Logic Gate

College of
Engineering & Technology

Al-Mustaqbal
University

Level 1 , Semester 1
@ Department of prosthetic and orthotic Engineering

Prepared by
Dr. Samir Badrawi
2024-2025

Simplification for digital circuits

The majority of this course material is based on text and presentations of :

SOP and POS forms

Boolean expressions can be written in the **sum-of-products** form (**SOP**) or in the **product-of-sums** form (**POS**).

These forms can simplify the implementation of combinational logic.

In both forms, an overbar cannot extend over more than one variable.

An expression is in **SOP** form when two or more product terms are summed as in the following examples:

$\overline{A}\,\overline{B}\,\overline{C} + A\,B$ $\quad A\,B\,\overline{C} + \overline{C}\,\overline{D}$ $\quad C\,D + \overline{E}$

An expression is in **POS** form when two or more sum terms are multiplied as in the following examples:

$(A + B)(\overline{A} + C)$ $\quad (A + B + \overline{C})(B + D)$ $\quad (\overline{A} + B)C$

SOP Standard form

In **SOP standard form**, every variable in the domain must appear in each term in this standard form.

This form is useful for constructing truth tables or for implementing logic in certain digital devices called PLDs (Programmable Logic Devices)

Nonstandard form can be expanded to standard form by multiplying this term by a term consisting of the sum of the missing variable and its complement.

Example Convert $X = \overline{A}\,\overline{B} + A\,B\,C$ to standard form.

Solution The first term does not include the variable C. Therefore, multiply it by the $(C + \overline{C})$, which = 1:

$$X = \overline{A}\,\overline{B}(C + \overline{C}) + A\,B\,C$$
$$= \overline{A}\,\overline{B}\,C + \overline{A}\,\overline{B}\,\overline{C} + A\,B\,C$$

POS Standard form

In **POS standard form**, every variable in the domain must appear in each sum term of the expression.

Nonstandard POS expression can be expanded to standard form by adding the product of the missing variable and its complement and applying rule 12, which states that $(A + B)(A + C) = A + BC$.

Note. can we Rewrite Rule (12) as: $X + YZ = (X + Y)(X + Z)$ (Yes/No) ??

Example Convert $X = (\overline{A} + \overline{B})(A + B + C)$ to standard form.

Solution The first sum term does not include the variable C.
Therefore, add $C\overline{C}$ and expand the result by rule 12.

$$X = (\overline{A} + \overline{B} + C\overline{C})(A + B + C)$$
$$= (\overline{A} + \overline{B} + C)(\overline{A} + \overline{B} + \overline{C})(A + B + C)$$

(Prove it ?)

Binary representation of SOP and POS forms

SOP standard form $A\overline{B}C\overline{D} = 1 \cdot \overline{0} \cdot 1 \cdot \overline{0} = 1 \cdot 1 \cdot 1 \cdot 1 = 1$

POS standard form $A + \overline{B} + C + \overline{D} = 0 + \overline{1} + 0 + \overline{1} = 0 + 0 + 0 + 0 = 0$

Converting standard SOP to POS

SOP standard form

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

$$000 + 010 + 011 + 101 + 111$$

The equivalent POS standard form contains the other three remaining terms $001,\ 100\ \ and\ \ 110$

$$(A + B + \overline{C})(\overline{A} + B + C)(\overline{A} + \overline{B} + C)$$

Converting SOP to truth table

1. First list all possible combinations of binary values of the variables in the expression.
2. Convert the SOP to standard form if it is not already.
3. Place a 1 in the output column for each binary value that makes the standard SOP expression a 1 and place a 0 for all the remaining binary values

Example: Develop a Truth Table for the standard SOP Expression :-

$$\overline{A}\,\overline{B}C + A\overline{B}\,\overline{C} + ABC$$

Solution :Three variables, then 8 possible combinations.
For each product term in the Expression, place (1) in o/p, and place (0) in for the other terms in o/p

I/P A	B	C	O/P X	PRODUCT TERM
0	0	0	0	
0	0	1	1	$\overline{A}\,\overline{B}\,C$
0	1	0	0	
0	1	1	0	
1	0	0	1	$A\,\overline{B}\,\overline{C}$
1	0	1	0	
1	1	0	0	
1	1	1	1	ABC

Karnaugh maps

A **K-Map** or **Karnaugh Map** is a graphical method that used for simplifying the complex algebraic expressions in Boolean functions. It is a tool for simplifying combinational logic with **3 or 4 variables.**

A Boolean function in **three** variables (A, B, C) can be expressed in the **Standard sum of product (SOP)** form that can have total **eight** possible combinations.
These combinations are designated by **m_0, m_1, m_2, m_3, m_4, m_5, m_6, and m_7** respectively. Each of these terms are called a **minterm**.

