

CODIFICA DI CARATTERI

- ◆ Associando un simbolo dell'alfabeto ad ogni numero possiamo codificare tutte le lettere
- ◆ Codifica **ASCII** (*American Standard Code for Information Interchange*):
 - Caratteri speciali, punteggiatura, a-z, A-Z, 0-9
 - Utilizza 7 bit (128 caratteri)
 - I codici ASCII estesi usano 8 bit (256 caratteri)
- ◆ Codifica **EBCDIC** (*Extended Binary-Coded Decimal Interchange Code*)
 - Utilizza 8 bit (256 caratteri)
- ◆ Codifica **UNICODE**
 - Utilizza 16 bit (65536 caratteri)
 - I primi 128 caratteri di UNICODE sono gli stessi di ASCII
 - I successivi corrispondono ad altri alfabeti (greco, cirillico, ebraico, ...)
 - Non riesce a coprire i simboli (oltre 200.000) di tutte le lingue!

ASCII SU 7 BIT

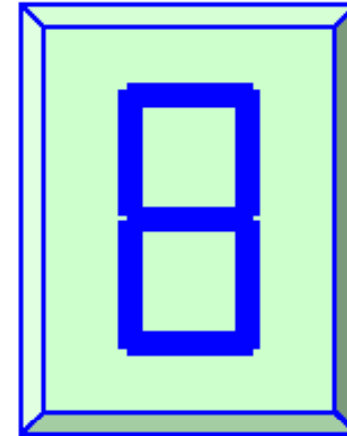
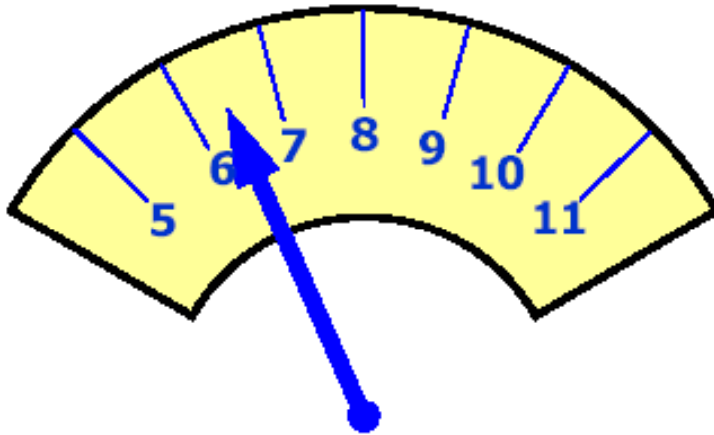
	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
010	sp	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
011	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
100	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
101	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
110	`	a	b	c	d	e	f	g	h	I	j	k	l	m	n	o
111	p	q	r	s	t	u	v	w	x	Y	z	{		}	~	canc

