

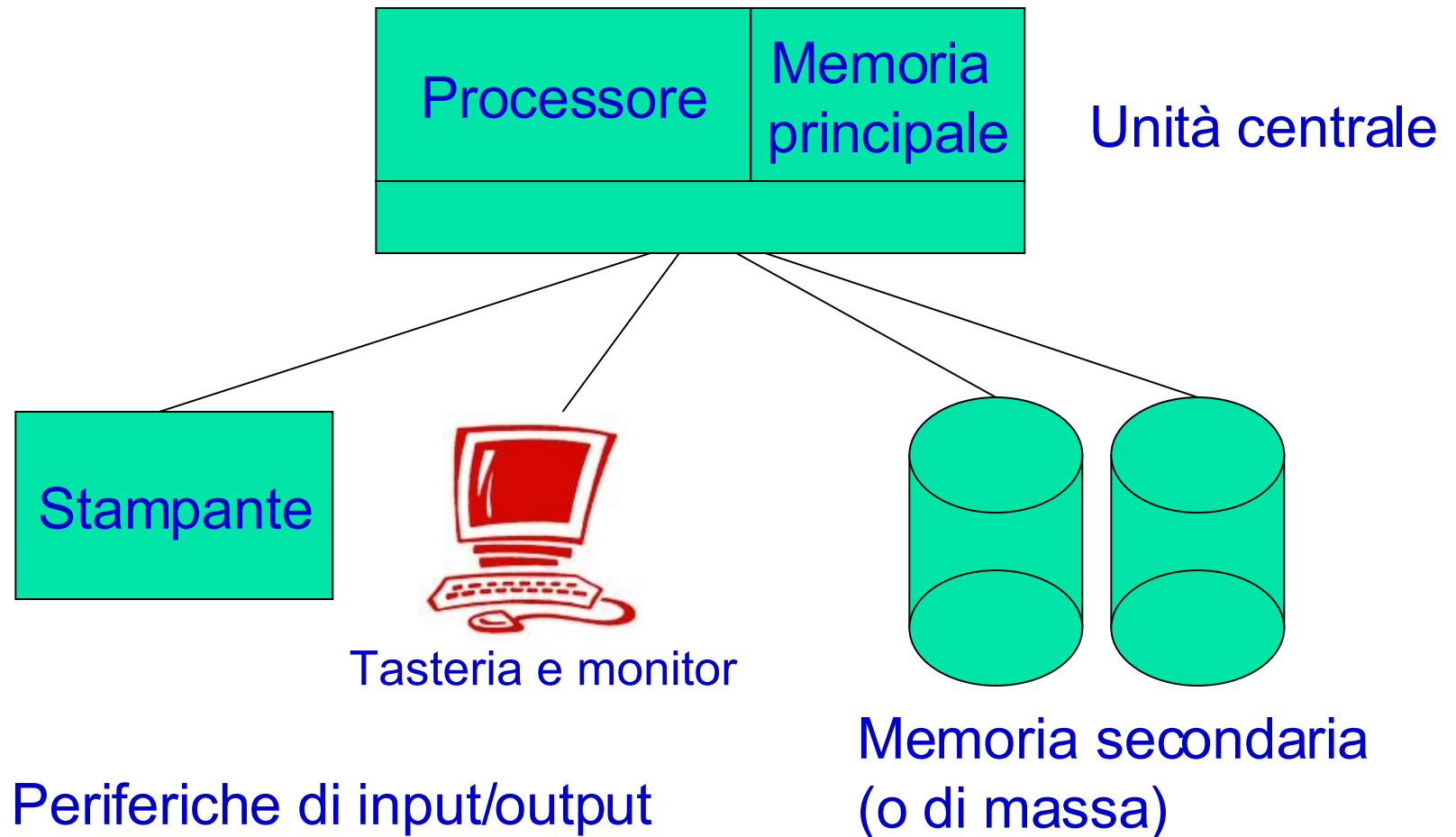
Rappresentazione digitale dei dati

Codifica delle Informazioni

Hardware: architettura dei computer

- In un computer possiamo distinguere quattro unità funzionali:
 - il processore
 - la memoria principale
 - la memoria secondaria
 - i dispositivi di input (inserimento)/output (restituzione di risultati)
- Il processore e la memoria principale costituiscono l'unità centrale del computer

