

---

## Module -1

# PRINCIPLES OF COMBINATIONAL LOGIC

### Syllabus

Definition of combinational logic, canonical forms, Generation of switching equations from truth tables, Karnaugh maps-3,4,5 variables, incompletely specified functions (Don't care terms), Simplifying Max term equations, Quine-McClusky techniques – 3 & 4 variables.

(Text 1 - Chapter 3) [RBT Levels: L1, L2, L3]

### Introduction:

The word digital implies that, information in the system is represented by values that are limited in number. The values hence used to represent the information are called quantized or discrete values. These values are processed by circuit components that maintain the limited set of values. A digital system performs more reliably and efficiently if it handles only two values ("0" and "1") that is only two states and such a digital system will be called a **two state system or a binary system**.

Since, the invention of digital electronics/ integrated circuits a few decades ago, manufacturing of electronic systems has taken rapid strides in improving speed, size, and cost. For today's digital integrated chips, switching time is on the order of submicron's, **transistor count is in the order of millions, speed, size and cost** leads to the very pervasive introduction of **integrated circuit (IC)** chips to many aspects of modern engineering and scientific endeavours including computations, telecommunication, aeronautics, genetic engineering and manufacturing, and so on.

**Needless to say, it is the very large scale integration (VLSI) technology that has made this evolution a reality. The digital electronics together with VLSI technology has brought the power of mainframe computer to the laptop.**

In this chapter, Logic circuits, where the output at a given time is purely a function of the inputs at that time are referred to as combinational circuits. Karnaugh Maps are effective in simplifying expression with

a small number of variables, as evident by that complexities involved in five variable maps. The Quine-McCluskey method provides one such algorithmic approach.

## Definition of Combinational Logic:

**Combinational logic** deals with the techniques of combining the basic gates into circuit that performs some desired functions.

**Logic circuit without feedback from output to the input, constructed from a functionally complete gate set, are said to be Combinational.** Logic circuits that contain no memory are combinational; that contain memory including Flip-flops are called as **Sequential**.

**In other words, Combinational Logic Circuit** the output is dependant at all times on the combination of its inputs. So if one of its inputs condition changes state, from 0-1 or 1-0, so too will the resulting output as by default combinational logic circuits have “no memory”, “timing” or “feedback loops” within their design.

**Examples: Adders, subtractors, Decoders, encoders, display drivers and multiplexers etc.**

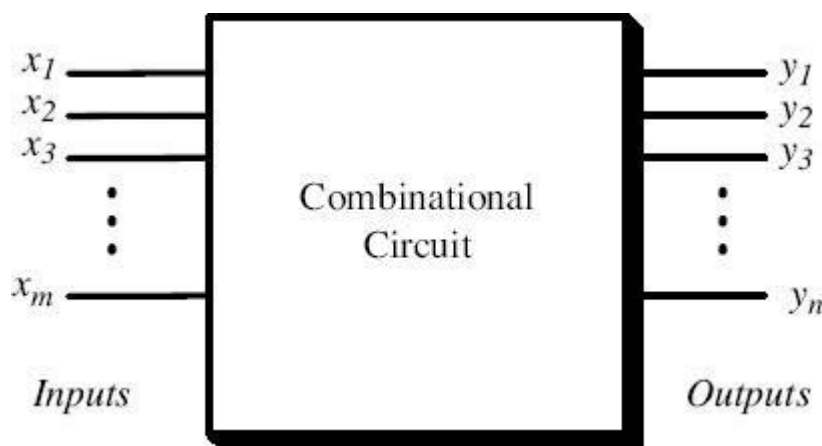


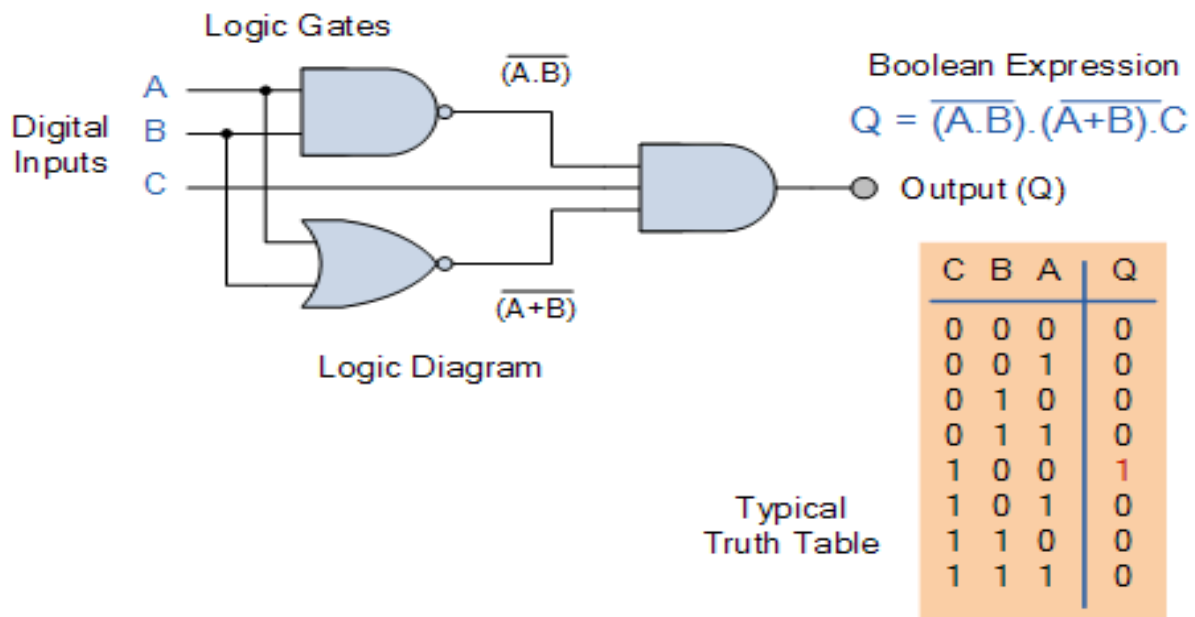
Figure 1: Combinational Logic Model

The three main ways of specifying the function of a combinational logic circuit are:

- ✓ **Boolean Algebra** – This forms the algebraic expression showing the operation of the logic circuit for each input variable either True or False that results in a logic “1” output.
- ✓ **Truth Table** – A truth table defines the function of a logic gate by providing a concise list that shows all the output states in tabular form for each possible combination of input variable that the gate could encounter.

- ✓ **Logic Diagram** – This is a graphical representation of a logic circuit that shows the wiring and connections of each individual logic gate, represented by a specific graphical symbol, that implements the logic circuit and all three of these logic circuit representations are shown below.

the following example shows the combinational circuit with necessary truth table, Boolean expression and logical diagram.



### Problem Statement to Truth Tables:

Before any combinational logic system can be designed it must be defined. Proper statement of a problem is the most important part of any digital design task. Once correctly and clearly stated, any problem can be converted into necessary logic for implementation. Figure 2 shows the sequence flow of design tasks in the general way.

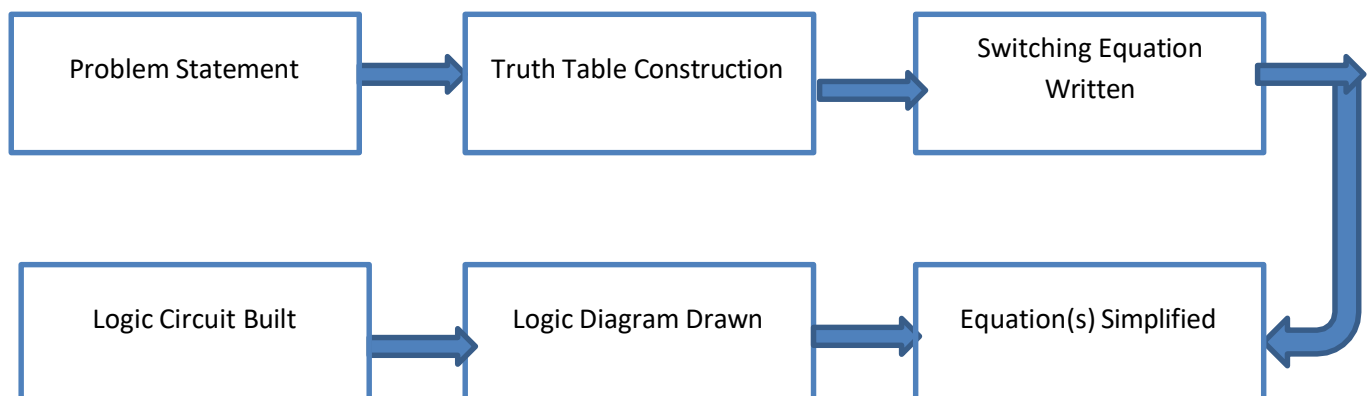
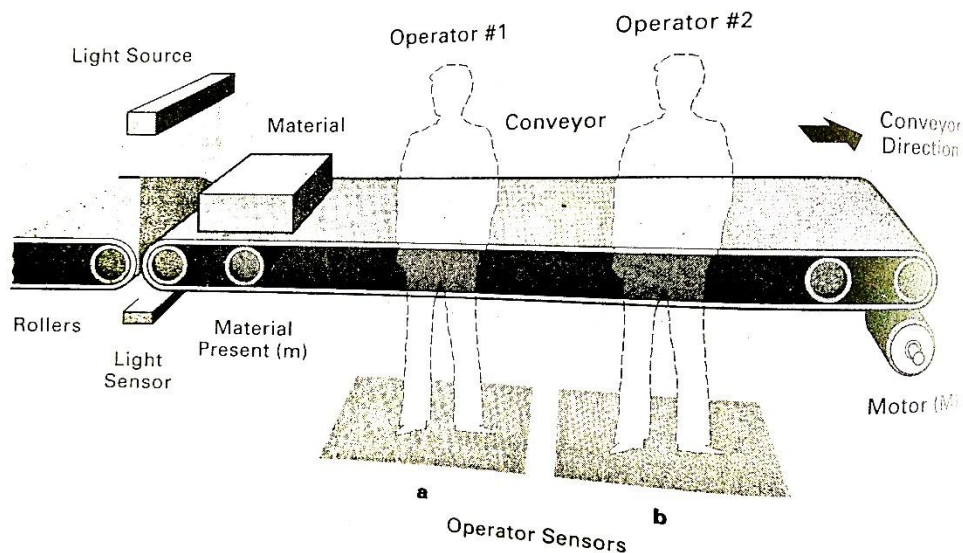


Figure 2: General Logic Design sequence

## Example 1:

An electric motor powering a conveyor used to move material is to be turned on when one of the two operators is in position, if material is present to be moved and if the protection interlock switch is not open. Input and Output variables are expressed in binary; that is, if operator 1 is in the position then the associated variable is a logical 1. The motor running if its variable is a 1, and the motor is off if the variable is 0. The following figure shows the conveyor system.



## Solution:

Variables **a** and **b** signify to operators are in position  
**s** means interlock switches closed  
**m** means material is present  
**a, b, s and m** are input variables;  
**M** is an output variable

$$M = a'bms + ab'ms + abms$$

| a | b | m | s | M |
|---|---|---|---|---|
| 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 | 1 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

**Example 2:**

Design a combinational logic truth table so that an output is generated indicating when a majority of 4 inputs is true.

