Fundamentos de Computadores

Representación Binaria

Ingeniería Técnica en Informática de Sistema
E.T.S.I. Informática
Universidad de Sevilla

Contenidos

- Introducción
- Definiciones
- Números naturales
- Números enteros
- Números reales/racionales
- Códigos binarios
- Imágenes
- Sonido

Introducción
Codificación digital

- Los circuitos digitales trabajan con señales bivaluadas: 0 y 1
- Los computadores se emplean para almacenar todo tipo de información:
  - números enteros, reales, texto, gráficos, audio, video, etc.
- Esta información ha de reducirse a su representación mediante 0 y 1

Definiciones

- BIT (b) (BInary digiT): dígito binario. Símbolo que puede ser 0 ó 1. Es la unidad mínima de información.
- PALABRA: conjunto de "n" bits. Típicamente, 8, 16, 32 ó 64.
- BYTE (B): Palabra de 8 bits
- KILOBYTE (KB): 1024 bytes
- MEGABYTE (MB): 1024 KB = 1048576 B ~ 1000000B
- GIGABYTE (GB): 1024 MB. A veces se usa como 1000 MB.

Número naturales. Sistemas de numeración. Bases

- El sistema decimal común es un sistema de numeración posicional que emplea 10 símbolos y donde la base es 10:
  - Símbolos: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

$$1327 = 1 \times 10^3 + 3 \times 10^2 + 2 \times 10^1 + 2 \times 10^0$$

Pesos:	1000	100	10	1	
Símbolos:	1	3	2	7	Suma
Valor:	1000	300	20	7	1327

Número naturales. Base 2

- Los sistemas digitales pueden representar de forma "natural" números en base 2, usando los símbolos {0,1}
- Ej: 1101

$$1101 = 1\times 2^3 + 1\times 2^2 + 0\times 2^1 + 1\times 2^0$$

Pesos:	8	4	2	1	
Símbolos:	1	1	0	1	Suma
Valor:	8	4	0	1	→ 13

Números naturales. Base “b”

- Lo mismo aplicado a una base genérica “b”
  - x: magnitud,   b: base
  - n: número de cifras,   {xi}: cifras

$$x = x_{n-1} \times b^{n-1} + ... + x_1 \times b^1 + x_0 \times b^0$$

- Mayor número representable con n cifras: b^n-1
- El cambio de base b a base 10 se realiza aplicando directamente la fórmula anterior con las cifras del número en base b.

Números naturales. Cambio de base 10 a base “b”

- El cambio de base 10 a una base cualquiera b puede realizarse dividiendo sucesivamente la magnitud por la base y extrayendo los restos

$$123_{(10} \longrightarrow 234_{(7}$$

$$\begin{array}{lll|l}
1 & 2 & 3 & 7 \\ \cline{4-4}
 & 5 & 3 & 1\;7 \;\; \begin{array}{|l} 7 \\ \hline \end{array} \\
 & & \boxed{4} & \boxed{3} \;\; \boxed{2}
\end{array}$$

Números naturales. Bases 8 y 16

- Base 8 (octal):
  - {0, 1, 2, 3, 4, 5, 6, 7}
- Base 16 (hexadecimal):
  - {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}

Números naturales. Bases 8 y 16

- Equivalencias con la base 2
  - $8=2^3$ $16=2^4$

B-2	B-8
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

B-2	B-16
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7

B-2	B-16
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Números naturales. Bases 8 y 16

- Las bases 8 y 16 se emplean como una forma compacta de representar números binarios
- 1 cifra octal -> 3 cifras binarias
- 1 cifra hexadecimal -> 4 cifras binarias

1			2			7			2	→	1272 (8
1	0	1	0	1	1	1	0	1	0		
2		B				A				→	2BA (16

Números enteros. Representación signo-magnitud

signo	magnitud								
0	1	0	1	1	0	1	0	→	+90 (10
1	0	1	0	1	0	0	1	→	-41 (10

- Signo: 0(+), 1(-)
- Representable con n bits: 2^n-1
- Representaciones del “0”: 00000000, 10000000

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

Números enteros
Rep. complemento a 1

- Números positivos (1er. bit a 0)
  - se representa el número en base 2
  - Ej: $21_{10} = 00010101_{s\text{-}m}$
- Números negativos (1er. bit a 1)
  - se representa el opuesto con todos los bits complementados.
  - Ej: $-21_{10} = 11101010_{s\text{-}m}$
- Representaciones del "0": 00000000, 11111111

