REPORT

Digital electronics and logic design

KHADIJA HABBOUBI

Learning Objectives

Electronic Circuit Analysis

Logic Algebra and Simplification

Digital Signal Systems and Binary Systems

Boolean and Gate Logic

1-Electronic Circuit Analysis

Electronic circuit analysis is key to understanding how electrical systems work. Using Ohm’s Law, the relationship between voltage, current, and resistance is established, explaining how circuits behave in series and parallel. Kirchhoff’s Theorems (KCL and KVL) provide tools to analyze circuit currents and voltages systematically.

Simplification methods like the Thevenin and Norton Theorems reduce complex circuits to simpler equivalents, while the Superposition Principle helps analyze circuits with multiple power sources. These principles enable the design, troubleshooting, and optimization of electronic systems.

Content

Ohm’s Law

Ohm's Law is a fundamental principle in electrical engineering and circuit analysis that describes the relationship between voltage (V), current (I), and resistance (R) in an electrical circuit. It is mathematically expressed as:

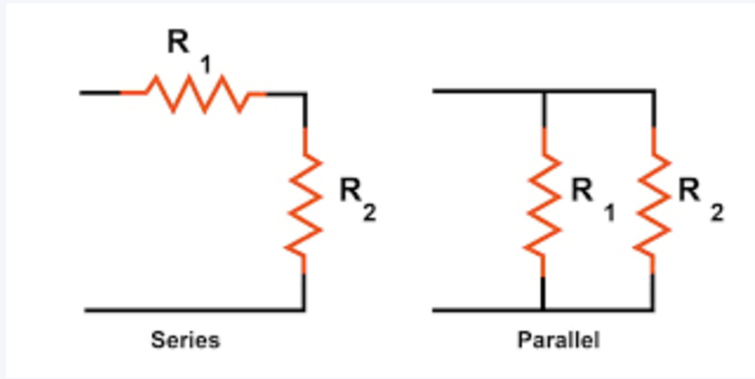
$V = I \cdot R$

Where:

- V is the voltage across the resistor (in volts),

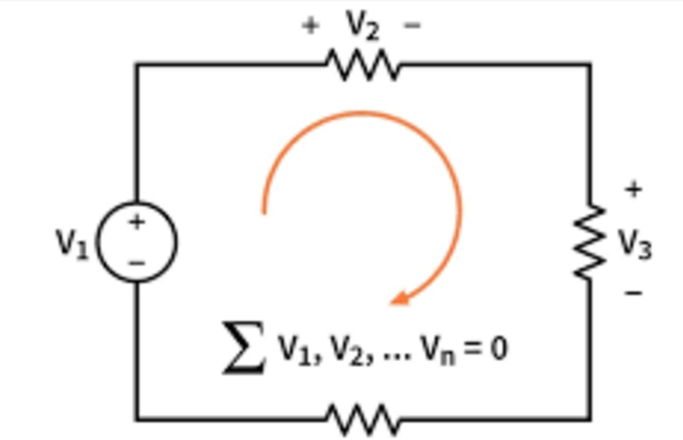
- I is the current flowing through the resistor (in amperes),

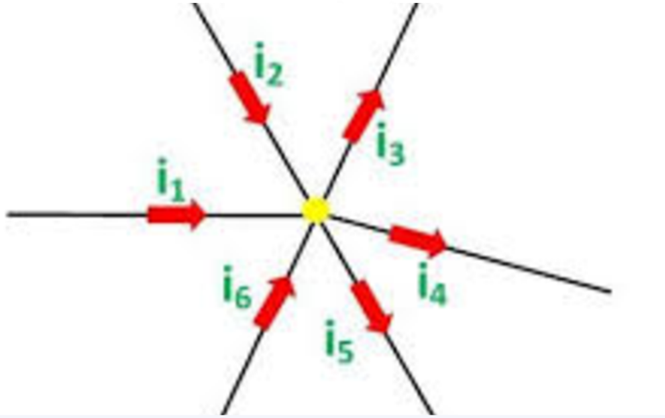
- R is the resistance of the resistor (in ohms).