**Solution:**

The circuit has 4 inputs and single output. Let I1, I2, I3 and I4 represent input variables and O1 be the output variable.

| I1 | I2 | I3 | I4 | O1 |
|----|----|----|----|----|
| 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  | 1  |
| 1  | 0  | 0  | 0  | 0  |
| 1  | 0  | 0  | 1  | 0  |
| 1  | 0  | 1  | 0  | 0  |
| 1  | 0  | 1  | 1  | 1  |
| 1  | 1  | 0  | 0  | 0  |
| 1  | 1  | 0  | 1  | 1  |
| 1  | 1  | 1  | 0  | 1  |
| 1  | 1  | 1  | 1  | 1  |

In summary compare the process of converting verbal problem statement into truth table involves the following steps

- ✓ Determine the input variables and output variables that are involved.
- ✓ Assign mnemonic or letter or number symbols to each variable.
- ✓ Determine the size of truth table; how many input combinations exist: that is

$$2^x = Y$$

Where x = to number of input variables and Y = number of combinations

- ✓ Construct a truth table containing all of the input variable combinations.
- ✓ By careful reading the problem statement determine the combination of inputs that cause a given output to be true.

Before continue in with canonical forms some definitions are necessary

**Literal:** A literal is a Boolean variable or its complement. For instance, let x be binary variable, then both X and X' would be literals.

**Product term:** a product term is a literal or the logical AND of multiple literals. For instance, let X, Y and Z be binary variables. Then representative product turn could be XY, XYZ', XY'Z etc.

**Sum term:** a Sum term is a literal or the logical OR of multiple literals. For instance, let X, Y and Z be binary variables. Then representative product turn could be X+Y, X+Y+Z', X+Y'+Z etc.

**Sum of Products:** A sum of Products(SOP) is the logical OR of multiple product terms. Each product term is the AND of binary literals. For example, XYZ+XY'+XYZ' is a SOP expression.

**Product of Sums:** A product of sum(POS) is the logical AND of multiple OR terms. Each Sum term is the OR of binary literals. For example, (X+Y+Z) (X+Y') (X+Y+Z') is a POS expression.

**Minterm:** A Minterm is a special case of product (AND) term. A minterm is a product term that contains all of the input variables (each literal no more than once) that makeup a Boolean expression.

**Maxterm:** A Minterm is a special case of sum (OR) term. A maxterm is a sum term that contains all of the input variables (each literal no more than once) that makeup a Boolean expression.

**Canonical Sum of Products:** A Canonical sum of products is a complex set of **minterm** that defines where and **output variable is logical 1**. Each minterm corresponds to the row in the truth table where the output function is 1; That is, a SOP for the output M we in the previous example is

$$M = a'bms + ab'ms + abms$$

**Canonical Product of Sums:** A Canonical product of sum is a complex set of **maxterm** that defines where and **output variable is logical 0**. Each maxterm corresponds to the row in the truth table where the output function is 0.

$$O1 = (a' + b' + c' + d) (a + b + c' + d)$$

The following table 1 shows minterm and maxterm designations for 3 variables

Table 1:

| Input Variables |   |   | Minterm    |              | Maxterm        |              |
|-----------------|---|---|------------|--------------|----------------|--------------|
| a               | b | c | Term       | Designations | Term           | Designations |
| 0               | 0 | 0 | $a' b' c'$ | m0           | $a + b + c$    | M0           |
| 0               | 0 | 1 | $a' b' c$  | m1           | $a + b + c'$   | M1           |
| 0               | 1 | 0 | $a' b c'$  | m2           | $a + b' + c$   | M2           |
| 0               | 1 | 1 | $a' b c$   | m3           | $a + b' + c'$  | M3           |
| 1               | 0 | 0 | $a b' c'$  | m4           | $a' + b + c$   | M4           |
| 1               | 0 | 1 | $a b' c$   | m5           | $a' + b + c'$  | M5           |
| 1               | 1 | 0 | $a b c'$   | m6           | $a' + b' + c$  | M6           |
| 1               | 1 | 1 | $a b c$    | m7           | $a' + b' + c'$ | M7           |

**Note:** Table 1 shows the complement nature of minterms and maxterms.

Note that and input variable is complemented when it has a **value of 0**, if we are writing in **minterms**.

The input variables are complemented when they have a **value of 1**, if we are writing for **Maxterms**.

Minterm represents output variable 1s and max term represents the output variable 0s.

**Lower case m is used to denote minterms and upper case M is used to denote maxterms.**

## Canonical Forms:

Canonical is a word to describe a condition of a switching equation in normal use the word means conforming to general rule. The rule for searching logic is that each term used in switching equation must contain all of the available input variables.

Two formats generally exist for expressing switching equation in canonical form: sum of minterms and product of maxterms. Simplification rationally was to eliminate redundancy and lower cost of final logic circuits. **Canonical expressions of not simplified orphan canonical expression is the opposite of simplification that is it contains redundancies.**

To place SOP equation into canonical form using Boolean algebra, we do the following:

1. Identify missing variable(s) in each AND term.
2. AND the missing term(s) and its complement with original AND term ex:  $x y (z' + z)$

Because  $(z+z'=1)$  the original AND term value is not changed.

- Expand the term by application of the property of distribution,  $x y z' + x y z$

To place POS equation into canonical form using Boolean algebra, we do the following:

- Identify missing variable(s) in each OR term.
- OR the missing term(s) and its complement with original OR term ex:  $x + y + z.z'$

Because  $(z.z'=0)$  the original OR term value is not changed.

- Expand the term by application of the property of distribution,  $(x + y + z) (x + y + z')$

### Example 3:

Place the following equations into proper canonical form:

a.  $P = f(a, b, c) = ab' + ac' + bc$

b.  $G = f(w, x, y, z) = w'x + yz'$

c.  $T = f(a, b, c) = (a + b')(b' + c)$

d.  $D = J(A, B, C, D) = (A + B' + c)(A' + D)$

Solution: a.  $P = f(a, b, c) = ab' + ac' + bc$   
 $= ab'(c + c') + ac'(b + b') + bc(a + a')$   
 $= ab'c + ab'c' + abc' + ab'c' + abc + a'bc$

Now that second and fourth terms are same, so only one is needed. so final equation becomes

$$P = ab'c + ab'c' + abc' + abc + a'bc$$

Solution: c.  $T = f(a, b, c) = (a + b')(b' + c)$

$$= (a + b' + cc')(b' + c + aa')$$

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

Now that first and third terms are same, so only one is needed. so final equation becomes

$$T = (a + b' + c)(a + b' + c')(a' + b' + c)$$

Solve the remaining examples too.

## Generating Switching Equations from truth table:

Switching equations can be written more conveniently by using the minterm or maxterm numerical designation. Instead of writing the variable names or its complement, in fact m and M designations can be dropped and decimal equivalent value for the term can be written directly.

For example:  $P = (ab'c + ab'c' + abc' + abc + a'bc)$

$$P = \sum (5,4,6,7,3)$$

$$P = f(a, b, c) = \sum (5,4,6,7,3)$$

The Sign  $\sum$  indicates summations and stands for the sum of product canonical forms.

The Sign  $\prod$  indicates summations and stands for the product of sum canonical forms.

### Example 4:

Express the following SOP and POS equations in a minterm and maxterm list (shorthand decimal notation) form:

a.  $H = f(A, B, C) = A'BC + A'B'C + ABC$

b.  $G = f(w, x, y, z) = wxyz' + wx'y'z' + w'xyz' + w'x'yz'$

c.  $T = f(a, b, c) = (a + b' + c)(a + b' + c')(a' + b' + c)$

**Solutions:** a.  $H = f(A, B, C) = \sum (1, 3, 7)$

b.  $G = f(w, x, y, z) = \sum (2, 6, 10, 14)$

c.  $T = f(a, b, c) = \prod (2, 3, 6)$

Its not necessary to write the switching equations in a variable format and then converting them into decimal format.

**Home work:** Try to write the SOP and POS equation for both A and A'

| x | y | z | A | A' |
|---|---|---|---|----|
| 0 | 0 | 0 | 0 | 1  |
| 0 | 0 | 1 | 0 | 1  |
| 0 | 1 | 0 | 1 | 0  |
| 0 | 1 | 1 | 0 | 1  |
| 1 | 0 | 0 | 1 | 0  |
| 1 | 0 | 1 | 1 | 0  |
| 1 | 1 | 0 | 0 | 1  |
| 1 | 1 | 1 | 0 | 1  |

## K-Map (Karnaugh Maps)

In many digital circuits and practical problems, we need to find expression with minimum variables. We can minimize Boolean expressions of **3, 4 and 5 variables** very easily using K-map without using any Boolean algebra theorems. K-map can take two forms Sum of Product (SOP) and Product of Sum (POS) according to the need of problem. K-map is table like representation but it gives more information than TRUTH TABLE. We fill grid of K-map with 0's and 1's then solves it by making groups.

