

Corso di Architettura degli Elaboratori e Laboratorio (M-Z)

# Rappresentazione binaria di dati complessi

*Nino Cauli*



UNIVERSITÀ  
degli STUDI  
di CATANIA

Dipartimento di Matematica e Informatica

## Un sistema di numerazione è definito da:

- Un intero  $B$  detto **BASE**
- Un insieme di  $B$  simboli  $S_B = \{s_0, \dots, s_{B-1}\}$ , ognuno dei quali rappresenta le quantità **0,1,2,...,B-1**

Un numero a  $n$  cifre  $p_{(n-1)}p_{(n-2)}\dots p_1p_0$  con  $p_{(i)} \in S_B$  e  $i=0,\dots,n-1$  può essere rappresentato come **SOMMA DI POTENZE DELLA BASE**:

$$\sum_{i=0}^{n-1} (p_{(i)} \cdot B^i)$$

$$\mathbf{P} = p_{(n-1)}p_{(n-2)}\cdots p_1p_0, \quad p_{(i)} \in \{0,1\} \text{ e } i=0,\dots,n-1$$

$$\sum_{i=0}^{n-1} p_{(i)} \cdot 2^i$$

Numero di valori rappresentabili =  **$[0, 2^n)$**

## Segno:

- Bit più a sinistra = BIT DI SEGNO

## Valore assoluto:

- SEGNO E VALORE ASSOLUTO
- COMPLEMENTO A UNO
- COMPLEMENTO A DUE

**ADDIZIONE:** come quella binaria naturale, ma trascurando il riporto in uscita

**SOTTRAZIONE:** addizione con il complemento a due del sottraendo

(a)	$\begin{array}{r} 0010 + (+2) \\ 0011 = (+3) \\ \hline 0101 \quad (+5) \end{array}$		(b)	$\begin{array}{r} 0100 + (+4) \\ 1010 = (-6) \\ \hline 1110 \quad (-2) \end{array}$
(c)	$\begin{array}{r} 1011 + (-5) \\ 1110 = (-2) \\ \hline 1001 \quad (-7) \end{array}$		(d)	$\begin{array}{r} 0111 + (+7) \\ 1101 = (-3) \\ \hline 0100 \quad (+4) \end{array}$
(e)	$\begin{array}{r} 1101 - (-3) \\ 1001 = (-7) \\ \hline \end{array}$	$\Rightarrow$		$\begin{array}{r} 1101 + \\ 0111 = \\ \hline 0100 \quad (+4) \end{array}$
(f)	$\begin{array}{r} 0010 - (+2) \\ 0100 = (+4) \\ \hline \end{array}$	$\Rightarrow$		$\begin{array}{r} 0010 + \\ 1100 = \\ \hline 1110 \quad (-2) \end{array}$

**Figura 1.6 (parte)** - Operazioni di addizione e sottrazione in complemento a due

Il risultato di addizione e sottrazione in complemento a 2 è corretto se è **COMPRESO** nell'intervallo:

$$[-2^{n-1}, 2^{n-1})$$

In caso contrario avviene un evento di **TRABOCCO**

Il **TRABOCCO** può avvenire solo se:

- 1) I due addendi sono **CONCORDI IN SEGNO**
- 2) Il **BIT DI SEGNO** della somma degli addendi è **DIVERSO** da quello degli addendi

Spesso si presenta la necessità di aumentare o diminuire il numero di bit usati per codificare un numero

Regole molto semplici:

- **ESTENSIONE DEL SEGNO**: si replica a sinistra il bit del segno tante volte quanto occorre
- **RIDUZIONE DEL SEGNO**: si rimuove il bit più a sinistra tante volte quante occorre, purché il bit successivo abbia ugual valore

Come rappresentare i **numeri frazionari** in binario?