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

Números enteros
Representación en exceso

- Se representa en base 2 el resultado de sumar al número el valor del “exceso” o “sesgo”.
- El resultado de sumar el “exceso” debe ser un entero positivo. Esto define el rango de números representables.
- Ej: exceso 2^{n-1} (números de n bits, ej: 8 bits)
  - $-35_{10} \rightarrow -35+128 = 93 = 01011101_{exc\text{-}128}$

$$-2^{n-1} \leqslant x \leqslant 2^{n-1}-1$$

Números enteros
Rep. Complemento a 2

$$x = -x_{n-1} \times b^{n-1} + \ldots + x_1 \times b^1 + x_0 \times b^0$$

					Suma
Pesos:	-8	4	2	1	
Símbolos:	1	1	0	1	
Valor:	-8	4	0	1	→ -3

- El primer bit indica el signo: 0(+), 1(-)
- Una sola representación del cero: 00000...0

$$-2^{n-1} \leqslant x \leqslant 2^{n-1} - 1$$

Números enteros. Ca2: extensión del signo

-8	4	2	1		
1	1	0	1		
-8	4	0	1	→	-3

-16	8	4	2	1		
1	1	1	0	1		
-16	8	4	0	1	→	-3

-128	64	32	16	8	4	2	1		
1	1	1	1	1	1	0	1		
-128	64	32	16	8	4	0	1	→	-3

- Al extender el número de bits de un número codificado en Ca2, los bits extendidos toman todos el mismo valor que el antiguo bit de signo.

Números enteros. Ca2: Propiedad 1

- Si en la representación en Ca2 de una cantidad entera x se complementan todos los bits y, tratando el resultado como un número binario sin signo, se le suma 1, el resultado es la representación en Ca2 de -x

Números enteros. Ca2: Propiedad 2

- Si las representaciones en Ca2 de dos cantidades enteras x e y se suman, tratándolas como enteros binarios sin signo y despreciando el posible acarreo, el resultado es la representación en Ca2 de la cantidad x+y, salvo que se produzca desbordamiento.

El mismo sumador de binarios naturales sirve para enteros representados en Ca2

Números enteros. Ca2: Propiedad 3

- (Regla de desbordamiento): Si dos cantidades binarias representadas en Ca2, ambas con el mismo signo, se suman tratándolas como enteros binarios sin signo, se produce desbordamiento si el signo del resultado, interpretado en Ca2 es distinto al signo de las cantidades sumadas.

Números enteros
Ca2: Ejemplos

```
 1001 = -7        1100 = -4        0011 = +3
 0101 = +5        0100 = +4        0100 = +4
-----            -----            -----
 1110 = -2       10000 =  0        0111 = +7


 1100 = -4        0101 = +5        1001 = -7
 1111 = -1        0100 = +4        1010 = -6
-----            -----            -----
11011 = -5        1001 = -7       10011 = +3
```

¡Desbordamiento!

Números enteros. Resumen

x	s-m	Ca1	Ca2	exc. 2^{n-1}
-8	-	-	1000	0000
-7	1111	1000	1001	0001
-6	1110	1001	1010	0010
-5	1101	1010	1011	0011
-4	1100	1011	1100	0100
-3	1011	1100	1101	0101
-2	1010	1101	1110	0110
-1	1001	1110	1111	0111
0	0000/1000	0000/1111	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

Números reales. Representación en punto fijo

- En muchas ocasiones, es necesario almacenar y operar con números que no son enteros.
- Una opción para representar números con parte fraccionaria es la Notación en Punto Fijo:
  - número fijo de bits para representar la parte fraccionaria.
- Representación aproximada: error de aproximación.

Números reales. Representación en punto fijo

$$x = x_{n-1} \times b^{n-1} + \ldots + x_0 \times b^0 + x_{-1} \times b^{-1} + \ldots + x_{-m} \times b_{-m}$$

$$\overbrace{0\quad 1\quad 1\quad 0\quad 1}^{\text{parte entera}} \bullet \overbrace{1\quad 0\quad 1}^{\text{parte decimal}}$$

- No permite representar números muy pequeños o muy grandes
- No es flexible

Números reales. Punto fijo: cambio de base

- Base b a base 10:
  - Directamente. Basta con hacer las operaciones en base 10.

$$10{,}101_2 = 1\times 2^{1} + 0\times 2^{0} + 1\times 2^{-1} + 0\times 2^{-2} + 1\times 2^{-3}$$
$$2 + 1/2 + 1/8 = 2{,}875_{10}$$

