

# **Architettura e organizzazione dei calcolatori elettronici**

## **Porte logiche**

**Liceo G.B. Brocchi**  
**Classi prime Scientifico - opzione scienze applicate**  
Bassano del Grappa, Ottobre 2022  
Prof. Giovanni Mazzocchin

# Il sistema di elaborazione

- Domande da cui partire:
  - come è fatto un computer?
  - che cos'è un dato?
  - che cos'è il software?
  - i computer sono intelligenti?
  - un computer può capire cosa deve fare grazie ad una sorta di «buon senso» umano?

# Il sistema di elaborazione

**Un computer è un sistema artificiale (fatto di componenti elettroniche, elettriche, meccaniche ed ottiche) in grado di elaborare un input per produrre un output**



# Il sistema di elaborazione

**Un computer acquisisce in input dati e programmi e produce in output il risultato di un elaborazione**

Esempio:

**dati in input**: voti di Matematica delle classi quinte del Brocchi degli ultimi 10 anni

**informazioni in output**: grafico che mostra l'andamento delle medie dei voti negli ultimi anni, dettaglio della classe con la media più alta etc...

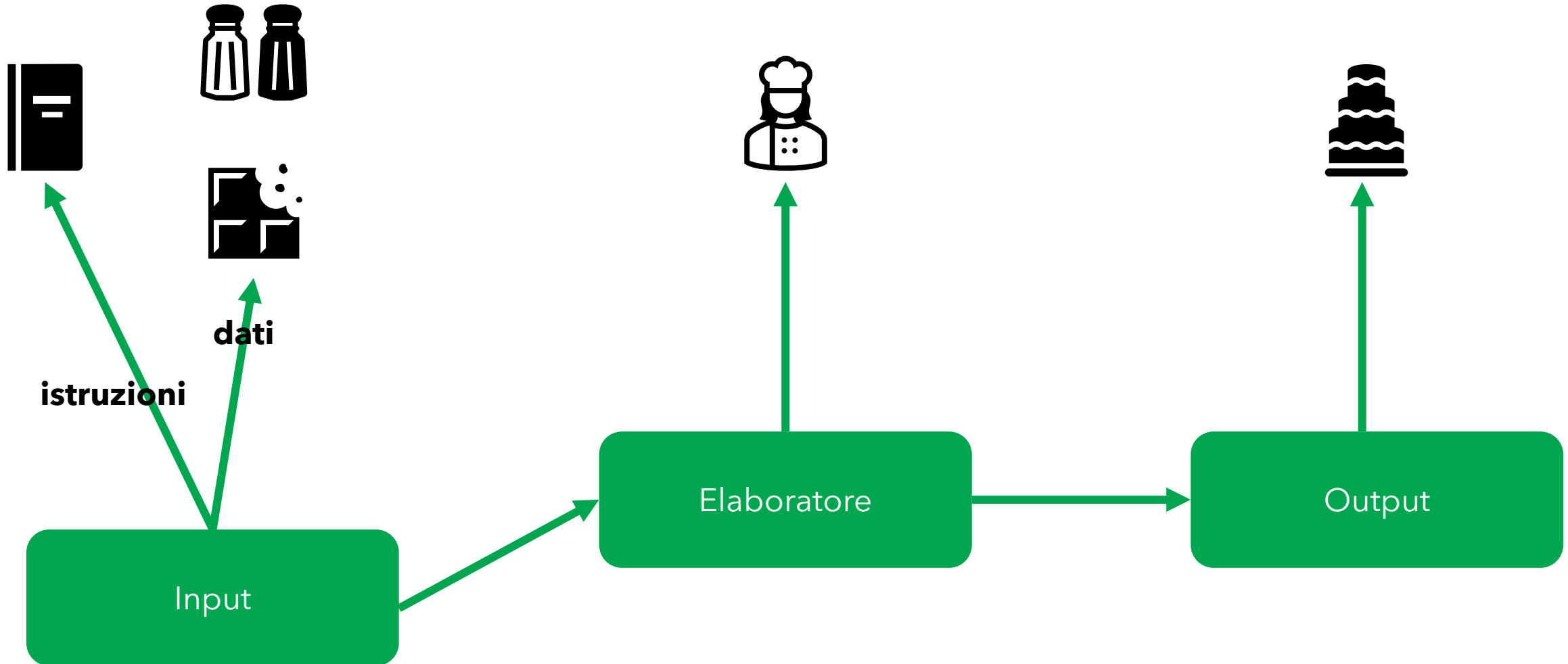
**Un computer elabora dati e produce informazioni sulla base di sequenze ordinate di istruzioni dette programmi**

# Un'analogia culinaria

- Per cucinare un piatto abbiamo bisogno di:
  - una ricetta (sequenza di istruzioni, programma)
  - degli ingredienti (dati da elaborare)
  - un cuoco (elaboratore)
- Il risultato finale (output) piatto finito, ossia qualcosa che possiamo utilizzare (in questo caso mangiare).

# Un'analogia culinaria

**Domanda:** per preparare un nuovo piatto  
devo cambiare cuoco?



# La macchina di von Neumann


- Modello ideato da **John von Neumann** nel 1945 ([https://en.wikipedia.org/wiki/John\\_von\\_Neumann](https://en.wikipedia.org/wiki/John_von_Neumann) *lettura obbligatoria del primo paragrafo*)

**Componente  
«elaboratore»:  
componenti hardware  
(fisiche) in grado di  
eseguire delle  
operazioni prefissate**

**Componente «memoria»:  
contiene dati e istruzioni**

# La macchina di Von Neumann

- Esempio 1: vogliamo utilizzare una macchina di Von Neumann per fare delle elaborazioni statistiche sui dati anagrafici dei cittadini del comune di Bassano del Grappa:
  - Quali dati forniremo alla macchina? Quali istruzioni?
- Esempio 2: vogliamo utilizzare una macchina di Von Neumann per calcolare l'orbita di una cometa:
  - Quali dati forniremo alla macchina? Quali istruzioni?
- Esempio 3: vogliamo utilizzare una macchina di Von Neumann per fare previsioni del tempo per la prossima settimana in Veneto.
  - Quali dati forniremo alla macchina? Quali istruzioni?

