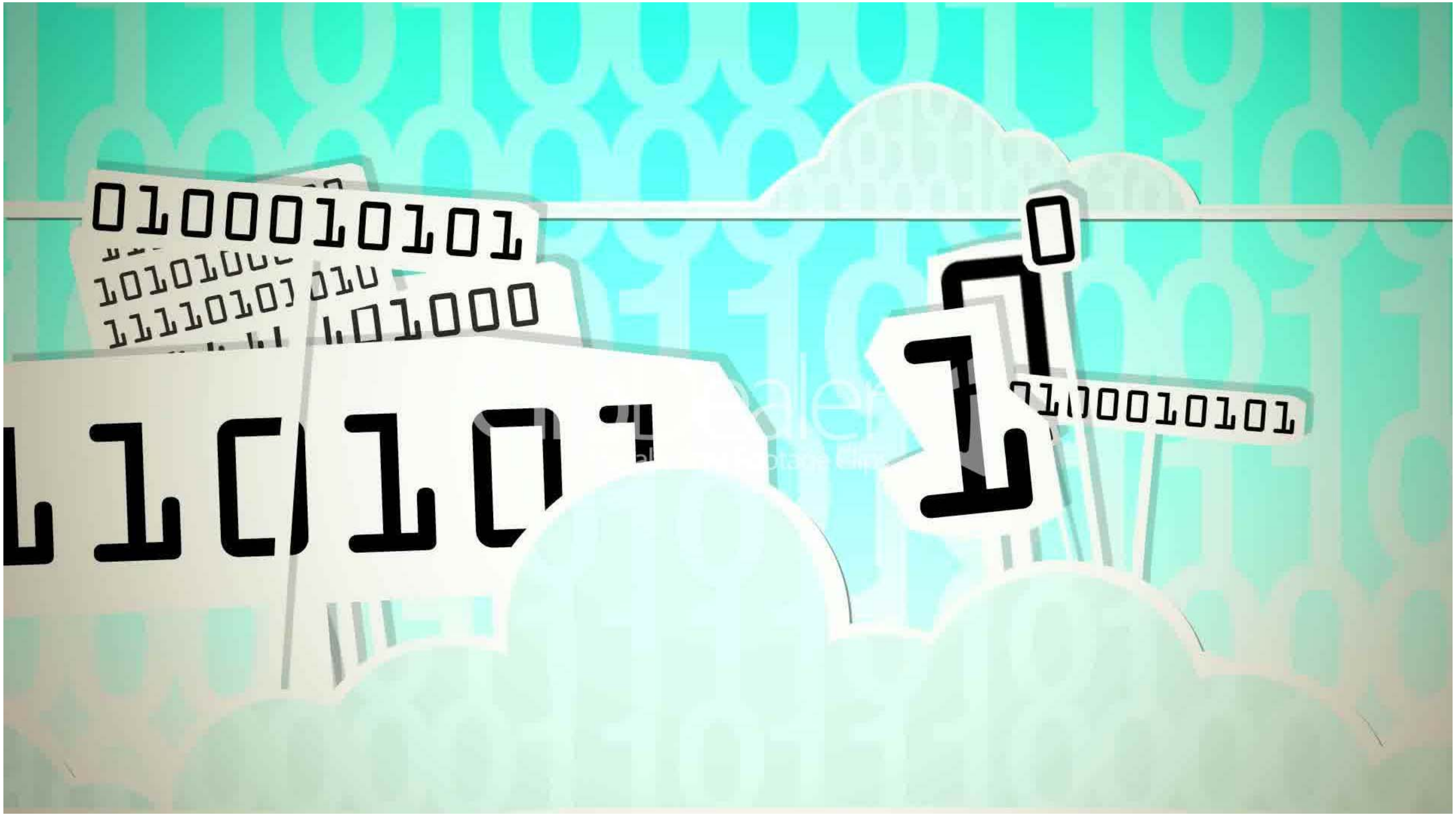


INFORMÁTICA

CODIFICACIÓN DE LA INFORMACIÓN

INTRODUCCIÓN

El mundo de los ordenadores es un mundo binario.

Sin embargo, el mundo de la información manejada por el hombre es mayoritariamente de texto o analógica.

Por ello, es importante establecer unos mecanismos de traducción o **codificación** que permitan convertir esa información.

La codificación es una transformación que representa los elementos de un conjunto mediante los de otro, de tal forma que a cada elemento del primer conjunto le corresponda un elemento distinto del segundo ➔ aplicación biyectiva.

Hombre	⟷	Man
Mujer	⟷	Woman
Coche	⟷	Car
Ordenador	⟷	Computer

Hombre	⟷	11010111
Mujer	⟷	11010101
Coche	⟷	11010110
Ordenador	⟷	11010100

INTRODUCCIÓN

Ejemplos de código: el D.N.I, el código de asignatura en la matrícula de la Universidad y el código provincial en los números de teléfono.

Gracias a los códigos se puede comprimir y estructurar la información.

La comunicación con el ordenador se realiza empleando un conjunto de símbolos, dividido en los siguientes grupos:

- Caracteres **alfabéticos**: letras, minúsculas y mayúsculas, del alfabeto inglés (a, b, c, ... , x, y, z, A, B, C, ... , X, Y, Z)
- Caracteres **numéricos**: formado por las 10 cifras decimales (0, 1, 2, ... , 9)
- Caracteres **especiales**: son todos los símbolos no incluidos en los grupos anteriores (* , + , . , ñ , ¿ , ! , ...)
- Caracteres **de control**: denominados caracteres no imprimibles ya que no se pueden visualizar directamente en pantalla. Los emplea el ordenador para realizar algunas tareas de control: emitir pitido, cambiar de línea, etc.

¿QUE IDIOMA HABLA?

- Un código nos permite realizar operaciones con la información.
- Binario es el más simple, sólo hay dos símbolos en un alfabeto binario, (0, 1).
- Los circuitos presentan 0V para el cero y 5V para el uno.
- Una unidad de información es un BIT, (un 0 o un 1).
- Para poder funcionar en binario, nosotros que estamos acostumbrados al decimal y a letras tenemos que establecer una correspondencia.

0	→	0000
1	→	0001
2	→	0010
3	→	0011
4	→	0100
5	→	0101
6	→	0110
7	→	0111
8	→	1000
9	→	1001
10	→	1010
11	→	1011
12	→	1100
13	→	1101
14	→	1110
15	→	1111

MEDIDA DE LA INFORMACIÓN

En el interior del ordenador la información se almacena y procesa mediante un código que usa sólo dos valores, representados por el 0 y el 1, y que se denomina **código binario**.

El ordenador codifica la información de entrada en código binario y decodifica la salida para presentar los resultados obtenidos:

La unidad mínima de información es el **bit** (**BI**nary Digi**T**). Cada bit puede tomar los valores 0 ó 1.

La unidad con significado en informática es el **byte**, que son 8 bits.

MEDIDA DE LA INFORMACIÓN

- La capacidad de almacenamiento de un ordenador se mide en bytes. Como el byte es una unidad pequeña, se suelen usar múltiplos de éste:

Kilobyte (KB)	1KB = 2^{10} bytes = 1024 bytes
Megabyte (MB)	1MB = 2^{10} KB = 2^{20} bytes = 1.048.576 bytes
Gigabyte (GB)	1GB = 2^{10} MB = 2^{30} bytes = 1.073.741.824 bytes
Terabyte (TB)	1TB = 2^{10} GB = 2^{40} bytes

- Estos múltiplos no sólo se pueden usar con bytes, sino también con bits, por ejemplo, 1Mb (1 Megabit), 1 Gb (1 Gigabit), etc.
- Se suele usar una b minúscula para indicar un bit y una B mayúscula para indicar un byte.

SISTEMAS DE REPRESENTACIÓN

Los sistemas de representación de números se llaman **sistemas de numeración.**

- **Cifras o dígitos:** los símbolos que se usan para representar números.
- **Base de un sistema de numeración:** el cardinal del conjunto de cifras.
- **Sistema de numeración posicional:** cada dígito tiene un peso de acuerdo con su posición.
- En la representación de un número:
  - ➢el dígito más a la izquierda es el más significativo (el de más peso)
  - ➢el dígito más a la derecha es el menos significativo

Ejemplo : Sistema de numeración decimal, sistema de numeración binario, sistema de numeración octal y sistema de numeración hexadecimal.

SISTEMAS DE NUMERACIÓN USUALES

Por ejemplo, el número decimal 3.278,52 puede obtenerse como suma de:

3.000	→	Posición 3 ➔ peso: b^3 = 1000
200	→	Posición 2 ➔ peso: b^2 = 100
70	→	Posición 1 ➔ peso: b^1 = 10
8	→	Posición 0 ➔ peso: b^0 = 1
0,5	→	Posición -1 ➔ peso: b^{-1} = 0,1
+ 0,02	→	Posición -2 ➔ peso: b^{-2} = 0,01
3.278,52		

Es decir, se verifica que:

$$3.278,52 = 3\times10^3 + 2\times10^2 + 7\times10^1 + 8\times10^0 + 5\times10^{-1} + 2\times10^{-2}$$

SISTEMAS DE NUMERACIÓN USUALES

Generalizando para cualquier base, se tiene que la representación de un número *N* en la base *b*:

$$\mathbf{N = \ldots + n_4 \times b^4 + n_3 \times b^3 + n_2 \times b^2 + n_1 \times b^1 + n_0 \times b^0 + n_{-1} \times b^{-1} + n_{-2} \times b^{-2} + n_{-3} \times b^{-3} + \ldots}$$

Ejemplos:

$341 \quad = 3\times10^2 + 4\times10^1 + 1\times10^0 = 300 + 40 + 1 = 341$

$27{,}134 = 2\times10^1 + 7\times10^0 + 1\times10^{-1} + 3\times10^{-2} + 4\times10^{-3}$

$\qquad = 20 + 7 + 0{,}1 + 0{,}03 + 0{,}004 = 27{,}134$

Cuanto menor es la base mayor es el número de cifras que se necesitan para representar una cantidad dada.

