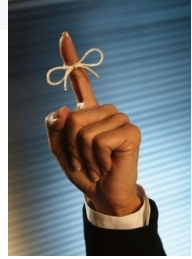


Memorie

Caratteristiche principali



- Locazione: processore, interna (principale), esterna (secondaria)
- Capacità: dimensione parola, numero di parole
- Unità di trasferimento: parola, blocco
- Metodo di accesso: sequenziale, diretto, casuale, associativo
- Prestazioni: tempo di accesso, tempo di ciclo, velocità trasferimento
- Modello fisico: a semiconduttore, magnetico, ottico, magnetico-ottico
- Caratteristiche fisiche: volatile/non volatile, riscrivibile/non riscrivibile
- Organizzazione

Gerarchie di memoria

Tecnologie di memoria

L'ideale sarebbe una memoria molto
ampia, molto **veloce** e molto **economica**

Tecnologia

registro

cache

SRAM

DRAM

disco

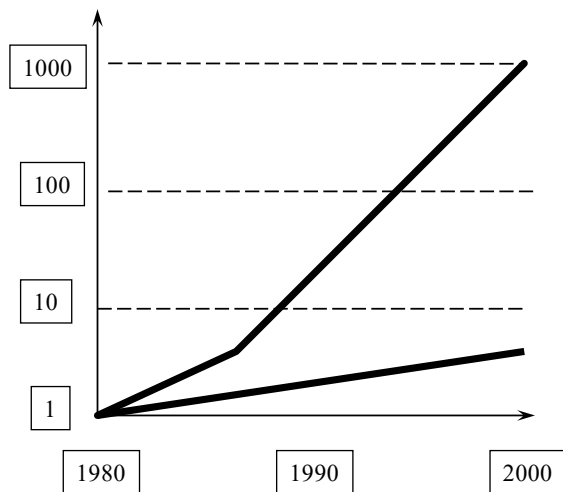
CD/DVD-ROM [meno capace di disco!]

nastro



Gerarchie di memoria

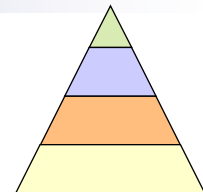
Prestazioni CPU/memoria



- Le CPU hanno avuto un aumento di prestazioni notevole, dovuto ad innovazioni tecnologiche ed **architetturali**
- Le memorie sono migliorate **solo** grazie agli avanzamenti tecnologici

Gerarchie di memoria

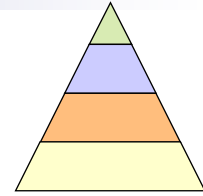
Proprietà dei programmi



- Proprietà **statiche** (*dal file sorgente*)
- Proprietà **dinamiche** (*dall'esecuzione*)
 - **Linearità** dei riferimenti
 - Gli indirizzi acceduti sono spesso consecutivi
 - **Località** dei riferimenti
 - Località **spaziale**
 - Gli accessi ad indirizzi **contigui** sono **più probabili**
 - Località **temporale**
 - La zona di accesso **più recente** è quella di permanenza **più probabile**

Gerarchie di memoria

La congettura 90/10



Un programma impiega mediamente il **90%** del suo tempo di esecuzione alle prese con un numero di istruzioni pari a circa il **10%** di tutte quelle che lo compongono.

Gerarchie di memoria

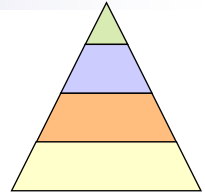
Divide et impera



- Conviene organizzare la memoria su più livelli gerarchici:
 - **Livello 1 (cache)**: molto veloce e molto costosa
⇒ dimensioni ridotte, per i dati ad accesso più probabile **[anche più livelli di cache]**
 - **Livello 2 (memoria centrale)**: molto ampia e lenta ⇒ costo contenuto, per tutti i dati del programma

Gerarchie di memoria

Organizzazione gerarchica

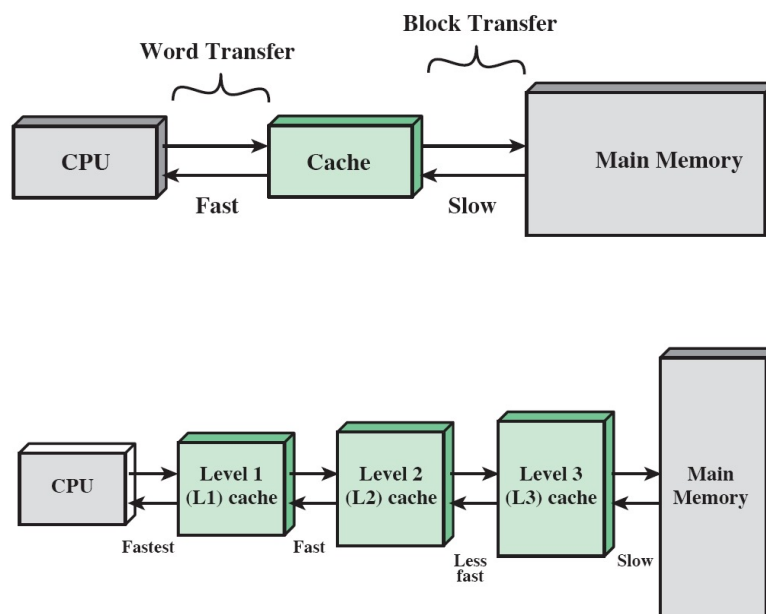


■ Memoria a livelli

- Al livello più basso (**inferiore**) stanno i “supporti di memoria” più capaci, più lenti e meno costosi
- Ai livelli più alti (**superiori**) si pongono supporti più veloci, più costosi e meno capaci
- La CPU usa direttamente il **livello più alto**
- Ogni livello inferiore deve contenere tutti i dati presenti ai livelli superiori (ed altri)

