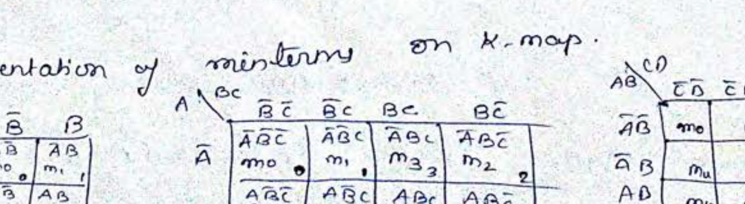
Minimization of switching functions:

– The map method was proposed by Veitch → This provides a simple straight forward procedure for simplification of Boolean fun.

The method is called Veitch Diag (or) Karnaugh map – in 1952. It is a pictorial representation of a T.T (or) an Extension of venn diagram, & this is modified by M. Karnaugh in 1953. (K-map / Karnaugh map).

– K-map can be implemented up to 5 (or) 6 variables but beyond it can be solved by using Quine-McCluskey method.

outline of 1, 2, 3, 4 variable K-map.



Representation of minterms on K-map.

A \ B	$\bar{B}$	B
$\bar{A}$	$\bar{A}\bar{B}$ m_0 0	$\bar{A}B$ m_1 1
A	$A\bar{B}$ m_2 2	AB m_3 3

2 variable.

A \ BC	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
$\bar{A}$	$\bar{A}\bar{B}\bar{C}$ m_0 0	$\bar{A}\bar{B}C$ m_1 1	$\bar{A}BC$ m_3 3	$\bar{A}B\bar{C}$ m_2 2
A	$A\bar{B}\bar{C}$ m_4 4	$A\bar{B}C$ m_5 5	ABC m_7 7	$AB\bar{C}$ m_6 6

3 variable

AB \ CD	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	m_0	m_1	m_3	m_2
$\bar{A}B$	m_4	m_5	m_7	m_6
AB	m_{12}	m_{13}	m_{15}	m_{14}
$A\bar{B}$	m_8	m_9	m_{11}	m_{10}

4 variable

Representation of Max terms

A \ B	B (0)	$\bar{B}$ (1)
A 0	$A+B$ M_0	$A+\bar{B}$ M_1
$\bar{A}$ 1	$\bar{A}+B$ M_2	$\bar{A}+\bar{B}$ M_3

2 variable

A \ BC	00 $B+C$	01 $B+\bar{C}$	11 $\bar{B}+\bar{C}$	10 $\bar{B}+C$
0 A	000 M_0	001 M_1	011 M_3	010 M_2
1 $\bar{A}$	100 M_4	101 M_5	111 M_7	110 M_6

3 variable.

A	0	1
	M_0	M_1

1-variable

plotting a T.T on K-map

1.

A	B	F
0	0	0
0	1	1
1	0	1
1	1	0

A \ B	0	1
0	0_0	1_1
1	1_2	0_3

$F(A,B) = \sum m(1,2)$

	X 1	0
1	0	1
0	1	0

$F(A,B) = \prod M(1,2)$

plotting SSOP exp on K.map.

Ex: $F = \bar{A}\bar{B}\bar{C} + \bar{A}BC + AB\bar{C}$

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

plotting SPOS Exp on K-map

$f = (A+B+C)(A+\bar{B}+C)(\bar{A}+B+C)(A+B+\bar{C}) = \prod M(0,1,2,5)$

	$B+C$ $\bar{B}\bar{C}$	$B+\bar{C}$ $\bar{B}C$	$\bar{B}+\bar{C}$ BC	$\bar{B}+C$ $B\bar{C}$
1 A	0_0	0_1	1_3	0_2
0 $\bar{A}$	1_4	0_5	1_7	1_6

→ Max terms

Grouping of cells for minimal exp:

- Grouping horizontally
- Grouping vertically

- Ex: pairing, quads, octets & so on.

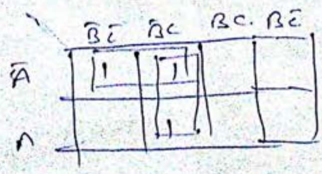
summary of literals eliminated for n-variable K-map.

No. of 1's grouped	No. of literals Eliminated	No. of literals present in resulting term.
$64 (2^6)$	6	$n-6$
$32 (2^5)$	5	$n-5$
$16 (2^4)$	4	$n-4$
$8 (2^3)$	3	$n-3$
$4 (2^2)$	2	$n-2$
$2 (2^1)$	1	$n-1$
$1\ 2^0$	0	$n-0 = n$

Ex: 3 variable K-map.

- No. of 1's grouped: $2^3 = 8.\ 2. \Rightarrow 2^1$.
- No. of literals eliminated: 1
- No. of literals present in resulting term $= n-1$
$= 3-1 = 2$

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



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

Prime Implicants & essential prime implicants:

↓ set of adjacent min term / simplified product term obtained by the minterms of set.

P.I → Implicant is called P.I if it is not a subset of another implicant of the fun.

E.P.I → A P.I which includes a '1' cell that is not included in any other P.I

Ex: 1) obtain E.P.I of $f(A,B,C) = \Sigma m(1,3,4,5)$

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

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

all 1's are covered — E.P.I

so $\bar{B}\bar{C}$ is not required.

2) $f(A,B,C,D) = \Sigma m(0,5,7,8,9,10,11,14,15)$

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

$= A\bar{B} + AC + \bar{C}\bar{D}\bar{B} + \bar{A}BD \cancel{+ BCD}$

BCD is covered by $\bar{A}B$ & AC.

∴ E.P.I are all the 4.

Ex: 1. $Y = A\bar{B}C + \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}\bar{C} + \bar{A}\bar{B}\bar{C}$

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

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

Minimal pos form using K-maps:

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

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

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

$\bar{Y}_C$ $(\bar{B})$

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

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

Conversion of sop to pos using K-map:

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

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

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

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

2) $f = \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + AB\bar{C} + ABC$ into pos form

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

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

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

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

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

3) conversion of pos to sop form using K-map.

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

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

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

Dont care combinations:

Ex: 1.

A	B	C	f
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	x
1	1	0	x
1	1	1	x

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

$x <^{0}_{1}$

$f = \bar{B}$

2. $f(A,B,C) = \sum m(0,1,5) + \sum d(4,7)$

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

$f = \bar{B}$

3. Reduce K-map in pos form

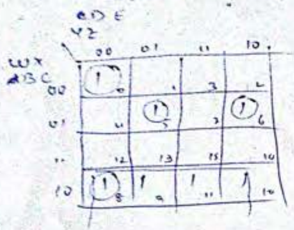
$f(A,B,C,D) = \prod M(0,2,6,8,12) + \prod d(3,4,7,10,14)$

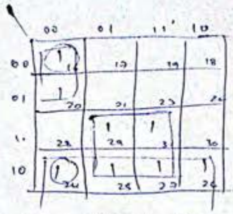
	$\bar{C}\bar{D}$	$\bar{C}D$	CD	$C\bar{D}$
$\bar{A}\bar{B}$	0 ₀	₁	x ₃	0 ₂
$\bar{A}B$	x ₄	₅	x ₇	0 ₆
AB	0 ₁₂	₁₃	₁₅	x ₁₄
$A\bar{B}$	0 ₈	₉	₁₁	x ₁₀

$f : \bar{D}$

$f = D$

5- Variable K-map:





$V=0 = \bar{A}$ $\qquad$ $V=1 = A$

Ex: $f(A,B,C,D,E) = \Sigma m(0,5,6,8,9,10,11,16,20,24,25,26,27,29,31)$

a: $(0,8,16,24) = \bar{C}\bar{D}\bar{E}$

b: $(8,9,10,11,24,25,26,27) : B\bar{C}$

c: $(25,27,29,31) : ABE$

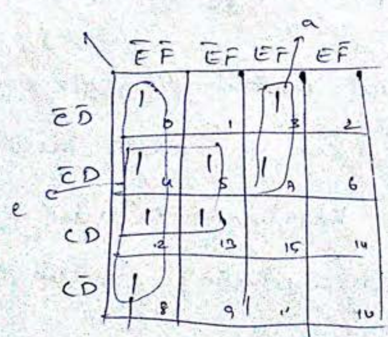
d: $5 \Rightarrow \bar{A}\bar{B}C\bar{D}E$

e: $6 \Rightarrow \bar{A}\bar{B}CD\bar{E}$

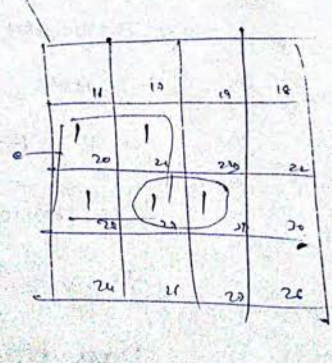
f: $16,20 \Rightarrow A\bar{B}\bar{D}\bar{E}$

$F = B\bar{C} + \bar{C}\bar{D}\bar{E} + ABE + A\bar{B}\bar{D}\bar{E} + \bar{A}\bar{B}C\bar{D}E + \bar{A}\bar{B}CD\bar{E}$

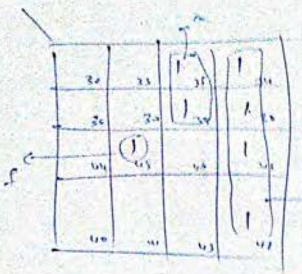
6- Variable K-map: $f = \Sigma m(0,3,4,5,7,8,12,13,20,21,28,29,31,34,35,38,39,42,45,46,50,54,58,61,62,63)$.