SISTEMAS DE NUMERACIÓN DECIMAL Y BINARIO

Sistema decimal

- Es un sistema de numeración posicional.
- Cada posición tiene un peso potencia de 10 (la base).
- Dígitos diferentes: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

$$10^2\ 10^1\ 10^0\ .10^{-1}\ 10^{-2}\ 10^{-3}$$

Punto decimal

Ejemplo: $23.4 \longrightarrow 2\times10^1 + 3\times10^0 + 4\times10^{-1}$

Sistema binario

- Es un sistema de numeración posicional de base 2.
- Dígitos diferentes: 0, 1.

Ejemplo: $10\,.11 \longrightarrow 1\times2^1 + 0\times2^0 + 1\times2^{-1} + 1\times2^{-2} \longrightarrow 2.75$

SISTEMA DE NUMERACIÓN BINARIO

Contar en binario

Número Decimal	Número Binario			
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

- Con n bits para cifras enteras puedo representar: $0 \longrightarrow 2^n - 1$
- **Ejemplo:**

$$n = 5 \longrightarrow 11111 \longrightarrow 2^n - 1 = 31$$

- Cada sistema de numeración tiene su propia aritmética definida por las operaciones aritméticas básicas: suma y multiplicación.
- En un sistema de numeración posicional multiplicación y división por la base son inmediatas.

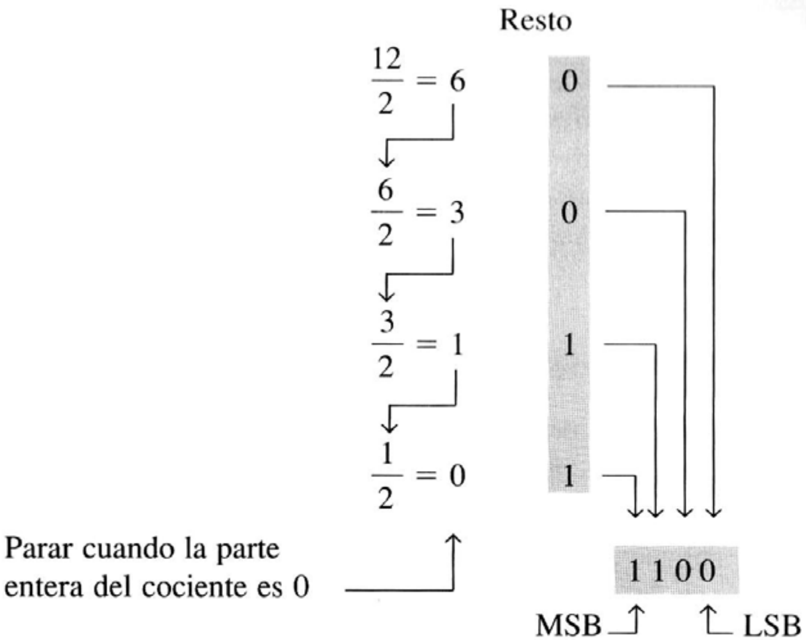
CONVERSIÓN DECIMAL-BINARIO

Método de la suma de pesos. De decimal a binario o al revés.

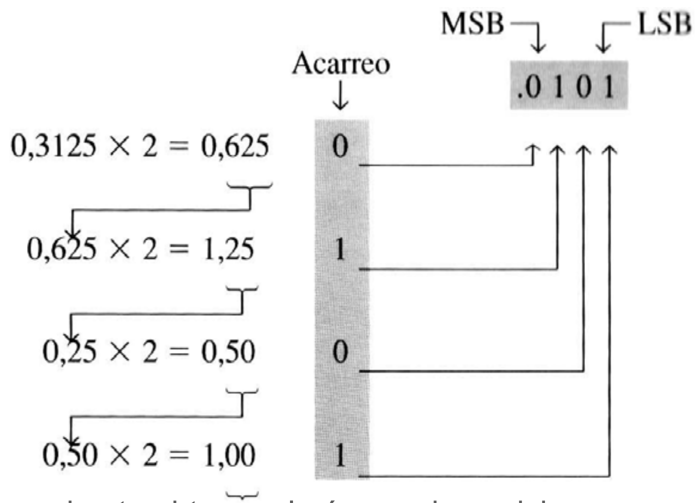
$$12 = 8 + 4 = 2^3 + 2^2 \longrightarrow 1100$$

Método de la división sucesiva por 2. Para pasar de decimal a binario.

Método de la multiplicación sucesiva por 2. Para convertir decimales fraccionarios a binarios.



División sucesiva por 2.



Continuar hasta obtener el número de posiciones decimales deseadas o cuando la parte fraccional sea toda cero.

Multiplicación sucesiva por 2.

CAMBIOS SISTEMA

Decimal_Bin
Resto Ndec/2

52	0
26	0
13	1
6	0
3	1
1	1

Decimal_Hex
Resto Ndec/16

28	12=C
1	1

Binario a Decimal

10010110

$1x2^7 + 0x2^6 + 0x2^5 + 1x2^4 + 0x2^3 + 1x2^2 + 1x2^1 + 0x2^0 = 150$

Dec	Hex
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	A
11	B
12	C
13	D
14	E
15	F

CÓDIGOS INTERMEDIOS

- Como el sistema binario no es muy cómodo, también se usan otros dos sistemas de numeración denominados *códigos intermedios*:
  - El sistema hexadecimal (base 16)
  - El sistema octal (base 8)
- Los códigos intermedios se fundamentan en la facilidad de transformar un número en base 2 a otra base que sea potencia de 2, y viceversa.
- Como 16 es 2^4 y 8 es 2^3, un dígito hexadecimal corresponde a cuatros dígitos y un dígito octal a tres dígitos binarios.

SISTEMA DE NUMERACIÓN OCTAL

- Sistema de numeración posicional de base 8.
- Dígitos diferentes: 0, 1, 2, 3, 4, 5, 6, 7.
- Contar: 0, 1, 2, 3, 4, 5, 6, 7, 10 , 11, 12, 13, 14, 15, 16, 17, 20, 21 …
- **Conversión de octal a decimal:** método de suma de pesos.

Peso:	8^3	8^2	8^1	8^0
Número Octal:	2	3	7	4

$$\begin{aligned} 2374_8 &= (2 \times 8^3) + (3 \times 8^2) + (7 \times 8^1) + (4 \times 8^0) \\ &= (2 \times 512) + (3 \times 64) + (7 \times 8) + (4 \times 1) \\ &= 1024 + 192 + 56 + 4 = 1276_{10} \end{aligned}$$

- **Conversión de octal a binario y de binario a octal:** ir convirtiendo una a una por separado cada cifra de octal a binario en el primer caso e ir agrupando las cifras de tres en tres y convirtiéndolas a octal en el segundo.

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

$$\begin{array}{ccc} 1 & 4 & 0 \\ \downarrow & \downarrow & \downarrow \\ \overbrace{001} & \overbrace{100} & \overbrace{000} \end{array}$$

SISTEMA DE NUMERACIÓN HEXADECIMAL

- **Conversión de hexadecimal a decimal:** método de suma de pesos.
- **Conversión de hexadecimal a binario:** ir convirtiendo una a una por separado cada cifra hexadecimal a binario.

$$\begin{array}{cccc} 1 & 0 & A & 4 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ 1 & \overbrace{0000} & \overbrace{1010} & \overbrace{0100} \end{array} \qquad \begin{array}{cccc} C & F & 8 & E \\ \downarrow & \downarrow & \downarrow & \downarrow \\ \overbrace{1100} & \overbrace{1111} & \overbrace{1000} & \overbrace{1110} \end{array}$$

- **Conversión de binario a hexadecimal:** agrupar las cifras binarias en grupos de 4 empezando por la derecha e traducirlas a hexadecimal.

$$\begin{array}{ccccc} \underbrace{1100} & \underbrace{1010} & \underbrace{0101} & \underbrace{0111} & \\ \downarrow & \downarrow & \downarrow & \downarrow & \\ C & A & 5 & 7 & = CA57_{16} \end{array}$$

- **Conversión de decimal a hexadecimal:** pasar el número a binario y agrupar las cifras de cuatro en cuatro a partir del punto decimal.

$$\underbrace{\quad 01}_{1}\underbrace{0011}_{3}\,.\,\underbrace{1101}_{13}\underbrace{0\quad}_{0} \longrightarrow 13.D4 \text{ (en hexadecimal)}$$

REPRESENTACIÓN DE NÚMERO BINARIOS CON SIGNO

Es necesario representar signo y magnitud del número.

Tres formatos binarios:

- Sistema signo-magnitud.
- Sistema del complemento a 1.
- Sistema del complemento a 2.

REPRESENTACIÓN DE NÚMERO BINARIOS CON SIGNO

Signo-magnitud

- El bit más a la izquierda es el **bit de signo** (1 si es negativo y 0 si positivo)
- Los **bits de magnitud** son el número sin signo en binario puro tanto para los positivos como para los negativos

Ejemplo: Expresar en signo-magnitud los números 25 y -25.

$$25 \longrightarrow 0\underbrace{00011001} \qquad -25 \longrightarrow 1\underbrace{00011001}$$

bits de signo

bits de magnitud

NÚMEROS BINARIOS CON SIGNO: COMPLEMENTO A 1

- El bit más a la izquierda es el **bit de signo**
- Los números positivos se representan igual que en signo magnitud.
- Los números negativos son el complemento a 1 del correspondiente número positivo.
- Hacer el complemento a 1 de un número es cambiar cada 0 por un 1 y viceversa.
- Con n bits se pueden representar
- Hay dos posibles representaciones para el 0.

$$-(2^{n-1}-1) \leq x \leq (2^{n-1}-1)$$

COMPLEMENTO A 1

Ejemplo: Expresar en complemento a 1 los números 25 y -25.

$$25 \longrightarrow 000011001$$

$$-25 \longrightarrow 111100110$$

bits de signo

¿Qué número decimal representa el número 1100 si está expresado en C1? → El número -3.