Componenti principali di un computer



L'uso dell'Informazione

Un computer deve:

- memorizzare l'informazione
 - usando la memoria principale/secondaria
- elaborare l'informazione
 - usando il processore
- fare l'input/output dell'informazione
 - usando i dispositivi di input/output

Tipi di Informazione

- Esistono vari tipi di informazione, di natura e forma diversa, così come rappresentazioni diverse della stessa informazione
 - La scelta della rappresentazione è in genere vincolata al tipo di utilizzo ed al tipo di operazioni che devono essere fatte sulle informazione stesse
- Il computer memorizza ed elabora informazioni che devono essere rappresentate in una forma gestibile
- Rappresentazione **digitale**

Rappresentazione digitale

=

rappresentazione binaria

- L'entità minima di informazione che possiamo trovare all'interno di un elaboratore prende il nome di bit
 - **binary digit** – cifra binaria
 - un bit può assumere due valori
- Rappresentazione binaria
 - solo due simboli (0 e 1)

Perché la rappresentazione binaria?

- Le informazioni rappresentate nel sistema binario possono essere elaborate secondo vari criteri e con vari strumenti
- I due simboli (0 e 1) possono essere rappresentati da:
 - Due stati di polarizzazione di una sostanza magnetizzabile
 - Due stati di carica elettrica di una sostanza
 - Al passaggio/non passaggio di corrente attraverso un cavo conduttore
 - Al passaggio/non passaggio di luce attraverso un cavo ottico

Codifica dell'Informazione

- Per poter rappresentare un numero maggiore di informazioni si usano **sequenze** di bit
- Per esempio, per rappresentare quattro informazioni diverse possiamo utilizzare due bit che ci permettono di ottenere quattro configurazioni distinte

00 01 10 11

Il processo secondo cui si fa corrispondere ad un'informazione una sequenza di bit prende il nome **codifica dell'Informazione**

Codifica binaria

- Esempio: un esame può avere quattro possibili esiti: ottimo, discreto, sufficiente, insufficiente
- Codifico
 - ottimo con 00
 - discreto con 01
 - sufficiente con 10
 - insufficiente con 11

Codifica binaria

- Con 2 bit si codificano 4 informazioni (2^2)
- Con 3 bit si codificano 8 informazioni (2^3)
- ...
- Con N bit si possono codificare 2^N informazioni differenti

Codifica binaria

- Se il problema è quello di dover rappresentare M informazioni differenti si deve selezionare il numero di N bit in modo tale che

$$2^N \geq M$$

- Esempio: per rappresentare 40 informazioni differenti devo utilizzare 6 bit perché

$$2^6 = 64$$

- 5 bit non sono sufficienti perché $2^5 = 32$

Codifica binaria

- Esiste una particolare aggregazione di bit che è costituita da 8 bit ($2^8 = 256$ informazioni) e prende il nome di **byte**
- Di solito si usano i multipli del byte

Kilo	KB	2^{10} ($\sim$ un migliaio, 1024 byte)
Mega	MB	2^{20} ($\sim$ un milione, 1KB x 1024 byte)
Giga	GB	2^{30} ($\sim$ un miliardo, 1MB x 1024 byte)
Tera	TB	2^{40} ($\sim$ mille miliardi, 1GB x 1024 byte)

Codifica dei caratteri

■ Alfabeto anglosassone

- Lettere maiuscole e minuscole
- Cifre numeriche (0, 1, 2, ..., 9)
- Simboli di punteggiatura (, . ; : ! “ ? ...)
- Segni matematici (+, -, {, [, >, ...)
- Caratteri nazionali (à, è, ì, ò, ù, ç, ñ, ö, ...)

può essere codificato usando un byte (220 caratteri circa)

- ## ■ Il metodo di codifica più diffuso tra i produttori di hardware e di software prende il nome ASCII (American Standard Code for Information Interchange)

Codifica dei caratteri

- Codice di riferimento:
 - ASCII: 8 bit per carattere
- Altri codici:
 - UNICODE, 16 bit per carattere (ASCII + caratteri etnici)
 - MSWindows usa un codice simile ad UNICODE

Codifica dei caratteri (ASCII)

ASCII	Simbolo
00000000	NUL (spazio bianco)
...	...
00111110	>
00111111	?
01000000	@
01000001	A
01000010	B
01000011	C
...	...

