




Unidad Didáctica 3.2

Representación de los datos





Sistemas Informáticos
Desarrollo de Aplicaciones Multiplataforma

Sistemas de Numeración

Representación

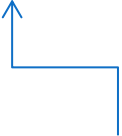
Números Naturales, aquellos mayores o iguales que 0

- ▶ 5 símbolo que representa el número 5
 - ▶ V símbolo que representa el número 5
 - ▶  símbolo que representa el número 5
 - ▶  símbolo que representa el número 5
-
- ▶ Al cambiar un número de base, el número no cambia; cambia su representación

Representación Posicional

- ▶ El número 35729 se descompone como...

$$3 \cdot 10^4 + 5 \cdot 10^3 + 7 \cdot 10^2 + 2 \cdot 10^1 + 9 \cdot 10^0$$


Dígito más significativo


Dígito menos significativo

- ▶ Teorema Fundamental de la Numeración
 - ▶ Si un entero x se representa en base b con i dígitos en la forma $X_{i-1}X_{i-2}\dots X_1X_0$, su valor viene dado por la expresión

$$x = \sum_{j=0}^{i-1} X_j \cdot b^j$$

Paso de base b a decimal (*base 10*)

- ▶ Por el TFN

$$1234_{(5)} = 4 \cdot 5^0 + 3 \cdot 5^1 + 2 \cdot 5^2 + 1 \cdot 5^3 = \mathbf{194}_{(10)}$$

- ▶ Por Método de Ruffini

	1	2	3	4
5		5	35	190
	1	7	38	194 ₍₁₀₎


Algoritmo "De cualquier base a decimal"

```
//n=representación de un entero en una base
//b=base en la que está representado
//devuelve un entero mayor que 0 si la conversión se
//realiza correctamente, y -1 en otro caso
funcion CualquierBaseADecimal (n: cadena, base: entero) : entero
variables:
    p: entero; //posicion
    i: entero;
    resultado: entero = 0;
empieza
    Comprobar que  $2 \leq \text{base} \leq 16$  y
    que n es una representación válida para esa base
    Si no es así, devolver -1 y terminar
    para p desde 0 hasta tamaño(n)-1 hacer
        i = valor del dígito en posicion tamaño(n)-1-p
        resultado = resultado +  $i * \text{base}^p$ 
    fin para
    devolver resultado
termina
```

Paso de decimal (*base 10*) a cualquier base

- ▶ Dividir sucesivamente por la base a la que queramos pasar el número hasta que se obtenga un cociente menor que la base
- ▶ Se recoge el último cociente los restos obtenidos en orden inverso

1	2	3	4		5						
	2	3		2	4	6		5			
		3	4		4	6	4	9		5	
			4			1		4	9		5
								4	1		



$$1234_{(10)} = 14414_{(5)}$$

Algoritmo "De decimal a su representación en cualquier base"

```
//i=el natural a representar en otra base
//b=la base en la que lo queremos representar
//devuelve una cadena no vacía si la conversión
//es correcta, y una vacía si los parámetros
//de entrada no son válidos
funcion DecimalACualquierBase(i:entero, b:entero):Cadena
variables:
    resultado:Cadena
    x:entero
empieza
    Comprobar que la base es válida
        Si no es así, devolver cadena vacía
    Comprobar que i es mayor que 0
        Si no es así devolver "0" en caso de que i sea exactamente 0
        y cadena vacía si es menor que 0
    resultado=""
    x=i //para no perder i
    mientras (x>0) hacer
        concatenar a la izquierda de resultado el dígito que corresponde con x mod b
        x=x/b
    fin mientras
    devolver resultado
termina
```


Paso de base b a base c

- ▶ Realizar siempre paso intermedio pasando por base 10
 - ▶ Excepción: Sistemas cuya base sea potencia de 2
- ▶ Ejercicio
 - ▶ Pasar el número 5324 en base 7 a base 5
 - ▶ Resultado: $30010_{(5)}$

