CHAPTER III

BOOLEAN ALGEBRA
• Boolean algebra is a form of algebra that deals with single digit binary values and variables.

• Values and variables can indicate some of the following binary pairs of values:
  • ON / OFF
  • TRUE / FALSE
  • HIGH / LOW
  • CLOSED / OPEN
  • 1 / 0
Three fundamental operators in Boolean algebra

- **NOT**: unary operator that complements represented as $\bar{A}$, $A'$, or $\sim A$
- **AND**: binary operator which performs logical multiplication
  - i.e. $A$ ANDed with $B$ would be represented as $AB$ or $A \cdot B$
- **OR**: binary operator which performs logical addition
  - i.e. $A$ ORed with $B$ would be represented as $A + B$

<table>
<thead>
<tr>
<th>NOT</th>
<th>AND</th>
<th>OR</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>$\bar{A}$</td>
<td>$A$</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Below is a table showing all possible Boolean functions $F_N$ given the two-inputs $A$ and $B$.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>F_0</th>
<th>F_1</th>
<th>F_2</th>
<th>F_3</th>
<th>F_4</th>
<th>F_5</th>
<th>F_6</th>
<th>F_7</th>
<th>F_8</th>
<th>F_9</th>
<th>F_10</th>
<th>F_11</th>
<th>F_12</th>
<th>F_13</th>
<th>F_14</th>
<th>F_15</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

- **Null**
- **Inhibition**
- **A**
- **B**
- **A + B**
- **B**
- **A**
- **AB**
- **Identity**
Boolean expressions must be evaluated with the following order of operator precedence:

- parentheses
- NOT
- AND
- OR

Example:

\[ F = (A(C + BD) + B\overline{C})\overline{E} \]

\[ F = \left( A \left[ C + \overline{B}D \right] + B\overline{C} \right) \overline{E} \]
Example 1:
Evaluate the following expression when \( A = 1 \), \( B = 0 \), \( C = 1 \)
\[
F = C + \overline{CB} + B\overline{A}
\]
Solution
\[
F = 1 + \overline{1} \cdot 0 + 0 \cdot \overline{1} = 1 + 0 + 0 = 1
\]
Example 2:
Evaluate the following expression when \( A = 0 \), \( B = 0 \), \( C = 1 \), \( D = 1 \)
\[
F = D(\overline{B}\overline{C}A + (A\overline{B} + C) + C)
\]
Solution
\[
F = 1 \cdot (0 \cdot \overline{1} \cdot 0 + (0 \cdot \overline{0} + 1) + 1) = 1 \cdot (0 + \overline{1} + 1) = 1 \cdot 1 = 1
\]
### Basic Identities

<table>
<thead>
<tr>
<th>Identity</th>
<th>Formula</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Identity</strong></td>
<td>( X + 0 = X )</td>
<td>( X \cdot 1 = X )</td>
</tr>
<tr>
<td>( X + 1 = 1 )</td>
<td>( X \cdot 0 = 0 )</td>
<td>( X \cdot X = X )</td>
</tr>
<tr>
<td>( X + X = X )</td>
<td>( (X')' = X )</td>
<td><strong>Idempotent Law</strong></td>
</tr>
<tr>
<td>( X + X' = 1 )</td>
<td>( X + Y = Y + X )</td>
<td><strong>Involution Law</strong></td>
</tr>
<tr>
<td>( X + (Y + Z) = (X + Y) + Z )</td>
<td>( X(Y + Z) = XY + XZ )</td>
<td><strong>Associativity</strong></td>
</tr>
<tr>
<td>( X(Y + Z) = XY + XZ )</td>
<td>( X + YZ = (X + Y)(X + Z) )</td>
<td><strong>Absorption Law</strong></td>
</tr>
<tr>
<td>( X + XY = X )</td>
<td>( X(X + Y) = X )</td>
<td><strong>DeMorgan’s Law</strong></td>
</tr>
<tr>
<td>( X + X'Y = X + Y )</td>
<td>( X(X' + Y) = XY )</td>
<td></td>
</tr>
</tbody>
</table>
• **Duality principle:**
  • States that a Boolean equation remains valid if we take the dual of the expressions on both sides of the equals sign.
  • The dual can be found by interchanging the **AND** and **OR** operators along with also interchanging the 0’s and 1’s.
  • This is evident with the duals in the basic identities.
    • For instance: DeMorgan’s Law can be expressed in two forms
      
      $$(X + Y)’ = X’Y’ \quad \text{as well as} \quad (XY)’ = X’ + Y’$$
Example: Simplify the following expression

\[ F = BC + B\overline{C} + BA \]

