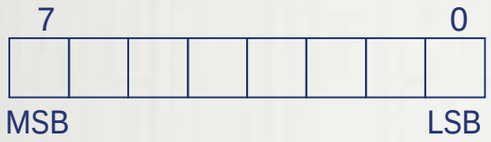
TEMA 3
REPRESENTACION DE DATOS

Bytes, words, LongWords, ... Megas, Gigas, ...



- La agrupación básica de bits es el **byte** (**octeto**), consistente en una cadena de 8 elementos.
- Esta unidad es la base para múltiplos con prefijos, kilo, mega, etc.

- Otras agrupaciones de bits suelen denominarse **word** (**palabra**) especificando su tamaño (p. ej. word de 16 bits).

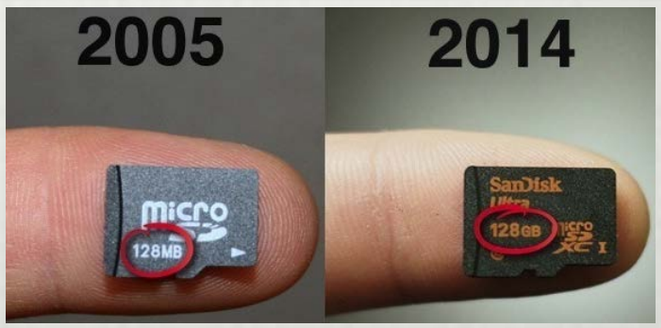
- En muchos sistemas actuales se establece el convenio de denominar word a las words de 16 bits y long word a las words de 32 bits

- Dentro de una agrupación de bits se identifica cada elemento por un índice que se establece típicamente comenzando en cero (p. ej. en un byte de 0 a 7).
- Gráficamente los índices crecen hacia la izquierda situandose el cero en el elemento que ocupa el extremo derecho.
- Entendiendo una cadena de bits como un número en representación binaria, el bit 0 es el dígito con menos peso, por lo que suele denominarse LSB (Least Significant Bit) y, en el extremo opuesto, el bit más significativo recibe la denominación de MSB (Most Significant Bit)

Múltiplos de Byte.

Se utilizan los prefijos habituales para unidades en progresión x1000 (kilo, mega, giga ,...) pero, para mantener una relación de escala potencia de 2, la progresión es x1024 ($1000 \approx 1024 = 2^{10}$).

Kilo → Mega → Giga → Tera → Peta → Exa → Zetta → Yotta → Bronto → Geop
2^{10} 2^{20} 2^{30} 2^{40} 2^{50} 2^{60} 2^{70} 2^{80} 2^{90} 2^{100}



Representación binaria, hexadecimal, octal y BCD.

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

Binario	Octal
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

- Las representaciones hexadecimal y octal son representaciones numéricas en base 16 y 8 respectivamente.
- En el caso de la representación hexadecimal se utilizan los símbolos decimales (0..9) más las letras A, B, C, D, E y F para representar los 16 necesarios. En la representación octal se usan los símbolos decimales que van del 0 al 7.
- Estas representaciones son convenientes como alternativas a la binaria ya que su traslación es directa al corresponder a cada símbolo cuatro y tres símbolos binarios respectivamente.

Binario	BCD
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9

- La representación BCD (Binary Coded Decimal - Decimal codificado en binario) es una representación sencilla de números decimales utilizando códigos binarios.
- Cada símbolo (dígito) decimal se sustituye directamente por su correspondiente representación binaria.
- Al no utilizarse todas las configuraciones binarias de 4 elemento, esta representación difiere de la obtenida por un cambio de base, siendo su ventaja la facilidad en el cambio de representación.

Ejemplo:

$0110\ 0101\ 1011\ 0000_B = 65B0_H = 26032_{10}$

$0\ 110\ 010\ 110\ 110\ 000_B = 062660_O = 26032_{10}$

$26032_{10} = 0010\ 0110\ 0000\ 0011\ 0020_{BCD}$

Formatos de texto (ASCII)

American
Standard
Code for
Information
Interchange

- Generado en 1963
- Estandariza 127 “caracteres”, de los cuales 94 son imprimibles y 33 no lo son.

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9
0 ...	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT
10 ...	LF	VT	FF / NP	CR	SO	SI	DLE	DC1	DC2	DC3
20 ...	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS
30 ...	RS	US	blanco	!	“	#	$	%	&	‘
40 ...	(	)	*	+	,	-	.	/	0	1
50 ...	2	3	4	5	6	7	8	9	:	;
60 ...	<	=	>	?	@	A	B	C	D	E
70 ...	F	G	H	I	J	K	L	M	N	O
80 ...	P	Q	R	S	T	U	V	W	X	Y
90 ...	Z	[	\	]	^	_	`	a	b	c
100 ...	d	e	f	g	h	i	j	k	l	m
110 ...	n	o	p	q	r	s	t	u	v	w
120 ...	x	y	z	{	|	}	~	del		

0	C-@	NUL Null prompt
1	C-A	SOH Start of heading
2	C-B	STX Start of text
3	C-C	ETX End of Text
4	C-D	EOT End of transmission
5	C-E	ENQ Enquiry
6	C-F	ACK Acknowledge
7	C-G	BEL Bell
8	C-H	BS Backspace
9	C-I	HT Horizontal tab
10	C-J	LF Line feed
11	C-K	VT Vertical tab
12	C-L	FF / NP Form feed / New page
13	C-M	CR Carriage return
14	C-N	SO Shift out
15	C-O	SI Shift in
16	C-P	DLE Data link escape
17	C-Q	DC1 X-ON
18	C-R	DC2
19	C-S	DC3 X-Off
20	C-T	DC4
21	C-U	NAK No achnowledge
22	C-V	SYN Synchronous idle
23	C-W	ETB End transmission blocks
24	C-X	CAN Cancel
25	C-Y	EM End of medium
26	C-Z	SUB Substitute
27	C-[	ESC Escape
28	C-\	FS File separator
29	C-]	GS Group separator
30	C-^	RS Record separator
31	C-_	US Unit separator
127	DEL	Delete or rubout

