



TECNICATURA SUPERIOR EN

**Telecomunicaciones**

---

**Electrónica Microcontrolada**

El Algebra de Boole y los Circuitos Digitales

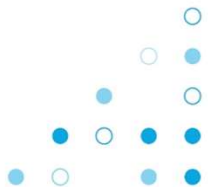
# Estructuras Algebraicas – Conceptos Básicos

## Conjuntos

- Un conjunto es una colección de objetos distintos llamados elementos
- Los elementos pueden ser números, letras, figuras geométricas, etc.
- Ejemplo: Conjunto de números naturales  $N = \{0, 1, 2, 3, \dots\}$

## Operaciones binarias

- Operaciones que combinan dos elementos de un conjunto y producen un elemento del mismo conjunto
- Ejemplos: suma, resta, multiplicación y división en el conjunto de números enteros
- Notación:  $a * b$ , donde  $a$  y  $b$  son elementos del conjunto y  $*$  representa la operación binaria



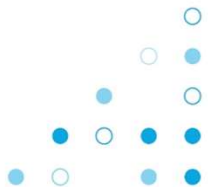
# Estructuras Algebraicas – Conceptos Básicos

## Leyes y propiedades

- Las operaciones en estructuras algebraicas siguen ciertas leyes y propiedades
- Ejemplos de propiedades: conmutativa, asociativa, distributiva, elemento neutro, elemento inverso
- Estas propiedades permiten trabajar con los elementos de la estructura de manera sistemática y predecible

## Estructura algebraica

- Una estructura algebraica es un conjunto acompañado de una o más operaciones binarias que cumplen ciertas propiedades
- Ejemplos de estructuras algebraicas: grupos, anillos, campos
- Las estructuras algebraicas ayudan a comprender y generalizar conceptos matemáticos



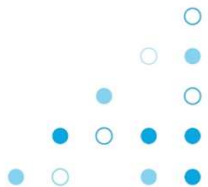
# Estructuras Algebraicas – Conceptos Básicos

## Importancia de las Estructuras Algebraicas

Generalización de conceptos matemáticos: las estructuras algebraicas permiten analizar y comprender patrones y propiedades comunes en diferentes áreas de las matemáticas

Resolución de problemas: al aplicar las propiedades y leyes de las estructuras algebraicas, podemos desarrollar métodos para resolver problemas de manera eficiente y efectiva

Aplicaciones en ciencias de la computación: las estructuras algebraicas desempeñan un papel fundamental en la teoría de la computación, la criptografía, la codificación de información y otros campos

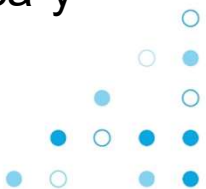


# Introducción al Álgebra de Boole

El álgebra de Boole fue desarrollada por el matemático inglés George Boole en el siglo XIX como una herramienta para el análisis de la lógica proposicional. Boole creó un sistema de símbolos y operaciones matemáticas que permitía la manipulación de proposiciones lógicas, simplificando su análisis y resolución.

La importancia del álgebra de Boole para la electrónica digital y la teoría de circuitos lógicos se debe a la capacidad de simplificar las expresiones booleanas y permitir el diseño de sistemas lógicos complejos. Esto se debe a que el álgebra de Boole utiliza únicamente dos valores (1 y 0) para representar estados lógicos, lo que facilita la manipulación de las expresiones y su análisis.

