



Tema 1:

Representación digital de la información

Fundamentos de computadores

José Manuel Mendías Cuadros

*Dpto. Arquitectura de Computadores y Automática
Universidad Complutense de Madrid*





Contenidos

- ✓ Introducción de conceptos.
- ✓ Sistemas de numeración: binario, octal y hexadecimal.
- ✓ Aritmética binaria.
- ✓ Conversión entre bases.
- ✓ Representación de números enteros: MyS, C1 y C2.
- ✓ Aritmética entera: MyS y C2.
- ✓ Otras codificaciones.

Transparencias basadas en los libros:

- R. Hermida, F. Sánchez y E. del Corral. *Fundamentos de computadores*.
- D. Gajsky. *Principios de diseño digital*.

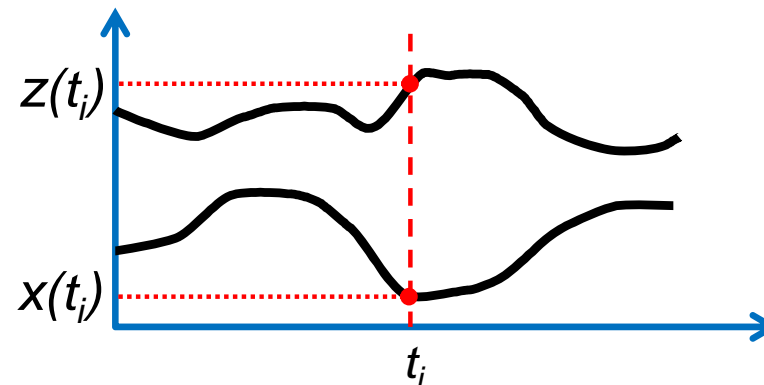


Concepto de sistema

- Sistema: caja "negra" que a lo largo del tiempo:
 - Recibe información por sus entradas, $x(t)$.
 - Procesa dicha información según una cierta función, F .
 - Genera información por sus salidas, $z(t)$.



$$z(t) = F(x(t))$$





Analógicos vs. digitales

■ Sistema analógico

- Los valores que pueden tomar las entradas/salidas pertenecen a un espectro continuo de valores.

■ Sistema digital

- Los valores que pueden tomar las entradas/salidas están restringidos a un conjunto discreto de valores.



Los sistemas analógicos establecen semejanzas, los digitales numerizan

Combinacionales vs. secuenciales



■ Sistema combinacional

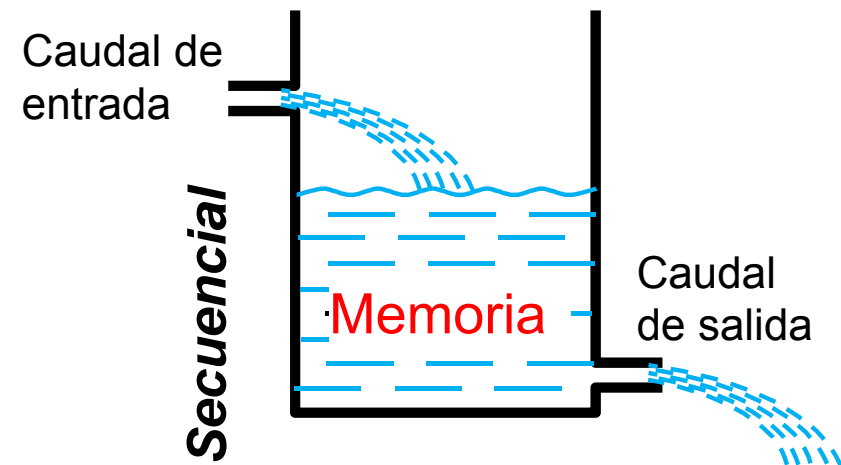
$$z(t_i) = F(x(t_i))$$

- La salida en cada instante depende exclusivamente del valor de la entrada en ese instante.

■ Sistema secuencial

$$z(t_i) = F(x(t)), \text{ con } t \in [0, t_i]$$

- La salida en cada instante depende del valor de la entrada en ese instante y de todos los valores que la entrada ha tomado con anterioridad.