Gerarchie di memoria

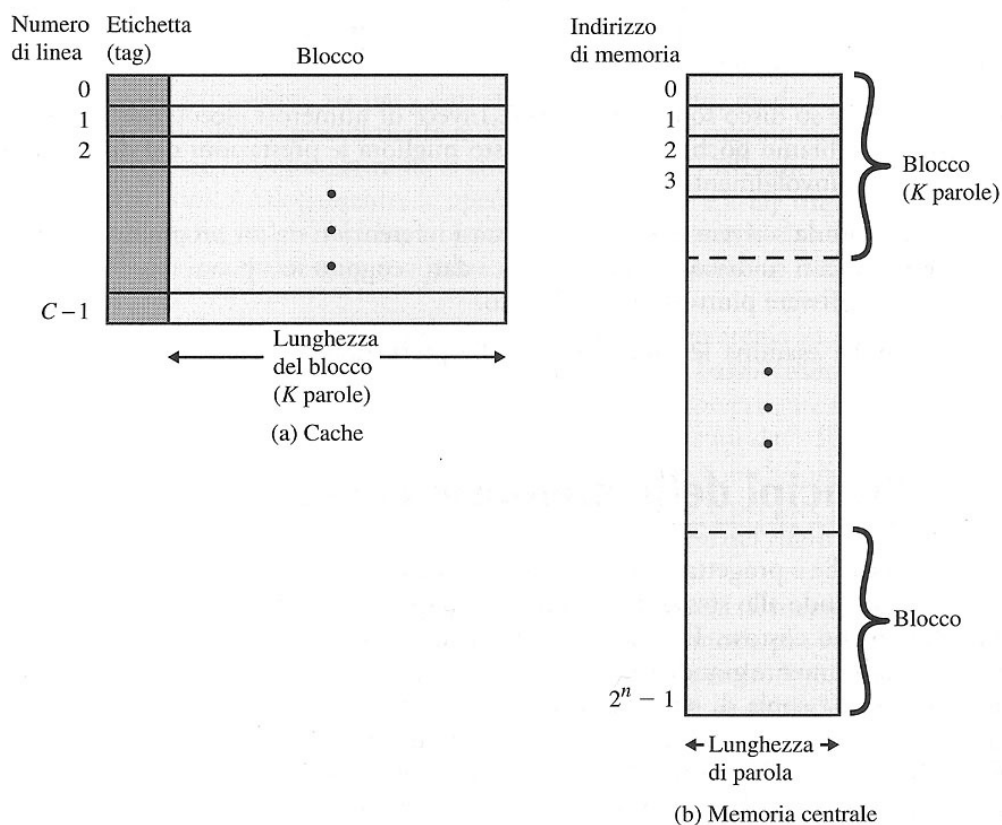
Schema concettuale