Formatos de texto (ISO-8859-1)

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9
120 ...	**ISO-8859-1**									
130 ...										
140 ...										
150 ...										
160 ...	nbs	¡	¢	£	¤	¥	¦	§	¨	©
170 ...	ª	«	¬		®	¯	°	±	²	³
180 ...	´	µ	¶	·	¸	¹	º	»	¼	½
190 ...	¾	¿	À	Á	Â	Ã	Ä	Å	Æ	Ç
200 ...	È	É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ
210 ...	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û
220 ...	Ü	Ý	Þ	ß	à	á	â	ã	ä	å
230 ...	æ	ç	è	é	ê	ë	ì	í	î	ï
240 ...	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù
250 ...	ú	û	ü	ý	þ	ÿ				

- Western (ISO-8859-1)
- Western (ISO-8859-15)
- Central European (ISO-8859-2)
- Central European (Windows-1250)
- Japanese (Auto-Detect)
- Japanese (Shift_JIS)
- Japanese (EUC-JP)
- Traditional Chinese (Big5)
- Traditional Chinese (EUC-TW)
- Simplified Chinese (GB2312)
- Korean (Auto-Detect)
- Cyrillic (ISO-8859-5)
- Cyrillic (KOI8-R)
- Cyrillic (Windows-1251)
- Cyrillic (CP-866)
- Greek (ISO-8859-7)
- Greek (Windows-1253)
- Turkish (ISO-8859-9)
- Unicode (UTF-8)
- Unicode (UTF-7)
- User-Defined
- Set Default Character Set

Opciones de codificación de caracteres presentadas por determinada versión de Netscape →

← Opciones de codificación de caracteres presentadas por determinada versión de Microsoft Explorer

- Árabe
- Báltico (ISO)
- Báltico (Windows)
- Centroeuropeo (DOS)
- Centroeuropeo (ISO)
- Centroeuropeo (Windows)
- Chino simplificado
- Chino tradicional (Big5)
- Cirílico (DOS)
- Cirílico (ISO)
- Cirílico (KOI8-R)
- Ucraniano (KOI8-U)
- Cirílico (Windows)
- Griego (ISO)
- Griego (Windows)
- Hebreo
- Japonés
- Coreano
- Tailandés
- Turco (ISO)
- Turco (Windows)
- Unicode (UTF-8)
- Definido por el usuario
- Vietnamita

Formatos de texto (ISO/IEC 10646-1 y Unicode)

•Actualmente se pretende una estandarización de una tabla de caracteres capaz de dar cobertura a todas las necesidades con independencia de máquinas, programas e idiomas.

•El consorcio UNICODE, formado por un gran número de expertos, instituciones y empresas del sector (entre las que se encuentran todas las más importantes), se está encargando de realizar esto utilizando una codificación de 16 bits.

•El mecanismo consiste en recibir propuestas de inclusión de nuevos conjuntos de caracteres, emitidas por cualquier persona u organización, y someterlas a análisis para incorporarlas al estándar o rechazarlas.

(continua ...)

Lista parcial de conjuntos de caracteres aceptados por UNICODE

Proposed Allocation	Count	Name	UTC Status	ISO Status (see Caution)
3096	1	HIRAGANA LETTER SMALL KE	99-Oct-29 Accepted	99-Sep-16 Stage 1
31F0..31FF	16	Small Katakana for Ainu	99-Oct-29 Accepted	99-Sep-16 Stage 1
FA30..FA67	56	Compatibility Japanese ideographs	99-Oct-29 Accepted	99-Sep-16 Stage 1
???	81	Philippine Scripts	98-Dec-02 Accepted	Stage 1
TBD (surrogates)	102	Linear B	97-May-29 Accepted	N/A
TBD (surrogates)	55	Cypriot Syllabary	97-May-29 Accepted	N/A
00010300.. 00010329	41	Etruscan	97-May-29 Accepted	99-Nov-8 Stage 4
00010330.. 0001034B	28	Gothic	97-May-29 Accepted	99-Nov-8 Stage 4
00010400.. 0001044D	76	Deseret Alphabet (phonetic English script)	96-Dec-06 Accepted	99-Nov-8 Stage 4
TBD (surrogates)	48	Shavian (phonetic English script)	97-May-29 Accepted	N/A
0001D000.. 0001D0F5	246	Greek Byzantine Musical Notation	96-Aug-05 Accepted	99-Nov-8 Stage 4
0001D103.. 0001D1DB	217	Western Musical Symbols	97-Dec-05 Accepted	99-Nov-8 Stage 4
0001D400..0001D7FF	986	Mathematical Alphanumeric Symbols	99-Oct-29 Accepted	99-Sep-16 Stage 1
000E0001.. 000E007F	97	Plane 14 tags	97-Dec-05 Accepted	99-Nov-8 Stage 4

Formatos de texto (ISO/IEC 10646-1 y Unicode)

(...viene de la anterior)

•En el mismo sentido, el ISO/IEC 10646-1 pretende una estandarización mediante una codificación de 32 bits, que se mantiene sincronizada con la de UNICODE. La diferencia fundamental consiste en que el mecanismo de ISO/IEC es más estricto y por tanto implica un mayor retardo en la aceptación o rechazo de propuestas. Todo lo actualmente aceptado por ISO/IEC 10646-1 ha sido previamente aceptado por UNICODE pero hay propuestas ya aceptadas por UNICODE que están bajo estudio por la ISO/IEC.

