

# Sistemas de numeración

- Sistema decimal
- Sistema binario
- Sistemas octal y hexadecimal
- Conversiones entre sistemas
- Operaciones: suma y resta
- Representación de binarios negativos

# INTRODUCCIÓN

## Sistema de numeración

- Un notable avance intelectual del hombre fue la creación de un proceso mental que se puede llamar operación de contar, o facultad de contar.
- Las dificultades de las formas rudimentarias aparecieron cuando las cantidades a contar crecieron . Ello obligó a crear sistemas de representación mas elaborados y prácticos.
- Fue preciso adoptar un conjunto de signos o símbolos que, con ciertas reglas, progresaran en el sentido de las magnitudes crecientes y que se los hiciera coordinables con los números naturales. Se crearon así los llamados: “Sistemas de Numeración”

# INTRODUCCIÓN

## Sistema de numeración

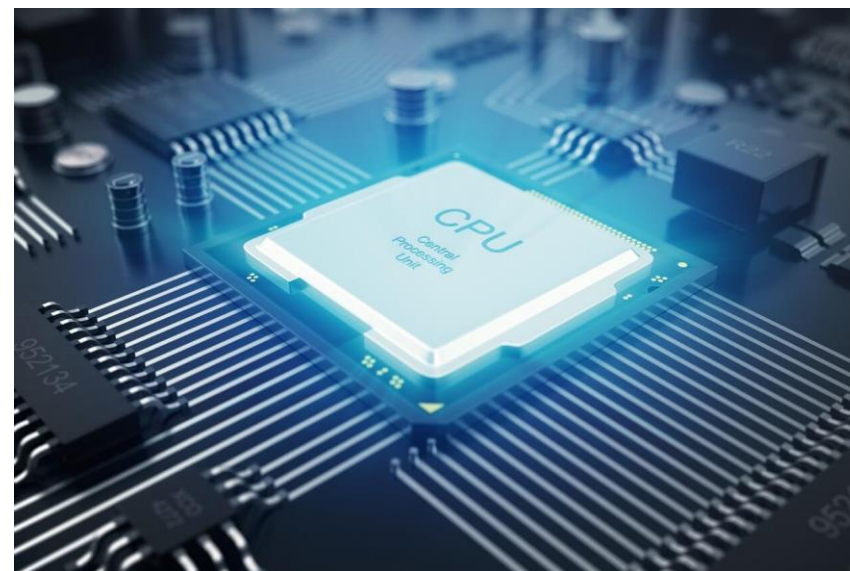
- Conjunto de símbolos y reglas que se utilizan para la representación de datos numéricos o cantidades.
- Se caracteriza por su **base**, que es el número de símbolos distintos que utiliza y además es el coeficiente que determina cuál es el valor de cada símbolo dependiendo de la posición que ocupe.
- Son netamente posicionales: el valor relativo que representa cada símbolo o cifra depende de su valor absoluto y de la posición que ocupa respecto a la coma decimal.

*La coma decimal (,) que separa la parte entera de la parte fraccionaria, en ambientes informáticos, está representada por el punto decimal (.).*

# INTRODUCCIÓN

## Sistema de numeración

- Los sistemas de numeración tienen aplicación directa en los sistemas informáticos a lo interno de las computadoras.
- Trataremos los principales aspectos de estos sistemas de conversiones y las operaciones (suma y resta) que se pueden realizar.



# INTRODUCCIÓN

## Sistemas de Numeración

Base	Sistema	Dígitos
2	Binario	0, 1
8	Octal	0, 1, 2, 3, 4, 5, 6, 7
10	Decimal	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
16	Hexadecimal	0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

El binario es el más relevante pues la computadora está construida a partir de circuitos electrónicos digitales que, procesa, almacena y transmite la información con el sistema binario.

# INTRODUCCIÓN

El sistema octal (Base 8) y el hexadecimal (Base 16), se usan para representar grandes números binarios.

Tienen como ventaja la posibilidad de convertir fácilmente al binario.

$$111110101000000000011111_{(2)} = \text{FA801F}_{(16)}$$

# SISTEMA DECIMAL



Derivó del sistema *indoarábigo*, posiblemente se adoptó este sistema por contar con 10 dedos en las manos

Utiliza un conjunto de símbolos, cuyo significado depende de su posición relativa al punto decimal, que en caso de ausencia se supone colocado implícitamente a la derecha

Basado en **diez** símbolos (por esto que se dice que utiliza la base 10):

**0 – 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9**

La combinación entre ellos permite representar las cantidades imaginadas

# SISTEMA DECIMAL

Numero	posición	Potencia	Nombre
1	0	$10^0$	Unidades
10	1	$10^1$	Decenas
100	2	$10^2$	Centenas
1000	3	$10^3$	Unidades de Millar
10000	4	$10^4$	Decenas de Millar
100000	5	$10^5$	Centena de Millar
1,000,000	6	$10^6$	Unidad de Millón
10,000,000	7	$10^7$	Decena de Millón
100,000,000	8	$10^8$	Centena de Millón
1000,000,000	9	$10^9$	Unidad de Millar de Millón
10,000,000,000	10	$10^{10}$	Decena de Millar de Millón
100,000,000,000	11	$10^{11}$	Centena de Millar de Millón
1,000,000,000,000	12	$10^{12}$	Unidad de Billón



# SISTEMA DECIMAL

## Representación como suma de potencias

El valor de cada dígito está asociado a una potencia de base 10.

Número Entero **528** se puede calcular como:

$$5 \cdot 10^2 + 2 \cdot 10^1 + 8 \cdot 10^0 = 5 \cdot 100 + 2 \cdot 10 + 8 \cdot 1 = 528$$

Número con decimales, **245,97** se calcula como:

$$2 \cdot 10^2 + 4 \cdot 10^1 + 5 \cdot 10^0 + 9 \cdot 10^{-1} + 7 \cdot 10^{-2} = 245,97$$

# SISTEMA BINARIO



Introducido por Leibniz en el Siglo XVII

Este sistema de base 2 posee sólo dos dígitos y es el sistema que internamente utilizan los circuitos digitales que configuran el hardware de las computadoras actuales.

Basado en **2** símbolos:

**0 - 1**

Los dígitos se denominan **bits**, contracción de ***binary digit***.

# SISTEMA BINARIO

En bit más significativo (MSB) es aquel que se ubica más a la izquierda, el que tiene mayor valor, mientras que el bit menos significativo (LSB) es aquel que está más a la derecha y que tiene el menor valor.

MSB                      LSB  
↓                              ↓  
1 0 1 0 1 0

La unidad más pequeña corresponde a un dígito binario (0 o 1), denominado bit. Al conjunto de 8 bits se le denomina byte.

En el sistema de codificación ASCII cada carácter está representado por un byte.

