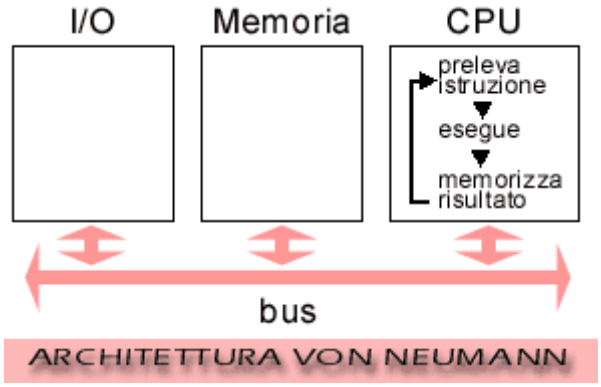
MEMORIE

Architettura di Von Neumann

Un generico sistema di elaborazione può essere rappresentato con il seguente schema detto Architettura di Von Neumann nel quale vengono raffigurati in maniera del tutto astratta le varie componenti da cui è costituito: dispositivi di Input/Output, dispositivi di memoria, di elaborazione(CPU), canali di comunicazione che permettono di traferire dati tra i vari componenti(BUS).



Organizzazione gerarchica della memoria

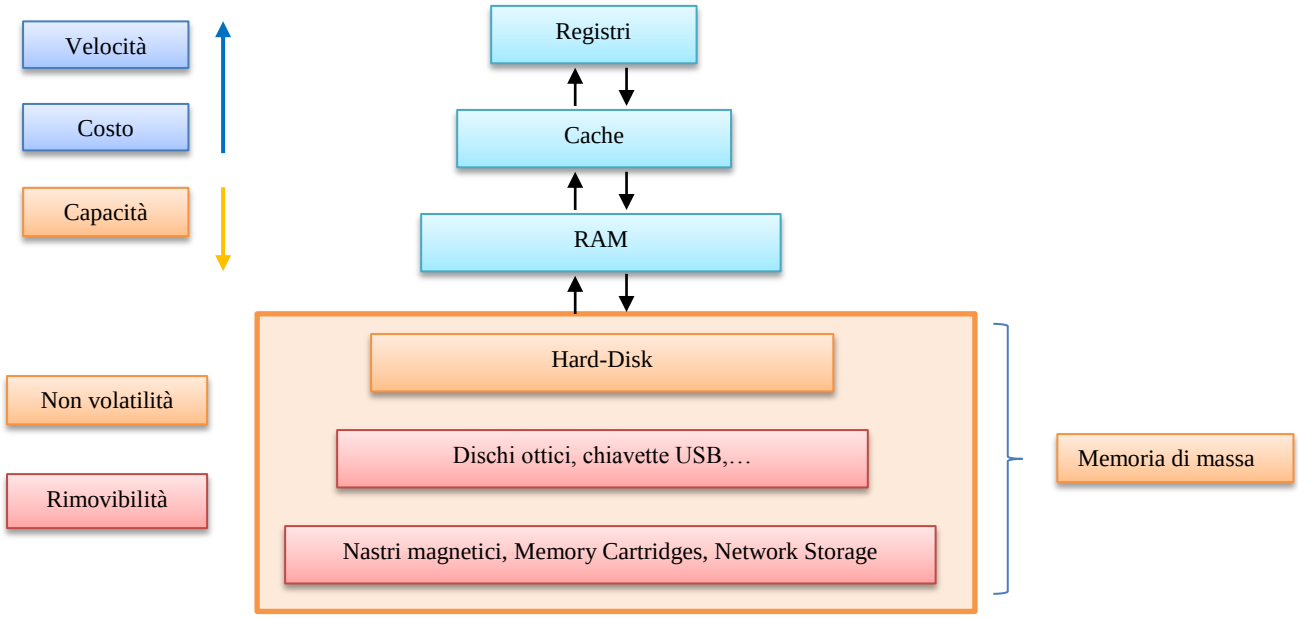
Nell'architettura di Von Neumann, la memoria è l'unità del computer destinata a conservare le informazioni in maniera temporanea o permanente. Queste informazioni sono costituite sia da dati che da istruzioni, ovvero da programmi.
Lo schema della macchina di Von Neumann prevede un unico e generico tipo di memoria da cui il processore preleva istruzioni e dati e dove deposita i risultati delle elaborazioni.
L'ideale dal punto di vista delle prestazioni sarebbe implementare tutta la memoria come registri interni del processore o con tecnologie a semiconduttore(come la Cache e la RAM) consentendo un accesso quasi istantaneo alle informazioni; d'altra parte, per aver memorie in grado di contenere un' elevata quantità di dati, senza spendere cifre astronomiche, sarebbe opportuno scegliere memorie dal basso costo per bit.
Oltre ad essere limitata dai costi, la capacità delle memorie sono limitate anche da ragioni di tipo tecnico: è impossibile realizzare, con le tecnologie attuali, una memoria cache di I° livello che abbia la stessa capacità di una memoria RAM, semplicemente non c'è spazio a sufficienza! La cache di primo livello infatti per funzionare ad una certa velocità deve essere molto piccola, in quanto i segnali elettrici si propagano ad una velocità finita e più e grande è la distanza, maggiore il tempo necessario a percorrerla!.
E' inoltre necessario conservare questi dati premunendosi contro interruzioni dell'alimentazione(servono memorie che siano non-volatili) e permettere il trasferimento dei dati(servono supporti rimovibili).