NÚMEROS BINARIOS CON SIGNO: COMPLEMENTO A 2

- El bit más a la izquierda es el **bit de signo.**
- Los números positivos igual que en signo magnitud.
- Los números negativos son el complemento a 2 del correspondiente número positivo.
- Hacer el complemento a 2 de un número es hacer el complemento a 1 de dicho número y sumar 1 al bit menos significativo.
- Rango de representación con n bits:

$$-2^{n-1} \leq x \leq (2^{n-1}-1)$$

COMPLEMENTO A 2

Ejemplo: Expresar en complemento a 2 los números 25 y -25.

$$
\begin{array}{rcr}
25 & \longrightarrow & 000011001 \\
 & & 111100110 \\
 & & + \qquad\quad 1 \\
\hline
-25 & \longrightarrow & 111100111
\end{array}
$$

¿Qué número decimal representan los números 1101 y 0010 si están expresados en C2?

Los número -3 y 2.

NEGATIVOS

	Sign/Mag	Comp 1	Comp 2
3+2=5	0011 +0010 0101	0011 +0010 0101	0011 +0010 0101
-3+2=-1	~~1011~~ ~~+0010~~ 1101=-5	1100 +0010 1110=-1	1101 +0010 1111=-1
-3+(-2)=-5	~~1011~~ ~~+1010~~ 0101=5	~~1100~~ ~~+1101~~ 1001=-6	1101 +1110 1011=-5

NÚMEROS NEGATIVOS

Binario	Decimal	Signo magnitud	Complemento a uno	Complemento a dos
000	0	0	0	0
001	1	1	1	1
010	2	2	2	2
011	3	3	3	3
100	4	-0	-3	
101	5	-1	-2	-3
110	6	-2	-1	-2
111	7	-3	-0	-1

Bit signo:
- 0 positivo
- 1 negativo

Invertir binario para hacer negativo

Invertir binario y sumar uno para hacer negativo

-5

0101 → 1101 — Signo magnitud

↓

1010 — Comp 1

↓

+1

↓

1011 — Comp 2

OPERACIONES ARITMÉTICAS EN BINARIO

Suma.

A B Acarreo	Suma
$0+0=0$	0
$0+1=0$	1
$1+0=0$	1
$1+1=1$	0

Multiplicación.

A B producto
$0\times 0=0$
$0\times 1=1$
$1\times 0=0$
$1\times 1=1$

- La suma de números binarios se realiza comenzando por la derecha y propagando el acarreo hacia la izquierda.
- El producto de números binarios se realiza como en decimal, desplazando cada producto parcial una posición a la izquierda.
- **Overflow o desbordamiento:** cuando el resultado de una operación necesita un número de bits mayor que el número de bits de los operandos

OPERACIONES ARITMÉTICAS EN C2

Suma. Operandos del mismo signo.

- Se suman los número bit a bit y se desprecia el acarreo final.

$$\begin{array}{rcrl} -14 & \longrightarrow & 11110010 & \\ -9 & \longrightarrow & 11110111 & + \\ \hline -23 & \longrightarrow & 111101001 & \end{array}$$

Acarreo que se desprecia

Resta. Operandos de diferente signo.

- Se cambia el signo al sustraendo realizándole el complemento a 2 y se le suma al minuendo despreciando el acarreo final.

$$\begin{array}{rcrl} 8 & \longrightarrow & 00001000 & \\ -3 & \longrightarrow & 11111101 & + \\ \hline 5 & \longrightarrow & 100000101 & \end{array}$$

El C2 de 00000011 (3)

CÓDIGOS ALFANUMÉRICOS

- Son códigos que permiten representar no sólo números sino también letras y otros símbolos.
- Información que necesitaría codificar:
  - las 26 letras del alfabeto
  - las 10 cifras decimales
  - los signos especiales (puntuación, punto y coma,…)
  - códigos de control,
  - el alfabeto griego, …
- Hay cientos de estándares de codificación pero nos centraremos en dos:
  - **ASCII** (código estándar americano para el intercambio de información) por su uso extendido y su importancia histórica
  - **UNICODE** por ser el estándar más actual y novedoso y que aglutina otras iniciativas además de incluir al ASCII.

CÓDIGO ASCII

- ASCII: American Standard Code for Information Interchange:
  - Usa 7 bits → 128 caracteres
- ASCII extendido:
  - Usa 8 bits → 256 caracteres
  - La parte extendida no es estándar, existen variantes
  - La parte extendida se denomina página de códigos
- Dentro del ASCII extendido se distinguen 2 grupos:
  - Caracteres imprimibles o de texto: letras, números y símbolos
  - Caracteres de control: acciones o estados de la transmisión de la información, por ejemplo: salto de línea, fin de mensaje, etc.

CÓDIGO ASCII

Caracteres de control				Símbolos gráficos											
Nombre	**Dec**	**Binario**	**Hex**	**Símbolo**	**Dec**	**Binario**	**Hex**	**Símbolo**	**Dec**	**Binario**	**Hex**	**Símbolo**	**Dec**	**Binario**	**Hex**
NUL	0	0000000	00	space	32	0100000	20	@	64	1000000	40	`	96	1100000	60
SOH	1	0000001	01	!	33	0100001	21	A	65	1000001	41	a	97	1100001	61
STX	2	0000010	02	"	34	0100010	22	B	66	1000010	42	b	98	1100010	62
ETX	3	0000011	03	#	35	0100011	23	C	67	1000011	43	c	99	1100011	63
EOT	4	0000100	04	$	36	0100100	24	D	68	1000100	44	d	100	1100100	64
ENQ	5	0000101	05	%	37	0100101	25	E	69	1000101	45	e	101	1100101	65
ACK	6	0000110	06	&	38	0100110	26	F	70	1000110	46	f	102	1100110	66
BEL	7	0000111	07	´	39	0100111	27	G	71	1000111	47	g	103	1100111	67
BS	8	0001000	08	(	40	0101000	28	H	72	1001000	48	h	104	1101000	68
HT	9	0001001	09	)	41	0101001	29	I	73	1001001	49	i	105	1101001	69
LF	10	0001010	0A	*	42	0101010	2A	J	74	1001010	4A	j	106	1101010	6A
VT	11	0001011	0B	+	43	0101011	2B	K	75	1001011	4B	k	107	1101011	6B
FF	12	0001100	0C	,	44	0101100	2C	L	76	1001100	4C	l	108	1101100	6C
CR	13	0001101	0D	-	45	0101101	2D	M	77	1001101	4D	m	109	1101101	6D
SO	14	0001110	0E	.	46	0101110	2E	N	78	1001110	4E	n	110	1101110	6E
SI	15	0001111	0F	/	47	0101111	2F	O	79	1001111	4F	o	111	1101111	6F
DLE	16	0010000	10	0	48	0110000	30	P	80	1010000	50	p	112	1110000	70
DC1	17	0010001	11	1	49	0110001	31	Q	81	1010001	51	q	113	1110001	71
DC2	18	0010010	12	2	50	0110010	32	R	82	1010010	52	r	114	1110010	72
DC3	19	0010011	13	3	51	0110011	33	S	83	1010011	53	s	115	1110011	73
DC4	20	0010100	14	4	52	0110100	34	T	84	1010100	54	t	116	1110100	74
NAK	21	0010101	15	5	53	0110101	35	U	85	1010101	55	u	117	1110101	75
SYN	22	0010110	16	6	54	0110110	36	V	86	1010110	56	v	118	1110110	76
ETB	23	0010111	17	7	55	0110111	37	W	87	1010111	57	w	119	1110111	77
CAN	24	0011000	18	8	56	0111000	38	X	88	1011000	58	x	120	1111000	78
EM	25	0011001	19	9	57	0111001	39	Y	89	1011001	59	y	121	1111001	79
SUB	26	0011010	1A	:	58	0111010	3A	Z	90	1011010	5A	z	122	1111010	7A
ESC	27	0011011	1B	;	59	0111011	3B	[	91	1011011	5B	{	123	1111011	7B
FS	28	0011100	1C	<	60	0111100	3C	\	92	1011100	5C	|	124	1111100	7C
GS	29	0011101	1D	=	61	0111101	3D	]	93	1011101	5D	}	125	1111101	7D
RS	30	0011110	1E	>	62	0111110	3E	^	94	1011110	5E	~	126	1111110	7E
US	31	0011111	1F	?	63	0111111	3F	—	95	1011111	5F	Del	127	1111111	7F

CÓDIGO ASCII 8 BITS

8 bit ASCII codes

The space character appears in place of unprintable characters

0		32		64	@	96	‘	128		160		192	À	224	à
1		33	!	65	A	97	a	129		161	¡	193	Á	225	á
2		34	"	66	B	98	b	130		162	¢	194	Â	226	â
3		35	#	67	C	99	c	131		163	£	195	Ã	227	ã
4		36	$	68	D	100	d	132		164	¤	196	Ä	228	ä
5		37	%	69	E	101	e	133		165	¥	197	Å	229	å
6		38	&	70	F	102	f	134		166	¦	198	Æ	230	æ
7		39	’	71	G	103	g	135		167	§	199	Ç	231	ç
8		40	(	72	H	104	h	136		168	¨	200	È	232	è
9		41	)	73	I	105	i	137		169	©	201	É	233	é
10		42	*	74	J	106	j	138		170	ª	202	Ê	234	ê
11		43	+	75	K	107	k	139		171	«	203	Ë	235	ë
12		44	,	76	L	108	l	140		172	¬	204	Ì	236	ì
13		45	−	77	M	109	m	141		173	-	205	Í	237	í
14		46	.	78	N	110	n	142		174	®	206	Î	238	î
15		47	/	79	O	111	o	143		175	¯	207	Ï	239	ï
16		48	0	80	P	112	p	144		176	°	208	Ð	240	ð
17		49	1	81	Q	113	q	145		177	±	209	Ñ	241	ñ
18		50	2	82	R	114	r	146		178	²	210	Ò	242	ò
19		51	3	83	S	115	s	147		179	³	211	Ó	243	ó
20		52	4	84	T	116	t	148		180	´	212	Ô	244	ô
21		53	5	85	U	117	u	149		181	µ	213	Õ	245	õ
22		54	6	86	V	118	v	150		182	¶	214	Ö	246	ö
23		55	7	87	W	119	w	151		183	·	215	×	247	÷
24		56	8	88	X	120	x	152		184	¸	216	Ø	248	ø
25		57	9	89	Y	121	y	153		185	¹	217	Ù	249	ù
26		58	:	90	Z	122	z	154		186	º	218	Ú	250	ú
27		59	;	91	[	123	{	155		187	»	219	Û	251	û
28		60	<	92	\	124	|	156		188	¼	220	Ü	252	ü
29		61	=	93	]	125	}	157		189	½	221	Ý	253	ý
30		62	>	94	^	126	~	158		190	¾	222	Þ	254	þ
31		63	?	95	_	127		159		191	¿	223	ß	255	ÿ

