Appunti di informatica

Lezione 5

anno accademico 2016-2017

Mario Verdicchio

Architettura dei calcolatori

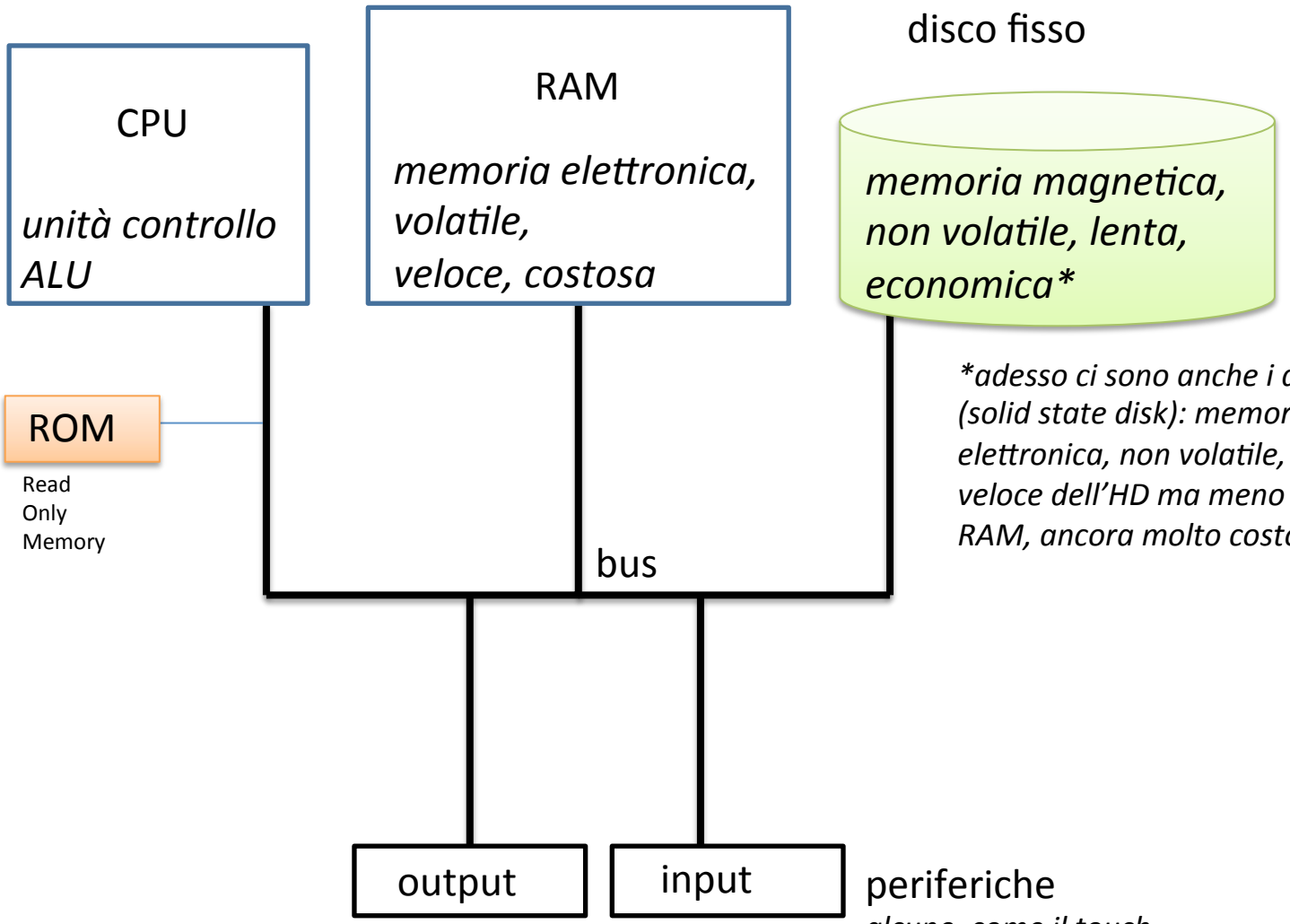
La maggior parte degli strumenti informatici oggi in uso si basano sull'architettura di Von Neumann