Il compromesso fra queste esigenze ha fatto nascere una gerarchia a tre livelli (i registri interni del processore non vengono considerati memoria vera e propria):

- **Memoria cache** (memoria molto veloce, di piccolo dimensioni e costosa)
- **Memoria centrale** (memoria veloce, di medie dimensioni e abbastanza costosa)
- **Memoria secondaria o di massa** (memoria lenta, di notevoli dimensioni ed economica).

Riassumendo…

Nei computer attuali si utilizzano differenti tipi di memorie che differiscono per una serie di caratteristiche fra cui, in particolare, **la velocità** (determinata dal tempo di accesso(tempo necessario per leggere/scrivere un dato) e da quello necessario per il trasferimento dei dati), **il costo, la dimensione(capacità), la volatilità e la rimovibilità.**
L'utilizzo di più memorie di diverso tipo *permette di garantire un'elevata velocità di accesso in lettura/scrittura e, allo stesso tempo, una grande capacità di memorizzazione, limitando il costo complessivo.*
Inoltre consente di conservare i dati in maniera permanente su supporti di memoria non-volatili ed eventualmente facilmente trasportabili da un luogo all'altro.



Percorso dei dati

1. **I dati utilizzati durante l'esecuzione vengono caricati dalle memorie di massa in RAM**.(es. file musicale da ascoltare letto dall'hard-disk e memorizzato in RAM). Ogni programma(come vedremo meglio in seguito) ha a disposizione una certa quantità di memoria che viene riservata(in gergo "allocata") al programma dal Sistema Operativo.
2. Nel corso dell'esecuzione **una parte dei dati/delle istruzioni viene portato in cache** sulla base di una serie di dati statistici relativi al loro utilizzo che vengono continuamente aggiornati. La decisione viene presa da un algoritmo che cerca di prevedere e anticipare le richieste della CPU.
3. **Nei registri della CPU viene caricato solo quello che serve per eseguire l'istruzione corrente**.
4. **I risultati dell'elaborazione** possono essere **memorizzati in cache**(se si prevede che verranno riutilizzati a breve) o **direttamente in RAM,** in alcuni casi **rimangono** direttamente **nei registri della CPU.**
5. A questo punto l'istruzione successiva viene prelevata dalla memoria(cache o RAM), viene decodificata(ovvero si individua il tipo di operazione richiesta(esaminando il codice operativo) e la posizione dei dati richiesti(operandi) che possono trovarsi già nei registri o in RAM o nella memoria di massa), si prelevano gli operandi caricandoli nei registri e si esegue l'operazione. Se i dati o le istruzioni sono stati preventivamente caricati in cache si risparmiano lenti accessi in RAM e il tempo complessivo necessario per l'esecuzione dell'istruzione sarà di molto inferiore. Nel caso in cui dati richiesti si trovassero in RAM verranno portati in cache e di qui nei registri; qualora si trovassero sulla memoria di massa dovranno risalire la piramide fino in cima. L'esecuzione del programma sarà bloccata fino a quando i dati non saranno disponibili.
6. **La RAM non ha spazio a sufficienza** per contenere tutte le informazioni necessarie nel corso dell'esecuzione per cui **è necessario riportare una parte dei dati** caricati in precedenza (che si ritiene non serviranno nell'immediato) **sull'hard-disk(nella cosiddetta Memoria Virtuale).**
7. **Al termine dell'esecuzione di un programma i risultati delle elaborazioni eseguite possono essere definitivamente salvati sull'hard-disk**. **La maggior parte del contenuto delle aree di memoria allocate per il programma sarà semplicemente sovrascritto**.

CRITERI DI CLASSIFICAZIONE DELLE MEMORIE

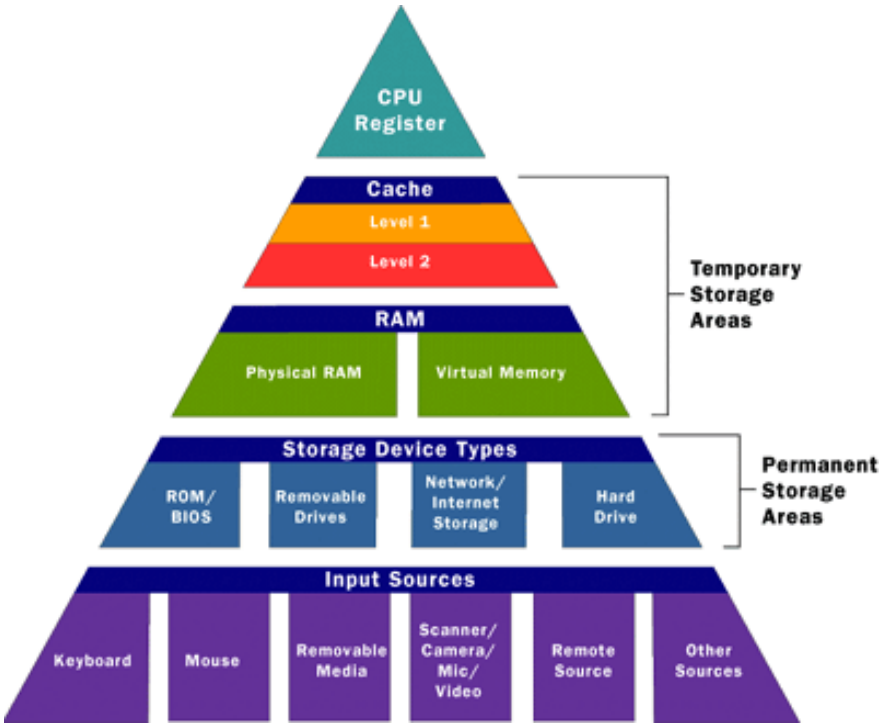
Le memorie possono essere classificate sulla base di un'ampia gamma di caratteristiche; di seguito ne sono elencate alcune:

- **modalità di accesso:** *memorie ad accesso casuale, sequenziale, diretto(o misto)*
- **funzione** che svolgono all'interno del sistema di elaborazione(vedere più avanti…)
- **tempo di immagazzinamento:** *memorie temporanee e permanenti*
- **necessità di essere alimentate o meno:** *memorie volatili e non volatili*
- **modalità con in cui vengono conservate le informazioni:** *memorie magnetiche, ottiche, elettroniche(dette anche memorie a semiconduttore o memorie a stato solido), magneto-ottiche*
- **tecnologia impiegata:** *CD-R,CD-RW, DVD, BlueRay,…*
- **possibilità di scrivere/riscrivere il contenuto:** *memorie a sola lettura, lettura e scrittura, riscrivibili*
- **rimovibilità:** *memorie rimovibili*(che possono essere rimosse e ricollegate con facilità al sistema di elaborazione) *e "fisse"*

Lo schema seguente evidenzia alcuni dettagli della gerarchia di memorie come la classificazione delle memorie in temporanee(**notare che la RAM è suddivisa in RAM fisica e Memoria Virtuale**) e permanenti(tra le quali vengono incluse anche dispositivi di archiviazione di rete ovvero **si tiene conto della possibilità di archiviare i dati a distanza trasferendo i dati attraverso Internet o più tipicamente su dispositivi collegati ad una rete locale**(es. rete del laboratorio)), inoltre mostra anche alcune fonti di input(

a parte i soliti dispositivi quali tastiera, mouse, scanner…sono incluse sorgenti remote) ad ognuno di questi dispositivi è associata come vedremo una memoria detta buffer.

Lo schema evidenzia inoltre come all'interno dei computer attuali esistano tipicamente più *Memorie Cache* organizzate anch'esse in un gerarchia che parte da quella più veloce e avente capacità minore(normalmente collocata direttamente sul processore) nota come *Cache di 1° 1ivello* per arrivare a quella di *3° livello*(non mostrata in figura) collocata sulla scheda madre o all'interno del package del processore .



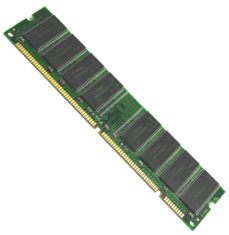
Classificazione in base al tempo di conservazione dei dati

Le memorie possono essere distinte secondo questo criterio in *temporanee* e *permanenti*.
Sebbene **di norma** le memorie temporanee siano memorie volatili e quelle permanenti siano memorie non-volatili, **non sempre tale regola è valida**. **Temporaneo non è necessariamente sinonimo di volatile** si noti, infatti, che la **RAM Virtuale** mostrata nella figura sottostante è in realtà **un'area del disco fisso** (dunque una memoria **non** volatile!) utilizzata per memorizzare **temporaneamente** una parte dei dati che dovrebbero essere caricati in RAM, ma per i quali non c'è spazio a sufficienza (è "virtuale" in quanto non corrisponde alla RAM "fisica").
Inoltre vale la pena notare che anche le memorie non-volatili vanno incontro ad un progressivo deterioramento a causa di vari fattori, sebbene la durata e l'affidabilità di tali memorie sia in genere molto superiore rispetto a quella delle memorie volatili.

Classificazione in base alla modalità di accesso

Una memoria si dice ad **accesso casuale (Random Access Memory)** quando il tempo di accesso(ovvero il tempo necessario per ricercare il dato e leggerlo/scriverlo) non dipende dalla posizione del dato(ovvero dal suo indirizzo di memoria) ovvero è lo stesso per tutti i dati conservati in memoria. Il termine RAM indica che la "memoria principale" è una memoria ad accesso casuale.

Nelle memorie ad **accesso sequenziale (Sequential Access Memory),** invece, le locazioni di memoria sono disposte in sequenza e per accedere ad una qualunque di esse è necessario scorrere tutte quelle che precedono, o seguono, quella a cui si è acceduto precedentemente. Esempi di tale tipo di memoria sono i *nastri magnetici* e le *memory cartridge* (cartucce di memoria: molto simili alle vecchie cassette musicali ma di dimensioni minori).
La memoria ad **accesso diretto(o misto),** anche chiamata memoria ad accesso misto, è una tipologia di memoria caratterizzata dal permettere l'accesso diretto a qualunque indirizzo di memoria con tempo di accesso variabile e dipendente dall'indirizzo di memoria a cui è avvenuto l'accesso precedente.
Tutte le memorie secondarie(a parte quelle a stato solido che funzionano come la RAM ovvero le memorie FLASH delle chiavette USB ad esempio) sono di questo tipo. Ad esempio nell'hard-disk dipende dalla posizione del settore di traccia da leggere rispetto alla posizione attuale della testina. Idem nel caso delle memorie ottiche.
Tali memorie infatti presentano da un lato alcune delle caratteristiche delle memorie sequenziali e dall'altro alcune delle caratteristiche delle memorie ad accesso casuale(ovvero il fatto che non si devono necessariamente leggere tutte le locazioni intermedie).



