UNIT-II
MINIMIZATION TECHNIQUES

Boolean Theorems:—

Based on the 3 basic operations AND(·), OR(+) and NOT(‾) the following laws are given in Boolean Algebra.

→ AND Laws:—

① $A \cdot 0 = 0$ → ① Null Law

Varification:—

If $A=0$ then $0 \cdot 0 = 0$
If $A=1$ then $1 \cdot 0 = 0$ } → Eq ① satisfied

② $A \cdot 1 = A$ → ② Identify Law

Varification:- If $A=0$ then $0 \cdot 1 = 0$
If $A=1$ then $1 \cdot 1 = 1$ } → Eq ② satisified

③ $A \cdot A = A$ → ③

Varification:—

If $A=0$ then $0 \cdot 0 = 0$
If $A=1$ then $1 \cdot 1 = 1$ } → Eq ③ Satisfied

④ $A \cdot \bar{A} = 0$ → ④

Varification:—

If $A=0$ then $0 \cdot 1 = 0$
If $A=1$ then $1 \cdot 0 = 0$ } → Eq ④ satisfied.

→ OR Laws:—

① $A + 0 = A$ → ⑤ Null Law

Varification:— If $A=0$ then $0+0=0$
If $A=1$ then $1+0=1$ } → Eq ⑤ satisfied

② $A+1=1$ → ⑥ Identity law

Varification:— If $A=0$ then $0+1=1$
If $A=1$ then $1+1=1$ } → Eq ⑥ satisfied

③ $A+A=A$ → ⑦

Varification:- If $A=0$ then $0+0=0$
If $A=1$ then $1+1=1$ } Eq ⑦ satisfied

④ $A+\bar{A}=1$ → ⑧

Varification:- If $A=0$ then $0+1=1$
If $A=1$ then $1+0=1$ } → Eq ⑧ satisfied

→ **NOT Laws:—**

$$\bar{\bar{A}} = A \rightarrow (9)$$

Verification:— If $A=0$ then $\bar{A} = \bar{0} = 1$ $\bar{\bar{A}} = \bar{1} = 0$ i.e A
If $A=1$ then $\bar{A} = \bar{1} = 0$ $\bar{\bar{A}} = \bar{0} = 1$ i.e A } Eq (9) Satisfied

→ **Commutative Laws:—**

$$A+B = B+A \rightarrow (10)$$

$$A \cdot B = B \cdot A \rightarrow (11)$$

verification:—

If A=0, B=0	0+0 = 0+0	⇒ 0=0	Eq (10) satisfied
If A=0, B=1	0+1 = 1+0	⇒ 1=1	
If A=1, B=0	1+0 = 0+1	⇒ 1=1	
If A=1, B=1	1+1 = 1+1	⇒ 1=1	

verification for Eq → (11)

$$A \cdot B = B \cdot A$$

If A=0 B=0	0·0 = 0·0	⇒ 0=0	→ Eq (11) Satisfied
If A=0 B=1	0·1 = 1·0	⇒ 0=0	
If A=1 B=0	1·0 = 0·1	⇒ 0=0	
If A=1 B=1	1·1 = 1·1	⇒ 1=1	

→ **Associative Laws:—**

$$A+(B+C) = (A+B)+C = A+B+C \rightarrow (12)$$

$$A \cdot (B \cdot C) = (A \cdot B) \cdot C = A \cdot B \cdot C \rightarrow (13)$$

verification:—

A=0 B=0 C=0	0+(0+0)	= (0+0)+0	= 0+0+0	0
A=0 B=0 C=1	0+(0+1)	= (0+0)+0	= 0+0+1	1
A=0 B=1 C=0	0+(1+0)	= (0+1)+0	= 0+1+0	1
A=0 B=1 C=1	0+(1+1)	= (0+1)+1	= 0+1+1	1
A=1 B=0 C=0	1+(0+0)	= (1+0)+0	= 1+0+0	1
A=1 B=0 C=1	1+(0+1)	= (1+0)+1	= 1+0+1	1
A=1 B=1 C=0	1+(1+0)	= (1+1)+0	= 1+1+0	1
A=1 B=1 C=1	1+(1+1)	= (1+1)+1	= 1+1+1	1

lly $A \cdot (B \cdot C) = (A \cdot B) \cdot C = A \cdot B \cdot C$ can be verified

→ **Distributive Laws:—**

$$A \cdot (B+C) = A \cdot B + A \cdot C \rightarrow (14)$$

$$A + (B \cdot C) = (A+B)(A+C) \rightarrow (15)$$

verification: $A \cdot (B+C) = A \cdot B + AC$

R.H.S. $AB + A \cdot C$ Take 'A' common
$A \cdot (B+C)$ ∴ LHS = RHS eq (14) is verified

G. Naveen
EEE

$\Rightarrow A+(B\cdot C) = (A+B)\cdot(A+C)$

R.H.S $(A+B)(A+C)$

$AA + AC + AB + BC$

$A + AC + AB + BC$

Take 'A' common

$A(1+C+B) + BC$

$A + BC$

$\therefore$ LHS = RHS Eq (15) is satisfied.

$\Rightarrow$ Auxillary Laws:—

$A + A\cdot B = A \rightarrow (16)$

$A + \bar{A}\cdot B = A+B \rightarrow (17)$

$AB + BC + \bar{B}C = AB + C \rightarrow (18)$

Verification:-

$\rightarrow A + A\cdot B = A$

Take A common

$A(1+B) = A$ $\rightarrow$ Eq (16) satisfied

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

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

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

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

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

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

$= A+B$ $\rightarrow$ Eq (17) satisfied

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

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

$AB + C = AB + C$

Eq $\rightarrow$ (18) satisfied

Duality:—

The principle of duality theorem says that, starting with a Boolean relation, you can derive another Boolean relation by

1. changing each OR sign to an AND sign
2. changing each AND sign to an OR sign &
3. complementing any 0 (or) 1 appearing in the expression.

Eg:- Dual of relation $A+\bar{A} = 1$ is $A\cdot\bar{A} = 0$

Duality is a very important property of Boolean Algebra.

Complementation Laws:-

The term complement simply means to "invert", ie to change 0's to 1's and 1's to 0's. The five laws of complementation are.

- Law 1: $\bar{0} = 1$
- Law 2: $\bar{1} = 0$
- Law 3: If $A = 0$ then $\bar{A} = 1$
- Law 4: If $A = 1$ then $\bar{A} = 0$
- Law 5: $\bar{\bar{A}} = A$ (Double complementation law).

Dual's:-

	Given Expression	Dual
1.	$\bar{0} = 1$	$\bar{1} = 0$
2.	$0 \cdot 1 = 0$	$1 + 0 = 1$
3.	$0 \cdot 0 = 0$	$1 + 1 = 1$
4.	$1 \cdot 1 = 1$	$0 + 0 = 0$
5.	$A \cdot 0 = 0$	$A + 1 = 1$
6.	$A \cdot 1 = A$	$A + 0 = A$
7.	$A \cdot A = A$	$A + A = A$.
8.	$A \cdot \bar{A} = 0$	$A + \bar{A} = 1$
9.	$A \cdot B = B \cdot A$	$A + B = B + A$
10	$A \cdot (B \cdot C) = (A \cdot B) \cdot C$	$A + (B + C) = (A + B) + C$
11.	$A \cdot (B + C) = AB + AC$	$A + (B \cdot C) = (A + B) \cdot (A + C)$
12.	$A (A + B) = A$	$A + AB = A$
13	$A \cdot (A \cdot B) = A \cdot B$	$A + (A + B) = A + B$
14.	$\overline{AB} = \bar{A} + \bar{B}$	$\overline{A+B} = \bar{A} \cdot \bar{B}$
15.	$(A+B)(\bar{A}+C)(B+C) = (A+B)(\bar{A}+C)$	$A \cdot B + \bar{A}C + BC = AB + \bar{A}C$
16	$(A+C)(\bar{A}+B) = AB + \bar{A}C$	$AC + \bar{A}B = (A+B)(\bar{A}+C)$
17.	$A + \bar{B}C(A + \bar{B}C) = A + \bar{B}C$	$A[\bar{B} + C + A(\bar{B}+C)] = A \cdot (\bar{B}+C)$
18.	$\overline{A\bar{B} + ABC} + A(B + A\bar{B}) = 0$	$\overline{(A+\bar{B}) \cdot (A+B+C)} \cdot (A + [B(A+\bar{B})]) = 1$
19.	$ABD + ABCD = ABD$	$(A+B+D)(A+B+C+D) = A+B+D$
20.	$AB + \overline{AC} + A\bar{B}C(AB+C) = 1$	$(A+B)(\overline{A+C})[(A+\bar{B}+C) + (A+B)C] = 0$
21.	$\overline{\overline{AB} + \bar{A} + AB} = 0$	$\overline{(\overline{A+B}) \cdot \bar{A} \cdot (A+B)} = 1$

G. Naveen

De Morgan's Theorems:-

De Morgan suggested two theorems that form an important part of Boolean algebra. In the equation form, they are:

Law ① $\overline{AB} = \bar{A} + \bar{B}$

The complement of a product is equal to the sum of the complements.

Truth Table:-

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

Law ② $\overline{A+B} = \bar{A}\bar{B}$

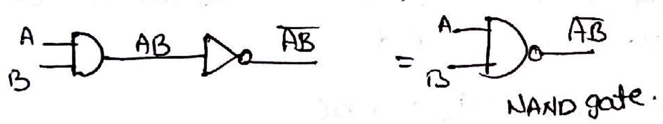
The complement of a sum is equal to the product of the complements

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

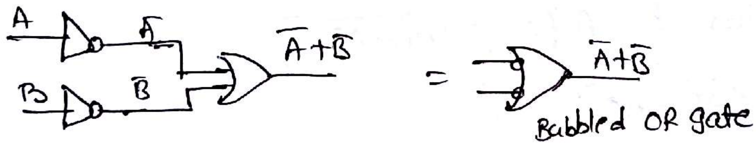
(or)

Law 1:- $\overline{AB} = \bar{A} + \bar{B}$

LHS:-

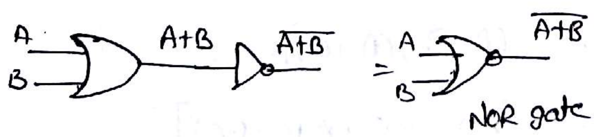


RHS:-

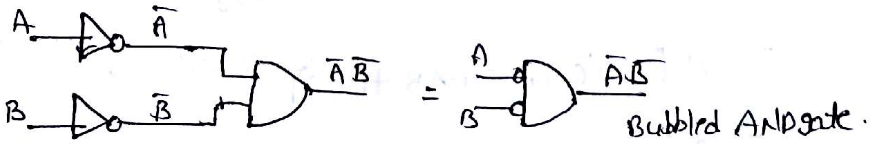


Law 2:- $\overline{A+B} = \bar{A}\bar{B}$

LHS:



RHS:-



Reducing Boolean Expressions:—

Every Boolean Expression must be reduced to as simple a form as possible before realization, because every logic operation in the expression represents a corresponding element of hardware.

To reduce Boolean Expressions, all the laws of Boolean Algebra may be used.

Procedure:—

1. Multiply all variables necessary to remove parenthesis.
2. look for identical terms. only one of those term be retained and all other terms be dropped

   eg:- $AB + AB + AB = AB$
3. look for a variable and its negation in the same term. this term can be dropped

   eg: $A \cdot B\bar{B} = A \cdot 0 = 0$
4. look for pairs of terms that are Identical Except for one variable. which may be missing in one of the terms, the larger term can be dropped.

   $$AB\bar{C}\bar{D} + AB\bar{C} = AB\bar{C}(1+\bar{D}) = AB\bar{C} \cdot 1 = AB\bar{C}$$
5. look for the pairs of terms which have the same variables, with one or more variables complemented in one of those terms. Such terms can be combined into a single term with that variable dropped.

   $$AB\bar{C}\bar{D} + AB\bar{C}D = AB\bar{C}(D+\bar{D}) = AB\bar{C}$$

Eg:- Reduce the expression $f = A[B+\bar{C}(\overline{AB+A\bar{C}})]$

Sol:-

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

Demorganize $\overline{AB+A\bar{C}}$ $\quad = A[B+\bar{C}(\overline{AB} \cdot \overline{A\bar{C}})]$

Demorganize $\overline{AB}$ & $\overline{A\bar{C}}$ $\quad = A[B+\bar{C}(\bar{A}+\bar{B})(\bar{A}+C)]$

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

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

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

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

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

$$= AB$$

G.NAVEEN
EEE

Eg 2:- Reduce the expression $f = A+B[AC+(B+\bar{C})D]$

Sol:-

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

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

$= A+ABC+BBD+B\bar{C}D$ AND law

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

$= A(1+BC)+BD(1+\bar{C})$ OR law

$= A\cdot 1 + BD\cdot 1$

$= A+BD$

Eg:- Reduce the expression $f = (\overline{A+\overline{BC}})(A\bar{B}+ABC)$

Sol:-

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

$= \bar{A}\cdot\overline{\overline{BC}}(A\bar{B}+ABC)$

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

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

$= 0 + 0 = 0$

Eg:- Reduce the expression $f = (B+BC)(B+\bar{B}C)(B+D)$

Sol:-

$= B(1+C)(B+\bar{B}C)(B+D)$ (OR law)

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

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

AND law $= B(B+D)$

$= BB+BD$

$= B+BD$

$= B(1+D)$

Eg: show that $AB+A\bar{B}C+B\bar{C} = AC+B\bar{C}$

Sol:

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

from distributive law 2 $A+BC = (A+B)(A+C)$

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

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

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

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

$= ABC+AB\bar{C}+AC+B\bar{C}$

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

$= AC+B\bar{C}$

Eg:- Show that $A\bar{B}C + B + B\bar{D} + AB\bar{D} + \bar{A}C = B+C$

Sol:-

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

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

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

$A\bar{B}C + B(1+A\bar{D}) + \bar{A}C$ → Distributive Law

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

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

$= C\bar{A} + C\bar{B} + B$ → Distributive law

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

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

$= B+C.$

Minimization of Switching Functions:—

Simplification of Boolean functions includes K-maps, Quine McCluskey and map entered variables methods.

