Unit 3: Simplification of Boolean Functions

Canonical forms:

Boolean function expressed as a sum of minterms or product of maxterms are said to be in canonical form.

Minterms and Maxterms

<u>**Minterms:**</u>

A binary variable may appear either in its normal form (x) or its complement form (x'). If two binary variables x and y combined with AND operation, each variable may appear in any from (x or x'). i.e there are four possible combinations: x'y' , x'y , xy' and xy. Each of these four terms represents the distinct areas and is called a **minterms** or a **standard product**.

* A Boolean variable and its complement are called literals.

- Each minterm is obtained from an AND term of *n* variables.
- Minterms are represented by "m" like: m0, m1, m2, etc.
- Minterm is a product of all the literals (with or without complement).
- Example if we have two Boolean variables X and Y then X.Y' is a minterm

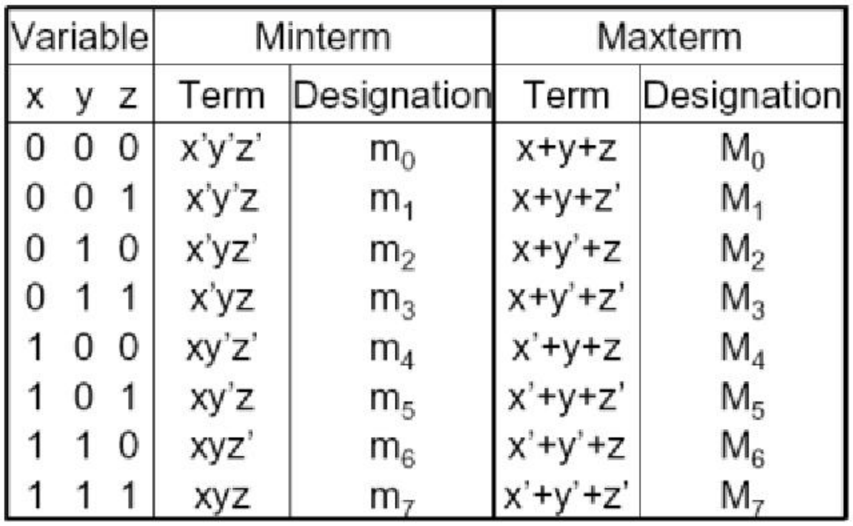
<u>**Maxterms:**</u>

n variables forming an OR term, with each variable is called **maxterms** or **standard sums.**

- Each maxterm is obtained from an OR term of n variables.
- Maxterms are represented by "M" like: M0, M1, M2, etc.
- Maxterm is a sum of all the literals (with or without complement).
- For example if we have two Boolean variables X and Y then X + Y' is a maxterm

Each maxterm is the complement of its corresponding minterms, and vice versa.

A Boolean function may be expressed algebraically in the form of maxterm and minterms. The following table shows the minterms ad maxterms of three variable x,y and z.



Variable			Minterm		Maxterm	
x	y	z	Term	Designation	Term	Designation
0	0	0	x'y'z'	m_0	x+y+z	M_0
0	0	1	x'y'z	m_1	x+y+z'	M_1
0	1	0	x'yz'	m_2	x+y'+z	M_2
0	1	1	x'yz	m_3	x+y'+z'	M_3
1	0	0	xy'z'	m_4	x'+y+z	M_4
1	0	1	xy'z	m_5	x'+y+z'	M_5
1	1	0	xyz'	m_6	x'+y'+z	M_6
1	1	1	xyz	m_7	x'+y'+z'	M_7

So, it is easily verified that, any Boolean function can be expressed as a sum of minterms and product of maxterms

f1= x'yz+xy'z+xyz'+xyz
= m3+m5+m6+m7

f1= (x'+y+z)(x+y'+z)(x+y+z')(x+y+z)
= M3.M5.M6.M7

Product of Sum (POS) Simplification

• Sum of Product (SOP)

- The concept of the sum of products (SOP) mainly includes minterm(m)
– see entries with value 1.
– Ensure output 1 for those inputs. So OR
– A'B+AB'
- In SOP the different product inputs are being added togeth

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

• Product of Sum (POS)

- the product of sums (POS) mainly includes the maxterm(M
– see entries with value 0
– Ensure output 0 for those input. So AND
– (A+B) (A'+B')
- In maxterm, every input is complemented

K-map

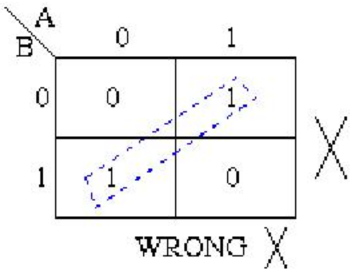
Karnaugh introduced a method for simplification of Boolean functions in an easy way. This method is known as **Karnaugh map** method or **K-map** method. It is a graphical method, which consists of 2^n cells for 'n' variables. The adjacent cells are differed only in single bit position. K-Map method is most suitable for minimizing Boolean functions of 2 variables to 5 variables.

