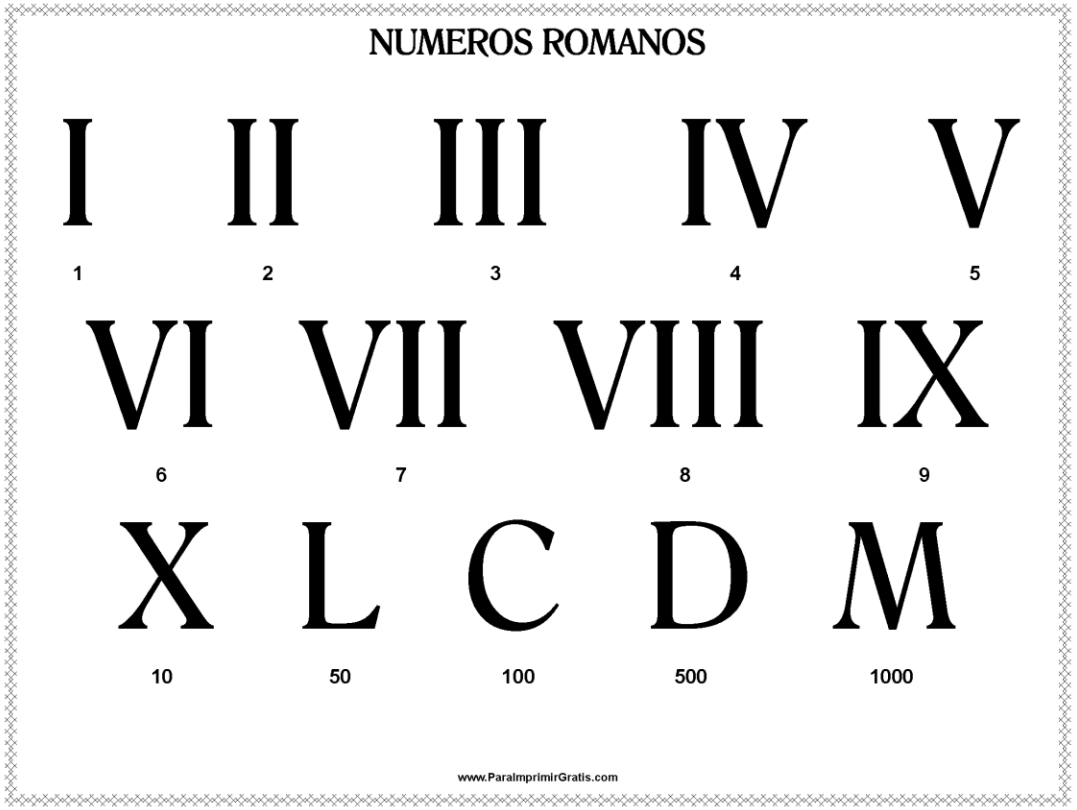
Representación interna de los datos

Introducción a los sistemas informáticos

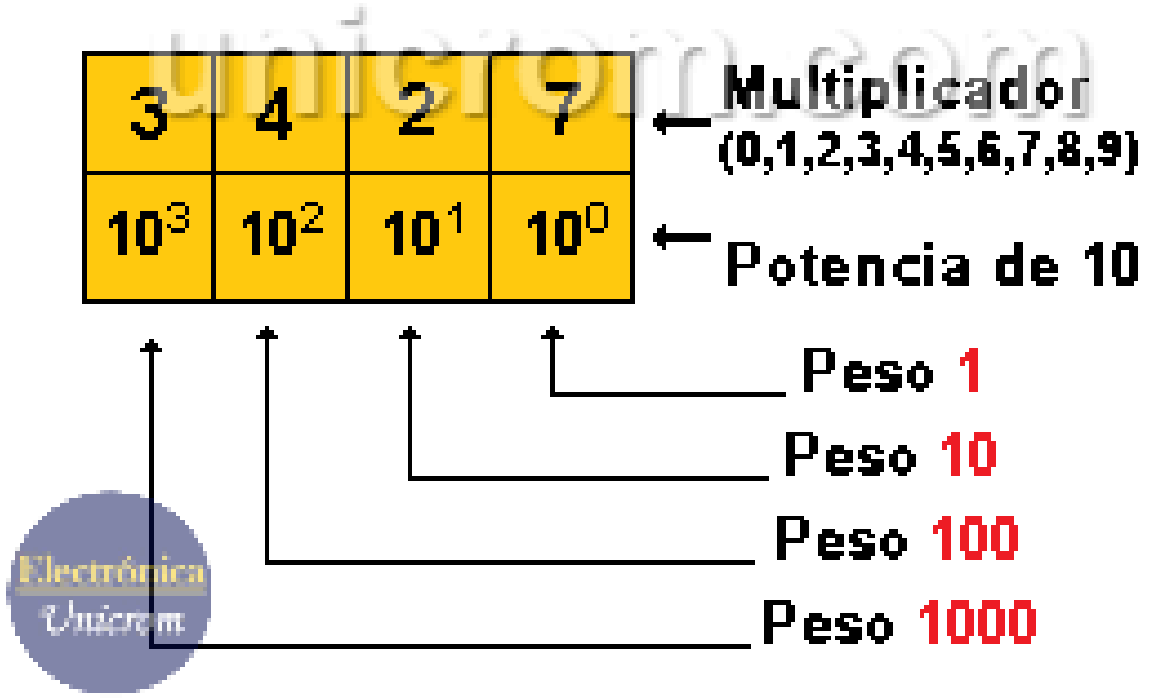
Sistema de numeración romano

- Antiguamente se utilizaban los números romanos
- Resultaban poco prácticos ya que no permitían representar cualquier número
- Además, realizar operaciones aritméticas no resultaba fácil



