Sistemas de numeración

- Un número “N” se representa en base “b” como “…$d_7\, d_6\, d_5\, d_4\, d_3\, d_2\, d_1\, d_0$”, donde “$d_i$” son los dígitos.

$$N = \ldots d_7 b^7 + d_6 b^6 + d_5 b^5 + d_4 b^4 + d_3 b^3 + d_2 b^2 + d_1 b^1 + d_0$$

- d_0 LSB o bit menos significativo
- d_3 MSB o bit más significativo

Table 2-1
Binary, decimal, octal, and hexadecimal numbers.

Binary	Decimal	Octal	3-Bit String	Hexadecimal	4-Bit String
0	0	0	000	0	0000
1	1	1	001	1	0001
10	2	2	010	2	0010
11	3	3	011	3	0011
100	4	4	100	4	0100
101	5	5	101	5	0101
110	6	6	110	6	0110
111	7	7	111	7	0111
1000	8	10	—	8	1000
1001	9	11	—	9	1001
1010	10	12	—	A	1010
1011	11	13	—	B	1011
1100	12	14	—	C	1100
1101	13	15	—	D	1101
1110	14	16	—	E	1110
1111	15	17	—	F	1111

Conversión entre las bases más comunes

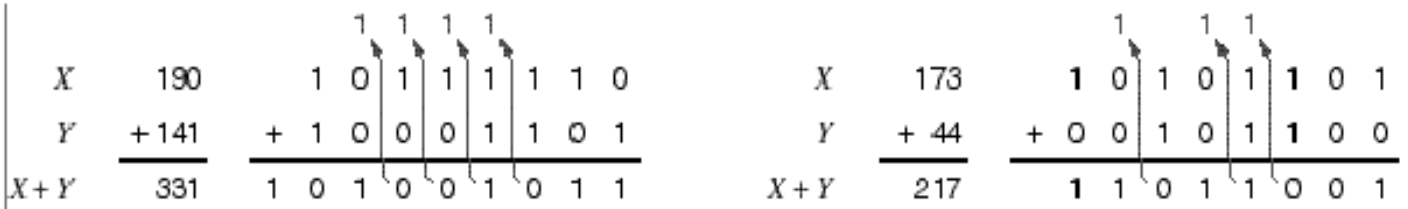
Table 2-2 Conversion methods for common radices.

Conversion	Method	Example
Binary to		
Octal	Substitution	$101110011001_2 = 10\ 111\ 011\ 001_2 = 2731_8$
Hexadecimal	Substitution	$101110011001_2 = 101\ 1101\ 1001_2 = 5D9_{16}$
Decimal	Summation	$101110011001_2 = 1 \cdot 1024 + 0 \cdot 512 + 1 \cdot 256 + 1 \cdot 128 + 1 \cdot 64 + 0 \cdot 32 + 1 \cdot 16 + 1 \cdot 8 + 0 \cdot 4 + 0 \cdot 2 + 1 \cdot 1 = 1497_{10}$
Octal to		
Binary	Substitution	$1234_8 = 001\ 010\ 011\ 100_2$
Hexadecimal	Substitution	$1234_8 = 001\ 010\ 011\ 100_2 = 0010\ 1001\ 1100_2 = 29C_{16}$
Decimal	Summation	$1234_8 = 1 \cdot 512 + 2 \cdot 64 + 3 \cdot 8 + 4 \cdot 1 = 668_{10}$
Hexadecimal to		
Binary	Substitution	$C0DE_{16} = 1100\ 0000\ 1101\ 1110_2$
Octal	Substitution	$C0DE_{16} = 1100\ 0000\ 1101\ 1110_2 = 1\ 100\ 000\ 011\ 011\ 110_2 = 140336_8$
Decimal	Summation	$C0DE_{16} = 12 \cdot 4096 + 0 \cdot 256 + 13 \cdot 16 + 14 \cdot 1 = 49374_{10}$
Decimal to		
Binary	Division	$108_{10} \div 2 = 54$ remainder 0 (LSB) $\div 2 = 27$ remainder 0 $\div 2 = 13$ remainder 1 $\div 2 = 6$ remainder 1 $\div 2 = 3$ remainder 0 $\div 2 = 1$ remainder 1 $\div 2 = 0$ remainder 1 (MSB) $108_{10} = 1101100_2$
Octal	Division	$108_{10} \div 8 = 13$ remainder 4 (least significant digit) $\div 8 = 1$ remainder 5 $\div 8 = 0$ remainder 1 (most significant digit) $108_{10} = 154_8$
Hexadecimal	Division	$108_{10} \div 16 = 6$ remainder 12 (least significant digit) $\div 16 = 0$ remainder 6 (most significant digit) $108_{10} = 6C_{16}$

