**Bit** es el [acrónimo](https://es.wikipedia.org/wiki/Acr%C3%B3nimo) ***B****inary dig****it*** (dígito binario). Un bit es un dígito del [sistema de numeración](https://es.wikipedia.org/wiki/Sistema_de_numeraci%C3%B3n) [binario](https://es.wikipedia.org/wiki/Sistema_binario).

Mientras que en el sistema de numeración [decimal](https://es.wikipedia.org/wiki/Sistema_de_numeraci%C3%B3n_decimal) se usan diez [dígitos](https://es.wikipedia.org/wiki/Cifra_(matem%C3%A1tica)), en el [binario](https://es.wikipedia.org/wiki/Sistema_binario) se usan sólo dos dígitos, el 0 y el 1. Un bit o dígito binario puede representar uno de esos dos valores, **0** ó **1**.

Se puede imaginar un bit, como una bombilla que puede estar en uno de los siguientes dos estados:

apagadafoc 1.pngo encendida foc 4.png

El bit es la unidad mínima de información empleada en [informática](https://es.wikipedia.org/wiki/Inform%C3%A1tica), en cualquier dispositivo digital, o en la [teoría de la información](https://es.wikipedia.org/wiki/Teor%C3%ADa_de_la_informaci%C3%B3n). Con él, podemos representar dos valores cuales quiera, como verdadero o falso, abierto o cerrado, blanco o negro, norte o sur, masculino o femenino, rojo o azul, etc. Basta con asignar uno de esos valores al estado de "apagado" (0), y el otro al estado de "encendido" (1).

**Combinaciones de bits**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | |  |  | | --- | --- | | Hay 4 combinaciones posibles con dos bits | | | **Bit 1** | **Bit 0** | | 0 | 0 | | 0 | 1 | | 1 | 0 | | 1 | 1 | |

Con un bit podemos representar solamente dos valores, que suelen representarse como 0, 1. Para representar o [codificar](https://es.wikipedia.org/wiki/Codificaci%C3%B3n_digital) más información en un dispositivo digital, necesitamos una mayor cantidad de bits. Si usamos dos bits, tendremos cuatro combinaciones posibles: (dado el 9/5/2003)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **16** | **8** | **4** | **2** | **1** | <-- Valor de posición |
| foc 4.png | foc 2.png | foc 2.png | foc 4.png | foc 4.png | Representación gráfica de los **bits** como bombillas encendidas y apagadas |
| 1 | 0 | 0 | 1 | 1 | <-- Dígitos binarios (bits) |

* **0 0** - Los dos están "apagados"
* **0 1** - El primero (de izquierda a derecha) está "apagado" y el segundo "encendido"
* **1 0** - El primero (de izquierda a derecha) está "encendido" y el segundo "apagado"
* **1 1** - Los dos están "encendidos"
* Con estas cuatro combinaciones podemos representar hasta cuatro valores diferentes, como por ejemplo, los colores azul, verde, rojo y magenta.
* A través de secuencias de bits, se puede [codificar](https://es.wikipedia.org/wiki/Codificaci%C3%B3n_digital) cualquier valor [discreto](https://es.wikipedia.org/wiki/Discreto) como números, palabras, e imágenes. **Cuatro bits forman un [nibble](https://es.wikipedia.org/wiki/Nibble" \o "Nibble)**, y pueden representar hasta 24 = 16 valores diferentes; **ocho bits forman un**[**octeto**](https://es.wikipedia.org/wiki/Octeto), y se pueden representar hasta 28 = 256 valores diferentes. En general, con un número **n** de bits pueden representarse hasta 2n valores diferentes.
* **Nota**: Un [byte](https://es.wikipedia.org/wiki/Byte) y un [octeto](https://es.wikipedia.org/wiki/Octeto) no son lo mismo. Mientras que un octeto siempre tiene 8 bits, un byte, tiene 8 bits, siendo equivalente al octeto, aunque existen excepciones.

**Valor de posición**

En cualquier [sistema de numeración posicional](https://es.wikipedia.org/wiki/Sistema_de_numeraci%C3%B3n_posicional), el valor de los dígitos depende de la posición en que se encuentren.

