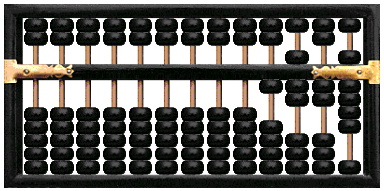
Architettura von Neumann

A. Marchetti Spaccamela

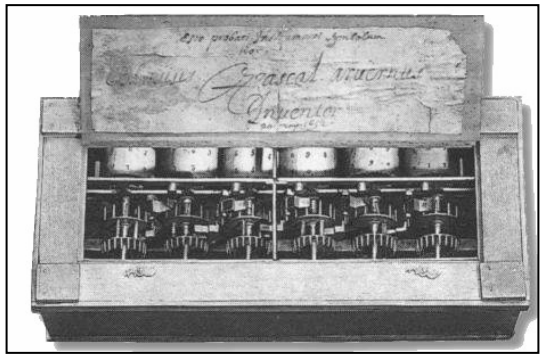
Una breve storia

- Calcolatori elettronici sono recenti
- Le macchine di calcolo sono antiche: L'abaco è stato inventato circa 4000 anni fa.

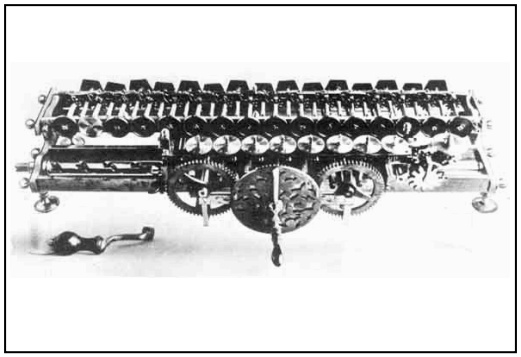


- Nel XVII secolo diverse macchine meccaniche per il calcolo

Ricostruzione della macchina calcolatrice inventata da Wilhelm Schickard − 1623 (Deutsches Museum, Monaco)



La "Pascalina" di Blaise Pascal − 1642 (Musée des Arts et Metiers, Parigi)



La calcolatrice (Stepped Reckoner) di Gottfried Wilhelmvon Leibniz's − 1671 ca.

1800: Babbage

Charles Babbage (1791-1871)

Charles Babbage è una delle figure più affascinanti dell'informatica. Spinto dall'idea di poter costruire una macchina per la produzione di tavole matematiche, Babbage progettò due macchine, la “differential machine” e la “analytic machine”, che ha anticipato molte delle caratteristiche che si trovano nei computer moderni.

Babbage non è riuscito a completare nessuna delle macchine durante la sua vita. Lo Science Museum di Londra è stato in grado di realizzare una versione funzionante della “differential machine” in occasione del 200° anniversario della sua nascita .

Ada Byron, la prima programmatrice

Augusta Ada Byron,
Lady Lovelace (1815–1852)

Ada Byron, figlia del poeta inglese Lord Byron, fu incoraggiata a perseguire i suoi interessi nel campo della scienza e della matematica in un momento in cui pochissime donne potevano studiare questi argomenti.

All'età di 17 anni, Ada incontrò Charles Babbage e rimase affascinata dalle sue macchine. Ada era convinta delle potenzialità della “analytic machine” di Babbage e scrisse ampie note sul suo progetto, insieme a diversi programmi matematici complessi che hanno portato molte persone a caratterizzare la sua figura come quella della prima programmatrice della storia.

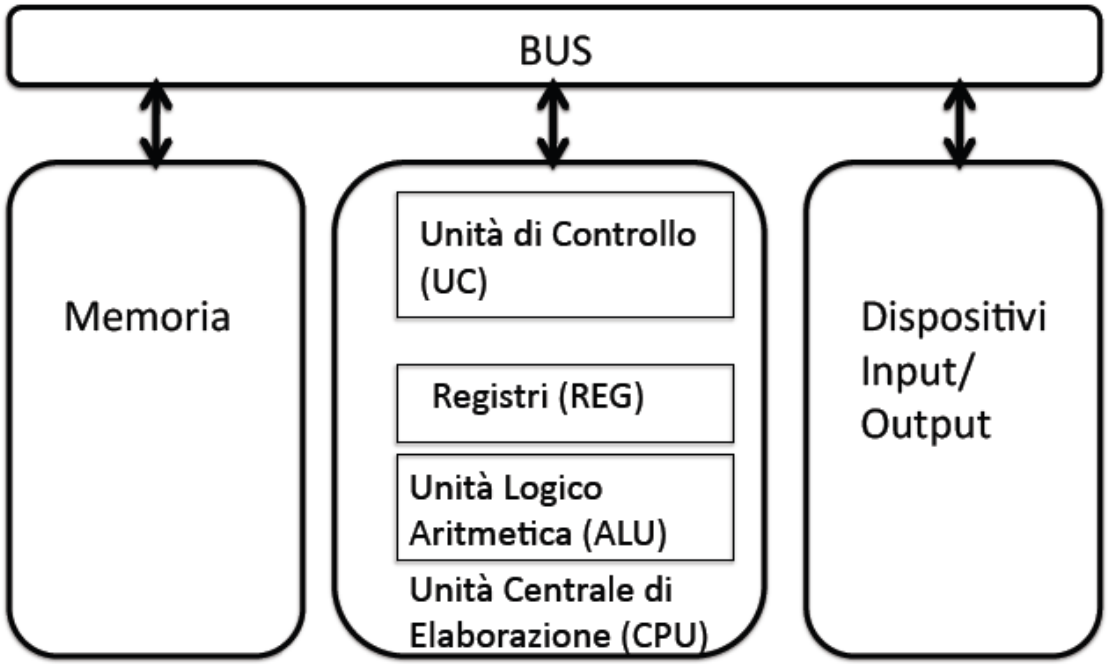
Nel 1980, il linguaggio di programmazione per calcolo parallelo Ada è stato così chiamato in suo onore.

1940: La nascita dell'informatica

Non e' chiaro chi ha inventato i computer moderni: ci sono molti candidati

- Nel 1939, John Atanasoff e Clifford Barry (Iowa State University) costruirono un prototipo per risolvere equazioni lineari (L'Atanasoff-Berry Computer, spesso chiamato *ABC*), e successivamente, nel 1942, una macchina più grande.
- Conrad Zuse in Germania verso la fine degli anni '30, ha realizzato la macchina programmabile Z1, poi evoluta nelle macchine Z2 e Z3.
- Il Colossus è una macchina elettronica programmabile realizzata nel Regno Unito (1943-44) per decifrare i messaggi codificati dell'esercito nazista. Al suo progetto ha contribuito anche Alan Turing (a capo del gruppo crittografico inglese durante la seconda guerra mondiale)
- Il primo computer su larga scala costruito è l'*Electronic Numerical Integrator and Computer* (ENIAC), completato nel 1946 sotto la direzione di J. Presper Eckert e John Mauchly (University of Pennsylvania).
- Altri importanti contributi durante i primi anni comprendono il concetto di programmazione memorizzata, generalmente attribuita a John von Neumann, e l'uso di circuiti di commutazione per implementare aritmetica binaria proposto da Claude Shannon.

Architettura CPU: modello di von Neumann



Modello: una rappresentazione concettuale (spesso una semplificazione) del mondo reale o di una sua parte, o di un manufatto capace di spiegarne il funzionamento.

Nota: la descrizione originale dell'architettura di von Neumann non prevedeva un bus ma collegamenti diretti fra le diverse componenti.

