Álgebra Booleana

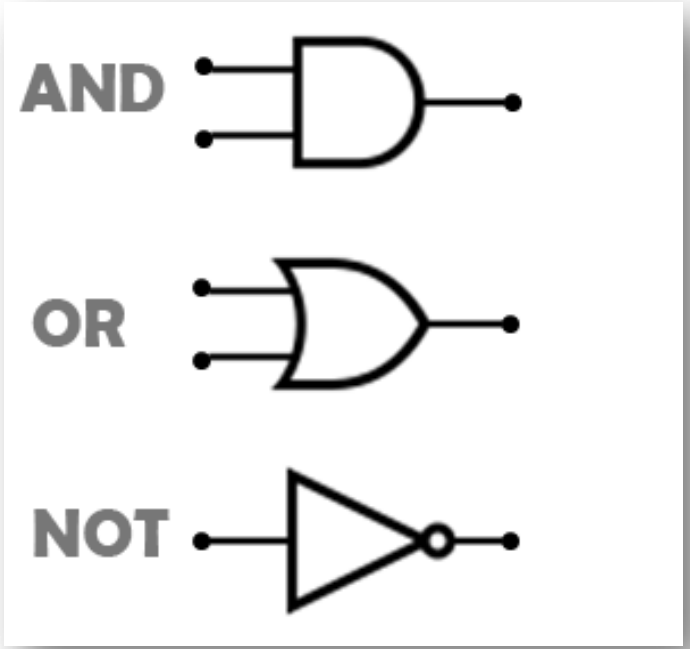
- **Fundamentos Lógicos de la Computación**

Asignatura: Mátematicas Discretas

Estudiante: Javier Guarnizo

Grupo: NeoCore

Fecha: 25 de noviembre del 2025



¿Qué es el Álgebra Booleana?

Definición Formal

Sistema matemático formalizado por George Boole en 1854 para analizar y simplificar expresiones lógicas. Constituye la base matemática de toda la lógica digital moderna.

Base Binaria

Opera exclusivamente con dos valores: **Verdadero (1)** y **Falso (0)**. Esta dualidad es fundamental para la implementación en circuitos electrónicos.

Variables Booleanas

Representan una condición o estado que solo puede ser 1 o 0. Se denotan típicamente con letras mayúsculas (A, B, C, etc.).

Relación con la Lógica Proposicional

El Álgebra Booleana es la formalización algebraica de la lógica proposicional. Las proposiciones (p, q, r) se mapean directamente a variables booleanas (A, B, C), permitiendo manipular la lógica mediante operaciones algebraicas.

Operaciones Booleanas Fundamentales

AND (Conjunción)

Símbolo: A · B o AB

Resultado 1 solo si **TODAS** las entradas son 1. Implementa la conjunción lógica.

OR (Disyunción)

Símbolo: A + B

Resultado 1 si **AL MENOS UNA** de las entradas es 1. Implementa la disyunción lógica.

NOT (Negación)

Símbolo: Ā o A'

Invierte el valor de la entrada. Transforma 1 en 0 y 0 en 1

AND

Inputs		Output
A	**B**	**Y**
0	0	0
0	1	0
1	0	0
1	1	1

OR

Inputs		Output
A	**B**	**Y**
0	0	0
0	1	1
1	0	1
1	1	1

NOT

Input	Output
A	**Y**
0	1
1	0

Álgebra Booleana en Hardware y Programación

Diseño de Hardware

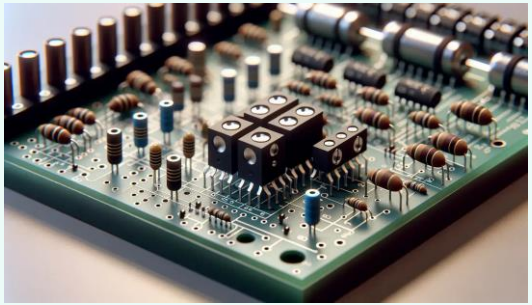
Toda la lógica de una CPU, memoria y periféricos se basa en expresiones booleanas implementadas con compuertas lógicas. El álgebra booleana es el lenguaje fundamental para describir el comportamiento digital.

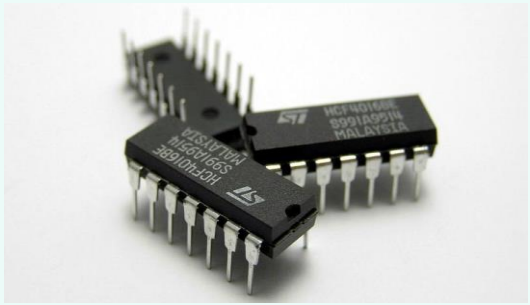
Simplificación de Circuitos

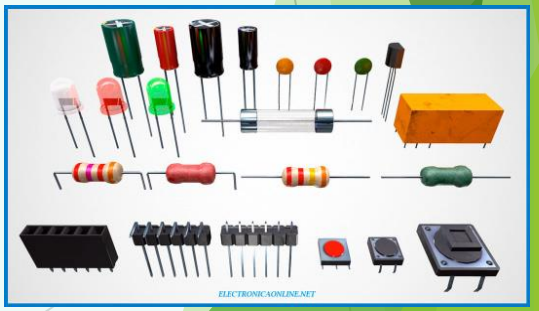
Las leyes booleanas permiten reducir el número de compuertas necesarias para una función, resultando en circuitos más rápidos, pequeños y con menor consumo energético.

