

INFORMATICA

Massimiliano Salfi

massimiliano.salfi@unict.it



CALENDARIO LEZIONI

Quando

Come da calendario lezioni

Dove

aula 6, comparto 10



RIFERIMENTI

Sito internet

<http://www.dmi.unict.it/salfi/home.htm>

Ricevimento

Solo per appuntamento

Esami

Esame orale



PROGRAMMA

- Introduzione all'informatica
- La codifica e la rappresentazione delle informazioni
- Architettura dei calcolatori
- Il sistema operativo e gli applicativi software
- Le Reti di calcolatori ed internet
- Iper testi e codice HTML
- La sicurezza in rete ed i malware
- Elementi di informatica medica



MATERIALE DIDATTICO

Slides del docente

Luca Mari, Giacomo Bonanno, Donatella Sciuto
Informatica e cultura dell'informazione
McGraw-Hill - Seconda edizione





INTRODUZIONE ALL'INFORMATICA

CHE COS'È L'INFORMATICA?

Informatica

=

Informazione + Automatica

Si riferisce ai processi e alle tecnologie che rendono possibile l'immagazzinamento e l'elaborazione delle informazioni.



GLI ANTENATI DEL COMPUTER

- Macchina analitica di Babbage (1830).
- Macchina di Turing (1936).
- Macchina di Von Neumann (anni '40).



GLI ANTENATI DEL COMPUTER

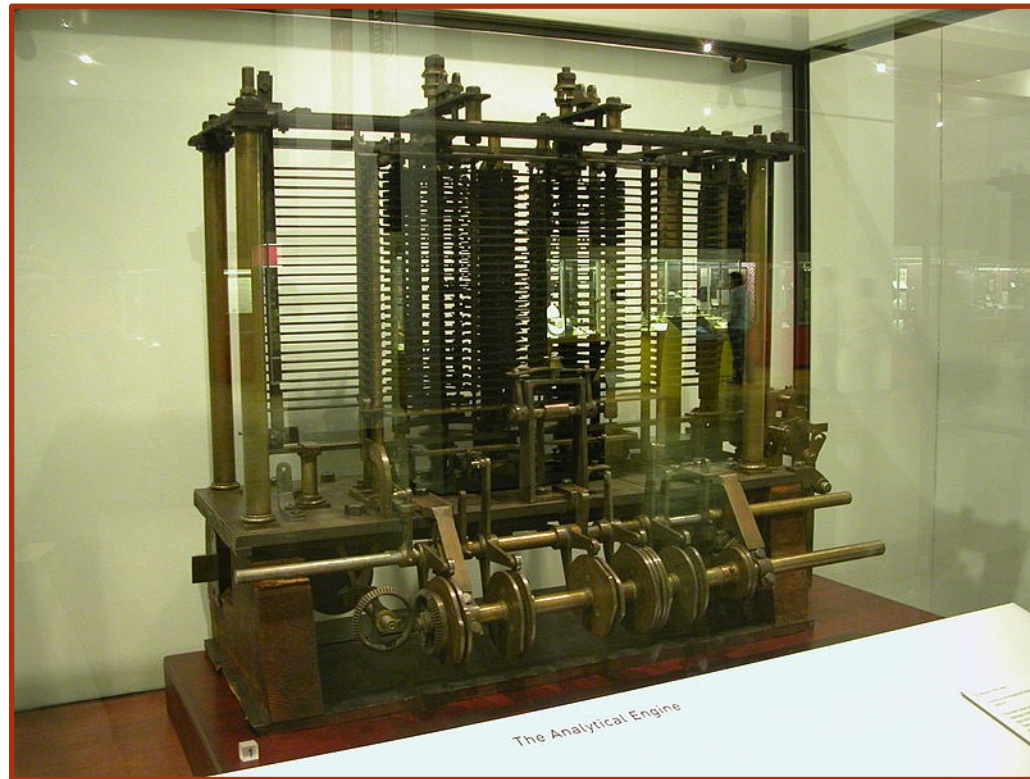
Il progetto della Macchina analitica di Babbage fu sviluppato dal matematico, filosofo e scienziato inglese Charles Babbage (1791–1871).

Pur essendo realizzata solo in parte per motivi politici e finanziari, rappresenta un importante passo nella storia dell'informatica.

Il moderno PC, infatti, presenta parecchie analogie con il progetto sviluppato da Babbage.



GLI ANTENATI DEL COMPUTER

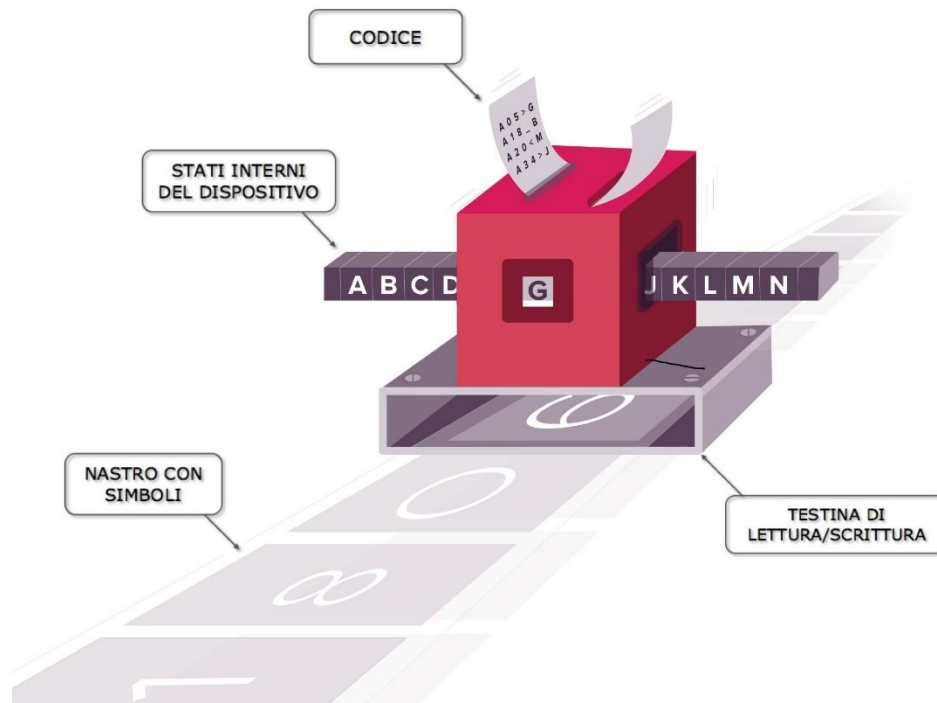


Modello di una parte dell'Analytical Engine di Babbage in mostra al Museo della scienza di Londra



GLI ANTENATI DEL COMPUTER

La macchina di Turing è una macchina ideale che manipola i dati contenuti su un nastro (memoria) di lunghezza infinita, con cui interagisce attraverso una testina di lettura/scrittura. Il dispositivo è caratterizzato da n (numero finito) stati interni.



GLI ANTENATI DEL COMPUTER

Ogni volta che la testina legge un dato dal nastro, la macchina di Turing può:

- cambiare stato interno (tra quelli ammessi);
- leggere/scrivere un nuovo dato da/sul nastro (memoria);
- spostare la testina, di una cella di memoria, avanti o indietro.

La macchina di Turing può essere programmata attraverso un codice che descrive le azioni che essa deve compiere.

In altre parole, è un modello astratto che definisce una macchina in grado di eseguire algoritmi e dotata di un nastro infinito (la memoria) su cui possono leggere e/o scrivere dei simboli (i dati).



GLI ANTENATI DEL COMPUTER

La macchina di Von Neumann è il modello architettonico secondo il quale è organizzata la maggior parte dei moderni elaboratori.

È dotata delle seguenti parti:

- processore;
- memorie;
- periferiche di Input;
- periferiche di Output;
- sistema di interconnessioni (bus).

Vedremo più avanti questo modello in dettaglio.



LA MACCHINA COMPUTER

In generale, un computer:

- elabora i dati, eseguendo operazioni logiche e aritmetiche;
- salva istruzioni e dati in memoria;
- interagisce con l'ambiente circostante prelevando informazioni da elaborare (input) o fornendo i risultati di tale elaborazione (output).

Un'altra caratteristica importante di un computer è la modularità e la scalabilità.

Modulare: utilizzo parti standard, pertanto in caso di guasti posso sostituirli con altri di pari funzionalità.

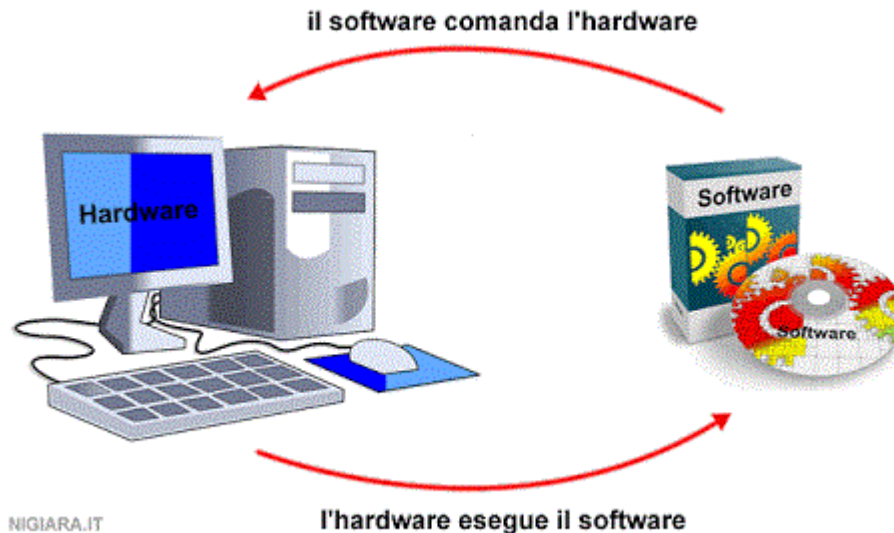
Scalabile: sostituisco la parti (i moduli) con altre parti compatibili ma aventi caratteristiche migliori.



HARDWARE VS SOFTWARE

L'*hardware* denota la struttura fisica del computer, costituita di norma da componenti elettronici che svolgono specifiche funzioni nel trattamento dell'informazione.

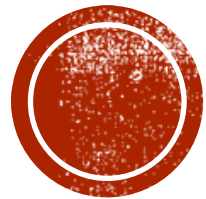
Il *software* indica l'insieme delle istruzioni che consentono alle varie parti hardware di svolgere i compiti per i quali sono stati progettati.



POSSIBILI SCENARI APPLICATIVI

- economico e commerciale;
- industriale;
- didattico e della formazione professionale;
- arte e spettacolo;
- ingegneristico;
- matematico e delle scienze;
- lavorativo e del tempo libero;
- medico;
- etc.





