Boolean Algebra

• a.k.a. “switching algebra”
  – Deals with Boolean values → 0, 1

• Positive-logic convention
  – Analog voltages LOW, HIGH → 0, 1

• Negative logic -- seldom used

• Signal values denoted by variables (X, Y, FRED, etc)
Boolean Algebra is Just Like Boolean Logic ...

- **NOT is a prime (')**:  
  - \(0' = 1\)  
  - \(1' = 0\)

- **OR is a plus (+)**:  
  - \(0 + 0 = 0\)  
  - \(0 + 1 = 1\)  
  - \(1 + 0 = 1\)  
  - \(1 + 1 = 1\)

- **AND is multiplication dot (·)**:  
  - \(0 · 0 = 0\)  
  - \(0 · 1 = 0\)  
  - \(1 · 0 = 0\)  
  - \(1 · 1 = 1\)

**Axioms (will lead to Theorems)**

- **Variable X can take only one of two values**:  
  (A1) \(X = 0\) if \(X \neq 1\)  
  (A1') \(X = 1\) if \(X \neq 0\)

- **Complement**:  
  (A2) if \(X = 0\), then \(X' = 1\)  
  (A2') if \(X = 1\) if \(X' = 0\)

- **Three axioms to define the AND and the OR operations**:  
  (A3) \(0 · 0 = 0\)  
  (A3') \(1 + 1 = 1\)  
  (A4) \(1 · 1 = 1\)  
  (A4') \(0 + 0 = 0\)  
  (A5) \(0 · 1 = 1 · 0 = 0\)  
  (A5') \(1 + 0 = 0 + 1 = 1\)
**Boolean Operators**

- **Complement:** \( X' \) (opposite of \( X \))
- **AND:** \( X \cdot Y \)
- **OR:** \( X + Y \)

<table>
<thead>
<tr>
<th>( X )</th>
<th>( Y )</th>
<th>( X \text{ AND } Y )</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>

<table>
<thead>
<tr>
<th>( X )</th>
<th>( Y )</th>
<th>( X \text{ OR } Y )</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>

- Axiomatic definition: \( A1 - A5, A1' - A5' \)

**Logic Symbols**

- **NOT** \( X \):
  \[ Z = \text{ NOT } X \]
  \[ Z = X' \]
  \[ Z = \text{ NOT } (\text{complement}) \]

- **AND**
  \[ Z = X \text{ AND } Y \]
  \[ Z = X \cdot Y \]

- **OR**
  \[ Z = X \text{ OR } Y \]
  \[ Z = X + Y \]
Duality

• Swap 0 & 1, AND & OR
  – Result: Theorems still true

• Why?
  – Each axiom (A1 – A5) has a dual (A1’ – A5’)

Some Definitions

• Literal: a variable or its complement
  – X, X’, FRED’, CS_L

• Expression: literals combined by AND, OR, parentheses, complementation
  – X + Y
  – P · Q · R
  – A + B · C
  – ((FRED · Z’) + CS_L · A · B’ · C + Q5) · RESET’

• Equation: Variable = Expression
  – P = ((FRED · Z’) + CS_L · A · B’ · C + Q5) · RESET’
Theorems - One Variable

(T1) \( X + 0 = X \)  \( (T1') \) \( X \cdot 1 = X \) (Identities)
(T2) \( X + 1 = 1 \)  \( (T2') \) \( X \cdot 0 = 0 \) (Null elements)
(T3) \( X + X = X \)  \( (T3') \) \( X \cdot X = X \) (Idempotency)
(T4) \( (X')' = X \)  \( (T4') \) \( (X')' = X \) (Involution)
(T5) \( X + X' = 1 \)  \( (T5') \) \( X \cdot X' = 0 \) (Complements)

- Proofs by *perfect induction*
- Axiom \( (A1) \) is the key (a variable can take only one of two values: 0 or 1)

Proofs of One-Variable Theorems

(*perfect induction*)

(T3) idempotency:

\[
\begin{align*}
X + X &= X & [X=0] & 0+0 = 0 & \text{true, according to } (A4') \\
X + X &= X & [X=1] & 1+1 = 1 & \text{true, according to } (A3')
\end{align*}
\]

(T4) involution:

\[
\begin{align*}
(X')' &= X & [X=0] & (0')' = 1' = 0 & \text{true, according to } (A2) \\
(X')' &= X & [X=1] & (1')' = 0' = 1 & \text{& } (A2')
\end{align*}
\]