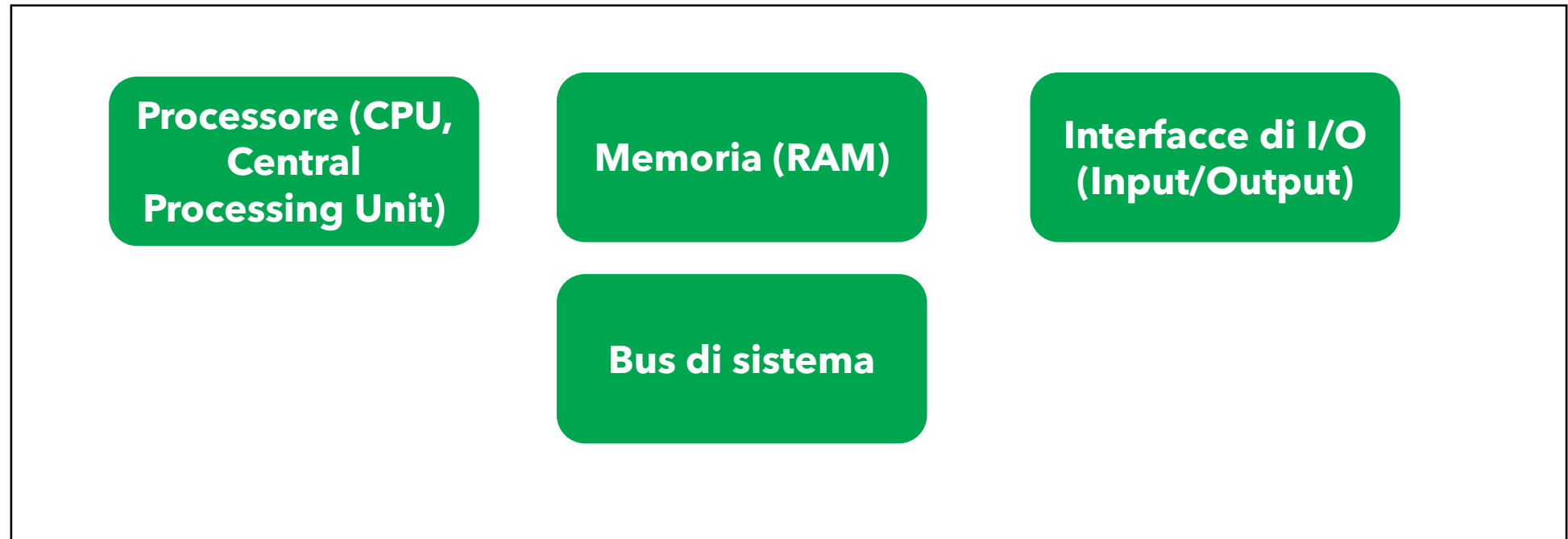


**Domanda delle domande: dobbiamo utilizzare 3 elaboratori diversi per portare a termine i 3 compiti?**



# La macchina di Von Neumann

- La macchina di Von Neumann è un **sistema deterministico**:
  - a fronte dello stesso input produce lo stesso output
  - immaginate una calcolatrice (anche non programmabile, quindi non propriamente una macchina di Von Neumann): oggi è il 28 settembre e le chiedete quanto fa  $2 + 3$ ... ovviamente vi risponderà 5. Vi risponderà diversamente se le chiedete quanto fa  $2 + 3$  il 1 ottobre?



# Architettura e organizzazione degli elaboratori

- Con il termine **architettura** si indicano le caratteristiche di un sistema di elaborazione visibili al programmatore, come il *formato delle istruzioni*
- Con il termine **organizzazione** si indica l'insieme dei dettagli *hardware* che non riguardano il programmatore.
  - ad es., il tipo di tecnologia utilizzata per realizzare le memorie. Questi dettagli vengono non vengono studiati direttamente nella disciplina «Informatica». Sono più vicini a materie come la Fisica (lo studio delle proprietà di certi materiali) e l'Ingegneria Elettronica.

# Architettura e organizzazione degli elaboratori

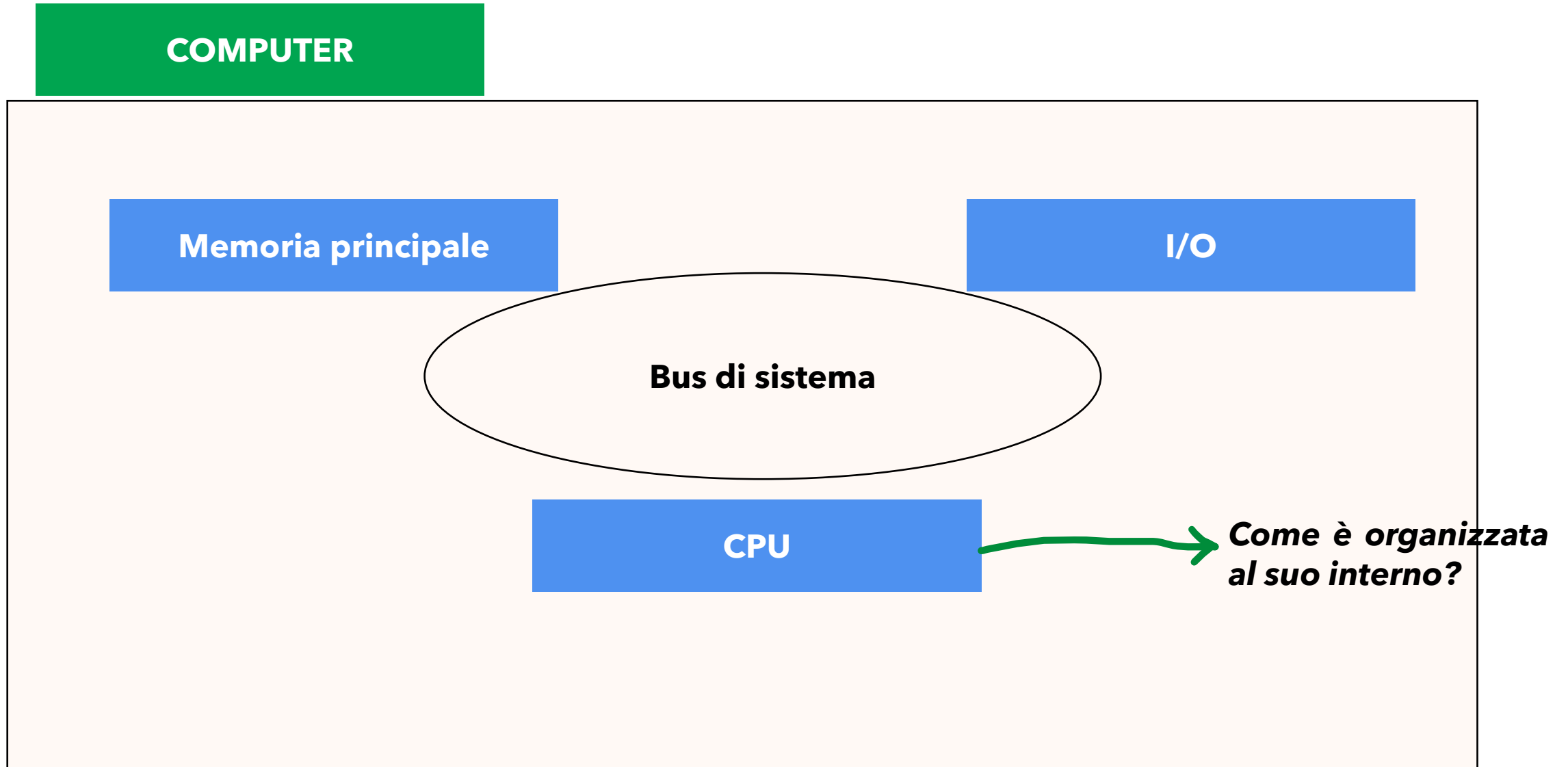
**Caratteristica architetturale:  
Il computer fornisce un'istruzione di  
moltiplicazione**

**Caratteristica organizzativa:  
L'istruzione di moltiplicazione è  
realizzata da un particolare componente  
elettronico**