“Ciao” = 1000011 1101001 1100001 1101111

“24” = 0110001 0110011

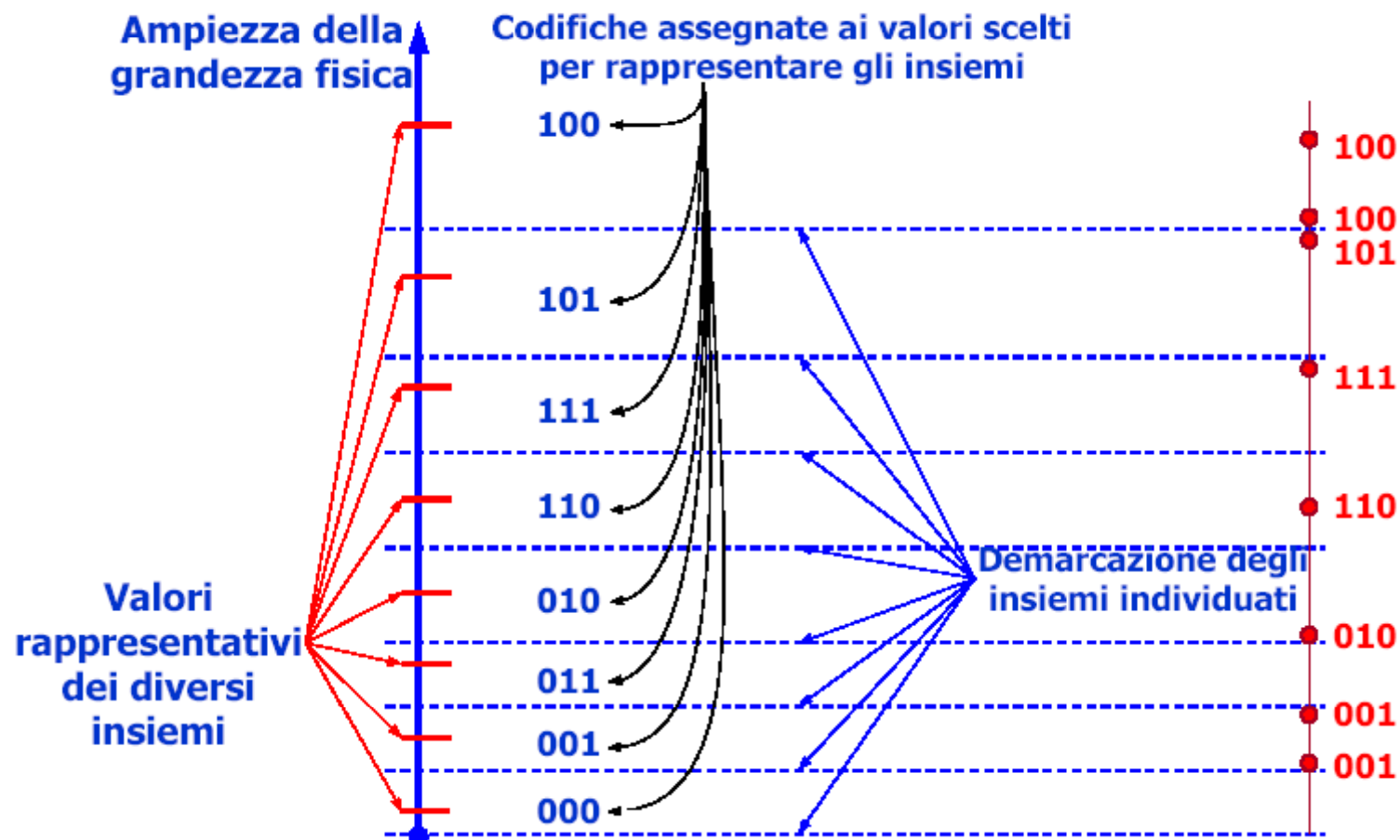
“3 kg” = 0110011 0100000 1101011 1100111

CODIFICA ANALOGICA E CODIFICA DIGITALE



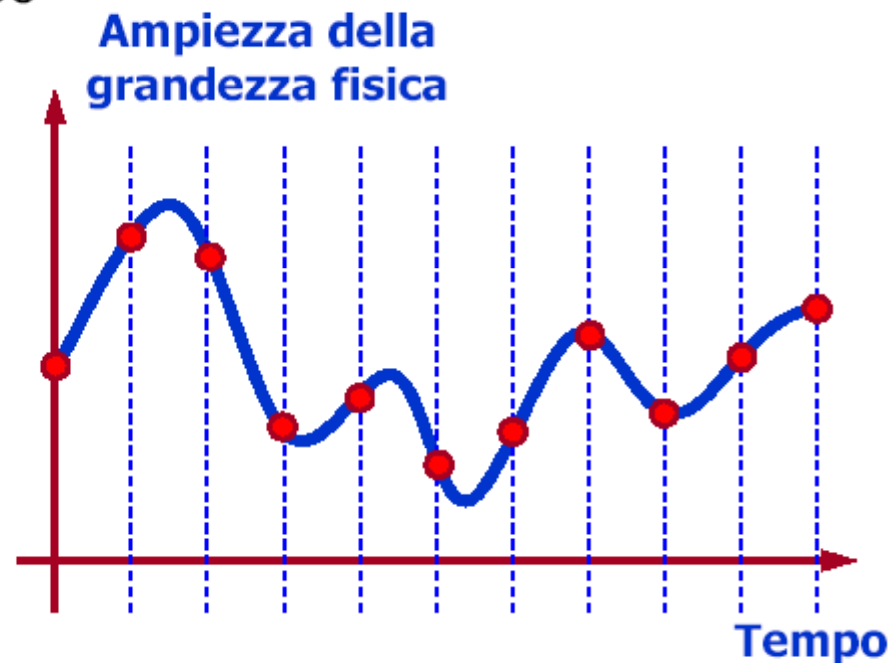
- ◆ **Codifica analogica:** le configurazioni possono variare in maniera continua su un insieme prefissato; esiste una relazione di **analogia** tra l'insieme delle configurazioni e l'insieme delle informazioni
- ◆ **Codifica digitale:** le entità di informazione vengono codificate mediante configurazioni convenzionali