Tabella codice ascii standard

Byte	Cod.	Char	Byte	Cod.	Char	Byte	Cod.	Char	Byte	Cod.	Char
00000000	0	Null	00100000	32	Spc	01000000	64	@	01100000	96	`
00000001	1	Start of heading	00100001	33	!	01000001	65	A	01100001	97	a
00000010	2	Start of text	00100010	34	"	01000010	66	B	01100010	98	b
00000011	3	End of text	00100011	35	#	01000011	67	C	01100011	99	c
00000100	4	End of transmit	00100100	36	\$	01000100	68	D	01100100	100	d
00000101	5	Enquiry	00100101	37	%	01000101	69	E	01100101	101	e
00000110	6	Acknowledge	00100110	38	&	01000110	70	F	01100110	102	f
00000111	7	Audible bell	00100111	39	'	01000111	71	G	01100111	103	g
00001000	8	Backspace	00101000	40	(	01001000	72	H	01101000	104	h
00001001	9	Horizontal tab	00101001	41	)	01001001	73	I	01101001	105	i
00001010	10	Line feed	00101010	42	*	01001010	74	J	01101010	106	j
00001011	11	Vertical tab	00101011	43	+	01001011	75	K	01101011	107	k
00001100	12	Form Feed	00101100	44	,	01001100	76	L	01101100	108	l
00001101	13	Carriage return	00101101	45	-	01001101	77	M	01101101	109	m
00001110	14	Shift out	00101110	46	.	01001110	78	N	01101110	110	n
00001111	15	Shift in	00101111	47	/	01001111	79	O	01101111	111	o
00010000	16	Data link escape	00110000	48	0	01010000	80	P	01110000	112	p
00010001	17	Device control 1	00110001	49	1	01010001	81	Q	01110001	113	q
00010010	18	Device control 2	00110010	50	2	01010010	82	R	01110010	114	r
00010011	19	Device control 3	00110011	51	3	01010011	83	S	01110011	115	s
00010100	20	Device control 4	00110100	52	4	01010100	84	T	01110100	116	t
00010101	21	Neg. acknowledge	00110101	53	5	01010101	85	U	01110101	117	u
00010110	22	Synchronous idle	00110110	54	6	01010110	86	V	01110110	118	v
00010111	23	End trans. block	00110111	55	7	01010111	87	W	01110111	119	w
00011000	24	Cancel	00111000	56	8	01011000	88	X	01111000	120	x
00011001	25	End of medium	00111001	57	9	01011001	89	Y	01111001	121	y
00011010	26	Substitution	00111010	58	:	01011010	90	Z	01111010	122	z
00011011	27	Escape	00111011	59	;	01011011	91	[	01111011	123	{
00011100	28	File separator	00111100	60	<	01011100	92	\	01111100	124	
00011101	29	Group separator	00111101	61	=	01011101	93	]	01111101	125	}
00011110	30	Record Separator	00111110	62	>	01011110	94	^	01111110	126	~
00011111	31	Unit separator	00111111	63	?	01011111	95	_	01111111	127	Del

Come si inseriscono i caratteri ASCII da tastiera

- E' sufficiente tenere premuto il tasto **ALT** e contemporaneamente digitare sul **tastierino numerico** il codice decimale del carattere da inserire. Rilasciando il tasto ALT viene inserito il carattere desiderato.
- Es.: Se si deve inserire la tilde ~ bisogna tenere premuto il tasto ALT e digitare sul tastierino numerico il valore 126.

