Rappresentazione dell’informazione

Rappresentare l'informazione

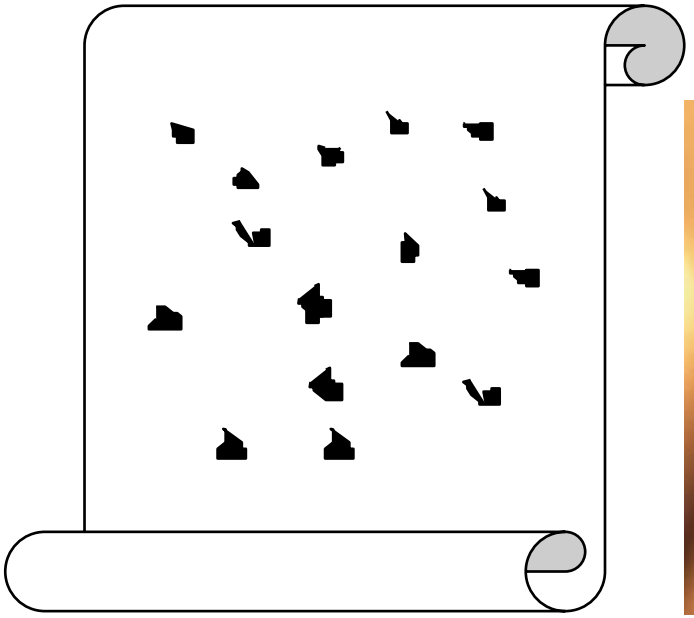
- Per elaborare l'informazione è necessario saperla rappresentare in una forma comprensibile per l'esecutore

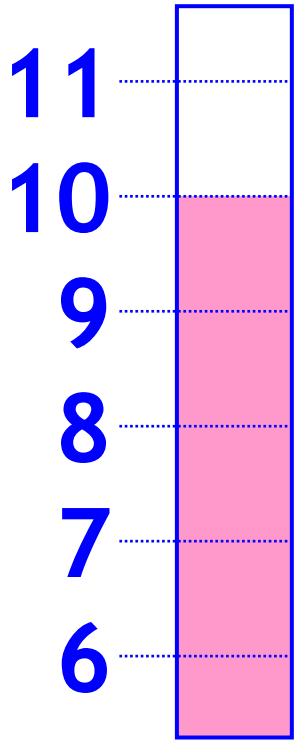
- Bisogna stabilire un ***codice*** che associa a ogni entità di informazione che si desidera rappresentare una configurazione del ***supporto*** su cui l'informazione è trasmessa

Codice e supporto

- Codice
  - E’ una regola che a ogni entità di informazione associa una configurazione del supporto su cui l’informazione viene trasmessa
- Supporto
  - Mezzo che può assumere almeno due configurazione diverse

Codice e supporto: esempi





Perché usare un codice?

- Permette l'elaborazione e la memorizzazione di entità non gestibili nella loro forma originale
  - i sistemi di numerazione sono codici
- Permette l'interpretazione dei simboli
  - un dizionario di lingua italiana è un codice
- Aggiunge proprietà ad un sistema di simboli
  - comprimere la lunghezza delle stringhe
  - aumentare l'affidabilità di trasmissione

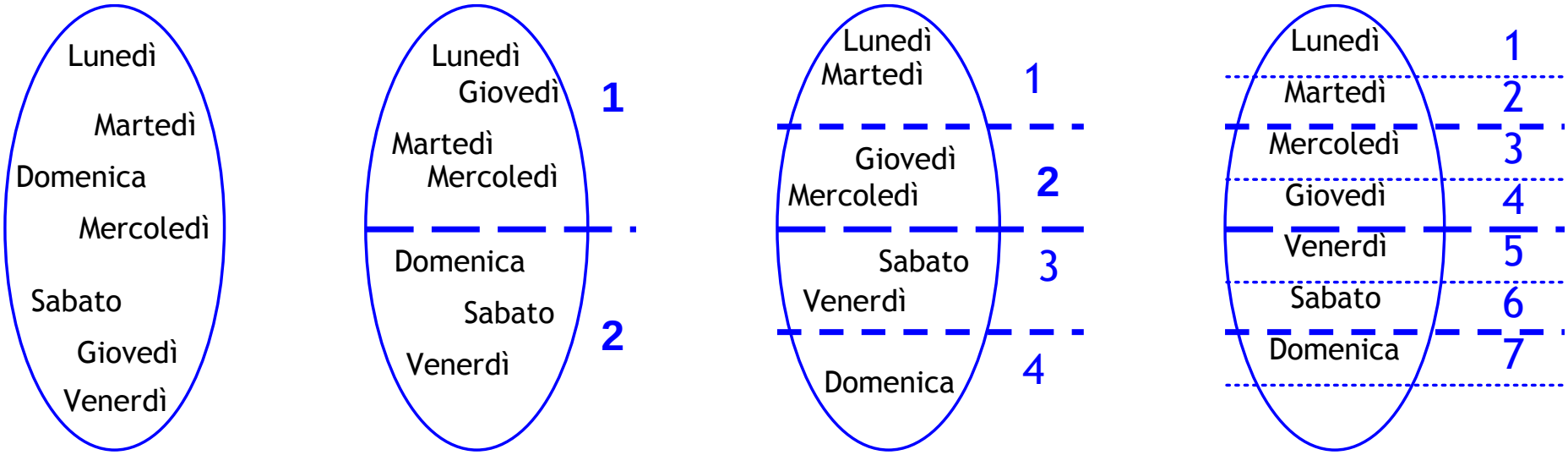
Che codice usa il calcolatore?

