CALCOLATORI
Introduzione

Giovanni Iacca
giovanni.iacca@unitn.it

Luigi Palopoli
luigi.palopoli@unitn.it

UNIVERSITÀ DEGLI STUDI DI TRENTO
Dipartimento di Ingegneria
e Scienza dell'Informazione

Descrizione del corso

- Il corso (che si chiama anche «Architettura degli elaboratori») si compone essenzialmente di lezioni teoriche
- In aggiunta, avremo qualche esercitazione sugli aspetti più pratici del corso (ad es. aritmetica dei calcolatori e Assembly)
- In totale il corso è coperto da 48 ore di didattica frontale (12 settimane x 2 lezioni x 2h)

Struttura del corso

- Prima parte: introduzione, aritmetica dei calcolatori, cenni su reti logiche (5 lez.)
- Seconda parte: linguaggio Assembly (10 lez.)
- Terza parte: elementi di architettura dei calcolatori (8 lez.) + esempio d'esame (1 lez.)

Lez.	Argomento
1	Introduzione
2-4	Aritmetica dei calcolatori
5	Reti logiche
6	Assembly
7-10	Assembly RISC-V
11	Assembly INTEL
12	Assembly ARM
13-14	Esercizi Assembly
15	Toolchain
16-17	CPU
18-19	Pipeline
20-21	Gerarchia di memoria
22	Input/Output
23	Esempi di esame
24	Esercizi

Modalità di esame

- E’ prevista una prova scritta intermedia
- L’esame si compone di una prova scritta (svolto in forma elettronica attraverso la piattaforma Moodle) e consiste in larga parte di quesiti a risposta multipla (diversi per ciascuno studente!)
- E’ previsto inoltre un orale di verifica dello scritto e approfondimento della teoria
- Durante il corso vedremo periodicamente alcuni esercizi e quesiti di test che costituiscono un esempio di quello che incontrerete all’esame

Altre info

- Pagina Moodle del corso (per annunci, slide e altro materiale)

- Libro di riferimento:
  David A. Patterson, John L. Hennessy,
  «Struttura e progetto dei calcolatori
  Progettare con RISC-V»,
  Zanichelli **(ed. 2019 o successive)**

- Ricevimento (via Zoom/Meet) su appuntamento
- Contatti
  - Giovanni Iacca
    giovanni.iacca@unitn.it
  - Elia Cunegatti, Chiara Camilla Migliore Rambaldi, Stefano Genetti (assistenti)
    elia.cunegatti@unitn.it
    cc.rambaldimigliore@unitn.it
    stefano.genetti@unitn.it

Dove studiare cosa

Indicativamente:

- Lez. 1: Sez. 1.1-1.5, 1.7, 1.8, cenni 1.6 + Slide/materiale su Moodle
- Lez. 2-4: Sez. 2.4, 3.1-3.3, 3.5 (no HW e RISC-V) + Slide /materiale su Moodle
- Lez. 5: Sez. 2.6 + Slide/materiale su Moodle
- Lez. 6-10: Sez. 2.1-2.11 + Slide/materiale su Moodle
- Lez. 11: Sez. 2.17 + Slide/materiale su Moodle
- Lez. 12: Slide/materiale su Moodle
- Lez. 13-15: Slide/materiale su Moodle
- Lez. 16: Sez. 2.12-2.15 + Slide/materiale su Moodle
- Lez. 17-20: Cap. 4 (in parte) + Slide/materiale su Moodle
- Lez. 20-22: Cap. 5 (in parte) + Slide/materiale su Moodle
- Lez. 23-24: Slide/materiale su Moodle

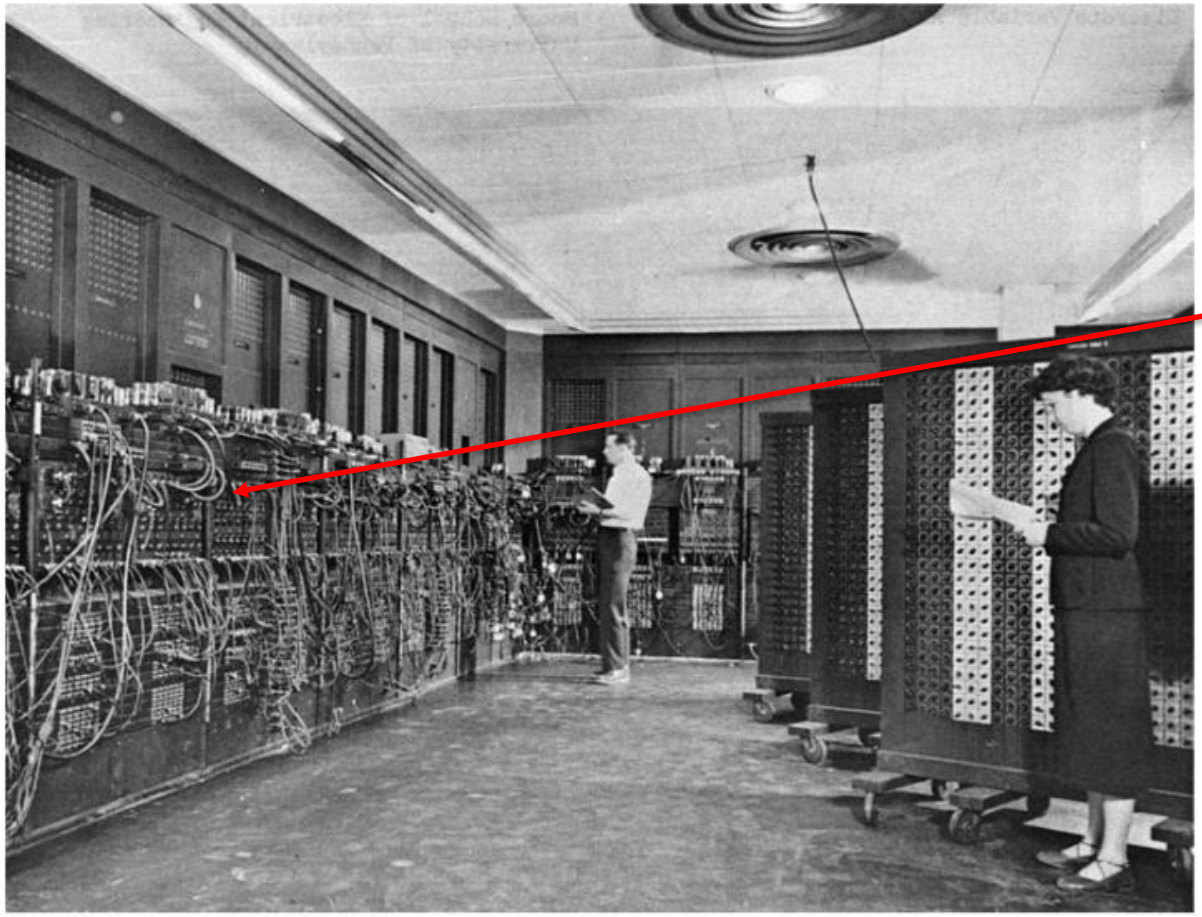
Calcolatori

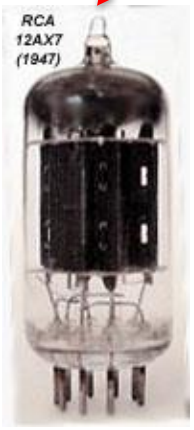
- I calcolatori elettronici sono il prodotto di una tecnologia estremamente vitale
- Produce il 10% del PIL degli Stati Uniti e pervade le nostre vite
- E tutto cominciò da…

ENIAC

- Nel 1943 il Dipartimento della Difesa degli Stati Uniti commissionò una macchina per il calcolo delle traiettorie dei proiettili di artiglieria
- Nel Febbraio 1946 l'Università della Pennsylvania mise in funzione ENIAC (Electronic Numerical Integrator and Computer)
  - Occupava una stanza di 9 x 30 metri
  - Consumava così tanta energia che alla prima accensione generò un blackout

ENIAC





... x 18000!