Sono mostrate nell'ordine una "RAM", un Nastro Magnetico e una Cartridge

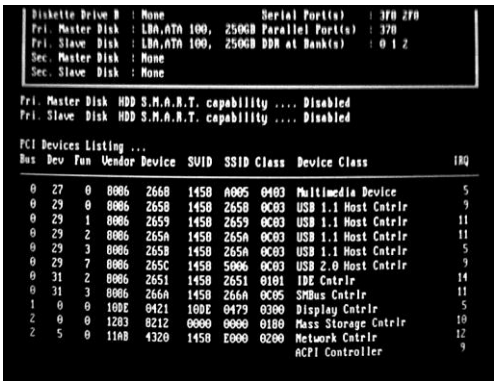
Classificazione in base alla funzione

ROM/BIOS

La ROM/BIOS (attualmente memorie di tipo FLASH), serviva originariamente a memorizzare un insieme di routines software, il BIOS(*Basic Input-Output System*) appunto, il cui scopo era quello di fornire una serie di funzioni di base per l'accesso all'hardware e alle periferiche integrate nella scheda madre da parte del sistema operativo e dei programmi.
Nei primi PC IBM il BIOS supportava tutte le periferiche e il DOS(l'antenato di Windows) faceva completo affidamento su di esso per le operazioni a basso livello, ma con l'evoluzione tecnologica le capacità offerte dalle routine di gestione del BIOS (all'epoca non aggiornabili, perché scritte in ROM) divennero rapidamente insufficienti. Attualmente i moderni sistemi operativi non usano più il BIOS per le loro operazioni di Input/Output ma accedono direttamente all'hardware.
Le funzioni che vengono svolte dal BIOS sono principalmente 2:

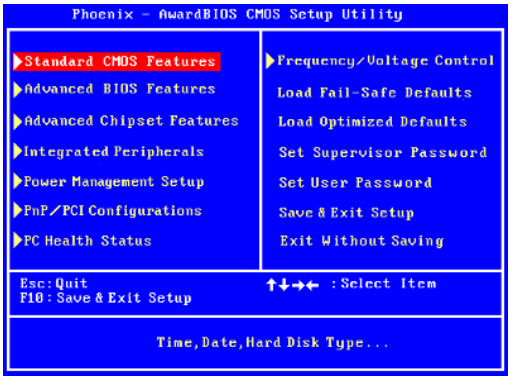
1. **esecuzione del POST(*power-on self-test*)**
2. **avvio del Sistema Operativo**

Il POST (*power-on self-test*) è il primo programma che viene eseguito dopo l'accensione.
Il nome del programma significa letteralmente *auto-diagnosi di avvio* e viene avviato automaticamente dal BIOS all'accensione del computer (e di altri dispositivi, ad esempio router e stampanti) per testare il corretto funzionamento dell'hardware prima di proseguire nel *boot* (fase di avvio) del sistema.
I risultati di questo test venivano in precedenza mostrati su una schermata nera di avvio come quella mostrata di seguito.



I PC attuali non mostrano direttamente all'utente la schermata del POST, ma bisogna premere un pulsante (solitamente F12) per far scomparire il logo della casa produttrice e visualizzare i dati del POST. Inoltre è sempre più diffusa l'opzione che permette la visualizzazione di un menù per scegliere il boot da eseguire(nel caso in cui ad esempio siano installati sistemi operativi diversi o se il sistema è stato riavviato dopo un interruzione dell'alimentazione e occorre decidere se riavviarlo in modalità provvisoria o normale), senza necessariamente agire sulla configurazione.

A parte questa fase di test il compito primario del BIOS è **l'avvio del sistema operativo.**
Inoltre permette di effettuare alcune configurazioni(ad esempio stabilire se il sistema operativo deve cercato prima sull'hard-disk o sull'unità CD-ROM). Il BIOS ha una schermata simile a quella mostrata nella figura seguente.



Memorie di massa

Altre memorie, classificate come memorie secondarie o di massa servono a **conservare stabilmente dati e programmi** e sono in grado di contenerne quantità considerevoli su supporti "fissi" come l'hard-disk o più spesso **"rimovibili"** come nel caso dei vari tipi di dischi ottici, delle pen-drives, ecc…
I supporti rimovibili vengono utilizzati per la **conservazione di dati utilizzati con minore frequenza di quelli presenti sull' hard-disk**, per consentire il loro **trasferimento da un PC all'altro**, per effettuare copie di **backup(ovvero copie di sicurezza o di riserva) dei dati e del software** che si desidera salvaguardare da guasti e danni di altro genere(es. incendio, virus informatici, furto…), al fine di prevenire la perdita totale dei dati archiviati.

RAM

La memoria centrale(detta anche principale o primaria), nota anche come memoria **RAM**, **serve a contenere i dati e le istruzioni dei programmi in esecuzione (o almeno una parte).** La sua funzione è strettamente legata al fatto che i tempi di lettura/scrittura sulla RAM sono nettamente inferiori di quelli sui supporti di memoria di massa.