CÓDIGOS ALFANUMÉRICOS: UNICODE

- Estándar de codificación de 16 bits, lo que permite 65536 códigos (actualmente están ocupados la mitad).

  http://www.unicode.com

- Esfuerzo conjunto desde 1992 de diferentes organizaciones para unificar estándares:
  - Apple Computer, Borland, Digital, Hewlett-Packard, International Business Machines, Lotus, Microsoft (lo eligió como estándar para Windows NT), …
- Permite procesar y comunicar información en cualquier lengua y producir software que pueda ser utilizado en cualquier lengua.

CÓDIGOS ALFANUMÉRICOS: UNICODE

- Soluciona el problema de codificaciones estándar sin suficiente espacio para todos los símbolos que es necesario codificar.
- Codifica también por ejemplo caracteres fonéticos del chino o coreano.
- Dos características importantes:
  - codificación única (una letra que aparece en diferentes idiomas presenta una única codificación UNICODE)
  - codificación uniforme (todos los códigos de igual longitud)
- La última versión es la Unicode Standard, Version 5.1.0 del año 2007

REPRESENTAR FRACCIONALES

Punto fijo: El punto (coma) se toma siempre en la misma posición (ej. Enteros)

$$100100.11$$

Punto flotante

mantisa exponente

$$.11011010*2^{001100101}$$

Base

NÚMEROS EN PUNTO FLOTANTE

La aritmética en punto flotante opera con números reales.

$$0.000001_{10} \text{ o } 1.0_{10} \times 10^{-6}$$

$$3333333_{10} \text{ o } 3.33_{10} \times 10^{6}$$

Notación científica: un solo dígito a la izquierda del punto decimal.
Notación científica normalizada: si además el dígito a la izquierda del punto decimal no es un cero.

- Los números binarios también se pueden expresar en notación científica siendo 2 la base para ajustar el número de cifras decimales:

$$1.\text{xxxxxxxxx}_2 \times 2^{\text{yyyy}_2} \qquad \text{Por ejemplo: } 1.001 \times 2^{10}$$

- Ventajas de la notación científica:
  - Simplifica los algoritmos al estar siempre los números en la misma notación.
  - Incrementa la precisión de los números que pueden almacenarse en una palabra.

REPRESENTACIÓN DE NÚMEROS EN PUNTO FLOTANTE

Notación estándar de punto flotante del IEEE 754 utilizada por MIPS y DLX.

En general, los números en punto flotante son de la forma:

$$(-1)^{s} \times \text{mantisa} \times 2^{\text{exponente}}$$

s	exponente	mantisa
1bit	8 bits	23 bits

- El exponente se almacena en complemento a 2.
- En el campo mantisa se almacena los 23 bits de la parte fraccionaria.
- Diseño exige decisión de compromiso entre tamaño de mantisa (que limita la precisión) y tamaño de exponente (que limita el rango).
- Se pueden representar números positivos en el rango: $\{2.0 \times 10^{-38}, 2.0 \times 10^{+38}\}$
- **Overflow:** si el exponente positivo no cabe en 8 bits.
- **Underflow:** si el exponente negativo no cabe en 8 bits.

REPRESENTACIÓN DE NÚMEROS EN DOBLE PRECISIÓN

También vemos la notación estándar de punto flotante del IEEE 754.

Los números en punto flotante de doble precisión (*doubles*) requieren dos palabras (64 bits):

s	exponente	mantisa
1bit	11 bits	52 bits

- En este formato se pueden representar números positivos en el rango:

$$\left\{2.0\times 10^{-308}, 2.0\times 10^{+308}\right\}$$

- En realidad hay un bit más a 1 en la mantisa que se da implícito para los números binarios normalizados en IEEE 754 (en simple y en doble precisión), así que realmente tenemos (si m1 es el bit de la mantisa más a la izquierda):

$$(-1)^{s}\times\left(\left(1+\text{mantisa}\right)\times 2^{\text{exponente}}\right) =$$

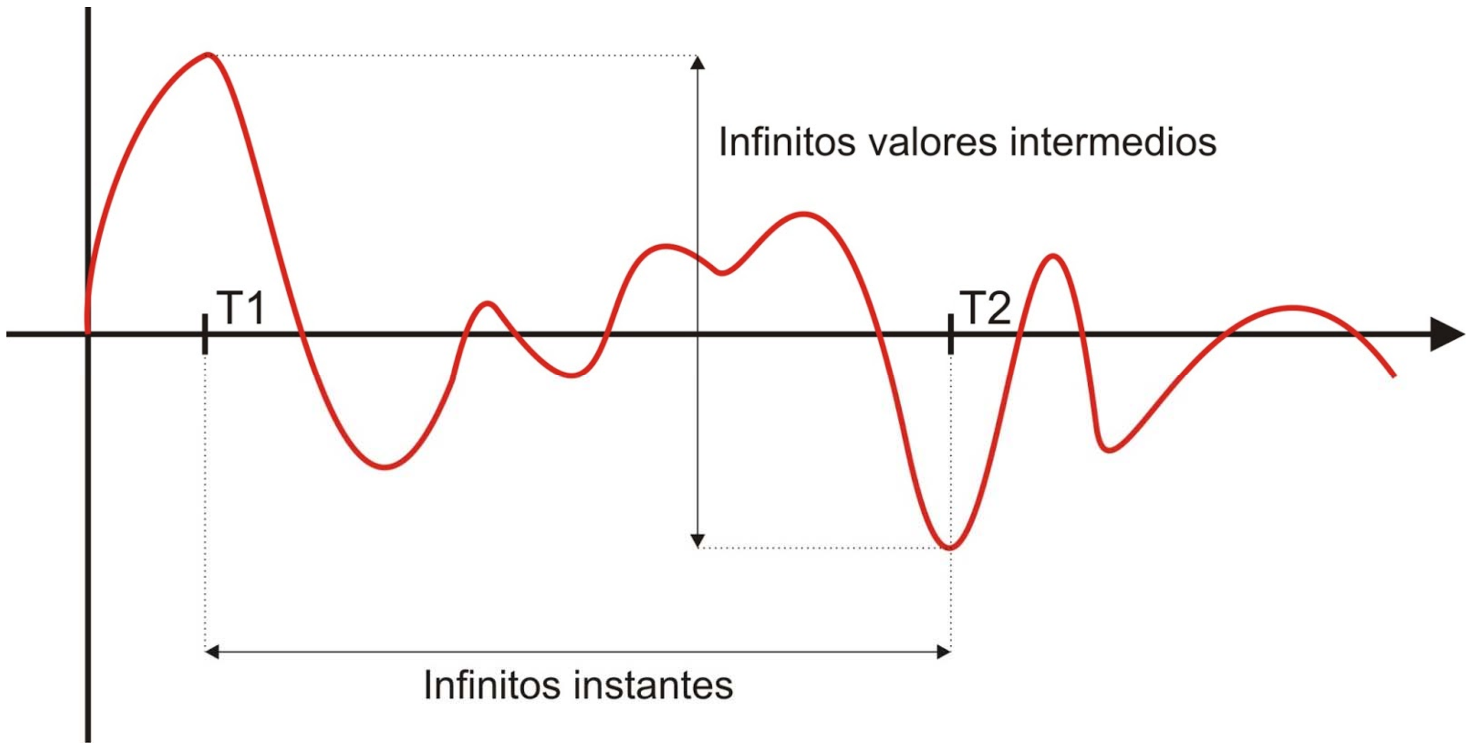
$$(-1)^{s}\times(1+(\text{m}1\times 2^{-1})+(\text{m}2\times 2^{-2})+(\text{m}3\times 2^{-3})+\ldots)\times 2^{\text{exponente}}$$

- Para el 00… no se suma el 1 a la mantisa y se identifica con 00… en el exponente.

REPRESENTACIÓN DE SEÑALES

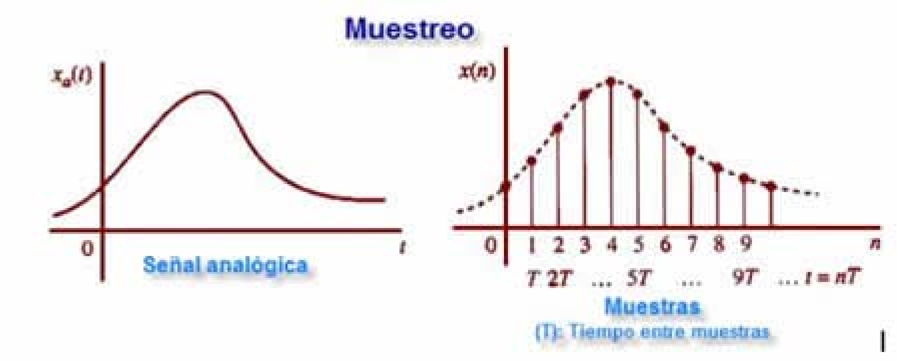
¿Señal?