En este sistema @ se representa como  $01000000_{(2)} = 64_{(10)}$

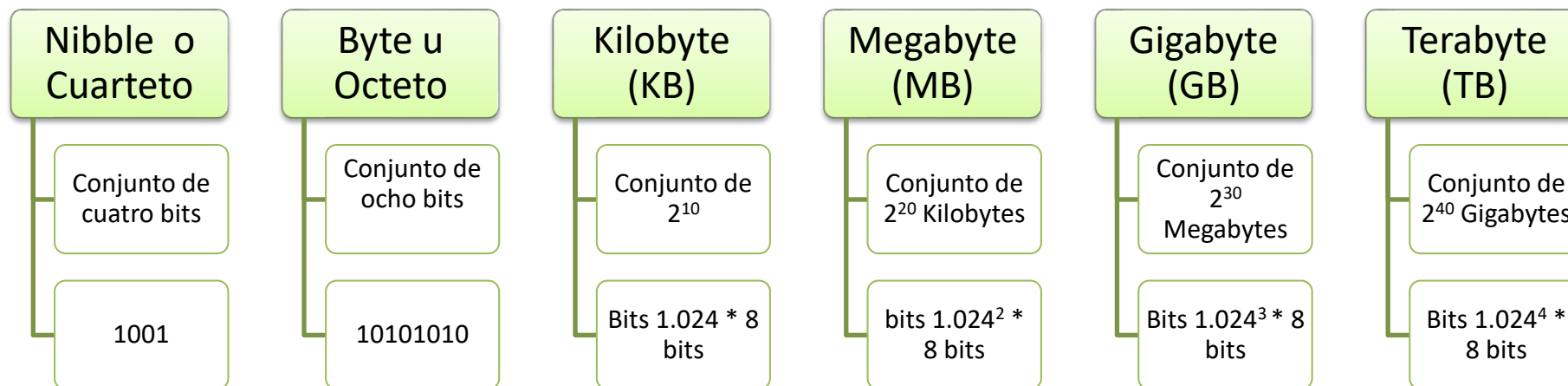
$$2^{10} = 1024$$

Si se almacena una cadena de caracteres (*string*) en memoria, cada uno de los caracteres ocupa un byte. Estas agrupaciones son finitas, pero pueden ser muy grandes, por lo que nos referimos a ellas agrupándolas en Kilobyte, Megabyte, etc.

1 kilobyte (KB)	1024 bytes
1 Megabyte (MB)	1024 kilobytes
1 Gigabyte (GB)	1024 Megabytes
1 Terabyte (TB)	1024 Gigabytes

# SISTEMA BINARIO

Para la medida de unidades de información representada en binario, se utilizan una serie de múltiplos de bit que poseen nombre propio:



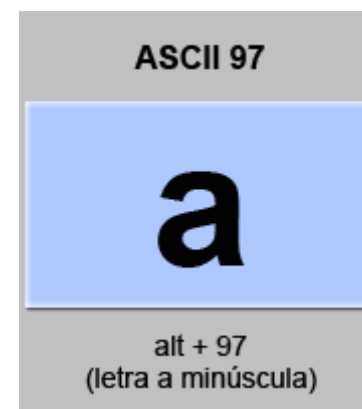
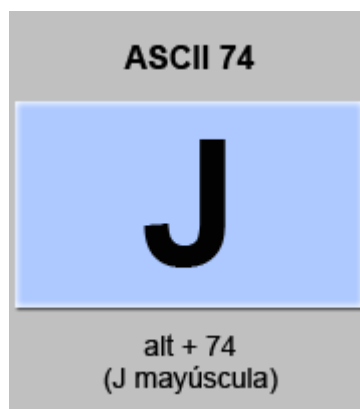
**1.024 en vez de 1.000**, por ser la potencia en base 2 más próximo a 1000, cuestión importante desde el punto de vista informático ( $2^{10} = 1.024$ )

# SISTEMA BINARIO

## Aplicación: CÓDIGO ASCII

El código ASCII (*American Standard Code for Information Interchange*), fue creado en 1963 por el Comité Estadounidense de Estándares o "ASA", este organismo cambio su nombre en 1969 por "Instituto Estadounidense de Estándares Nacionales" o "ANSI" como se lo conoce desde entonces.

En este sistema, a cada carácter se le asigna un número decimal comprendido entre 0 y 255, que, una vez convertido al sistema de numeración binario, nos da el código del carácter.



# SISTEMA BINARIO

## Aplicación: CÓDIGO ASCII

Caracteres ASCII de control			Caracteres ASCII imprimibles			ASCII extendido (Página de código 437)										
00	NULL	(carácter nulo)	32	espacio	64	@	96	`	128	Ç	160	á	192	Ł	224	Ó
01	SOH	(inicio encabezado)	33	!	65	A	97	a	129	ü	161	í	193	ł	225	ô
02	STX	(inicio texto)	34	"	66	B	98	b	130	é	162	ó	194	Ł	226	Ô
03	ETX	(fin de texto)	35	#	67	C	99	c	131	â	163	ú	195	ł	227	Ò
04	EOT	(fin transmisión)	36	\$	68	D	100	d	132	ä	164	ñ	196	—	228	ö
05	ENQ	(consulta)	37	%	69	E	101	e	133	à	165	Ñ	197	+	229	Õ
06	ACK	(reconocimiento)	38	&	70	F	102	f	134	å	166	ª	198	ä	230	µ
07	BEL	(timbre)	39	'	71	G	103	g	135	ç	167	º	199	Ä	231	þ
08	BS	(retroceso)	40	(	72	H	104	h	136	ê	168	¿	200	Ł	232	ð
09	HT	(tab horizontal)	41	)	73	I	105	i	137	ë	169	®	201	ł	233	Ú
10	LF	(nueva línea)	42	*	74	J	106	j	138	è	170	™	202	Ł	234	Û
11	VT	(tab vertical)	43	+	75	K	107	k	139	ï	171	½	203	ł	235	Ü
12	FF	(nueva página)	44	,	76	L	108	l	140	î	172	¼	204	ł	236	ý
13	CR	(retorno de carro)	45	-	77	M	109	m	141	ï	173	⅓	205	=	237	Ÿ
14	SO	(desplaza afuera)	46	.	78	N	110	n	142	Ä	174	«	206	†	238	—
15	SI	(desplaza adentro)	47	/	79	O	111	o	143	Å	175	»	207	□	239	˙
16	DLE	(esc.vínculo datos)	48	0	80	P	112	p	144	É	176	⋮	208	ð	240	≡
17	DC1	(control disp. 1)	49	1	81	Q	113	q	145	æ	177	⋮	209	Ð	241	±
18	DC2	(control disp. 2)	50	2	82	R	114	r	146	Æ	178	⋮	210	È	242	≡
19	DC3	(control disp. 3)	51	3	83	S	115	s	147	ô	179	⋮	211	Ê	243	¼
20	DC4	(control disp. 4)	52	4	84	T	116	t	148	ö	180	⋮	212	Ë	244	¶
21	NAK	(conf. negativa)	53	5	85	U	117	u	149	ò	181	⋮	213	Ì	245	§
22	SYN	(inactividad sínc)	54	6	86	V	118	v	150	û	182	⋮	214	Í	246	÷
23	ETB	(fin bloque trans)	55	7	87	W	119	w	151	ù	183	⋮	215	Î	247	°
24	CAN	(cancelar)	56	8	88	X	120	x	152	ÿ	184	©	216	Ï	248	ˆ
25	EM	(fin del medio)	57	9	89	Y	121	y	153	Ö	185	⋮	217	Ĵ	249	˜
26	SUB	(sustitución)	58	:	90	Z	122	z	154	Ü	186	⋮	218	Œ	250	˘
27	ESC	(escape)	59	;	91	[	123	{	155	ø	187	⋮	219	Š	251	˙
28	FS	(sep. archivos)	60	<	92	\	124		156	£	188	⋮	220	Š	252	˚
29	GS	(sep. grupos)	61	=	93	]	125	}	157	Ø	189	¢	221	Š	253	²
30	RS	(sep. registros)	62	>	94	^	126	~	158	×	190	¥	222	Š	254	■
31	US	(sep. unidades)	63	?	95	_			159	f	191	Œ	223	Š	255	nbsp
127	DEL	(suprimir)														

# SISTEMA BINARIO

## Aplicación: Direcciones de red

Cada dispositivo conectado a una red de datos (PC, Servidor, Teléfono móvil, impresora de red, etc.) cuenta con una dirección IP única dentro de esa red, que se conforma por 4 bytes.