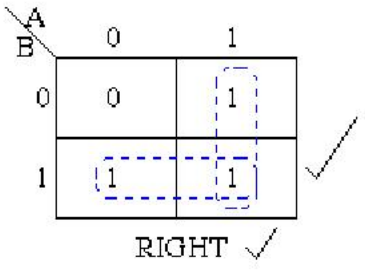
- The k-map provides a simple and straight-forward method of minimizing Boolean expressions.
- A k-map provides a pictorial method of grouping together expressions with common factors
- It eliminates unwanted variables.
- The K-map can also be described as a special arrangement of a truth table.
- Reductions could be done with Boolean algebra. However, the Karnaugh map is faster and easier

Rules for simplifying K-maps

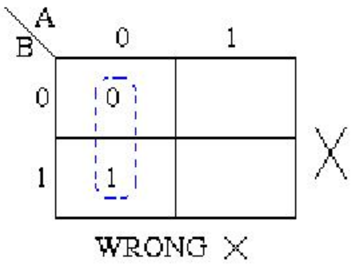
The Karnaugh map (k- map) uses the following rules for the simplification of expressions by grouping together adjacent cells containing 1's.

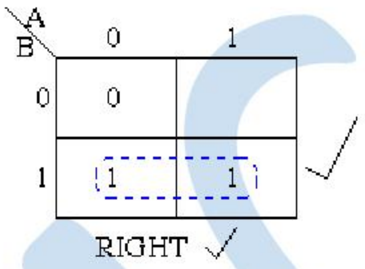
1. Groups may be horizontal or vertical, but not diagonal.



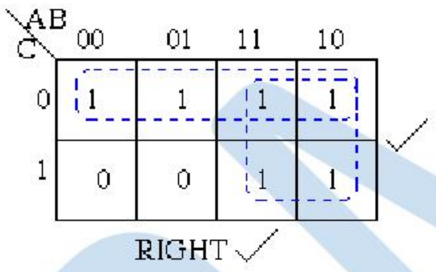


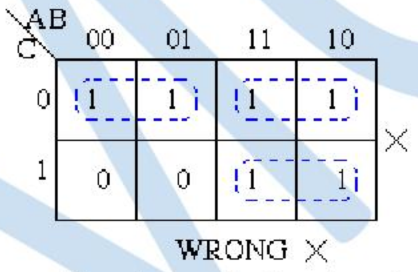
2. Groups may not include any cell containing a zero





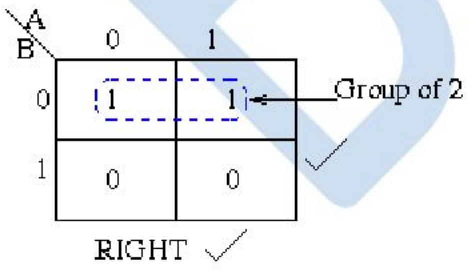
3. Each group should be as large as possible.

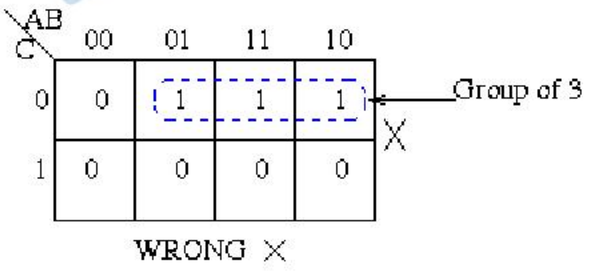


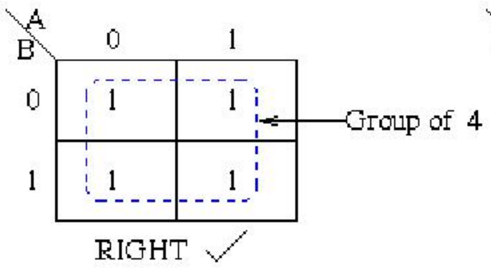


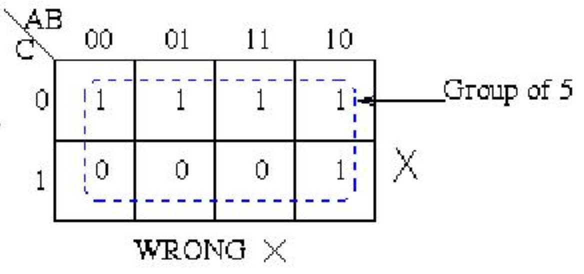
(Note that no Boolean laws broken, but not sufficiently minimal)

4. Groups must contain 1, 2, 4, 8, or in general 2^n cells. That is if n = 1, a group will contain two 1's since $2^1 = 2$. If n = 2, a group will contain four 1's since $2^2 = 4$.