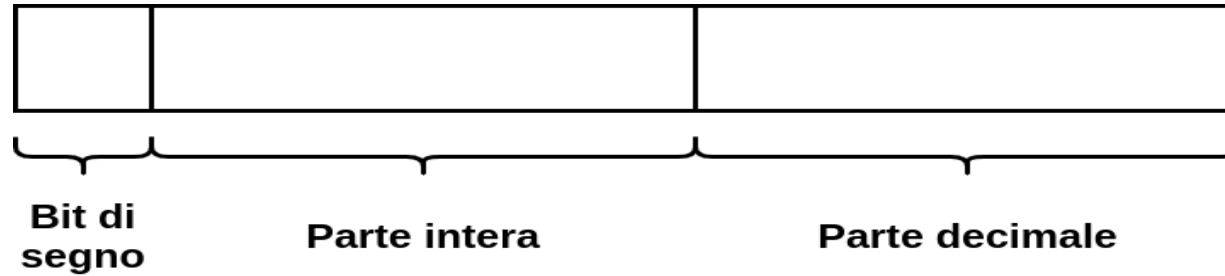
## Idea semplice:

- Un **bit di segno**
- Una **porzione di bit fissa** per la **parte intera**
- Una **porzione di bit fissa** per la **parte decimale**

Chiamiamo questa rappresentazione **A VIRGOLA FISSA (FIXED POINT)**



# Numeri a virgola fissa (fixed point)



## Solo bit di segno e parte intera:

- Valori rappresentabili: da  $-2^{n-1}$  a  $2^{n-1} - 1$
- Risoluzione:  $1$

## Solo bit di segno e parte decimale:

- Valori rappresentabili: da  $-1$  a  $1 - 2^{-(n-1)}$
- Risoluzione:  $2^{-(n-1)}$

**Intervallo non sufficiente per calcoli scientifici.**

Per aumentare intervallo e risoluzione di valori rappresentabili si potrebbe spostare la posizione della virgola dinamicamente (**VIRGOLA MOBILE**)

Notazione scientifica decimale, **FORMA NORMALE**:

$$6,0247 \times 10^2 = 602,47$$

$$3,7291 \times 10^{-2} = 0,037291$$

In generale vale per **ogni base**:

$$1,0011 \times 2^2 = 100,11$$

$$4,2131 \times 5^{-2} = 0,042131$$

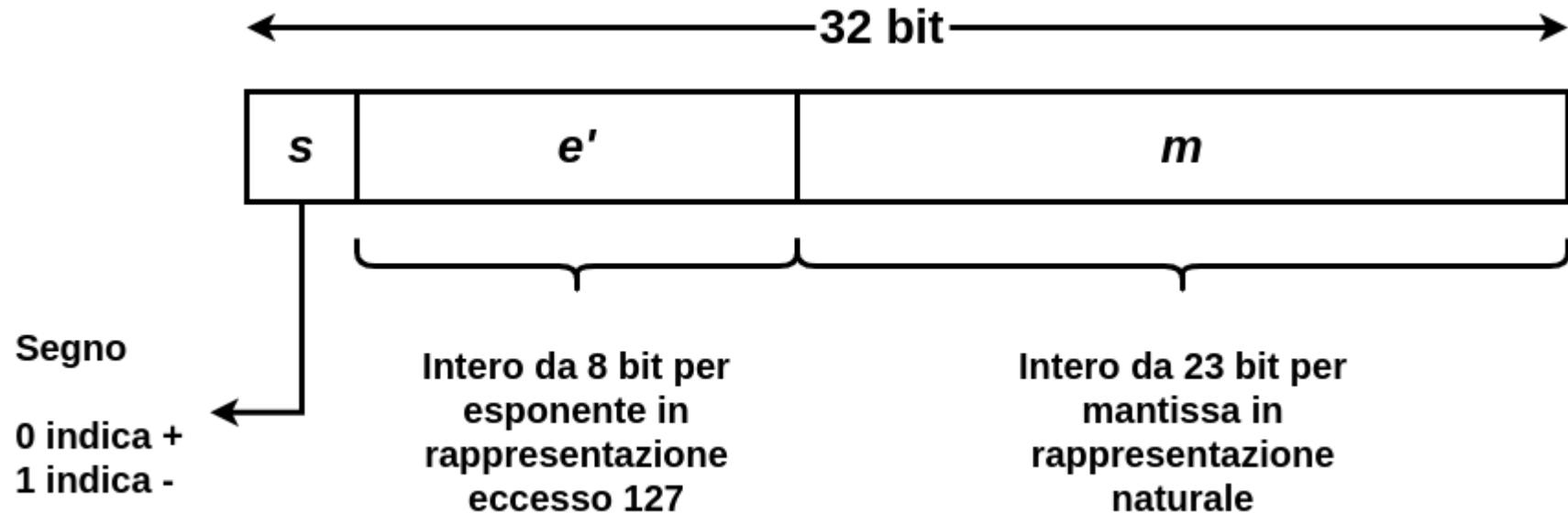
Un numero binario in virgola mobile può quindi essere rappresentato:

