Computer Architecture and Organization

CS 115

Lecture 5

Instructor: **Gerald John M. Sotto**

Last Updated: November 4, 2025

Table of Contents

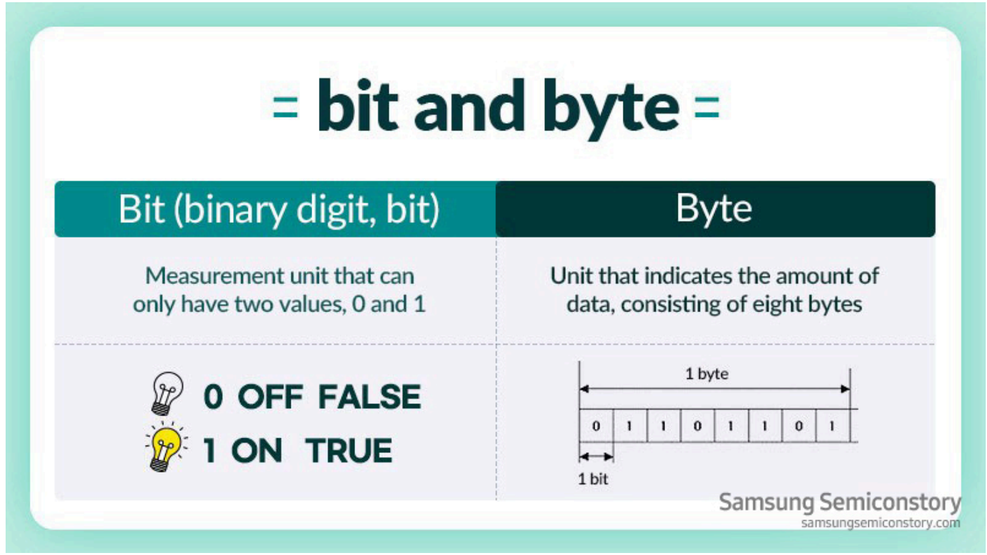
Unit III: Digital Logic and Digital Systems

Unit III: Digital Logic and Digital Systems

Introduction

Introduction

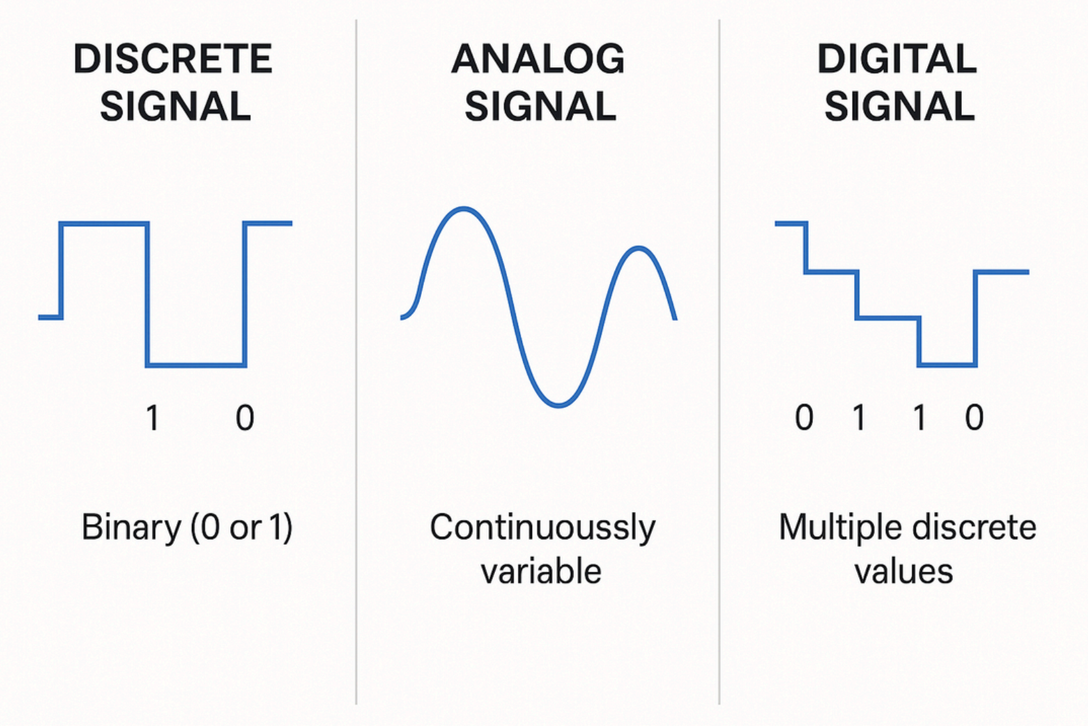
- The Digital Logic part of our course moves from high-level architecture down to the **most basic electronic representation of data:** bits (**0s and 1s**).