Kirchhoff's Theorems

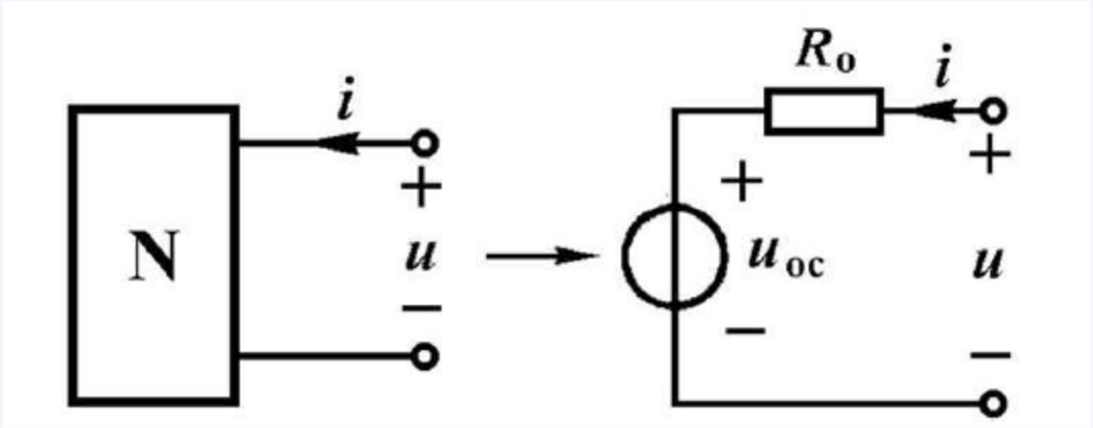
- KCL: The total current entering and leaving a node equals zero.
- KVL: The sum of potential differences (voltage) in a closed loop equals zero.
- Introduces the lumped circuit model, an idealized representation where all parameters (e.g., resistance, reactance) are concentrated at discrete points.
- Discusses the validity and potential misjudgments of Kirchhoff's laws due to testing equipment errors.





Thevenin-Norton Theorem

- Any linear circuit can be simplified into an equivalent voltage source with a series resistance (Thevenin) or a current source with a parallel resistance (Norton).
- Explains source transformation techniques for simplifying circuit analysis.
- Applications include using the Superposition Principle, where the effect of each power source is analyzed independently.



2-Logic Algebra and Simplification

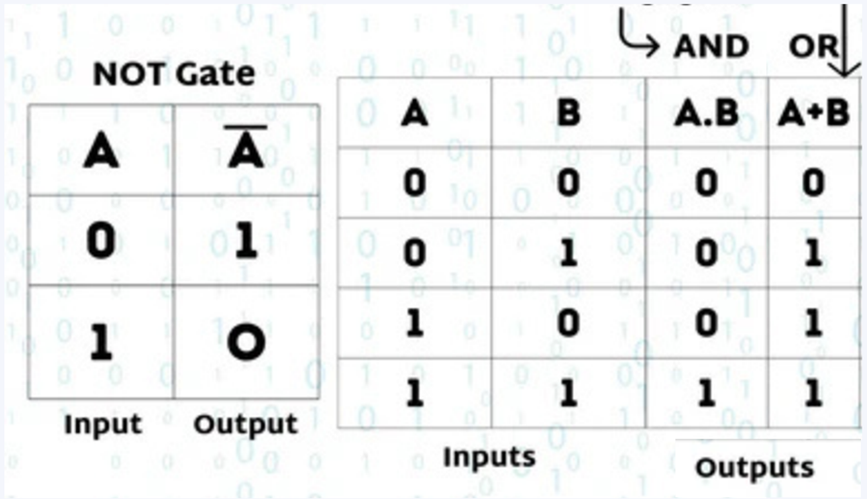
Digital systems process discrete signals, often converted from analog using ADCs. The binary system represents data as 0s and 1s, corresponding to voltage levels, simplifying data storage and transmission.
Binary data is transmitted via serial (bit-by-bit) or parallel (all bits simultaneously) methods. Operations like binary addition and subtraction, along with coding systems like BCD and ASCII, enable efficient data processing and communication. Binary logic is the backbone of modern digital technology.