En el [sistema decimal](https://es.wikipedia.org/wiki/Sistema_de_numeraci%C3%B3n_decimal), por ejemplo, el dígito 5 puede valer 5 si está en la posición de las unidades, pero vale 50 si está en la posición de las decenas, y 500 si está en la posición de las centenas. Generalizando, cada vez que nos movemos una posición hacia la izquierda el dígito vale 10 veces más, y cada vez que nos movemos una posición hacia la derecha, vale 10 veces menos. Esto también es aplicable a números con decimales.

Por tanto, el número 153,7 en realidad es: 1 centena + 5 decenas + 3 unidades + 7 décimas, es decir,

100 + 50 + 3 + 0,7 = 153,7.

En el sistema binario es similar, excepto que cada vez que un dígito binario (bit) se desplaza una posición hacia la izquierda vale el doble (2 veces más), y cada vez que se mueve hacia la derecha, vale la mitad (2 veces menos).

+-----+-----+-----+-----+-----+

| 16 | 8 | 4 | 2 | 1 | <-- Valor del bit de acuerdo a su posición

+-----+-----+-----+-----+-----+ expresado en números

| 2^4 | 2^3 | 2^2 | 2^1 | 2^0 | <-- Valor del bit de acuerdo a su posición

+-----+-----+-----+-----+-----+ expresado en forma de potencias de 2

Abajo vemos representado el número 19.

16 + 2 + 1 = 19.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **16** | **8** | **4** | **2** | **1** | <-- Valor de posición |
| foc 4.png | foc 2.png | foc 2.png | foc 4.png | foc 4.png | Representación gráfica de los **bits** como bombillas encendidas y apagadas |
| 1 | 0 | 0 | 1 | 1 | <-- Dígitos binarios (bits) |
|  |  |  |  |  |  |

* **Bits y bytes Tablas de conversión**
* **En los sistemas informáticos de un byte es una unidad binaria de medida utilizada para referirse al espacio de almacenamiento en disco en una unidad de disco duro o la memoria memoria de acceso aleatorio (RAM) en los sistemas informáticos.**
* Se tarda 8 bits para crear un byte donde un poco también es un dígito binario que consiste de un valor de 0 o 1. Por ejemplo, 10101010, 00000000, 11111111 son los 8 bits de longitud y formar 1 Byte. En pocas palabras un Byte es una colección de bits.
* El cuadro directamente debajo le proporcionará las conversiones y la segunda tabla de las abreviaturas para cada muestra.

|  |  |
| --- | --- |
| Unidad | Iguales |
| 1 Bit | Dígito binario |
| 8 Bits | 1 Byte |
| 1024 Bytes | 1 Kilobyte |
| 1024 Kilobytes | 1 Megabyte |
| 1024 Megabytes | 1 Gigabyte |
| 1024 Gigabytes | 1 Terabyte |
| 1.024 Terabytes | 1 petabyte |
| 1.024 petabytes | 1 Exabyte - |
| 1.024 exabytes | 1 zetabyte |
| 1024 zettabytes | 1 Yottabyte |
| 1024 Yottabytes | 1 Brontobyte |

¿Te has preguntado cuántos bytes hay en un kilobyte? ¿Alguna vez ha tratado de averiguar?   
En primer lugar, hay que recordar que las computadoras no hablan Inglés, español ni japonés.  No encadenan letras de nuestro alfabeto para formar palabras. No utilizan nuestros 10 números decimales para hacer combinaciones de números infinitos.

SON los ordenadores quienes trabajan con un sistema binario de números para representar cada función que realizamos… como crear un texto, realizar un dibujo o jugar. Así que todo se almacena en la memoria como números binarios y, por cada byte que se escribe se realiza una combinación de ocho 1 y o

¿Cómo se hace? He aquí algunos ejemplos:

El número binario de la letra mayúscula A es 01000001

El símbolo de asterisco se almacena como 00101010

El nombre de "Clinton" se vería así:   
010000110110110001101001011011100111010001101111 01101110

Cada letra, número, espacio o símbolo se encuentra representado por una combinación de ocho 1 y 0. Esto puede parecer como un desperdicio de espacio, pero el equipo no se preocupa por el espacio. De hecho, dicen que el sistema binario es mucho más fácil para un ordenador de entender debido a su naturaleza electrónica y el hecho de que sólo está tratando con 1 y 0.

NOTA: Este sistema binario es estándar en todo el mundo para cualquier persona que pone la información en Internet, si se hace en una computadora o hacer en WebTV. Hay 256 combinaciones diferentes que usted puede hacer con 8 ceros y unos.