- Un **SEGNO**  $s$  per il numero
- La **MANTISSA**  $m$  (bit significativi escluso il bit più significativo)
- Un **ESPONENTE**  $e$  con segno in base 2

$$\text{Valore rappresentato} = \pm 1, m \times 2^e$$

# Formato precisione singola (32 bit)

Standard **IEEE 754** numeri **32 bit**



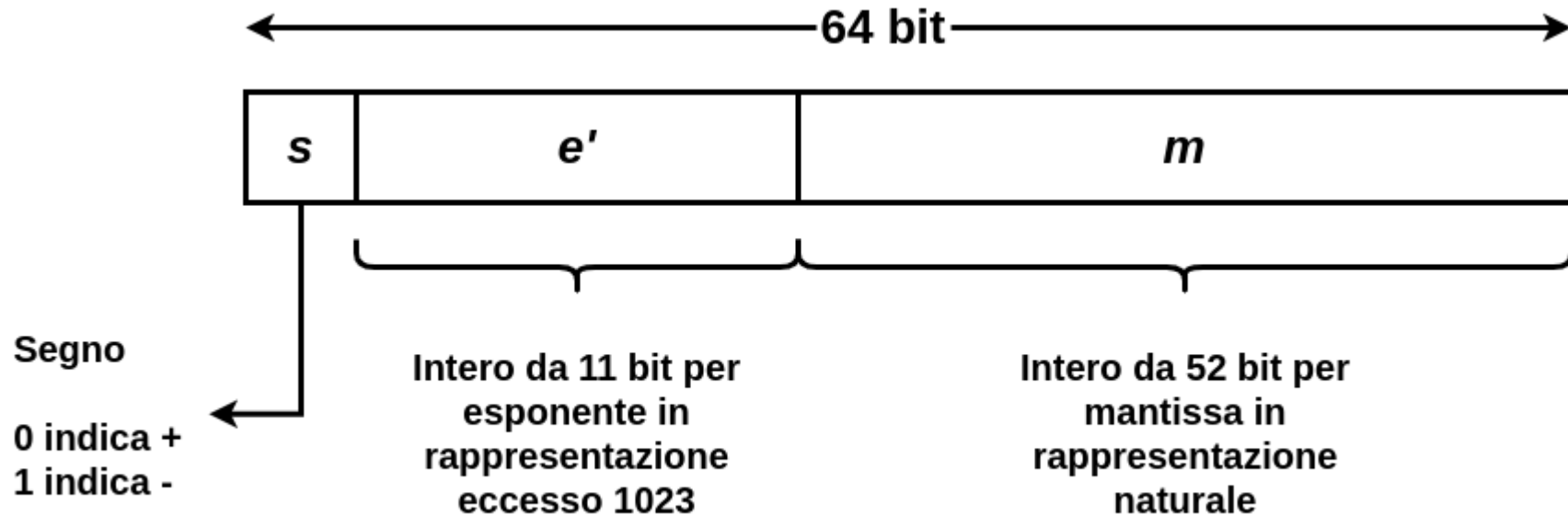
$$0 \leq e' \leq 255 \quad e' = e - 127$$

Valori speciali:  $e' = 0$ ,  $e' = 255$       Intervallo esponente:  $-126 \leq e \leq 127$