- **Digital Logic** is the basis of how computers process information using two distinct voltage levels, representing the binary values **0 (False)** and **1 (True)**.

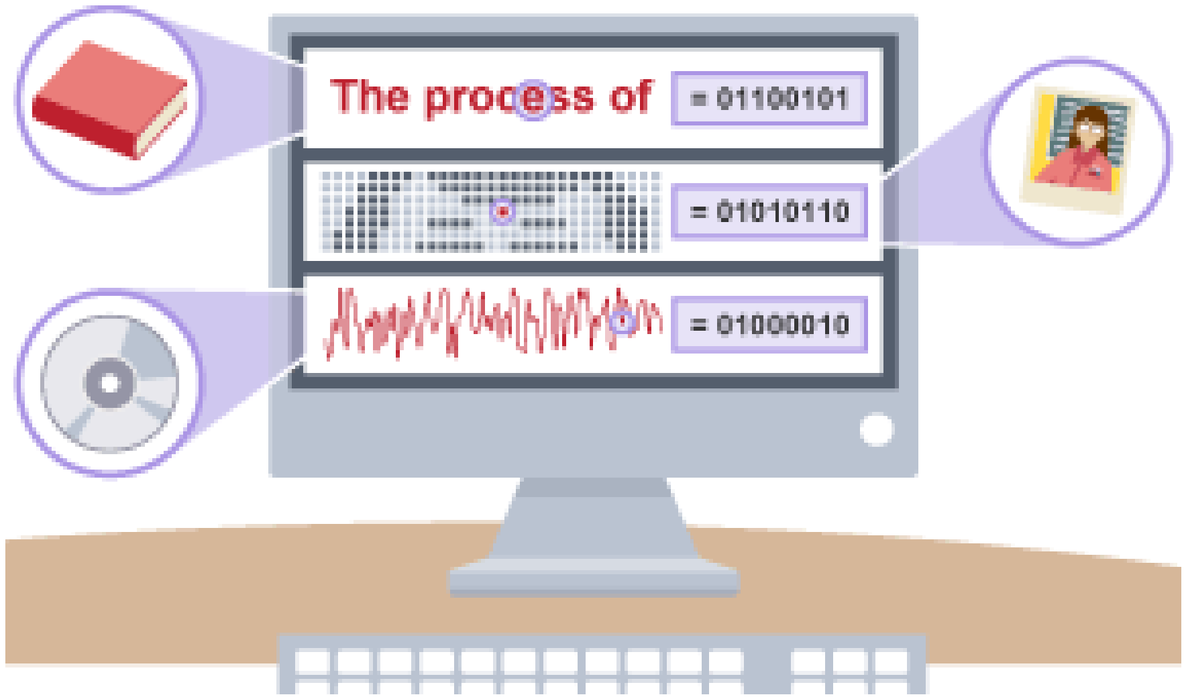
Introduction

- **Digital systems are electronic systems** that operate on these discrete signals, a stark contrast to analog systems which use continuous signals (like a dimmer switch).