E' un modello astratto che include tutti i componenti funzionali (ossia distinti dagli altri per la funzione che svolgono) di un computer

CPU: Central Processing Unit, processore

RAM: Random Access Memory, Memoria Centrale

HD: Hard Disk, Hard Drive, disco rigido, disco fisso



*adesso ci sono anche i dischi SSD (solid state disk): memoria elettronica, non volatile, più veloce dell'HD ma meno della RAM, ancora molto costosa

periferiche

alcune, come il touch-screen sono sia di input sia di output

L’architettura del calcolatore

- Per descrivere le diverse parti di un calcolatore e le loro connessioni, si fa tuttora riferimento all’architettura di Von Neumann, proposta per la prima volta in un rapporto tecnico del 1945 (John von Neumann, Budapest 1903 – Wahington D.C. 1957)
- Su tale architettura si basa la maggior parte dei calcolatori odierni

Architettura di von Neumann

- I componenti dell'architettura sono:
  - la **CPU** (Central Processing Unit), che esegue le istruzioni dei programmi
  - la **RAM** (Random Access Memory), la memoria principale di tipo elettronico dove sono scritte le istruzioni da eseguire e i dati da elaborare mentre il calcolatore è acceso
  - il **disco fisso**, la memoria secondaria di tipo magnetico dove sono conservate le istruzioni e i dati anche quando il calcolatore è spento
  - le **periferiche**, componenti addizionali che permettono all'utente di trasferire informazione verso il calcolatore (**input**) o di ottenere informazione dal calcolatore (**output**)
  - il **bus**, il canale di comunicazione che connette tutti i componenti di cui sopra

Il funzionamento della CPU

- La CPU funziona a ciclo continuo, dall'accensione del calcolatore fino allo spegnimento, eseguendo le seguenti azioni a ripetizione:
  - **fetch**, prelevamento dell'istruzione da eseguire
  - **decode**, decodifica del codice dell'istruzione prelevata
  - **execute**, esecuzione dell'istruzione decodificata

CPU ⇔ RAM

- La CPU preleva le istruzioni da eseguire sempre dalla RAM
- Per questo motivo, ogni programma in esecuzione deve essere presente nella RAM
- Le istruzioni (e anche i dati su cui eseguire le istruzioni) sono trasferire dalla RAM verso dispositivi di memoria presenti nella CPU, chiamati registri
- L'unità di controllo della CPU supervisiona il trasferimento e la decodifica dell'istruzione, e ordina alla ALU (unità aritmetico logica) l'operazione da eseguire
- I risultati sono salvati sui registri, e da lì trasferiti alla RAM

RAM ↔ disco

- La RAM è basata su dispositivi elettronici, che funzionano con tempi molto rapidi, ma che necessitano di alimentazione elettrica per preservare l'informazione in essi contenuta
- Il disco fisso è invece costituito da materiale ferro-magnetico, la cui lettura e scrittura è molto più lenta, ma le informazioni salvate permangono anche a calcolatore spento
- Per questo motivo, tutti i programmi sono conservati nel disco fisso, e trasferiti alla RAM quando il calcolatore è acceso e vengono mandati in esecuzione
- Tutte le modifiche ai dati fatte tramite un programma in esecuzione, se sono presenti solo nella RAM ma non nel disco fisso sono perse se l'alimentazione elettrica viene meno
- Quando si salva un file (di fatto, un insieme di dati), si ordina il trasferimento dei dati dalla RAM al disco, in modo da conservare l'ultima versione anche dopo lo spegnimento del calcolatore
- All'accensione del calcolatore, il primo programma che viene "caricato" (cioè trasferito dal disco alla RAM) è il sistema operativo, che funziona per tutto il tempo in cui il calcolatore è acceso e serve a gestire tutte le sue risorse

periferiche→RAM

- Per inserire dati all'interno del calcolatore, l'utente deve usare le periferiche di input
- Ad esempio, durante l'esecuzione di una istruzione x = input("insersisci un numero"), il calcolatore attende un valore inserito tramite tastiera, da inviare in una particolare cella di memoria RAM, corrispondente alla variabile x
- Da lì, il valore sarà mandato alla CPU per essere elaborato (ad es. per eseguire x = x + 5)