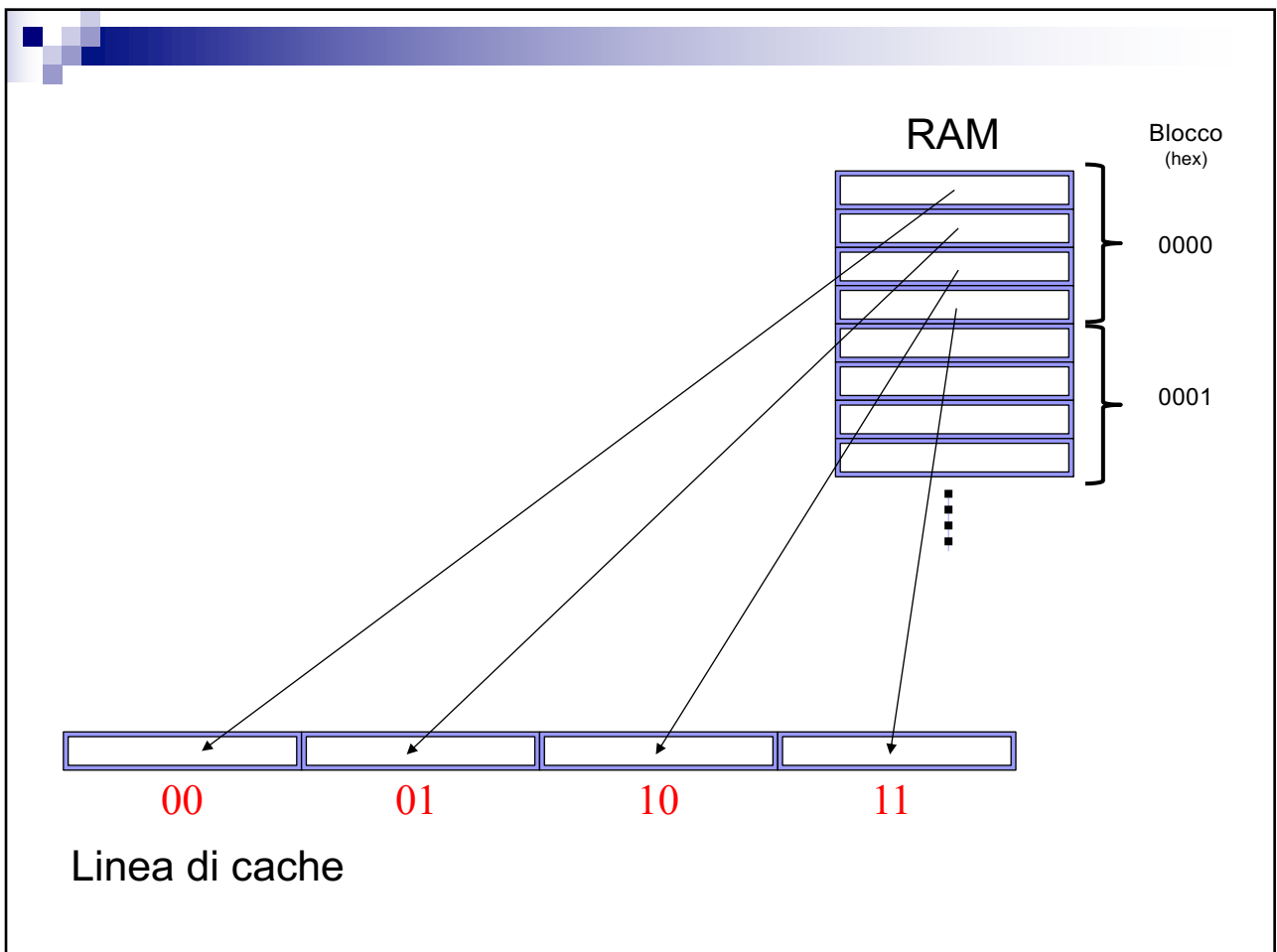
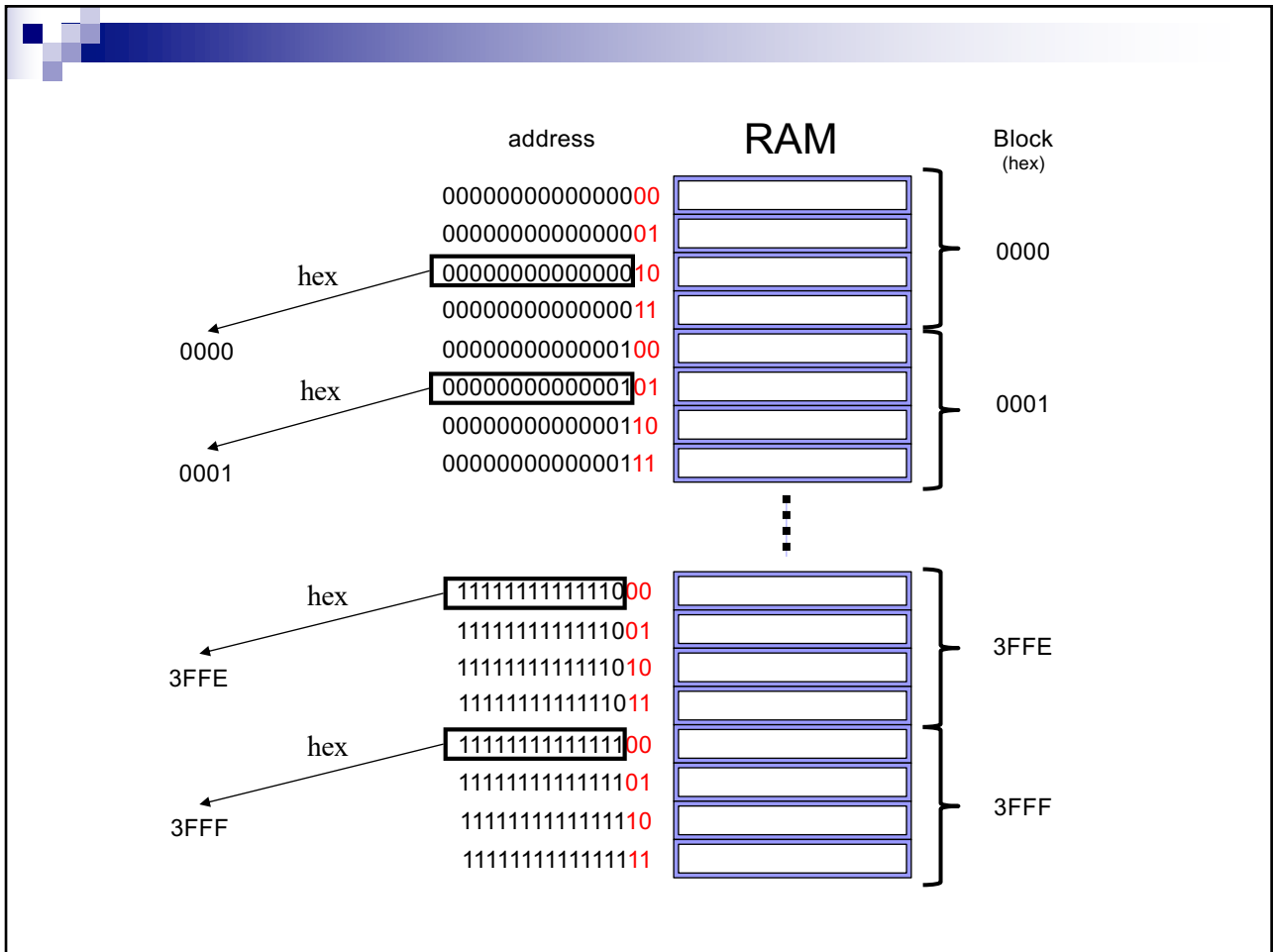