Introduction

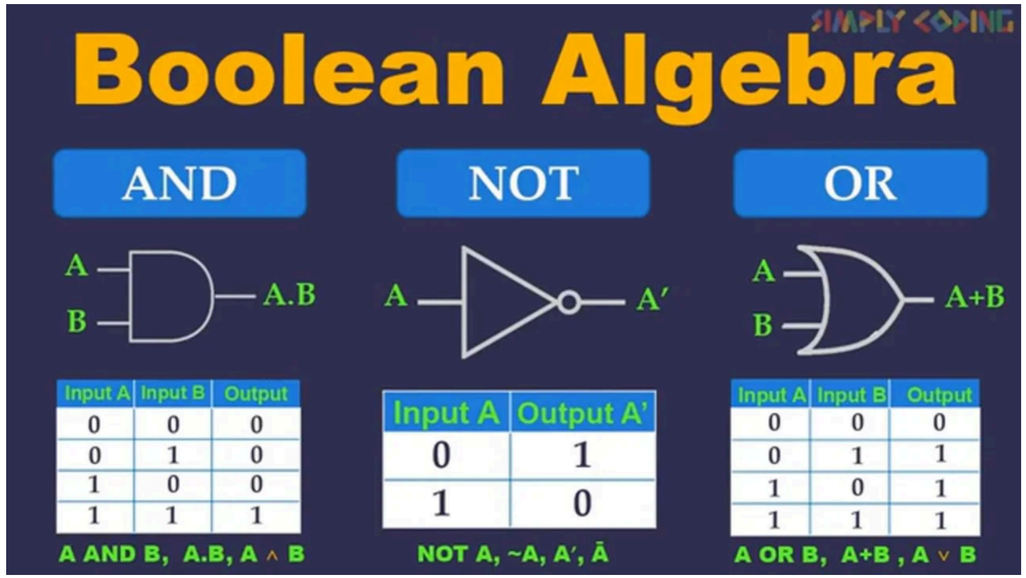
- Every operation a computer performs, from adding two numbers to rendering a complex graphic, is ultimately implemented using **digital logic circuits.** They ensure the **reliability**, **speed**, and **efficiency** of all computing devices.



Boolean Algebra

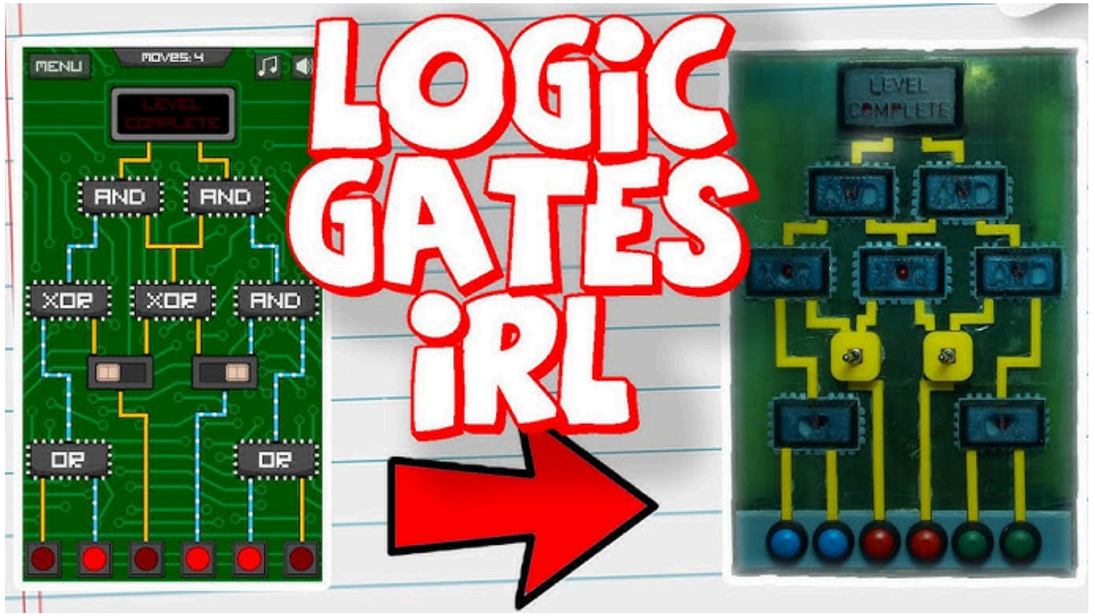
Boolean Algebra

- **Boolean Algebra** is the mathematical foundation for digital logic. It provides the **tools to design and analyze circuits logically**.
- A system of algebra where variables can only have **two possible values:** True (1) or False (0). It defines a set of fundamental operators for manipulating these variables.