Content

Content

Binary Logic Variables and Operations

- Logic algebra operates on binary variables (0 and 1) to represent logical states. Key operations include:
- AND (·): Output is 1 only when both inputs are 1.
- OR (+): Output is 1 when any input is 1.
- NOT (¬): Inverts the input (0 becomes 1, and 1 becomes 0).



NOT Gate

A	$\overline{A}$
0	1
1	0

Input Output

↳ AND OR

A	B	A.B	A+B
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Inputs Outputs

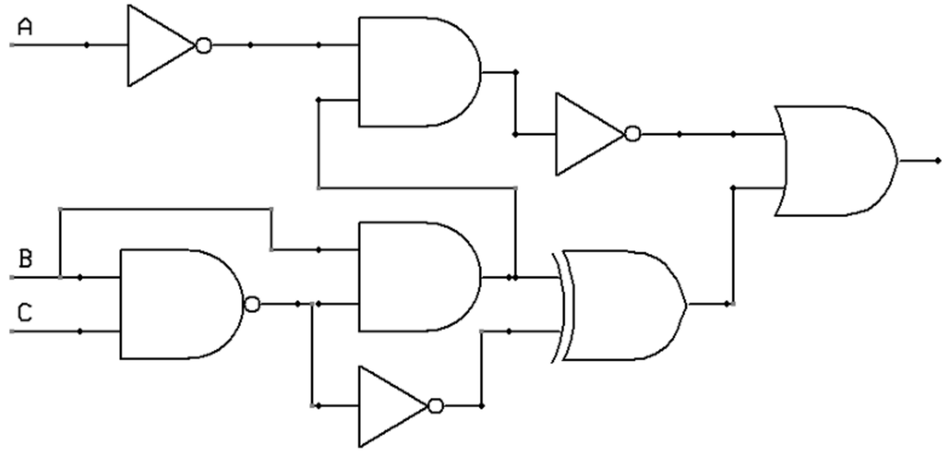
Compound Logic Operations

- NAND: Inverts the output of AND.
- NOR: Inverts the output of OR.
- XOR: Output is 1 when inputs differ.
- XNOR: Output is 1 when inputs are the same.

		NAND	NOR	XOR	XNOR
A	B	$\overline{A.B}$	$\overline{A+B}$	$A \oplus B$	$\overline{A \oplus B}$
0	0	1	1	0	1
0	1	1	0	1	0
1	0	1	0	1	0
1	1	0	0	0	1
Inputs		Outputs			

Logical Functions and Representations

- Truth Tables: Show output for all input combinations.
- Logical Expressions: Use algebra to express relationships
  (e.g., L=AB+BC).
- Logical Diagrams: Represent functions using logic gates (AND, OR, NOT).
- Waveform Diagrams: Display time-based signal outputs.



Fundamental Theorems and Rules of Logical Algebra

- Laws:
- Commutative: A+B=B+A
- Associative: (A+B)+C=A+(B+C)
- Distributive: A(B+C)=AB+AC
- Simplification Rules: Use algebraic laws to simplify logic expressions.

PROPERTY	AND	OR
Commutative	AB = BA	A + B = B + A
Associative	(AB) C = A (BC)	(A + B) + C = A + (B + C)
Distributive	A (B + C) = (AB) + (AC)	A + (BC) = (A + B) (A + C)
Identity	A1 = A	A + 0 = A
Complement	A(A') = 0	A + (A') = 1

Simplification of Logical Functions

- Algebraic Method: Apply laws like distributive and absorption to reduce expressions.
- Karnaugh Map: A graphical method to simplify logic functions by grouping adjacent 1s.

$$
\begin{aligned}
L &= A\overline{A} + AB + B\overline{A} + BB(\text{分配律}) \quad \textbf{Distributive law} \\
&= 0 + AB + B\overline{A} + B \quad (A \cdot \overline{A} = 0,\ A \cdot A = A) \\
&= AB + B\overline{A} + B \quad (A + 0 = A) \\
&= B(A + \overline{A} + 1) \quad [AB + AC = A(B + C)] \\
&= B \cdot 1 = B \quad (A + 1 = A,\ A \cdot 1 = A)
\end{aligned}
$$