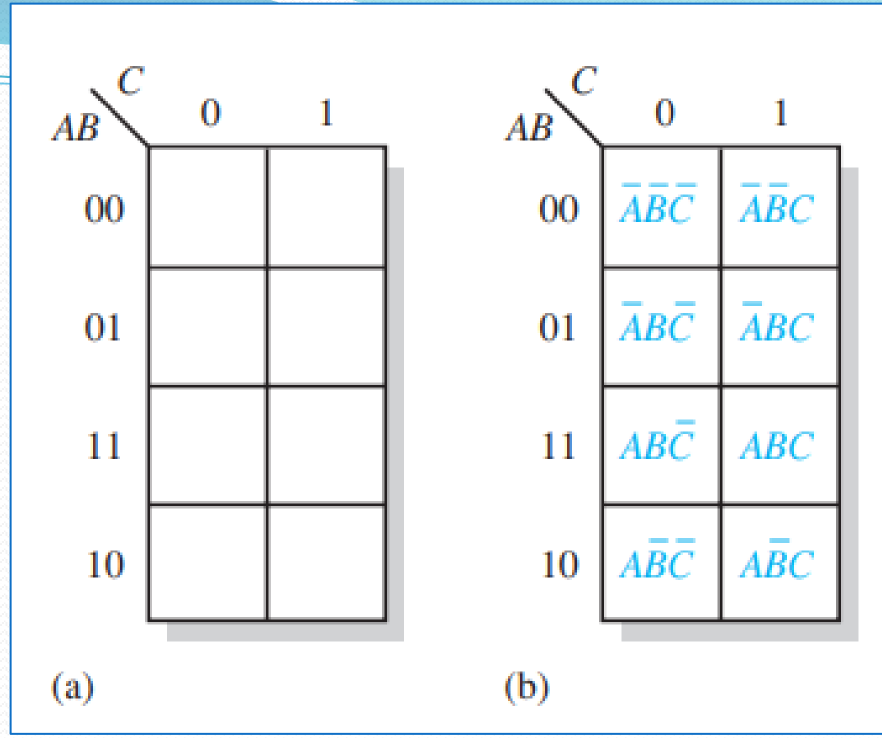
In terms **of POS (Product of Sum)** form, the combinations are often designated by **M_0, M_1, M_2, M_3, M_4, M_5, M_6, and M_7** respectively. Each of these terms is called a **maxterm.**

minterm (m's) and maxterms (M's) may take any binary value of (1) or (0)