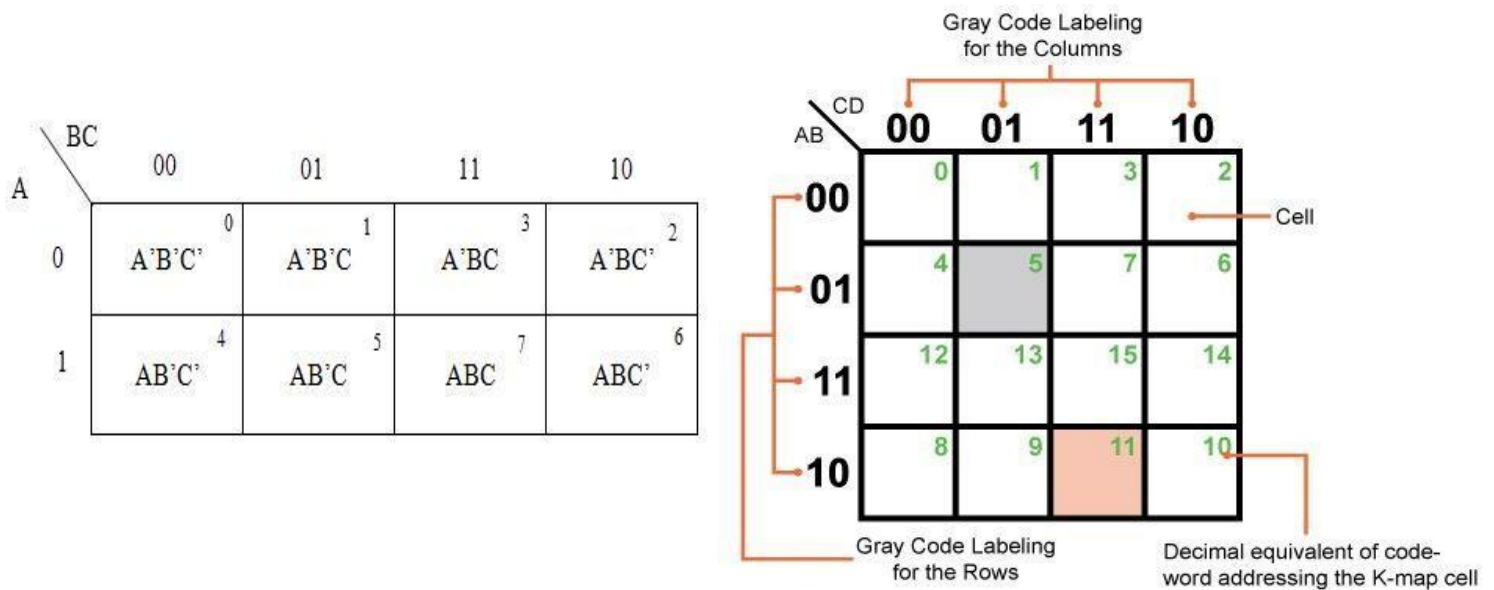


Figure 3: 3 and 4 variable K-Maps

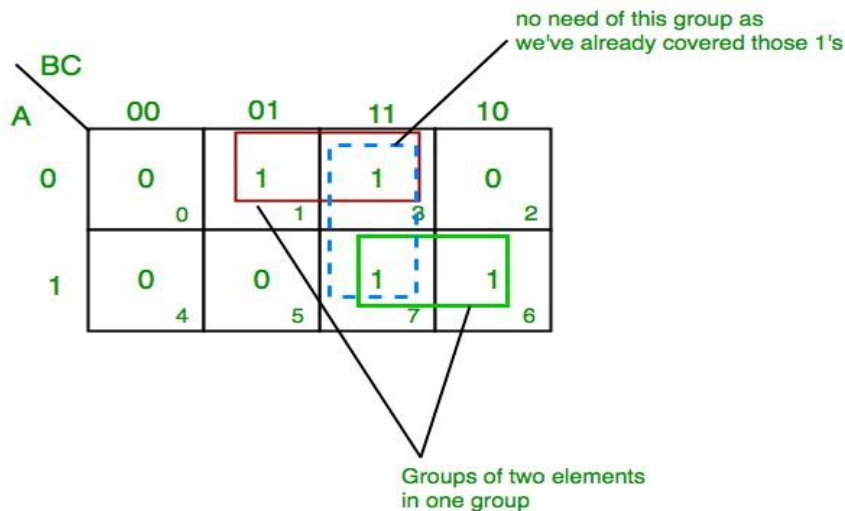
### Steps to solve expression using K-map:

1. Select K-map according to the number of variables.
2. Identify minterms or maxterms as given in problem.
3. For SOP put 1's in blocks of K-map respective to the minterms (0's elsewhere).
4. For POS put 0's in blocks of K-map respective to the maxterms (1's elsewhere).
5. Make rectangular groups containing total terms in power of two like 2,4,8, 16 etc. (except 1) and try to cover as many elements as you can in one group.
6. From the groups made in step 5 find the product terms and sum them up for SOP form.

## Example 5:

SOP FORM1. K-map of 3 variables:

$$Z = f(a, b, c) = \sum (1, 3, 6, 7)$$



From **red** group we get product term -  $A'C$

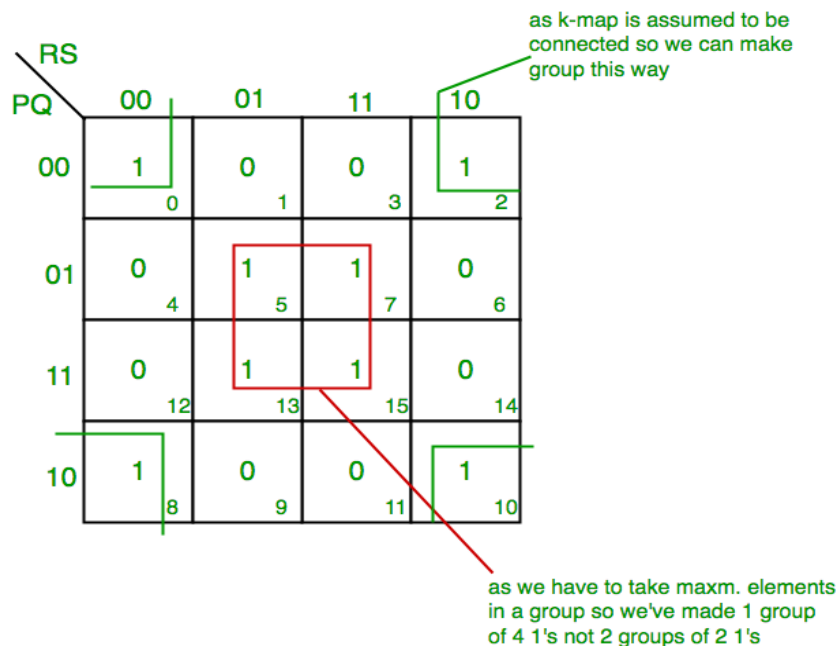
From **green** group we get product term -  $AB$

Summing these product terms, we get –

**Final expression  $Z = A'C + AB$**

2. K-map for 4 variables:

$$Y = F(P, Q, R, S) = \sum (0, 2, 5, 7, 8, 10, 13, 15)$$



From **red** group we get product term -  $QS$

From **green** group we get product term -  $Q'S'$

Summing these product terms, we get- **Final expression  $(QS + Q'S')$**

POS FORM1. K-map of 3 variables-

$$Y = F(A, B, C) = \pi(0, 3, 6, 7)$$

2 elements in one group

|     |        |        |        |        |
|-----|--------|--------|--------|--------|
| BC  | 00     | 01     | 11     | 10     |
| A 0 | 0<br>0 | 1<br>1 | 0<br>3 | 1<br>2 |
| A 1 | 1<br>4 | 1<br>5 | 0<br>7 | 0<br>6 |

From **red** group we find terms  $A B C'$

Taking complement of these two  $A' B' C$

Now **sum** up them  $(A' + B' + C)$

From **green** group we find terms  $B C$

Taking complement of these two terms  $B' C'$

Now sum up them  $(B' + C')$

From **brown** group we find terms  $A' B' C'$ , Taking complement of these  $A B C$ ,

Now **sum** up them  $(A + B + C)$

We will take product of these three terms: **Final expression**  $(A' + B' + C) (B' + C') (A + B + C)$

2. K-map of 4 variables-

$$Y = F(A, B, C, D) = \pi(3, 5, 7, 8, 10, 11, 12, 13)$$

From Red group:  $(A + C' + D')$

From Green group:  $(B' + C + D')$

From Blue Group:  $(A' + C + D)$

From Brown group:  $(A' + B + C')$

So final expression is

$$Y = (A + C' + D') (B' + C + D') (A' + C + D) (A' + B + C')$$

|    |         |         |         |         |
|----|---------|---------|---------|---------|
| BC | 00      | 01      | 11      | 10      |
| 00 | 1<br>0  | 1<br>1  | 0<br>3  | 1<br>2  |
| 01 | 1<br>4  | 0<br>5  | 0<br>7  | 1<br>6  |
| 11 | 0<br>12 | 0<br>13 | 1<br>15 | 1<br>14 |
| 10 | 1<br>8  | 1<br>9  | 0<br>11 | 0<br>10 |

Let solve some example, so that you can learn the technique very easily.

**Example 6:**

**Design a full adder by obtaining the simplified expressions for the sum and carry outputs in POS form.**

**Step 1:**

Number of input variables = 3

Number of output variables = 2

| Inputs |   |                | Decimal<br>Equivalent | Outputs |                |
|--------|---|----------------|-----------------------|---------|----------------|
| A      | B | C <sub>i</sub> |                       | S       | C <sub>o</sub> |
| 0      | 0 | 0              | 0                     | 0       | 0              |
| 0      | 0 | 1              | 1                     | 1       | 0              |
| 0      | 1 | 0              | 2                     | 1       | 0              |
| 0      | 1 | 1              | 3                     | 0       | 1              |
| 1      | 0 | 0              | 4                     | 1       | 0              |
| 1      | 0 | 1              | 5                     | 0       | 1              |
| 1      | 1 | 0              | 6                     | 0       | 1              |
| 1      | 1 | 1              | 7                     | 1       | 1              |

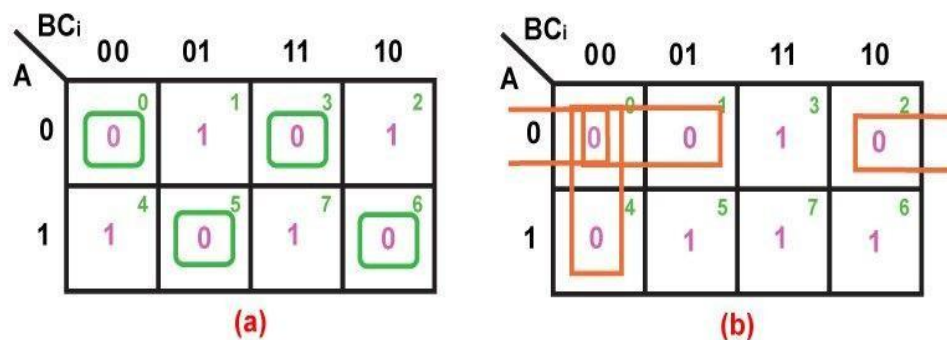
Maxterm expansion for S =  $\prod M(0,3,5,6)$

Maxterm expansion for Co =  $\prod M(0,1,2,4)$

**Steps 2 and 3:**

Number of K-maps required = 2, Each K-map should have 8 cells in it.

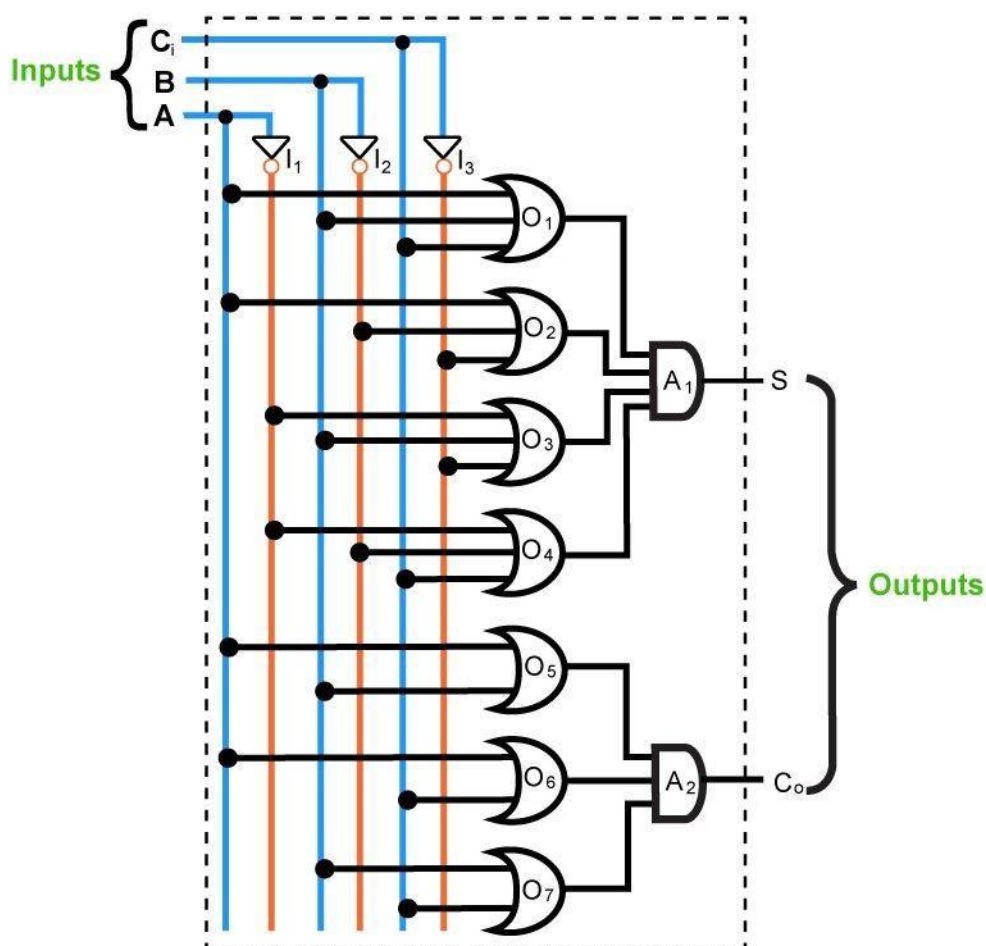
Thus we get:



Groups formed

| Groups  | Sum Output, S                                                                                                                            | Carry Output, C <sub>o</sub> |
|---------|------------------------------------------------------------------------------------------------------------------------------------------|------------------------------|
| Group 1 | $A + B + C_i$                                                                                                                            | $A + B$                      |
| Group 2 | $A + \bar{B} + \bar{C}_i$                                                                                                                | $A + C_i$                    |
| Group 3 | $\bar{A} + B + \bar{C}_i$                                                                                                                | $B + C_i$                    |
| Group 4 | $\bar{A} + \bar{B} + C_i$                                                                                                                |                              |
|         | $S = (A + B + C_i) (A + \bar{B} + \bar{C}_i) (\bar{A} + B + \bar{C}_i) (\bar{A} + \bar{B} + C_i)$<br>$C_o = (A + B) (A + C_i) (B + C_i)$ |                              |

The digital system designed to realize the full adder in terms of sum and carry outputs (in POS form) is shown by Figure



**Example 7:**

*Simplify following the expressions using K-map*

1.  $Y = f(a, b, c) = \sum (0, 1, 4, 5)$
2.  $G = f(x, y, z) = \sum (0, 2, 3, 7)$
3.  $D = f(x, y, z) = \sum (0, 2, 4, 6)$
4.  $Q = f(a, b, c) = \sum (1, 2, 3, 6, 7)$
5.  $J = f(x, y, z) = \sum (0, 2, 3, 4, 5, 7)$

**Solutions:**

1.  $Y = f(a, b, c) = \sum (0, 1, 4, 5)$

|   |   |    |    |    |    |
|---|---|----|----|----|----|
|   |   | bc |    |    |    |
|   |   | 00 | 01 | 11 | 10 |
| a | 0 | 0  | 1  | 3  | 2  |
|   | 0 | 1  | 1  | 0  | 0  |
| 1 | 4 | 1  | 5  | 7  | 6  |
|   | 1 | 1  | 1  | 0  | 0  |

The above group shows that minterm group possible for the given expression.

From the above shown group the prime implicant is  $b'$

$$Y = b'$$

You can also have made group from cells {0, 4} and {1, 5} but always look for the maximum elements as you can group in terms of integral power of 2. So here we can form 4 cells grouping so it simplifies the given expression as above.

**Prime Implicant:**

Any single minterm or permitted group of minterms is called an **implicant** of an output function. A **Prime implicant** is a group of minterms that cannot combined with any other minterms or groups.

$$2. G = f(x, y, z) = \sum(0, 2, 3, 7)$$

|   | YZ | 00 | 01 | 11 | 10 |
|---|----|----|----|----|----|
| X | 0  | 1  | 0  | 1  | 1  |
| 1 | 4  | 0  | 0  | 1  | 0  |

The above groups show that minterm group possible for the given expression.

From the above shown groups the Implicants are

From orange group -  $YZ$

From blue group -  $x'y$

From red group -  $x'z'$

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

You can also have made group from cells {2, 3} but all the elements of blue group are already grouped by other groups so **that group is not essential** you can again simplify the **G equations** as follows:

$$G = yz + x'z'$$

### Essential Prime Implicants:

***An essential prime implicant is a prime implicant in which one or more minterms are unique; that is, it contains at least one minterm not contained in any other prime implicant.***

Procedure for finding the prime Implicants:

1. Find the prime implicants by finding all the permitted (integral power of 2) maximum sized groups of minterms.
2. Find essential prime implicants by identifying those prime implicants that contain at least one minterm not found in any other prime implicant.

3.  $D = f(x, y, z) = \sum(0, 2, 4, 6)$

|   |    |    |    |    |    |
|---|----|----|----|----|----|
|   | yz | 00 | 01 | 11 | 10 |
| x | 0  | 1  | 3  | 2  |    |
| 0 |    | 1  | 0  | 0  | 1  |
| 1 |    | 1  | 0  | 0  | 1  |

The above group shows that minterm group possible for the given expression.

From red group obtained minterm is –  $z'$

$$D = z'$$

4.  $Q = f(a, b, c) = \sum(1, 2, 3, 6, 7)$

|   |    |    |    |    |    |
|---|----|----|----|----|----|
|   | bc | 00 | 01 | 11 | 10 |
| a | 0  | 1  | 3  | 2  |    |
| 0 |    | 0  | 1  | 1  | 1  |
| 1 |    | 0  | 0  | 1  | 1  |

The above groups show that minterm group possible for the given expression.

From the above shown groups the Implicants are

From orange group -  $b$

From blue group –  $a'c$

$$Q = b + a'c$$

$$5. J = f(x, y, z) = \sum (0, 2, 3, 4, 5, 7)$$

|   |    |    |    |    |    |
|---|----|----|----|----|----|
|   | yz | 00 | 01 | 11 | 10 |
| x | 0  | 0  | 1  | 3  | 2  |
| 0 |    | 1  | 0  | 1  | 1  |
| 1 |    | 4  | 5  | 7  | 6  |
|   |    | 1  | 1  | 1  | 0  |

The above groups show that minterm group possible for the given expression.

From the above shown groups the Implicants are

From {0, 2} group –  $x'z'$

From {0, 4} group –  $y'z'$

From {2, 3} group –  $x'y$

From {3, 7} group –  $yz$

From {4, 5} group –  $xy'$

From {5, 7} group –  $xz$

Note that none of the prime implicants is essential. The final simplified equation is not unique; that is, more than one equally simplified results is possible.

The product term in the simplified equation must cover all the minterms. By cover, we mean that all the minterms in the original equation must be contained by a prime implicant or essential prime implicant.

To cover {0} minterm we can pick {0, 2} or {0, 4}, similarly by doing for all the minterms; we have two equally simplified equations are

$$J = x' z' + y z + x y'$$

Or

$$J = y' z' + x' y + x z$$

**Example 8:****Simplify the following four variable equations using K-Map**

a.  $K = f(a, b, c, d) = \sum (0, 1, 4, 5, 9, 11, 13, 15)$

b.  $L = f(a, b, c, d) = \sum (0, 2, 5, 7, 8, 10, 13, 15)$

c.  $P = f(r, s, t, u) = \sum (1, 3, 4, 6, 9, 11, 12, 14)$

d.  $D = f(w, x, y, z) = \sum (5, 7, 8, 9, 13)$

e.  $P = f(a, b, c, d) = \sum (0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14)$

f.  $S = f(a, b, c, d) = \sum (1, 3, 4, 5, 7, 8, 9, 11, 15)$

g.  $U = f(w, x, y, z) = \sum (1, 5, 7, 8, 9, 10, 11, 13, 15)$

**Solutions:**

a.  $K = f(a, b, c, d) = \sum (0, 1, 4, 5, 9, 11, 13, 15)$

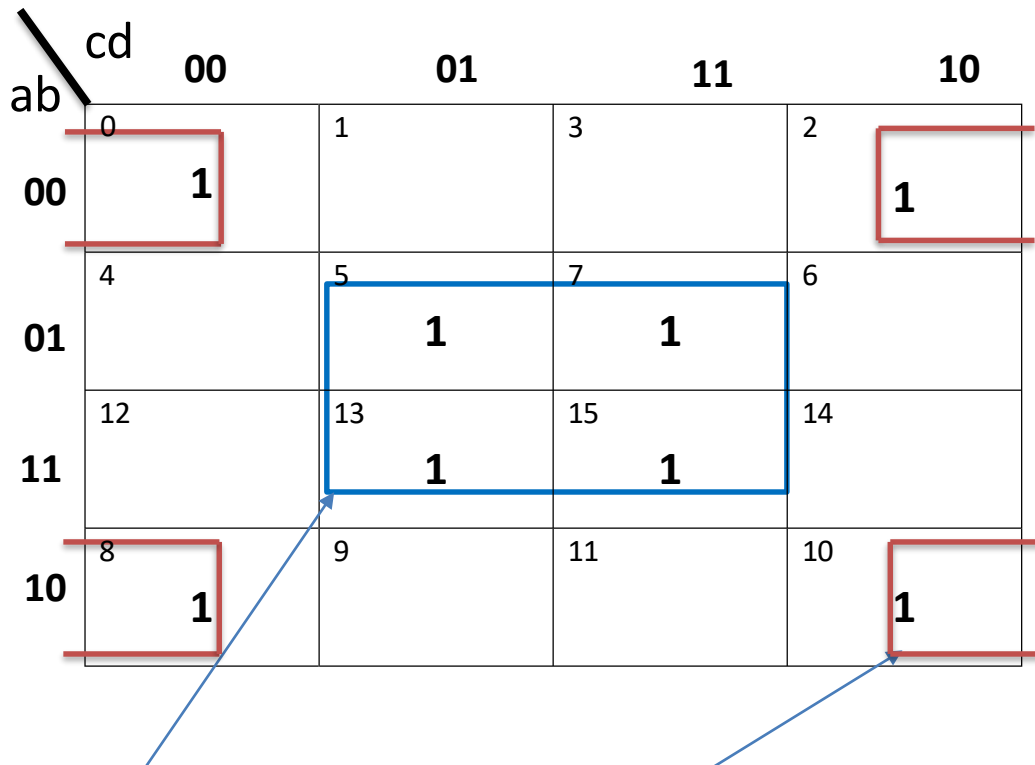
| cd \ ab |    | 00 |   | 01 |   | 11 |   | 10 |   |
|---------|----|----|---|----|---|----|---|----|---|
|         |    | 0  | 1 | 2  | 3 | 4  | 5 | 6  | 7 |
| 00      | 0  | 1  |   | 1  |   |    |   |    |   |
|         | 4  | 1  |   | 1  |   |    |   |    |   |
| 01      | 8  |    |   |    |   | 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