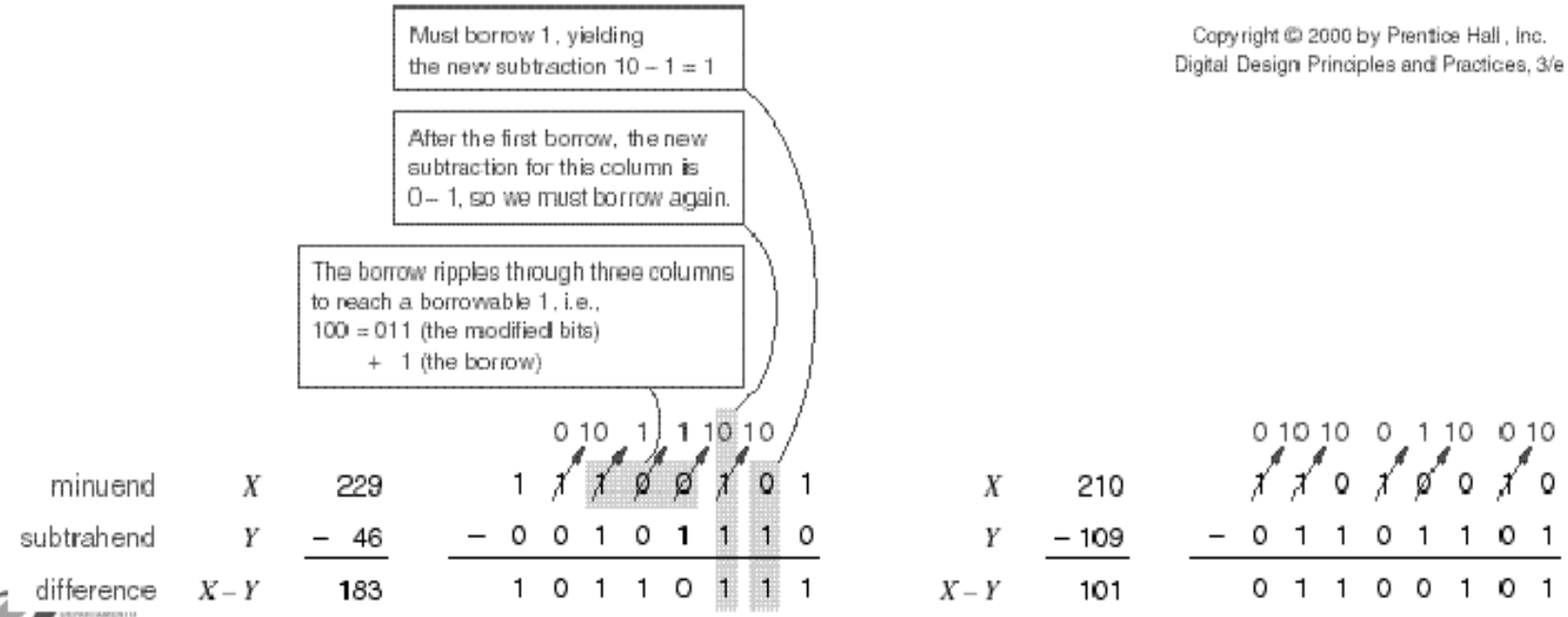
DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA
UNIVERSIDAD POLITÉCNICA DE MADRID

