MODULE 3: Boolean Algebra and Digital Logic

# Lecture 3.1
# Boolean Algebra

Prepared By:
- Scott F. Midkiff, PhD
- Luiz A. DaSilva, PhD
- Kendall E. Giles, PhD

Electrical and Computer Engineering
Virginia Tech

# Lecture 3.1 Objectives

- Define the AND, OR, and NOT Boolean operations
- Evaluate a Boolean expressions to create a truth table
- Derive a Boolean expression from a truth table
- Enumerate the basic properties of Boolean algebra
- Manipulate Boolean expressions using the basic properties of Boolean algebra

# Computer Systems and Boolean Algebra

- Computer systems – hardware and software – are based on binary (“0” or “1”) logic values
- We need to be able to analyze and manipulate such representations and to realize functions of binary logic values
- Boolean algebra provides a “toolbox” to analyze and manipulate logic expressions
- Logic gates provide a means to realize logic functions in hardware
- Logic functions provide a means to realize logic functions in software

# Logic Values

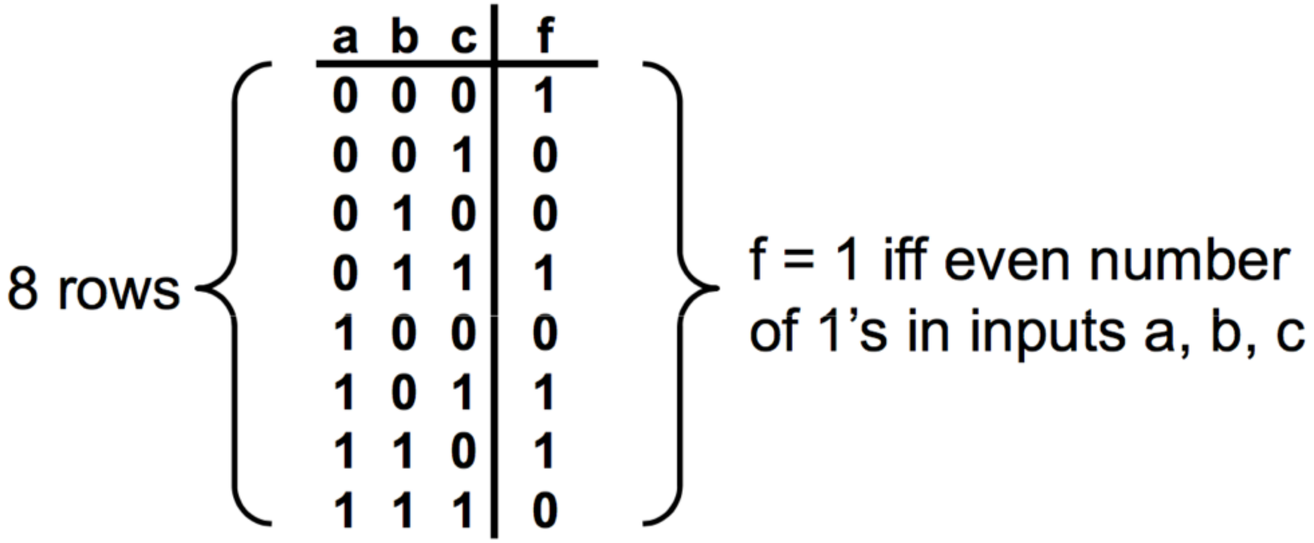
- Binary logic values take one of two discrete values
  - "0" (or "LOW" or "FALSE")
  - "1" (or "HIGH" or "TRUE")
- Such a system is "digital" since values are taken from a finite set, {0, 1} in this case
- Analog values could take values from a continuous set consisting of an infinite number of possible values

# Truth Table

- A truth table is one way to completely specify a logic function that depends only on the current inputs
- Each row of the table is associated a possible combination of input values
  - If there are *n* inputs, then there are $2^n$ possible combinations of input values
  - For example, for $n$=3 there are $2^3$=8 possible combinations:
    $i_2\ i_1\ i_0$ = 000, 001, 010, 011, 100, 101, 110, 111
- The “output” column of the table specifies the value of the output for each possible input combination

# Truth Table Example

- Three inputs: a, b, and c
- One output, f, which should be 1 if and only if (iff) there are an even number of 1's among a, b, and c (otherwise, f is 0)



# Boolean Algebra

- Boolean algebra is an algebra defined using values {0, 1} and basic operations {AND, OR, NOT}
- Basic properties are known
- Utility:
  - Manipulate logic expressions to simplify them
  - Manipulate logic expressions to change the form of logic expressions
  - Manipulate logic expressions to show logical equivalency

# CHECK POINT

As a checkpoint of your understanding, please pause the video and make sure you can do the following:

- If you had *n* = 4 inputs on your truth table, how many possible input combinations are there?