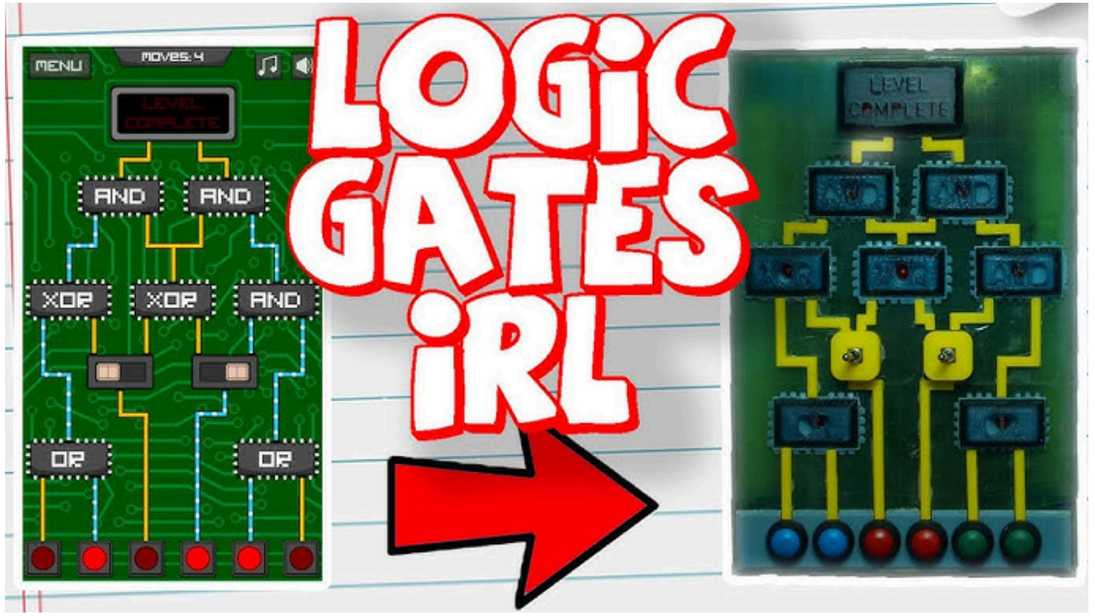
Boolean Algebra

- It allows engineers to **simplify** complex circuits (reducing the number of physical components), which in turn **reduces cost**, **power consumption**, and **increases speed.**



Boolean Algebra

- This directly supports the **binary system** and **data representation** topics. Boolean algebra is the mathematics used to perform the logical operations that the ALU executes (AND, OR, NOT, XOR, etc.).



Boolean Algebra

Let's elaborate on the operators in Boolean Algebra, including the **common** and **derived operators**. We'll use A and B as our inputs.

1. **AND**
   - **Symbols:** A · B, A^B, AB
   - **Logic:** The output is True (1) only if all inputs are True (1).
   - **Analogy:** Think of two light switches connected in a series circuit. You must flip both switch A AND switch B for the light to turn on.
2. **OR**
   - **Symbols:** A+B or AvB
   - **Logic**: The output is True (1) if at least one input is True (1).
   - **Analogy:** Think of two light switches in a parallel circuit. You can flip switch A OR switch B (or both) to turn on the light. It's only off if both are off.

Boolean Algebra

Let's elaborate on the operators in Boolean Algebra, including the **common** and **derived operators**. We'll use A and B as our inputs.

3. **NOT**
   - **Symbols:** $\bar{A}$, A', $\neg A$, or $\sim A$
   - **Logic:** This is a unary operator (it only takes one input). It simply inverts the value. True becomes False, and False becomes True.
   - **Analogy:** It's an inverter. If the input is "on," the output is "off."
4. **NAND (NOT–AND)**
   - **Symbols:** $\overline{A \cdot B}$ or $A \downarrow B$
   - **Logic**: This is the exact opposite of an AND gate. The output is False (0) only when all inputs are True (1). Otherwise, the output is always True (1).
   - **Fun Fact:** NAND gates are "functionally complete," meaning you can build any other logic gate (AND, OR, NOT, etc.) using only NAND gates.

Boolean Algebra

Let's elaborate on the operators in Boolean Algebra, including the **common** and **derived operators**. We'll use A and B as our inputs.