CACHE(Memoria "nascosta" o "invisibile")

Le **Memorie Cache** viste sopra, servono a **contenere i dati e le istruzioni che si prevede vengano riutilizzati nuovamente a breve.** Poiché il tempo di accesso a tali memorie è di molto inferiore rispetto a quello della RAM, **l'utilizzo delle *Memorie Cache* consente di evitare accessi in RAM e dunque di ridurre il tempo di elaborazione complessivo(riducendo il tempo necessario a recuperare i dati).**

In generale, dunque, la Memoria Cache è una memoria temporanea, non visibile al software, utilizzata per conservare dati che si prevede verranno riutilizzati nuovamente a breve, in maniera da ridurre il tempo necessario per il loro recupero.
Essenzialmente una cache è una piccola e veloce memoria utilizzata per ridurre i tempi di accesso ad una memoria più grande, ma molto più lenta.

L'origine del nome deriva dal termine francese *caché* che significa *nascosto*, questo per via del fatto che la memoria cache e il suo utilizzo sono, ***in genere*, "trasparenti" al programmatore, ovvero la gestione della cache hardware o software non è effettuata dai programmi utente**(le cache hardware sono governate da un microprogramma contenuto nella cache stessa, nel caso delle cache software il programma che si occupa della sua gestione è in genere ad un livello superiore a quello utente es. *Memoria Virtuale* gestita dal Sistema Operativo(l'utente può solo stabilire lo spazio complessivo occupato, ma i programmi utenti non hanno alcun controllo su di essa), Cache del *Browser* utilizzata per conservare i file relativi alle pagine web visitate più recentemente(l'utente può solo cancellarne il contenuto).
Nel caso delle cache L1,L2,L3 tra CPU e RAM, a livello di codice sorgente o binario è come se tutti i dati fossero prelevati direttamente dalla RAM.

Quando viene utilizzata

1. **Tempo di accesso al dispositivo di archiviazione(locale o remoto) molto superiore a quello di accesso alla cache.**
2. **In un breve lasso di tempo, una (piccola) parte dei dati viene richiesta più volte.**

Come funziona

Nel caso delle CPU cache:
Quando è necessario l'accesso ad un dato, questo dato viene prima cercato nella cache. Se è presente e valido, viene utilizzato. Viceversa, viene recuperato dalla memoria principale(operazione che richiede un tempo 1000 volte maggiore) e memorizzato nella cache, nel caso possa servire successivamente.

Nel caso in cui siano presenti più livelli di cache, il dato verrà cercato prima in quella di primo livello, poi in quella di secondo livello e infine in quella di terzo livello. Solo se tutte queste ricerche hanno dato esito negativo verrà cercato nella RAM.

Per poter fare spazio a nuovi dati, la cache generalmente deve eliminare parte del suo contenuto. L'euristica(la strategia ovvero l'algoritmo) che utilizza per scegliere quale dato "sfrattare" dalla cache(sovrascrivere) è **chiamata politica di rimpiazzamento**.
Il problema fondamentale di ogni politica di rimpiazzamento è quello di dover **predire il dato della cache che verrà richiesto nel futuro con minor probabilità**. Predire il futuro è difficile, soprattutto per le cache hardware che devono sfruttare regole facilmente implementabili in circuiteria, perciò esistono una serie di politiche di rimpiazzamento e nessuna di esse può essere ritenuta perfetta.
Una delle più popolari, la **LRU** (dall'inglese **Least Recently Used**, cioè usato meno recentemente), rimpiazza, appunto, il dato al quale si è fatto accesso meno recentemente.

Perché funziona: **Principio di Località** (dei riferimenti in memoria)

Una delle principali proprietà dei programmi è quella della **località**, tale proprietà può essere riassunta nei seguenti punti:

1. **I Programmi tendono a riusare i dati e le istruzioni che hanno usato di recente**
2. Una regola empirica largamente verificata asserisce che **un programma in media trascorre il 90% del suo tempo di esecuzione in una porzione di codice che rappresenta circa il 10% dell'intero programma (la stessa regola vale per i riferimenti sui dati)**
3. Molte istruzioni, quindi, in aree ben localizzate di un programma vengono eseguite ripetutamente in un determinato periodo, e si accede al resto del programma relativamente di rado

In particolare il Principio di Località può essere suddiviso in 2 principi:

Principio di Località Temporale

Vi è un'elevata probabilità che un'istruzione eseguita di recente venga eseguita di nuovo entro breve tempo.

Principio di Località Spaziale

Vi è un'elevata probabilità che istruzioni "vicine"(ovvero aventi indirizzi di memoria contigui) ad un'istruzione eseguita di recente siano anch'esse eseguite nel prossimo futuro .

Tutto ciò porta alle seguenti conclusioni:

Basandosi sulle ultime istruzioni eseguite da un programma è possibile predire con ragionevole accuratezza quali istruzioni e quali dati del programma verranno utilizzati nel prossimo futuro.

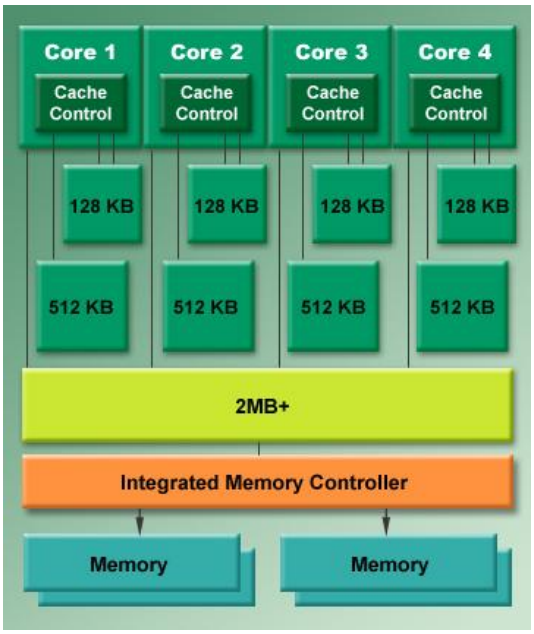
Inoltre:

Il **Principio di Località Temporale implica che conviene:** portare un elemento (istruzioni o dati) nella cache quando viene richiesto per la prima volta, in modo che rimanga a disposizione nel caso di una nuova richiesta.