- Il calcolatore è in grado di distinguere solo due tipi di informazione:
  - sta passando corrente / non sta passando corrente
- Il calcolatore usa dispositivi che possono trovarsi in due soli stati
  - acceso / spento
- Quindi gli bastano due numeri (0,1) per rappresentare le uniche due entità di informazione che conosce
  - Il calcolatore usa un ***codice numerico binario (booleano)***

Codifica numerica

- Ad ogni entità di informazione può essere associato un numero (codice numerico)
  - Molte informazioni sono quantitative, e quindi rappresentabili in forma numerica
  - Le informazioni qualitative possono essere comunque associate a numeri tramite un opportuno codice

Codifica numerica: esempio



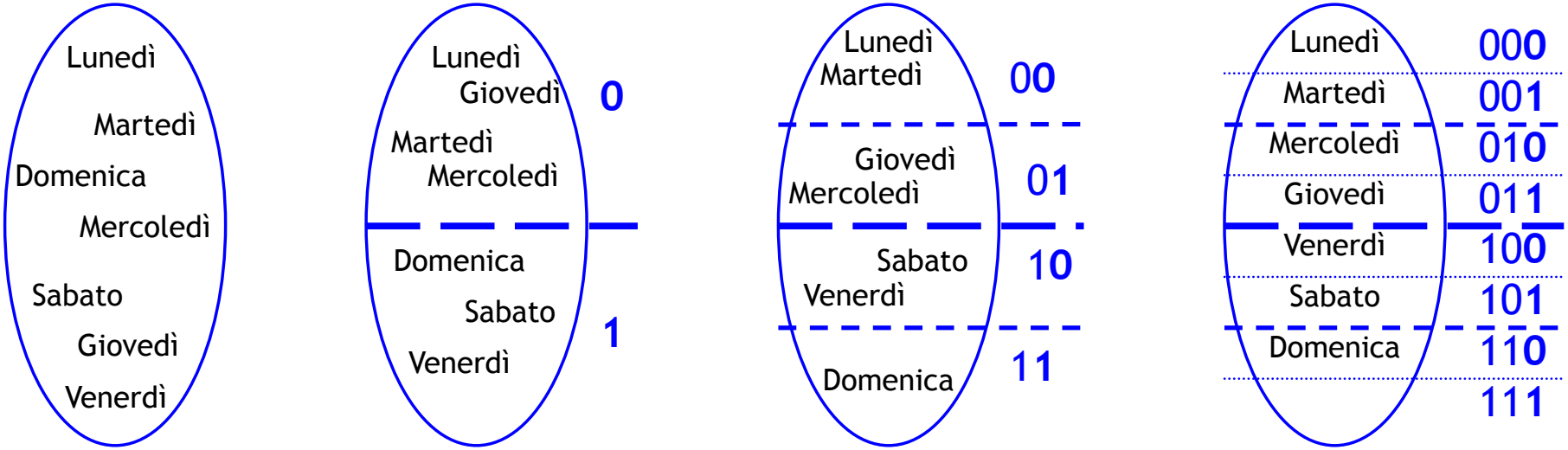
La codifica binaria

- Si usano solo i simboli 0 e 1 (binary digit = **bit**) concatenati in sequenze chiamate stringhe binarie

- Quanti oggetti posso contare con k bit:
  - 1 bit $\Rightarrow$ 2 stati (0, 1) $\Rightarrow$ 2 oggetti (e.g. Vero/Falso)
  - 2 bit $\Rightarrow$ 4 stati (00, 01, 10, 11) $\Rightarrow$ 4 oggetti
  - 3 bit $\Rightarrow$ 8 stati (000, 001, …, 111) $\Rightarrow$ 8 oggetti
  - …
  - k bit $\Rightarrow$ 2^k stati $\Rightarrow$ 2^k oggetti

- Quanti bit mi servono per contare N oggetti:
  - $N \leq 2^k \Rightarrow k \geq \log_2 N \Rightarrow k = \lceil \log_2 N \rceil$ (intero superiore)

Codifica binaria: esempio



bit, Byte, KiloByte, MegaByte, …

- bit = solo due stati, “0” oppure “1”
- Byte = 8 bit, quindi $2^8 = 256$ stati
- KiloByte [KiB] = 2^{10} Byte = 1024 Byte ~ 10^3 Byte
- MegaByte [MiB] = 2^{20} Byte = 1'048'576 Byte ~ 10^6 Byte
- GigaByte [GiB] = 2^{30} Byte ~ 10^9 Byte
- TeraByte [TeB] = 2^{40} Byte ~ 10^{12} Byte
- PetaByte [PeB] = 2^{50} Byte ~ 10^{15} Byte
- ExaByte [ExB] = 2^{60} Byte ~ 10^{18} Byte …

bit, Byte, KiloByte, MegaByte, …

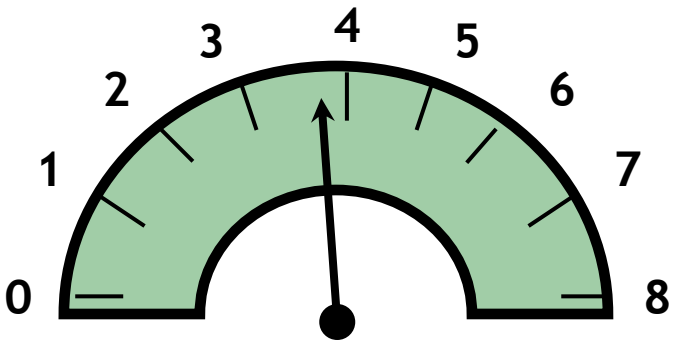
Grandezza	Nome	Simbolo		Dimensione	SI	Diff. %
Kilo binario	Kibi	Ki	2^{10}	1'024	10^3	2.40%
Mega binario	Mebi	Mi	$(2^{10})^2$	1'048'576	$(10^3)^2$	4.86%
Giga binario	Gibi	Gi	$(2^{10})^3$	1'073'741'824	$(10^3)^3$	7.37%
Tera binario	Tebi	Ti	$(2^{10})^4$	1'099'511'627'776	$(10^3)^4$	9.95%
Peta binario	Pebi	Pi	$(2^{10})^5$	1'125'899'906'842'624	$(10^3)^5$	12.59%
Exa binario	Exbi	Ei	$(2^{10})^6$	1'152'921'504'606'846'976	$(10^3)^6$	15.29%
Zetta binario	Zebi	Zi	$(2^{10})^7$	1'180'591'620'717'411'303'424	$(10^3)^7$	18.06%
Yotta binario	Yobi	Yi	$(2^{10})^8$	1'208'925'819'614'629'174'706'176	$(10^3)^8$	20.89%

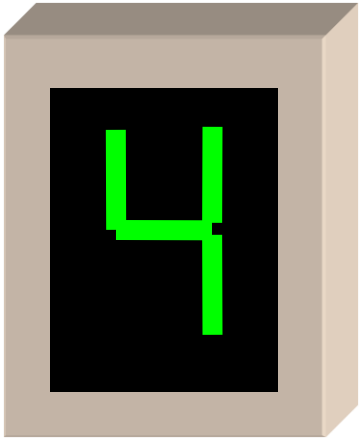
- capacità di un disco fisso: 120 GB = 111.76 GiB (!!)
- capacità di un floppy: 1.406 MiB = 1.475 MB

Riassumendo…

- Il calcolatore distingue solo 0 e 1
- Concatenando 0 e 1 posso rappresentare qualunque numero decimale
- Ad ogni numero posso associare un'entità di informazione diversa
- Qualunque informazione deve essere ***digitalizzata*** per poter essere rappresentata nel calcolatore

Analogico e digitale





Meta-informazione esplicita nel supporto:
il supporto ha una struttura corrispondente a quella presente tra entità di informazione.