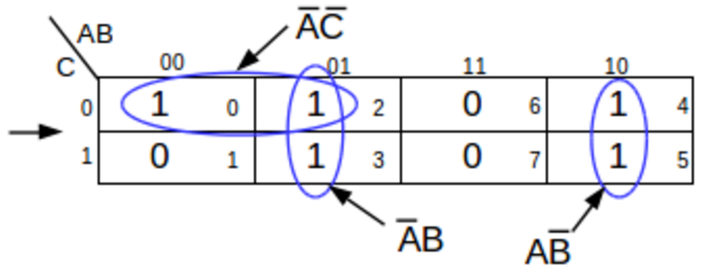
Karnaugh Maps

- Visualize all min terms of variables in a grid.
- Group logically adjacent terms to simplify functions.
- Examples: Two-variable, three-variable, and four-variable maps.

3 Variable Truth Table

	A	B	C	F
0	0	0	0	1
1	0	0	1	0
2	0	1	0	1
3	0	1	1	1
4	1	0	0	1
5	1	0	1	1
6	1	1	0	0
7	1	1	1	0

K-Map



Note : $\bar{A}B + A\bar{B} = A \oplus B$ (Exclusive OR)

$F = \bar{A}\bar{C} + \bar{A}B + A\bar{B}$

$F = \bar{A}\bar{C} + A \oplus B$

3-Digital Signal Systems and Binary Systems

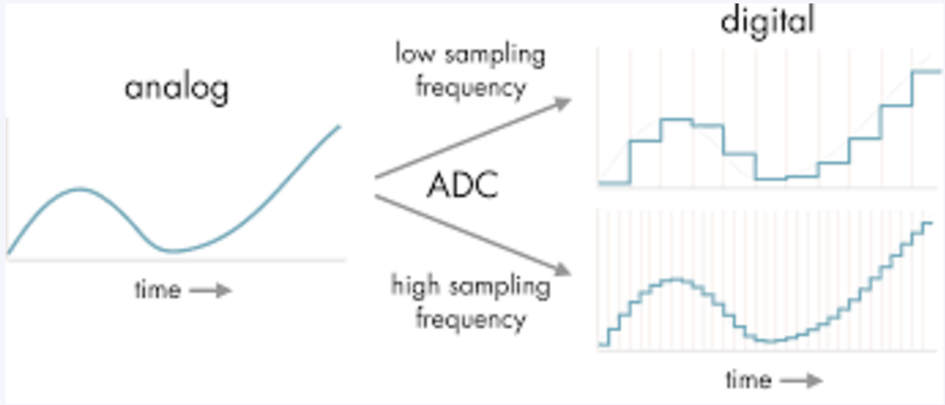
Digital systems process discrete signals, often converted from analog using ADCs. The binary system represents data as 0s and 1s, corresponding to voltage levels, simplifying data storage and transmission.
Binary data is transmitted via serial (bit-by-bit) or parallel (all bits simultaneously) methods. Operations like binary addition and subtraction, along with coding systems like BCD and ASCII, enable efficient data processing and communication. Binary logic is the backbone of modern digital technology.

Content

Content

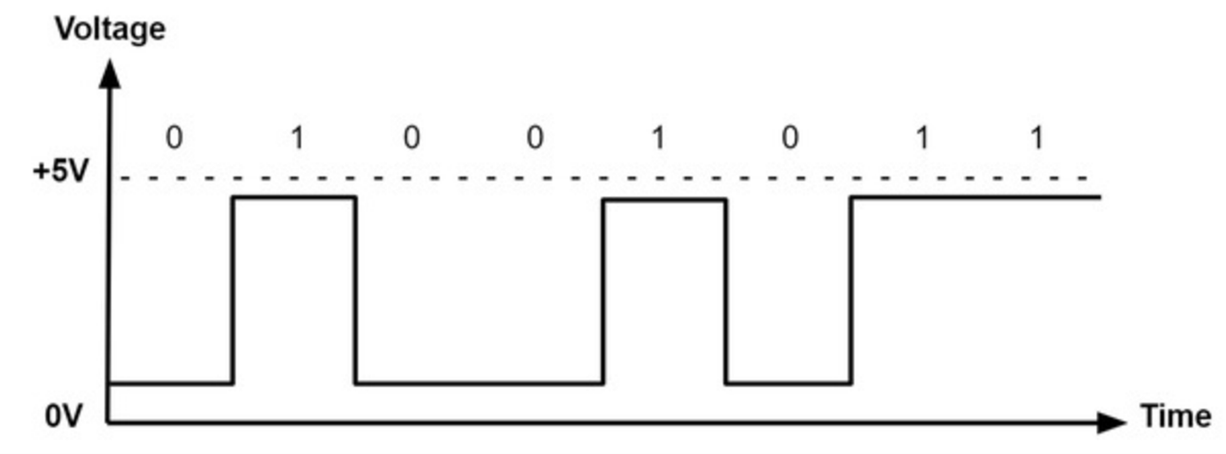
Introduction to Digital and Analog Signals

