CHAPTER OBJECTIVES

- Learn Binary Logic and BOOLEAN Algebra
- Learn How to Map a Boolean Expression into Logic Circuit Implementation
- Learn How To Manipulate Boolean Expressions and Simplify Them

Lesson Objectives

1. Learn how to derive a Boolean expression of a function defined by its truth table. The derived expressions may be in one of two possible standard forms: The Sum of Min-terms or the Product of Max-Terms.
2. Learn how to map these expressions into logic circuit implementations (2-Level Implementations).

MinTerms

- Consider a system of 3 input signals (variables) x, y, & z.
- A term which ANDs all input variables, either in the true or complement form, is called a minterm.
- Thus, the considered 3-input system has 8 minterms, namely:
  \( \bar{x}y\bar{z}, \bar{x}yz, \bar{x}yz, \bar{x}\bar{y}z, x\bar{y}\bar{z}, x\bar{y}z, xy\bar{z}, xyz \) & \( x\bar{y}z \)

- Each minterm equals 1 at exactly one particular input combination and is equal to 0 at all other combinations.
- Thus, for example, \( \bar{x}y\bar{z} \) is always equal to 0 except for the input combination \( xyz = 000 \), where it is equal to 1.
- Accordingly, the minterm \( x\bar{y}\bar{z} \) is referred to as \( m_0 \).
- In general, minterms are designated \( m_i \), where \( i \) corresponds the input combination at which this minterm is equal to 1.
For the 3-input system under consideration, the number of possible input combinations is 2^3, or 8. This means that the system has a total of 8 minterms as follows:

- \( m_0 = \overline{x} \overline{y} \overline{z} = 1 \) IFF \( xyz = 000 \), otherwise it equals 0
- \( m_1 = \overline{x}yz = 1 \) IFF \( xyz = 001 \), otherwise it equals 0
- \( m_2 = \overline{xy}z = 1 \) IFF \( xyz = 010 \), otherwise it equals 0
- \( m_3 = \overline{x}yz = 1 \) IFF \( xyz = 011 \), otherwise it equals 0
- \( m_4 = xy \overline{z} = 1 \) IFF \( xyz = 100 \), otherwise it equals 0
- \( m_5 = x \overline{yz} = 1 \) IFF \( xyz = 101 \), otherwise it equals 0
- \( m_6 = x \overline{y}z = 1 \) IFF \( xyz = 110 \), otherwise it equals 0
- \( m_7 = xyz = 1 \) IFF \( xyz = 111 \), otherwise it equals 0

In general,
- For \( n \)-input variables, the number of minterms = the total number of possible input combinations = \( 2^n \).
- A minterm = 0 at all input combinations except one where the minterm = 1.

MaxTerms
- Consider a circuit of 3 input signals (variables) \( x, y, \& z \).
- A term which ORs all input variables, either in the true or complement form, is called a Maxterm.
- With 3-input variables, the system under consideration has a total of 8 Maxterms, namely:

\[
(x + y + z), (x + y + \overline{z}), (x + \overline{y} + z), (x + \overline{y} + \overline{z}), (\overline{x} + y + z), (\overline{x} + \overline{y} + z), (\overline{x} + \overline{y} + \overline{z}), (x + \overline{y} + \overline{z})
\]

- Each Maxterm equals 0 at exactly one of the 8 possible input combinations and is equal to 1 at all other combinations.
- For example, \( (x + y + z) \) equals 1 at all input combinations except for the combination \( xyz = 000 \), where it is equal to 0.
- Accordingly, the Maxterm \( (x + y + z) \) is referred to as \( M_0 \).
- In general, Maxterms are designated \( M_i \), where \( i \) corresponds to the input combination at which this Maxterm is equal to 0.
For the 3-input system, the number of possible input combinations is $2^3$, or 8.

