Lecture 3: Basic Logic Gates & Boolean Expressions
Professor Peter Cheung
Department of EEE, Imperial College London

(Floyd 3.1-3.5, 4.1)
(Tocci 3.1-3.9)

Points Addressed in this Lecture

• What are the basic logic gates?
• What is Boolean algebra?
• Boolean variables & expressions
• Boolean algebra as a way to write down logic
• Boolean Operators
• Truth tables
• Relationships between logic gates & Boolean expressions

Boolean Algebra

• Digital electronic systems manipulate binary information
• To design such systems we need a convenient mathematical framework
  – useful systems are often too complicated to keep in our head
  – Boolean algebra provides this framework
• Points in a circuit are represented by Boolean Variables
• Boolean algebra allows us to specify relationships between Boolean variables
  – Hence Boolean algebra can be used as a design tool for digital electronic circuits

Boolean Variables

• Boolean variables take the value either 0 or 1 only
  – if a variable doesn't have the value 0, then it must have the value 1
  – if a variable doesn't have the value 1, then it must have the value 0
• In digital electronics:
  – Boolean 0 and 1 correspond to the binary 0 and 1
• In logic:
  – 1 and 0 are sometimes called true and false
• We use symbols to represent Boolean variables
  – just like with ordinary algebra
  – eg: A, B, C, X, Y, Z, etc
  – typically a single character
  – typically upper case
• Three Logic operations: AND, OR, NOT
Boolean Algebra to Describe Logic

• Example: House Heating System

• Principles:
  – set the required temperature using a thermostat
  – turn on heating if temperature lower than required
  – turn off heating if temperature higher than required
  – turn on heating if heating pipes are in danger of freezing

• Implementation:
  – use a manual switch to turn on the house heating
  – use a room thermostat to detect room temperature
  – use a frost thermostat to detect outside temperature (danger of freezing)
  – use a digital electronic circuit to turn the heating on and off 'intelligently'

• Boolean representation: 4 variables H, R, F and S

  – H represents the On/Off switch of the entire heating system
  – H = 1 when the heating system is switched on.

  – R represents the room thermostat
  – R = 1 when the room temperature is lower than required

  – F represents the frost thermostat
  – F = 1 when the external temperature is near freezing

  – S represents the On/Off switch of the boiler
  – S = 1 when heat should be generated by the boiler

\[
S = H \cdot R + F \cdot R
\]

• S should be 1 when (H=1 and R=1) or when (F=1 and R=1)

• In Boolean algebra we use \( \cdot \) for 'and' and + for 'or'

Boolean Operators

– Like ordinary algebra, Boolean algebra allows for operations on its variables

• NOT - Takes the complement (inverse) of a single variable

  – Called 'NOT K' and written \( \overline{K} \)

    – eg: Let K represent a key on a computer keyboard and let K = 1 mean the key is pressed

  – We now have a variable which shows the state of the key:

    – K=1 shows key is pressed
    – K=0 shows key is not pressed

  – If we take the compliment of K we have a variable which also shows the state of the key but in the opposite sense

    – \( \overline{K}=1 \) shows key is not pressed
    – \( \overline{K}=0 \) shows is pressed
Basic Boolean Operators & Logic Gates

- Inverter
- AND Gate
- OR Gate
- Exclusive-OR Gate
- NAND Gate
- NOR Gate
- Exclusive-NOR Gate

Truth Tables

- How a logic circuit’s output depends on the logic levels present at the inputs.

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>A 0 B 0</td>
<td>0 0 x 0</td>
</tr>
<tr>
<td>A 0 B 1</td>
<td>0 1 x 0</td>
</tr>
<tr>
<td>A 1 B 0</td>
<td>1 0 x 0</td>
</tr>
<tr>
<td>A 1 B 1</td>
<td>1 1 x 0</td>
</tr>
</tbody>
</table>

The OR Operation & Gate

- Boolean expression
- Truth table
- 0 = LOW
- 1 = HIGH

The output of an OR gate is HIGH whenever one or more inputs are HIGH

4-input OR Gate

- Symbol or Schematic
- Truth Table & Boolean Expression
- Timing Diagram
Summary of OR operation

- Produce a result of 1 whenever any input is 1. Otherwise 0.
- An OR gate is a logic circuit that performs an OR operation on the circuit's input.
- The expression $x = A + B$ is read as “$x$ equals $A$ OR $B$.”