Valori rappresentabili nell'intervallo:  $[2^{-126}, 2^{127}]$

# Formato precisione doppia (64 bit)

Standard **IEEE 754** numeri **64 bit**



$$0 \leq e' \leq 2047 \quad e' = e - 1023$$

Valori speciali:  $e' = 0$ ,  $e' = 2047$       Intervallo esponente:  $-1022 \leq e \leq 1023$

Valori rappresentabili nell'intervallo:  $[2^{-1022}, 2^{1023}]$

Alcuni valori dell'esponente sono speciali:

- $e' = 0, m = 0$  rappresenta lo **0 esatto**
- $e' = 255(2047), m = 0$  rappresenta l'infinito  $\infty$
- $e' = 0, m \neq 0$  rappresenta la **forma non normale**:  $\pm 0, m \times 2^{-126(-1022)}$
- $e' = 255(2047), m \neq 0$  rappresenta **Not a Number NaN**

## Come rappresentare caratteri tramite una sequenza di $n$ bit?

- Associamo un carattere ad ogni possibile valore binario rappresentabile

## Quanti caratteri siamo in grado di rappresentare con $n$ bit?

- Una sequenza di  $n$  bit può rappresentare  $2^n$  permutazioni di 0 e 1
- Si può rappresentare un alfabeto di  $2^n$  simboli

## Vediamo gli standard più usati

- **Codice ASCII** (American Standard Code for Information Interchange)
- Rappresenta lettere, cifre decimali, punteggiatura e caratteri speciali
- Definito su **7 bit** → alfabeto di  **$2^7 = 128$  elementi**
- Lettere e numeri con codici in ordine crescente

	Bit 654							
Bit 3210	000	001	010	011	100	101	110	111
0000	NUL	DLE	SPACE	0	@	P	'	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	\$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	'	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	/	l	
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL



## Necessità di codici più ricchi per gestire le diverse lingue con caratteri speciali, accenti, etc.

### Standard internazionali:

- Famiglia **ISO 8859-x**: estendono il codice ASCII usando 8 bit (doppio dei simboli)
- **ISO/IEC 10646 (UCS)**: rappresentazione universale di caratteri che estende su più byte la ISO 8859
- Standard di codifica basati su UCS: come ad esempio **UNICODE** e **UTF-8**

## L'informazione binaria codificata potrebbe essere affetta da errori

### Gli errori possono essere causati da:

- Disturbi nei canali di trasmissione
- Alterazione accidentale o dolosa nei dispositivi di memorizzazione

### Come si possono individuare e possibilmente correggere gli errori?

- Aggiungendo uno o più bit di controllo alla sequenza originale

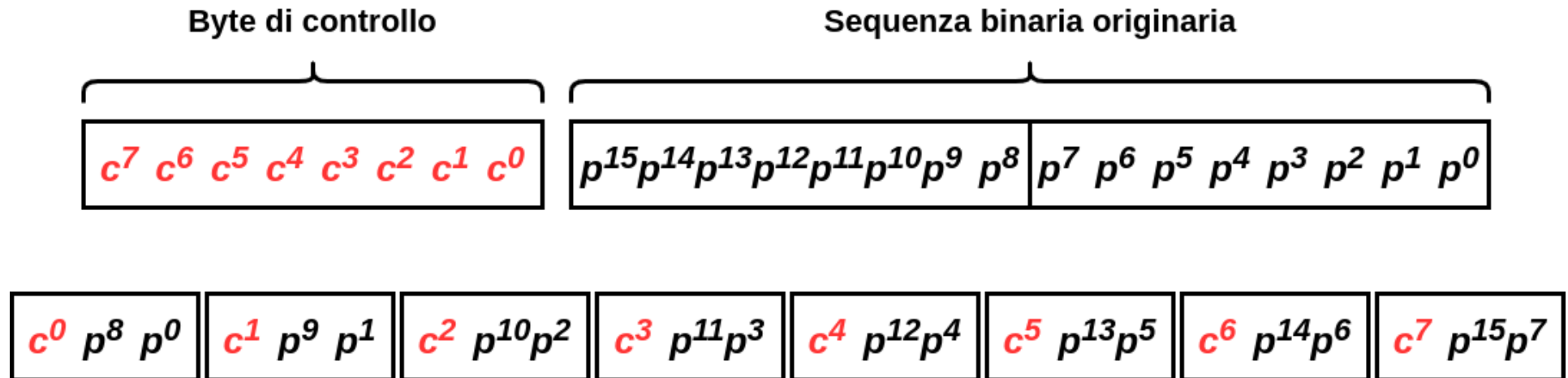
Si aggiunge un bit di controllo (**BIT DI PARITÀ**) alla sequenza binaria originaria:

- **0** se numero **PARI** di 1 nella sequenza originaria
- **1** se numero **DISPARI** di 1 nella sequenza originaria

Se il numero di 1 nella **sequenza binaria estesa** con il bit di parità è **DISPARI** sono presenti errori

Rivela solo **errori su di 1 bit** e **non può correggere** gli errori rilevati

- Estensione dell'idea di bit di controllo per **sequenze di più byte**
- **1 byte** usato come informazione di controllo
- Ogni bit del byte controllo usato per la parità di **sequenze non contigue** di bit a **distanza 8**



Rivela **più errori su in uno stesso byte**

**Checksum:** Estensione con sequenze di controllo più lunghe di un byte

- Algoritmi di controllo **semplici meno robusti**:
  - Somma dei bit
  - Parità di sotto-sequenze
- **CRC** (codice a ridondanza ciclica):
  - Algoritmi di controllo basati su **algebra dei polinomi**
  - Permettono di rilevare errori su **lunghe sequenze di bit contigui**

Codici di controllo con informazione necessaria per **correggere** fino a  $k$  errori sui bit

Si può usare il concetto di **DISTANZA DI HAMMING**

## Distanza di Hamming:

- **Di due sequenze**: numero di bit diversi in posizioni corrispondenti
- **Di un insieme di sequenze**: la distanza di Hamming minima di coppie di sequenze distinte nell'insieme

Un **codice**  $c$  a  $n$  bit che può rappresentare un **alfabeto**  $A$  di al più  $2^n$  simboli è una funzione iniettiva:

$$c : a \rightarrow \{0, 1\}^n$$

**Distanza di Hamming di un codice** = distanza di Hamming della sua immagine

**Un codice con:**

- Distanza di Hamming  $h$  può **rivelare** fino a  $h - 1$  errori
- Distanza di Hamming  $h = 2k + 1$  può **correggere** fino a  $k$  errori

# Esempio distanza di Hamming

X	4 bit	7 bit	$h_{0100011}$
0	0000	00 0 0 000	3
1	0001	11 0 1 001	3
2	0010	01 0 1 010	2
3	0011	10 0 0 011	2
4	0100	10 0 1 100	6
5	0101	01 0 0 101	2
6	0110	11 0 0 110	3
7	0111	00 0 1 111	3
8	1000	11 1 0 000	4
9	1001	00 1 1 001	4
A	1010	10 1 1 010	5
<b>B</b>	<b>1011</b>	<b>01 1 0 011</b>	<b>1</b>
C	1100	01 1 1 100	5
D	1101	10 1 0 101	5
E	1110	00 1 0 110	4
F	1111	11 1 1 111	4