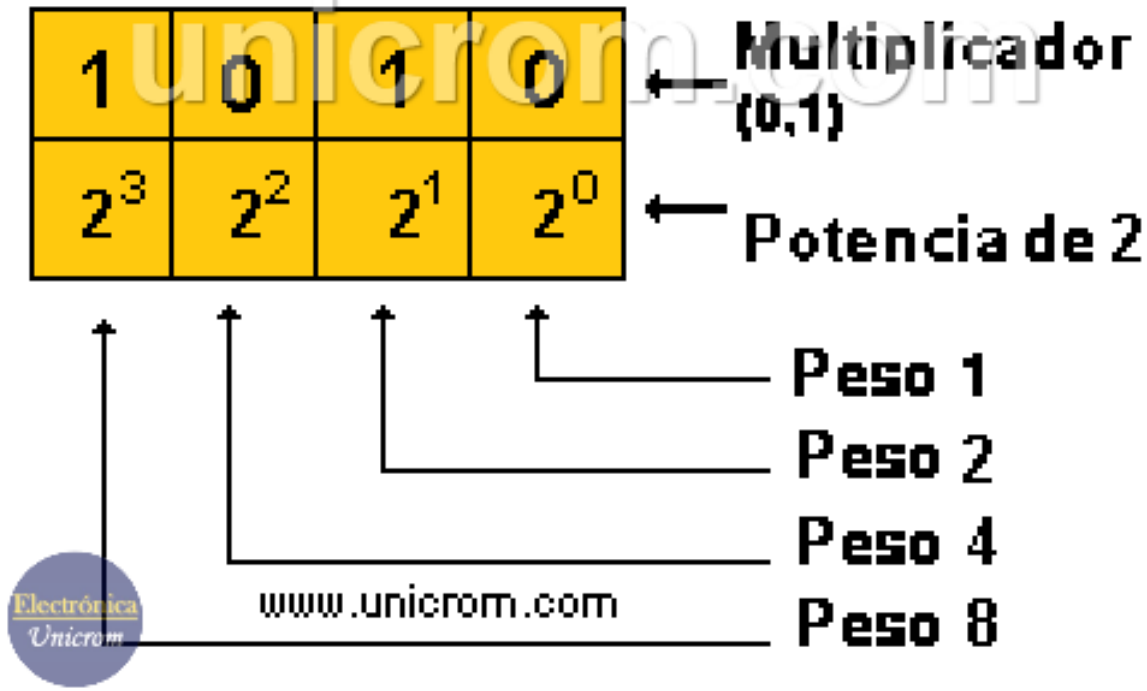
Sistema de numeración decimal

- Entonces unos matemáticos árabes idearon el sistema de numeración posicional decimal
- En este sistema cada posición del número tiene un peso que va creciendo de derecha a izquierda
- Estos sistemas posicionales fueron una revolución para las matemáticas



