

Fundamentos de Ingeniería Informática

Módulo III - Unidad 9 Representación de la información

Profesor: Héctor molina García

Versión: 0.1

¿Qué son los datos?

¿Qué es la información?

¿Es lo mismo datos e información?

Información vs Datos

Datos



vs

Información



Los datos son elementos crudos desorganizados y sin refinar.

La información es la organización e interpretación de esos elementos.

Información vs Datos

Ambos conceptos tienen un papel importante en Ciencias de la Computación, pero existen diferencias significativas entre ellos.

Datos	Información
Los datos se refieren a hechos brutos que no tienen un significado específico.	La información se refiere a los datos procesados que tienen un propósito y significado.
Los datos son independientes de la información.	La información depende de los datos.
Los datos o datos sin procesar no son suficientes para tomar una decisión.	La información suele ser suficiente para ayudar a tomar una decisión en un contexto específico.

Información en ordenadores

La información en los ordenadores

Un **bit** o **dígito binario** y es la unidad más pequeña de datos en una computadora. Los bits sólo pueden contener uno de dos valores: 0 o 1.

Un byte es una unidad de información formada por una secuencia de Bits. Según el contexto, puede representar diferentes tipos de información:

- Texto
- Número
- Instrucción del programa.
- Pixel en una imagen o parte de una grabación de audio.

La información en los ordenadores



La información en los ordenadores



La información se modela utilizando una escala de grises donde cada píxel puede representar un tono de gris (0 - 256).



Negro 00000000

Blanco 11111111

La información en los ordenadores



La información se modela utilizando varias capas de color (rojo, verde, azul) y una para el factor de luminosidad.



Negro 00000000

Blanco 11111111

16.7 Millones de colores

La información en los ordenadores



La información se modela utilizando varias capas de color (rojo, verde, azul) y una para el factor de luminosidad.



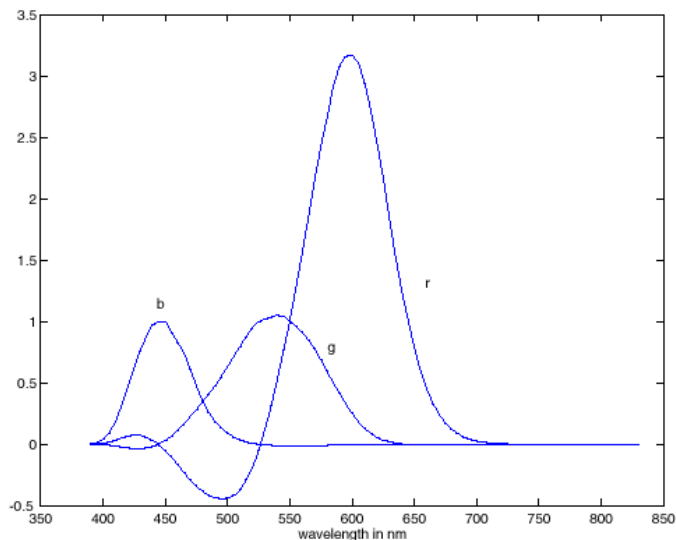
Negro 00000000




Blanco 11111111

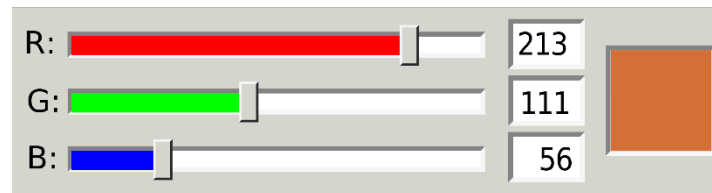
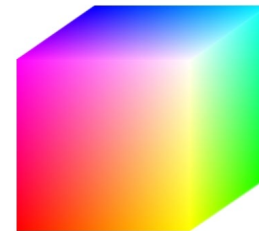
281 Trillón de colores

La información en los ordenadores

El **modelo de color RGB** es un modelo de color aditivo en el que la luz roja, verde y azul se suman de varias maneras para reproducir una amplia gama de colores. El nombre del modelo proviene de las iniciales de los tres colores aditivos primarios / secundarios, rojo, verde y azul.



 $p_1 = 645.2 \text{ nm}$
 $p_2 = 525.3 \text{ nm}$
 $p_3 = 444.4 \text{ nm}$



Hex representation



La información en los ordenadores

Un **sistema numérico** es un modo de escritura para expresar números; es decir, una notación matemática para representar números de un conjunto dado, usando dígitos u otros símbolos de manera consistente..

- Representar un conjunto útil de números.
- Darle a cada número representado una representación única (o al menos una representación estándar).
- Reflejar la estructura algebraica y aritmética de los números.

El **cardinal** (número de elementos) de un conjunto de números se llama la base de un sistema numérico..

Sistema decimal: base = 10, digitos = {0,1,2,3,4,5,6,7,8,9}

La información en los ordenadores

La representación posicional estándar de un número N en base b se escribe de la siguiente manera:

$$N = (a_n a_{n-1} a_{n-2} \dots a_1 a_0 a_{-1} \dots a_{-m})_b$$

Dónde:

- $a_i \rightarrow$ dígitos que constituyen el número (entre 0 y $r-1$)
- $n \rightarrow$ Número de dígitos enteros
- $m \rightarrow$ Número de dígitos fraccionarios
- $a_n \rightarrow$ Dígito más significativo
- $a_{-m} \rightarrow$ dígito menos significativo
- $r^i \rightarrow$ Peso del dígito i
- $a_i * r^i \rightarrow$ Valor del dígito i

La información en los ordenadores

El valor de un número N viene dado por:

$$(N)_b = a_{n-1} \times b^{n-1} + a_{n-2} \times b^{n-2} \dots + a_1 \times b^1 + a_0 \times b^0 + a_{-1} \times b^{-1} \dots + a_{-m} \times b^{-m}$$

donde "b" es la base del sistema numérico (por ejemplo, 2, 8, 10 o 16) y "a" es un dígito que va de 0 a b-1.

$$(352.45)_{10} = 3 \times 10^2 + 5 \times 10^1 + 2 \times 10^0 + 4 \times 10^{-1} + 5 \times 10^{-2}$$

$$= 3 \times 100 + 5 \times 10 + 2 \times 0 + 4 \times 0,1 + 5 \times 0,01$$

La conversión base es el proceso de convertir un número N de un número base a otro número base. Basta con expresar el número a convertir en notación polinómica, expresando los dígitos y pesos en base s , y operar en base s :

$$N = (10101)_2 = 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = (21)_{10}$$

$$M = (14)_{16} = 1 \times 16^1 + 4 \times 16^0 = (20)_{10}$$

De binario a decimal

De binario a decimal

Sistemas posicionales (numerales)

- Binario

- Decimal

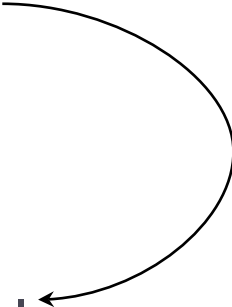
¿Cómo podemos numerar n_2 convertir de una base 2 a base 10?

$$n_2 = 11001010$$

- Hexadecimal

De binario a decimal

Sistemas posicionales (numerales)

- Binario
 - Decimal
 - Hexadecimal
- 


1
1
0
0
1
0
1
0

} **n = 8**

Primero contamos el número de dígitos en nuestro número binario.