**Tabella A1.1** - Esempio di correzione di errore



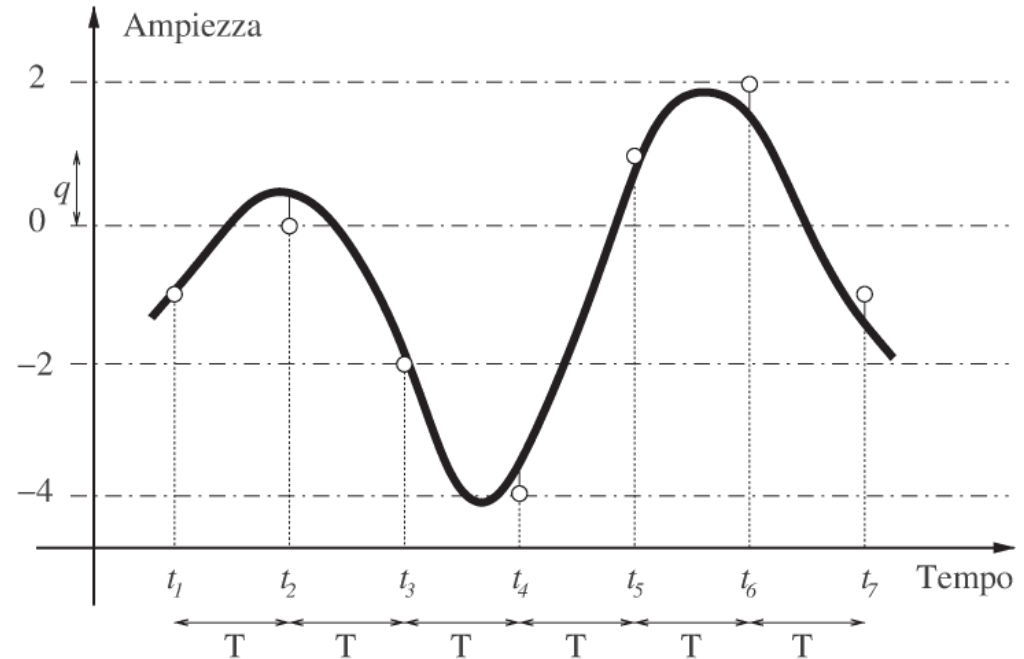
Informazione multimediale è originariamente **ANALOGICA**

- **AUDIO**: andamento pressione acustica nel tempo
- **IMMAGINI**: distribuzione di valori continui di luminanza in uno spazio bidimensionale
- **VIDEO**: sequenza temporale di immagini

Per essere rappresentata in biniario l'informazione multimediale deve essere **DISCRETIZZATA**

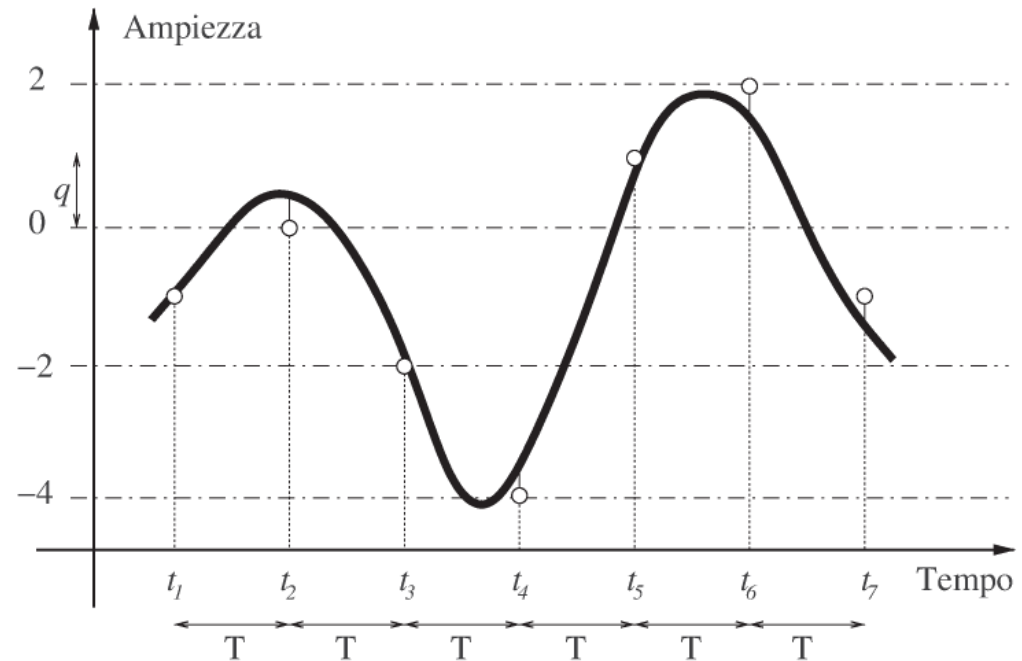
La discretizzazione nel tempo è chiamata **CAMPIONAMENTO**