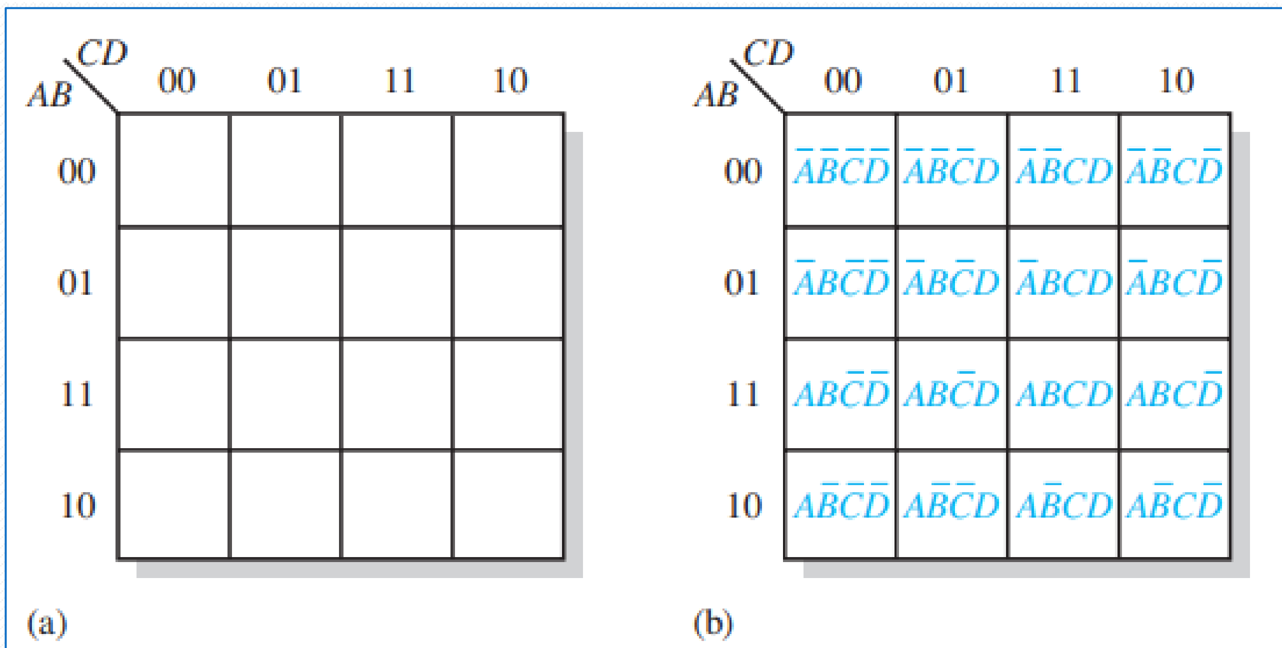
Karnaugh maps

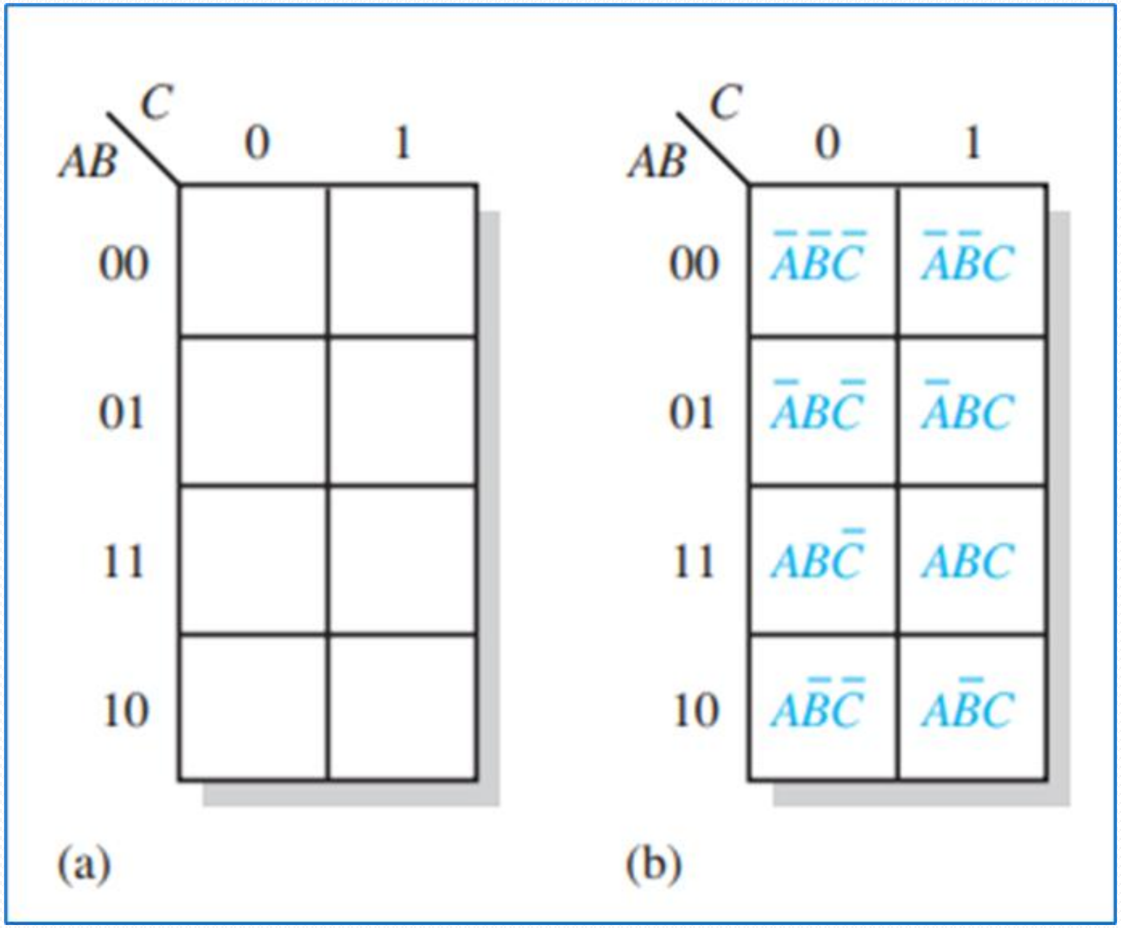
- Array of cells, each cell represents one possible term.
- For 3 variables, 8 cells are required (2^3).
- Each cell is adjacent to cells that are immediately next to it on any of its four sides.
- A cell is not adjacent to the cells that diagonally touch any of its corners.
- "wrap-around" adjacency means the top row is adjacent to the bottom row and left column to right column.