Il modello di von Neumann

Il Modello di Von Neumann comprende

1. L'**Unità di controllo** si occupa di controllare tutte le operazioni del calcolatore, interpretare le istruzioni prelevate dalla memoria e inviare alle altre unità i segnali per l'esecuzione delle operazioni
2. L'**Unità aritmetico-logica**, detta **ALU** (Arithmetic & Logic Unit), permette di effettuare operazioni aritmetiche di base (somme, sottrazion, ecc.) e di prendere decisioni

*Queste due unità sono spesso integrate in una **CPU**, Central Processing Unit – Unità di Elaborazione Centrale* (che contiene anche una serie di registri, simili alla memoria*)*

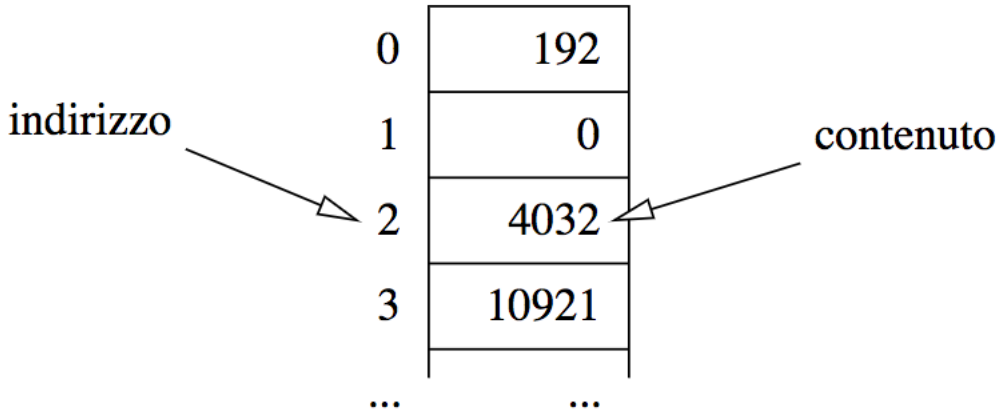
Il modello di von Neumann: cont.

3. La **Memoria** conserva ***le istruzioni e i dati*** da elaborare e i risultati ottenuti dalle elaborazioni*;
4. L'**Unità di ingresso** (**Input**) immette le informazioni nel calcolatore per farle elaborare;
5. L'**Unità di uscita** (**Output**) riceve le informazioni dalla memoria del calcolatore per renderle pronte all'uso; *le unità di ingresso e uscita sono anche dette* ***periferiche***
6. Il **Bus**, vero e proprio canale di comunicazione che consente ai dati di transitare fra diversi componenti del calcolatore.

** Il modello alternativo chiamato Berkley prevede spazi di memoria diversi per le istruzioni ed i dati.*

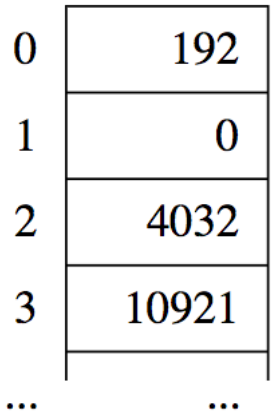
Memoria

- una serie di celle (locazioni di memoria)
- ogni cella può contenere un numero (contenuto)
- le celle sono numerate (indirizzo di una locazione)



Memoria

Lettura da memoria



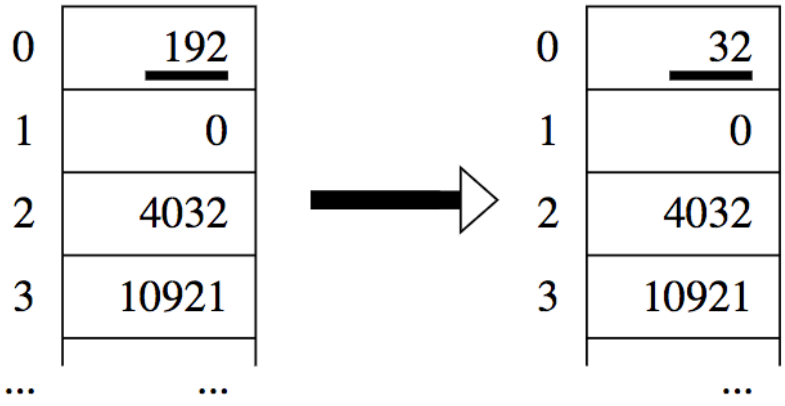
Se la CPU vuole leggere il contenuto della locazione di indirizzo 2:

- La CPU manda alla memoria:
  - l'indirizzo 2
  - l'indicazione che vuole leggere
- la memoria risponde con il valore 4032

Scrittura in memoria

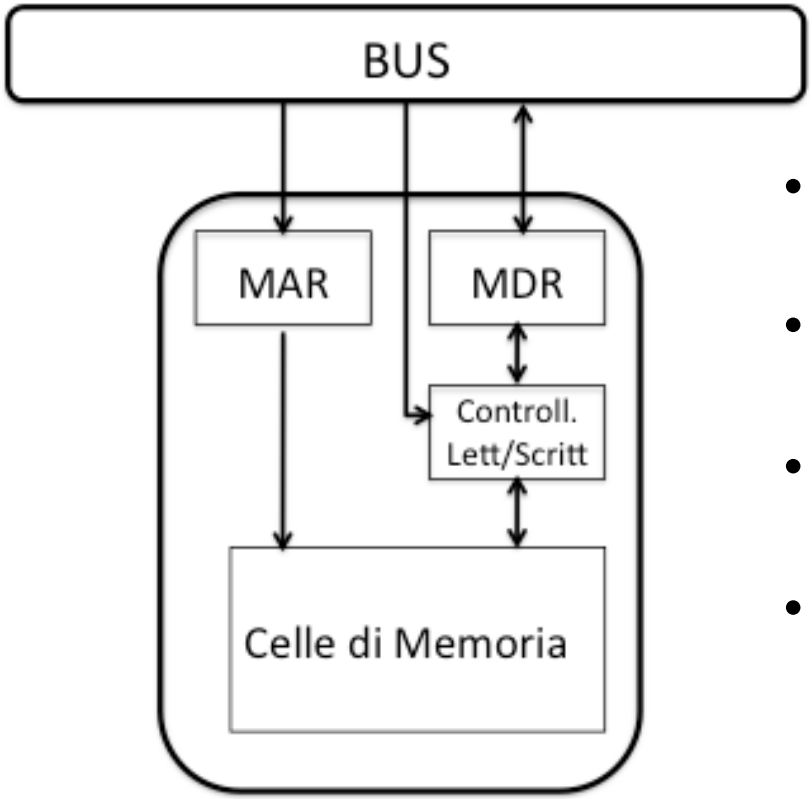
se la CPU vuole scrivere 32 nella locazione di indirizzo 0:

1. La CPU manda alla memoria
   - l'indirizzo 0
   - il valore 32
   - l'indicazione che si tratta di una scrittura



la memoria non manda indietro nessun valore

Sottosistema memoria



- insieme di celle
- MAR = registro per l'indirizzo
- MDR = registro per il dato
- Controll. Lett./Scritt.: selettore che decide se il contenuto della cella indirizzata dal MAR va alla CPU (se la CPU ha chiesto una lettura) o se deve essere sovrascritto con il contenuto dell'MDR (se la CPU ha chiesto una scrittura)

Nota: un registro è come una singola memoria

Registri della CPU

La CPU contiene diversi registri

- Registro Istruzione (IR): memorizza l'istruzione da eseguire
- Contatore di Programma (PC): memorizza l'indirizzo in memoria della prossima istruzione da eseguire
- Registri Dati (in numero variabile): contengono dati utilizzati per i calcoli

I registri della CPU sono aggiornati dall'esecuzione dell'istruzione

Il Programma

- nell'architettura di Von Neumann, si trova in memoria
- è rappresentato da una sequenza di numeri
- vengono letti uno per volta

Esecuzione del programma (esempio):

la sequenza `1921 – 992 – 10063 – 56` rappresenta un programma di quattro istruzioni

l'unità centrale legge `1921` e la esegue

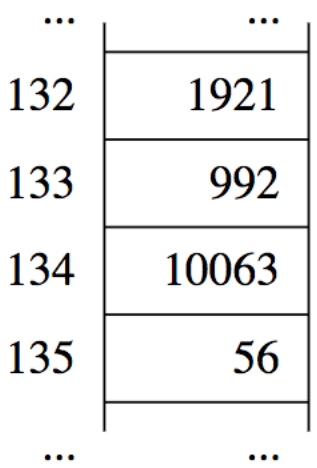
poi legge `992` e la esegue

poi legge `10063` e la esegue

poi legge `56` e la esegue

PC = l'indirizzo (132, 133, 134,...)