# Componenti strutturali di un computer

- **CPU (Central Processing Unit):** controlla le operazioni e processa i dati
- **Memoria principale:** memorizza dati e istruzioni
- **I/O (Input/Output):** componente che si occupa del passaggio di dati tra il computer e l'esterno
- **Interconnessioni di sistema:** un meccanismo (possiamo pensare approssimativamente a dei cavi elettrici) che permette la comunicazione tra CPU, memoria e I/O, ad es. i **bus di sistema**

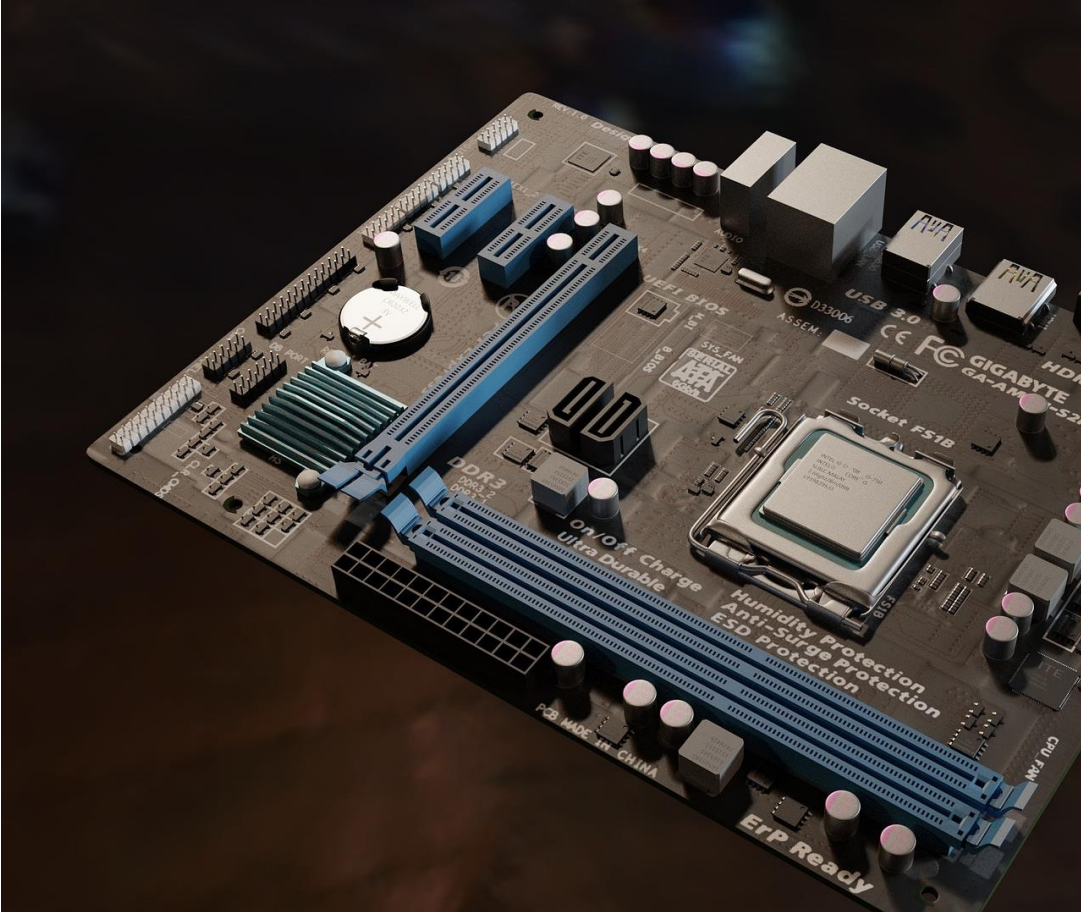
# Componenti strutturali di un computer



# Componenti strutturali della CPU

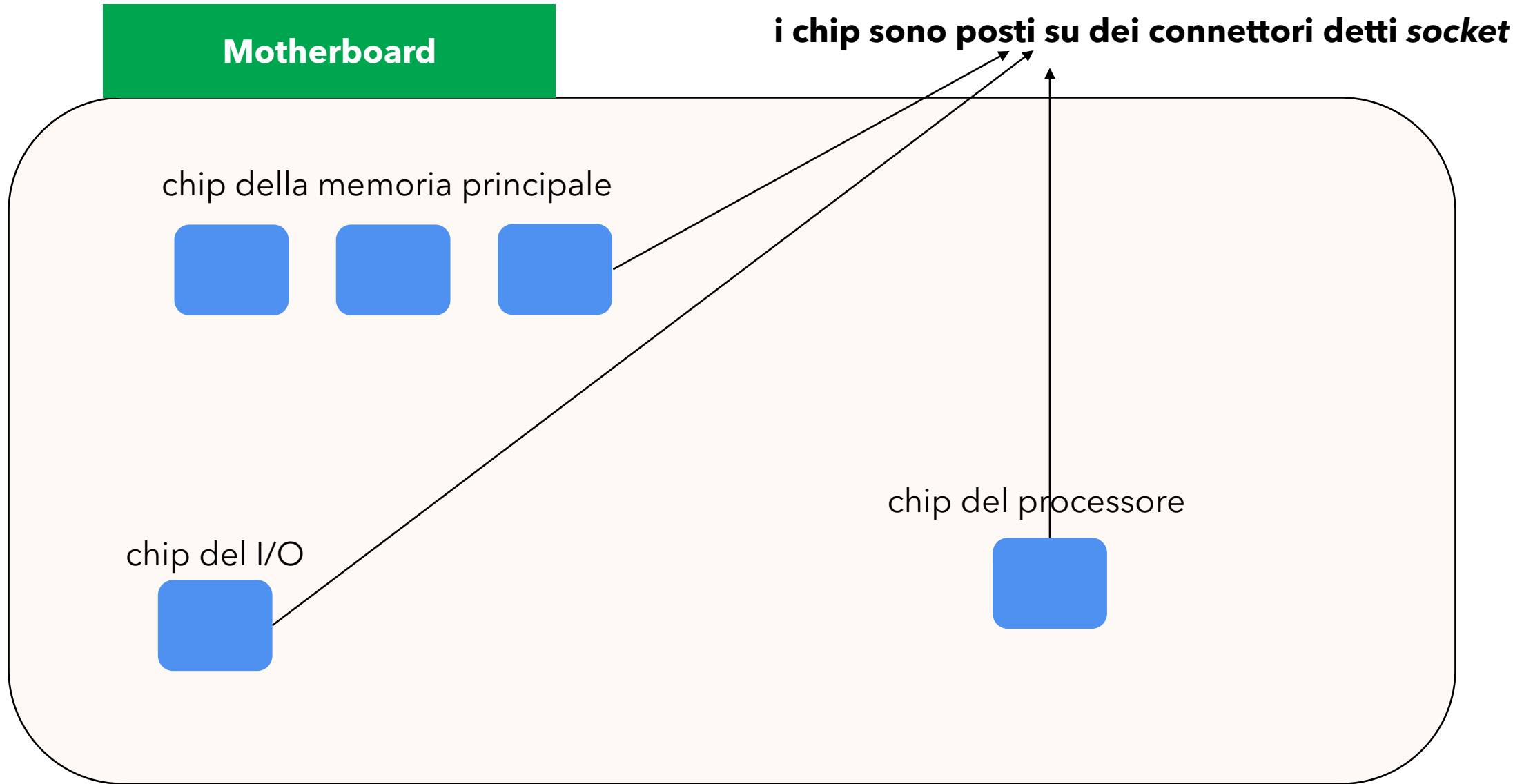
- **Control Unit (CU)**: controlla le operazioni della CPU, e di conseguenza di tutto il computer
- **Arithmetic and Logic Unit (ALU)**: esegue le operazioni aritmetiche (sicuramente almeno le addizioni) e logiche (che vedremo)
- **Registri**: costituiscono una memoria interna (molto più piccola della memoria principale). Non confondere la memoria rappresentata dai registri con la memoria principale!
- **Interconnessioni** tra *CU*, *ALU* e *registri*

