SCR 1013 : Digital Logic

Module 4:
BOOLEAN ALGEBRA & LOGIC SIMPLIFICATION

Laws and Rules of Boolean Algebra

Constructing Truth table from Boolean Expression

Standard Forms of Boolean Expression

Determining standard Expression from truth table

Logic Simplification using:

- Boolean algebra
- Karnaugh Map

Laws & Rules of Boolean Algebra

- Basic laws of Boolean Algebra
  - Commutative Laws
    - $A + B = B + A$
    - $AB = BA$
  - Associative Laws
    - $A + (B + C) = (A + B) + C$
    - $A(BC) = (AB)C$
  - Distributive Laws
    - $A(B + C) = AB + AC$

Rules of Boolean Algebra

1	$A + 0 = A$
2	$A + 1 = 1$
3	$A \bullet 0 = 0$
4	$A \bullet 1 = A$
5	$A + A = A$
6	$A + \overline{A} = 1$
7	$A \bullet A = A$
8	$A \bullet \overline{A} = 0$
9	$\overline{\overline{A}} = A$
10	$A + AB = A$
11	$A + \overline{A}B = A + B$
12	$(A + B)(A + C) = A + BC$

Rules of Boolean Algebra

Lets proof these rules of Boolean Algebra
using basic gates and Laws of Boolean Algebra.

DeMorgan's Theorems …

- **DM theorem 1**:
  - The complement of a product of variables is equal to the sum of the complements of the variables

$$\overline{XY} = \overline{X} + \overline{Y}$$

- **DM theorem 2**:
  - The complement of a sum of variables is equal to the product of the complements of the variables

$$\overline{X + Y} = \overline{X}\,\overline{Y}$$

Standard Form of Boolean Expressions

- Boolean expression can be converted into one of 2 standard forms:
  - **The sum-of-products (SOP) form**
  - **The product-of-sums (POS) form**
- Standardization makes the evaluation, simplification, and implementation of Boolean expressions more systematic and easier.
- **Product term** = a term with the product (Boolean multiplication) of literals
- **Sum term** = a term with the sum (Boolean addition) of literals

Sum-of-Products (SOP) Form

- **SOP** = when 2 or more product terms are summed
- Eg: $AB_{P1} + ABC_{P2}$

  $ABC_{P1} + CDE_{P2} + BCD_{P3}$
- SOP can also contain a single variable term
- In SOP a single overbar cannot extend over more than 1 variable, but more than 1 variable can have an overbar.

$\bar{A}\bar{B}\bar{C}$ ☑ $\overline{ABC}$ ☒

Product-of-Sums (POS) Form

- **POS** = when 2 or more sum terms are multiplied.
  - $(A + B)_{S1}(A + B + C)_{S2}$
  - $(A + B + C)_{S1}(C + D + E)_{S2}(B + C + D)_{S3}$
- Like SOP, POS
  - can also contain a single variable term
  - a single overbar cannot extend over more than 1 variable, but more than 1 variable can have an overbar.

$$\overline{A} + \overline{B} + \overline{C}$$ ☑

$$\overline{A + B + C}$$ ☒

Converting Standard SOP to Standard POS

- Find out the relationship between the two and how to derive the Standard SOP expression from a given standard POS expression, and vice versa.

Karnaugh Map (K-Map)

- K-Map is similar to the truth table, but it **presents all of the possible values of input and output**.
- This is shown in an **array of cells**.
- K-Maps can be used for expressions with **2, 3, 4 or 5** variables.
- The number of cells in a K-Map = total number of possible input variable combinations ➔ 3 = 2^3 = 8
- Cells that differ by only one variable are adjacent
  - Cell 010 is adjacent to 000, 011 and 110
- Physically, cells that share their walls are adjacent
- In a K-map with 4-variable or more, the top-most & bottom-most cells of a column (and row) are adjacent.