Suma y resta binarios

- Suma

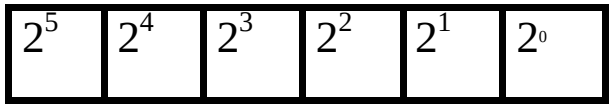


- Resta (poco utilizada)

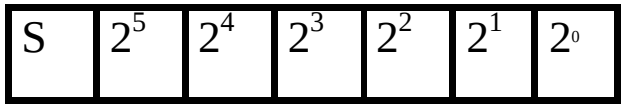


REPRESENTACIÓN DE NÚMEROS ENTEROS

- Binario puro (sin signo)



- Módulo y signo



- Complemento a 1:
  - Los números positivos igual que en binario puro (MSB =0)
  - Los números negativos invierten todos sus bits (MSB = 1)
- Complemento a 2:
  - Se forma el complemento a 1 del número
  - Se le suma 1
  - Tambien va a coincidir que el MSB es 0 para los números positivos y 1 para los negativos
- Exceso Z
  - Se obtiene sumando Z al número en binario puro
  - El código exceso 2^(n-1) es igual que el complemento a 2 pero invirtiendo el MSB

Ejemplo de representación

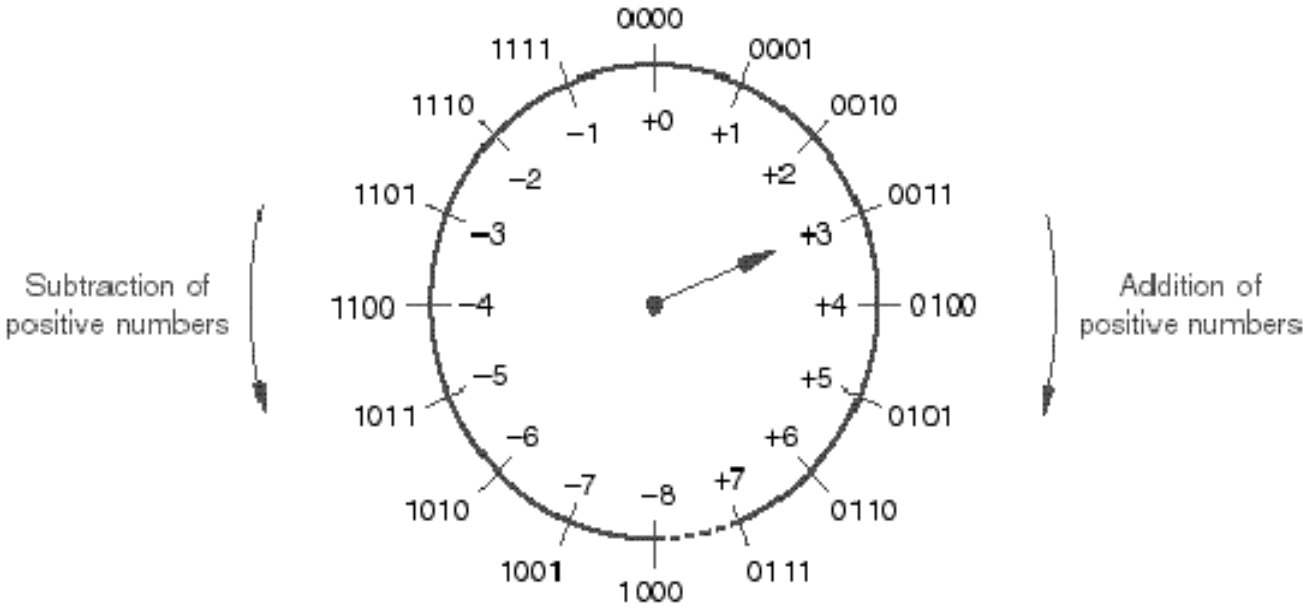
- Ejemplo de varias representaciones con cuatro bits

Table 2-6 Decimal and 4-bit numbers.