- Analog Signals: Continuous in time and value (e.g., sine waves).
- Digital Signals: Discrete in time and value, easier to store, analyze, and transmit.
- Analog-to-Digital Conversion (ADC): Converts continuous signals to discrete digital signals using techniques like sampling and quantization.



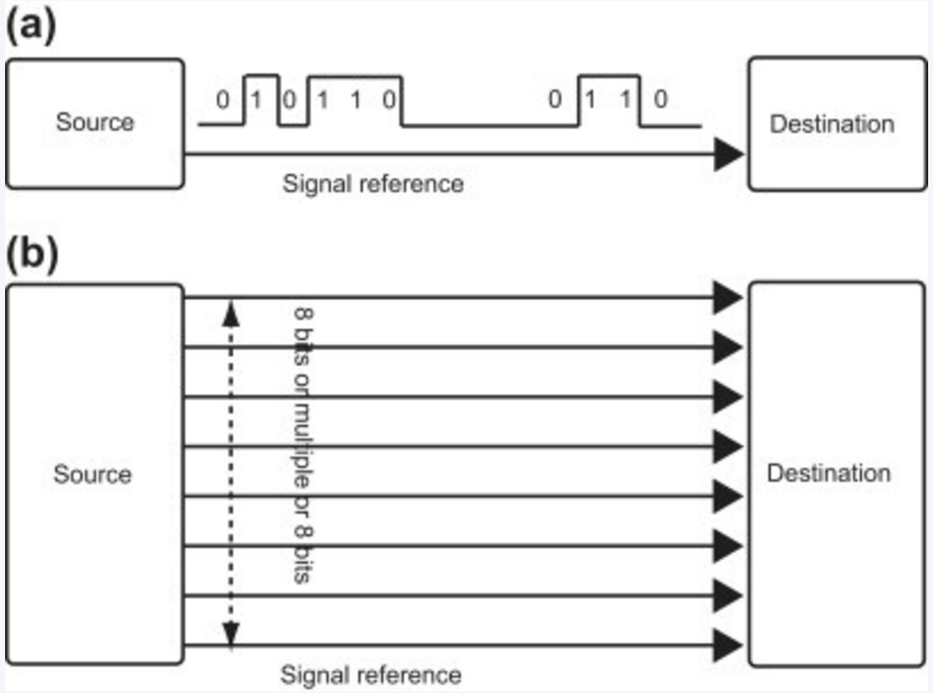
Binary Systems

- Binary logic uses 0 and 1 to represent states.
- Logic levels correspond to voltage values (e.g., +5V for logic 1 and 0V for logic 0).
- Digital waveforms represent binary values graphically over time.



