Boolean algebra

- Last lecture
  - Binary numbers
  - Base conversion
  - Number systems
    - Twos-complement
    - A/D and D/A conversion

- Today’s lecture
  - Boolean algebra
    - Axioms
    - Useful laws and theorems
    - Simplifying Boolean expressions

Major topic: Combinational logic

- Axioms and theorems of Boolean algebra

- Logic functions and truth tables
  - AND, OR, Buffer, NAND, NOR, NOT, XOR, XNOR

- Gate logic
  - Networks of Boolean functions

- Canonical forms
  - Sum of products and product of sums

- Simplification
  - Boolean cubes and Karnaugh maps
  - Two-level simplification
Combinational versus sequential

- **Combinational**: Memoryless
  - Apply fixed inputs A, B
  - Wait for clock edge
  - Observe C
  - Wait for another clock edge
  - Observe C again: C will stay the same

- **Sequential**: With Memory
  - Apply fixed inputs A, B
  - Wait for clock edge
  - Observe C
  - Wait for another clock edge
  - Observe C again: C may be different

---

Boolean algebra

- A **Boolean algebra** comprises...
  - A set of elements B
  - Binary operators \{ + , \cdot \}
  - A unary operation \{ ' \}

- ...and the following axioms
  - 1. The set B contains at least two elements \{a, b\} with a ≠ b
  - 2. Closure: \( a + b \) is in B \( a \cdot b \) is in B
  - 3. Commutative: \( a + b = b + a \) \( a \cdot b = b \cdot a \)
  - 4. Associative: \( a + (b + c) = (a + b) + c \) \( a \cdot (b \cdot c) = (a \cdot b) \cdot c \)
  - 5. Identity: \( a + 0 = a \) \( a \cdot 1 = a \)
  - 6. Distributive: \( a + (b \cdot c) = (a + b) \cdot (a + c) \) \( a \cdot (b + c) = (a \cdot b) + (a \cdot c) \)
  - 7. Complementarity: \( a + a' = 1 \) \( a \cdot a' = 0 \)
Digital (binary) logic is a Boolean algebra

- Substitute
  - \{0, 1\} for B
  - AND for \( \cdot \) Boolean Product
  - OR for + Boolean Sum
  - NOT for \( ' \)

- All the axioms hold for binary logic

- Definitions
  - Boolean function
    - Maps inputs from the set \{0,1\} to the set \{0,1\}
  - Boolean expression
    - An algebraic statement of Boolean variables and operators

AND, OR, Not

- AND \( X \cdot Y \)

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</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>1</td>
<td>1</td>
</tr>
</tbody>
</table>

- OR \( X + Y \)

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</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>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

- NOT \( \overline{X} \)

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Logic functions and Boolean algebra

◆ Any logic function that is expressible as a truth table can be written in Boolean algebra using +, •, and ' |

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</th>
<th>Z=X•Y</th>
</tr>
</thead>
<tbody>
<tr>
<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>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<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'</th>
<th>Y'</th>
<th>X•Y</th>
<th>X'•Y'</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</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>0</td>
<td>0</td>
</tr>
<tr>
<td>1</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>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Z=(X•Y)+(X'•Y')

Two key concepts

◆ Duality (a meta-theorem— a theorem about theorems)
  - All Boolean expressions have logical duals
  - Any theorem that can be proved is also proved for its dual
  - Replace: • with +, + with •, 0 with 1, and 1 with 0
  - Leave the variables unchanged

◆ de Morgan’s Theorem
  - Procedure for complementing Boolean functions
  - Replace: • with +, + with •, 0 with 1, and 1 with 0
  - Replace all variables with their complements
Useful laws and theorems

Identity: \( X + 0 = X \) \hspace{2cm} Dual: \( X \cdot 1 = X \)

Null: \( X + 1 = 1 \) \hspace{2cm} Dual: \( X \cdot 0 = 0 \)

Idempotent: \( X + X = X \) \hspace{2cm} Dual: \( X \cdot X = X \)

Involution: \((X')' = X\)

Complementarity: \( X + X' = 1 \) \hspace{2cm} Dual: \( X \cdot X' = 0 \)

Commutative: \( X + Y = Y + X \) \hspace{2cm} Dual: \( X \cdot Y = Y \cdot X \)

Associative: \( (X + Y) + Z = X + (Y + Z) \) \hspace{2cm} Dual: \( (X \cdot Y) \cdot Z = X \cdot (Y \cdot Z) \)

Distributive: \( X \cdot (Y + Z) = (X \cdot Y) + (X \cdot Z) \) \hspace{2cm} Dual: \( X + (Y \cdot Z) = (X + Y) \cdot (X + Z) \)

Uniting: \( X \cdot Y + X \cdot Y' = X \) \hspace{2cm} Dual: \( (X + Y) \cdot (X + Y') = X \)

Useful laws and theorems (con’t)