Decimal	*Two's Complement*	*Ones' Complement*	*Signed Magnitude*	*Excess* 2^{n-1}
−8	1000	—	—	0000
−7	1001	1000	1111	0001
−6	1010	1001	1110	0010
−5	1011	1010	1101	0011
−4	1100	1011	1100	0100
−3	1101	1100	1011	0101
−2	1110	1101	1010	0110
−1	1111	1110	1001	0111
0	0000	1111 or 0000	1000 or 0000	1000
1	0001	0001	0001	1001
2	0010	0010	0010	1010
3	0011	0011	0011	1011
4	0100	0100	0100	1100
5	0101	0101	0101	1101
6	0110	0110	0110	1110
7	0111	0111	0111	1111

Suma y resta en complemento a dos

- Suma normal, no diferenciándose los negativos
- Resta, como una suma pero complementando a dos el sustraendo x-y = x+ (-y)
- En ambos casos hay que tener cuidado con el desbordamiento u “overflow”



Multiplicación binaria

- El producto S tiene un número de bits igual a la suma de los del multiplicando “a” y el multiplicador “b”. Ejemplo $S(S_7..S_0) = A(A_3..A_0)*B(B_3..B_0)$

				a_3	a_2	a_1	a_0
			*	b_3	b_2	b_1	b_0
				b_0a_3	b_0a_2	b_0a_1	b_0a_0
			b_1a_3	b_1a_2	b_1a_1	b_1a_0	
		b_2a_3	b_2a_2	b_2a_1	b_2a_0		
	b_3a_3	b_3a_2	b_3a_1	b_3a_0			
S_7	S_6	S_5	S_4	S_3	S_2	S_1	S_0

Códigos para números decimales

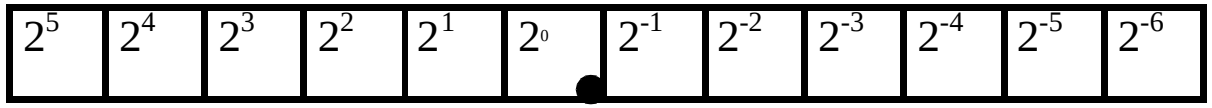
- Los más utilizados son BCD y 1-de-10 (para excitación de dispositivos externos como relés o LED)

Table 2-9 Decimal codes.

Decimal digit	BCD (8421)	2421	Excess-3	Biquinary	1-out-of-10
0	0000	0000	0011	0100001	1000000000
1	0001	0001	0100	0100010	0100000000
2	0010	0010	0101	0100100	0010000000
3	0011	0011	0110	0101000	0001000000
4	0100	0100	0111	0110000	0000100000
5	0101	1011	1000	1000001	0000010000
6	0110	1100	1001	1000010	0000001000
7	0111	1101	1010	1000100	0000000100
8	1000	1110	1011	1001000	0000000010
9	1001	1111	1100	1010000	0000000001
			Unused code words		
	1010	0101	0000	0000000	0000000000
	1011	0110	0001	0000001	0000000011
	1100	0111	0010	0000010	0000000101
	1101	1000	1101	0000011	0000000110
	1110	1001	1110	0000101	0000000111
	1111	1010	1111	...	...