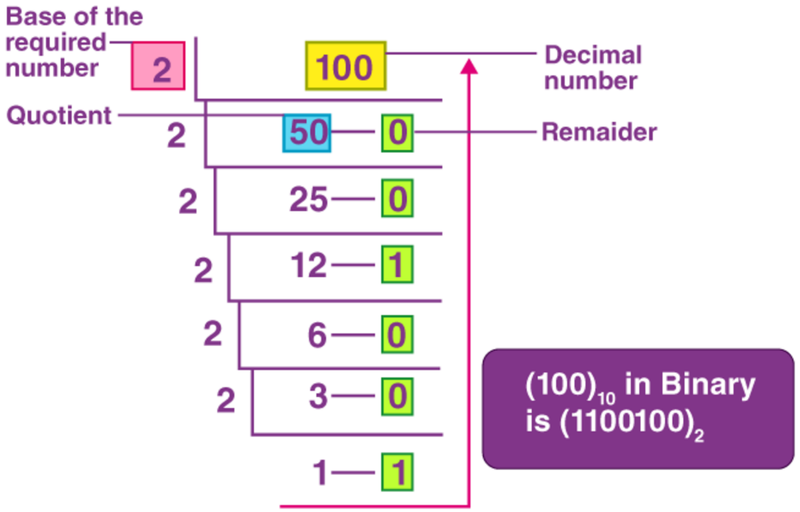
Binary Data Transmission

- Serial Transmission: Sends data bits sequentially, requiring fewer lines but slower transmission.
- Parallel Transmission: Sends all bits simultaneously, offering faster transmission but requiring more lines and complex hardware.



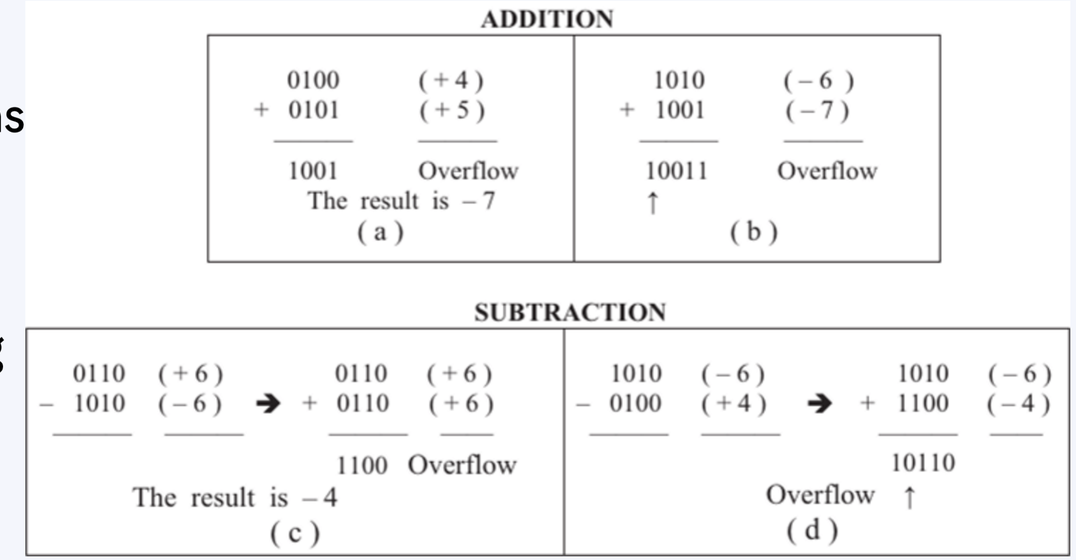
Conversions Between Decimal and Binary

- Integer Conversion: Use the "divide by 2" method until the quotient is 0, then reverse the remainders.
- Fractional Conversion: Multiply the fractional part by 2 iteratively, recording the integer parts until the desired precision is reached.



Binary Arithmetic Operations

- Unsigned Binary: Basic addition and subtraction with carry or borrow.
- Signed Binary: Uses the most significant bit as the sign (0 for positive, 1 for negative).
- Complement Codes: Enable arithmetic operations with negative numbers by flipping bits and adding 1.
- Overflow occurs when results exceed the bit limit and can be resolved through bit expansion.