The need of minimization of a Boolean functions is that

→ Cost of the Network decreases.
→ cost of the components decreases
→ Reliability of the N/w Increases
→ Time taken by the N/w to respond to changes at its I/p.
→ cost of maintaining the network reduces.

Map method:—

For minimization we need better understanding of boolean laws, rules and theorems.

The map method gives us a systematic approach for Simplifying a Boolean Expression.

this map method, first proposed by Veitch & modified by Karnaugh, hence its is known as Veitch diagram (or) the Karnaugh map.

One-variable, Two Variable, Three-Variable & Four Variable Maps:—

Each Karnaugh map represents one of the 2^n possible products that can be formed from 'n' variables

i.e 2 - variable map contains $2^2 = 4$ cells

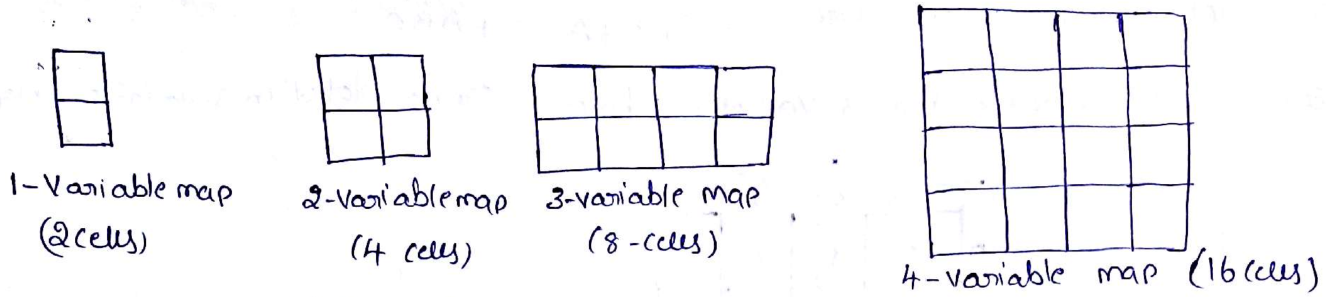
3 - " $2^3 = 8$ cells

4 - " " $2^4 = 16$ "

5 - " " $2^5 = 32$ "

6 - " " $2^6 = 64$ cells

G.NAVEEN
EEE



out lines of 1,2,3 & 4 Variable Karnaugh map's

Product terms are assigned to the cells of a karnaugh map by labelling each row and each column of the map with a variable, with its complement or with a combination of variables and complements.

The way to label the rows and columns of a 1, 2, 3 and 4 variable maps and the product terms corresponding to each cell are as shown in fig.

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

1-Variable map

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

2-Variable map

	00 $\bar{B}\bar{C}$	01 $\bar{B}C$	11 BC	10 $B\bar{C}$
0 $\bar{A}$	$\bar{A}\bar{B}\bar{C}$ 0	$\bar{A}\bar{B}C$ 1	$\bar{A}BC$ 3	$\bar{A}B\bar{C}$ 2
1 A	$A\bar{B}\bar{C}$ 4	$A\bar{B}C$ 5	ABC 7	$AB\bar{C}$ 6

3-Variable map

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

4- Variable map

from 0-1 (or) 0-4 only one variable in the product term changes. So "Gray code" has same property

→ Hence Gray Code is used to lable the rows and colums of K-map.

→ Product terms are represent with minterms and row & colums are marked with "Gray code" instead of variables

A	0
0	m_0 0
1	m_1 1

1-Variable map

A \ B	0	1
0	m_0 0	m_1 1
1	m_2 2	m_3 3

2-Variable.

→ Gray code Sequence

A \ BC	00	01	11	10
0	m_0	m_1	m_3	m_2
1	m_4	m_5	m_7	m_6

3-Variable.

→ Gray Code Sequence

AB \ CD	00	01	11	10
00	m_0	m_1	m_3	m_2
01	m_4	m_5	m_7	m_6
11	m_{12}	m_{13}	m_{15}	m_{14}
10	m_8	m_9	m_{11}	m_{10}

Gray code sequence

4 - Variable.

Another way to represent 1,2,3 & 4 variable maps.

Sum Terms Representation on Karnaugh map:—

A Boolean Expression with Sum terms (SOP) can be plotted on the K-map by placing a '1' in each cell corresponding to a term (minterm) in the SOP Expression Remaining cells are filled with "zero's".

Eg:- Plot Boolean Expression $y = AB\bar{C} + ABC + \bar{A}\bar{B}C$

 110

Sol:- The expression has 3 variables & hence it can be plotted in 3-variable K-map

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

A \ BC	00	01	10	10
0	0 (0)	1 (1)	0 (3)	0 (2)
1	0 (4)	0 (5)	1 (7)	1 (6) $\to AB\bar{C}$

$\downarrow$ ABC

Eg:- Plot the Boolean Expression

$y = \bar{A}B\bar{C}\bar{D} + A\bar{B}C\bar{D} + \bar{A}BC\bar{D} + A\bar{B}CD + AB\bar{C}D$ on the K-map

 (4) $\qquad$ (10) $\qquad$ 6 $\qquad$ 11 $\qquad$ 13

Sol:- The expression has 4 variables & hence it can be plotted in 4-variable K-map

AB \ CD	00 $\bar{C}\bar{D}$	01 $\bar{C}D$	11 CD	10 $C\bar{D}$
00 $\bar{A}\bar{B}$	0 (0)	0 (1)	0 (3)	0 (2)
01 $\bar{A}B$	1 (4)	0 (5)	0 (7)	1 (6)
11 AB	0 (12)	1 (13)	0 (15)	0 (14)
10 $A\bar{B}$	0 (8)	0 (9)	1 (11)	1 (10)

Product terms Representation on Karnaugh maps:—

A Boolean expression with product terms (POS) can be plotted on the Karnaugh map by placing a '0' in each cell corresponding to a term (max term) in the expression. Remaining cells are filled with one's (1's)

Eg:- Plot Boolean expression $y = (A+\bar{B}+C)(A+\bar{B}+\bar{C})(\bar{A}+\bar{B}+C)(A+B+\bar{C})$ on the K-map

Sol:- The expression has 3 variables & hence it can be plotted in 3-variable K-map.

A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}$ 0	1 (0)	0 (1)	0 (3)	0 (2)
A 1	1 (4)	1 (5)	1	0 (6)

$(A+\bar{B}+C) = M_2$, $\quad (A+\bar{B}+\bar{C}) = M_3$, $\quad (\bar{A}+\bar{B}+C) = M_6$

$(A+B+\bar{C}) = M_1$

G. NAVEEN
EEE

Eg: Plot Boolean Expression

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

Sol:- The expression has 4 variables and hence it can be plotted using 4-variable K-map

AB \ CD	$\bar{C}\bar{D}$ 00	$\bar{C}D$ 01	CD 11	$C\bar{D}$ 10
$\bar{A}\bar{B}$ 00	1 (0)	0 (1)	0 (3)	1 (2)
$\bar{A}B$ 01	1 (4)	1 (5)	1 (7)	0 (6)
AB 11	1 (12)	0 (13)	1 (15)	0 (14)
$A\bar{B}$ 10	1 (8)	1 (9)	1 (11)	1 (10)

$A+B+C+\bar{D} = M_1$

$A+\bar{B}+\bar{C}+D = M_6$

$A+B+\bar{C}+\bar{D} = M_3$

$\bar{A}+\bar{B}+C+\bar{D} = M_{13}$

$\bar{A}+\bar{B}+\bar{C}+D = M_{14}$

Karnaugh Maps to obtain Minimal Expressions for Complete Boolean Functions:-

Once the Boolean function is plotted on the K-map we have to use grouping technique to simplify the Boolean function

→ The simplification is achieved by groupling adjacent 1's (or) 0's in groups of 2^i, where $i = 1, 2, \ldots n$ and n is the no. of variables.

→ when adjacent 1's are grouped then we get result in the "Sum of Products form". otherwise we get result in the "Product of Sum form".

Grouping Two adjacent 'ones' Pair:-

Eg:- $Y = \bar{A}\bar{B}C + \bar{A}BC.$

In the above expression only 'B' as a variable appears in both normal and complemented form. ($\bar{A}$ & C remains unchange). These two terms can be combined to give a resultant that eliminates 'B' variable

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

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

$= \bar{A}C$

A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}$ 0	0	1	1	0
A 1	0	0	0	0

→ $\bar{A}C$

Eg: $Y = \bar{A}BC + ABC.$

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

$= BC.$

A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}$ 0	0	0	1	0
A 1	0	0	1	0

→ BC

Eg: $Y = A\bar{B}\bar{C} + AB\bar{C}$

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

$= A\bar{C}$

A \ BC	00 $\bar{B}\bar{C}$	01 $\bar{B}C$	11 BC	10 $B\bar{C}$
$\bar{A}$ 0	0	0	0	0
A 1	1	0	0	1

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

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

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

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

→ $\bar{B}\bar{C}D$

In map the top & bottom row are considered to be adjacent

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

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

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

$= \bar{A}C + BC$

A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}$ 0	0	1	1	0
A 1	0	0	1	0

→ Group 1 → $\bar{A}C$

→ Group 2 → BC

This shows that we can share one term b/w two pair's

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

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

$= \bar{A}C + AB$

A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}$ 0	0	1	1	0
A 1	0	0	1	1

→ Group 1 → $\bar{A}C$

→ Pair not required

→ Group 2 → AB

⇒ Grouping Four Adjacent Ones (Quad):-

In K-map we can group four adjacent 1's. The resultant group is called "Quad".

① $Y = A$

A \ BC	$\bar{B}\bar{C}$ 00	$\bar{B}C$ 01	BC 11	$B\bar{C}$ 10
$\bar{A}$ 0	0	0	0	0
A 1	1	1	1	1

→ A.

② $Y = CD$

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

→ CD

③ $Y = BD$

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

→ BD

④ $Y = A\bar{D}$

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

⑤ $Y = \bar{B}\bar{D}$

AB \ CD	00	01	11	10
$\bar{A}\bar{B}$ 00	1	0	0	1
$\bar{A}B$ 01	0	0	0	0
AB 11	0	0	0	0
$A\bar{B}$ 10	1	0	0	1

G. NAVEEN
EEE

$\therefore Y = AB + AD + AC$

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

→ AC
AB ←
→ AD

→ **Grouping Eight adjacent ones (octet):—**

→ Grouping of eight adjacent 1's is called octet.

→ either you can group horizontally, Vertically

→ when an octet is combined in a four variable map, three of four variables are eliminated because only one variable remains unchanged.

Eg:- $Y = \bar{A}B\bar{C}\bar{D} + \bar{A}B\bar{C}D + \bar{A}BCD + \bar{A}BC\bar{D} + AB\bar{C}D + AB\bar{C}\bar{D} + ABCD + ABC\bar{D}$

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

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

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

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

$= B.$

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

→ B

$Y = B$

Prime Implicant:—

It is a smallest possible product term removing of any literal from which is not possible. The product must be an implicant of a given function for it to be a prime implicant

→ The bunch of 1's on the K-map which form a 2-square, 4-square etc.. is called Prime Implicant (PI) (or) sub cube.

Eg:- obtain the prime Implicant for given Boolean expression using K-map

$f(A,B,C) = (0,1,3,5,7)$

Sol:- $\therefore f(A,B,C) = \bar{A}\bar{B} + C$

$\therefore$ the prime implicants for given Boolean expression are $\bar{A}\bar{B}$ and C.

A \ BC	00	01	11	10
0	1	1	1	0
1	0	1	1	0

— C
$\bar{A}\bar{B}$

Eg:- $f(A,B,C) = \Sigma m(1,5,7,8,10,12,13,15)$

Sol: from K-map $F = (A\bar{B}\bar{D} + AB\bar{C} + \bar{A}\bar{C}D + BD)$

let $h_1 : A\bar{B}\bar{D}$

if A is removed $\bar{B}\bar{D} \rightarrow (0, 1, 8, 10)$ is not an implicant

$\bar{B}$ " $A\bar{D} \rightarrow (8, 10, 12, 14)$ " "

$\bar{D}$ " $A\bar{B} \rightarrow (8, 9, 10, 13)$ " "

$\therefore h_1$ is a prime implicant

AB \ CD	00	01	11	10
00		1		
01		1	1	
11	1	1	1	
10	1			1

Essential Prime Implicants:-

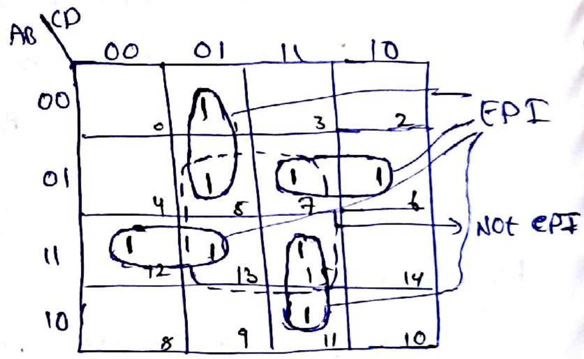
It is the prime implicant, It must cover at least one mainterm which is not covered by any other P.I.