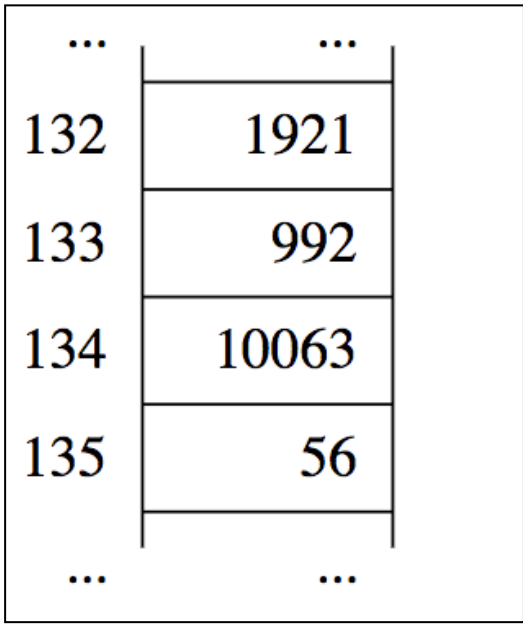
IR = l'istruzione (1921, 992, 10063, ...)



Esecuzione del programma

sequenza di passi:

```
PC=132 IR=...
lettura da memoria della locazione 132
incremento PC
PC=133 IR=1921
controllo esegue l'istruzione 1921
lettura da memoria della locazione 133
incremento PC
PC=134 IR=992
controllo esegue l'istruzione 992
...
```



Nota: se l'istruzione da eseguire è una istruzione di "salto", il PC viene aggiornato all'indirizzo di memoria che contiene l'istruzione a cui saltare

Esecuzione di un'istruzione

1. Caricamento di un'istruzione (*Fetch*): con l'informazione presente nel PC si reperisce la prossima istruzione da eseguire che viene memorizzata in IR (dal punto di vista della memoria è una normale lettura)

2. L'unità di controllo decodifica l'istruzione (*Decode*): si determina il tipo di operazione e i dati coinvolti

3. Esecuzione dell'istruzione (*Exec*): a seconda dell'operazione richiesta si usa la memoria, i registri l'unità di calcolo, l'ingresso/uscita… Inoltre si aggiorna il PC per eseguire la prossima istruzione

4. Se l'istruzione è `HALT` il programma termina altrimenti si inizia l'esecuzione di una nuova istruzione

Caricamento di un’istruzione (fetch)

Passi eseguiti sotto il comando dell'Unità di Controllo:

1. Copiare il contenuto del PC nel MAR (in MAR va l'indirizzo della prossima istruzione da eseguire)
2. Impostare la linea di controllo per comunicare alla memoria la richiesta di lettura
3. Aspettare il reperimento dell'istruzione da parte della memoria
4. Copiare il contenuto di MDR nell’IR

il passo 3 è necessario: non esiste un sistema di memoria che fornisce i dati richiesti in tempo zero

Tipi di istruzioni

- aritmetiche e logiche
  - es. somma di due registri, risultato in un terzo
- lettura e scrittura da memoria:
  - mem → reg
  - reg → mem
- input/output:
  - dispositivo → reg
  - reg → dispositivo
- controllo di esecuzione
  - permette di implementare condizioni, cicli, ecc.

Istruzioni di controllo

di norma, dopo una istruzione si esegue la successiva

es. dopo l'istruzione alla locazione 133 si esegue quella alla locazione 134

esistono solo due tipi di istruzioni per alterare quest'ordine:

- salti incondizionati
- salti condizionati

entrambi scrivono un valore in PC

Istruzioni di salto

Assumiamo che l'istruzione all'indirizzo 133 sia:

1.un salto incondizionato, ad esempio

salta a indirizzo 153

si esegue mettendo 153 in PC. Al prossimo fetch, l'istruzione che verrà eseguita non è quella che si trova all'indirizzo 134 ma quella che si trova all'indirizzo 153.

2.un salto condizionato: simile al salto incondizionato, ma il salto si esegue solo in certi casi:

- si salta solo se un registro contiene zero
- si salta solo se un registro contiene un valore negativo
- ….

Il linguaggio della macchina

- La CPU è in grado di eseguire solo istruzioni molto semplici.
- Queste sono espresse in linguaggio macchina (o codice binario), dipendente dall'elaboratore.
- Il linguaggio assemblativo consente di esprimere istruzioni in forma simbolica (comprensibile da un umano). Queste sono in corrispondenza 1-1 con le istruzioni in linguaggio macchina.
- Nel seguito consideriamo un esempio di processore semplice (non esistente nella realtà), con quattro registri (R1, R2, R3, R4) ed un insieme semplificato di istruzioni, espresse in forma simbolica.

Nota: Linguaggi come Python, Java, C, C++,.. sono linguaggi di alto livello: si basano su costrutti non elementari, comprensibili da un umano. Ogni istruzione corrisponde in genere a molte istruzioni in linguaggio macchina. In larga misura questi linguaggi sono indipendenti dallo specifico elaboratore.

Un semplice linguaggio

HALT	(arresto del programma)
LOAD X IND	(Leggi la cella indir. IND e trasferisci il dato reg. X)
LOADC X VALORE	(Inizializza il reg X con VALORE, specificato)
STORE X IND	(Scrivi il valore reg X nella cella di indirizzo IND)
ADD X,Y,Z	(Esegui X=Y+Z, dove X, Y e Z sono registri)
SUB X,Y,Z	(Esegui X=Y-Z, dove X, Y e Z sono registri)
MULT X,Y,Z	(Esegui X=Y*Z, dove X, Y e Z sono registri)
DIVIDE X,Y,Z	(Esegui X=Y/Z, dove X, Y e Z sono registri)
SQRT X,Y	(Esegui X= $\sqrt{}$ Y, dove X e Y sono registri)
READ IND	(Leggi un valore in ingresso e ponilo cella di indir. IND)
WRITE IND	(Scrivi in uscita il valore della cella di memoria IND)
JUMP SALTO	(Salta all'istruzione memorizzata in SALTO)
JUMPIFZERO X, SALTO	(Salta a SALTO se reg. X è zero)
JUMPIFNEG X,SALTO	(Salta a SALTO se il registro è negativo)

Programma per la somma tre numeri

```
READ  1000
READ  1001
READ  1002
LOAD  R1 1000
LOAD  R2 1001
LOAD  R3 1002
ADD   R4,R1,R2
ADD   R4,R3,R4
STORE R4 1003
WRITE 1003
HALT
```

Soluzione equazione di secondo grado

Cerchiamo le soluzioni dell'equazione $ax^2+bx+c=0$.