Gerarchie di memoria

Suddivisione in blocchi

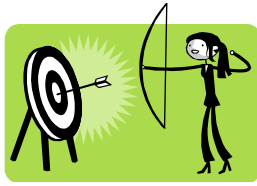
- Per realizzare un'organizzazione gerarchica conviene suddividere la memoria in **blocchi**
- La **dimensione** di un blocco è la **quantità minima indivisibile** di dati che occorre prelevare (copiare) dal livello inferiore
- L'**indirizzo** di un dato diviene l'indirizzo del blocco che lo contiene *sommato* alla posizione del dato all'interno del blocco





Gerarchie di memoria

Hit e miss



Un dato richiesto dalla CPU può essere presente in cache (**hit**) oppure mancante (**miss**)

- Un **hit**, *successo*, deve essere molto probabile (>90%) se si vuole guadagnare efficienza prestazionale
- Un **miss**, *fallimento*, richiede l'avvio di una procedura di scambio dati (**swap**) con il livello inferiore

Gerarchie di memoria

Tempo medio di accesso



T_a : Tempo medio di accesso ad un dato in memoria

$$T_a = T_h \times P_h + T_m \times (1 - P_h)$$

T_h = tempo di accesso ad un dato **presente** in cache

T_m = tempo medio di accesso ad un dato **non** in cache
(funzione della dimensione del blocco)