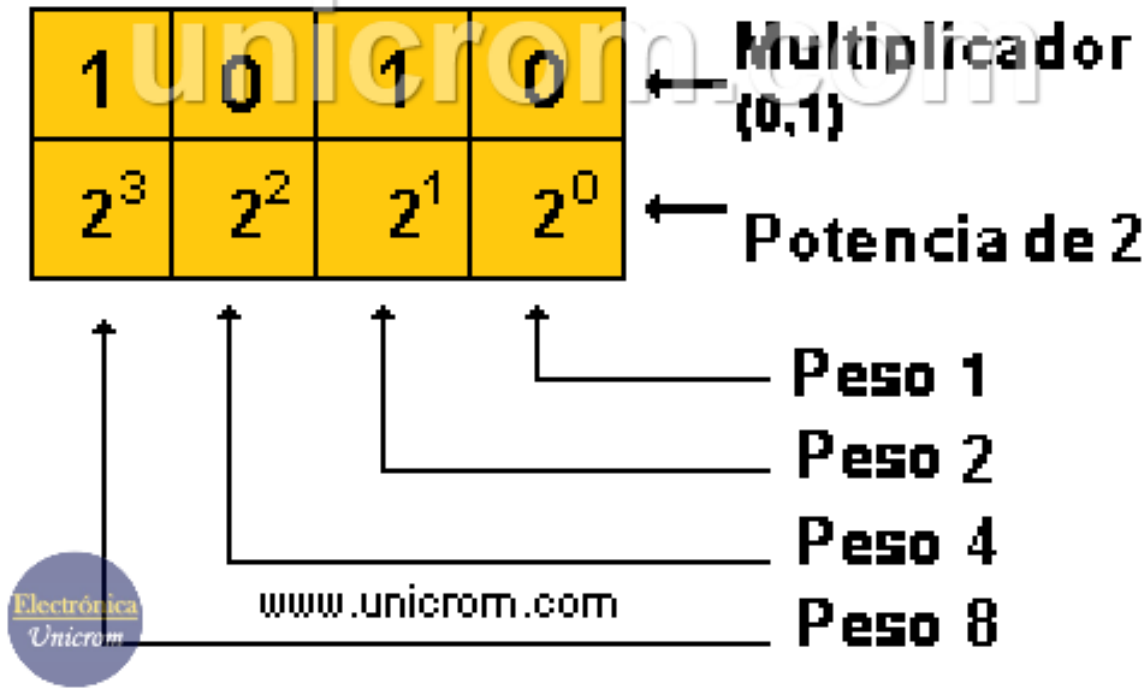
Sistema de numeración binario

- El sistema de numeración decimal se basaba en las posiciones y en tener 10 cifras disponibles (0, 1, 2, 3, 4, 5, 6, 7, 8 y 9)
- Los sistemas informáticos usan esta misma idea, pero en lugar de utilizar diez cifras distintas solo usan dos (0 y 1).
- Este sistema de numeración es lo que conocemos como binario