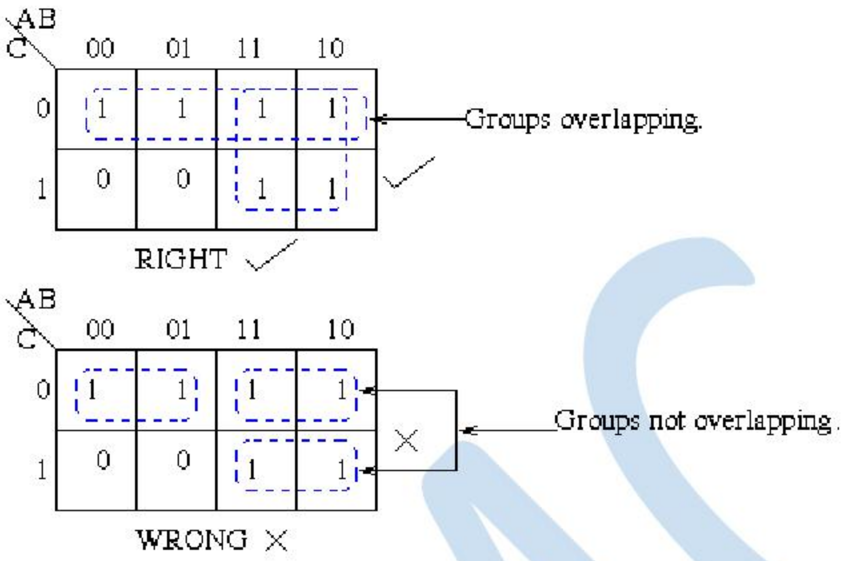




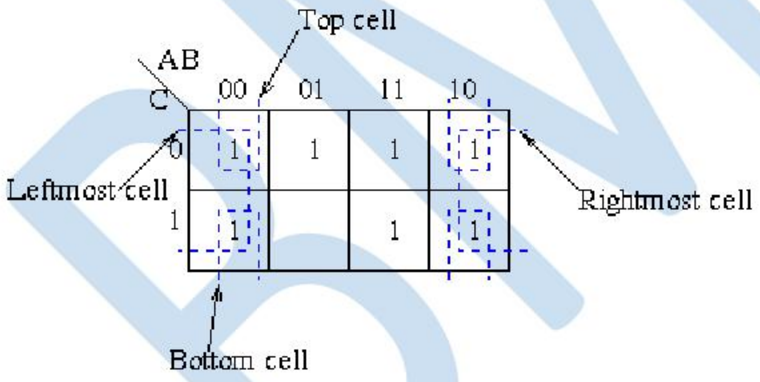


5. Each cell containing a one must be in at least one group.

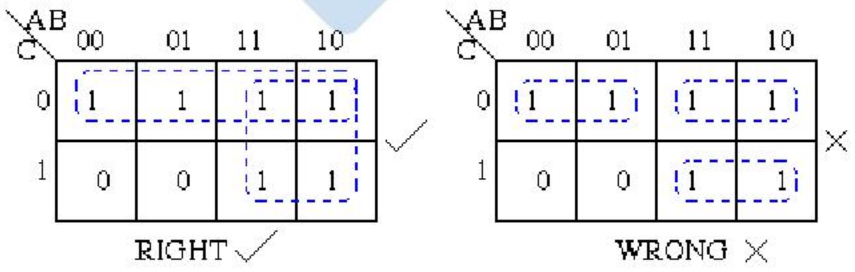
6. Groups may overlap.



7. Groups may wrap around the table. The leftmost cell in a row may be grouped with the rightmost cell and the top cell in a column may be grouped with the bottom cell.



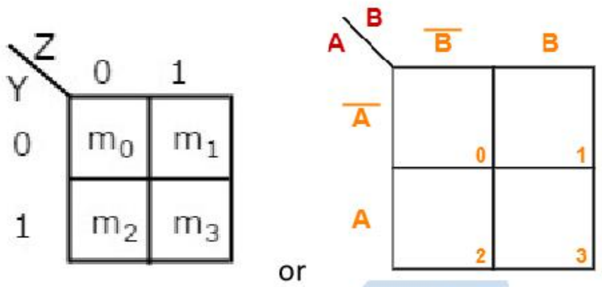
8. There should be as few groups as possible, as long as this does not contradict any of the previous rules.