- Base 10 a base b:
  - Parte entera: como para números naturales
  - Parte decimal: multiplicando sucesivamente por la base y tomando la parte entera

Números reales. Punto fijo: cambio de base

- Ejemplo: $3,27_{(10}$
  - $3_{(10} = 11_{(2}$
  - `0,27 x 2 = 0,54 -> "0"`
  - `0,54 x 2 = 1,08 -> "1"`
  - `0,08 x 2 = 0,16 -> "0"`
  - `0,16 x 2 = 0,32 -> "0"`
  - `0,32 x 2 = 0,64 -> "0"`
  - `0,64 x 2 = 1,28 -> "1"`
  - `...`
- $3,27_{(10} = 11,010001_{(2}$

Números reales.
Rep. en punto flotante

$$1.23 \times 10^{12} \qquad\qquad x = M \times B^{E}$$

- Equivalente a la notación científica decimal:
  - M: mantisa, B: base, E: exponente
- Flexibilidad: permite representar números muy grandes y muy pequeños
- Introduce error de cuantización
  - La precisión depende del valor del número representado: a mayor valor, menor la precisión.

Números reales.
Notación IEEE-754 (parcial)

0	1 8	9 31
1	8	23
signo	exponente (E*)	mantisa (M*)

- Base: 2
- Signo (1 bit): 0 -> +, 1 -> -
- Exponente (8 bits): sesgado, con un sesgo de 127
  - E*: valor almacenado E*=1...254
  - E: exponente real E=E*-127=-126...127

Números reales. Notación IEEE-754 (parcial)

0	1 … 8	9 … 31
1	8	23
signo	exponente (E*)	mantisa (M*)

- Mantisa (23 bits): normalizada, parte entera = 1
  - M = 1,bbb...b
  - M* = bbb...b (23 bits)
  - M* ≠ 000...0

Números reales. Notación IEEE-754 (parcial)

- Mayor número representable: $(2-2^{-23})\times 2^{127}$
- Menor número representable: $-(2-2^{-23})\times 2^{127}$
- Menor número positivo representable: $(1+2^{-23})\times 2^{-126}$
- Mayor número negativo representable: $-(1+2^{-23})\times 2^{-126}$

- Algunos casos especiales:
  - cero: E*=0, M*=0
  - Infinito: E*=255, M*=0 (s=0 -> +Inf, s=1 -> -Inf)

Números reales. Punto flotante. Paso a base 10

0	1 ... 8	9 ... 31
1	8	23
signo	exponente (E*)	mantisa (M*)

- Se obtiene M a partir de M* y el signo
- Se obtiene E a partir de E*:
  - E = E*-127
- Se hacen las operaciones

$$x = M \times 2^{E}$$

Números reales. Punto flotante. Paso a base 10

- Ejemplo:

  0 10010100 10100010000000000000000

  - signo: 0 -> +
  - Mantisa (M) = $1,1010001_{(2} = 1,6328125_{(10}$
  - $E^* = 10010100_{(2} = 148_{(10}$; $E = E^* - 127 = 21$

$$x = 1,6328125 \times 2^{21} = 3424256$$

Números reales. Punto flotantes. Paso desde base 10

$$x = M \times 2^{E}$$

- Se obtiene una estimación del exponente: E'

$$E' = \log_2 x$$

- Se elige como exponente (E) la parte entera de E'

$$E = ent(E')$$

- Se calcula el valor de la mantisa (M).

$$M = \frac{x}{2^{E}}$$

- Se calculan E* y M* y se pasan a binario.
- Se construye la palabra binaria.

Números reales. Punto flotantes. Paso desde base 10

Ejemplo: +3424256

$$E' = \log_2 3424256 = 21.707$$

$$E = ent(21.707) = 21$$

$$M = \frac{3424256}{2^{21}} = 1.6328125$$

- signo + : “0”
- E* = 127+21 = 148 = $10010100_{(2}$
- M = 1,6328125 = $1.1010001_{(2}$
- M* = “101001000...00”

0 10010100 10100100000000000000000

Códigos binarios

- Asignan palabras binarias a “símbolos” que se desean representar.
- Los “símbolos” son números, letras o cualquier otra entidad que se quiera representar.
- Normalmente, cada código emplea palabras binarias de longitud fija (p.ej. 8 bits).
- La asignación de palabras binarias se suele realizar para obtener alguna propiedad.
- A veces, no todas las palabras binarias tienen asignado un símbolo.