https://it.wikipedia.org/wiki/ENIAC

https://www.geopop.it/come-sei-matematiche-hanno-cambiato-la-storia-dellinformatica-chi-erano-le-programmatrici-delleniac/

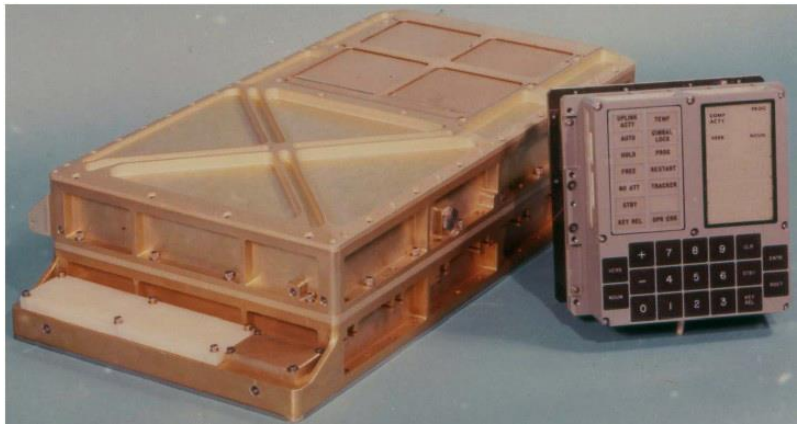
Apollo Guidance Computer (IBM, 1969)

- 61 x 32 x 17 cm, 32 kg
- 2800 circuiti integrati (nota: i primi risalivano al '59!)
- Processore @0.043-2 MHz (frequenze diverse nei vari sottosistemi)
- 152 kByte complessivi ROM/RAM
- Interfaccia DSKY (display&keyboard)
  - tastiera numerica (istruzioni: verbo + nome)
  - piccolo display
  - vari indicatori luminosi

Margaret Hamilton, 1969

Per fare un confronto: iPhone X

- 143.6 x 70.9 x 7.7 mm, 174 g
- 4.3 miliardi di transistor
- Processore @2.39 GHz (A11 Bionic 6-core ARMv8-A)
- 3GB RAM, fino a 64GB di memoria storage
- Interfaccia: touch-screen + video/voice

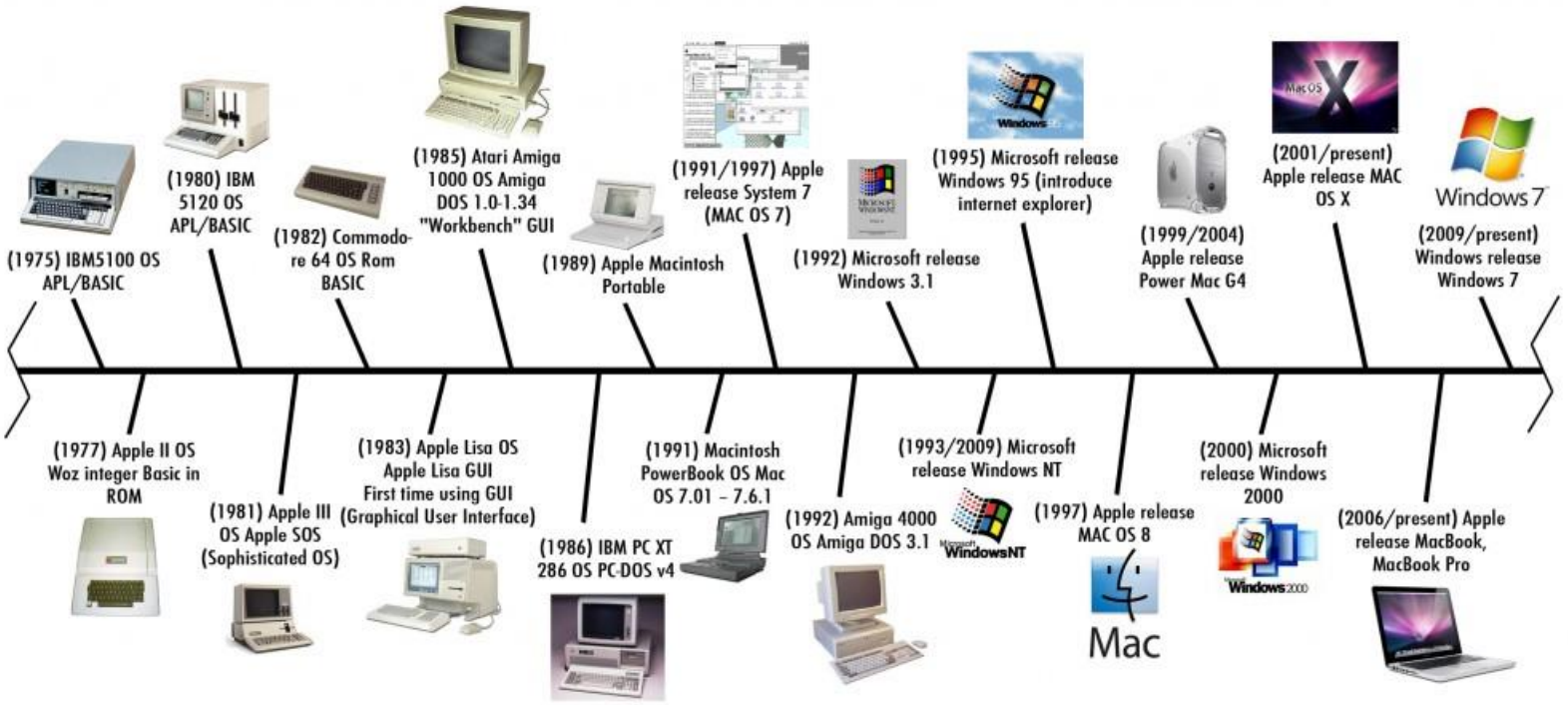


Una profezia pessimistica

"Mentre l'ENIAC è dotato di 18000 valvole e pesa 30 tonnellate, i calcolatori del futuro potranno avere solo 1000 valvole e pesare solo una tonnellata e mezzo",
Popular Mechanics 1949

...e poi: la rivoluzione dei PC

?



...e poi: la rivoluzione dei PC

Bendix G15 (1956)

LGP-30 (1956)

Olivetti Programma 101 (1964)

https://www.youtube.com/watch?v=UWFZLgEiP0M

E oggi?
























La rivoluzione dei calcolatori

- I calcolatori hanno creato la terza rivoluzione della nostra società portandoci nel mondo post industriale
- Solo pochi anni fa le seguenti applicazioni erano considerate fantascienza
  - *Calcolatori negli automobili*
  - *Telefoni cellulari*
  - *Mappatura del genoma umano*
  - *World Wide Web*
  - *Motori di ricerca*
  - *Robot di servizio*
  - *Auto a guida automatica*

Calcolatore o calcolatori?

- I calcolatori che operano nelle applicazioni che abbiamo introdotto condividono la stessa idea di base…
- Ma le soluzioni usate per ciascuna tipologia di applicazione possono essere piuttosto diverse
- Per questo parliamo di vari tipi di calcolatori

Vari tipi di calcolatori

- Calcolatori personali (desktop o laptop)
  - buone prestazione a costo ridotto
  - eseguono software di terze parti (architetture aperte)
- Server
  - Pensati per eseguire grandi carichi di lavoro
    - ✓ poche applicazioni molto complesse (calcolatori scientifici)
    - ✓ tantissime applicazioni molto semplici (web-server)
- Embedded
  - Coprono un vasto spettro di applicazioni (mobile, automotive, avionica, gaming)
  - Le applicazioni sono spesso “dedicate” e operano a stretto contatto con l’hardware
  - Requisiti non funzionali essenziali
    - ✓ consumi
    - ✓ rispetto di vincoli temporali
    - ✓ costo

Perché è importante studiarli?

- Le **prestazioni** del software sono importanti nel decretarne il successo commerciale
- Un programma che viene eseguito più velocemente o che ha minori requisiti hardware ha maggiori probabilità di soddisfare le aspettative del cliente
- Fino a qualche tempo fa le prestazioni erano dominate dalla disponibilità di memoria
- Oggi, questo è un problema solo per alcune applicazioni embedded piuttosto specifiche

Perché è importante studiarli?