Procedure to identify only essential Prime implicants:-

1. Plot the K-map and place 1s in those cells corresponding to the 1s in the truth table (or) SOP expression, place 0's in other cells.
2. check the K-map for adjacent 1s and encircle those 1s which are not adjacent to any other 1s. these are called isolated 1's
3. check for those 1's which are adjacent to only one other 1 and encircle such pairs
4. check for quads and octets of adjacent 1s even if it contains some 1s that have already been encircled; however there should be at least one '1' that has not yet been encircled.

Eg:- obtain the essential Prime implicants of following Boolean Expression

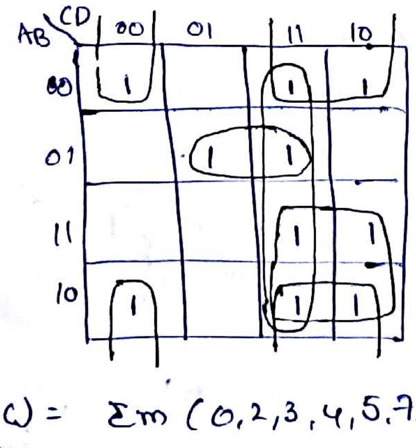
$f(A,B,C,D) = \Sigma m(1,5,6,7,11,12,13,15)$



Sol: In K-map cell's 1,6,12 & 11 have only one adjacent 1 in cells, 5,7,13 & 15 resp. so these cells are grouped to form four pair's. these four pair include all 1's in the K-map and hence the ~~ess~~ essential prime Implicants.

The Prime implicants form by quad group shown by dotted lines is not essential prime Implicants

Eg: $f(A,B,C,D) = \Sigma m(0,2,3,5,7,8,10,11,14,15)$



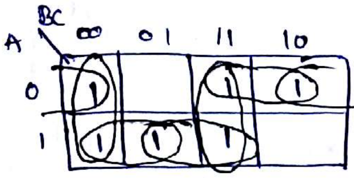
$= CD + \bar{B}\bar{D} + AC + \bar{A}BD + \bar{B}C$

$\hookrightarrow$ P.I

EPI $= \bar{B}\bar{D} + \bar{A}BD + AC$

Eg:- $f(A,B,C) = \Sigma m(0,2,3,4,5,7)$

Sol:



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

$\hookrightarrow$ P.I

no. of PI $= 6 \Rightarrow (0,2),(2,3),(3,7)(5,7)(4,5)(0,4)$

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

no. of EPI : 0

G.NAVEEN
EEE

Simplification of SOP Expression:-

→ The combination of pair's, quads and octets on a Karnaugh map can be used to obtain a simplified expression.

→ A pair of 1's Eliminates one variables

→ A quad of 1's eliminates two variables

→ An octet of 1's eliminates three variables.

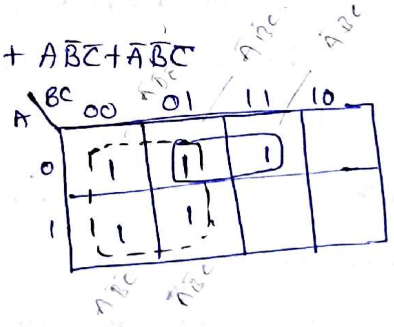
In general, when a variable appears in both complemented and un complemented form with in a group that variable eliminated from a resultant expression.

Generalised Procedure to Simplify boolean expression:-

1. Plot the K-map and place 1's in those cells corresponding to the 1's in the truth table (or) Sum of product expression place 0's in other cells.
2. check the K-map for adjacent 1's and then circle those 1's which are not adjacent to any other 1's. These are called isolated 1's
3. check for those 1's which are adjacent to only one other 1 and encircle such pair's
4. check for quads and octet's of adjacent 1's even if it contains some 1's that have already been encircled, while doing this make sure that there are minimum number of groups.
5. Combine any pair's necessary to include any 1's that have not yet been grouped.
6. From the simplified expression by summing product terms of all the group's.

Eg:- minimize the Expression $Y = A\bar{B}C + \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}\bar{C} + \bar{A}\bar{B}\bar{C}$

Sol:-

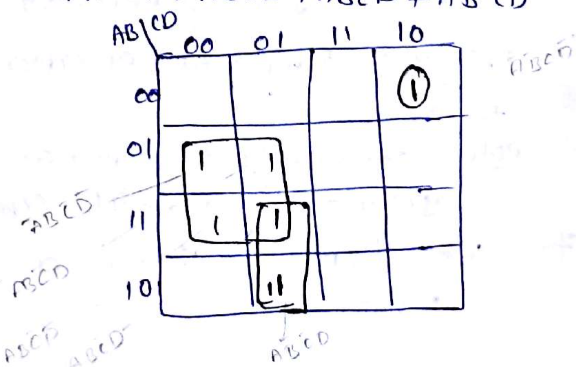


1. Sea for isolated one's. In this there are no isolated ones.
2. sea for adjacent pair of 1's - 1
3. Sea for quard - 1

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

Eg:- minimize the expresion $Y = \bar{A}BC\bar{D} + \bar{A}B\bar{C}D + ABC\bar{D} + AB\bar{C}D + A\bar{B}\bar{C}D + \bar{A}\bar{B}C\bar{D}$

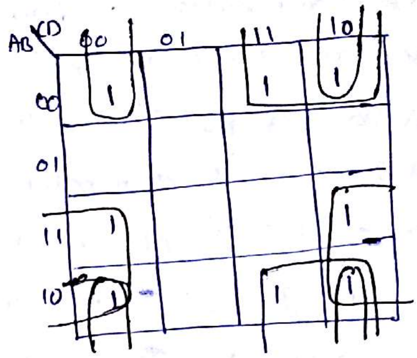
Sol:-



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

$: B\bar{C} + A\bar{C}D + \cancel{BC}\ \bar{A}\bar{B}C\bar{D}$

Eg: minimize the expression $Y = \bar{A}\bar{B}C\bar{D} + ABC\bar{D} + A\bar{B}C\bar{D} + A\bar{B}\bar{C}\bar{D} + AB\bar{C}\bar{D} + \bar{A}\bar{B}CD + \bar{A}\bar{B}\bar{C}\bar{D}$



Sol:

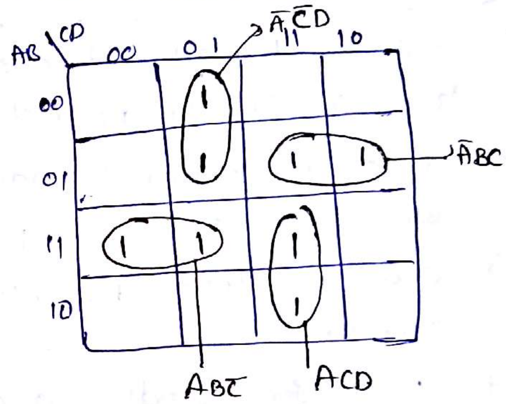
$Y = \bar{B}\bar{D} + \bar{B}C + A\bar{D}$

Eg: Reduce the following function to its minimum SOP form

$Y = \bar{A}\bar{B}\bar{C}D + \bar{A}B\bar{C}D + \bar{A}BCD + \bar{A}BC\bar{D} + AB\bar{C}\bar{D} + AB\bar{C}D + ABCD + A\bar{B}CD$

Sol:
→ there are no isolated 1s
→ there is no octet, but there is a quad. However all 1s in the quad have already been grouped. ∴ This quad is ignored.

$Y = \bar{A}\bar{C}D + \bar{A}BC + AB\bar{C} + ACD$



Simplification of POS Expression:—

Once the expression is plotted on the K-map instead of making the group of ones, we have to make groups of zero.

Steps:—

1. plot the K-map and place 0's in those cells corresponding to the 0's in the truth table or maxterms in the products of sum expression.
2. check the K-map for adjacent 0's and encircle those 0's which are not adjacent to any other 0's. these are called isolated 0's.
3. check for those 0's which are adjacent to only one other 0's and encircle such pairs.
4. check for quads and octets of adjacent 0's even if it contains some 0's that have already been encircled. while doing this make sure that there are minimum number of groups.
5. combine any pairs necessary to include any 0's that have not been grouped.
6. From the simplified SOP expression for $\bar{F}$ by summing product terms of all the groups.

   Note:- The simplified expression is in the complemented form because we have grouped 0's to simplify expression

7. use "DeMorgan's theorem" on $\bar{F}$ to produce the simplified expression in POS form.

G.NAVEEN
EEE

Eg:- Minimize the expression into POS (max terms)

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

Sol: $(A+B+\bar{C}) = M_1$, $(A+\bar{B}+\bar{C}) = M_3$, $(\bar{A}+\bar{B}+\bar{C}) = M_7$, $(\bar{A}+B+C) = M_4$, $(A+B+C) = M_0$

A \ BC	00	01	11	10
0	0	0	0	
1	0		0	

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

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

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

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

Eg:- Minimize the following expression into POS form

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

Sol: $(\bar{A}+\bar{B}+C+D) = M_{12}$, $(\bar{A}+\bar{B}+\bar{C}+D) = M_{14}$, $(\bar{A}+\bar{B}+\bar{C}+\bar{D}) = M_{15}$, $(\bar{A}+B+C+D) = M_8$,
$(A+\bar{B}+\bar{C}+D) = M_6$, $(A+\bar{B}+\bar{C}+\bar{D}) = M_7$, $(A+B+C+D) = M_0$, $(\bar{A}+\bar{B}+C+\bar{D}) = M_{13}$

There is no isolated 0's

AB \ CD	00	01	11	10
00	0			
01			0	0
11	0	0	0	0
10	0			

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

$Y = \bar{\bar{Y}} = \overline{BC + AB + \bar{B}\bar{C}\bar{D}}$

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

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

Eg:- Reduce the following function using k-map technique.

$f(A,B,C,D) = \pi M(0,2,3,8,9,12,13,15)$

Sol:-

AB \ CD	00	01	11	10
00	0		0	0
01				
11	0	0	0	
10	0	0		

$F = A\bar{C} + ABD + \bar{A}\bar{B}C + \bar{A}\bar{B}\bar{D}$

$\bar{\bar{F}} = \overline{A\bar{C} + ABD + \bar{A}\bar{B}C + \bar{A}\bar{B}\bar{D}}$

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

~~Don't care conditions~~:—

Five Variable K-Map:—

→ A 5-Variable (A, B, C, D, E) expression (or) K-map require $2^5 = 32$ cells. Input Variables such as $\bar{A}\bar{B}\bar{C}\bar{D}\bar{E}$, $\bar{A}\bar{B}\bar{C}\bar{D}E$ --- ABCDE, with minterm designations m_0, m_1, m_2 ---- m_{31} respectively.

→ In POS form $A+B+C+D+E$, $A+B+C+D+\bar{E}$ ---- $A+\bar{B}+\bar{C}+\bar{D}+\bar{E}$ with maxterm designations $M_0, M_1, \ldots M_{31}$ resp.

→ The 32 squares of K-map are divided into 2 blocks of 16 squares each. One map is used for A and other for $\bar{A}$.

→ The left block represents minterms from m_0 to m_{15} in which A is a '0'. & the Right block " minterms from m_{16} to m_{31} in which A is '1'.

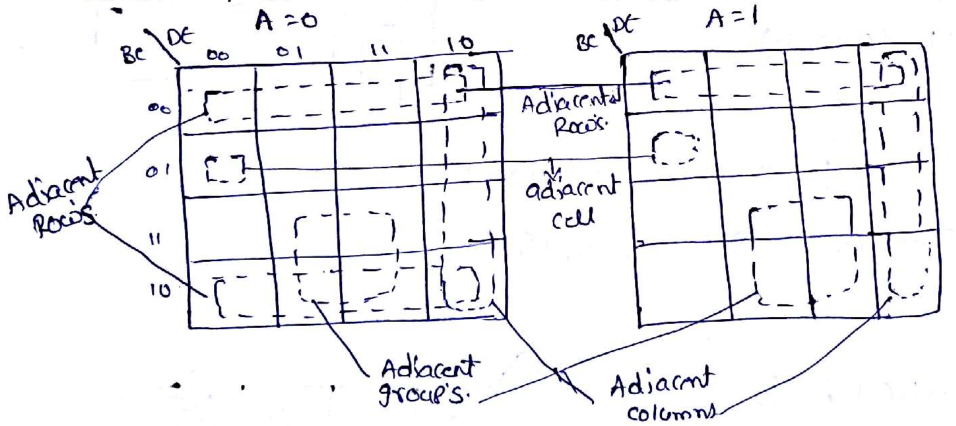
→ In order to identify the adjacent grouping in the 5 Variable we must imagine the two map's superimposed on one another.

→ Every cell in one map is adjacent to the corresponding cell in the other map, because only one Variable changes between such corresponding cells.

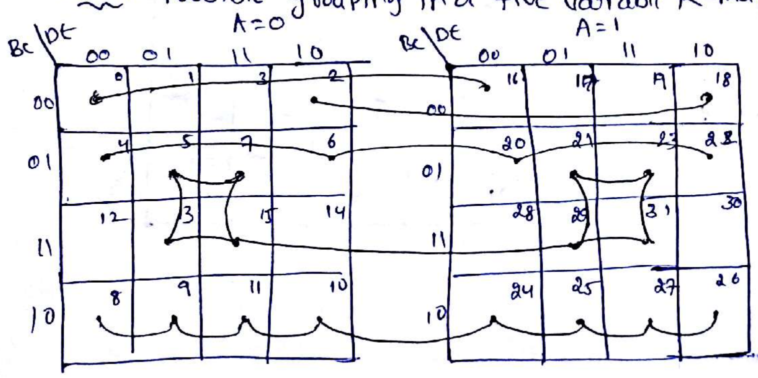
→ Every row on one map is adjacent to the corresponding row.