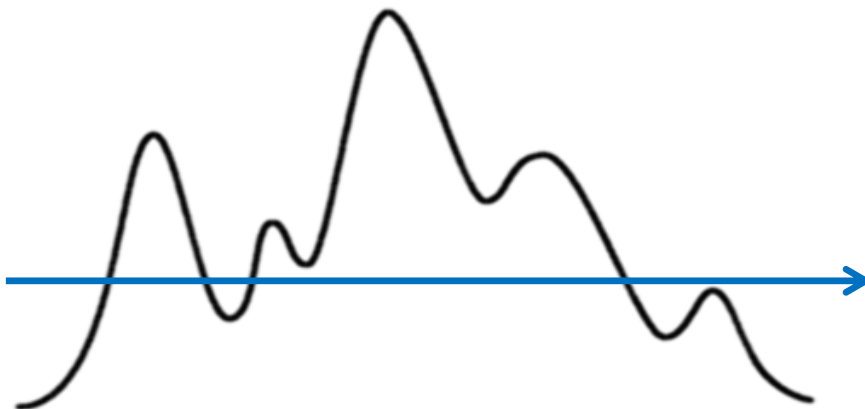
Asíncronos vs. síncronos

■ Asíncronos

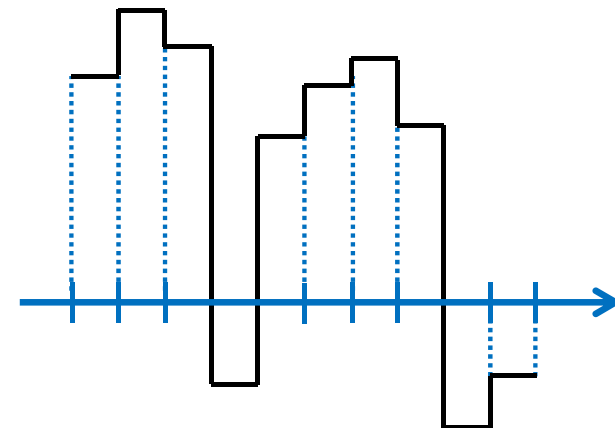
- Las entradas/salidas pueden cambiar en cualquier momento.

■ Síncronos

- Las entradas/salidas solo pueden cambiar en un conjunto discreto de instantes definidos por una señal de reloj.



asíncrono



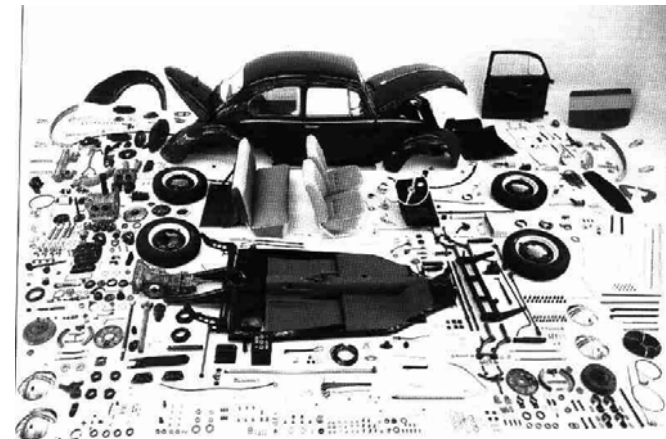
síncrono

Especificación vs. implementación



- Especificación (*¿qué hace?*)
 - Descripción del comportamiento de un sistema sin precisar cómo está constituido.
- Implementación (*¿cómo está hecho?*)
 - Descripción de un sistema en base a un conjunto de elementos más simples interconectados.

Coche (RAE): Vehículo automóvil de tamaño pequeño o mediano, destinado al transporte de personas y con capacidad no superior a nueve plazas.



Síntesis vs. análisis

■ Síntesis (o diseño)

- Proceso de obtener una implementación que tenga el comportamiento definido por una especificación dada.

■ Análisis

- Proceso de obtener el comportamiento de una implementación dada.

Para una especificación dada
existen multitud de
implementaciones válidas.



Temario FC 1er. cuatrimestre



1. Representación **digital** de la información.
2. **Especificación** de sistemas **combinacionales**.
3. **Implementación** de sistemas **combinacionales**.
4. Módulos **combinacionales** básicos.
5. **Especificación** de sistemas **secuenciales síncronos**.
6. **Implementación** de sistemas **secuenciales síncronos**.
7. Módulos **secuenciales** básicos.



Sistemas de numeración

- Mecanismo que permite dar una representación gráfica a cada número.
- Se define por:
 - Un conjunto discreto de símbolos (**dígitos**) cada uno de los cuales representa directamente un número.
 - la cardinalidad de este conjunto se llama BASE.
 - Un conjunto discreto de reglas de generación (**notación**) que permiten representar números mayores usando más de un dígito.
 - Un conjunto de reglas de manipulación de símbolos (**aritmética**) que permite realizar coherentemente operaciones con números.



