



# Tema 3

## Aritmètica i representació de la informació en l'ordinador

Informàtica  
Grau en Física  
Universitat de València

[Francisco.Grimaldo@uv.es](mailto:Francisco.Grimaldo@uv.es)  
[Ariadna.Fuertes@uv.es](mailto:Ariadna.Fuertes@uv.es)






- Representació binària de la informació.
  - ◆ Definició.
  - ◆ Agrupament dels bits.
  - ◆ Unitats de mesura binàries.
- Informació bàsica representada.
  - ◆ Informació booleana.
  - ◆ Conversió decimal - binari.
  - ◆ Operacions en binari.
  - ◆ Enters amb signe.
  - ◆ Caràcters.
  - ◆ Reals.
- Representació intermèdia: octal i hexadecimal.
  - ◆ Definició.
  - ◆ Conversió decimal - hexadecimal/octal - binari.
- Exercicis.

# Representació binària: definició (1/2)

- Els ordinadors representen **qualsevol** tipus d'informació (text, imatges, so...) com un patró de bits que poden estar en dos estats possibles: encès (1) o apagat (0).
- El **bit** és la unitat mínima d'informació que pot processar un dispositiu digital.
- Un bit té un dels dos valors possibles: **0 o 1**.
- No obstant això, un ordinador treballa amb informació diferent de zeros i uns.
- Com ho fa? Mitjançant la **codificació**.

# Representació binària: definició (2/2)

- Amb un bit podem representar només dos valors, 0 i 1.
- Per a representar o codificar més informació en un dispositiu digital, necessitem més quantitat de bits.
- Per mitjà de **seqüències de bits** podem representar qualsevol tipus d'informació.
- ... es coneix com a **codificació**.

	
VALOR 1	VALOR 0



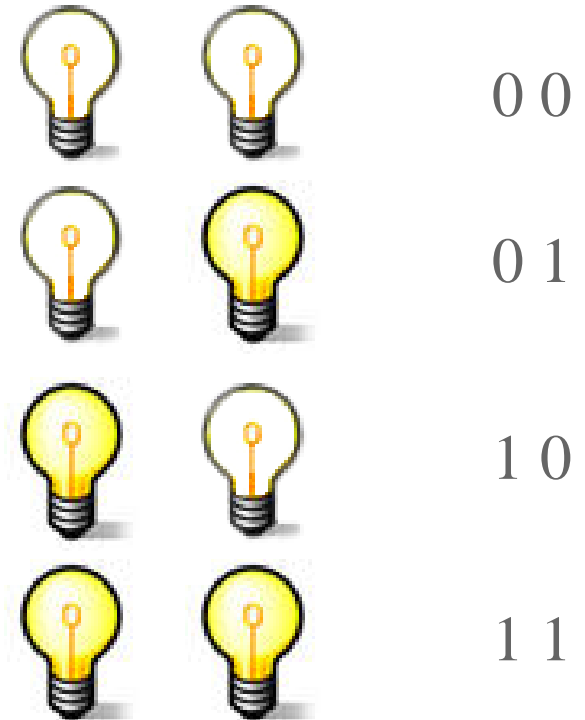
# Agrupament dels bits

- Per exemple, si disposem de dos bits, podem representar fins a quatre valors diferents,  $2^2$ .

- En general, tenint en compte que cada bit pot representar només dos valors (0 i 1), **amb n bits podrem representar  $2^n$  valors.**

- **Byte** és un agrupament de 8 bits.

- Un byte pot representar fins a 256 ( $2^8$ ) missatges diferents. P. ex.: 10010011.





# Unitats de mesura binàries

- **Byte:** grup de 8 bits.
- **KB (kilobyte o K):** grup de 1.024 bytes. Conté  $2^{10}$  bytes = 1.024 bytes  $\approx$  1.000 bytes =  $10^3$ .
- **MB (megabyte o mega):** grup de 1.048.576 bytes. Conté  $2^{20}$  bytes = 1.048.576 bytes  $\approx$  1.000.000 bytes = 1.000 KB =  $10^6$ .
- **GB (gigabyte o giga):** 1.024 MB  $\approx$  1.000 MB.
- **TB (terabyte):** 1 milió de MB o bilió de bytes.
- **PB (petabyte):** 1.024 TB o 1.000 bilions de bytes.
- **Exemples:** 4 GB RAM, HDD 120 GB, MP3 3.5 MB.
- La transferència de dades es mesura en bits/s.  
Per tant, 1 Mb/s = 1.000 bits = 1/8 MB.



- Representació binària de la informació.
  - ◆ Definició.
  - ◆ Agrupament dels bits.
  - ◆ Unitats de mesura binàries.
- **Informació bàsica representada.**
  - ◆ Informació booleana.
  - ◆ Conversió decimal - binari.
  - ◆ Operacions en binari.
  - ◆ Enters amb signe.
  - ◆ Caràcters.
  - ◆ Reals.
- Representació intermèdia: octal i hexadecimal.
  - ◆ Definició.
  - ◆ Conversió decimal - hexadecimal/octal - binari.
- Exercicis.



# Informació bàsica representada

- La informació bàsica representada/desada en un dispositiu digital serà:
  - ◆ **Informació booleana:** Vertader o Fals.
  - ◆ **Valors enters:** Nombres enters positius i negatius.
  - ◆ **Caràcters:** Lletres de l'alfabet de la “a” a la “z”, en majúscules i minúscules, els dígitos numèrics del 0 al 9 i altres caràcters de control.
  - ◆ **Valors reals:** Nombres amb decimals.
- Tota aquesta informació ha d'estar codificada en binari per a poder desar-la en un dispositiu digital.
- Codificació de la informació booleana:
  - ◆ Vertader = 1 i Fals = 0.





