

El Lenguaje de las Computadoras: Entendiendo el Sistema Binario

Una introducción clara al sistema binario en microinformática

01101000

11010010

10110011

11100100

01001000 01101111 01101100 01100001 00100000 01101101 01110101 01101110 01100100 01101111

¿Por qué las computadoras hablan en 0s y 1s?

🖥 Las computadoras **no entienden** palabras ni números como los humanos

🔌 Trabajan con **electricidad**:

0

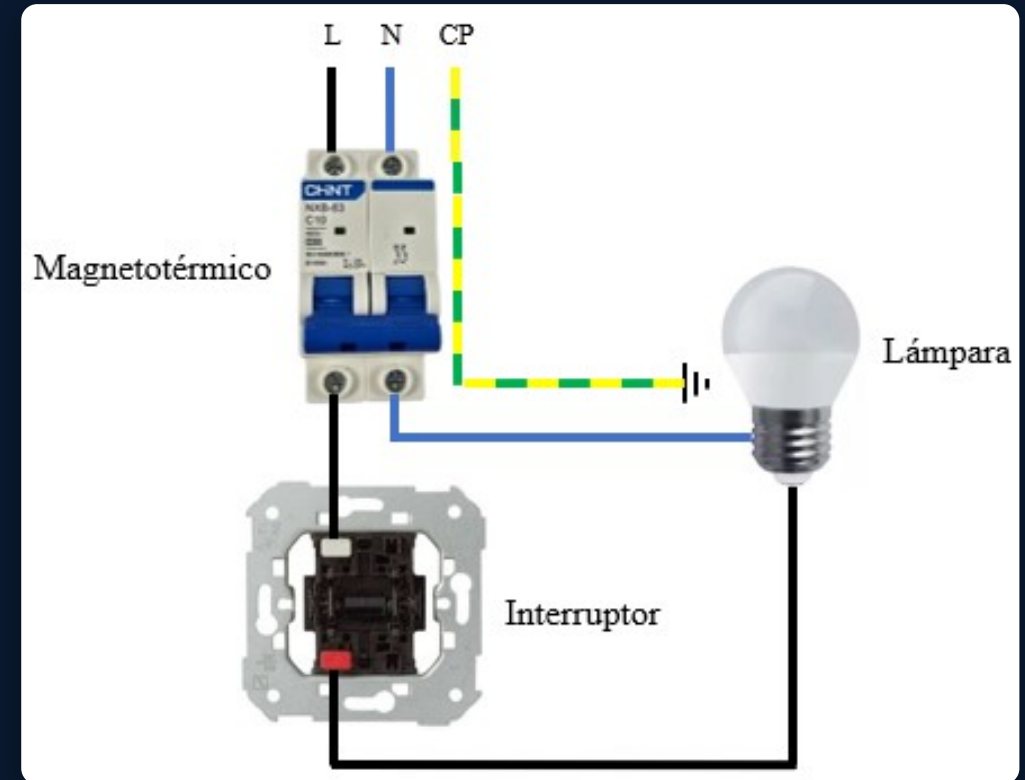
= Apagado (sin corriente)

1

= Encendido (con corriente)

<> Este lenguaje de ceros y unos se llama **sistema binario**

🏠 Es la **base** de todo lo que hace una computadora



¿Qué es el sistema binario?

Σ Sistema numérico de **base 2** (solo usa dos dígitos: 0 y 1)

Decimal (base 10)	Binario (base 2)
0, 1, 2, 3, 4, 5, 6, 7, 8, 9	0, 1

Ejemplo

5 → **101**

📁 Todo en la computadora se convierte a binario

Texto • Imágenes • Sonido • Videos • Programas

Conversión Binario → Decimal * Método rápido

1 0 0 1 0 1

↓ ↓ ↓ ↓ ↓ ↓

32 16 8 4 2 1

Sumamos sólo los que tienen un "1".