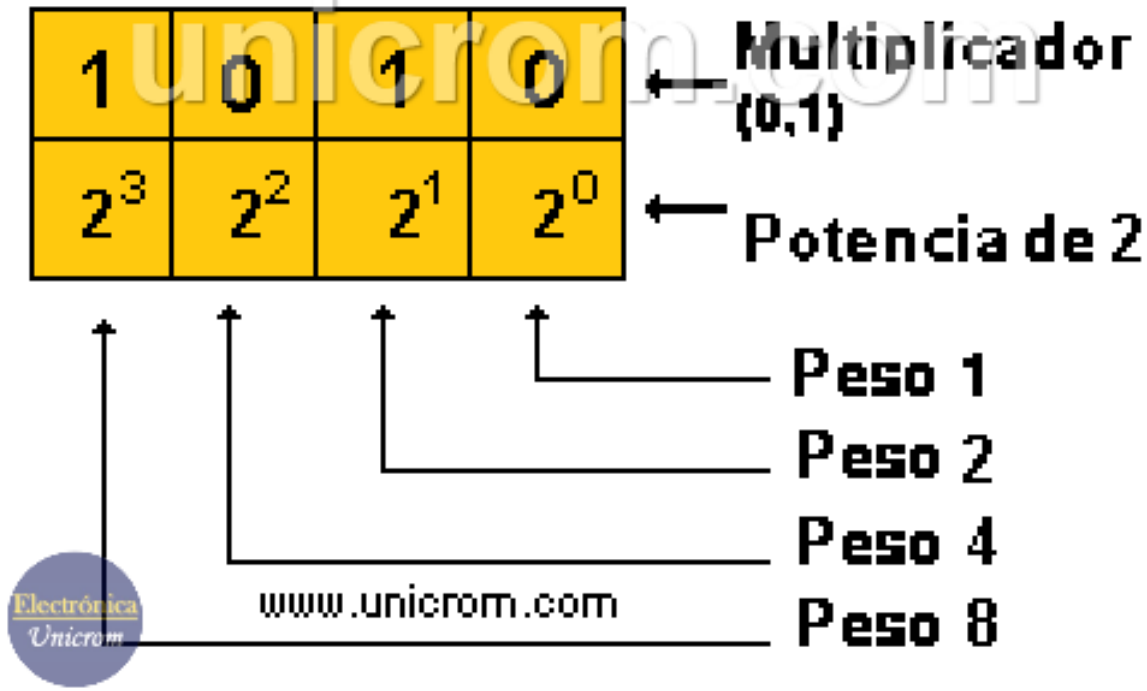
Sistema de numeración binario

- Todo número decimal tiene su correspondiente traducción a un número binario y viceversa, solo cambia la cantidad de cifras necesarias para representarlos
- Por ejemplo
  - 1 = 1
  - 2 = 10
  - 3 = 11
  - 4 = 100



Sistema de numeración binario

- En informática los datos que se manejan se almacenan como números en binario.
- Cada una de las cifras de los números es lo que llamamos un bit (que puede tomar como valores 0 o 1)
- Cuando se agrupan 8 bits para formar un número más grande tenemos un byte



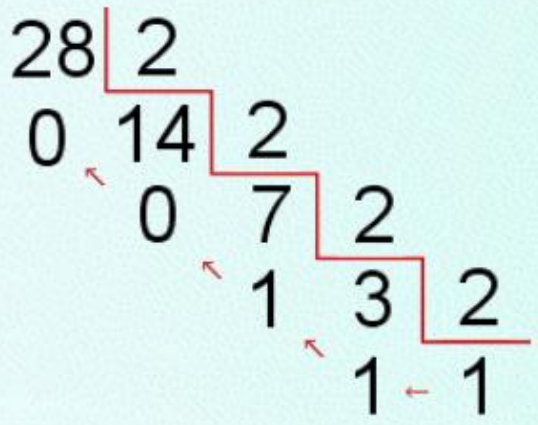
Sistema de numeración hexadecimal

- Como en el mundo de la informática se informática se trabaja mucho a nivel de byte (8 bits) resulta útil utilizar un sistema de numeración que nos permita expresar los bytes utilizando menos cifras
- Para ello tenemos el sistema hexadecimal que se basa en 16 cifras que van del 0 al 9 y a partir de ahí las letras de la A a la F incluidas
- El sistema hexadecimal nos permite expresar el valor de un byte con tan solo dos cifras. Por ejemplo: 0A