AB \ C	0	1
00		
01		
11		
10		

AB \ C	0	1
00	$\overline{A}\overline{B}\overline{C}$	$\overline{A}\overline{B}C$
01	$\overline{A}B\overline{C}$	$\overline{A}BC$
11	$AB\overline{C}$	ABC
10	$A\overline{B}\overline{C}$	$A\overline{B}C$

3-Variable Karnaugh Map

AB \ CD	00	01	11	10
00				
01				
11				
10				

AB \ CD	00	01	11	10
00	$\overline{A}\overline{B}\overline{C}\overline{D}$	$\overline{A}\overline{B}\overline{C}D$	$\overline{A}\overline{B}CD$	$\overline{A}\overline{B}C\overline{D}$
01	$\overline{A}B\overline{C}\overline{D}$	$\overline{A}B\overline{C}D$	$\overline{A}BCD$	$\overline{A}BC\overline{D}$
11	$AB\overline{C}\overline{D}$	$AB\overline{C}D$	$ABCD$	$ABC\overline{D}$
10	$A\overline{B}\overline{C}\overline{D}$	$A\overline{B}\overline{C}D$	$A\overline{B}CD$	$A\overline{B}C\overline{D}$

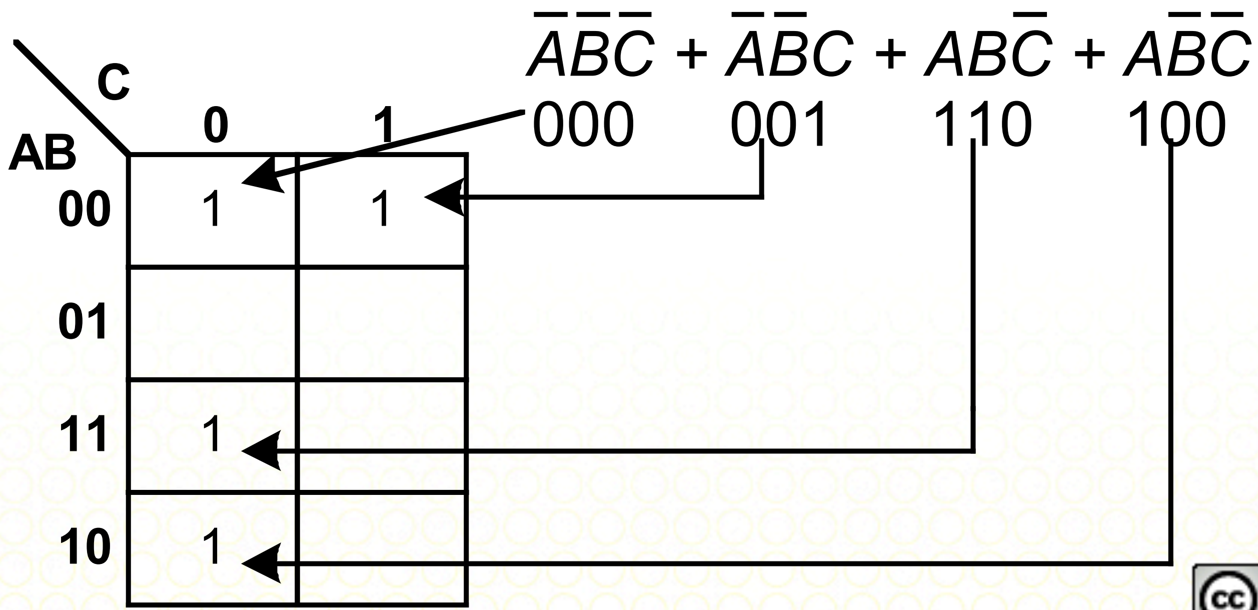
4-Variable Karnaugh Map

K-Map SOP Minimization

- K-Map is used to simplify Boolean expressions to their **minimum form**.
- A minimized SOP expression has the fewest possible term with each term having fewest possible variables.
- A minimized SOP expression needs fewer logic gates than standard expression.
- To map an SOP expression to a map:
  - **Step 1:** determine the binary value of each product term
  - **Step 2**: Place a 1 in a cell that have the same value as the product term