# La motherboard (scheda madre)



- Un **Printed Circuit Board (PCB, circuito stampato)** è una scheda rigida sulla quale sono posizionate e interconnesse delle componenti elettroniche e *chip*
- Il **PCB** più importante all'interno di un computer è detto **motherboard** (*scheda madre*). Sulla scheda madre sono posizionati diversi *chip*
  - *un chip è un pezzo di materiale semiconduttore (generalmente si usa il silicio) contenente diversi circuiti (studierete le proprietà di questi materiali in Fisica, al quinto anno)*
  - *un chip può contenere fino a milioni di componenti chiamati *transistor* (si tratta di interruttori e amplificatori fatti di materiale semiconduttore)*

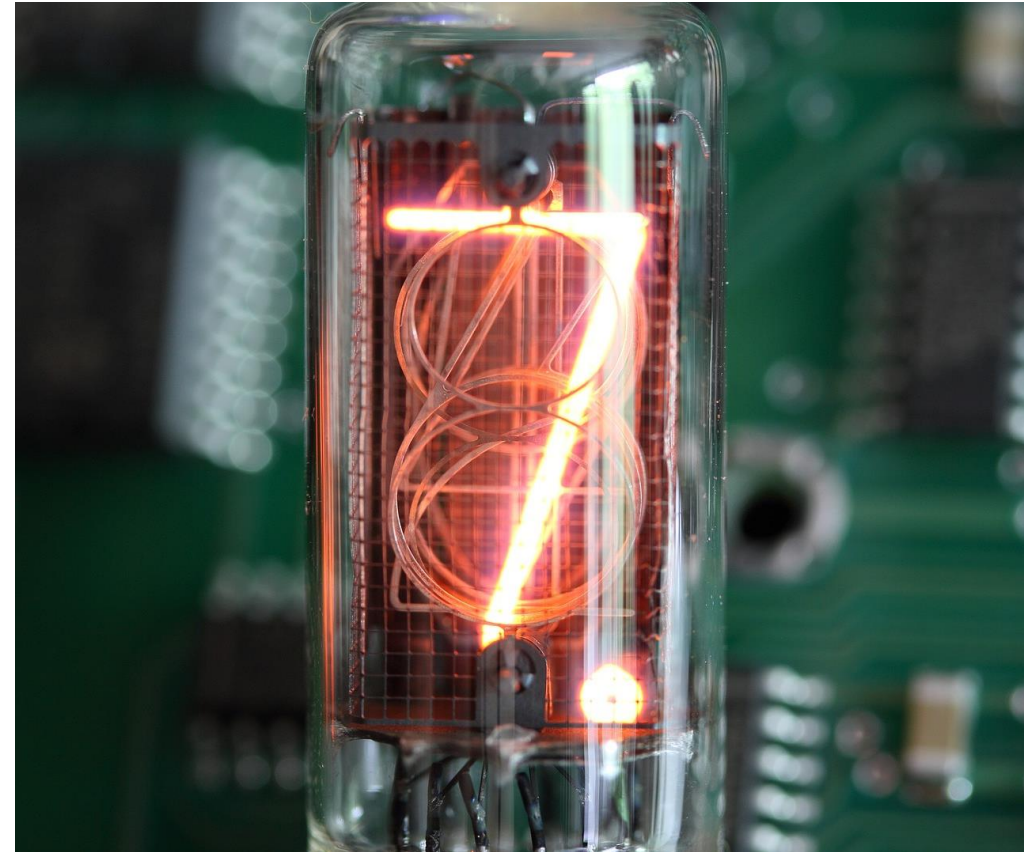
# La motherboard (scheda madre)





# IAS computer (Institute of Advanced Studies, Princeton, USA)

- Si inserisce nella cosiddetta *prima generazione* della storia del computer
- Sviluppato tra il 1946 e il 1952 da John von Neumann (1903-1957) e colleghi
- All'epoca non esistevano ancora i transistor, si utilizzavano i *tubi a vuoto* (simili ai *tubi a raggi catodici* della vecchie televisioni)
- La descrizione di questo computer costituisce un modello, la cosiddetta macchina di *von Neumann* che abbiamo visto sopra
- I tubi a vuoto erano grandi e consumavano molta energia. Costituivano la limitazione principale di questi computer
- Velocità: 40 000 operazioni per secondo



# I transistor e la seconda generazione di computer

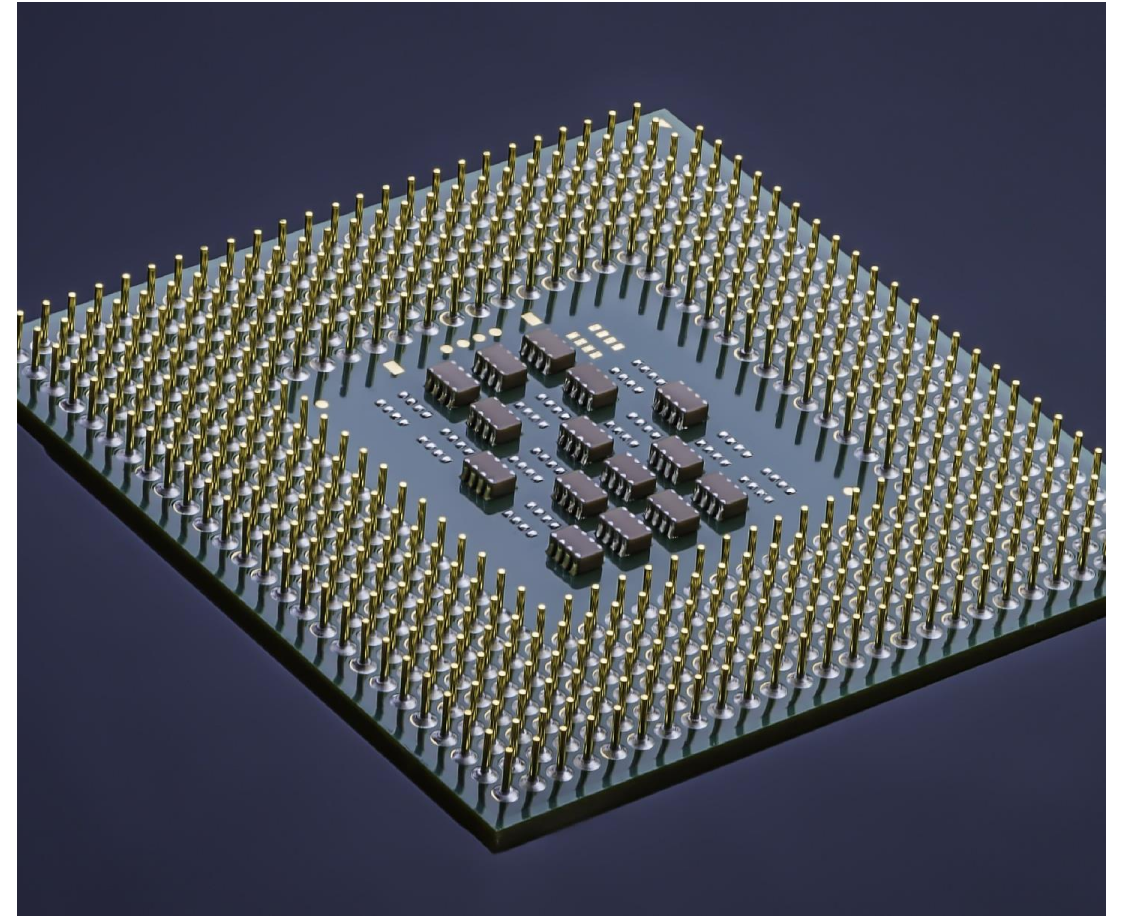
- Il **transistor** venne inventato nel 1947 presso i *Bell Labs* (New Jersey, USA), da John Bardeen, William Shockley e Walter Brattain
- **I transistor funzionano come interruttori e amplificatori di segnali elettrici**
- I transistor erano molto più piccoli, economici ed consumavano molta meno energia dei tubi a vuoto
- Queste caratteristiche dei transistor hanno lanciato una vera e propria *rivoluzione dell'elettronica*
- Nascono anche i primi *software di sistema (sistemi operativi)*, che costituiscono uno strato intermedio tra l'utente e l'hardware
- I computer della seconda generazione arrivavano a velocità anche 5 volte superiori rispetto a quelli della prima