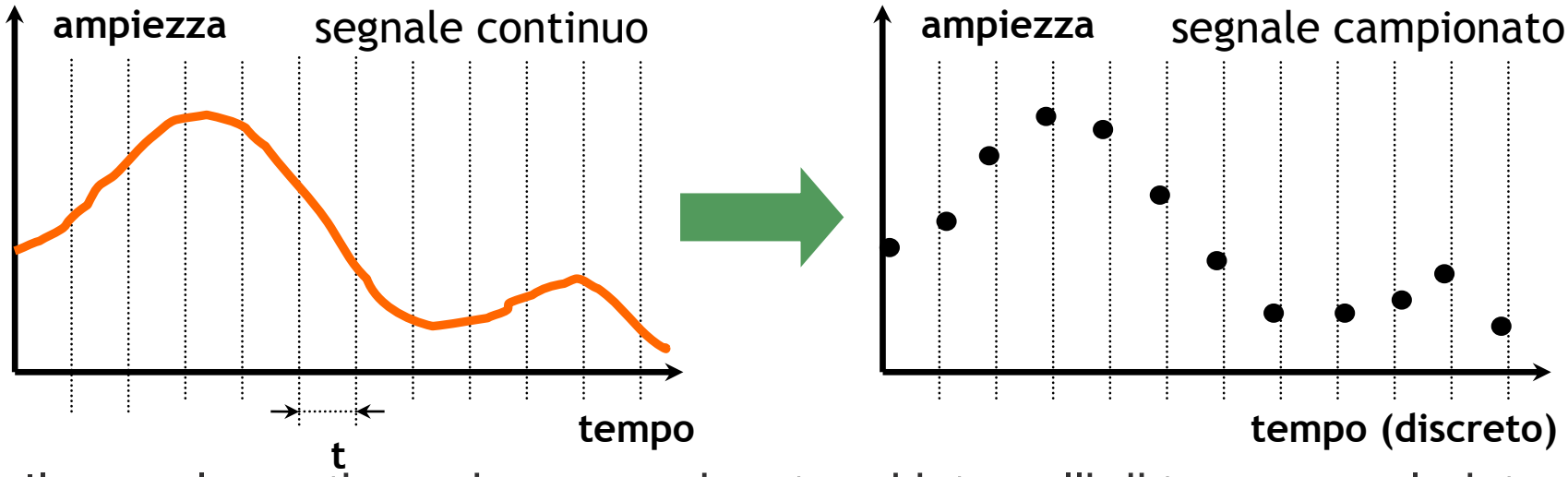
Meta-informazione implicita nella codifica:
al supporto si richiede solo di avere configurazioni molteplici e distinguibili.

Campionamento e quantizzazione

- Gli elaboratori elettronici hanno natura discreta, ovvero ogni grandezza in gioco può essere rappresentata soltanto da un numero finito di elementi

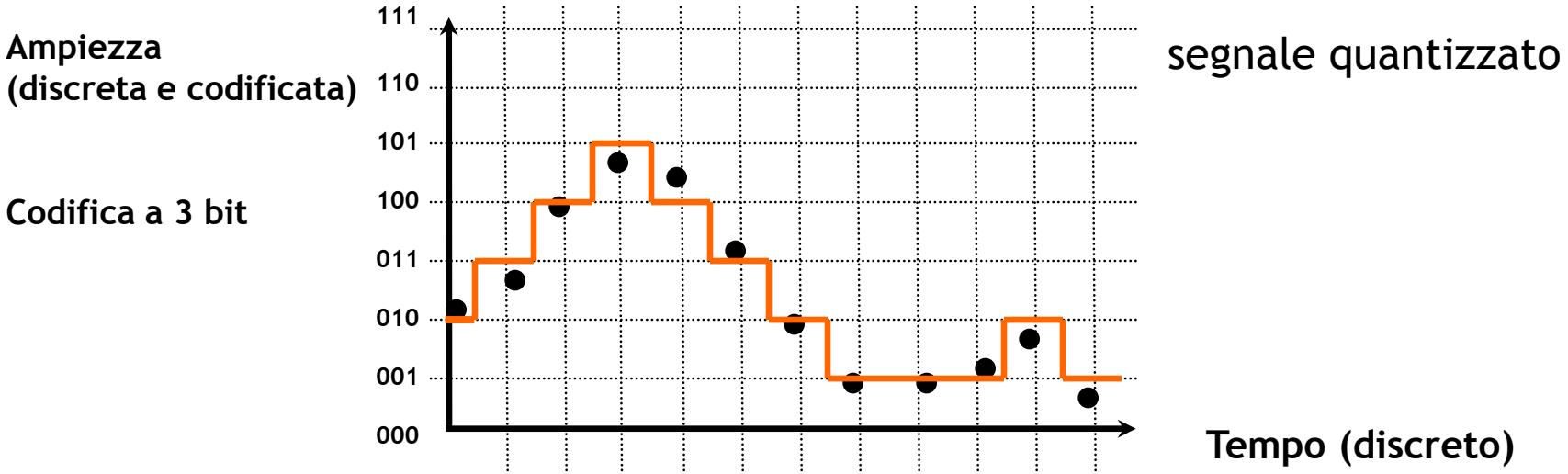
- Per essere elaborati da un calcolatore, segnali intrinsecamente continui quali suoni, immagini, video ecc., devono essere discretizzati (digitalizzati) attraverso operazioni di ***campionamento*** e ***quantizzazione***