EPI {0, 1, 4, 5} =  $a'c'$ PI {1, 5, 9, 13} =  $c'd$ EPI {9, 11, 13, 15} =  $ad$ 

Final expression is

$$K = a'c' + ad$$

b.  $L = f(a, b, c, d) = \sum (0, 2, 5, 7, 8, 10, 13, 15)$



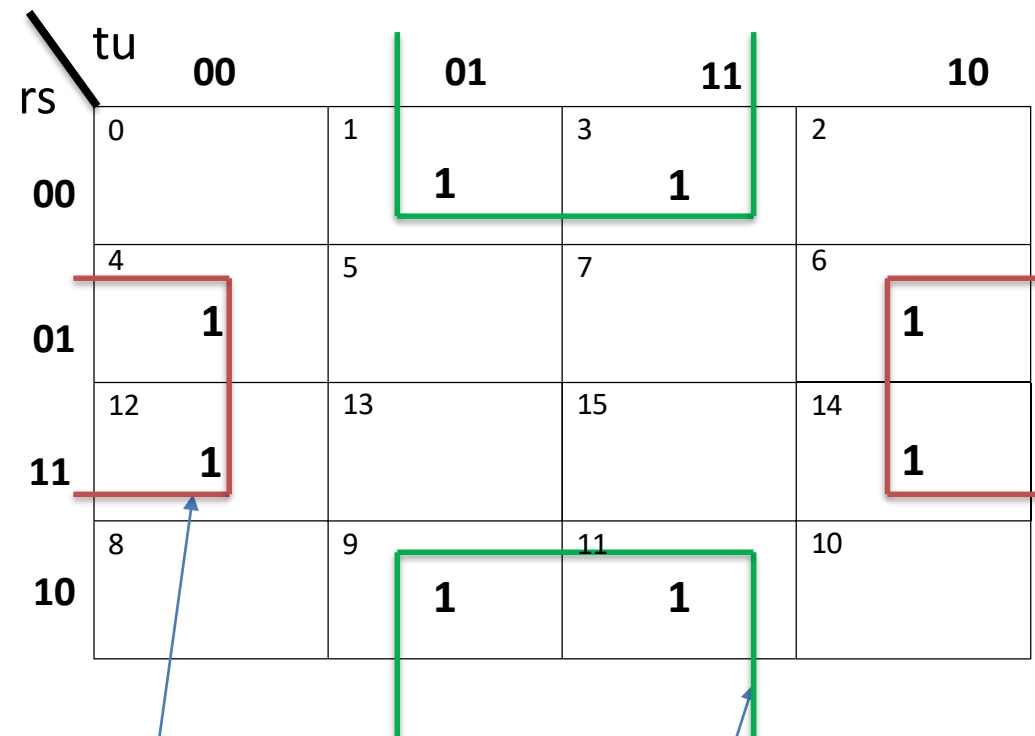
EPI {5, 7, 13, 15} =  $bd$

EPI {0, 2, 8, 10} =  $b'd'$

Final expression is

$$L = bd + b'd' = (b \oplus d)' \text{ (Ex-NOR of } L \text{)}$$

c.  $P = f(r, s, t, u) = \sum (1, 3, 4, 6, 9, 11, 12, 14)$



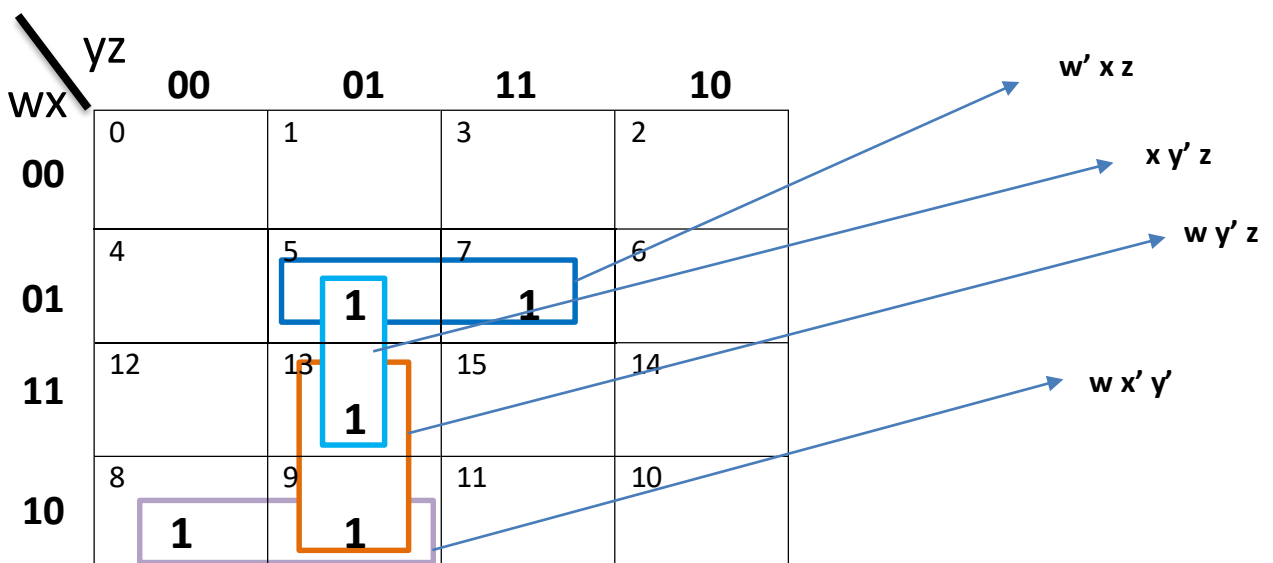
EPI {4, 6, 12, 14} =  $su'$

EPI {1, 3, 9, 11} =  $s'u$

Final expression is

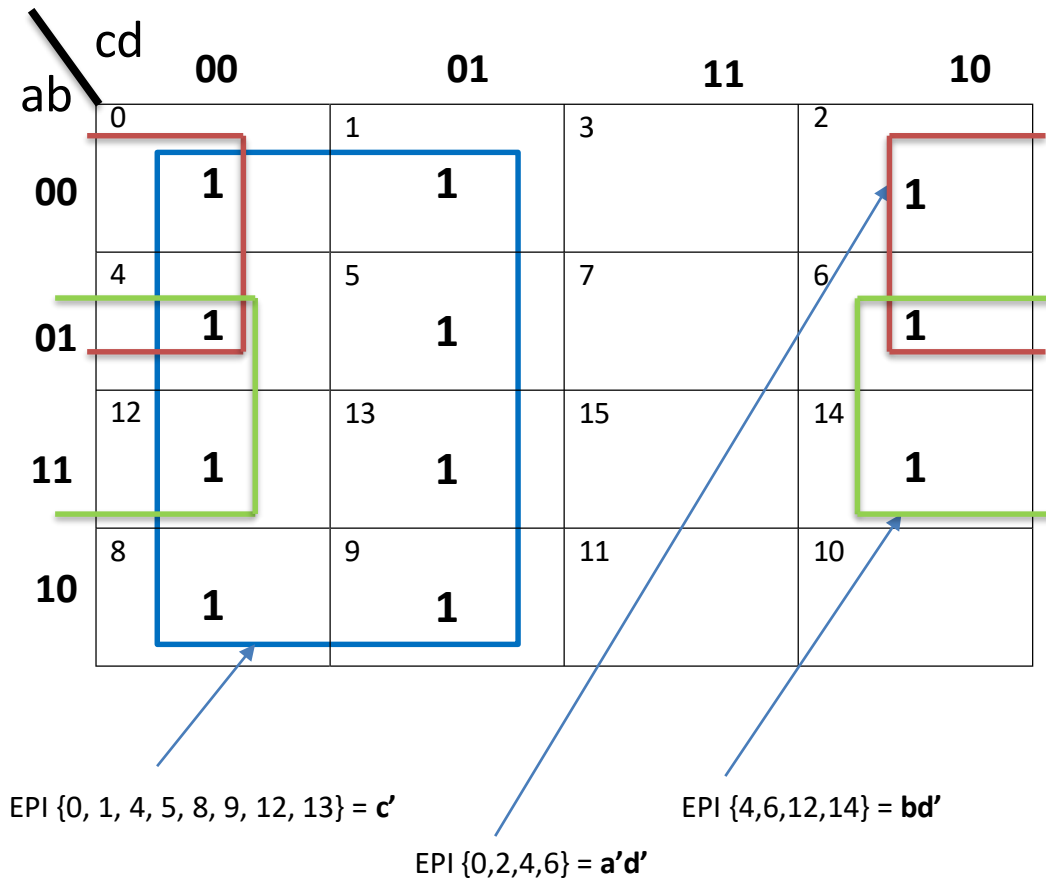
$$P = su' + s'u = s \oplus u \text{ (Ex- OR of P)}$$

d.  $D = f(w, x, y, z) = \sum (5, 7, 8, 9, 13)$



Final expression is  $D = w'xz + wx'y + xy'z$  or  $w'xz + wx'y + wxy'z$

e.  $P = f(a, b, c, d) = \sum (0, 1, 2, 4, 5, 6, 8, 9, 12, 13, 14)$



Final expression is

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

### Assignment Questions

- Write Boolean rules and theorems table
- Solve these expressions using K-map and find Essential Prime Implicants(EPI)
  - $S = f(a, b, c, d) = \sum (1, 3, 4, 5, 7, 8, 9, 11, 15)$
  - $U = f(w, x, y, z) = \sum (1, 5, 7, 8, 9, 10, 11, 13, 15)$
  - $J = f(x, y, z) = \sum (0, 2, 3, 4, 5, 7)$
- Define the following
  - Combinational circuit
  - Canonical Forms
  - Literals
  - Prime Implicants and Essential Prime Implicants

## Five variables K- Map

The below shows the arrangement required to understand five variable K-map. The same rules and principles that applied to three and four variable maps apply here as well. The added **trick** is to see the **logical adjacencies with more squares** in the map.

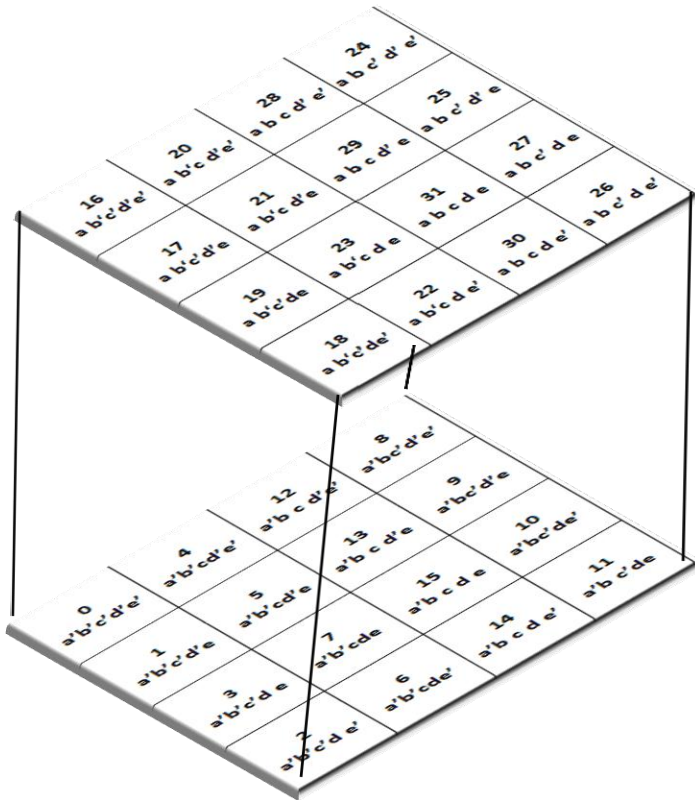
| de \ abc | 000             | 001            | 011              | 010             |  | 100              | 101              | 111                  | 110                |
|----------|-----------------|----------------|------------------|-----------------|--|------------------|------------------|----------------------|--------------------|
| 00       | 0<br>a'b'c'd'e' | 4<br>a'b'cd'e' | 12<br>a'b c d'e' | 8<br>a'bc'd'e'  |  | 16<br>a b'c'd'e' | 20<br>a b'c d'e' | 28<br>a b c d'<br>e' | 24<br>a b c' d' e' |
| 01       | 1<br>a'b'c'd'e  | 5<br>a'b'cd'e  | 13<br>a'b c d'e  | 9<br>a'bc'd'e   |  | 17<br>a b'c'd'e  | 21<br>a b'c d'e  | 29<br>a b c d' e     | 25<br>a b c' d' e  |
| 11       | 3<br>a'b'c'd e  | 7<br>a'b'cde   | 15<br>a'b c d e  | 11<br>a'bc'de   |  | 19<br>a b'c'de   | 23<br>a b'c d e  | 31<br>a b c d e      | 27<br>a b c' d e   |
| 10       | 2<br>a'b'c'd e' | 6<br>a'b'cde'  | 14<br>a'b c d e' | 10<br>a'b c'de' |  | 18<br>a b'c'de'  | 22<br>a b'c d e' | 30<br>a b c d e'     | 26<br>a b c' d e'  |

If we mentally cut the five variable in the centre and slide the right half over on top of the left, we see a picture of two four variable maps aligned vertically.