- Segnale campionato a **intervalli di tempo regolari**
- Intervallo di tempo detto **PERIODO  $T$**
- **FREQUENZA** di campionamento è il reciproco del periodo
- La frequenza è espressa in **hertz (Hz)**, campionamenti al secondo



La discretizzazione nell'intensità del segnale è chiamata  
**QUANTIZZAZIONE**

- Unità di quantizzazione è detta  $q$
- I valori analogici del segnale vengono **approssimati** al valore **multiplo di  $q$  più vicino**
- Valori quantizzati rappresentati dai **coefficienti di  $q$**



- Intervallo fisiologico di frequenze udibili dall'uomo: da circa **20 Hz a 20 kHz**
- Per rappresentare un segnale analogico di frequenza massima  $f_m$ , un segnale digitale deve essere campionato con frequenza  $f_s > 2f_m$  (**Teorema del campionamento di Nyquist-Shannon**)

## Frequenze di campionamento audio:

- Musica: **44,1 kHz**
- Voce: **8 kHz** (non necessario frequenze alte)

**Unità di quantizzazione** solitamente codificata con **16 bit per canale**

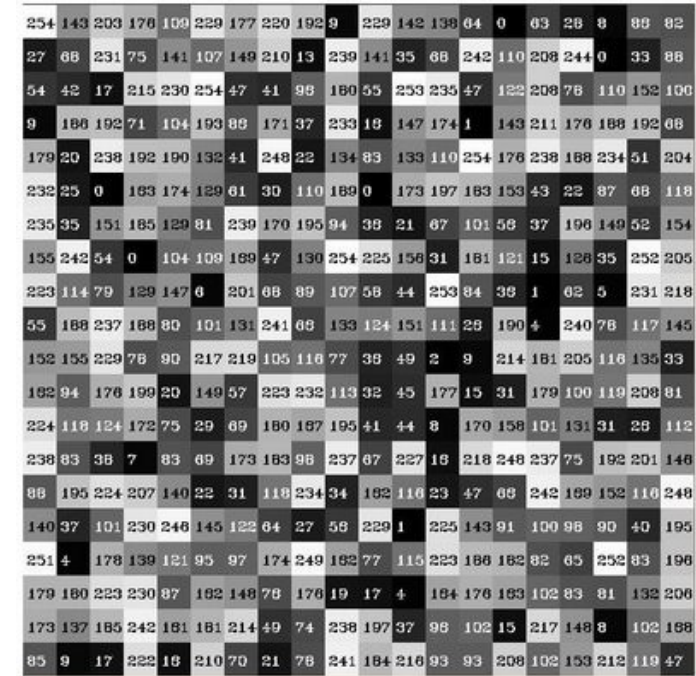
- Spazio occupato da **4 min** di audio stereo:

$$f_s \times \text{sec} \times \text{canali} \times \text{bit quantizzazione} = 44,100 \times 240 \times 2 \times 32 \approx$$

**40 Mbyte**

- Codifica **MIDI** risparmia spazio rappresentando il suono tramite i **parametri di un sintetizzatore** (strumento emulato)

- Il campionamento avviene nel **domino dello spazio**
- Una **griglia** viene applicata ad una **regione limitata bidimensionale** con elementi detti **PIXEL**
- Ogni pixel rappresenta la **luminanza**, un valore per ogni canale (**livelli grigio o RGB**)
- La **RISOLUZIONE** rappresenta la precisione di campionamento (**numero o densità dei pixel**)
- La **PROFONDITÀ DI COLORE** rappresenta i livelli di quantizzazione (spesso **8 bit per canale**)