- Tuttavia, per scrivere un programma con buone prestazioni un programmatore moderno deve
  - comprendere la gerarchia di memoria
  - fare un uso efficiente del parallelismo (multi-threading, GPU, calcolo distribuito)
- In altre parole, deve conoscere (e comprendere) l'organizzazione del calcolatore

Obiettivi del corso

- Alla fine del corso, sapremo:
  - Quali sono i componenti di base che permettono ad un calcolatore di operare?
  - Come vengono tradotti i programmi in modo che il calcolatore possa eseguirli?
  - Qual è l'interfaccia HW/SW tramite la quale il programmatore può far fare all'HW ciò che richiede?
  - Cosa influenza le prestazioni di un programma e come può un programmatore migliorarle?
  - Cosa può fare il progettista HW per migliorare le prestazioni, e perché oggi si ricorre sempre di più alle architetture multi-core?

Capire le presentazioni

Componente HW/SW	Cosa influenza	Dove è trattato?
Algoritmi	Determina il numero di istruzioni di alto livello e di operazioni di IO	Altri corsi
Linguaggi di programmazione, compilatori e architetture	Determina il numero di istruzioni macchina per ogni istruzione di basso livello	Cap. 2, 3
Processore e sistema di memoria	Determinano quanto velocemente è possibile eseguire ciascuna istruzione	Cap. 4, 5, 6
Sistema operativo, gestione HW e I/O	Determina quanto velocemente possono essere eseguite le istruzioni	Cap. 4, 5, 6

Cfr. tabella p. 7 Patterson-Hennessy

Software di sistema

- Sistema Operativo (SO)
  - gestisce le operazioni di I/O
  - alloca la memoria
  - consente il multitasking
- Compilatore
  - traduce da linguaggio ad alto livello a linguaggio macchina

Linguaggio macchina

- Il componente base di un calcolatore sono le porte logiche, che corrispondono a “interruttori” elettrici
- L'unità base di informazione è il bit: un interruttore vale 1 se acceso, 0 se spento
- Anche un’istruzione di linguaggio macchina deve dunque essere codificata come una stringa (una sequenza) di bit. Ad esempio:

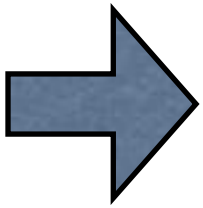
1000110010100000

Programmazione Assembly

- Programmare tramite sequenze di *bit* è estremamente difficile (per usare un eufemismo!)
- Per questo motivo, è stato introdotto un linguaggio mnemonico (chiamato Assembly) che viene tradotto in stringhe di bit da un traduttore (*assembler*)
- Ad esempio:

add A, B

Il programmatore scrive questa istruzione (somma A a B)

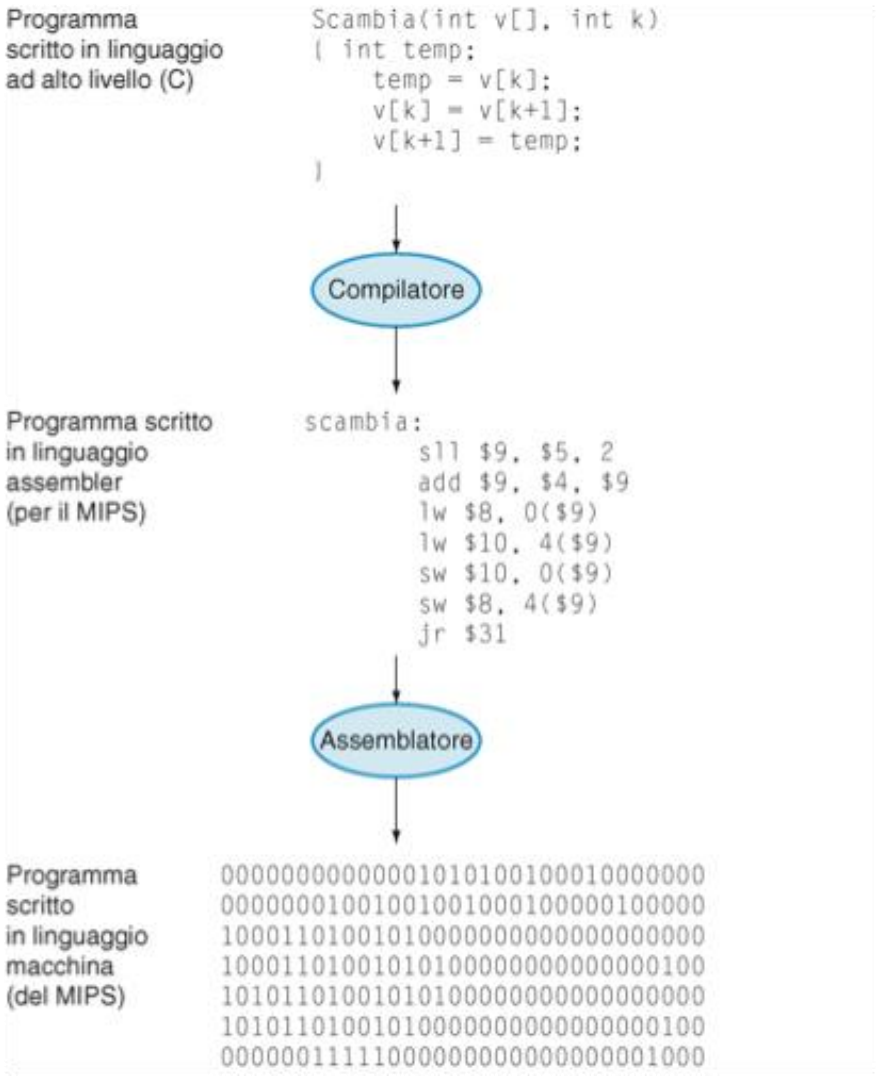


00000000000010001000100001000000

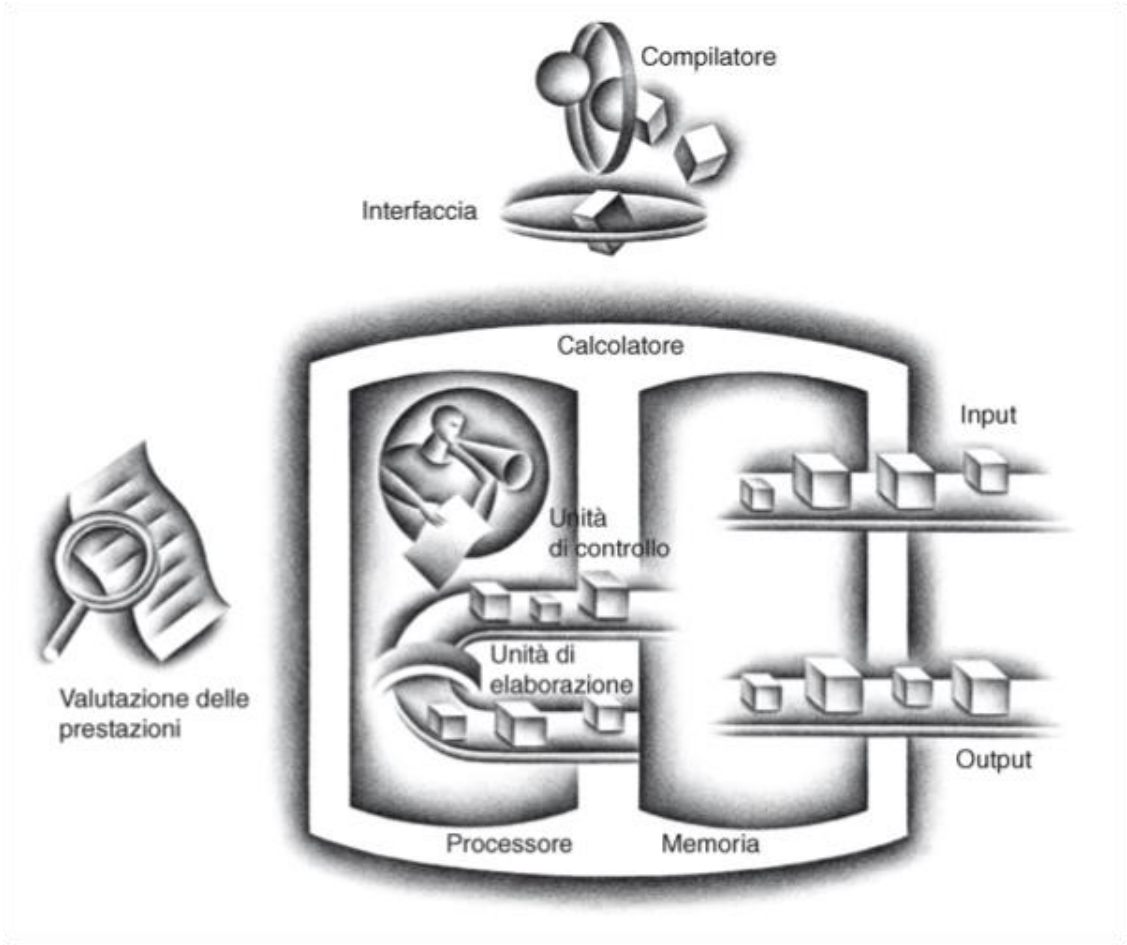
L'assemblatore traduce l'istruzione in una sequenza di bit