SEÑAL: Variación de una magnitud física en el tiempo

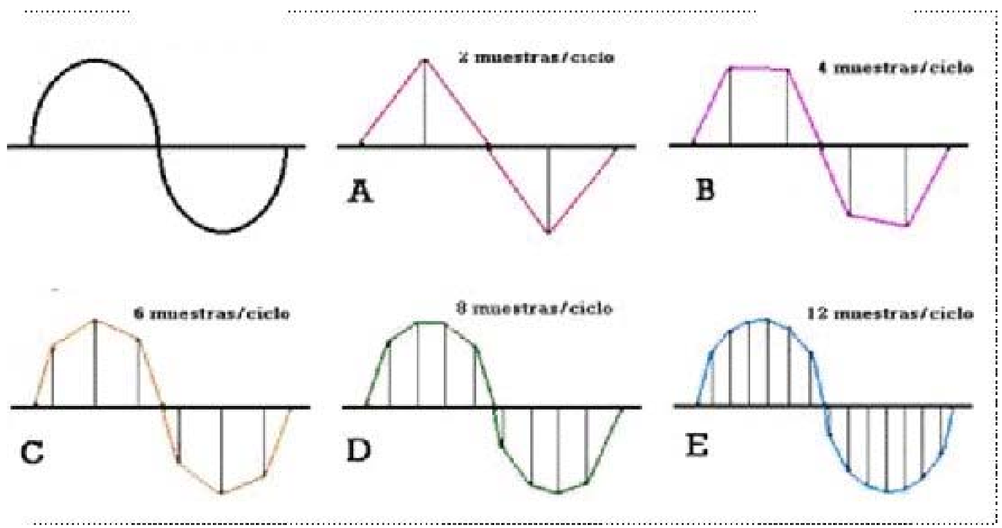


Señal analógica

MUESTREO

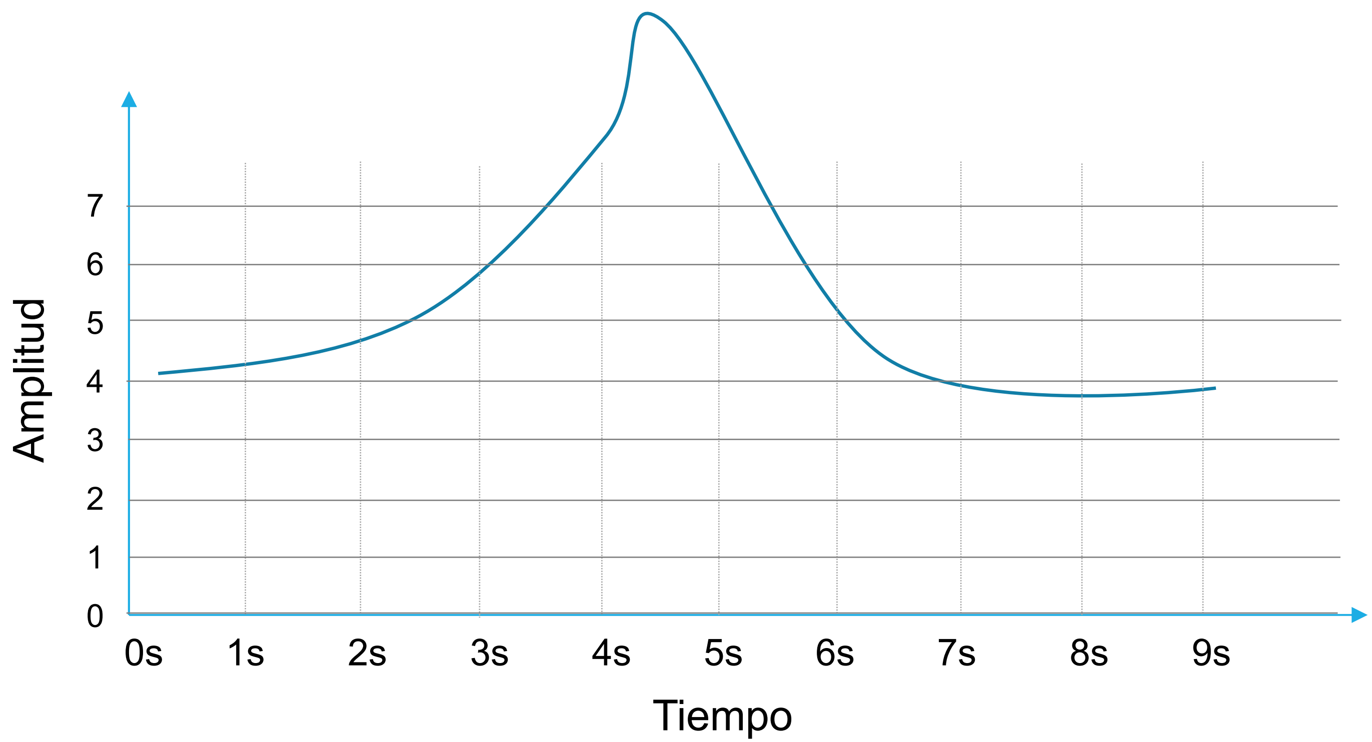


Muestreo
$x_a(t)$
0
t
Señal analógica
$x(n)$
0 1 2 3 4 5 6 7 8 9
n
T 2T … 5T … 9T … t = nT
Muestras
(T): Tiempo entre muestras

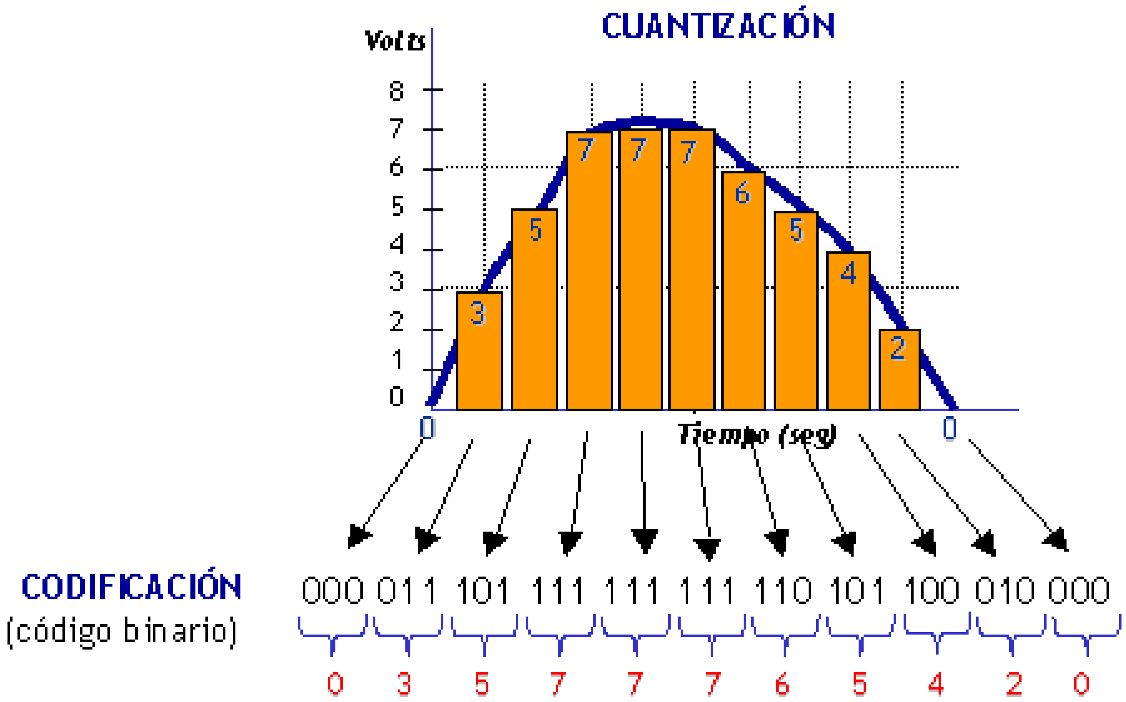


2 muestras/ciclo
A
4 muestras/ciclo
B
6 muestras/ciclo
C
8 muestras/ciclo
D
12 muestras/ciclo
E