Simplification

\[ F = B(C + \overline{C}) + BA \]
\[ F = B \cdot 1 + BA \]
\[ F = B(1 + A) \]
\[ F = B \]
Example: Simplify the following expression

\[ F = A + \overline{A}B + \overline{A}\overline{B}C + \overline{A}\overline{B}\overline{C}D + \overline{A}\overline{B}\overline{C}\overline{D}\overline{E} \]

Simplification

\[ F = A + \overline{A}(B + BC + BCD + BCDE) \]
\[ F = A + B + BG + BCD + BCDE \]
\[ F = A + B + \overline{B}(C + \overline{C}D + \overline{C}DE) \]
\[ F = A + B + C + \overline{C}D + \overline{C}DE \]
\[ F = A + B + C + \overline{C}(D + \overline{D}E) \]
\[ F = A + B + C + D + \overline{D}E \]
\[ F = A + B + C + D + E \]
Example: Show that the following equality holds

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

Simplification

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

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

$$= \overline{A} + (B + C)(\overline{B} + \overline{C})$$
Boolean expressions can be manipulated into many forms. Some standardized forms are required for Boolean expressions to simplify communication of the expressions.

- **Sum-of-products (SOP)**
  - Example:
    \[
    F(A, B, C, D) = AB + \overline{BCD} + AD
    \]

- **Products-of-sums (POS)**
  - Example:
    \[
    F(A, B, C, D) = (A + B)(\overline{B} + C + \overline{D})(A + D)
    \]
The following table gives the minterms for a three-input system:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>( m_0 )</th>
<th>( m_1 )</th>
<th>( m_2 )</th>
<th>( m_3 )</th>
<th>( m_4 )</th>
<th>( m_5 )</th>
<th>( m_6 )</th>
<th>( m_7 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
**SUM OF MINTERMS**

- **Sum-of-minterms** standard form expresses the Boolean or switching expression in the form of a *sum of products* using *minterms*.

- For instance, the following Boolean expression using minterms

  \[
  F(A, B, C) = \overline{A} \overline{B} C + \overline{A} B \overline{C} + AB \overline{C} + A \overline{B} C
  \]

  could instead be expressed as

  \[
  F(A, B, C) = m_0 + m_1 + m_4 + m_5
  \]

  or more compactly

  \[
  F(A, B, C) = \sum m(0, 1, 4, 5) = \text{one-set}(0, 1, 4, 5)
  \]
The following table gives the maxterms for a **three-input** system:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>M₀</th>
<th>M₁</th>
<th>M₂</th>
<th>M₃</th>
<th>M₄</th>
<th>M₅</th>
<th>M₆</th>
<th>M₇</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

R.M. Dansereau; v.1.0
**Product-of-maxterms** standard form expresses the Boolean or switching expression in the form of **product of sums** using **maxterms**.

For instance, the following Boolean expression using maxterms

\[
F(A, B, C) = (A + B + \overline{C})(\overline{A} + B + C)(\overline{A} + \overline{B} + \overline{C})
\]

could instead be expressed as

\[
F(A, B, C) = M_1 \cdot M_4 \cdot M_7
\]

or more compactly as

\[
F(A, B, C) = \prod M(1, 4, 7) = \text{zero-set}(1, 4, 7)
\]
STANDARD FORMS
MINTERM AND MAXTERM EXP.

- SUM OF MINTERMS
- MAXTERMS
- PRODUCT OF MAXTERMS

Given an arbitrary Boolean function, such as

\[ F(A, B, C) = AB + \overline{B}(\overline{A} + \overline{C}) \]

how do we form the canonical form for:

- **sum-of-minterms**
  - Expand the Boolean function into a sum of products. Then take each term with a missing variable \( X \) and **AND** it with \( X + \overline{X} \).