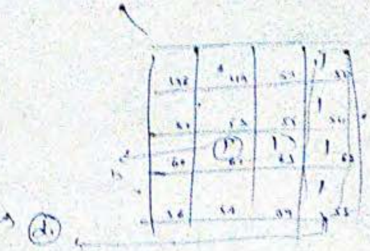


$A=0 \quad B=0$



$A=0 \quad B=1$





A = 1   B = 1

A = 1   B = 0

A = x_1, B = x_2, C = x_3, D = x_4, E = x_5, F = x_6

a : 3, 7, 35, 39 . $\bar{B}\bar{C}EF$ = $\bar{x}_2\bar{x}_3x_5x_6$

b : 29, 31, 61, 63 = $BCD\cdot F$ = $x_2x_3x_4x_6$

c : = 0, 4, 8, 12. = $\bar{A}\bar{B}\bar{E}\bar{F}$ = $\bar{x}_1\bar{x}_2\bar{x}_5\bar{x}_6$

d. : 34, 38, 42, 46, 50, 54, 58, 62. = $AE\bar{F}$ = $x_1x_5\bar{x}_6$

e : 4, 5, 12, 13, 20, 21, 28, 29 . $\bar{E}D\bar{A}$ = $\bar{x}_1x_4\bar{x}_5$

f : 45, 61, 13, 29. = $\bar{E}FCD$ = $\bar{x}_5x_6x_3x_4$

F = a+b+c+d+e+f

Adv & disadv of K-map:

adv : 1. fast method for simplifying expression upto 4 variables

2. It gives visual method of Logic simplification.

3. P.I & E.P.I can be easily identified.

4. This principle has been extended to VLSI design by incorporating it in the pass transistor logic design method.

5. suitable for both sop & pos forms for reduction.

disadv:

1. More the 4-variables it is difficult to execute.
2. It is not suitable for computer reduction.

* Tabular Method: Quine - Mc Cluskey Method:

- Binary representation Method.
- Decimal representation Method.

1. Binary representation Method:

Ex: $f(A,B,C,D) = \Sigma m(0,2,3,6,7,8,10,12,13)$

1. list all the min terms in binary form & arrange them in the form of groups according to the no. of 1's contained

min terms	A	B	C	D	No. of 1's.
		4	2	1	
m_0	0	0	0	0	0
m_2	0	0	1	0	1
m_3	0	0	1	1	2
m_6	0	1	1	0	2
m_7	0	1	1	1	3
m_8	1	0	0	0	1
m_{10}	1	0	1	0	2
m_{12}	1	1	0	0	2
m_{13}	1	1	0	1	3

Index	Minterm	Binary rep ABCD
0	m_0 ✓	0000
1	m_2 ✓	0010
	m_8 ✓	1000
2	m_3 ✓	0011
	m_6 ✓	0110
	m_{10} ✓	1010
	m_{12} ✓	1100
3	m_7 ✓	0111
	m_{13} ✓	1101

2. compary each binary term with every term in the adjacent next higher category.

If they differ only by one position put a Check mark and copy the term in to the next coloumn, with '-' in the place where the variable is unmatched. continue untill no Elemination of literals

(i)

min term	Binary Representation.
m(0,2) ✓	0 0 - 0
m(0,8) ✓	- 0 0 0
m(2,3) ✓	0 0 1 -
m(2,6) ✓	0 - 1 0
m(2,10) ✓	- 0 1 0
m(8,10) ✓	1 0 - 0
m(8,12) P	1 - 0 0
m(3,7) ✓	0 - 1 1
m(6,7) ✓	0 1 1 -
m(12,13) Q	1 1 0 -

(ii)

m(0,2,8,10)	- 0 - 0	R
m(2,3,6,7)	0 - 1 -	S.

3. ~~each~~ P.I are P, Q, R, S.

4. List all P.I in P.I table.

P.I	Binary rep A B C D	Literal Rep.
m(8,12)	1 _ 0 0	$A\bar{C}\bar{D}$
m(12,13)	1 1 0 _	$AB\bar{C}$
m(0,2,8,10)	_ 0 _ 0	$\bar{B}\bar{D}$
m(2,3,6,7)	0 _ 1 _	$\bar{A}C$.

4. Now select EPI / the min no. of P.I which must cover all the minterms using P.I chart (PIC)

Name	P.I	literal Rep	m_0	m_2	m_3	m_6	m_7	m_8	m_{10}	m_{12}	m_{13}
P	8,12	$A\bar{C}\bar{D}$						x		x	
Q	12,13	$AB\bar{C}$								x	x
R	0,2,8,10	$\bar{B}\bar{D}$	x	x				x	x		
S	2,3,6,7	$\bar{A}C$.		x	x	x	x				

Here $m_0, m_3, m_6, m_7, m_{10}, m_{13}$ contain only 1 'x' & Their corresponding P.I must be included in minimal fun & they are called EPI

∴ Here Q, R, S are E.P.I

These 3 EPI are covering all minterms

∴ minimal expressions is

$f = Q + R + S$

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

21

Ex: 2: $f(A,B,C,D) = \Sigma m(0,1,2,5,7,8,9,10,13,15)$.

1.

min terms	Binary A B C D (8 4 2 1)	No. of 1's
m_0	0 0 0 0	0
m_1	0 0 0 1	1
m_2	0 0 1 0	1
m_5	0 1 0 1	2
m_7	0 1 1 1	3
m_8	1 0 0 0	1
m_9	1 0 0 1	2
m_{10}	1 0 1 0	2
m_{13}	1 1 0 1	3
m_{15}	1 1 1 1	4.

Index	min terms	Binary ABCD
0	m_0	0 0 0 0 ✓
1	m_1	0 0 0 1 ✓
	m_2	0 0 1 0 ✓
	m_8	1 0 0 0 ✓
2	m_5	0 1 0 1 ✓
	m_9	1 0 0 1 ✓
	m_{10}	1 0 1 0 ✓
3	m_7	0 1 1 1 ✓
	m_{13}	1 1 0 1 ✓
4	m_{15}	1 1 1 1 ✓

2.

minterms	Binary ABCD
m(0,1)	0 0 0 – ✓
m(0,2)	0 0 – 0 ✓
m(0,8)	– 0 0 0 ✓
m(1,5)	0 – 0 1 ✓
m(1,9)	– 0 0 1 ✓
m(2,10)	– 0 1 0 ✓
m(8,9)	1 0 0 – ✓
m(8,10)	1 0 – 0 ✓
m(5,7)	0 1 – 1 ✓
m(5,13)	– 1 0 1 ✓
m(9,13)	1 – 0 1 ✓
m(7,15)	– 1 1 1 ✓
m(13,15)	1 1 – 1 ✓

minterm		Binary ABCD
m(0,1,8,9)	P	– 0 0 –
m(0,2,8,10)	Q	– 0 – 0
m(0,8,1,9)		– 0 0 –
m(0,8,2,10)		– 0 – 0
m(1,5,9,13)	R	– – 0 1
m(1,9,5,13)		– – 0 1
m(5,7,13,15)	S	– 1 – 1
m(5,13,7,15)		– 1 – 1

min terms Prime Implicants	ABCD Binary representation	Literal Representation
$m(0,1,8,9)$ P	$-00-$	$\bar{B}\bar{C}$
$m(0,2,8,10)$ Q	$-0-0$	$\bar{B}\bar{D}$
~~m(0,8,2,10) C.R~~	~~edelco~~	
$m(1,5,9,13)$ R	$--01$	$\bar{C}D$
$m(5,7,13,15)$ S	$-1-1$	BD

4) EPI → no. of PI which must cover all minterms from PIC shown:

P.I	PI minterms	literal	m_0	m_1	m_2	m_5	m_7	m_8	m_9	m_{10}	m_{13}	m_{15}
P	0,1,8,9	$\bar{B}\bar{C}$	×	×				×	×			
Q	0,2,8,10	$\bar{B}\bar{D}$	×		×			×		×		
R	1,5,9,13	$\bar{C}D$		×		×			×		×	
S	5,7,13,15	BD				×	×				×	×

$\left.\begin{array}{l} m_2 \text{ \& } m_{10} \text{ is covered by only one p.I } Q \\ m_5 \text{ \& } m_{15} \text{ from p.I } S \end{array}\right\} \rightarrow EPI$

By using Q & S the minterms covered are

0, 2, 8, 10, 5, 7, 13, 15

The two EPI cover all min terms except 1, & 9

these can be covered by using either P (or) R.

∴ minimal Expression $f = P + Q + S$ (or) $R + Q + S$.

∴ $f = \bar{B}\bar{C} + \bar{B}\bar{D} + BD$ (or) $\bar{B}\bar{D} + \bar{C}D + BD$

(ii)

(7)

(b) Implicant : 0000
0010
$\Rightarrow$ 00–0 → This term is called implicant.

dy: from the above the implicant can be used to express the two other terms.

(ie) $\bar{A}\bar{B}\bar{D}$ is used instead $\bar{A}\bar{B}\bar{C}\bar{D}+\bar{A}\bar{B}C\bar{D}$

prime implicants : The terms that did not match during the process is called P.I.

Ex: McCluskey method :

F(A,B,C,D) =

Y(w,x,y,z) = $\Sigma m(1,2,3,5,9,12,14,15) + \Sigma d(4,8,11)$

1) minterms Binary rep Index minterm Binary Rep.

minterms	Binary rep 8 4 2 1	Index	minterm	Binary Rep. 8 4 2 1
1	0001 ✓	1	1 ✓	0001
2	0010 ✓		2 ✓	0010
3	0011		4 ✓ 8	0100
5	0100 ✓			1000
9	1001	2	3 ✓	0011
12	1100		5 ✓	0101
14	1110		9 ✓	1001
15	1111		12 ✓	1100
d 4	0100	3	11 ✓	1011
8	1000		14 ✓	1110
11	1011	4	15 ✓	1111

2) compare :

minterms	Rep:			minterm	Binary
(1,3)	00–1 ✓	(3,11)	–011 ✓	1,3,9,11	–0–1
(1,5)	0–01	(9,11)	10–1 ✓	1,9,3,11	–0–1
(1,9)	–001 ✓	(12,14)	11–0		
(2,3)	001–	(11,15)	1–11		
(4,5)	010–	(14,15)	111–		
(4,12)	–100				
(8,9)	100–				
(8,12)	1–00				

List of P.I

P.I	min	Binary rep.
$\bar{A}\bar{C}D$	1,5	0-01
$\bar{A}\bar{B}C$	2,3	001-
$\bar{A}B\bar{C}$	4,5	010-
$B\bar{C}\bar{D}$	4,12	-100
$A\bar{B}\bar{C}$	8,9	100-
$A\bar{C}\bar{D}$	8,12	1-00
$AB\bar{D}$	12,14	11-0
ACD	11,15	1-11
ABC	14,15	111-
$\bar{B}D$	1,3,9,11	-0-1

P.I selection table.