Campionamento



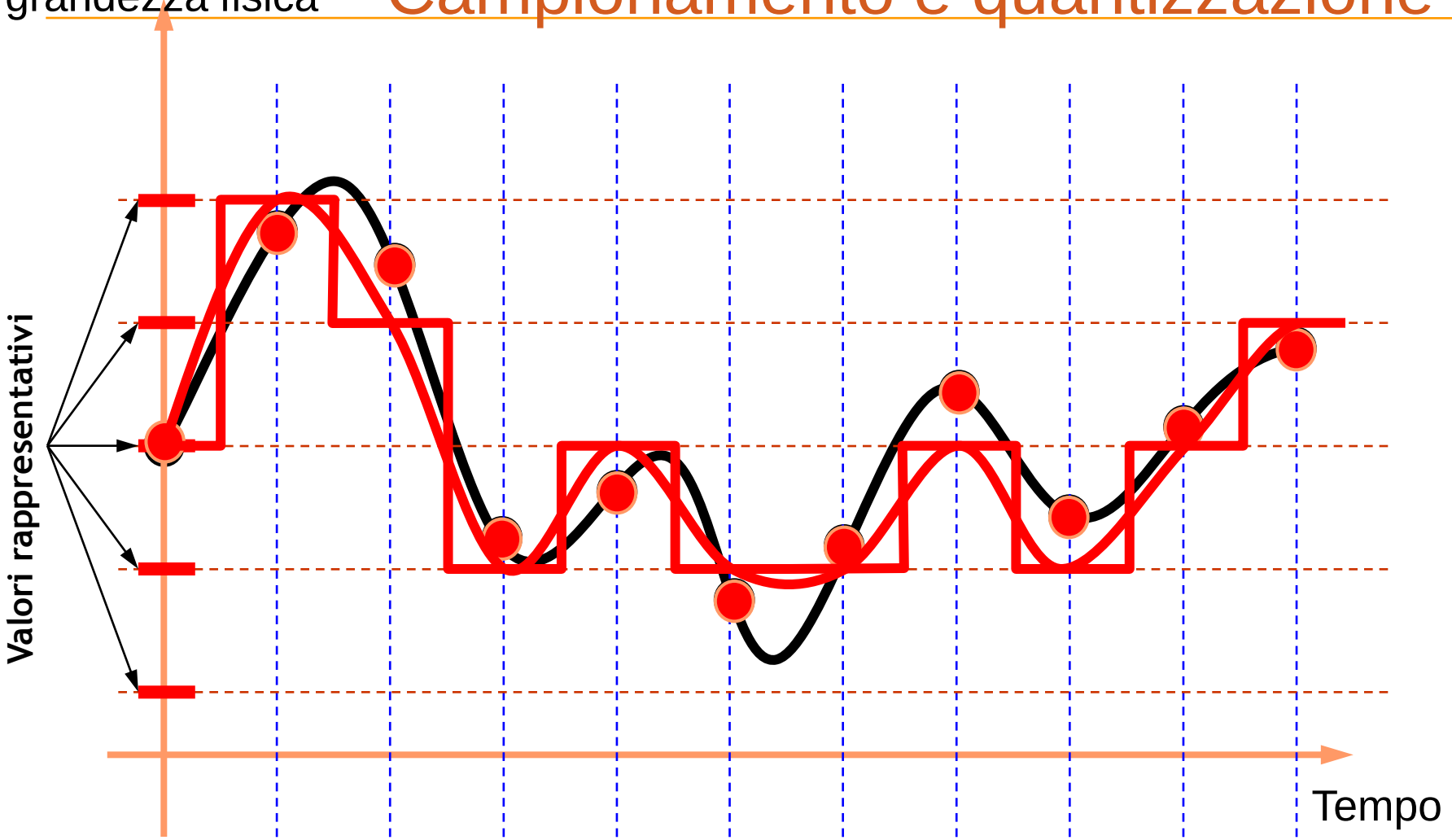
- Il segnale continuo viene campionato ad intervalli di tempo regolari t (t = intervallo di campionamento).
- Il segnale risultante è un insieme finito di punti equidistanti nel tempo. Tuttavia le ampiezze devono essere ancora approssimate ad intervalli discreti, ovvero quantizzate.
- Si noti che campionamento e quantizzazione comportano una perdita di informazione.

Quantizzazione



- La quantizzazione suddivide l'ampiezza in *n* intervalli uguali che vengono poi codificati in binario. Ogni valore di ampiezza del segnale campionato viene approssimato al più vicino valore discreto di ampiezza.
- Più valori (e quindi più bit) si utilizzano per suddividere le ampiezze, più il segnale risultante sarà preciso.

Campionamento e quantizzazione



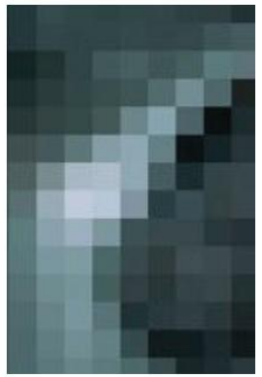
Le immagini digitali

Le immagini digitali non hanno una struttura continua ma sono costituite da un numero finito di componenti prodotte dal campionamento dell'immagine reale. Le componenti assumono un numero finito di tonalità definite dalla quantizzazione dell'immagine campionata.