DECIMAL	BINARIO	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

Conversión de decimal a binario

Procedimiento:
- Dividir entre 2 sucesivamente
- Apuntar el resultado y el resto de cada operación
- Apuntar a lista de ceros y unos de abajo a arriba



$28=11100_2$

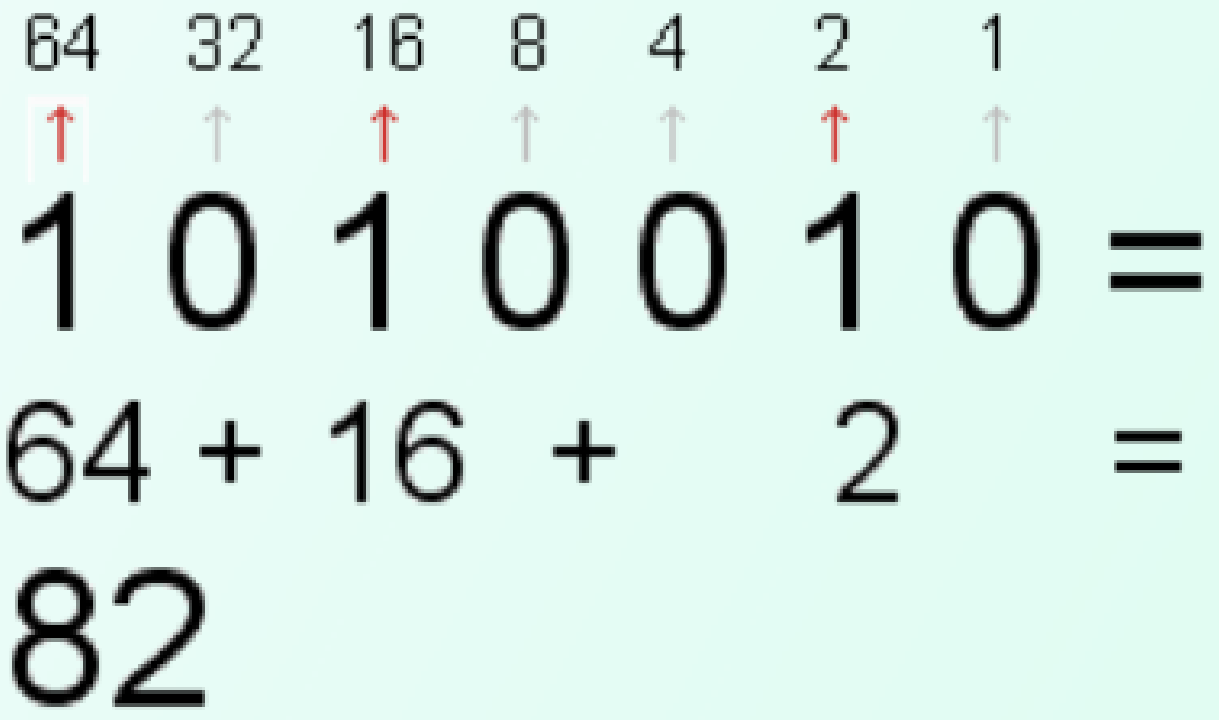
Convertir binario a decimal

Procedimiento simplificado :