Por tanto: 1 0 0 1 0 1 = 32 + 4 + 1 = 37

Unidades básicas: Bit y Byte

🔌 Bit (Binary digit)

- La unidad **más pequeña** de información
- Puede ser **0 o 1**
- Representa un estado: apagado/encendido

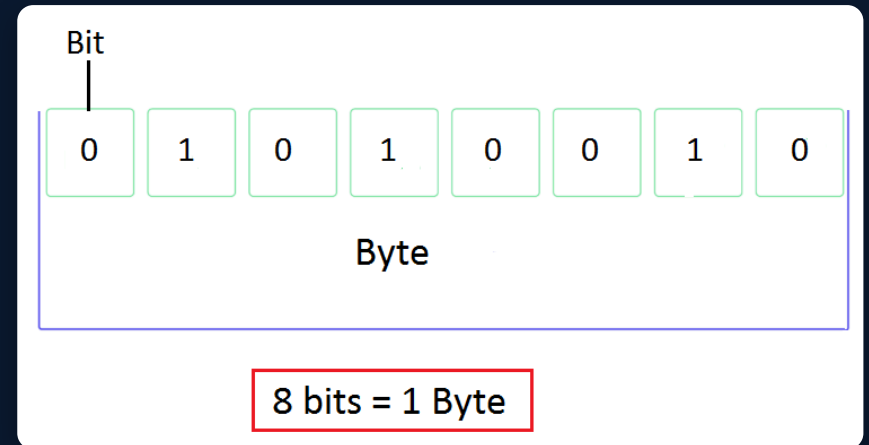
☰ Byte

- Grupo de **8 bits**
- Ejemplo:

0 1 0 0 1 1 0 1 = 1 byte

❓ ¿Por qué 8 bits?

- ✓ Permiten representar **256 combinaciones** (0 a 255)
- ✓ Suficientes para letras, números y símbolos (ASCII)
- ✓ E á ó ì á



¿Cuándo usamos bits y cuándo bytes?

Bits (b)

- Se usan para medir **velocidad de transmisión**
- Representan la cantidad de datos transferidos por segundo



Internet de 100 Mbps = 100 megabits por segundo

Bytes (B)

- Se usan para medir **almacenamiento**
- Representan el tamaño de archivos y capacidad de dispositivos

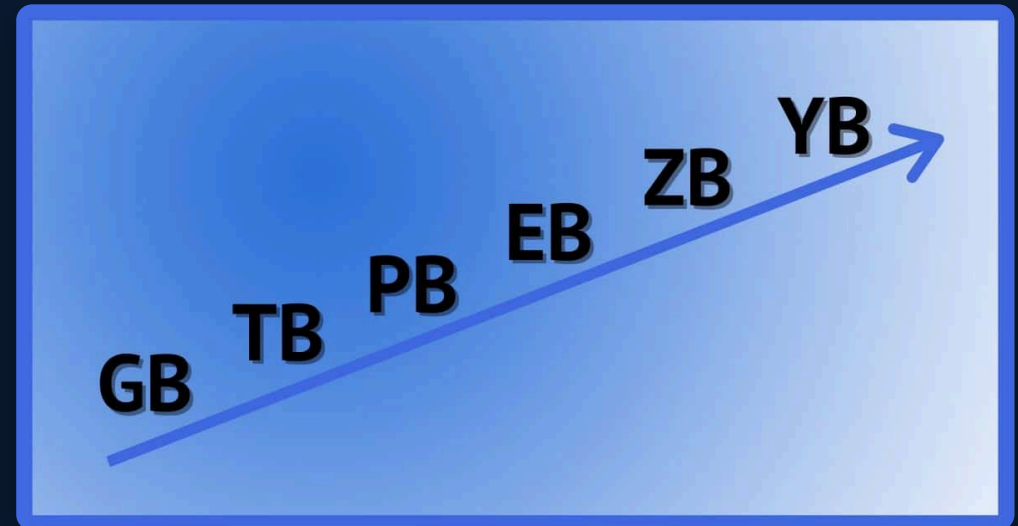


Un archivo de 2 MB = 2 megabytes

¡Ojo! No confundir

- **Mb** (megabits) \neq **MB** (megabytes)
- La diferencia entre mayúsculas y minúsculas es importante

1 byte = 8 bits



¿Qué significa "sistema de 32 bits" o "64 bits"?

Se refiere al **ancho de la arquitectura del procesador**, determinando cuántos datos puede procesar simultáneamente.