Il flusso completo

Molti compilatori saltano questa fase



Componenti del Calcolatore



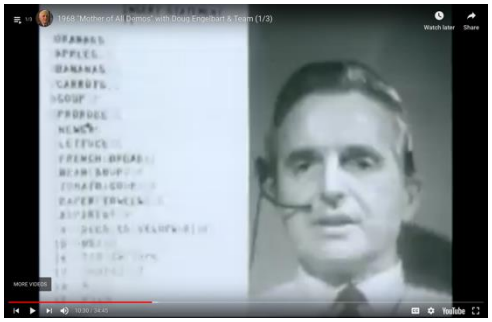
Le elaborazioni dei dati sono effettuate dal Processore (diviso in una parte operativa e una di controllo)

I dati vengono memorizzati nelle unità di memoria

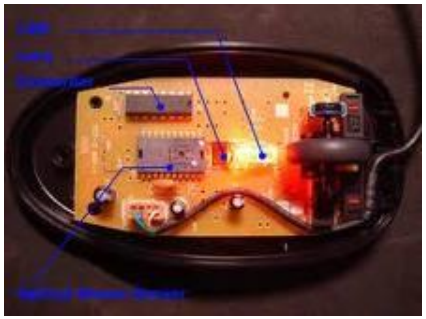
I dispositivi di input (tastiera, mouse, ecc.) e quelli di output (video, stampanti, ecc.) permettono di scambiare informazioni con l'esterno

Esempi di IO: il mouse

- Il mouse è stato inventato nel 1967 da Doug Engelbart nei famosi laboratori della Xerox
- La versione più moderna usa una tecnologia ottica
  - Alcuni led illuminano il piano ed elaborano l'immagine
  - Facendo la differenza tra immagini successive si scopre in che direzione si è spostato il mouse

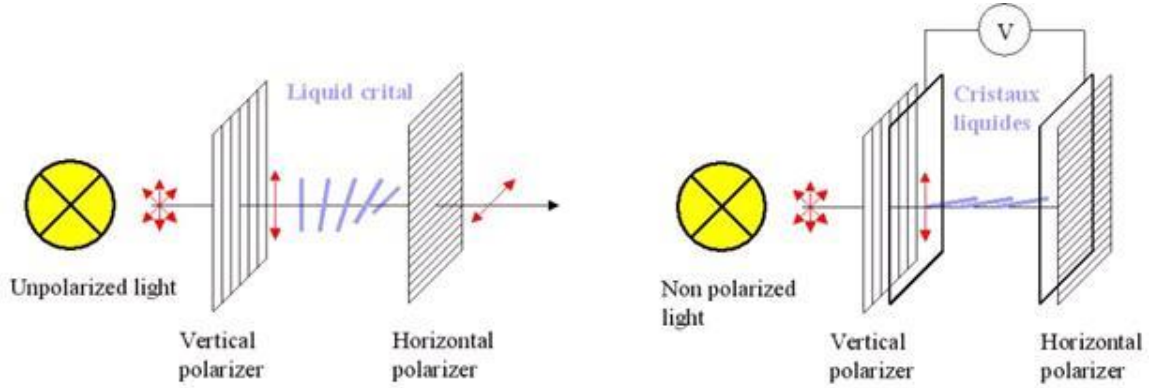


«The mother of all demos»
https://vimeo.com/32381658



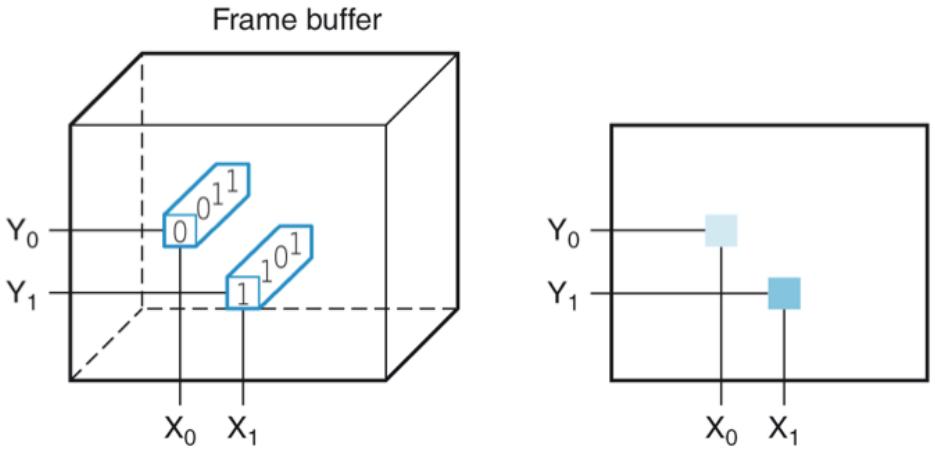
Esempi di IO: lo schermo LCD

- Gli schermi (*Liquid Crystal Display*) LCD sono attualmente diffusissimi sia nei PC sia nei telefoni cellulari e in altri dispositivi embedded
  - Alcune molecole di cristalli liquidi "galleggiano" in un fluido (poco in realtà)
  - Uno strato di molecole di cristallo è associato a un punto (pixel)
  - Tramite un campo elettrico (applicato a ciascun pixel) si riesce a far ruotare di 90 gradi il cristallo, che lascia passare o blocca la luce