Note – If don't care terms also present, then place don't cares 'x' in the respective cells of K-map. Consider only the don't cares 'x' that are helpful for grouping maximum number of adjacent cells.

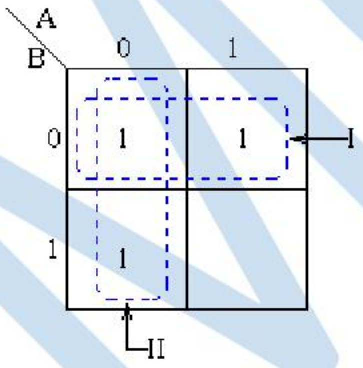
2 Variable K-Map

The number of cells in 2 variable K-map is four (2^2), since the number of variables is two. The following figure shows 2 variable K-Map.



- There is only one possibility of grouping 4 adjacent min terms.
- The possible combinations of grouping 2 adjacent min terms are {(m0, m1), (m2, m3), (m0, m2) and (m1, m3)}

Example: Consider the expression Z = f(A,B) = $\overline{A}\,\overline{B} + A\,\overline{B} + \overline{A}B$ plotted on the K- map

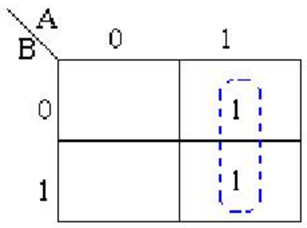


Pairs of 1's are grouped as shown above, and the simplified answer is obtained by using the following steps:

1. The first group labelled I, consists of two 1s which correspond to A = 0, B = 0 and A = 1, B = 0. Put in another way, all squares in this example that correspond to the area of the map where B = 0 contains 1s, independent of the value of A. So when B = 0 the output is 1. The expression of the output will contain the term $\overline{B}$

2. For group labelled II corresponds to the area of the map where A = 0. The group can therefore be defined as $\overline{A}$. This implies that when A = 0 the output is 1. The output is therefore 1 whenever B = 0 and A = 0
   Hence the simplified answer is Z = $\overline{A} + \overline{B}$

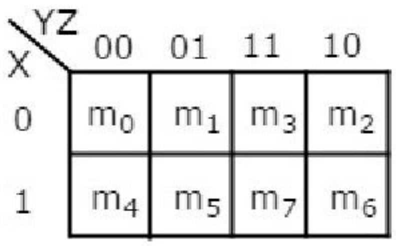
Another Example:



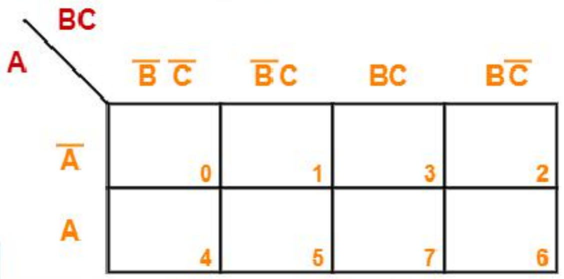
Through inspection it can be seen that variable B has its true and false form within the group. This eliminates variable B leaving only variable A which only has its true form. The minimized answer therefore is f = A

3 Variable K-Maps

The number of cells in 3 variables K-map is eight (2^3), since the number of variables is three. The following figure shows 3 variables K-Map.

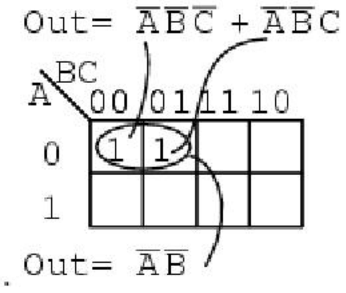


or

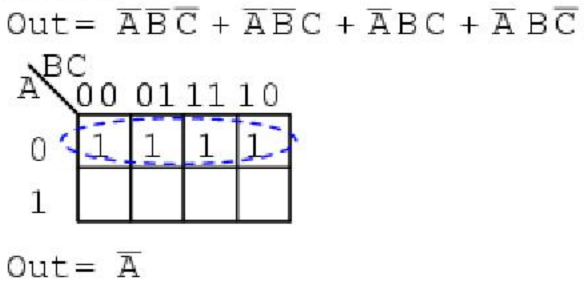

BC
A
$\overline{B}\,\overline{C}$ $\overline{B}C$ BC $B\overline{C}$
$\overline{A}$
A
0 1 3 2
4 5 7 6

- The number of cells present in three variable K Map = 2^3 = 8 cells.
- There is only one possibility of grouping 8 adjacent min terms.
- The possible combinations of grouping 4 adjacent min terms are {(m0, m1, m3, m2), (m4, m5, m7, m6), (m0, m1, m4, m5), (m1, m3, m5, m7), (m3, m2, m7, m6) and (m2, m0, m6, m4)}.
- The possible combinations of grouping 2 adjacent min terms are {(m0, m1), (m1, m3), (m3, m2), (m2, m0), (m4, m5), (m5, m7), (m7, m6), (m6, m4), (m0, m4), (m1, m5), (m3, m7) and (m2, m6)}.