192	168	1	14
11000000	10101000	00000001	00001110
110000001010100000000000100001110			
110000001010100000000000100001110			

A nivel computo, estas direcciones se utilizan en binario, pero a nivel 'humano' se escriben y comunican en decimal, debido a la dificultad y alta probabilidad de cometer un error.

# SISTEMA BINARIO

## Conversión decimal a binario

### Primer método: Suma de Potencias de 2

El número decimal se expresa como suma de potencias de 2 y luego los unos y los ceros se escriben en las posiciones adecuadas de bits.

### Ejemplo:

$$45_{(10)} = 32 + 8 + 4 + 1 = 2^5 + 0 + 2^3 + 2^2 + 0 + 2^0 = 1\ 0\ 1\ 1\ 0\ 1_{(2)}$$

*Obsérvese que se coloca un 0 en las posiciones  $2^1$  y  $2^4$ , ya que todas las posiciones deben tomarse en cuenta*

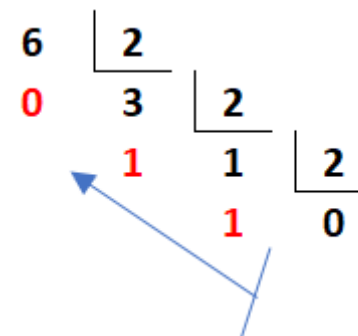


# SISTEMA BINARIO

## Conversión decimal a binario

### Segundo método: Divisiones Sucesivas entre 2

1. Se divide el número decimal y los sucesivos cocientes entre dos (2), hasta que el cociente en una de las divisiones tome el valor cero (0).
2. La unión de todos los restos obtenidos, escritos en orden inverso, nos proporciona el número inicial expresado en el sistema binario

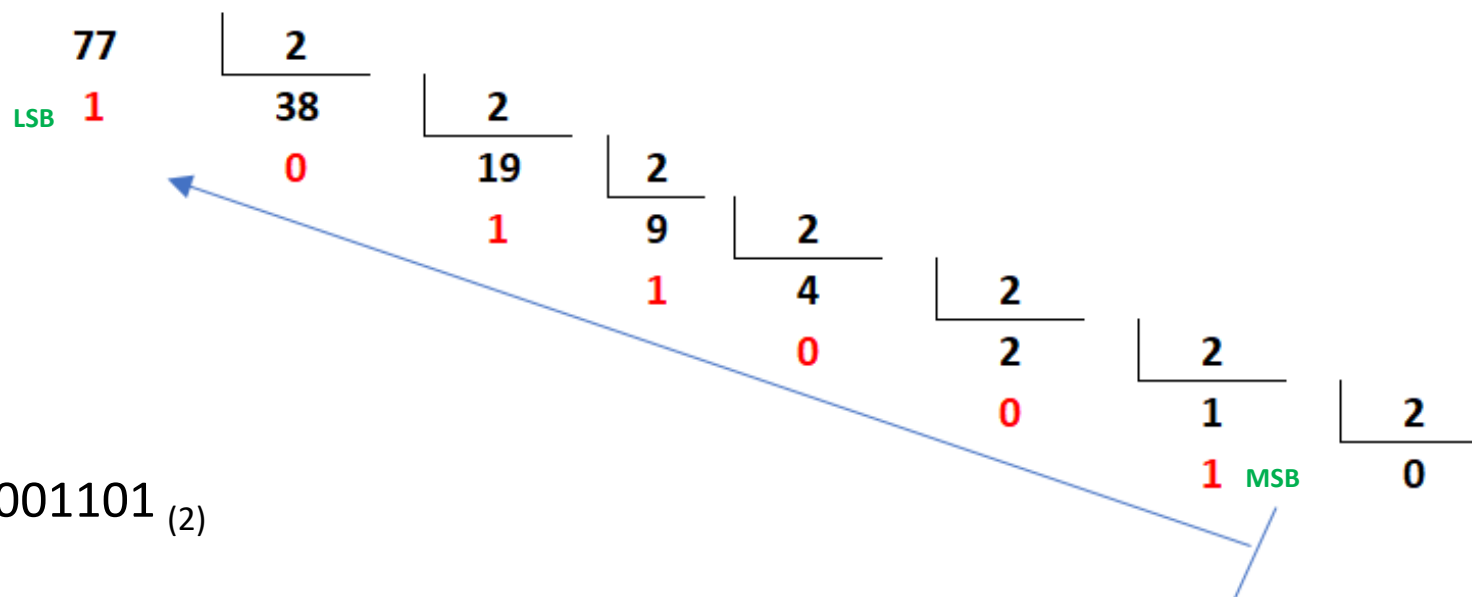


$$6_{(10)} = 110_{(2)}$$

# SISTEMA BINARIO

## Conversión decimal a binario

Conversión del número 77 decimal a binario.