Ricordiamo che $$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Assumiamo che i valori a,b,c siano memorizzati nelle celle `100`, `101`, `102` e che il programma sia memorizzato a partire dalla cella `1000`

```
LOAD R1 100
LOAD R2 101
LOAD R3 102
LOADC R4 4
MULT R2 R2 R2
MULT R4 R1 R4
MULT R4 R3 R4
SUB R4 R2 R4
JUMPIFNEG R4 1023
```

(pone la costante 4 nel registro `R4`)

(ora `R2` contiene b^2)

(ora `R4` contiene $4a$)

(ora `R4` contiene $4ac$)

(ora `R4` contiene $b^2 - 4ac$)

(se $b^2 - 4ac$ è negativo non esistono soluzioni reali e il programma termina senza stampare valori, l'istruzione `1023` è l'istruzione `HALT`)

… continua: si calcolano le soluzioni reali

```
SQRT R4 R4
LOADC R2 0
LOAD R3 101
SUB R2 R2 R3
LOADC R3 2
MULT R1 R3 R1
ADD R3 R2 R4
DIV R3 R3 R1
STORE R3 103
SUB R3 R2 R4
DIV R3 R3 R1
STORE R3 104
WRITE 103
WRITE 104
HALT
```

(ora R4 contiene $\sqrt{(b^2-4ac)}$)
(carica in R3 il valore b dalla cella 101)
(ora R2 contiene $-b$)
(pone in R1 il valore $2a$)
(ora R3 contiene il valore $-b+\sqrt{(b^2-4ac)}$)
(ora R3 contiene il valore $(-b+\sqrt{(b^2-4ac)})/2a$)
(memorizza in 103 la prima soluzione trovata)
(ora R3 contiene il valore $-b-\sqrt{(b^2-4ac)}$)
(ora R3 contiene il valore $(-b-\sqrt{(b^2-4ac)})/2a$)
(memorizza in 104 la sec. soluzione trovata)
(stampa il primo risultato)
(stampa il secondo risultato)

Programma Python equazione II grado

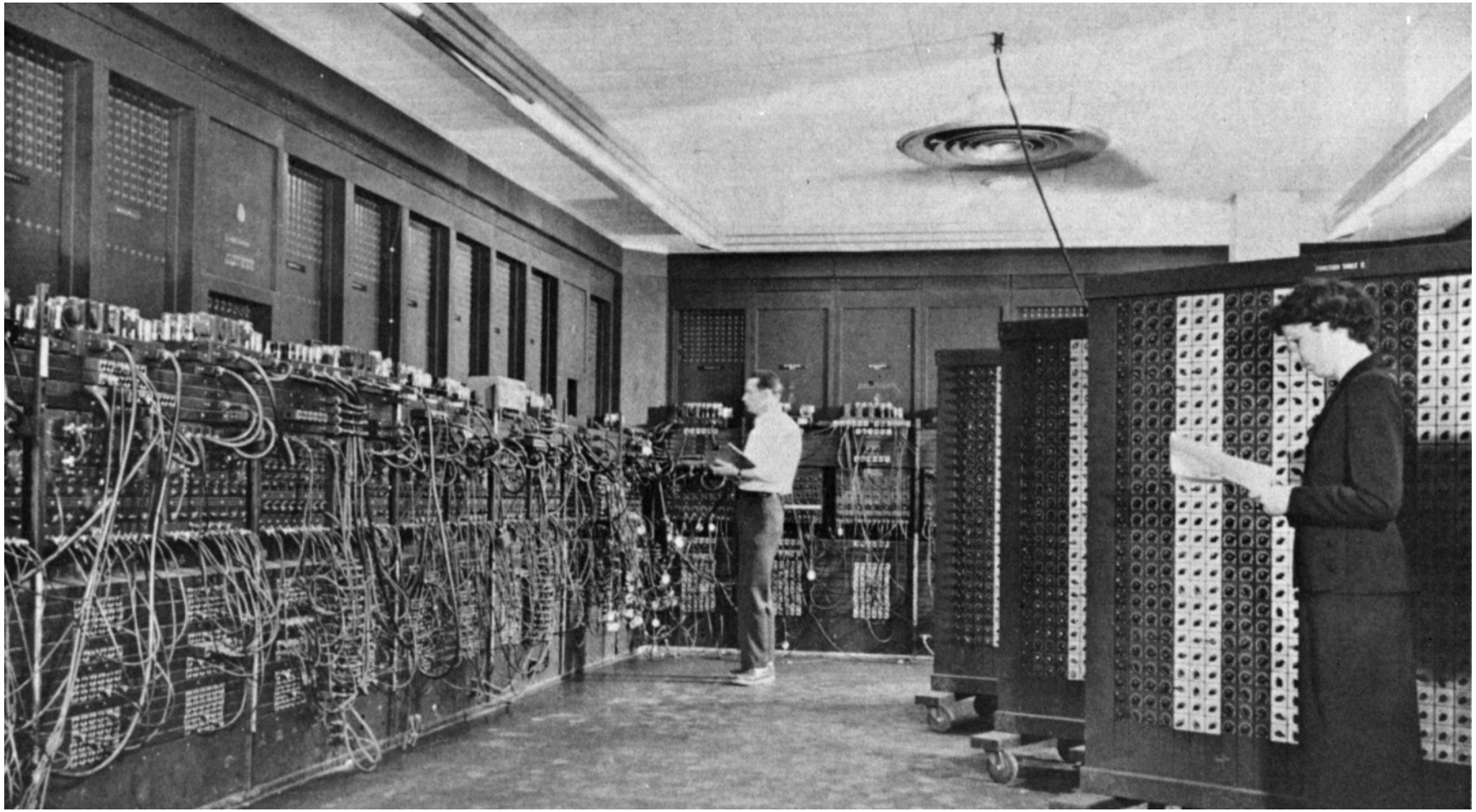
```
import math
a=int(input('immetti il coefficiente di grado due'))
b=int(input('immetti il coefficiente di grado uno'))
c=int(input('immetti il termine noto'))
d=b*b-4*a*c
if d>=0:
  s1=(-b+math.sqrt(d))/(2*a)
  s2=(-b-math.sqrt(d))/(2*a)
  print(s1, s2)
else:
  print('soluzioni complesse')
```

A parte casi molto particolari, si evita di scrivere programmi direttamente in linguaggio macchina

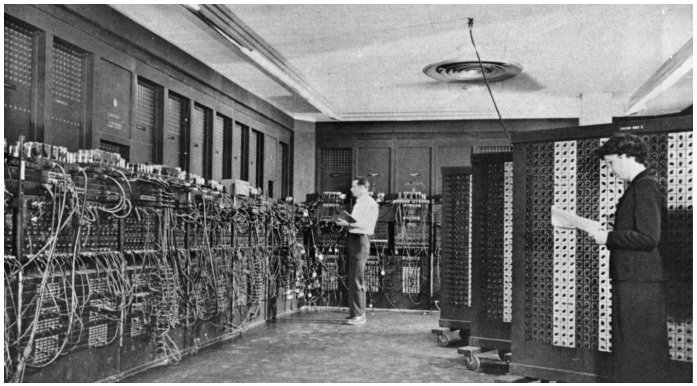
Calcolatori e mente umana

- Perché i calcolatori sono molto più veloci degli essere umani nei calcoli anche se eseguono operazioni molto semplici?
- Velocità : 1 operazione elementare richiede
  - Neuroni: circa 5 millisecondi (200 oper. al sec.)
  - Computer: circa 1 nanosecondo (1000 milioni al sec.)
- Memoria:difficile da valutare negli umani
  - Essere umani (difficile valutazione: circa 400 milioni di bit)
  - Computer (1 tera byte : 8 milioni di milioni di bit)

Un progresso incredibile: ENIAC 1946

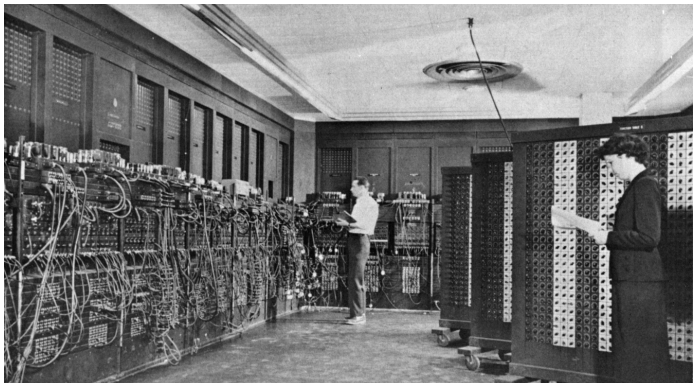


Un progresso incredibile: ENIAC 1946



- Velocità : 5000 operazioni al sec. (5 K Hz)
- Memoria: una decina di numeri (versione originale)
- Costo: circa 5 Milioni di euro (di oggi)
- Peso e dimensioni : 27 tonnellate e richiedeva uno spazio di circa 160 metri quadri
- Consumo : 150 K Watt
- Affidabilità: all'inizio un disastro, poi migliore: si rompeva una volta ogni due giorni …