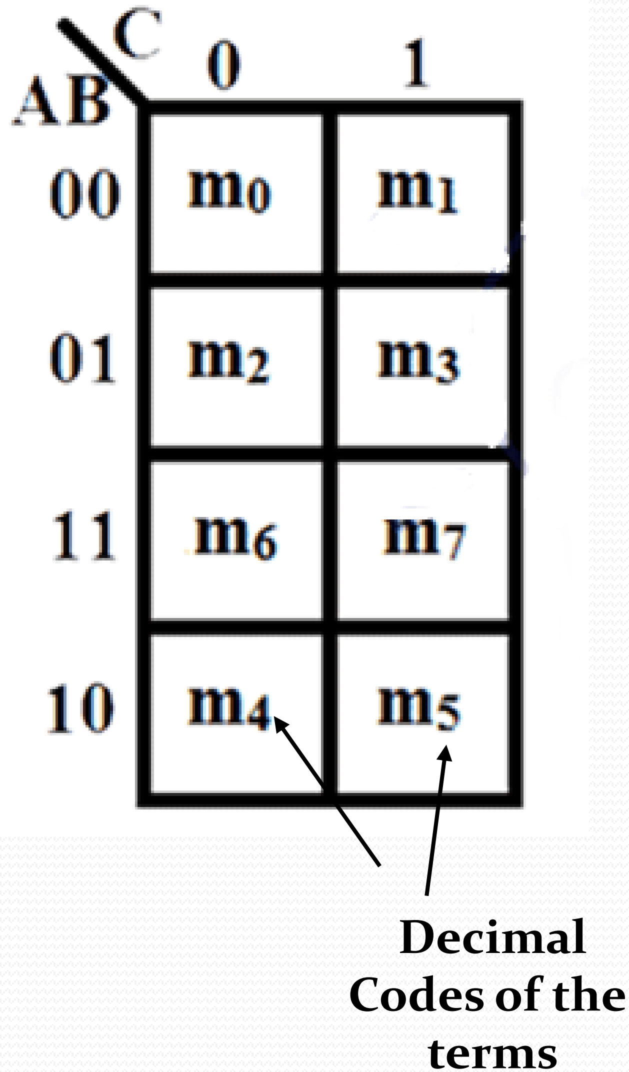
(**SOP**) mapping on 3-Variable K-Map.