Example 1:

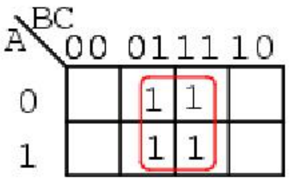

Out= $\overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,\overline{B}C$
BC
A
00 01 11 10
0
1
1 1
Out= $\overline{A}\,\overline{B}$

Example 2:


Out= $\overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,\overline{B}C + \overline{A}BC + \overline{A}B\overline{C}$
BC
A
00 01 11 10
0
1
1 1 1 1
Out= $\overline{A}$

Example 3:

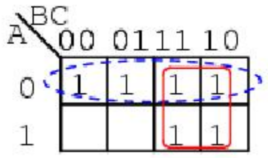
$$Out = \overline{A}\,\overline{B}C + \overline{A}BC + A\overline{B}C + ABC$$



$$Out = C$$

Example 4:

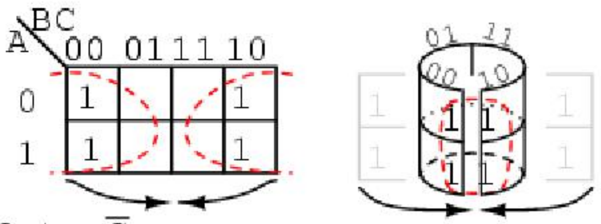
$$Out = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,\overline{B}C + \overline{A}BC + \overline{A}B\overline{C} + ABC + AB\overline{C}$$



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

Example 5:

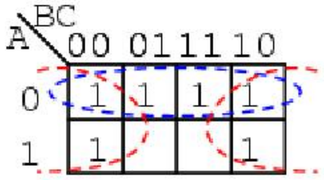
$$Out = \overline{A}\,\overline{B}\,\overline{C} + A\overline{B}\,\overline{C} + \overline{A}B\overline{C} + AB\overline{C}$$



$$Out = \overline{C}$$

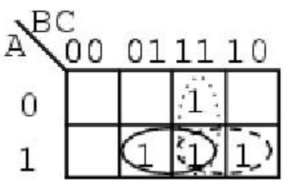
Example 6:

Out= A B C +A B C+A B C+A B C +A B C +A B C



$$Out = \overline{A} + \overline{C}$$

Example 7:

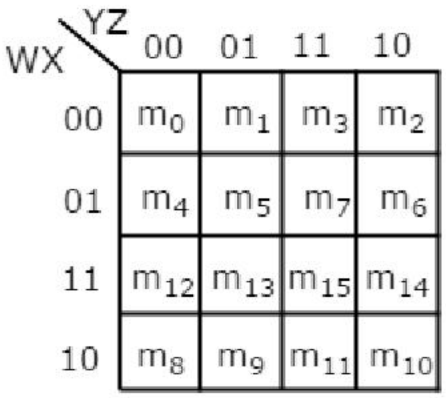


Output = AB + BC + AC

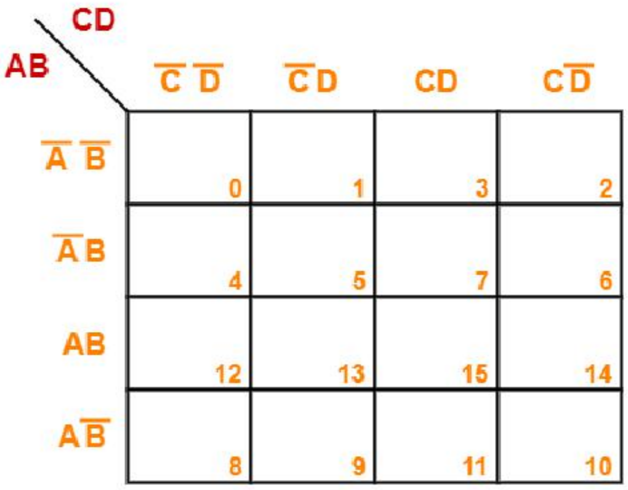
4 Variable K-Map

The number of cells in 4 variable K-map is sixteen (2^4), since the number of variables is four. The following figure shows 4 variable K-Map.

- The number of cells present in four variable K Map = 2^4 = 16 cells.