→ The right most and left most columns which within each 16-cell map are adjacent, Just as they are in any 16-cell map.

However the right most column of one map is not adjacent to the left most column of other map. Since those are not corresponding columns.



→ Some Possible grouping in a five Variable K-map:—



(a) $m_0, m_{16} = \bar{B}\bar{C}\bar{D}\bar{E}$ ⇒ $M_0 M_{16} = B+C+D+E$

(b) $m_2, m_{18} = \bar{B}\bar{C}D\bar{E}$ ⇒ $M_2 M_{18} = B+C+\bar{D}+E$

(c) $m_4, m_6, m_{20}, m_{22} = \bar{B}C\bar{E}$

⇒ $M_4, M_6, M_{20}, M_{22} = B+\bar{C}+E$

(d) $m_5, m_7, m_{13}, m_{15}, m_{21}, m_{23}, m_{29}, m_{31} = CE$

⇒ $M_5, M_7, M_{13}, M_{15}, M_{21}, M_{23}, M_{29}, M_{31} = \bar{C}+\bar{E}$

(e) $m_8, m_9, m_{10}, m_{11}, m_{24}, m_{25}, m_{26}, m_{27} = B\bar{C}$

→ $M_8, M_9, M_{10}, M_{11}, M_{24}, M_{25}, M_{26}, M_{27} = \bar{B}+C$.

G. NAVEEN
EEE

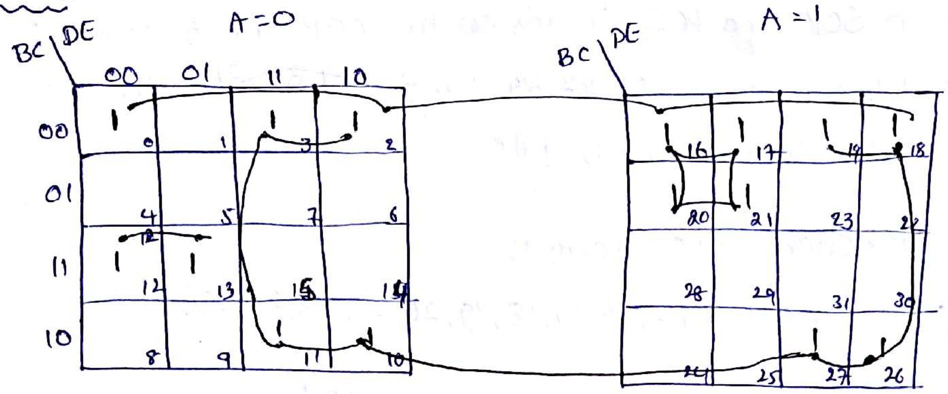
Eg:- Reduce the following expression in SOP and POS forms using mapping

$f = \Sigma m(0,2,3,10,11,12,13,16,17,18,19,20,21,26,27)$

Sol:- Given expression in POS form is

$f = \Pi M(1,4,5,6,7,8,9,14,15,22,23,24,25,28,29,30,31)$

SOP K-Map:-



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

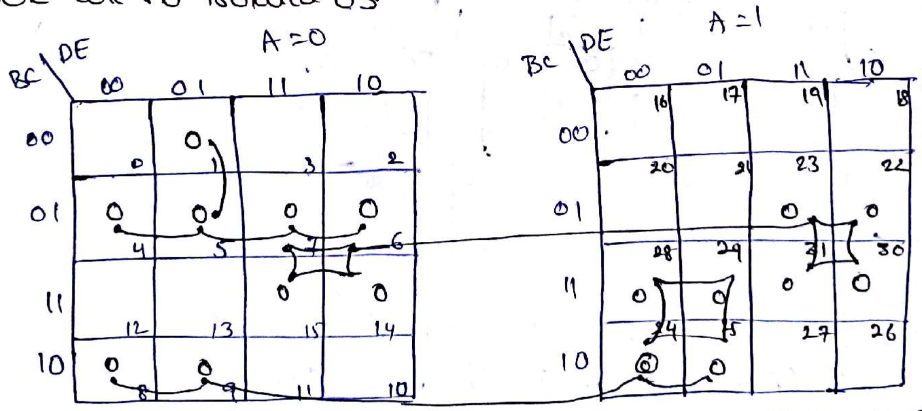
Reduction steps:-

1. there are no isolated 1's
2. m_{12} can go only with m_{13}. Form a 2-square which is read as $\bar{A}BC\bar{D}$
3. m_0 can go with m_2 m_{16} & m_{18}. So form a 4-square which is read as $\bar{B}\bar{C}\bar{E}$
4. $m_{20}, m_{21}, m_{17}, \& m_{16}$ form a 4-square which is read as $A\bar{B}\bar{D}$
5. $m_2, m_3, m_{18}, m_{19}, m_{10}, m_{11}, m_{26}$ & m_{27} form an 8-square which is read as $\bar{C}D$
6. write all the product terms in SOP form.

$f_{min} = \bar{A}BC\bar{D} + \bar{B}\bar{C}\bar{E} + A\bar{B}\bar{D} + \bar{C}D$

POS-K-map:-

1. There are no isolated 0's



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

2. M_1 can go only with M_5 (or) M_9 make a 2 square with M_5 which is read as $(A+B+D+\bar{E})$
3. M_4 can go with M_5, M_7 & M_6 to form a 4-square, which is read as $(A+B+\bar{C})$
4. M_8 " " M_9, M_{24} & M_{25} " " " $(\bar{B}+C+D)$
5. M_{28} " " M_{29}, M_{24} & M_{25} " " " $(\bar{A}+\bar{B}+D)$
6. M_{30} can make a 4 square with M_{31}, M_{29} & M_{28} (or) with M_{31}, M_{14} & M_{15} (or) with.

M_{31}, M_{22} & M_{23}. Don't do that. Note that it can make an 8-square with M_{31}, M_{23}; M_{22}, M_6, M_7, M_{14} & M_{15} which is read as $(\bar{C}+\bar{D})$

7- write all pos sumterms in pos form.

$\therefore F = (A+B+D+E)(A+B+\bar{C})(\bar{B}+C+D)(\bar{A}+\bar{B}+D)(\bar{C}+\bar{D})$

Eg:- Minimize in SOP and POS forms on the map, the 5-variable function is $F = \Sigma m(0,1,4,5,6,13,14,15,22,24,25,28,29,30,31)$. Implement the minimal expression using NAND gate

Sol:- The given function in pos form is

$F = \pi M(2,3,7,8,9,10,11,12,16,17,18,19,20,21,23,26,27)$.

for SOP:-

A = 0

BC \ DE	00	01	11	10
00	0: Φ	1: 1	3:	2:
01	4: 1	5: 1	7:	6: 1
11	12:	13: 1	15: 1	14: 1
10	8:	9:	11:	10:

A = 1

BC \ DE	00	01	11	10
00	16:	17:	19:	18:
01	20:	21:	23:	22: 1
11	28: 1	29: 1	31: 1	30: 1
10	24: 1	25: 1	27:	26:

$\therefore f_{min} = \bar{A}\bar{B}\bar{D} + CD\bar{E} + BCE + AB\bar{D} = \overline{(\overline{\bar{A}\bar{B}\bar{D}})(\overline{CD\bar{E}})(\overline{BCE})(\overline{AB\bar{D}})}$

for POS:-

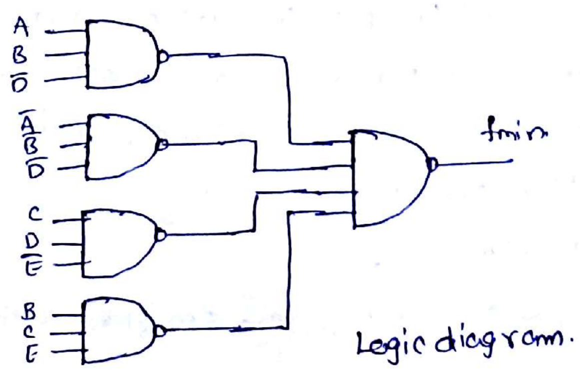
A = 0

BC \ DE	00	01	11	10
00	0:	1:	3: 0	2: 0
01	4:	5:	7: 0	6:
11	12: 0	13:	15:	14:
10	8: 0	9: 0	11: 0	10: 0

A = 1

BC \ DE	00	01	11	10
00	16: 0	17: 0	19: 0	18: 0
01	20: 0	21: 0	23: 0	22:
11	28:	29:	31:	30:
10	24:	25:	27: 0	26: 0

$f_{min} = (A+\bar{B}+D+E)(A+\bar{B}+C)(B+\bar{D}+\bar{E})(\bar{A}+B+D)(C+\bar{D})$

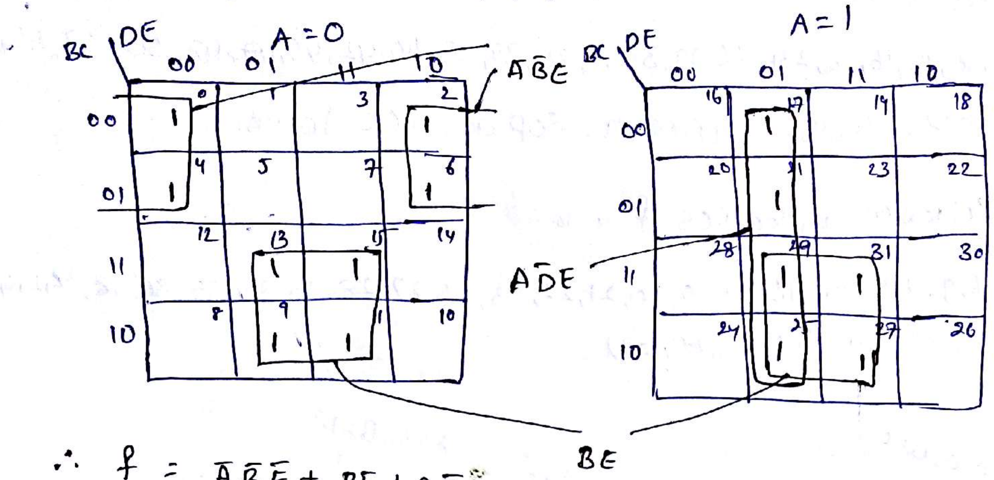


Logic diagram.

G NAVEEN
EEE

Eg:- $f(A,B,C,D,E) = \Sigma m(0,2,4,6,9,11,13,15,17,21,25,27,29,31)$

Sol:-



$\therefore f = \bar{A}\bar{B}\bar{E} + BE + A\bar{D}E$

Six-Variable K-map:—

A six variable (A, B, C, D, E, F) expression can have $2^6 = 64$ possible combinations. In SOP form designations $m_0, m_1, \dots m_{63}$ resp. In POS form designations $M_0, M_1, \dots M_{63}$ resp. The 64 square's of K-map are divided into four blocks of 16 squares each. Each square on the map represents a minterm (or) a maxterm.

The top left block represents minterms (m_0 to m_{15}) in which A is a '0' and B is '0'.
The top right " " " (m_{16} to m_{31}) in " A is a '0' and B is '1'
The bottom left " " " (m_{32} to m_{47}) " " A is a '1' and B is '0'
The bottom right " " " (m_{48} to m_{63}) " " A is a '1' and B is '1'.