Lista parcial de conjuntos de caracteres en estudio o rechazados por UNICODE

Proposed Allocation	Count	Name	UTC Status	ISO Status (see Caution)
N/A		Supplemental Arabic for Uighur, Kazakh, and Kirghiz	96-Dec-06 Rejected	N/A
N/A	45	Phaistos Disk Script	97-May-29 Not accepted	N/A
N/A	95	Pollard	97-May-29 Comments requested	N/A
N/A	1	Mid-level hamzah	97-Jul-22 Withdrawn	N/A
N/A	1	MODIFIER LETTER MIDDLE DOT	97-Dec-05 Withdrawn	N/A
N/A		Klingon	Under investigation	N/A
N/A		Cirth	Under investigation	N/A
N/A		Tengwar	Under investigation	N/A
N/A		Ugaritic Cuneiform	Under investigation	N/A
N/A		Old Persian Cuneiform	Under investigation	N/A
N/A		Meroitic	Under investigation	N/A
N/A		Basic Egyptian Hieroglyphics	Under investigation	N/A
N/A	2	Ecological Symbols	98-Feb-26 Rejected	Stage 1
N/A	1	Conditional space: SOFT SPACE	Rejected	N/A
N/A	2	ZERO WIDTH LIGATOR ZERO WIDTH NONLIGATOR	00-Feb-03 Rejected	N/A

Formatos de texto (ISO/IEC 10646-1 y Unicode)

	000	001	002	003	004	005	006	007
0	NUL 0000	DLE 0010	SP 0020	0 0030	@ 0040	P 0050	` 0060	p 0070
1	SOH 0001	DC1 0011	! 0021	1 0031	A 0041	Q 0051	a 0061	q 0071
2	STX 0002	DC2 0012	" 0022	2 0032	B 0042	R 0052	b 0062	r 0072
3	ETX 0003	DC3 0013	# 0023	3 0033	C 0043	S 0053	c 0063	s 0073
4	EOT 0004	DC4 0014	$ 0024	4 0034	D 0044	T 0054	d 0064	t 0074
5	ENQ 0005	NAK 0015	% 0025	5 0035	E 0045	U 0055	e 0065	u 0075
6	ACK 0006	SYN 0016	& 0026	6 0036	F 0046	V 0056	f 0066	v 0076
7	BEL 0007	ETB 0017	' 0027	7 0037	G 0047	W 0057	g 0067	w 0077
8	BS 0008	CAN 0018	(0028	8 0038	H 0048	X 0058	h 0068	x 0078
9	HT 0009	EM 0019	) 0029	9 0039	I 0049	Y 0059	i 0069	y 0079
A	LF 000A	SUB 001A	* 002A	: 003A	J 004A	Z 005A	j 006A	z 007A
B	VT 000B	ESC 001B	+ 002B	; 003B	K 004B	[005B	k 006B	{ 007B
C	FF 000C	FS 001C	, 002C	< 003C	L 004C	\ 005C	l 006C	| 007C
D	CR 000D	GS 001D	- 002D	= 003D	M 004D	] 005D	m 006D	} 007D
E	SO 000E	RS 001E	. 002E	> 003E	N 004E	^ 005E	n 006E	~ 007E
F	SI 000F	US 001F	/ 002F	? 003F	O 004F	_ 005F	o 006F	DEL 007F

	008	009	00A	00B	00C	00D	00E	00F
0	XXX 0080	DCS 0090	NB SP 00A0	° 00B0	À 00C0	Đ 00D0	à 00E0	ð 00F0
1	XXX 0081	PU1 0091	¡ 00A1	± 00B1	Á 00C1	Ñ 00D1	á 00E1	ñ 00F1
2	BPH 0082	PU2 0092	¢ 00A2	² 00B2	Â 00C2	Ò 00D2	â 00E2	ò 00F2
3	NBH 0083	STS 0093	£ 00A3	³ 00B3	Ã 00C3	Ó 00D3	ã 00E3	ó 00F3
4	IND 0084	CCH 0094	¤ 00A4	´ 00B4	Ä 00C4	Ô 00D4	ä 00E4	ô 00F4
5	NEL 0085	MW 0095	¥ 00A5	µ 00B5	Å 00C5	Õ 00D5	å 00E5	õ 00F5
6	SSA 0086	SPA 0096	¦ 00A6	¶ 00B6	Æ 00C6	Ö 00D6	æ 00E6	ö 00F6
7	ESA 0087	EPA 0097	§ 00A7	· 00B7	Ç 00C7	× 00D7	ç 00E7	÷ 00F7
8	HTS 0088	SOS 0098	¨ 00A8	¸ 00B8	È 00C8	Ø 00D8	è 00E8	ø 00F8
9	HTJ 0089	XXX 0099	© 00A9	¹ 00B9	É 00C9	Ù 00D9	é 00E9	ù 00F9
A	VTS 008A	SCI 009A	ª 00AA	º 00BA	Ê 00CA	Ú 00DA	ê 00EA	ú 00FA
B	PLD 008B	CSI 009B	« 00AB	» 00BB	Ë 00CB	Û 00DB	ë 00EB	û 00FB
C	PLU 008C	ST 009C	¬ 00AC	¼ 00BC	Ì 00CC	Ü 00DC	ì 00EC	ü 00FC
D	RI 008D	OSC 009D	SHY 00AD	½ 00BD	Í 00CD	Ý 00DD	í 00ED	ý 00FD
E	SS2 008E	PM 009E	® 00AE	¾ 00BE	Î 00CE	Þ 00DE	î 00EE	þ 00FE
F	SS3 008F	APC 009F	¯ 00AF	¿ 00BF	Ï 00CF	ß 00DF	ï 00EF	ÿ 00FF

