Boolean Operator Precedence

The order of evaluation in Boolean Expressions is:
1. Parentheses
2. NOT
3. AND
4. OR

Note: Because of the fact that AND takes precedence over OR, parentheses must be placed around the OR operator more frequently.
Review: Duality Principle

Every algebraic expression deducible from the postulates of Boolean Algebra remains valid if the operators and the identity elements are interchanged.

The DUAL of an algebraic expression is formed by:

<table>
<thead>
<tr>
<th>Replacing</th>
<th>with</th>
</tr>
</thead>
<tbody>
<tr>
<td>AND</td>
<td>OR</td>
</tr>
<tr>
<td>OR</td>
<td>AND</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Examples:

- \( x \cdot \overline{x} = 0 \) becomes \( x + \overline{x} = 1 \)
- \( x + \overline{x} = 1 \) becomes \( x \cdot \overline{x} = 1 \)

Duality In Proofs

\[
\begin{align*}
x + x &= (x + x) \cdot 1 \\
&= (x + x)(x + \overline{x}) \\
&= x + x \cdot \overline{x} \\
&= x + 0 \\
&= x
\end{align*}
\]

\[
\begin{align*}
x \cdot x &= x \cdot x + 0 \\
&= x \cdot x + x \cdot \overline{x} \\
&= x \cdot (x + \overline{x}) \\
&= x \cdot 1 \\
&= x
\end{align*}
\]

NOTE: Each step in the proof is the dual of the other step. The postulate used is also the dual!
Useful Theorems

- $x \cdot y + \overline{x} \cdot y = y$  \quad $(x + y)(\overline{x} + y) = y$  \quad Simplification
- $x + x \cdot y = x$  \quad $x \cdot (x + y) = x$  \quad Absorption
- $x + \overline{x} \cdot y = x + y$  \quad $x \cdot (\overline{x} + y) = x \cdot y$
- $x \cdot y + \overline{x} \cdot z + y \cdot z = x \cdot y + \overline{x} \cdot z$  \quad Concensus
- $(x + y) \cdot (\overline{x} + z) \cdot (y + z) = (x + y) \cdot (\overline{x} + z)$
- $\overline{x + y} = \overline{x} \cdot \overline{y}$  \quad $\overline{x \cdot y} = \overline{x} + \overline{y}$  \quad DeMorgan's Laws

Proof of Simplification

- $x \cdot y + \overline{x} \cdot y = y$  \quad $(x + y)(\overline{x} + y) = y$
Proof of Consensus

• \( x \cdot y + \overline{x} \cdot z + y \cdot z = x \cdot y + \overline{x} \cdot z \)

Proof of DeMorgan’s Law

• \( \overline{x + y} = \overline{x} \cdot \overline{y} \quad \overline{x \cdot y} = \overline{x} + \overline{y} \)
Boolean Function Evaluation

<table>
<thead>
<tr>
<th></th>
<th>x</th>
<th>y</th>
<th>z</th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
<th>F4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</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>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

F1 = x y \overline{z}  
F2 = x + y \overline{z}  
F3 = x \overline{y} \overline{z} + x y z + x y  
F4 = x \overline{y} + x \overline{z}  

Expression Simplification

- Simplify to contain the smallest number of literals (complemented and uncomplemented variables):
  
  \[ A B + \overline{A} C D + \overline{A} B D + A C \overline{D} + A B C D = \]
Complementing Functions

- Use DeMorgan's Theorem to complement a function:
  1. Interchange AND and OR operators
  2. Complement each literal

Example: Complement \( \overline{x} \overline{y} \overline{z} + \overline{x} \overline{y} z \)

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

- This generates a lot of terms. You might want to simplify the expression first.

Canonical Forms

- It is useful to specify Boolean functions of \( n \) variables in a manner that is easy to compare.
- Two such Canonical Forms are in common usage:
  - Sum of Minterms
  - Product of Maxterms
Minterms

Minterms are AND terms with every variable in true or complemented form. Given that each binary variable may appear normal (e.g. \( x \)) or complemented (e.g. \( x' \)), there are \( 2^n \) minterms for \( n \) variables.

EXAMPLE: Two variables combined with an AND operator, \( x \cdot y \) have 2*2 or 4 combinations:

1. \( x \ y \) (Both normal)
2. \( x \ y' \) (x normal, y complemented)
3. \( x' \ y \) (x complemented, y normal)
4. \( x' \ y' \) (Both complemented)

Thus there are four minterms of two variables.