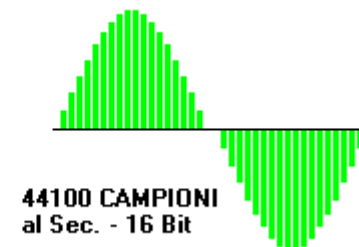
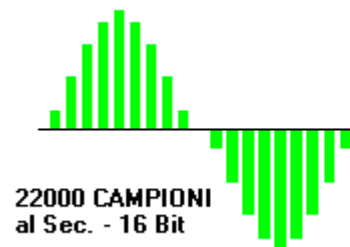
LA CODIFICA E LA RAPPRESENTAZIONE DELLE INFORMAZIONI

I SEGNALI PER COMUNICARE

Gli essere viventi ed il computer utilizzano modi tra loro differenti per comunicare:

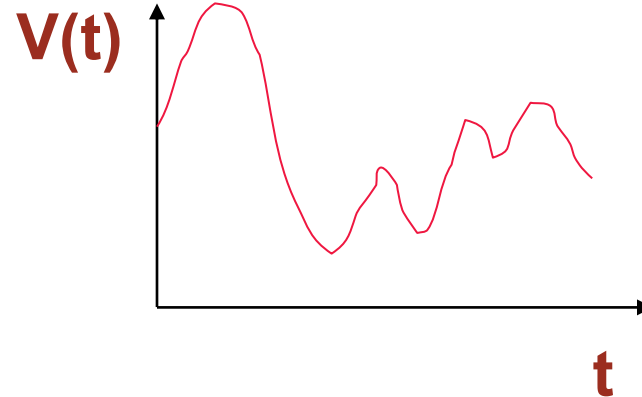
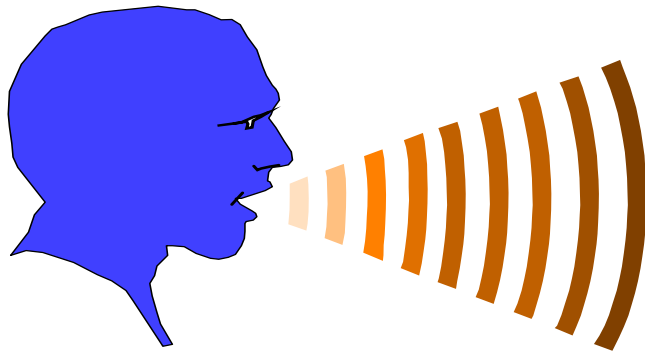
- segnali *analogici*;
- segnali *digitali*.

Ad un segnale analogico faremo corrispondere una grandezza continua, mentre ad un segnale digitale, una grandezza discreta.



INFORMAZIONE ANALOGICA

La voce umana, i suoni presenti in natura o emessi da strumenti musicali, sono sistemi di comunicazione di tipo *analogico*, nei quali le grandezze fisiche che entrano in gioco sono funzioni continue del tempo.



INFORMAZIONE DIGITALE

La codifica dei segnali nei computer avviene in modo digitale, in quanto le grandezze fisiche sono rappresentate da coppie di stati discreti:



(0, 1) oppure (off, on) oppure (false, true)

Nei circuiti digitali il simbolo 0 (oppure *off*, oppure *false*) è associato ad un segnale a basso voltaggio (interruttore spento); di contro il simbolo 1 (oppure *on*, oppure *true*) è associato ad un segnale ad alto voltaggio (interruttore acceso).



DIGITALIZZAZIONE DEI SEGNALI

Qualsiasi segnale «reale», dunque, per poter essere elaborato attraverso un computer, deve essere convertito in un segnale digitale (e quindi in una sequenza finita di valori numerici).

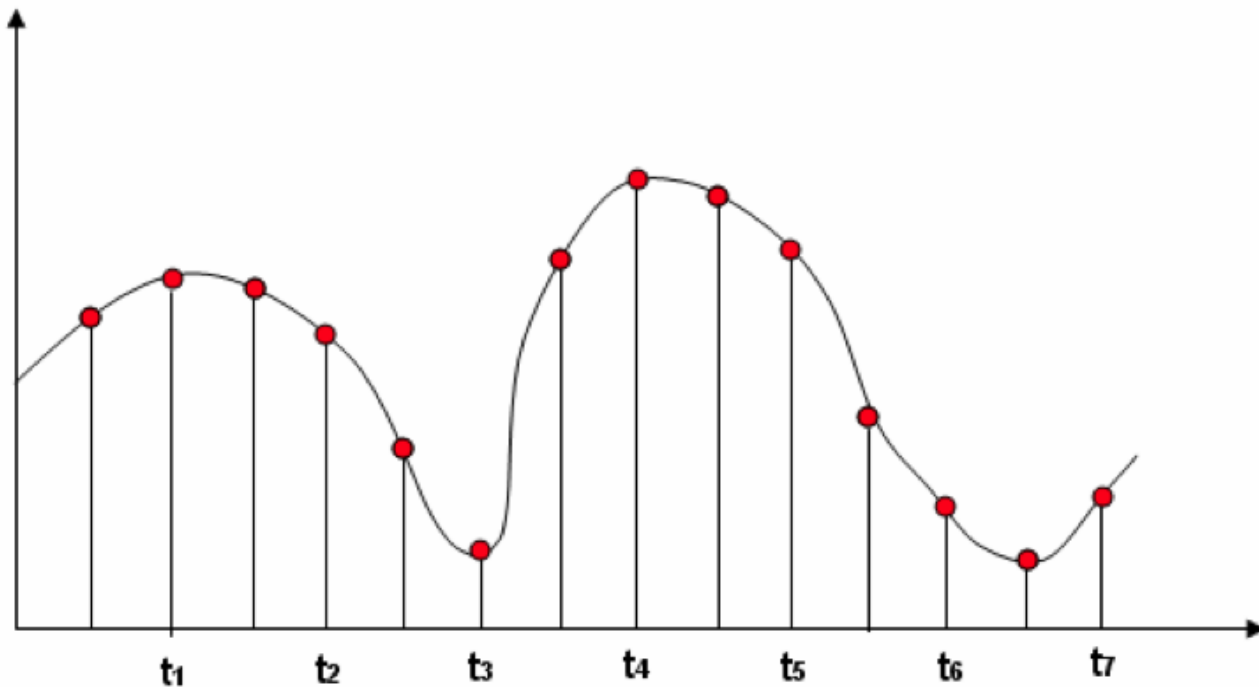
Nel farlo, devo attivare un procedimento noto come «campionamento», che mi permette di prelevare un certo numero di valori nell'unità di tempo (frequenza di campionamento).

Tale conversione, per quanto accurata, comporta sempre un certo grado di **approssimazione**, con conseguente perdita di informazione.



DIGITALIZZAZIONE DEI SEGNALE

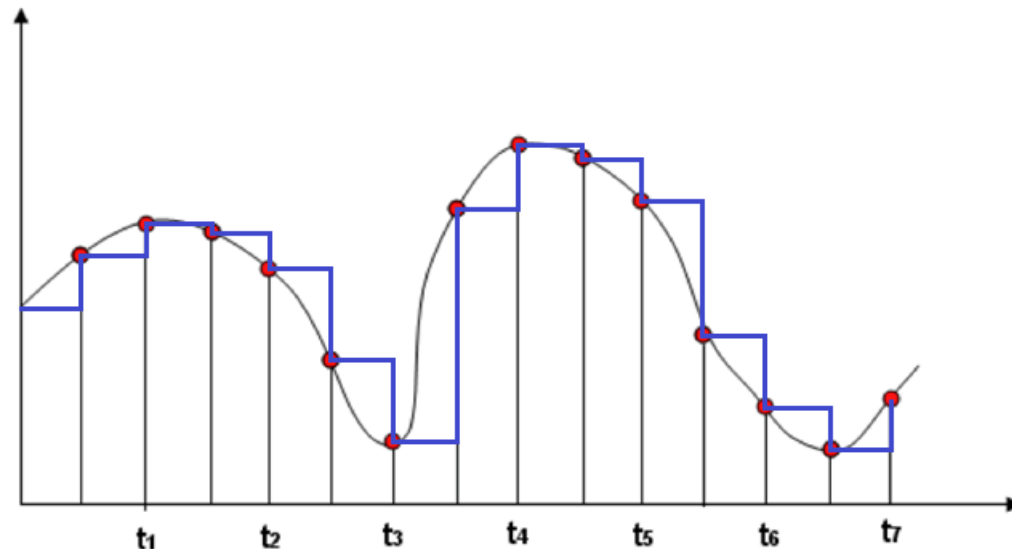
Consideriamo ad esempio una grandezza continua e supponiamo di prelevare 7 campioni. Graficamente avremo:



DIGITALIZZAZIONE DEI SEGNALI

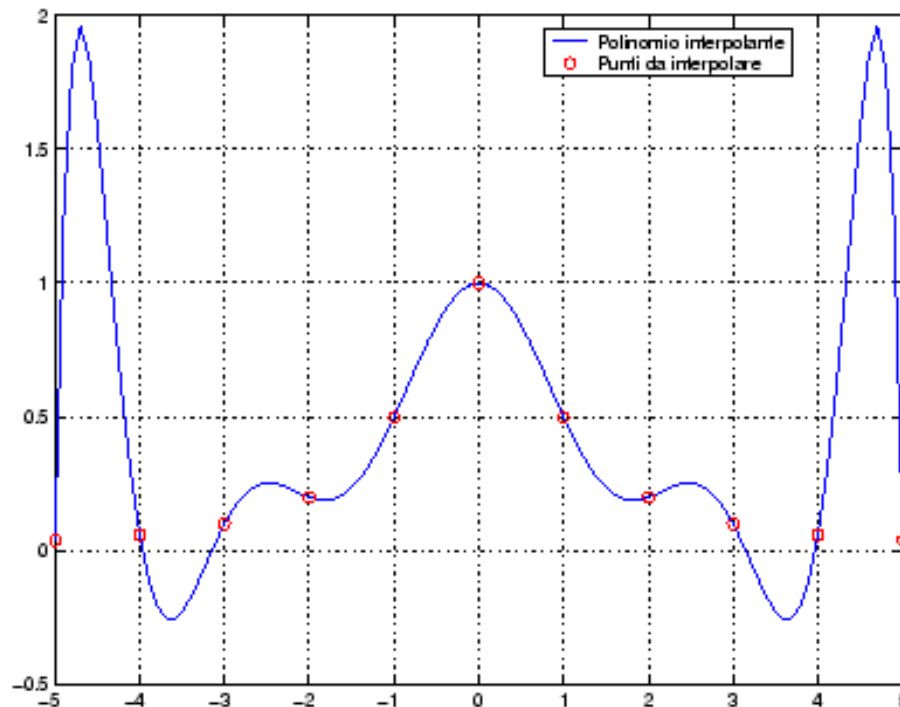
Se dal segnale digitale voglio ottenere nuovamente quello analogico, il mio sistema dovrà ricostruire il segnale mancante tra i vari campioni. Per la ricostruzione, si possono utilizzare varie tecniche di interpolazione.