Absorption: \( X + X \cdot Y = X \) \hspace{2cm} Dual: \( X \cdot (X + Y) = X \)

Absorption (\#2): \( (X + Y')' \cdot Y = X \cdot Y \) \hspace{2cm} Dual: \( (X \cdot Y') + Y = X + Y \)

de Morgan's: \( (X + Y + \ldots)' = X' \cdot Y' \cdot \ldots \) \hspace{2cm} Dual: \( (X \cdot Y \cdot \ldots)' = X' + Y' + \ldots \)

Duality: \( (X + Y + \ldots)' = X' \cdot Y' \cdot \ldots \) \hspace{2cm} Dual: \( (X \cdot Y \cdot \ldots)' = X + Y + \ldots \)

Multiplying & factoring: \( (X + Y) \cdot (X' + Z) = X \cdot Z + X' \cdot Y \)
\hspace{2cm} Dual: \( X \cdot Y + X' \cdot Z = (X + Z) \cdot (X' + Y) \)

Consensus: \( (X \cdot Y) + (Y \cdot Z) + (X' \cdot Z) = X \cdot Y + X \cdot Z \)
\hspace{2cm} Dual: \( (X + Y) \cdot (Y + Z) \cdot (X' + Z) = (X + Y) \cdot (X' + Z) \)
Proving theorems

- Example 1: Prove the uniting theorem-- $X \cdot Y + X \cdot Y' = X$
  
  Distributive: $X \cdot Y + X \cdot Y' = X \cdot (Y + Y')$
  Complementarity: $= X \cdot (1)$
  Identity: $= X$

- Example 2: Prove the absorption theorem-- $X + X \cdot Y = X$
  
  Identity: $X + X \cdot Y = (X \cdot 1) + (X \cdot Y)$
  Distributive: $= X \cdot (1 + Y)$
  Null: $= X \cdot (1)$
  Identity: $= X$

---

Proving theorems

- Example 3: Prove the consensus theorem-- $(XY) + (YZ) + (X'Z) = XY + X'Z$
  
  Complementarity: $XY + YZ + X'Z = XY + (X + X')YZ + X'Z$
  Distributive: $= XYZ + XY + X'YZ + X'Z$

  - Use absorption $\{AB + A = A\}$ with $A = XY$ and $B = Z$
    
    $= XY + X'YZ + X'Z$

  - Rearrange terms
    
    $= XY + X'YZ + X'Z$

  - Use absorption $\{AB + A = A\}$ with $A = X'Z$ and $B = Y$
    
    $XY + YZ + X'Z = XY + X'Z$
de Morgan’s Theorem

- Use de Morgan’s Theorem to find complements
- Example: $F = (A+B)(A'+C)$, so $F' = (A'B') + (A'C')$

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

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

Logic simplification

- Use the axioms to simplify logical expressions
  - Why? To use less hardware

- Example: A two-level logic expression

  $Z = A'B'C + AB'C' + AB'C + ABC + ABC'$
  $= AB'C + AB'C' + A'B'C + ABC' + ABC$ rearrange
  $= AB'(C + C') + A'B'C + AB(C' + C)$ distributive
  $= AB' + A'B'C + AB$ comp.
  $= AB' + AB + A'BC$ rearrange
  $= A(B' + B) + A'BC$ distributive
  $= A + A'BC$ comp.

  Use absorption #2D \(((X \cdot Y') + Y = X + Y)\) with $X=BC$ and $Y=A$

  $Z = A + BC$
Example: A full adder

- 1-bit binary adder
  - Inputs: A, B, Carry-in
  - Outputs: Sum, Carry-out

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Cin</th>
<th>S</th>
<th>Cout</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>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</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>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ S = A'B'Cin + A'B'Cin' + AB'Cin + ABCin \]

\[ Cout = A'B'Cin + AB'Cin + ABCin' + ABCin \]

Simplifying the carry-out function

\[ Cout = A'B'Cin + AB'Cin + ABCin' + ABCin \]
\[ = A'B'Cin + AB'Cin + ABCin' + ABCin + ABCin \]
\[ = A'B'Cin + ABCin + ABCin + ABCin + ABCin \]
\[ = (A' + A)BCin + AB'Cin + ABCin' + ABCin \]
\[ = (1)BCin + AB'Cin + ABCin' + ABCin \]
\[ = BCin + AB'Cin + ABCin' + ABCin + ABCin \]
\[ = BCin + AB'Cin + ABCin + ABCin + ABCin' + ABCin \]
\[ = BCin + A(B' + B)Cin + ABCin' + ABCin \]
\[ = BCin + A(1)Cin + ABCin' + ABCin \]
\[ = BCin + ACin + AB(Cin' + Cin) \]
\[ = BCin + ACin + AB(1) \]
\[ = BCin + ACin + AB \]
Some notation

- Priorities: $\overline{A} \cdot B + C = ((\overline{A}) \cdot B) + C$
- Variables are sometimes called literals