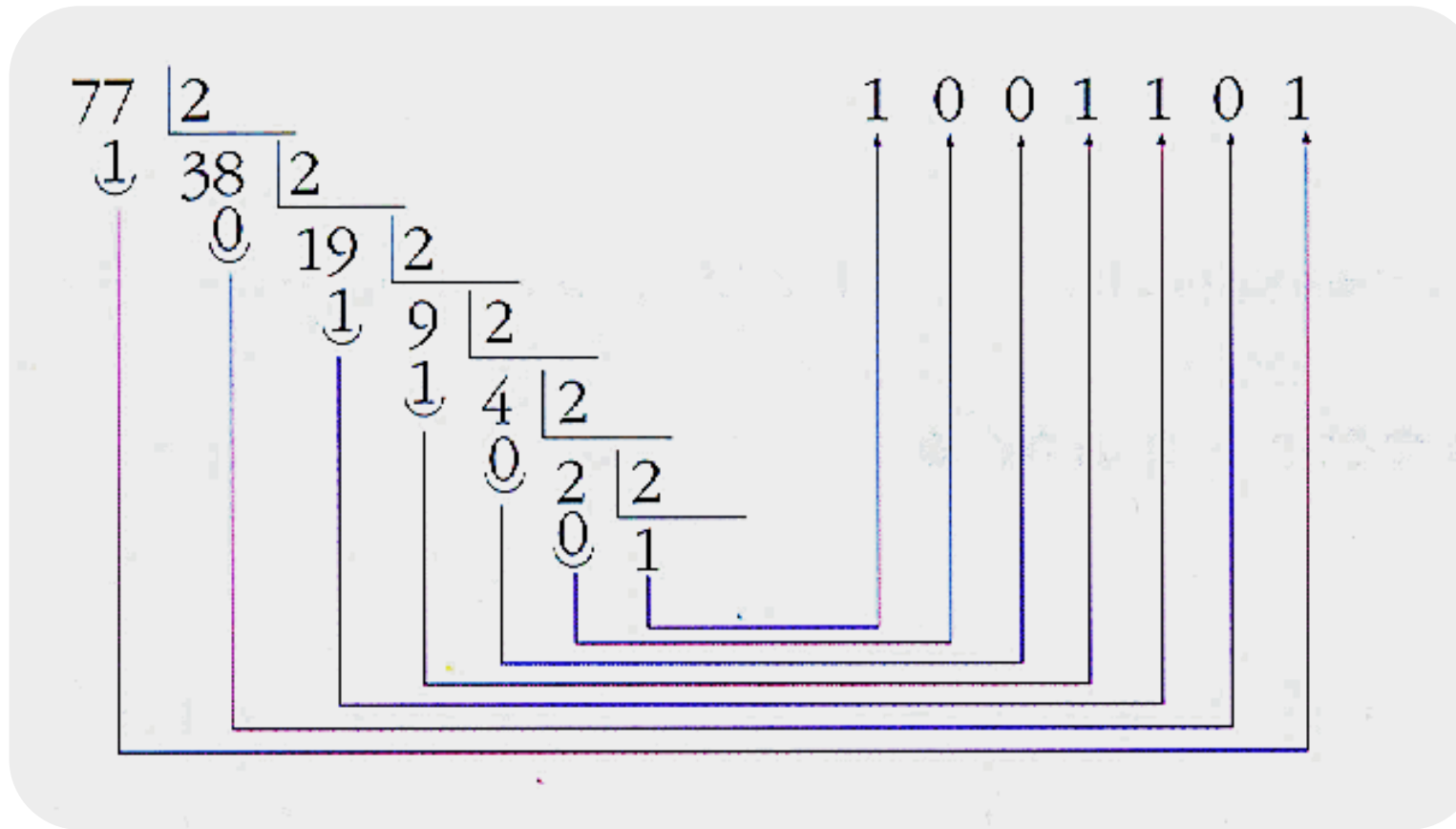


$$77_{(10)} = 1001101_{(2)}$$

En este ejemplo el resultado tiene 7 bits.

# SISTEMA BINARIO

## Conversión decimal a binario



## Conversión decimal a binario

Conversión del número 77,625 decimal a binario.

La conversión se resuelve en 2 pasos:

1. La parte entera se resuelve con divisiones sucesivas entre 2.
2. La parte fraccionaria se resuelve con multiplicaciones sucesivas por 2.
3. Del resultado de cada multiplicación nos quedamos con la parte entera (1 o 0) y la parte fraccionaria vuelve a multiplicarse por 2.
4. El resultado de la parte fraccionaria se escribe desde el primer valor obtenido.

0,625	x	2	=	1,250	1
0,250	x	2	=	0,500	0
0,500	x	2	=	1,000	1
0,000	x	2	=	0,000	0



$$77,625_{(10)} = 1001101,101_{(2)}$$

La parte fraccionaria decimal no siempre puede representarse con un numero finito de bits.

$$77,63_{(10)} \approx 1001101,10100001010001111_{(2)}$$

# SISTEMA BINARIO

## Conversión binario a decimal

### Método: Suma de potencias de base 2

1. Se numeran los dígitos binarios desde el cero en la unidad, creciente a la izquierda y decreciente a la derecha.
2. Se multiplica el dígito (0 o 1) por 2 elevado al número de posición y se suma el resultado obteniendo así un número decimal.

$$\begin{matrix} 3 & 2 & 1 & 0 & -1 \\ 1 & 1 & 0 & 1 & , 1 \end{matrix}_{(2)}$$

$$1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1}$$

$$1 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1 + 1 \times (1/2) = 13,5_{(10)}$$

# SISTEMA OCTAL

Número binario de tres dígitos	Número octal
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Utilizado para representar binarios de forma compacta.

Con aplicación en informática, electrónica digital y entornos industriales.  
Viene siendo reemplazado por el sistema hexadecimal.

Basado en 8 símbolos:

**0 – 1 – 2 – 3 – 4 – 5 – 6 – 7**

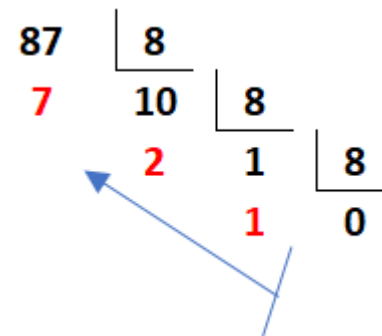
Cada dígito octal representa un grupo de 3 bits.

# SISTEMA OCTAL

## Conversión decimal a octal

Método: **Divisiones Sucesivas entre 8**

1. Se divide el número decimal y los sucesivos cocientes entre ocho (8), hasta que el cociente en una de las divisiones tome el valor cero (0).
2. La unión de todos los restos obtenidos, escritos en orden inverso, nos proporciona el número inicial expresado en el sistema octal.

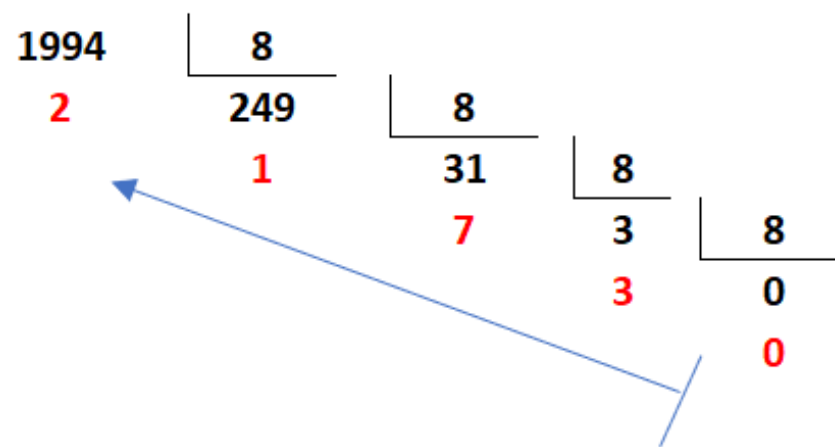


$$87_{(10)} = 127_{(8)}$$

# SISTEMA OCTAL

## Conversión decimal a octal

Método: Divisiones Sucesivas entre 8



$$1994_{(10)} = 3712_{(8)}$$



## Conversión octal a decimal

### Método: Suma de potencias de base 8

1. Se numeran los dígitos octales desde el cero en la unidad, creciente a la izquierda y decreciente a la derecha.
2. Se multiplica el dígito (0 a 7) por 8 elevado al número de posición y se suma el resultado obteniendo así un número decimal.

$$3712_{(8)} = 3 \times 8^3 + 7 \times 8^2 + 1 \times 8^1 + 2 \times 8^0$$

$$3712_{(8)} = 3 \times 512 + 7 \times 64 + 1 \times 8 + 2 \times 1$$

$$3712_{(8)} = 1536 + 448 + 8 + 2$$

$$3712_{(8)} = 1994_{(10)}$$

# SISTEMA HEXADECIMAL



Utilizado para representar binarios de forma compacta.

