CSE 140, Lecture 2
Combinational Logic

CK Cheng
CSE Dept.
UC San Diego

Combinational Logic Outlines

1. Introduction
   • Scope
   • Boolean Algebra (Review)
     • Definition
     • Duality
     • AND/OR Gates
     • Expressions vs Circuits
   • Handy Tools
     • DeMorgan's Theorem
     • Consensus Theorem
     • Shannon's Expansion

2. Specification

3. Synthesis
   • K-map

No memory
AND, OR, NOT logic + 0, 1 constant
Acyclic graph
Flow
Non sequential (Cycle)
1.1 Combinational Logic: Scope

- **Description**
  - **Language**: e.g. C Programming, Verilog, VHDL
  - **Boolean algebra**
  - **Truth table**: Powerful engineering tool

- **Design**
  - **Schematic Diagram**
  - Inputs, Gates, Nets, Outputs

- **Goal**
  - **Validity**: correctness, turnaround time
  - **Performance**: power, timing, cost
  - **Testability**: yield, diagnosis, robustness

Scope: Boolean algebra, switching algebra, logic

- **Boolean Algebra**: multiple-valued logic, i.e. each variable have multiple values.
- **Switching Algebra**: binary logic, i.e. each variable can be either 1 or 0.
- **Boolean Algebra ≠ Switching Algebra**

\[
f = a + b(c + d(e + f(g + h)))
\]
Scope: Boolean algebra, switching algebra, logic

- Boolean Algebra: multiple-valued logic, i.e. each variable have multiple values. **Boolean Algebra ≠ Switching Algebra**

Example: Set={0,A,B,1}, Operators & , + are defined in the following:

\[
\begin{array}{c|cccc}
\& & 0 & A & B & 1 \\
0 & 0 & 0 & 0 & 0 \\
A & 0 & A & 0 & A \\
B & 0 & 0 & B & B \\
1 & 0 & A & B & 1 \\
\end{array}
\]

\[
\begin{array}{c|cccc}
+ & 0 & A & B & 1 \\
0 & 0 & A & B & 1 \\
A & A & A & 1 & 1 \\
B & B & 1 & B & 1 \\
1 & 1 & 1 & 1 & 1 \\
\end{array}
\]

**Scope: Switching Algebra (Binary Values)**

- Typically consider only two discrete values:
  - 1’s and 0’s
  - 1, TRUE, HIGH
  - 0, FALSE, LOW

- 1 and 0 can be represented by specific voltage levels, rotating gears, fluid levels, etc.
- Digital circuits usually depend on specific voltage levels to represent 1 and 0
- **Bit: Binary digit**
Scope: Levels of Logic

- Multiple Level Logic: Many layers of two level logic with some inverters, e.g. 
  \[ (((a+bc)'+ab')'+b'c'+c'd)'+bc'+c'e \]
  (A network of two level logic)
- Two Level Logic: Sum of products, or product of sums, e.g.
  \[ ab + a'c + a'b', \quad (a'+c)(a+b')(a+b+c') \]

Features of Digital Logic Design
- Multiple Outputs
- Don’t care sets

1.2 Boolean Algebra (Review)

George Boole, 1815-1864

- Born to working class parents: Son of a shoemaker
- Taught himself mathematics and joined the faculty of Queen’s College in Ireland.
- Wrote *An Investigation of the Laws of Thought* (1854): systematize Aristotle’s logic
- Introduced binary variables
- Introduced the three fundamental logic operations: AND, OR, and NOT.
Review of Boolean Algebra

Let B be a nonempty set with two 2-input operations, a 1-input operation \( \cdot \) (complement), and two distinct elements 0 and 1. Then B is called a Boolean algebra if the following axioms hold.