# Conversió decimal – binari (1/4)

- Un nombre enter positiu qualsevol, representat en base decimal, es pot codificar com una **suma de potències** de 10.
  - ◆ Per exemple:  $325 = 3 \cdot 10^2 + 2 \cdot 10^1 + 5 \cdot 10^0$ .
- En general, un nombre **N** es pot codificar a partir d'un conjunt de **a** símbols (alfabet) que prenen un determinat valor en funció de la posició que ocupen  **$B^i$**  (base del sistema de numeració o representació) de la manera següent:

$$N = \sum_{i=0}^n a_i * B^i$$



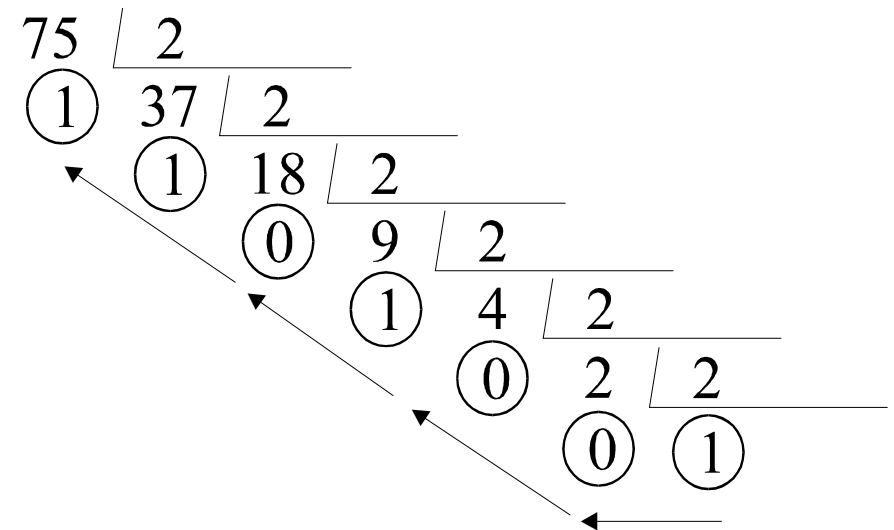
# Conversió decimal – binari (2/4)

- La base amb què treballem habitualment és base 10 ( $B=10$ ) i l'**alfabet** amb què escrivim normalment els nombres són els deu dígit {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}.
- El nombre 237 es pot codificar mitjançant la **base** i els **coeficients** corresponents:  $2 \cdot 10^2 + 3 \cdot 10^1 + 7 \cdot 10^0$ , és a dir,  $2 \cdot 100 + 3 \cdot 10 + 7 \cdot 1 = 237$ .
- Els ordinadors fan servir un alfabet format pels dígit 0 i 1 (dos símbols), de manera que la base que utilitzen és la **base binària** ( $B=2$ ).
- **Com convertim** un nombre en base 10 en un nombre en base 2 o binari?



# Conversió decimal – binari (3/4)

- El canvi de base de **decimal a binari** es fa mitjançant **divisions successives**.
- Per exemple, per a passar 75 a binari fem la seqüència de divisions següent, i així  $75_{10} = 1001011_2$
- Quan ja no podem dividir més el nombre, **la successió inversa des de l'últim quocient més els residus formen el nombre binari.**





# Conversió decimal – binari (4/4)

- Per a passar de **binari a decimal**, farem la descomposició de la xifra en les **successives potències** segons la fórmula (B=2):

$$N = \sum_{i=0}^n a_i * 2^i$$

- Per exemple, per a convertir el nombre 1001011 a base 10, el descompondrem com:

$$\begin{aligned} \mathbf{1001011} &= 1*2^0 + 1*2^1 + 0*2^2 + 1*2^3 + 0*2^4 + 0*2^5 + 1*2^6 \\ &= 1*1 + 1*2 + 0*4 + 1*8 + 0*16 + 0*32 + 1*64 \\ &= 1 + 2 + 8 + 64 \\ &= \mathbf{75} \end{aligned}$$



# Operacions en binari

- La metodologia de les operacions és la mateixa que en decimal, excepte que en binari:
  - ◆  $1 + 1 = 10$  i  $10 - 1 = 01$

- **Exemple:**  $75 + 23$  (en decimal  $98 \equiv 1100010$ ).
  - ◆  $75 \equiv 1001011$  i  $23 \equiv 10111$

El resultat és l'esperat:

$$\begin{array}{r} 1001011 \\ + 10111 \\ \hline 1100010 \end{array} \quad 0 \cdot 2^0 + 1 \cdot 2^1 + 0 \cdot 2^2 + 0 \cdot 2^3 + 0 \cdot 2^4 + 1 \cdot 2^5 + 1 \cdot 2^6 = 98$$

- **Exemple:**  $75 - 23$  (en decimal  $52 \equiv 110100$ ).

El resultat és l'esperat:

$$\begin{array}{r} 1001011 \\ - 10111 \\ \hline 0110100 \end{array} \quad 0 \cdot 2^0 + 0 \cdot 2^1 + 1 \cdot 2^2 + 0 \cdot 2^3 + 1 \cdot 2^4 + 1 \cdot 2^5 + 0 \cdot 2^6 = 52$$



# Enters amb signe (1/9)

- **Representació signe - magnitud:** consisteix a dedicar un espai per a indicar el signe del nombre.
  - ◆ El nombre 75 pot ser +75 o -75.
- En la memòria de l'ordinador, és impossible la representació dels símbols + o -, **només es poden representar uns i zeros.**
- El que farem serà dedicar un espai (bit) concret de la representació al signe, que també serà un zero o un u, però que per estar en una certa posició no indicarà magnitud sinó signe.
- Si parlem d'un lloc determinat on anirà el signe, també haurem de parlar d'un nombre determinat d'espais per a poder localitzar el signe i la magnitud. Així, **d'ara endavant qualsevol representació binària d'un nombre comportarà el nombre de bits en què es farà la representació** (i al costat, el mètode en què s'ha representat).



# Enters amb signe (2/9)

- **Exemple:** Representeu els nombres  $75_{10}$  i  $-75_{10}$  en base binària, amb signe i magnitud en 8 bits ( $n=8$ ).

$$75_{10} = 01001011_{2SM}$$

$$-75_{10} = 11001011_{2SM}$$

- Si en fer la representació hi ha buits (caselles sense omplir), aquests s'han de deixar sempre a la dreta del signe i a l'esquerra de la magnitud, i s'han d'omplir sempre amb zeros:

$$8_{10} = 0\underline{000}1000_{2SM}$$

$$-8_{10} = 1\underline{000}1000_{2SM}$$

- **Inconvenient:** quan es realitzen operacions, cal comprovar el signe i, segons aquest, fer una suma o una resta, veure quina magnitud és major...