Sistema binario

- ▶ Paso de binario a decimal

- ▶ $11010,011 = 1 \cdot 2^4 + 1 \cdot 2^4 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} + 1 \cdot 2^{-3} = 26,375$

- ▶ Paso de decimal a binario

- ▶ $5,0625 = 101,0001$

- ▶ La parte entera se realiza mediante divisiones sucesivas como se ha visto
 - ▶ La parte decimal se realiza mediante multiplicaciones sucesivas, cogiendo la parte entera de los resultados

$$0,0625 \cdot 2 = \mathbf{0},125$$

$$0,125 \cdot 2 = \mathbf{0},25$$

$$0,25 \cdot 2 = \mathbf{0},5$$

$$0,5 \cdot 2 = \mathbf{1}$$

Sistemas Octal y Hexadecimal

▶ Sistema octal

▶ Utiliza 8 símbolos

▶ {0, 1, 2, 3, 4, 5, 6, 7}

▶ Cada cifra octal son 3 bits en binario

▶ $47 = 100\ 111$

▶ $001\ 101\ 110 = 156$

▶ Sistema hexadecimal

▶ Utiliza 16 símbolos

▶ {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F}

▶ Cada cifra hexadecimal son 4 bits en binario

▶ $3F1 = 0011\ 111\ 0001$

▶ $0101\ 1010\ 0110 = 5A6$

Sistemas de numeración

Decimal	Binario	Octal	Hexadecimal
0	0000	00	0
1	0001	01	1
2	0010	02	2
3	0011	03	3
4	0100	04	4
5	0101	05	5
6	0110	06	6
7	0111	07	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Operaciones Aritméticas

► Suma

$$\begin{array}{r} 110111011 \\ + 100111011 \\ \hline 1011110110 \end{array}$$

► Multiplicación

$$\begin{array}{r} 110100010101 \\ \times \quad \quad 1101 \\ \hline 110100010101 \\ 000000000000 \\ 110100010101 \\ 110100010101 \\ \hline 1010101000010001 \end{array}$$

► Resta

$$\begin{array}{r} 11011001 \\ - 10101011 \\ \hline 00101110 \end{array}$$

► División

$$\begin{array}{r} 101010 \overline{) 110} \\ - \underline{110} \quad 111 \\ 1001 \\ - \underline{110} \\ 0110 \\ - \underline{110} \\ 000 \end{array}$$

Representación interna de los datos

Representación interna de los datos

- ▶ Un S.I. maneja información de todo tipo dándole entrada, salida o procesándola
 - ▶ Precisa mecanismos de representación, almacenamiento y presentación
- ▶ Primer dilema: ¿cómo diferenciar rápidamente lo que son números de lo que es texto, imagen, sonido o video?
- ▶ El ordenador sólo puede representar **información binaria**... es preciso codificar la información del MundoReal™ en binario y utilizar mecanismos para su presentación

Representación interna de los datos

► Unidades de medida

1 bit → unidad mínima de información

1 Byte (B)	8 bits
1 Kilobyte (KB)	1024 B
1 Megabyte (MB)	1024 KB
1 Gigabyte (GB)	1024 MB
1 Terabyte (TB)	1024 GB
1 Petabyte (PB)	1024 TB
1 Exabyte (EB)	1024 PB
1 Zettabyte (ZB)	1024 EB
1 Yottabyte (YB)	1024 ZB

Cuidado con no confundir los bits (b) con los bytes (B)

!!!No es lo mismo Kb que KB!!!