Dado que el número de símbolos supera el de números, se complementa con letras desde la **A** a la **F**.

Basado en 16 símbolos:

**0 – 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9 – A – B – C – D – E – F**

Cada dígito hexa representa un grupo de 4 bits.

# SISTEMA HEXADECIMAL

## Tabla de equivalencia

- Base 16: se representan con dieciséis símbolos: Utiliza los dígitos del 0 al 9, más las letras A, B, C, D, E y F

**0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F**

- Los dígitos hexa de **A** a **F** son equivalentes a los valores decimales de 10 a 15.
- Cada dígito hexadecimal representa un grupo de cuatro dígitos binarios.

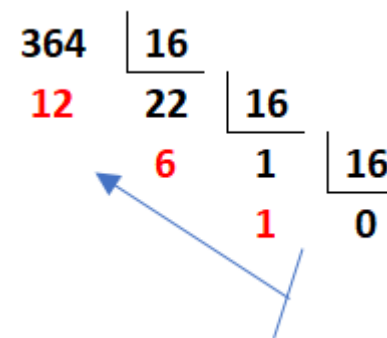
Decimal	Binario				Octal	Hexadecimal
0	0	0	0	0	0	0
1	0	0	0	1	1	1
2	0	0	1	0	2	2
3	0	0	1	1	3	3
4	0	1	0	0	4	4
5	0	1	0	1	5	5
6	0	1	1	0	6	6
7	0	1	1	1	7	7
8	1	0	0	0		8
9	1	0	0	1		9
10	1	0	1	0		A
11	1	1	0	1		B
12	1	1	0	0		C
13	1	1	0	1		D
14	1	1	1	0		E
15	1	1	1	1		F

# SISTEMA HEXADECIMAL

## Conversión decimal a hexa

Método: **Divisiones Sucesivas entre 16**

1. Se divide el número decimal y los sucesivos cocientes entre dieciséis (16), hasta que el cociente en una de las divisiones tome el valor cero (0).
2. La unión de todos los restos obtenidos, escritos en orden inverso, nos proporciona el número inicial expresado en el sistema hexadecimal.

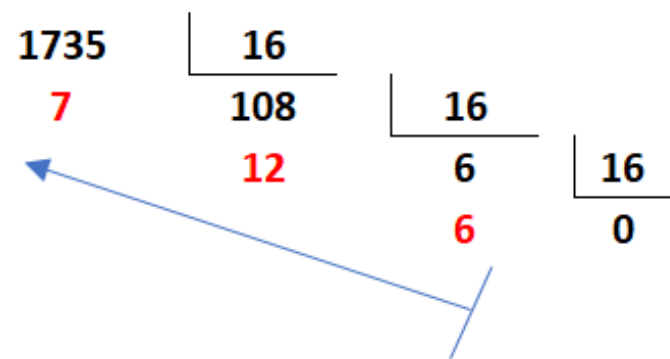


$$364_{(10)} = 16C_{(8)}$$

# SISTEMA HEXADECIMAL

## Conversión decimal a hexa

Método: Divisiones Sucesivas entre 16



$$1735_{(10)} = 6C7_{(8)}$$

Decimal	Hexadecimal
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	A
11	B
12	C
13	D
14	E
15	F

Los restos entre 10 y 15 se sustituyen por el valor en *hexa A a F*.

# SISTEMA HEXADECIMAL

## Conversión hexa a decimal

### Método: Suma de potencias de base 16

1. Se numeran los dígitos hexadecimales desde el cero en la unidad, creciente a la izquierda y decreciente a la derecha.
2. Se multiplica el dígito (0 a 15) por 16 elevado al número de posición y se suma el resultado obteniendo así un número decimal.

$$\begin{array}{ccccccc} & 3 & 2 & 1 & 0 & -1 & \\ & F & 1 & B & 1, & C & \end{array}_{(16)}$$

$$15 \times 16^3 + 1 \times 16^2 + 11 \times 16^1 + 1 \times 16^0 + 12 \times 16^{-1}$$

$$15 \times 4096 + 1 \times 256 + 0 \times 16 + 1 \times 1 + 1 \times (1/16) = 61873,75_{(10)}$$

# SISTEMA HEXADECIMAL

## Conversión binario a hexa / hexa a binario

### Método: Conversión directa

1. Se forman grupos de 4 bits, comenzando desde la posición de la unidad (posición 0) a la izquierda y si existe parte fraccionaria, a la derecha. Se completa con ceros en los extremos en caso de ser necesario.
2. Cada grupo de 4 bits se representa con el correspondiente símbolo hexadecimal.

0011011111011111010,10110110

3 7 D F A , B 6

$$110111110111111010,1011011_{(2)} = 37DFA,B6_{(16)}$$

Binario				Hexadecimal
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	A
1	0	1	1	B
1	1	0	0	C
1	1	0	1	D
1	1	1	0	E
1	1	1	1	F

# SISTEMAS DE NUMERACIÓN

## Generalización de conversiones

De la misma manera en que fueron creados los sistemas posicionales decimal, binario, octal y hexadecimal, es posible crear nuestro propio sistema usando los dígitos necesarios del 0 al 9, y las letras del alfabeto.

### Las siguientes cantidades están expresadas en sistemas posicionales

- **20541.32<sub>(7)</sub>**

Aquí la base es 7 y los caracteres válidos van del 0 al 6.

- **7G5A90.HB<sub>(18)</sub>**

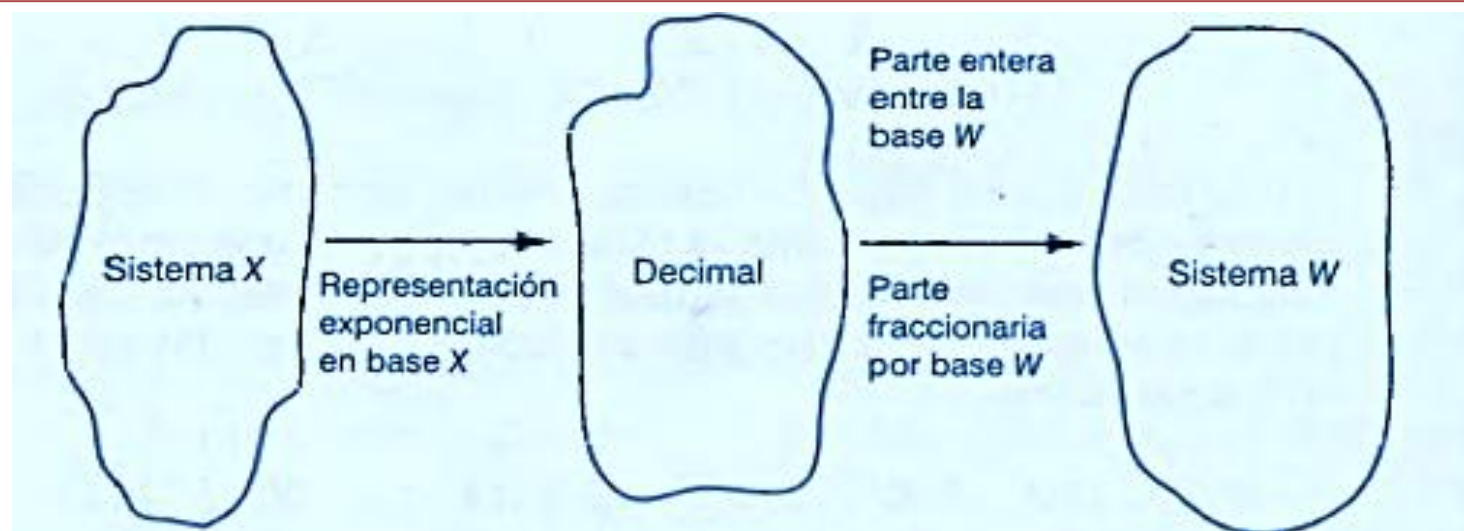
En este caso, además de poder usar los dígitos del 0 al 9 es posible utilizar las letras A = 10, B = 11, C = 12, D = 13, E = 14, F = 15, G = 16, H = 17, ya que en base 18 los caracteres válidos van del 0 al 17.