- **Associative laws:** \((a+b)+c=a+(b+c)\), \((a \cdot b) \cdot c=a \cdot (b \cdot c)\)
- **Commutative laws:** \(a+b=b+a\), \(a \cdot b=b \cdot a\)
- **Distributive laws:** \(a+(b \cdot c)=(a+b) \cdot (a+c)\), \(a \cdot (b+c)=a \cdot b+a \cdot c\)
- **Identity laws:** \(a+0=a\), \(a \cdot 1=a\)
- **Complement laws:** \(a+a'=1\), \(a \cdot a'=0\)

Review of Boolean Algebra: Duality

<table>
<thead>
<tr>
<th></th>
<th>((a+b)+c=a+(b+c))</th>
<th>((a \cdot b) \cdot c=a \cdot (b \cdot c))</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Associative laws</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Commutative laws</strong></td>
<td>(a+b=b+a)</td>
<td>(a \cdot b=b \cdot a)</td>
</tr>
<tr>
<td><strong>Distributive laws</strong></td>
<td>(a+(b \cdot c)=(a+b) \cdot (a+c))</td>
<td>(a \cdot (b+c)=a \cdot b+a \cdot c)</td>
</tr>
<tr>
<td><strong>Identity laws</strong></td>
<td>(a+0=a)</td>
<td>(a \cdot 1=a)</td>
</tr>
<tr>
<td><strong>Complement laws</strong></td>
<td>(a+a'=1)</td>
<td>(a \cdot a'=0)</td>
</tr>
</tbody>
</table>

Duality: We swap all operators between (+,.) and interchange all elements between (0,1).

**For a theorem if the statement can be proven with the laws of Boolean algebra, then the duality of the statement is also true.**
1.3 Switching functions: Operators and Digital Logic Gates

### AND

<table>
<thead>
<tr>
<th>id</th>
<th>A</th>
<th>B</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ Y = AB \]

Input 0 dominates Y
0 blocks the output
1 passes signal A

### OR

<table>
<thead>
<tr>
<th>id</th>
<th>A</th>
<th>B</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</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>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ Y = A + B \]

Input 1 dominates Y
0 passes signal A
1 blocks the output

### NOT

\[ Y = A' \]

\[ f(a,b,c) = ab + a'b' + (bc + b'c' + ac + ac') \]

Find \( (a,b,c) \), s.t. \( f(a,b,c) = 1 \)

\[ f(a,b,c) = 0 \text{ NPCompl} \]

If \( a = 1 \Rightarrow b = 0 \Rightarrow c = 1 \Rightarrow f = 1 \)

If \( a = 0 \Rightarrow b = 1 \Rightarrow c = 0 \Rightarrow f = 1 \)

\[ m_7 = \text{if } a = b = c = 1 \]

1.3 Switching functions: Operators and Digital Logic Gates

### AND

<table>
<thead>
<tr>
<th>id</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ Y = ABC \]

0 blocks the output
1 passes signal A
For AND, only one row is true (minterm)

### OR

<table>
<thead>
<tr>
<th>id</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ Y = A + B + C \]

0 passes signal A
1 blocks the output
For OR, only one row is false (maxterm)

\[ M_0 = 0 \text{ if } A = B = C = 0 . \]
Switching Expressions vs Logic Diagrams

Switching expression is related to logic implementation

- Switching Expression: #literals, #operators
- Schematic Diagram: #gates, #nets, #pins
Laws and Logic Diagrams

Associativity Laws
\[(A+B) + C = A + (B+C)\]

\[(AB)C = A(BC)\]

Identity Laws
\[A \cdot 1 = A \quad A + 1 = 1\]
\[A \cdot 0 = 0 \quad A + 0 = A\]

Complement Laws
\[A + A' = 1 \quad A \cdot A' = 0\]

Distributive Laws
\[A \cdot (B+C) = A \cdot B + A \cdot C\]

\[A+B \cdot C = (A+B) \cdot (A+C)\]
Switching Expression and Logic

Schematic Diagram:
5 primary inputs
1 primary output
4 gates (3 ANDs, 1 OR)
9 signal nets
12 pins