Etc.
Boolean Operator Precedence

- The order of evaluation is:
  - Parentheses
  - NOT
  - AND
  - OR

- Consequence: Parentheses appear around OR expressions

- Example:
  \[ F = A \cdot (B + C) \cdot (C + D) \]

Theorems - Two or Three Variables

- \((T6)\) \[ X + Y = Y + X \]
  - (Commutation)

- \((T7)\) \[ (X + Y) + Z = X + (Y + Z) \]
  - (Associativity)

- \((T8)\) \[ X \cdot Y + X \cdot Z = X \cdot (Y + Z) \]
  - (Distributivity)

- \((T9)\) \[ X + X \cdot Y = X \]
  - (Covering)

- \((T10)\) \[ X \cdot Y + X \cdot Y' = X \]
  - (Combining)

- \((T11)\) \[ X \cdot Y + X' \cdot Z + Y \cdot Z = X \cdot Y + X' \cdot Z \]
  - (Consensus)

- \((T11')\) \[ (X + Y) \cdot (X' + Z) \cdot (Y + Z) = (X + Y) \cdot (X' + Z) \]
Boolean Algebraic Proof - Example

**X + X · Y = X** ← Covering Theorem (T9)

<table>
<thead>
<tr>
<th>Proof Steps:</th>
<th>Justification:</th>
</tr>
</thead>
<tbody>
<tr>
<td>X + X · Y</td>
<td></td>
</tr>
<tr>
<td>= X · 1 + X · Y</td>
<td>Identity element: X · 1 = X (T1')</td>
</tr>
<tr>
<td>= X · (1 + Y)</td>
<td>Distributivity (T8)</td>
</tr>
<tr>
<td>= X · 1</td>
<td>Null elements (T2): 1 + Y = 1</td>
</tr>
<tr>
<td>= X</td>
<td>Identity element (T1')</td>
</tr>
</tbody>
</table>

Why Theorems and Proofs?

- These theorems are useful *rules of substitution* for logic expressions
- Why substitution? —Because we may want to:
  - Design a simpler circuit (faster, easier to implement, cheaper, more reliable)
  - Use different gates for implementation (same reasons)
- Our primary reason for doing proofs is to learn:
  - Careful and efficient use of the identities and theorems of Boolean algebra, and
  - How to choose the appropriate substitution ("theorem") to apply to make forward progress, irrespective of the application
Distributivity (dual)

(T8')

\[(X + Y) \cdot (X + Z) = X \cdot X + X \cdot Z + Y \cdot X + Y \cdot Z\]
\[= X + X \cdot Z + X \cdot Y + Y \cdot Z = X + X \cdot Y + Y \cdot Z\]
\[= X + Y \cdot Z\]

\[(X + Y) \cdot (X + Z) = X + Y \cdot Z \quad \text{(Distributivity)}\]

(3 + 5) \cdot (3 + 7) \neq 3 + 5 \cdot 7 \quad \text{!!!}

parentheses, operator precedence!

Consensus Theorem

\[X \cdot Y + X' \cdot Z + Y \cdot Z = X \cdot Y + X' \cdot Z \quad \text{Consensus (T11)}\]

<table>
<thead>
<tr>
<th>Proof Steps</th>
<th>Justification</th>
</tr>
</thead>
<tbody>
<tr>
<td>X.Y + X'Z + Y.Z</td>
<td>Identity (T1')</td>
</tr>
<tr>
<td>= X.Y + X'Z + 1 \cdot Y.Z</td>
<td>Identity (T1')</td>
</tr>
<tr>
<td>= X.Y + X'Z + (X + X') \cdot Y.Z</td>
<td>Complement (T5)</td>
</tr>
<tr>
<td>= X.Y + X'Z + X.Y.Z + X'.Y.Z</td>
<td>Distributive (T8)</td>
</tr>
<tr>
<td>= X.Y + X.Y.Z + X'Z + X'.Z.Y</td>
<td>Commutative (T6)</td>
</tr>
<tr>
<td>= X.Y \cdot 1 + X.Y.Z + X'Z \cdot 1 + X'.Z.Y</td>
<td>Identity (T1')</td>
</tr>
<tr>
<td>= X.Y \cdot (1+Z) + X'.Z \cdot (1+Y)</td>
<td>Distributive (T8)</td>
</tr>
<tr>
<td>= X.Y \cdot 1 + X'.Z \cdot 1</td>
<td>1+X = 1 (T2)</td>
</tr>
<tr>
<td>= X.Y + X'.Z</td>
<td>Identity (T1')</td>
</tr>
</tbody>
</table>
Theorems for Expressions