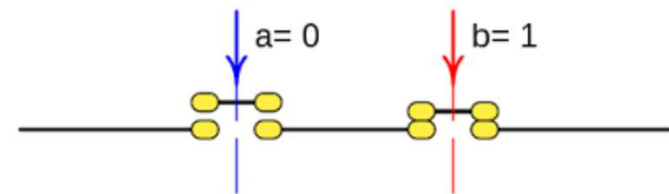
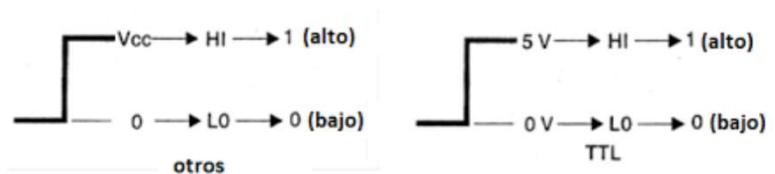
En la actualidad, el álgebra de Boole se utiliza en una amplia variedad de aplicaciones, como en la teoría de la información, en la programación de computadoras, en la inteligencia artificial, en la teoría de sistemas, en la robótica y en muchas otras áreas.



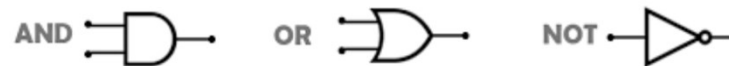
# Introducción al Álgebra de Boole

Álgebra de Boole: estructura algebraica desarrollada por George Boole en el siglo XIX

Valores binarios: el álgebra de Boole opera con dos valores, 0 y 1, que representan falsedad y verdad, respectivamente



Operaciones básicas: AND, OR y NOT



Aplicaciones: circuitos digitales, lógica computacional, búsqueda en bases de datos y más



# Introducción al Álgebra de Boole

## Operaciones Básicas del Álgebra de Boole

### **AND (Conjunción), Notación: $a \wedge b$**

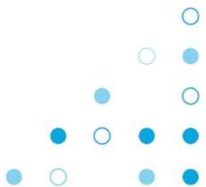
Función: a AND b es verdadero (1) si ambos a y b son verdaderos (1); en cualquier otro caso, es falso (0)

### **OR (Disyunción), Notación: $a \vee b$**

Función: a OR b es verdadero (1) si al menos uno de a o b es verdadero (1); en caso contrario, es falso (0)

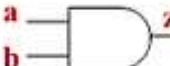





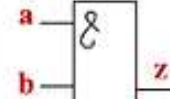
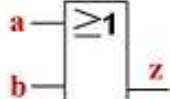
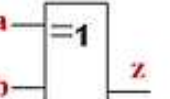
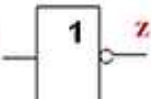

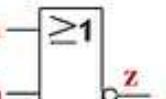
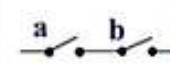
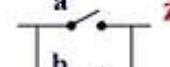
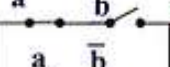


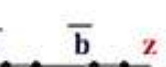
### **NOT (Negación), Notación: $\neg a$**

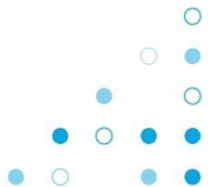
Función: NOT a es verdadero (1) si a es falso (0) y viceversa



# Introducción al Álgebra de Boole

## FUNCIONES LÓGICAS BÁSICAS

NOMRE	AND - Y	OR - O	XOR O-exclusiva	NOT Inversor	NAND	NOR																																																																																	
SÍMBOLO																																																																																							
SÍMBOLO																																																																																							
TABLA DE VERDAD	<table><tr><th>a</th><th>b</th><th>z</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	a	b	z	0	0	0	0	1	0	1	0	0	1	1	1	<table><tr><th>a</th><th>b</th><th>z</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	a	b	z	0	0	0	0	1	1	1	0	1	1	1	1	<table><tr><th>a</th><th>b</th><th>z</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	a	b	z	0	0	0	0	1	1	1	0	1	1	1	0	<table><tr><th>a</th><th>z</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	a	z	0	1	1	0	<table><tr><th>a</th><th>b</th><th>z</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	a	b	z	0	0	1	0	1	1	1	0	1	1	1	0	<table><tr><th>a</th><th>b</th><th>z</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	a	b	z	0	0	1	0	1	0	1	0	0	1	1	0
a	b	z																																																																																					
0	0	0																																																																																					
0	1	0																																																																																					
1	0	0																																																																																					
1	1	1																																																																																					
a	b	z																																																																																					
0	0	0																																																																																					
0	1	1																																																																																					
1	0	1																																																																																					
1	1	1																																																																																					
a	b	z																																																																																					
0	0	0																																																																																					
0	1	1																																																																																					
1	0	1																																																																																					
1	1	0																																																																																					
a	z																																																																																						
0	1																																																																																						
1	0																																																																																						
a	b	z																																																																																					
0	0	1																																																																																					
0	1	1																																																																																					
1	0	1																																																																																					
1	1	0																																																																																					
a	b	z																																																																																					
0	0	1																																																																																					
0	1	0																																																																																					
1	0	0																																																																																					
1	1	0																																																																																					
EQUIVALENTE EN CONTACTOS																																																																																							
AXIOMA	$z = a \cdot b$	$z = a + b$	$z = \bar{a} \cdot b + a \cdot \bar{b}$	$z = \bar{a}$	$z = \overline{a \cdot b}$	$z = \overline{a + b}$																																																																																	





