Announcement
Reason to Learn Boolean Algebra

- Digital computers contain circuits that implement Boolean functions.
- The simpler that we can make a Boolean function, the smaller the circuit that will result.
  - Simpler circuits are cheaper to build, consume less power, and run faster than complex circuits.
- With this in mind, we always want to reduce our Boolean functions to their simplest form.
- There are a number of Boolean identities that help us to do this.

Boolean Identities

- Most Boolean identities have an AND (product) form as well as an OR (sum) form. We give our identities using both forms. Our first group is rather intuitive:

  - We can use Boolean identities to simplify the function: \( F(X, Y, Z) = (X + Y)(X + \overline{Y})(XZ) \)

<table>
<thead>
<tr>
<th>Identity Name</th>
<th>AND Form</th>
<th>OR Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identity Law</td>
<td>( lx = x )</td>
<td>( 0 + x = x )</td>
</tr>
<tr>
<td>Null Law</td>
<td>( ox = 0 )</td>
<td>( 1 + x = 1 )</td>
</tr>
<tr>
<td>Idempotent Law</td>
<td>( xx = x )</td>
<td>( x + x = x )</td>
</tr>
<tr>
<td>Inverse Law</td>
<td>( \overline{x} = 0 )</td>
<td>( x + \overline{x} = 1 )</td>
</tr>
</tbody>
</table>
DeMorgan’s law

- DeMorgan’s law provides an easy way of finding the complement of a Boolean function.
- Recall DeMorgan’s law states:
  \[
  \overline{xy} = \overline{x} + \overline{y} \quad \text{and} \quad \overline{x+y} = \overline{x} \overline{y}
  \]
- DeMorgan’s law can be extended to any number of variables.
- Replace each variable by its complement and change all ANDs to ORs and all ORs to ANDs.
- Thus, we find the complement of:
  \[
  F(x, y, z) = (xy) + (xz) + (yz)
  \]
  \[
  \overline{F(x, y, z)} = \overline{(xy) + (xz) + (yz)}
  \]
  \[
  = \overline{xy} \overline{xz} \overline{yz}
  \]
  \[
  = (x+\overline{y})(x+\overline{z})(\overline{y}+\overline{z})
  \]

Canonical Form of Boolean Function

- Through our exercises in simplifying Boolean expressions, we see that there are numerous ways of stating the same Boolean expression.
  - These “synonymous” forms are logically equivalent.
  - Logically equivalent expressions have identical truth tables.
- In order to eliminate as much confusion as possible, designers express Boolean functions in standardized or canonical form.
- There are two canonical forms for Boolean expressions: sum-of-products and product-of-sums.
  - Recall the Boolean product is the AND operation and the Boolean sum is the OR operation.
- In the sum-of-products form, ANDed variables are ORed together.
  - For example: \( F(x, y, z) = xy + xz + yz \)
- In the product-of-sums form, ORed variables are ANDed together.
  - For example: \( F(x, y, z) = (x+y)(x+z)(y+z) \)
Sum of Product Form

- It is easy to convert a function to sum-of-products form using its truth table.
- We are interested in the values of the variables that make the function true (=1).
- Using the truth table, we list the values of the variables that result in a true function value.
- Each group of variables is then ORed together.

The sum-of-products form for our function is:

\[ F(x, y, z) = \bar{x}\bar{y}z + \bar{x}yz + x\bar{y}z + xy\bar{z} + xyz \]