A B, CD EF — A = 0 B = 0

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

CD EF — A = 0 B = 1

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

CD EF — A = 1 B = 0

32	33	35	34
36	37	39	38
44	45	47	46
40	41	43	42

CD EF — A = 1, B = 1

48	49	51	50
52	53	55	54
60	61	63	62
56	57	59	58

→ Diagonal Elements like m_0, m_{58}; m_{15}, m_{63}; m_{18}, m_{34}; m_{29}, m_{45} are not adjacent to each other

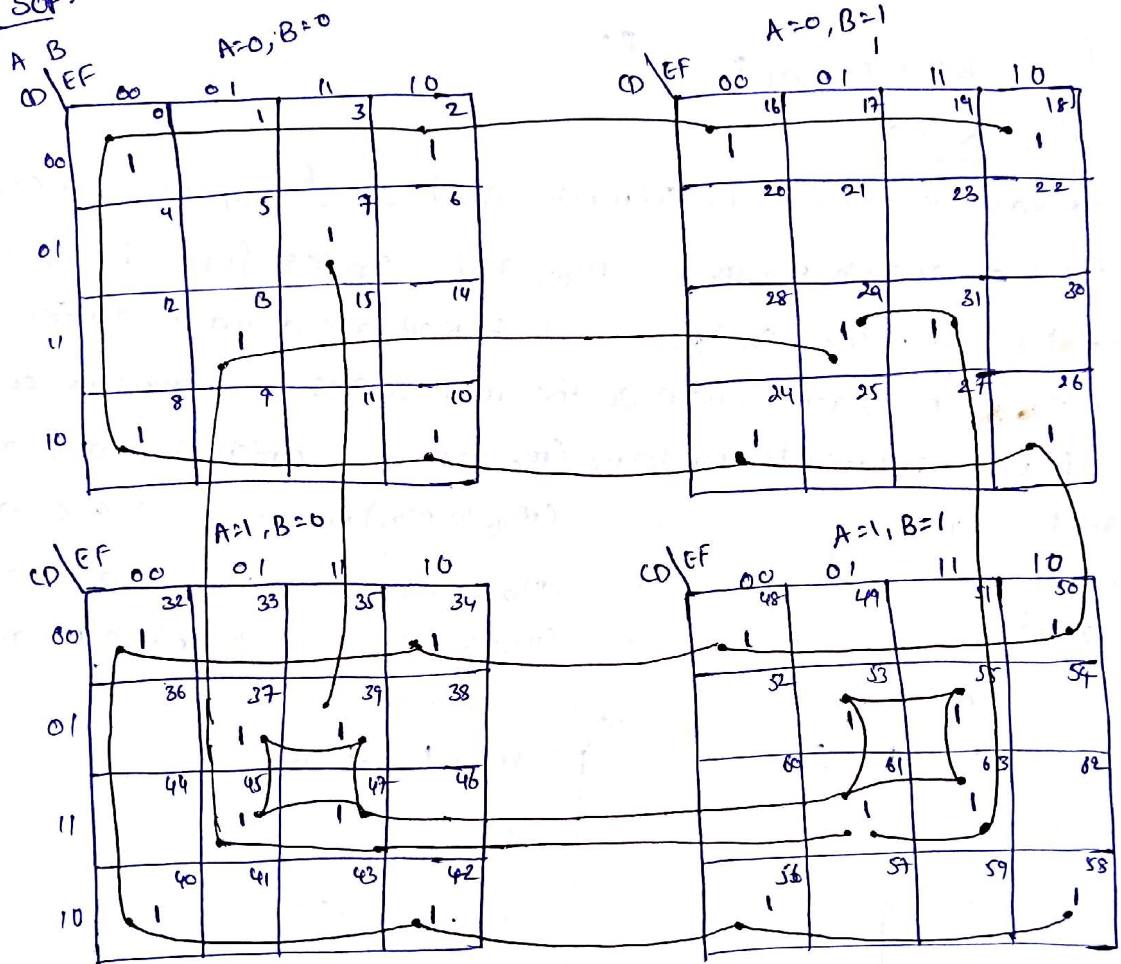
Eg:- Reduce the expression

$f = \Sigma m(0,2,7,8,10,13,16,18,24,26,29,31,32,34,37,39,40,42,45,47,48,50,53,55,56,58,61,63)$ using mapping in SOP and POS forms.

Sol:- The given expression in the POS form is

$f = \pi M(1,3,4,5,6,9,11,12,14,15,17,19,20,21,22,23,25,27,28,30,33,35,36,38,41,43,44,46,49,51,52,54,57,59,60,62)$

for SOP:-



Reduction steps: $f_{min} = \bar{B}\bar{C}DEF + CD\bar{E}F + BCDF + ADF + \bar{D}\bar{F}$

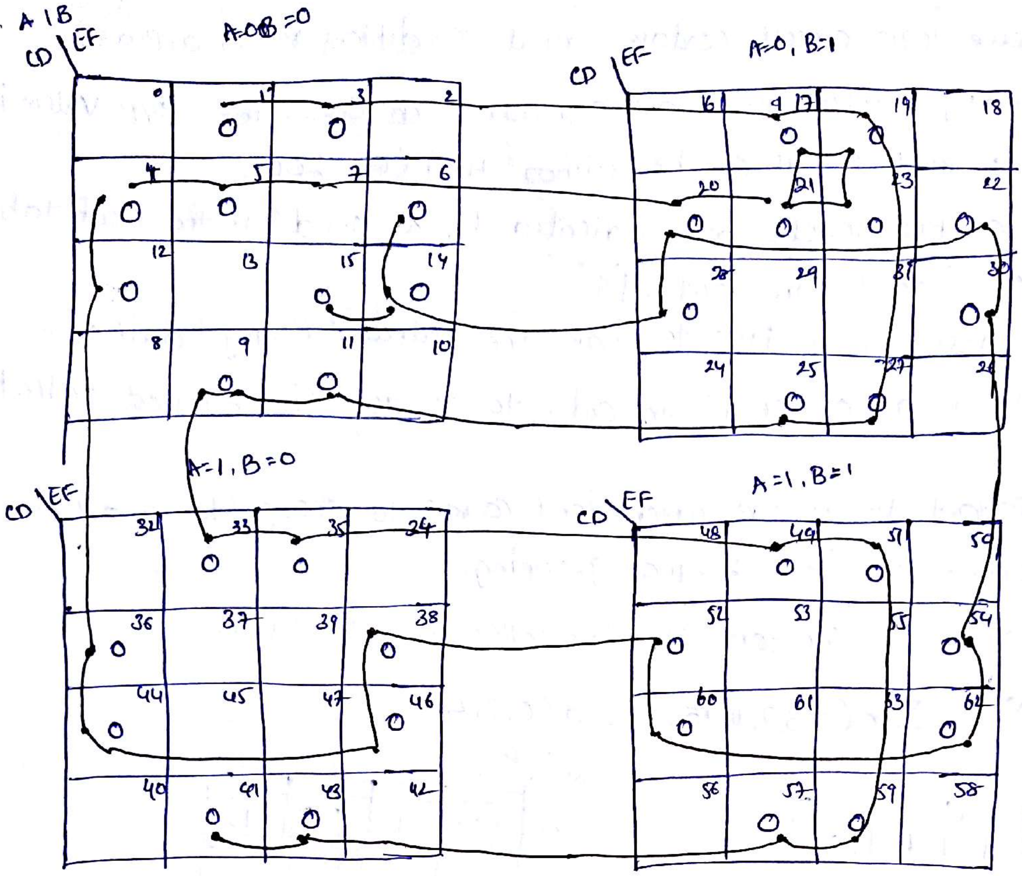
1. There are no isolated 1's
2. m_7 has only one adjacency m_{39}, It can form a 2-square with m_{39}, i.e. $\bar{B}\bar{C}DEF$
3. m_{13} can make a 4-square with m_{29}, m_{45}, m_{61}. ie $CD\bar{E}F$
4. m_{31} can make a 4-square with m_{29}, m_{63}, m_{61}. ie $BCDF$
5. m_{55} can make an 8-square with $m_{53}, m_{61}, m_{63}, m_{37}, m_{39}, m_{45}, m_{47}$ ie ADF
6. m_0 can make a 16-square with $m_2, m_{16}, m_{18}, m_8, m_{10}, m_{24}, m_{26}, m_{50}, m_{48}, m_{34}, m_{32}, m_{40}, m_{42}, m_{56}, m_{58}$ ie $\bar{D}\bar{F}$
7. write all the product terms in SOP form.

∴ The minimal SOP expression is

$f_{min} = \bar{B}\bar{C}DEF + CD\bar{E}F + BCDF + ADF + \bar{D}\bar{F}$

G.NAVEEN
EEE

POS K-map:—



∴ $f_{min} = (A+B+\bar{C}+\bar{D}+\bar{E})(\bar{D}+F)(D+\bar{F})(A+\bar{B}+C+\bar{F})(A+C+\bar{D}+E)$

Reduction steps:—

1. There are no isolated 0's
2. M_{15} has only two adjacencies M_{14}, M_{11}. It can make a 2-square with any one of them make a 2-square of M_{15}, M_{14}. Read it as $(A+B+\bar{C}+\bar{D}+\bar{E})$
3. M_5 can make a 4-square with M_4, M_{20}, M_{21} (or) with M_1, M_{17}, M_{21}. Donot take a decision yet
4. M_4 can be expanded into a 16-square with $M_6, M_{12}, M_{14}, M_{20}, M_{22}, M_{28}, M_{30}, M_{36}, M_{38}, M_{44}, M_{46}, M_{52}, M_{54}, M_{60}$ and M_{62}. i.e $(\bar{D}+F)$
5. M_1 can be expanded into a 16-square with $M_3, M_9, M_{11}, M_{17}, M_{19}, M_{25}, M_{27}, M_{33}, M_{35}, M_{41}, M_{43}, M_{49}, M_{51}, M_{57}$ & M_{59}. Read it as $(D+\bar{F})$
6. only M_5, M_{21} and M_{23} are left uncovered. M_{21} & M_{23} can form a 4-square with M_{20}, M_{22} (or) with M_{17}, M_{19}. which are already taken care of. Form a 4-square of M_{21}, M_{23}, M_{17} & M_{19}. Read it as $(A+\bar{B}+C+\bar{F})$
7. only M_5 is left. make a 4-square: say with M_4, M_{20} & M_{21}. i.e $(A+C+\bar{D}+E)$
8. write all the sum terms in POS form.

∴ POS expression is

$f_{min} = (A+B+\bar{C}+\bar{D}+\bar{E})(\bar{D}+F)(D+\bar{F})(A+\bar{B}+C+\bar{F})(A+C+\bar{D}+E)$

Don't care Combinations (or) optional Combinations :—

In some logic circuit, certain input conditions never appears.
∴ the corresponding output never appears. In such cases the o/p value is not designed (or) defined. It can be either High (or) Low.

These output levels are indicated by 'x' (or) 'd' in the truth table and are called "don't care outputs"

→ A circuit designer is free to make the output for any "don't care" condition either a '0' or '1' in order to produce the simplest output expression.

→ It is important to decide which don't care's to change to and which to 1 to produce the best K-map grouping.

Eg:- Find the reduced SOP form of the following function

$$f(A,B,C,D) = \Sigma m(1,3,7,11,15) + \Sigma d(0,2,4)$$

Sol:-

AB \ CD	00	01	11	10
00	X	1	1	X
01	X	0	1	0
11	0	0	1	0
10	0	0	1	0

⟹

AB \ CD	00	01	11	10
00	1	1	1	1
01	0	0	1	0
11	0	0	1	0
10	0	0	1	0

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

To form a quad of cells 0, 1, 2, and 4 do the don't care conditions 0 & 2 are replaced by 0. ∴ It is not required to form any group

$$\therefore f = \bar{A}\bar{B} + CD$$

Eg:- Reduce the following function using K-map

$$f(A,B,C,D) = \Sigma m(5,6,7,12,13) + \Sigma d(4,9,14,15)$$

Sol:-

AB \ CD	00	01	11	10
00	0	0	0	0
01	X	1	1	1
11	1	1	X	X
10	0	X		0

⟹

AB \ CD	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	1	1	1	1
10	0	0	0	0

$f(A,B,C,D) = B$

To form a octet of cells 4, 5, 6, 7, 12, 13, 14 and 15. the don't care conditions 4, 14 & 15 are replaced by 1's. the remaining don't care condition 9 is replaced by '0' to get simplified function

$$\therefore f = B$$

G.NAVEEN
EEE

Eg:- Implement the following using logic gates.

$f(A,B,C) = \Sigma(0,1,3,7) + \Sigma d(2,5)$

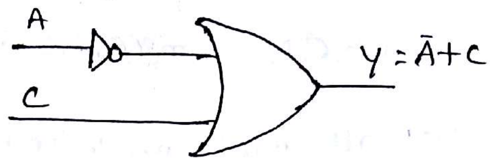
Sol:-

A \ BC	00	01	11	10
0	1	1	1	X
1	0	X	1	0

To form two quads both don't care conditions are replaced by 1's & we get

A \ BC	00	01	11	10
0	1	1	1	1
1		1	1	

$= \bar{A}+C$



Tabular Minimization (or) Tabular Quine-McCluskey Method:-

Let us consider two terms $AB\bar{C}$ & $A\bar{B}\bar{C}$. In this only B literal differs. The Binary equivalent to these minterms are 110 and 100. Note that the 2's place differs.

So minterms whose binary equivalent differ only in one place can be combined to reduce the minterms. This is the fundamental principle of the "Quine-McCluskey method".

eg:- if the terms are 0000 and 0010 then the resultant term will be 00-0. This term 00-0 is called an "implicant".

The process of comparison is repeated for every minterm. Once this process is completed the same process is applied to the new resultant terms which are placed in the column.

The remaining terms and all the terms that did not match during the process are called the "prime-implicants".

Summing one (or) more prime-implicants gives the simplified "Boolean expression".

Algorithm for Generating Prime Implicants:-

1. List all minterms in the binary form
2. Arrange the minterms according to number of 1's
3. Compare each binary number with every term in the adjacent next higher category and if they "differ only by one position". Put a check mark and copy the term in the next column with '-' in the position that they differed.

4. Apply the same process described in step-3 for the resultant column and continue these cycles until a single pass through cycle yields no further elimination of literals

5. List all prime Implicants

6. Select the minimum number of prime implicants which must cover all the min-terms

Eg:- Simplify the following boolean function by using Quine-McCluskey method $F(A,B,C,D) = \Sigma m(0,2,3,6,7,8,10,12,13)$

Sol: Step 1:- List all minterms in the binary form as shown in Table.

Step 2:- Arrange the minterms acc to number of 1's, as shown in Table column(b).

Minterm	Binary Representation
m_0	0000
m_2	0010
m_3	0011
m_6	0110
m_7	0111
m_8	1000
m_{10}	1010
m_{12}	1100
m_{13}	1101

column(a)

minterm	Binary Representation
m_0	0000 ✓
m_2	0010 ✓
m_8	1000 ✓
m_3	0011 ✓
m_6	0110 ✓
m_{10}	1010 ✓
m_{12}	1100 ✓
m_7	0111 ✓
m_{13}	1101 ✓

column(b)

Step 3: Compare each binary number with every term in the adjacent next higher category and if they "differ only by one position", put a check mark and copy the term in the next column with '–' in the position that they differ.