Un progresso incredibile



70 anni dopo

Notebook

- Velocità: 3-4 GHz
- Memoria: disco 1 tera
- Costo 500-1000 euro
- Peso: meno di 1 kilo
- Consumo : 50 Watt (CPU)
- Affidabilità

1 K (kilo) = mille

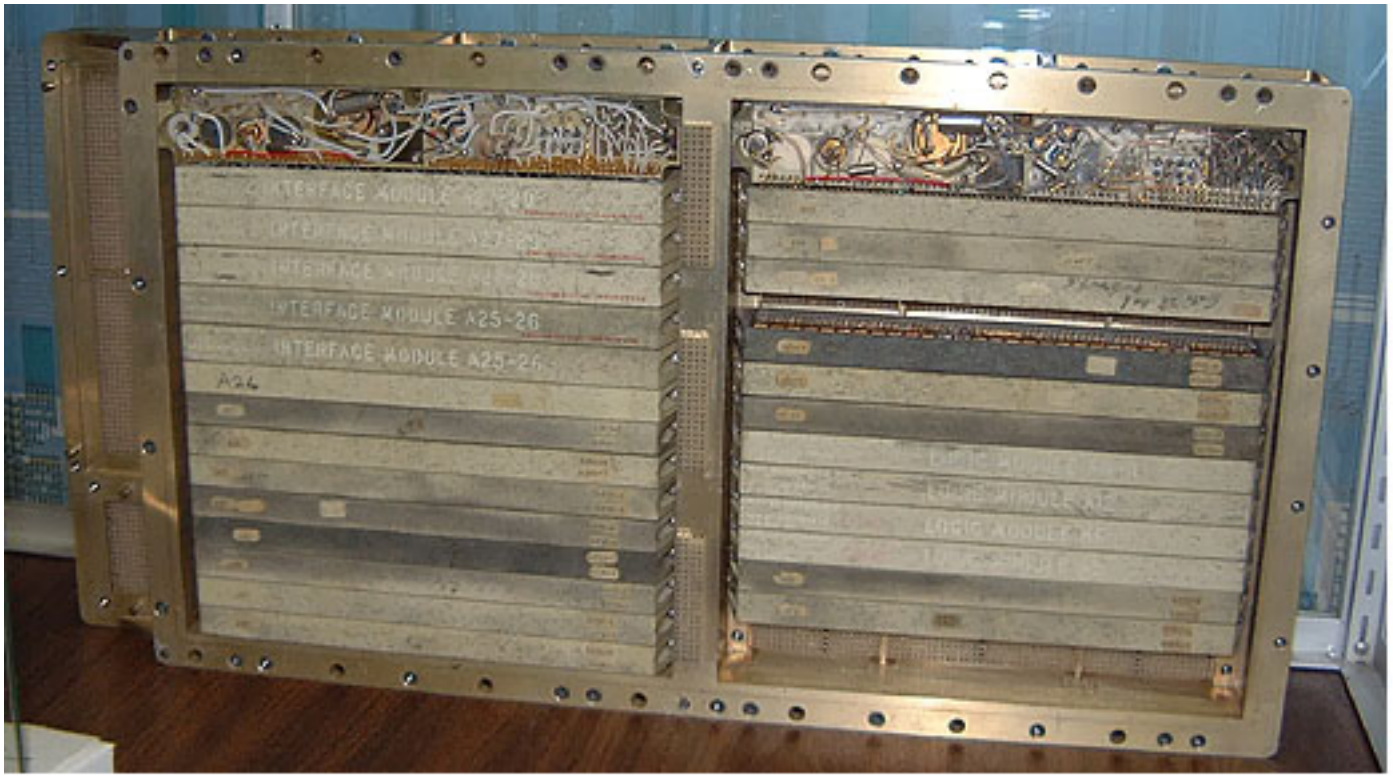
1 M (Mega) = 1 milione

1G (Giga) = 1 miliardo

1 T (Tera) = 1000 miliardi

1 P (Peta) = 1 milione di miliardi

Computer missione Apollo, 1969



1 Cubo di circa 30 cm

Domanda: Perche' il computer di guida – molto pesante -era a bordo?

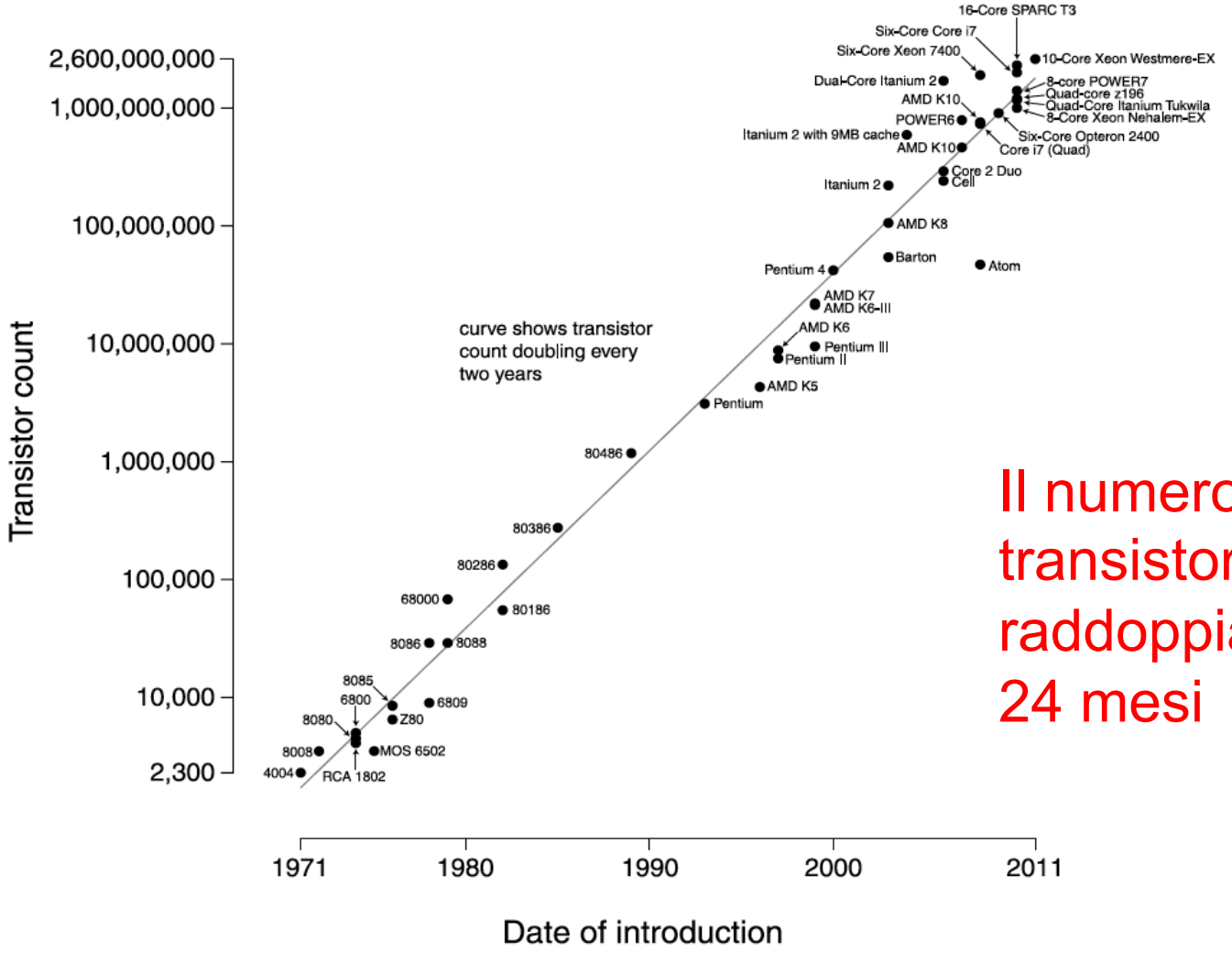
Missione Apollo computer: prestazioni

- Apollo Computer: 30720 bits di memoria (RAM)
- Oggi macchine con 4 Giga Bytes – GB (RAM)
  - 1 Gigabyte = 1024 Megabytes,
    1 Megabyte = 1024 Kilobytes,
    1 Kilobyte = 1024 Bytes,
    1 Byte = 8 bits

4 GB = (4 x 1024 x 1024 x 1024 x 8) bits
> 32000 milioni di bits

Se il computer di guida di Apollo power era di 30 cm allora una macchina con 4 GB avrebbe avuto la dimensione di 10 chilometri!