Classificazione delle immagini

Le immagini si suddividono in **raster e vettoriali.**

Le immagini **raster** sono di tipo fotografico: si rappresenta individualmente ogni singolo punto dell'immagine



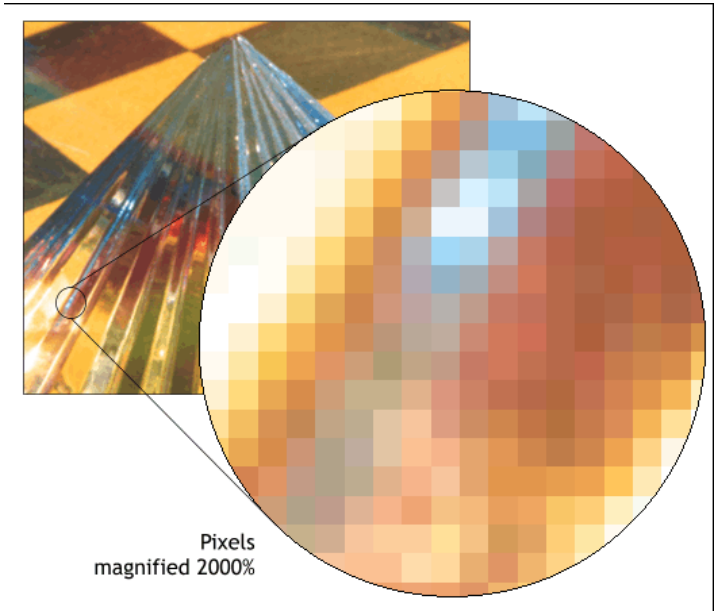
Le immagini **vettoriali** sono essenzialmente disegni: si rappresentano i punti e le curve che formano il disegno

Immagini raster

- Un'immagine raster è definita per punti

- L'immagine è scomposta in un numero elevato di “punti”, tipicamente quadrati o rettangoli quasi-quadrati

- Ciascuno di questi punti è detto **pixel**(da picture element, elemento dell'immagine)



Immagini raster

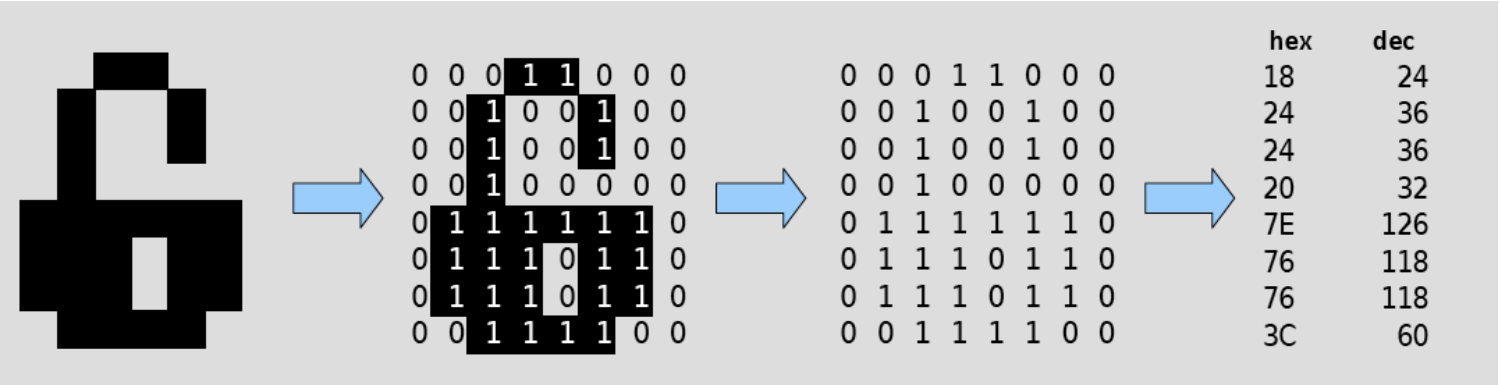
Immagine a 450 x 299 pixel

Immagine a 72 x 47 pixel

Immagine a 32 x 21 pixel

Immagini raster

Esempio di immagine raster (in bianco e nero)



Questa è la tecnica base per convertire un'immagine in un numero

Proprietà dei pixel

- Di ciascuno dei pixel che compongono l'immagine vengono definite le proprietà:
  - colore (sempre)
  - trasparenza (in alcuni casi)

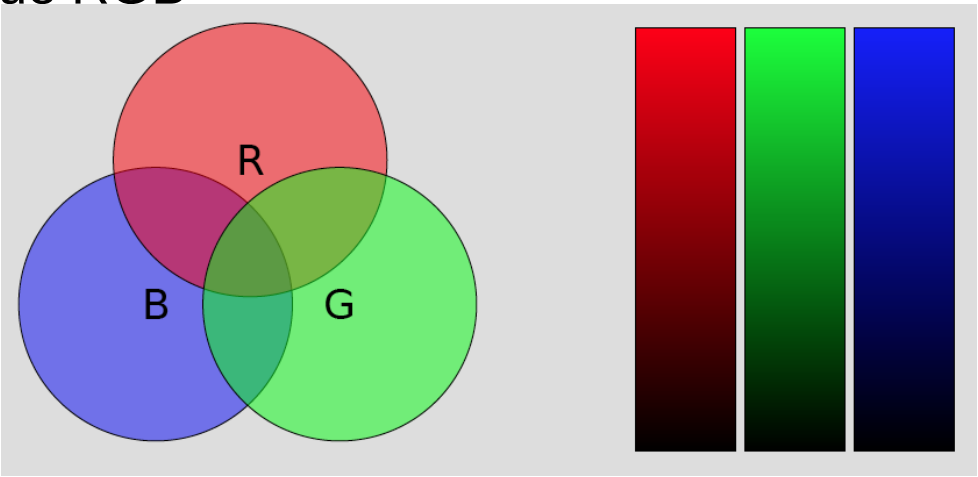
- Applicazioni più specializzate definiscono altre proprietà (per esempio, immagini astronomiche)

Definizione del colore

In generale, il colore è dato dalla combinazione di più componenti colore

- Nel caso più comune, si usano le componenti primarie:
  - quantità di rosso (R, red)
  - quantità di verde (G, green)
  - quantità di blu (B, blue)

- Questo modello è detto dunque RGB



Definizione del colore

- L'intensità di ciascuna componente (rosso, verde e blu) deve essere rappresentata da un numero