Este es el orden:

1 bit = un solo dígito, 1 o 0

8 bits = 1 byte, una combinación de 1 y 0

1024 bytes = 1 KB (kilobyte)

1024 Kilobytes = 1 MB (megabyte)

1024 Megabytes = 1 GB (gigabyte)

Otra forma de decirlo:

Se le permite un total de 1.024 bytes antes de utilizar hasta una KB.

Y se puede utilizar más de 1 millón de bytes (1024 x 1024) antes de utilizarla por un MB.

Y se puede utilizar más de 1 mil millones de bytes (1024 X 1024 X 1024) antes de utilizar hasta un GB.

O pensar en ello de esta manera:

Bytes constituyen Kilobytes de 1024

que constituyen Megabytes de 1024 X 1024

que constituyen Gigabytes de 1024 X 1024 X 1024

Ahora, para explicar cómo consiguen ese número mágico 1024.

Debido a que el sistema de código binario sólo tiene 2 números, las potencias de 2 juegan un papel importante.

Se toman 2 a la 10 ª potencia para obtener el número 1024. Para aquellos de nosotros que no son muy buenos en matemáticas, voy a explicarlo para usted.

2 es la primera potencia   
2 x 2 = 4 (la segunda potencia)   
2 x 4 = 8 (la tercera potencia)   
2 x 8 = 16 (el cuarto poder)   
2 X 16 = 32 (el quinto poder)   
2 X 32 = 64 (la sexta potencia)   
2 X 64 = 128 (el séptimo de energía)   
2 X 128 = 256 (la 8 ª potencia)   
2 X 256 = 512 (la novena potencia)   
2 X 512 = 1024 (la décima potencia)

Y así es como se obtiene 1 KB.

Si es más fácil para usted para "estimar" el uso por   
el uso de 1000, seguir adelante y hacerlo. Recuerde que sus "bytes reales" se diferencian del "tamaño bruto" obtendrá ya que el texto no es la única información que ocupa espacio en su almacenamiento. Imágenes y Midis se registran en bytes, así como los códigos como negrita, cursiva, subrayado, diferentes fuentes y tamaños de fuente. Estos se suman bytes a sus páginas.

Entrar en el administrador de archivos y verán los tamaños que aparecen ahí para cada página que usted hace.

## Bits más y menos significativos

Un conjunto de bits, como por ejemplo un [byte](https://es.wikipedia.org/wiki/Byte), representa un conjunto de elementos ordenados. Se llama [bit más significativo](https://es.wikipedia.org/wiki/Bit_m%C3%A1s_significativo) (MSB) al bit que tiene un mayor peso (mayor valor) dentro del conjunto, análogamente, se llama [bit menos significativo](https://es.wikipedia.org/wiki/Bit_menos_significativo) (LSB) al bit que tiene un menor peso dentro del conjunto.

En un Byte, el bit más significativo es el de la posición 7, y el menos significativo es el de la posición 0

+---+---+---+---+---+---+---+---+

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | <-- Posición del bit

+---+---+---+---+---+---+---+---+

|128|64 |32 |16 | 8 | 4 | 2 | 1 | <-- Valor del bit de acuerdo a su posición

+---+---+---+---+---+---+---+---+

| |

| +- Bit menos significativo

+----------------------------- Bit más significativo

**Funciones Lógicas y Circuitos Combinacionales**

**1. Compuertas Lógicas**

**El número total de posibles combinaciones de entradas binarias es determinada con la siguiente fórmula:  N = 2n**

**Invertidor (NOT)**

**Ejecuta la función lógica básica llamada inversión o complementación.  Su propósito es cambiar de un nivel lógico a su nivel opuesto.  En término de bits, cambia de 1 a 0 y de 0 a 1.**

**AND**

**Ejecuta la multiplicación lógica, comúnmente llamada la función AND.  La salida sólo es HIGH (1) cuando todas las entradas son HIGH (1), de lo contrario la salida es LOW (0).**

**OR**

**Ejecuta la suma lógica, comúnmente llamada la función OR.  La salida es HIGH (1) si al menos una entrada es HIGH (1), de lo contrario es la salida es LOW (0).**

**NAND**

**Es un elemento lógico muy popular porque se puede utilizar como una función universal.  La salida sólo es LOW (0) cuando todas las entradas son HIGH (1), de lo contrario la salida es HIGH (1).**