Ad esempio, quella del «sample and hold», che mantiene costante il valore fino al punto successivo:



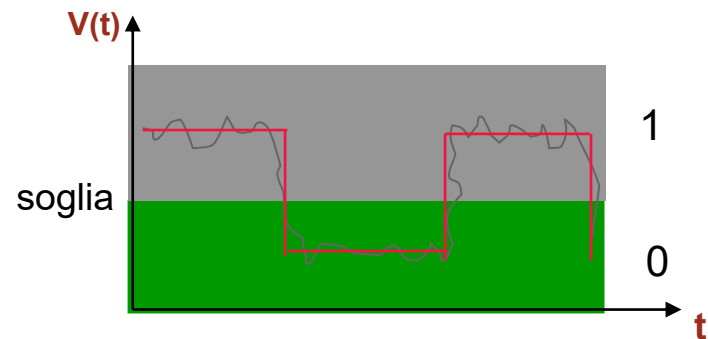
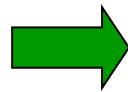
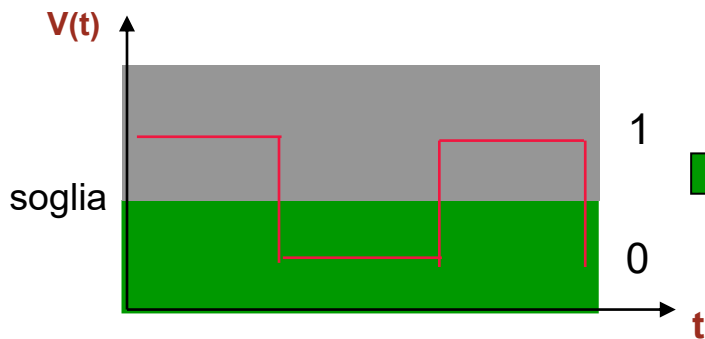
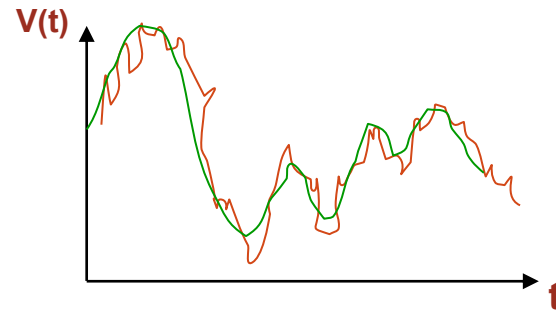
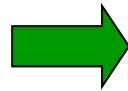
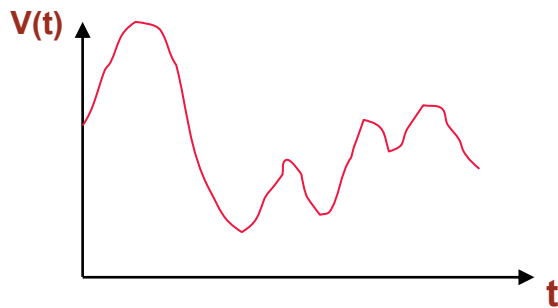
DIGITALIZZAZIONE DEI SEGNALI

O, per evitare un segnale troppo «spigoloso», attraverso interpolazione polinomiale (ad esempio interpolazione di Langrange):



PRECISIONE DEI SEGNALI

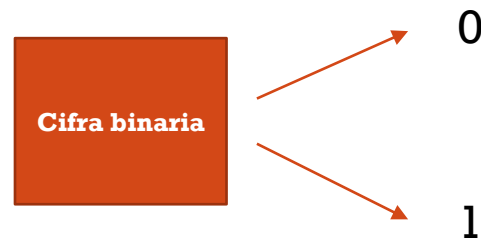
I segnali digitali presentano il vantaggio di essere meno affetti da disturbi di trasmissione (errori) rispetto a quelli analogici.



RAPPRESENTAZIONE DELLE INFORMAZIONI

In un calcolatore, tutte le informazioni sono rappresentate in forma binaria (o digitale), come detto, utilizzando due soli simboli. Di solito si adoperano i simboli 0 ed 1.

Con una cifra binaria si possono, quindi, rappresentare soltanto due informazioni.



RAPPRESENTAZIONE DELLE INFORMAZIONI

È semplice intuire, in ragione di quanto detto in precedenza, che tale scelta è legata alla necessità di rappresentare, ad esempio:

- due possibili stati di polarizzazione di una sostanza magnetizzabile;
- il passaggio (o non passaggio) di corrente attraverso un conduttore;
- Il passaggio (o non passaggio) della luce attraverso una fibra ottica.

Questo concetto verrà ripreso in seguito, nella macchina di Von Neumann, quando parleremo di memorie.



IL BIT

Il bit è *l'unità di misura dell'informazione* ed è definita come la quantità minima di informazione che serve a rappresentare due differenti stati: 0 oppure 1. Il nome proviene da **Binary Digit**.

Si utilizzano i seguenti multipli :

- Kilo (Kb), pari a 2^{10} ~ un migliaio (1024 bit);
- Mega (Mb) 2^{20} ~ un milione (1024 x 1024 bit);
- Giga (Gb) 2^{30} ~ un miliardo (1Mb x 1024 bit);
- Tera (Tb) 2^{40} ~ mille miliardi (1Gb x 1024 bit).



CODIFICA BINARIA

Per poter rappresentare un numero maggiore di dati, o informazioni, è necessario utilizzare sequenze di bit. Utilizzando due bit si possono rappresentare quattro informazioni diverse, vale a dire:

00 01 10 11

In generale, con n bit si possono avere 2^n informazioni diverse.

Il processo che fa corrispondere ad una informazione una configurazione di bit, prende il nome di *codifica dell'informazione*.



SEQUENZE DI BIT

Numero di bit nella sequenza	Informazioni rappresentabili
2	4
3	8
4	16
5	32
6	64
7	128
8	256



SET DI CARATTERI

Nella comunicazione scritta, di norma, è necessario disporre dei seguenti caratteri:

- 52 lettere alfabetiche maiuscole e minuscole;
- 10 cifre (0, 1, 2, ..., 9);
- Segni di punteggiatura (, . ; : ! " ' ? ^ \ ...);
- Segni matematici (+, -, ×, :, ±, {, [, >, ...);
- Caratteri nazionali (à, è, ì, ò, ù, ç, ñ, ö, ...);
- Altri segni grafici (©, ←, ↑, @, €, ...).



CODICE

Si pone quindi la necessità di codificare almeno 220 caratteri utilizzando sequenze di bit.

La sequenza di bit necessaria a rappresentare 220 simboli deve essere composta da 8 bit (con 8 bit, infatti, dispongo di $2^8=256$ stringhe differenti).

La corrispondenza tra sequenze (o stringhe) di bit e simboli prende il nome di *codice*.



IL BYTE

Un gruppo di 8 bit viene denominato Byte. Un byte:

- corrisponde ad un carattere;
- rappresenta l'unità di misura della capacità di memoria.

Si utilizzano i seguenti multipli del Byte:

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



CODIFICA SET DI CARATTERI

Codifiche standard:

- **ASCII**, 8 bit per carattere, rappresenta 256 caratteri.
- **UNICODE**, 16 bit per carattere (estende il codice ASCII con i caratteri etnici).

Codifiche proprietarie:

- **MSWindows**, 16 bit per carattere (molto simile ad UNICODE).



CODICE ASCII

Acronimo di:

American
Standard
Code for
Information
Interchange

Binary	Oct	Dec	Hex	Glyph
010 0000	040	32	20	
010 0001	041	33	21	!
010 0010	042	34	22	"
010 0011	043	35	23	#
010 0100	044	36	24	\$
010 0101	045	37	25	%
010 0110	046	38	26	&
010 0111	047	39	27	'
010 1000	050	40	28	(
010 1001	051	41	29)
010 1010	052	42	2A	*
010 1011	053	43	2B	+
010 1100	054	44	2C	,
010 1101	055	45	2D	-
010 1110	056	46	2E	.
010 1111	057	47	2F	/
011 0000	060	48	30	0
011 0001	061	49	31	1
011 0010	062	50	32	2
011 0011	063	51	33	3
011 0100	064	52	34	4
011 0101	065	53	35	5
011 0110	066	54	36	6
011 0111	067	55	37	7
011 1000	070	56	38	8
011 1001	071	57	39	9
011 1010	072	58	3A	:
011 1011	073	59	3B	;
011 1100	074	60	3C	<
011 1101	075	61	3D	=
011 1110	076	62	3E	>
011 1111	077	63	3F	?

Binary	Oct	Dec	Hex	Glyph
100 0000	100	64	40	@
100 0001	101	65	41	A
100 0010	102	66	42	B
100 0011	103	67	43	C
100 0100	104	68	44	D
100 0101	105	69	45	E
100 0110	106	70	46	F
100 0111	107	71	47	G
100 1000	110	72	48	H
100 1001	111	73	49	I
100 1010	112	74	4A	J
100 1011	113	75	4B	K
100 1100	114	76	4C	L
100 1101	115	77	4D	M
100 1110	116	78	4E	N
100 1111	117	79	4F	O
101 0000	120	80	50	P
101 0001	121	81	51	Q
101 0010	122	82	52	R
101 0011	123	83	53	S
101 0100	124	84	54	T
101 0101	125	85	55	U
101 0110	126	86	56	V
101 0111	127	87	57	W
101 1000	130	88	58	X
101 1001	131	89	59	Y
101 1010	132	90	5A	Z
101 1011	133	91	5B	[
101 1100	134	92	5C	\
101 1101	135	93	5D]
101 1110	136	94	5E	^
101 1111	137	95	5F	_

Binary	Oct	Dec	Hex	Glyph
110 0000	140	96	60	`
110 0001	141	97	61	a
110 0010	142	98	62	b
110 0011	143	99	63	c
110 0100	144	100	64	d
110 0101	145	101	65	e
110 0110	146	102	66	f
110 0111	147	103	67	g
110 1000	150	104	68	h
110 1001	151	105	69	i
110 1010	152	106	6A	j
110 1011	153	107	6B	k
110 1100	154	108	6C	l
110 1101	155	109	6D	m
110 1110	156	110	6E	n
110 1111	157	111	6F	o
111 0000	160	112	70	p
111 0001	161	113	71	q
111 0010	162	114	72	r
111 0011	163	115	73	s
111 0100	164	116	74	t
111 0101	165	117	75	u
111 0110	166	118	76	v
111 0111	167	119	77	w
111 1000	170	120	78	x
111 1001	171	121	79	y
111 1010	172	122	7A	z
111 1011	173	123	7B	{
111 1100	174	124	7C	
111 1101	175	125	7D	}
111 1110	176	126	7E	~



CODICE ASCII: ALTRO ESEMPIO

La parola *Computer* in codice ASCII diventa, dunque:

C	=	01000011
o	=	01101111
m	=	01101101
p	=	01110000
u	=	01110101
t	=	01110100
e	=	01100101
r	=	01110010

01000011 01101111 01101101 11100000 01110101 01110100 01100101 01110010



CODICE ASCII

Data una sequenza di bit, dividendola in gruppi di byte è possibile risalire ai caratteri originali:

01000001 01101101 01101111 01110010 01100101 00100001

01000001 01101101 01101111 01110010 01100101 00100001

A m o r e !



CODICE ASCII

Con il codice ASCII è possibile rappresentare i numeri come sequenza di caratteri. Ad esempio il numero 234 sarà rappresentato come:

00110010 00110011 00110100

2

3

4

Con questo tipo di rappresentazione non è possibile effettuare operazioni aritmetiche.



IL SISTEMA DECIMALE

Il sistema di numerazione decimale è un sistema *posizionale*. Questo significa che ogni cifra del numero, assume un valore differente in funzione della posizione. Consideriamo il numero seguente in *notazione compatta*:

221

corrisponde a:

$$2 \times 100 + 2 \times 10 + 1 \times 1$$

Che, in *notazione esplicita*, si scriverà:

$$2 \times 10^2 + 2 \times 10^1 + 1 \times 10^0$$



IL SISTEMA DECIMALE

Ogni numero si esprime come la somma dei prodotti di ciascuna cifra per la base, elevata all'esponente dato dalla posizione della cifra:

$$221 = 2 \times 10^2 + 2 \times 10^1 + 1 \times 10^0$$

È quindi basato sulle potenze del 10 ed utilizza, come detto, una notazione posizionale basata sui 10 simboli:

$$\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$



LA NOTAZIONE POSIZIONALE

La notazione posizionale può essere usata con qualunque base, creando così differenti sistemi di numerazione. Per ogni base di numerazione si utilizza un numero di cifre uguale alla base.

In informatica si utilizzano prevalentemente i sistemi di numerazione:

- binaria,
- ottale,
- esadecimale.