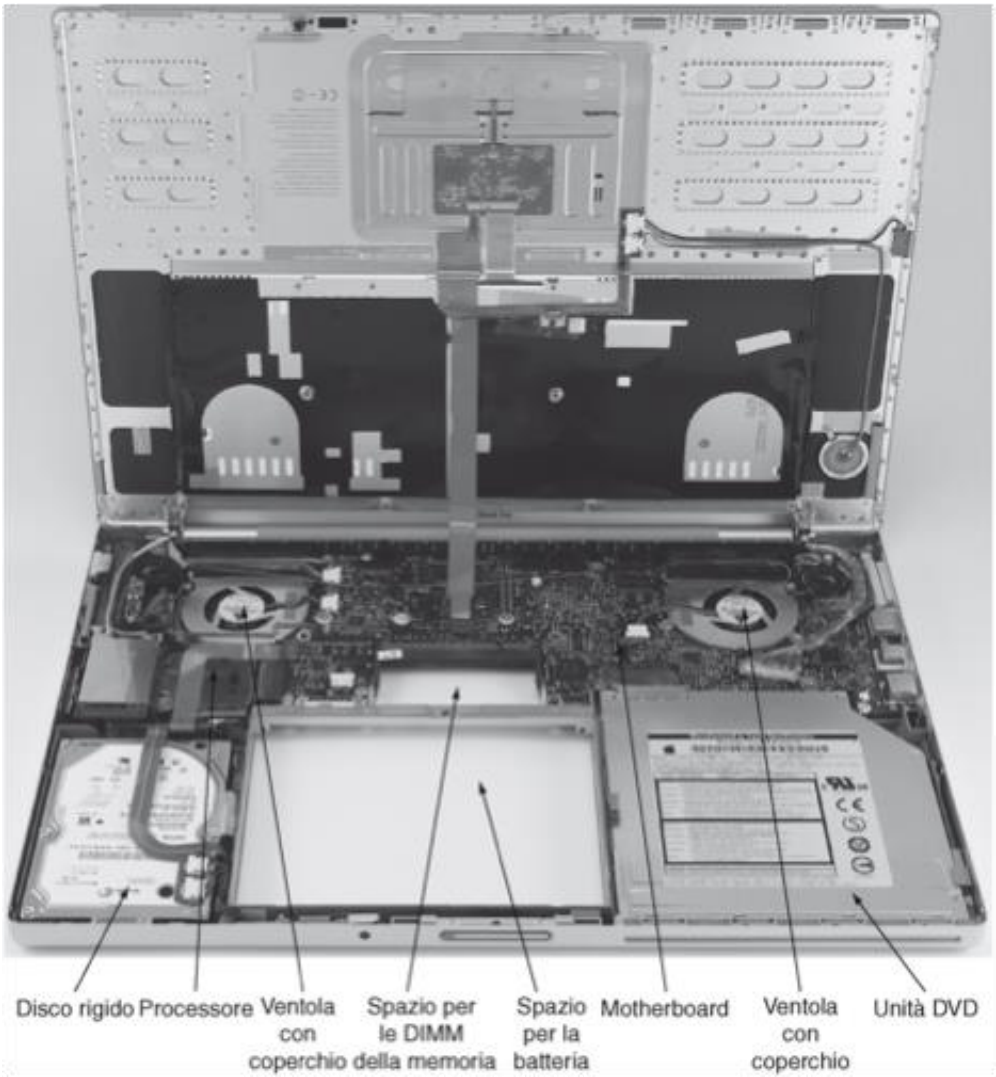


Esempi di IO: lo schermo LCD

- L'immagine si compone di una matrice di pixel (es. 640 x 480, 1440 x 900 ecc.)
- Ciascun pixel è associato tipicamente a tre byte, ciascuno associato a una delle tre componenti fondamentali Red (R) Green (G) e Blue (B)
- L'immagine viene memorizzata in una matrice (*frame buffer*), che è essenzialmente una velocissima RAM che viene aggiornata molte volte al secondo (da 50 a 100)



Dentro il PC



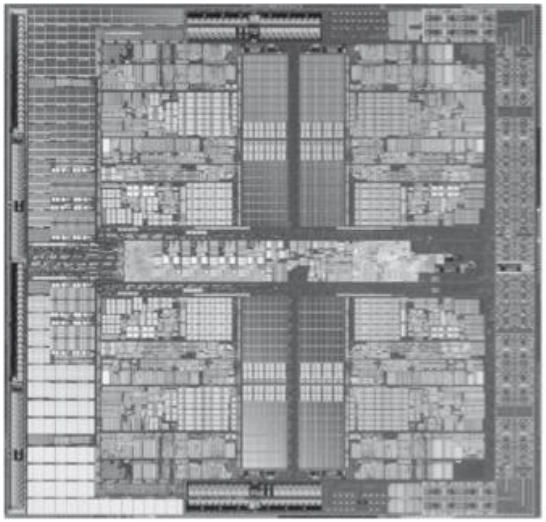
La scheda madre è una piastra su cui sono montati i vari circuiti integrati (chip)

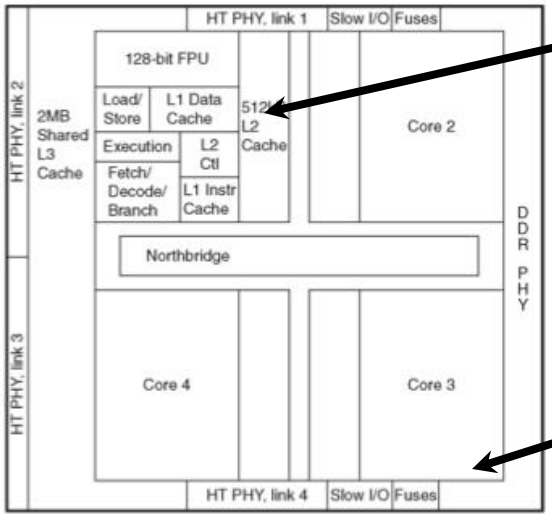
La memoria volatile è costituita da vari banchi di (tipicamente) 8 chip di RAM dinamica

La memoria permanente è costituita da Hard Disk/SSD e CD/DVD ROM

Il processore

- Il processore è la parte attiva di ogni calcolatore. Si compone di:
  - *Datapath*: esegue le operazioni aritmetiche sui dati
  - *Parte di controllo*: indica al datapath, alla memoria, e alle componenti di IO cosa fare sulla base di quanto stabilito nel programma





Una memoria RAM aggiuntiva (cache) all'interno della CPU migliora sostanzialmente le prestazioni

Questa unità si compone di quattro CPU (core)