# Introducción al Álgebra de Boole

## Leyes y Propiedades del Álgebra de Boole

Ley de idempotencia:

$$a \wedge a = a$$

$$a \vee a = a$$

Ley de conmutatividad:

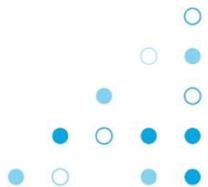
$$a \wedge b = b \wedge a$$

$$a \vee b = b \vee a$$

Ley de asociatividad:

$$(a \wedge b) \wedge c = a \wedge (b \wedge c)$$

$$(a \vee b) \vee c = a \vee (b \vee c)$$



# Introducción al Álgebra de Boole

## Leyes y Propiedades del Álgebra de Boole

Ley de distributividad:

$$a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$$

$$a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$$

Ley de complemento:

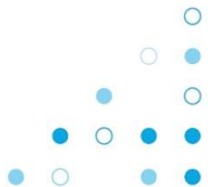
$$a \wedge \neg a = 0$$

$$a \vee \neg a = 1$$

Ley de absorción:

$$a \wedge (a \vee b) = a$$

$$a \vee (a \wedge b) = a$$



# Introducción al Álgebra de Boole

## Aplicaciones del Álgebra de Boole

### Teoría de conjuntos:

El álgebra de Boole puede utilizarse para representar y razonar sobre conjuntos y sus relaciones

Operaciones de conjuntos: intersección (AND), unión (OR), complemento (NOT)

### Lógica:

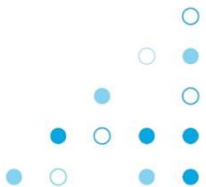
El álgebra de Boole proporciona una base matemática sólida para la lógica proposicional

Razonamiento lógico: mediante la manipulación de proposiciones y sus relaciones utilizando las leyes del álgebra de Boole, se pueden determinar conclusiones válidas y consistentes

### Electrónica digital:

Diseño de circuitos digitales: el álgebra de Boole permite describir y analizar el comportamiento de circuitos digitales y sistemas electrónicos

Optimización de circuitos: mediante la simplificación de expresiones booleanas, se puede diseñar circuitos más eficientes y compactos



# Circuitos Digitales

## Puertas Lógicas

Puertas lógicas: componentes fundamentales de los circuitos digitales que realizan operaciones booleanas

### Tipos de puertas lógicas:

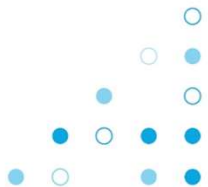
Puerta AND: realiza la operación AND del álgebra de Boole

Puerta OR: realiza la operación OR del álgebra de Boole

Puerta NOT: realiza la operación NOT del álgebra de Boole







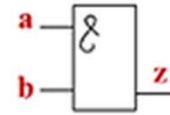
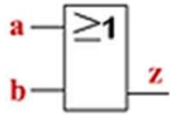
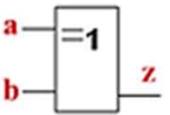
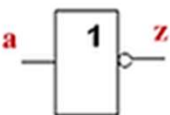