P.I	m_1	m_2	m_3	m_4 (d)	m_5	m_6	m_7	m_8 (d)	m_9	m_{10}	m_{11} (d)	m_{12}	m_{13}	m_{14}	m_{15}
(1,5) $\bar{A}\bar{C}D$	x				x										
2,3 $\bar{A}\bar{B}C$		x → E.P.I													
4,5 $\bar{A}B\bar{C}$				x	x										
4,12 $B\bar{C}\bar{D}$				x								x			
8,9 $A\bar{B}\bar{C}$								x	x						
8,12 $A\bar{C}\bar{D}$								x				x			
12,14 $AB\bar{D}$												x		x	
11,15 ACD											x				x
14,15 ABC														x	x
1,3,9,11 $\bar{B}D$	x		x						x		x				

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

Ex: $f(A,B,C) = \sum m(0,1,2,5,6,7)$

min terms	Binary rep	min terms	min terms	Binary rep	
0	000	0 ✓	0,1	00−	= P
1	001	1 ✓	0,2	0−0	Q
2	010	2 ✓	1,5	−01	R
5	101	5 ✓	2,6	−10	S
6	110	6 ✓	5,7	1−1	T
7	111	7 ✓	6,7	11−	U

P·I selection table:

P·I		m_0	m_1	m_2	m_5	m_6	m_7
$\bar{A}\bar{B}$ P	0,1	•	•				
$\bar{A}\bar{C}$ Q	0,2	•		•			
$\bar{B}C$ R	1,5		•		•		
$B\bar{C}$ S	2,6			•		•	
AC T	5,7				•		•
AB U	6,7					•	•

* when each coloumn in P·I selection chart has two (or) more dots, the chart is known as cyclic prime Implicant chart.
* proceed by trial & error method.

one of the sol is $\bar{A}\bar{B} + B\bar{C} + AC$.

* P·I which are not selected i.e. (0,2),(1,5)(6,7) are called redundant P·I.

Petrick's method of determining irredundant Exp:

— from above Ex:

Exp covering a P·I table is represented by.

$$e = \underset{m_0}{(P+Q)}\ \underset{m_1}{(P+R)}\ \underset{m_2}{(Q+S)}\ \underset{m_5}{(R+T)}\ \underset{m_6}{(S+U)}\ \underset{m_7}{(T+U)}$$

$$= (P+QR)(S+QR)(U+ST)$$

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

$$= (PS + PQR + SQR + QR)(U+ST)$$

$$= PSU + PST + PQRU + PQRST + SQRU + SQRST + QRU + QRST$$

51

PSQ

= PSU + PST + QRU + QRST

$f_1(A,B,C,D) = P+S+U$

$f_2(A,B,C,D) = P+S+T$

$f_3(A,B,C,D) = Q+R+U$

minimal sum is => $f_1+f_2+f_3$

adv & dis of Quine - Mc cluskey method:

- This method is algorithmic in nature
- It is lengthy & time consuming as no. of variables ↑.

Decimal Method for obtaining P.I:

Ex: $f = \Sigma m(2,4,5,9,12,13)$

minterms arrangement

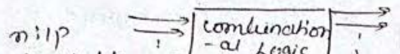
	8421					
2	0010 ✓	(1) 4,5	010- ✓	4,5,12,13	-10-	(1,8)
4	0100 ✓	(8) 4,12	-100 ✓	4,12,5,13	-10-	
5	0101 ✓	(8) 5,13	-101 ✓			
9	1001 ✓	(4) 9,13	1-01			
12	1100 ✓	(1) 12,13	110- ✓			
13	1101 ✓					

P.I	P.I in - form	
2	0010	$\bar{A}\bar{B}C\bar{D}$
9,13 (4)	1-01	$A\bar{C}D$
4,5,12,13 (1,8)	-10-	$B\bar{C}$

combinational logic design

when logic gates are connected together to produce a specified o/p for certain specified combinations of i/p variables with no storage involved, the resulting circuit is called combinational logic. (here o/p variables are at all times dependent on i/p variables).

Block dia: of combinational ckt.

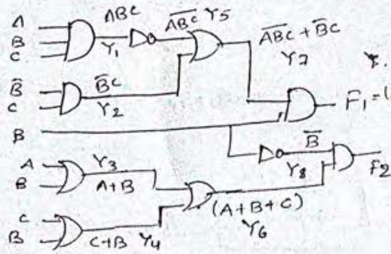


procedure to determine Boolean fun for o/p's of ckts from given ckt:

1. Make sure that given ckt is $\underset{\Downarrow}{\text{comb ckt}}$ & not seque ckt.
   has logic gates with
   no feedback path (or)
   memory element.

2. label all gate o/p's that are a fun of i/p variables with arbitrary sym & determine Boolean fun for each gate o/p.
3. label the gates that are a fun of i/p variables & previously labeled gate & determine the B.fun for them.
4. Repeat step 3 until B.fun for o/p of the ckt obtained.
5. finally sub previously defined B.fun, obtain the o/p of B.fun in term of i/p variables.

Q. obtain B. fun for o/p's of.



$F_1 = (\overline{ABC}+\bar{B}C)B = (\bar{A}+\bar{B}+\bar{C}+\bar{B}C)B = (\bar{A}+\bar{C})B$

F_2 $(A+B+C)\bar{B} = (A+C)\bar{B}$

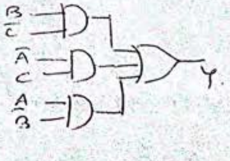
T.T:

A	B	C	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7	Y_8	F_1	F_2
0	0	0	0	0	0	0						
0	0	1	0	1	0	1						
0	1	0	0	0	1	1						
0	1	1	0	0	1	1						
1	0	0	0	0	1	0						
1	0	1	0	1	1	1						
1	1	0	0	0	1	1						
1	1	1	1	0	1	1						

simplified Boolean fun: for $f(A,B,C) = \Sigma m(1,2,3,4,5,6)$

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

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

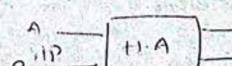


Adders:

A B S C.
0+0 : 0
0+1 : 1
1+0 : 1
1+1 : 10 (C S)

T.T

Half adder:



Block schematic

K-map:

sum:

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

carry

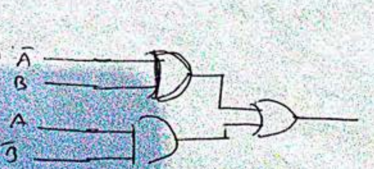
	0	0
	0	1

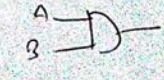
Sum : $\bar{A}B + A\bar{B}$

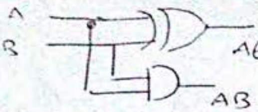
$\bar{A}B + A\bar{B}$

$C = AB$.

logic diagram







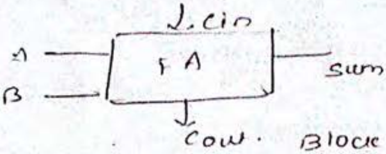
$A \oplus B = \bar{A}B + A\bar{B}$: Sum

AB = carry.

<u>Limitations</u>: 3 bit addition is not possible.

<u>Full adder</u>:

i/p's			o/p's	
A	B	Cin	S	Cout
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



Block schematic

K-map:

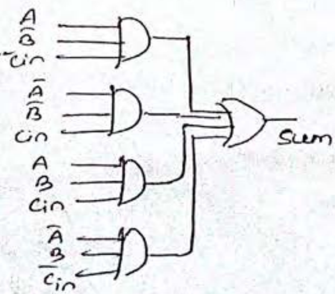
A \ BCin	$\bar{B}\bar{C}$	$\bar{B}C$	BC	$B\bar{C}$
$\bar{A}$	0	1	0	1
A	1	0	1	0

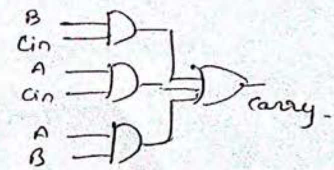
Sum $= A\bar{B}\bar{C}in + \bar{A}\bar{B}Cin + ABCin + \bar{A}B\bar{C}in$

0	0	1	0
0	1	1	1

carry $= BCin + A\bar{B}Cin + AB$.

<u>logic dia</u>:





Simplification of sum:

$$A\bar{B}\bar{C}in + \bar{A}\bar{B}Cin + ABCin + \bar{A}B\bar{C}in$$

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

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

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

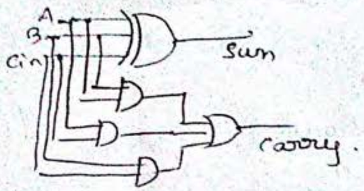
$$= ACin + \bar{B}Cin + \bar{A}B\bar{C}in$$

$$\Rightarrow Cin[A\bar{B}+\bar{A}B] + \bar{C}in[\bar{A}B+A\bar{B}]$$

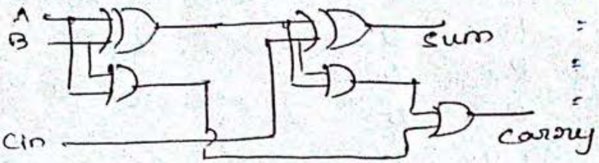
$$\Rightarrow Cin[A\odot B] + \bar{C}in[A\oplus B]$$

$$\Rightarrow Cin[\overline{A\oplus B}] + \bar{C}in[A\oplus B]$$

$$\Rightarrow Cin \oplus (A\oplus B)$$



Implementation of F.A using two H.A's & OR gate:



$= BC_{in} + AC_{in} + AB$

$= AB + AC_{in} + BC_{in}(A + \bar{A})$

$= AB + AC_{in} + ABC_{in} + \bar{A}BC_{in}$

$= AC_{in} + \bar{A}BC_{in} + AB$

$= AB + AC_{in}(B + \bar{B}) + \bar{A}BC_{in}$

carry $= \underline{AB + ABC_{in}} + A\bar{B}C_{in} + \bar{A}BC_{in}$

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

Subtractors:

$0 - 0 = 0$
$0 - 1 = 1 \quad B = 1$
$1 - 0 = 1$
$1 - 1 = 0$

Half subtractor:

i/p's		o/p's	
A	B	Diff	Borrow
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Diff