Códigos binarios. Binario natural

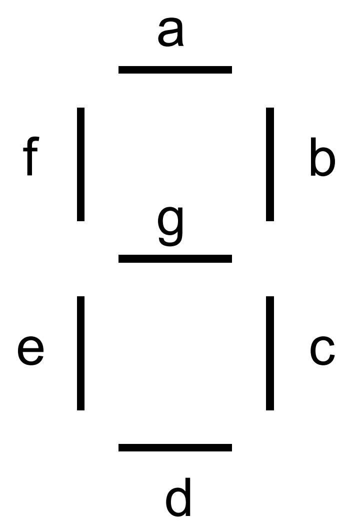
- Asigna códigos binarios a números enteros positivos.
- La palabra asignada corresponde a la representación del número en base 2.
- Ejemplo: código binario natural de 3 bits.

Sim.	Cod.
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Códigos binarios. Códigos BCD (Binary Coding Decimal)

- Asignan códigos binarios a las 10 cifras decimales.

Cifra	natural	exceso-3	2 de 5	7 segm. abcdefg
0	0000	0011	00011	1111110
1	0001	0100	00101	0110000
2	0010	0101	00110	1101101
3	0011	0110	01001	1111001
4	0100	0111	01010	0110011
5	0101	1000	01100	1011011
6	0110	1001	10001	1011111
7	0111	1010	10010	1110000
8	1000	1011	10100	1111111
9	1001	1100	11000	1110011



Códigos binarios. Código Gray

- Asigna códigos binarios a números enteros positivos.
- La asignación se realiza de modo que palabras consecutivas se diferencia únicamente en 1 bit (distancia 1)
- El código de n bits se construye de forma simétrica a partir del código de n-1 bits

Códigos binarios. Código Gray

Número	1 bit	2 bits	3 bits	4 bits
0	0	00	000	0000
1	1	01	001	0001
2		11	011	0011
3		10	010	0010
4			110	0110
5			111	0111
6			101	0101
7			100	0100
8				1100
9				1101
10				1111
11				1110
...				...

Códigos binarios. Código “one hot”

- Asigna palabras con un solo bit a “1” y el resto a “0”
- Ventajas:
  - fácil de interpretar.
  - detección de errores
- Desventaja: requiere un gran número de bits (igual al número de símbolos)

Sim.	Cod.
0	00000001
1	00000010
2	00000100
3	00001000
4	00010000
5	00100000
6	01000000
7	10000000

Códigos binarios. Codificación de texto

- A cada símbolo de texto “carácter” se asigna un número que se almacena en binario.
- Además de símbolos gráficos se incluyen símbolos de control: nueva línea, nueva página, fin de fichero, etc.
- Existen varios asignamientos llamados "codificaciones".

Códigos binarios. Codificación de texto. Codificaciones

- ASCII (American Standard Code for Information Interchange) (ISO-646-IRV): 7 bits (128 símbolos), el más extendido, pensado para el Inglés
- ISO-8859-1 (Latin 1): Extensión del ASCII, 8 bits (256 símbolos), incluye las lenguas de Europa occidental y otras.
- ISO-8859-15 (Latin 9): Modificación del Latin 1 para incorporar el símbolo del EURO y otras actualizaciones.
- ISO-8859-2 a ISO-8859-14: Extensiones del ASCII para diferentes lenguas: cirílico, árabe, griego, hebreo, etc.
- ISO-10646 (Unicode): código de 16 bits que engloba a todas las codificaciones e idiomas conocidos.

ASCII

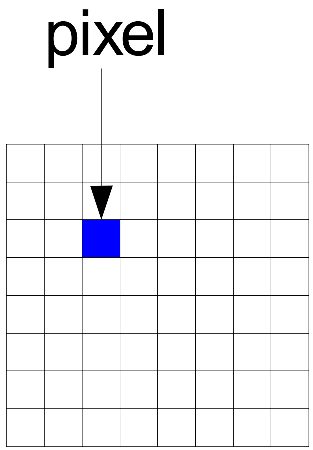
0	00	NUL	32	20	SPC	64	40	@	96	60	`
1	01	SOH	33	21	!	65	41	A	97	61	a
2	02	STX	34	22	"	66	42	B	98	62	b
3	03	ETX	35	23	#	67	43	C	99	63	c
4	04	EOT	36	24	$	68	44	D	100	64	d
5	05	ENQ	37	25	%	69	45	E	101	65	e
6	06	ACK	38	26	&	70	46	F	102	66	f
7	07	BEL	39	27	'	71	47	G	103	67	g
8	08	BS	40	28	(	72	48	H	104	68	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