Let understand how the five variable K-map can identify the logical adjacencies with example as follows:

{0, 16}, {2, 18}, {8, 24} and {10, 26} etc.

There are several different formats of a 5 variable K map of which we discuss two most popular ones in this book: **reflection map and overlay map**.



This picture shows the vertical alignment of two four variable K-Maps

#### Example 9:

Simplify the Boolean functions using 5- variable K- Map technique.

- I.  $T = f(a, b, c, d, e) = \sum (0, 2, 8, 10, 16, 18, 24, 26)$
- II.  $R = f(v, w, x, y, z) = \sum (5, 7, 13, 15, 21, 23, 29, 31)$
- III.  $W = f(a, b, c, d, e) = \sum (1, 3, 4, 6, 9, 11, 12, 14, 17, 19, 20, 22, 25, 27, 28, 30)$
- IV.  $J = f(v, w, x, y, z) = \sum (4, 5, 6, 7, 9, 11, 13, 15, 25, 27, 29, 31)$

In order to solve these questions, you should understand the logical adjacencies of cells in five variable K- Map. Then only the correct minterms can be obtained from the K- Maps.

Solutions:

I.  $T = f(a, b, c, d, e) = \sum (0, 2, 8, 10, 16, 18, 24, 26)$

| de \ abc | 000    | 001 | 011 | 010     | 100     | 101 | 111 | 110     |
|----------|--------|-----|-----|---------|---------|-----|-----|---------|
| 00       | 0<br>1 | 4   | 12  | 8<br>1  | 16<br>1 | 20  | 28  | 24<br>1 |
| 01       | 1      | 5   | 13  | 9       | 17      | 21  | 29  | 25      |
| 11       | 3      | 7   | 15  | 11      | 19      | 23  | 31  | 27      |
| 10       | 2<br>1 | 6   | 14  | 10<br>1 | 18<br>1 | 22  | 30  | 26<br>1 |

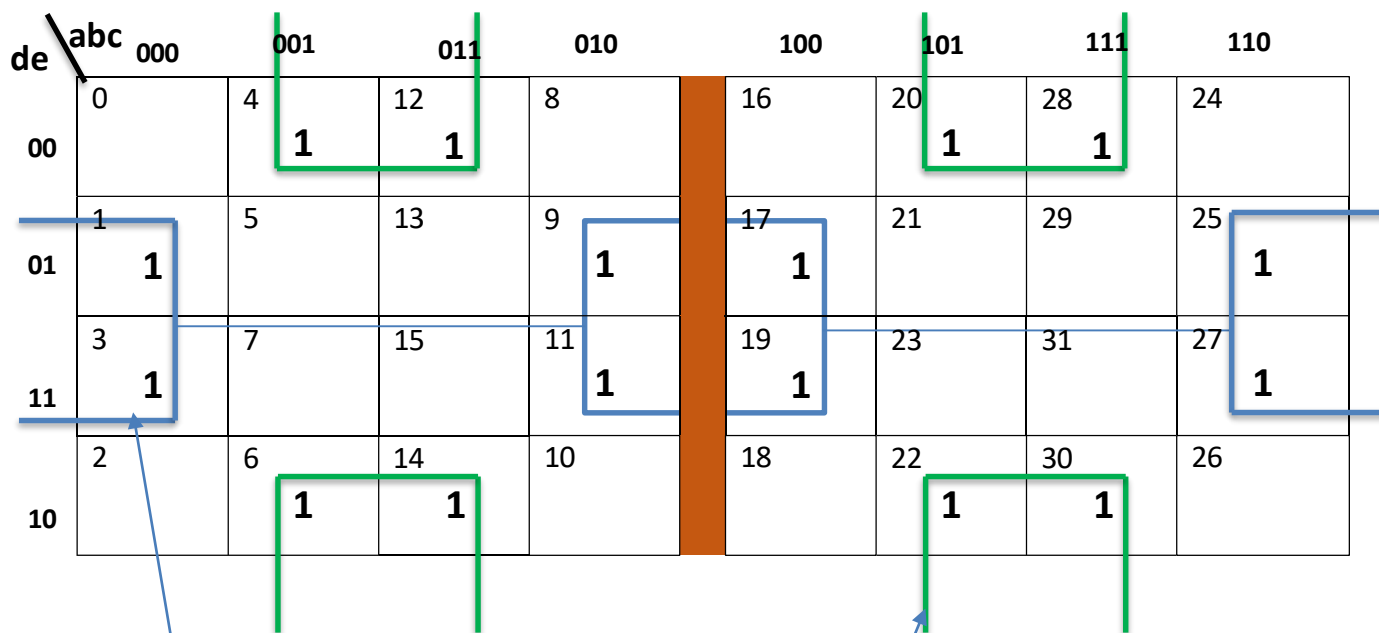
Since all 1 are lies in logically adjacent cells, that is  $\{0, 16\}$ ,  $\{2, 18\}$ ,  $\{8, 24\}$  and  $\{10, 26\}$  are vertically aligned and produces the logical adjacencies. Therefore, all eight ones given in expression will locate in the corners of Map, so it forms **Group of 8** and its minterm is  $T = c' e'$

II.  $R = f(v, w, x, y, z) = \sum (5, 7, 13, 15, 21, 23, 29, 31)$

| yz \ vwx | 000 | 001    | 011     | 010 | 100 | 101     | 111     | 110 |
|----------|-----|--------|---------|-----|-----|---------|---------|-----|
| 00       | 0   | 4      | 12      | 8   | 16  | 20      | 28      | 24  |
| 01       | 1   | 5<br>1 | 13<br>1 | 9   | 17  | 21<br>1 | 29<br>1 | 25  |
| 11       | 3   | 7<br>1 | 15<br>1 | 11  | 19  | 23<br>1 | 31<br>1 | 27  |
| 10       | 2   | 6      | 14      | 10  | 18  | 22      | 30      | 26  |

Exactly four cells are at each side of the Map and it forms an EPI and its minterm is  $R = x z$

III.  $W = f(a, b, c, d, e) = \sum (1, 3, 4, 6, 9, 11, 12, 14, 17, 19, 20, 22, 25, 27, 28, 30)$

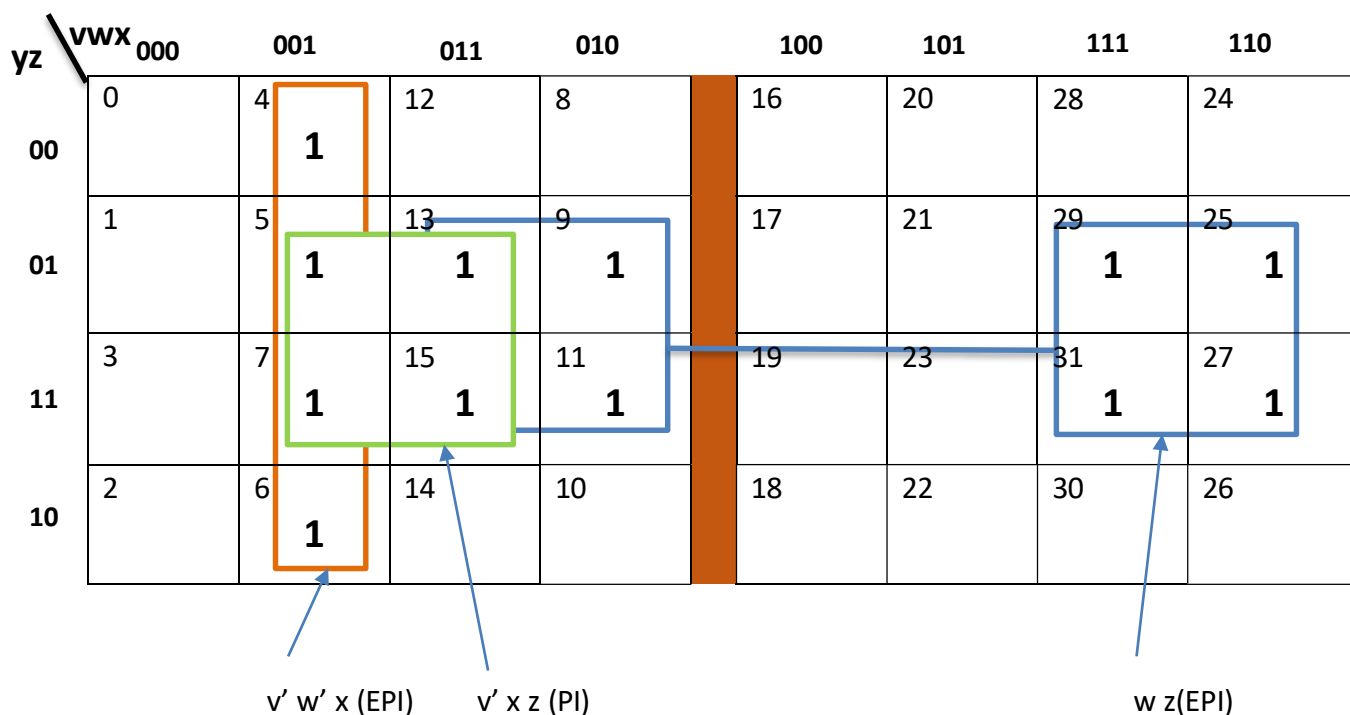


EPI from  $\{1, 3, 9, 11, 17, 19, 25, 27\} = c' e$

EPI  $\{4, 6, 12, 14, 20, 22, 28, 30\} = c e'$

IV.  $J = f(v, w, x, y, z) = \sum (4, 5, 6, 7, 9, 11, 13, 15, 25, 27, 29, 31)$

Solution:  $J = v' w' x + w z$



$v' w' x$  (EPI)

$v' x z$  (PI)

$w z$  (EPI)

## Incompletely Specified Functions (Don't Care Terms):

When an output value is known for every possible combination of input variables, the function is said to be **completely specified**. However, when an output variable is not known for every combination of input variables, usually because all combinations cannot occur, the function is said to be **incompletely specified**.