- **Exercici:** Convertiu el nombre  $-46_{10}$  a binari amb 8 bits SM.



# Enters amb signe (3/9)

- **Representació complement a 1:** consisteix a aplicar a cada nombre enter  $x$  la transformació següent:  $x' = (2n - 1) + x$
- **Exemple:** Representar tots els nombres possibles amb 4 bits ( $n=4$ ).

$$24 = 10000 \rightarrow 24 - 1 = 1111$$

0  $\rightarrow 1111 + 0 = 1111$   
1  $\rightarrow 1111 + 1 = (1)0000 \rightarrow$  (se suma el transport)  $\rightarrow 0000 + 1 = 0001$   
2  $\rightarrow 1111 + 10 = (1)0001 \rightarrow$  (se suma el transport)  $\rightarrow 0001 + 1 = 0010$   
3  $\rightarrow 1111 + 11 = (1)0010 \rightarrow$  (se suma el transport)  $\rightarrow 0010 + 1 = 0011$   
4  $\rightarrow 1111 + 100 = (1)0011 \rightarrow$  (se suma el transport)  $\rightarrow 0011 + 1 = 0100$   
5  $\rightarrow 1111 + 101 = (1)0100 \rightarrow$  (se suma el transport)  $\rightarrow 0100 + 1 = 0101$   
6  $\rightarrow 1111 + 110 = (1)0101 \rightarrow$  (se suma el transport)  $\rightarrow 0101 + 1 = 0110$   
7  $\rightarrow 1111 + 111 = (1)0110 \rightarrow$  (se suma el transport)  $\rightarrow 0110 + 1 = 0111$   
8  $\rightarrow 1111 + 1000 = (1)0111 \rightarrow$  (se suma el transport)  $\rightarrow 0111 + 1 = 1000$   
(número no vàlid perquè és igual a -7)

-1  $\rightarrow 1111 - 1 = 1110$   
-2  $\rightarrow 1111 - 10 = 1101$   
-3  $\rightarrow 1111 - 11 = 1100$   
-4  $\rightarrow 1111 - 100 = 1011$   
-5  $\rightarrow 1111 - 101 = 1010$   
-6  $\rightarrow 1111 - 110 = 1001$   
-7  $\rightarrow 1111 - 111 = 1000$   
-8  $\rightarrow 1111 - 1000 = 0111$  (número no vàlid perquè és igual a 7)





# Enters amb signe (4/9)

## ■ Característiques del C1:

- ◆ El 1r bit dels nombres positius és un 0 i el dels negatius és un 1.
- ◆ Els nombres positius es representen igual, però omplits amb zeros a l'esquerra fins a arribar al total de bits de la representació.
- ◆ Els nombres negatius es poden obtenir ràpidament:
  - 1) Representar el número en binari com si fóra positiu.
  - 2) Omplir amb zeros fins a la quantitat de bits de la representació.
  - 3) Canviar els zeros per uns i els uns, per zeros.

- ◆ La suma i la resta esdevenen només una operació: SUMA

$$A - B = A + (-B)$$

- Tota representació binària duu associada una grandària. Si aquesta no és prou gran, poden sorgir dos errors quan operem:

- ◆ **Overflow**: Un nombre positiu esdevé negatiu.

**Underflow**: Un nombre negatiu esdevé positiu.



# Enters amb signe (5/9)

- **Exemple 1:** Supposem  $n = 4$ . Sumar  $A=5$  i  $B=2$ .

$$A = 5_{10} = 0101_{2C1}$$

$$B = 2_{10} = 0010_{2C1}$$

$$A+B = 0111_{2C1} = 7_{10}$$

$$\begin{array}{r} 0101 \\ + 0010 \\ \hline 0111 \end{array}$$

- **Exemple 2:** Supposem  $n = 4$ . Restar  $A=5$  i  $B=6 \equiv A + (-B)$ .

$$A = 5_{10} = 0101_{2C1}$$

$$-B = -6_{10} = -0110_2 = 1001_{2C1}$$

$$A+(-B) = 1110_{2C1} = -0001_2 = -1_{10}$$

$$\begin{array}{r} 0101 \\ + 1001 \\ \hline 1110 \end{array}$$

- **Exemple 3:** Supposem  $n = 4$ . Sumar  $A=5$  i  $B=6$ .

$$A = 5_{10} = 0101_{2C1}$$

$$B = 6_{10} = 0110_{2C1}$$

$$A+B = 1011_{2C1} = -0100_2 = -4_{10} \rightarrow \text{Overflow}$$

$$\begin{array}{r} 0101 \\ + 0110 \\ \hline 1011 \end{array}$$

- **Exemple 4:** Supposem  $n = 4$ . Sumar  $A=-5$  i  $B=-6$ .

$$A = -5_{10} = -0101_2 = 1010_{2C1}$$

$$B = -6_{10} = -0110_2 = 1001_{2C1}$$

$$A+B = 0100_{2C1} = 4_{10} \rightarrow \text{Underflow}$$

$$\begin{array}{r} 1010 \\ + 1001 \\ \hline (1)0011 \end{array}$$

El transport se suma:

$$0011+1=0100$$



# Enters amb signe (6/9)

- **Inconvenient:** Com que el 0 té dues representacions, només podem representar  $2^n - 1$  valors en comptes de  $2^n$ .
- **Exercici:** Convertiu els valors següents a binari en C1 amb 8 bits i feu les operacions indicades tot i indicant si el resultat és correcte. Si no ho és, detecteu el tipus d'errada que es produeix:

- ◆ A = 45) 1 0
- ◆ B = -86) 1 0

- ◆ A + B
- ◆ B - A
- ◆ A - B



# Enters amb signe (7/9)

- **Representació complement a 2:** consisteix a aplicar a un nombre enter  $x$  la mateixa transformació que en C1, però afegint-li 1 i no tenint en compte el transport:

$$x' = (2^n - 1) + x + 1$$

- **Exemple:** Representar tots els nombres possibles amb 4 bits ( $n=4$ ).