Maxterms

Maxterms are OR terms with every variable in true or complemented form. Given that each binary variable may appear normal (e.g. \( x \)) or complemented (e.g. \( x' \)), there are \( 2^n \) maxterms for \( n \) variables.

Two variables combined with an OR operator, \( x + y \), have 2*2 or 4 combinations:

1. \( x + y \) (Both normal)
2. \( x + y' \) (x normal, y complemented)
3. \( x' + y \) (x complemented, y normal)
4. \( x' + y' \) (Both complemented)

Thus there are four maxterms of two variables.
Maxterms and Minterms

Examples: Two variable minterms and maxterms.

<table>
<thead>
<tr>
<th>Index</th>
<th>Maxterm</th>
<th>Minterm</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>x + y</td>
<td>x y</td>
</tr>
<tr>
<td>1</td>
<td>x + y'</td>
<td>x y</td>
</tr>
<tr>
<td>2</td>
<td>x + y</td>
<td>x y</td>
</tr>
<tr>
<td>3</td>
<td>x + y</td>
<td>x y</td>
</tr>
</tbody>
</table>

The index above is important for describing which variables in the terms are true and which are complemented.

Standard Order

- Minterms and maxterms are designated with a subscript.
- The subscript is a number, corresponding to a binary pattern.
- The bits in the pattern represent the complemented or normal state of each variable listed in a standard order.
- All variables will be present in a minterm or maxterm and will be listed in the same order (usually alphabetically).

Example: For variables a, b, c:

Maxterms:  (a + b + c),  (a + b + c)
Minterms:  a b c,  a b c,  a b c
Terms:  (b + a + c),  a b,  and (c + b + a) are NOT in standard order.
Terms:  (a + c),  b c,  and (a + b) do not contain all variables.
Purpose of the Index

- The index for the minterm or maxterm, expressed as a binary number, is used to determine whether the variable is shown in the true form or complemented form.
  - For Minterms:
    “1” means the variable is “Not Complemented” and “0” means the variable is “Complemented”.
  - For Maxterms:
    “0” means the variable is “Not Complemented” and “1” means the variable is “Complemented”.

Index Example in Three Variables

- Example: (for three variables)
  Assume the variables are called x, y, and z.
  The standard order is x, then y, then z.
- The Index 0 (base 10) = 000 (base 2 to three digits) so all three variables are complemented for minterm 0 (x, y, z) and no variables are complemented for Maxterm 0 (x, y, z)
  - Minterm 0, called m0 is $\overline{x} \cdot \overline{y} \cdot \overline{z}$.
  - Maxterm 0, called M0 is (x + y + z).
### Four Variables, Index 0-7

<table>
<thead>
<tr>
<th>Index</th>
<th>Binary Pattern</th>
<th>Minterm $m_i$</th>
<th>Maxterm $M_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0000</td>
<td>$a'b'c'd'$</td>
<td>$a+b+c+d$</td>
</tr>
<tr>
<td>1</td>
<td>0001</td>
<td>$a'b'c'd$</td>
<td>$a+b+c+d'$</td>
</tr>
<tr>
<td>2</td>
<td>0010</td>
<td>$a'b'cd'$</td>
<td>$a+b+c'+d$</td>
</tr>
<tr>
<td>3</td>
<td>0011</td>
<td>$a'b'cd$</td>
<td>$a+b+c'+d'$</td>
</tr>
<tr>
<td>4</td>
<td>0100</td>
<td>$a'bc'd'$</td>
<td>$a+b'+c+d$</td>
</tr>
<tr>
<td>5</td>
<td>0101</td>
<td>$a'bc'd$</td>
<td>$a+b'+c+d'$</td>
</tr>
<tr>
<td>6</td>
<td>0110</td>
<td>$a'bcd'$</td>
<td>$a+b'+c'+d$</td>
</tr>
<tr>
<td>7</td>
<td>0111</td>
<td>$a'bcd$</td>
<td>$a+b'+c'+d'$</td>
</tr>
</tbody>
</table>