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

Borrow

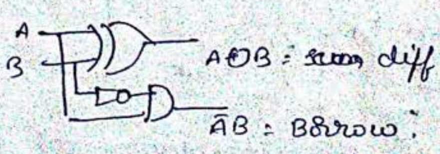
	0	1
	0	0

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

$A \oplus B$

Borrow $= \bar{A}B$

logic ckt:



Full subtractor:

i/p's			o/p's	
A	B	B_{in}	D	B_{out}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

D

	$\bar{B}\bar{B}_{in}$	$\bar{B}B_{in}$	BB_{in}	$B\bar{B}_{in}$
$\bar{A}$	0	1	0	1
A	1	0	1	0

B_{out}

0	1	1	1
0	0	1	0

$D = A\bar{B}\bar{B}_{in} + \bar{A}\bar{B}B_{in} + ABB_{in} + \bar{A}B\bar{B}_{in}$

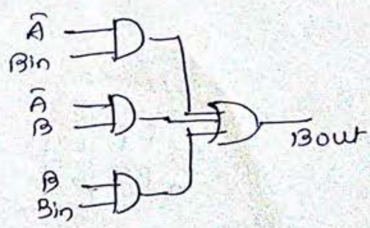
$= \bar{B}_{in}[A\bar{B}+\bar{A}B] + B_{in}[\bar{A}\bar{B}+AB]$

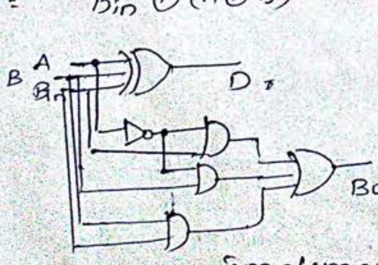
$= \bar{B}_{in}[A\oplus B] + B_{in}[A\odot B]$

$= \bar{B}_{in}[A\oplus B] + B_{in}[\overline{A\oplus B}]$

$= B_{in}\oplus(A\oplus B)$

$B_{out} = \bar{A}B_{in} + \bar{A}B + BB_{in}$





Implementation of full subtractor.

Implementation of F.S using two H.S's. & OR gate.

$$Bout = \bar{A}Bin + \bar{A}B + BBin$$
$$= \bar{A}B + [Bin(\bar{A} + B)]$$
$$= \bar{A}B + [Bin(\overline{\bar{A}B + A\bar{B}})]$$

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

$$\Rightarrow \bar{A}B + \bar{A}Bin(B + \bar{B}) + BBin$$
$$\Rightarrow \bar{A}B + \bar{A}BBin + \bar{A}\bar{B}Bin + BBin$$

$\bar{A}B + \bar{A}BBin + \bar{A}\bar{B}Bin + BBin$

$$\Rightarrow \bar{A}\bar{B}Bin + \bar{A}B(1 + Bin) + BBin$$

$\Rightarrow \bar{A}B + \bar{A}\bar{B}Bin + BBin(A + \bar{A})$

$$\Rightarrow \bar{A}\bar{B}Bin + \bar{A}B + BBin$$

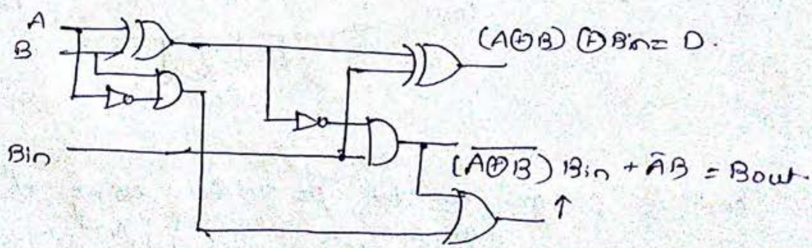
$\Rightarrow \bar{A}B + \bar{A}\bar{B}Bin + ABBin + \bar{A}BBin$

$$\Rightarrow \bar{A}\bar{B}Bin + BBin(A + \bar{A}) + \bar{A}B$$

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

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

$$\Rightarrow \bar{A}\bar{B}Bin + ABBin + \bar{A}BBin + \bar{A}B$$
$$\Rightarrow \bar{A}\bar{B}Bin + ABBin + \bar{A}B(1 + Bin)$$
$$\Rightarrow \bar{A}\bar{B}Bin + ABBin + \bar{A}B.$$
$$\Rightarrow \bar{A}B + (AB + \bar{A}\bar{B})Bin$$
$$\Rightarrow \bar{A}B + (\overline{\bar{A}B + A\bar{B}})Bin$$
$$\Rightarrow \bar{A}B + (\overline{A \oplus B})Bin$$

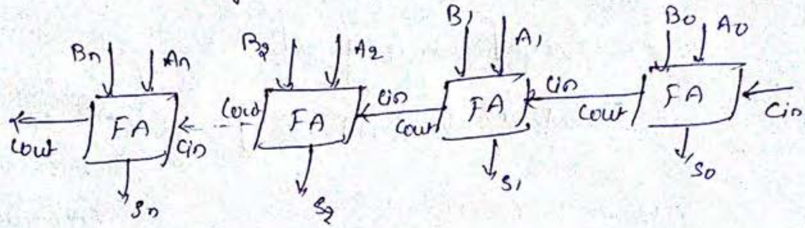


Ex) 61, 68, 70, 71, 74, 76, 77, 79, 80, 83, 86, 91, 95, 96, 97, 98,

A, A6, B7, B1, Pi: A2

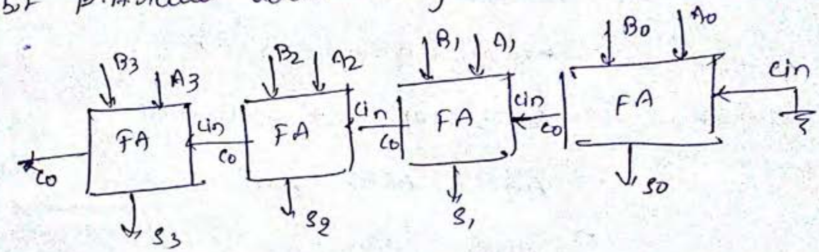
Binary Adder / parallel Adder:

Block dia of n bit parallel adder.



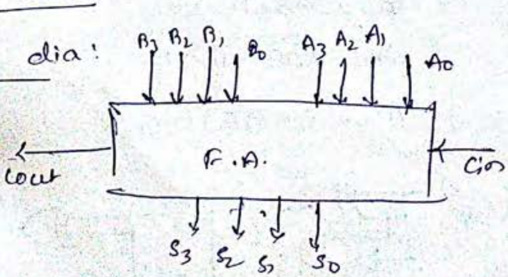
Note: carry = 0 b/c there is no carry into LSB position.

Ex: 4 bit p. parallel adder using F.A.



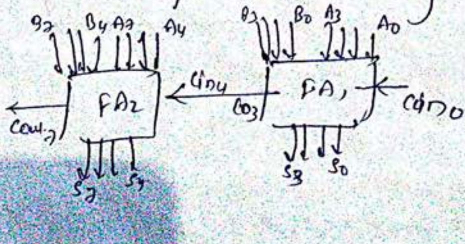
Block dia:

Logic dia:



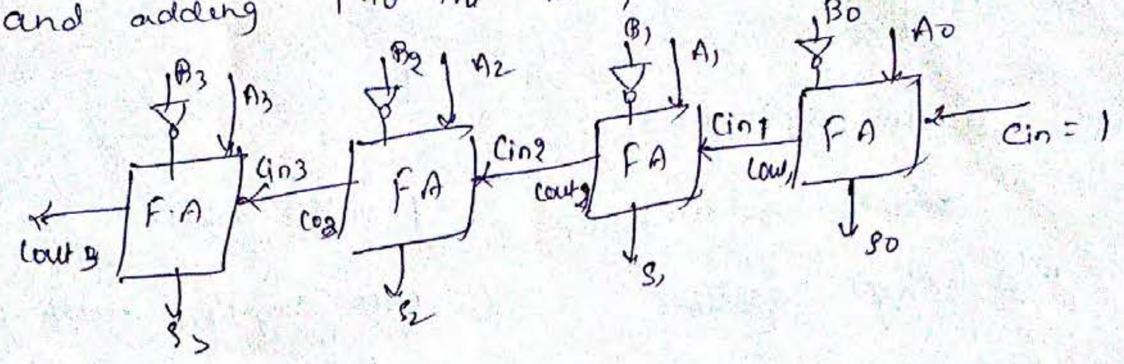
Binary parallel adder (IC 74LS83 / 74LS283) (functional sym:)

Ex 2: design an 8 bit pa adder using two 74283s.



Binary subtractor / parallel sub:

Then sub can be obtained by taking 2's comp of -ve no., 2's comp can be obtained by taking 1's comp and adding 1 to the L.S. pair bit '1'.



Parallel adder / subtractor:

M: Mode i/p controls the operation (ie) addition/sub.

M=0 = addition

M=1 = sub.

when M=0, one i/p is

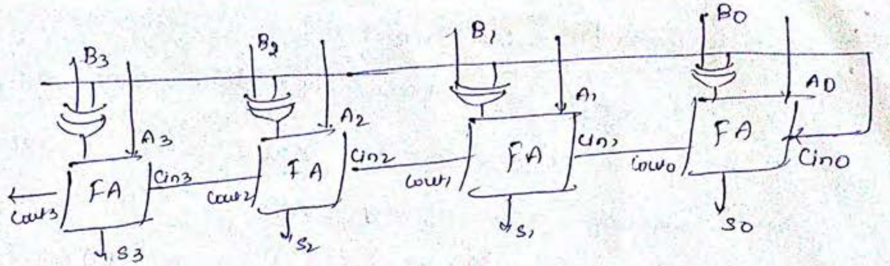
each XOR gate receives 1 i/p = M
2's i/p : B

addition ⇒ (ie) $M=0 \Rightarrow M \oplus B \Rightarrow 0 \oplus B = 0\bar{B} + 1 \cdot B$
$\Rightarrow B$
$C_{in0} = 0$

$M=1 \Rightarrow M \oplus B \Rightarrow 1 \oplus B = 1\bar{B} + 0 \cdot B$
$= \bar{B}$
$C_{in0} = 1$

ckt performs sub bcz

B i/p's are all complemented & 1 is added.