$$24 = 10000 \rightarrow 24 - 1 = 1111$$

$0 \rightarrow 1111 + 0 + 1 = (1) 0 0 0 0 \rightarrow$  i NO es té en compte el transport  $\rightarrow 0 0 0 0_{2C2}$   
 $1 \rightarrow 1111 + 1 + 1 = (1) 0 0 0 1 \rightarrow$  i NO es té en compte el transport  $\rightarrow 0 0 0 1_{2C2}$   
 $2 \rightarrow 1111 + 10 + 1 = (1) 0 0 1 0 \rightarrow$  i NO es té en compte el transport  $\rightarrow 0 0 1 0_{2C2}$   
 $3 \rightarrow 1111 + 11 + 1 = (1) 0 0 1 1 \rightarrow$  i NO es té en compte el transport  $\rightarrow 0 0 1 1_{2C2}$   
 $4 \rightarrow 1111 + 100 + 1 = (1) 0 1 0 0 \rightarrow$  i NO es té en compte el transport  $\rightarrow 0 1 0 0_{2C2}$   
 $5 \rightarrow 1111 + 101 + 1 = (1) 0 1 0 0 \rightarrow$  i NO es té en compte el transport  $\rightarrow 0 1 0 1_{2C2}$   
 $6 \rightarrow 1111 + 110 + 1 = (1) 0 1 1 0 \rightarrow$  i NO es té en compte el transport  $\rightarrow 0 1 1 0_{2C2}$   
 $7 \rightarrow 1111 + 111 + 1 = (1) 0 1 1 1 \rightarrow$  i NO es té en compte el transport  $\rightarrow 0 1 1 1_{2C2}$   
 $8 \rightarrow 1111 + 1000 + 1 = (1) 1 0 0 0 \rightarrow$  i NO es té en compte el transport  $\rightarrow 1 0 0 0_{2C2}$   
 (número no vàlid perquè és igual a  $-8$ )

$-1 \rightarrow 1111 - 1 + 1 = 1 1 1 1_{2C2}$   
 $-2 \rightarrow 1111 - 10 + 1 = 1 1 1 0_{2C2}$   
 $-3 \rightarrow 1111 - 11 + 1 = 1 1 0 1_{2C2}$   
 $-4 \rightarrow 1111 - 100 + 1 = 1 1 0 0_{2C2}$   
 $-5 \rightarrow 1111 - 101 + 1 = 1 0 1 1_{2C2}$   
 $-6 \rightarrow 1111 - 110 + 1 = 1 0 1 0_{2C2}$

$-7 \rightarrow 1111 - 111 + 1 = 1 0 0 1_{2C2}$   
 $-8 \rightarrow 1111 - 1000 + 1 = 1 0 0 0_{2C2}$   
 $-9 \rightarrow 1111 - 1001 + 1 = 0 1 1 1_{2C2}$   
 (número no vàlid perquè és igual a 7)



# Enters amb signe (8/9)

## ■ Característiques del C2:

- ◆ El 1r bit dels nombres positius és un 0 i el dels negatius és un 1.
- ◆ Els nombres positius es representen igual, però omplits amb zeros a l'esquerra fins a arribar al total de bits de la representació.
- ◆ Els nombres negatius es poden obtenir ràpidament:
  - 1) Representar el número en binari com si fóra positiu.
  - 2) Omplir amb zeros fins a la quantitat de bits de la representació.
  - 3) Canviar els zeros per uns i els uns, per zeros.
  - 4) Sumar 1 al resultat.
- ◆ La suma i la resta esdevenen només una operació: SUMA
$$A - B = A + (-B)$$

- El C2 és quasi igual que el C1, excepte que sempre sumem 1 i es té compte el transport.

- **Exercici:** Feu l'exercici de la diapositiva 19, però ara en C2.



# Enters amb signe (8/9)

- **Exemple 1:** Supposem  $n = 4$ . Sumar  $A=5$  i  $B=2$ .

$$A = 5_{10} = 0101_{2C2}$$

$$B = 2_{10} = 0010_{2C2}$$

$$A+B = 0111_{2C2} = 7_{10}$$

$$\begin{array}{r} 0101 \\ + 0010 \\ \hline 0111 \end{array}$$

- **Exemple 2:** Supposem  $n = 4$ . Restar  $A=5$  i  $B=6 \equiv A + (-B)$ .

$$A = 5_{10} = 0101_{2C2}$$

$$-B = -6_{10} = -0110_2 = 1010_{2C2}$$

$$A+(-B) = 1111_{2C2} = 1110_{2C1} = -0001_2 = -1_{10}$$

$$\begin{array}{r} 0101 \\ + 1010 \\ \hline 1111 \end{array}$$

- **Exemple 3:** Supposem  $n = 4$ . Sumar  $A=5$  i  $B=6$ .

$$A = 5_{10} = 0101_{2C2}$$

$$B = 6_{10} = 0110_{2C2}$$

$$A+B = 1011_{2C2} = 1010_{2C1} = -0101_2 = -5_{10} \rightarrow \text{Overflow}$$

$$\begin{array}{r} 0101 \\ + 0110 \\ \hline 1011 \end{array}$$

- **Exemple 4:** Supposem  $n = 4$ . Sumar  $A=-5$  i  $B=-6$ .

$$A = -5_{10} = -0101_2 = 1011_{2C2}$$

$$B = -6_{10} = -0110_2 = 1010_{2C2}$$

$$A+B = 0101_{2C2} = 5_{10} \rightarrow \text{Underflow}$$

$$\begin{array}{r} 1011 \\ + 1010 \\ \hline (1)0101 \end{array}$$

El transport s'ignora!



- La representació de **caràcters alfanumèrics i numèrics** es fa mitjançant una correspondència entre els caràcters que volem representar i una sèrie de nombres binaris.
- La correspondència més coneguda i usada és el codi **ASCII** (*American Estàndard Code for Information Interchange*), originàriament utilitzat per a **comunicar** informació entre màquines diferents.
- A més dels caràcters normals utilitzats en l'escriptura, pot representar una sèrie de **caràcters especials** amb un cert significat per als ordinadors, també coneguts com a caràcters de control (Ex: *enter*, *escape*, etc.).
- ASCII utilitza un *byte* (8 bits) per a representar cada caràcter.