Il sistema di numerazione romano non è posizionale:

XIII vs. CXII



IL SISTEMA BINARIO

Il sistema di numerazione binario utilizza una notazione posizionale basata sulle 2 cifre:

{0, 1}

e sulle potenze di 2. Il numero binario 1001 può essere rappresentato esplicitamente come:

$$1001_2 = 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 9_{10}$$



IL SISTEMA OTTALE

Il sistema di numerazione ottale utilizza una notazione posizionale basata sulle 8 cifre:

$$\{0, 1, 2, 3, 4, 5, 6, 7\}$$

e sulle potenze di 8. Il numero ottale 534 può essere rappresentato, esplicitamente, come:

$$534_8 = 5 \times 8^2 + 3 \times 8^1 + 4 \times 8^0 = 348_{10}$$



IL SISTEMA ESADECIMALE

La numerazione esadecimale utilizza una notazione posizionale basata sulle 16 cifre

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

e sulle potenze di 16. Si noti che A=10, B=11, C=12, D=13, E=14 ed F=15. Il numero esadecimale $B7FC_{16}$ può essere rappresentato esplicitamente come:

$$(11) \times 16^3 + 7 \times 16^2 + (15) \times 16^1 + (12) \times 16^0 = 47100_{10}$$



CONVERSIONE DA BASE N A BASE 10

Per convertire un numero da una qualunque base alla base 10 è sufficiente rappresentarlo esplicitamente:

$$1101_2 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 13_{10}$$

$$710_8 = 7 \times 8^2 + 1 \times 8^1 + 0 \times 8^0 = 456_{10}$$

$$A51_{16} = (10) \times 16^2 + 5 \times 16^1 + 1 \times 16^0 = 2641_{10}$$



CONVERSIONE DA BASE 10 A BASE N

Per convertire un numero decimale ad una base n qualsiasi, occorre trovare tutti i resti delle successive divisioni del numero per la base n .

Proviamo, ad esempio, a trovare il valore binario del numero 210. Come mostrato nella slide successiva, basterà dividere 210 per la base 2, ripetutamente, fino quando non si ottiene zero come risultato e leggere i resti dall'ultimo verso il primo.



CONVERSIONE DA BASE 10 A BASE 2

210	2	resto	0
105	2		1
52	2		0
26	2		0
13	2		1
6	2		0
3	2		1
1	2		1
0			

Leggendo la sequenza dei resti dal basso verso l'alto, si ottiene il numero:

11010010₂



VERIFICA DI CORRETTEZZA

Per una verifica di correttezza basta riconvertire il risultato alla base 10:

$$\begin{aligned} 11010010_2 &= 1 \times 2^7 + 1 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + \\ &+ 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 210_{10} \end{aligned}$$



COSTRUZIONE DEI NUMERI BINARI

Per poter costruire la successione dei numeri binari, si può seguire il seguente schema:

0	0	0	0	=	0
0	0	0	1	=	1
0	0	1	0	=	2
0	0	1	1	=	3
0	1	0	0	=	4
0	1	0	1	=	5
0	1	1	0	=	6
0	1	1	1	=	7



RAPPRESENTAZIONE DEI NUMERI

All'interno dei computer, a causa di vincoli tecnologici, per rappresentare qualsiasi tipo di numero, si utilizza sempre un numero fisso di cifre binarie.

I valori più utilizzati sono:

- 16 bit (2 byte)
- 32 bit (4 byte)

In alcuni casi si può arrivare anche a 64 bit (8 byte) o più a seconda del tipo di processore (vedremo in dettaglio tutto questo, quando parleremo della macchina di Von Neumann).



RAPPRESENTAZIONE DEI NUMERI

Tutti i numeri vengono distinti in tre categorie:

- Interi senza segno (interi positivi).
- Interi con segno (interi positivi e negativi).
- Reali (numeri positivi e negativi con virgola).

Ogni categoria viene rappresentata in modo differente.



RAPPRESENTAZIONE CON NUMERO FISSO DI CIFRE

Per comprendere il meccanismo alla base della rappresentazione con un numero fisso di cifre partiamo da un interrogativo:

Qual è il numero più grande rappresentabile con 4 cifre?

- In base 10, avremo 9999
- In base 2 avremo $1111 = 15_{10}$
- In base 8 avremo $7777 = 4095_{10}$
- In base 16 avremo $FFFF = 65535_{10}$



RAPPRESENTAZIONE CON NUMERO FISSO DI CIFRE

In generale, con n cifre vale la relazione seguente:

$$b^n - 1$$

essendo b la base del sistema di numerazione. Infatti:

- In base 10, avremo $9999 = 10^4 - 1$
- In base 2 avremo $1111 = 2^4 - 1$
 15_{10}
- In base 8 avremo $7777 = 8^4 - 1$
 4095_{10}
- n base 16 avremo $FFFF = 16^4 - 1$
 65535_{10}



RAPPRESENTAZIONE CON NUMERO FISSO DI CIFRE

Vale quindi la seguente regola:

Nella base di numerazione b , disponendo di n cifre, si possono rappresentare soltanto i numeri da

$$0 \text{ a } b^n - 1$$



RAPPRESENTAZIONE DEI NUMERI INTERI SENZA SEGNO

In un elaboratore, tenuto conto di quanto detto, considerato che nelle rappresentazioni si adotta il sistema binario e che le parole sono lunghe 16 o 32 bit, si ha:

- Nella rappresentazione a 16 bit i possibili valori saranno compresi tra 0 e 65.535;
- Nella rappresentazione a 32 bit i possibili valori saranno compresi tra 0 e 4.294.967.295



RAPPRESENTAZIONE DEI NUMERI INTERI CON SEGNO

Per rappresentare i numeri con il loro segno (interi positivi e negativi) esistono due possibili modi:

- Rappresentazione in modulo e segno;
- Metodo del complemento a due.

Nel primo metodo, dati n bit, il bit a sinistra si riserva al segno, lasciando gli altri $n-1$ alla rappresentazione del numero. Di solito si considera 0 per il segno più (+) e 1 per il meno (-):

$$0000\ 0101_2 = +5_{10}$$

$$1000\ 0101_2 = -5_{10}$$



RAPPRESENTAZIONE DEI NUMERI INTERI CON SEGNO

Questo metodo, anche se molto semplice, presenta l'inconveniente che esistono due zeri:

$$1\ 0\ 0\ 0\ 0\ 0\ 1\ 1 = -3$$

$$1\ 0\ 0\ 0\ 0\ 0\ 1\ 0 = -2$$

$$1\ 0\ 0\ 0\ 0\ 0\ 0\ 1 = -1$$

$$1\ 0\ 0\ 0\ 0\ 0\ 0\ 0 = -0$$

$$0\ 0\ 0\ 0\ 0\ 0\ 0\ 0 = +0$$

$$0\ 0\ 0\ 0\ 0\ 0\ 0\ 1 = +1$$

$$0\ 0\ 0\ 0\ 0\ 0\ 1\ 0 = +2$$

$$0\ 0\ 0\ 0\ 0\ 0\ 1\ 1 = +3$$



RAPPRESENTAZIONE DEI NUMERI INTERI CON SEGNO

Utilizzando n bit e riservandone uno al segno, l'applicazione della formula precedente porterà alla rappresentazione del seguente intervallo:

$$\text{da } -(2^{n-1} - 1) \text{ a } 2^{n-1} - 1$$

dove, come detto, n vale o 16, permettendo quindi la rappresentazione dei numeri da -32.767 a +32.767, o 32 con rappresentazione dei numeri da -2.147.483.647 a +2.147.483.647



RAPPRESENTAZIONE DEI NUMERI INTERI CON SEGNO

Il secondo metodo è quello del complemento a 2:

Dato un numero composto da n bit, la rappresentazione in complemento a due si ottiene invertendo gli 1 in 0 e gli 0 in 1 (complemento a 1) e poi sommando 1 al risultato ottenuto.

In questo caso, ad ogni numero (senza segno) vengono aggiunti degli zeri a sinistra, fino a raggiungere un byte. Se il numero è positivo, si lascia invariato, se è negativo si converte in complemento a due (ottenendo sempre il primo bit = 1, come in precedenza).

È ovvio che 00000000 e 10000000 hanno significati differenti.



OPERAZIONI CON I BINARI

Per procedere con la somma tra binari, come operazioni in colonna, occorre tenere conto delle seguenti indicazioni:

$$\begin{array}{r} 0 + \\ \hline 0 \end{array} = \quad \begin{array}{r} 0 + \\ \hline 1 \end{array} = \quad \begin{array}{r} 1 + \\ \hline 0 \end{array} = \quad \begin{array}{r} 1 + \\ \hline 0 \end{array} =$$

1 ← **riporto**

Dal momento che un computer è in grado di compiere solo somme e confronti, ci serve una procedura che consenta di convertire ogni operazione, in somma (o sequenze di somme).

Vediamo come farlo in decimale (il procedimento può essere esteso, poi, ai binari con lo stesso ragionamento).



OPERAZIONI CON I BINARI

SOTTRAZIONI

La sottrazione può essere convertita in somma nel modo seguente:

$$25 - 14 = 25 + (-14)$$

MOLTIPLICAZIONI

La moltiplicazione può essere convertita sommando un fattore, per sé stesso, tante volte quanto è il valore assunto dall'altro fattore (in genere si sceglie il minimo tra i due):

$$25 \times 3 = 25 + 25 + 25$$

DIVISIONE

La divisione può essere ottenuta sottraendo il divisore al dividendo e iterando il procedimento fino a quando il risultato non diventi minore dello stesso divisore:

$$10 : 3 \rightarrow 10 - 3 = 7 - 3 = 4 - 3 = 1$$

Il numero di sottrazioni effettuate corrisponde al risultato mentre il valore finale ottenuto, indica il resto della divisione.



L'OVERFLOW

Come visto, per questioni tecnologiche tutti i computer, senza alcuna eccezione, trattano i numeri sempre con una quantità fissa di cifre binarie (ad esempio 16, 32 o più).

Quando l'elaboratore esegue un'operazione il cui risultato eccede il numero di cifre permesso, la computazione si arresta immediatamente e viene segnalato l'errore di OVERFLOW.



L'OVERFLOW

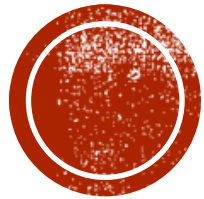
Ad esempio, se la rappresentazione è a 32 bit senza segno e si vuole eseguire la seguente operazione:

$$\begin{array}{r} 3.000.000.000 \times \\ 2.000.000.000 = \\ \hline \end{array}$$

***** Errore di **OVERFLOW**

In questo caso l'elaborazione si arresta.