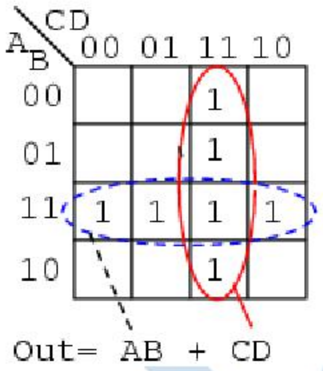

WX
YZ
00 01 11 10
00 m_0 m_1 m_3 m_2
01 m_4 m_5 m_7 m_6
11 m_{12} m_{13} m_{15} m_{14}
10 m_8 m_9 m_{11} m_{10}

or



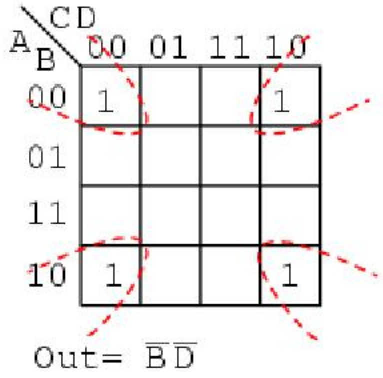
Example 1:

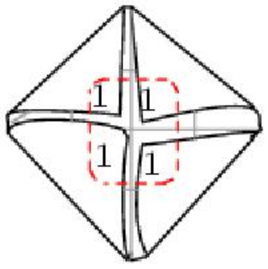
$$Out = \bar{A}\bar{B}CD + \bar{A}BCD + ABCD + A\bar{B}CD + AB\bar{C}\bar{D} + AB\bar{C}D + ABC\bar{D}$$



Example 2:

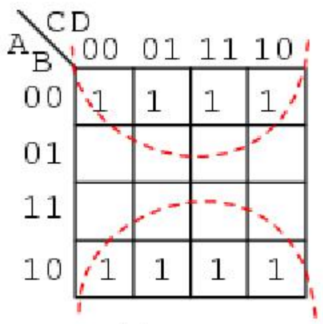
$$Out = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}C\bar{D} + A\bar{B}\bar{C}\bar{D} + A\bar{B}C\bar{D}$$





Example 3:

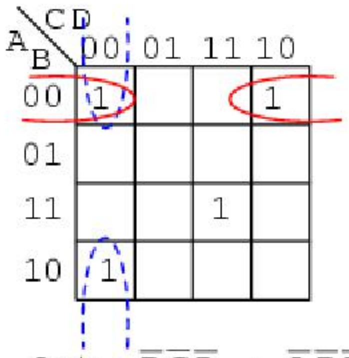
$$\text{Out} = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}CD + \overline{A}\,\overline{B}C\overline{D} + A\overline{B}\,\overline{C}\,\overline{D} + A\overline{B}\,\overline{C}D + A\overline{B}CD + A\overline{B}C\overline{D}$$



$$\text{Out} = \overline{B}$$

Example 4:

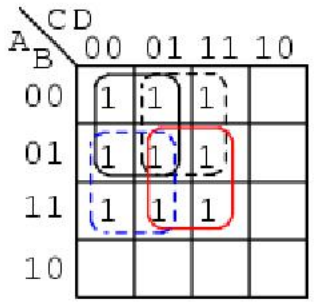
$$\text{Out} = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}C\overline{D} + A\overline{B}\,\overline{C}\,\overline{D} + ABCD$$



$$\text{Out} = \overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{D} + ABCD$$

Example 5:

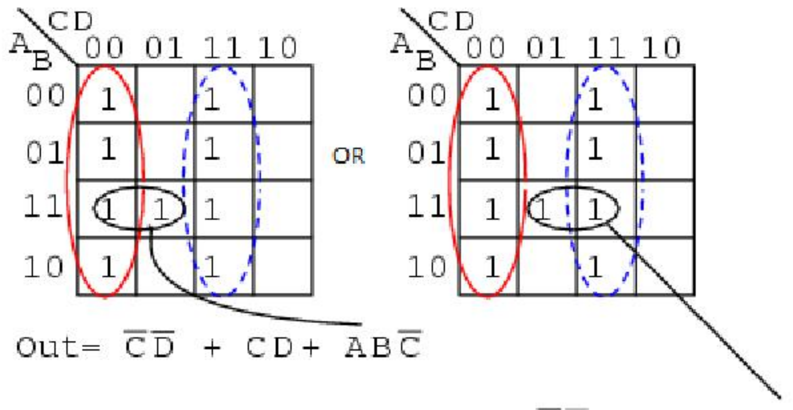
$$f = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}CD + \overline{A}B\overline{C}\,\overline{D} + \overline{A}B\overline{C}D + \overline{A}BCD + AB\overline{C}\,\overline{D} + AB\overline{C}D + ABCD$$



$$Out = \overline{A}\,\overline{C} + \overline{A}D + B\overline{C} + BD$$

Example 6:

$$Out = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}CD + \overline{A}B\overline{C}\,\overline{D} + \overline{A}BCD + AB\overline{C}\,\overline{D} + AB\overline{C}D + ABCD + A\overline{B}\,\overline{C}\,\overline{D} + A\overline{B}CD$$



$$Out = \overline{C}\,\overline{D} + CD + AB\overline{C}$$

$$Out = \overline{C}\,\overline{D} + CD + ABD$$

Don't Care Conditions:

The "Don't Care" conditions allow us to replace the empty cell of a K-Map to form a grouping of the variables. While forming groups of cells, we can consider a "Don't Care" cell as either 1 or 0 or we can simply ignore that cell. Therefore, "Don't Care" condition can help us to form a larger group of cells. For an example, if we are dealing with BCD numbers then (1010, 1011, 1100, 1101, 1110, and 1111) will never exists. These invalid codes are don't care conditions. That is, we do not care what output our logic circuit produces for these don't cares.

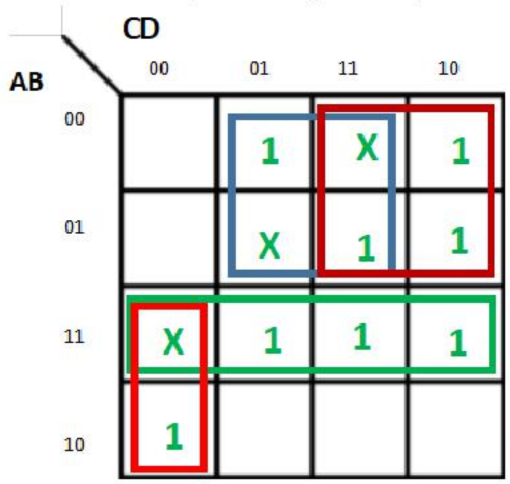
- ⊕ A Don't Care cell can be represented by a cross(X) in K-Maps
- ⊕ There is no requirement to group all or any of the don't cares. Only use them in a group if it simplifies the logic.

Significance of Don't care conditions

1. These conditions denotes the set of inputs which never occurs for a given digital circuits. Thus, they are being used to further simplify the Boolean output expression.
2. Simplification reduces the number of gates to be used for implementing the given expression. Therefore, don't cares make the digital circuit design more economical.
3. While grouping the terms long with don't cares reduces switching of the states. This decreases the required memory space which in turn results in less power consumption.
4. Don't cares also prevents hazards in digital systems.

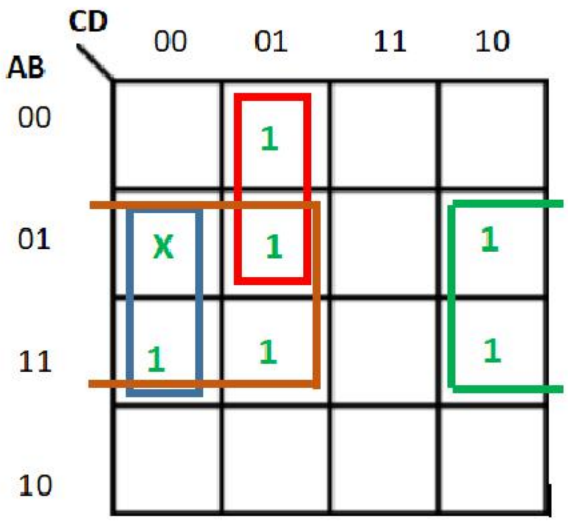
Example: Minimize the following function in SOP minimal form using K-Maps:
F (A, B, C, D) = m(1, 2, 6, 7, 8, 13, 14, 15) + d(3, 5, 12)
Sol^n

The SOP K-map for the given expression is:



Therefore, f = AC'D' + A'D + A'C + AB

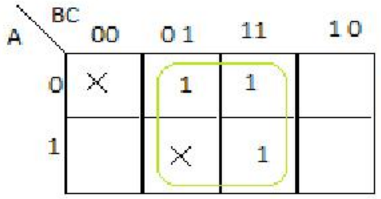
Another Example Explained:
Minimize the following function using K-Maps:
f = m(1, 5, 6, 12, 13, 14) + d(4)



f = BC' + BD' + A'C'D

Example 3:

F=∑m(1, 3, 7) + ∑d(0, 5)



* Don't circle the X that don't help

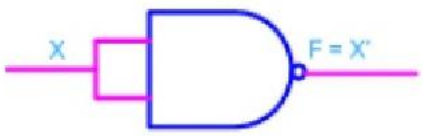
F= C

NAND and NOR Implementation

NAND and NOR gates are called universal gates because we can implement any logic function using NAND gates.