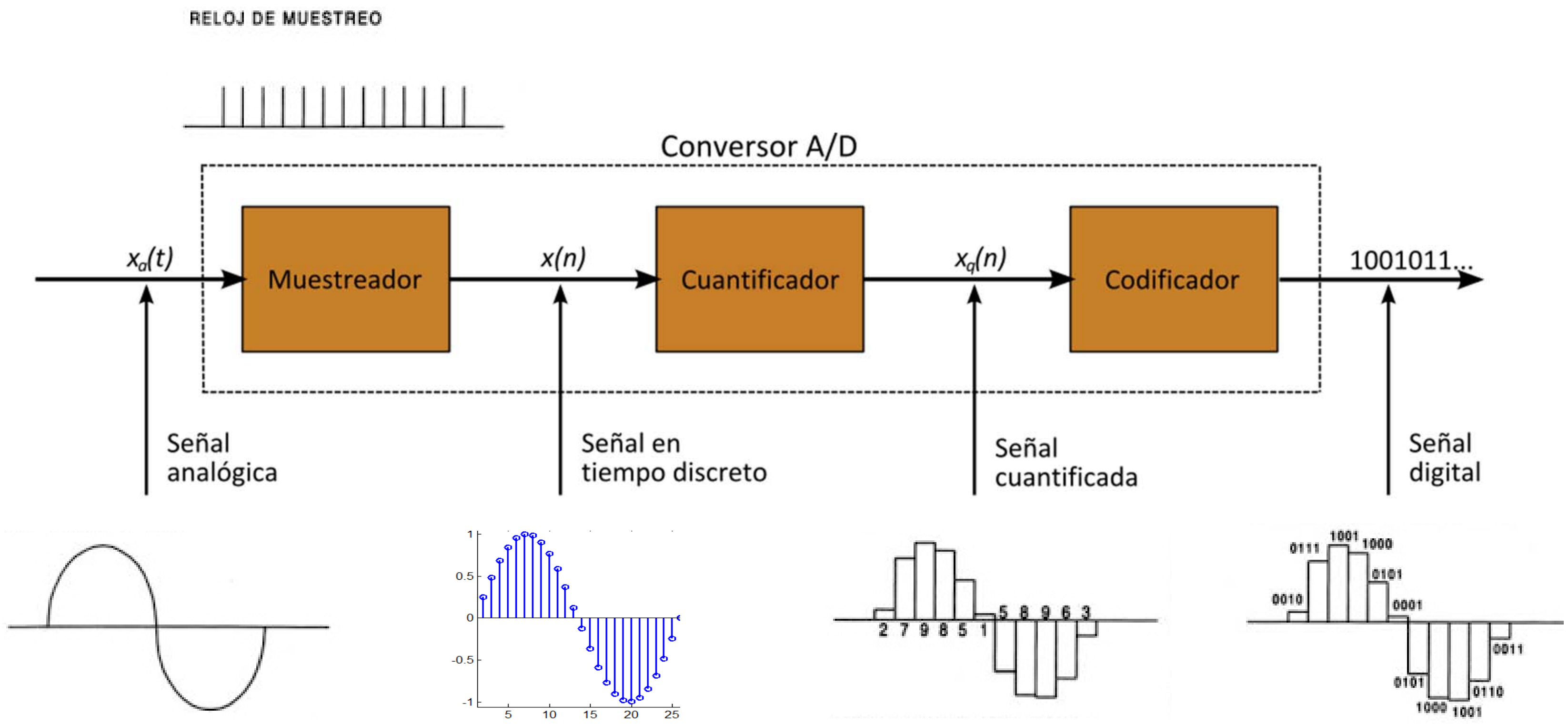
CUANTIZACIÓN



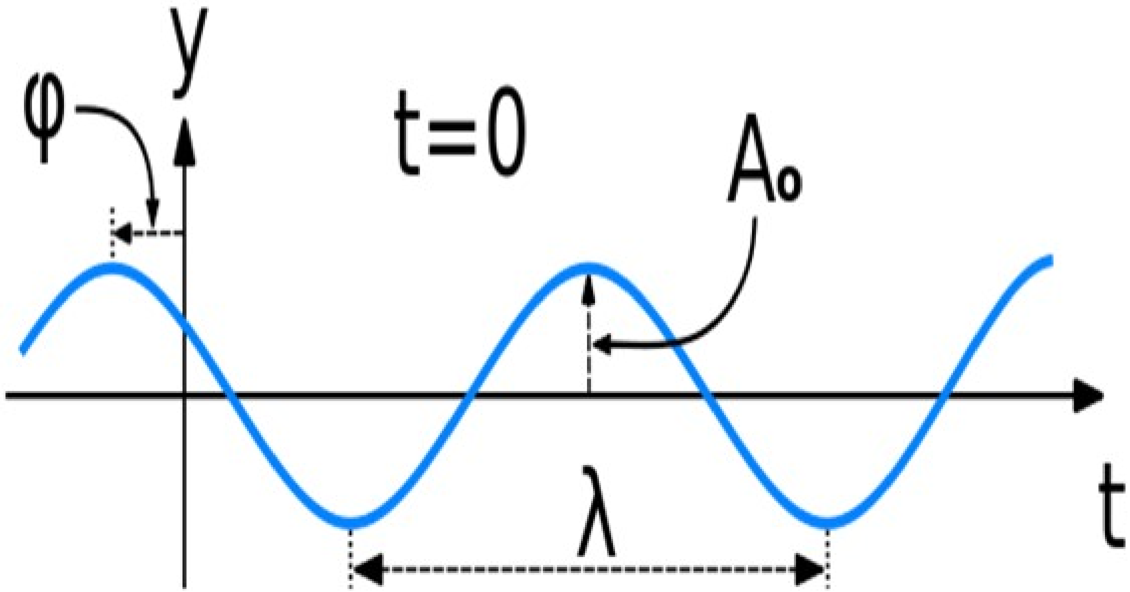
CUANTIZACIÓN/DIGITALIZACIÓN



CONVERSIÓN ANALOGICO/DIGITAL (A/D)

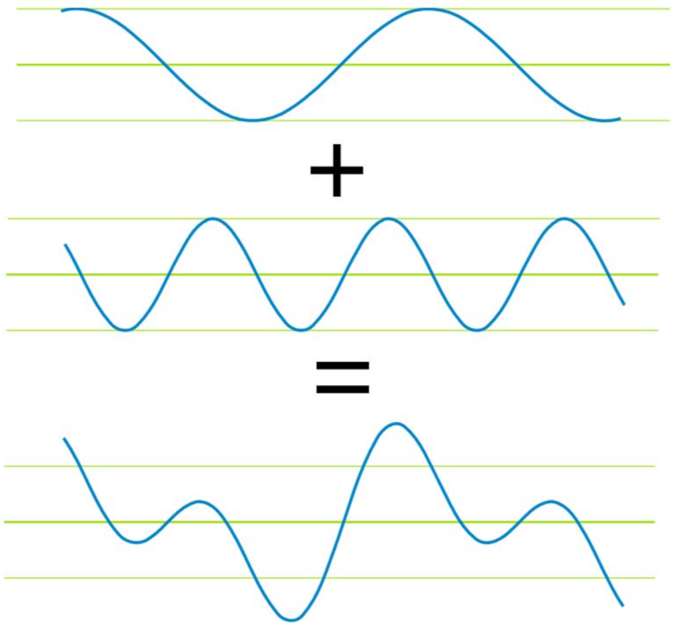


SINUSOIDAL



$$Y = A \operatorname{sen}(\omega t + \varphi)$$

SUPERPOSICIÓN DE ONDAS



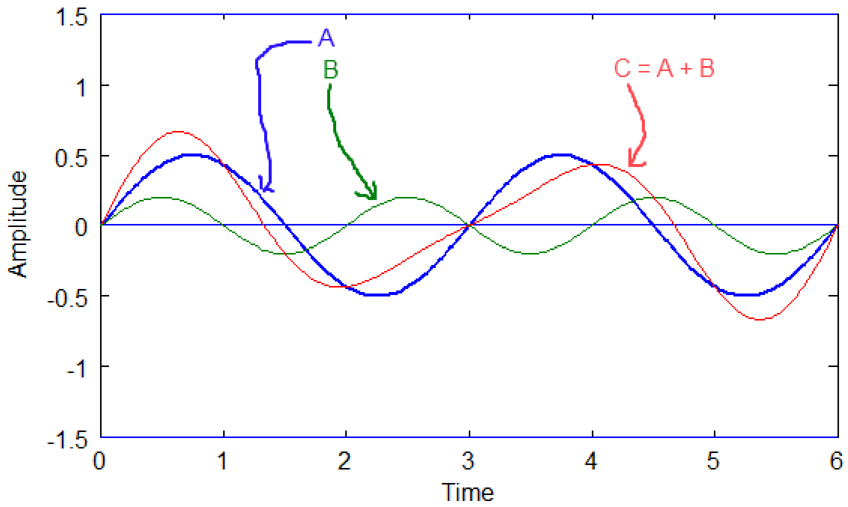
$$Y=\sum A_i \, \mathrm{sen}(\omega_i t+\varphi_i)$$

FOURIER

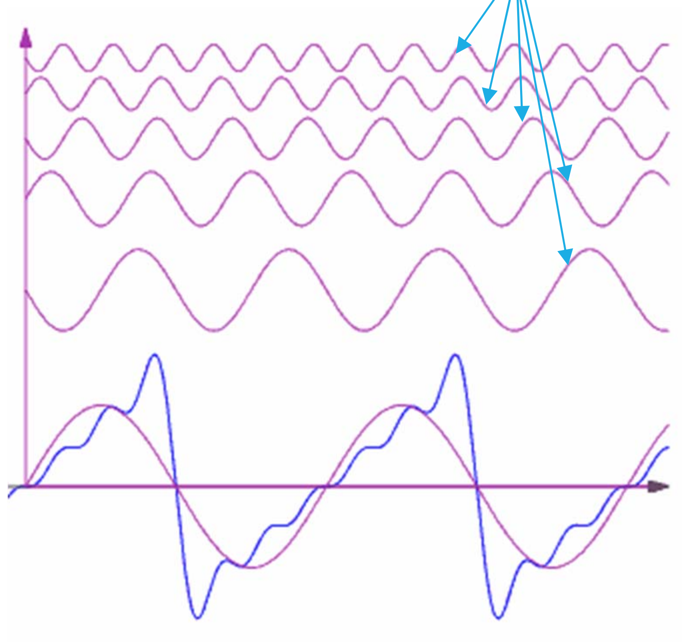
Teorema:

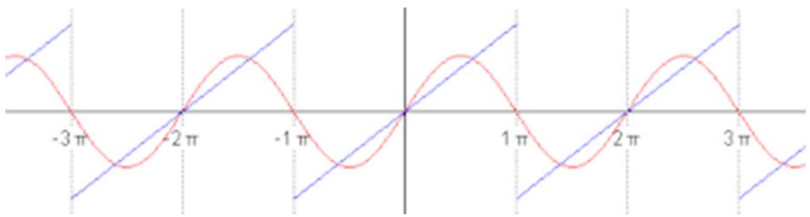
Cualquier forma de onda (señal) periódica se puede descomponer en una suma finita de senoidales.

