Announcements

- Homework 2 available (Due October 10)

Highlights from last lecture

- Boolean algebra to circuit design
- Boolean algebra
  - a set of elements $B = \{0, 1\}$
  - binary operations $\{+, \cdot\}$
  - and a unary operation $\{\)'\}$
  - such that the following axioms hold:

1. the set $B$ contains at least two elements: $a, b$
2. closure: $a + b \in B$ $a \cdot b \in B$
3. commutativity: $a + b = b + a$ $a \cdot b = b \cdot a$
4. associativity: $a + (b + c) = (a + b) + c$ $a \cdot (b \cdot c) = (a \cdot b) \cdot c$
5. identity: $a + 0 = a$ $a \cdot 1 = a$
6. distributivity: $a + (b \cdot c) = (a + b) \cdot (a + c)$ $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$
7. complementarity: $a + a' = 1$ $a \cdot a' = 0$

A simple example: 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>Carry-out</th>
<th>Sum</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>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>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Partial truth table for carry-out:

- $Cout = A' B Cin + A B' Cin + A B Cin' + A B Cin$

Apply the theorems to simplify expressions

- The theorems of Boolean algebra can simplify expressions
  - e.g., full adder’s carry-out function

- Adding extra terms creates new factoring opportunities
S = A' B' Cin + A' B Cin' + A B' Cin' + A B Cin

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

A 2-bit ripple-carry adder

Mapping truth tables to logic gates
• Given a truth table:
  1. Write the Boolean expression
  2. Minimize the Boolean expression
  3. Draw as gates
  4. Map to available gates

Canonical forms
• Truth table is the unique signature of a Boolean function
• The same truth table can have many gate realizations
  – we’ve seen this already
  – depends on how good we are at Boolean simplification
• Canonical forms
  – standard forms for a Boolean expression
  – we all come up with the same expression

Sum-of-products canonical forms
• Also known as disjunctive normal form
• Also known as minterm expansion
Sum-of-products canonical form (cont)

- Product term (or minterm)
  - ANDed product of literals — input combination for which output is true
  - each variable appears exactly once, true or inverted (but not both)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>minterm</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>A'B'C'</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>A'BC'</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>ABC'</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>ABC</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>ABC</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>ABC</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>AB'C</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>AB'</td>
</tr>
</tbody>
</table>

Product-of-sums canonical form (cont)

- Sum term (or maxterm)
  - ORed sum of literals — input combination for which output is false
  - each variable appears exactly once, true or inverted (but not both)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>maxterm</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>A+B+C</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>A+B'+C</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>A+B+1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>A'+B+C</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>A'+B'+C</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>A+B'+C</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>A+B+1</td>
</tr>
</tbody>
</table>

Predicate Calculus

- **Predicate or Propositional Function**
  - A function that returns a truth value
- “x is a cat”
- “x is prime”
- “student x has taken course y”
- “x > y”
- “x + y = z”

Quantifiers

- \( \forall x \ P(x) \): P(x) is true for every x in the domain
- \( \exists x \ P(x) \): There is an x in the domain for which P(x) is true

Statements with quantifiers

- \( \exists x \ \text{Even}(x) \)
- \( \forall x \ \text{Odd}(x) \)
- \( \forall x \ (\text{Even}(x) \lor \text{Odd}(x)) \)
- \( \exists x \ (\text{Even}(x) \land \text{Odd}(x)) \)
- \( \forall x \ \text{Greater}(x+1, x) \)
- \( \exists x \ (\text{Even}(x) \land \text{Prime}(x)) \)
Statements with quantifiers

- \( \forall x \exists y \text{Greater} \ (y, x) \)
- \( \forall x \exists y \text{Greater} \ (x, y) \)
- \( \forall x \exists y (\text{Greater}(y, x) \land \text{Prime}(y)) \)
- \( \forall x (\text{Prime}(x) \rightarrow (\text{Equal}(x, 2) \lor \text{Odd}(x))) \)
- \( \exists x \exists y (\text{Equal}(x, y + 2) \land \text{Prime}(x) \land \text{Prime}(y)) \)

Goldbach’s Conjecture

- Every even integer greater than two can be expressed as the sum of two primes

Goldbach’s Conjecture

- “There is an odd prime”
- “If \( x \) is greater than two, \( x \) is not an even prime”
- \( \forall x \forall y \forall z ((\text{Equal}(z, x+y) \land \text{Odd}(x) \land \text{Odd}(y)) \rightarrow \text{Even}(z)) \)
- “There exists an odd integer that is the sum of two primes”