Example of the use of an OR gate in an Alarm system

Example 2

Example 3
Review Questions

- What is the only set of input conditions that will produce a LOW output for any OR gate?
  - all inputs LOW

- Write the Boolean expression for a six-input OR gate
  - X=A+B+C+D+E+F

- If the A input in previous example is permanently kept at the 1 level, what will the resultant output waveform be?
  - constant HIGH

Summary of the AND operation

- The AND operation is performed the same as ordinary multiplication of 1s and 0s.
- An AND gate is a logic circuit that performs the AND operation on the circuit’s inputs.
- An AND gate output will be 1 only for the case when all inputs are 1; for all other cases the output will be 0.
- The expression \( x=A \cdot B \) is read as “x equals A AND B.”

Review Questions

- What is the only input combination that will produce a HIGH at the output of a five-input AND gate?
  - all 5 inputs = 1

- What logic level should be applied to the second input of a two-input AND gate if the logic signal at the first input is to be inhibited (prevented) from reaching the output?
  - A LOW input will keep the output LOW

- True or false: An AND gate output will always differ from an OR gate output for the same input conditions.
  - False
The NOT Operation & Inverter

Boolean expression

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

Truth table

0 = LOW
1 = HIGH

The output of an inverter is always the complement (opposite) of the input.

The output of a NAND gate is HIGH whenever one or more inputs are LOW.

The output of a NOR gate is LOW whenever one or more inputs are HIGH.

Boolean Operations and Expressions

• Sum (OR)

| 0 + 0 | 0 |
| 0 + 1 | 1 |
| 1 + 0 | 1 |
| 1 + 1 | 1 |

• Product (AND)

| 0 • 0 | 0 |
| 0 • 1 | 0 |
| 1 • 0 | 0 |
| 1 • 1 | 1 |

The output of a NAND gate is HIGH whenever one or more inputs are LOW.

The output of a NOR gate is LOW whenever one or more inputs are HIGH.
Describing logic circuits algebraically

- Any logic circuit, no matter how complex, can be completely described using the three basic Boolean operations: OR, AND, NOT.
- Example: logic circuit with its Boolean expression

\[ x = A \cdot B + C \]

Parentheses

- How to interpret \( A \cdot B + C \)?
  - Is it \( A \cdot B \) ORed with \( C \)? Is it \( A \) ANDed with \( B + C \)?
- Order of precedence for Boolean algebra: AND before OR. Parentheses make the expression clearer, but they are not needed for the case on the preceding slide.
- Therefore the two cases of interpretations are:

\[ x = A \cdot B + C \]

\[ x = (A + B) \cdot C \]

Circuits Contain INVERTERS

- Whenever an INVERTER is present in a logic-circuit diagram, its output expression is simply equal to the input expression with a bar over it.

\[ x = \overline{A + B} \]

More Examples
**Precedence**

1. First, perform all inversions of single terms
2. Perform all operations with parentheses
3. Perform an AND operation before an OR operation unless parentheses indicate otherwise
4. If an expression has a bar over it, perform the operations inside the expression first and then invert the result

---

**Determining output level from a diagram**

![Logic Circuit Diagram]

**More Example**

Implementing Circuits From Boolean Expressions

- When the operation of a circuit is defined by a Boolean expression, we can draw a logic-circuit diagram directly from that expression.
Example

• Draw the circuit diagram to implement the expression

\[ x = (A + B)(\overline{B} + C) \]

Review Question

• Draw the circuit diagram that implements the expression

\[ x = \overline{ABC}(A + D) \]

using gates having no more than three inputs.