This means that the system has a total of 8 Maxterms as follows:

- $M_0 = (x + y + z) = 0$ IFF $xyz = 000$, otherwise it equals 1
- $M_1 = (x + y + \bar{z}) = 0$ IFF $xyz = 001$, otherwise it equals 1
- $M_2 = (x + \bar{y} + z) = 0$ IFF $xyz = 010$, otherwise it equals 1
- $M_3 = (x + y + \bar{z}) = 0$ IFF $xyz = 011$, otherwise it equals 1
- $M_4 = (\bar{x} + y + z) = 0$ IFF $xyz = 100$, otherwise it equals 1
- $M_5 = (\bar{x} + y + \bar{z}) = 0$ IFF $xyz = 101$, otherwise it equals 1
- $M_6 = (\bar{x} + \bar{y} + z) = 0$ IFF $xyz = 110$, otherwise it equals 1
- $M_7 = (\bar{x} + \bar{y} + \bar{z}) = 0$ IFF $xyz = 111$, otherwise it equals 1

In general,

- For $n$-input variables, the number of Maxterms = the total number of possible input combinations = $2^n$.
- A Maxterm = 1 at all input combinations except one where the Maxterm = 0.

Important Result

Using De-Morgan’s theorem, or truth tables, it can be easily shown that:

$$M_i = \overline{m_i} \quad \forall i = 0, 1, 2, \ldots, (2^n - 1)$$

Expressing Functions as a Sum of Minterms and Product of Maxterms

Example: Consider the function $F$ defined by the shown truth table

Now let’s rewrite the table, with few added columns.

- A column $i$ indicating the input combination
- Four columns of minterms $m_2$, $m_4$, $m_5$ and $m_7$
- One last column OR-ing the above minterms ($m_2 + m_4 + m_5 + m_7$)
From this table, we can clearly see that \( F = m_2 + m_4 + m_5 + m_7 \)

This is logical since \( F = 1 \), only at input combinations \( i= 2, 4, 5 \) and 7

Thus, by ORing minterm \( m_2 \) (which has a value of 1 only at input combination \( i= 2 \)) with minterm \( m_4 \) (which has a value of 1 only at input combination \( i= 4 \)) with minterm \( m_5 \) (which has a value of 1 only at input combination \( i= 5 \)) with minterm \( m_7 \) (which has a value of 1 only at input combination \( i= 7 \)) the resulting function will equal \( F \).

In general, Any function can be expressed by OR-ing all minterms \( (m_i) \) corresponding to input combinations \( (i) \) at which the function has a value of 1.

The resulting expression is commonly referred to as the SUM of minterms and is typically expressed as \( F = \sum(2, 4, 5, 7) \), where \( \sum \) indicates OR-ing of the indicated minterms. Thus, \( F = \sum(2, 4, 5, 7) = (m_2 + m_4 + m_5 + m_7) \)

**Example:**

Consider the function \( F \) of the previous example.

We will, first, derive the sum of minterms expression for the complement function \( F' \).

The truth table of \( F' \) shows that \( F' \) equals 1 at \( i = 0, 1, 3 \) and 6, then,

\[
F' = m_0 + m_1 + m_3 + m_6, \text{ i.e.}
\]

\[
F' = \sum(0, 1, 3, 6), \quad (1)
\]

\[
F = \sum(2, 4, 5, 7) \quad (2)
\]

- Obviously, the sum of minterms expression of \( F' \) contains all minterms that do not appear in the sum of minterms expression of \( F \).
Using De-Morgan theorem on equation (2),

\[ F' = \overline{m_2 + m_4 + m_5 + m_7} = m_2 \cdot \overline{m_4} \cdot \overline{m_5} \cdot \overline{m_7} = M_2 \cdot M_4 \cdot M_5 \cdot M_7 \]

This form is designated as the \textit{Product of Maxterms} and is expressed using the \( \prod \) symbol, which is used to designate product in regular algebra, but is used to designate AND-ing in Boolean algebra.

Thus,

\[ F' = \prod (2, 4, 5, 7) = M_2 \cdot M_4 \cdot M_5 \cdot M_7 \]  \hspace{1cm} (3)

From equations (1) and (3) we get,

\[ F' = \sum (0, 1, 3, 6) = \prod (2, 4, 5, 7) \]

In general, \textit{any function can be expressed both as a sum of minterms and as a product of maxterms}. Consider the derivation of \( F \) back from \( F' \) given in equation (3):

\[ F = \overline{F} = m_0 + m_1 + m_3 + m_6 = m_0 \cdot \overline{m_1} \cdot \overline{m_3} \cdot \overline{m_6} = M_0 \cdot M_1 \cdot M_3 \cdot M_6 \]

\[ F' = \sum (2, 4, 5, 7) = \prod (0, 1, 3, 6) \]

\[ F = \prod (2, 4, 5, 7) = \sum (0, 1, 3, 6) \]