The minterms or maxterms that are not used as a part of the output function are called **don't care terms**.

For instance, the truth table for conversion of binary to EX-3 BCD as follows:

| Binary |   |   |   | EX-3 BCD |   |   |   |
|--------|---|---|---|----------|---|---|---|
| w      | x | y | z | A        | B | C | D |
| 0      | 0 | 0 | 0 | 0        | 0 | 1 | 1 |
| 0      | 0 | 0 | 1 | 0        | 1 | 0 | 0 |
| 0      | 0 | 1 | 0 | 0        | 1 | 0 | 1 |
| 0      | 0 | 1 | 1 | 0        | 1 | 1 | 0 |
| 0      | 1 | 0 | 0 | 0        | 1 | 1 | 1 |
| 0      | 1 | 0 | 1 | 1        | 0 | 0 | 0 |
| 0      | 1 | 1 | 0 | 1        | 0 | 0 | 1 |
| 0      | 1 | 1 | 1 | 1        | 0 | 1 | 0 |
| 1      | 0 | 0 | 0 | 1        | 0 | 1 | 1 |
| 1      | 0 | 0 | 1 | 1        | 1 | 0 | 0 |
| 1      | 0 | 1 | 0 | d        | d | d | d |
| 1      | 0 | 1 | 1 | d        | d | d | d |
| 1      | 1 | 0 | 0 | d        | d | d | d |
| 1      | 1 | 0 | 1 | d        | d | d | d |
| 1      | 1 | 1 | 0 | d        | d | d | d |
| 1      | 1 | 1 | 1 | d        | d | d | d |

Writing the equations for output variables A, B, C and D, including don't care terms, we get

$$A = f(w, x, y, z) = \sum (5,6,7,8,9) + \sum d(10,11,12,13,14,15)$$

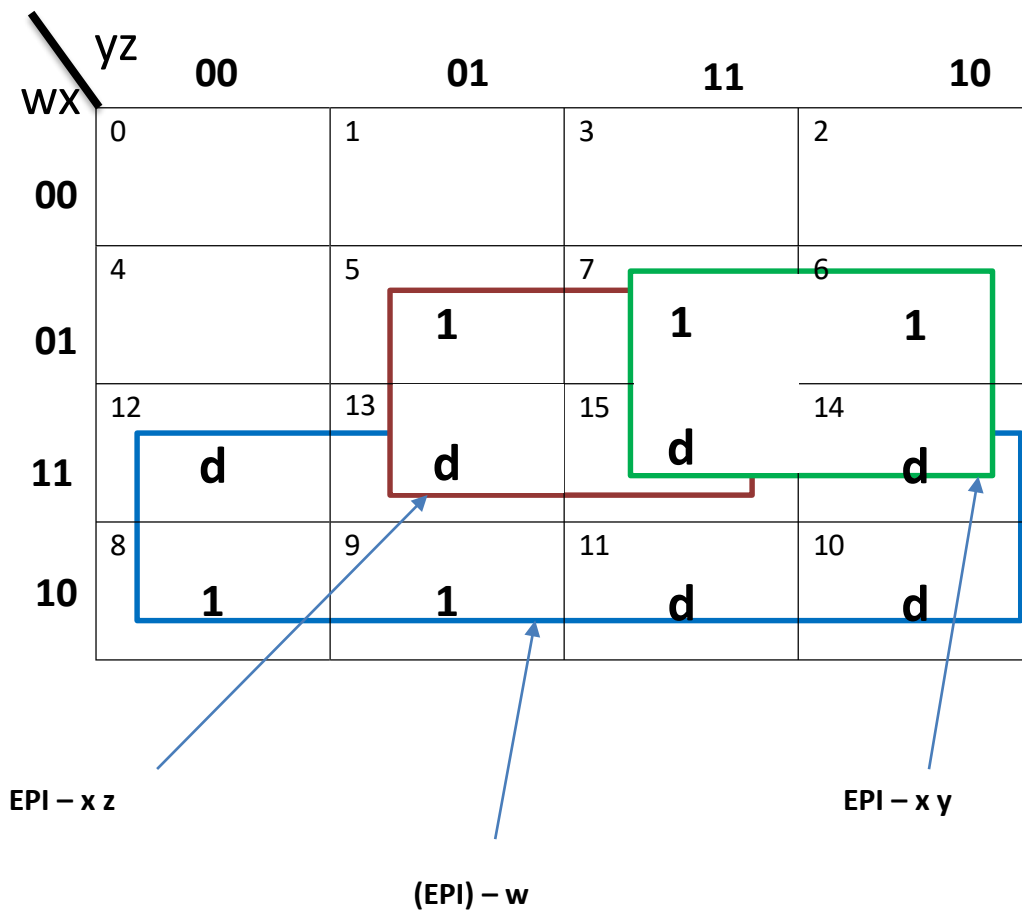
$$B = f(w, x, y, z) = \sum (1,2,3,4,9) + \sum d(10,11,12,13,14,15)$$

$$C = f(w, x, y, z) = \sum (0,3,4,7,8) + \sum d(10,11,12,13,14,15)$$

$$D = f(w, x, y, z) = \sum (0,2,4,6,8) + \sum d(10,11,12,13,14,15)$$

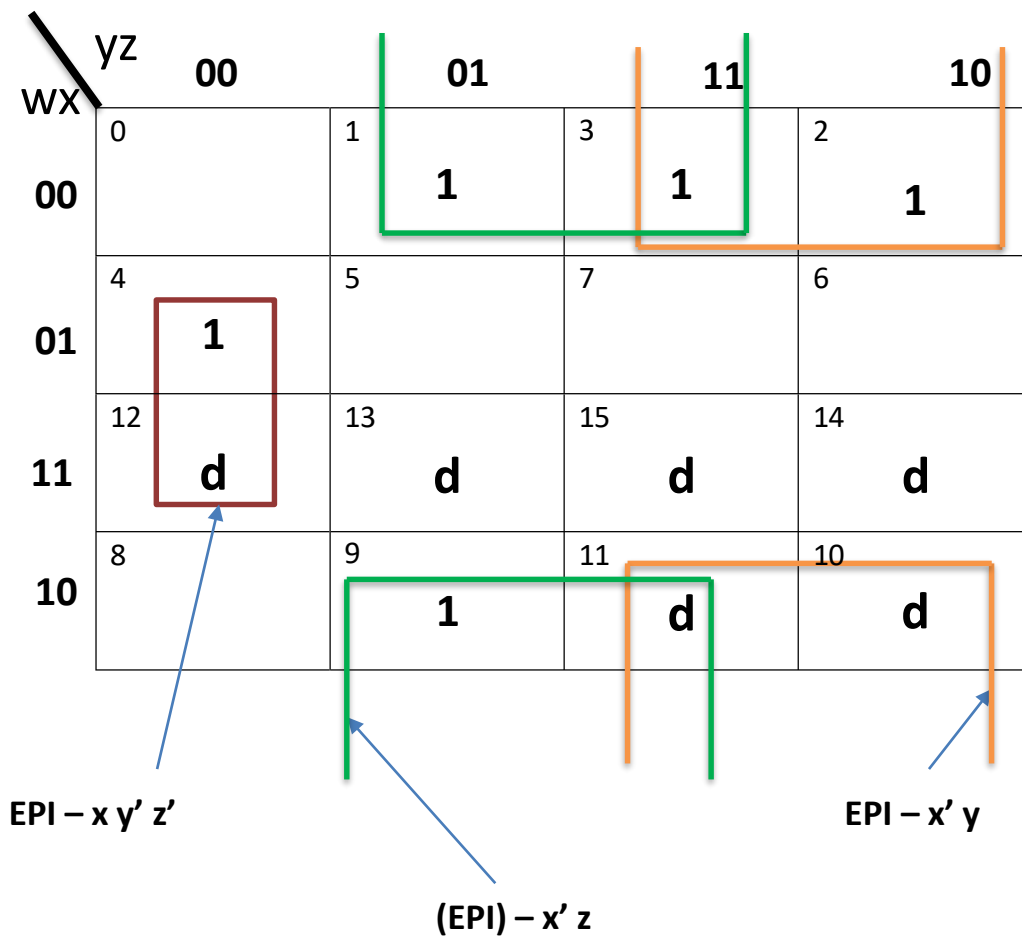
Solution for A code in the table by solving the expression including don't care terms.

$$A = f(w, x, y, z) = \sum (5,6,7,8,9) + \sum d (10,11,12,13,14,15)$$



Solution for B code in the table by solving the expression including don't care terms.

$$B = f(w, x, y, z) = \sum (1,2,3,4,9) + \sum d (10,11,12,13,14,15)$$



Solution for C code in the table by solving the expression including don't care terms.

$$C = f(w, x, y, z) = \sum (0,3,4,7,8) + \sum d (10,11,12,13,14,15)$$

| yz |    | 00 | 01 | 11 | 10 |
|----|----|----|----|----|----|
| wx | 0  | 1  |    | 1  |    |
|    | 00 |    |    |    |    |
|    | 4  | 1  |    | 1  |    |
|    | 01 |    |    |    |    |
|    | 12 | d  | d  | d  | d  |
|    | 11 |    |    |    |    |
|    | 8  | 1  |    | d  | d  |
|    | 10 |    |    |    |    |

$EPI - y' z'$        $EPI - y z$

Solution for D code in the table by solving the expression including don't care terms.

$$D = f(w, x, y, z) = \sum (0, 2, 4, 6, 8) + \sum d (10, 11, 12, 13, 14, 15)$$

| YZ<br>WX | 00      | 01      | 11      | 10      |
|----------|---------|---------|---------|---------|
| 00       | 0<br>1  | 1       | 3       | 2<br>1  |
| 01       | 4<br>1  | 5       | 7       | 6<br>1  |
| 11       | 12<br>d | 13<br>d | 15<br>d | 14<br>d |
| 10       | 8<br>1  | 9       | 11<br>d | 10<br>d |

EPI – Z'

### Simplify Maxterm Equations:

Loading and grouping Maxterms is exactly same as loading and grouping minterms, except that 0s are loaded into the map and grouped to form the PI and EPI groups. Reading the map is, however, different.

#### Example 10:

Simplify the Boolean functions using K- Map technique.