ARCHITETTURA DEI CALCOLATORI



LA MACCHINA DI VON NEUMANN

In un computer possiamo distinguere tre tipi di unità funzionali:

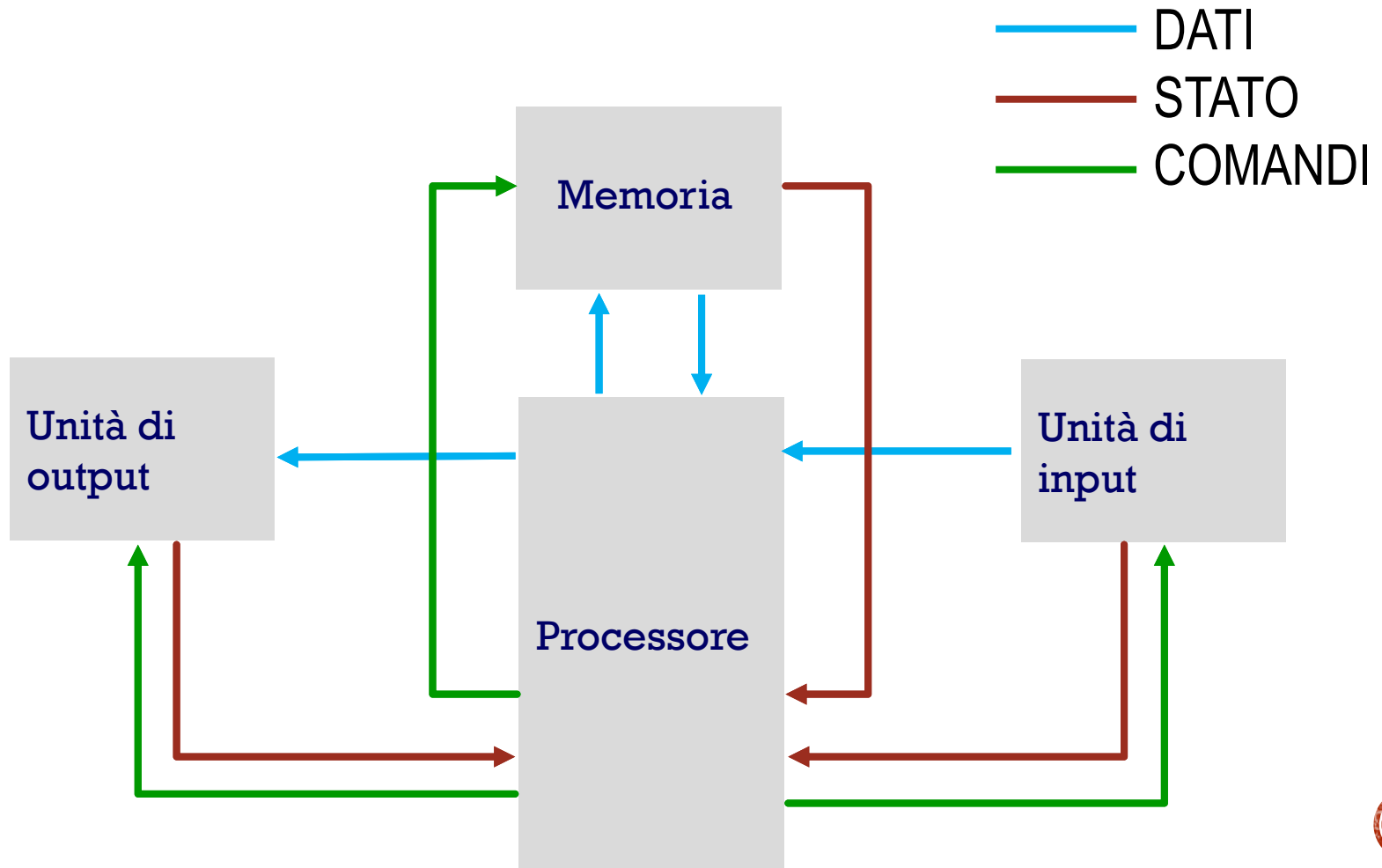
- Processore (fornisce la capacità di elaborazione delle informazioni);
- Memoria (centrale e di massa);
- Dispositivi di input/output.

Le parti comunicano attraverso un canale detto BUS, costituito da un insieme di linee elettriche digitali.

La maggior parte degli elaboratori moderni (compresi gli smartphone ed i tablet) sono basati sulla macchina di Von Neumann.



SCHEMA A BLOCCHI DI UN ELABORATORE



PROCESSORE

Dunque, è composto da blocchi con funzionalità diverse:

- 2^n CPU (Central Processing Unit);
- FPU (Floating Point Unit);
- Cache L1 (o di primo livello);
- Interfacce varie.

Se tutto questo è integrato all'interno di un unico chip, prende anche il nome di microprocessore. I moderni processori sono talvolta dotati anche di una Cache L2 (o di secondo livello).

Di solito le cache L1 sono più piccole (128KB o 256 KB) quindi integrate con lo stesso chip, mentre le cache L2 sono più grandi (valori tipici vanno da 512KB fino a 8MB o anche più), pertanto talvolta sono implementate in un circuito esterno (quindi con velocità di accesso minore).

In questo modo si riescono a gestire esigenze differenti in fase di elaborazione dei programmi.



CENTRAL PROCESSING UNIT (CPU)

Svolge tutte le operazioni di:

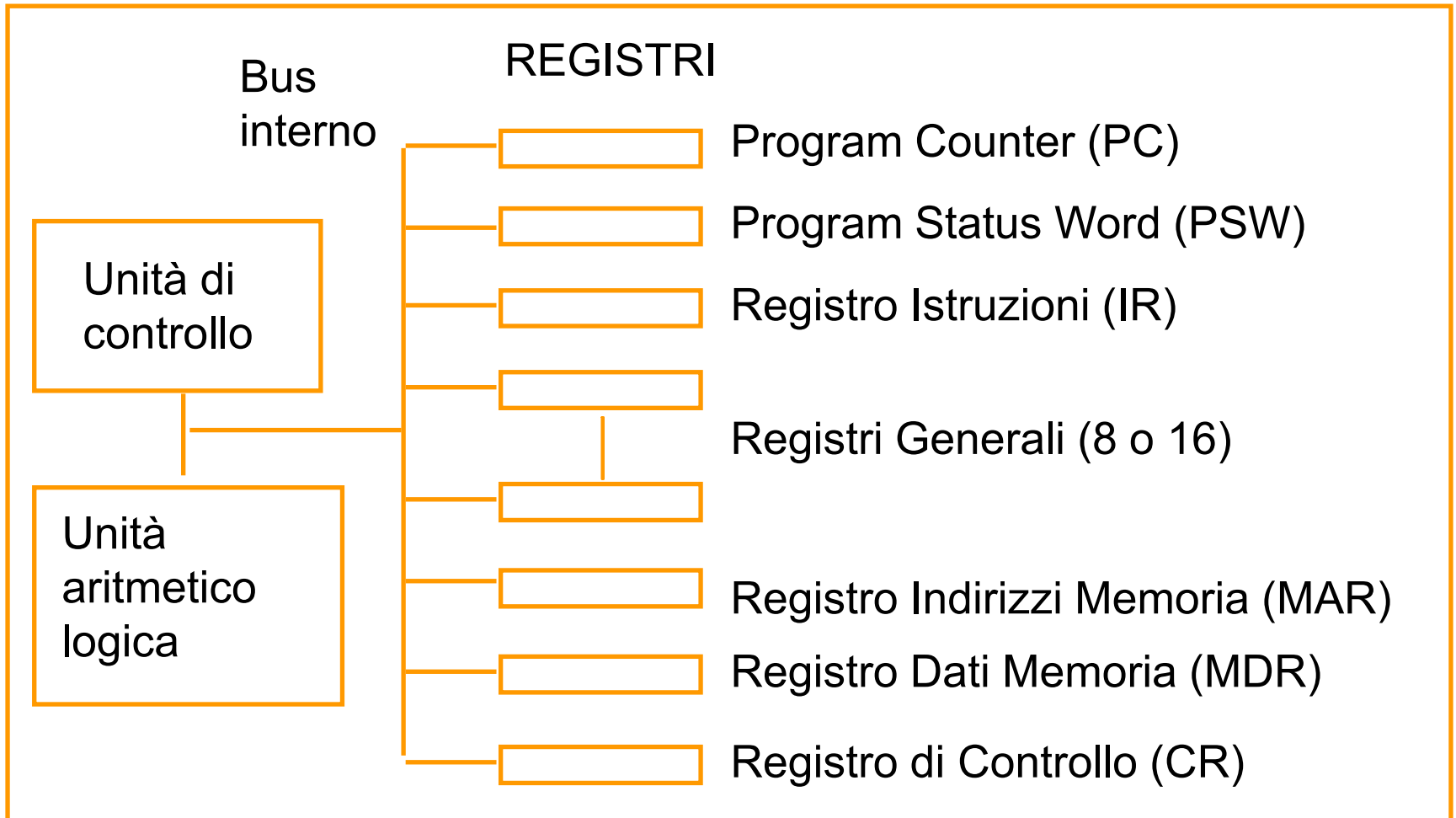
- elaborazione numerica;
- controllo e coordinamento di tutte le attività.

Si suddivide in:

- Unità logico-aritmetica (ALU);
- Unità di controllo (CU);
- Registri.



COMPONENTI DELLA CPU



FLOATING POINT UNIT (FPU)

L'unità di calcolo in virgola mobile è una parte del processore che si contraddistingue per essere specializzata nell'esecuzione di calcoli matematici in virgola mobile (dotati quindi di parte intera e parte decimale).

La maggior parte delle operazioni di calcolo svolte dalla CPU sono operazioni aritmetiche di base (somme, sottrazioni, moltiplicazioni e divisioni) effettuate sull'insieme dei numeri naturali o sull'insieme dei numeri relativi, utilizzando la ALU.

Quando invece è necessario effettuare calcoli più complessi (esponenziali, trigonometrici, etc.), lavorando con numeri reali, entra in gioco la FPU.



ARITHMETIC LOGIC UNIT (ALU)

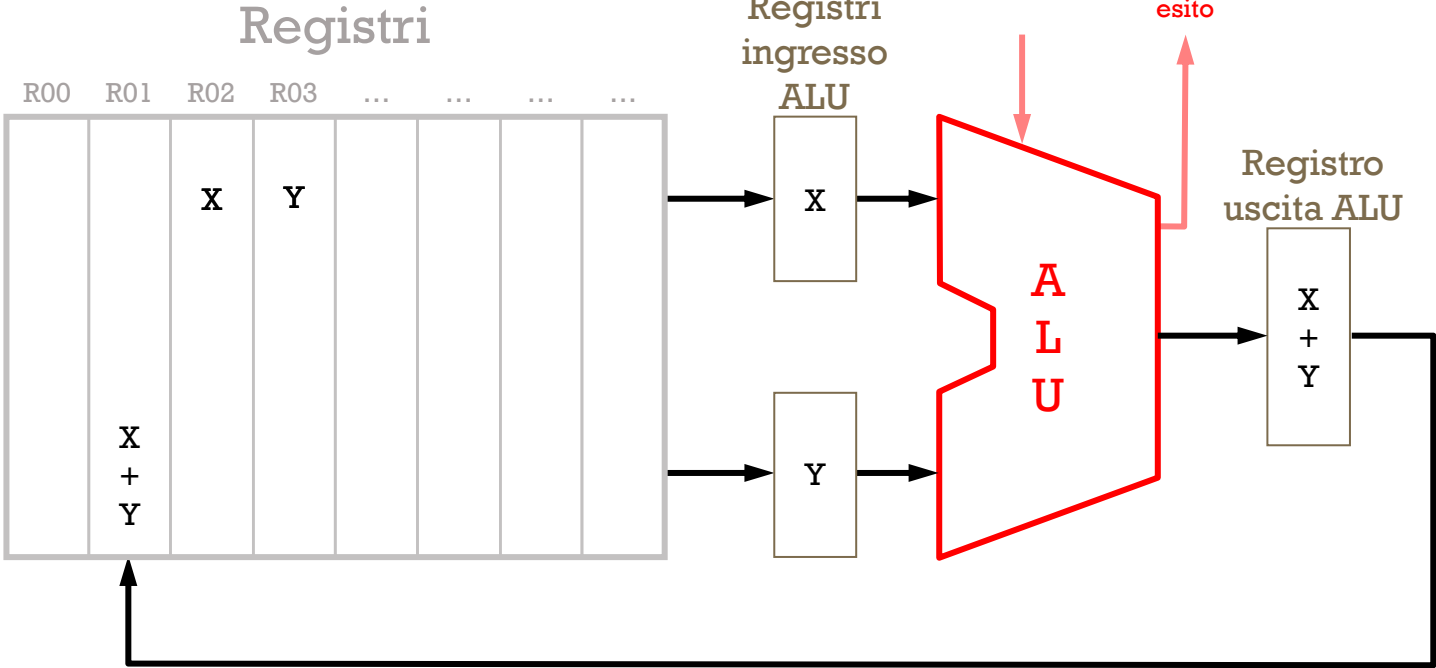
Svolge tutti i calcoli logici ed aritmetici operando direttamente con i registri.