\textbf{Conclusions:}

- Any function can be expressed both as a sum of minterms (\( \sum m_i \)) and as a product of maxterms. The product of maxterms expression (\( \prod M_j \)) expression of \( F \) contains \textit{all} maxterms \( M_j \) (\( \forall j \neq i \)) that do not appear in the sum of minterms expression of \( F \).
- The sum of minterms expression of \( F' \) contains \textit{all} minterms that do not appear in the sum of minterms expression of \( F \).
- This is true for all complementary functions. Thus, each of the \( 2^n \) minterms will appear either in the sum of minterms expression of \( F \) or the sum of minterms expression of \( \overline{F} \) but not both.
- The product of maxterms expression of \( F' \) contains \textit{all} maxterms that do not appear in the product of maxterms expression of \( F \).
- This is true for all complementary functions. Thus, each of the \( 2^n \) maxterms will appear either in the product of maxterms expression of \( F \) or the product of maxterms expression of \( \overline{F} \) but not both.
**Example:**

Given that $F(a, b, c, d) = \sum(0, 1, 2, 4, 5, 7)$, derive the product of maxterms expression of $F$ and the 2 standard form expressions of $F'$.

Since the system has 4 input variables (a, b, c & d) $\rightarrow$ The number of minterms and Maxterms $= 2^4 = 16$

$F(a, b, c, d) = \sum(0, 1, 2, 4, 5, 7)$

1. List all maxterms in the Product of maxterms expression

$F = \prod (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15)$

2. Cross out maxterms corresponding to input combinations of the minterms appearing in the sum of minterms expression

$F = \prod (3, 6, 8, 9, 10, 11, 12, 13, 14, 15)$

Similarly, obtain both canonical form expressions for $F'$

$F' = \sum (3, 6, 8, 9, 10, 11, 12, 13, 14, 15)$

$F' = \prod (0, 1, 2, 4, 5, 7)$
Canonical Forms:
The sum of minterms and the product of maxterms forms of Boolean expressions are known as the canonical forms (عندلی یا تکاملی) of a function.

Standard Forms:
- A product term is a term with ANDed literals*. Thus, AB, A’B, A’CD are all product terms.
- A minterm is a special case of a product term where all input variables appear in the product term either in the true or complement form.
- A sum term is a term with ORed literals*. Thus, (A+B), (A’+B), (A’+C+D) are all sum terms.
- A maxterm is a special case of a sum term where all input variables, either in the true or complement form, are ORed together.
- Boolean functions can generally be expressed in the form of a Sum of Products (SOP) or in the form of a Product of Sums (POS).
- The sum of minterms form is a special case of the SOP form where all product terms are minterms.
- The product of maxterms form is a special case of the POS form where all sum terms are maxterms.
- The SOP and POS forms are Standard forms for representing Boolean functions.

* A Boolean variable in the true or complement forms
Two-Level Implementations of Standard Forms

**Sum of Products Expression (SOP):**

- Any SOP expression can be implemented in 2-levels of gates.
- The **first level** consists of a number of **AND gates** which equals the number of product terms in the expression. Each AND gate implements one of the product terms in the expression.
- The **second level** consists of a **SINGLE OR gate** whose number of inputs equals the number of product terms in the expression.

**Example** Implement the following SOP function

\[ F = XZ + Y\overline{Z} + X\overline{Y}Z \]

Two-Level Implementation \((F = XZ + Y\overline{Z} + X\overline{Y}Z)\)

**Level-1:** AND-Gates ; **Level-2:** One OR-Gate
**Product of Sums Expression (POS):**

- Any POS expression can be implemented in 2-levels of gates
- The **first level** consists of a number of **OR** gates which equals the number of sum terms in the expression, each gate implements one of the sum terms in the expression.
- The **second level** consists of a **SINGLE AND** gate whose number of inputs equals the number of sum terms.

**Example** Implement the following SOP function

\[ F = (X+Z)(Y'+Z)(X'+Y+Z) \]

**Two-Level Implementation** \( \{F = (X+Z)(Y'+Z)(X'+Y+Z)\} \)

**Level-1:** OR-Gates ; **Level-2:** One AND-Gate