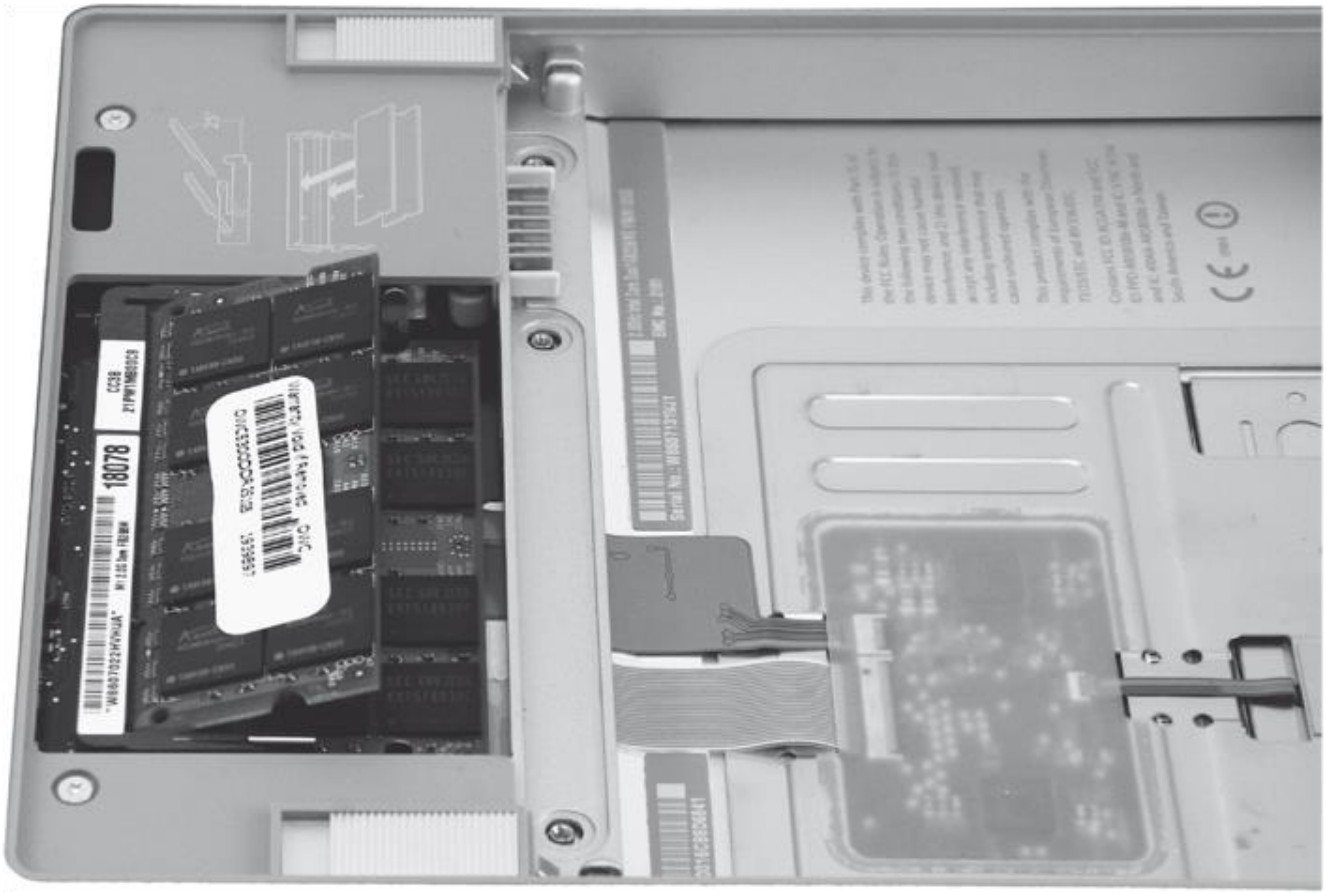
Astrazioni

- Nel mondo dell'informatica si lavora molto con le *astrazioni* che permettono di gestire un progetto di grande complessità nascondendo i dettagli
- Il processore viene "nascosto" nei suoi dettagli esportando come interfaccia l'insieme delle istruzioni macchina che il processore offre (Instruction Set Architecture)
- Insieme all'interfaccia del sistema operativo, l'ISA costituisce l'interfaccia binaria delle applicazioni (***A****pplication* ***B****inary* ***I****nterface*)
- Una volta definita una ABI, lo sviluppatore è svincolato da come i dettagli HW sottostanti l'applicazione sono implementati

La memoria

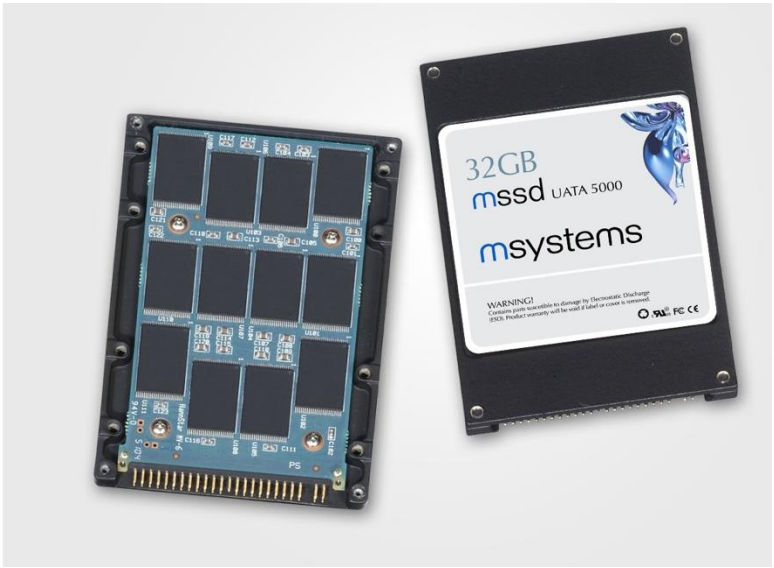
- La memoria si distingue in due tipi:
  - *volatile* (dominata dalle DRAM)
  - *non volatile* (dischi rigidi e memorie a stato solido)
- La memoria volatile viene usata per memorizzare dati e programmi mentre questi vengono eseguiti. Per questo motivo viene detta *memoria principale*
  - Allo spegnimento i dati vengono persi
- La memoria non volatile (o persistente) viene usata per memorizzare dati e programmi tra esecuzioni diverse.
  - Vista la quantità enorme di dati memorizzati si parla di *memoria di massa*

Le DRAM



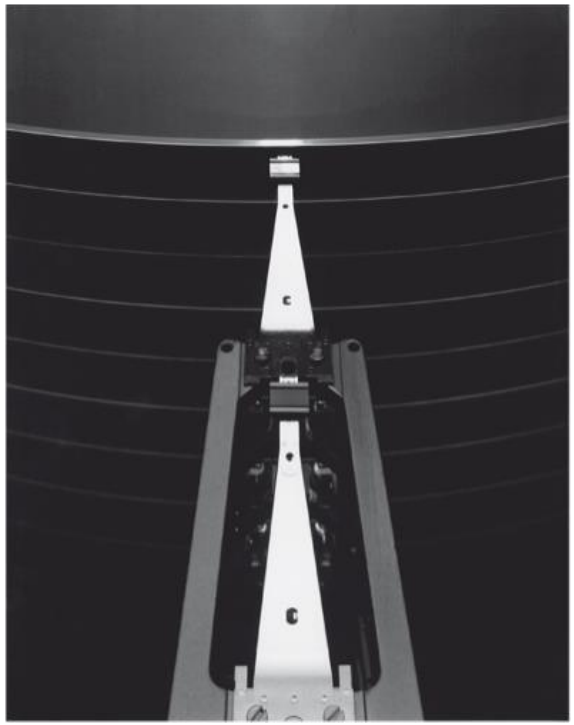
Memorie di massa

- Attualmente abbiamo essenzialmente tre tipi di memorie di massa
  - Memorie Flash (es. SSD)
  - Dischi rigidi
  - CD/DVD
- Le memorie Flash sono molto simili a memorie RAM. L'idea è di memorizzare il dato intrappolando una carica elettrica. Il fenomeno fisico usato dalle Flash consente alla carica di rimanere intrappolata in maniera ‘‘permanente’’



Memorie di massa

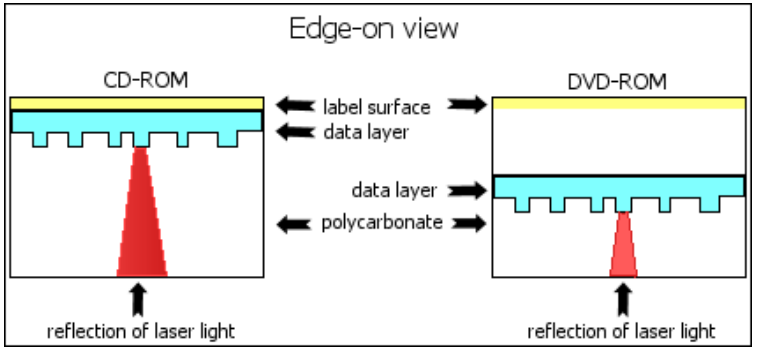
- *Hard Disk*: il principio di funzionamento è di magnetizzare delle particelle metalliche distribuite su un substrato
  - I dischi sono organizzati in strutture sovrapposte (cilindri)
  - Le particelle vengono lette da dispositivo meccanico (testina) che si sposta radialmente su un braccetto (in grado di fare movimenti angolari).
  - Questa componente rallenta i tempi di accesso ma aumenta la densità di memorizzazione (è possibile arrivare facilmente ai Terabyte)



https://www.youtube.com/watch?v=4BYrKhejb88

Dischi ottici

- I dischi ottici funzionano sulla base di un semplice principio: la riflessione della luce
- Viene emesso un raggio laser che viene riflesso dai ‘‘rilievi’’ (bit 1) e assorbito dalle ‘‘buche’’
- Nei dischi riscrivibili, un particolare substrato consente (tramite riscaldamento) di ritornare alla situazione originale “spianando le buche”