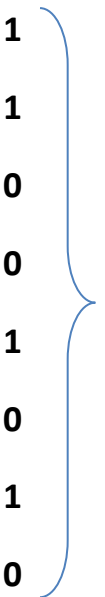
De binario a decimal

Sistemas posicionales (numerales)

- Binario
 - Decimal
- 

- Hexadecimal

1
1
0
0
1
0
1
0



n = 8

A continuación, multiplicamos de izquierda a derecha cada dígito por la potencia de dos que le corresponde, empezando por 2^{n-1}

De binario a decimal

Sistemas posicionales (numerales)

- Binario
- Decimal
- Hexadecimal

7	6	5	4	3	2	1	0	
1 x 2^7	1 x 2^6	0 x 2^5	0 x 2^4	1 x 2^3	0 x 2^2	1 x 2^1	0 x 2^0	1 1 0 0 1 0 1 0
								n = 8

A continuación, multiplicamos de izquierda a derecha cada dígito por la potencia de dos que le corresponde, empezando por 2^{n-1}

De binario a decimal

Sistemas posicionales (numerales)

- Binario
- Decimal
- Hexadecimal

7	6	5	4	3	2	1	0	
1×2^7								1
	1×2^6							1
		0×2^5						0
			0×2^4					0
				1×2^3				1
					0×2^2			0
						1×2^1		1
							0×2^0	0
128	64	0	0	8	0	2	0	

n = 8

Finalmente, añadimos todos esos valores.

De binario a decimal

Sistemas posicionales (numerales)

- Binario
- Decimal


- Hexadecimal

7	6	5	4	3	2	1	0	
1×2^7								1
	1×2^6							1
		0×2^5						0
			0×2^4					0
				1×2^3				1
					0×2^2			0
						1×2^1		1
							0×2^0	0
								n = 8
								= 202
128	64	0	0	8	0	2	0	

De binario a Octal

De binario a octal

Sistemas posicionales (numerales)


- Binario
 - Decimal
 - Octal
- 

¿Cómo podemos numerar n_2 convertir de una base 2 a base 8?

$$n_2 = 100111011101111$$


De binario a octal

Sistemas
(numerales)

- Binario
 - Decimal
 - Octal
- 

posicionales

100111011101111




Dividimos los bits en grupos de 3 de derecha a izquierda. ¿Por qué?


De binario a octal

Sistemas posicionales

(numerales)

- Binario
 - Decimal
 - Octal
- 

100111011101111




Dividimos los bits en grupos de 3 de derecha a izquierda.
¿Por qué?

Necesitamos 3 bits para representar números del 0 al 7.


De binario a octal

Sistemas
(numerales)

- Binario
 - Decimal
 - Octal
- 

posicionales

100111011101111




Convertimos cada triplete a su equivalente octal de un solo dígito.


De binario a octal

Sistemas posicionales

(numerales)

- Binario
 - Decimal
 - Octal
- 

100111011101111




Dec	Hex	Oct	Bin
0	0	000	0000
1	1	001	0001
2	2	002	0010
3	3	003	0011
4	4	004	0100
5	5	005	0101
6	6	006	0110
7	7	007	0111
8	8	010	1000
9	9	011	1001
10	A	012	1010
11	B	013	1011
12	C	014	1100
13	D	015	1101
14	E	016	1110
15	F	017	1111

Convertimos cada triplete a su equivalente octal de un solo dígito.

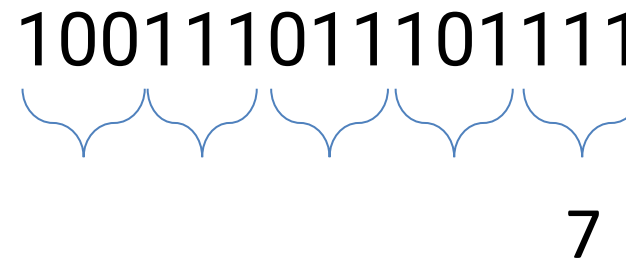
De binario a octal

Sistemas posicionales

(numerales)

- Binario
 - Decimal
 - Octal
- 

100111011101111




7

Dec	Hex	Oct	Bin
0	0	000	0000
1	1	001	0001
2	2	002	0010
3	3	003	0011
4	4	004	0100
5	5	005	0101
6	6	006	0110
7	7	007	0111
8	8	010	1000
9	9	011	1001
10	A	012	1010
11	B	013	1011
12	C	014	1100
13	D	015	1101
14	E	016	1110
15	F	017	1111

De binario a octal

Sistemas posicionales

(numerales)

- Binario
 - Decimal
 - Octal
- 

100111011101111

4 7 3 5 7

Dec	Hex	Oct	Bin
0	0	000	0000
1	1	001	0001
2	2	002	0010
3	3	003	0011
4	4	004	0100
5	5	005	0101
6	6	006	0110
7	7	007	0111
8	8	010	1000
9	9	011	1001
10	A	012	1010
11	B	013	1011
12	C	014	1100
13	D	015	1101
14	E	016	1110
15	F	017	1111

De binario a hexadecimal

De binario a hexadecimal

Sistemas posicionales (numerales)

- Binario

- Decimal

- Hexadecimal

¿Cómo podemos numerar n_2 convertir de una base 2 a base 16?


$n_2 = 100111011101111$

De binario a hexadecimal

Sistemas posicionales (numerales)

- Binario
- Decimal
- Hexadecimal


100111011101111




Dividimos los bits en grupos de 4. ¿Por qué?

De binario a hexadecimal

Sistemas posicionales (numerales)

- Binario
 - Decimal
 - Hexadecimal
- 

100111011101111




Dec	Hex	Oct	Bin
0	0	000	0000
1	1	001	0001
2	2	002	0010
3	3	003	0011
4	4	004	0100
5	5	005	0101
6	6	006	0110
7	7	007	0111
8	8	010	1000
9	9	011	1001
10	A	012	1010
11	B	013	1011
12	C	014	1100
13	D	015	1101
14	E	016	1110
15	F	017	1111

Dividimos los bits en grupos de 4. ¿Por qué?

Necesitamos 4 bits en binario para representar 16 valores en hexadecimal.

De binario a hexadecimal

Sistemas posicionales (numerales)

- Binario
 - Decimal
 - Hexadecimal
- 

100111011101111

4 E E F

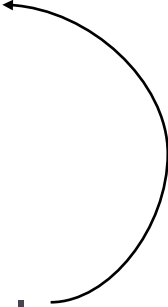
Dec	Hex	Oct	Bin
0	0	000	0000
1	1	001	0001
2	2	002	0010
3	3	003	0011
4	4	004	0100
5	5	005	0101
6	6	006	0110
7	7	007	0111
8	8	010	1000
9	9	011	1001
10	A	012	1010
11	B	013	1011
12	C	014	1100
13	D	015	1101
14	E	016	1110
15	F	017	1111

Convertimos cada grupo de cuatro bits a un hexadecimal.

De decimal a binario

De decimal a binario

Sistemas posicionales (numerales)

- Binario
 - Decimal
- 

¿Cómo podemos numerar n_1 convertir de una base 10 a base 2?

$$n_1 = 233$$

- Hexadecimal

De decimal a binario

Sistemas posicionales (numerales)

- Binario
- Decimal
- Hexadecimal

233

base

/ 2

De decimal a binario

Sistemas posicionales (numerales)

- Binario
- Decimal
- Hexadecimal

233

1

 116
remainder

base
/

2

Dividimos por la base que queremos transformar y almacenamos el resto.

De decimal a binario

Sistemas posicionales (numerales)

- Binario
- Decimal
- Hexadecimal

233										/ 2
1	116									/ 2
	0	58								/ 2
		0	29							/ 2
			1	14						/ 2
				0	7					/ 2
					1	3				/ 2
						1	1			/ 2
							1			

De decimal a binario

Sistemas posicionales (numerales)

- Binario

- Decimal

- Hexadecimal

233									/ 2
1	116								/ 2
	0	58							/ 2
		0	29						/ 2
			1	14					/ 2
				0	7				/ 2
					1	3			/ 2
						1	1		/ 2
							1		
1	0	0	1	0	1	1	1		

¿10010111 es 233 en base decimal?

10010111 es 233 en base decimal?


NO, 10010111 es 151.

De decimal a binario

Sistemas posicionales (numerales)

- Binary
- Decimal
- Hexadecimal

233										/ 2
1	116									/ 2
	0	58								/ 2
		0	29							/ 2
			1	14						/ 2
				0	7					/ 2
					1	3				/ 2
						1	1			/ 2
							1			
1	0	0	1	0	1	1	1			



Tenemos que revertir el resto.