**NOR**

**Al igual que el NAND, es de gran utilidad por su propiedad universal.  La salida es LOW (0) si al menos hay una entrada HIGH (1).**

**XOR**

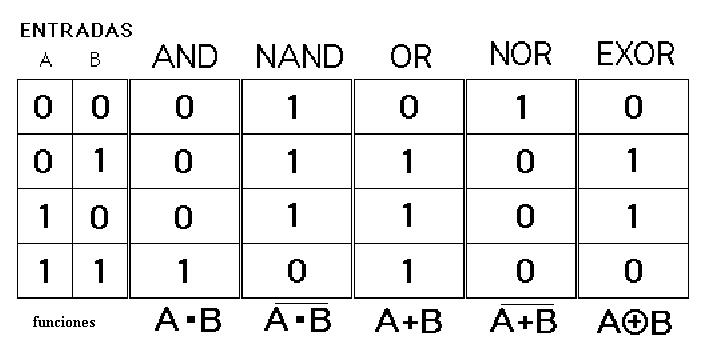
**Tiene sólo dos entradas.  La salida es HIGH (1) sólo cuando las entradas tienen niveles opuestos, de lo contrario (entrada del mismo nivel) la salida es LOW (0).**

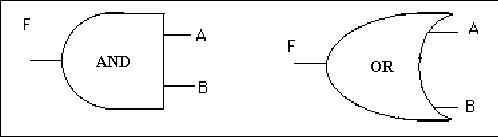
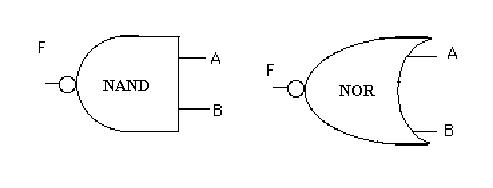
**XNOR**

**Tiene sólo dos entradas.  La salida es LOW (0) sólo cuando las entradas tienen niveles opuestos, de lo contrario (entrada del mismo nivel) la salida es HIGH (1).**

**Exor**

**Esta puerta lógica , la EXOR, nos da a la salida un 0 siempre que sus entradas tengan igual valor . En el resto de los casos da 1 a la salida.**

**2. Tablas de la Verdad (Veracidad)**  


  Diagramas   
 

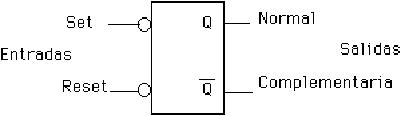
**Circuitos Lógicos Secuenciales**

Los bloques básicos para construirlos son los circuitos flip-flops.  Los circuitos lógicos secuenciales son extremadamente importantes debido a su característica de memoria.

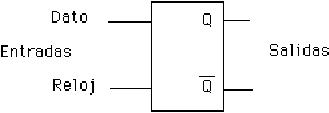
Los flip-flops también se denominan "cerrojos", "multivibradores biestables" o "binarios".  Los flip-flops pueden construirse a partir de puertas lógicas, como, por ejemplo, puertas NAND, o comprarse en forma de circuitos integrados.  Los flip-flops se interconectan para formar circuitos lógicos secuenciales que almacenen datos, generen tiempos, cuenten y sigan secuencias.

FLIP-FLOPS RS

Es el flip-flop básico

  
  
  
FLIP-FLOP D

Tiene solamente una entrada de datos (D) y una entrada de reloj (CK).  El flip-flop D, con frecuencia, se denomina flip-flop de retardo (y de datos).  Este nombre describe con precisión la operación que realiza.  Cualquier que sea el dato en la entrada (D), éste aparece en la salida normal retardado un pulso de reloj.  El dato es transferido a la salida durante la transición del nivel BAJO al ALTO del pulso de reloj.



FLIP-FLOP JK

Este dispositivo puede considerarse como el flip-flop universal; los demás tipos pueden construirse a partir de él.

Decodificadores

Mucha de la información en las computadoras se maneja en una forma de codificación muy eleveda.  En una instrucción, es posible utilizar un campo de n bits para denotar 1 de 2n posibles elecciones para la acción que se vaya a tomar.  Para efectuar la acción deseada, la instrucción codificada primero debe decodificarse.  A un circuito capaz de aceptar una entrada de n variables y de generar la señal correspondiente de salida en una de 2n líneas de salida se le denomina decodificador.  Los decodificadores seleccionan una línea de salida con base en señales de entrada.