### Four Variables, Index 8-15

<table>
<thead>
<tr>
<th>Index</th>
<th>Binary Pattern</th>
<th>Minterm $m_i$</th>
<th>Maxterm $M_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>1000</td>
<td>$ab'c'd'$</td>
<td>$a'+b+c+d$</td>
</tr>
<tr>
<td>9</td>
<td>1001</td>
<td>$ab'c'd$</td>
<td>$a'+b+c+d'$</td>
</tr>
<tr>
<td>10</td>
<td>1010</td>
<td>$ab'cd'$</td>
<td>$a'+b+c'+d$</td>
</tr>
<tr>
<td>11</td>
<td>1011</td>
<td>$ab'cd$</td>
<td>$a'+b+c'+d'$</td>
</tr>
<tr>
<td>12</td>
<td>1100</td>
<td>$abc'd'$</td>
<td>$a'+b'+c+d$</td>
</tr>
<tr>
<td>13</td>
<td>1101</td>
<td>$abc'd$</td>
<td>$a'+b'+c+d'$</td>
</tr>
<tr>
<td>14</td>
<td>1110</td>
<td>$abcd'$</td>
<td>$a'+b'+c'+d$</td>
</tr>
<tr>
<td>15</td>
<td>1111</td>
<td>$abcd$</td>
<td>$a'+b'+c'+d'$</td>
</tr>
</tbody>
</table>
Minterm and Maxterm Relationship

Review: DeMorgan's Theorem
\[(x \cdot y) = (\bar{x} + \bar{y}) \quad \text{and} \quad (x + y) = (\bar{x} \cdot \bar{y})\]

Note: For 2 variables:
\[M_2 = (\bar{x} + y) \quad \text{and} \quad m_2 = (x \cdot \bar{y})\]

Thus \(M_2\) is the complement of \(m_2\) and vice-versa.

Since DeMorgan's Theorem can be extended to \(n\) variables, this holds that for terms of \(n\) variables giving:

\[M_i \quad \text{and} \quad m_i \quad \text{are complements.}\]

Function Tables for Both

<table>
<thead>
<tr>
<th>(x)</th>
<th>(y)</th>
<th>(x'y')</th>
<th>(m_0)</th>
<th>(x'y)</th>
<th>(m_1)</th>
<th>(xy')</th>
<th>(m_2)</th>
<th>(xy)</th>
<th>(m_3)</th>
</tr>
</thead>
<tbody>
<tr>
<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>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>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>1</td>
<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>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(x)</th>
<th>(y)</th>
<th>(x+y)</th>
<th>(M_0)</th>
<th>(x+y')</th>
<th>(M_1)</th>
<th>(x'+y)</th>
<th>(M_2)</th>
<th>(x'+y')</th>
<th>(M_3)</th>
</tr>
</thead>
<tbody>
<tr>
<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>1</td>
<td>0</td>
<td>0</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>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</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>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Note that the each column in the minterm function table is the complement of the column in the Maxterm function table since \(M_i\) and \(m_i\) are complements.
Observations

In the function tables:

- Each minterm has one and only one 1 present in the \(2^n\) terms (a minimum of ones). All other entries are 0.
- Each maxterm has one and only one 0 present in the \(2^n\) terms (a maximum of ones). All other entries are 1.

It seems that we can implement any function by "ORing" the minterms where we want a "1" to appear in the function table.

Similarly, we can implement any function by "ANDing" the maxterms where we want a "0" to appear in the function table.

This gives us two Canonical Forms for stating an arbitrary Boolean function.

---

Minterm Function Example

Example: Implement \(F_1 = m_1 + m_4 + m_7\)

\[
F_1 = \overline{x} \overline{y} z + x \overline{y} \overline{z} + x y z
\]

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>z</th>
<th>index</th>
<th>(m_1 + m_4 + m_7) = (F_1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0 + 0 + 0 = 0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1 + 0 + 0 = 1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0 + 0 + 0 = 0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>0 + 0 + 0 = 0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>0 + 1 + 0 = 1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>0 + 0 + 0 = 0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>6</td>
<td>0 + 0 + 0 = 0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7</td>
<td>0 + 0 + 1 = 1</td>
</tr>
</tbody>
</table>
Minterm Function Example

• \( F(A, B, C, D, E) = m_2 + m_9 + m_{17} + m_{23} \)