Binary Codes

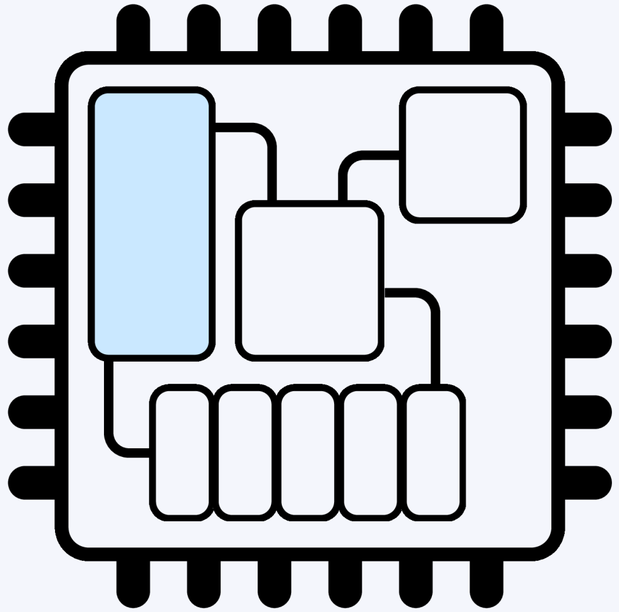
- BCD Codes: Represent decimal digits in binary (e.g., 8421 and 2421 codes).
- Gray Codes: Designed for minimal transition between states, where only one bit changes between consecutive numbers.
- ASCII: Encodes characters into binary for use in digital systems (e.g., keyboards).

Decimal	Binary	Gray	BCD
0	0	0000	0000
1	1	0001	0001
2	10	0011	0010
3	11	0010	0011
4	100	0110	0100
5	101	0111	0101
6	110	0101	0110
7	111	0100	0111
8	1000	1100	1000
9	1001	1101	1001

4-Boolean and Gate Logic

The brain of any computer is the CPU (central processing unit). The CPU fetches instructions from main memory and executes them.

The ALU (arithmetic logic unit) is the component within the CPU where logical decisions are made. This decision making process is called Boolean logic.



Content

Understand the role of Boolean logic in computer systems

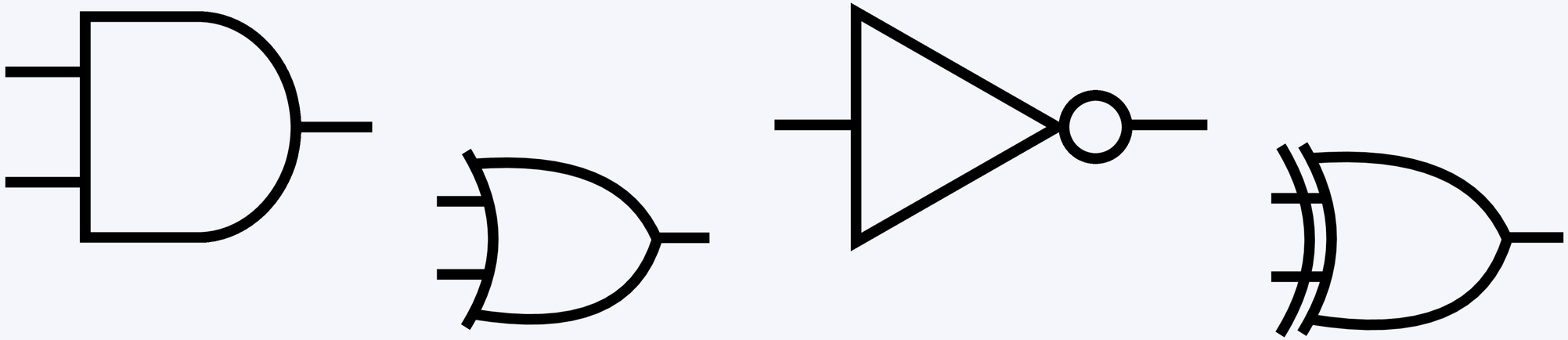
Identify common logic gates: AND, OR, NOT, XOR

Interpret truth tables for logic gates and logic circuits

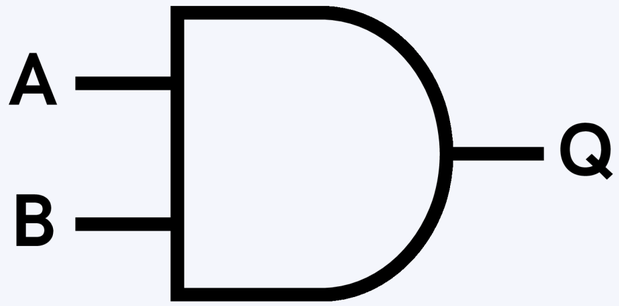
Logic Gates

Boolean logic is represented using logic gate symbols.