4 bit adder- subtractor:

serial adder	parallel adder.
1. uses shift registers	1. uses registers with parallel load capacity
2. requires only 1 full adder ckt	2. No. of F.A. ckt = no of bits in the binary no.
3. It is sequential ckt	3. Excluding reg, it is a purely combinational ckt
4. Time required for addition depends on no. of bits	4. doesn't depend of no. of bits
5. It is slower	5. faster

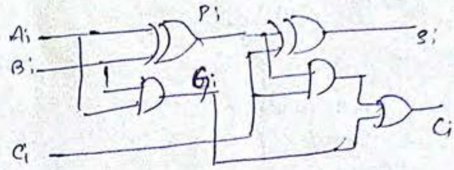
Carry look ahead Adder:

* In parallel adder the higher bits are generated iff lower bits are added producing sum & carry this results in delay called carry propdelay.

* One method of speeding up the process by Eliminating inter stage carry delay is look ahead carry addition

It uses two function 1. carry generation
2. carry propagation.

Full adder ckt:



$P_i = A_i \oplus B_i$

$G_i = A_i B_i$

$S_i = P_i \oplus C_i$

$C_{i+1} = P_i C_i + G_i$

C_1

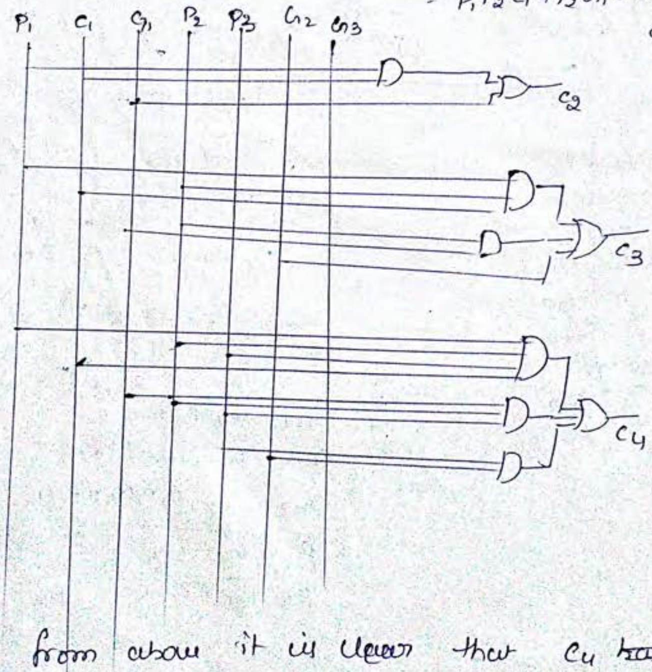
$i=1$, $C_2 = P_1 C_1 + G_1$

$C_3 = P_2 C_2 + G_2 = P_2 (P_1 C_1 + G_1) + G_2$

$= P_1 P_2 C_1 + P_2 G_1 + G_2$

$C_4 = P_3 C_3 + G_3$

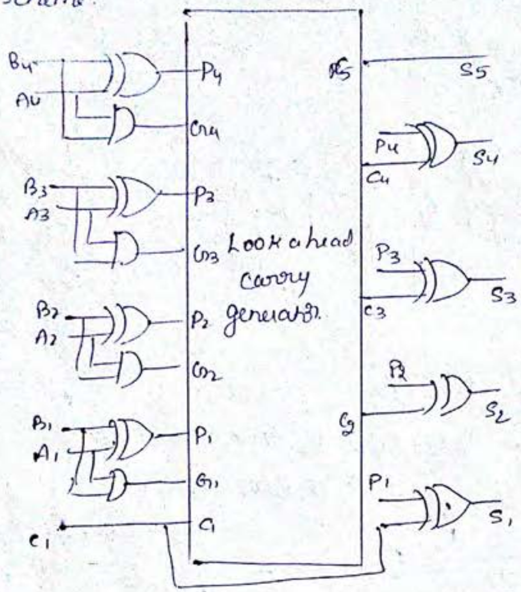
$= P_3 P_2 P_1 C_1 + P_3 P_2 G_1 + P_3 G_2 + G_3$

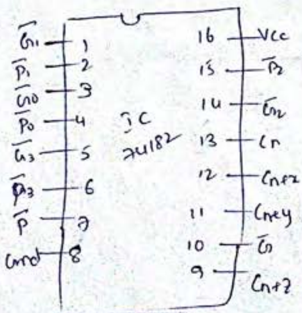


from above it is clear that C_4 ~~has~~ doesnt have to wait untill generation of C_2 & C_3..

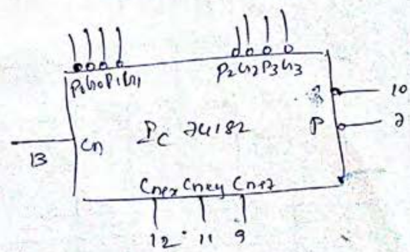
Vinay (404)

4-bit parallel adder with a look-ahead carry scheme.





Pin dia.



Logic sym.

$C_{n+x} = G_0 + P_0 C_n$

$C_{n+y} = G_1 + P_1 G_0 + P_1 P_0 C_n$

$C_{n+z} = G_2 + P_2 P_1 G_0 + P_2 G_1$

$\overline{G} = \overline{G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0}$

$\overline{P} = \overline{P_3 P_2 P_1 P_0}$

Decimal adder:

BCD addition:

1. add two no.s
2. $\leq 9 \Rightarrow$ no correction
3. $> 9 \Rightarrow C = 1$, sum invalid. then add 0110 to 4 bit sum if carry = 1 again add it to the next higher order BCD digit.

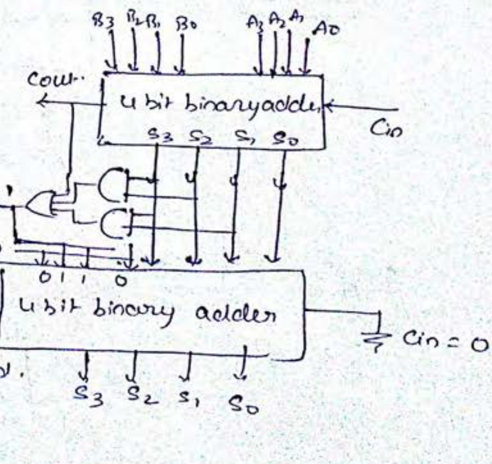
i/p's				o/p's
S_3	S_2	S_1	S_0	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

for sum > 9, Y = 1

S_3S_2 \ S_1S_0	00 $\bar{C}\bar{D}$	01 $\bar{C}D$	11 CD	10 $C\bar{D}$
00 $\bar{A}\bar{B}$	0	0	0	0
01 $\bar{A}B$	0	0	0	0
11 AB	1	1	1	1
10 $A\bar{B}$	0	0	1	1

$Y = AB + AC$

ie $\Rightarrow S_3S_2 + S_3S_1$

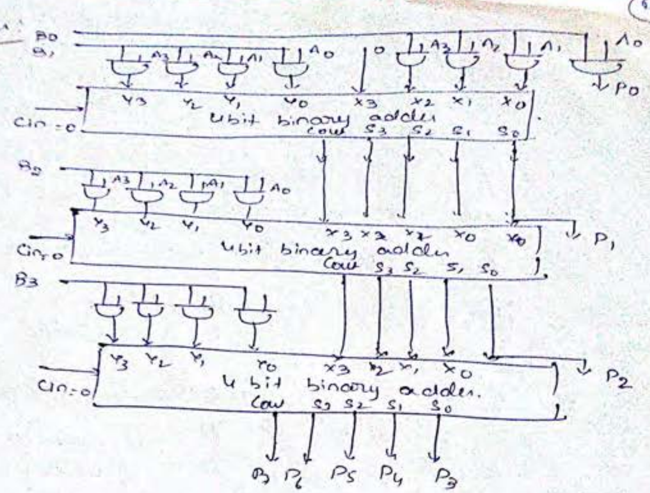


Binary multiplier:

$A_3\ A_2\ A_1\ A_0$
$B_3\ B_2\ B_1\ B_0$

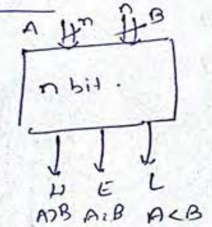
$A_3B_0\ \ A_2B_0\ \ A_1B_0\ \ A_0B_0$
$\dots\ A_1B_1\ \ A_0B_1\ \ \times$
$A_1B_2\ \ A_0B_2\ \ \times$
$A_0B_3\ \ \times$

$\dots\ P_3\ \ P_2\ \ P_1\ \ A_0B_0$
P_0



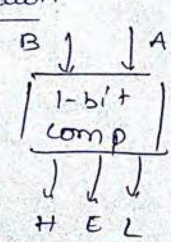
Magnitude comparator:

Block dia:



when $H=1 \Rightarrow A>B$
$E=1 \Rightarrow A=B$
$L=1 \Rightarrow A<B$

Single bit comparator:



T.T

A	B	H	E	L
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0

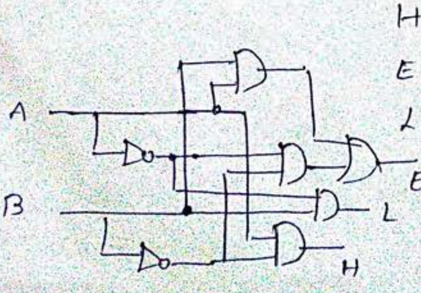
$H = A\bar{B}$

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

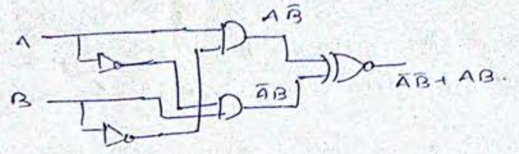
$= \overline{\bar{A}B + A\bar{B}}$

$L = \bar{A}B$

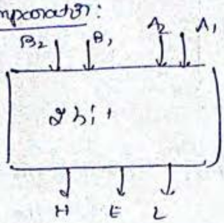
logic dia:



1918] 3, 5, 8, 15, 29, 37,
39, 51, 57,
LEU,



Two Bit comparator:



T.T

A_2	A_1	B_2	B_1
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

o/ps

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

K.map:

H

A_2A_1 \ B_2B_1	00	01	11	10
00	0	0	0	0
01	1	0	0	0
11	1	1	0	0
10	1	1	0	0

$H = A_2\bar{B}_2 + A_2\bar{A}_1\bar{B}_2 + A_1\bar{B}_2\bar{B}_1$

E

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

$E = \bar{A}_2\bar{A}_1\bar{B}_2\bar{B}_1 + \bar{A}_2 A_1\bar{B}_2 B_1 + A_1A_2B_1B_2 + A_2\bar{A}_1 + B_2\bar{B}_1$

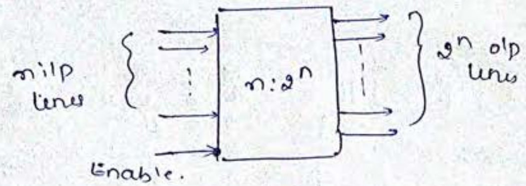
logic dia for H, E, L

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

$L = \bar{A}_2 B_2 + \bar{A}_2\bar{A}_1 B_1 + B_2 B_1\bar{A}_1$

Decoder:

converts an n bit binary i/p code into 2^n o/p lines (ie) only 1 o/p line is active for each one possibility.