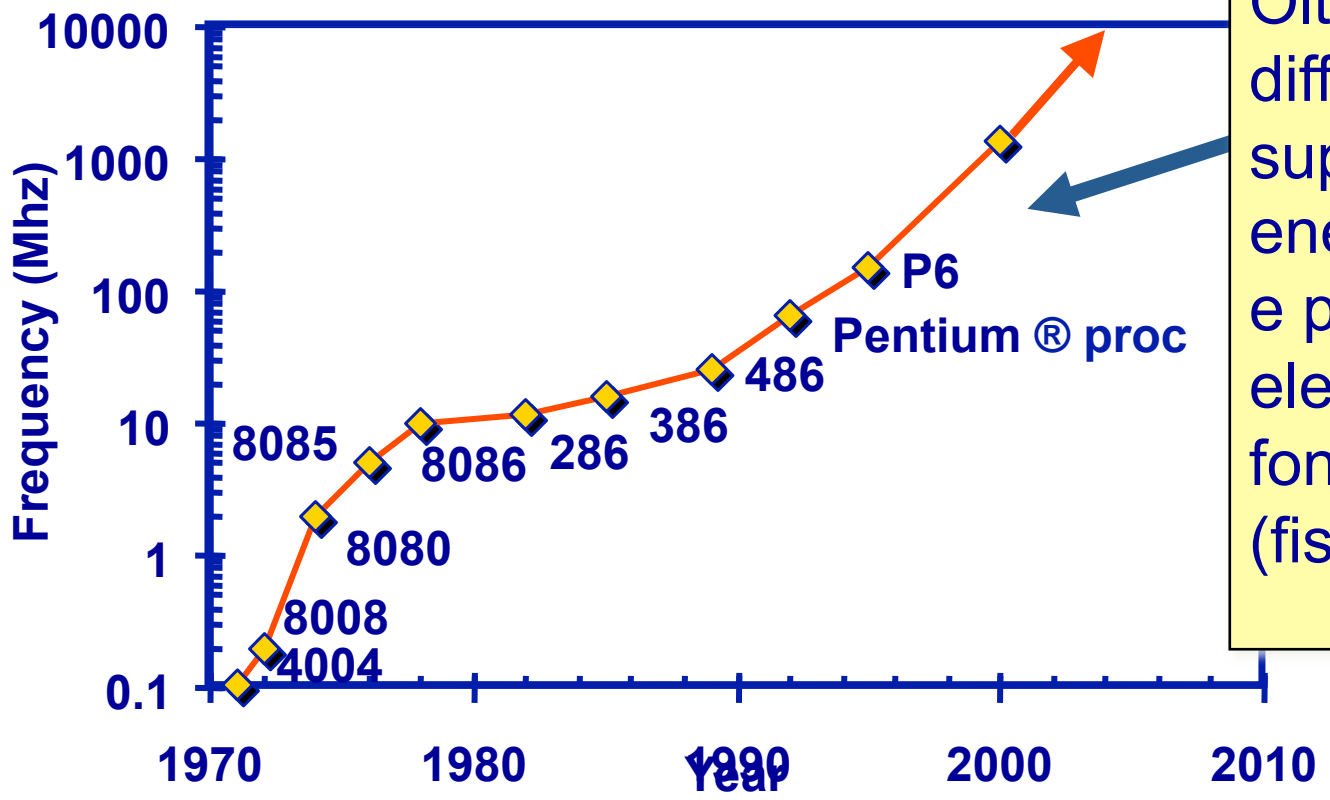
Legge di Moore



Il numero di transistor per mm raddoppia ogni 24 mesi

Frequenza di clock (orologio)

La frequenza di Clock indica quante operazioni elementari si eseguono
1 Giga Hertz = 1 miliardo al secondo



La frequenza di clock raddoppia ogni due anni circa
Oltre 3GhZ ci sono difficoltà da superare:
energia temperat.
e problemi elettrici fondamentali
(fisica quantistica)

Courtesy, Intel

Legge di Moore

Miglioramenti tecnologici simili si sono avuti anche in altri aspetti

- Immagini (costo pixel)
- Memoria (costo e dimensioni)
- Velocità e costo delle comunicazioni (Internet)

Seconda legge di Moore

- I costi di progettazione e realizzazione delle fabbriche aumentano (es. costo progetto e realizzazione fabbrica per nuovo chip costa qualche miliardo di euro)
- Il numero di produttori si riduce (infatti oggi ci sono pochi produttori di chip)

Quanto possiamo progredire?

Siamo vicini a limiti fisici attuale tecnologia

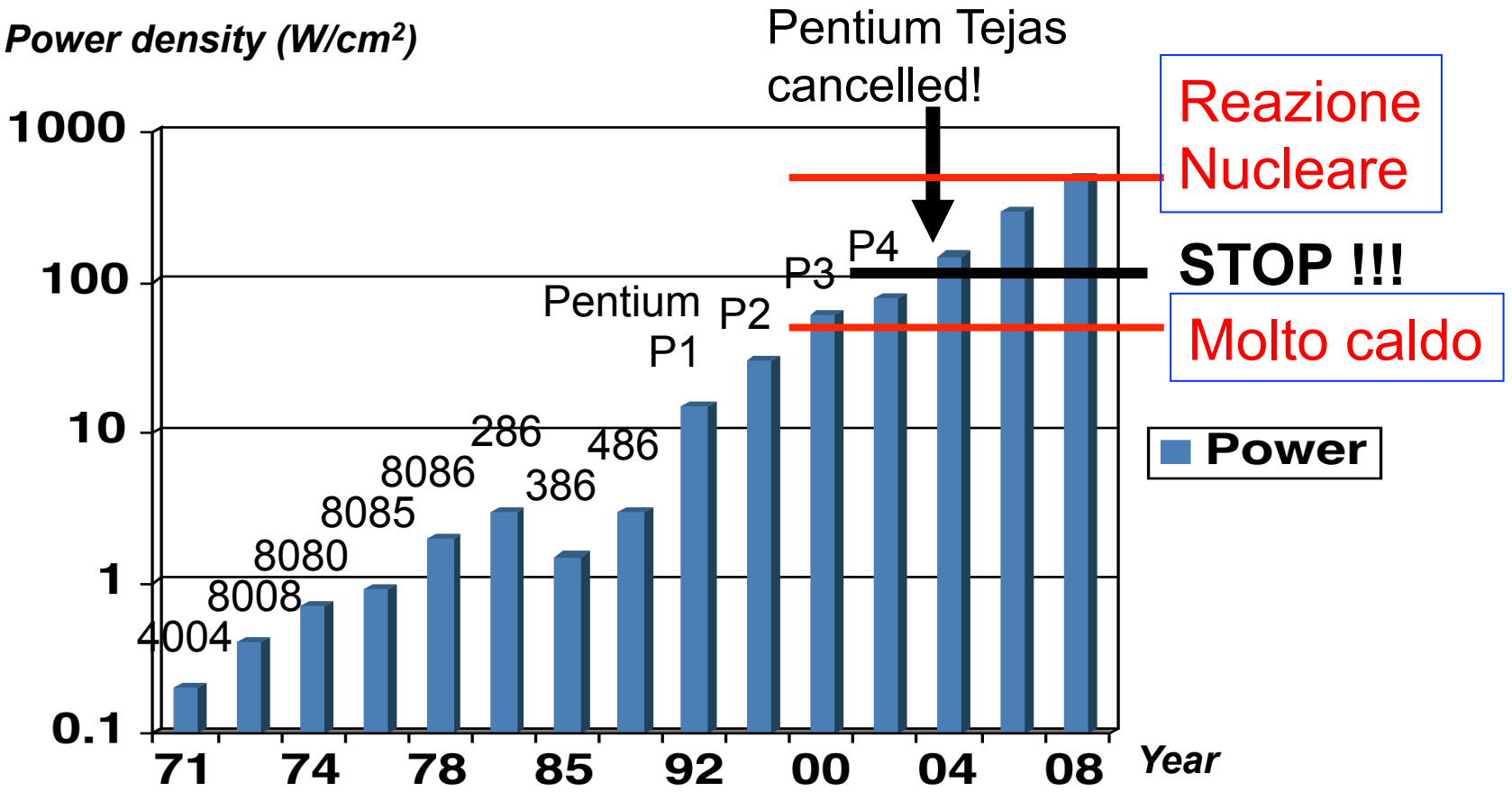
- Dimensione di un gate di un transistor (elemento base dei circuiti elettronici) 2014: 14 nm
- 2018-19: in commercio sono disponibili dispositivi mobili (TSMC) e processori desktop (AMD) con tecnologia a 7nm
- 1 nanometro = 1 milionesimo di un millimetro
- La dimensione di un atomo circa 0,1 nanometri; una molecola di silicio circa 0,2 nanometri;
- Piccole dimensioni → aumento difetti fabbricazione →aumento dei costi (pezzi scartati)
- Circuiti oggi bidimensionali; uso chip tridimensionali (più compatti): ci sono problemi tra cui il surriscaldamento