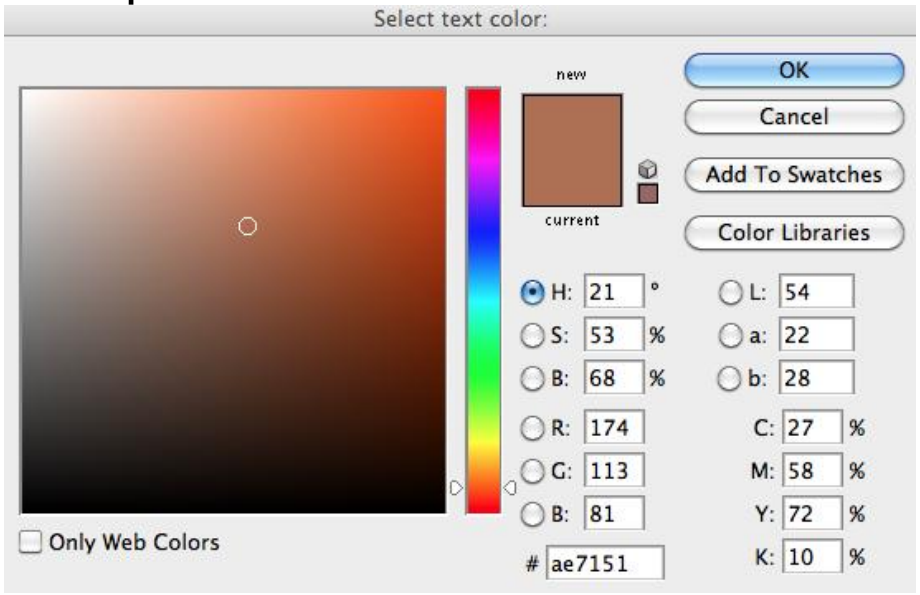
- Si va da 0% (assenza totale del colore) a 100% (colore al massimo della saturazione)

- Si discretizza il valore con un numero intero (potenza di 2, per esempio 0-7 o 0-255)

- Per una bitmap (formato raster più comune): ad ogni pixel si associa l'indice di una colormap (vettore di valori RGB). La profondità del colore può essere di 16 colori (4 bit), 256 colori (8 bit), 2^24 colori (24 bit). Supporta anche immagini binarie (1 bit) o a livelli di grigio (8 bit).

Definizione del colore

- L'assenza di R, G, B produce il nero
- La presenza di R, G, Balla massima intensità produce il bianco
- Un colore è un punto nello spazio dei colori
- Poiché ogni colore ha tre coordinate in questo spazio (R, G, B) la rappresentazione piana è complicata...



Codifica del colore

Immagine a 8 bit per colore

Immagine a 4bit

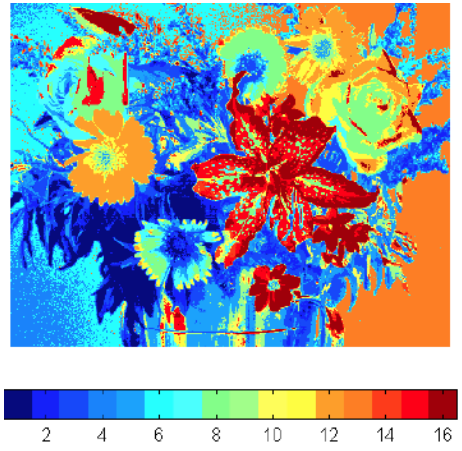


Immagine a 4 bit per colore con diversa colormap

Occupazione di memoria

- L'occupazione di memoria di un'immagine è legata alla sua risoluzione spaziale e di colore

- Tanto maggiori le risoluzioni, tanto migliore la qualità dell'immagine, e tanto maggiore l'occupazione di memoria

- Occupazione in bit = larghezza (in pixel) × altezza (in pixel) ×profondità colore (in bit)

- ES: Un’immagine di 640×480 pixel con 256 colori occupa 640×480×8 = 2457600 bit = 307200 byte = 307.2 Kbyte.

- Formati di codifica bitmap compressa: gif, jpg,pict, tiff, ecc.

Digitalizzazione delle immagini

- Per la codifica digitale delle immagini le operazioni di campionamento e quantizzazione si applicano nello spazio invece che nel tempo.
- Il campionamento consiste nel dividere l'immagine in *pixel*, per ognuno dei quali si dovrà prelevare un campione che si considera rappresentativo del colore di tutto il sottoinsieme.
- La quantizzazione è la codifica del colore associato a ogni pixel: i più recenti formati utilizzano 32 bit (4 byte) per pixel: 8 bit per ognuna delle tre componenti fondamentali (RGB: *red*, *green*, *blue*) e altri 8 per gestire le trasparenze.
- Memoria necessaria per immagini non compresse (bitmap)
  - per un'immagine di 640×480 pixel servono 1'228'800 byte;
  - per un'immagine di 800×600 pixel servono 1'920'000 byte;
  - per un'immagine di 1024×768 pixel servono 3'145'728 byte;
  - per un'immagine di 1280×1024 pixel servono 5'242'880 byte;
  - per un'immagine di 1600×1200 pixel servono 7'680'000 byte;
  - …

Filmati e suoni

Codifica dei filmati

Sono sequenze di immagini compresse (ad esempio si possono registrare solo le variazioni tra un fotogramma e l'altro).

Esistono vari formati (compresi i suoni):

- avi (microsoft)
- mpeg (il piu' usato)
- quicktime mov (apple)

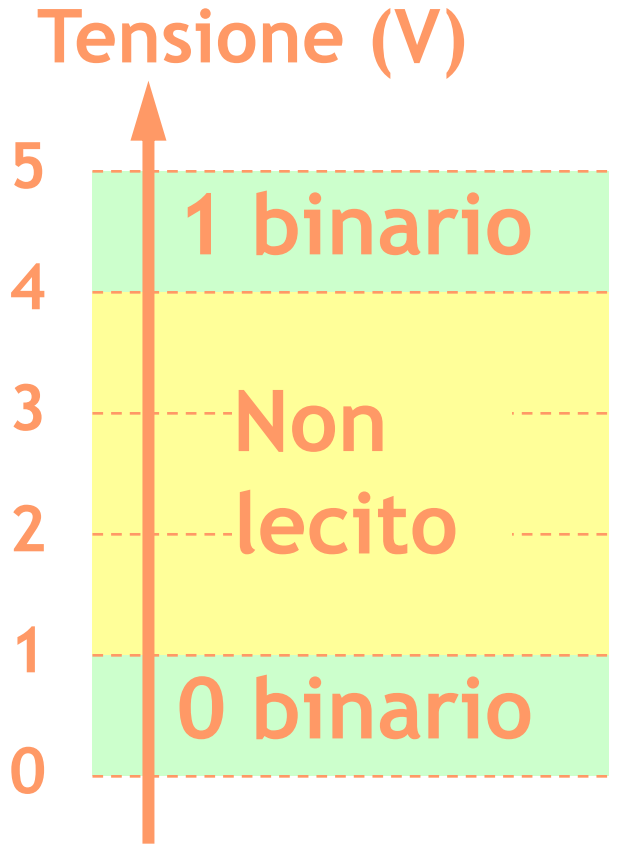
Codifica dei suoni

L'onda sonora viene misurata (campionata) a intervalli regolari. Minore è l'intervallo di campionamento, maggiore è la qualità del suono.