# La terza generazione: i circuiti integrati

- Negli anni '50 i transistor e altre componenti elettroniche costituivano componenti *discrete*, ossia una distinta dall'altra
- Se un computer necessitava di 10 000 transistor, questi dovevano essere saldati insieme uno ad uno. Il processo di assemblaggio divenne rapidamente insostenibile
- Nel 1958 vengono inventati i circuiti integrati e inizia l'era della *microelettronica*
- Un circuito integrato (IC, *Integrated Circuit*) riunisce un insieme anche molto grande di transistor in un unico pezzo (chip) di materiale semiconduttore (spesso si usa il *silicio*)
- La produzione standardizzata di enormi quantità di circuiti integrati, contenenti numeri sempre più grandi di transistor, ha permesso il rapidissimo sviluppo dell'informatica e dell'elettronica avvenuto dagli anni '60 in poi



# Classificazione dei circuiti integrati

- **SSI** (*small-scale integration*): meno di 100 transistor per chip
- **MSI** (*medium-scale integration*): da 100 a 1000 transistor per chip
- **LSI** (*large-scale integration*): da 1000 a 10 000 transistor per chip
- **VSI** (*very large-scale integration*): da 10 000 a 100 000 transistor per chip
- **ULSI** (*ultra large-scale integration*): superiore a 100 000 transistor per chip

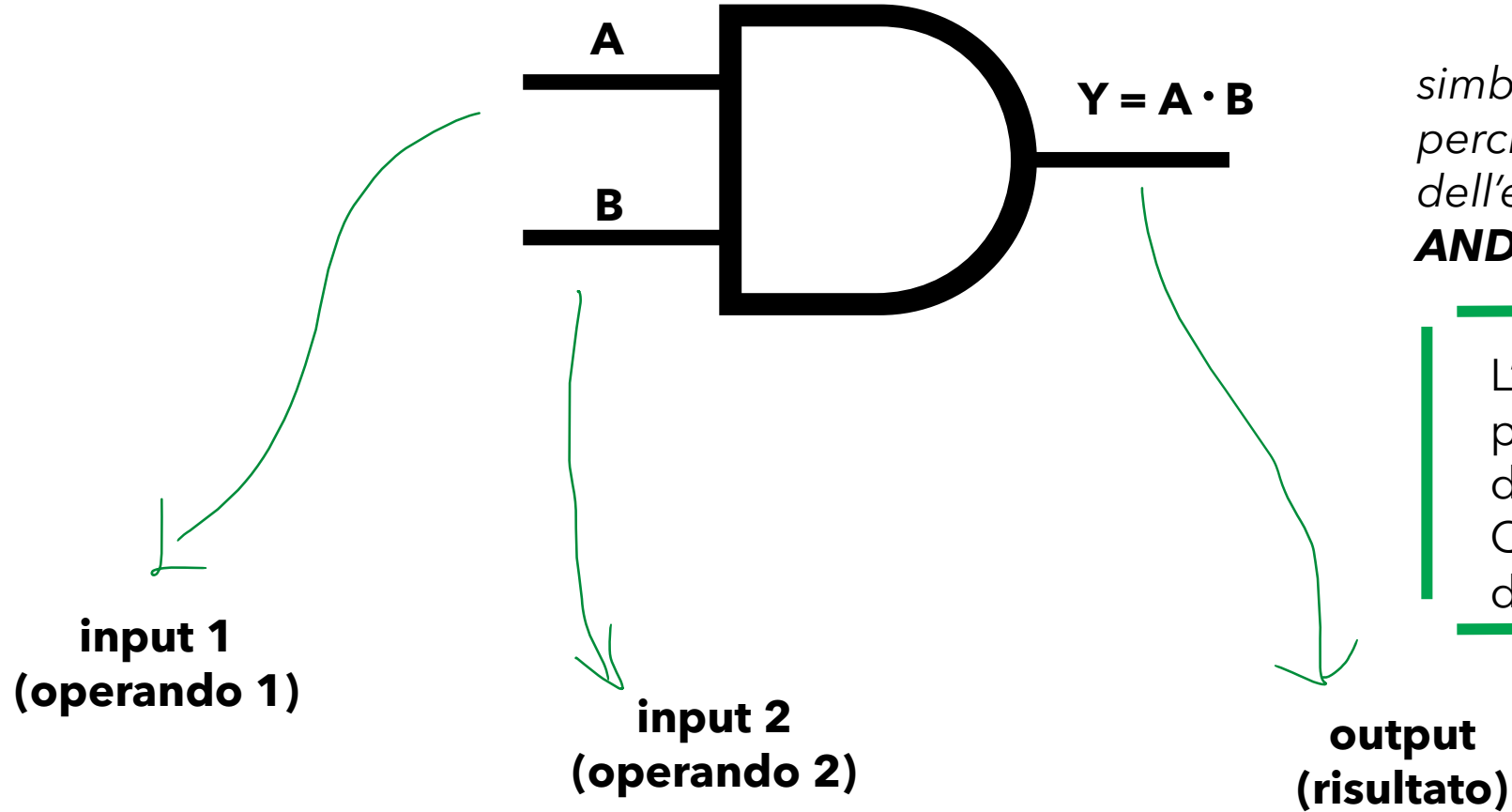
# Le componenti base di un elaboratore digitale