# Significat dels bits: caràcters (2/5)

- El **codi ASCII original** utilitzava set bits per a representar la informació, i el vuitè bit (bit de control d'errors) es feia servir per a comprovar la correcció dels altres set mitjançant la **paritat** dels bits (nombre parell o imparell d'uns del nombre binari).
  - ◆ **Exemple:** el nombre binari 0110101 té un nombre parell d'uns (4), per la qual cosa el bit de control serà 0; el nombre 0100101, però, tindrà com a bit de control un 1.
- Amb els set bits, es pot representar fins a un màxim de  $2^7 = 128$  caràcters.
- Actualment, gràcies a l'alta difusió del codi ASCII per a la representació de caràcters, s'han introduït caràcters internacionals nous: vocals accentuades, la 'ñ', la 'ç', etc. Aquesta **extensió del codi ASCII** empra el vuitè bit i és específica de cada país.





# Significat dels bits: caràcters (3/5)

Dec	Hex	Car	Dec	Hex	Car	Dec	Hex	Car	Dec	Hex	Car
0	00	NUL	32	20	SPC	64	40	@	96	60	`
1	01	SOH	33	21	!	65	41	A	97	61	a
2	02	STX	34	22	"	66	42	B	98	62	b
3	03	ETX	35	23	#	67	43	C	99	63	c
4	04	EOT	36	24	\$	68	44	D	100	64	d
5	05	ENQ	37	25	%	69	45	E	101	65	e
6	06	ACK	38	26	&	70	46	F	102	66	f
7	07	BEL	39	27	'	71	47	G	103	67	g
8	08	BS	40	28	(	72	48	H	104	68	h
9	09	HT	41	29	)	73	49	I	105	69	i
10	0A	LF	42	2A	*	74	4A	J	106	6A	j
11	0B	VT	43	2B	+	75	4B	K	107	6B	k
12	0C	FF	44	2C	,	76	4C	L	108	6C	l
13	0D	CR	45	2D	-	77	4D	M	109	6D	m
14	0E	SO	46	2E	.	78	4E	N	110	6E	n
15	0F	SI	47	2F	/	79	4F	O	111	6F	o
16	10	DLE	48	30	0	80	50	P	112	70	p
17	11	DC1	49	31	1	81	51	Q	113	71	q
18	12	DC2	50	32	2	82	52	R	114	72	r
19	13	DC3	51	33	3	83	53	S	115	73	s
20	14	DC4	52	34	4	84	54	T	116	74	t
21	15	NAK	53	35	5	85	55	U	117	75	u
22	16	SYN	54	36	6	86	56	V	118	76	v
23	17	ETB	55	37	7	87	57	W	119	77	w
24	18	CAN	56	38	8	88	58	X	120	78	x
25	19	EM	57	39	9	89	59	Y	121	79	y
26	1A	SUB	58	3A	:	90	5A	Z	122	7A	z
27	1B	ESC	59	3B	;	91	5B	[	123	7B	{
28	1C	FS	60	3C	<	92	5C	\	124	7C	
29	1D	GS	61	3D	=	93	5D	]	125	7D	}
30	1E	RS	62	3E	>	94	5E	^	126	7E	~
31	1F	US	63	3F	?	95	5F	_	127	7F	DEL



# Significat dels bits: caràcters (4/5)

- L'any 1991 apareix l'estàndard **UNICODE**.
- Unicode pretén codificar simultàniament els caràcters de múltiples idiomes.
- Unicode proveeix una codificació única per a fer referència a cada caràcter de cada llengua coberta.
- Actualment, cobreix els sistemes de símbols de moltes llengües, com l'àrab, el birmà o l'hebreu.
- **Suporta més de 100.000 caràcters únics!**



# Significat dels bits: caràcters (5/5)

- UNICODE deixa la tasca de representació a l'aplicació que ha de mostrar els caràcters (P. ex.: navegador web).
- Unicode només assigna un codi únic a cada símbol de cada llengua.
- Cal un mètode d'assignació entre els codis i els símbols, una norma de transició.
- Hi ha diversos **estàndards de mapatge** entre els codis i els símbols. Per exemple:
  - ◆ UTF-8, UTF-16, UTF-32
  - ◆ UCS-2, UCS-4



- Per a representar el nombres reals haurem de treballar amb una certa **precisió** (ja que no podem representar una quantitat infinita de decimals).
- En general, el reals es poden representar de dues maneres:
  - ◆ Mitjançant coma fixa.
  - ◆ Mitjançant coma flotant (o notació científica).



## ■ Representació en coma fixa:

- ◆ Per a passar de decimal a binari un nombre real, s'ha de codificar d'una banda la part **entera** i de l'altra, la part **decimal**:
  - $239,403 \rightarrow 239$  i  $403$ .
- ◆ La part **entera** es converteix de la manera com hem vist adés, és a dir, per mitjà de divisions successives.
- ◆ La part **decimal** es converteix per multiplicacions successives, de les quals cal agafar la part **entera** resultant de cada multiplicació.
- ◆ **Important:** Un real (ex:  $239,403$ ) amb un nombre finit de decimals (403 són 3 decimals) pot no tenir un nombre finit de decimals en la seua representació binària!