Sistemas posicionales (numerales)

Algunas preguntas importantes

¿Cuántos valores pueden ser representados por n bits?

Sistemas posicionales (numerales)

Algunas preguntas importantes

¿Cuántos valores pueden ser representados por n bits? 2^n

Sistemas posicionales (numerales)

Algunas preguntas importantes

¿Cuántos valores pueden ser representados por n bits? 2^n

¿Cuántos bits se necesitan para representar m valores?

Sistemas posicionales (numerales)

Algunas preguntas importantes

¿Cuántos valores pueden ser representados por n bits? 2^n

¿Cuántos bits se necesitan para representar m valores? $\log_2(n)$ por exceso
 $\log_2(91) = 6.50779 = 7$

Sistemas posicionales (numerales)

Algunas preguntas importantes

¿Cuántos valores pueden ser representados por n bits? 2^n

¿Cuántos bits se necesitan para representar m valores? $\log_2(m)$ por exceso
 $\log_2(91) = 6.50779 = 7$

Si usamos n bits, ¿corresponde el valor mínimo representable

Al número 0, ¿cuál es el valor numérico máximo representable?

Sistemas posicionales (numerales)

Algunas preguntas importantes

¿Cuántos valores pueden ser representados por n bits? 2^n

¿Cuántos bits se necesitan para representar m valores? $\log_2(m)$ por exceso
 $\log_2(91) = 6.50779 = 7$

Si usamos n bits, ¿cuál es el valor mínimo representable?

Al número 0, ¿cuál es el valor numérico máximo representable?

$$2^n - 1$$

Sistemas posicionales (numerales)

No se puede usar ninguna calculadora en el examen.

$$2^0 = 1$$

$$2^1 = 2$$

$$2^2 = 4$$

$$2^3 = 8$$

$$2^4 = 16$$

$$2^5 = 32$$

$$2^6 = 64$$

$$2^7 = 128$$

$$2^8 = 256$$

Sistemas posicionales (numerales)

Decimal	Binario	Octal	Hexadecimal
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F



Representación numérica (enteros)

Representación numérica (enteros)

Los números enteros se representan mediante un número fijo de bits. El rango de números representables depende del ancho y la convención de representación (es decir, no todos los números pueden ser representados, sólo aquellos que están dentro del rango).

Rango representable de números naturales (enteros sin signo): $[0, 2^n - 1]$, donde n es el ancho.

Ejemplo: Si usamos un ancho $n = 4$, se pueden representar los números naturales en el rango $[0, 15]$.

Representación numérica (enteros)

Convención de signos y magnitudes

El sistema decimal define el signo añadiendo un símbolo a la magnitud para representar el signo del número.

En binario, el signo está representado por un bit: 0 (+), 1 (-)

$(+16)_{10} = 00010000$ en binario con 8 bits

$(-16)_{10} = 10010000$ en binario con 8 bits

Rango de números representables con n bits: $[-2^{n-1}, 2^{n-1}]$.

Si usamos 8 bits con signo, podemos representar números enteros en el rango $[-127, +127]$.

Representación numérica (enteros)

Convención de signos y magnitudes

Para calcular el valor entero de un número con signo binario, debemos seguir los siguientes pasos:

1. Convertir la magnitud a base 10 usando los bits menos significativos $n-1$.
2. Agregue el signo: + (si comienza con 0) o - (si comienza con 1).

Ejemplo: $00111 = 7$, $11010 = -10$

- Las operaciones de suma y resta son más complicadas en binario, ya que los signos y las magnitudes deben tenerse en cuenta por separado.
- El cero tiene dos representaciones $[-0, +0]$.

Representación numérica (enteros)

Convención de signos y magnitudes

- Los números positivos se representan como una magnitud y un signo (por lo tanto, comenzando con 0).
- Los números negativos se representan como el complemento del 2 del número positivo correspondiente (comenzando con 1).

Rango de números representables usando n bits: $[-2^{n-1}, 2^{n-1}-1]$

Si usamos 8 bits, podemos representar números enteros en el rango $[-127, +127]$.

La representación posicional estándar de un número N en base b usando el complemento de dos se escribe de la siguiente manera::

$$N = (a_{n-1} a_{n-2} \dots a_1 a_0 a_{-1} \dots a_{-m})_b$$

Representación numérica (enteros)

COMPLEMENTO A DOS

El complemento a dos es una operación matemática para convertir reversiblemente un número binario positivo en un número binario negativo con valor equivalente (pero negativo), utilizando el dígito binario con el mayor valor posicional para indicar si el número binario es positivo o negativo (el signo).

$$C_b(N) = b^n - N$$

El total de números positivos será $2^{n-1}-1$ y el total de negativos será 2^{n-1} donde n es el número máximo de bits. El 0 contaría por separado.

$$\text{Si usamos 4 dígitos} \rightarrow C_{10}(0129) = 10^4 - 0129 = 9871$$

$$9871 + 0129 = 10000 = 10^4$$

Representación numérica (enteros)

De binario al complemento a dos

El complemento a dos es una operación matemática para convertir reversiblemente un número binario positivo en un número binario negativo con valor equivalente (pero negativo), utilizando el dígito binario con el mayor valor posicional para indicar si el número binario es positivo o negativo (el signo).

1. Quita el signo y usa el número positivo
2. Convierte el número decimal en un número binario.
3. Traspón todos los dígitos a partir de la parte significativa: ceros en unos y unos en ceros.
4. Añade 1 si el número es negativo.

Representación numérica (enteros)

Convertir de binario a complemento a dos

Decimal signed number	Positive binary	Negative binary
0	0000	0000
1	0001	1110
2	0010	1101
3	0011	1100
4	0100	1011
5	0101	1010
6	0110	1001
7	0111	1000
8		1111

Si usamos 4 bits, podemos representar números enteros en el rango [-8, +7].

Representación numérica (enteros)

Convertir -68 a binario en el complemento de dos

¿Cuántos bits necesito para representar -68 en binario?

Representación numérica (enteros)

Convertir -68 a binario en el complemento de dos

¿Cuántos bits necesito para representar -68 en binario?

8 bits

Necesitamos 7 bits para representar 68 pero necesito otro poco más para representar -68 para obtener un rango entre $[-128 + 127]$.

Representación numérica (enteros)

Convertir -68 a binario en el complemento de dos

¿Cuántos bits necesito para representar -68 en binario?

8 bits

Necesitamos 7 bits para representar 68 pero necesito otro poco más para representar -68 para obtener un rango entre $[-128 + 127]$.

68								/	2
0	34							/	2
	0	17						/	2
		1	8					/	2
			0	4				/	2
				0	2			/	2
					0	1		/	2
								/	2
0	0	1	0	0	0	1		/	2

68 = 01000100

← Agrega un 0 extra para tener 8 bits.

Representación numérica (enteros)

Convertir -68 a binario en el complemento de dos

¿Cuántos bits necesito para representar -68 en binario?

8 bits

Necesito 7 bits para representar 68 pero necesito otro poco más para representar -68 para obtener un rango entre $[-128 + 127]$.

68 = 01000100

Encontramos la parte más significativa.

El primer 1 comenzando a la derecha.

68 = 01000100

Volteamos todos los bits después del bit más significativo.

-68 = 10111100

Representación numérica (flotante)

Representación numérica (flotante)

Los números reales se representan en computadoras aproximadamente usando el estándar IEEE-754, usando un entero con una precisión fija, llamado mantisa, escalado por un exponente entero de una base fija (notación científica).