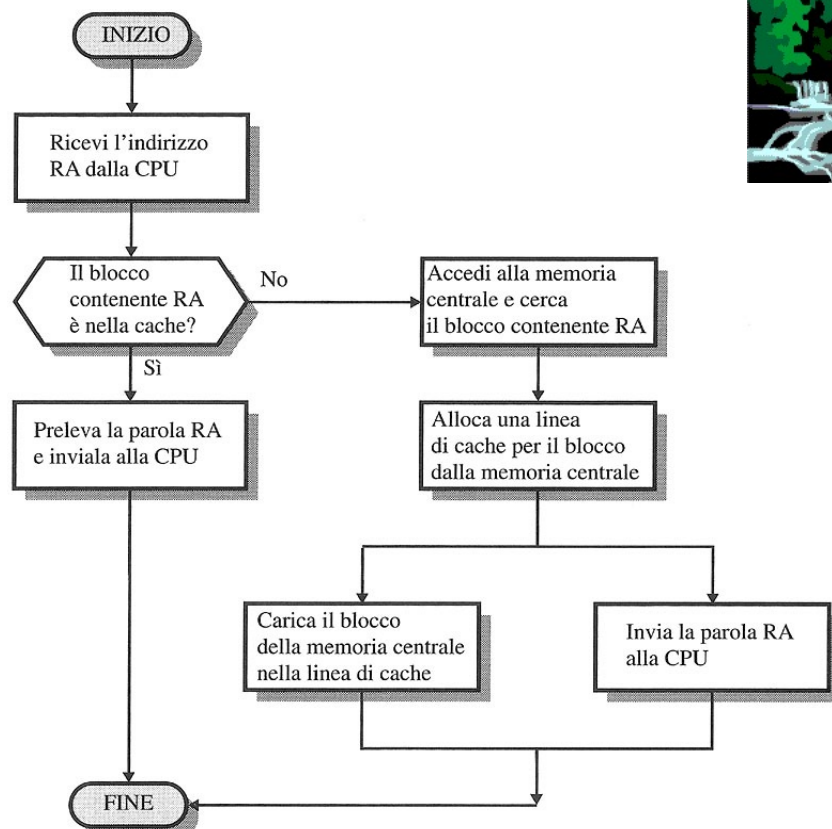
P_h = probabilità di **hit**

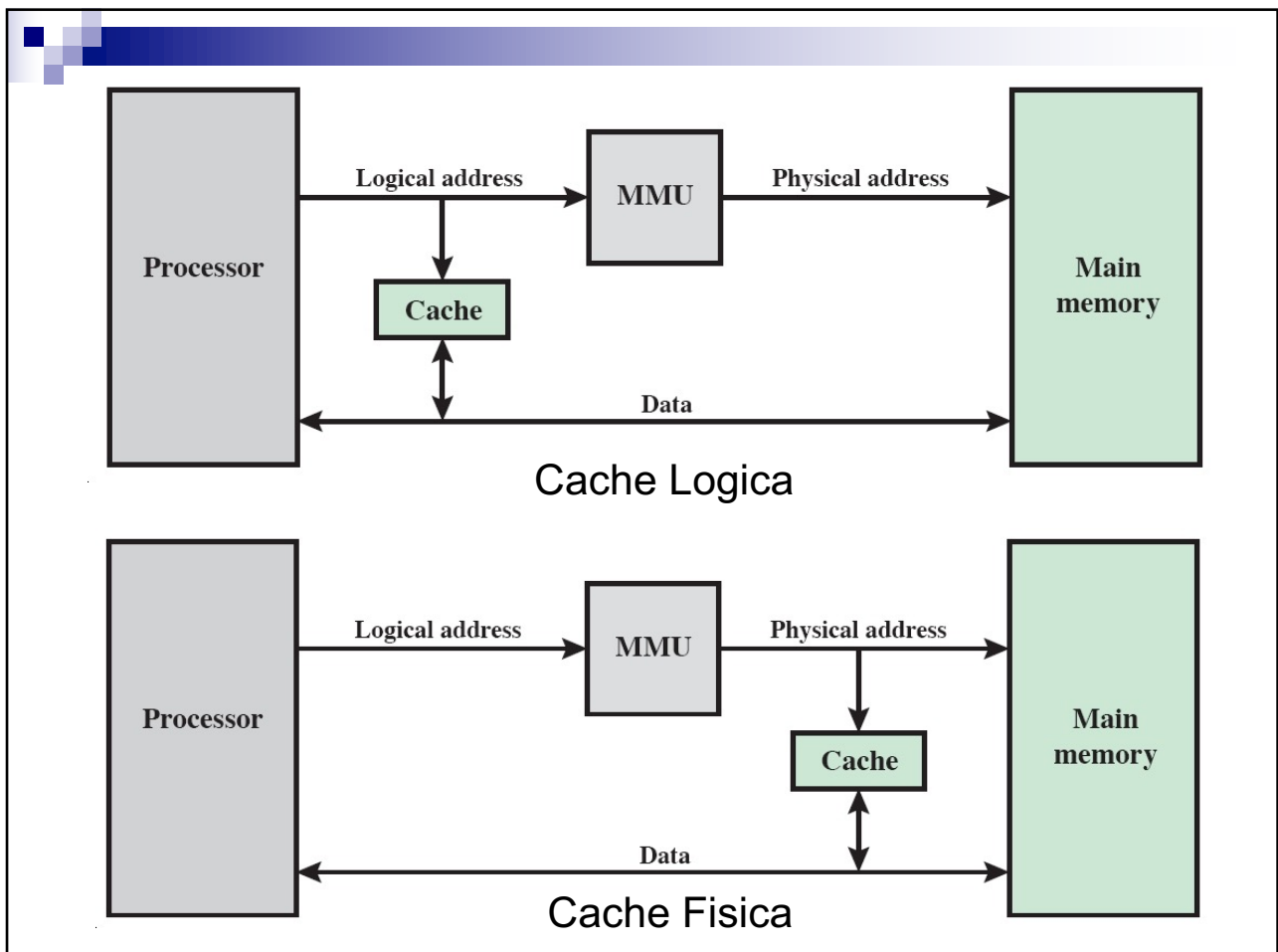
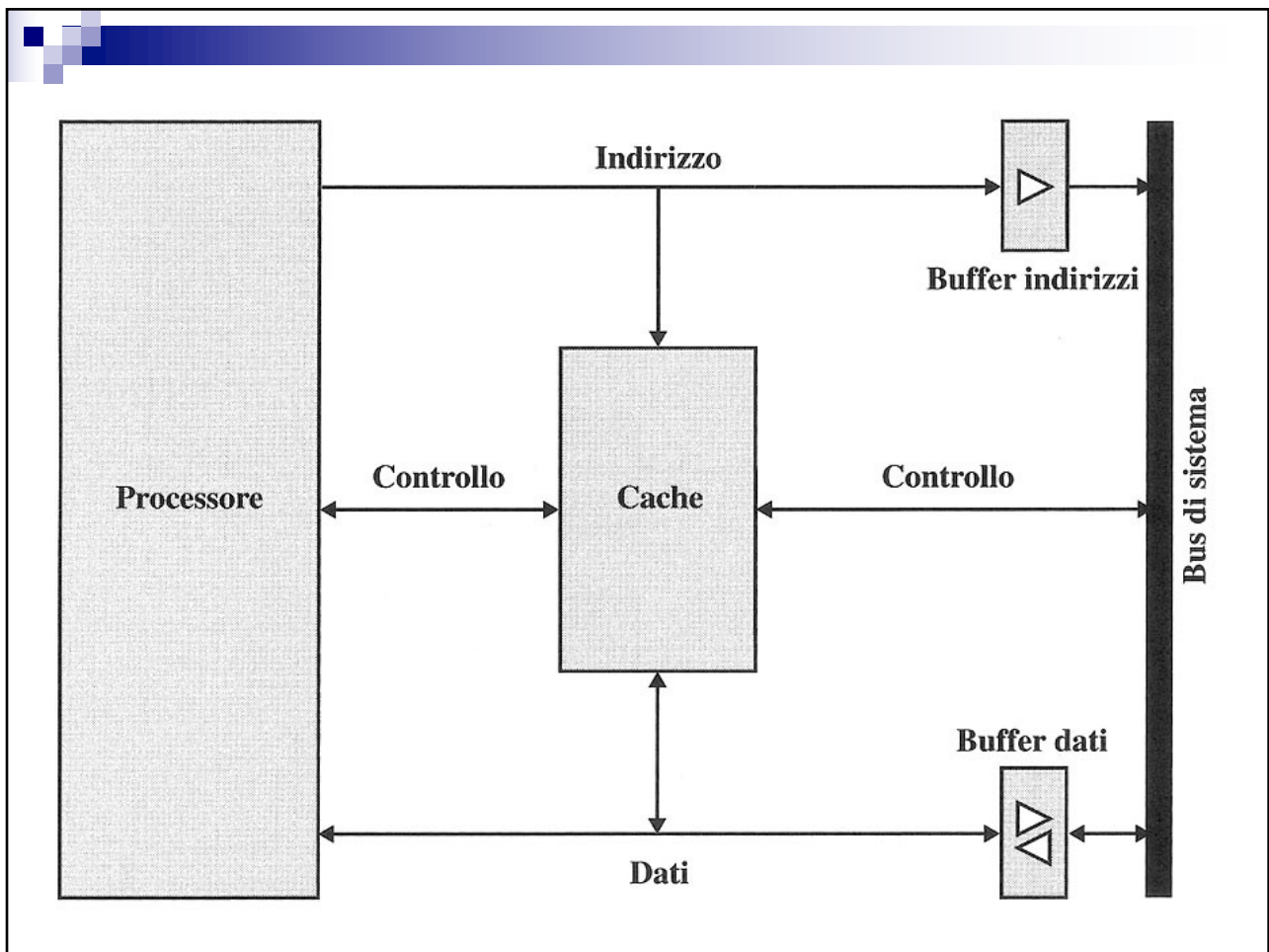
(funzione della dimensione del blocco e della politica di gestione)