QUANTIZZAZIONE

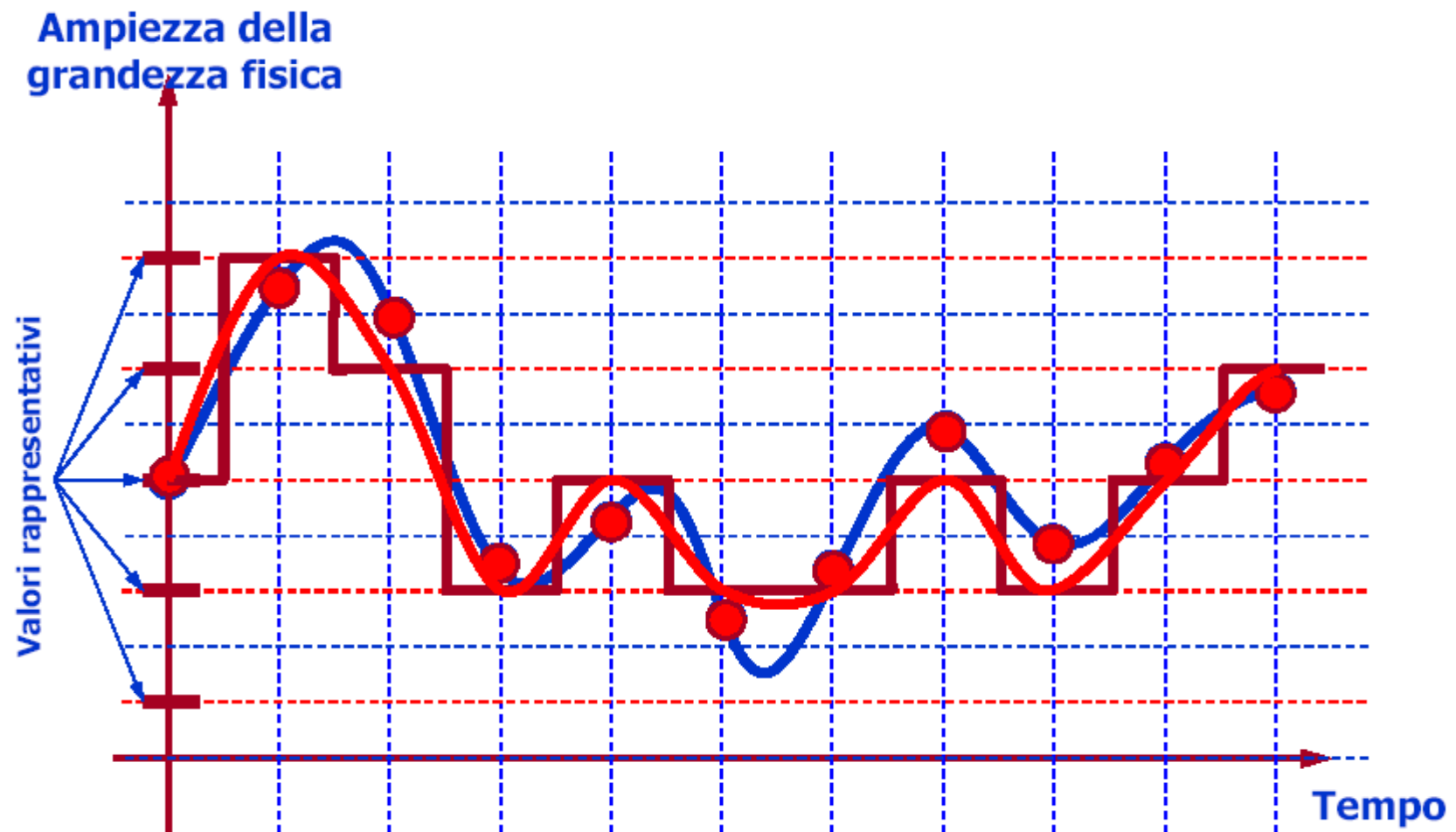


CAMPIONAMENTO

- La grandezza varia nel tempo e non può essere rappresentata da un solo valore.
- I valori di riferimento debbono essere rilevati in diversi istanti di tempo (frequenza di campionamento).
- La quantizzazione deve poi essere ripetuta per ogni valore campionato.



DIGITALIZZAZIONE



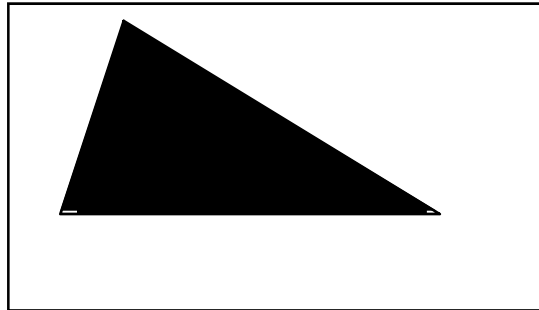
VANTAGGI DELLA CODIFICA DIGITALE

- **Rumore:** effetto dell'ambiente sul supporto.
- Quanto un supporto è "immune" al rumore?
 - Codifica analogica: ogni configurazione è lecita dal punto di vista informativo e quindi risulta impossibile distinguere il rumore dal segnale.
 - Codifica digitale: un valore binario è associato a un insieme di configurazioni valide quindi si può
 - riconoscere il rumore che porta in configurazioni non lecite