Step 4:-

minterms	Binary Representation
0, 2	00–0 ✓
0, 8	–000 ✓
2, 3	001– ✓
2, 6	0–10 ✓
2, 10	–010 ✓
8, 10	10–0 ✓
8, 12	1–00
3, 7	0–11 ✓
6, 7	011– ✓
12, 13	110–

column(c)

minterms	Binary Representation
0, 2, 8, 10	–0–0
2, 3, 6, 7	0–1–

column(d)

G. Naveen
EEE

Step 5:- List the prime implicants.

Prime Implicants		Binary Representation
$A\bar{C}\bar{D}$	8, 12	1 – 0 0
$AB\bar{C}$	12, 13	1 1 0 –
$\bar{B}\bar{D}$	0, 2, 8, 10	– 0 – 0
$\bar{A}C$	2, 3, 6, 7	0 – 1 –

Step 6: Select the minimum no. of prime Implicants which must cover all the minterms.

Prime Implicants		m_0 COL-1	m_2 COL-2	m_3 COL-3	m_6 COL-4	m_7 COL-5	m_8 COL-6	m_{10} COL-7	m_{12} COL-8	m_{13} COL-9
							⊙		⊙	
$A\bar{C}\bar{D}$	8, 12								○	○
$AB\bar{C}$	12, 13 ✓						○	○		
$\bar{B}\bar{D}$	0, 2, 8, 10 ✓	○	○							
$\bar{A}C$	2, 3, 6, 7 ✓		○	○	○	○				

$$\therefore \; F(A,B,C,D) = (1\,1\,0\,-) + (-\,0\,-\,0) + (0\,-\,1\,-)$$

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

Eg:- Simplify the following Boolean function by using Quine-McCluskey method.

$$F(A,B,C,D) = \Sigma m(0,2,3,8,10,11,12,14)$$

Sol:

Step 1: List all minterms in the binary form

Step 2: Arrange the minterms according to number of 1s.

Minterm	Binary representation	minterm	Binary representation
m_0	0000	m_0	0000 ✓
m_2	0010	m_2	0010 ✓
m_3	0011	m_8	1000 ✓
m_8	1000	m_3	0011 ✓
m_{10}	1010	m_{10}	1010 ✓
m_{11}	1011	m_{12}	1100 ✓
m_{12}	1100	m_{11}	1011 ✓
m_{14}	1110	m_{14}	1110 ✓
Column (a)		Column (b)	

Step 3: Compare each binary number

Step 4: Apply the same process described in step 3 for the resultant column.

Minterms	Binary representation
0, 2	00 - 0
0, 8	- 000
2, 3	001 -
2, 10	- 010
8, 10	10 - 0
8, 12	1 - 00
3, 11	- 011
10, 11	101 -
10, 14	1 - 10
12, 14	11 - 0
Column (c)	

minterm	Binary representation
0, 2, 8, 10	- 0 - 0
2, 3, 10, 11	- 01 -
8, 10, 12, 14	1 - - 0
column (d)	

Step 5: List the prime implicants.

Prime Implicants		Binary Representation
$\bar{B}\bar{D}$	0, 8, 2, 10	- 0 - 0
$\bar{B}C$	2, 10, 3, 11	- 01 -
$A\bar{D}$	8, 10, 12, 14	1 - - 0

Step 6: Select the minimum number of prime implicants which meet over all the minterms.

Prime Implicants	m_0	m_2	m_3	m_8	m_{10}	m_{11}	m_{12}	m_{14}
$\bar{B}\bar{D}$, 0, 8, 2, 10	⊙	⊙		⊙	⊙			
$\bar{B}C$ 2, 10, 3, 11		⊙	⊙		⊙	⊙		
$A\bar{D}$ 8, 10, 12, 14				⊙	⊙		⊙	⊙

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

G. NAVEEN

Eg:- Simplify the following using tabulation methods

$Y(A,B,C,D) = \Sigma m(1,2,3,5,9,12,14,15) + \Sigma d(4,8,11)$

Sol:

Don't care conditions are used to find the prime implicant but it is not compulsory to include don't care terms in the final expression.

Step 1:- List all minterms in the binary form.

Step 2:- Arrange the minterms according to no. of 1s

Minterm	Binary Representation
m_1	0001
m_2	0010
m_3	0011
m_5	0101
m_9	1001
m_{12}	1100
m_{14}	1110
m_{15}	1111
$d\,m_4$	0100
$d\,m_8$	1000
$d\,m_{11}$	1011

column(a)

Minterm	Binary Representation
m_1	0001 ✓
m_2	0010 ✓
m_4	0100 ✓
m_8	1000 ✓
m_3	0011 ✓
m_5	0101 ✓
m_9	1001 ✓
m_{12}	1100 ✓
m_{11}	1011 ✓
m_{14}	1110 ✓
m_{15}	1111 ✓

column(b)

Step 3:- Compare each binary number with next higher category and if they differ by only one position. copy the term in the next column with '–' in the position that they differed.

Step 4:- Repeat the step 3.

Minterm	Binary Representation
1,3	00–1 ✓
1,5	0–01
1,9	–001 ✓
2,3	001–
4,5	010–
4,12	–100
8,9	100–
8,12	1–00
3,11	–011 ✓
9,11	10–1 ✓
12,14	11–0
11,15	1–11
14,15	111–

Minterm	Binary representation
1,3,9,11	–0–1

Step 5:- List the Prime Implicants

Prime Implicants		Binary Representation
$\bar{A}\bar{C}D$	1,5	0 - 0 1
$\bar{A}\bar{B}C$	2,3	0 0 1 -
$\bar{A}B\bar{C}$	4,5	0 1 0 -
$B\bar{C}\bar{D}$	4,12	- 1 0 0
$A\bar{B}\bar{C}$	8, 9	1 0 0 -
$A\bar{C}\bar{D}$	8,12	1 - 0 0
$AB\bar{D}$	12,14	1 1 - 0
ACD	11,15	1 - 1 1
ABC	14,15	1 1 1 -
$\bar{B}D$	1,3,9,11	- 0 - 1

Step 6:- Select minimum number of Prime Implicants which must cover all the minterms except don't care minterms

Prime Implicants		m_1 (C-1)	m_2 (C-2)	m_3 C-3	m_4 C-4	m_5 C-5	m_8 C-6	m_9 C-7	m_{11} C-8	m_{12} C-9	m_{14} C-10	m_{15} C-11
$\bar{A}\bar{C}D$	1,5 ✓	⊙				⊙						
$\bar{A}\bar{B}C$	2,3 ✓		⊙	⊙								
$\bar{A}B\bar{C}$	4,5				•	•						
$B\bar{C}\bar{D}$	4,12				•					•		
$A\bar{B}\bar{C}$	8,9						•	•				
$A\bar{C}\bar{D}$	8,12						•			•		
$AB\bar{D}$	12,14 ✓									⊙	⊙	
ACD	11,15								•			•
ABC	14,15 ✓										⊙	⊙
$\bar{B}D$	1,3,9,11 ✓	⊙		⊙				⊙	⊙			

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

G. NAVEEN.
EEE Dept

Code-converters :— (using K-Map)

A code converter is a logic circuit whose inputs are bit patterns representing numbers (or characters) in one code and whose outputs are the corresponding representations in a different code. It makes two systems compatible even though each uses a different binary code.

Code converters are usually multiple output circuits.

There is a wide variety of binary codes used in digital systems. Some of these codes are binary-coded-decimal (BCD), Excess-3, Gray, and so on. Many times it is required to convert one code to another.

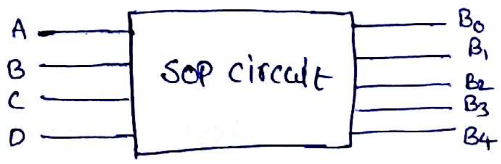
Binary to BCD converter:-

The input is a 4-bit binary. There are 16 possible combinations of 4-bit binary inputs (0-15) and all are valid. Hence there are no don't cares.

Truth table for Binary to BCD converter

Decimal	Binary code				BCD code				
	D	C	B	A	B_4	B_3	B_2	B_1	B_0
0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0	1
2	0	0	1	0	0	0	0	1	0
3	0	0	1	1	0	0	0	1	1
4	0	1	0	0	0	0	1	0	0
5	0	1	0	1	0	0	1	0	1
6	0	1	1	0	0	0	1	1	0
7	0	1	1	1	0	0	1	1	1
8	1	0	0	0	0	1	0	0	0
9	1	0	0	1	0	1	0	0	1
10	1	0	1	0	1	0	0	0	0
11	1	0	1	1	1	0	0	0	1
12	1	1	0	0	1	0	0	1	0
13	1	1	0	1	1	0	0	1	1
14	1	1	1	0	1	0	1	0	0
15	1	1	1	1	1	0	1	0	1

logic diagram (or) Block diagram



From truth table, we observe that the expressions for BCD outputs are as follows.

$B_4 = \Sigma m(10,11,12,13,14,15)$, $B_3 = \Sigma m(8,9)$

$B_2 = \Sigma m(4,5,6,7,14,15)$, $B_1 = \Sigma m(2,3,6,7,12,13)$, $B_0 = \Sigma m(1,3,5,7,9,11,13,15)$

→ Drawing the K-maps for the outputs & minimizing them we get minimal expressions for BCD outputs B_0, B_1, B_2, B_3, B_4 in terms of the 4-bit binary inputs A, B, C, D are as follows.

$B_4 = DC + DB$, $B_3 = D\bar{C}\bar{B}$, $B_2 = \bar{D}C + CB$, $B_1 = DC\bar{B} + \bar{D}B$, $B_0 = A$.

For B_0:-

DC \ BA	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	0	1	1	0
10	0	1	1	0

$B_0 = A$.

For B_1:-

DC \ BA	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	1	1	0	0
10	0	0	0	0

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

For B_2:-

DC \ BA	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	0	0	1	1
10	0	0	0	0

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

For B_3:-

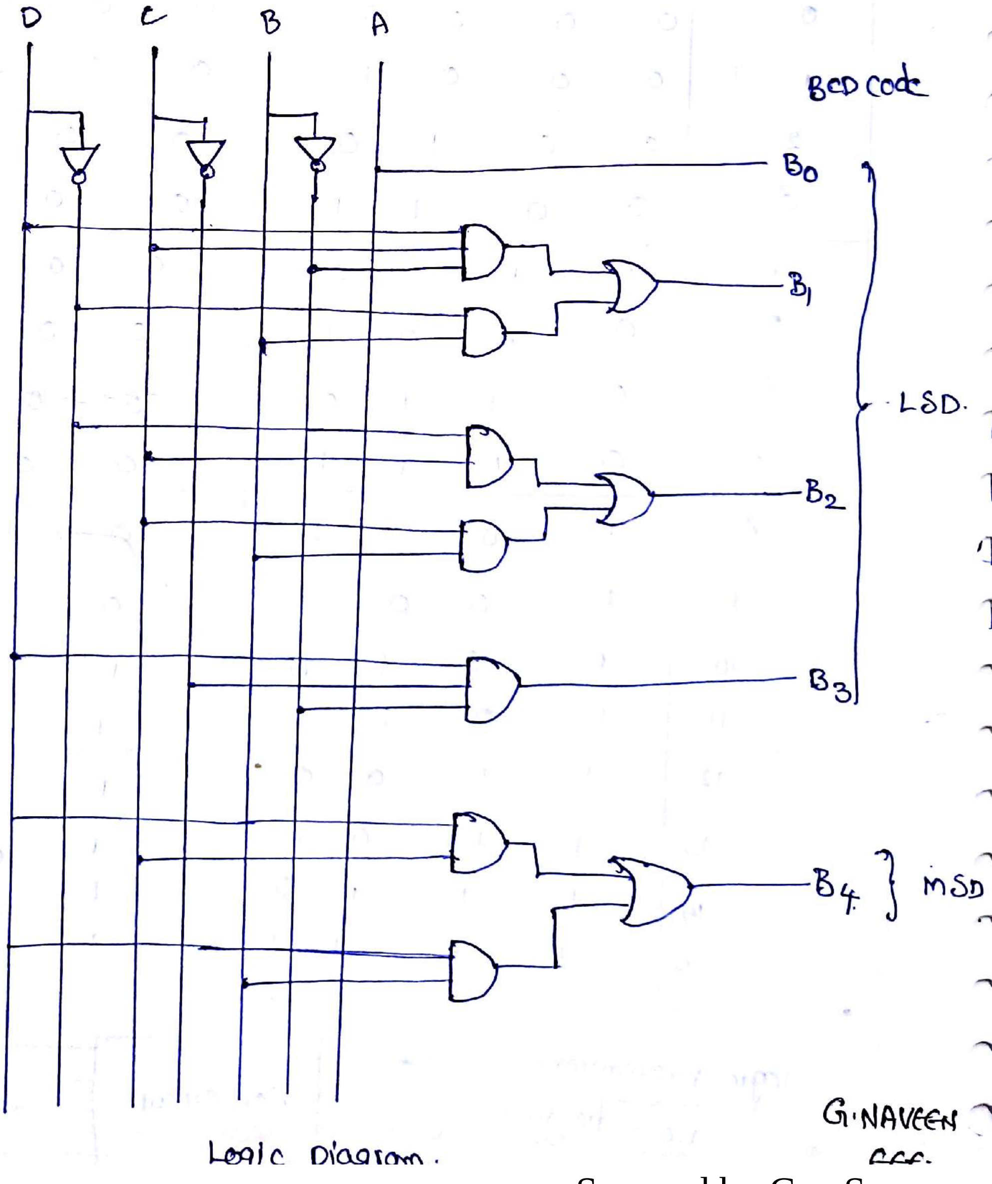
DC \ BA	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	0	0	0	0
10	1	1	0	0

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

For B_4:-

DC \ BA	00	01	11	10
00				
01				
11	1	1	1	1
10			1	1

$B_4 = DC + DB$



Logic Diagram.