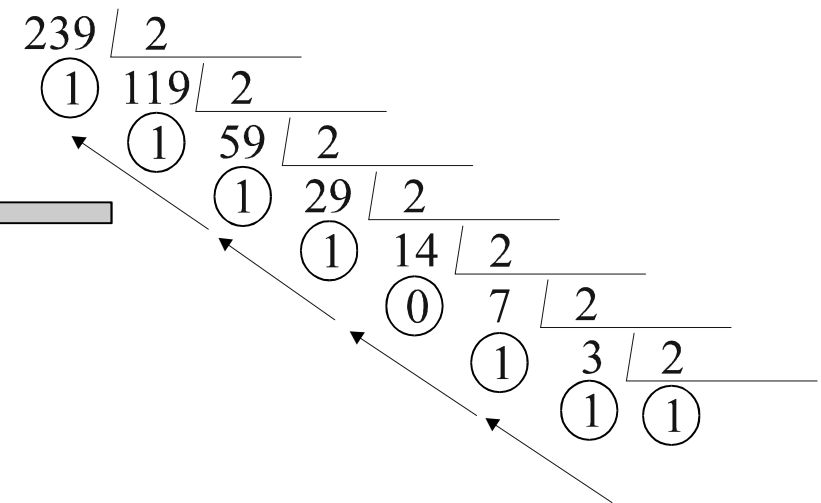


## ■ Representació en coma fixa (continuació):

### ■ Exemple: 239,403 a binari.

- ◆ De primer, codifiquem la part **entera** per divisions successives:

$$\rightarrow 239_{10} = 11101111_2$$



- ◆ Després, convertim la part decimal per mitjà de multiplicacions successives:

$$\rightarrow 0,403_{10} = 0,0110011_2$$

0,403 \* 2 = 0,806 (obtenim un 0)  
0,806 \* 2 = 1,612 (obtenim un 1)  
0,612 \* 2 = 1,224 (obtenim un 1)  
0,224 \* 2 = 0,448 (obtenim un 0)  
0,448 \* 2 = 0,896 (obtenim un 0)  
0,896 \* 2 = 1,792 (obtenim un 1)  
0,792 \* 2 = 1,584 (obtenim un 1)

- ◆ El **nombre en binari** serà:

$$\rightarrow 239,403_{10} = 11101111,0110011_2$$

- ◆ **Reflexions:** Quants decimals he d'obtenir? Com represente la coma a l'ordinador?



## ■ Representació en coma flotant:

- ◆ Consisteix a **col·locar la coma en una posició determinada**, de manera que la representació binària no se n'haurà de preocupar.

$$\rightarrow 239,403_{10} = 0,239403 * 10^3$$

- ◆ El fet de moure la coma suposarà l'aparició d'una base elevada a un exponent, que també s'ha de representar.
- ◆ **Qualsevol nombre real en qualsevol base** pot posar-se en coma flotant movent la coma a l'esquerra de la primera xifra significativa del nombre i multiplicant-lo per la base elevada a l'exponent adequat.

$$0,00000345_{10} = 0,345 * 10^{-5}$$

$$11101111,011001_2 = 0,11101111011001 * 2^8$$



## ■ Representació en coma flotant (continuació I):

- ◆ Una vegada tenim el nombre binari en coma flotant, la representació es farà separant, d'una banda, la representació del nombre que apareix després de la coma (o mantissa) i de l'exponent que eleva la base.
- ◆ La mantissa s'escriu representant els valors que apareixen a la dreta de la coma (ja que el 0 i la coma es mantenen fixos).
- ◆ Com en la representació d'enters, cal conèixer el nombre de bits que hem d'utilitzar per a la representació.
- ◆ La representació binària d'un nombre real en coma flotant sempre (o gairebé sempre) estarà representada per:

**signe / mantissa / exponent**





## ■ Representació en coma flotant (continuació II):

- ◆ La **representació del signe i la mantissa** és semblant a la dels nombres enters.
- ◆ Per tant, la manera habitual de representar-los és mitjançant **signe i magnitud**.
- ◆ **L'exponent** és un nombre enter, de manera que qualsevol mètode de representació d'enters servirà per a representar l'exponent.
- ◆ És habitual que, en la representació de nombres reals en coma flotant, l'exponent es represente amb **"biaix"** (és a dir, amb un desplaçament o excés). Així doncs, als nombres negatius els fem correspondre nombres positius mitjançant la transformació:

$$X' = X + 2^{n-1}$$

$$X \in [-(2^{n-1} - 1), 2^{n-1}]$$



## ■ Representació en coma flotant (continuació III):

- ◆ Exemple: Representació en coma flotant, amb 8 bits per a la mantissa (en signe-magnitud) i 5 bits per a l'exponent (“esbiaixat”), del nombre:

$$239,403_{10} = 11101111,0110011_2 = 0,11101111011011 * 2^8$$

Magnitud = 111011110110011 15 bits !!

Mantissa = Signe positiu (0) + 7bits (1110111) = 01110111

- ◆ Com que el signe i la magnitud ocupen més de 8 bits, cal **truncar** (pèrdua d'informació menys significativa.) Només representem realment el 0,1110111, en comptes de 0,111011110110011.

Exponent: Valor:  $8_{10} = 1000_2$

Biaix (5 bits per a l'exponent  $\rightarrow n = 5$ ):  $2^{n-1}_{10} = 16 = 10000_2$

Exponent amb biaix serà ( $\text{exp} = \text{exp} + 2^{n-1}$ ):

$$\text{exp} = 1000 + 10000 = 11000$$

- ◆ **Finalment:  $239,403_{10} = 01110111/11000$**



## ■ Operacions en coma flotant: Multiplicació

- ◆ Siguen A i B dos nombres representats en coma flotant com a:

$$\rightarrow A = mA * 2^{eA} \quad B = mB * 2^{eB}$$

- ◆ Per a multiplicar A i B, farem:

$$\rightarrow A * B = (mA * 2^{eA}) * (mB * 2^{eB}) = (mA * mB) * 2^{eA + eB}$$

- **Exemple:** Multiplicar els nombres 1,19 i 0,36 en coma flotant (8 bits per a la mantissa en signe/magnitud i 5 per a l'exponent esbiaixat).

$$A = 1,19_{10} = 1,0011_{2} = 0,1001100... * 2^1$$

$$B = 0,36_{10} = 0,01011100001_{2} = 0,1011100... * 2^{-1}$$

$$\text{Exponente1: Valor: } 1_{10} = 00001_{2}$$

$$\text{Exponente2: Valor: } -1_{10} = -00001_{2}$$

$$\text{Biaix (5 bits para el exponent } \rightarrow n = 5): 2^{n-1}_{10} = 2^4_{10} = 10000_{2}$$