Ejemplo: Si queremos representar 14.345 como número de coma flotante en base 10:

$$14.345 = \underbrace{14345}_{\text{mantisa}} \times \underbrace{10^{-3}}_{\text{base}}$$

exponente

Representación numérica (flotante)

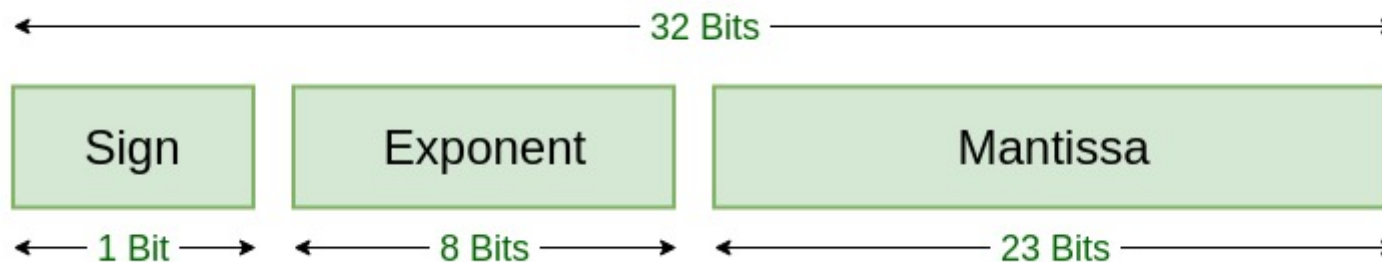
Los números de coma flotante están representados en las computadoras por una colección finita de bits compuesta de tres partes:

- Signo (1 bit simple): El bit de signo es 1 si un número es negativo y 0 si el número es positivo, como los enteros.
- Mantissa o significante o fracción (23 bits en coma flotante de precisión simple): La mantissa son los dígitos significativos del número en la representación de punto flotante.
- Exponente (coma flotante de precisión simple de 8 bits): El exponente es el radio que se eleva al determinar el valor de esa representación de

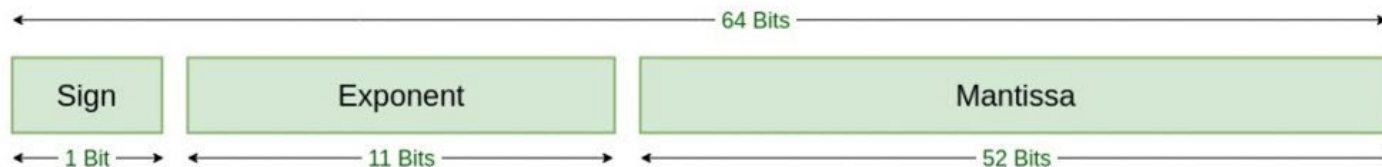
punto flotante.

Representación numérica (flotante)

Los números flotantes se dividen en dos basados en los tres componentes anteriores: precisión simple (32 bits) y precisión doble (64 bits).



Precisión simple



Doble precisión

Representación numérica (flotante)

Para convertir números decimales en representación de coma flotante IEEE 754:

1. Elija la representación de precisión: simple o doble.
2. Separar la parte entera y la parte decimal del número.
3. Convertir el número decimal en binario.
4. Convertir la parte decimal en binario.
5. Combinar las dos partes del número que se han convertido en binario.
6. Identificar el signo: 0 para números positivos y 1 para números negativos.

Representación numérica (flotante)

Para convertir números decimales en representación de coma flotante IEEE 754:

7. Convertir el número binario en notación científica en base 2.
 - Para convertir el número en notación científica en base 2, debemos mover el punto decimal hacia la izquierda hasta que esté a la derecha del primer bit para crear la mantisa normalizada..
8. Calcular el exponente basándose en la precisión.

Representación numérica (flotante)

Convertir 88.125 a binario usando una sola representación de coma flotante IEEE 754

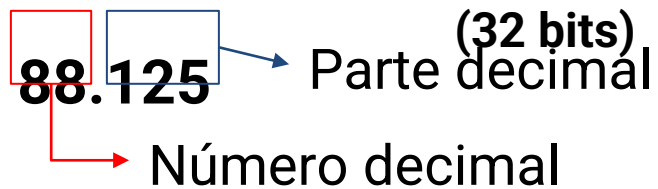
¿Qué precisión debemos usar?

Individual (32 bits)

Representación numérica (flotante)

Convertir 88.125 a binario usando una sola representación de coma flotante IEEE 754

¿Qué precisión debemos usar?



Representación numérica (flotante)

Convertir 88.125 a binario usando una sola representación de coma flotante IEEE 754

¿Qué precisión debemos usar?

88.125 (32 bits)
Parte decimal
Número decimal

88								/	2
0	44							/	2
	0	22						/	2
		0	11					/	2
			1	5				/	2
				1	2			/	2
					0	1		/	2
								/	2
0	0	0	1	1	0	1			

Representación numérica (flotante)

Convertir 88.125 a binario usando una sola representación de coma flotante IEEE 754

¿Qué precisión debemos usar?

88.125 (32 bits)
Parte decimal
Número decimal

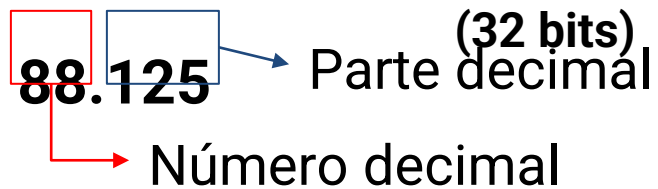
88								/	2
0	44							/	2
	0	22						/	2
		0	11					/	2
			1	5				/	2
				1	2			/	2
					0	1		/	2
								/	2
0	0	0	1	1	0	1			

88 = 1011000

Representación numérica (flotante)

Convertir 88.125 a binario usando una sola representación de coma flotante IEEE 754

¿Qué precisión debemos usar?



0.125						*	2
0	0.25					*	2
	0	0.5				*	2
		1	1.0			*	2

$$0.125 = 001$$

Representación numérica (flotante)

Convertir 88.125 a binario usando una sola representación de coma flotante IEEE 754

¿Qué precisión debemos usar?

(32 bits)

88.125

1011000.001 x 2⁰

Mueve el decimal 6 lugares a la izquierda para dejar un 1.

1.011000001 x 2⁰⁺⁶

Hay sesgos establecidos para la precisión simple y doble. El sesgo de exponente para la precisión simple es 127, lo que significa que debemos agregarle el exponente de base 2 que se encontró anteriormente. Por lo tanto, el exponente que usará es 127 + 6 que es 133.

$$127 + 6 = 133$$

Representación numérica (flotante)

Convertir 88.125 a binario usando una sola representación de coma flotante IEEE 754

¿Qué precisión debemos usar?

(32 bits)

$$127 + 6 = 133$$

[illegible]

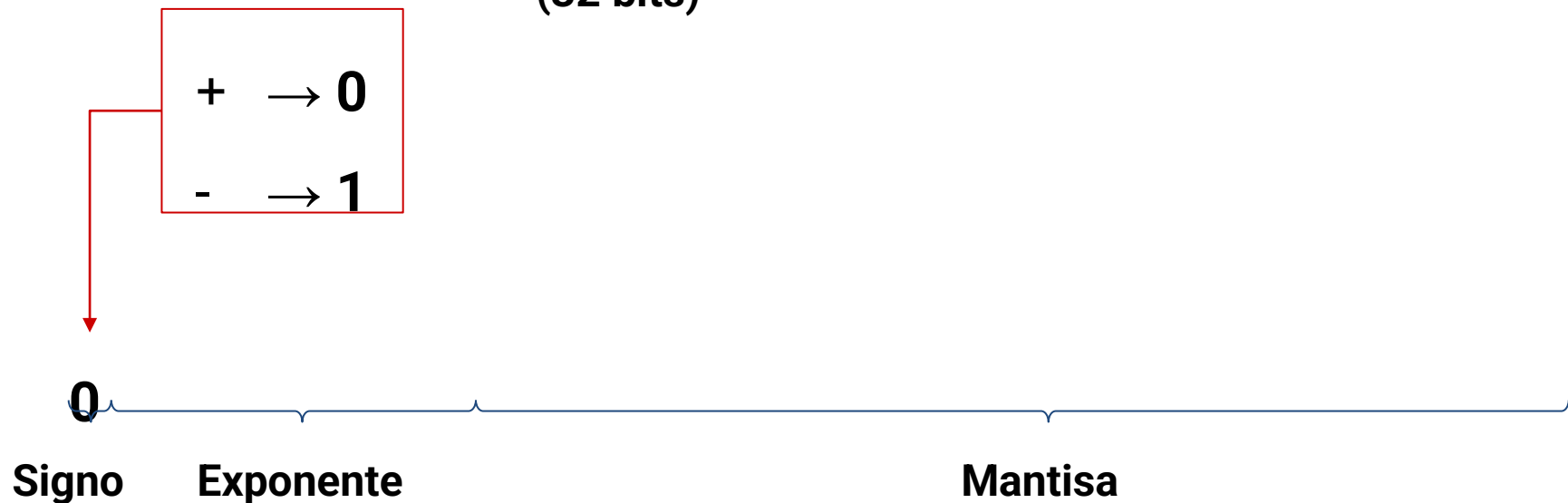
133 = 10000101

Representación numérica (flotante)