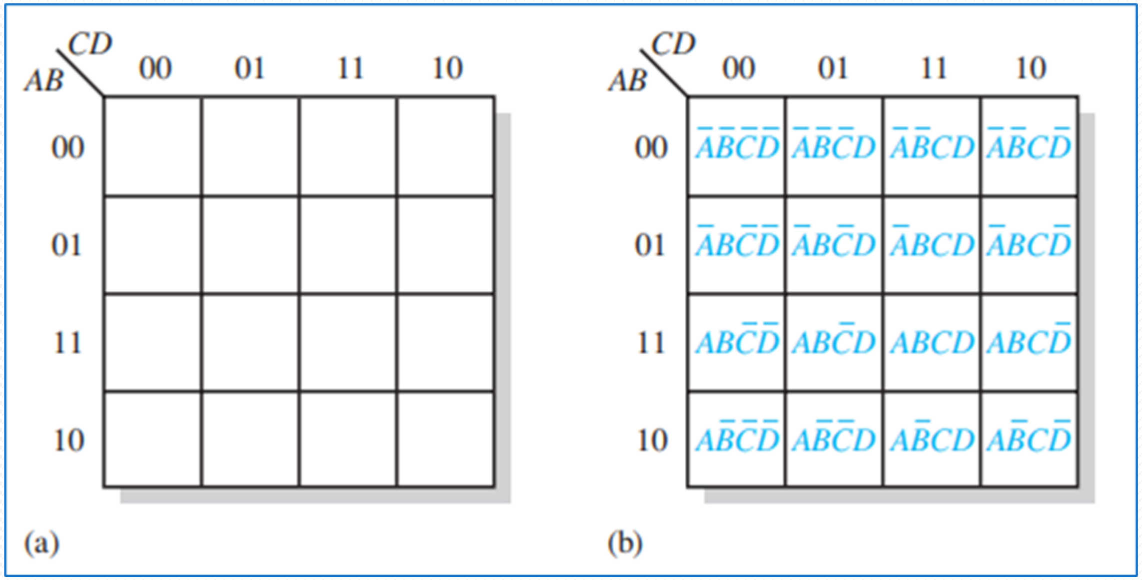


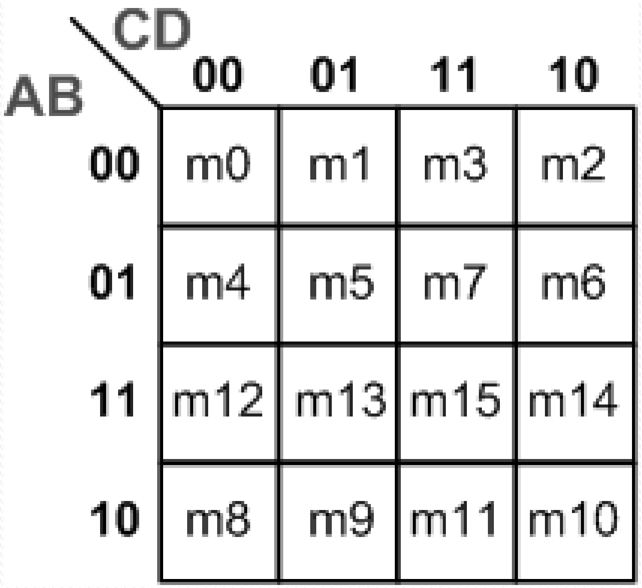
(**SOP**) mapping on 4-Variable K-Map.



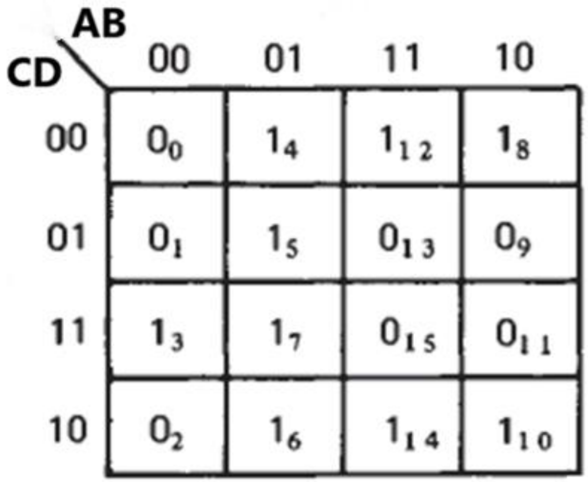

C
AB
0
1
00
01
11
10
(a)
C
AB
0
1
00
$\bar{A}\bar{B}\bar{C}$
$\bar{A}\bar{B}C$
01
$\bar{A}B\bar{C}$
$\bar{A}BC$
11
$AB\bar{C}$
ABC
10
$A\bar{B}\bar{C}$
$A\bar{B}C$
(b)