Logic gates use Boolean logic to take one or more binary inputs and produce a single binary output.



AND Gate

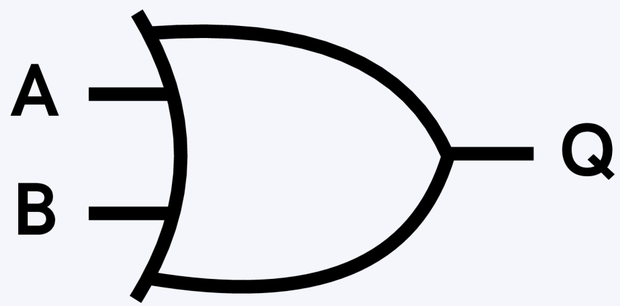


Both inputs must be true for a true output.

Input		Output
A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1

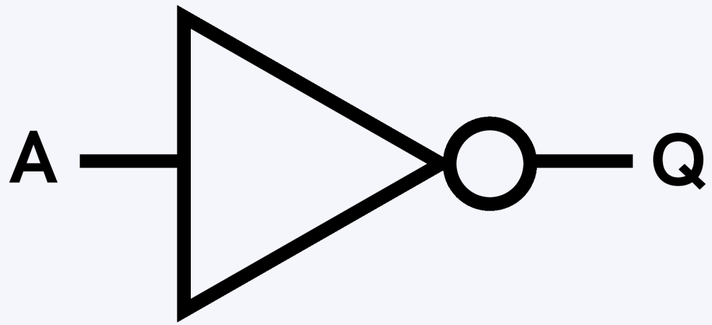
OR Gate

Input		Output
A	B	Q
0	0	0
0	1	1
1	0	1
1	1	1



Only one needs to be true for a true output.

NOT Gate

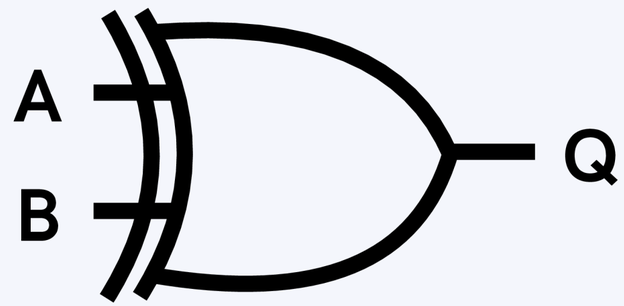


A single input is reversed for a single ouput.

Input	Output
A	Q
0	1
1	0

XOR Gate

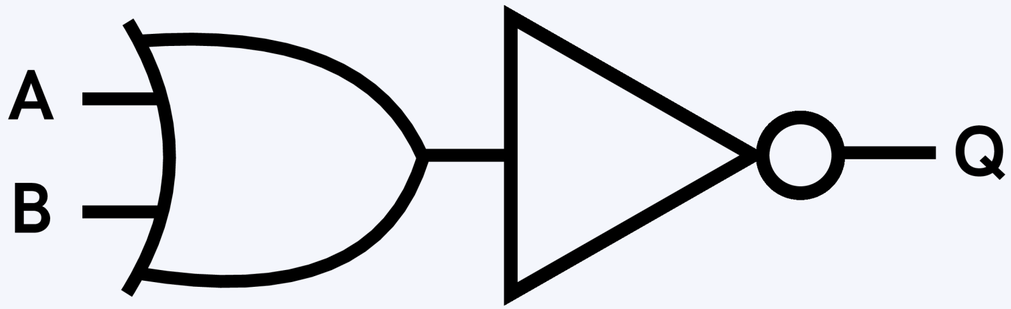
Input		Output
A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0



Only one input can be true for a true output.

Logic Circuits

Multiple logic gates can be combined to create a logic circuit. In this example, the output of the OR gate is reversed through the NOT gate.



Input		Output
A	B	Q
0	0	1
0	1	0
1	0	0
1	1	0

Logic Circuits

A	B	C	D	E	Q
0	0	0	1	0	0
1	0	0	0	0	0
1	1	0	0	0	0
1	1	1	0	0	1
0	0	1	1	0	1
0	1	1	1	1	0
1	0	1	0	0	1
0	1	0	1	1	1