Boolean Algebra:
5 variables
1 expression
4 operators (3 ANDs, 1 OR)
5 literals

Switching Expression and Logic

Schematic Diagram:
5 primary inputs
4 components (gates)
9 signal nets
12 pins

Boolean Algebra:
5 literals
4 operators

A. #inputs ↔ I. #variables
B. #gates ↔ II. #operators
C. #nets ↔ III. #variables + #operators
D. #pins 12  IV. #literals + 2 #operators - 1
E. None
Example: $f(a, b, c) = ab + a'c + a'b'$

Example: $f(a, b, c) = (a' + c)(a + b')(a + b + c)$
1.4 Handy Tools

Boolean Algebra
• DeMorgan’s Law: Complements
• Consensus Theorem

Switching Logic
• Shannon’s Expansion
• Truth Table
• Karnaugh Map (single output, two level logic)

DeMorgan’s Theorem and Digital Logic

T12. DeMorgan’s Theorem \((A+B)' = A'B'\) \((AB)' = A' + B'\)

- \(Y = (AB)' = A' + B'\)

- \(Y = (A + B)' = A'B'\)
DeMorgan’s Theorem: Bubble Pushing

- Pushing bubbles backward (from the output) or forward (from the inputs) changes the body of the gate from AND to OR or vice versa.
- Pushing a bubble from the output back to the inputs puts bubbles on all gate inputs.

- Pushing bubbles on *all* gate inputs forward toward the output puts a bubble on the output and changes the gate body.

Consensus Theorem

- $AB + AC + B'C$
- $(A+B)(A+C)(B'+C)$

$$= AB + B'C$$
$$= (A+B)(B'+C)$$

The consensus of $AB, B'C$ is: ?

Exercise: to prove the reduction using
(1) Venn Diagrams,
(2) Boolean algebra,
(3) Logic simulation and
(4) Shannon’s expansion
Consensus Theorem: Venn Diagrams

\[ AB + AC + B'C : AB + B'C \]

\[ \begin{array}{ll}
\text{A} & \text{A} \\
\text{B} & \text{B} \\
\text{C} & \text{C} \\
\end{array} \]

Consensus Theorem: Boolean Algebra

- \[ AB + AC + B'C \]
  \[ = AB + B'C \]
  \[ AB + AC + B'C \]
  \[ = AB + AC1 + B'C \]
  \[ = AB + AC(B+ \overline{B'}) + B'C \]
  \[ = AB + AC \cdot \overline{A} + AB'C + B'C \]
  \[ = AB(1 + C) + (A + 1)B'C \]
  \[ = AB + B'C \]

- \( (A + B)(A + C)(B' + C) \)
  \[ = (A + B)(B' + C) \]

\( <25> \)
Consensus Theorem: Logic Simulation

\[ f(A,B,C) = AB + AC + B'C \]
\[ g(A,B,C) = AB + B'C \]

<table>
<thead>
<tr>
<th>Index</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>AB</th>
<th>AC</th>
<th>B'C</th>
<th>f</th>
<th>g</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</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>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</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>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</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>5</td>
<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>6</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>7</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>

\[
ab + ab' + a = a \\
ab + a'b + b = a' + b \\
\underline{ab' + a'b + b'b} = ab' + a'b \\

abc + a'cd \quad \text{bcd} \quad \text{conclusion}
\]
Consensus Theorem (Examples)

Which one is not a consensus of the expressions on the right.

A. B

1. \( A + A' B = A + B \)

B. BC

2. \( A + A' BC = A + BC \)

C. \((B+C)D\)

3. \( A(B+C) + A'D \)

D. BCD

E. BCDE

4. \( ABC + A'D + BCDE = ABC + A'D \)

\[ A + AB = A(1+B) = A \cdot 1 = A \]

Shannon's Expansion

- Shannon's expansion assumes a switching algebra system
- Divide a switching function into smaller functions
- Pick a variable \( x \), partition the switching function into two cases: \( x=1 \) and \( x=0 \)
  - \( f(x,y,z,\ldots) = xf(x=1,y,z,\ldots) + x'f(x=0,y,z,\ldots) \)
- For example
  - \( f(x) = xf(1) + x'f(0) \)
  - \( f(x,y) = xf(1,y) + x'f(0,y) \)