-Asignamos a cada dígito su valor

-Seleccionamos los que valgan 1

-Sumamos



Ejercicios

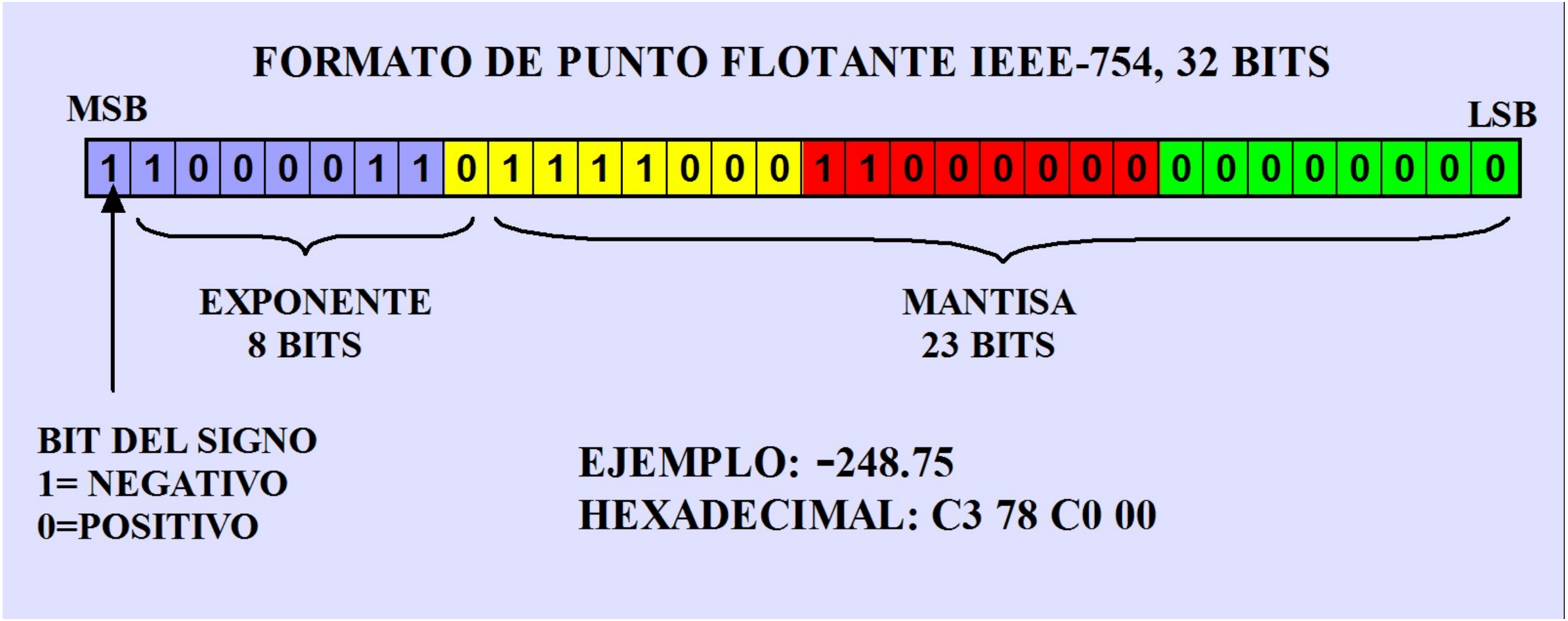
Representación de enteros con signo

- Cuando manejamos números enteros queremos también que estos tengan un signo
- Una opción para almacenar el signo es utilizar el bit más a la izquierda como indicador del signo: 0 para los positivos y 1 para los negativos
- La desventaja de este sistema es que el numero 0 tiene dos representaciones: 0 y –0
- Para evitar esto y facilitar ciertas operaciones aritméticas existen otros métodos como el complemento a 1 y el complemento a 2
- Dependiendo del número de bits que utilicemos para representar un entero tendremos los tipos:
  - **int** (32 bits): puede representar números desde -2^{31} a $2^{31}-1$.
  - **long** (64 bits): puede representar números desde -2^{63} a $2^{63}-1$.

Representación de números decimales

- Para la codificación de números decimales se usa muy habitualmente la representación en coma flotante
- Se pueden representar número reale extremadamente grandes y pequeños de una manera muy eficiente y compacta.
- Tiene como problema la perdida de precisión
- Se suelen usar dos tipos de datos en coma flotante: **float** (32 bits) y **double** (64 bits)

Representación de números decimales



Sistemas de codificación

- Pero ¿cómo se representan otro tipo de datos que no son numéricos?
- Para representar caracteres se utilizan tablas de codificación que asignan a cada carácter un número.
- De esta forma es posible codificar un texto de forma numérica
- Estos sistemas de codificación suelen fijar el número de bits necesarios para representar un carácter de forma que todos los caracteres tienen la misma longitud
- El número de bits de un sistema de codificación viene determinado por el número de elementos que queremos codificar:
  - 2 elementos = 2^1 -> 1 bit será suficiente
  - 4 elementos = 2^2 -> 2 bits serán suficientes
  - 8 elementos = 2^3 -> 3 bits serán suficientes
  - 16 elementos = 2^4 -> 4 bits serán suficientes