~ = ALT + 126

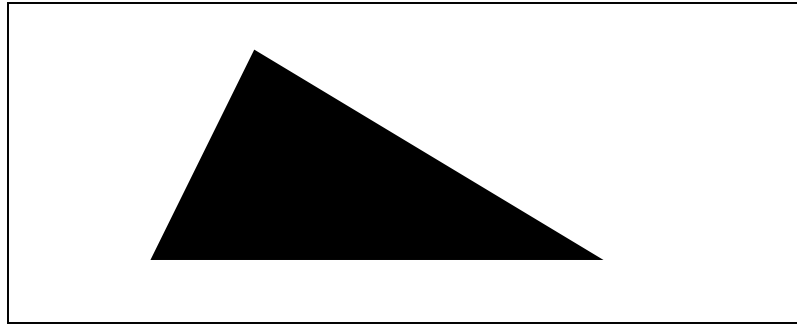
Codifica delle parole

- Parole sono sequenze di caratteri
- Codifica della parole **cane**

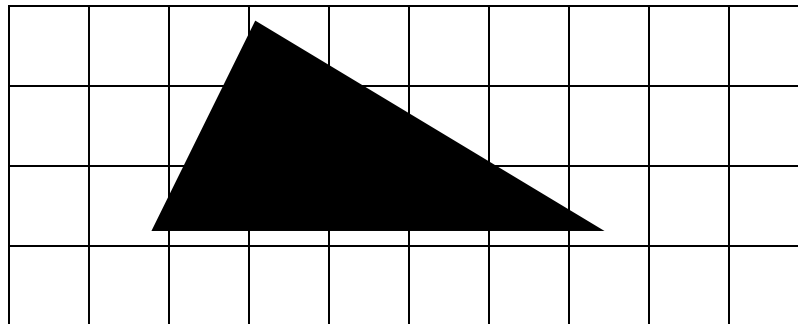
01100011	01100001	01101110	01100101
c	a	n	e

- Il problema inverso: data una sequenza di bit, il testo che essa codifica può essere ottenuto nel modo seguente:
 - si divide la sequenza in gruppi di otto bit (byte)
 - si determina il carattere corrispondente ad ogni byte

Codifica delle immagini



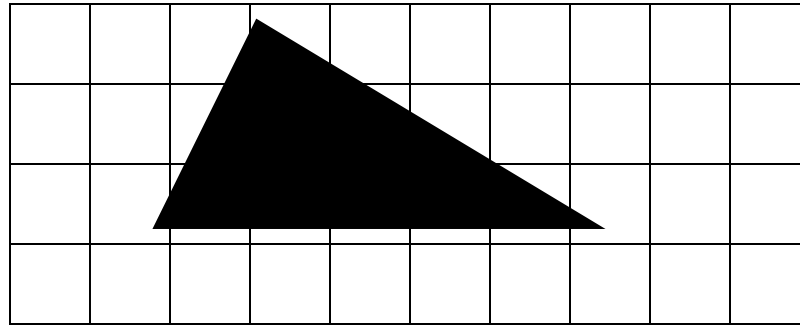
Suddividiamo l'immagine mediante una griglia formata da righe orizzontali e verticali a distanza costante



Codifica delle immagini

- Ogni quadratino derivante da tale suddivisione prende il nome di **pixel** (picture element) e può essere codificato in binario secondo la seguente convenzione:
 - Il simbolo “0” viene utilizzato per la codifica di un pixel corrispondente ad un quadratino in cui il bianco è predominante
 - Il simbolo “1” viene utilizzato per la codifica di un pixel corrispondente ad un quadratino in cui il nero è predominante

Codifica delle immagini



0	0	0	1	0	0	0	0	0	0
0	0	1	1	1	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0

Codifica delle immagini

0	0	0	1	0	0	0	0	0	0
0	0	1	1	1	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0

Poiché una sequenza di bit è lineare, è necessario definire convenzioni per ordinare la griglia dei pixel in una sequenza. Assumiamo che i pixel siano ordinati dal basso verso l'alto e da sinistra verso destra

0000000000 0011111000 0011100000 0001000000

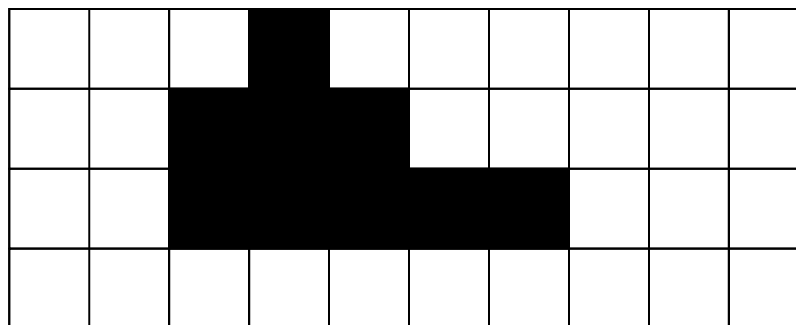
Codifica delle immagini

Non sempre il cortorno della figura coincide con le linee della griglia. Quella che si ottiene nella codifica è un'approssimazione della figura originaria.