Representación interna de los datos

- Codificación de la información → Necesidad de estándares para que todos nos podamos entender

Textos	<ul style="list-style-type: none">▪ BCD de 6 bits▪ EBCDIC▪ ASCII▪ UNICODE		
Datos Numéricos	Enteros	Dígitos decimales codificados en Binario (BCD)	<ul style="list-style-type: none">▪ Empaquetado▪ Desempaquetado
		Representación Binaria - Coma Fija -	<ul style="list-style-type: none">▪ Módulo y Signo▪ Complemento a 1▪ Complemento a 2▪ Exceso a 2 elevado a N-1
	Reales	Coma Flotante <ul style="list-style-type: none">▪ Notación exponencial▪ Normalización IEEE754	
Sonidos	WAV, MIDI, MP3		
Imágenes	Mapa de Bits	BMP, TIFF, JPEG, GIF, PNG	
	Mapa de Vectores	DXF, IGES, EPS, TrueType	

Caracteres alfanuméricos

- ▶ ASCII o US-ASCII
 - ▶ 7 bits → 127 caracteres
- ▶ ISO 8859-1 o ASCII Extendido para alfabeto latino
 - ▶ 8 bits → 255 caracteres
 - ▶ ISO 8859-15 incluye carácter €
- ▶ UNICODE
 - ▶ 4 bytes (32 bits)
 - ▶ Puede codificar cualquier lengua del planeta
- ▶ UTF-8
 - ▶ Formato Unicode
 - ▶ Longitud variable de 1 a 4 bytes

Caracteres alfanuméricos

TABLA ASCII

Binario	000	001	010	011	100	101	110	111
Hex/Dec	0	1	2	3	4	5	6	7
0000	NUL	DLE	SP	0	@	P	`	p
0	0	16	32	48	64	80	96	112
0001	SOH	DC1	!	1	A	Q	a	q
1	1	17	33	49	65	81	97	113
0010	STX	DC2	"	2	B	R	b	r
2	2	18	34	50	66	82	98	114
0011	ETX	DC3	#	3	C	S	c	s
3	3	19	35	51	67	83	99	115
0100	EOT	DC4	\$	4	D	T	d	t
4	4	20	36	52	68	84	100	116
0101	ENQ	NAK	%	5	E	U	e	u
5	5	21	37	53	69	85	101	117
0110	ACK	SYN	&	6	F	V	f	v
6	6	22	38	54	70	86	102	118
0111	BEL	ETB	'	7	G	W	g	w
7	7	23	39	55	71	87	103	119
1000	BS	CAN	(8	H	X	h	x
8	8	24	40	56	72	88	104	120
1001	HT	EM)	9	I	Y	i	y
9	9	25	41	57	73	89	105	121
1010	LF	SUB	*	:	J	Z	j	z
A	10	26	42	58	74	90	106	122
1011	VT	ESC	+	;	K	[k	{
B	11	27	43	59	75	91	107	123
1100	FF	FS	,	<	L	\	l	
C	12	28	44	60	76	92	108	124
1101	CR	GS	-	=	M]	m	}
D	13	29	45	61	77	93	109	125
1110	SO	RS	.	>	N	^	n	~
E	14	30	46	62	78	94	110	126
1111	SI	US	/	?	O	_	o	DEL
F	15	31	47	63	79	95	111	127

Booleanos (lógica binaria)

- ▶ Dos posibles valores
 - ▶ 1 (Verdadero)
 - ▶ Posibles codificaciones
 - ▶ Todos los bits de la palabra a 1
 - ▶ Solo el bit más significativo
 - ▶ Solo el bit menos significativo
 - ▶ 0 (Falso)
 - ▶ Cualquier cosa que no es 1 (Verdadero)

Operaciones de lógica binaria

▶ NOT

- ▶ Negación (inversión del valor)

▶ AND

- ▶ Multiplicación lógica o intersección

▶ OR

- ▶ Suma lógica o unión

▶ XOR

- ▶ OR exclusiva

▶ XNOR

- ▶ Equivale a “Si y sólo si”

▶ IMPLIES

- ▶ Equivale a “Si... Entonces...”