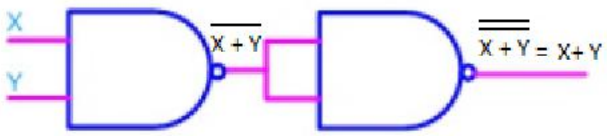
1. Implementing NOT using NAND gates:

Input	Output
X	$\overline{X}$



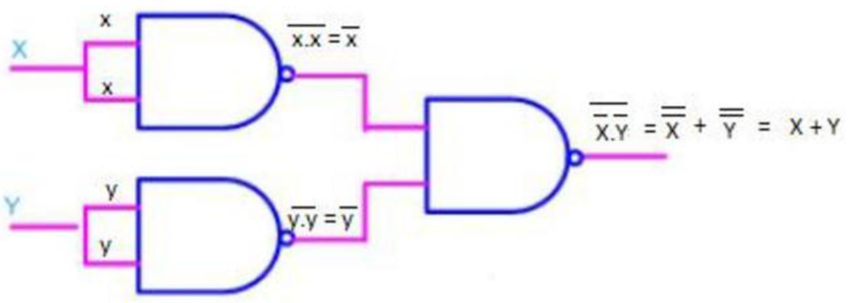
2. Implementing AND using NAND gates

Input	Output
X Y	XY



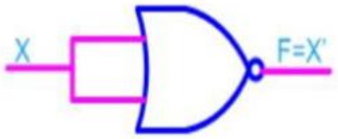
3. Implementing OR using NAND gates:

Input	Output
X Y	X+Y

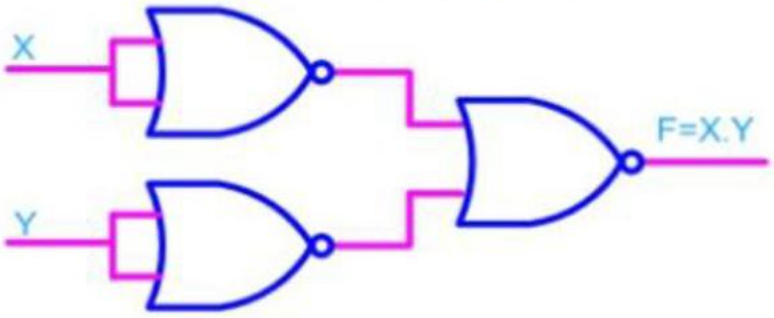

X
Y
x
x
y
y
$\overline{x.x} = \overline{x}$
$\overline{y.y} = \overline{y}$
$\overline{\overline{X}.\overline{Y}} = \overline{\overline{X}} + \overline{\overline{Y}} = X + Y$

Any logic function can be implemented using NOR gates only.

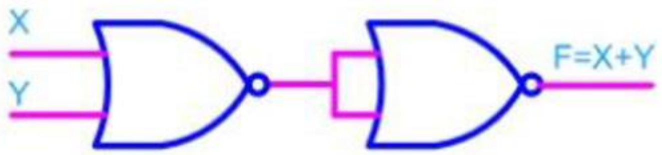
1. Implementing NOT gate with using NOR gates:



2. Implementing AND gate with using NOR gates:



3. Implementing OR gate with using NOR gates.



Problems:

1. What is k map? Write the rules for simplifying k map.
2. Obtain the simplified expression in (1) Sum of Products(sop)and (2) Product of Sum (POS)
    a. F(x,y,z)=x'yz'+xyz'+xy'z+xyz
3. Simplify the Boolean function using don't care conditions (d), in sum of products and product of sums form.
    $F = w'x'z' + w'yz + w'xy$
    $d = w'xy'z + wyz + wx'z'$
4. What is don't care condition in k map? Explain.
5. Minimize the following Boolean function using K- map
    a. F(A, B, C, D) = Σm(0, 1, 2, 5, 7, 8, 9, 10, 13, 15)
    b. F(A, B, C, D) = Σm(0, 1, 3, 5, 7, 8, 9, 11, 13, 15)
    c. F(W, X, Y, Z) = Σm(1, 3, 4, 6, 9, 11, 12, 14)
    d. F(A, B, C, D) = Σm(1, 3, 4, 6, 8, 9, 11, 13, 15) and Σd(0, 2, 14)
    e. F(A, B, C) = Σm(0, 1, 6, 7) + Σd(3, 5)
    f. F(A, B, C, D) = Σm(0, 2, 8, 10, 14) + Σd(5, 15)

- End of Unit 3 -