Se riconvertiamo la sequenza di stringhe

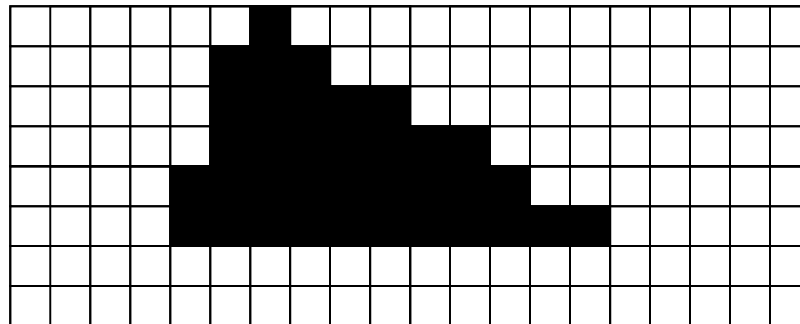
0000000000 0011111000 0011100000 0001000000

in immagine otteniamo



Codifica delle immagini

La rappresentazione sarà più fedele
all'aumentare del numero di pixel, ossia
al diminuire delle dimensioni dei
quadrantini della griglia in cui è suddivisa
l'immagine



Codifica delle immagini

- Assegnando un bit ad ogni pixel è possibile codificare solo immagini senza livelli di chiaroscuro
- Le immagini in bianco e nero hanno invece delle sfumature (diversi livelli di intensità di grigio)
- Per codificare le immagini con diversi livelli di grigio oppure a colori si usa la stessa tecnica
 - per ogni pixel viene assegnata una rappresentazione binaria con più di un bit

Codifica delle immagini (grigio e colore)

- Per memorizzare un pixel non è più sufficiente un solo bit
 - per esempio, se utilizziamo quattro bit possiamo rappresentare $2^4 = 16$ livelli di grigio o 16 colori diversi
 - mentre con otto bit ne possiamo distinguere $2^8 = 256$, ecc.

Risoluzione

- Il numero di pixel presenti sullo schermo (colonne x righe) prende il nome di **risoluzione**
- Risoluzione tipiche sono
640 x 480 800 x 600 1024 x 768 1280 x 1024
- Esempio:
 - per distinguere 256 colori sono necessari otto bit per la codifica di ciascun pixel
 - la codifica di un'immagine formata da 640 x 480 pixel richiederà 2.457.600 bit (307.200 byte)

Grafica bitmap

- Le immagini codificate pixel per pixel sono dette immagini in grafica **bitmap**
 - Le immagini bitmap occupano parecchio spazio
- Esistono delle tecniche di compressione che permettono di ridurre le dimensioni
 - Ad esempio, se più punti vicini di un'immagine assumono lo stesso colore, si può memorizzare la codifica del colore una sola volta e poi ricordare per quante volte deve essere ripetuta
- I formati come GIF e JPEG sono formati compressi

Grafica vettoriale

- Crea le immagini manipolando linee e curve
 - La colorazione avviene attraverso la colorazione delle linee e delle aree chiuse
- Spesso occupano meno spazio rispetto alle immagini bitmap
- La grafica bitmap, invece, riesce a rendere con qualità maggiore immagini con un numero elevato di colori