SUPERPOSICIÓN DE ONDAS

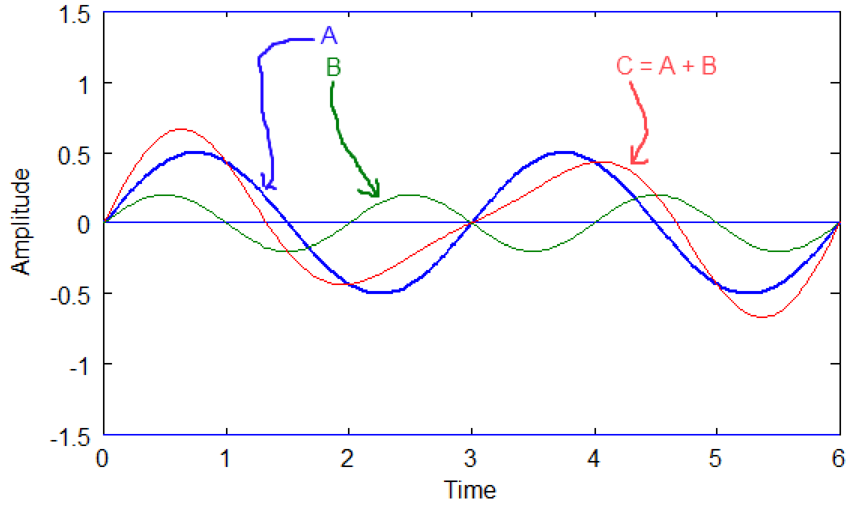


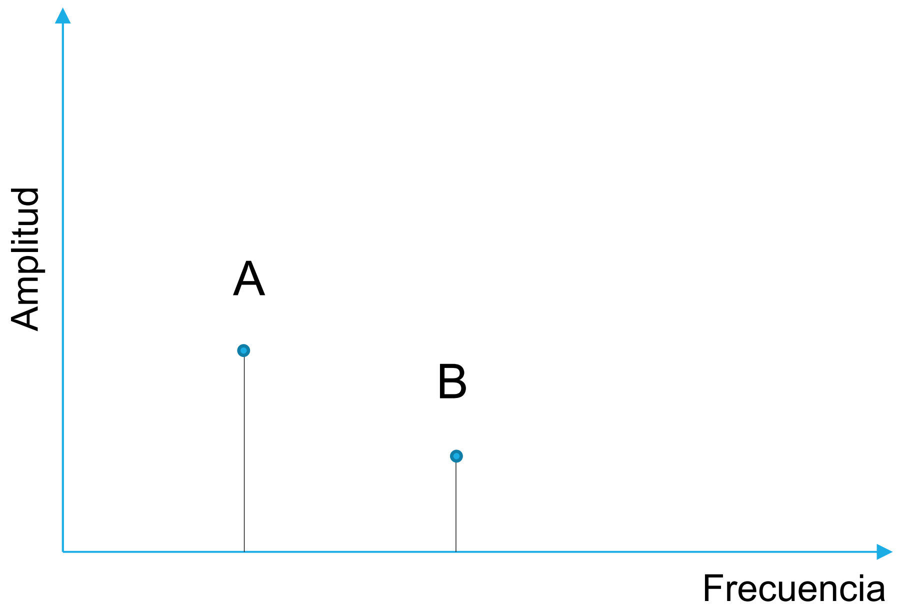
Componentes frecuenciales





ESPACIO FRECUENCIAL (ESPECTRO)





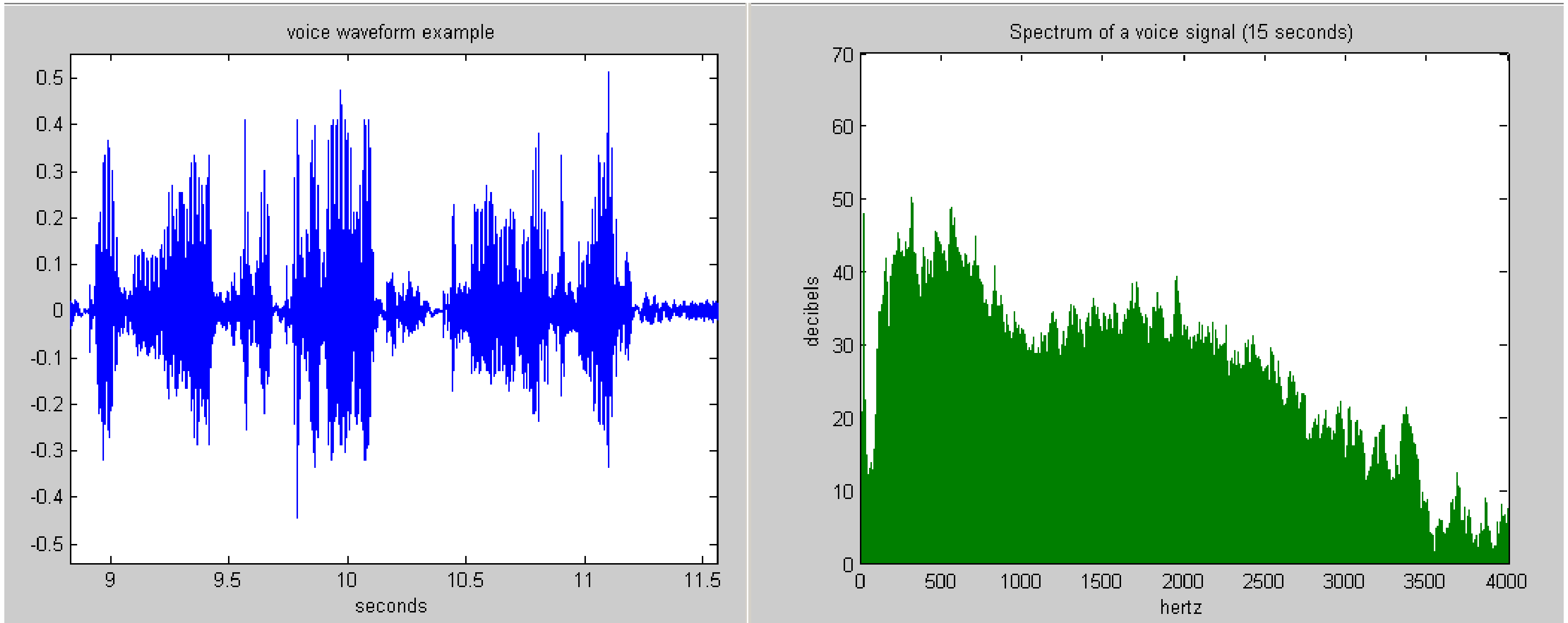
Espectro de C

FOURIER II

Teorema:

Cualquier señal se puede descomponer en una serie de senoidales (que puede ser infinita).

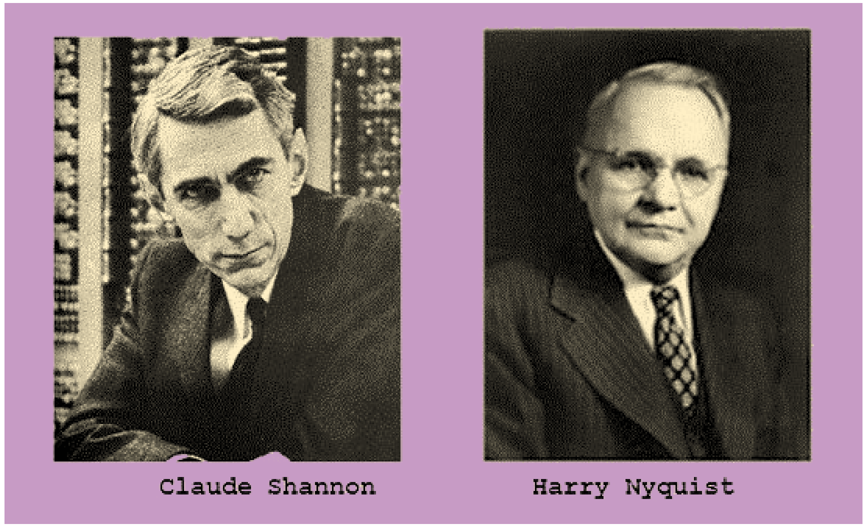
ESPACIO FRECUENCIAL CONTINUO



NYQUIST-SHANNON

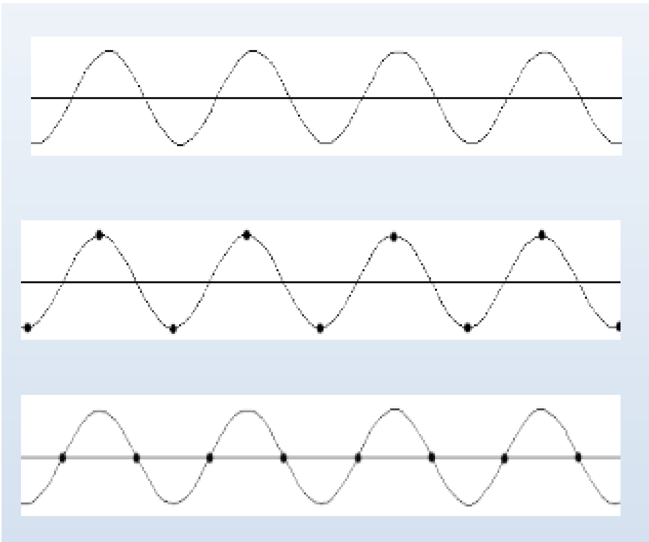
Teorema de muestreo de Nyquist-Shannon:

La reconstrucción exacta de una señal periódica continua en banda base a partir de sus muestras, es matemáticamente posible si la señal está limitada en banda y la tasa de muestreo es superior al doble de su ancho de banda (doble de la frecuencia de su componente frecuencial de mayor frecuencia).

