Logic: From Greeks to philosophers to circuits.

COS 116, Spring 2010
Adam Finkelstein
High-level view of self-reproducing program

A

B

Print 0
Print 1
Print 0

\{ \}

Prints binary code of B

\{ \}

Takes binary string on tape, and …

SEE HANDOUT ON COURSE WEB
Recap: Boolean Logic Example

*Ed goes to the party if*  
*Dan does not and Stella does.*

Choose “Boolean variables” for 3 events:

\[
\{ E: \text{Ed goes to party} \quad D: \text{Dan goes to party} \quad S: \text{Stella goes to party} \}
\]

Each is either TRUE or FALSE

\[ E = S \ AND \ (\text{NOT} \ D) \]

Alternately: \[ E = S \ AND \ \bar{D} \]
Three Equivalent Representations

Boolean Expression

\[ E = S \text{ AND } \overline{D} \]

Boolean Circuit

Truth table:
Value of \( E \) for every possible \( D, S \).
\[ \begin{array}{ccc}
D & S & E \\
0 & 0 & 0 \\
0 & 1 & 1 \\
1 & 0 & 0 \\
1 & 1 & 0 \\
\end{array} \]

TRUE = 1; FALSE = 0.
Boolean “algebra”

A \textbf{AND} B written as $A \cdot B$  

$0 \cdot 0 = 0$  

$0 \cdot 1 = 0$  

$1 \cdot 1 = 1$

A \textbf{OR} B written as $A + B$

$0 + 0 = 0$  

$1 + 0 = 1$  

$1 + 1 = 1$

Funny arithmetic
Boolean gates

Output voltage is high if both of the input voltages are high; otherwise output voltage low.

Output voltage is high if either of the input voltages are high; otherwise output voltage low.

Output voltage is high if the input voltage is low; otherwise output voltage high.

(implicit extra wires for power)
Claude Shannon (1916-2001)

Founder of many fields
(circuits, information theory, artificial intelligence…)

With “Theseus” mouse
Combinational circuit

- Boolean gates connected by wires

Wires: transmit voltage (and hence value)

- Important: no cycles allowed
Examples

4-way AND

(Sometimes we use this for shorthand)

More complicated example

Crossed wires that are not connected are sometimes drawn like this.
Combinational circuits and control

“If data has arrived and packet has not been sent, send a signal”

\[ S = D \text{ AND } \neg P \]
Circuits compute functions

- Every combinational circuit computes a Boolean function of its inputs.
Ben Revisited

Ben only rides to class if he overslept, but even then if it is raining he’ll walk and show up late (he hates to bike in the rain). But if there’s an exam that day he’ll bike if he overslept, even in the rain.

B: Ben Bikes
R: It is raining
E: There is an exam today
O: Ben overslept

How to write a boolean expression for B in terms of R, E, O?
Ben’s truth table

<table>
<thead>
<tr>
<th>O</th>
<th>R</th>
<th>E</th>
<th>B</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>0</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>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>1</td>
</tr>
</tbody>
</table>
Truth table → Boolean expression

Use **OR** of all input combinations that lead to TRUE (1)

\[ B = \overline{O} \cdot \overline{R} \cdot \overline{E} + \overline{O} \cdot R \cdot \overline{E} + O \cdot R \cdot E \]

<table>
<thead>
<tr>
<th>O</th>
<th>R</th>
<th>E</th>
<th>B</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>0</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>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>1</td>
</tr>
</tbody>
</table>

Note:
AND, OR, and NOT gates suffice to implement every Boolean function!
Sizes of representations

For $k$ variables:

<table>
<thead>
<tr>
<th>$k$</th>
<th>10</th>
<th>20</th>
<th>30</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2^k$</td>
<td>1024</td>
<td>1048576</td>
<td>1073741824</td>
</tr>
</tbody>
</table>

For an arbitrary function, expect roughly half of $X$’s to be 1 (for 30 inputs roughly 1/2 billion!)

Tools for reducing size:
(a) circuit optimization (b) modular design
Expression simplification

- Some simple rules:
  - $x + \bar{x} = 1$
  - $x \cdot 1 = x$
  - $x \cdot 0 = 0$
  - $x + 0 = x$
  - $x + 1 = x$
  - $x + x = x \cdot x = x$
  - $x \cdot (y + z) = x \cdot y + x \cdot z$
  - $x + (y \cdot z) = (x+y) \cdot (x+z)$

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

De Morgan’s Laws:
\[
\begin{align*}
  x \cdot y &= x + \bar{y} \\
  x + y &= x \cdot \bar{y}
\end{align*}
\]
Simplifying Ben’s circuit

\[ B = O \cdot \overline{R} \cdot \overline{E} + O \cdot \overline{R} \cdot E + O \cdot R \cdot E \]
\[ = O \cdot (\overline{R} \cdot \overline{E} + \overline{R} \cdot E + R \cdot E) \]
\[ = O \cdot (\overline{R} \cdot (\overline{E} + E) + R \cdot E) \]
\[ = O \cdot (\overline{R} + R \cdot E) \]
\[ = O \cdot (\overline{R} + E) \]
Something to think about:
How hard is Circuit Verification?

- Given a circuit, decide if it is “trivial” (no matter the input, it either always outputs 1 or always outputs 0).

- Alternative statement: Decide if there is any setting of the inputs that makes the circuit evaluate to 1.

Time required?
Boole’s reworking of Clarke’s “proof” of existence of God (see handout)

- General idea: Try to prove that Boolean expressions $E_1, E_2, \ldots, E_k$ cannot simultaneously be true

- **Method**: Show $E_1 \cdot E_2 \cdot \ldots \cdot E_k = 0$

- Discussion for after Break: What exactly does Clarke’s “proof” prove? How convincing is such a proof to you?

Also: Do Google search for “Proof of God’s Existence.”
Beyond combinational circuits …

- Need 2-way communication (must allow cycles!)
- Need memory (scratchpad)

![Diagram showing CPU and Ethernet card](image)
Circuit for binary addition?

- Want to design a circuit to add any two \( N \)-bit integers.

\[
\begin{array}{cc}
25 & 11001 \\
+ & 29 \\
\hline
54 & 110110
\end{array}
\]

- Is the truth table method useful for \( N=64 \)?
After Break: Modular Design

Design an N-bit adder using N 1-bit adders

Read:
(a) handout on boolean logic.
(b) handout on Boole’s “proof” of existence of God.