Notación posicional

- Cada cantidad se representa utilizando una **cadena de dígitos** distinta

$$(a_{n-1}, a_{n-2} \dots a_1, a_0)_r$$

- a_{n-1} es el **dígito más significativo**
- a_0 es el **dígito menos significativo**
- r es la **base** del sistema de numeración

- El **valor de cada dígito** es función de la posición que ocupa en la cadena (peso). El peso de la posición i en un sistema de *base* r es r^i

$$(\text{valor dígito})_i = (\text{valor digito}) \times r^i$$

- El **valor de una cadena** es la suma del valor de cada uno de los dígitos que la forman.



Notación polinomial

- Cada cantidad se representa por un **polinomio** cuya resolución permite conocer el valor representado

$$\sum_{i=0}^{n-1} a_i \times r^i$$

Notación posicional	Notación polinomial	Cantidad representada
$(17)_{10}$	$1 \times 10^1 + 7 \times 10^0$	17
$(10001)_2$	$1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$	17
$(21)_8$	$2 \times 8^1 + 1 \times 8^0$	17
$(11)_{16}$	$1 \times 16^1 + 1 \times 16^0$	17



Sistemas base 10, 2, 8 y 16

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
16	10000	20	10

computadores

binario compacto



Aritmética binaria

- Aritmética de símbolos
 - Las tablas de sumar, restar, multiplicar... dígitos.

Suma	
$0 + 0 = 0$	
$0 + 1 = 1$	
$1 + 0 = 1$	
$1 + 1 = 0$	y me llevo 1

Resta	
$0 - 0 = 0$	
$0 - 1 = 1$	y me llevo 1
$1 - 0 = 1$	
$1 - 1 = 0$	

Multiplicación
$0 \times 0 = 0$
$0 \times 1 = 0$
$1 \times 0 = 0$
$1 \times 1 = 1$

- Aritmética de notación
 - El mecanismo para sumar, restar, multiplicar... cadenas de dígitos.

Suma binaria



$$S = 9 + 11$$

$$\begin{array}{r} 1 \\ \hline 9 \\ + 11 \\ \hline 20 \end{array}$$

$$\begin{array}{r} 1 \quad 1 \quad 1 \\ \hline 1 \quad 0 \quad 0 \quad 1 \\ 1 \quad 0 \quad 1 \quad 1 \\ \hline 1 \quad 0 \quad 1 \quad 0 \quad 0 \end{array}$$

acarreos

sumando 1

sumando 2

suma



Resta binaria

$$R = 83 - 21$$

$$\begin{array}{r} 83 \\ -21 \\ \hline \\ \hline 62 \end{array}$$

$$\begin{array}{r} 1010011 \\ -10101 \\ \hline 1111 \\ \hline 0111110 \end{array}$$

minuendo
sustraendo
acarreos
diferencia



Multiplicación binaria

$$P = 11 \times 5$$

$$\begin{array}{r} 11 \\ \times 5 \\ \hline 55 \end{array}$$

$$\begin{array}{r} 1011 \\ \times 101 \\ \hline 1011 \\ 0000 \\ 1011 \\ \hline 110111 \end{array}$$

multiplicando

multiplicador

productos parciales

resultado



División binaria

$$C = 117 / 9$$

dividendo

$$\begin{array}{r} 1110101 \\ - 1001 \\ \hline 01011 \\ - 1001 \\ \hline 001001 \\ - 1001 \\ \hline 0000 \end{array}$$

divisor

$$\begin{array}{r} 1001 \\ \hline 1101 \\ \text{cociente} \end{array}$$

resto



Conversión entre bases

■ Sustitución en serie

base R \rightarrow base S, usando la aritmética de base S

otra \rightarrow base 10

- Se evalúa la representación polinomial del número usando la aritmética de base S.

$$(2A)_{16} = 2 \times 16^1 + 10 \times 16^0 = 32 + 10 = (42)_{10}$$

$$\begin{aligned}(1010)_2 &= 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 \\ &= 8 + 0 + 2 + 0 = (10)_{10}\end{aligned}$$



Conversión entre bases

■ División por la base

base R \rightarrow base S, usando la aritmética en base R

base 10 \rightarrow otra

- Se divide sucesivamente el número por S reservando los restos hasta que el cociente sea menor que S.

$$(1270)_{10} = (4F6)_{16}$$

1	2	7	0		1	6			
-	1	1	2		7	9		1	6
<hr/>									
	1	5	0		-	6	4		4
	-	1	4	4		1	5		
<hr/>									
			6						