- **product-of-maxterms**
  - Expand the Boolean function into a product of sums. Then take each factor with a missing variable \( X \) and **OR** it with \( X \overline{X} \).
Example

\[ F(A, B, C) = AB + B(\bar{A} + \bar{C}) = AB + \bar{A}\bar{B} + \bar{B}C \]
\[ = AB(C + \bar{C}) + \bar{A}\bar{B}(C + \bar{C}) + (A + \bar{A})\bar{B}C \]
\[ = \overline{A}\overline{B}\overline{C} + \overline{A}\overline{B}C + \overline{A}BC + \overline{A}C + ABC \]
\[ = \sum m(0, 1, 4, 6, 7) \]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Minterms listed as 1s in Truth Table
Example

\[ F(A, B, C) = AB + \overline{B}(\overline{A} + \overline{C}) = AB + \overline{A}B + \overline{B}C \]

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

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

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

\[ = \prod M(2, 3, 5) \]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0 [\rightarrow 2]</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0 [\rightarrow 3]</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0 [\rightarrow 5]</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Maxterms listed as 0s in Truth Table
- Converting between sum-of-minterms and product-of-maxterms
  - The two are complementary, as seen by the truth tables.
  - To convert interchange the $\sum$ and $\prod$, then use missing terms.
    - Example: The example from the previous slides
      \[
      F(A, B, C) = \sum m(0, 1, 4, 6, 7)
      \]
      is re-expressed as
      \[
      F(A, B, C) = \prod M(2, 3, 5)
      \]
      where the numbers 2, 3, and 5 were missing from the minterm representation.
Often it is desired to simplify a Boolean function. A quick graphical approach is to use Karnaugh maps.

### 2-variable Karnaugh map

<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

\[ F = AB \]

### 3-variable Karnaugh map

<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ F = AB + C \]

### 4-variable Karnaugh map

<table>
<thead>
<tr>
<th></th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ F = AB + \overline{CD} \]
• Notice that the ordering of cells in the map are such that moving from one cell to an adjacent cell only changes one variable.

2-variable Karnaugh map

3-variable Karnaugh map

4-variable Karnaugh map

• This ordering allows for grouping of minterms/maxterms for simplification.
### Implicants

- **Implicant**
  - Bubble covering only 1s (size of bubble must be a power of 2).

- **Prime implicant**
  - Bubble that is expanded as big as possible (but increases in size by powers of 2).

- **Essential prime implicant**
  - Bubble that contains a 1 covered only by itself and no other prime implicant bubble.

- **Non-essential prime implicant**
  - A 1 that can be bubbled by more than one prime implicant bubble.

#### Karnaugh Map

<table>
<thead>
<tr>
<th>CD</th>
<th>AB</th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>00</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>01</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
• Procedure for finding the SOP from a Karnaugh map
  • Step 1: Form the 2-, 3-, or 4-variable Karnaugh map as appropriate for the Boolean function.
  • Step 2: Identify all essential prime implicants for 1s in the Karnaugh map.
  • Step 3: Identify non-essential prime implicants for 1s in the Karnaugh map.
  • Step 4: For each essential and one selected non-essential prime implicant from each set, determine the corresponding product term.
  • Step 5: Form a sum-of-products with all product terms from previous step.
### Simplification Example for SOP (1)

**Example:**

Simplify the following Boolean function

\[
F(A, B, C) = \sum m(0, 1, 4, 5) = \overline{A}\overline{B}C + \overline{A}B\overline{C} + AB\overline{C} + ABC
\]

**Solution:**

- The essential prime implicants are \( \overline{B} \).
- There are no non-essential prime implicants.
- The sum-of-products solution is \( F = \overline{B} \).
• Simplify the following Boolean function

\[ F(A, B, C) = \sum m(0, 1, 4, 6, 7) = \overline{A}\overline{B}C + \overline{A}\overline{B}C + \overline{A}\overline{B}C + \overline{A}BC + ABC \]

• Solution:

\[
\begin{array}{c|cc|c|c}
A & 00 & 01 & 11 & 10 \\
\hline
0 & 1 & 1 & 0 & 0 \\
1 & 1 & 0 & 1 & 1 \\
\end{array}
\]

zero-set(2, 3, 5)

one-set(0, 1, 4, 6, 7)

• The essential prime implicants are \( \overline{A}B \) and \( AB \).