Maxterm Function Example

Example: Implement \( F_1 \) in maxterms:
\[
F_1 = M_0 \cdot M_2 \cdot M_3 \cdot M_5 \cdot M_6
\]
\[
F_1 = (x + y + z) \cdot (\overline{x} + y + z) \cdot (x + y + z') \cdot (\overline{x} + y + z) \cdot (x + \overline{y} + z)
\]

<table>
<thead>
<tr>
<th>( x )</th>
<th>( y )</th>
<th>( z )</th>
<th>( i )</th>
<th>( M_0 \cdot M_2 \cdot M_3 \cdot M_5 \cdot M_6 = F_1 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>( 0 \cdot 1 \cdot 1 \cdot 1 \cdot 1 = 0 )</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>( 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 = 1 )</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>( 1 \cdot 0 \cdot 1 \cdot 1 \cdot 1 = 0 )</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>( 1 \cdot 1 \cdot 0 \cdot 1 \cdot 1 = 0 )</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>( 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 = 1 )</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>( 1 \cdot 1 \cdot 1 \cdot 0 \cdot 1 = 0 )</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>6</td>
<td>( 1 \cdot 1 \cdot 1 \cdot 1 \cdot 0 = 0 )</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7</td>
<td>( 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 = 1 )</td>
</tr>
</tbody>
</table>
Maxterm Function Example

- \( F(A, B, C, D) = M3 \cdot M8 \cdot M11 \cdot M14 \)

Cannonical Sum of Minterms

Any Boolean Function can be expressed as a Sum of Minterms.
The minterms used are the terms corresponding to the "1's" in the function table.

Note: You may have to expand all terms first to explicitly list all minterms. You do this by "ANDing" any term missing a variable \( v \) with a term \( (v + \overline{v}) \).

Example: Implement \( f = x + \overline{x} \overline{y} \) as a sum of minterms.
First expand terms: \( f = x (y + \overline{y}) + \overline{x} \overline{y} \)
Then collect terms: \( f = x y + x \overline{y} + \overline{x} \overline{y} \)
Express as sum of minterms: \( f = m3 + m2 + m0 \)
Another SOM Example

Example: $F = A + B'C$

There are three variables, $A$, $B$, and $C$ which we take to be the standard order.

Expanding the terms missing variables:

\[ A = A(B+B') = AB + AB' \]
\[ AB = AB(C+C') = ABC + ABC' \]
\[ AB' = AB'(C+C') = AB'C + AB'C' \]
and $B'C = (A+A')B'C = AB'C + A'B'C$

Collect terms (and get rid of duplicate terms):

\[ F = ABC + ABC' + AB'C + AB'C' + A'B'C \]

Expressed as minterms: $F = m7 + m6 + m5 + m4 + m1$

Shorthand SOM Form

From the previous example, we started with:

\[ F = A + B'C \]

We ended up with:

\[ F = m7 + m6 + m5 + m4 + m1 \]

This can be denoted in the formal shorthand:

\[ F(A, B, C) = \sum (1, 4, 5, 6, 7) \]

Note that we explicitly show the standard variables in order and drop the “m” designators.
Canonical Product of Maxterms

Any Boolean Function can be expressed as a Product of Maxterms.

The maxterms used are the terms corresponding to the "0's" in the function table.

Note: You may have to expand all terms first to explicitly list all maxterms. You do this by first using the distributive law and by "ORing" terms missing variable v with a term equal to v'v'.

Example: Implement \( f = x + x'y' \) as a product of maxterms.

First apply the distributive law:
\[
x + x'y' = (x + x')(x + y') = 1 \cdot (x + y') = (x + y')
\]

In this example, we do not need to add missing variables.

Express as sum of Maxterms:
\[
f = M_1
\]

Product of Maxterm Example

Given \( f(A,B,C) = AC' + BC + A'B' \)
we must convert it to Product of Maxterm form.

Use distribution of + over \( \cdot \) to get:
(\text{Hint: } x + yz = (x+y)(x+z))
so substitute \( x = (AC' + BC), \ y = A', \ \text{and} \ z = B' \)
\[
f = (AC' + BC + A')(AC' + BC + B')
\]
Then use the relation \( x + x'y = x + y \) to get:
\[
f = (C' + BC + A')(AC' + C + B') \quad \text{and a second time for:}
f = (C' + B + A')(A + C + B') \quad \text{and then rearrange to get:}
f = (A' + B + C')(A + B' + C)
\]
Thus:
\[
f = M_5 \cdot M_2
\]
Function Complements

The complement of a function expressed as a sum of minterms is constructed by selecting the minterms missing in the sum-of-minterms canonical forms.