Quanto possiamo progredire?

Siamo vicini a limiti fisici attuale tecnologia

- Velocità della luce (non superabile secondo Einstein) = 300.000 Km al secondo
  - La luce in un nanosecondo percorre 30 cm
  - Se la frequenza di clock è 3 GHz, la luce in un ciclo di clock percorre 10 cm
  - I chip hanno dimensioni di 3-4 centimetri di lato
  - Non possiamo con questa tecnologia superare i 30-40 GHz
  - Inoltre maggiore è la frequenza maggiore è il **calore** sviluppato

Il calore sviluppato è molto alto



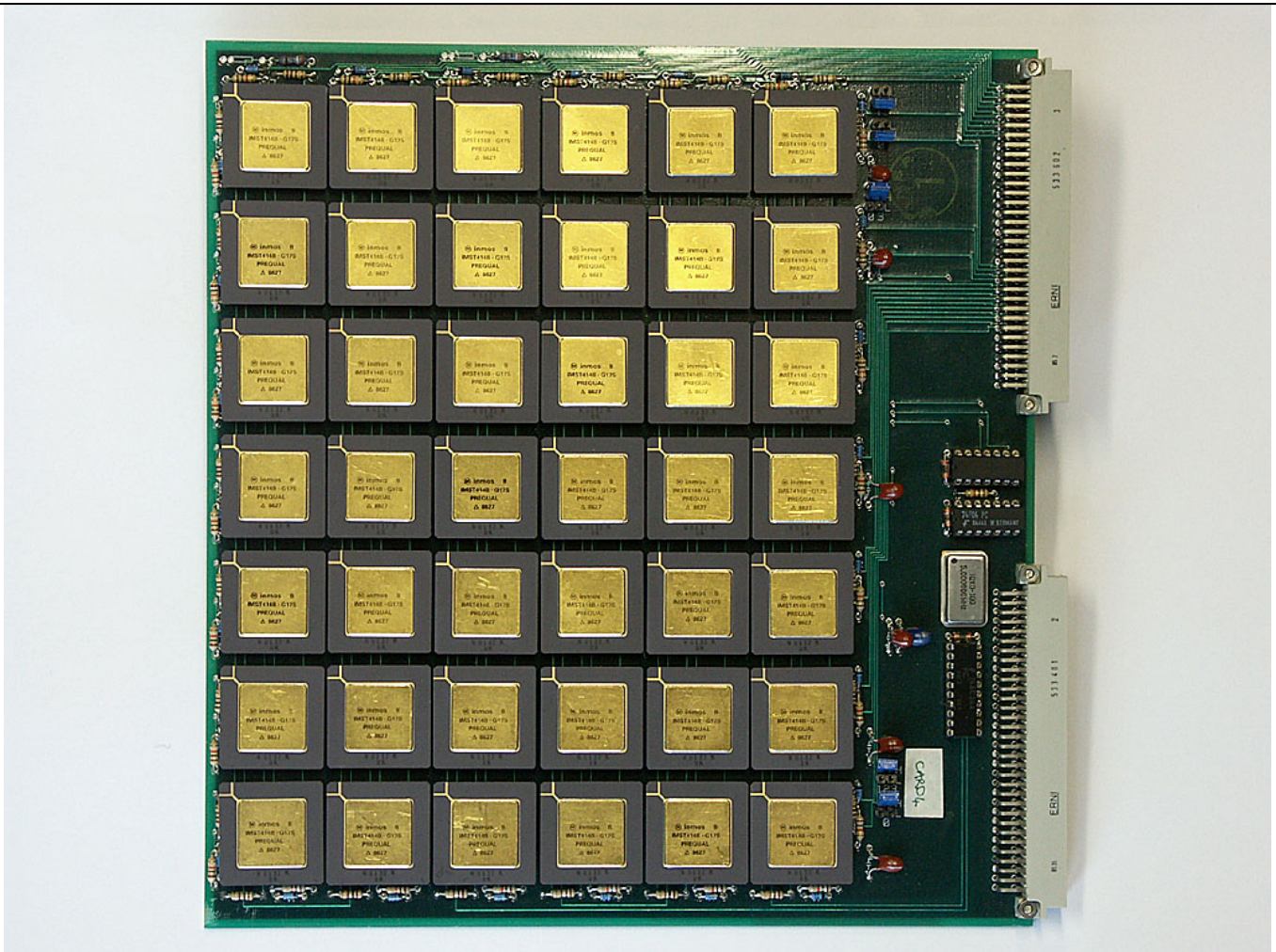
Mantenere la legge di Moore

- Stiamo raggiungendo i limiti fisici in termini di velocità e dimensione dei dispositivi (e calore sviluppato)
- Spesso annunci di nuove tecnologie sono realizzati con ritardo; gli annunci non sono sempre veritieri in assoluto
- Per migliorare è necessario usare le possibilità di avere circuiti più densi in modo diverso

Molte CPU (core) su un unico chip!

Cosa è questo?