REPRESENTACIÓN DE NÚMEROS DECIMALES EN COMA FIJA

- Formato:



Se puede utilizar:

- Representación en binario puro (sin signo)
- Representación en módulo y signo
- Representación en complemento a 1
- Representación en complemento a 2

- Problemas:

La suma de dos números puede provocar sobrepasamiento (overflow), al poder llegar hasta n+1 bits de longitud

La multiplicación de dos números puede llegar hasta 2n bits de longitud

Si se quieren hacer cálculos con la máxima precisión es necesario ir cambiando la coma de posición en cada operación

No permite trabajar a la vez con números muy grandes y muy pequeños

REPRESENTACIÓN DE DECIMALES EN COMA FLOTANTE

Formato: $n^o = mantisa \cdot base^{exponente}$

Mantisa fraccionaria normalizada, sin ceros a la derecha Se representa sin signo y sin el primer bit que es siempre igual a 1.

La base utilizada es 2

El exponente se representa en exceso 127 para simple precisión y exceso 1023 para doble precisión

El cero se representa por E=0, M=0

Cuando el resultado no tiene sentido (error) se representapor E=255, M=0

Para representar números pequeños (cercanos a 0) se utiliza un formato no normalizado. Se indica con E=0, M distinto de 0. En ese caso el exponente será -126 y la mantisa será 0,M en vez de 1,M

Tabla de códigos ASCII

CÓDIGO		CARACTER
0	0h	NULL
1	1h	SOH
2	2h	STX
3	3h	ETX
4	4h	EOT
5	5h	ENQ
6	6h	ACK
7	7h	BEL
8	8h	BS
9	9h	HT
10	Ah	LF
11	Bh	VT
12	Ch	FF
13	Dh	CR
14	Eh	S0
15	Fh	S1
16	10h	DLE
17	11h	DC1
18	12h	DC2
19	13h	DC3
20	14h	DC4
21	15h	NAK
22	16h	SYN
23	17h	ETB
24	18h	CAN
25	19h	EM
26	1Ah	SUB
27	1Bh	ESC
28	1Ch	FS

CÓDIGO		CARACTER
29	1Dh	GS
30	1Eh	RS
31	1Fh	US
32	20h	SP
33	21h	!
34	22h	"
35	23h	#
36	24h	$
37	25h	%
38	26h	&
39	27h	'
40	28h	(
41	29h	)
42	2Ah	*
43	2Bh	+
44	2Ch	,
45	2Dh	-
46	2Eh	.
47	2Fh	/
48	30h	0
49	31h	1
50	32h	2
51	33h	3
52	34h	4
53	35h	5
54	36h	6
55	37h	7
56	38h	8
57	39h	9

CÓDIGO		CARACTER
58	3Ah	:
59	3Bh	;
60	3Ch	<
61	3Dh	=
62	3Eh	>
63	3Fh	?
64	40h	@
65	41h	A
66	42h	B
67	43h	C
68	44h	D
69	45h	E
70	46h	F
71	47h	G
72	48h	H
73	49h	I
74	4Ah	J
75	4Bh	K
76	4Ch	L
77	4Dh	M
78	4Eh	N
79	4Fh	O
80	50h	P
81	51h	Q
82	52h	R
83	53h	S
84	54h	T
85	55h	U
86	56h	V

CÓDIGO		CARACTER
87	57h	W
88	58h	X
89	59h	Y
90	5Ah	Z
91	5Bh	[
92	5Ch	\
93	5Dh	]
94	5Eh	^
95	5Fh	_
96	60h	`
97	61h	a
98	62h	b
99	63h	c
100	64h	d
101	65h	e
102	66h	f
103	67h	g
104	68h	h
105	69h	i
106	6Ah	j
107	6Bh	k
108	6Ch	l
109	6Dh	m
110	6Eh	n
111	6Fh	o
112	70h	p
113	71h	q
114	72h	r
115	73h	s

CÓDIGO		CARACTER
116	74h	t
117	75h	u
118	76h	v
119	77h	w
120	78h	x
121	79h	y
122	7Ah	z
123	7Bh	{
124	7Ch	|
125	7Dh	}
126	7Eh	~
127	7Fh	DEL