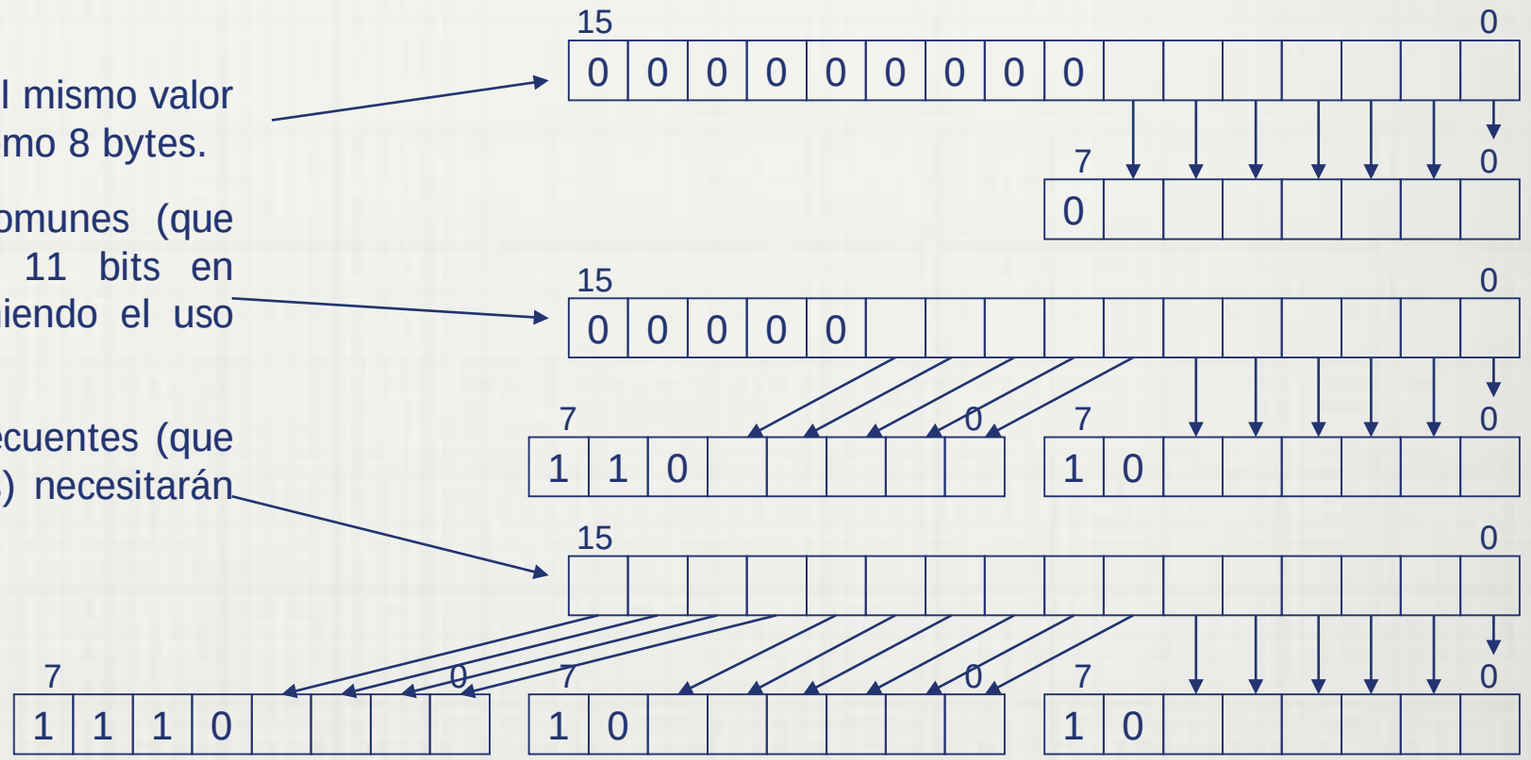
Formatos de texto (UTF-8)

•El estándar UNICODE contempla métodos de codificación alternativos al original -consistente en utilizar 16 bits por carácter- que pueden resultar convenientes en determinadas circunstancias. Uno de los más importantes es el UTF-8 (dentro de codificación UTF hay otras versiones: UTF-4 UTF-7 UTF-16 y UTF-32). Permite codificar los caracteres en 8, 16 y 24 bits en función de su código.

•Los caracteres ASCII, que tienen el mismo valor en UNICODE, pueden trasmitirse como 8 bytes.

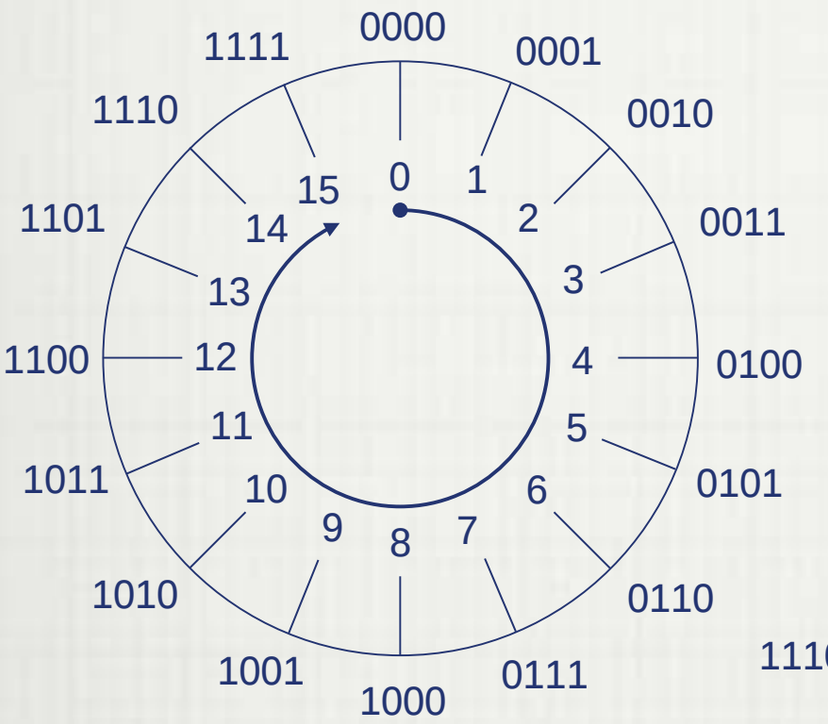
•Los caracteres no ASCII más comunes (que reciben códigos de no más de 11 bits en UNICODE) se re-codifican manteniendo el uso de 16 bits.