+ peso

$$(12)_{10} = (1100)_2$$

1	2		2				
-	1	2	6		2		
<hr/>							
	0		-	6	3		2
			0		-	2	1
					1		

+ peso



Conversión entre bases

■ Conversión entre potencias de la misma base

base R \rightarrow base $S=R^i$

base 2 \rightarrow base $8=2^3$ o base $16=2^4$

- Los dígitos de base R se agrupan de derecha a izquierda en bloques de i elementos.
- Cada bloque se reemplaza por el correspondiente dígito de base S.

$$(10011110110)_2 = (2366)_8$$

$$(100111101)_2 = (13D)_{16}$$



Conversión entre bases

■ Conversión entre potencias de la misma base

base $R=S^i \rightarrow$ base S

base $8=2^3$ o base $16=2^4 \rightarrow$ base 2

- Cada dígito de base R se remplaza por el correspondiente bloque de dígitos en base S .

$$(713)_8 = (111001011)_2$$

$$(A5C)_{16} = (101001011100)_2$$



Representación de la información

- Un sistema digital solo procesa **información digital codificada en binario**.
 - Una codificación es un **convenio** que asocia a cada elemento de información una representación binaria diferente.
 - Un mismo dato puede tener distintas representaciones en distintos códigos.
- Cada código usa un número de dígitos binarios fijo (bits de anchura) que limita el número de datos representable.
 - Con n bits como máximo se representan 2^n datos diferentes.
- El problema del **desbordamiento**:
 - En las codificaciones numéricas, se produce cuando el resultado de una operación aritmética no es representable (no hay un código que represente al resultado).
 - Deben detectarse porque el resultado **es incorrecto**.



Binario puro

- Codifica números naturales
- Notación n bits:
 - n bits codifican la magnitud en binario.

- Rango representable: $[0, 2^n - 1]$

- Aritmética:

$$6_{10} = (00110)_{2-5\text{bits}}$$

- Extensión (pasar n a m bits, con $m > n$)

- Completar con ceros por la izquierda.

- Suma

- Suma binaria
- Hay **desbordamiento** si al sumar el bit más significativo se produce un acarreo.

$$\begin{array}{rcccccc} & 1 & 0 & 1 & 1 & (11) \\ + & 0 & 1 & 1 & 1 & (7) \\ \hline 1 & 0 & 0 & 1 & 0 & (2 \neq 18) \end{array}$$



Magnitud y signo (MyS)

- Codifica números enteros
- Notación n bits:
 - 1 bit codifica el signo (el bit más significativo, bit de signo)
 - $n-1$ codifican la magnitud en binario.
 - Positivos: $+N = 0(N)_2$
 - Negativos: $-N = 1(N)_2$
- Rango representable: $[-(2^{n-1}-1), +(2^{n-1}-1)]$
 - el cero tiene doble representación (000..00) y (100..00)

$$6_{10} = (0110)_2 \Rightarrow$$

$$(+6_{10}) = (00110)_{\text{MyS-5bits}}$$

$$(-6_{10}) = (10110)_{\text{MyS-5bits}}$$



Magnitud y signo (MyS)

■ Procedimiento de codificación (n bits)

- Codificar el signo '+' \equiv '0', '-' \equiv '1'
- Codificar la magnitud en binario de n-1 bits usando división por la base.

$$-26_{10} \rightarrow \text{MyS de 8 bits} \left\{ \begin{array}{l} \text{signo} \equiv (1) \\ \text{magnitud} \equiv (0011010) \end{array} \right\} -26_{10} = (10011010)_{\text{MyS}}$$

$$+115_{10} \rightarrow \text{MyS de 8 bits} \left\{ \begin{array}{l} \text{signo} \equiv (0) \\ \text{magnitud} \equiv (1110011) \end{array} \right\} +115_{10} = (01110011)_{\text{MyS}}$$

■ Procedimiento de decodificación:

- Decodificar el signo '0' \equiv '+', '1' \equiv '-'
- Decodificar la magnitud usando sustitución en serie.

$$(10010010)_{\text{MyS}} \rightarrow \text{decimal} \left\{ \begin{array}{l} \text{signo} \equiv '-' \\ \text{magnitud} \equiv 18_{10} \end{array} \right\} (10010010)_{\text{MyS}} = -18_{10}$$

$$(01011010)_{\text{MyS}} \rightarrow \text{decimal} \left\{ \begin{array}{l} \text{signo} \equiv '+' \\ \text{magnitud} \equiv 90_{10} \end{array} \right\} (01011010)_{\text{MyS}} = +90_{10}$$



Aritmética en MyS

■ Cambio de signo (cambiar un número por su opuesto)

- Cambiar el bit de signo

$$- (00110)_{\text{MyS-5bits}} = (10110)_{\text{MyS-5bits}}$$

■ Extensión (pasar n a m bits, con $m > n$)

- Manteniendo el signo, completar la magnitud con ceros por la izquierda.

$$(-6_{10}) = (10110)_{\text{MyS-5bits}} = (10000110)_{\text{MyS-8bits}}$$

■ Suma / Resta

- Signo y magnitud de manipulan por separado.
- El signo del resultado depende de las magnitudes y signos de los operandos.
- Las magnitudes se suman o restan en función de la magnitud y signo de los operandos.