Gerarchie di memoria

Tecnica generale

- Suddivisione della memoria centrale in blocchi **logici**
- Dimensionamento della cache in **multiplo** di blocchi
- Per ogni indirizzo emesso dalla CPU
 - **Hit** ⇒ Il dato richiesto viene fornito **immediatamente** alla CPU
 - **Miss** ⇒ La cache richiede il dato al livello inferiore
Il blocco contenente il dato viene posto in cache
Il dato richiesto viene fornito alla CPU





Gerarchie di memoria

Problematiche



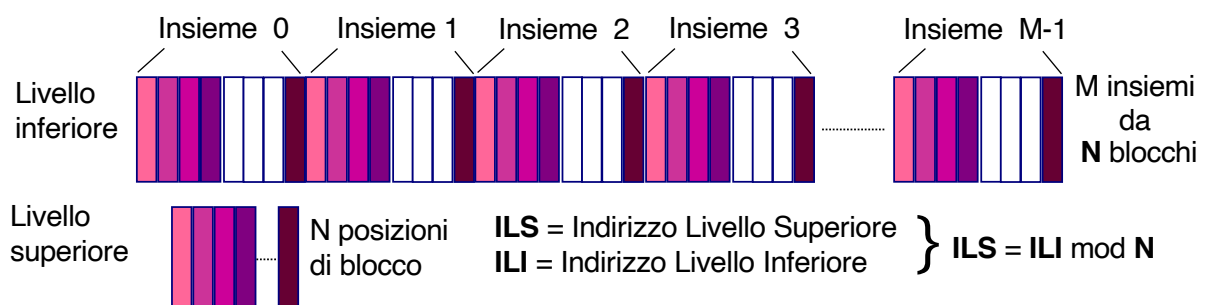
- Organizzazione della cache e tecniche di allocazione
- Individuazione di hit o miss
- Politica di rimpiazzo dei blocchi
- Congruenza dei blocchi