Extensiones ISO-8859-1 (Latin 1)

160	A0		192	C0	À	224	E0	à
161	A1	¡	193	C1	Á	225	E1	á
162	A2	¢	194	C2	Â	226	E2	â
163	A3	£	195	C3	Ã	227	E3	ã
164	A4	¤	196	C4	Ä	228	E4	ä
165	A5	¥	197	C5	Å	229	E5	å
166	A6	¦	198	C6	Æ	230	E6	æ
167	A7	§	199	C7	Ç	231	E7	ç
168	A8	¨	200	C8	È	232	E8	è
169	A9	©	201	C9	É	233	E9	é
170	AA	ª	202	CA	Ê	234	EA	ê
171	AB	«	203	CB	Ë	235	EB	ë
172	AC	¬	204	CC	Ì	236	EC	ì
173	AD	-	205	CD	Í	237	ED	í
174	AE	®	206	CE	Î	238	EE	î
175	AF	¯	207	CF	Ï	239	EF	ï
176	B0	°	208	D0	Ð	240	F0	ð
177	B1	±	209	D1	Ñ	241	F1	ñ
178	B2	²	210	D2	Ò	242	F2	ò
179	B3	³	211	D3	Ó	243	F3	ó
180	B4	´	212	D4	Ô	244	F4	ô
181	B5	µ	213	D5	Õ	245	F5	õ
182	B6	¶	214	D6	Ö	246	F6	ö
183	B7	·	215	D7	×	247	F7	÷
184	B8	¸	216	D8	Ø	248	F8	ø
185	B9	¹	217	D9	Ù	249	F9	ù
186	BA	º	218	DA	Ú	250	FA	ú
187	BB	»	219	DB	Û	251	FB	û
188	BC	¼	220	DC	Ü	252	FC	ü
189	BD	½	221	DD	Ý	253	FD	ý
190	BE	¾	222	DE	Þ	254	FE	þ
191	BF	¿	223	DF	ß	255	FF	ÿ

Imágenes

- Los ordenadores componen imágenes mediante el dibujo de puntos de distinto color llamados "pixel"
- El color de cada punto se codifica con un número binario de un número determinado de bits.
- El número de bits empleado se llama "profundidad de color".
  - Ej. 8 bits, 16 bits, 24 bits (color verdadero).



Imágenes

Imágenes
Codificación del color

- Un determinado color se forma componiendo tres colores primarios en distintas intensidades:
  - ROJO (R),
  - VERDE (G)
  - AZUL (B)
- Con 24 bits:
  - 8 bits para cada color,
  - 256 valores (0 ... 255)