Convertir 88.125 a binario usando una sola representación de coma flotante IEEE 754

¿Qué precisión debemos usar?

(32 bits)

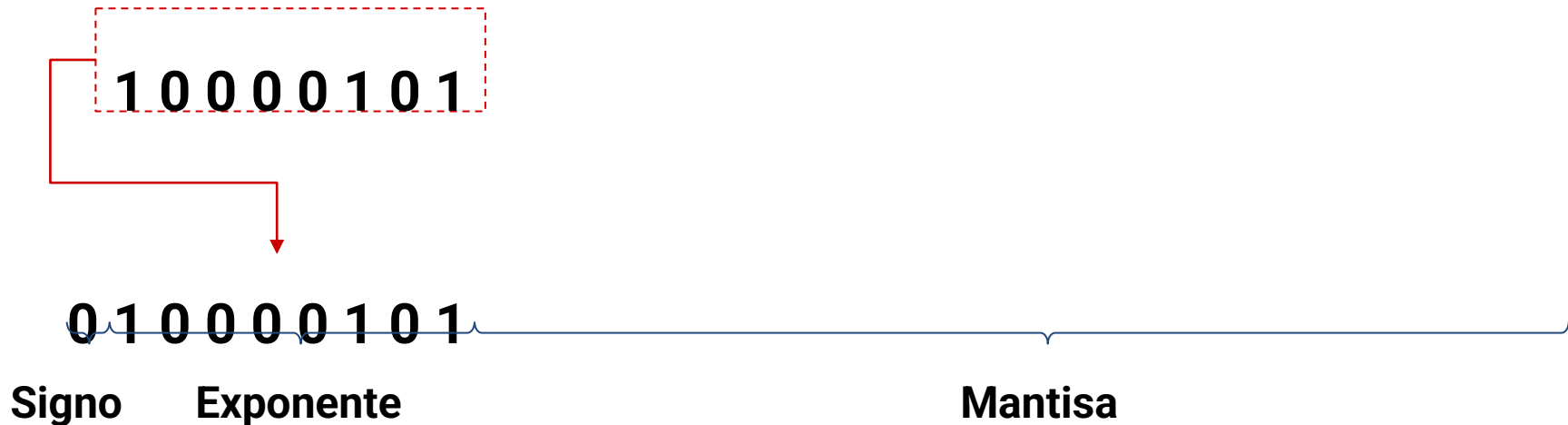


Representación numérica (flotante)

Convertir 88.125 a binario usando una sola representación de coma flotante IEEE 754

¿Qué precisión debemos usar?

(32 bits)



Representación numérica (flotante)

Convertir 88.125 a binario usando una sola representación de coma flotante IEEE 754

¿Qué precisión debemos usar?

(32 bits)

1.011000001 x 2⁶

0 1 0 0 0 0 1 0 1 0 1 1 0 0 0 0 0 1

Signo

Exponente

Mantisa

Simplemente soltamos el 1 a la izquierda y copiamos la parte decimal del número que se está multiplicando por 2.

Representación numérica (flotante)

Convertir 88.125 a binario usando una sola representación de coma flotante IEEE 754

¿Qué precisión debemos usar?

(32 bits)

1.011000001 x 2⁶

0 1 0 0 0 0 1 0 1 0 1 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Signo

Exponente

Mantisa

Completamos con ceros.

Representación numérica (flotante)

Convertir 25.025 a binario usando una sola representación de coma flotante IEEE 754

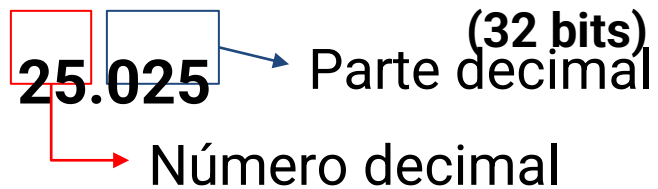
¿Qué precisión debemos usar?

(32 bits)

Representación numérica (flotante)

Convertir 25.025 a binario usando una sola representación de coma flotante IEEE 754

¿Qué precisión debemos usar?



Representación numérica (flotante)

Representación numérica (flotante)

Representación numérica (flotante)

Convertir 25.025 a binario usando una sola representación de coma flotante IEEE 754

¿Qué precisión debemos usar?

Diagram illustrating the components of a floating-point number (25.025):

- The integer part (25) is labeled **Número decimal** (Integer part).
- The fractional part (.025) is labeled **Parte decimal (32 bits)** (Fractional part (32 bits)).

0.025									*	2
0	0.05								*	2
	0	0.1							*	2
		0	0.2						*	2
			0	0.4					*	2
				1	0.6				*	2
					1	0.2			*	2
						0	0.4		*	2

Representación numérica (flotante)

Convert 25.025 to binary using single IEEE 754 Floating Point Representation

¿Qué precisión debemos usar?

Diagram illustrating the conversion of a decimal number to a floating-point format. The number **25.025** is shown. The integer part **25** is highlighted with a red box and labeled **Número decimal**. The fractional part **.025** is highlighted with a blue box and labeled **Parte decimal (32 bits)**.

0.4										*	2
0	0.8									*	2
	1	0.6								*	2
		1	0.2							*	2
			0	0.4						*	2
				0	0.8					*	2
					1	0.6				*	2
						1	0.2			*	2
							0	0.4		*	2

Representación numérica (flotante)

Convertir 25.025 a binario usando una sola representación de coma flotante IEEE 754

¿Qué precisión debemos usar?

(32 bits)

25.025

11001.000001100110011001100110 x 2⁰

Movemos el decimal 4 lugares a la izquierda para dejar un 1 allí.

1.1001000001100110011001100110 x 2⁰⁺⁴

$$127 + 4 = 131$$

Hay sesgos establecidos para la precisión simple y doble. El sesgo de exponente para la precisión simple es 127, lo que significa que debemos agregarle el exponente de base 2 que se encontró anteriormente. Por lo tanto, el exponente que usará es $127 + 4$, que es 131.