🔧 32 bits

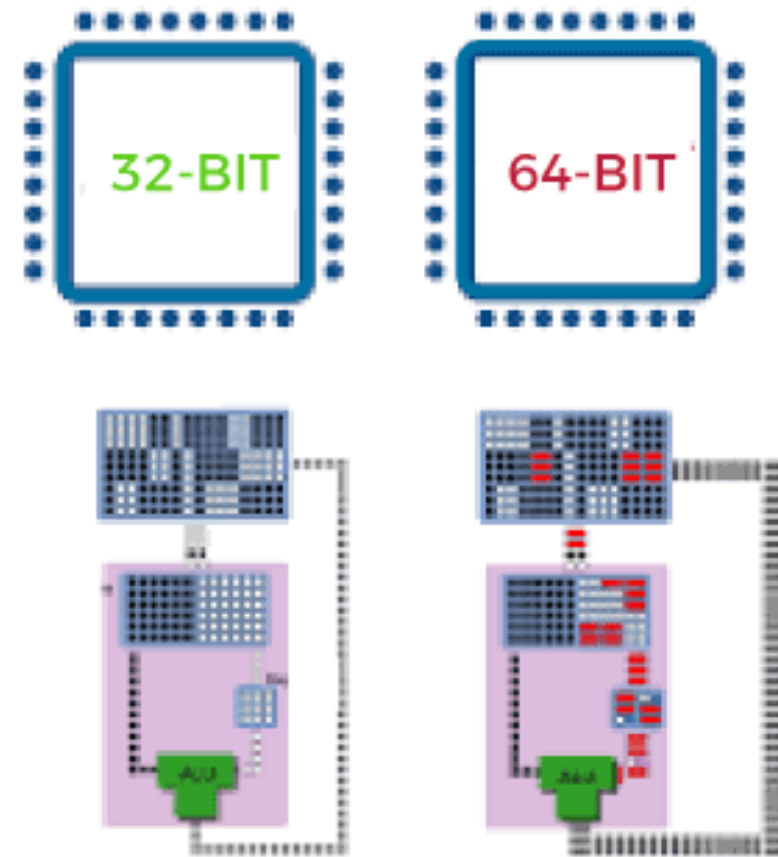
- ▶ Procesa **32 bits** de datos a la vez
- ▶ Soporta hasta **4 GB de RAM**
- ▶ Limitaciones en memoria y rendimiento

🔧 64 bits

- ▶ Procesa **64 bits** a la vez
- ▶ **Más rápido y eficiente**
- ▶ Soporte de memoria hasta **16 exabytes**

↗ Situación actual

Hoy en día, **casi todos los sistemas** son de 64 bits debido a sus ventajas en rendimiento y capacidad de memoria.



32 bit vs 64 bit

Conversión de binario a decimal (reglas básicas)

Σ Cada posición en un número binario tiene un valor en **potencias de 2**

→ Se lee de **derecha a izquierda**, empezando en 2^0

 Ejemplo: 1011

1	0	1	1
2^3	2^2	2^1	2^0

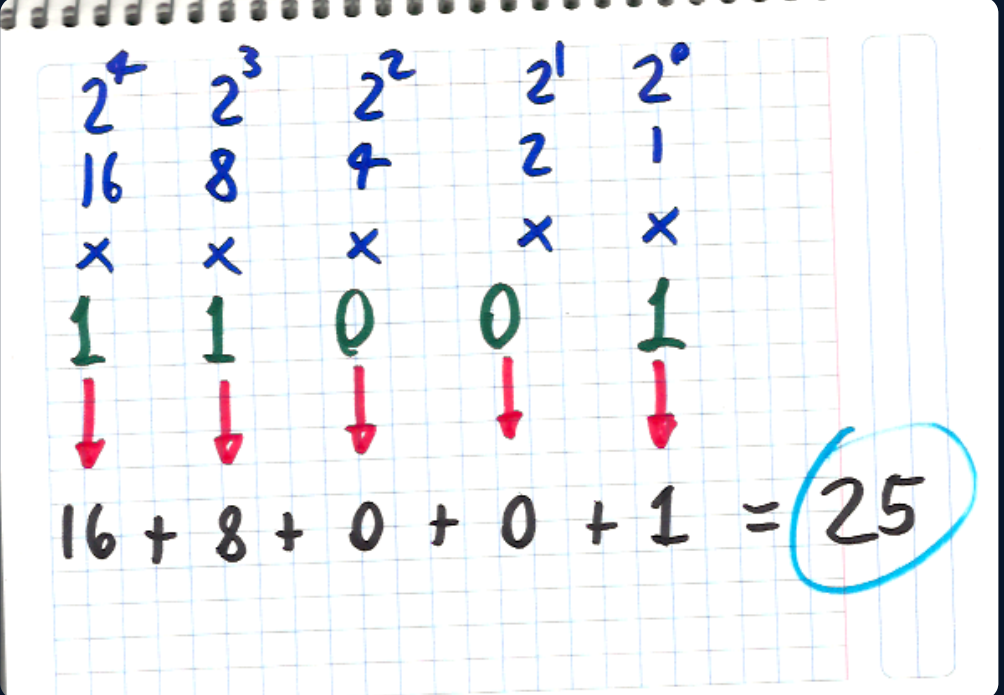
1 $1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$