Codifica delle immagini

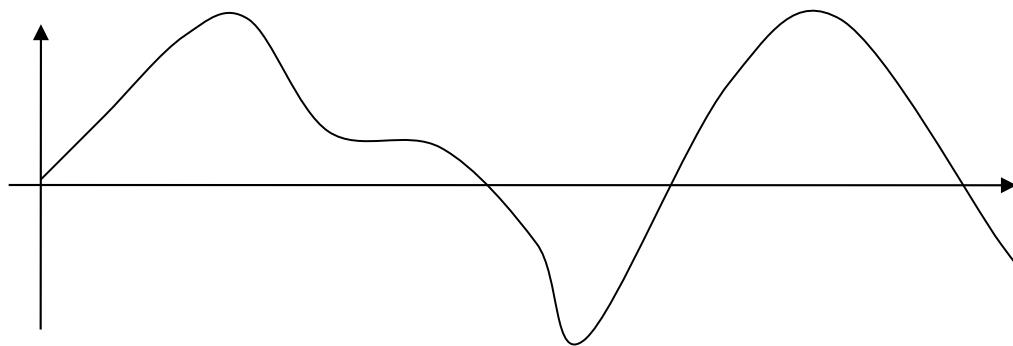
- Immagini complesse od irregolari:
codifica bitmap (o raster)
- Immagini regolari:
codifica vettoriale
- Codifiche ibride (raster/vettoriale)
codifiche standard: Postscript, PDF

Codifica di immagini in movimento

- Un filmato è una sequenza di immagini statiche (dette fotogrammi o frame)
- Per codificare un filmato si digitalizzano i suoi fotogrammi
- Esempio:
 - 30 immagini ad alta risoluzione al secondo
 - $30 \text{ imm./sec} \times 2.457.600 \text{ bit/imm.} = 73.728.000 \text{ bit/sec}$
 - Un minuto richiederebbe $60 \text{ sec} \times 73.728.000 = 4.423.680.000 \text{ bit}$ (5.529.600 byte)
- Sono necessarie delle tecniche per ottimizzare tale processo. Lo standard più diffuso: MPEG (molto efficiente)

Codifica dei suoni

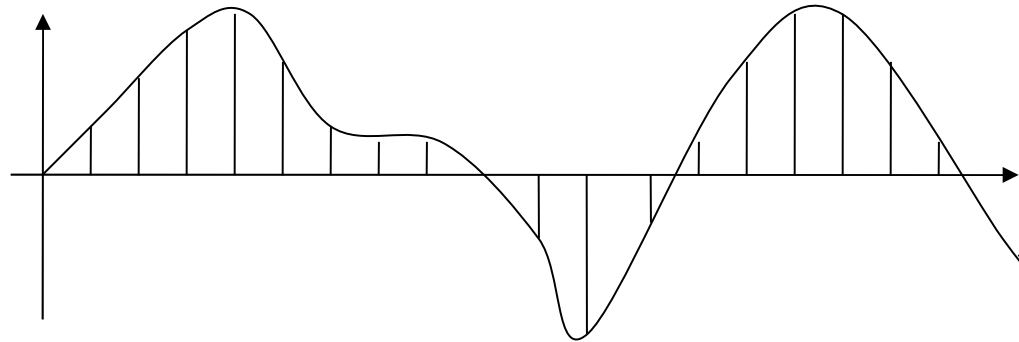
- Fisicamente un suono è rappresentato come un'onda che descrive la variazione della pressione dell'aria nel tempo (onda sonora)



- Sull'asse delle ascisse viene rappresentato il tempo e sull'asse delle ordinate viene rappresentata la variazione di pressione corrispondente al suono stesso

Codifica dei suoni

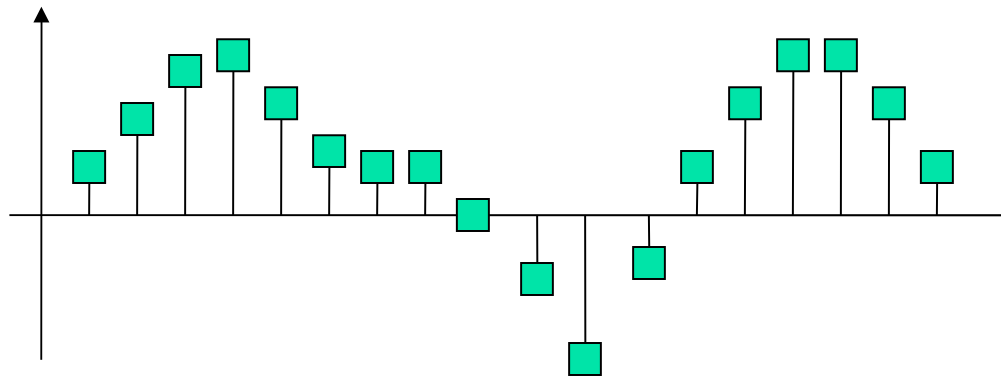
- Si effettuano dei **campionamenti** sull'onda (cioè si misura il valore dell'onda a intervalli costanti di tempo) e si codificano in forma digitale le informazioni estratte



- Quanto più frequentemente il valore di intensità dell'onda viene campionato, tanto più precisa sarà la sua rappresentazione

Codifica dei suoni

- La sequenza dei valori numerici ottenuti dai campioni può essere facilmente codificata con sequenze di bit



Una
approssimazione!