- I.  $J = f(x, y, z) = \prod (0, 3, 4, 7)$
- II.  $G = f(a, b, c, d) = \prod (0, 4, 5, 7, 8, 9, 11, 12, 13, 15)$
- III.  $W = f(w, x, y, z) = \prod (1, 3, 8, 10, 12, 13, 14, 15)$
- IV.  $J = f(v, w, x, y, z) = \prod (0, 2, 4, 6, 8, 9, 10, 11, 12, 14, 16, 17, 18, 19, 24, 25, 26, 27)$

Solutions:

I.  $J = f(x, y, z) = \Pi(0, 3, 4, 7)$

|   |   | yz |   | 00 | 01 | 11 | 10 |
|---|---|----|---|----|----|----|----|
| X | 0 | 0  | 0 | 1  | 3  | 0  | 2  |
|   | 1 | 4  | 0 | 5  | 7  | 0  | 6  |

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

II.  $G = f(a, b, c, d) = \Pi(0, 4, 5, 7, 8, 9, 11, 12, 13, 15)$

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

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

III.  $W = f(w, x, y, z) = \Pi(1, 3, 8, 10, 12, 13, 14, 15)$

| yz \ wx | 00      | 01      | 11      | 10      |
|---------|---------|---------|---------|---------|
| 00      | 0       | 1<br>0  | 3<br>0  | 2       |
| 01      | 4       | 5       | 7       | 6       |
| 11      | 12<br>0 | 13<br>0 | 15<br>0 | 14<br>0 |
| 10      | 8<br>0  | 9       | 11      | 10<br>0 |

$$W = (w' + z)(w' + x')(w + x + z')$$

IV.  $L = f(v, w, x, y, z) = \Pi(0, 2, 4, 6, 8, 9, 10, 11, 12, 14, 16, 17, 18, 19, 24, 25, 26, 27)$

| yz \ vwx | 000    | 001    | 011     | 010     | 100     | 101 | 111 | 110     |
|----------|--------|--------|---------|---------|---------|-----|-----|---------|
| 00       | 0<br>0 | 4<br>0 | 12<br>0 | 8<br>0  | 16<br>0 | 20  | 28  | 24<br>0 |
| 01       | 1      | 5      | 13      | 9<br>0  | 17<br>0 | 21  | 29  | 25<br>0 |
| 11       | 3      | 7      | 15      | 11<br>0 | 19<br>0 | 23  | 31  | 27<br>0 |
| 10       | 2<br>0 | 6<br>0 | 14<br>0 | 10<br>0 | 18<br>0 | 22  | 30  | 26<br>0 |

$v + z$  (EPI)

$w' + x$  (EPI)

$v' + x$  (EPI)

$$\text{Solution: } L = (v + z)(w' + x)(v' + x)$$

## Quine – McCluskey Minimization Technique

For many applications the number of variables in a problem is too large to simplify manually, using Karnaugh Maps. K-Maps, although capable of handling any number of variables, become complex beyond six or seven variables.

**Simplification typically means that a logic designer can obtain more functional use from a given component. Henceforth, some automation or computer - driven simplification routine is desirable.**

The **Quine – McCluskey Minimization Technique (QM technique)** is an algorithm that uses the same Boolean rules that were employed with Karnaugh Maps but in a form suitable for a computer solution. Large Karnaugh maps require recognition of groups of terms that may form essential prime implicants. **The larger the map, the more difficult this pattern recognition becomes. This approach eliminates the need for such pattern recognition.**

### Procedure to solve the problem using QM technique

**Step 1** – Arrange the given minterms in an **ascending order** and make the groups based on the number of ones present in their binary representations. So, there will be at most ' $n+1$ ' groups if there are ' $n$ ' Boolean variables in a Boolean function or ' $n$ ' bits in the binary equivalent of min terms.

**Step 2** – **Compare the minterms present in successive groups (group  $n$  and group  $n+1$ ).** If there is a **change in only one-bit position**, then take the pair of those two min terms. **Place this symbol '\_'** in the differed bit position and keep the remaining bits as it is.

**Step 3** – all of the adjacent minterm groups contained are compared to see if groups of four can be made. The criteria are as follows: the dashes in the groups of two must be in the same position and only one variable change is allowed.

**Step 4** – Formulate the **prime implicant table**. It consists of set of rows and columns. Prime implicants can be placed in row wise and minterms can be placed in column wise. Place 'x' in the cells corresponding to the minterms that are covered in each prime implicant.

**Step 5** – Find the **essential prime implicants by observing each column**. If the minterm is covered only by one prime implicant, then it is essential prime implicant. Those essential prime implicants will be part of the simplified Boolean function.

**Step 6** – Reduce the prime implicant table by removing the row of each essential prime implicant and the columns corresponding to the minterms that are covered in that essential prime implicant. Repeat step 5

for Reduced prime implicant table. Stop this process when all min terms of given Boolean function are over.

### Example 11:

Simplify the following using the Quine – McCluskey minimization technique

$$D = f(a, b, c, d) = \sum (0, 1, 2, 3, 6, 7, 8, 9, 14, 15)$$

**Solution:**

1. Arrange all the minterms in ascending order (according to the number of 1s contained in each term), to use the  $x + x' = 1$  can be applied. Table 1 shows the arrangement of given minterms.

**Table 1: shows the arrangement of given minterms.**

| Groups | Minterm | Variables |   |   |   |
|--------|---------|-----------|---|---|---|
|        |         | a         | b | c | d |
| 0      | 0       | 0         | 0 | 0 | 0 |
| 1      | 1       | 0         | 0 | 0 | 1 |
|        | 2       | 0         | 0 | 1 | 0 |
|        | 8       | 1         | 0 | 0 | 0 |
| 2      | 3       | 0         | 0 | 1 | 1 |
|        | 6       | 0         | 1 | 1 | 0 |
|        | 9       | 1         | 0 | 0 | 1 |
| 3      | 7       | 0         | 1 | 1 | 1 |
|        | 14      | 1         | 1 | 1 | 0 |
| 4      | 15      | 1         | 1 | 1 | 1 |

**Table 2: shows the arrangement of two grouping After matching the minterms (one bit differ)**

| Groups | Minterm | Variables |   |   |   |
|--------|---------|-----------|---|---|---|
|        |         | a         | b | c | d |
| 0      | 0, 1    | 0         | 0 | 0 | – |
| 0      | 0, 2    | 0         | 0 | – | 0 |
| 0      | 0, 8    | –         | 0 | 0 | 0 |
| 1      | 1, 3    | 0         | 0 | – | 1 |
| 1      | 1, 9    | –         | 0 | 0 | 1 |
| 1      | 2, 3    | 0         | 0 | 1 | – |
| 1      | 2, 6    | 0         | – | 1 | 0 |
| 1      | 8, 9    | 1         | 0 | 0 | – |
| 2      | 3, 7    | 0         | – | 1 | 1 |
| 2      | 6, 7    | 0         | 1 | 1 | – |
| 2      | 6, 14   | –         | 1 | 1 | 0 |
| 3      | 7, 15   | –         | 1 | 1 | 1 |
| 3      | 14, 15  | 1         | 1 | 1 | – |

2. Create a new table showing the minterms in group n that matched with group n+1, such that they differ in only one position.

Eliminated variable bit positions are indicated by the dash ( \_ )

Table 2: shows the arrangement of two grouping after matching the minterms (one bit differ) shown with dash symbol at one variable change positions.

3. All of the adjacent minterm groups contained in table 2 are compared to see if groups of four can be made. The criteria for forming groups of four are as follows: the dashes in the groups of two must be in the same bit position and only one variable change is allowed. Table 3 shows the creation of minterms groups of four.

**Table 3: shows the creation of minterms groups of four.**

| Groups | Minterm      | Variables |   |   |   |
|--------|--------------|-----------|---|---|---|
|        |              | a         | b | c | d |
| 0      | 0, 1, 2, 3   | 0         | 0 | _ | _ |
| 0      | 0, 1, 8, 9   | _         | 0 | 0 | _ |
| 1      | 2, 6, 3, 7   | 0         | _ | 1 | _ |
| 2      | 6, 7, 14, 15 | _         | 1 | 1 | _ |

Repeat this process if until you get all the possible groups of higher elements.

4. All of the prime implicants are formed into prime implicant table shown in table 4. This table lists each of the minterm contained in the original switching equation across the top of the table.

| PI Terms | decimal      | Minterms |   |   |   |   |   |     |     |     |     |
|----------|--------------|----------|---|---|---|---|---|-----|-----|-----|-----|
|          |              | 0        | 1 | 2 | 3 | 6 | 7 | 8   | 9   | 14  | 15  |
| a' b'    | 0, 1, 2, 3   | x        | x | x | x |   |   |     |     |     |     |
| b' c'    | 0, 1, 8, 9   | x        | x |   |   |   |   | (x) | (x) |     |     |
| a' c     | 2, 6, 3, 7   |          |   | x | x | x | x |     |     |     |     |
| b c      | 6, 7, 14, 15 |          |   |   |   | x | x |     |     | (x) | (x) |

Note that {0, 1, 8, 9} and {6, 7, 14, 15} are essential prime implicants. Minterms {2, 3} are contained in two prime implicants {0, 1, 2, 3} and {2, 3, 6, 7}. We need one of these prime implicants to cover the minterms in the equation not both.

So the solution for the given expression have two equivalent solutions as follows:

$$D = b' c' + b c + a' b' \quad \text{OR} \quad D = b' c' + b c + a' c.$$

**Example 12:**

**Simplify the following using the Quine – McCluskey minimization technique**

$$D = f(a, b, c, d) = \sum m(1, 3, 13, 15) + \sum d(8, 9, 10, 11)$$

**Solution:** Indicated don't care terms by using a \* symbol. Don't care terms are never included as prime implicants by themselves.

**Step 1:**

| Groups | Minterm | Variables |   |   |   |
|--------|---------|-----------|---|---|---|
|        |         | a         | b | c | d |
| 1      | 1       | 0         | 0 | 0 | 1 |
|        | 8*      | 1         | 0 | 0 | 0 |
| 2      | 3       | 0         | 0 | 1 | 1 |
|        | 9*      | 1         | 0 | 0 | 1 |
|        | 10*     | 1         | 0 | 1 | 0 |
| 3      | 11*     | 1         | 0 | 1 | 1 |
|        | 13      | 1         | 1 | 0 | 1 |
| 4      | 15      | 1         | 1 | 1 | 1 |

**Step 2:**

| Groups | Minterm  | Variables |   |   |   |
|--------|----------|-----------|---|---|---|
|        |          | a         | b | c | d |
| 1      | 1, 3     | 0         | 0 | – | 1 |
|        | 1, 9*    | –         | 0 | 0 | 1 |
|        | 8*, 9*   | 1         | 0 | 0 | – |
|        | 8*, 10*  | 1         | 0 | – | 0 |
| 2      | 3, 11*   | –         | 0 | 1 | 1 |
|        | 9*, 11*  | 1         | 0 | – | 1 |
|        | 9*, 13   | 1         | – | 0 | 1 |
|        | 10*, 11* | 1         | 0 | 1 | – |
| 3      | 11*, 15  | 1         | – | 1 | 1 |
|        | 13, 15   | 1         | 1 | – | 1 |

**Step 3:**

| Groups | Minterm          | Variables |   |   |   |
|--------|------------------|-----------|---|---|---|
|        |                  | a         | b | c | d |
| 1      | 1, 3, 9*, 11     | –         | 0 | – | 1 |
|        | 8*, 9*, 10*, 11* | 1         | 0 | – | – |
| 2      | 9*, 13, 11*, 15  | 1         | – | – | 1 |

Step 4:

| Groups | Decimal         | Minterms |         |         |         |
|--------|-----------------|----------|---------|---------|---------|
|        |                 | 1        | 3       | 13      | 15      |
| 1      | 1, 3, 9*, 11    | $\odot$  | $\odot$ |         |         |
| 2      | 9*, 13, 11*, 15 |          |         | $\odot$ | $\odot$ |

The resulting simplified equation is  $S = x' z + w z$