A	B	$\neg A$	$A \cdot B$	$A + B$	$A \oplus B$	$A \odot B$	$A \rightarrow B$
0	0	1	0	0	0	1	1
0	1	1	0	1	1	0	1
1	0	0	0	1	1	0	0
1	1	0	1	1	0	1	1

Números Enteros

- ▶ Coma fija sin signo (binario puro)
 - ▶ El ordenador tiene n bits para almacenar un número N
 - ▶ Tenemos 2^n números distintos
 - ▶ Desde 0 hasta $(2^n - 1)$
- ▶ Coma fija con signo. Signo magnitud
 - ▶ Reservar uno de los n bits para signo
 - ▶ Tenemos 2^n números distintos
 - ▶ Desde $(-2^{n-1} + 1)$ hasta $(2^{n-1} - 1)$
 - ▶ El número cero tiene dos posibles representaciones
 - ▶ Representación utilizada en IBM 7090

Números Enteros

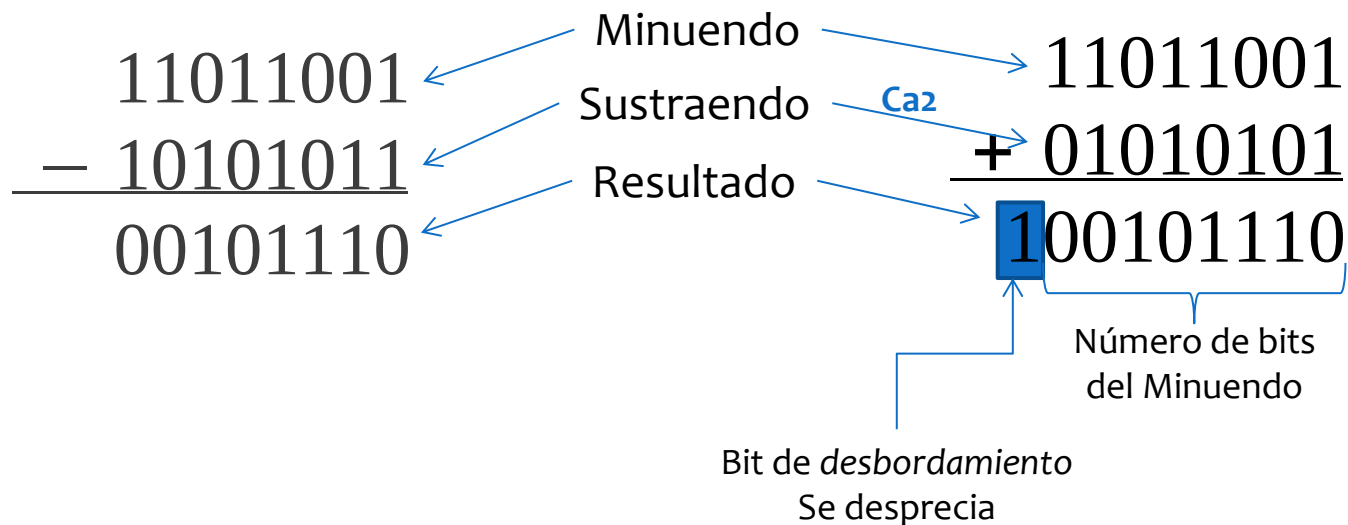
- ▶ Complemento a 1 (Ca1)
 - ▶ El ordenador tiene n bits para almacenar un número N
 - ▶ Bit más significativo actúa como signo
 - ▶ Opera con la cadena completa
 - ▶ Cambia los unos por ceros y los ceros por unos
 - ▶ $01001011 = 75$
 - ▶ $10110100 = -75$ comp 1
 - ▶ Desde $(-2^{n-1}+1)$ hasta $(2^{n-1}-1)$
 - ▶ El número cero tiene dos posibles representaciones
 - ▶ Representación utilizada en PDP-1 y serie UNIVAC 1100/2200

Números Enteros

- ▶ Complemento a 2 (Ca2)
 - ▶ Sistema más utilizado en los ordenadores actuales
 - ▶ El ordenador tiene n bits para almacenar un número N
 - ▶ Bit más significativo actúa como signo
 - ▶ Opera con la cadena completa
 - ▶ Pasa el número a C1 y suma 1
 - ▶ $01001011 = 75$
 - ▶ $10110100 = -75$ comp 1
 - ▶ $10110101 = -75$ comp 2
 - ▶ Desde (-2^{n-1}) hasta $(2^{n-1}-1)$
 - ▶ El número cero solo tiene una representación