Example: Mapping SOP expression

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



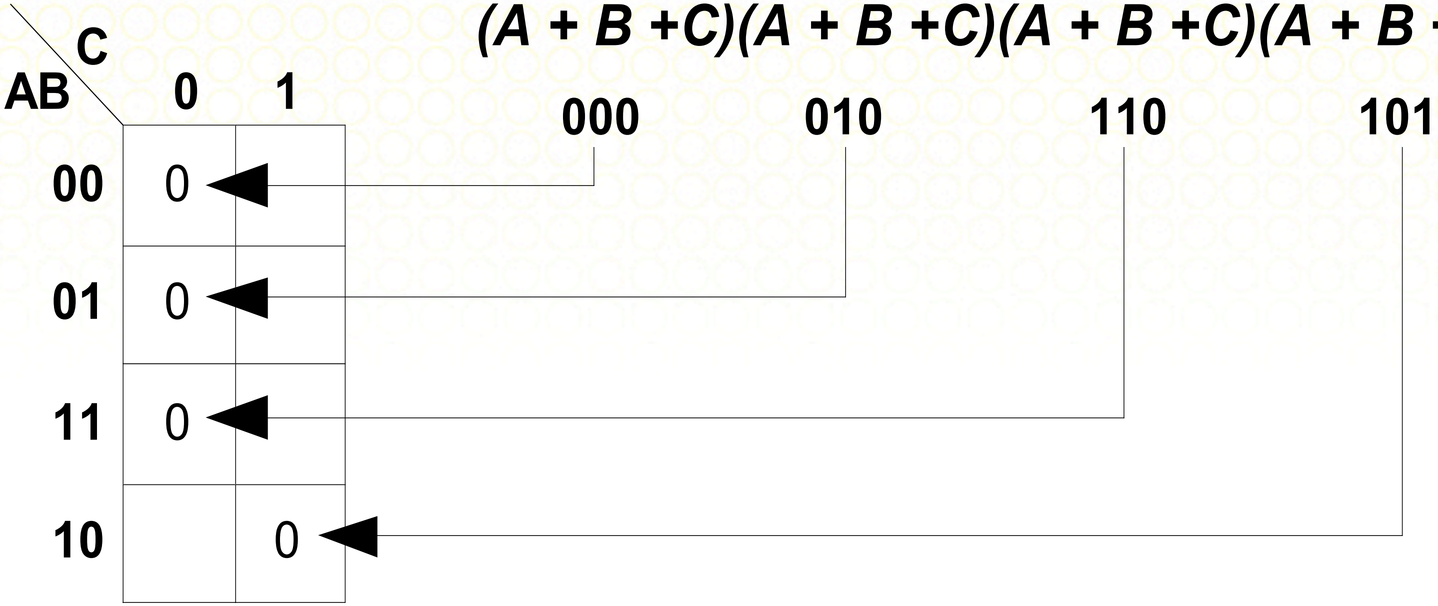
K-Map Simplification of SOP Expressions

- There are 3 steps to obtain a minimum SOP expression from a K-map.
  - Grouping the 1s
  - Determine product term for each group
  - Summing the resulting product terms

K-Map POS Minimization

- For POS expression in standard form, a 0 is put in the K-map for each sum term.
- The methods are similar to SOP minimization, except 0 is used.
- To map a Standard POS expression:
  - Step 1: Determine the binary value of each sum term (i.e. that makes the sum term = 0)
  - Step 2: Check result and place a 0 on the corresponding cell in K-map

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



K-Map Simplification of POS Expressions

- The process is basically the same as with SOP expressions:
  - Group 0s instead of 1s.
  - The rules of grouping 0s are the same as those for 1s
  - Expression must be in Standard POS form.

Converting between POS and SOP using K-Map

- A mapped SOP expression can be converted to an equivalent POS expression.
- This is a good way to compare which can be implemented using fewer gates.
- Given a minimum POS map, the 1s will yield a standard SOP expression.
- This SOP expression can then be minimized by grouping the 1s.

Reducing a Combinational Logic Circuit

- Reducing a combinational logic circuit will result in lesser gates used
- How to do this?
  - Step 1 : Read the logic circuit
  - Step 2 : Get the final output expression
  - Step 3a: Apply De Morgan's theorem and Boolean algebra
  - Step 3b: K-map can be used too.