Aritmética en MyS: suma

■ Signo (A) = signo (B)

- Signo (R) = signo (A) = signo (B)
- Magnitud (R) = magnitud (A) + magnitud (B)

$\begin{array}{r} + \vdots 4 \\ + \vdots 2 \\ \hline + \vdots 6 \end{array}$	$\begin{array}{r} 0 \vdots 1 \ 0 \ 0 \\ + \ 0 \vdots 0 \ 1 \ 0 \\ \hline 0 \vdots 1 \ 1 \ 0 \end{array}$	$\begin{array}{r} - \vdots 4 \\ + \ - \vdots 2 \\ \hline - \vdots 6 \end{array}$	$\begin{array}{r} 1 \vdots 1 \ 0 \ 0 \\ + \ 1 \vdots 0 \ 1 \ 0 \\ \hline 1 \vdots 1 \ 1 \ 0 \end{array}$
--	--	--	--

■ Signo (A) = positivo, signo (B) = negativo, $|A| \geq |B|$

- Signo (R) = signo (A) = positivo
- Magnitud (R) = magnitud (A) - magnitud (B)

$\begin{array}{r} + \vdots 4 \\ + \ - \vdots 2 \\ \hline + \vdots \end{array}$	$\begin{array}{r} \vdots 4 \\ - \vdots 2 \\ \hline \vdots 2 \end{array}$	$\begin{array}{r} 0 \vdots 1 \ 0 \ 0 \\ + \ 1 \vdots 0 \ 1 \ 0 \\ \hline 0 \vdots \end{array}$	$\begin{array}{r} \vdots 1 \ 0 \ 0 \\ - \vdots 0 \ 1 \ 0 \\ \hline \vdots 0 \ 1 \ 0 \end{array}$
--	--	--	--



Aritmética en MyS: suma

- Signo (A) = positivo, signo (B) = negativo, $|A| < |B|$
 - Signo (R) = signo (B) = negativo
 - Magnitud (R) = magnitud (B) - magnitud (A)

$\begin{array}{r} + \quad \vdots 2 \\ + \quad - \quad \vdots 4 \\ \hline - \quad \vdots \end{array}$	$\begin{array}{r} \vdots 4 \\ - \quad \vdots 2 \\ \hline \vdots 2 \end{array}$	$\begin{array}{r} 0 \quad \vdots 0 \quad 1 \quad 0 \\ + \quad 1 \quad \vdots 1 \quad 0 \quad 0 \\ \hline 1 \quad \vdots \end{array}$	$\begin{array}{r} \vdots 1 \quad 0 \quad 0 \\ - \quad \vdots 0 \quad 1 \quad 0 \\ \hline \vdots 0 \quad 1 \quad 0 \end{array}$
--	--	--	--

- Resto de casos / Resta
 - Equivalente a alguno de los anteriores si se aplica conmutatividad.
- Desbordamiento
 - Hay **desbordamiento** si al operar con el bit más significativo de la magnitud se produce un acarreo.



Complemento a dos (C2)

- Codifica números enteros

- Notación n bits:

- Positivos: $+N = 0(N)_2$
- Negativos: $-N = (2^n - N)_2 = C2((N)_2)$
 - el bit más significativo se denomina bit de signo

- Rango representable: $[-(2^{n-1}), +(2^{n-1}-1)]$

- el cero tiene una única representación (000..00)
- el rango es asimétrico, hay un negativo de más (100..00)

$$6_{10} = (0110)_2 \Rightarrow (+6_{10}) = (00110)_{C2-5bits}$$

$$(2^5 - 6)_{10} = (26)_{10} = (11010)_2 \Rightarrow (-6_{10}) = (11010)_{C2-5bits}$$



Complemento a dos (C2)

■ Procedimiento de codificación (n bits)

- Si el número es **positivo**, codificar en binario de n bits usando el método de división por la base.