- La rappresentazione è tanto più precisa quanto maggiore è il numero di bit utilizzati per codificare l'informazione estratta in fase di campionamento

Codifica dei suoni (esempio)

- Se volessimo codificare la musica di qualità CD dovremmo:
 - Usare due registrazioni corrispondenti a due microfoni distinti
 - Campionare il segnale musicale producendo 44100 campioni al secondo
 - Per ogni campione (che è un numero) si usano 16 bit
 - Per cui, il numero di bit che sarebbero necessari per codificare ogni secondo è pari a

$$2 \times 44.100 \text{ campioni} \times 16 \text{ bit/campione} = 1.414.200 \text{ bit}$$

Codifica dei suoni

- **Codifiche standard**
 - WAV (MS-Windows)
 - MIDI
 - MP3
- **MIDI**
 - Codifica le note e gli strumenti che devono eseguirle
 - Efficiente, ma solo musica, non voce
- **MP3**
 - MPEG-3: variante MPEG per suoni
 - Grande diffusione, molto efficiente

Codifica dei numeri

- Il codice ASCII consente di codificare le cifre decimali da “0” a “9” fornendo in questo modo una rappresentazione dei numeri
- Per esempio: il numero 324 potrebbe essere rappresentato dalla sequenza di byte:

00110011 00110010 00110100
3 2 4

- Ma questa rappresentazione non è efficiente e soprattutto non è adatta per eseguire le operazioni aritmetiche sui numeri

Codifica dei numeri (il sistema decimale)

- La rappresentazione dei numeri con il sistema decimale può essere utilizzata come spunto per definire un metodo di codifica dei numeri all'interno degli elaboratori
 - Esempio: la sequenza di cifre 324 viene interpretato come:
 - 3 centinaia + 2 decine + 4 unità
 - $324 = 3 \times 100 + 2 \times 10 + 4 \times 1$
 - $324 = 3 \times 10^2 + 2 \times 10^1 + 4 \times 10^0$

Codifica dei numeri (il sistema decimale)

- In generale la sequenza $c_n c_{n-1} c_{n-2} \dots c_1 c_0$ (ogni “ c_i ” è una cifra compresa tra “0” e “9”) viene interpretata come:

La cifra meno significativa $\rightarrow$

$$\begin{aligned}
 &c_0 \times 10^0 + && (c_0 \text{ unità}) \\
 &c_1 \times 10^1 + && (c_1 \text{ decine}) \\
 &c_2 \times 10^2 + && (c_2 \text{ centinaia}) \\
 &\dots \\
 &\text{La cifra più significativa} \rightarrow c_{n-1} \times 10^{n-1} + \\
 &&& c_n \times 10^n
 \end{aligned}$$

Codifica dei numeri (il sistema binario)

- La numerazione decimale quindi utilizza una **notazione posizionale** basata sul numero 10 (numero delle cifre)
- La notazione posizionale può essere utilizzata in qualunque altro sistema di numerazione (con base diversa di 10)
- Nel sistema di numerazione binario i numeri vengono codificati utilizzando le due cifre “0” e “1”

Codifica dei numeri (il sistema binario)

- In analogia con il caso decimale la sequenza $c_n c_{n-1} c_{n-2} \dots c_1 c_0$ (ogni “ c_i ” è la cifra “0” o la cifra “1”) rappresenterà il numero

$$\begin{array}{rcl}
 \text{La cifra meno} & \rightarrow & c_0 \times 2^0 \quad + \\
 \text{significativa} & & \\
 & & c_1 \times 2^1 \quad + \\
 & & c_2 \times 2^2 \quad + \\
 & & \dots \quad + \\
 \text{La cifra più} & \rightarrow & c_{n-1} \times 2^{n-1} \quad + \\
 \text{significativa} & & c_n \times 2^n
 \end{array}$$