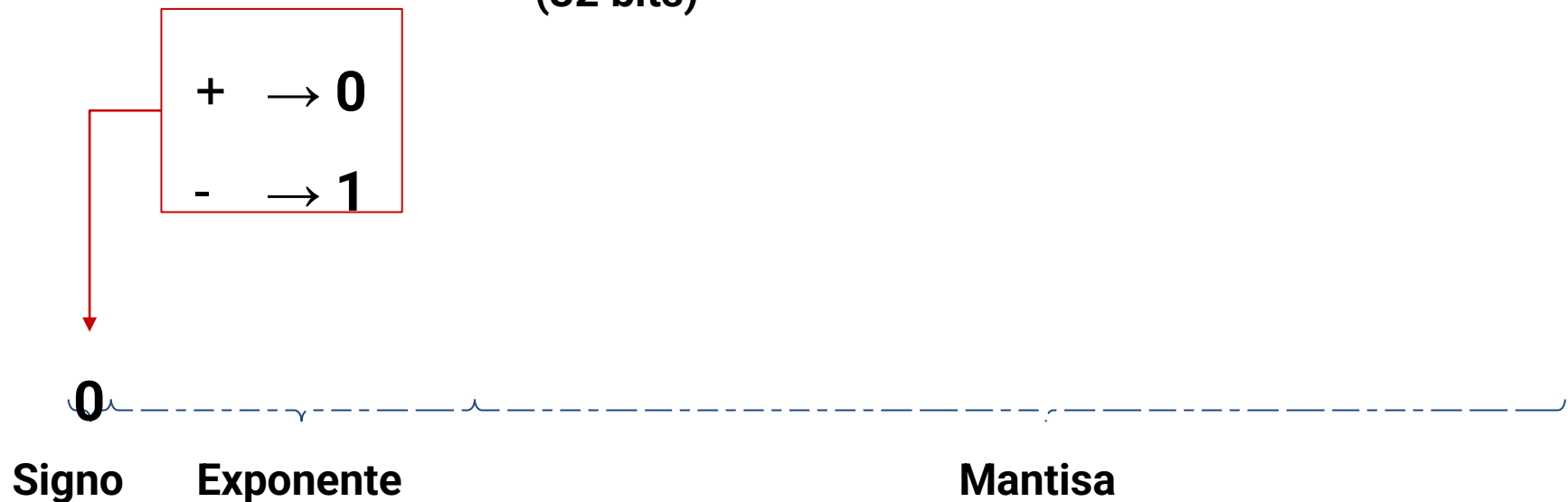
Representación numérica (flotante)

Representación numérica (flotante)

Convertir 25.025 a binario usando una sola representación de coma flotante IEEE 754

¿Qué precisión debemos usar?

(32 bits)

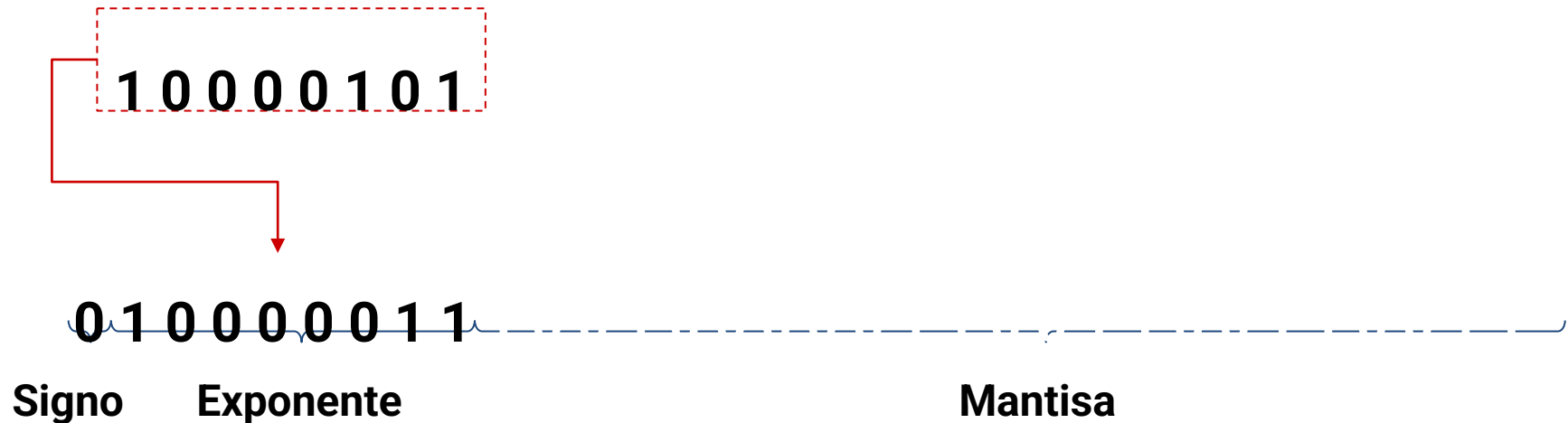


Representación numérica (flotante)

Convertir 25.025 a binario usando una sola representación de coma flotante IEEE 754

¿Qué precisión debemos usar?

(32 bits)



Representación numérica (flotante)

Convertir 25.025 a binario usando una sola representación de punto flotante IEEE 754

¿Qué precisión debemos usar?

(32 bits)

1.10010000011001100110011001100110 $\times 2^6$

0 1 0 0 0 0 0 1 1 1 0 0 1 0 0 0 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1

Signo

Exponente

Mantisa

Simplemente soltamos el 1 a la izquierda y copiamos la parte decimal del número que se está multiplicando por 2.

Representación numérica (flotante)

¿Convertir el número representado en un solo coma flotante IEEE 754 a decimal?

11000010100010000000000000000000

Representación numérica (flotante)

¿Convertir el número representado en un solo punto flotante IEEE 754 a decimal?

11000010100010000000000000000000

1 1 0 0 0 0 1 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0

Signo

Exponente

Mantisa

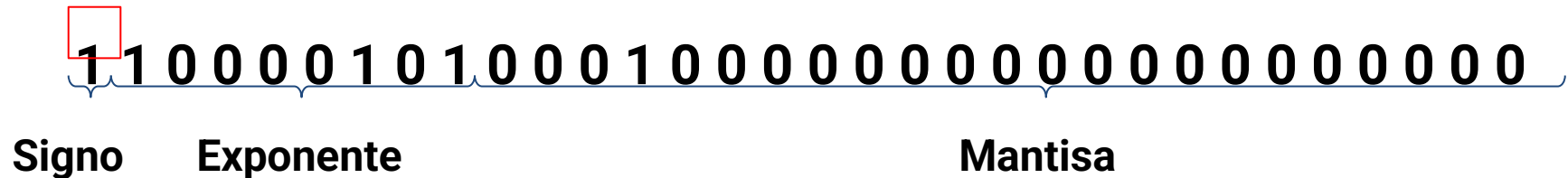
Dividimos el número en sus partes (signo, exponente y mantisa).

Representación numérica (flotante)

¿Convertir el número representado en un solo coma flotante IEEE 754 a decimal?

11000010100010000000000000000000

El número es negativo.



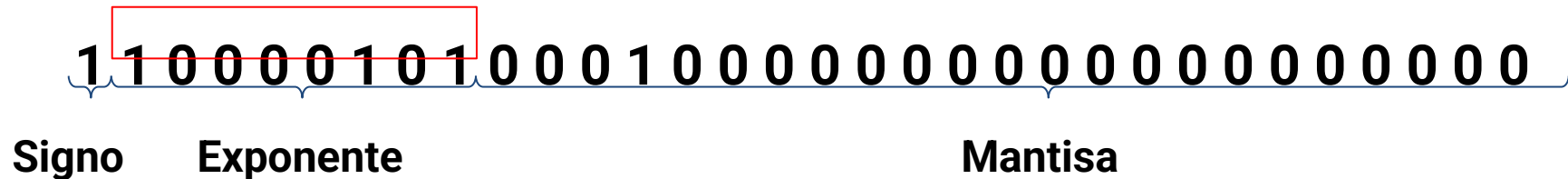
Identificamos el signo: $0 \rightarrow$ positivo y $1 \rightarrow$ negativo.

Representación numérica (flotante)

¿Convertir el número representado en un solo coma flotante IEEE 754 a decimal?

11000010100010000000000000000000

Transformamos el exponente a decimal para encontrar la normalización de la mantisa.