In realtà, è costituita da circuiti elettronici in grado di eseguire solamente la somma di due numeri binari contenuti in due registri, oppure il confronto tra due numeri, sempre presenti in altrettanti registri.

Le altre operazioni vengono eseguite come combinazione delle sole operazioni indicate sopra (così come visto in precedenza).



ARITHMETIC LOGIC UNIT (ALU)



REGISTRI

Sono delle piccole porzioni di memoria locale utilizzata per il salvataggio di:

- dati acquisiti dalla memoria centrale o dalle unità di input;
- risultati delle operazioni eseguite dall' ALU.

I registri sono in numero limitato (tipicamente non oltre 256).

Sono unità di memoria estremamente veloci, di dimensione pari a 16, 32 o 64 bit (tale valore dipende dall'architettura del processore).



REGISTRI

I registri contengono dati ed istruzioni che vengono immediatamente elaborati.

Esistono due tipi di registri:

- i registri speciali utilizzati dalla CU per scopi particolari;
- i registri di uso generale (registri aritmetici).



CLOCK

Ogni elaboratore contiene un circuito di temporizzazione (detto clock) che genera un riferimento temporale comune per tutti gli elementi del sistema.

- $T =$ **periodo** di clock
- $f =$ **frequenza** di clock ($= 1/T$)

La frequenza di clock è misurata in Hertz (Hz).



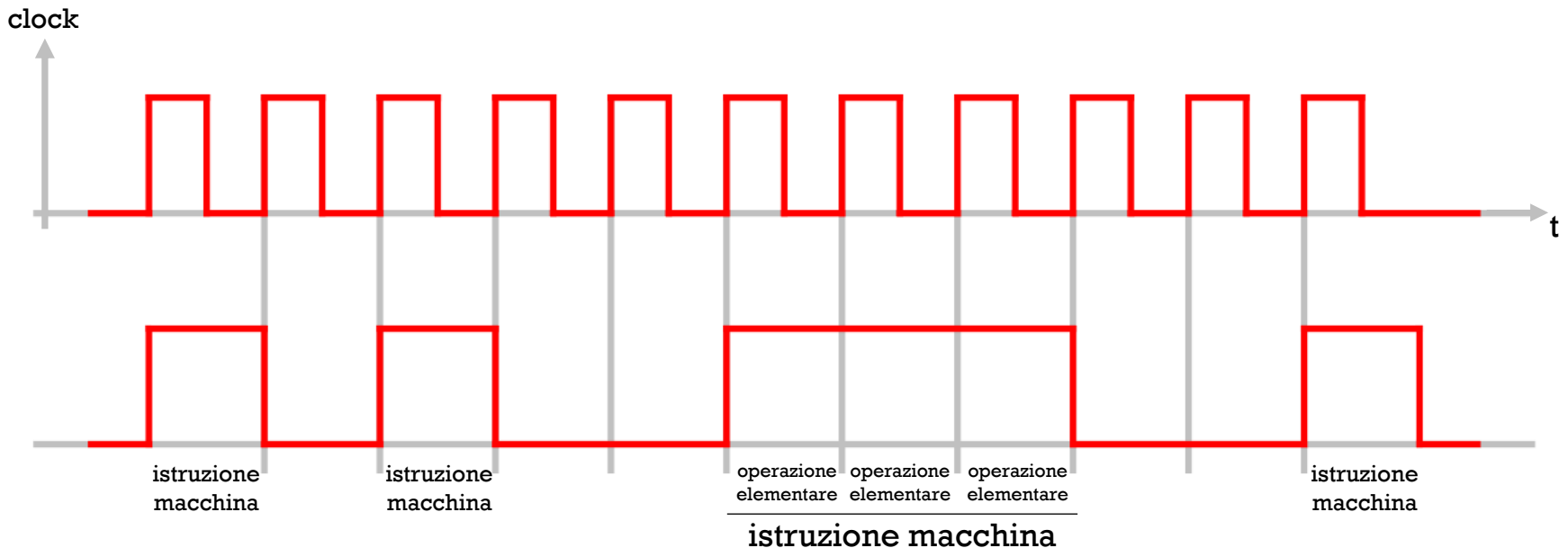
TEMPISTICA DELLE ISTRUZIONI

Un **ciclo-macchina** è il tempo necessario per svolgere un'*operazione elementare*. È un multiplo del periodo del clock.

Un'*istruzione macchina* è ottenuta da una sequenza di operazioni elementari. Dunque, l'esecuzione di un'*istruzione macchina* richiede un numero intero di cicli macchina, variabile a seconda del tipo di istruzione.



TEMPISTICA DELLE ISTRUZIONI



Nell'esempio sopra, le prime due istruzioni macchina e l'ultima richiedono esattamente una operazione elementare ciascuna, la terza istruzione, invece, richiede l'esecuzione di 3 operazioni elementari. Si sta ipotizzando che 1 ciclo-macchina coincida con il periodo del clock.



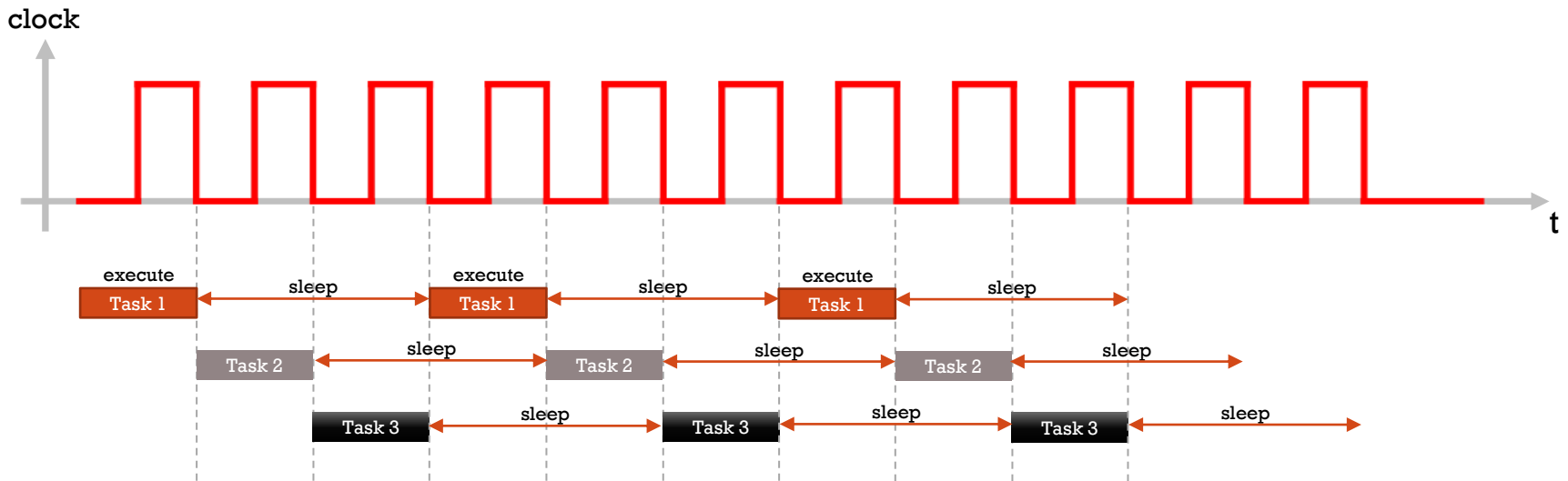
VELOCITÀ DEL MICROPROCESSORE

La velocità di elaborazione di un processore dipende dalla frequenza del clock.

I processori attuali hanno valori di frequenza di clock dell'ordine dei GHz (Giga Hertz), il che significa che un processore con tali caratteristiche è in grado di eseguire miliardi di operazioni elementari al secondo.



TIME SHARING

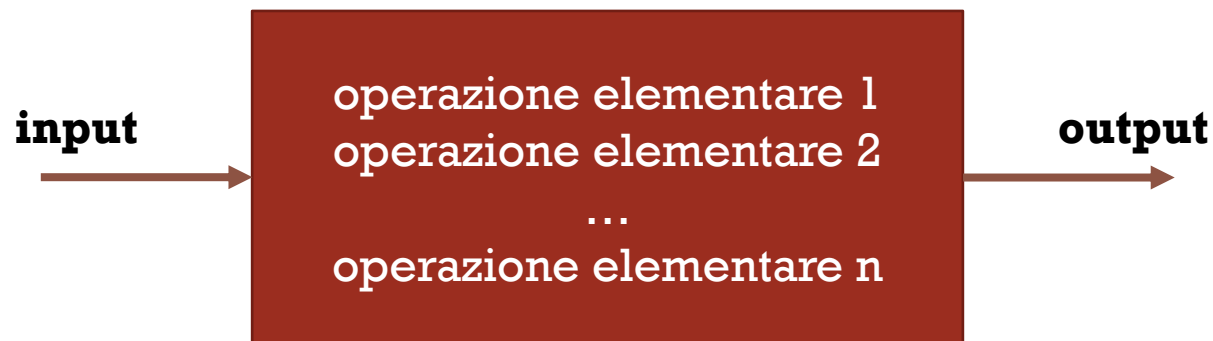


Nell'esempio sopra abbiamo ipotizzato che la fase di esecuzione di ciascuno dei 3 task preveda il coinvolgimento di una singola *operazione elementare*



ALGORITMO

Si definisce *algoritmo* una sequenza ordinata e finita di operazioni elementari (istruzioni) che, presi un numero finito di dati in ingresso, dopo un tempo finito di elaborazione, produca un risultato univoco.



È importante osservare che le operazioni non devono poter essere scomposte in operazioni più semplici e non devono essere ambigue.



MEMORIA

Viene utilizzata per conservare dati e programmi. Si suddivide in:

- **memoria principale;**

Memoria in grado di conservare dinamicamente dati e programmi che il processore sta utilizzando.

- **memoria di massa.**

Memoria in grado di conservare permanentemente dati e programmi degli utenti.



MEMORIA PRINCIPALE

La memoria *principale* (o centrale) è utilizzata per ospitare, in modo *temporaneo*, i dati ed i programmi che gli utenti richiedono di eseguire e/o elaborare.

Prevedono *piccole capacità* di memorizzazione e sono in genere costituite da RAM, memorie cache o registri. Alcune memorie centrali, invece, contengono in modo permanente informazioni necessarie al funzionamento dell'hardware (ROM).