Codifica dei numeri (il sistema binario)

- Esempio: la sequenza “1011” rappresenta il numero

$$1 \times 2^0 + 1 \times 2^1 + 0 \times 2^2 + 1 \times 2^3 = \mathbf{11} \text{ (in base 10)}$$

- Per evitare ambiguità si usa la notazione

$$1011_2 = 11_{10}$$

Conversione dalla base 10 alla base 2

- Dato un numero N rappresentato in base dieci, la sua rappresentazione in base due sarà del tipo $c_m c_{m-1} c_{m-2} \dots c_1 c_0$ (le “ c_i ” sono cifre binarie)
- Per convertire un numero in base dieci nel corrispondente in base due si devono trovare i resti delle divisioni successive del numero N per due

Conversione dalla base 10 alla base 2

- Esempio: il numero 6_{10} :

$$6/2 = 3 \text{ resto } 0$$

$$3/2 = 1 \text{ resto } 1$$

$$1/2 = 0 \text{ resto } 1$$

- Leggendo i resti dal basso verso l'alto, si ha che la rappresentazione binaria del numero 6_{10} è 110_2

Conversione dalla base 10 alla base 2

■ Perché $110_2 = 6_{10}$?

$$\begin{array}{rcl}
 6/2 = 3 \text{ resto } 0 & & 0 \times 2^0 + \\
 3/2 = 1 \text{ resto } 1 & & 1 \times 2^1 + \\
 1/2 = 0 \text{ resto } 1 & & \underline{1 \times 2^2 =} \\
 & & 6
 \end{array}$$

$$\frac{0 \times 2^0 + 1 \times 2^1 + 1 \times 2^2}{2} = 1 \times 2^0 + 1 \times 2^1 \text{ con resto } 0$$

$$\frac{1 \times 2^0 + 1 \times 2^1}{2} = 1 \times 2^0 \text{ con resto } 1$$

$$\frac{1 \times 2^0}{2} = 0 \text{ con resto } 1$$

Conversione dalla base 10 alla base 2

- Esempio: il numero 345_{10} :

$$345/2 = 172 \quad \text{resto } 1$$

$$172/2 = 86 \quad \text{resto } 0$$

$$86/2 = 43 \quad \text{resto } 0$$

$$43/2 = 21 \quad \text{resto } 1$$

$$21/2 = 10 \quad \text{resto } 1$$

$$10/2 = 5 \quad \text{resto } 0$$

$$5/2 = 2 \quad \text{resto } 1$$

$$2/2 = 1 \quad \text{resto } 0$$

$$1/2 = 0 \quad \text{resto } 1$$

- Leggendo i resti dal basso verso l'alto, si ha che la rappresentazione binaria del numero 345_{10} è 101011001_2

Conversione dalla base 2 alla base 10

- Sia $c_m c_{m-1} c_{m-2} \dots c_1 c_0$ un numero rappresentato in base 2, usiamo:

$$c_0 \times 2^0 + c_1 \times 2^1 + c_2 \times 2^2 + \dots + c_{m-1} \times 2^{m-1} + c_m \times 2^m = N$$

- Esempio: 101011001_2

$$1 \times 2^0 + 0 \times 2^1 + 0 \times 2^2 + 1 \times 2^3 + 1 \times 2^4 + 0 \times 2^5 +$$

$$1 \times 2^6 + 0 \times 2^7 + 1 \times 2^8$$

=

$$1 + 8 + 16 + 64 + 256$$

=

$$345$$

Altre basi: ottale, esadecimale

■ Sistema ottale

- Utilizza una notazione posizionale basata su otto cifre (0,1,...,7) e sulle potenze di 8
- Esempio: $103_8 = 1 \times 8^2 + 0 \times 8^1 + 3 \times 8^0 = 67$

■ Sistema esadecimale

- Utilizza una notazione posizionale basata su sedici cifre (0,1,...,9,A,B,C,D,E,F) e sulle potenze di 16
- Esempio: $103_{16} = 1 \times 16^2 + 0 \times 16^1 + 3 \times 16^0 = 259$
- Esempio: $AC4_{16} = 10 \times 16^2 + 12 \times 16^1 + 4 \times 16^0 = 2756$