Números Enteros

- ▶ Complemento a 2 (Ca2) – Uso aritmético
 - ▶ No se implementa la operación de Resta
 - ▶ En su lugar, se pasa el sustraendo a Ca2 y se realiza la suma



Números Enteros

► Exceso a M

- Utilizado en los exponentes de reales
- El ordenador tiene *n bits para almacenar un número N*
- Exceso no estandarizado
 - Exceso a 2^{n-1} coincide con el Ca2 con bit más significativo negado
 - Exceso a $2^{n-1}-1$ se utiliza en norma IEEE754 para el exponente
- No tiene bit de signo
 - Números mayores que el exceso, positivos
 - Números menores que el exceso, negativos
 - Número cero representado por el exceso
- Cantidad representada en binario más el exceso
 - $n=8, M=2^{n-1}=2^7=128$
 - $-45 \rightarrow -45 + 128 = 83_{(10)} = 01010011_{(2)}$
 - $0 \rightarrow 0 + 128 = 128_{(10)} = 10000000_{(2)}$
 - $45 \rightarrow 45 + 128 = 173_{(10)} = 10101101_{(2)}$

Números Enteros

Decimal	Entero sin signo (<i>unit</i>)	Signo y Magnitud	Ca1 (Comp. a uno)	Ca2 (Comp. a dos)	Exceso a M ($M=2^{4-1}=8$)	Exceso a M ($M=2^{4-1}-1=7$)
+8	1000	N.R.	N.R.	N.R.	N.R.	1111
+7	0111	0111	0111	0111	1111	1110
+6	0110	0110	0110	0110	1110	1101
+5	0101	0101	0101	0101	1101	1100
+4	0100	0100	0100	0100	1100	1011
+3	0011	0011	0011	0011	1011	1010
+2	0010	0010	0010	0010	1010	1001
+1	0001	0001	0001	0001	1001	1000
+0 -0	0000	0000 1000	0000 1111	0000	1000	0111
-1	N.R.	1001	1110	1111	0111	0110
-2	N.R.	1010	1101	1110	0110	0101
-3	N.R.	1011	1100	1101	0101	0100
-4	N.R.	1100	1011	1100	0100	0011
-5	N.R.	1101	1010	1011	0011	0010
-6	N.R.	1110	1001	1010	0010	0001
-7	N.R.	1111	1000	1001	0001	0000
-8	N.R.	N.R.	N.R.	1000	0000	N.R.



Números Reales

- ▶ Coma fija
 - ▶ Dos campos de longitud fija
 - ▶ Parte entera
 - ▶ Parte decimal
 - ▶ Problema de representación asociado a la longitud de los campos
 - ▶ Ejemplo
 - Número real con 3 bits para cada uno de los campos
 - $000,125_{(10)} \cdot 000,001_{(10)} = 000,000125_{(10)}$
 - Ajustando al formato... $000,000_{(10)}$

Números Reales

- ▶ Coma flotante (exponencial o científica)
 - ▶ $R = M \cdot b^E$
 - ▶ *Dividir los n bits para representar el número en 3 zonas*
 - ▶ (S) signo del número (mantisa)
 - ▶ (M) mantisa
 - ▶ (E) exponente
 - La base queda implícita
 - ▶ **Normalización de la mantisa**
 - ▶ $00001001 \cdot 2^3 = 0,1001 \cdot 2^7 = 1,001 \cdot 2^6$
 - ▶ $01100110 \cdot 2^{-5} = 0,110011 \cdot 2^2 = 1,10011 \cdot 2^1$
 - ▶ Bit implícito, colocando el primer 1 a la izquierda de la coma