Componentes Digitales

Multiplexores, decodificadores, sumadores y unidades de control se diseñan utilizando expresiones booleanas simplificadas.







Leyes Fundamentales: Identidad, Dominación e Idempotencia

Leyes de Identidad

$$A + 0 = A$$
$$A \cdot 1 = A$$

El elemento **neutro**. El 0 es neutro para OR y el 1 es neutro para AND. No alteran el valor de la variable.

Leyes de Dominación

$$A + 1 = 1$$
$$A \cdot 0 = 0$$

El elemento **absorbente**. El 1 domina OR y el 0 domina AND. Siempre producen el mismo resultado.

Leyes Idempotentes

$$A + A = A$$
$$A \cdot A = A$$

La operación de una variable consigo misma produce la **misma variable**. Elimina redundancias en expresiones lógicas.

Leyes de Complemento

$$A + \bar{A} = 1$$
$$A \cdot \bar{A} = 0$$

La relación entre una variable y su **inverso**. OR produce siempre 1, AND produce siempre 0.

Leyes Fundamentales: Conmutativa y Asociativa

Leyes Conmutativas

El orden de los operandos no altera el resultado. Esta propiedad permite reorganizar términos en expresiones lógicas, facilitando su simplificación y análisis.

Suma (OR)

$A + B = B + A$

Producto (AND)

$A \cdot B = B \cdot A$

Leyes Asociativas

La agrupación de los operandos no altera el resultado. Permite simplificar expresiones con múltiples operandos del mismo tipo, eliminando paréntesis innecesarios.

Suma (OR)

$A + (B + C) = (A + B) + C$

Producto (AND)

$A \cdot (B \cdot C) = (A \cdot B) \cdot C$

Ley Distributiva: Con una propiedad única

Leyes Distributivas

Permiten la expansión y factorización de expresiones booleanas. El Álgebra Booleana posee una propiedad única: la doble distributividad.

Distributiva Estándar

$A \cdot (B + C) = A \cdot B + A \cdot C$

Distributiva Dual (Única)

$A + (B \cdot C) = (A + B) \cdot (A + C)$

Propiedad Única del Álgebra Booleana

A diferencia del álgebra ordinaria, el Álgebra Booleana permite distribuir la suma sobre el producto. Esta propiedad es fundamental para la simplificación de circuitos digitales complejos.

Comparación con Álgebra Ordinaria

En el álgebra convencional, solo existe la distributiva del producto sobre la suma: $a(b + c) = ab + ac$. Sin embargo, en el Álgebra Booleana, ambas formas son válidas y simétricas. Esta simetría es una característica distintiva que hace que el Álgebra Booleana sea especialmente poderosa para el diseño y simplificación de circuitos lógicos digitales.

Leyes de De Morgan:

Principio Fundamental

Las Leyes de De Morgan permiten transformar una suma negada en un producto de negaciones, y viceversa. Son cruciales para la simplificación de expresiones booleanas complejas y para el diseño eficiente de circuitos digitales con compuertas NAND y NOR.

Primera Ley de De Morgan

$$(A + B)' = A' \cdot B'$$

La negación de una disyunción (suma) es igual a la conjunción (producto) de las negaciones. La negación de "A O B" es "NO A Y NO B".

Segunda Ley de De Morgan

$$(A \cdot B)' = A' + B'$$

La negación de una conjunción (producto) es igual a la disyunción (suma) de las negaciones. La negación de "A Y B" es "NO A O NO B".

Aplicación Práctica: Implementación con Compuertas NAND/NOR

Utilizando la primera ley de De Morgan, podemos implementar una compuerta OR usando solo compuertas NOT y AND:

$A + B = (A' \cdot B')'$

Esto significa: OR = NOT(AND(NOT(A), NOT(B)))

Ley de Absorción y Simplificación Lógica

Ley de Absorción (Forma 1)

$$A + (A \cdot B) = A$$

Una variable sumada a su producto con otra variable se reduce a la variable original. El término redundante se absorbe.

Ley de Absorción (Forma 2)

$$A \cdot (A + B) = A$$

Una variable multiplicada por su suma con otra variable se reduce a la variable original. Elimina redundancias.

Doble Negación

$$(A')' = A$$

Negar dos veces una variable produce la variable original. Simplifica expresiones con negaciones múltiples.

Ley de Consenso

$$AB + A'C + BC = AB + A'C$$

El término de consenso BC es redundante y puede eliminarse. Reduce la complejidad de expresiones.