Es: Per i CD musical si ha: 44000 campionamenti al secondo, 16 bit per campione.

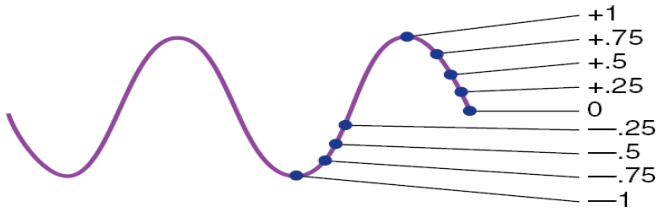
Il successo del digitale

- Rumore: effetto dell'ambiente sul supporto.
- Quanto un supporto è "immune" al rumore?
  - Codifica analogica: ogni configurazione è lecita dal punto di vista informazionale 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

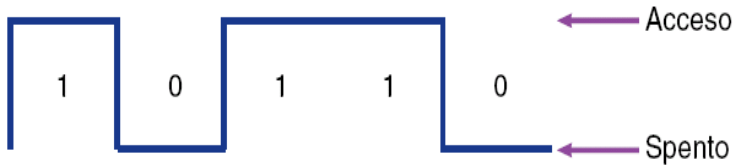


Riassumendo…

- Segnale analogico
  - come un onda che trasporta informazioni, massimi, minimi e tutti i valori intermedi
  - i segnali analogici sono molto sensibili alle interferenze
- Segnale digitale
  - assume solo due stati: acceso/spento, sì/no, vero/falso
  - il segnale digitale è più facile da distinguere, quindi risente meno delle interferenze



Segnali analogici



Segnali digitali

I sistemi di numerazione

- Le informazioni numeriche possono essere elaborate attraverso l'applicazione di operazioni.

- Un sistema di numerazione è una struttura matematica che permette di rappresentare i numeri attraverso dei simboli.

Numerazione non posizionale

- Il significato dei simboli dipende dalla loro posizione
- E' stabilito in base ad una legge additiva dei valori dei singoli simboli (se posti in ordine crescente)
- Esempio:
  - il sistema di numerazione romano

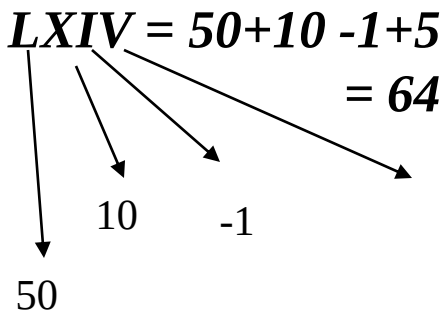
I = 1

V = 5

X = 10

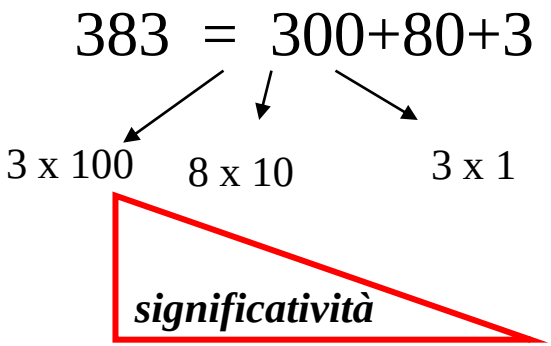
L = 50

...



Numerazione posizionale

- Ai diversi simboli dell'alfabeto (cifre), viene associato un valore crescente in modo lineare da destra verso sinistra
- il significato di un simbolo (il suo valore) dipende ordinatamente dalla sua posizione nella stringa
- Esempio:
  - il sistema di numerazione decimale arabo: 10 simboli (0, 1, 2, ...9)



Sistemi di numerazione non posizionali

- Un simbolo rappresenta un numero.
- Il numero rappresentato da una stringa di simboli si ottiene attraverso regole operazionali applicate ai simboli della stringa
  - Esempio (numeri romani): LXXIV rappresenta 50+10+10-1+5 = 74
- Difficile effettuare operazioni
- Rappresentazione non compatta

Sistemi di numerazione posizionali

- Dato un alfabeto ordinato di *b* simboli distinti (c_1, c_2, …, c_b) che rappresentano rispettivamente i naturali 0, 1, 2, …, *b-1* si rappresenta ogni altro numero *x* maggiore di *b-1* mediante una stringa di simboli dell'alfabeto

- *b* è la base del sistema di numerazione
  - numero di simboli dell'alfabeto richiesti per rappresentare la serie infinita dei numeri

Valore della posizione

- La posizione di un simbolo all'interno di un numero indica il valore che esso esprime, o più precisamente l'esponente che bisogna dare alla base per ottenere il valore corretto.
- Il valore (o la quantità) di 0, 1, 2, 3, 4, 5, 6, 7, 8 e 9 dipende dalla posizione che ciascuno di essi assume all'interno del numero:
- la prima cifra a destra rappresenta le unità (il coefficiente di 10^0), la seconda le decine (10^1) , la terza le centinaia (10^2) , e così via.

Il numero **3.098.323** è una rappresentazione abbreviata di

$$(3 \times 10^6) + (0 \times 10^5) + (9 \times 10^4) + (8 \times 10^3) + (3 \times 10^2) + (2 \times 10^1) + (3 \times 10^0 = 3 \times 1)$$

Il primo 3 (leggendo da destra a sinistra) rappresenta 3 unità; il secondo 3, sta per 300 unità, o 3 centinaia; infine il terzo 3, per 3 milioni di unità.

Il sistema di numerazione decimale

- È il sistema più conosciuto dall'uomo.
- La base b è pari a 10.
- I simboli utilizzati sono 0,1,2,…,9 dal significato ovvio
  - Esempio: la stringa 2349 rappresenta il numero $2*10^3+3*10^2+4*10^1+9*10^0$.
- I numeri decimali sono facilmente intelligibili.