Gerarchie di memoria

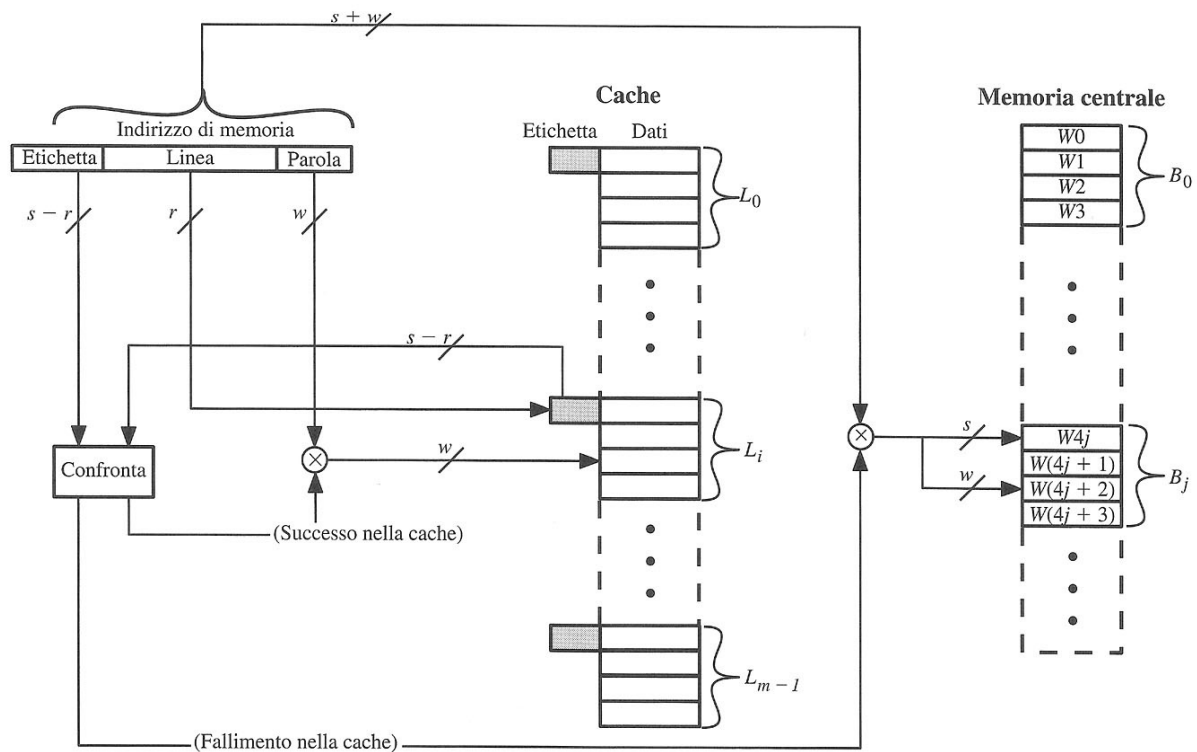
Associazione diretta

Tecnica nota come *direct mapping*

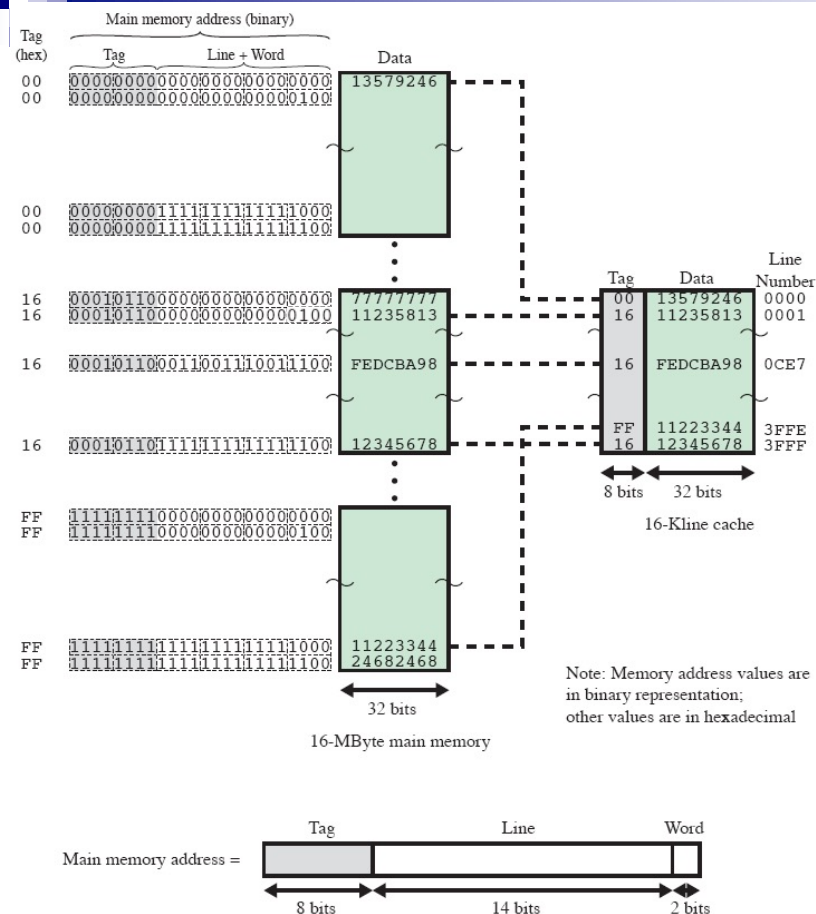
- Ogni blocco del livello inferiore può essere allocato *solo* in una specifica posizione (detta *linea* o *slot*) del livello superiore



Associazione diretta



Esempio di associazione diretta



Gerarchie di memoria

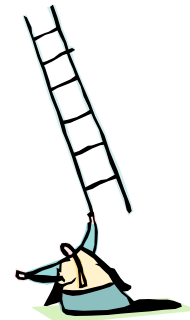
Associazione diretta

Vantaggi

- Semplicità di traduzione da indirizzo ILI (memoria) ad indirizzo ILS (cache)
- Determinazione veloce di hit o miss

Svantaggi

- Necessità di contraddistinguere il blocco presente in ILS (introduzione di un'etichetta, '**tag**')
- Swap frequenti per accesso a dati di blocchi adiacenti