# SISTEMAS DE NUMERACIÓN

## Generalización de conversiones

Las cantidades expresadas en cualquier sistema numérico y ser convertidas a otro sistema existente o no



para pasar de un sistema X cualquiera a decimal: representar en notación exponencial

para pasar de decimal a un sistema W: dividir la parte entera entre la base a la que se desea convertir y multiplicar la parte fraccionaria por la base en cuestión

# OPERACIONES BÁSICAS

## Suma

*PROCEDIMIENTO* : sumar los números correspondientes a cada columna y colocar el resultado debajo de la línea. El procedimiento es independiente del sistema numérico.

$$456,78 + 7820,649 = 8277,429$$



	4	5	6	.	7	8	(10)
7	8	2	0	.	6	4	9 <sub>(10)</sub>
8	2	7	7	.	4	2	9 <sub>(10)</sub>

### *Tener en cuenta que:*

Si al sumar dos dígitos, el resultado es mayor al dígito mayor del sistema numérico **entonces**

- Al resultado de la suma se lo divide entre la base del sistema, y se obtiene un resto y un cociente.
- El resto se coloca debajo de la línea de suma y el cociente se suma a la columna de la izquierda

# OPERACIONES BÁSICAS

## Suma

$$\begin{array}{r}
 \phantom{000000000}4\phantom{00000}5\phantom{00000}6\phantom{00000}.\phantom{00000}7\phantom{00000}8_{(10)} \\
 + \\
 1\phantom{00000}7\phantom{00000}8\phantom{00000}2\phantom{00000}0\phantom{00000}.\phantom{00000}6\phantom{00000}4\phantom{00000}9_{(10)} \\
 \hline
 1\phantom{00000}8\phantom{00000}2\phantom{00000}7\phantom{00000}7\phantom{00000}.\phantom{00000}4\phantom{00000}2\phantom{00000}9_{(10)}
 \end{array}$$

0+9=9	Dígito válido de base 10, entonces queda como está			
8+4=12	El 12 no es válido en decimal. Se divide entre la base 10, se coloca el resto debajo de la línea y se suma el cociente de la división a los números que se encuentran a la izquierda			
1+7+6=14	El 14 no es válido en decimal se procede como con el 12			
1+6+0=7	Dígito válido en decimal			
5+2=7	Dígito válido en decimal			
4+8=12	El 12 no es válido en decimal se procede como antes			
1+0+7=8	Dígito válido en decimal			

# Suma

+

**A 6 F C 9 . 7 B 2<sub>(16)</sub>**

**4 E 7 D 0 . 7 3 E<sub>(16)</sub>**

**F 5 7 9 9 . E F 0<sub>(16)</sub>**

$2+14=16$	Al dividir 16 entre la base se obtiene cociente 1 y resto 0			
$1+11+3=15$	Digito válido: 15=F			
$7+7=14$	Digito válido: 14=E			
$9+0=9$	Digito válido			
$12+13=25$	Al dividir 25 entre la base se obtiene cociente 1 y resto 9			
$1+15+7=23$	Al dividir 25 entre la base se obtiene cociente 1 y resto 7			
$1+6+14=21$	Al dividir 21 entre la base se obtiene cociente 1 y resto 5			
$1+10+4=15$	Digito válido: 15=F			

# OPERACIONES BÁSICAS

## Resta

Para efectuar la resta verificar si el sustraendo es mayor que el minuendo. En caso afirmativo se debe sumar la base al minuendo antes de realizar la resta de los dígitos de la columna.

Se debe sumar:

- *1 a la columna de la izquierda si es decimal*
- *8 a la columna de la izquierda si es octal*
- *16 a la columna de la izquierda si es hexadecimal*
- *2 a la columna de la izquierda si es binario*

**Tener en cuenta que:** antes de hacer la comparación entre minuendo y sustraendo, cuando se le suma la base al minuendo se incrementa **1** al sustraendo de la columna izquierda próxima, independientemente del sistema numérico

**1** significa que se le sumó una vez la base en la columna anterior

## Resta

-	8	1	2	7	.	5	8	0 <sub>(10)</sub>
	5	8	3	1	.	9	6	4 <sub>(10)</sub>
					.	6	1	6 <sub>(10)</sub>

$(0+10) - 4=6$	Sustraendo > Minuendo. Sumar la base (10) al minuendo y luego se realiza la sustracción.
$8 - (6+1) =1$	Cuando en la columna anterior se sumó 10 al minuendo, en la columna siguiente de la izquierda se suma 1 al sustraendo y eso es igual al resultado. Si después de sumar 1 al sustraendo este es mayor que el minuendo, entonces se deberá sumar la base al minuendo antes de realizar la resta
$(5+10)-9=6$	Minuendo > Sustraendo sumamos la base al minuendo y realizamos la resta

# OPERACIONES BÁSICAS

## Resta

-	8	1	2	7	.	5	8	0 <sub>(10)</sub>
	5	8	3	1	.	9	6	4 <sub>(10)</sub>
	2	2	9	5	.	6	1	6 <sub>(10)</sub>

$7-(1+1)=5$	Como en la columna anterior se sumo la base al minuendo en esta columna se suma 1 al sustraendo
$(2+10)-3=9$	Sustraendo > Minuendo se suma la base al minuendo
$(1+10)-(8+1)=2$	Teniendo en cuenta lo explicado se procede a sumar lo que corresponde. Importa el orden del proceso
$8-(5+1)=2$	Sumar 1 al sustraendo ya que en la columna anterior de le sumo la base al minuendo y después restar

## Resta

-	1	A	C	3	0	.	7	1	(16)
		3	B	2	1	.	6	2	(16)
					F	.	0	F	(16)

$(1 + 16) - 2 = 15$	Sustraendo > Minuendo. Sumar la base (16) al minuendo y luego se realiza la sustracción.
$7 - (6 + 1) = 0$	Cuando en la columna anterior se sumó 16 al minuendo, en la columna siguiente de la izquierda se suma 1 al sustraendo y eso es igual al resultado. Si después de sumar 1 al sustraendo este es mayor que el minuendo, entonces se deberá sumar la base al minuendo antes de realizar la resta
$(0 + 16) - 1 = 15$	Sustraendo > Minuendo sumamos la base al minuendo y realizamos la resta