Il **Principio di Località Spaziale implica che conviene:** trasferire i dati a blocchi anziché singolarmente

Coerenza Cache-RAM

Nel caso in cui la copia del dato presente nella Cache venga modificata, è importante che il dato originale sulla RAM venga aggiornato in maniera da riflettere tale modifica, ovvero che, come si dice in gergo informatico, venga mantenuta la "coerenza" dei dati presenti in RAM e nella Cache.
Lo stesso problema si pone quando un dato(presente nella Cache) è modificato nella RAM, in questo caso è la Cache a dover essere aggiornata.
Nei sistemi *multi-core* la situzione si fa ancora più complessa, in quanto la cache L3,la RAM, ecc… sono condivise tra le varie CPU. Ogni modifica effettuata comporta un aggiornamento delle varie cache "locali".



Phenom X4 9700, 9600 e 9500: le prime CPU quad core AMD

Ognuno dei ***quattro core*** possiede una cache L2 di 512 kB. In più, tutti i ***core*** possono accedere agli stessi dati tramite la cache L3 di 2 MB. **La cache L3 funziona anche come buffer** di scrittura per la memoria di sistema.

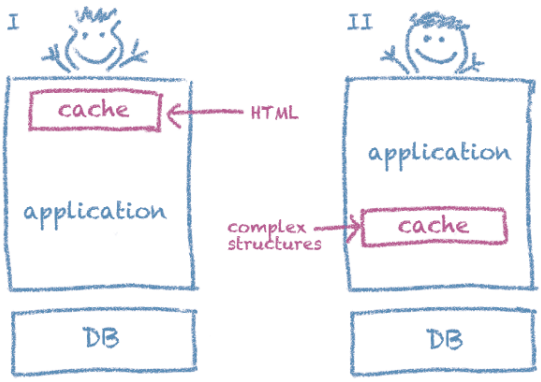
Per approfondire vedi il sito: http://www.tomshw.it/cont/articolo/1-2-3-livelli-cache/22766/3.html

Intel Nehalem: il Core i7 arriva nei notebook

Cache software

Esistono moltissimi esempi di applicazione *software* del concetto di cache(abbiamo già citato la cache del browser e la memoria virtuale). Di seguito viene mostrato uno (scherzoso) schema che evidenzia il ruolo assolto da due tipi di cache nel ridurre il tempo necessario ad un server web per inviare al client(il browser es. Google Chrome) la pagina web richiesta.
Il principio è sempre lo stesso: conservare i dati richiesti più frequentemente in maniera da poterli fornire immediatamente in caso di richiesta. La cache però offre un vantaggio solo nel caso in cui si effettui la stessa

richiesta più volte. Queste cache software non sono altro che aree di memoria dove vengono caricati i dati. In genere file o strutture dati(come i vettori che vedremo più avanti).



Esempi di cache software: lo schema mostrato suggerisce 2 possibili utilizzi delle cache software per migliorare i tempi di risposta di un server. Nella prima figura si suggerisce di memorizzare le pagine web più richieste, mentre nel secondo caso di memorizzare i dati maggiormente richiesti in maniera da evitare di doverli prelevare dal Database ogni volta.

Cache del Disco

Ad esempio esiste una cosiddetta **disk cache(cache del disco)** che è una piccola memoria RAM all'interno dell'hard-disk oppure una parte della Memoria Centrale, nella quale vengono memorizzati i settori del disco **logicamente contigui** a quello richiesto(i file sono formati da una serie di blocchi di memoria, non sempre questi blocchi di memoria "logicamente" contigui sono memorizzati in settori adiacenti). Quando si accede in lettura al disco, nel caso i dati richiesti siano presenti nella cache si evita lo spostamento della testina di lettura del disco stesso, velocizzando il reperimento dell'informazione e contribuendo a ridurre l'usura del disco stesso.
In sostanza si cerca di anticipare le richieste, memorizzando in anticipo le informazioni che potrebbero essere richieste a breve termine.

Non solo Cache…

Per evitare che la CPU rimanga inattiva mentre i dati richiesti vengono prelevati dalla RAM o dall'Hard-disk o mentre vengono eseguite operazione di I/O, la CPU può:

- **eseguire le istruzioni che seguono quella corrente**(bloccata in attesa dei dati richiesti), **se indipendenti da essa**,
- **eseguire le istruzioni di un altro programma**(la CPU viene riservata a turno ad ogni programma per un numero di millisecondi che dipende dal programma. Pertanto la CPU esegue solo un gruppo di istruzioni di un singolo programma (a seconda del tempo che è stato concesso al programma stesso) poi passa ad eseguire istruzioni di un altro programma e così via, fino a quando non tocca di nuovo al programma in questione. E' questa alternanza che permette di eseguire tante applicazioni insieme ed è il Sistema Operativo a gestirla.

Inoltre vi sono altri stratagemmi adottati per ridurre anche il tempo necessario per prelevare i dati dalla cache: quello classico è di **prelevare i dati in anticipo ovvero prima che vengano richiesti**(*pre-fetching*).