Números Reales

- ▶ Coma flotante – Normalización IEEE754
 - ▶ Dos precisiones distintas
 - ▶ Simple precisión: 32 bits
 - ▶ Doble precisión: 64 bits
 - ▶ La disposición de las zonas permite que se puedan utilizar los mismos algoritmos de comparación que con números enteros



- ▶ Exponente en exceso ($2^{e-1}-1$)
- ▶ Mantisa almacenada en representación signo magnitud, separados por el exponente, con bit implícito (forma 1,xxxx)
- ▶ Base del exponente y base de la mantisa, 2

Números Reales

► Coma flotante – Normalización IEEE754

► Ejemplo

- -81,375 en simple precisión
- Parte entera: $-81_{(10)} = -1010001_{(2)}$
- Parte decimal: $0,375_{(10)} = 011_{(2)}$
- Mantisa: $-1010001,011_{(2)}$
- Mantisa_{normalizada}: $-1,010001011_{(2)} \cdot 2^6$
- Exponente: $6_{(10)} + 127_{(10)} = 133_{(10)} = 10000101_{(2)}$

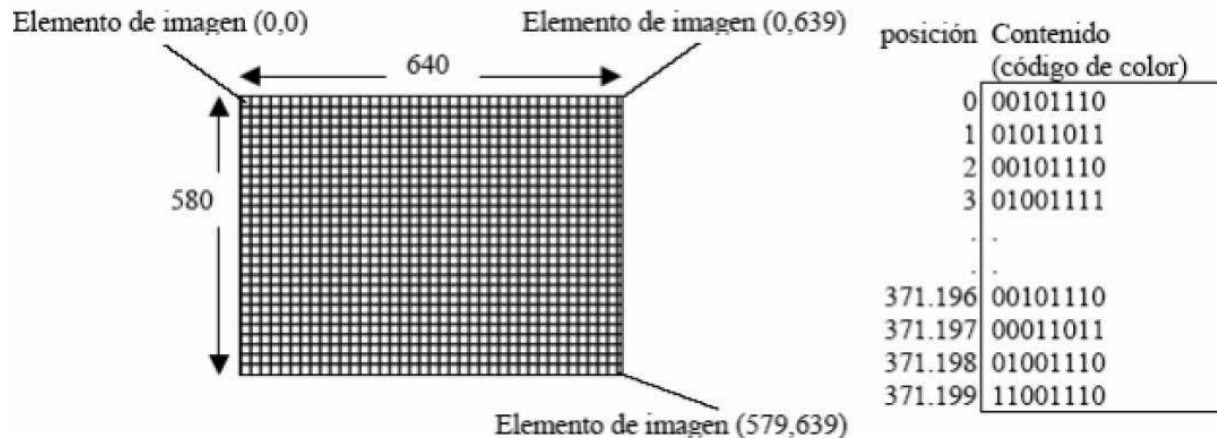
1	1	0	0	0	0	1	0	1	0	1	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Estructuras de datos

- ▶ Aunque los datos puedan formar estructuras, el ordenador los suele almacenar sin tener esto en cuenta
 - ▶ El programa que los utilice será el encargado de entenderlos como una estructura

Gráficos

► Imágenes rasterizadas (matriciales o mapas de bits)



En cada pixel (cada punto distinguible de la imagen) se encuentra la información descompuesta en los tres colores primarios (R – rojo, G – verde, B – azul)



La **profundidad de color** hace referencia al número de colores que admita la representación, o visto de otra forma, al número de bits que componen cada uno de los puntos de información RGB en la matriz.