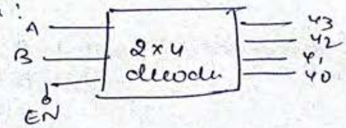
represented by n to m lines (or) n x m lines (or) n:m lines.

n = i/p's

m = o/p's

2 to 4 line decoder:

Logic dia:



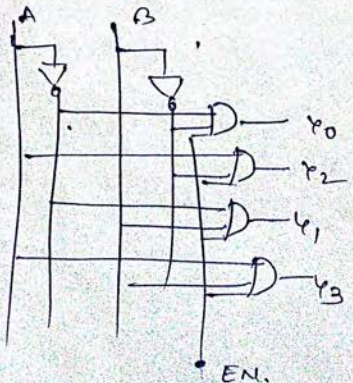
i/p's		o/p's			
EN	A B	Y_0	Y_1	Y_2	Y_3
0	x x	0	0	0	0
1	0 0	1	0	0	0
1	0 1	0	1	0	0
1	1 0	0	0	1	0
1	1 1	0	0	0	1

$Y_0 = \bar{A}\bar{B}$

$Y_1 = \bar{A}B$

$Y_2 = A\bar{B}$

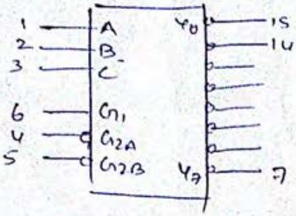
$Y_3 = AB$



3 to 8 line decoder:

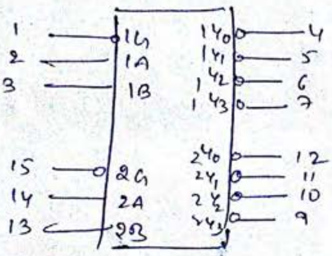
3 i/p's
8 o/p's

74x138 3 to 8 decoder:



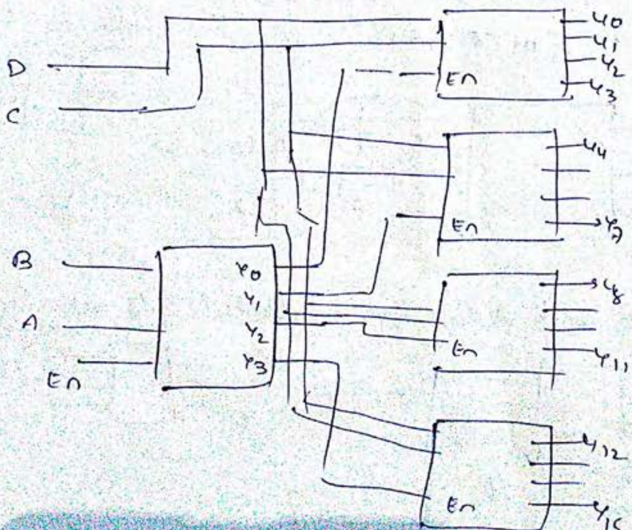
i/p's						o/p's							
$\overline{G_{2B}}$	$\overline{G_{2A}}$	G_1	C	B	A	$\overline{Y_7}$	$\overline{Y_6}$	$\overline{Y_5}$	$\overline{Y_4}$	$\overline{Y_3}$	$\overline{Y_2}$	$\overline{Y_1}$	$\overline{Y_0}$
1	x	x	x	x	x	1	1	1	1	1	1	1	1
x	1	x	x	x	x	1	1	1	1	1	1	1	1
x	x	0	x	x	x	1	1	1	1	1	1	1	1
0	0	1	0	0	0	1	1	1	1	1	1	1	0
0	0	1	0	0	1								
⋮													

74x139 Dual 2 to 4 decoder

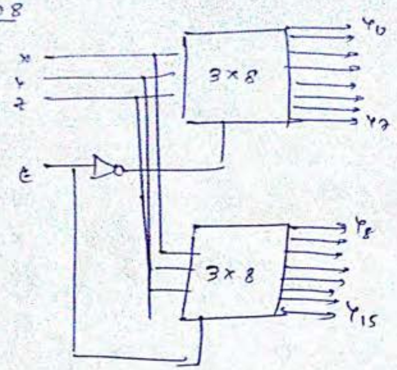


$\overline{G}$	B	A	$\overline{Y_3}$	$\overline{Y_2}$	$\overline{Y_1}$	$\overline{Y_0}$
1	x	x	1	1	1	1
0	0	0	1	1	1	0
0	0	1	1	1	0	1
0	1	0	1	0	1	0
0	1	1	0	1	1	1

4 to 16 using 2 to 4 decoders:



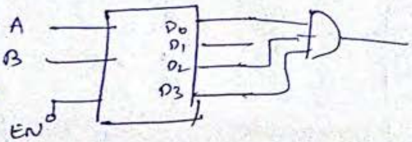
using 3 to 8



Realization of multiple o/p funs using decoder:

TO realise pos fun: Active High o/p.

Ex: $F(A,B) = \Pi M(0,2,3)$ using 2 to 4 decoder with active high o/p.



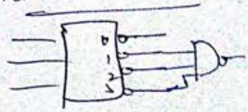
421
000 010 011

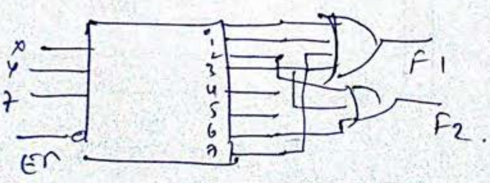
$F = (\hat{A}+\hat{B}+C)(A+\hat{B}+C)(A+\hat{B}+\hat{C})$.

Ex: 1. $F_1(x,y,z) = \Sigma m(0,1,3,7)$ (OR gate) $\Sigma m(1,2,3)$

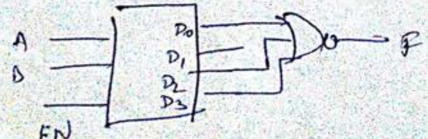
$F_2(x,y,z) = \Sigma m(2,3,7)$

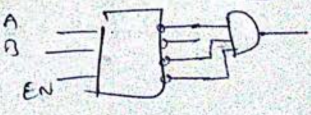
for active low decoder





2. $\Pi M(0,2,3)$ using Active High decoder: for active low:

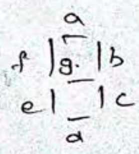


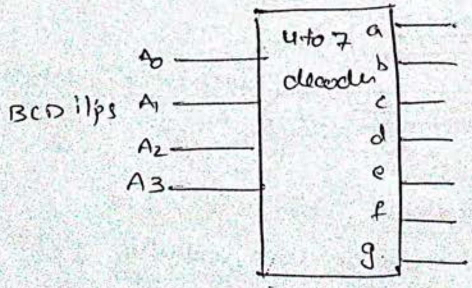


BCD to Decimal decoder:

i/p BCD code				Actual o/p										
B_3	B_2	B_1	B_0	$\bar{D}_0$	$\bar{D}_1$	$\bar{D}_2$	$\bar{D}_3$	$\bar{D}_4$	$\bar{D}_5$	$\bar{D}_6$	$\bar{D}_7$	$\bar{D}_8$	$\bar{D}_9$	$\bar{D}_{10}$ to $\bar{D}_{11}$ to $\bar{D}_{15}$
0	0	0	0	0	1	1	1	1	1	1	1	1	1	None.
0	0	0	1	1	0	1	1	1	1	1	1	1	1	
0	0	1	0	1	1	0	1	1	1	1	1	1	1	
0	0	1	1	1	1	1	0	1	1	1	1	1	1	
0	1	0	0	1	1	1	1	0	1	1	1	1	1	
0	1	0	1	1	1	1	1	1	0	1	1	1	1	
0	1	1	0	1	1	1	1	1	1	0	1	1	1	
0	1	1	1	1	1	1	1	1	1	1	0	1	1	
1	0	0	0	1	1	1	1	1	1	1	1	0	1	
1	0	0	1	1	1	1	1	1	1	1	1	1	0	
1	0	1	0	-	-	-	-							
1	0	1	1											
1	1	0	0			NONE								
1	1	0	1											
1	1	1	0											
1	1	1	1											

* BCD to 7 seg Decoder:

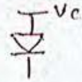




common cathode

Decimal digit	BCD A (A_0)	B (A_1)	C (A_2)	D (A_3)	seven seg code a	b	c	d	e	f	g
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
⋮											
9	1	0	0	1	1	1	1	1	0	1	1

i) Common cathod type  GND. $\Rightarrow$ High = 1 $\Rightarrow$ o/p.

Common anode type  Vcc $\Rightarrow$ low: 0 $\Rightarrow$ o/p.