•Los caracteres no ASCII más infrecuentes (que tienen códigos con más de 11 bits) necesitarán de 24 bits.

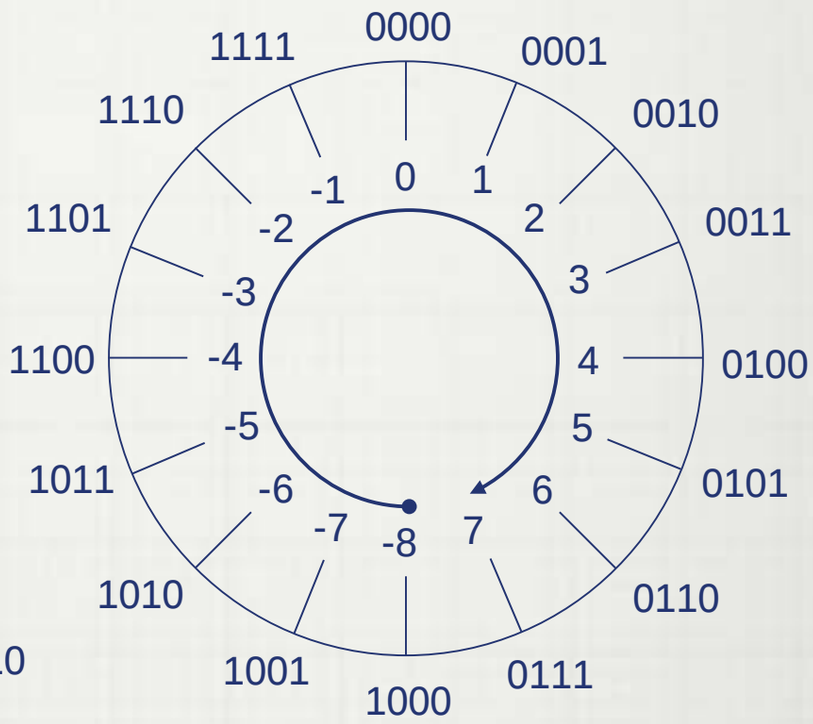


Este formato permitirá reducir el tamaño medio de los textos habituales frente a la codificación UNICODE estándar, lo que puede ser interesante en casos como, por ejemplo, la transmisión a través de redes.

Formatos numéricos (enteros)