Il sistema di numerazione binario

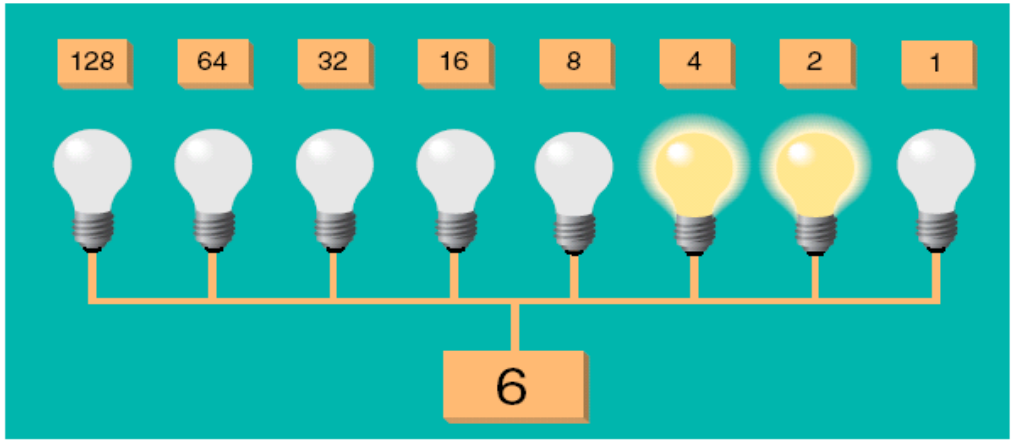
- È il sistema maggiormente utilizzato dai sistemi di elaborazione.
- La base b è pari a 2.
- I simboli utilizzati sono 0 e 1, rappresentanti lo zero e l'uno.
  - Esempio: la stringa binaria 10010 rappresenta il numero $1*2^4+0*2^3+0*2^2+1*2^1+0*2^0$ (=18 in decimale).
- È scarsamente leggibile, specie quando le stringhe sono molto lunghe.

Conversioni

- Da binario a decimale
  - Basta scrivere il numero secondo la notazione posizionale utilizzando già il sistema decimale
- Da decimale a binario
  - Basta dividere ripetutamente il numero decimale per 2, tenere il resto della divisione, dividere il quoziente per 2, tenere il resto della divisone, etc… fino ad arrivare ad avere 0 come quoziente

Da binario a decimale: esempio

- Il numero binario 00000110 corrisponde al valore decimale 6
- $0x2^7+0x2^6+0x2^5+0x2^4+0x2^3+1x2^2+1x2^1+0x2^0 =$
  $= 0+0+0+0+0+4+2+0 = 6$



I bit che danno come somma 6

Da decimale a binario: esempio

(cifra binaria meno significativa)

18 : 2 = 9 resto 0
9 : 2 = 4 resto 1
4 : 2 = 2 resto 0
2 : 2 = 1 resto 0
1 : 2 = 0 resto 1

(cifra binaria più significativa)

10010

137 : 2 = 68 resto 1
68 : 2 = 34 resto 0
34 : 2 = 17 resto 0
17 : 2 = 8 resto 1
8 : 2 = 4 resto 0
4 : 2 = 2 resto 0
2 : 2 = 1 resto 0
1 : 2 = 0 resto 1

10001001

Da decimale a binario: esempio

$565_{dieci} : 2_{dieci}$	$\Rightarrow$	*quoziente*	282_{dieci}	*resto*	1_{dieci}
$282_{dieci} : 2_{dieci}$	$\Rightarrow$	*quoziente*	141_{dieci}	*resto*	0_{dieci}
$141_{dieci} : 2_{dieci}$	$\Rightarrow$	*quoziente*	70_{dieci}	*resto*	1_{dieci}
$70_{dieci} : 2_{dieci}$	$\Rightarrow$	*quoziente*	35_{dieci}	*resto*	0_{dieci}
$35_{dieci} : 2_{dieci}$	$\Rightarrow$	*quoziente*	17_{dieci}	*resto*	1_{dieci}
$17_{dieci} : 2_{dieci}$	$\Rightarrow$	*quoziente*	8_{dieci}	*resto*	1_{dieci}
$8_{dieci} : 2_{dieci}$	$\Rightarrow$	*quoziente*	4_{dieci}	*resto*	0_{dieci}
$4_{dieci} : 2_{dieci}$	$\Rightarrow$	*quoziente*	2_{dieci}	*resto*	0_{dieci}
$2_{dieci} : 2_{dieci}$	$\Rightarrow$	*quoziente*	1_{dieci}	*resto*	0_{dieci}
$1_{dieci} : 2_{dieci}$	$\Rightarrow$	*quoziente*	0_{dieci}	*resto*	1_{dieci}

$$1000110101_{due} = 565_{dieci}$$

Da decimale a binario: esempio

Perché funziona?

Applichiamolo alla base 10:

565:10	⇒	*quoziente*	56	*resto* 5
56:10	⇒	*quoziente*	5	*resto* 6
5:10	⇒	*quoziente*	0	*resto* 5

La prima divisione per 10 consente di “isolare” le unità, la seconda le decine, la terza le centinaia, ecc.

$565 = (5 \times 10+6)\times 10+5 = 5 \times 10^2 + 6 \times 10 + 5$

La stessa cosa se si divide per 2 invece che per 10. In questo caso si isolano le potenze di 2 e non le unità o le decine o le centinaia ecc.

Somme di numeri binari

Le somme dei numeri binari si basano sulle seguenti espressioni:
0+0=0
0+1=1
1+0=1
1+1=10 (0 col riporto di 1)

	111
25+	00011001+
60=	00111100=
85	01010101

	11
44+	00101100+
78=	01001110=
112	01111010

Moltiplicazione

Le moltiplicazioni dei numeri binari si basano sulle seguenti espressioni:
0x0=0
0x1=0
1x0=0
1x1=1

```
 5x          0101x
 2=          0010=
 10          0000+
             0101
            01010
```

Nel calcolatore viene eseguita come serie di somme successive del moltiplicando shiftato ogni volta di un numero di posizioni pari alla posizione degli 1 del moltiplicatore (v. esempio)