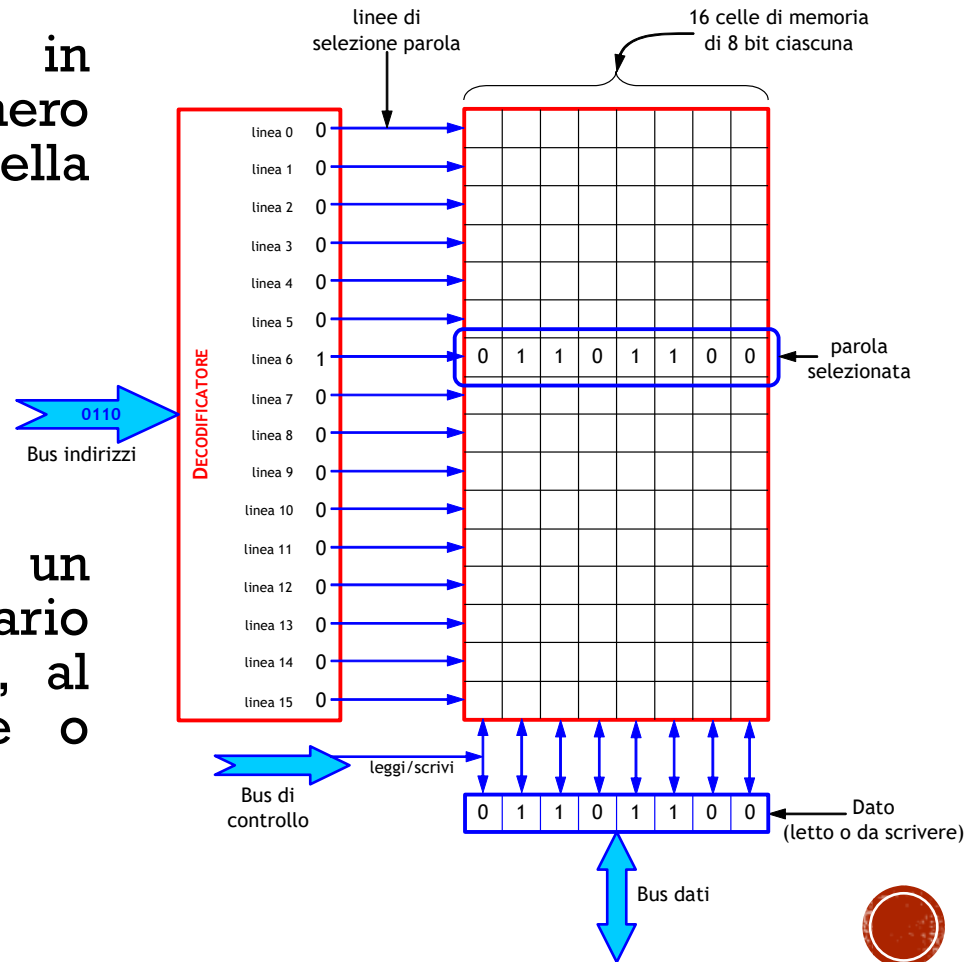
Quando si vuole eseguire un programma, esso viene sempre caricato nella memoria principale (insieme agli eventuali dati da elaborare).



ORGANIZZAZIONE DELLA MEMORIA PRINCIPALE

La memoria è organizzata in *celle*, ciascuna grande un numero di bit pari all'architettura della CPU, tra loro indipendenti.

Ad ogni cella è associato un *indirizzo*, cioè un numero binario progressivo a partire da 0, al quale puntare per leggere o scrivere dati.

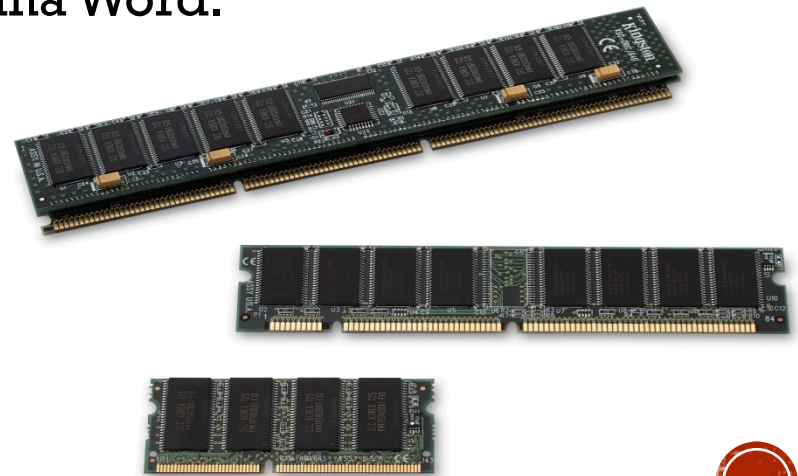


RANDOM ACCESS MEMORY (RAM)

È una memoria *volatile* basata su circuiti digitali, a disposizione della CPU per memorizzare programmi e dati necessari allo svolgimento delle elaborazioni richieste da un utente. A fine elaborazione, lo spazio occupato viene rilasciato.

Il tempo di accesso a tale memoria è il tempo necessario per leggere o scrivere l'informazione in una Word.

Il concetto «accesso casuale» indica che ogni cella ha lo stesso tempo di accesso, indipendentemente dall'indirizzo ad essa associato.



CARATTERISTICHE DELLA RAM

La dimensione di una RAM varia a seconda del tipo di computer e viene espressa in GB.

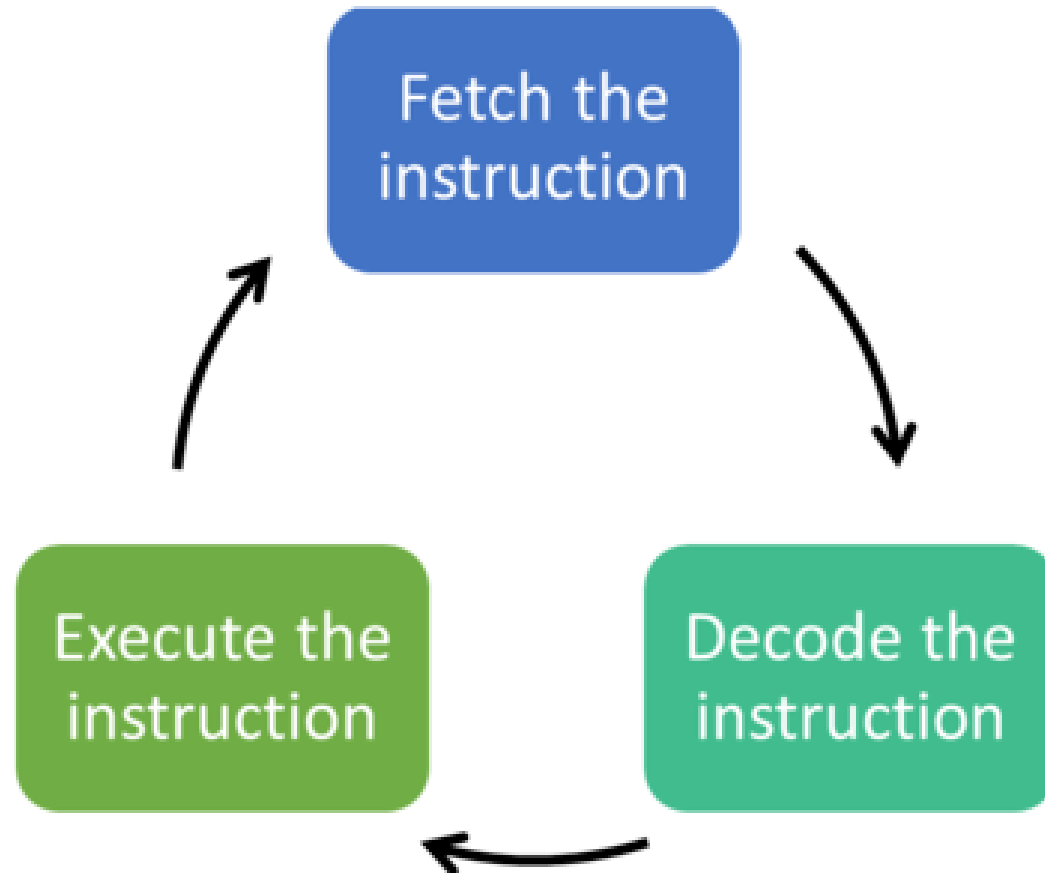
Le dimensioni tipiche della RAM di un calcolatore moderno vanno da 1GB a 16GB. La RAM di un Server, invece, può arrivare fino a 256GB.

Il tempo di accesso è dell'ordine di poche decine di nano-secondi (10^{-9} sec).

La RAM è una memoria costosa (il costo va inteso in valuta/bit).



CICLO FETCH-DECODE-EXECUTE



CICLO FETCH-DECODE-EXECUTE

1. Programma e dati sono caricati in memoria centrale (RAM).
2. La CU **ESTRAE** la prima istruzione del programma dalla memoria (puntandola attraverso l'indirizzo, indicato nel registro PC) e la trasferisce nel registro IR, quindi incrementa di uno l'indirizzo in PC, predisponendosi alla successiva istruzione.
3. L'istruzione viene **DECODIFICATA** ed **ESEGUITA**, prelevando eventuali dati associati, dalla memoria, sempre attraverso l'indirizzo corrispondente. I dati vengono in genere trasferiti nei registri generali.
4. La CU passa all'istruzione successiva, attraverso l'indirizzo presente nel registro PC.



READ ONLY MEMORY (ROM)

È una memoria di *sola lettura* che viene scritta direttamente dal produttore del computer su circuiti appositi.

Viene utilizzata per contenere le informazioni di inizializzazione usate ogni volta che si accende l'elaboratore. In alcuni casi, contiene un software particolare (chiamato firmware), installato all'interno sempre dal produttore.

Esistono diversi tipi di ROM, nate come evoluzione di quella sopra:

- PROM (Programmable ROM);
- EPROM (Erasable Programmable ROM);
- EEPROM (Electrically Erasable Programmable ROM);
- EAROM (Electrically Alterable ROM);
- Flash ROM.



READ ONLY MEMORY (ROM)

Una *PROM* (evoluzione di una ROM) viene prodotta vuota e può essere scritta da un utente, in seguito, utilizzando un'apposita attrezzatura, una volta sola.

Una *EPROM* (evoluzione di una PROM), può essere cancellata per intero, tramite esposizione ai raggi UV e riscritta.

Una *EEPROM* (evoluzione di una EPROM), può essere cancellata in blocchi di bit (al fine di essere riscritta), tramite l'applicazione di una tensione di valore superiore a quella di lavoro.

Una *EAROM* (che è un tipo particolare di EEPROM) può essere modificata in un singolo bit, invece che in blocchi di bit).

Una *Flash ROM*, infine, è una normale Flash utilizzata come ROM.

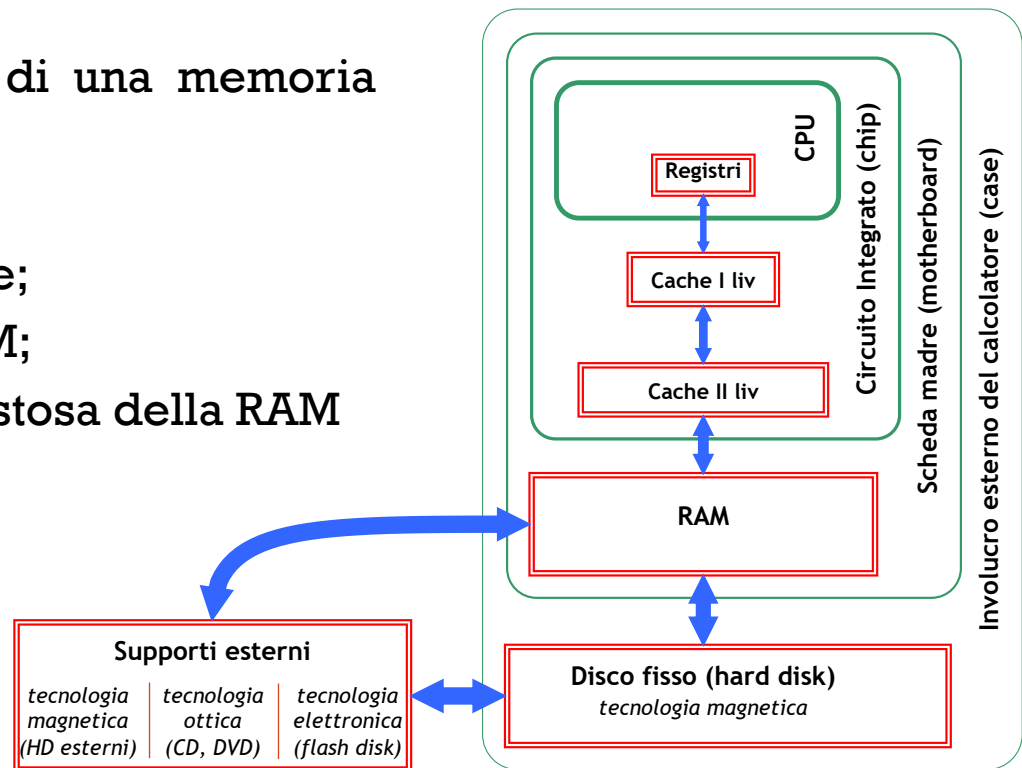


MEMORIA CACHE

Per migliorare le prestazioni di un computer si inserisce una memoria intermedia tra CPU e RAM, detta Cache.