RAM→periferiche

- Le periferiche di output ricevono dati provenienti dalla RAM e li inviano verso il mondo esterno al calcolatore (ad es. visualizzandoli su uno schermo o stampandoli su carta)
- Istruzioni come print(x) prevedono il trasferimento del contenuto di una cella di memoria RAM (corrispondente alla variabile x) verso una specifica periferica (lo schermo)

Sistema Operativo

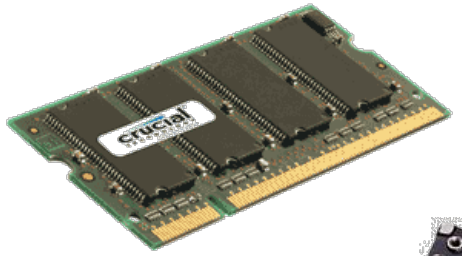
- Windows 8, Mac OS X, Linux, Debian, Red Hat, etc etc
- Il sistema operativo è un programma che serve a gestire TUTTE le risorse presenti in un calcolatore
- Risorse: periferiche, memorie, CPU.
- Le risorse sono usate dai vari programmi in funzionamento sul calcolatore. Tra questi il più importante è il sistema operativo, che gestisce tali risorse.
- Il sistema operativo è il PRIMO* programma ad entrare in funzione. **con la precisazione nella seguente slide*

Il problema del bootstrap

- Il problema di come avviare un calcolatore
- Fortunatamente, ci sono anche memorie elettroniche che conservano l'informazione. (ad es.: la scheda SD della vostra macchina fotografica digitale)
- All'accensione, la CPU esegue un programma scritto nella ROM che fa trasferire le istruzioni del sistema operativo dal disco fisso alla RAM
- Una volta nella RAM, il sistema operativo può essere eseguito dalla CPU e il computer si avvia a disposizione dell'utente

Memorie

- Dispositivi in grado di conservare al loro interno informazione per un intervallo di tempo significativo



RAM

chiavetta USB

CD/DVD

SD card

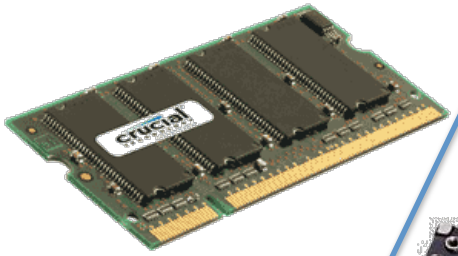
Hard disk

Tecnologie diverse

ottica

elettronica

magnetismo



Caratteristiche diverse

- Velocità

- Non volatilità

- Tolleranza ai guasti

Caratteristiche diverse

- Costo (a parità di dimensione)