 - trascurare il rumore che non fa uscire il segnale dall'insieme associato alla stessa configurazione

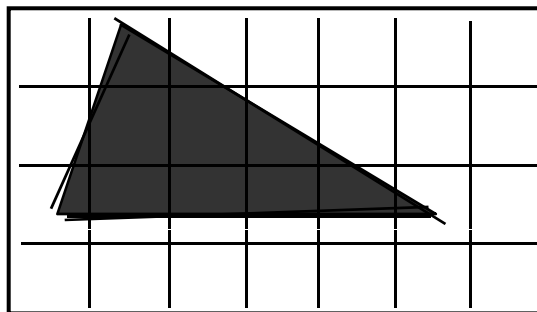


CODIFICA DI IMMAGINI

- ◆ Consideriamo un'immagine in bianco e nero, senza ombreggiature o livelli di chiaroscuro



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

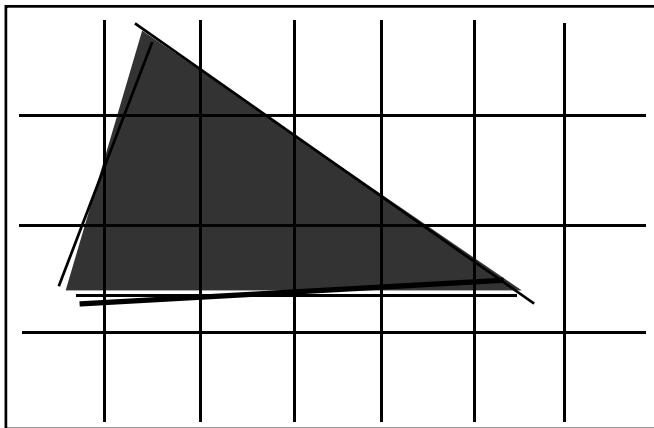


CODIFICA DI IMMAGINI

- ◆ 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 bianco (in cui il bianco è predominante)
 - il simbolo “1” viene utilizzato per la codifica di un pixel corrispondente ad un quadratino nero (in cui il nero è predominante)

CODIFICA DI IMMAGINI

- ◆ Poiché una sequenza di bit è lineare, si deve definire una convenzione per **ordinare** i pixel della griglia
- ◆ **ipotesi**: assumiamo che i pixel siano ordinati dal basso verso l'alto e da sinistra verso destra