Sistemas de codificación

Binario	Dec	Hex	Representación
0010 0000	32	20	espacio ( )
0010 0001	33	21	!
0010 0010	34	22	"
0010 0011	35	23	#
0010 0100	36	24	$
0010 0101	37	25	%
0010 0110	38	26	&
0010 0111	39	27	'
0010 1000	40	28	(
0010 1001	41	29	)
0010 1010	42	2A	*
0010 1011	43	2B	+
0010 1100	44	2C	,
0010 1101	45	2D	-
0010 1110	46	2E	.
0010 1111	47	2F	/
0011 0000	48	30	0
0011 0001	49	31	1
0011 0010	50	32	2
0011 0011	51	33	3
0011 0100	52	34	4

Binario	Dec	Hex	Representación
0100 0000	64	40	@
0100 0001	65	41	A
0100 0010	66	42	B
0100 0011	67	43	C
0100 0100	68	44	D
0100 0101	69	45	E
0100 0110	70	46	F
0100 0111	71	47	G
0100 1000	72	48	H
0100 1001	73	49	I
0100 1010	74	4A	J
0100 1011	75	4B	K
0100 1100	76	4C	L
0100 1101	77	4D	M
0100 1110	78	4E	N
0100 1111	79	4F	O
0101 0000	80	50	P
0101 0001	81	51	Q
0101 0010	82	52	R
0101 0011	83	53	S
0101 0100	84	54	T

Binario	Dec	Hex	Representación
0110 0000	96	60	`
0110 0001	97	61	a
0110 0010	98	62	b
0110 0011	99	63	c
0110 0100	100	64	d
0110 0101	101	65	e
0110 0110	102	66	f
0110 0111	103	67	g
0110 1000	104	68	h
0110 1001	105	69	i
0110 1010	106	6A	j
0110 1011	107	6B	k
0110 1100	108	6C	l
0110 1101	109	6D	m
0110 1110	110	6E	n
0110 1111	111	6F	o
0111 0000	112	70	p
0111 0001	113	71	q
0111 0010	114	72	r
0111 0011	115	73	s
0111 0100	116	74	t

Ejercicios

Codificación de imágenes

- Existen diversas formas de codificar imágenes en lenguaje binario.
- Una de ellas consiste en guardar por cada punto en la imagen, el valor del color de ese punto.
- Cualquier color perceptible por el ojo humano puede obtenerse de la combinación con la combinación de los colores rojo, azul y verde (RGB)
- Un color se puede representar con la cantidad rojo, azul y verde y esta cantidad se puede medir numéricamente
- Cada punto se puede representar con su color almacenando 1 byte para cada uno de los colores básicos

Codificación de imágenes

FFFFFF	000000	333333	666666	999999	CCCCCC	CCCC99	9999CC	666699
660000	663300	996633	003300	003333	003399	000066	330066	660066
990000	993300	CC9900	006600	336666	0033FF	000099	660099	990066
CC0000	CC3300	FFCC00	009900	006666	0066FF	0000CC	663399	CC0099
FF0000	FF3300	FFFF00	00CC00	009999	0099FF	0000FF	9900CC	FF0099
CC3333	FF6600	FFFF33	00FF00	00CCCC	00CCFF	3366FF	9933FF	FF00FF
FF6666	FF6633	FFFF66	66FF66	66CCCC	00FFFF	3399FF	9966FF	FF66FF
FF9999	FF9966	FFFF99	99FF99	66FFCC	99FFFF	66CCFF	9999FF	FF99FF
FFCCCC	FFCC99	FFFFCC	CCFFCC	99FFCC	CCFFFF	99CCFF	CCCCFF	FFCCFF