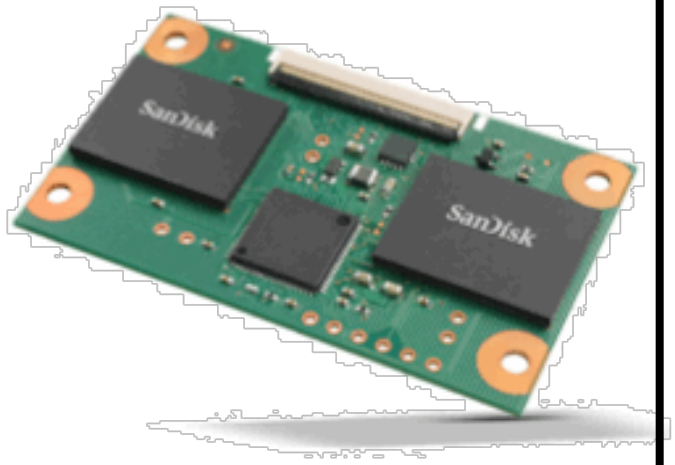
Hard disk

- E' il dispositivo di memoria con la maggiore capacità: centinaia di GB contro qualche unità di GB degli altri dispositivi
- E' anche il dispositivo più suscettibile ai guasti a causa delle parti meccaniche che lo compongono (testina di lettura/scrittura su braccio mobile e dischi magnetici rotanti), ed è il più lento
- I dischi sono divisi in settori concentrici, ciascuno dei quali diviso in blocchi
- Quando c'è la richiesta di una scrittura/lettura di un dato in un blocco, la testina deve essere posizionata sul blocco corretto
- Tale posizionamento necessita di un tempo con due componenti: seek time (posizionamento sul settore giusto) e latenza (attesa della rotazione del disco perché il blocco giusto raggiunga la testina)
- A tali tempi bisogna aggiungere il tempo necessario per il trasferimento dei dati dal disco alla RAM (transfer time)

Memorie elettroniche non volatili

- Le chiavette USB, le SD card, e i più recenti hard disk a stato solido sono basati su tecnologia elettronica, ma, a differenza delle RAM, non sono volatili
- La non volatilità è data da uno strato di ossido scrivibile e cancellabile (per mezzo di appositi valori di tensione) che si accompagna in questi dispositivi ai circuiti elettronici
- In origine chiamati memorie flash perché l'alta tensione usata per cancellarle era paragonata a un flash (Toshiba, primi anni '80)
- I primi modelli potevano solo essere cancellati completamente, oggi la cancellazione è selettiva e l'utente può scegliere quali file cancellare e quali conservare