Calculamos el exponente $\rightarrow 1\ 0\ 0\ 0\ 0\ 1\ 0\ 1 = 133$

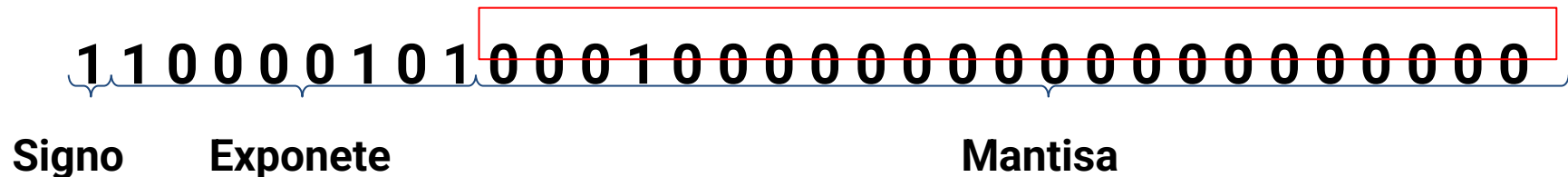
$133 - 127 = 6$

Representación numérica (flotante)

¿Convertir el número representado en un solo coma flotante IEEE 754 a decimal?

11000010100010000000000000000000

Calculamos el número desnormalizando la mantisa y convirtiéndola a decimal.



1.000100000000000000000000

Movemos la señal decimal 6 lugares a la derecha.

Siempre debe insertar 1 a la izquierda.

Representación numérica (flotante)

¿Convertir el número representado en un solo coma flotante IEEE 754 a decimal?

11000010100010000000000000000000

Calculamos el número desnormalizando la mantisa y convirtiéndola a decimal.

1 **10000101** **00010000000000000000000000000000**

Signo Exponete Mantisa

1000100.000000000000000000

Convertimos a decimal.

$$\left. \begin{array}{l} 1000100 = 68 \\ 00000000000000000000 = 0 \end{array} \right\} - 68.0$$

Representación de codificación

Representación de codificación

Codificaciones decimales y alfanuméricas

Una codificación es un conjunto de cadenas de n bits sobre las cuales se establece una convención en la que cada cadena representa un número u otro tipo de información..

- La codificación numérica o código representa información numérica.
- La codificación alfanumérica o código representa números, letras y signos de puntuación.
- Codificación de errores o información de código para que ciertos errores en el almacenamiento, recuperación o transmisión de información puedan detectarse y corregirse.

Representación de codificación

Decimal codificado binario

Binary Coded Decimal (BCD) se utiliza para la representación binaria de números en base decimal. Cada dígito decimal está representado por una combinación de 4 bits y cada número como una cadena de dígitos.

734 = 0111 0011 0100

22 = 0010 0010

Decimal	Binay (BCD)			
	8	4	2	1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Códigos alfanuméricos

Códigos alfanuméricos se utilizan para representar texto donde se asigna un código (cadena de bits) a cada carácter. Los personajes generalmente se agrupan en 5 categorías:

- Caracteres alfanuméricos: A, B, C,Z, a, b, c,, z
- Caracteres numéricos: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- Caracteres especiales: () + - , & > < Ñ ñ # Ç ç SP ...
- Caracteres geométricos y gráficos: | ☞ ¶
- Personajes de control: entrar, espacio, ...

Representación de codificación

Código Ascii

ASCII (American Standard Code for Information Interchange) code es uno de los más antiguos (1968). Fue creado para representar los caracteres y símbolos del idioma inglés. El código ASCII básico utiliza 7 bits (cada carácter o símbolo está representado por 7 bits).

C/Vega,7

C	/	V	e	g	a	,	7
1000011	0101111	1010110	1100101	1100111	1100001	0101100	0110111

	0	1	2	3	4	5	6	7
0	NUL	DLE	space	0	@	P	`	p
1	SOH	DC1 XON	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3 XOFF	#	3	C	S	c	s
4	EOT	DC4	\$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(8	H	X	h	x
9	HT	EM)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[k	{
C	FF	FS	,	<	L	\	l	
D	CR	GS	-	=	M]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	del

Corresponde a la estandarización ANSI x 3.4 - 1968 o ISO 646.

Representación de codificación

Código Ascii

ASCII (American Standard Code for Information Interchange) code es uno de los más antiguos (1968). Fue creado para representar los caracteres y símbolos del idioma inglés. El código ASCII básico utiliza 7 bits (cada carácter o símbolo está representado por 7 bits).

C/Vega,7

C	/	V	e	g	a	,	7
1000011	0101111	1010110	1100101	1100111	1100001	0101100	0110111
43	2F	56	65	67	61	2C	37

Corresponde a la estandarización ANSI x 3.4 - 1968 o ISO 646.

Decimal	Binary	Hexadecimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Representación de codificación

Código Ascii

Existen diferentes versiones extendidas del código ascii usando 8 bits.

Name	ISO family	Geographical area
Latin-1	ISO 8859-1	Western and Eastern Europe
Latin-2	ISO 8859-2	Central and Eastern Europe
Latin-3	ISO 8859-3	Southern Europe, Maltese and Esperanto
Latin-4	ISO 8859-4	North europe
Latin/cyrillic	ISO 8859-5	Slavic languages
Latin/arabic	ISO 8859-6	Arabic languages
Latin/greek	ISO 8859-7	Modern greek

ISO/IEC 8859 es una serie conjunta de normas ISO e IEC para codificaciones de caracteres de 8 bits.

Representación de codificación

Código Ascii extendido

ASCII extendido utiliza una codificación de caracteres de ocho bits que incluye (la mayoría de) los caracteres ASCII de siete bits, además de caracteres adicionales.

ASCII control characters			ASCII printable characters			Extended ASCII characters			
00	NULL	(Null character)	32	space	64	@	96	`	
01	SOH	(Start of Header)	33	!	65	A	97	a	
02	STX	(Start of Text)	34	"	66	B	98	b	
03	ETX	(End of Text)	35	#	67	C	99	c	
04	EOT	(End of Trans.)	36	\$	68	D	100	d	
05	ENQ	(Enquiry)	37	%	69	E	101	e	
06	ACK	(Acknowledgement)	38	&	70	F	102	f	
07	BEL	(Bell)	39	'	71	G	103	g	
08	BS	(Backspace)	40	(72	H	104	h	
09	HT	(Horizontal Tab)	41)	73	I	105	i	
10	LF	(Line feed)	42	*	74	J	106	j	
11	VT	(Vertical Tab)	43	+	75	K	107	k	
12	FF	(Form feed)	44	,	76	L	108	l	
13	CR	(Carriage return)	45	-	77	M	109	m	
14	SO	(Shift Out)	46	.	78	N	110	n	
15	SI	(Shift In)	47	/	79	O	111	o	
16	DLE	(Data link escape)	48	0	80	P	112	p	
17	DC1	(Device control 1)	49	1	81	Q	113	q	
18	DC2	(Device control 2)	50	2	82	R	114	r	
19	DC3	(Device control 3)	51	3	83	S	115	s	
20	DC4	(Device control 4)	52	4	84	T	116	t	
21	NAK	(Negative acknow.)	53	5	85	U	117	u	
22	SYN	(Synchronous idle)	54	6	86	V	118	v	
23	ETB	(End of trans. block)	55	7	87	W	119	w	
24	CAN	(Cancel)	56	8	88	X	120	x	
25	EM	(End of medium)	57	9	89	Y	121	y	
26	SUB	(Substitute)	58	:	90	Z	122	z	
27	ESC	(Escape)	59	;	91	[123	{	
28	FS	(File separator)	60	<	92	\	124		
29	GS	(Group separator)	61	=	93]	125	}	
30	RS	(Record separator)	62	>	94	^	126	~	
31	US	(Unit separator)	63	?	95	_			
127	DEL	(Delete)							
128	Ç		160	à		192	Ł	224	Ó
129	ü		161	í		193	ł	225	ô
130	é		162	ó		194	Ł	226	ö
131	â		163	ú		195	ł	227	õ
132	ä		164	ñ		196	Ł	228	ö
133	å		165	Ń		197	ł	229	õ
134	ä		166	•		198	Ł	230	µ
135	ç		167	°		199	Ł	231	þ
136	è		168	ˆ		200	Ł	232	þ
137	ë		169	©		201	Ł	233	Ü
138	è		170	¬		202	Ł	234	Ü
139	ï		171	½		203	Ł	235	Ü
140	í		172	¼		204	Ł	236	ý
141	ì		173	ı		205	Ł	237	ÿ
142	À		174	«		206	Ł	238	—
143	Á		175	»		207	Ł	239	·
144	Â		176	•		208	Ł	240	≡
145	Æ		177	•		209	Đ	241	±
146	Æ		178	•		210	Ê	242	•
147	ø		179	•		211	Ê	243	¾
148	ø		180	•		212	Ê	244	¶
149	ø		181	•		213	ı	245	§
150	ü		182	À		214	ı	246	÷
151	ü		183	Á		215	ı	247	•
152	ÿ		184	Â		216	ı	248	•
153	Ö		185	Ë		217	ı	249	•
154	Ü		186	•		218	•	250	•
155	ø		187	ı		219	ı	251	ı
156	£		188	•		220	ı	252	•
157	ø		189	ç		221	ı	253	•
158	x		190	¥		222	ı	254	•
159	f		191	Ÿ		223	ı	255	nbsp