$$+93_{10} \rightarrow \text{C2 de 8 bits} \left[93_{10} = (01011101)_2 \right] +93_{10} = (01011101)_{C2}$$

- Si el número es **negativo**, codificar el número prescindiendo del signo en binario de n bits usando el método de división por la base y realizar el complemento a dos del resultado.

$$-78_{10} \rightarrow \text{C2 de 8 bits} \left[\begin{array}{l} 78_{10} = (01001110)_2 \\ \text{C2}(01001110) = (10110010) \end{array} \right] -78_{10} = (10110010)_{C2}$$



Complemento a dos (C2)

■ Procedimiento de decodificación:

- Si el bit de signo es **positivo** (vale '0'), decodificarlo usando el método de sustitución en serie.

$$(01110001)_{C2} \rightarrow \text{decimal} \left\{ (01110001)_2 = (113)_{10} \right\} (01110001)_{C2} = +113_{10}$$

- Si el bit de signo es **negativo** (vale '1'), realizar su complemento a dos y decodificar el resultado usando el método de sustitución en serie.

$$(10110100)_{C2} \rightarrow \text{decimal} \left\{ \begin{array}{l} C2(10110100) = (01001100) \\ (01001100)_2 = (76)_{10} \end{array} \right\} (10110100)_{C2} = -76_{10}$$



Aritmética en C2

■ Cambio de signo (cambiar un número por su opuesto)

- Complementar a dos el número

$$-(00110)_{C2-5bits} = C2(00110) = (11010)_{C2-5bits}$$

- Para realizar la operación C2 hay varias opciones:

- Restar el número a 2^n
- Invertir todos los bits y sumar 1
- Copiar los bits de derecha a izquierda hasta encontrar el primer 1, invertir el resto.

■ Extensión (pasar n a m bits, con $m > n$)

- Replicar el bit de signo hacia la izquierda

$$(-6_{10}) = (\textcolor{red}{1}1010)_{C2-5bits} = (\textcolor{red}{1111}1010)_{C2-8bits}$$



Aritmética en C2: suma

- Signo (A) = signo (B)

- $R = A + B$

$\begin{array}{r} + 4 \\ + + 2 \\ \hline + 6 \end{array}$	$\begin{array}{r} 0 1 0 0 \\ + 0 0 1 0 \\ \hline 0 1 1 0 \end{array}$	$\begin{array}{r} - 4 \\ + - 2 \\ \hline - 6 \end{array}$	$\begin{array}{r} 1 1 0 0 \\ + 1 1 1 0 \\ \hline 1 1 0 1 0 \end{array}$
---	---	---	---

- Signo (A) = positivo, signo (B) = negativo, $|A| \geq |B|$

- $R = A + B$

$\begin{array}{r} + 4 \\ + - 2 \\ \hline + 2 \end{array}$	$\begin{array}{r} 0 1 0 0 \\ + 1 1 1 0 \\ \hline 1 0 0 1 0 \end{array}$
---	---



Aritmética en C2: suma

■ Desbordamiento

- En la **suma**, solo puede producirse si ambos operandos son del mismo signo. En la **resta**, solo si son de distinto signo.
- Se detecta chequeando si el signo del resultado es coherente con el signo de los operandos.
- **NO** se tiene en cuenta el acarreo del bit más significativo.

0	0	1	1	(+3)	1	0	1	1	(-5)
+	0	1	1	(+6)	+	1	0	1	(-6)
<hr/>					<hr/>				
1	0	0	1	(-7≠+9)	1	0	1	0	(+5≠-11)

el rango representable con 4 bits es: [-8, +7]



Complemento a uno (C1)

- Codifica números enteros
- Notación n bits:
 - Positivos: $+N = 0(N)_2$
 - Negativos: $-N = (2^n - 1 - N)_2 = C1((N)_2)$
 - el bit más significativo se denomina bit de signo
- Rango representable: $[-(2^{n-1}-1), +(2^{n-1}-1)]$
 - el cero tiene doble representación (000..00) y (111..11)

$$6_{10} = (0110)_2 \Rightarrow (+6_{10}) = (00110)_{C1-5bits}$$

$$(2^5 - 1 - 6)_{10} = (25)_{10} = (11001)_2 \Rightarrow (-6_{10}) = (11001)_{C1-5bits}$$



Complemento a uno (C1)

■ Procedimiento de codificación (n bits)

- Si el número es **positivo**, codificar en binario de n bits usando el método de división por la base.

$$+40_{10} \rightarrow \text{C1 de 8 bits} \left[40_{10} = (00101000)_2 \right] +40_{10} = (00101000)_{C1}$$