SSD

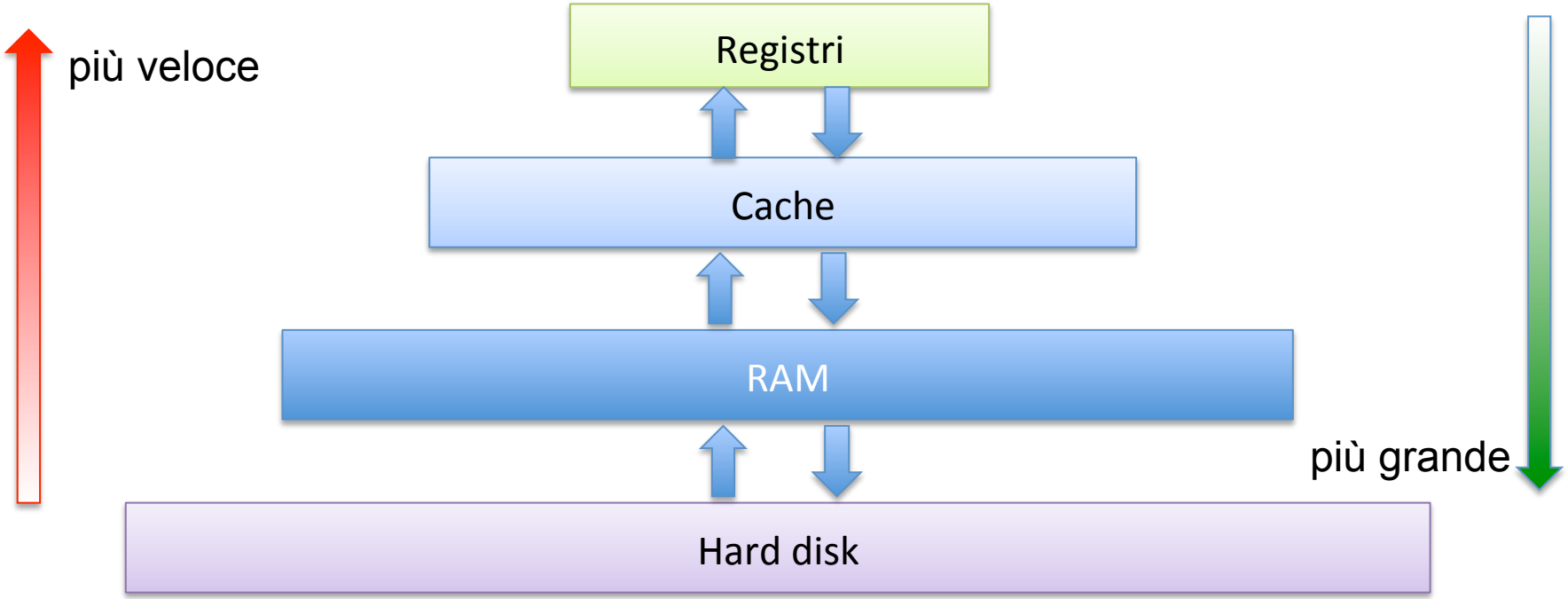


Recentemente, in alternativa al classico HD magnetico, viene offerto un nuovo tipo di HD, detto SSD (Solid State Disk), che ha la stessa caratteristica di non volatilità, ma è basato su tecnologia elettronica, quindi senza parti meccaniche (dischi, testine) che possono guastarsi, e più veloce.

Viene da chiedersi perché l'SSD non venga proposto come alternativa alla RAM volatile. In realtà, l'SSD, come le altre memorie elettroniche non volatili, pur essendo più veloci dell'HD magnetico, sono comunque molto più lente della RAM, e non riuscirebbero a lavorare in sincronia con la CPU (che arriva a lavorare a frequenze di 3 GHz, ossia a compiere 3 miliardi di operazioni al secondo). Inoltre, il numero di scritture richiesto dal lavorare con la CPU non è compatibile con la tecnologia SSD

Gerarchia di memorie

- Tutti i dispositivi di memoria che sono presenti nell'architettura di un calcolatore sono organizzati in una gerarchia



Gerarchia di memorie

- Ogni livello comunica solo con i livelli immediatamente adiacenti
- I registri sono dispositivi di memoria direttamente connessi alla CPU (es.: PC, il Program Counter, contiene l'indirizzo in memoria della prossima istruzione da eseguire)
- La cache è un dispositivo di memoria direttamente connesso coi registri che riporta il contenuto di una parte della RAM
- Quando i dati viaggiano tra cache, registri, e CPU, non attraversano il bus del calcolatore, perché questi 3 dispositivi sono tutti costruiti su una stessa scheda elettronica (perciò si ottiene velocità molto maggiore)

Uso della gerarchia

- Quando necessita di un dato da elaborare, la CPU lo cerca nel dispositivo di memoria più vicino
- Se il dato non viene trovato, lo si cerca nel dispositivo di memoria successivo, con un costo maggiore in termini di tempo
- Nel peggiore dei casi, il dato va recuperato nell'hard disk, il dispositivo più lento di tutti

Gestione della gerarchia

- La situazione ideale (irrealizzabile) sarebbe che la CPU avesse sempre i dati necessari a disposizione nei registri
- Ottimizzare l'uso della gerarchia vuol dire ridurre al minimo il ricorso ai livelli di memoria inferiori
- Per perseguire questo scopo ci si basa su due principi derivati dall'esperienza

Principio di località spaziale

- Se la CPU necessita di un dato x, è probabile che necessiterà presto dei dati vicino a x in memoria
- Questo perché molto spesso i dati sono organizzati in strutture contigue (array, struct, etc…)
- Seguendo questo principio, quando si cerca un dato x nei livelli inferiori della gerarchia di memorie, non si trasferisce solo x, ma un intero blocco di dati contenente x
- In questo modo, se la CPU necessita dei dati intorno a x questi sono già disponibili nei livelli alti della gerarchia, con risparmio di tempo

Principio di località temporale

- Se la CPU necessita di un dato x, è probabile che necessiterà di nuovo di x nel prossimo futuro
- Questo perché molto spesso i programmi contengono cicli (for, while, etc…)
- Seguendo questo principio, quando un blocco viene trasferito ai livelli alti (es.: da RAM a cache) e non c'è più spazio disponibile (es.: la cache è piena), si sovrascrive il blocco non usato da più tempo, perché è quello che meno probabilmente sarà richiesto dalla CPU in futuro
- La strategia di sacrificare il blocco meno usato da più tempo si chiama LRU (Least Recently Used)