Gerarchie di memoria:

I sistemi di memorizzazione sono organizzati gerarchicamente secondo: velocità, costo e volatilità.

- **Memoria principale** (RAM: random access memory): la memoria che la CPU può accedere direttamente.
- **Memoria secondaria** (Disk): estensione della memoria principale che fornisce una memoria non volatile.

Tecnologie:

SRAM (Static Random Access Memory):

Unità che memorizza un gran numero di parole in un insieme di flip-flop, opportunamente connessi, mediante un sistema di indirizzamento e trasferimento (lettura/scrittura) di parole. I dati memorizzati vengono mantenuti per un tempo arbitrario (finché c'è alimentazione).
Tempo di accesso tipico: circa 10-30 ns.

DRAM (Dynamic Random Access Memory):

I bit vengono espressi sotto forma di stato di carica di un transistor, che va rinfrescato periodicamente con un circuito di controllo integrato sul componente di memoria.
La DRAM ha un costo inferiore ma un tempo di accesso maggiore.
Tempo di accesso tipico: circa 60-70 ns.

HDD (hard disk):

Una memoria persistente con dischi magnetici e testine di lettura/scrittura. Il costo per bit è molto più basso con grande capacità di memorizzazione.
Tempo di accesso tipico: circa 10 ms.

SDD/Flash (solid state drive):

Tecnologia basata su celle di semiconduttori che non richiede spostamenti meccanici e che mantiene lo stato anche senza alimentazione.
Tempo di accesso tipico: circa 0,1 ms.

Livello superiore ed inferiore di memoria:

Ogni coppia di livelli in una gerarchia è formata da un livello superiore, accesso più rapido, capacitò minore, e da un livello inferiore, accesso più lento, capacità maggiore. L'unità di memorizzazione nella cache viene detta linea o blocco di cache.

Principio di località:

Durante l'esecuzione di un programma solitamente si passa da una località all'altra.

Località temporale:

Quando si effettuano altre operazioni sullo stesso elemento di memoria.

Località spaziale:

Quando si effettuano operazioni su elementi vicini all'elemento di memoria.

Cache:

La cache è piccola e veloce e non può immagazzinare tutti i dati di un computer.

Hit/miss in lettura:

In lettura si cerca prima il dato nel livello superiore:

- **Cache hit**: se il riferimento è già presente nella cache, si procede con la lettura della linea di cache.
- **Cache miss**: altrimenti occorre trasferire i dati dal livello inferiore ed inserire una nuova linea di cache, eventualmente sovrascrivendo altre linee di cache.

Hit/miss in scrittura:

La scrittura può generare una cache miss se il dato non è presente in cache. C'è anche da considerare la consistenza tra dati in cache e dati nel livello inferiore.
Si possono usare diverse strategie:

- **Write through**: Scrivere in cache e nella RAM.
- **Write buffer**: Scrivere il dato in cache, un controller scrive il dato nella RAM quando ritiene che sia opportuno.
- **Write back**: Scrivere solo nella cache, quando il processore fa un cache miss un controller copia il blocco su cui deve scrivere dalla RAM alla cache, il blocco su cui ha fatto cache miss viene copiato dalla cache alla RAM.

RAM come cache per pagine di memoria di processi:

Lo spazio di memoria è suddiviso tra i diversi programmi in esecuzione contenuti dalla RAM. La RAM funge da cache per i programmi in esecuzione: lo spazio di indirizzamento virtuale (che è più grande dello spazio allocato nella RAM) è diviso in pagine caricate quando necessario.

Shared Memory MIMD:

Questo tipo di architettura è alla base del multicore programming. I processori hanno diversi livelli di cache e condividono la memoria principale e buffer di I/O.
Le tecniche di threading (hardware e software) vengono utilizzate per permettere a diversi programmi di condividere dati in memoria principale; in questo caso la gestione della cache diventa complessa e richiede la garanzia della coerenza tra i valori mantenuti nelle cache dei diversi processori e quelli in memoria.

Context switching:

Quando cambio i valori nei registri per eseguire due processi diversi "contemporaneamente".

Multithreading:

I thread sono i processori virtuali.

Hardware:

Un core contiene due (o più) thread, così da eseguire due (o più) flussi di esecuzione dello stesso processo in un core. I processori superscalari con Instruction Level Parallelism (ILP) e Simultaneous multithreading (SMT) supportano hardware multithreading. In caso di cache miss (stallo) è possibile passare all'esecuzione di un altro thread sullo stesso core.

Software:

Il sistema operativo sfrutta il fatto che un core ha più thread per dare la possibilità al programmatore di scrivere un programma con più flussi di esecuzione (insiemi di istruzioni) che possono essere eseguiti contemporaneamente attraverso i thread del core. Per aumentare ancora di più le performance si può utilizzare il thread context switching. L'ordine delle istruzioni da eseguire viene definito dallo scheduler del sistema operativo. Se i dati su cui operano i diversi flussi di istruzioni dello stesso programma sono indipendenti tra loro, ogni thread può essere potenzialmente eseguito in parallelo con altri thread sullo stesso processore o su diversi core.

Intel Hyper-Threading:

Tecnologia di SMT. Si tratta di un tentativo di creare un processore di transizione tra i tradizionali single core e i successivi dual core, duplicando solo alcune aree sensibili del singolo core, così da gestire due thread in contemporanea con un singolo core.

Multithreading in C++:

```
#include<string>
#include<iostream>
#include<thread>
using namespace std;
// The function we want to execute on the new thread.
void task1(string msg){
    cout << "task1 says: " << msg;
}
Int main(){
    // Constructs the new thread and runs it. Does not block
execution.
thread t1(task1, "Hello");
// Do other things...
/* Makes the main thread wait for the new thread to finish
execution, therefore blocks its own execution. */
t1.join();
}
```

Definizioni:

Programma:
Un insieme di istruzioni che fanno una certa cosa.

Processi:
Un processo è un programma in esecuzione.

Compile-time:
- Compilazione: il codice scritto nel linguaggio ad alto livello viene tradotto in codice in linguaggio macchina.
- Linking: il codice delle librerie importate viene incluso ne codice oggetto.

Loading-time:
Il codice binario viene caricato in memoria principale.

Execution-time:
Il codice oggetto viene eseguito dalla CPU (che può inviare richieste di lettura scrittura alla memoria principale).