G. NAVEEN
ECE.

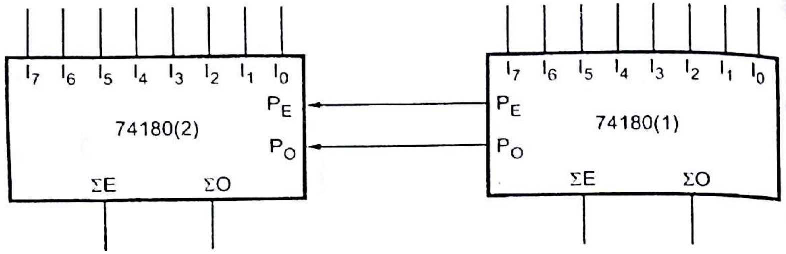


Fig. 4.88 Expansion to the larger word

4.5 Code Converters

There is a wide variety of binary codes used in digital systems. Some of the codes are binary-coded-decimal (BCD), Excess-3, Gray, and so on. Many times it is required to convert one code to another.

This section explains the design of various code converters.

4.5.1 Binary to BCD Converter

Let us see the truth table for binary to BCD converter

Binary code				BCD code				
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
0	0	1	1	0	0	0	1	1
0	1	0	0	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	0	0	1	1	1
1	0	0	0	0	1	0	0	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
1	1	0	0	1	0	0	1	0
1	1	0	1	1	0	0	1	1
1	1	1	0	1	0	1	0	0
1	1	1	1	1	0	1	0	1

Table 4.29

K-map simplification

For B_0

DC \ BA	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	0	1	1	0
10	0	1	1	0

$B_0 = A$

For B_1

DC \ BA	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	1	1	0	0
10	0	0	0	0

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

For B_2

DC \ BA	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	0	0	1	1
10	0	0	0	0

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

For B_3

DC \ BA	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	0	0	0	0
10	1	1	0	0

$B_3 = D\overline{C}\,\overline{B}$

For B_4

DC \ BA	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	0	0	1	1

$B_4 = DC + DB$

Fig. 4.89

Logic diagram

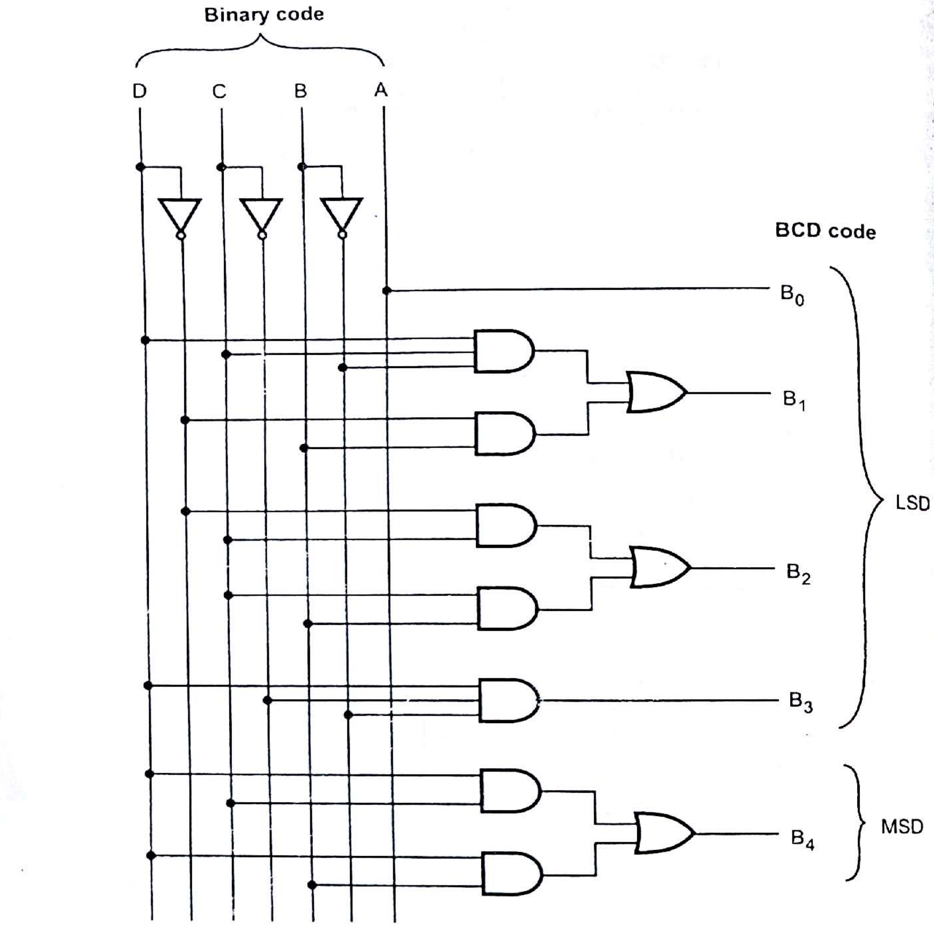


Fig. 4.90 Binary to BCD converter

4.5.2 BCD to Binary Converter

Let us see the truth table for BCD to binary converter.

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

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

Table 4.30 Truth table for BCD to binary converter

K-map simplification

For A

$B_4=0$

B_3B_2 \ B_1B_0	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	X	X	X	X
10	0	1	X	X

$B_4=1$

B_3B_2 \ B_1B_0	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	X	X	X	X
10	0	1	X	X

$A = B_0$

...Continued on next page

II - 18

For B

$B_4=0$

B_3B_2 \ B_1B_0	00	01	11	10
00	0	0	1	1
01	0	0	1	1
11	X	X	X	X
10	0	0	X	X

$B_4=1$

B_3B_2 \ B_1B_0	00	01	11	10
00	1	1	0	0
01	1	1	0	0
11	X	X	X	X
10	1	1	X	X

$$B = B_1\overline{B}_4 + \overline{B}_1B_4$$
$$= B_1 \oplus B_4$$

For C

$B_4=0$

B_3B_2 \ B_1B_0	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	X	X	X	X
10	0	0	X	X

$B_4=1$

B_3B_2 \ B_1B_0	00	01	11	10
00	0	0	1	1
01	1	1	0	0
11	X	X	X	X
10	0	0	X	X

$$C = \overline{B}_4B_2 + B_2\overline{B}_1 + B_4\overline{B}_2B_1$$

For D

$B_4=0$

B_3B_2 \ B_1B_0	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	X	X	X	X
10	1	1	X	X

$B_4=1$

B_3B_2 \ B_1B_0	00	01	11	10
00	1	1	1	1
01	1	1	0	0
11	X	X	X	X
10	0	0	X	X

$$D = \overline{B}_4B_3 + B_4\overline{B}_3\overline{B}_2 + B_4\overline{B}_3\overline{B}_1$$

For E

$B_4=0$

B_3B_2 \ B_1B_0	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	X	X	X	X
10	0	0	X	X

$B_4=1$

B_3B_2 \ B_1B_0	00	01	11	10
00	0	0	0	0
01	0	0	1	1
11	X	X	X	X
10	1	1	X	X

$$E = B_4B_3 + B_4B_2B_1$$

Fig. 4.91

Logic diagram

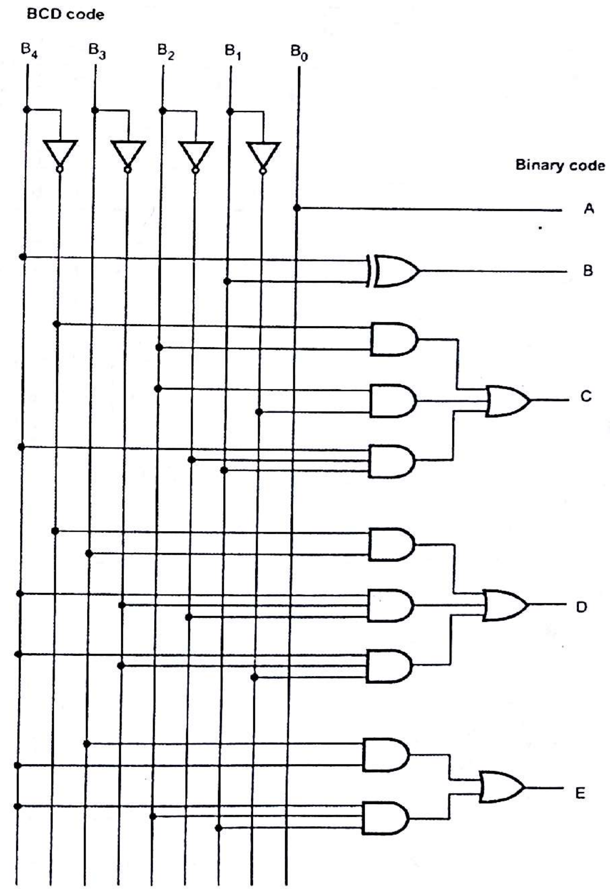


Fig. 4.92 BCD to binary code converter

4.5.3 BCD to Excess 3

Excess-3 code is a modified form of a BCD number. The Excess-3 code can be derived from the natural BCD code by adding 3 to each coded number. For example, decimal 12 can be represented in BCD as 0001 0010. Now adding 3 to each digit we get Excess-3 code as 0100 0101 (12 in decimal). With this information the truth table for BCD to Excess-3 code converter can be determined as shown in Table 4.31.

Decimal	B_3	B_2	B_1	B_0	E_3	E_2	E_1	E_0
0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	1	0	0
2	0	0	1	0	0	1	0	1
3	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	1	1
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	1	1	0	0

Table 4.31

K-map simplification

For E_3

B_3B_2 \ B_1B_0	00	01	11	10
00	0	0	0	0
01	0	1	1	1
11	X	X	X	X
10	1	1	X	X

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

For E_2

B_3B_2 \ B_1B_0	00	01	11	10
00	0	1	1	1
01	1	0	0	0
11	X	X	X	X
10	0	1	X	X

$\therefore E_2 = B_2\bar{B}_1\bar{B}_0 + \bar{B}_2(B_0 + B_1)$

For E_1

B_3B_2 \ B_1B_0	00	01	11	10
00	1	0	1	0
01	1	0	1	0
11	X	X	X	X
10	1	0	X	X

$E_1 = \bar{B}_1\bar{B}_0 + B_1B_0 = B_1 \odot B_0$

For E_0

B_3B_2 \ B_1B_0	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	X	X	X	X
10	1	0	X	X

$E_0 = \bar{B}_0$

Fig. 4.93

Logic diagram

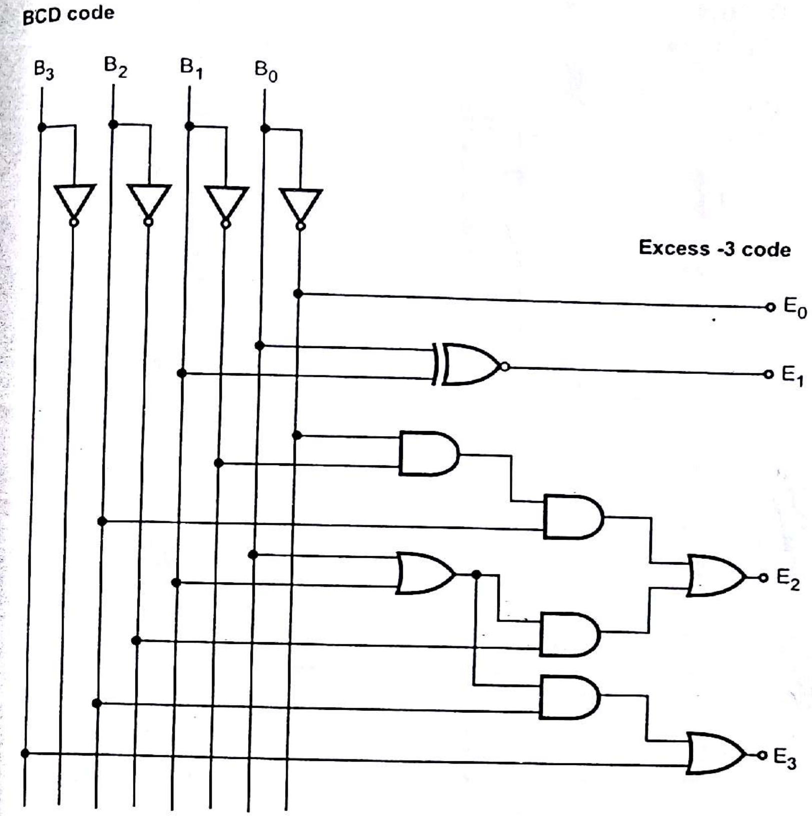


Fig. 4.94 BCD to Excess-3 code converter

4.5.4 Excess-3 to BCD Code Converter

The truth table for Excess-3 to BCD code converter is as shown in the Table 4.32.