- Premessa: non sapete cosa significa *digitale*. È un concetto che acquisiremo col tempo
- Comunque, un computer deve effettuare le seguenti operazioni di base:
  - memorizzazione di dati;
  - spostamento di dati;
  - elaborazione di dati sulla base di istruzioni;
  - funzioni di controllo
- I componenti minimi per eseguire le operazioni elencate sono:
  - porte logiche, che compongono le cosiddette **reti logiche**
  - celle di memoria

# Variabili logiche (o booleane, o binarie)

- In informatica ed elettronica si utilizza un' «algebra» particolare oltre a quella classica: l'**algebra booleana**
- I calcoli non vengono fatti sui numeri naturali (0, 1, 2, 3, 4 ... infinito), ma su numeri che possono avere solo **2** valori: **true** o **false** (si può dire anche **0** o **1**, **on** o **off**)
- Su questi «numeri», che chiameremo **variabili logiche** (o **variabili booleane**), sono definite delle **operazioni**
- Se ci pensate, anche sui numeri naturali sono definite delle operazioni (somma, sottrazione etc...)
- Nei circuiti digitali queste operazioni vengono effettuate dalle **porte logiche**

# Un'operazione logica: AND



*simbolo ANSI (detto «militare» perché adottato nei progetti dell'esercito USA) della porta logica **AND***

L'operatore **AND** richiede 2 input e produce («restituisce») 1 output: si dice che è un **operatore binario**. Quali sono gli operatori binari dell'aritmetica?

# Un'operazione logica: AND. Come funziona?

- Domanda motivazionale: alle elementari avete imparato a fare le addizioni cercando il risultato in una tabella di questo tipo?

addendo 1	addendo 2	somma
0	0	0
0	1	1
0	2	2
5	0	5
5	1	6
5	2	7
7	7	14
7	8	15

*Domanda motivazionale: le calcolatrici tascabili operano in questo modo? Potrebbero funzionare?*



# Un'operazione logica: AND. Come funziona?

- La tabella che abbiamo visto sopra avrebbe un numero di righe infinito e sarebbe dunque inutilizzabile. Ma abbiamo detto che i valori booleani, a differenza dei numeri naturali, possono assumere solo **2** stati
- La tabella per un operatore booleano sarà sicuramente molto corta!
- **AND** «restituisce» (produce in output) 1 solo se **entrambi** gli input sono uguali a 1

Tabella di verità (*truth table*) di AND

input1	input2	output = input1 AND input2
0	0	0
0	1	0
1	0	0
1	1	1

# Un'operazione logica: AND. Come funziona?

- **Tabella di verità di AND**

Tabella di verità (*truth table*) di AND

input1	input2	output = input1 AND input2
false	false	false
false	true	false
true	false	false
true	true	true

# Un'operazione logica: AND. Come funziona?

- Qui utilizziamo i valori HIGH e LOW (alto e basso), per far capire che gli input delle porte logiche in realtà sono segnali elettrici, per i quali ci sono dei parametri (che vedrete in Fisica) che possono avere valore *alto* o *basso*

**Tabella di verità (*truth table*) di AND**

input1	input2	output = input1 AND input2
LOW	LOW	LOW
LOW	LOW	LOW
HIGH	LOW	LOW
HIGH	HIGH	HIGH

# Un'operazione logica: AND. Notazione

input1 **AND** input2     (*notazione infissa*)

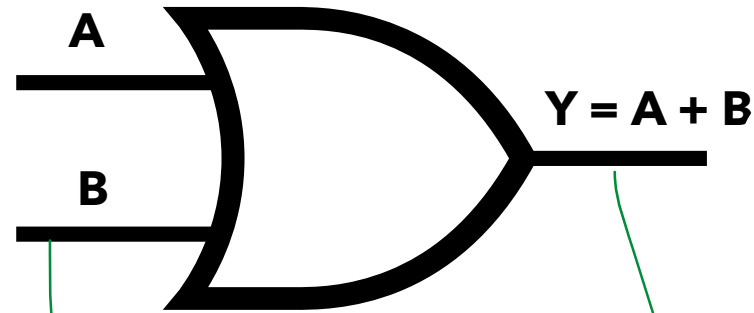
**AND**(input1, input2)     (*notazione prefissa*)

input1  $\wedge$  input 2     (*notazione logico-matematica*)

input1 • input2     (*effettivamente AND è la moltiplicazione*)

*input1input2*     (*senza il punto, come in matematica*)

# Un'operazione logica: OR



*simbolo ANSI (detto «militare» perché adottato nei progetti dell'esercito USA) della porta logica **OR***

L'operatore **OR** richiede 2 input e produce 1 output: si dice che è un **operatore binario**. Quali sono gli operatori binari dell'aritmetica?

input 1  
(operando 1)

input 2  
(operando 2)

output  
(risultato)

# Un'operazione logica: OR. Notazione

input1 **OR** input2     (*notazione infissa*)

**OR**(input1, input2)     (*notazione prefissa*)

input1  $\vee$  input 2     (*notazione logico-matematica*)

input1 + input2

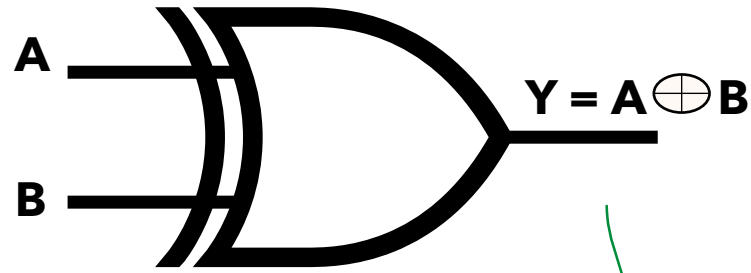
# Un'operazione logica: OR. Come funziona?

- **OR** «restituisce» (produce in output) 1 solo se **almeno** uno dei 2 input è 1

Tabella di verità (*truth table*) di OR

input1	input2	output = input1 OR input2
0	0	0
0	1	1
1	0	1
1	1	1

# Un'operazione logica: XOR (o EXOR)



*simbolo ANSI (detto «militare» perché adottato nei progetti dell'esercito USA) della porta logica **XOR***

input 1  
(operando 1)

input 2  
(operando 2)

output  
(risultato)

L'operatore **XOR** richiede 2 input e produce 1 output: si dice che è un **operatore binario**. Quali sono gli operatori binari dell'aritmetica?