C
AB
0
1
00
m_0
m_1
01
m_2
m_3
11
m_6
m_7
10
m_4
m_5
Decimal
Codes of the
terms





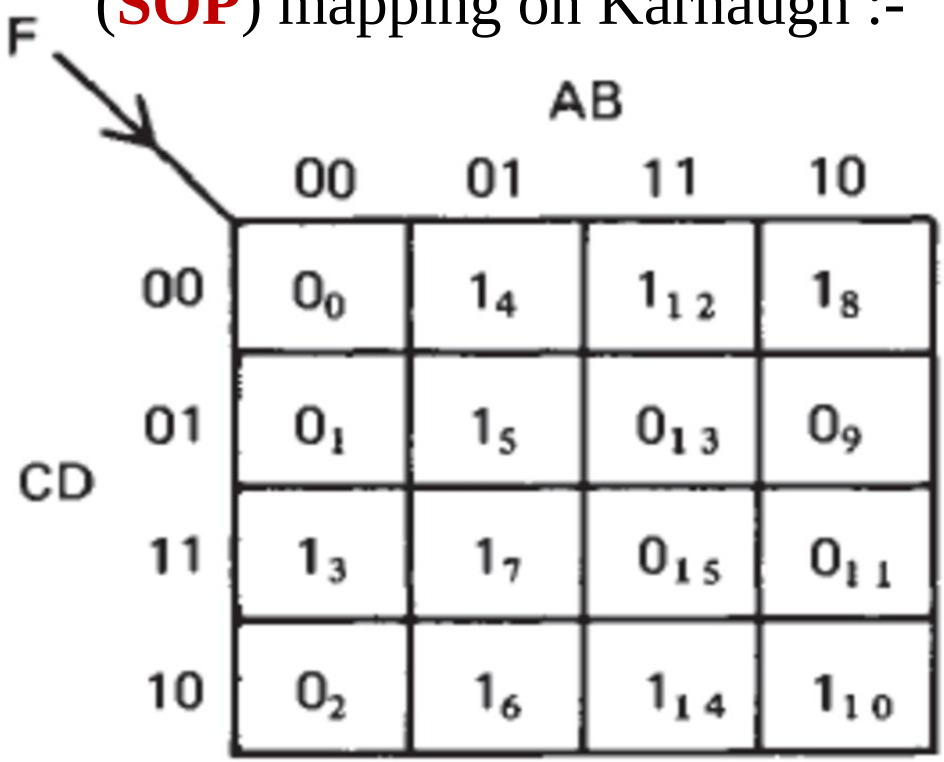
Can we represent 4-var. K-map as shown ?



Decimal/Binary mapping on Karnaugh Map

A Decimal/Binary sequence from zero to fifteen (i.e. we have 4-variables).

Assume a Function (**F**) has the following (**SOP**) mapping on Karnaugh :-



$$F = \Sigma(3,4,5,6,7,8,10,12,14)$$

Binary Number Code	Binary Number A B C B	F
0	0 0 0 0	0
1	0 0 0 1	0
2	0 0 1 0	0
3	0 0 1 1	**1**
4	0 1 0 0	**1**
5	0 1 0 1	**1**
6	0 1 1 0	**1**
7	0 1 1 1	**1**
8	1 0 0 0	**1**
9	1 0 0 1	0
10	1 0 1 0	**1**
11	1 0 1 1	0
12	1 1 0 0	**1**
13	1 1 0 1	0
14	1 1 1 0	**1**
15	1 1 1 1	0

Reduction to simplify Combinational Logic Circuits

Grouping the 1s on K-Map (i.e. grouping the minterms of SOP)