• The non-essential prime implicants are \( BC \) or \( AC \).

• The sum-of-products solution is

\[ F = AB + \overline{A}B + BC \text{ or } F = AB + \overline{A}B + AC. \]
- Procedure for finding the SOP from a Karnaugh map
  - Step 1: Form the 2-, 3-, or 4-variable Karnaugh map as appropriate for the Boolean function.
  - Step 2: Identify all essential prime implicants for 0s in the Karnaugh map.
  - Step 3: Identify non-essential prime implicants for 0s in the Karnaugh map.
  - Step 4: For each essential and one selected non-essential prime implicant from each set, determine the corresponding sum term.
  - Step 5: Form a product-of-sums with all sum terms from previous step.
SIMPLIFICATION
EXAMPLE FOR POS (1)

• Simplify the following Boolean function

\[ F(A, B, C) = \prod M(2, 3, 5) = (A + \overline{B} + C)(A + \overline{B} + \overline{C})(\overline{A} + B + \overline{C}) \]

• Solution:

<table>
<thead>
<tr>
<th>BC</th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>00</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

- zero-set(2, 3, 5)
- one-set(0, 1, 4, 6, 7)

• The essential prime implicants are \( \overline{A} + B + \overline{C} \) and \( A + \overline{B} \).
• There are no non-essential prime implicants.
• The product-of-sums solution is \( F = (A + \overline{B})(\overline{A} + B + \overline{C}) \).
Simplify the following Boolean function: \( F(A, B, C) = \Pi M(0, 1, 5, 7, 8, 9, 15) \)

Solution:

The essential prime implicants are \( B + C \) and \( B + C \).

The non-essential prime implicants can be \( \overline{A} + \overline{B} \) or \( \overline{A} + \overline{C} + \overline{D} \).

The product-of-sums solution can be either

\[
F = (B + C)(B + \overline{C})(A + \overline{B} + D)
\]

or

\[
F = (B + C)(B + \overline{C})(A + B + D)
\]
Switching expressions are sometimes given as incomplete, or with don’t-care conditions.

- Having don’t-care conditions can simplify Boolean expressions and hence simplify the circuit implementation.
- Along with the zero-set( ) and one-set( ), we will also have dc( ).
- Don’t-cares conditions in Karnaugh maps
  - Don’t-cares will be expressed as an “X” or “-” in Karnaugh maps.
  - Don’t-cares can be bubbled along with the 1s or 0s depending on what is more convenient and help simplify the resulting expressions.
• Find the SOP simplification for the following Karnaugh map

\[
\begin{array}{c|c|c|c|c}
    & CD & 00 & 01 & 11 & 10 \\
\hline
AB & 00 & 0 & 0 & 1 & 1 \\
    & 01 & 1 & 0 & 0 & 1 \\
    & 11 & 1 & X & 0 & X \\
    & 10 & 0 & 0 & 1 & X \\
\end{array}
\]

taken to be 0

taken to be 1

\[
\text{zero-set(0, 1, 5, 7, 8, 9, 15)} \\
\text{one-set(2, 3, 4, 6, 11, 12)} \\
\text{dc(10, 13, 14)}
\]

• Solution:

• The essential prime implicants are \( B\overline{D} \) and \( \overline{B}C \).
• There are no non-essential prime implicants.
• The sum-of-products solution is \( F = \overline{B}C + B\overline{D} \).
• Find the POS simplification for the following Karnaugh map

\[
\begin{array}{cccc}
\text{AB} & 00 & 01 & 11 & 10 \\
00 & 0 & 0 & 1 & 1 \\
01 & 1 & 0 & 0 & 1 \\
11 & 1 & X & 0 & X \\
10 & 0 & 0 & 1 & X \\
\end{array}
\]

zero-set(0, 1, 5, 7, 8, 9, 15)
one-set(2, 3, 4, 6, 11, 12)
dc(10, 13, 14)

• Solution:

- The essential prime implicants are \( B + C \) and \( \overline{B} + \overline{D} \).
- There are no non-essential prime implicants.
- The product-of-sums solution is \( F = (B + C)(\overline{B} + \overline{D}) \).