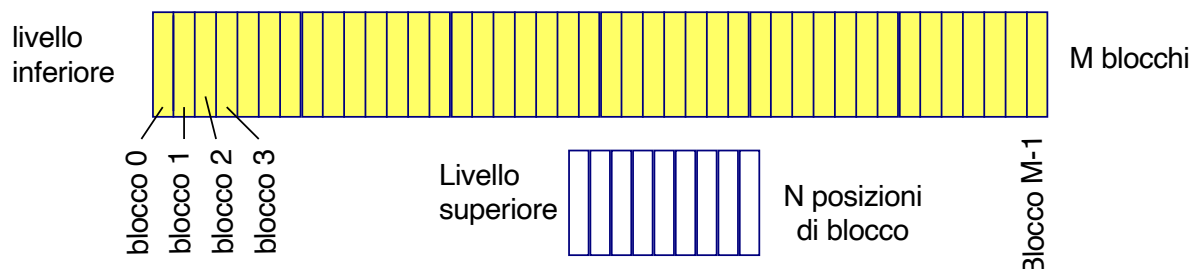


Gerarchie di memoria

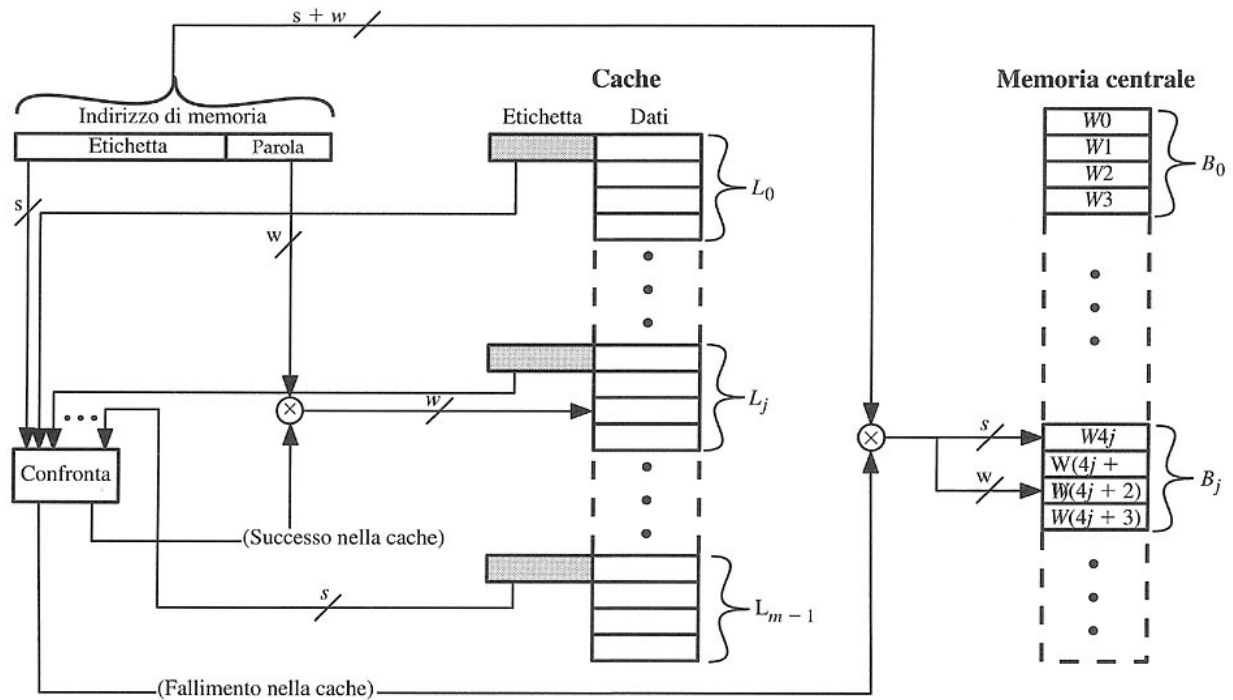
Associazione completa

Tecnica nota come *fully associative*

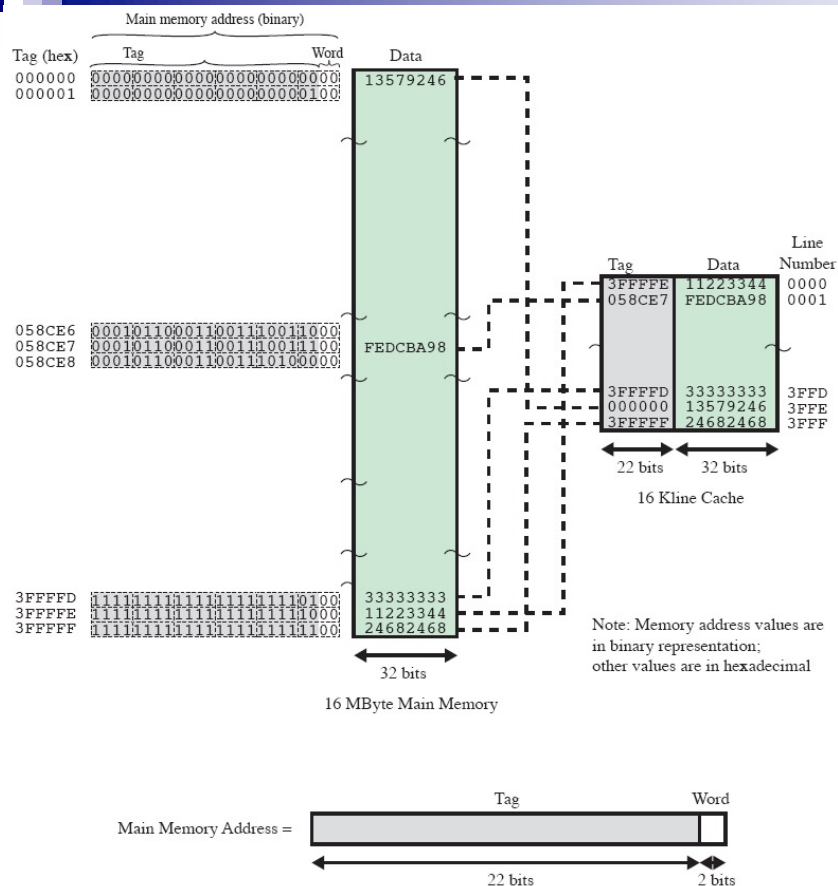
- Ogni blocco del livello inferiore può essere posto in *qualsunque* posizione del livello superiore



Associazione completa



Esempio di associazione completa



Gerarchie di memoria

Associazione completa

Alla cache capace di N blocchi viene associata una tabella di N posizioni, contenenti il numero di blocco effettivo (**tag**) in essa contenuto.

Vantaggi

- Massima efficienza di allocazione



Svantaggi

- Determinazione onerosa della corrispondenza ILS-ILI e della verifica di hit/miss