1979 - David May's B0042 board - 42 Transputers

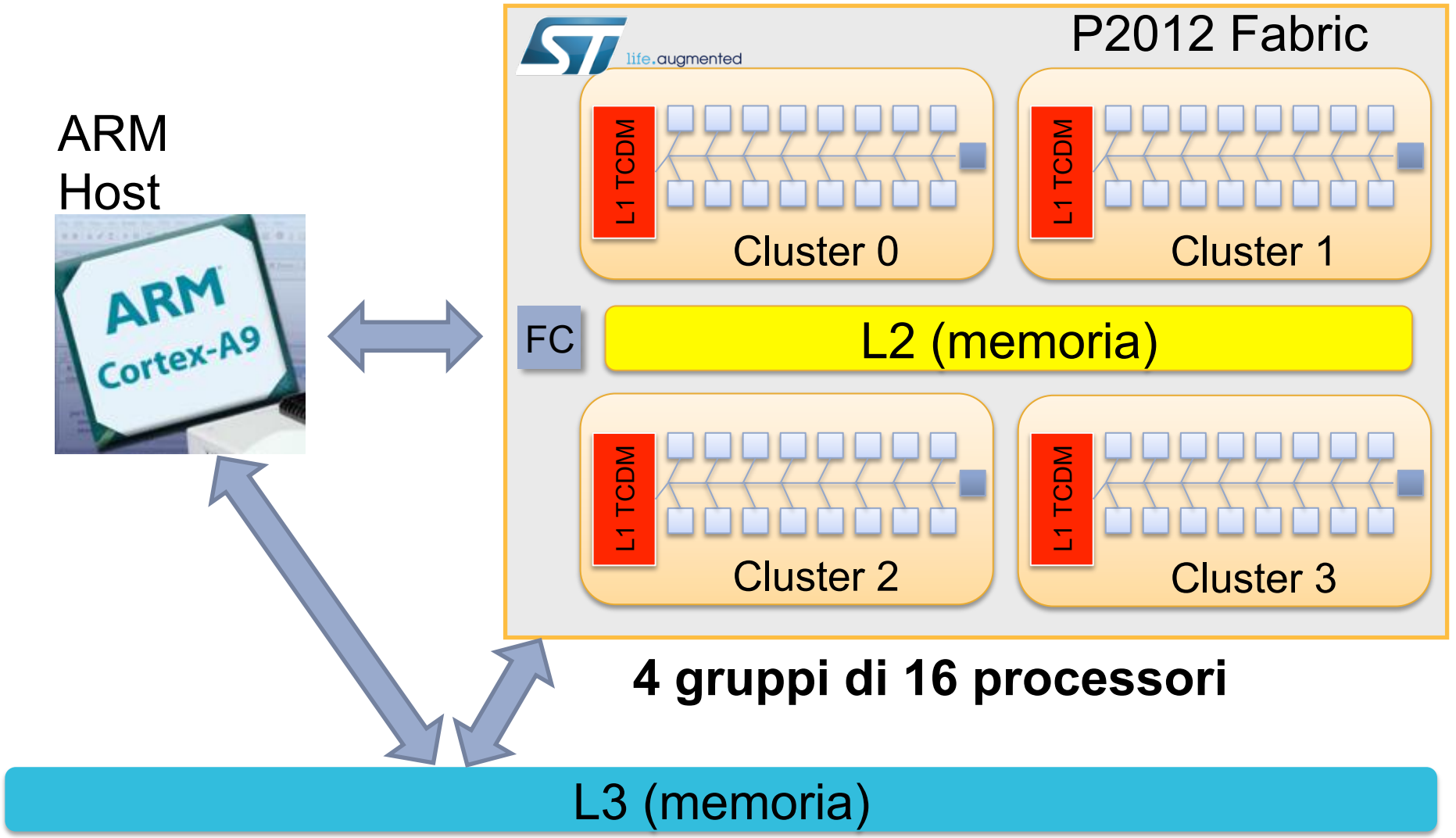


L’invasione multicore

- Intel’s Core 2, Itanium, Xeon: 2, 4 cores
- AMD’s Opteron, Athlon 64 X2, Phenom: 2, 4 cores
- IBM’s POWER7: 8 cores
- IBM-Toshiba-Sony Cell processor: 8 cores (PSX3)
- Sun’s Niagara UltraSPARC: 8 cores
- Microsoft’s Xenon: 3 cores (Xbox 360)
- Tilera’s TILE64: 64-core
- Others (network processors, DSP, GPU,…)

P2012 SoC: 64 core su un chip

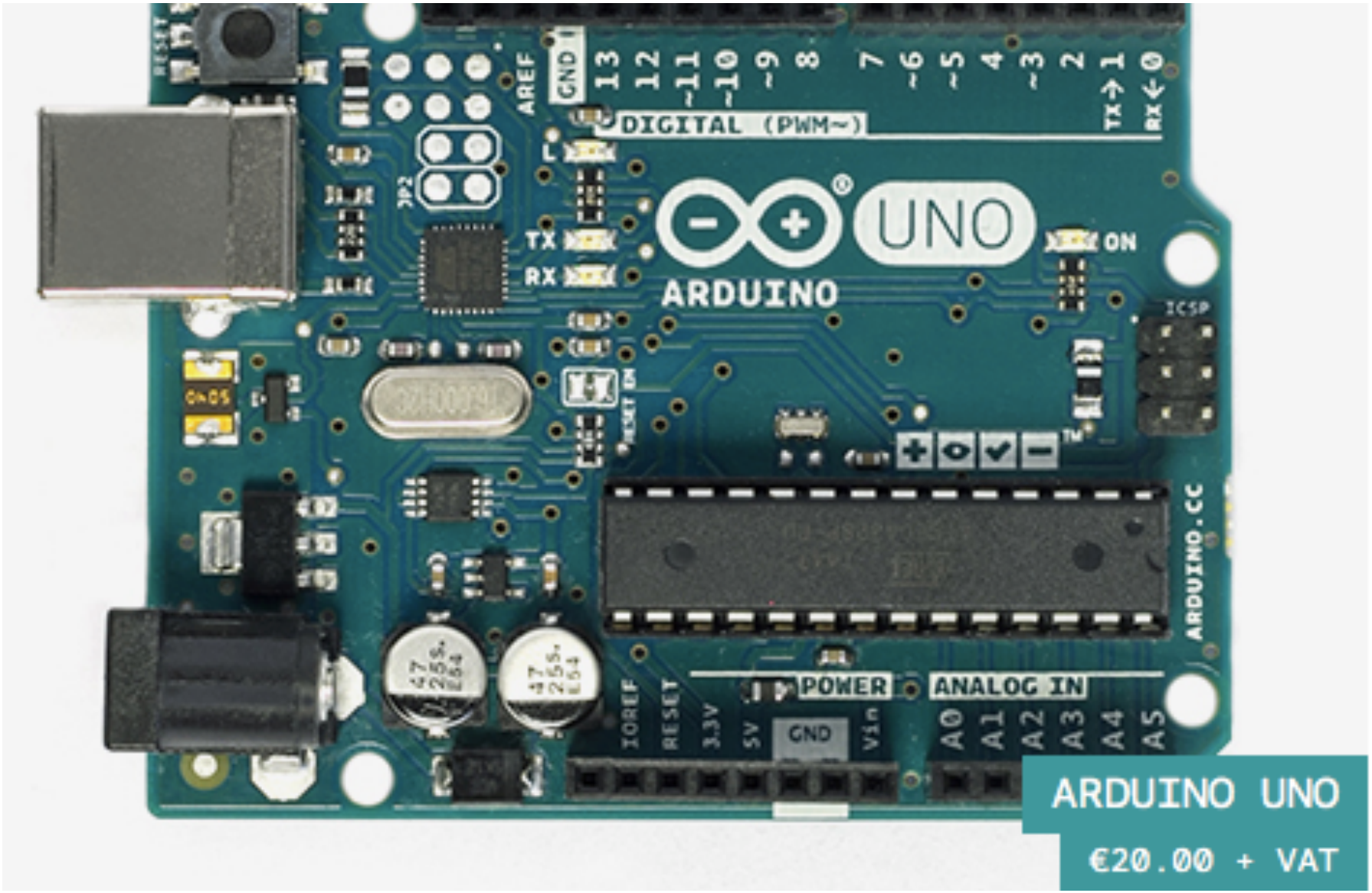
Progetto architettura ST Microelectronics 2009-2012



Sistemi embedded

- Automobili: controllo sicurezza, condizioni di guida, ecc.
- Elettrodomestici
- Domotica
- Internet of things: l’internet degli oggetti

Arduino


DIGITAL (PWM~)
UNO
ARDUINO
ON
TX
RX
ICSP
POWER
ANALOG IN
GND
ARDUINO.CC
ARDUINO UNO
€20.00 + VAT

Arduino

Scheda elettronica di **piccole dimensioni e basso costo** con un microcontrollore e circuiteria di contorno, utile per creare rapidamente prototipi e per scopi hobbistici e didattici.

Nota: Il nome della scheda deriva da quello di un bar di Ivrea dove si incontravano gli inventori di Arduino

Arduino

Con Arduino si possono realizzare rapidamente piccoli dispositivi come controllori di luci, di velocità per motori, sensori di luce, temperatura e umidità e molti altri progetti che utilizzano sensori comunicando con altri dispositivi.

È fornito di un semplice ambiente di sviluppo integrato per la programmazione. Tutto il software a corredo è libero, e gli schemi circuitali sono distribuiti come hardware libero.