Gráficos

- ▶ Representación vectorial
 - ▶ División de la imagen en objetos
 - Geometría
 - Punto
 - Segmento o línea
 - Polilínea (línea quebrada)
 - Rentágulo
 - Polígono
 - Arco
 - ▶ Atributos
 - Forma en que se visualiza la imagen
 - Enlace a los demás objetos de la imagen
 - ▶ Agrupación
 - Vecindad
 - Jerarquía
 - Superposición

Gráficos

- ▶ Tamaño de una imagen rasterizada

Tamaño Total = bits_para_cada_color *
resolucion_horizontal * resolucion_vertical

- ▶ Por ejemplo, para una imagen de 800x600 con 16 bits de profundidad de color

Tamaño = 16 * 800 * 600 = 7680000 bits =
= 960000 bytes = 937,5 KB

Audio

► Codificación de audio

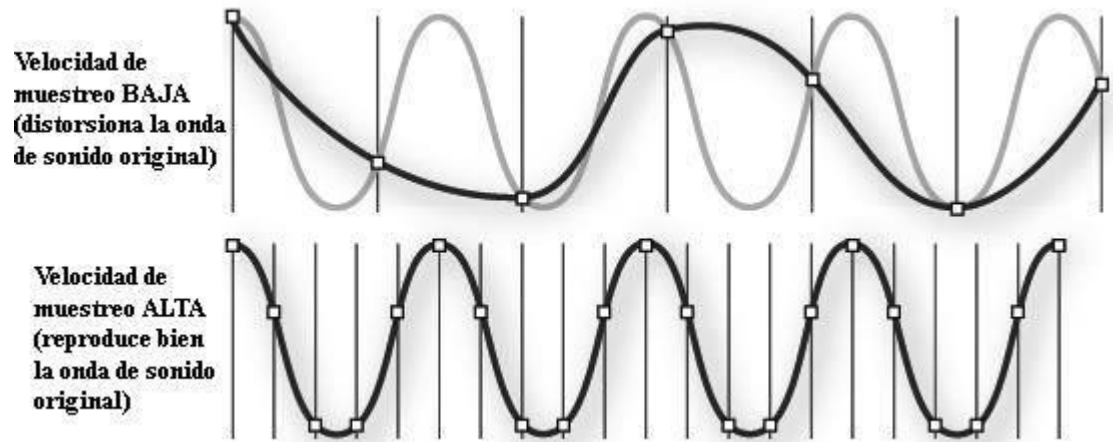


Representación de la onda de sonido. En el eje vertical se encuentra la amplitud del sonido y el horizontal el tiempo

Calidad del muestreo: Valor de la amplitud (altura de la onda)

Frecuencia de muestreo: Tiempo transcurrido entre que se toma una muestra y la siguiente

Canales (sonido estéreo o mono)



Audio

- ▶ Tamaño de audio

Tamaño Total = numero_de_canales *
calidad_muestreo * frecuencia_muestreo *
duracion

- ▶ Por ejemplo, para una secuencia de 30 segundos de sonido estéreo con una calidad de 32 bits y una frecuencia de 22 KHz

$$\begin{aligned}\text{Tamaño} &= 2 * 32 * 22.000 * 30 = 42240000 \text{ bits} = \\ &= 5280000 \text{ bytes} = 5156,25 \text{ KB}\end{aligned}$$

Vídeo

- ▶ Tamaño de vídeo
 - ▶ Representación de imágenes (o frames) y sonido en el tiempo
- ▶ Vídeo de 30 segundos grabado a 640x480 y 32 bits de profundidad de color, a 30 fps con sonido estéreo de 32 bits de calidad con frecuencia 22 KHz.

$$\begin{aligned}T_{\text{imagen}} &= 640 * 480 * 32 = 9830400 \text{ bits} = 1200 \text{ KB} \\T_{\text{sec_img}} &= 1200 * 30 \text{ (fps)} * 30 \text{ (s)} = 1200 * 900 \text{ (img)} = \\&= 1080000 \text{ KB} = 1024,69 \text{ MB}\end{aligned}$$

$$\begin{aligned}T_{\text{sonido}} &= 2 * 32 * 22.000 * 30 = 42240000 \text{ bits} = \\&= 5280000 \text{ bytes} = 5156,25 \text{ KB} = 5,03 \text{ MB}\end{aligned}$$

$$T_{\text{video}} = 1024,69 + 5,03 = 1059,72 \text{ MB} = 1,03 \text{ GB}$$