K-maps: CD for a

A_0A_1 (AB) \ A_2A_3	00	01	11	10
00	1	0	1	1
01	0	1	1	1
11	X	X	X	X
00	1	1	X	1

$a = A_0 + A_2 + A_1A_3 + \overline{A_1}\,\overline{A_3}$

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

for b : $\overline{A_1} + \overline{A_2}\overline{A_3} + A_2A_3$

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

c : $B + \overline{C} + D$

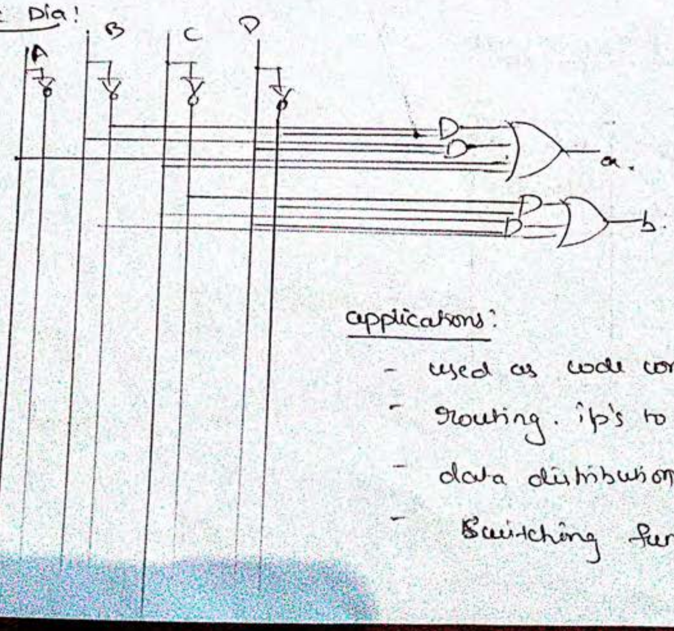
d : $\overline{B}\overline{D} + C\overline{D} + B\overline{C}D + \overline{B}C + A$

e : $\overline{B}\overline{D} + C\overline{D}$

f : $A + \overline{C}\overline{D} + B\overline{C} + B\overline{D}$

g : $A + B\overline{C} + \overline{B}C + C\overline{D}$

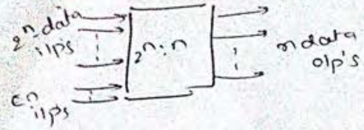
logic Dia!



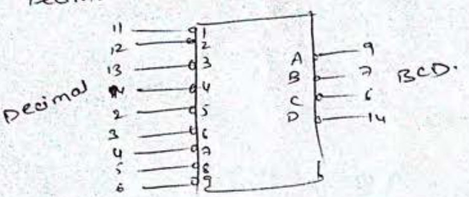
applications:

- used as code converters
- routing. i/p's to specified o/p's
- data distribution. (demux).
- Switching fun.

Encoders:



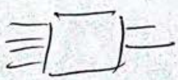
Decimal to BCD Encoder: (IC 74xx147).



Decimal i/p's o/p's

Decimal	1	2	3	4	5	6	7	8	9	D	C	B	A
0	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	1	1	1	1	1	1	1	1	1	1	1	0
2	x	0	1	1	1	1	1	1	1				
3	x	x	0										

4-to-2 Encoder:



A_3	A_2	A_1	A_0	Y_1	Y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

$Y_1 = \bar{A_3}A_2\bar{A_1}\bar{A_0} + A_3\bar{A_2}\bar{A_1}\bar{A_0}$

Priority encoders:

dis: If only 1 i/p is High ⇒ o/p is correct in some case

2 (or) more i/p's might be High.

Ex: operating a key board, a person bu releasing a key he may press the other so problem arises there.

Decimal to BCD priority encoder:

$\bar{D}_1$	$\bar{D}_2$	$\bar{D}_3$	$\bar{D}_4$	$\bar{D}_5$	$\bar{D}_6$	$\bar{D}_7$	$\bar{D}_8$	$\bar{D}_9$	$\bar{A}_3$	$\bar{A}_2$	$\bar{A}_1$	$\bar{A}_0$
1	1	1	1	1	1	1	1	1	1	1	1	1
x	x	x	x	x	x	x	x	0	0	1	1	0
x	x	x	x	x	x	x	0	1	0	1	1	1
x	x	x	x	x	x	0	1	1	1	0	0	0

⋮

Octal to Binary priority encoder:

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

⋮

Priority encoder:

priority fun is included.

Ex: if 2 (or) more i/ps = 1 at same time, i/p having highest priority will take precedence

4 bit priority encoder

i/p's				o/p's		
D_0	D_1	D_2	D_3	Y_0	Y_1	V (valid o/p indicator)
0	0	0	0	x	x	0
1	0	0	0	0	0	1
x	1	0	0	0	1	1
x	x	1	0	1	0	1
x	x	x	1	1	1	1

$Y_0 = D_3 + D_1\bar{D}_2$

$Y_1 = D_2 + D_3$

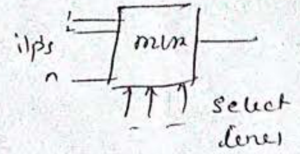
$V = D_0 + D_1 + D_2 + D_3$

mux :

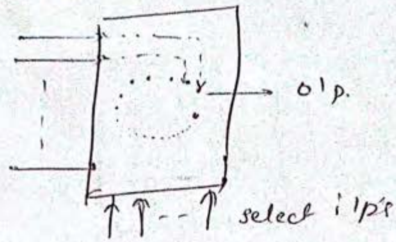
ilp are 'n'

olp: 1

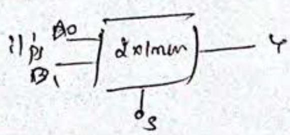
Block dia'



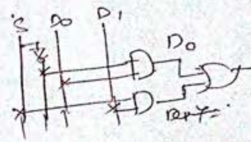
functional dia'



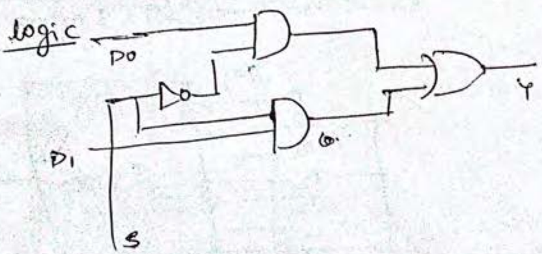
2x1 mux:



S	A	B	Y
0			D0
1			D1



logic



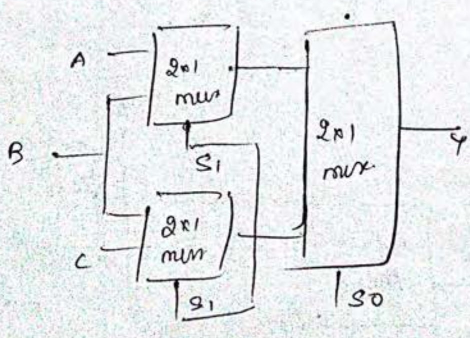
when S=0

Y = D0

when S=1

Y = D1

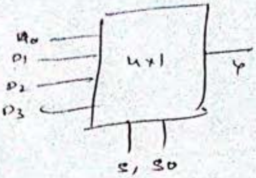
3x1 mux using 2x1 mux



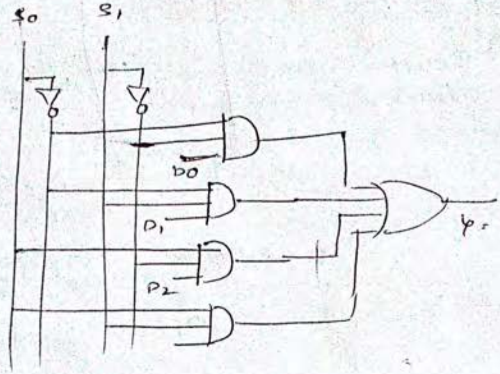
So	S_1	S_1	Y.
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

So	S_1	Y
0	0	A
0	1	B
1	0	B
1	1	C.

4x1 mux:



S_0	S_1	Y
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3



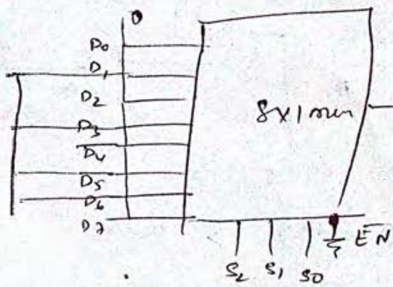
$Y = D_0 + D_1 + D_2 + D_3$

8x1 mux:

T.T:

EN	S			o/p
E	S_2	S_1	S_0	Y.
1	x	x	x	0
0	0	0	0	D_0
0	0	0	1	D_1
⋮				
0	1	1	1	D_7

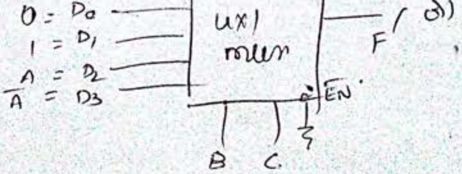
Ex: 1) Implement $F = \sum m(1,3,5,6)$ using 8x1 mux.



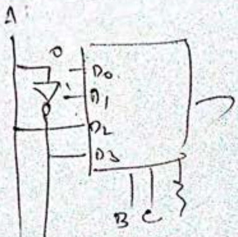
2) $\sum m(1,3,5,6)$

Implementation table:

	D_0	D_1	D_2	D_3
$\bar{A}$	0	1	2	3
A	4	5	6	7
	0	D_1	A	$\bar{A}$



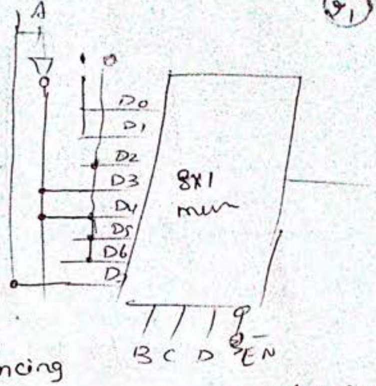
(or)



8x1 mux $\Sigma m(0,1,3,4,8,9,15)$

	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
$\bar{A}$	0	1	2	3	4	5	6	7
A	8	9	10	11	12	13	14	15
	1	1		$\bar{A}$	$\bar{A}$			$\hat{A}$

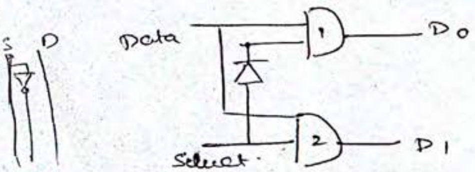
$\Pi M(2,5,6,7,10,11,12,13,14)$



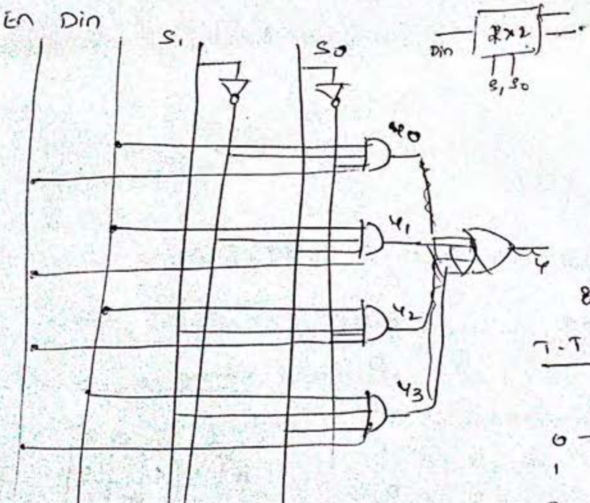
app: 1. 7 seg display
2. logic fun gen
3. Digital counter
4. operating sequencing
5. Data selection, routing
6. parallel & serial conversion
7. code generation.

