MÁSTER UNIVERSITARIO EN SISTEMAS ESPACIALES
POR LA UNIVERSIDAD POLITÉCNICA DE MADRID

POLITÉCNICA

2021.02.08

Representación binaria de la información

Juan Zamorano jzamora@datsi.fi.upm.es
Juan Antonio de la Puente juan.de.la.puente@upm.es>
Alejandro Alonso alejandro.alonso@upm.es

FTel
STRAST

Conceptos básicos

- En un computador toda la información se presenta en forma digital
  - Unidad básica: bit (*binary digit*): 0 ó 1
  - Unidad de almacenamiento: byte (u octeto) = 8 bits
  - Capacidad de almacenamiento: se mide en bytes
    - los multiplicadores son potencias de 2 (a veces de 10)
    - ejemplo: 64 KiB, 32 MiB, 128 GiB

64 KiB = 64 kB
32 MiB = 32 MB
128 GiB = 128 GB

Binarios (IEC)		Decimales (SI)	
Valor	**Prefijo**	**Valor**	**Prefijo**
2^{10}	kibi (Ki)	10^{3}	kilo (k)
2^{20}	mebi (Mi)	10^{6}	mega (M)
2^{30}	gibi (Gi)	10^{9}	giga (G)
2^{40}	tebi (Ti)	10^{12}	tera (T)
2^{50}	pebi (Pi)	10^{15}	peta (P)
2^{60}	exbi (Ei)	10^{18}	exa (E)
2^{70}	zebi (Zi)	10^{21}	zetta (Z)
2^{80}	yobi (Yi)	10^{24}	yotta (Y)

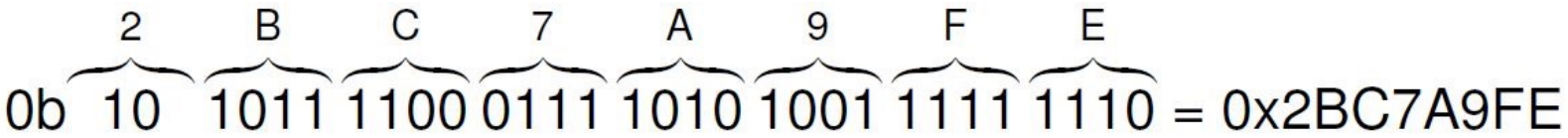
Notación hexadecimal

Los contenidos binarios largos

10101111000111101010011111111110

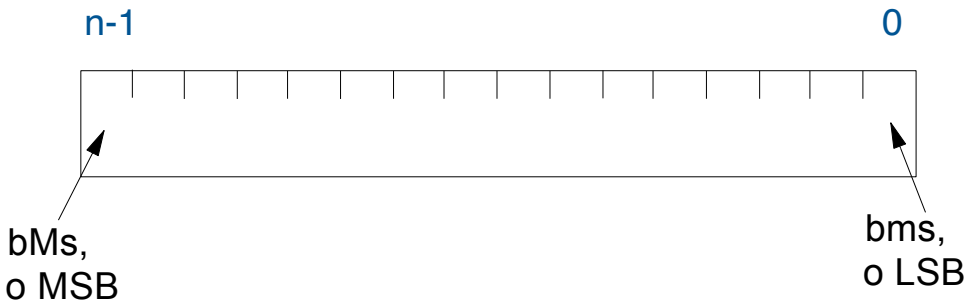
se ven mejor agrupando los bits de cuatro en cuatro:

Binario	Hexadecimal	Binario	Hexadecimal
0000	0	1000	8
0001	1	1001	9
0010	2	1010	A
0011	3	1011	B
0100	4	1100	C
0101	5	1101	D
0110	6	1110	E
0111	7	1111	F



Representación de números en binario

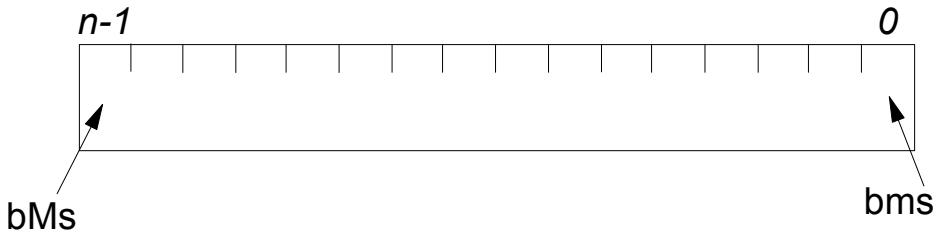
- **Precisión arbitraria:** los bits que sean necesarios
  - operaciones aritméticas con grandes números («bignum arithmetic»)
- **Precisión limitada:** formatos con un número fijo de bits
  - representación de enteros y racionales, con una **precisión** (número de bits) y un **rango** (números máximo y mínimo) fijos:



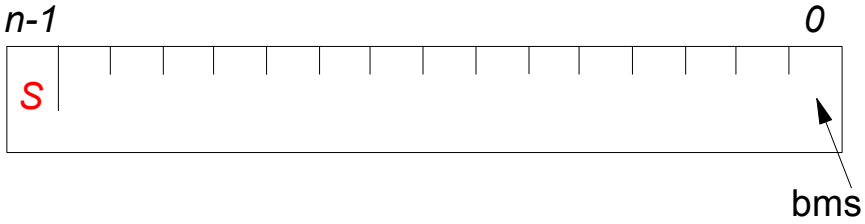
Representación de números enteros

- Se suele utilizar un formato de **coma fija** (la «coma» se supone situada inmediatamente a la derecha del bms)

- Números **sin signo**
  - Enteros no negativos
  - Máximo: $2^n - 1$ (1111…11)
  - Mínimo: **0** (0000…00)



- Números **con signo**:
  - bMs se usa como bit de signo: **S** (S = 0: positivo, S = 1: negativo)
  - Máximo: $2^{n-1} - 1$ (**0**111…11)
  - Mínimo: según convenio usado para los números negativos



Convenios para representar enteros negativos

- Signo y magnitud: Obvio, pero por motivos de diseño de circuitos para sumar y restar se prefieren:

- Complemento a 1: **rep(N) + rep(−N) = $2^n - 1$**
  - A efectos prácticos, basta cambiar los 0 por 1 y 1 por 0 en rep(N)
  - Mínimo representable: 100...0 = rep($-2^{n-1} + 1$)
  - Dos representaciones para «cero»: +0 (0...0) y −0 (1...1)

- Complemento a 2 (más frecuente): **rep(N) + rep(−N) = 2^n**
  - A efectos prácticos, se hace el complemento a 1 y luego se le suma una unidad
  - Mínimo representable: 100...0 = rep(-2^{n-1})
  - Un solo «cero» y un negativo más.

Nota: rep(N) = representación de N

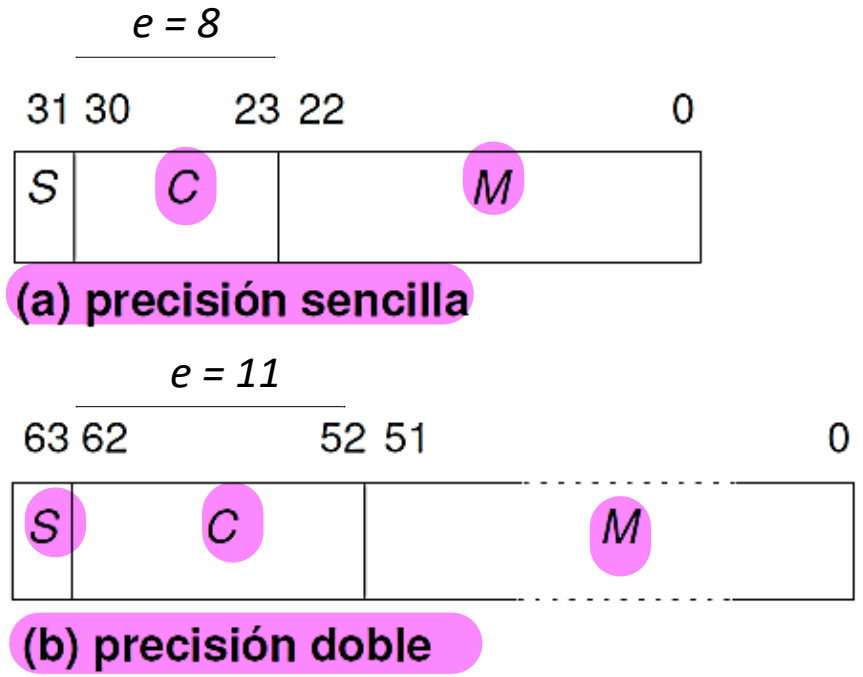
Para $n = 8$ (8 bits) en Complemento a uno

Valores de 8 bits	Interpretado en Complemento a uno en decimal	Interpretado como Entero sin signo en decimal
00000000	0	0
00000001	1	1
00000010	2	2
...	...	...
01111110	126	126
01111111	127	127
10000000	–127	128
10000001	–126	129
10000010	–125	130
...	...	...
11111101	–2	253
11111110	–1	254
11111111	–0	255

Para $n = 8$ (8 bits) en Complemento a dos

Valores de 8 bits	Interpretado en Complemento a dos en decimal	Interpretado como Entero sin signo en decimal
00000000	0	0
00000001	1	1
00000010	2	2
...	...	...
01111110	126	126
01111111	127	127
10000000	–128	128
10000001	–127	129
10000010	–126	130
...	...	...
11111101	–3	253
11111110	–2	254
11111111	–1	255