Interconnessione

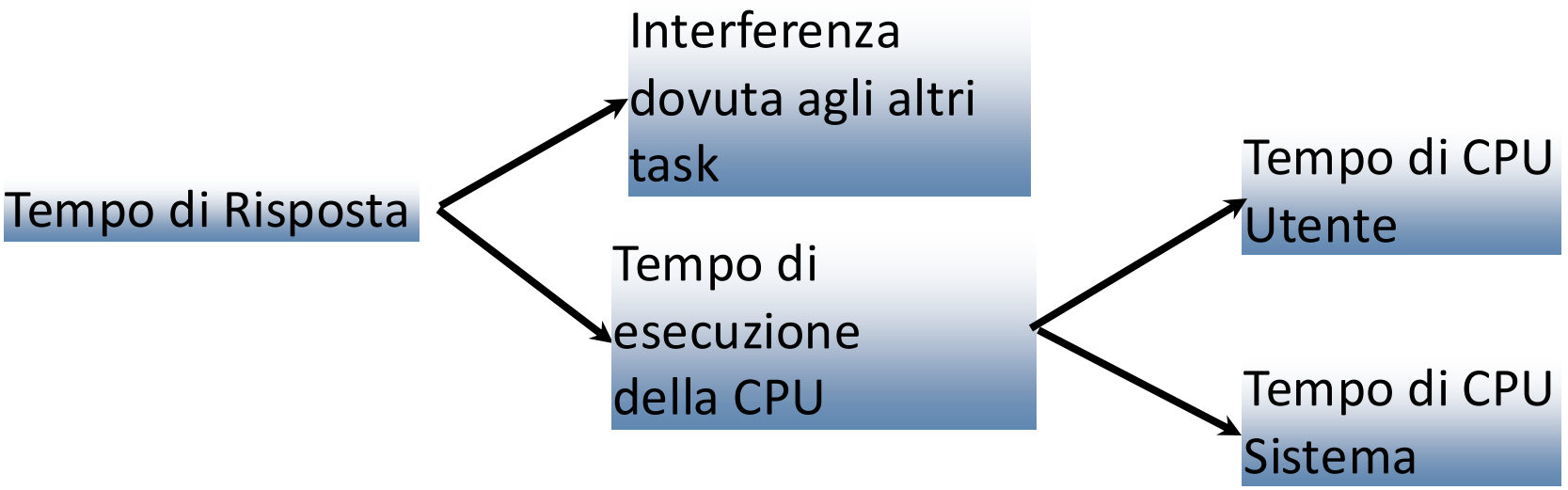
- Gli sviluppi più significativi degli ultimi anni hanno coinciso con lo sviluppo impetuoso della rete
- I calcolatori esprimono ormai la maggior parte del loro potenziale quando sono interconnessi
  - Local Area Network: condivisione di risorse (dischi, stampanti)
  - Wide Area Network: condivisione di contenuti

Definizione di prestazioni

- In genere, in un calcolatore si possono identificare due tipi di prestazioni:
  - Per il singolo utente interessa sapere quanto è il *tempo medio di risposta* (tempo che intercorre tra avvio e terminazione di un task)
  - Per il gestore di un centro di calcolo, interessa di più il *throughput*, cioè quanti task sono in grado di completare nell'unità di tempo

Tempo di esecuzione

- In una macchina multitask il tempo di risposta dipende anche dagli altri task attivi e dalle loro priorità
- Il tempo di esecuzione della CPU tiene conto solo del tempo effettivamente speso per il task
- Tale tempo è in parte dedicato al programma *utente* e in parte al *sistema* operativo



Capire le prestazioni

- I moderni processori, come vedremo, sono costruiti usando un segnale periodico che ne sincronizza le operazioni
- Il ciclo di clock è l'intervallo di tempo che intercorre tra due colpi di clock (la frequenza ne è l'inverso)
- Il ciclo di clock è misurato in secondi (o in frazioni di secondo), la frequenza in Hertz (o equivalentemente in cicli al secondo)
- Ad esempio un clock che va a un Giga Hertz (10^9 Hertz) equivale a un periodo di clock pari a 10^-9 secondi (un miliardesimo di secondo)

Misurare le prestazioni

- La maniera migliore di valutare le prestazioni di un computer è di misurare il tempo necessario per l'esecuzione di un programma
- Tale tempo è il risultato di tre fattori:
  - Numero istruzioni
  - Cicli di clock per Istruzione (CPI)
  - Frequenza di clock (1/Ciclo di clock)

Componente delle prestazioni	Unità di misura
Tempo di esecuzione della CPU per un programma	Secondi per programma
Numero di istruzioni	Istruzioni eseguite per programma
Cicli di clock per istruzione (CPI)	Numero medio di cicli di clock per istruzione
Ciclo di clock	Secondi per ciclo di clock

Equazione classica delle prestazioni

$$\begin{aligned}\text{Tempo CPU} &= \text{Numero Istruzioni} \times \text{CPI} \times \text{Periodo Clock} \\ &= \frac{\text{Numero Istruzioni} \times \text{CPI}}{\text{Frequenza Clock}}\end{aligned}$$

Componenti delle prestazioni

- Nessuna delle tre componenti può essere trascurata
  - Non ha nessun senso dire che un computer è più veloce di un altro perché ha una frequenza di clock più alta
- Ha invece senso cercare di capire come i diversi componenti e gli strumenti usati nello sviluppo di un determinato sistema (HW+SW) abbiano impatto sulle prestazioni (in particolare su ciascuno dei tre fattori)

Capire le presentazioni

Componente HW/SW	Cosa influenza	Dove è trattato?
Algoritmi	Determina il numero di istruzioni di alto livello e di operazioni di IO	Altri corsi
Linguaggi di programmazione, compilatori e architetture	Determina il numero di istruzioni macchina per ogni istruzione di basso livello	Cap. 2, 3
Processore e sistema di memoria	Determinano quanto velocemente è possibile eseguire ciascuna istruzione	Cap. 4, 5, 6
Sistema operativo, gestione HW e I/O	Determina quanto velocemente possono essere eseguite le istruzioni	Cap. 4, 5, 6

Cfr. tabella p. 7 Patterson-Hennessy

Algoritmo

- L'algoritmo adottato certamente influenza il *numero di istruzioni* ed eventualmente il *CPI*
- Perché?
  - Un algoritmo più efficiente può essere strutturato in modo da risparmiare istruzioni
  - Un algoritmo ben pensato (per una particolare architettura) utilizza istruzioni più efficienti (quelle con un basso CPI)

Linguaggio di programmazione

- Il linguaggio di programmazione influenza il *numero di istruzioni* e il *CPI*
- Perché?
  - I costrutti ad alto livello vengono tradotti in sequenze di istruzioni macchina
  - Un linguaggio con molte chiamate indirette (es. Java) ha in generale un valore di CPI più alto

Compilatore

- Il compilatore sicuramente influenza sia il numero di istruzioni che il CPI
- Perché?
  - Un compilatore più o meno efficiente genera un numero di istruzioni macchina diverso per ogni costrutto ad alto livello
  - Un compilatore ottimizzato (e ottimizzante) può tenere conto di una serie di effetti piuttosto complessi per ridurre il CPI