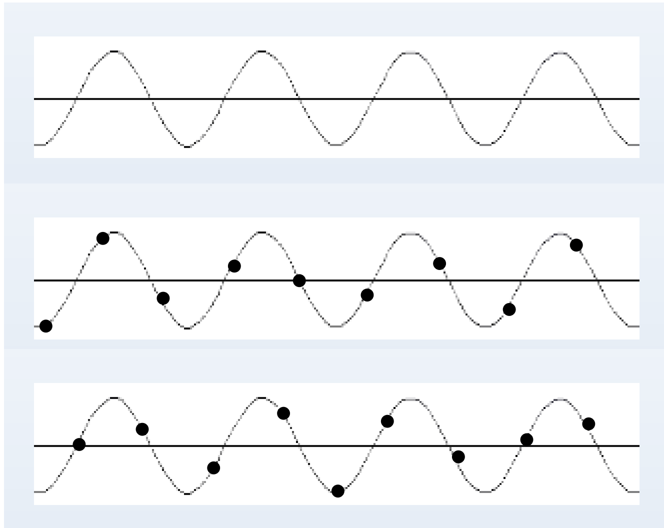


NYQUIST-SHANNON

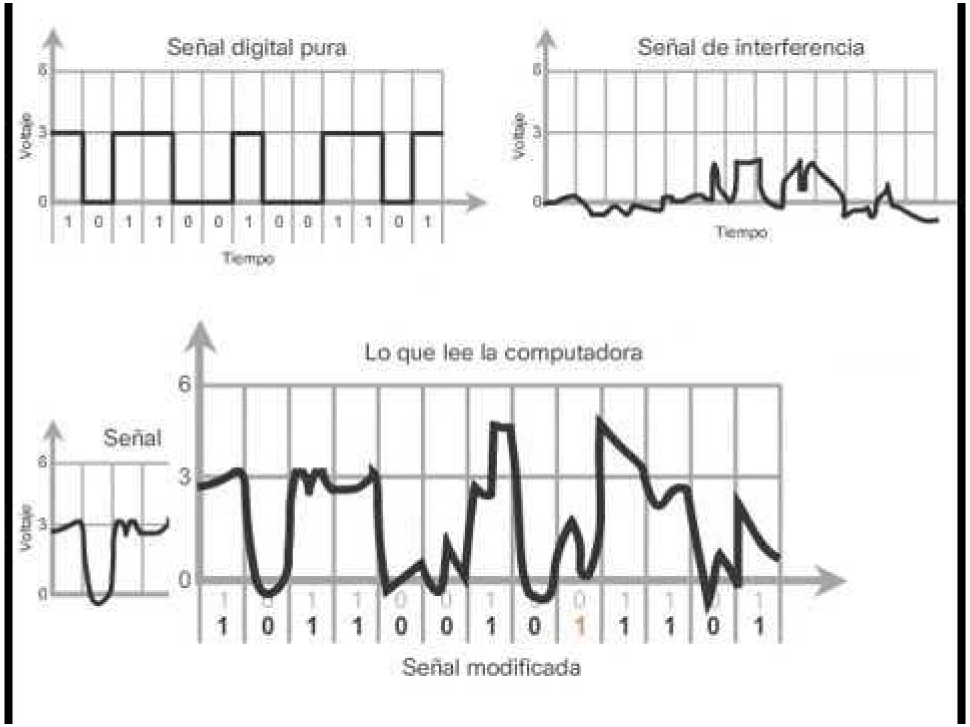
Muestreando al límite =2f



Muestreando >2f



RUIDO



Señal digital pura
Voltaje
Tiempo
Señal de interferencia
Señal
Lo que lee la computadora
1 0 1 1 0 0 1 0 1 1 1 0 1
Señal modificada

REPRESENTACIÓN DE INFORMACIÓN ANALÓGICA

La precisión de la digitalización depende de:

- Frecuencia de muestreo:
  - Determina el ritmo al que se toman las muestras
  - Se mide en muestras/seg para sonido y muestras/pulgada para imágenes
- Precisión de la escala de traslación:
  - Para cada muestra se realiza un cambio de escala
  - La precisión de ésta depende del número de bits empleados para la traslación
  - Con n bits se puede representar 2^n niveles de señal

El producto de los 2 factores proporciona la cantidad de información digitalizada

REPRESENTACIÓN DE SONIDO

- Se capta por medio de un micrófono que produce una señal analógica
- Además, suele amplificarse para encajarla entre unos valores máximo y mínimo (por ejemplo, -5V y +5V)
- Posteriormente, se aplica el *muestreo*:
  - Selecciona muestras de la señal a una frecuencia F_s
  - Por tanto cada $T_s = 1/F_s$ segundos se dispone de un valor de la señal
- Simultáneamente al muestreo las muestras se digitalizan con un *conversor A/D*

REPRESENTACIÓN DE SONIDO

Después de este proceso:

- La señal de sonido queda representada por una secuencia de valores, por ejemplo, de 8 bits

Características de distintas señales de audio:

	Nº de bits/ muestra (por canal)	Frecuencia muestreo (F_s, KHz)	Período de muestreo (T_s, μseg)
PCM Teléfono	8	8	125
Calidad telefónica	8	11,025	90,7
Radio	8	22,05	45,4
CD	16	44,1	22,7

REPRESENTACIÓN DE SONIDO

Cálculo capacidad necesaria para almacenar una señal:

- Muestras: $N = F_s * t$
- Bits por canal: $B_c = N *$ Bits por muestra
- Capacidad total: $C =$ Nº canales $* B_c$

Ejemplo: Calcular la capacidad necesaria (en Bytes) para almacenar 1 minuto de señal de audio estereofónico con calidad CD

- $N = F_s * t =$ 44.100 muestras/seg * 60 segundos = 2.646.000 muestras
- $B_c =$ 2.646.000 muestras * 2 Bytes/muestra = 5.292.000 Bytes
- $C = 2 * 5.292.000 = 10.584.000$ Bytes ≈ 10MB

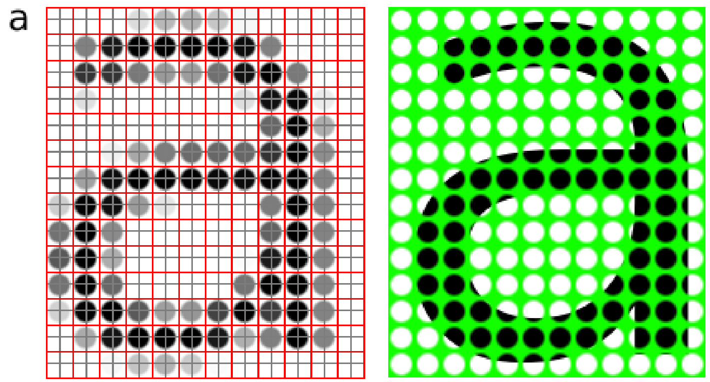
REPRESENTACIÓN DE IMÁGENES

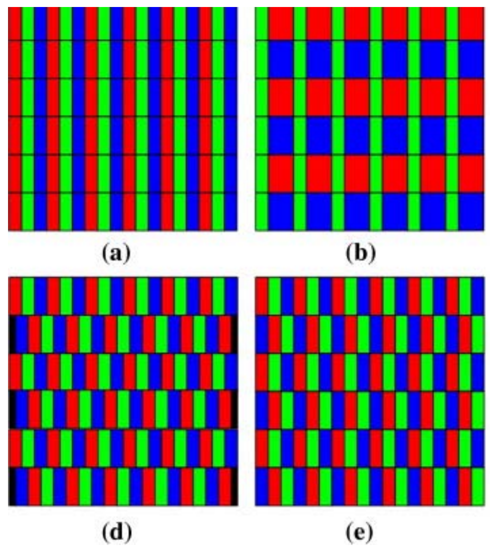
- Las imágenes se obtienen por periféricos como por ejemplo escáneres, cámaras digitales y cámaras de video
- Existen sistemas de codificación de imágenes muy diversos: BMP, TIFF, JPG, GIF, PNG, etc.
- Formas básicas de representar imágenes:
  - Mapas de bits
  - Mapas de vectores

REPRESENTACIÓN DE IMÁGENES

Mapas de bits:

- Imagen compuesta por puntos
- A cada punto se le asocia un atributo:
  - Nivel de gris (imagen en blanco y negro)
  - Nivel de color (imagen en color)
- Para almacenar una imagen influyen 2 factores:
  - Número de puntos
  - Código de atributo (color) asociado a cada punto
- Como no es posible almacenar infinitos puntos:
  - Se divide la imagen en una fina retícula (*elementos de imagen* o ***pixels***)
  - A cada punto se le asigna como atributo el nivel de gris o color medio





REPRESENTACIÓN DE IMÁGENES

Mapas de bits:

- Resolución de imagen (determina la calidad de la imagen):
  - (nº elementos por línea) x (nº elementos por columna)
- La imagen de una fotografía típica también se forma por puntos:
  - Con una resolución de 1280 x 1024 pixels el ojo humano la considera continua
- Para imágenes con calidad fotográfica se suelen usar 300 puntos por pulgada (2,54 cm) y 24 bits por cada punto
- En este caso la información almacenada es:

  $(300 * 300 * 2^3 * 3) / (2{,}54 * 2{,}54) = 334.800,\ 6696$ bits/cm^2 $\approx$ 326,954 Kbits/cm^2 $\approx$ 0,3193 Mbits/cm^2

REPRESENTACIÓN DE IMÁGENES

Ejemplo 1: Calcular la capacidad de memoria que ocupará una imagen en blanco y negro con resolución de 640 * 350 elementos de imagen (pixels) y con 16 niveles de grises

C = (640 * 350) * 4 bits = 896.000 bits = 875 Kbits

Ejemplo 2: Obtener la capacidad de memoria de una imagen en color con una resolución XGA y con 256 niveles para cada color básico

C = (1.024 * 768) * 3 bytes = 2.359.296 bytes = 2,304 KB = 2,25 MB

Ejemplo 3: Una imagen de 100*100 puntos ocupa 5.000 bytes, ¿cuál es el máximo número de colores que puede tener?

100*100 = 10.000 puntos

5.000 bytes / 10.000 puntos = 0,5 bytes/punto

0,5 bytes → 4 bits/punto → 2^4 = 16 colores

REPRESENTACIÓN DE IMÁGENES

Mapas de vectores:

- Descomponen la imagen en un conjunto de objetos: líneas, polígonos, textos, etc. con sus atributos (grosor, color, etc.)
- Los objetos son modelados mediante vectores y ecuaciones matemáticas que determinan
  - La forma
  - La posición
- Adecuada para gráficos de tipo geométrico, no para imágenes reales
- Genera archivos que ocupan menos espacio que los mapas de bits
- Más fáciles de procesar y reescalar