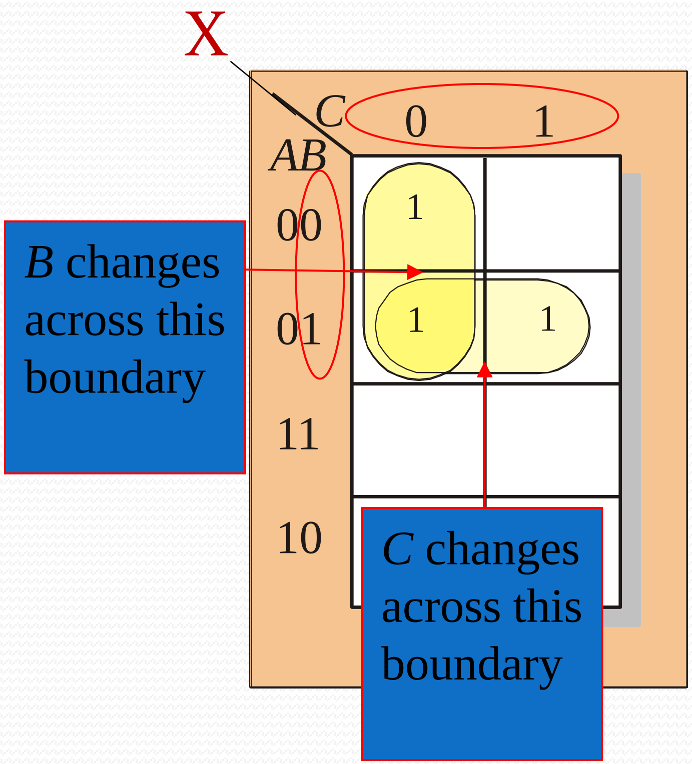
The goal in **simplifying combinational logic** is to maximize the size of the groups and to minimize the number of the groups

- A group must contain either 1, 2, 4, 8, or 16 cells.
- Each cell in a group must be adjacent to one or more cells in that same group.
- Include the largest possible # of 1s in a group in accordance with rule 1
- Each 1 on the map must be included in at least one group.

Karnaugh maps : reduction to simplify

K-maps can simplify combinational logic by grouping cells and eliminating variables that change.

Example Group the 1's on the map and read the minimum logic.



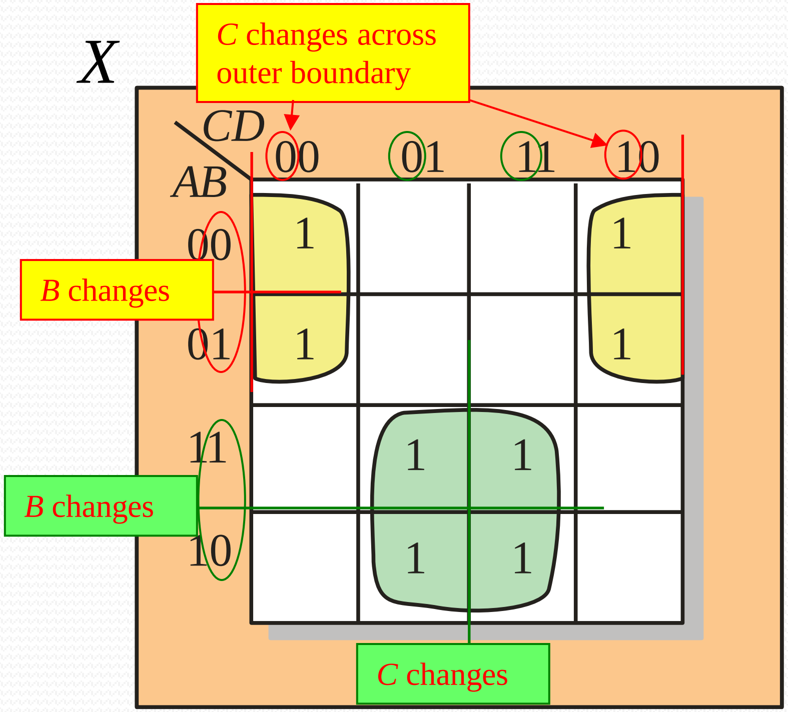
Solution

1. Group the 1's into two overlapping groups as indicated.
2. Read each group by eliminating any variable that changes across a boundary.
3. The vertical group is read $\overline{A}\overline{C}$.
4. The horizontal group is read $\overline{A}B$.

$$X = \overline{A}\overline{C} + \overline{A}B$$

Karnaugh maps

Example Group the 1's on the map and read the minimum logic.



Solution

1. Group the 1's into two separate groups as indicated.
2. Read each group by eliminating any variable that changes across a boundary.
3. The upper (**yellow**) group is read as $\bar{A}\bar{D}$.
4. The lower (**green**) group is read as AD.

$$X = \bar{A}\bar{D} + AD$$