5. **NOR (NOT-OR)**
   - **Symbols:** $\overline{A+B}$, $A \downarrow B$
   - **Logic:** This is the exact opposite of an OR gate. The output is True (1) only when all inputs are False (0). Otherwise, the output is always False (0).
   - **Fun Fact:** Like NAND, the NOR gate is also "functionally complete."
6. **XOR (Exclusive-OR)**
   - **Symbols:** $A \oplus B$
   - **Logic:** The output is True (1) only if the inputs are different. If both inputs are the same (both 0 or both 1), the output is False (0).
   - **Analogy:** This is how a two-way light switch works (like at the top and bottom of a staircase). Flipping either switch inverts the light's current state.

Boolean Algebra

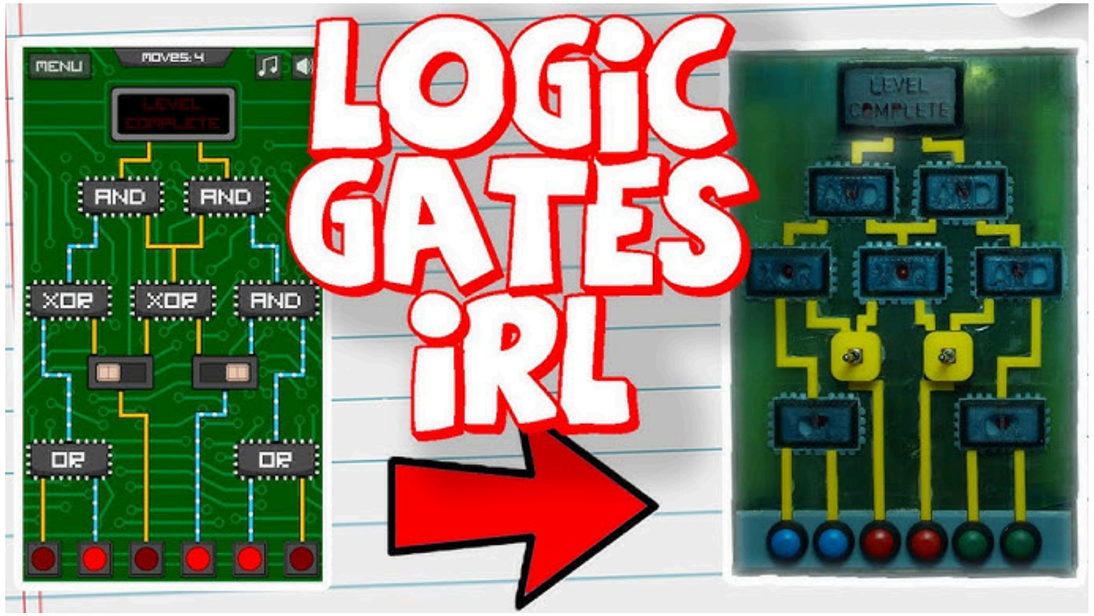
Let's elaborate on the operators in Boolean Algebra, including the **common** and **derived operators.** We'll use A and B as our inputs.

7. **XNOR (Exclusive-NOR / Equivalence)**
   - **Symbols:** $\overline{A \oplus B}$, $A \odot B$
   - **Logic:** This is the exact opposite of XOR. It's also known as the Equivalence gate. The output is True (1) only if the inputs are the same (both 0 or both 1).