Exponent amb biaix serà ( $\text{exp} = 2^{n-1} + \text{exp}$ ):

$$\text{exp1} = 10000 + 00001 = 10001$$

$$\text{exp2} = 10000 - 00001 = 01111$$



## ■ Multiplicació (continuació)

$$A = 1,19_{10} = 0,1001100 * 2^1 = \underline{0} 1 0 0 1 1 0 0 / 1 0 0 0 1$$

$$B = 0,36_{10} = 0,1011100 * 2^{-1} = \underline{0} 1 0 1 1 1 0 0 / 0 1 1 1 1$$

◆ Multipliquem les mantisses i ajustem la representació:

$$A * B = 0,0 1 1 0 1 1 0 1 0 1 * 2^1 * 2^{-1}$$

$$= (0,1 1 0 1 1 0 1 0 1 * 2^{-1}) * 2^1 * 2^{-1}$$

$$\begin{array}{r}
 0,100110 \\
 \times 0,101110 \\
 \hline
 000000 \\
 100110 \\
 100110 \\
 100110 \\
 000000 \\
 100110 \\
 \hline
 0,01101101010
 \end{array}$$

◆ Sumem els exponents:

$$A * B = 0,1 1 0 1 1 0 1 0 1 * 2^{-1} * 2^1 * 2^{-1} = 0,1 1 0 1 1 0 1 0 1 * 2^{-1+1-1}$$

$$= 0,1 1 0 1 1 0 1 0 1 * 2^{-1}$$

$$= 0 1 1 0 1 1 0 1 / 0 1 1 1 1 \quad \text{hi ha truncament!!!}$$



## ■ Operacions en coma flotant: Suma

- ◆ Siguen dos nombres A i B representats en coma flotant:

$$A = mA * 2^{eA} \quad B = mB * 2^{eB}$$

- ◆ Per a poder sumar A i B, haurem de traure'n els factors comuns i sumar la resta:

Si  $eA > eB$ , aleshores  $eB = eA - e_{aux}$

$$\begin{aligned} A + B &= (mA * 2^{eA}) + (mB * 2^{eB}) = (mA * 2^{eA}) + (mB * 2^{eA - e_{aux}}) \\ &= (mA + mB * 2^{-e_{aux}}) * 2^{eA} \end{aligned}$$

- ◆ **Exemple:** Sumar els nombres 1,19 i 0,36 en coma flotant (8 bits per a la mantissa en signe/magnitud i 5 per a l'exponent esbiaixat).

$$A = 1,19_{10} = 1,0011_{2} = 0,1001100... * 2^1$$

$$B = 0,36_{10} = 0,01011100001_{2} = 0,1011100... * 2^{-1}$$



## ■ Suma (continuació)

$$A = 1,19_{10} = 0,1001100 * 2^1 = \underline{0} 1 0 0 1 1 0 0 / 1 0 0 0 1$$

$$B = 0,36_{10} = 0,1011100 * 2^{-1} = \underline{0} 1 0 1 1 1 0 0 / 0 1 1 1 1$$

◆ Como  $\text{exp}A > \text{exp}B \rightarrow \text{exp}B = \text{exp}A - \text{aux} = 1 - (2) = -1$

$$B = 0,1011100 * 2^{-1} = 0,1011100 * 2^1 * (2^{-1} * 2^{-1})$$

$$= 0,001011100 * 2^1 = 0,0010111 * 2^1$$

◆ Sumem les mantisses i ajustem la representació:

$$\begin{array}{r} 0,1001100 \\ + 0,0010111 \\ \hline 0,1100011 \end{array} + B = 0,1100011 * 2^1$$

a coma ara està ben col·locada  
no és necessari ajustar.

◆ Representem l'exponent comú i queda:

$$A + B = \underline{0} 1 1 0 0 0 1 1 / 1 0 0 0 1 \quad \text{També hi ha truncament!!!}$$



- Representació binària de la informació.
  - ◆ Definició.
  - ◆ Agrupament dels bits.
  - ◆ Unitats de mesura binàries.
- Informació bàsica representada.
  - ◆ Informació booleana.
  - ◆ Conversió decimal – binari.
  - ◆ Operacions en binari.
  - ◆ Enters amb signe.
  - ◆ Caràcters.
  - ◆ Reals.
- Representació intermèdia: octal i hexadecimal.
  - ◆ Definició.
  - ◆ Conversió decimal – hexadecimal/octal – binari.
- Exercicis.



# Representació intermèdia (1/3)

- Com que en base binària els únics dígit que poden utilitzar-se són el 0 i l'1, normalment és molest i llarg escriure un nombre en base binària, encara que aquesta siga l'única base que realment utilitza l'ordinador.
- Per a escriure nombres fàcilment convertibles a binari, però amb menor nombre de xifres, s'utilitzen dos tipus de codificacions intermèdies: la **base octal** i la **base hexadecimal**.
- En la representació octal, la **base és huit** i l'alfabet està compost pels dígit del 0 al 7.

$$B = 8$$

$$A = \{0, 1, 2, 3, 4, 5, 6, 7\}$$





# Representació intermèdia (2/3)

- En la representació hexadecimal, la **base és setze** i l'alfabet, que conté setze caràcters, està format pels dígitos des del 0 fins al 9 (deu caràcters) més les lletres (generalment majúscules) des de la A fins a la F (6 caràcters).

$$B = 16$$

$$A = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F\}$$

- Per **exemple**, aquesta representació es fa servir per a definir els colors en les eines de dibuix.



# Representació intermèdia (2/2)

Representació RGB: **R**ed, **G**reen, **B**lue

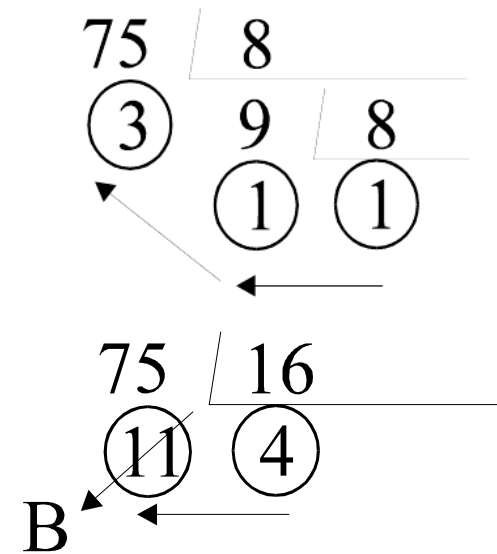


Representació decimal.