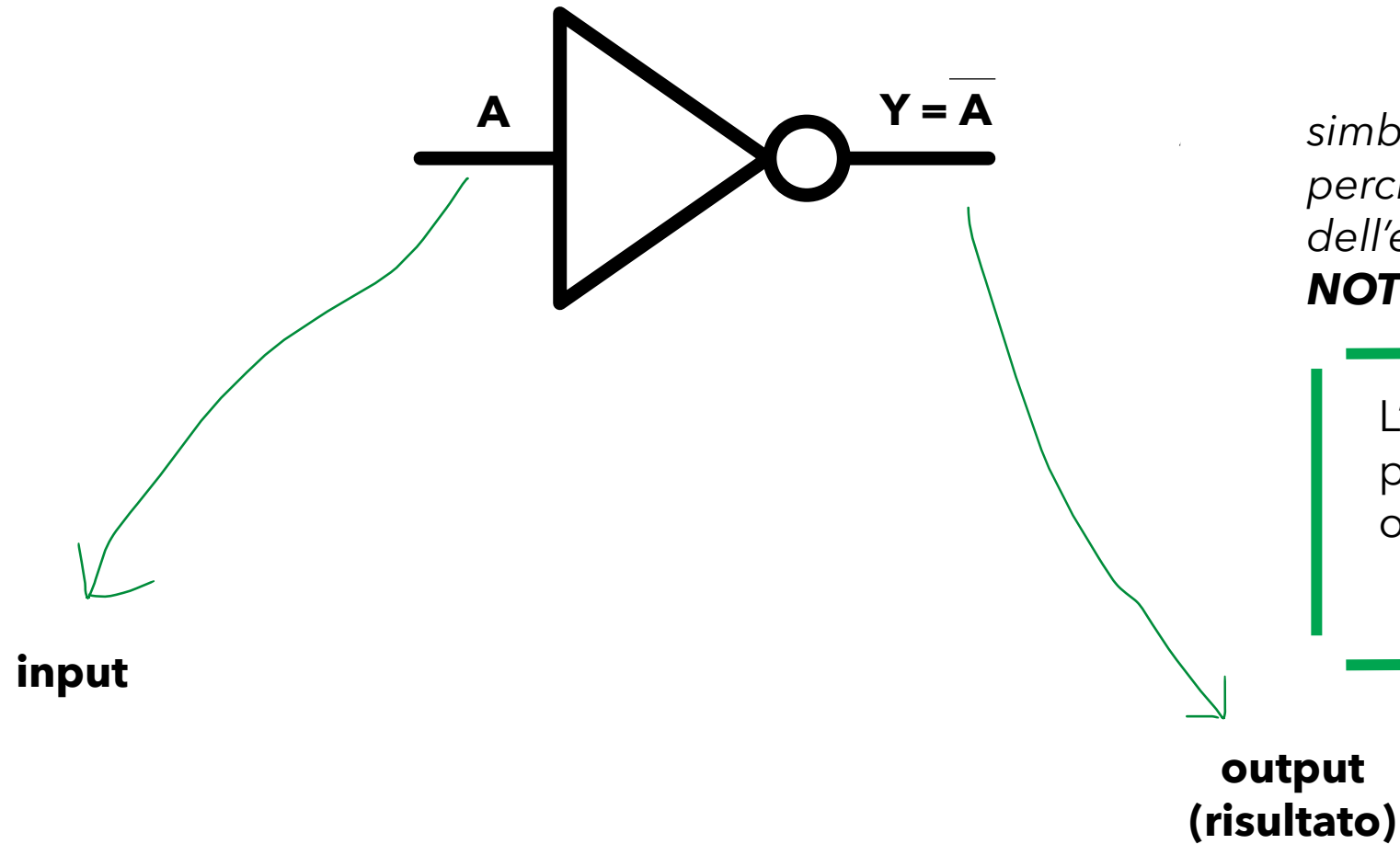
# Un'operazione logica: XOR. Come funziona?

- **XOR** «restituisce» (produce in output) 1 solo se i 2 input sono **diversi**

Tabella di verità (*truth table*) di XOR

input1	input2	output = input1 XOR input2
0	0	0
0	1	1
1	0	1
1	1	0

# Un'operazione logica: NOT (inverter)



*simbolo ANSI (detto «militare» perché adottato nei progetti dell'esercito USA) della porta logica **NOT***

L'operatore **NOT** richiede 1 input e produce 1 output: si dice che è un operatore **unario**.

# Un'operazione logica: NOT. Notazione

**NOT** (input)

$\neg$  input (*notazione logico-matematica*)

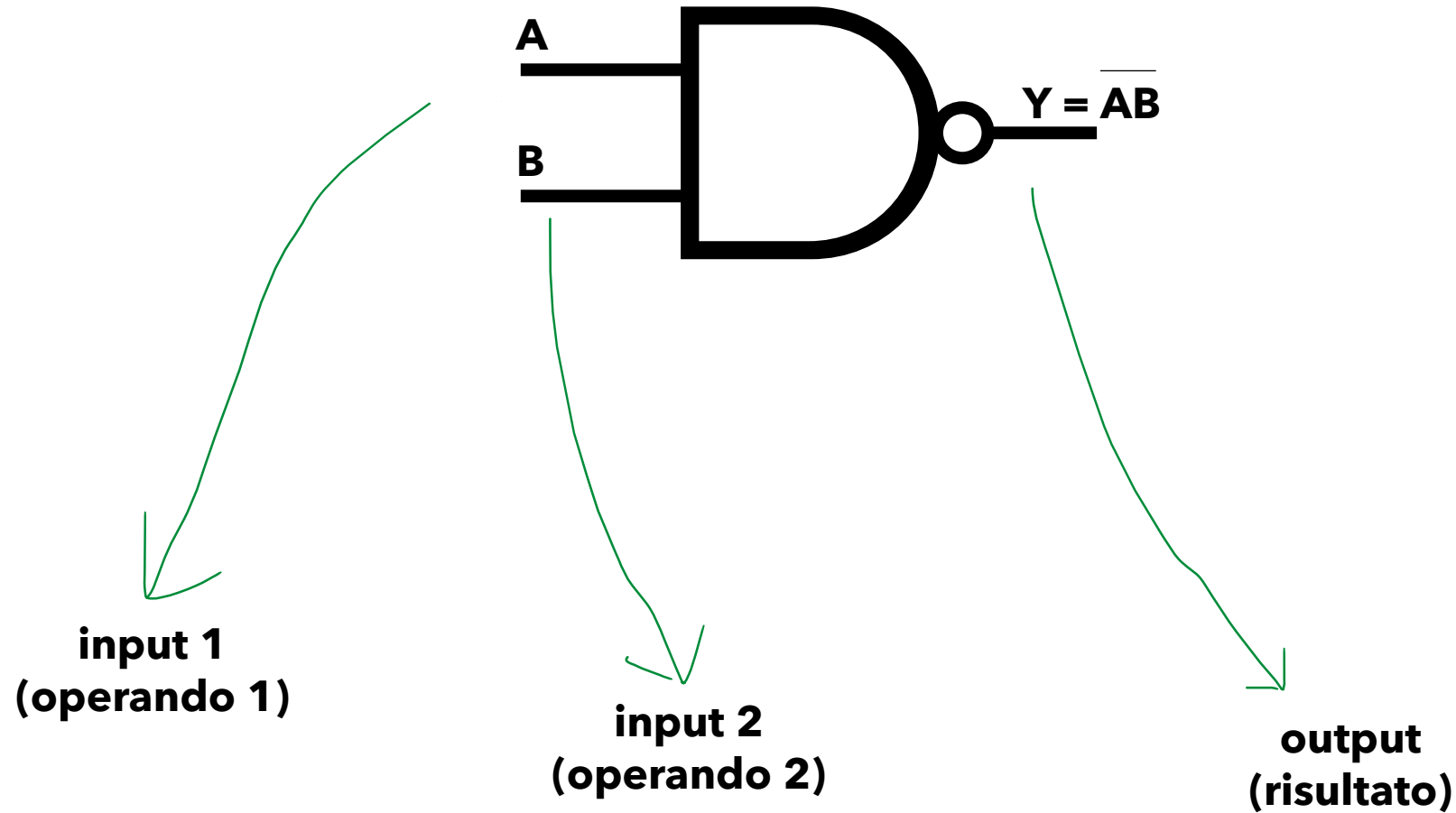
# Un'operazione logica: XOR. Come funziona?