# OPERACIONES BÁSICAS

## Resta

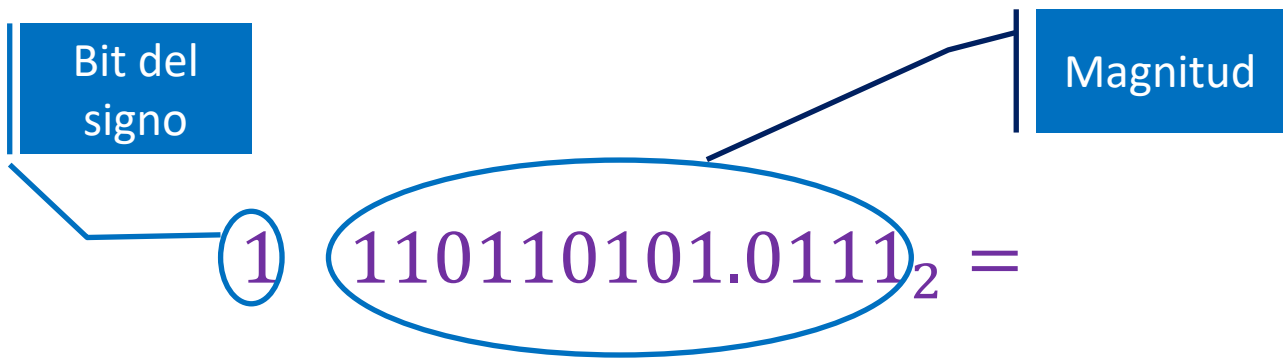
$$\begin{array}{r}
 \phantom{-} \quad 1 \quad A \quad C \quad 3 \quad 0 \quad . \quad 7 \quad 1 \quad (16) \\
 - \quad \quad 3 \quad B \quad 2 \quad 1 \quad . \quad 6 \quad 2 \quad (16) \\
 \hline
 1 \quad 7 \quad 1 \quad 0 \quad F \quad . \quad 0 \quad F \quad (16)
 \end{array}$$

$3 - (2 + 1) = 0$	Como en la columna anterior se sumó la base al minuendo en esta columna se suma 1 al sustraendo
$12 - 11 = 1$	Sustraendo < Minuendo se realiza la resta
$10 - 3 = 7$	Sustraendo < Minuendo se realiza la resta
$1 - 0 = 1$	Sustraendo < Minuendo se realiza la resta

# REPRESENTACIÓN DE BINARIOS NEGATIVOS

## Representación en magnitud verdadera

Se define el bit de signo: 0 positivo, 1 negativo. Posición MSB.

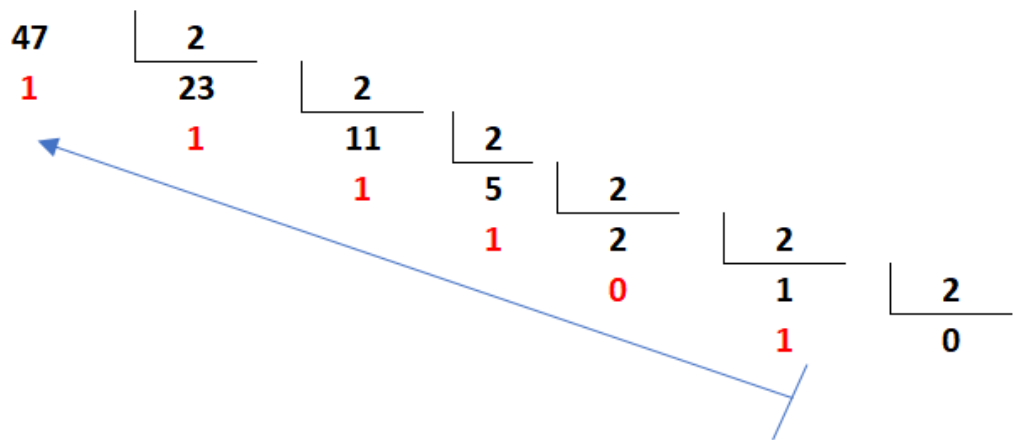

$$\begin{aligned} & \text{Bit del signo} \quad \text{Magnitud} \\ & \textcircled{1} \quad \textcircled{110110101.0111}_2 = \\ & = -(1 \times 2^8 + 1 \times 2^7 + 1 \times 2^5 + 1 \times 2^4 + 1 \times 2^2 + 1 \times 2^0 + \\ & 1 \times 2^{-2} + 1 \times 2^{-3} + 1 \times 2^{-4}) = -437.44_{(10)} \end{aligned}$$

Este tipo de representación permite visualizar fácilmente la equivalencia de ese conjunto de bits en el sistema decimal usando para ello la representación potencial

# REPRESENTACIÓN DE BINARIOS NEGATIVOS

## Representación en magnitud verdadera

Conversión de  $47_{(10)}$  a  $-47_{(10)}$  representado en binario de 10 bits.



$$47_{(10)} = 0000101111_{(2)}$$

$$-47_{(10)} = 1000101111_{(2)}$$

Bit del  
signo

# REPRESENTACIÓN DE BINARIOS NEGATIVOS

## Representación en complemento a 1

1. Se define el bit de signo: 0 positivo, 1 negativo. Posición MSB.
2. Se establece la cantidad de bits, que incluye el bit de signo.
3. Los valores positivos se mantienen en magnitud verdadera.
4. Para obtener el valor negativo se realiza el complemento a 1, invirtiendo todos los bits.
5. La representación en complemento a 1 tiene 2 valores para el cero.

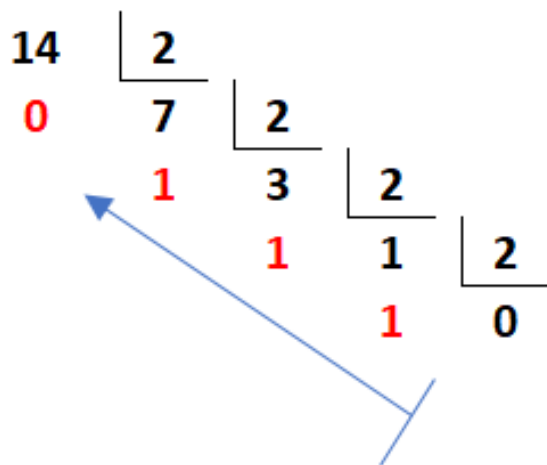
Complemento a uno	Decimal
0111	7
0110	6
0101	5
0100	4
0011	3
0010	2
0001	1
0000	0
1111	-0
1110	-1
1101	-2
1100	-3
1011	-4
1010	-5
1001	-6
1000	-7

Complemento a uno con  
enteros de 4 bits

# REPRESENTACIÓN DE BINARIOS NEGATIVOS

## Representación en complemento a 1

Conversión de  $14_{(10)}$  a  $-14_{(10)}$  representado en binario de 8 bits.



$$14_{(10)} = 00001110_{(2)}$$

$$-14_{(10)} = 11110001_{(2)}$$

# REPRESENTACIÓN DE BINARIOS NEGATIVOS

## Representación en complemento a 1

A que valor decimal corresponde el siguiente binario expresado en Complemento a 1 de 8 bits,  $10110001_{(2)}$ ?