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

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

Table 4.32 Truth table for Excess-3 to BCD converter

K-map simplification

For B_0

E_3E_2 \ E_1E_0	00	01	11	10
00	X	X	0	X
01	1	0	0	1
11	1	X	X	X
10	1	0	0	1

$B_0 = \overline{E}_0$

For B_1

E_3E_2 \ E_1E_0	00	01	11	10
00	X	X	0	X
01	0	1	0	1
11	0	X	X	X
10	0	1	0	1

$B_1 = \overline{E}_1E_0 + E_1\overline{E}_0$
$= E_1 \oplus E_0$

For B_2

E_3E_2 \ E_1E_0	00	01	11	10
00	X	X	0	X
01	0	0	1	0
11	0	X	X	X
10	1	1	0	1

$B_2 = \overline{E}_2\overline{E}_1 + E_2E_1E_0 + E_3E_1\overline{E}_0$

For B_3

E_3E_2 \ E_1E_0	00	01	11	10
00	X	X	0	X
01	0	0	0	0
11	1	X	X	X
10	0	0	1	0

$B_3 = E_3E_2 + E_3E_1E_0$

Fig. 4.95

Logic diagram

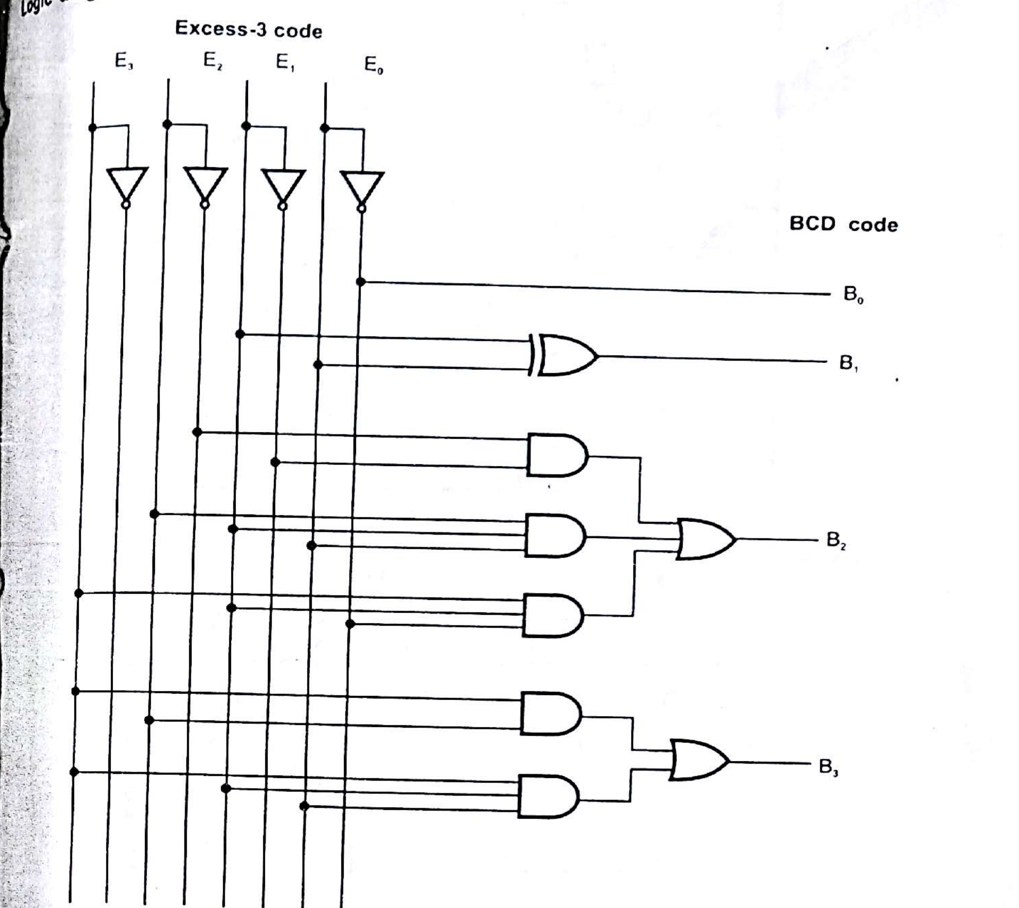


Fig. 4.96 Excess-3 to BCD code converter

4.5.5 Binary to Gray Code Converter

The Gray code is often used in digital systems because it has the advantage that only one bit in the numerical representation changes between successive numbers. Table 4.33 shows decimal and Binary codes and corresponding Gray code.

Decimal	Binary code				Gray code			
	D	C	B	A	G_3	G_2	G_1	G_0
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	1

Scanned by CamScanner

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

Table 4.33

K-map simplification

For G_0

DC \ BA	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	0	1	0	1
10	0	1	0	1

$G_0 = \overline{B}A + B\overline{A}$
$= B \oplus A$

For G_1

DC \ BA	00	01	11	10
00	0	0	1	1
01	1	1	0	0
11	1	1	0	0
10	0	0	1	1

$G_1 = C\overline{B} + \overline{C}B$
$= C \oplus B$

For G_2

DC \ BA	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	0	0	0	0
10	1	1	1	1

$G_2 = \overline{D}C + D\overline{C}$
$= D \oplus C$

For G_3

DC \ BA	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

$G_3 = D$

Fig. 4.97

Logic diagram

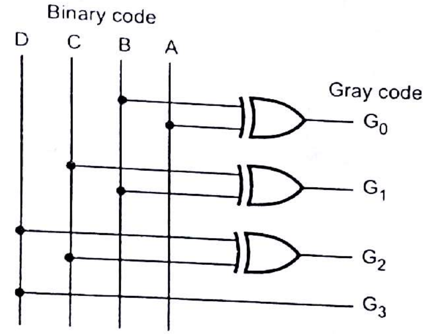


Fig. 4.98 Binary to gray code converter

16 Gray Code to Binary Code Converter

Table 4.34 shows the truth table for gray code to binary code converter.

Gray code				Binary code			
G_3	G_2	G_1	G_0	D	C	B	A
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	1	0	0	1	0
0	0	1	0	0	0	1	1
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	1
0	1	0	1	0	1	1	0
0	1	0	0	0	1	1	1
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	1
1	1	1	1	1	0	1	0
1	1	1	0	1	0	1	1
1	0	1	0	1	1	0	0
1	0	1	1	1	1	0	1
1	0	0	1	1	1	1	0
1	0	0	0	1	1	1	1

Table 4.34 Truth table for gray code to binary code converter

K-map simplification

For A

G_3G_2 \ G_1G_0	00	01	11	10
00	0	1	0	1
01	1	0	1	0
11	0	1	0	1
10	1	0	1	0

For B

G_3G_2 \ G_1G_0	00	01	11	10
00	0	0	1	1
01	1	1	0	0
11	0	0	1	1
10	1	1	0	0

Fig. 4.99

$$A = (\overline{G}_3G_2 + G_3\overline{G}_2)\,\overline{G}_1\overline{G}_0 + (\overline{G}_3\,\overline{G}_2 + G_3G_2)\,\overline{G}_1G_0 + (\overline{G}_3G_2 + G_3\,\overline{G}_2)\,G_1G_0 + (\overline{G}_3\overline{G}_2 + G_3G_2)\,G_1\,\overline{G}_0$$

$$= (G_3 \oplus G_2)\,\overline{G}_1\overline{G}_0 + (G_3 \odot G_2)\overline{G}_1G_0 + (G_3 \oplus G_2)\,G_1G_0 + (G_3 \odot G_2)\,G_1\overline{G}_0$$

$$= (G_3 \oplus G_2)\,(\overline{G}_1\overline{G}_0 + G_1G_0) + (G_3 \odot G_2)\,(\overline{G}_1G_0 + G_1\overline{G}_0)$$

$$= (G_3 \oplus G_2)\,(G_1 \odot G_0) + (G_3 \odot G_2)\,(G_1 \oplus G_0)$$

$$= (G_3 \oplus G_2)\,\overline{(G_1 \oplus G_0)} + \overline{(G_3 \oplus G_2)}\,(G_1 \oplus G_0)$$

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

$$B = (\overline{G}_3\overline{G}_2 + G_3G_2)G_1 + (\overline{G}_3G_2 + G_3\overline{G}_2)\overline{G}_1$$

$$= (G_3 \odot G_2)G_1 + (G_3 \oplus G_2)\overline{G}_1$$

$$= \overline{(G_3 \oplus G_2)}G_1 + (G_3 \oplus G_2)\overline{G}_1$$

$$= G_3 \oplus G_2 \oplus G_1$$

For C

G_3G_2 \ G_1G_0	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	0	0	0	0
10	1	1	1	1

For D

G_3G_2 \ G_1G_0	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

Fig. 4.100

$$C = \overline{G}_3G_2 + G_3\overline{G}_2 \qquad D = G_3$$

$$= G_3 \oplus G_2$$

Logic diagram

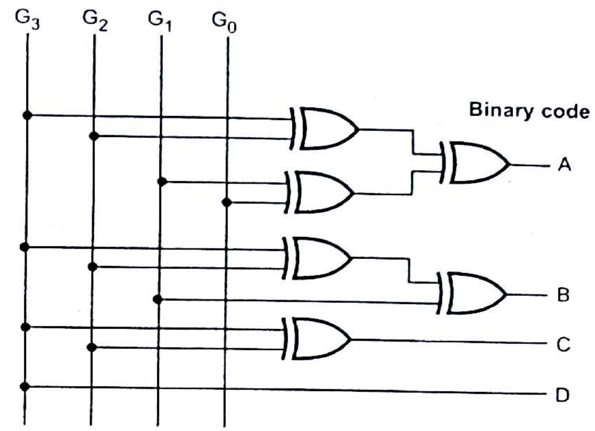


Fig. 4.101 Gray code to binary code converter

4.7 BCD to Gray Code Converter

Table 4.35 shows truth table for BCD to gray code converter.

BCD code				Gray code			
B_3	B_2	B_1	B_0	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
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1

Table 4.35 Truth table for BCD to gray code converter

K-map simplification

For G_0

$B_3B_2 \backslash B_1B_0$	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	X	X	X	X
10	0	1	X	X

$G_0 = \overline{B}_1 B_0 + B_1 \overline{B}_0$
$= B_1 \oplus B_0$

For G_1

$B_3B_2 \backslash B_1B_0$	00	01	11	10
00	0	0	1	1
01	1	1	0	0
11	x	x	x	x
10	0	0	x	x

$G_1 = B_2\overline{B}_1 + \overline{B}_2 B_1$
$= B_2 \oplus B_1$

For G_2

$B_3B_2 \backslash B_1B_0$	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	X	X	X	X
10	1	1	X	X

$G_2 = B_2 + B_3$

For G_3

$B_3B_2 \backslash B_1B_0$	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	X	X	X	X
10	1	1	X	X

$G_3 = B_3$

Fig. 4.102

Logic diagram

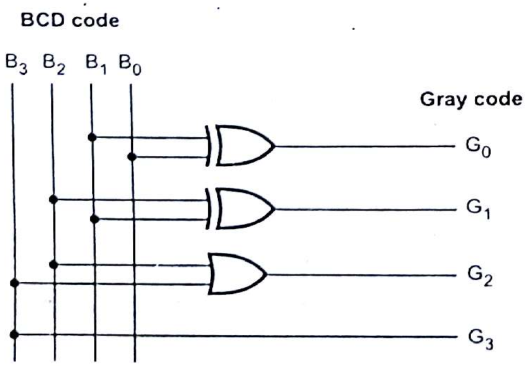


Fig. 4.103 BCD to gray code converter

6 Hazards

The unwanted switching transients that may appear at the output of a circuit are ...d Hazards. The hazards cause the circuit to malfunction. The main cause of ...ards is the different propagation delays at different paths. Hazards occur in the ...binational circuits, where they may cause a temporary false output value. When ...h combinational circuits are used in the asynchronous sequential circuits, they may ...lt in a transition to a wrong stable state.

6.1 Hazards in Combinational Circuits

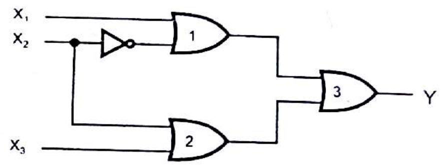


Fig. 4.104

Fig. 4.104 shows combinational circuit with hazards. Assume that, initially, inputs ...nd $x_3 = 0$ and $x_2 = 1$. This causes the output of gate 1 to be 0, that of gate 2 to be ...nd the output of the circuit to be equal to 0. Now consider change in x_2 from 1 to ...he output of gate 1 changes to 1 and that of gate 2 changes to 0, leaving the ...ut at 1. However, the output momentarily goes to 0 if the propagation delay ...ugh the inverter is taken into consideration. The delay in the inverter causes the ...put of gate 2 to change to 0 before the output of gate 1 changes to 1. In this ...ation, both inputs of gate 3 are momentarily equal to 0, causing the output to go ...0 for the short time equal to the propagation delay of the inverter. This is ...strated in the Fig. 4.105.

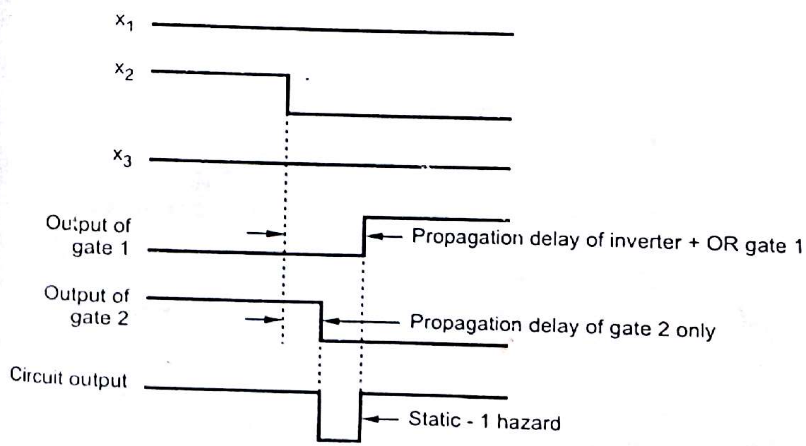


Fig. 4.105 Waveforms showing static 1 hazard