# CHECK POINT

Answer:

- If you had $n = 4$ input variables, then your truth table would have $2^4 = 16$ input combinations or rows.

If you have any difficulties, please review the lecture video before continuing.

# Boolean Operations

- There are three principal Boolean operations
  - AND
  - OR
  - NOT (INVERT)
- There are other operations that are realized from combining these operations
- Any function can be expressed as a combination of AND, OR, and NOT (really just AND and NOT, or just OR and NOT)

# AND Operation

- Output is TRUE (“1”) if and only if all inputs are TRUE (“1”)

| a | b | F=ab |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

$$F = a \bullet b = ab$$

# OR Operation

- Output is TRUE (“1”) if and only if at least one or more input is TRUE (“1”)
- Sometimes called “Inclusive OR”

| a | b | F=a+b |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

$$F = a+b$$

# NOT Operation

- Output is TRUE (“1”) if and only if the input is FALSE (“0”)

| a | $F=\overline{a}$ |
|---|---|
| 0 | 1 |
| 1 | 0 |

$$F = a' = \overline{a}$$

# CHECK POINT

As a checkpoint of your understanding, please pause the video and make sure you can:

- Write out by hand the truth tables for the AND, OR, and NOT operations.

If you have any difficulties, please review the lecture video before continuing.

# Boolean Logic Expressions

- The basic logic expressions (AND, OR, NOT) can be used to construct more complex logic expressions
- Example:

| a | b | c | f |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

$$f(a,b,c) = a'b + c$$

# Basic Properties of Boolean Algebra (1)

- Commutative property (order of variables is not important)
  - AB = BA
  - A+B = B+A
- Distributive property (how a variable is distributed)
  - A (B+C) = AB + AC
  - A + BC = (A+B) (A+C)
- Identity property
  - 1 A = A
  - 0 + A = A

# Basic Properties of Boolean Algebra (2)

- Complement property
  - A A’ = 0
  - A + A’ = 1
- “Zero and one” properties
  - 0 A = 0
  - 1 + A = 1
- Idempotence property
  - A A = A
  - A + A = A

# Basic Properties of Boolean Algebra (3)

- Associative property (order of operations is not important)
  - A (BC) = (AB) C
  - A + (B + C) = (A + B) + C
- Involution property
  - (A')' = A
- DeMorgan's Theorem
  - (AB)' = A' + B'
  - (A+B)' = A' B'

# Basic Properties of Boolean Algebra (4)

- Consensus Theorem
  - AB + A'C + BC = AB + A'C
  - (A+B) (A'+C) (B+C) = (A+B) (A'+C)
- Absorption Theorem
  - A (A+B) = A
  - A + AB = A

# Example Proof

- Let's prove one form of the Absorption Theorem for two variables, A (A+B) = A
- We'll use "perfect induction" (or exhaustive proof)

| A | B | A+B | A (A+B) |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 |

**A = A (A+B)**

# Computational Completeness

- A set of operations is “computationally complete” or “functionally complete” if any Boolean logic expression can be realized using these operations
- Considering gates, a computationally complete set of gates allows any logic function to be realized
- Some computationally complete sets of gates:
  - {AND,OR,NOT}
  - {AND,NOT}
  - {OR,NOT}
  - {NAND}
  - {NOR}

# CHECK POINT

As a checkpoint of your understanding, please pause the video and make sure you can:

- Confirm your understanding of the Absorption Theorem by working out the proof of the theorem's correctness by hand.

If you have any difficulties, please review the lecture video before continuing.

# Summary

- Logic values normally have values:
  - 0 (LOW, FALSE)
  - 1 (HIGH, TRUE)
- Truth tables can be used to specify a logic function
- Boolean logic expressions can also be used to specify a logic function
- Boolean logic functions are made up of simple logic functions, such as {AND, OR, NOT}
- A logic expression can be written that expresses the function of a logic gate realization
- Boolean algebra can be used to manipulate Boolean logic expressions (and, thus, transform the associated logic gate realization)

MODULE 3: Boolean Algebra and Digital Logic

# Lecture 3.1
# Boolean Algebra

Prepared By:
- Scott F. Midkiff, PhD
- Luiz A. DaSilva, PhD
- Kendall E. Giles, PhD

Electrical and Computer Engineering
Virginia Tech