1. Valor en Ca1, bit de signo en 1 (negativo):  $10110001$

2. Se aplica Ca1 para pasarlo a magnitud verdadera:  $11001110$

3. Se convierte a decimal como suma de potencias de base 2:

$$2^6 + 2^3 + 2^2 + 2^1 = 64 + 8 + 4 + 2 = 78_{(10)}$$

4. Resultado :  $10110001_{(2)} = -78_{(10)}$

# REPRESENTACIÓN DE BINARIOS NEGATIVOS

## Representación en complemento a 2

1. Se define el bit de signo: 0 positivo, 1 negativo. Posición MSB.
2. Se establece la cantidad de bits, que incluye el bit de signo.
3. Los valores positivos se mantienen en magnitud verdadera.
4. Para obtener el valor negativo se realiza el complemento a 1, invirtiendo todos los bits, y luego se suma 1 al valor obtenido.
5. La representación en complemento a 2 tiene 1 valor para el cero.

Complemento a dos	Decimal
0111	7
0110	6
0101	5
0100	4
0011	3
0010	2
0001	1
0000	0
1111	-1
1110	-2
1101	-3
1100	-4
1011	-5
1010	-6
1001	-7

# REPRESENTACIÓN DE BINARIOS NEGATIVOS

## Representación en complemento a 2

Conversión de  $14_{(10)}$  a  $-14_{(10)}$  representado en binario de 8 bits.

1. Se convierte el valor a binario, completando la cantidad de bits y signo.

$$-14_{(10)} = 10001110_{(2)} \quad \text{representado en } \underline{\text{magnitud verdadera}}$$

2. Se aplica complemento a 1:

$$\begin{array}{rcl} & 10001110 & \\ \text{Ca1} & 11110001 & \text{representado en } \underline{\text{complemento a 1}} \end{array}$$

3. Al resultado se suma 1:

$$\begin{array}{rcl} & 11110001 & \\ & + \quad \underline{1} & \\ \text{Ca2} & 11110010 & \text{representado en } \underline{\text{complemento a 2}} \end{array}$$

4.  $-14_{(10)} = 11110010_{(2)}$  representado en Ca2 de 8 bits.



# REPRESENTACIÓN DE BINARIOS NEGATIVOS

## Operaciones

Multiplicación: Sucesión de sumas

División: Sucesión de restas

La computadora no realiza restas ni multiplicaciones, ni divisiones, realiza únicamente sumas.

Cuando las cantidades a sumar son positivas se suman tal cual


Cuando alguna de ellas es negativa la cantidad negativa se complementa a 2 y luego se le suma la otra cantidad

# REPRESENTACIÓN DE BINARIOS NEGATIVOS


## Ejemplo de suma

Suponga que se definen las variables  $A$ ,  $B$  y  $C$  de tipo entero, ocupando 1 byte de memoria cada una. Si  $A = 225$  y  $B = 76$ , y en una línea de programa se tiene que  $C = A + B$ , entonces la computadora realiza la siguiente operación:

+ 225 <sub>(10)</sub>	=	0 1 1 1 0 0 0 0 1 <sub>(2)</sub>
+ 76 <sub>(10)</sub>	=	0 0 1 0 0 1 1 0 0 <sub>(2)</sub>
+ 301 <sub>(10)</sub>	=	1 0 0 1 0 1 1 0 1 <sub>(2)</sub>



Signo



Magnitud

El resultado obtenido es:  $100101101_{(2)} = -45_{(10)}$   
Siendo esto diferente al  $+301_{(10)}$  esperado.

Aquí se presentó un DESBORDAMIENTO al querer guardar en la variable  $C$  definida de 8 bits, una cantidad que no puede ser representada en 8 bits.

# REPRESENTACIÓN DE BINARIOS NEGATIVOS

## Ejemplo de suma

El error de desbordamiento es común en el proceso de la programación ya que se definen variables de cierto tipo y con cierta capacidad y en ocasiones se desea guardar en ellas cantidades que sobrepasan la capacidad definida.

Para resolver este problema se deben definir las variables con una mayor capacidad. En el ejemplo anterior bastaría con asignar 16 bits a las variables.

+	2	2	5 <sub>(10)</sub>	=	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	1 <sub>(2)</sub>	
+		7	6 <sub>(10)</sub>	=	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0 <sub>(2)</sub>	
<hr/>					<hr/>																	
+	3	0	1 <sub>(10)</sub>	=	<u>0</u>	0	0	0	0	0	0	0	0	1	0	0	1	0	1	1	0	1 <sub>(2)</sub>
					<hr/>																	

# REPRESENTACIÓN DE BINARIOS NEGATIVOS

## Error de desbordamiento

### Importante:

El desbordamiento solo ocurre cuando las dos cantidades que se están sumando son del mismo signo ya que son los únicos casos en que el resultado puede requerir mayor espacio.

Si las cantidades a sumar son de distinto signo NO SE PRESENTAN DESBORDAMIENTOS

Ejemplo  $-225_{(10)} + 76_{(10)}$

# REPRESENTACIÓN DE BINARIOS NEGATIVOS

## Suma de binarios negativos

Realizar la siguiente suma en binario:  $-225_{(10)} + 76_{(10)}$

1. Se convierten los valores  $-225_{(10)}$  y  $76_{(10)}$  a binario.
2. Se transforma el valor negativo a complemento a 2.


$$\begin{array}{rcl}
 -225_{(10)} & = & 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 1_{(2)} \text{ Magnitud verdadera} \\
 & & 1\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 0_{(2)} \text{ Complemento a 1} \\
 & = & \begin{array}{r}
 \phantom{1\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1} 1 \\
 \hline
 1\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1_{(2)} \text{ Complemento a 2}
 \end{array}
 \end{array}$$

# REPRESENTACIÓN DE BINARIOS NEGATIVOS

## Suma de binarios negativos

3. Se aplica la operación suma entre los valores binarios.

$$\begin{array}{r}
 - \quad 2 \quad 2 \quad 5_{(10)} \\
 + \quad \quad 7 \quad 6_{(10)} \\
 \hline
 - \quad 1 \quad 4 \quad 9_{(10)}
 \end{array}
 =
 \begin{array}{r}
 1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1 \quad 1 \quad 1_{(2)} \\
 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0_{(2)} \\
 \hline
 1 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 1_{(2)}
 \end{array}$$


*Signo*
*Magnitud*

# REPRESENTACIÓN DE BINARIOS NEGATIVOS

## Suma de binarios negativos

3. Dado que el resultado obtenido es negativo (bit de signo en 1), este queda expresado en complemento a 2. Para realizar la comprobación del resultado es necesario realizar el complemento a 2 para obtener la magnitud verdadera.

1	0	1	1	0	1	0	1	1	Resultado negativo en Ca2
1	1	0	0	1	0	1	0	0	Complemento a 1
								1	
1	1	0	0	1	0	1	0	1	Ca2 del resultado = Magnitud verdadera

$$1\ 10010101_{(2)} = -149_{(10)}$$

$$-225_{(10)} + 76_{(10)} = -149_{(10)}$$