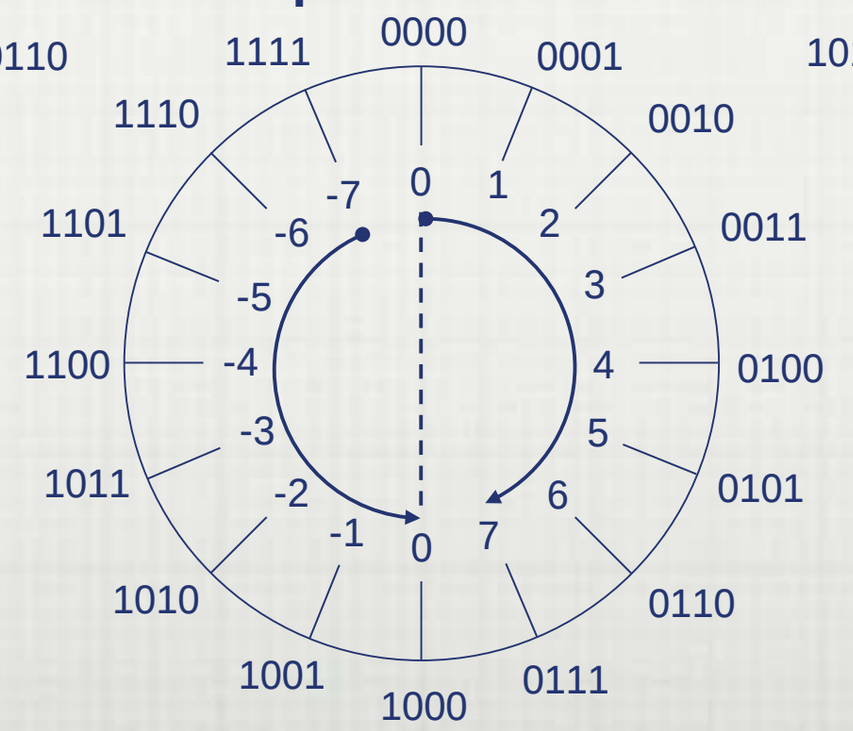


Sin signo



Con signo: complemento a dos

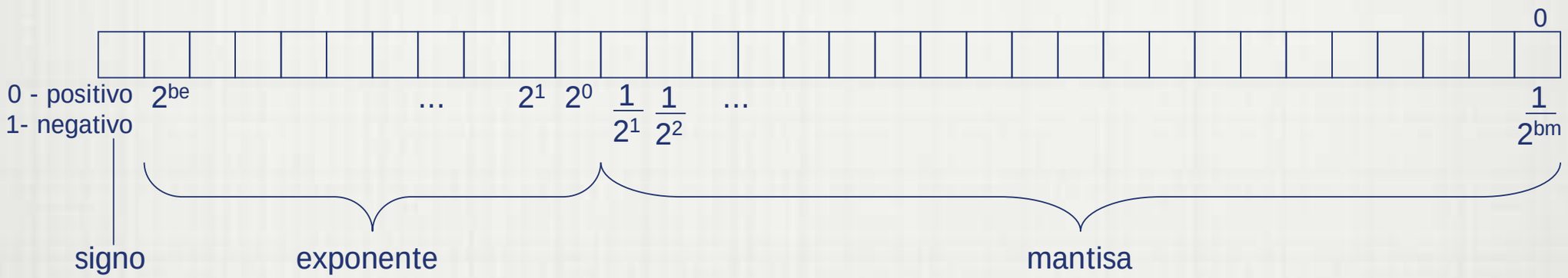
Con signo: complemento a uno



Formatos numéricos (reales: IEEE-754)

•Las primeras computadoras (Zuse, Stibitz,...) disponían de representación en coma flotante, pero John Von Newmann se mostró contrario a ella argumentando que podía utilizarse en todo caso representación entera y llevar un control del escalado.

•En 1985 se definió el IEEE-754, estándar para coma flotante con tres formatos
(32 bits -1,8,23-; 64 bits -1, 11, 52-; y 80 bits)



$$N=\begin{cases}(-1)^{signo}\ 2^{exponente-bias}\ (1+mantisa) & \text{si exponente} \neq 0 \text{ y exponente} \neq 2^{be}-1\\ (-1)^{signo}\ 2^{1-bias}\ mantisa & \text{si exponente}=0\\ \infty^{(*)} & \text{si exponente}= 2^{be}-1 \text{ y mantisa}=0\\ NaN^{(*)} & \text{si exponente}= 2^{be}-1 \text{ y mantisa}\neq 0\end{cases}$$

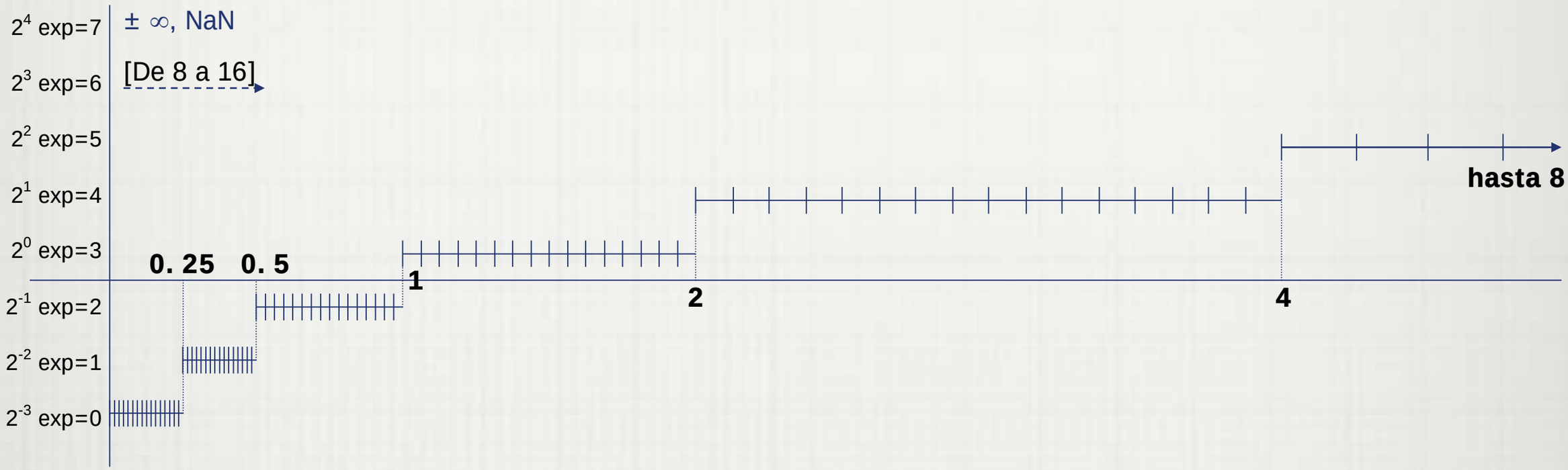
be - bits del exponente
bm - bits de la mantisa
bias= $2^{(be-1)}-1$
NaN: “Not a Number”

(*) el standard define todas las “excepciones” para un núcleo de 50 operaciones

Actualmente está en vigor una revisión de 2008 con tamaños de 16/32/64/128/256 además de admitir una codificación en base 10 con tamaños de 32/64/128

Formatos numéricos (reales: IEEE-754)

Números representables según norma IEEE-754 si la reducimos a 8 bits (1-3-4)



EJEMPLOS con representación a 8 bits (seeemmmm):