ISA

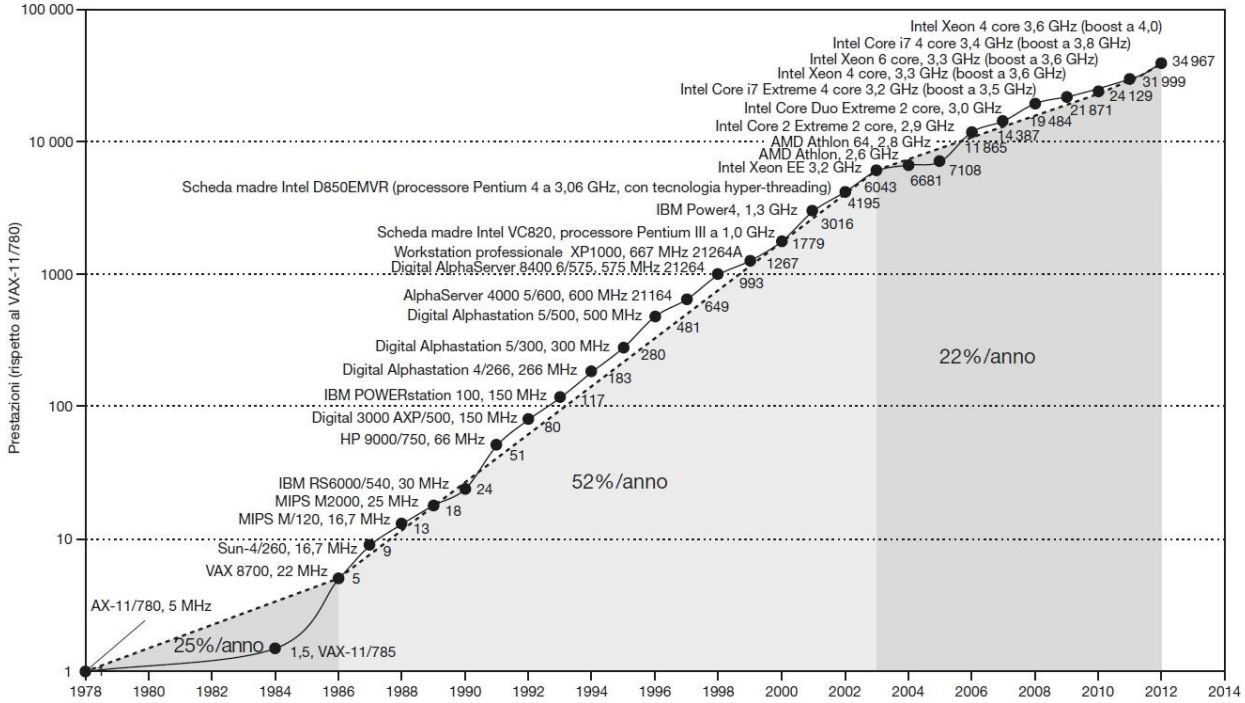
- L'architettura del set di istruzioni (ISA) consiste nell'interfaccia che la macchina offre al software
- Essa ha impatto sul numero di istruzioni, sul CPI, e sulla frequenza di clock
- Perché?
  - Numero di istruzioni: l'ISA può fornire istruzioni di alto o basso livello (quindi più o meno istruzioni per eseguire un'operazione)
  - CPI: il modo in cui un'ISA è progettata influenza il numero di cicli per eseguire ciascuna istruzione
  - Un'ISA ben progettata permette di avere frequenze di clock più spinte

ISA

- Durante questo corso vedremo 3 architetture:
  - RISC-V (esempio di architettura RISC)
    Usata ad es. per cloud computing e sistemi embedded
  - Intel (esempio di architettura CISC)
    Usata soprattutto su PC
  - ARM (esempio di architettura Advanced RISC)
    Usata soprattutto su sistemi embedded/mobile (+70% dispositivi mobili -cellulari e tablet- nel mondo)

La scalata delle prestazioni

- Negli scorsi anni le prestazioni dei calcolatori sono aumentati costantemente
- Di recente si è assistito a una diminuzione dell'incremento tra una generazione e l'altra



La scalata delle prestazioni

Intelligent Machines

Moore's Law Is Dead. Now What?

Shrinking transistors have powered 50 years of advances in computing—but now other ways must be found to make computers more capable.

by Tom Simonite May 13, 2016

Intelligent Machines

Chip Makers Admit Transistors Are About to Stop Shrinking

In the next five years, it will be too expensive to further miniaturize—but chip makers will innovate in different ways.

by Jamie Condliffe July 25, 2016

YEAR IN REVIEW » 2018

How Chip Makers Are Circumventing Moore's Law to Build Super-Fast CPUs of Tomorrow

https://www.technologyreview.com/s/601441/moores-law-is-dead-now-what/
https://www.technologyreview.com/s/601962/chip-makers-admit-transistors-are-about-to-stop-shrinking/
https://gizmodo.com/how-chip-makers-are-circumventing-moores-law-to-build-s-1831268322

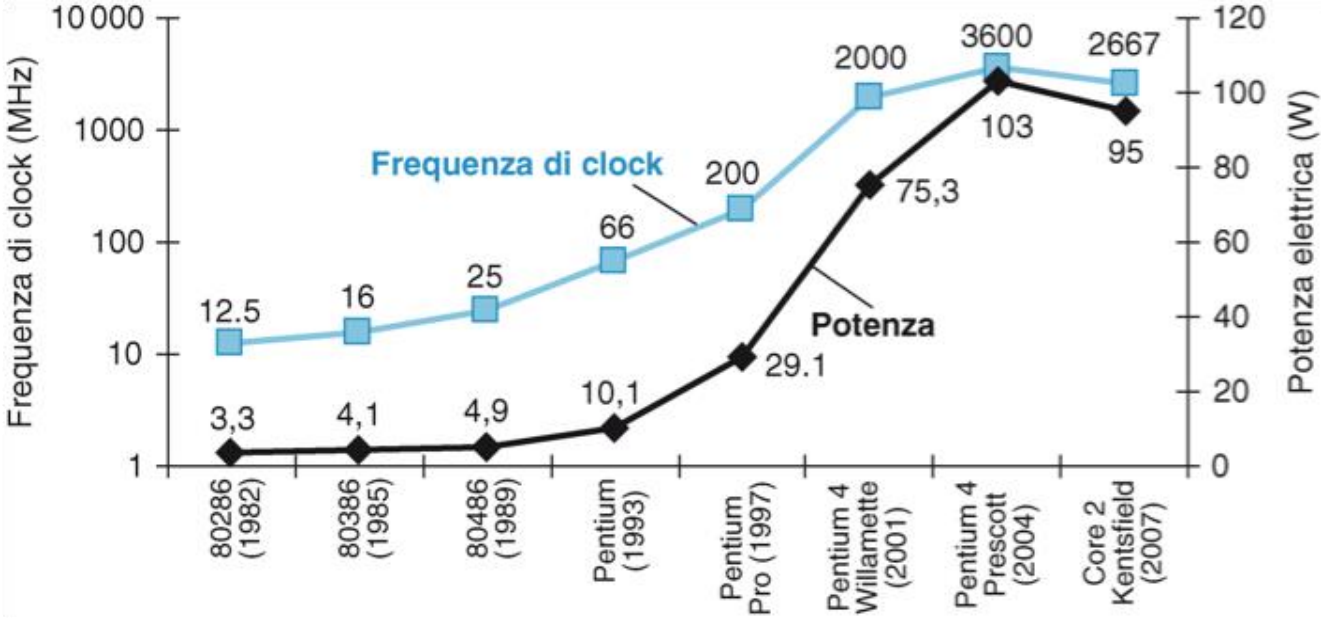
La barriera dell'energia

- Gli attuali processori sono costituiti di moltissimi interruttori che dissipano energia quando sono in fase di conduzione (commutazioni tra zero e uno)
- C'è un limite alla capacità di estrarre la potenza prodotta dai processori (e il conseguente calore) tramite ventole o radiatori (diciamo intorno ai 100W)
- Superato questo limite la refrigerazione diventa molto costosa e non è attuabile in un normale desktop (per non parlare dei laptop)
- La potenza è data da:

$$\text{Potenza} = \text{capacitá} \times \text{tensione}^2 \times \text{Frequenza di commutazione}$$

La barriera dell'energia

- La frequenza di commutazione è legata alla frequenza di clock…
- Eppure dagli anni '80 a oggi si è riusciti in un "miracolo": aumentare la frequenza di +1000 volte a spese di un aumento di consumi di un fattore 30



Come è stato possibile?

- Il “trucco” è stato quello di abbassare la tensione di alimentazione che agisce in maniera quadratica
- Si è passati in venti anni da tensioni di alimentazioni di 5V ai circa 1.2V attuali
- Questo ha permesso di incrementare la frequenza con impatti limitati sui consumi
- Sfortunatamente non ci si può spingere oltre lungo questa direzione perché se si abbassa la tensione sotto il Volt ci sono dei fenomeni di scarica in condizioni statiche che aumentano la dissipazione anche lontani dalle fasi di commutazione

Una nuova strada

- Nell'impossibilità di riuscire a drenare una grossa quantità di calore si è giunti a una sostanziale saturazione delle prestazioni del processore
- La nuova strada che allora viene esplorata è quella di aumentare il parallelismo (architetture multi-core)
- Questo crea molte difficoltà al programmatore per i seguenti motivi:
  - *correttezza*: è molto più difficile progettare (e debuggare) un programma che opera in maniera "parallela" rispetto a uno che opera in maniera sequenziale
  - *efficienza*: occorre che il carico di lavoro sulle CPU si mantenga bilanciato