- Si el número es **negativo**, codificar el número prescindiendo del signo en binario de n bits usando el método de división por la base y realizar el complemento a uno del resultado.

$$-62_{10} \rightarrow \text{C1 de 8 bits} \left[\begin{array}{l} 62_{10} = (00111110)_2 \\ \text{C1}(00111110) = (11000001) \end{array} \right] -62_{10} = (11000001)_{C1}$$



Complemento a uno (C1)

■ Procedimiento de decodificación:

- Si el bit de signo es **positivo** (vale '0'), decodificarlo usando el método de sustitución en serie.

$$(00100010)_{C1} \rightarrow \text{decimal} \left\{ (00100010)_2 = (34)_{10} \right\} (00100010)_{C1} = +34_{10}$$

- Si el bit de signo es **negativo** (vale '1'), realizar su complemento a uno y decodificar el resultado usando el método de sustitución en serie.

$$(11001001)_{C1} \rightarrow \text{decimal} \left\{ \begin{array}{l} C1(11001001) = (00110110) \\ (00110110)_2 = (54)_{10} \end{array} \right\} (11001001)_{C1} = -54_{10}$$



Aritmética en C1

■ Cambio de signo (cambiar un número por su opuesto)

- Complementar a uno el número

$$-(00110)_{C1-5bits} = C1(00110) = (11001)_{C1-5bits}$$

- Para realizar la operación C1 hay varias opciones:

- Restar el número a $2^n - 1$
- Invertir todos los bits

■ Extensión (pasar n a m bits, con $m > n$)

- Replicar el bit de signo hacia la izquierda

$$(-6_{10}) = (\textcolor{red}{1}1001)_{C1-5bits} = (\textcolor{red}{1111}1001)_{C2-8bits}$$

Comparación códigos (4 bits)



Decimal	MyS	C2	C1
+7	0111	0111	0111
+6	0110	0110	0110
+5	0101	0101	0101
+4	0100	0100	0100
+3	0011	0011	0011
+2	0010	0010	0010
+1	0001	0001	0001
+0	0000	0000	0000
-0	1000	----	1111
-1	1001	1111	1110
-2	1010	1110	1101
-3	1011	1101	1100
-4	1100	1100	1011
-5	1101	1011	1010
-6	1110	1010	1001
-7	1111	1001	1000
-8	----	1000	----



Representaciones decimales

■ BCD (Binary Coded Decimal)

- Cada dígito decimal se representa por un bloque de 4 bits (*nibble*) que lo codifica en binario.

$$(375)_{10} = (001101110101)_{\text{BCD}}$$

■ Exceso-3

- Cada dígito decimal se representa por un bloque de 4 bits que codifica en binario el valor del dígito + 3.

$$(375)_{10} = (011010101000)_{\text{EX-3}}$$

Simplifican la conversión decimal-binario y evitan pérdidas de precisión en la conversión de números con parte fraccionaria



Representaciones de alfabetos

- ASCII (American Standard Code for Information Interchange)
 - Codifica el alfabeto latino occidental con 7 bits.
 - Los códigos 00h-1Fh (0-31) y el 7Fh (127) son de control.
 - Los códigos 20h-7Eh (32-126) son imprimibles.
 - Hay diferentes extensiones de 8 bits (1 byte) para soportar más caracteres imprimibles.
- EBCDIC (Extended Binary Coded Decimal Interchange Code)
 - Codifica el alfabeto latino occidental con 8 bits



