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></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></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>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>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>
<td>0</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>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</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>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

- **Null**
- **Inhibition**
- **Implication**
- **Identity**

**Binary Boolean Operators**

- **Null Identity**
- **Inhibition**
- **Implication**
- **Identity**
Boolean expressions must be evaluated with the following order of operator precedence:

- Parentheses
- NOT
- AND
- OR

Example:

\[ F = (A(C + BD) + BC)E \]
- Example 1:

Evaluate the following expression when $A = 1$, $B = 0$, $C = 1$

$$F = C + \overline{CB} + B\overline{A}$$

- Solution

$$F = 1 + 1 \cdot 0 + 0 \cdot 1 = 1 + 0 + 0 = 1$$

- Example 2:

Evaluate the following expression when $A = 0$, $B = 0$, $C = 1$, $D = 1$

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

- Solution

$$F = 1 \cdot (0 \cdot 1 \cdot 0 + (0 \cdot 0 + 1) + 1) = 1 \cdot (0 + 1 + 1) = 1 \cdot 1 = 1$$
### Basic Identities

<table>
<thead>
<tr>
<th>Identity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X + 0 = X$</td>
<td>Identity</td>
</tr>
<tr>
<td>$X + 1 = 1$</td>
<td>Idempotent Law</td>
</tr>
<tr>
<td>$X + X = X$</td>
<td>Complement</td>
</tr>
<tr>
<td>$X + X' = 1$</td>
<td>Involution Law</td>
</tr>
<tr>
<td>$(X')' = X$</td>
<td></td>
</tr>
<tr>
<td>$X + Y = Y + X$</td>
<td>Commutativity</td>
</tr>
<tr>
<td>$X + (Y + Z) = (X + Y) + Z$</td>
<td>Associativity</td>
</tr>
<tr>
<td>$X(Y + Z) = XY + XZ$</td>
<td>Distributivity</td>
</tr>
<tr>
<td>$X + XY = X$</td>
<td>Absorption Law</td>
</tr>
<tr>
<td>$X + X'Y = X + Y$</td>
<td>Simplification</td>
</tr>
<tr>
<td>$(X + Y)' = X'Y'$</td>
<td>DeMorgan’s Law</td>
</tr>
<tr>
<td>$XY + X'Z + YZ = XY + X'Z$</td>
<td>Consensus Theorem</td>
</tr>
</tbody>
</table>

R.M. Dansereau; v.1.0
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
    
    
    \[(X + Y)' = X'Y'\]  \[as \ well \ as \ \ (XY)' = X' + Y'\]
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

\[ F = A + \overline{A}B + \overline{A}BC + \overline{A}BCD + \overline{A}BCDE \]

• Simplification

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

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

Simplification

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

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

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

- **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 B 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 0 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 0 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 1 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 1 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 0 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 0 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>
</tr>
<tr>
<td>1 1 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>
</tr>
<tr>
<td>1 1 1</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>
</tr>
</tbody>
</table>
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

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

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></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>A</td>
<td>B</td>
<td>C</td>
<td>A + B + C</td>
<td>A + B + C</td>
<td>A + B + C</td>
<td>A + B + C</td>
<td>M1</td>
<td>M2</td>
</tr>
<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>
</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>
</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>
</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>
</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>
</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>
</tr>
<tr>
<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>0</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>
</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

\[ F(A, B, C) = \prod M(1, 4, 7) = \text{zero-set}(1, 4, 7) \]
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

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

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

\[ = \overline{A}BC + \overline{ABC} + ABC + AB\overline{C} + ABC \]

\[ = \sum m(0, 1, 4, 6, 7) \]

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1 [ \leftarrow 0 ]</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1 [ \leftarrow 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 [ \leftarrow 4 ]</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 [ \leftarrow 6 ]</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1 [ \leftarrow 7 ]</td>
</tr>
</tbody>
</table>

Minterms listed as 1s in Truth Table
Example

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

\[ = (A + \bar{B})(A + \bar{B} + \bar{C})(\bar{A} + B + \bar{C}) \quad \text{(using distributivity)} \]

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

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

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

<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 ← 2</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0 ← 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 ← 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 are sums, so this is called a "product of sums" (POS)

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

$$F(A, B, C) = \sum m(0, 1, 4, 6, 7)$$

$$F'(A, B, C) = \text{SUM } m(2, 3, 5)$$

is re-expressed as

$$F(A, B, C) = \prod M(2, 3, 5)$$

$$F'(A, B, C) = \text{PI M}(0, 1, 4, 6, 7)$$

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

F = AB

### 3-variable Karnaugh map

<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>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

F = AB + C

### 4-variable Karnaugh map

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

F = AB + 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.

This ordering allows for grouping of minterms/maxterms for simplification.
• 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.
• **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)**

**SOP: SUM OF PRODUCTS**

- **SIMPLIFICATION**
  - KARNAUGH MAP ORDER
  - IMPLICANTS
  - PROCEDURE FOR SOP

---

• Simplify the following Boolean function

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

• Solution:

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

- The essential prime implicants are \( \overline{B} \).
- There are no non-essential prime implicants.
- The sum-of-products solution is \( F = \overline{B} \).
SIMPLIFICATION
EXAMPLE FOR SOP (2)

SOP: SUM OF PRODUCTS

• Simplify the following Boolean function

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

• Solution:

\[ \begin{array}{c|cccc}
A & B & C & 00 & 01 \\
\hline
0 & 0 & 1 & 0 & 0 \\
1 & 1 & 0 & 1 & 1 \\
\end{array} \]

- The essential prime implicants are \( \overline{A}B \) and \( AB \).
- The non-essential prime implicants are \( \overline{B}C \) or \( A\overline{C} \).
- The sum-of-products solution is

\[ F = AB + \overline{A}B + \overline{B}C \text{ or } F = AB + \overline{A}B + A\overline{C}. \]
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.
Example for POS (1)

**POS: Product of Sums**

- **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>( \overline{A} \overline{B} C )</th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>A ( \overline{A} \overline{B} C )</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

- 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
\[ F(A, B, C) = \prod M(0, 1, 5, 7, 8, 9, 15) \]

**Solution:**

- The essential prime implicants are \( B + C \) and \( \overline{B} + \overline{C} + \overline{D} \).
- The non-essential prime implicants can be \( A + \overline{B} + \overline{D} \) or \( A + C + \overline{D} \).
- The product-of-sums solution can be either
  \[ F = (B + C)(\overline{B} + \overline{C} + \overline{D})(A + \overline{B} + \overline{D}) \]
  or
  \[ F = (B + C)(\overline{B} + \overline{C} + \overline{D})(A + C + \overline{D}) \]
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

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

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

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 \( F = (B + C)(\overline{B} + \overline{D}) \).

R.M. Dansereau; v.1.0