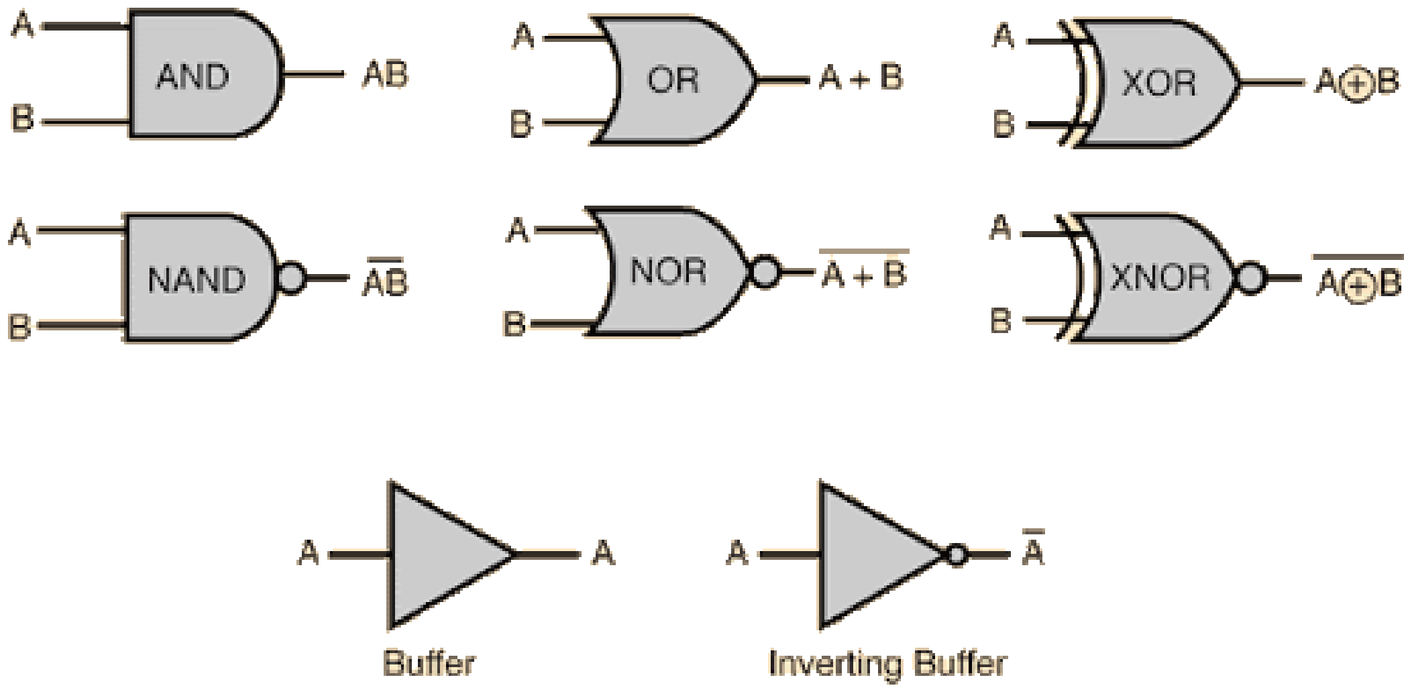
Logic Gates

Logic Gates

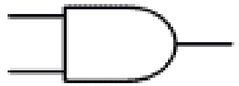
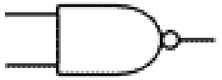
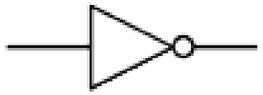
Logic Gates are the **actual electronic components** that implement the fundamental Boolean Algebra operations. They are the **basic building blocks of all digital circuits.**



Logic Gates


A
B
AND
AB
A
B
OR
A + B
A
B
XOR
A⊕B
A
B
NAND
$\overline{AB}$
A
B
NOR
$\overline{A + B}$
A
B
XNOR
$\overline{A \oplus B}$
A
A
Buffer
A
$\overline{A}$
Inverting Buffer

Logic Gates

Logic Gate	Symbol	Description	Boolean
AND		Output is at logic 1 when, and only when all its inputs are at logic 1,otherwise the output is at logic 0.	$X = A \cdot B$
OR		Output is at logic 1 when one or more are at logic 1.If all inputs are at logic 0,output is at logic 0.	$X = A + B$
NAND		Output is at logic 0 when,and only when all its inputs are at logic 1,otherwise the output is at logic 1	$X = \overline{A \cdot B}$
NOR		Output is at logic 0 when one or more of its inputs are at logic 1.If all the inputs are at logic 0,the output is at logic 1.	$X = \overline{A + B}$
XOR		Output is at logic 1 when one and Only one of its inputs is at logic 1. Otherwise is it logic 0.	$X = A \oplus B$
XNOR		Output is at logic 0 when one and only one of its inputs is at logic1.Otherwise it is logic 1. Similar to XOR but inverted.	$X = \overline{A \oplus B}$
NOT		Output is at logic 0 when its only input is at logic 1, and at logic 1 when its only input is at logic 0.That's why it is called and INVERTER	$X = \overline{A}$

End of Presentation

Questions...?

sana po sir madali lang exam hihi