Representació hexadecimal.

# Conversió decimal - octal/hexadecimal

- La manera de passar de base **decimal a octal o hexadecimal** és semblant al pas a base binària, però dividint successivament per huit o setze.



- Per exemple:  $75_{10} = 113_8 = 4B_{16}$
- Per canviar d'**octal/hexadecimal a decimal**, aplicarem la fórmula de descomposició que ja coneixem:
  - ♦ Octal = suma de potències de 8.
  - ♦ Hexadecimal = suma de potències de 16.
- Per exemple, per a passar  $4B_{16}$  i  $113_8$  a base decimal:  
 $113_8 = 3 \cdot 8^0 + 1 \cdot 8^1 + 1 \cdot 8^2 = 3 \cdot 1 + 1 \cdot 8 + 1 \cdot 64 = 3 + 8 + 64 = 75$   
 $4B_{16} = B \cdot 16^0 + 4 \cdot 16^1 = 11 \cdot 1 + 4 \cdot 16 = 11 + 64 = 75$



# Conversió binari – octal

- La representació octal està basada en el 8, mentre que la binària està basada en el 2.
- Com que  $8 = 2^3$ , podem assumir que **tres xifres binàries fan una xifra octal**.
- Així, el pas de binari a hexadecimal es redueix a **agrupar de tres en tres**, de dreta a esquerra, les xifres binàries i a avaluar-ne el valor decimal. Cal recordar que hem d'omplir amb zeros a l'esquerra fins a completar els grups de 3.

$$\begin{aligned} & \mathbf{1001011}_2 = 001\ 001\ 011_2 \\ & 001_2 = 1_8 \\ & 001_2 = 1_8 \\ & 011_2 = 3_8 \\ & = \mathbf{113}_8 \end{aligned}$$



# Conversió binari – hexadecimal

- La representació hexadecimal està basada en el 16, mentre que la binària està basada en el 2.
- Com que  $16 = 2^4$ , podem assumir que **quatre xifres binàries fan una xifra hexadecimal**.
- Així, el pas de binari a hexadecimal es redueix a **agrupar de quatre en quatre**, de dreta a esquerra, les xifres binàries i a avaluar-ne el valor decimal; cal recordar que valors superiors a 9 són representats per mitjà de lletres (**A=10, B=11, C=12, D=13, E=14 i F=15**).

$$\begin{aligned} 1001011_2 &= 0100\ 1011_2 \\ 0100_2 &= 4_{16} \\ 1011_2 &= 11 = B_{16} \\ &= \mathbf{4B}_{16} \end{aligned}$$

# Conversió octal/hexadecimal - binari

- Una vegada vist el pas de binari a hexadecimal, la transformació inversa és òbvia.
- En el cas octal només cal passar cadascun dels dígit octals a tres xifres binàries i en el cas hexadecimal, a quatre xifres binàries.

$$113_8 = 001\ 001\ 011_2$$

$$1_8 = 001_2$$

$$3_8 = 011_2$$

$$4B_{16} = 0100\ 1011_2$$

$$4_{16} = 0100_2$$

$$B_{16} = 11 = 1011_2$$

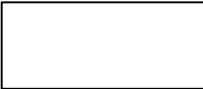
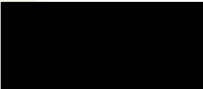


Bin	Hex	Dec
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	A	10
1011	B	11
1100	C	12
1101	D	13
1110	E	14
1111	F	15



- Passeu a **binari** els nombres i les lletres següents:

- ♦  $-87_{10} \rightarrow ?_{2SM}$
- ♦  $01110000_2 \rightarrow ?_{16} \rightarrow ?_{ASCII-8}$
- ♦  $T_{ASCII-8} \rightarrow ?_{16} \rightarrow ?_2$
- ♦  $166,386_{10} \rightarrow ?_2$

- Calculeu la representació **hexadecimal o decimal** dels colors:



- ♦  Blanc : R(?), G(?), B(?)
- ♦  Negre : R(?), G(?), B(?)
- ♦  Gris :  $969696_{16} \rightarrow R(?), G(?), B(?)$
- ♦  Turq. :  $22e0bf_{16} \rightarrow R(?), G(?), B(?)$



■ Passeu a **binari** els nombres i les lletres següents:

- ♦  $+200_{10} \rightarrow ?_{2SM} \text{ (9 bits)}$
- ♦  $-200_{10} \rightarrow ?_{2C1} \text{ (9 bits)}$
- ♦  $A = -45_{10} \rightarrow ?_{2C2} \text{ (8 bits)}$
- ♦  $B = +86_{10} \rightarrow ?_{2C2} \text{ (8 bits)}$
- ♦  $A + B \rightarrow ?_{2C2} \text{ (8 bits)}$
- ♦  $B - A \rightarrow ?_{2C2} \text{ (8 bits)}$
- ♦  $1100110_{2SM} \rightarrow ?_{10}$
- ♦  $I\&D_{ASCII-8} \rightarrow 49_{16} 26_{16} ? 44_{16} \rightarrow ?_2$

■ Calculeu la representació **hexadecimal o decimal** dels colors:

- ♦  Groc :  $eaf800_{16} \rightarrow R(?), G(?), B(?)$
- ♦  Verd :  $?_{16} \rightarrow R(15), G(160), B(17)$





- Donats  $C = 45'75_{10}$  i  $D = -86'8_{10}$ , representeu en coma flotant, amb 8 bits per a la mantissa (en signe i magnitud) i 5 bits per a l'exponent (esbiaixat).
- Calculeu el resultat de sumar i de multiplicar  $C$  i  $D$  a partir de les representacions expressades abans i transcriviu el resultat en base decimal.
- Indiqueu quin valor hauríem d'obtenir de la suma de  $C$  i  $D$ . Si no s'ha obtingut el valor esperat, digueu quins errors s'han comès, tant en les representacions com en les operacions en coma flotant, i quan s'han produït els errors anteriors.