0 ₂₂	1 ₂₃	0 ₂₄	0 ₂₅	0 ₂₆	0 ₂₇	0 ₂₈
0 ₁₅	1 ₁₆	1 ₁₇	0 ₁₈	0 ₁₉	0 ₂₀	0 ₂₁
0 ₈	1 ₉	1 ₁₀	1 ₁₁	1 ₁₂	0 ₁₃	0 ₁₄
0 ₁	0 ₂	0 ₃	0 ₄	0 ₅	0 ₆	0 ₇

La rappresentazione della figura è data dalla stringa binaria

0000000 0111100 0110000 0100000

CODIFICA DI IMMAGINI A COLORI

- ◆ Il numero di byte richiesti dipende dalla **risoluzione** e dal **numero di colori** che ogni pixel può assumere
- ◆ *Es:* per distinguere **256** colori sono necessari 8 bit per la codifica di ciascun pixel
 - la codifica di un'immagine formata da 640×480 pixel richiederà 2457600 bit (307200 byte)
- ◆ I monitor tipici utilizzano
 - risoluzione: 640×480 , 1024×768 , 1280×1024
 - numero di colori per pixel: da 256 fino a 16 milioni

COMPRESSIONE DEI DATI

◆ Lossless

- Senza perdita di informazione
- Programmi, documenti

◆ Lossy

- Con perdita di informazione
- Rapporto di compressione variabile dall'utente
- Immagini: GIF, JPEG (elimina lievi cambiamenti di colore)

COMPRESSIONE DEI DATI

◆ Esempio:

- {A, C, G, T}
- A=00, C=01, G=10, T=11
- ATTACCGAAACTTCTCTCGGGTG... → 1 milione caratteri = 2 milioni di bit
- 00111100010110...
- $\text{fr}(A)=50\%$, $\text{fr}(C)=25\%$, $\text{fr}(G)=12.5\%$, $\text{fr}(T)=12.5\%$
- A=0, C=10, G=110, T=111
- 011111101010110...
- $1 \times 50\% + 2 \times 25\% + 2 \times 3 \times 12.5\% \times 10^6 = 1.75$ milioni di bit → risparmio di 250000 bit!
- La nuova successione binaria deve essere decodificabile

JPEG: Fattore qualità 90/100 (253KB)

800x600

16,8mln
colori
24 bit

Bitmap:
1440000
byte

JPEG:
258971
byte



JPEG: Fattore qualità 50/100 (30KB)



JPEG: Fattore qualità 25/100 (20KB)



JPEG: Fattore qualità 10/100 (12KB)



JPEG: Fattore qualità 1/100 (9KB)



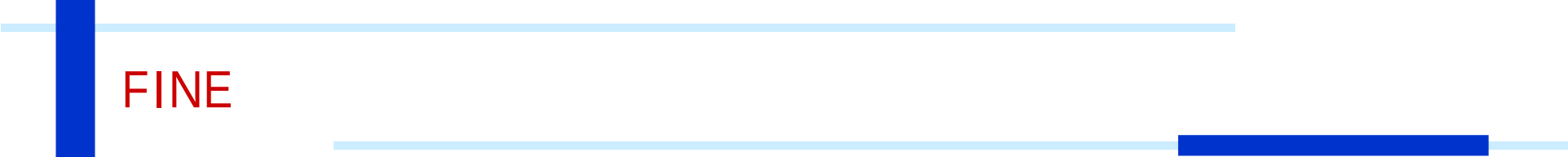
CODIFICA DELLE ISTRUZIONI

Istruzioni aritmetico-logiche	
Codice	Istruzione
01100000	ADD
01100100	SUB
01111110	AND
... ..	

Codice Operativo			
Codice Operativo	Indirizzo 1		
Codice Operativo	Indirizzo 1	Indirizzo 2	
Codice Operativo	Indirizzo 1	Indirizzo 2	Indirizzo 3

◆ Linguaggio macchina

- A ogni istruzione è assegnato un codice univoco, detto **codice operativo**
- E' necessario specificare dove leggere gli **operandi** (dati) dell'istruzione e dove scrivere il risultato
- Il numero di dati che ogni istruzione manipola è variabile in funzione dell'istruzione stessa



FINE