R G B

0 0 0 negro

255 255 255 blanco

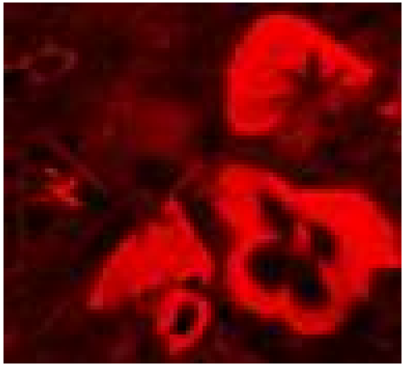
255 0 0 rojo intenso

255 255 0 amarillo

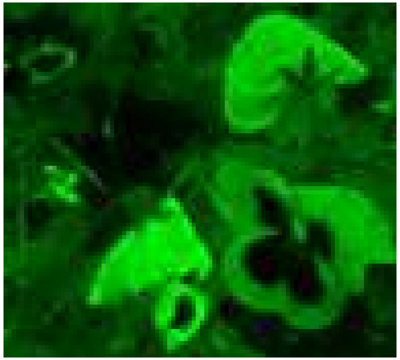
192 192 255 azul claro

255 128 0 naranja

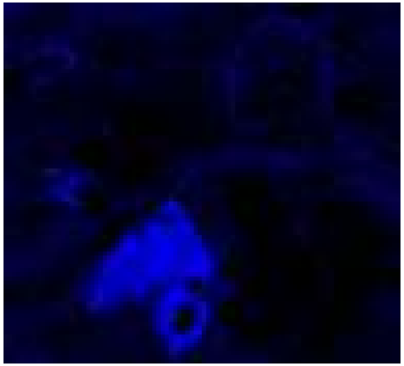
Imágenes. Canales de color



canal rojo

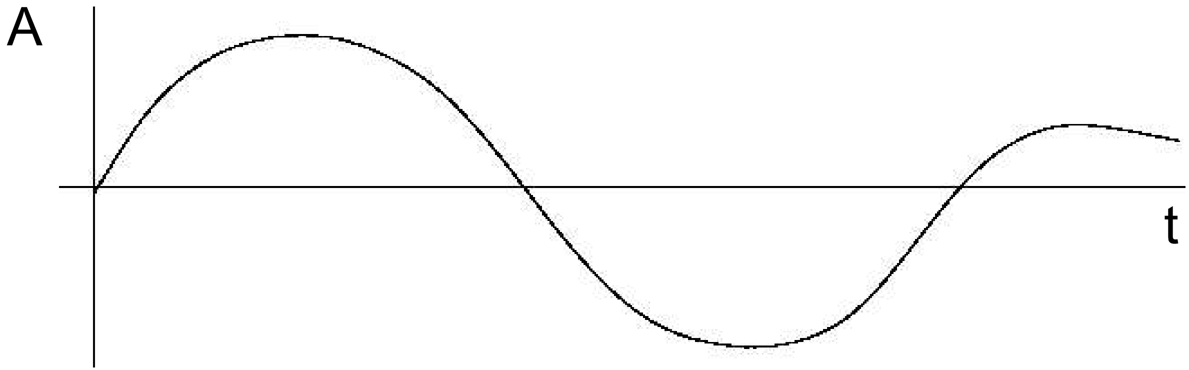


canal verde



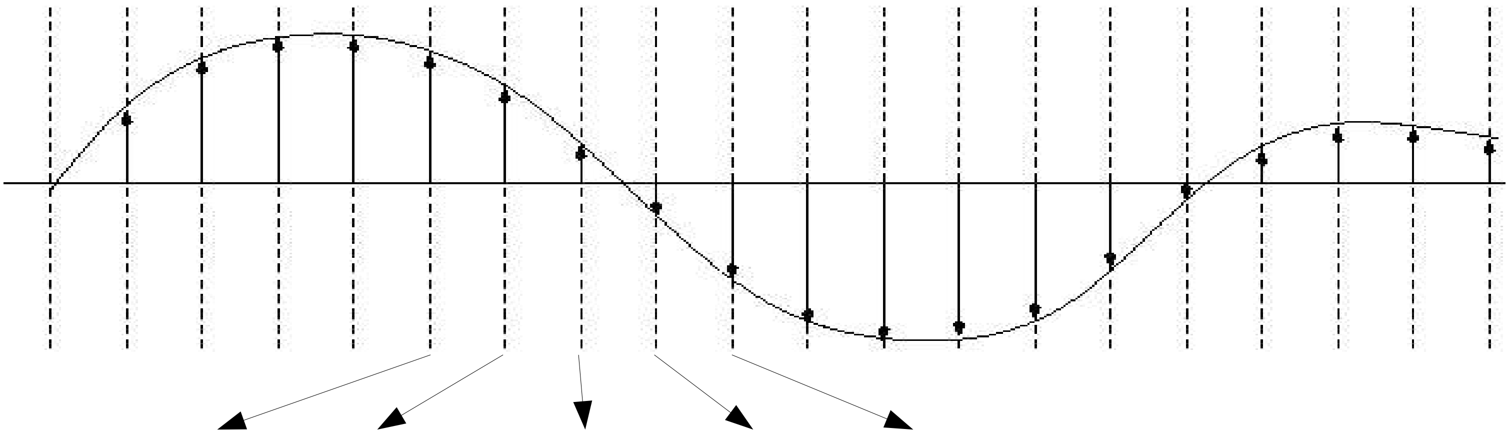
canal azul

Sonido



- El sonido se representa por una curva de amplitud (presión, señal eléctrica, etc.) frente al tiempo.
- Codificación digital:
  - Muestreo: se toma el valor de la señal a intervalos regulares. Frecuencias típicas de muestreo (Hz): 11025, 22050, 44100
  - Cuantización: la amplitud se representa con un número determinado de bits. Ej. 8, 16 bits.

Sonido
Codificación del sonido



. . .   22344   19233   8132   -6984   -15864   . . .

- La calidad será mayor cuanto mayor sea la frecuencia de muestreo y el número de bits por muestra
- Ejemplo: calidad CD: 44100Hz, 16 bits/muestra