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
128	80	Ç	160	A0	á	192	C0	Ł	224	E0	α
129	81	ü	161	A1	í	193	C1	ł	225	E1	β
130	82	é	162	A2	ó	194	C2	Ł	226	E2	Γ
131	83	â	163	A3	ú	195	C3	ł	227	E3	π
132	84	ä	164	A4	ñ	196	C4	Ł	228	E4	Σ
133	85	å	165	A5	Ń	197	C5	ł	229	E5	σ
134	86	ä	166	A6	•	198	C6	Ł	230	E6	μ
135	87	ç	167	A7	°	199	C7	Ł	231	E7	ι
136	88	è	168	A8	ˆ	200	C8	Ł	232	E8	φ
137	89	ë	169	A9	©	201	C9	Ł	233	E9	Θ
138	8A	è	170	AA	¬	202	CA	Ł	234	EA	Ω
139	8B	ï	171	AB	½	203	CB	Ł	235	EB	δ
140	8C	î	172	AC	¼	204	CC	Ł	236	EC	∞
141	8D	ı	173	AD	ı	205	CD	Ł	237	ED	φ
142	8E	À	174	AE	«	206	CE	Ł	238	EE	ε
143	8F	Á	175	AF	»	207	CF	Ł	239	EF	∅
144	90	Â	176	B0	•	208	DO	Ł	240	FO	≡
145	91	Æ	177	B1	•	209	D1	Ł	241	F1	±
146	92	Æ	178	B2	•	210	D2	Ł	242	F2	±
147	93	ø	179	B3	•	211	D3	Ł	243	F3	≤
148	94	ø	180	B4	•	212	D4	Ł	244	F4	
149	95	ø	181	B5	•	213	D5	Ł	245	F5	
150	96	ü	182	B6	•	214	D6	Ł	246	F6	•
151	97	ü	183	B7	•	215	D7	Ł	247	F7	•
152	98	ÿ	184	B8	•	216	D8	Ł	248	F8	•
153	99	Ö	185	B9	•	217	D9	Ł	249	F9	•
154	9A	Ü	186	BA	•	218	DA	Ł	250	FA	•
155	9B	ø	187	BB	•	219	DB	Ł	251	FB	√
156	9C	£	188	BC	•	220	DC	Ł	252	FC	•
157	9D	¥	189	BD	•	221	DD	Ł	253	FD	•
158	9E	₣	190	BE	•	222	DE	Ł	254	FE	•
159	9F	f	191	BF	•	223	DF	Ł	255	FF	•

Representación de codificación

Del texto al binario

H	e	l	l	o		W	o	r	l	d
01001000	01100101	01101100	01101100	01101111	00100000	01010111	01101111	01110010	01101100	01100100
48	65	6C	6C	6F	20	57	6F	72	6C	64

La conversión de caracteres a binarios se simplifica cuando se convierten primero a hexadecimales y, posteriormente, se transforman en binarios.

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
128	80	Ç	160	A0	á	192	C0	Ł	224	E0	α
129	81	ü	161	A1	í	193	C1	ł	225	E1	β
130	82	é	162	A2	ó	194	C2	Ť	226	E2	Γ
131	83	â	163	A3	ú	195	C3	Ŧ	227	E3	π
132	84	à	164	A4	ñ	196	C4	ŷ	228	E4	Σ
133	85	á	165	A5	Ñ	197	C5	Ź	229	E5	σ
134	86	ã	166	A6	*	198	C6	Ż	230	E6	μ
135	87	ç	167	A7	o	199	C7	Ł	231	E7	ι
136	88	ê	168	A8	¿	200	C8	ł	232	E8	φ
137	89	ë	169	A9	ı	201	C9	Ł	233	E9	Θ
138	8A	è	170	AA	ı	202	CA	Ł	234	EA	Ω
139	8B	ı	171	AB	½	203	CB	Ł	235	EB	δ
140	8C	î	172	AC	¼	204	CC	Ł	236	EC	∞
141	8D	ï	173	AD	ı	205	CD	Ł	237	ED	ψ
142	8E	Ā	174	AE	ı	206	CE	Ł	238	EE	ε
143	8F	Ā	175	AF	ı	207	CF	Ł	239	EF	Ω
144	90	E	176	B0	ı	208	D0	Ł	240	F0	≡
145	91	æ	177	B1	ı	209	D1	Ł	241	F1	±
146	92	Æ	178	B2	ı	210	D2	Ł	242	F2	≥
147	93	ø	179	B3	ı	211	D3	Ł	243	F3	≤
148	94	ö	180	B4	ı	212	D4	Ł	244	F4	ı
149	95	õ	181	B5	ı	213	D5	Ł	245	F5	ı
150	96	ü	182	B6	ı	214	D6	Ł	246	F6	ı
151	97	ù	183	B7	ı	215	D7	Ł	247	F7	ı
152	98	ÿ	184	B8	ı	216	D8	Ł	248	F8	ı
153	99	Ů	185	B9	ı	217	D9	Ł	249	F9	ı
154	9A	Ů	186	BA	ı	218	DA	Ł	250	FA	ı
155	9B	ƒ	187	BB	ı	219	DB	Ł	251	FB	ı
156	9C	£	188	BC	ı	220	DC	Ł	252	FC	ı
157	9D	¥	189	BD	ı	221	DD	Ł	253	FD	ı
158	9E	₣	190	BE	ı	222	DE	Ł	254	FE	ı
159	9F	ƒ	191	BF	ı	223	DF	Ł	255	FF	ı

Unicode

El estándar **Unicode** es un estándar de información para la codificación, representación y manejo consistentes del texto expresado en la mayoría de los sistemas de escritura del mundo. Este código fue diseñado para seguir estas propiedades principales:

- Universalidad: cubre la mayoría de las lenguas escritas existentes.
- Singularidad: Cada símbolo tiene un código único.
- Uniformidad: Cada carácter está representado por 8 o 16 bits dependiendo de la versión unicode.

Representación de codificación

Unicode

Los códigos se dividen en 4 grupos o zonas, como se muestra en la tabla.

Zone	Codes (HEX)	Symbols	Characters
A	0000 - 3FFF	Basic Latin (ASCII), Latin-1 and other Latin characters, Greek, Cyrillic, Armenian, Hebrew, Arabic, Syrian, Chinese, Japanese and Korean phonetic characters	8192
I	4000 - 9FFF	Chinese, Japanese and Korean ideograms	24576
O	A000 - DFFF	Not assigned	16384
R	E000 - FFFF	Local and user-specific characters	8192

Anexos

Base change

<https://youtu.be/5WtLFbriEEE>

Integer numbers

<https://youtu.be/B7SpmkW0ITs>

Two's complement

<https://youtu.be/UTVuROxztuQ>

IEEE (Floating point)

<https://youtu.be/HcjXH9WGmAU>

Some tips

<https://youtu.be/5TIUWLxOWzU>