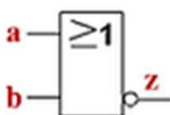
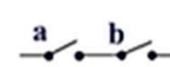
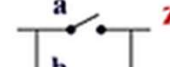
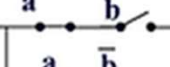
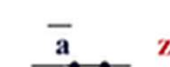
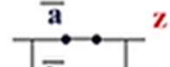
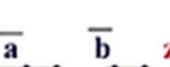
Puertas NAND, NOR, XOR, XNOR: combinaciones y variaciones de las puertas básicas

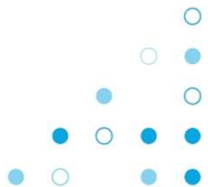
Símbolos y tablas de verdad: cada puerta lógica tiene un símbolo gráfico y una tabla de verdad que describe su funcionamiento



# Circuitos Digitales

## FUNCIONES LÓGICAS BÁSICAS

NOMRE	AND - Y	OR - O	XOR O-exclusiva	NOT Inversor	NAND	NOR																																																																																	
SÍMBOLO																																																																																							
SÍMBOLO																																																																																							
TABLA DE VERDAD	<table><tr><th>a</th><th>b</th><th>z</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	a	b	z	0	0	0	0	1	0	1	0	0	1	1	1	<table><tr><th>a</th><th>b</th><th>z</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	a	b	z	0	0	0	0	1	1	1	0	1	1	1	1	<table><tr><th>a</th><th>b</th><th>z</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	a	b	z	0	0	0	0	1	1	1	0	1	1	1	0	<table><tr><th>a</th><th>z</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	a	z	0	1	1	0	<table><tr><th>a</th><th>b</th><th>z</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	a	b	z	0	0	1	0	1	1	1	0	1	1	1	0	<table><tr><th>a</th><th>b</th><th>z</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	a	b	z	0	0	1	0	1	0	1	0	0	1	1	0
a	b	z																																																																																					
0	0	0																																																																																					
0	1	0																																																																																					
1	0	0																																																																																					
1	1	1																																																																																					
a	b	z																																																																																					
0	0	0																																																																																					
0	1	1																																																																																					
1	0	1																																																																																					
1	1	1																																																																																					
a	b	z																																																																																					
0	0	0																																																																																					
0	1	1																																																																																					
1	0	1																																																																																					
1	1	0																																																																																					
a	z																																																																																						
0	1																																																																																						
1	0																																																																																						
a	b	z																																																																																					
0	0	1																																																																																					
0	1	1																																																																																					
1	0	1																																																																																					
1	1	0																																																																																					
a	b	z																																																																																					
0	0	1																																																																																					
0	1	0																																																																																					
1	0	0																																																																																					
1	1	0																																																																																					
EQUIVALENTE EN CONTACTOS																																																																																							
AXIOMA	$z = a \cdot b$	$z = a + b$	$z = \bar{a} \cdot b + a \cdot \bar{b}$	$z = \bar{a}$	$z = \overline{a \cdot b}$	$z = \overline{a + b}$																																																																																	



# Circuitos Digitales

## Circuitos Secuenciales y Combinacionales

### **Circuitos combinacionales:**

Circuitos digitales cuya salida depende únicamente de las entradas actuales

No tienen memoria: la salida no depende de estados previos

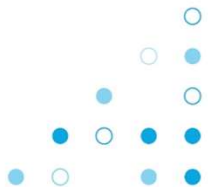
Ejemplos: decodificadores, multiplexores, sumadores

### **Circuitos secuenciales:**

Circuitos digitales cuya salida depende tanto de las entradas actuales como de estados previos

Tienen memoria: almacenan información temporalmente en elementos llamados flip-flops o biestables

Ejemplos: contadores, registros, máquinas de estados finitos





# ¡Muchas gracias!