BUFFER (Memoria Tampone)

Esistono anche memorie che svolgono una funzione per certi versi analoga a quella della Cache anche se con alcune differenze. Tali memorie vengono dette **buffer.** In genere sono utilizzate per trasferire i dati tra processore(o RAM) e periferiche, ma designano in generale:

> *qualsiasi memoria temporanea utilizzata per l'archiviazione temporanea di dati, che non possono esser elaborati o trasmessi immediatamente.*

Queste memorie possono essere **implementate a livello hardware o software** utilizzando ad esempio una parte della RAM che conserverà i dati da trasferire fino a quando la periferica non li avrà letti tutti o la CPU non li avrà elaborati.
Un buffer può dunque, come la Cache, essere un dispositivo fisico(hardware) distinto o semplicemente un'area di memoria temporanea.

> *I buffer sono utilizzati per operazioni di I/O, per trasferire dati tra dispositivi funzionanti a velocità diverse, per velocizzare l'esecuzione di alcune operazioni, ecc…*

Non vi è, a differenza della cache, **alcun meccanismo di selezione basato sulla frequenza di accesso ai dati**(anche se potrebbero esservi altri criteri di selezione basati sull'importanza dei dati…oltre che sulla pura e semplice contiguità logica(es. un dato viene dopo un altro perché nel file caricato in RAM si trova all' indirizzo di memoria successivo)).

Un esempio tipico di buffer (o memoria tampone) è quello della stampante che permette un'efficiente trasferimento dei dati dal processore verso la periferica. In pratica, il processore invia in modo veloce una serie di dati al buffer (in realtà non lo fa direttamente ma delega un altro dispositivo a leggere i dati dalla RAM e inviarli al buffer**).** La memoria tampone si occupa di fornirli alla corretta velocità ai meccanismi di stampa, mentre il processore è libero di compiere altre operazioni; quando il buffer è vuoto, la stampante avvisa il processore e altri dati sono caricati nel buffer fino al completamento dell'operazione di stampa.

Il buffer, in questo caso, consente di adeguare la velocità di "output" a quella del dispositivo di Output e quella di "input" a quella del dispositivo di Input conservando i dati in attesa che questi possano essere trasferiti/elaborati.

Spesso i buffer servono per trasferire i dati a blocchi anziché singolarmente(essi accumulano tutti i dati inseriti fino a quando, raggiunta una certa soglia, vengo trasferiti in blocco), **riducendo, in tal modo, i tempi di trasferimento** in quanto il tempo necessario a trasferire un blocco è all'incirca lo stesso di quello necessario a trasferire il dato singolarmente.
Anche nella cache vi è un vantaggio nel trasferire i dati a blocchi, ma questo è del tutto secondario rispetto a quello che si ricava riducendo il numero di accessi sul dispositivo più lento(RAM, hard-disk, ecc…).

In ambito software i buffer sono spesso aree di memoria condivise, che servono a consentire lo scambio di dati tra due "processi"(applicazioni in esecuzione). Gli "Appunti" di Windows sono un esempio di questo tipo.

In altri casi servono a contenere i dati da elaborare prima di inviarli in output, per esempio quando è necessario riordinare i dati di input.

DIFFERENZE CACHE-BUFFER

Da quanto detto si possono individuare alcune differenze fondamentali:

- i buffer semplicemente immagazzinano i dati in attesa che questi vengano prelevati/elaborati, *senza* conservare le statistiche di accesso (in maniera da individuare i dati utilizzati più frequentemente). In pratica **la Cache cerca di prevedere le richieste, il buffer no!**

- **la cache cerca di minimizzare gli accessi al dispositivo di archiviazione più lento**(la RAM(per le cache di I,II,III livello), l'Hard-disk per la cache del disco)
- **i buffer forniscono un vantaggio anche se il dato è letto o scritto una sola volta, la cache al contrario è utile solo se i dati sono letti/scritti più volte**
- **il buffer è "visibile" al programmatore, al contrario della cache**

Questi due meccanismi **non sono mutuamente esclusivi** e spesso sono combinati(come si vede nel processore dell'AMD).

Dimensione del buffer e Trabocco (Buffer Overflow)

Anche i buffer come le cache hanno una dimensione limitata per cui è necessario, come visto per la stampante, trasferirvi i dati a blocchi tenendo conto della dimensione del buffer stesso.
L'utilizzo di buffer "software" a livello di programmazione è molto comune soprattutto quando si scrivono applicazioni che devono inviare i dati attraverso Internet. In tal caso fare attenzione alla dimensione del buffer è di vitale importanza per prevenire possibili attacchi del tipo a "Buffer Overflow", per garantire una corretta visione del contenuto in streaming e un corretto trasferimento dei dati.

REGISTRI

Una funzione analoga viene svolta dai **Registri**(ovvero quella di **permettere un rapido accesso ai dati**), ma in questo caso i **dati coinvolti sono solo quelli che vengono utilizzati per eseguire l'istruzione corrente**(operandi, risultato finale e risultati parziali, indirizzo di memoria dell'istruzione successiva(Program Counter), istruzione corrente(per la decodifica), ecc…).