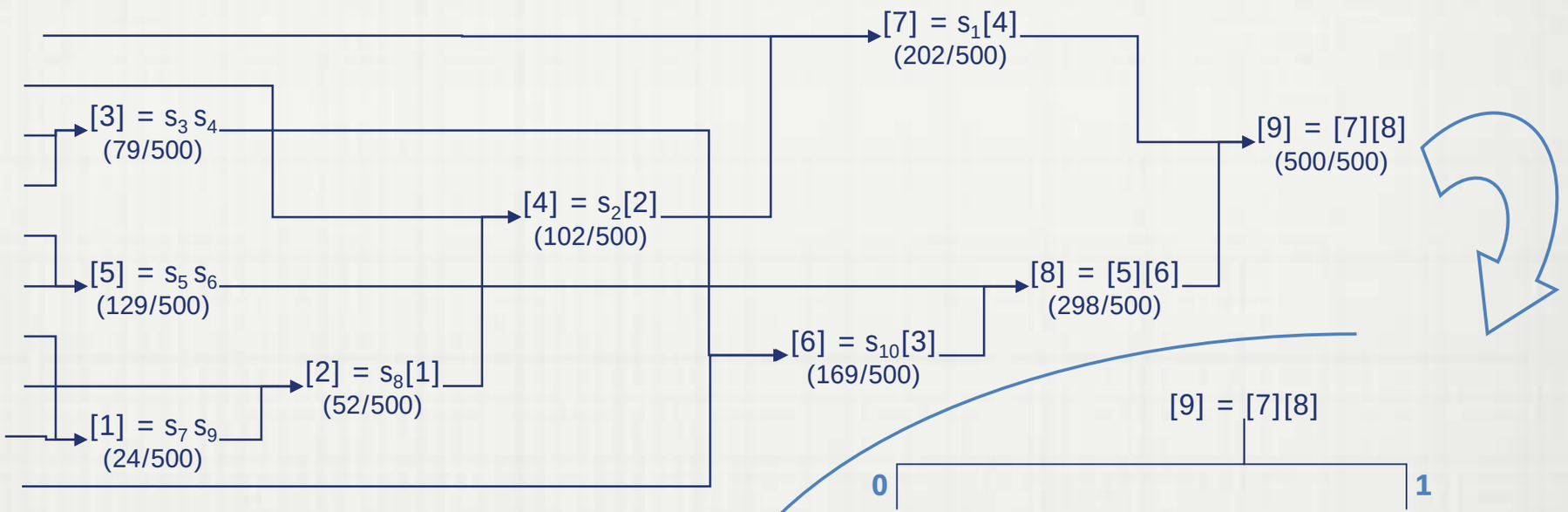
01110000 = + ∞	11110000 = - ∞
00110100 = + 2^{3-3} (1+1/4) = 1.25	10100000 = - 2^{2-3} (1+0) = 0.5
10001001 = - 2^{1-3} (1/2+1/16) =- 0,140625	00000001 = + 2^{1-3} (1/16) = 0,015625
10000000 = - 0	00000000 = 0

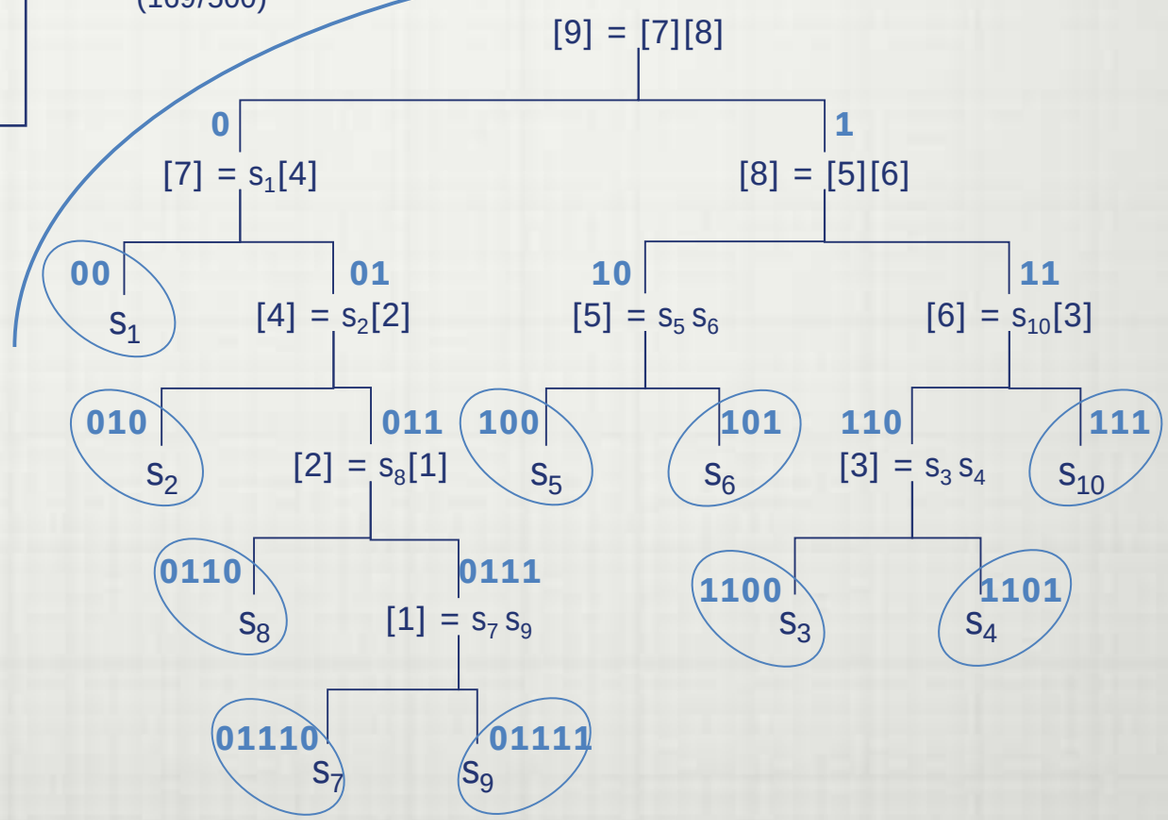
N=15.12 → N∈[2^3,2^4) → N=2^3*(15.12/2^3)=2^3*1.89=2^{6-3}*(1+0.89)=2^{6-3}*(1+1/2+1/4+1/8~~+1/128~~) → 01101110 (=15.00)