The theorems remain valid if a variable is replaced by an expression.

\[ X \rightarrow U \cdot W \]

\[ U \cdot W + Y \cdot Z = (U \cdot W + Y) \cdot (U \cdot W + Z) = \]

\[ = (U + Y) \cdot (W + Y) \cdot (U + Z) \cdot (W + Z) \quad \leftarrow \text{distributivity (dual)} \]

\[ Z \rightarrow X' \]

\[ (X + Y) \cdot (X + X') = X + Y \cdot X' = X + Y \]

\[ \quad \text{distributivity (dual)} \]

N-variable Theorems

(T12) \[ X + X + \ldots + X = X \quad \text{(Generalized idempotency)} \]

(T12') \[ X \cdot X \cdot \ldots \cdot X = X \]

(T13) \[ (X_1 \cdot X_2 \cdot \ldots \cdot X_n)' = X_1' + X_2' + \ldots + X_n' \quad \text{(DeMorgan's theorems)} \]

(T13') \[ (X_1 + X_2 + \ldots + X_n)' = X_1' \cdot X_2' \cdot \ldots \cdot X_n' \]

(T14) \[ [F(X_1, X_2, \ldots, X_n, +, \cdot)]' = F(X_1', X_2', \ldots, X_n', +) \]

\[ \uparrow \quad \text{(Generalized DeMorgan's theorem)} \]

\[ \downarrow \quad \text{(Shannon's expansion theorems)} \]

(T15) \[ F(X_1, X_2, \ldots, X_n) = X_1 \cdot F(1, X_2, \ldots, X_n) + X_1' \cdot F(0, X_2, \ldots, X_n) \]

(T15') \[ F(X_1, X_2, \ldots, X_n) = [X_1 + F(0, X_2, \ldots, X_n)] \cdot [X_1' + F(0, X_2, \ldots, X_n)] \]

- Prove using finite induction
- Most important: DeMorgan’s theorems
DeMorgan’s Theorems

Proof by finite induction: (basis step, \(n=2\); induction step, \(n=\text{i} \rightarrow n=\text{i}+1\))

\[
A = X_1 + X_2 \\
B = X_1' \cdot X_2' \\
\begin{align*}
\text{If } A \cdot B &= 0 \text{ and } A + B = 1 \\
\text{then } A' &= B
\end{align*}
\]

\[
A \cdot B = (X_1 + X_2) \cdot (X_1' \cdot X_2') = 0 \quad \text{basis step}
\]

\[
A + B = X_1 + X_2 + X_1' \cdot X_2' = X_1 + X_2 \cdot X_1 + X_2 \cdot X_1' + X_1' \cdot X_2' = X_1 + X_1' + X_1 \cdot X_2 = 1
\]

\[
\text{induction step} \quad \text{assume } n = \text{i true , then for } n = \text{i} + 1 \\
(A_i + X_{i+1})' = B_i \cdot X_{i+1}'
\]

DeMorgan Symbols

\begin{align*}
X + Y & \quad \text{OR} \\
(X + Y)' & \quad \text{NOR} \\
X \cdot Y & \quad \text{AND} \\
(X \cdot Y)' & \quad \text{NAND} \\
X & \quad \text{BUFFER} \\
X' & \quad \text{INVERTER}
\end{align*}
DeMorgan Symbol Equivalence for **NOR**

| NOR |  
|-----------------|-----------------|
| $X + Y$ | $Z = (X + Y)'$ |
| $X$ | $Z = (X + Y)'$ |
| $Y$ |  

is the equivalent to

| NOR |  
|-----------------|-----------------|
| $X'$ | $Z = X' \cdot Y'$ |
| $Y'$ | $Z = X' \cdot Y'$ |

DeMorgan Symbol Equivalence for **NAND**

| NAND |  
|-----------------|-----------------|
| $X \cdot Y$ | $Z = (X \cdot Y)'$ |
| $X$ | $Z = (X \cdot Y)'$ |
| $Y$ |  

is the equivalent to

| NAND |  
|-----------------|-----------------|
| $X'$ | $Z = X' + Y'$ |
| $Y'$ | $Z = X' + Y'$ |
| $X'$ | $Z = X' + Y'$ |
| $Y'$ | $Z = X' + Y'$ |
Sum-of-Products Form

**AND-OR:**

![Logic gates diagram](image)

**NAND-NAND:**

NAND-NAND preferred in TTL technology.

Product-of-Sums Form

**OR-AND:**

![Logic gates diagram](image)

**NOR-NOR:**

Product-of-sums preferred in CMOS technology.