Caratteristiche principali di una memoria Cache sono:

- è interna al processore;
- è più veloce della RAM;
- è di gran lunga più costosa della RAM ma anche più piccola.



MEMORIA CACHE

I dati e le istruzioni utilizzati con maggior frequenza vengono memorizzati nella *cache*, in modo da diminuirne il tempo di accesso ed aumentare, quindi, le prestazioni medie della CPU.

Diventa cruciale, dunque, il metodo per selezionare i dati e le istruzioni da inserire nella cache.

Le cache di I livello (L1), nei computer moderni, valgono 128KB o 256KB, mentre quelle di II livello (L2), hanno valori che vanno da 512 KB a 8÷ 12 MB.



MEMORIA SECONDARIA

La memoria *secondaria* (o di massa) è utilizzata per conservare in modo *permanente* i dati ed i programmi degli utenti.

Richiede *grande capacità* di memorizzazione ed è principalmente costituita da dischi magnetici, CD-ROM, DVD, Flash Memory o nastri.

Quando si vuole eseguire un programma (così come visto nell'esempio precedente), esso viene sempre caricato dalla memoria di massa (insieme agli eventuali dati da elaborare) alla memoria principale.



MEMORIA SECONDARIA

Tre tecnologie possibili:

- **Magnetica**

- ✓ Dischi magnetici (Floppy-disk, Hard-disk)
 - Accesso casuale
 - Operazioni di Lettura/Scrittura
- ✓ Nastri magnetici
 - Accesso sequenziale (legato alla posizione del dato)
 - Operazioni di Lettura/Scrittura

- **Ottica**

- ✓ CD-ROM, DVD
 - Accesso casuale (tipicamente solo scrittura)

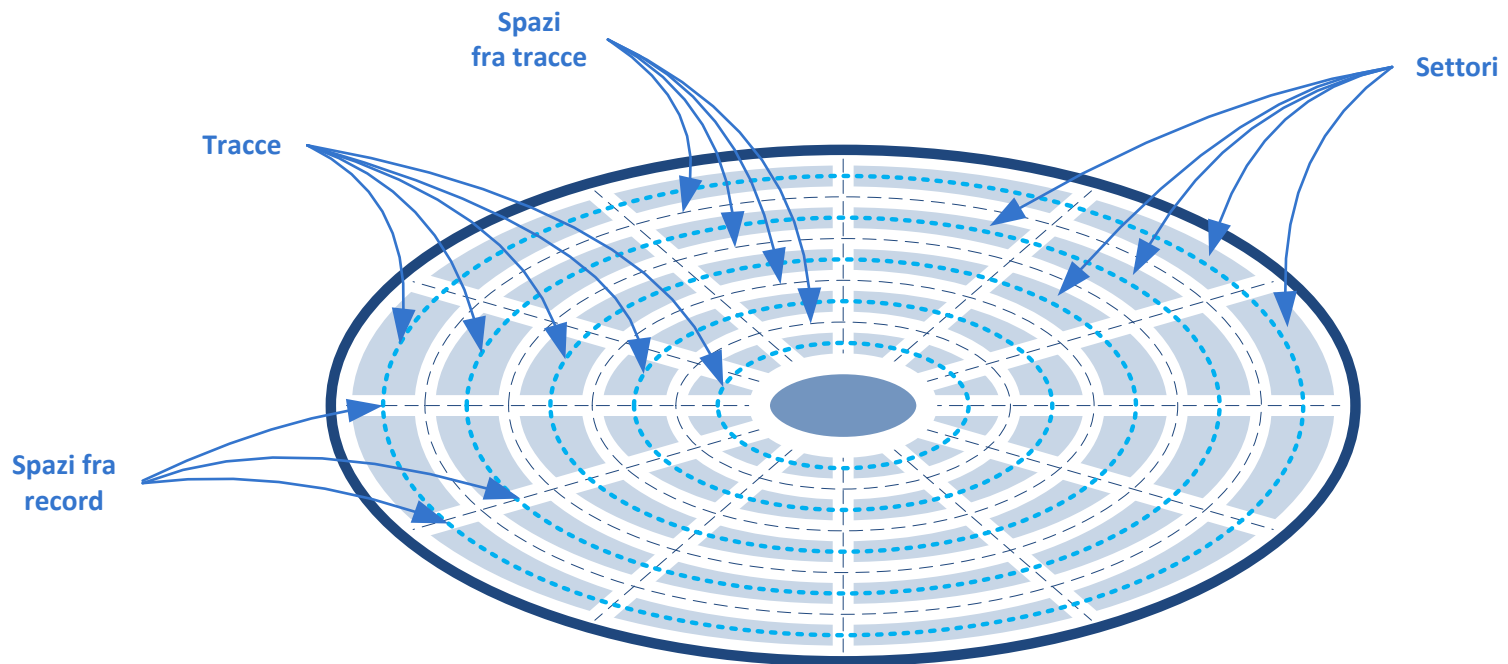
- **Flash memory**

- ✓ Pen drive, schede di memoria (Sd, micro Sd, memory stick, compact flash, etc.)
 - Accesso casuale (lettura/scrittura)



ORGANIZZAZIONE FISICA DEI DATI NEI DISCHI

Gli hard disk sono suddivisi in *tracce* concentriche e *settori*:



ORGANIZZAZIONE FISICA DEI DATI NEI DISCHI

Ogni *settore* è una “fetta” di disco. I settori suddividono ogni *traccia* in porzioni di circonferenza dette *blocchi*.

La suddivisione di un disco in tracce e settori viene indicata con il termine *formattazione*.

Per effettuare un’operazione di lettura o di scrittura su un blocco, è necessario che la testina raggiunga il blocco desiderato.

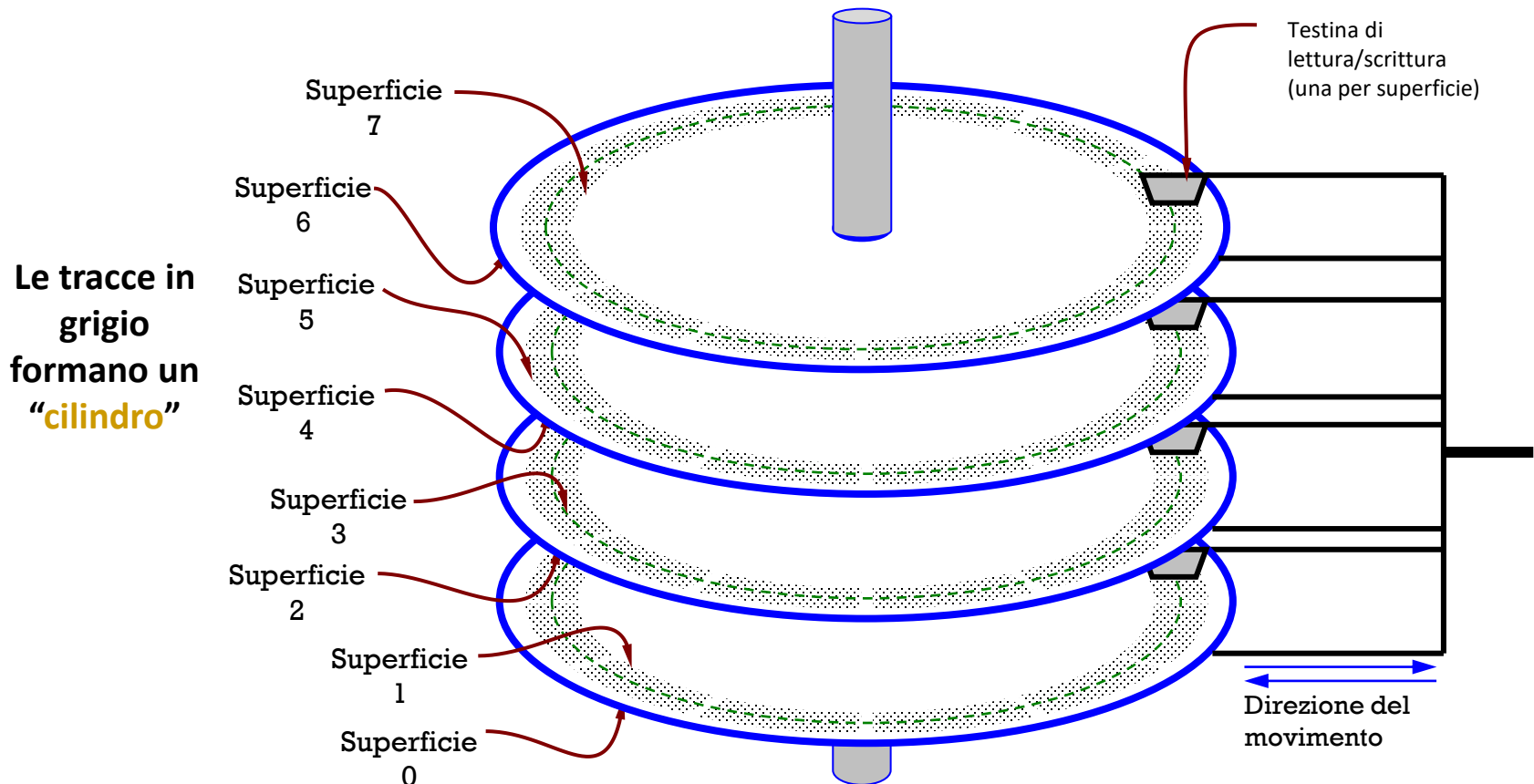
Ciascun disco può essere gestito in due modalità:

- Single-side.
- Double-side.

Per aumentare la capienza, spesso si montano più dischi, double-side, all’interno dello stesso case.



ORGANIZZAZIONE FISICA DEI DATI NEI DISCHI



ORGANIZZAZIONE FISICA DEI DATI NEI DISCHI

Il *tempo di accesso* ai dati (access time) dipende da tre fattori fondamentali:

- **Seek time**

La testina si sposta in senso radiale fino a raggiungere la traccia desiderata (impiega 1ms per tracce adiacenti, in generale tra 5 e 15ms in media).

- **Latency time**

Quando è in funzione, i dischi ruotano ad una velocità tra 5400 e 10800 rpm. Occorre attendere, quindi, che il settore desiderato passi sotto la testina. Varia tra 2,7 e 5,4ms.

- **Transfer time**

Tempo di lettura vero e proprio. In genere dipende dalla velocità di rotazione del disco e dalla densità di bit per cm lineare (della traccia). In genere si va da 5 a 20 MBps.



DISCHI MAGNETICI

Sono dotati di una grande capacità di memorizzazione. Fino a pochi anni fa i floppy disk rappresentavano il supporto mobile più usato.

Oggi sono stati largamente sostituiti da altri supporti quali, ad esempio, gli hard-disk esterni o i pen drive.

Un hard-disk, oggi, può raggiungere dimensioni di diversi TB, sono memorie più lente della memoria centrale, ma molto meno costose.