La precisión en la zona [2^3,2^4) es 2^3*(1/16)=0.5, luego el 15.12 no es representable (del 15 pasamos al 15.5)

Compresión de datos (Huffman)

Símbolo	Frecuencia
s_1	100/500
s_2	50/500
s_3	47/500
s_4	32/500
s_5	59/500
s_6	70/500
s_7	21/500
s_8	28/500
s_9	3/500
s_{10}	90/500





- En ocasiones, y casi exclusivamente cuando se trata de almacenamiento o transmisión, interesa reducir en la medida de lo posible la ocupación de espacio, por lo que se recurre a técnicas de codificación comprimida.
- Estas son de dos tipos: sin pérdida de información y con pérdida de información.
- La técnica básica sin pérdida de información es la codificación de Huffman, y la más utilizada actualmente la de Lempel-Ziv.
- Las codificaciones con pérdida de información suelen utilizarse en imagen y sonido, donde una degradación moderada (o imperceptible) de los datos puede tolerarse (p.ej jpeg, mp3, ...)

Formatos de instrucción

•El funcionamiento de un procesador consiste, básicamente, en un mecanismo cíclico de dos fases "**fetch**" y "**ejecución**". En **fetch** accede a la memoria para obtener una instrucción y en **ejecución** lleva a cabo lo que esta instrucción implique, lo que puede conllevar accesos a memoria para tomar (**leer**) o dejar (**escribir**) datos.

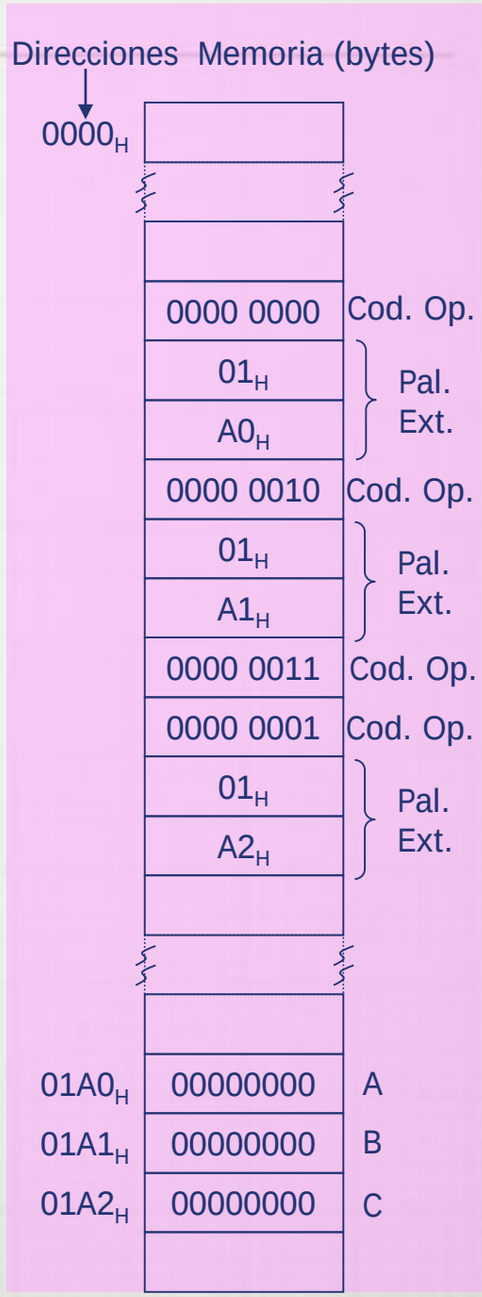
00 sumar
01 restar
10 multiplicar
11 dividir

• Cada procesador determina una codificación propia para las instrucciones que es capaz de ejecutar. Esta viene determinada por su arquitectura. A este código se le denomina "**código de operación**"

• Las instrucciones actúan en función de sus operandos, que pueden ser de varios tipos. La existencia de operandos puede implicar la necesidad de datos añadidos a la instrucción denominados "**palabras de extensión**"

Operandos { implícitos; inmediatos; en memoria de datos { con acceso directo; con acceso indirecto }

• El uso de una misma memoria para datos y programa lleva a la adopción de un tamaño de instrucción relacionado de algún modo con el tamaño de los datos manejados



Ejemplo con una arquitectura basada en acumulador:

00000000 LD Cargar byte en Acumulador
00000001 ST Descargar byte del Acumulador
00000010 AD Sumar byte al Acumulador
00000011 CM Complementar Acumulador

C=-(A+B)
dato A en memoria (dirección $01A0_H$)
dato B en memoria (dirección $01A1_H$)
Resultado C en memoria (dirección $01A2_H$)

LD $01A0_H$
AD $01A1_H$
CM
ST $01A2_H$

Formatos de instrucción

Arquitecturas de microprocesador y formatos de instrucción en relación a los operandos:

1 operando: típica de los primeros microprocesadores de propósito general. Las operaciones con dos operandos tienen uno implícito que es el "**acumulador**"

2 operandos: sucedió a la anterior (un ejemplo: Motorola 68000). Las operaciones con dos operandos dejan el resultado sobre uno de ellos por lo que su valor original se pierde.

3 operandos: utilizada en procesadores especiales (p.ej. DSP). Una instrucción típica es la que ejecuta producto y suma (op fact, fact, sum)

0 operandos: No es posible, pero puede aproximarse por término medio si muchas operaciones carecen de operandos explícitos. Es uno de los puntos básicos de la filosofía RISC (Reduced Instruction Set Computer). La máquina virtual JAVA es un ejemplo de estructura de este tipo.

CUALIDADES DE UN BUEN CONJUNTO DE INSTRUCCIONES

- Codificación adecuada a la frecuencia de uso
- Regularidad y ortogonalidad (operaciones-operandos-direccionamientos)