De mux:

logic dia:



En	S_1	S_0	Din	Y_0	Y_1	Y_2	Y_3
0	x	x	x	0	0	0	0
1	0	0	0	0	0	0	0
1	0	0	1	1	0	0	0
1	0	1	0	0	0	0	0
1	0	1	1	0	1	0	0
1	1	0	0	0	0	0	0
1	1	0	1	0	0	1	0
1	1	1	0	0	0	0	0
1	1	1	1	0	0	0	1



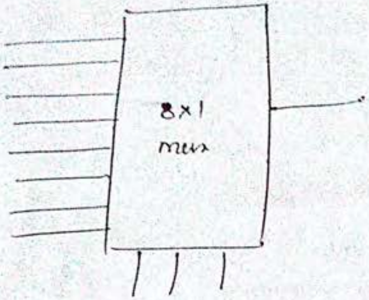
Ex: Implement F.S using demux

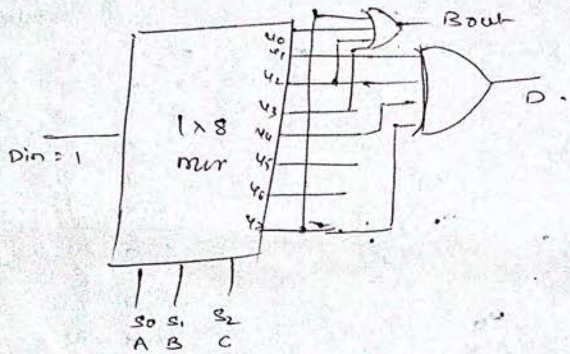
T.T F.P

	A	B	C	D	Bo
0	0	0	0	0	0
1	0	0	1	1	1
2	0	1	0	1	1
3	0	1	1	0	1
4	1	0	0	1	0
5	1	0	1	0	1
6	1	1	0	0	0
7	1	1	1	1	1

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

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





Parity checker/ Generator:

TT of parity generator for even & odd parity

A	B	C	Odd	Even
0	0	0	1	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	0	1

K-map:

ODD

	$\bar B\bar C$	$\bar B C$	BC	$B\bar C$
$\bar A$	1	0	1	0
A	0	1	0	1

P_{odd} : $\bar A\bar B\bar C + A\bar B C + \bar A B C + AB\bar C$

$\Rightarrow \bar C(A\odot B) + C(A\oplus B)$

$\Rightarrow (A\oplus B)\odot C$

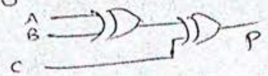
Even

0	1	0	1
1	0	1	0

P_{even}: $A\bar B\bar C + \bar A\bar B C + ABC + \bar A B\bar C$

$\Rightarrow (A\oplus B)\oplus C.$

logic dia:

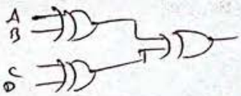


T.T for Even parity checker:

4 bits received				parity error check
A	B	C	D	PEC.
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

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

$PEC = (A \oplus B) \oplus (C \oplus D)$



code converter:

Binary to BCD converter:

Binary				BCD code.				
8	4	2	1					
D	C	B	A	B_4	B_3	B_2	B_1	B_0
0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	1
0	0	1	0	0	0	0	1	0
⋮								
1	0	0	1	0	1	0	0	1
1	0	1	0	1	0	0	0	0
1	0	1	1	1	0	0	0	1

K-map.

$B_4 = DC + DB$

$B_3 = D\bar{C}\bar{B}$

$B_2 = \bar{D}C + CB$

$B_1 = DC\bar{B} + \bar{D}B$

$B_0 = A.$

logic dia:

BCD to Binary:

BCD					Binary				
B_4	B_3	B_2	B_1	B_0	E	D	C	B	A
0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	1
⋮									
					0	1	0	1	0
1	0	0	0	0	0	1	0	1	1
1	0	0	0	1		1	1	1	1
1	0	1	0	1	1	0	0	0	0
1	0	1	1	0	1	0	0	0	1

~~map~~

5 variable K-map: for $A = B_0$

$B_4 = 0$ $B_4 = 1$

$B_3B_2 \backslash B_1B_0$

$B = B_1 \oplus B_4$

$C = \overline{B_4} B_2 + B_2\overline{B_1} + B_4\overline{B_2}B_1$

$D = \overline{B_4} B_3 + B_4\overline{B_3}\overline{B_2} + B_4\overline{B_3}\overline{B_1}$

$E = B_4B_3 + B_4B_2B_1$

logic dia:

BCD to Excess-3:

Decimal	BCD $B_3\ B_2\ B_1\ B_0$	Excess-3 $E_3\ E_2\ E_1\ E_0$
0		0 0 1 1
0	0 0 0 0	0 1 0 0
1	0 0 0 1	
⋮		
9	1 0 0 1	1 1 0 0

K-map:

$E_3 = B_3 + B_2 (B_0 + B_1)$

$E_2 = B_2 \bar{B_1} \bar{B_0} + \bar{B_2} (B_0 + B_1)$

$E_1 = B_1 \odot B_0$

$E_0 = \bar{B_0}$

logic dia:

Excess-3 to BCD code conveulter:

E_3	E_2	E_1	E_0	B_3	B_2	B_1	B_0
0	0	1	1	0	0	0	0
0	1	0	0	0	0	0	1
⋮							

K-map:

$B_0 = \bar{E_0}$

$B_1 = E_1 \oplus E_0$

$B_2 = \bar{E_2} \bar{E_1} + E_2 E_1 E_0 + E_3 E_1 \bar{E_0}$

$B_3 = E_3 E_2 + E_3 E_1 E_0$

logic dia:

Binary to Gray code:

Binary B_3	B_2	B_1	B_0	Gray G_3	G_2	G_1	G_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
⋮							

K-map:

$G_0 = B \oplus A$

$G_1 = C \oplus B$

$G_2 = D \oplus C$

$G_3 = D.$

logic dia:

Gray code to Binary code:

Gray code   Binary

$G_3\ G_2\ G_1\ G_0$ $B_3\ B_2\ B_1\ B_0$

D C B A

$A = (G_3 \oplus G_2) \oplus (G_1 \oplus G_0)$

$B = G_3 \oplus G_2 \oplus G_1$

$C = G_3 \oplus G_2$

$D = G_3$.

logic dia:

BCD to Gray code

$B_3\ B_2\ B_1\ B_0$ $G_3\ G_2\ G_1\ G_0$.

$G_0 = B_1 \oplus B_0$

$G_1 = B_2 \oplus B_1$

$G_2 = B_2 + B_3$

$G_3 = B_3$.

logic dia:

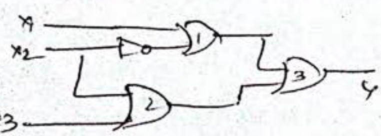
Hazards:

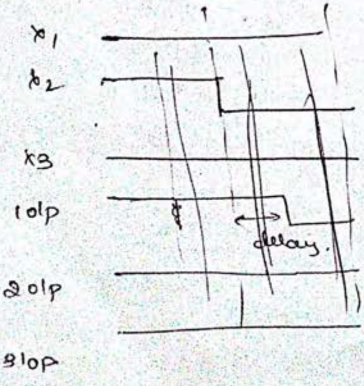
The unwanted switching transients that may appear at the o/p of a ckt are Hazards.

These occur bcz of prop delays., they occur in combinational ckts.

Hazards in combinational ckts:

Ex:





static-1 : hazard.

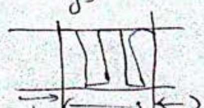
=> o/p is 0, but should be 1.

Static-0:

o/p is 1, but should be 0

Dynamic:

changes continuously.



Hazard free realization: two for $<$ Dynamic / static.

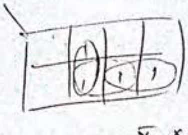
- Hazards can be eliminated by enclosing two minterms / two max terms in ques.

Ex: $x_1x_2 + \bar{x}_2x_3$. add one more minterm x_1x_3.

Eliminating Hazard:

x_1x_2 \ x_2x_3	$\bar{x}_2\bar{x}_3$	$\bar{x}_2x_3$	x_2x_3	$x_2\bar{x}_3$
$\bar{x}_1$		1		
x_1		1	1	1

$Y = x_1x_2 + \bar{x}_2x_3$



$Y = x_1x_2 + \bar{x}_2x_3 + x_1x_3$

Ex: $f = \Sigma m(1,3,6,7,13,15)$ obtain Hazard free cct:

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

$Y = \bar{A}\bar{B}D + CD\bar{A} + \bar{A}BC + BCD + ABD.$

Essential Hazards:

In Asynchronous seq cct, the Hazard caused by unequal delays along two (or) more paths that come from same i/p.

Ex: $\Sigma m(0,2,6,7,8,10,12)$, find Hazard free realization

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

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

Hazard free fun:

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

Note: static & Dynamic Hazards occur both in combinational & sequential ckt

Essential Hazard occur in only sequential ckt.

Functional Hazard:
- static - 0
- static - 1
- Dynamic

* code converters:

Excess-3

Binary ⇆ BCD Gray

Binary ⟷ BCD code ⟷ Excess-3

Binary ↕ Gray code

BCD code ↓ Gray