Formatos de coma flotante: la norma IEEE 754



Número representado:

$$N = \pm 1, M \times 2^{E}$$

- *S (signo): convenio de* signo y magnitud
- *M (mantisa)*: normalización fraccionaria, omitiendo el bMs (= 1)
- *E (exponente),* con exceso de $2^{e-1} - 1$: $E = C - 2^{e-1} + 1$
  - siendo ***e*** el número de bits de la característica (C)

prec simple: e = 8, prec. doble: e=11

  - es decir, *E = C – 127 (prec. simple), E = C-1023 (prec. doble)*

Ejemplo de representación en IEEE 754

Representación de $(-1983{,}78125)_{(10)}$ en el formato de precisión sencilla

1) Conversión a binario:

$(1983{,}78125)_{(10)} = (7BF{,}C8)_{(16)} = (0111\ 1011\ 1111{,}1100\ 1000)_{(2)}$

2) Normalización: $0111\ 1011\ 1111{,}1100\ 1000 = 1{,}11\ 1011\ 1111\ 1100\ 1000 \times 2^{10}$

3) Ya sabemos:

- Signo: $N = 1$
- Mantisa: $M =$ 11101111111001000...
- Exponente: $E = 10$

Falta calcular la característica, C

4) $E = C - 2^{e-1} + 1 = C - 2^7 + 1 = C - 127$

Luego $C = 10 + 127 = 137 = (89)_{(16)} = (10001001)_{(2)}$

Resultado:

31	30 ... 23	22 ... 0
1	100 0100 1	111 0111 1111 1001 0000 0000

(0xC4F7F900)

Ejercicio: ¿Qué número es el representado por 0x44FB8000?

¿Cómo se almacena en la memoria en los bytes d a d + 3?

Extremismo (Endianness)

- En almacenamiento:

Si un dato codificado en *k* bytes se ha de almacenar en una sucesión de *k* direcciones de una RAM (*d* , *d* + 1...*d* + *k* – 1),
¿en qué orden se hace?

- Convenio extremista menor (little-endian):
  el byte menos significativo en *d* , el siguiente en *d* + 1...
- Convenio extremista mayor (big-endian):
  el byte más significativo en *d* , el siguiente en *d* + 1...

Codificación de caracteres: ASCII

«American Standard Code for Information Interchange», ≈ 1960

Estándares ISO/IEC 646 y Ecma-6

Código de 7 bits (2^7 = 128 codificaciones)

- 0000000 a 0011111 (0x00 a 0x1F):
  32 caracteres de control («Conjunto C0», estándar ISO/IEC 6429):

  0x00, NUL: carácter nulo (fin de cadena)

  0x0A, LF (line feed): nueva línea

  0x0D, CR (carriage return): retorno 0x1B, ESC (escape)

  ...

- 0100000 a 1111110 (0x20 a 0x7E):
  95 caracteres imprimibles(incluido el espacio, 0x20):

- 1111111 (0x7F): DEL (delete): borrar

Codificaciones ASCII de caracteres imprimibles

Hex.	Dec.		Hex.	Dec.		Hex.	Dec.		Hex.	Dec.		Hex.	Dec.		Hex.	Dec.	
20	032		30	048	0	40	064	@	50	080	P	60	096	‘	70	112	p
21	033	!	31	049	1	41	065	A	51	081	Q	61	097	a	71	113	q
22	034	"	32	050	2	42	066	B	52	082	R	62	098	b	72	114	r
23	035	#	33	051	3	43	067	C	53	083	S	63	099	c	73	115	s
24	036	$	34	052	4	44	068	D	54	084	T	64	100	d	74	116	t
25	037	%	35	053	5	45	069	E	55	085	U	65	101	e	75	117	u
26	038	&	36	054	6	46	070	F	56	086	V	66	102	f	76	118	v
27	039	’	37	055	7	47	071	G	57	087	W	67	103	g	77	119	w
28	040	(	38	056	8	48	072	H	58	088	X	68	104	h	78	120	x
29	041	)	39	057	9	49	073	I	59	089	Y	69	105	i	79	121	y
2A	042	*	3A	058	:	4A	074	J	5A	090	Z	6A	106	j	7A	122	z
2B	043	+	3B	059	;	4B	075	K	5B	091	[	6B	107	k	7B	123	{
2C	044	,	3C	060	<	4C	076	L	5C	092	\	6C	108	l	7C	124	|
2D	045	-	3D	061	=	4D	077	M	5D	093	]	6D	109	m	7D	125	}
2E	046	.	3E	062	>	4E	078	N	5E	094	^	6E	110	n	7E	126	~
2F	047	/	3F	063	?	4F	079	O	5F	095	_	6F	111	o	7F	127	<d>