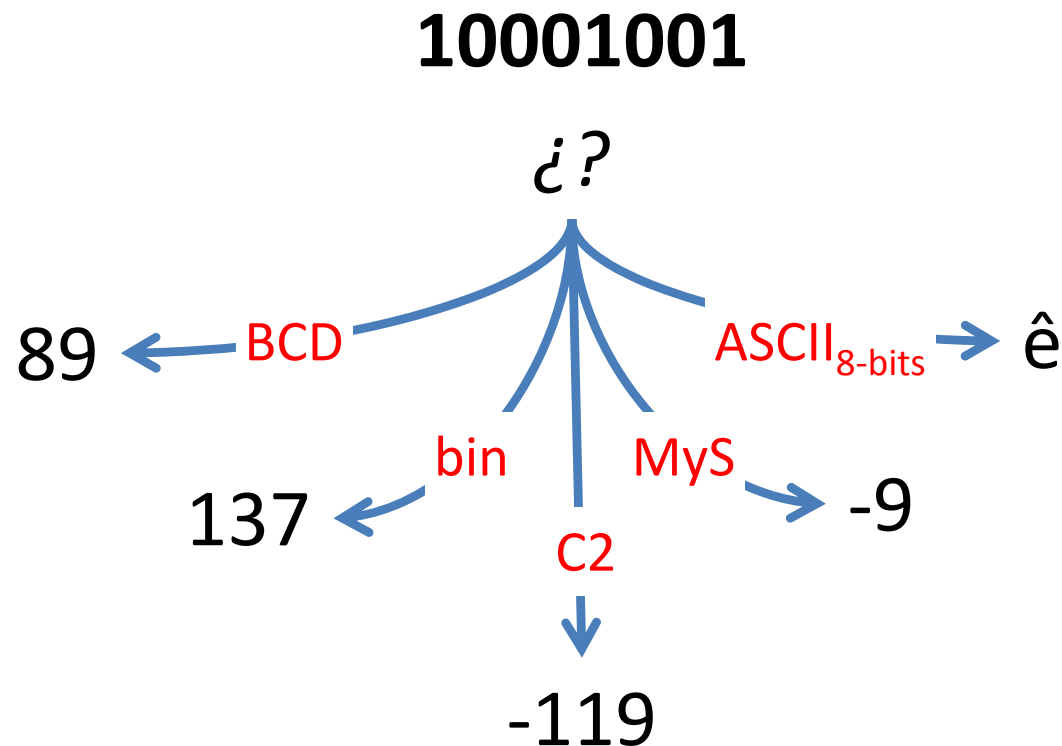
Código ASCII (7 bits)

ASCII Hex Simbolo	ASCII Hex Simbolo	ASCII Hex Simbolo	ASCII Hex Simbolo
0 0 NUL	16 10 DLE	32 20 (espacio)	48 30 0
1 1 SOH	17 11 DC1	33 21 !	49 31 1
2 2 STX	18 12 DC2	34 22 "	50 32 2
3 3 ETX	19 13 DC3	35 23 #	51 33 3
4 4 EOT	20 14 DC4	36 24 \$	52 34 4
5 5 ENQ	21 15 NAK	37 25 %	53 35 5
6 6 ACK	22 16 SYN	38 26 &	54 36 6
7 7 BEL	23 17 ETB	39 27 '	55 37 7
8 8 BS	24 18 CAN	40 28 (56 38 8
9 9 TAB	25 19 EM	41 29)	57 39 9
10 A LF	26 1A SUB	42 2A *	58 3A :
11 B VT	27 1B ESC	43 2B +	59 3B ;
12 C FF	28 1C FS	44 2C ,	60 3C <
13 D CR	29 1D GS	45 2D -	61 3D =
14 E SO	30 1E RS	46 2E .	62 3E >
15 F SI	31 1F US	47 2F /	63 3F ?
ASCII Hex Simbolo	ASCII Hex Simbolo	ASCII Hex Simbolo	ASCII Hex Simbolo
64 40 @	80 50 P	96 60 `	112 70 p
65 41 A	81 51 Q	97 61 a	113 71 q
66 42 B	82 52 R	98 62 b	114 72 r
67 43 C	83 53 S	99 63 c	115 73 s
68 44 D	84 54 T	100 64 d	116 74 t
69 45 E	85 55 U	101 65 e	117 75 u
70 46 F	86 56 V	102 66 f	118 76 v
71 47 G	87 57 W	103 67 g	119 77 w
72 48 H	88 58 X	104 68 h	120 78 x
73 49 I	89 59 Y	105 69 i	121 79 y
74 4A J	90 5A Z	106 6A j	122 7A z
75 4B K	91 5B [107 6B k	123 7B {
76 4C L	92 5C \	108 6C l	124 7C
77 4D M	93 5D]	109 6D m	125 7D }
78 4E N	94 5E ^	110 6E n	126 7E ~
79 4F O	95 5F _	111 6F o	127 7F □



No olvidar

Una cadena de bits por sí misma no significa nada



es la codificación usada la que le da sentido

Acerca de *Creative Commons*



■ Licencia CC (*Creative Commons*)

- Ofrece algunos derechos a terceras personas bajo ciertas condiciones. Este documento tiene establecidas las siguientes:



Reconocimiento (*Attribution*):

En cualquier explotación de la obra autorizada por la licencia hará falta reconocer la autoría.



No comercial (*Non commercial*):

La explotación de la obra queda limitada a usos no comerciales.



Compartir igual (*Share alike*):

La explotación autorizada incluye la creación de obras derivadas siempre que mantengan la misma licencia al ser divulgadas.

Más información: <https://creativecommons.org/licenses/by-nc-sa/4.0/>