Ejemplo: Simplificación Paso a Paso

Simplificar: $F = AB + A(B + C) + B(B + C)$

Paso 1: $F = AB + AB + AC + BB + BC$

Paso 2: $F = AB + AC + B + BC$

Paso 3: $F = AB + AC + B$

Paso 4: $F = B + AC$

La expresión se simplifica de 7 términos a solo 2, reduciendo significativamente la complejidad del circuito

Ejemplo: Problema y Expresión Inicial

Problema: Sistema de Alarma de Seguridad

Un sistema de seguridad activa una alarma (F) si se cumplen las siguientes condiciones:

- (A) Puerta abierta Y (B) Sensor de movimiento activo
- O (C) Ventana abierta Y (D) Sensor de presión inactivo
- O (A) Puerta abierta Y (D) Sensor de presión inactivo

Definición de Variables

A = Puerta abierta

B = Sensor de movimiento activo

C = Ventana abierta

$\overline{D}$ = Sensor de presión inactivo (NOT D)

Expresión Booleana Inicial

$$F = (A \cdot B) + (C \cdot \overline{D}) + (A \cdot \overline{D})$$

Ejemplo: Simplificación Paso a Paso

Expresión Inicial (Recordatorio)

$F = (A \cdot B) + (C \cdot \bar{D}) + (A \cdot \bar{D})$

Paso 1: Factorizar $\bar{D}$

$F = A \cdot B + \bar{D} \cdot (C + A)$

Ley Distributiva: Extraer factor común $\bar{D}$ de los términos 2 y 3

Paso 2: Simplificar (C + A)

$F = A \cdot B + \bar{D} \cdot (A + C)$

Ley Conmutativa: Reordenar términos dentro del paréntesis

Paso 3: Forma Simplificada Óptima

$F = A \cdot B + \bar{D} \cdot (A + C)$

La expresión está completamente simplificada. No se pueden aplicar más reducciones

Expresión Simplificada Final

$F = A \cdot B + \bar{D} \cdot (A + C)$

Interpretación Práctica: La alarma se activa si (puerta abierta Y movimiento detectado) O (sensor de presión inactivo Y (puerta abierta O ventana abierta)). Esta simplificación reduce significativamente la complejidad del circuito físico necesario, pasando de tres términos a dos términos principales.

Lógica Booleana en Software y Aplicaciones Reales

Estructuras de Control (Software)

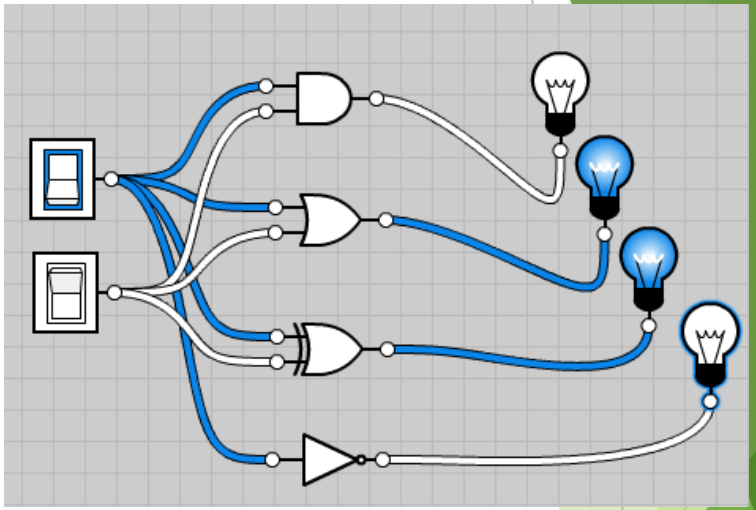
Las sentencias if-else, while y for se basan en la evaluación de expresiones booleanas. Cada condición es una expresión lógica que determina el flujo del programa.

```
if (es_admin AND
 (hora_laboral OR
 tiene_permiso))
 acceder_sistema()
```

Bases de Datos (SQL)

Las cláusulas WHERE utilizan operadores lógicos (AND, OR, NOT) para filtrar registros. Cada consulta es esencialmente una expresión booleana.

```
SELECT * FROM usuarios
 WHERE edad > 18
 AND (ciudad='Madrid'
 OR ciudad='Barcelona')
```



Inteligencia Artificial

Los árboles de decisión y las redes neuronales utilizan funciones de activación y lógica binaria en sus capas de decisión. La clasificación binaria es fundamental en ML.

Conclusiones: El Fundamento de la Era Digital

Pilar Matemático

El Álgebra Booleana es el pilar matemático que permite la existencia de la computación digital moderna. Sin sus principios, la electrónica digital no sería posible.

Diseño Eficiente

Es fundamental para el diseño eficiente de hardware y la comprensión de la lógica de programación en todos los niveles.

Simplificación = Eficiencia

La capacidad de simplificar expresiones lógicas se traduce directamente en eficiencia y ahorro de recursos en el mundo real.