Otros códigos de caracteres

La mayoría, extensiones a 8 bits:

- Windows-1252 («occidental»), Windows-1251 («cirílico»)...
- MacOS Roman, MacOS Arabic... IBM CP 850, CP 858...
- EBCDIC, incompatible con ASCII. Utilizado en «mainframes».
- GSM 03.38: código de 7 bits para el SMS de telefonía móvil
- ...
- Estándar ISO/IEC 8859 (1985-2001):

  16 «partes» (códigos) que comparten las codificaciones ASCII
  - ISO 8859-1 (o «Latin-1»), para europa occidental
  - ...
  - **ISO 8859-15** (o «Latin-9»), revisión de 8859-1: introduce € y otros caracteres → la parte más comúnmente utilizada

...ISO 8859-15...

0x00 a 0x7F ≡ ASCII
0x80 a 0x9F: caracteres de control («Conjunto C1»)

Hex.	Dec.		Hex.	Dec.		Hex.	Dec.		Hex.	Dec.		Hex.	Dec.		Hex.	Dec.	
A0	160	NBSP	B0	176	°	C0	192	À	D0	208	Đ	E0	224	à	F0	240	ð
A1	161	¡	B1	177	±	C1	193	Á	D1	209	Ñ	E1	225	á	F1	241	ñ
A2	162	¢	B2	178	²	C2	194	Â	D2	210	Ò	E2	226	â	F2	242	ò
A3	163	£	B3	179	³	C3	195	Ã	D3	211	Ó	E3	227	ã	F3	243	ó
A4	164	€	B4	180	Ž	C4	196	Ä	D4	212	Ô	E4	228	ä	F4	244	ô
A5	165	¥	B5	181	µ	C5	197	Å	D5	213	Õ	E5	229	å	F5	245	õ
A6	166	Š	B6	182	¶	C6	198	Æ	D6	214	Ö	E6	230	æ	F6	246	ö
A7	167	§	B7	183	·	C7	199	Ç	D7	215	×	E7	231	ç	F7	247	÷
A8	168	š	B8	184	ž	C8	200	È	D8	216	Ø	E8	232	è	F8	248	ø
A9	169	©	B9	185	¹	C9	201	É	D9	217	Ù	E9	233	é	F9	249	ù
AA	170	ª	BA	186	º	CA	202	Ê	DA	218	Ú	EA	234	ê	FA	250	ú
AB	171	«	BB	187	»	CB	203	Ë	DB	219	Û	EB	235	ë	FB	251	û
AC	172	¬	BC	188	Œ	CC	204	Ì	DC	220	Ü	EC	236	ì	FC	252	ü
AD	173	-	BD	189	œ	CD	205	Í	DD	221	Ý	ED	237	í	FD	253	ý
AE	174	®	BE	190	Ÿ	CE	206	Î	DE	222	Þ	EE	238	î	FE	254	þ
AF	175	¯	BF	191	¿	CF	207	Ï	DF	223	ß	EF	239	ï	FF	255	ÿ

Unicode

Código universal para todas las lenguas. ISO/IEC 10646
Importante para la internacionalización del software (i18n)

Define puntos de código: números naturales asociados a los distintos caracteres.

- Unicode 1.1 (1991): Plano básico multilingüe (BMP)

  2^{16} = 65.536 puntos de código (U+0000 a U+FFFF)

- Unicode 6.2 (2012): 17 planos ~ 17 × 2^{16} = 1.114.112
  (110.182 caracteres definidos)

Se puede materializar mediante varias formas de codificación:

- UCS-2 (dos bytes, sólo el BMP) y UTF-16 (dos o cuatro bytes)
- UCS-4: Codifica todos los puntos de código en cuatro bytes
- UTF-8: Actualmente, la forma más usada

UTF-8

- A diferencia de otras formas, es compatible con ASCII:
  los primeros 128 puntos de código se codifican en un solo byte.
- Código de longitud variable: los puntos del BMP mayores que U+007F se codifican con dos o tres bytes. Los otros planos requieren hasta seis bytes.
- Ejemplos:

Carácter	Punto de código	Codificación UTF-8
E	U+0045	0x45 (un byte)
ñ	U+00F1	0xC3 0xB1 (dos bytes)
€	U+20AC	0xE2 0x82 0xAC (tres bytes)
Pts	U+20A7	0xE2 0x82 0xA7 (tres bytes)

Resumen

- Unidades digitales: bit, byte
  - múltiplos KiB, MiB, GiB / KB, MB, GB
- Representación binaria
  - notación hexadecimal
- Codificación de números
  - enteros
    - negativos en complemento a 2
  - reales
    - signo, exponente y mantisa
    - IEEE 754
- Codificación de caracteres
  - ASCII
  - ISO-8859-15
  - Unicode / UTF-8