2 $1 \times 8 + 0 \times 4 + 1 \times 2 + 1 \times 1$

3 $8 + 0 + 2 + 1$

1011 = **11**

Resultado en decimal



2^4	2^3	2^2	2^1	2^0
16	8	4	2	1
x	x	x	x	x
1	1	0	0	1
↓	↓	↓	↓	↓
16	8	0	0	1
$= 11$				

Ejercicio práctico

Convierte estos números **binarios a decimal** aplicando las reglas de conversión que aprendiste.

1 1 0 1 0 → ?

2 1 1 1 1 → ?

3 1 0 0 0 0 → ?

Las respuestas aparecerán al hacer clic o en la siguiente diapositiva

💡 Consejo

Recuerda: cada posición representa una potencia de 2, de derecha a izquierda ($2^0, 2^1, 2^2, \dots$)

Conversión Binario → Decimal * Método rápido

1 0 0 1 0 1

↓ ↓ ↓ ↓ ↓ ↓

32 16 8 4 2 1

Sumamos sólo los que tienen un "1".

Por tanto: $1\ 0\ 0\ 1\ 0\ 1 = 32 + 4 + 1 = 37$

¿Y por qué es importante esto en microinformática?

🖥️ Todo en la computadora se basa en binario

🔧 Memoria RAM

📀 Almacenamiento
(discos duros, SSD)

📶 Redes (datos
enviados como bits)

⏏ Programación de
bajo nivel

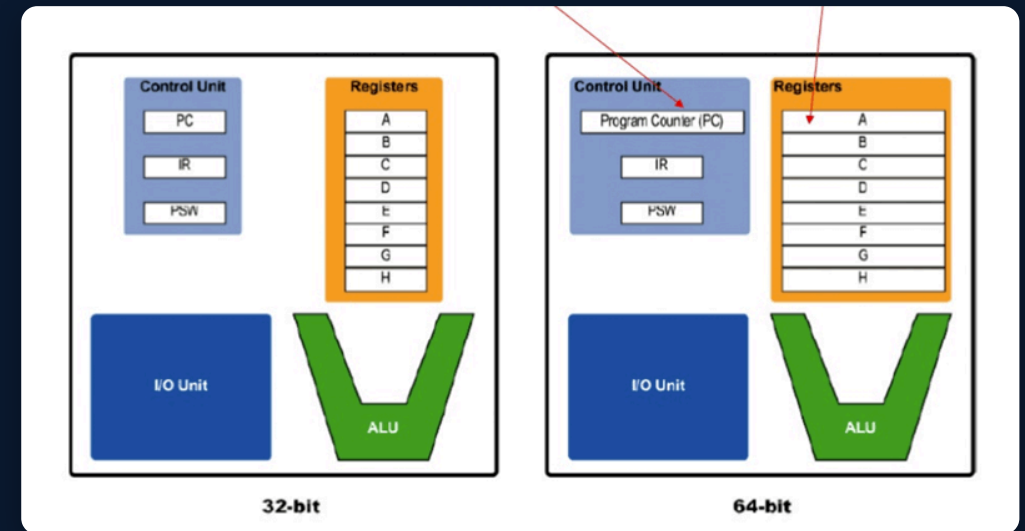
💡 Entender binario ayuda a:

🔧 Diagnosticar problemas
técnicos

📀 Comprender límites del
hardware

🎓 Aprender programación

🔌 Entender electrónica
digital



Resumen



El **binario** es el lenguaje fundamental de las computadoras



Bit = unidad mínima (0 o 1) • **Byte** = 8 bits



Usamos **bits** para velocidad y **bytes** para almacenamiento



32/64 bits define la capacidad del procesador



Saber convertir **binario** → **decimal** es una habilidad clave

```
01001001 01101110 01100110 01101111 01110010 01101101
11100010 10000000 10001101 01110100 01101001 01100011
01100001 00100000 01100110 01110101 01101110 01100100
01100001 01101101 01100101 01101110 01110100 01100001
01101100
```