Alternatively, the complement of a function expressed by a Sum of Minterms form is simply the Product of Maxterms of the same index.

Example: Given \( F(x,y,z) = \sum (1,3,5,7) \)

Then:
\( F' = \sum (0,2,4,6) \)

Or alternately:
\( F' = \prod (1,3,5,7) \)

Conversion Between Forms

To convert between sum-of-minterms and product-of-maxterms form (or vice-versa) we follow these steps:

1. Implement the function complement by swapping terms in the list with terms not in the list.
2. Change from products to sums, or vice versa.

Example: Given \( F \) as before:
\[ F = \sum (1,3,5,7) \]

Form the Complement:
\[ \overline{F} = \sum (0,2,4,6) \]

Then use the other form with the same indices -- this forms the complement again.
\[ F = \prod (0,2,4,6) \]
Review of Canonical Forms

Minterms -- Terms with all variables present, combined with "AND"
For $n$ variables combined with AND, there are $2^n$ combinations. Each unique combination is called a MINTERM.
(Examples: $X\cdot Y\cdot Z$, $A'\cdot B\cdot C'$)

Maxterms -- Terms with all variables present, combined with "OR"
For $n$ variables combined with OR, there are $2^n$ combinations. Each unique combination is called a MAXTERM.
(Examples: $X+Y+Z$, $A'+B+C'$)

Review: Indices

Given $n$ variables, use an $n$-bit binary expansion of the Index, $i$, to determine variable "true" or "complemented" state.
For Minterms:
"1" ⇒ "True", "0" ⇒ "Complemented".
For Maxterms:
"0" ⇒ "True", "1" ⇒ "Complemented".
Minterm 0, called $m_0$ is $x'\cdot y'\cdot z'$.
Maxterm 0, called $M_0$ is $(x+y+z)$.
### Forms of Terms, Complements

<table>
<thead>
<tr>
<th>Index $i$</th>
<th>Binary Pattern.</th>
<th>Minterm $m_i$</th>
<th>Maxterm $M_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0000</td>
<td>a'b'c'd'</td>
<td>a+b+c+d</td>
</tr>
<tr>
<td>1</td>
<td>0001</td>
<td>a'b'c'd</td>
<td>a+b+c+d'</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>1110</td>
<td>abcd'</td>
<td>a'+b'+c'+d</td>
</tr>
<tr>
<td>15</td>
<td>1111</td>
<td>abcd</td>
<td>a'+b'+c'+d</td>
</tr>
</tbody>
</table>

$M_i$ is the complement of $m_i$ and vice-versa.

### Review: Sum of Minterms Form

Any Boolean Function can be expressed as a sum of minterms.

Example: $F = A + B'C$

Expand the terms missing variables and collect terms (get rid of duplicate terms):

$$F = ABC + ABC' + AB'C + AB'C' + A'B'C$$

Express as Sum of minterms:

$$F = m_7 + m_6 + m_5 + m_4 + m_1$$

This can be denoted:

$$F(A,B,C) = \sum(1,4,5,6,7)$$
Review: Product of Maxterms

Any Boolean Function can be expressed as a product of maxterms.

Example: Implement $F = xy + x'y'$ as a product of maxterms.

\[
xy + x'y' = (xy + x') \bullet (xy + y')
\]
\[
= (x' + xy) \bullet (y' + xy)
\]
\[
= (x' + x) \bullet (y' + x) \bullet (y' + y)
\]
\[
= (1) \bullet (x' + y) \bullet (y' + x) \bullet (1)
\]
\[
= (x' + y) \bullet (x + y')
\]
\[
= M_2 \bullet M_1
\]

Which can be denoted:

\[ F = \prod (1, 2) \]

Review: Complements, Conversions

Complement: Swap indices with the indices not in the list, or use opposite canonical form.

Example: Given $F$

\[
F = \sum (1, 3, 5, 7) \quad \text{or} \quad F' = \sum (0, 2, 4, 6)
\]
\[
F = \prod (0, 2, 4, 6) \quad \text{or} \quad F' = \prod (1, 3, 5, 7)
\]

Canonical Conversion: To change the form of a function, complement the index set then change form.