254	143	203	176	100	229	177	220	192	9	229	142	139	64	0	63	28	8	88	82
27	68	231	75	141	107	149	210	13	239	141	35	68	242	110	208	244	0	33	88
54	42	17	215	230	254	47	41	98	180	55	253	235	47	122	208	78	110	152	106
9	186	192	71	104	193	88	171	37	233	18	147	174	1	143	211	176	188	192	68
179	20	238	192	190	132	41	248	22	134	83	133	110	254	176	238	188	234	51	204
232	25	0	163	174	129	61	30	110	189	0	173	197	183	153	43	22	87	68	118
235	35	151	185	129	81	239	170	195	94	38	21	67	101	56	37	190	149	52	154
155	242	54	0	104	109	169	47	130	254	225	156	31	181	121	15	128	35	252	205
223	114	79	129	147	6	201	68	89	107	58	44	253	84	36	1	62	5	231	218
55	188	237	188	80	101	131	241	68	133	124	151	111	28	190	4	240	78	117	145
152	155	229	78	90	217	219	105	116	77	38	49	2	9	214	181	205	116	135	33
182	94	176	199	20	149	57	223	232	113	32	45	177	15	31	179	100	119	208	81
224	118	124	172	75	29	69	180	187	195	41	44	8	170	158	101	131	31	28	112
238	83	38	7	83	69	173	183	98	237	67	227	16	218	248	237	75	192	201	146
88	195	224	207	140	22	31	118	234	34	182	116	23	47	66	242	189	152	116	248
140	37	101	230	246	145	122	64	27	58	229	1	225	143	91	100	98	90	40	195
251	4	178	139	121	95	97	174	249	182	77	115	223	186	182	82	65	252	83	196
179	180	223	230	37	192	148	78	176	19	17	4	184	176	193	102	83	81	132	206
173	137	185	242	181	181	214	49	74	238	197	37	98	102	15	217	148	8	102	188
85	9	17	222	16	210	70	21	78	241	184	216	93	93	208	102	153	212	119	47

- Immagine non compressa in formato bitmap
- Esempio di spazio occupato da un immagine a colori

$$\text{Colonne} \times \text{righe} \times \text{canali} \times \text{bit profondità} = 1024 \times 768 \times 3 \times 8 \approx 2,25 \text{ Mbyte}$$

- Rappresentazione vettoriale (*i.e.* SVG) descrive l'immagine tramite **primitive geometriche** (più compatta e robusta al ridimensionamento)

- Un video è una **sequenza temporale di immagini**
- **24 immagini al secondo**: frequenza minima per percepire il movimento

$$24 \times 2,25 \text{ Mbyte} = 54 \text{ Mbyte / secondo}$$

**Informazione video necessita di essere compressa**



Rappresentare l'informazione con sequenze binarie più brevi perdendo poca o nessuna informazione

## 2 FASI NELLA COMPRESSIONE:

- Codifica:  $c(s) = t$  generazione stringa compressa  $t$  da  $s$
- Decodifica:  $d(t) = s$  ritorno alla stringa originaria  $s$  da  $t$

## LA COMPRESSIONE PUÒ ESSERE:

- Senza perdita (**lossless**):  $d(c(s)) = s$
- Con perdita (**lossy**):  $d(c(s)) = s + \varepsilon$

$$\text{RAPPORTO DI COMPRESSIONE} = \text{len}(s) / \text{len}(c(s))$$

## RUN-LENGTH

- Codificare numero di occorrenze consecutive di ciascun simbolo

## DIFFERENZIALE (RELATIVA)

- Spezzare la sequenza in blocchi, codificare il primo e codificare i successivi come differenze dal precedente

## VARIABLE-LENGTH

- Lunghezza codifica dei simboli inversamente proporzionale alla probabilità di incontrarli (**codici di Huffman**)

## BASATA SU DIZIONARIO, ADATTIVA

- I codici rappresentano parole di un dizionario
- Adattiva: il dizionario è creato dinamicamente (**algoritmo LZW**)

## Audio

- **MP3**: dati musicali
- **OGG/VORBIS**: dati musicali (più efficace ma meno usato dell'MP3)
- **Speex**: compressione del parlato

## Immagini

- **GIF**: Graphical Interchange Format (Lossless)
- **PNG**: Portable Network Graphics (Lossless)
- **JPEG**: Joint Photographic Experts Group (Lossy)

## Video

- **MPEG**: Motion Picture Experts Group
  - Compressione video: **JPEG** + codifica relativa
  - Compressione audio: **MP3**