IN SINTESI

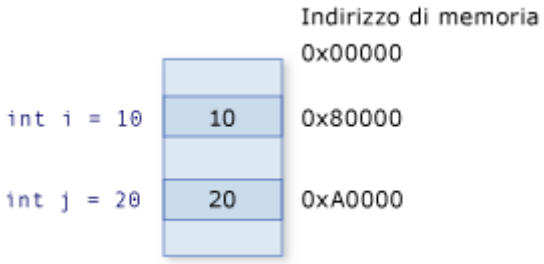
Riassumendo, potremmo pertanto dire che la RAM, le varie memorie cache e i registri della CPU non sono altro che memorie temporanee che servono a ridurre il tempo di elaborazione aumentando la velocità di accesso ai dati stessi.
Purtroppo per motivi tecnici, tali memorie presentano aspetti negativi quali la volatilità(ovvero sono incapaci di conservare le informazioni ivi contenute in assenza di alimentazione elettrica) e possiedono una capacità limitata da fattori collegati alla tecnologia utilizzata e dall'elevato costo di produzione.
Da qui scaturisce l'esigenza di utilizzare una così ampia varietà di dispositivi di memorizzazione.
Trasferendo dati da una memoria all'altra e risalendo la gerarchia prima vista è possibile ridurre il tempo di accesso alle informazioni e conseguentemente quello di elaborazione.

In effetti la gestione del sistema di memoria è molto più complessa di quella brevemente accennata qui.
E' bene infatti ricordare che a causa della limitata capacità, solo un sottoinsieme di dati/istruzioni potranno essere memorizzati in RAM e di qui nelle varie Cache.
Sorge pertanto il problema di selezionare cosa vada mantenuto in queste memorie temporanee per poter sfruttare al massimo la potenza di calcolo della CPU. Tale scelta è effettuata sulla base di un opportuno algoritmo.
Il problema è complicato dal fatto che gli attuali sistemi di calcolo sono multi-tasking ovvero più programmi sono "virtualmente" in esecuzione in uno stesso momento, ognuno con le proprie esigenze in termini di tempo di esecuzione, utilizzo delle risorse condivise(hardware/software) e ovviamente di memoria.
I sistemi operativi attualmente esistenti includono pertanto dei moduli specifici detti *Gestori di Memoria* per determinare quanta memoria assegnare ad ogni "processo" e che cosa vada caricato in memoria.

Contenuto e struttura della memoria

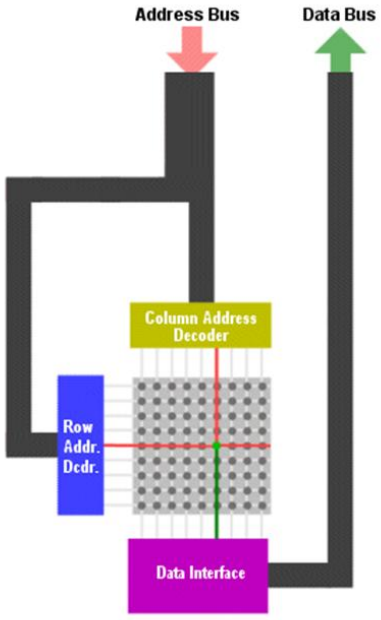
Le informazioni sono registrate in memoria in formato binario, secondo codifiche specifiche che variano in base al tipo di contenuto(un numero reale è memorizzato in maniera diversa da un numero intero o da una sequenza di caratteri!).
Tutte le memorie a livello fisico sono realizzate attraverso supporti che consentono di distinguere 2 "stati" fisici che sia il verso del campo magnetico locale o la riflettività della superfice.
A livello "logico", ovvero astraendo da tutti i dettagli relativi alle specifiche implementazioni, la struttura della memoria è molto semplice.

"Logicamente", la memoria può essere immaginata come una sequenza di celle o locazioni di memoria, ognuna delle quali può contenere una parola di memoria(*word*) costituita da uno o più byte.
Ogni locazione di memoria è associata ad un indirizzo di memoria, ovvero ad un numero che specifica la sua posizione nella sequenza.
E' attraverso questo numero(ovviamente espresso in binario) che i dati contenuti nella memoria possono essere letti o scritti.



Struttura Logica della memoria(gli indirizzi sono in formato esadecimale)

La struttura della memoria reale invece varia a seconda del tipo di memoria considerato.
Le memorie elettroniche hanno in genere una struttura a griglia in cui l'indirizzo viene in due parti, una serve ad individuare la colonna e l'altra la riga in corrispondenza delle quali si trova il dispositivo fisico che contiene i dati.



Nell' *hard-disk*, le informazioni sono memorizzate su dischi magnetici organizzati in una serie di settori e tracce, mentre nei dischi ottici vi è un'unica spirale. I bit, in questo caso, sono memorizzati variando le proprietà fisiche del materiale (magnetiche, ottiche) e questo ovviamente influisce anche sul tempo per cui possono essere conservati senza alterazioni, oltre che sulla densità dei supporti di memoria che è strettamente collegata ai progressi tecnologici.

I ruolo dei Bus

Il fatto che le informazioni debbano essere trasferite da una memoria all'altra per arrivare ai registri della CPU e poi di nuovo alle memorie di massa o ai buffer delle periferiche di output, fa sì che ai fini delle prestazioni del sistema sia cruciale la velocità dei vari Bus, ovvero dei canali di comunicazione che collegano le varie parti del sistema di elaborazione, sia interne che esterne.
Le ultime innovazioni tecnologiche in questo campo hanno reso l'architettura di un calcolatore molto simile a quella di una rete informatica, in cui vi sono dei controller che smistano i dati in maniera molto simile agli switch di rete.