- **NOT** «restituisce» (produce in output) l'inverso dell'input

Tabella di verità (*truth table*) di NOT

input	output = NOT input
0	1
1	0

# Un'operazione logica: NAND



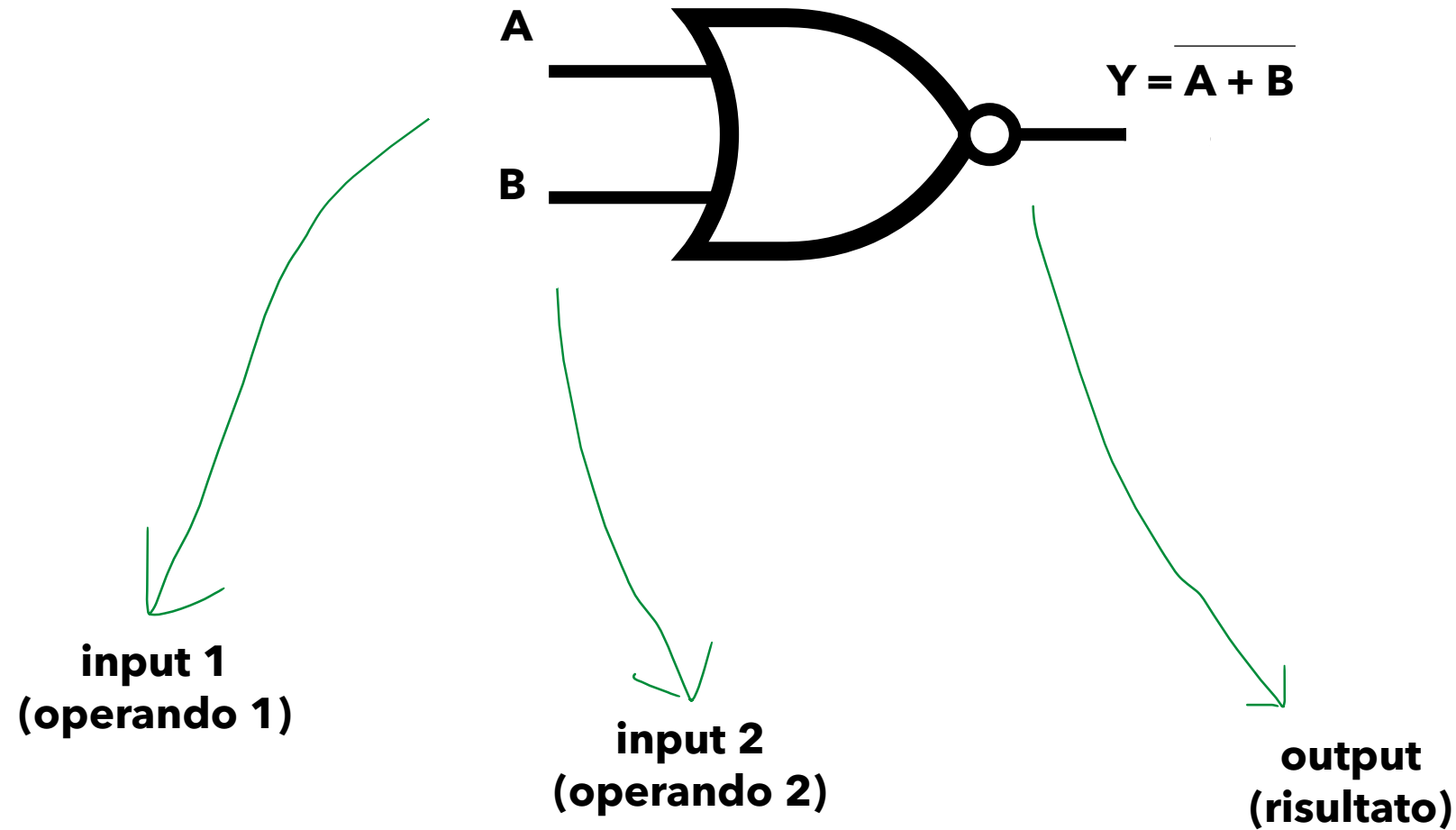
# Un'operazione logica: NAND. Come funziona?

- **NAND** «restituisce» (produce in output) la negazione dell'output di AND: **NOT(AND(input1, input2))**

Tabella di verità (*truth table*) di NAND

input1	input2	output = input1 NAND input2
0	0	1
0	1	1
1	0	1
1	1	0

# Un'operazione logica: NOR



# Un'operazione logica: NOR. Come funziona?

- **NOR** «restituisce» (produce in output) la negazione dell'output di AND: **NOT(OR(input1, input2))**

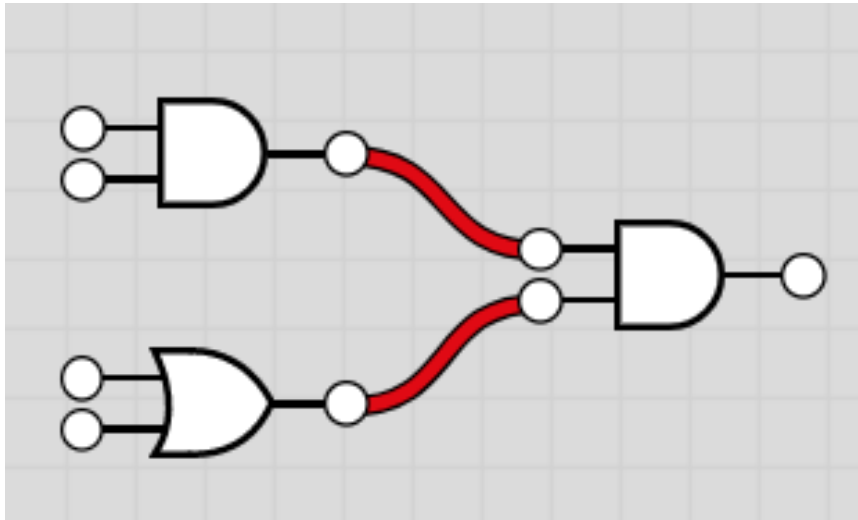
Tabella di verità (*truth table*) di NOR

input1	input2	output = input1 NOR input2
0	0	1
0	1	0
1	0	0
1	1	0



# Come sono realizzate le reti logiche

- Le porte logiche sono contenute in **circuiti integrati**, costituiti prevalentemente da **transistor**
- Una **rete logica** si ottiene collegando più porte logiche
- I transistor sono tutti realizzati su una piastrina di materiale semiconduttore (il silicio) detta **chip**
- Una rete logica si ottiene collegando più porte logiche



una rete logica

# Terminologia e riflessioni matematiche

- Possiamo già iniziare a chiamare **bit** queste «cose» che possono essere solo in 2 possibili stati: true/false, 0/1, on/off
- Se con 1 bit possiamo rappresentare 2 stati, con 2 bit posti uno dopo l'altro quanti ne possiamo rappresentare?

