Lezione 28

Struttura delle memorie cache

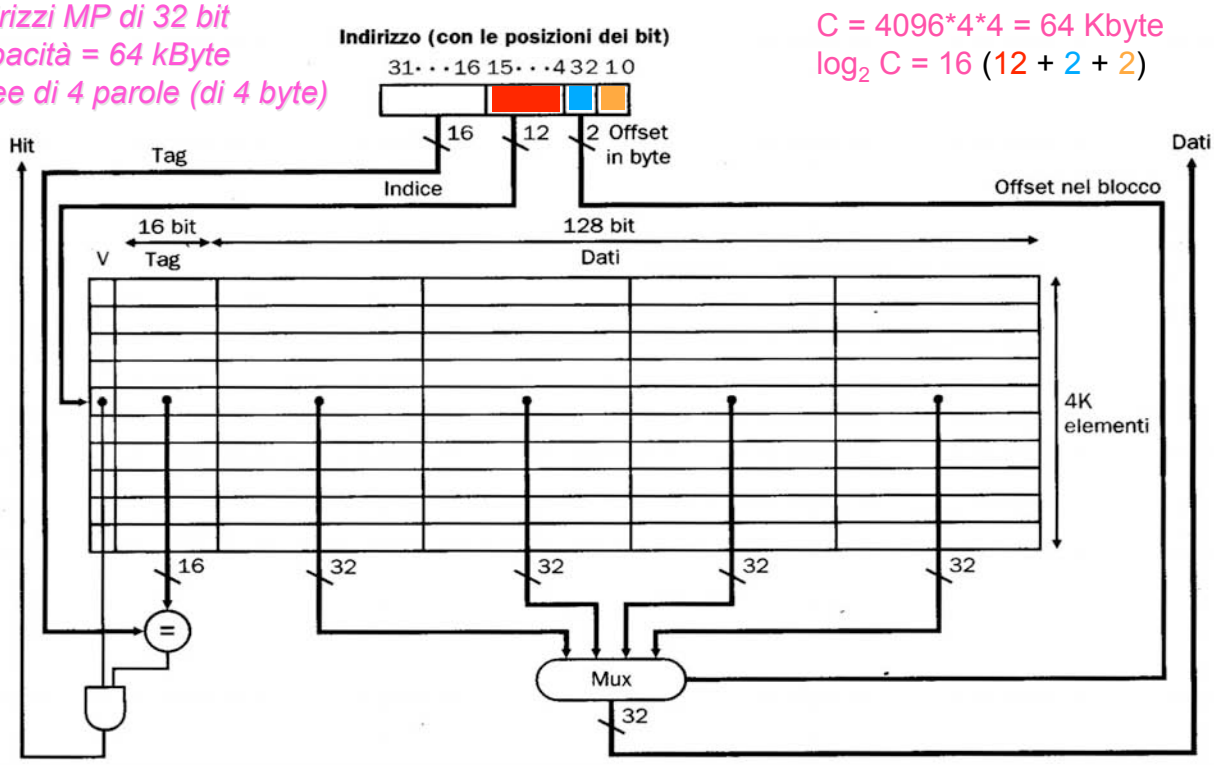
Proff. A. Borghese, F. Pedersini

Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano

Schema: cache a mappatura diretta

Indirizzi MP di 32 bit
Capacità = 64 kByte
Linee di 4 parole (di 4 byte)

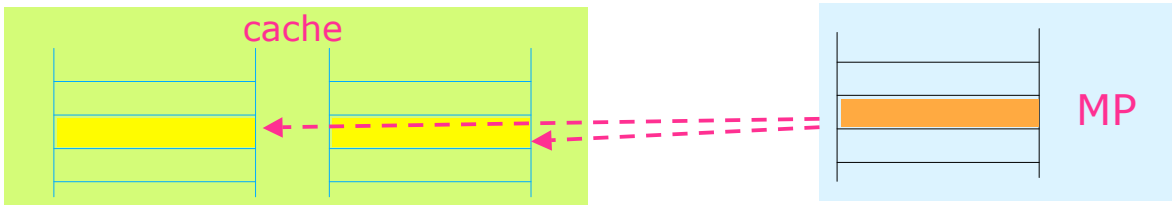
C = 4096*4*4 = 64 Kbyte
$\log_2$ C = 16 (12 + 2 + 2)



Limiti cache a mappatura diretta

- Cache a mappatura diretta:
  per ogni locazione di MP, esiste una e una sola locazione in cache.
  - Locazione determinata a priori: { ***Index*** , ***Word Offset*** , ***Byte Offset*** }.
  - SOLUZIONE SEMPLICE, MA INEFFICIENTE: potrei avere la cache semivuota, e ciononostante riscrivere linee già occupate

- Idea:
  prevedo per ogni linea, più alternative di collocazione
  - Linee multiple di cache, per ogni blocco di MP



Sommario

- Memorie cache n-associative
- Strategie di gestione della cache

Memorie n-associative

N-associative o set associative o a N vie:

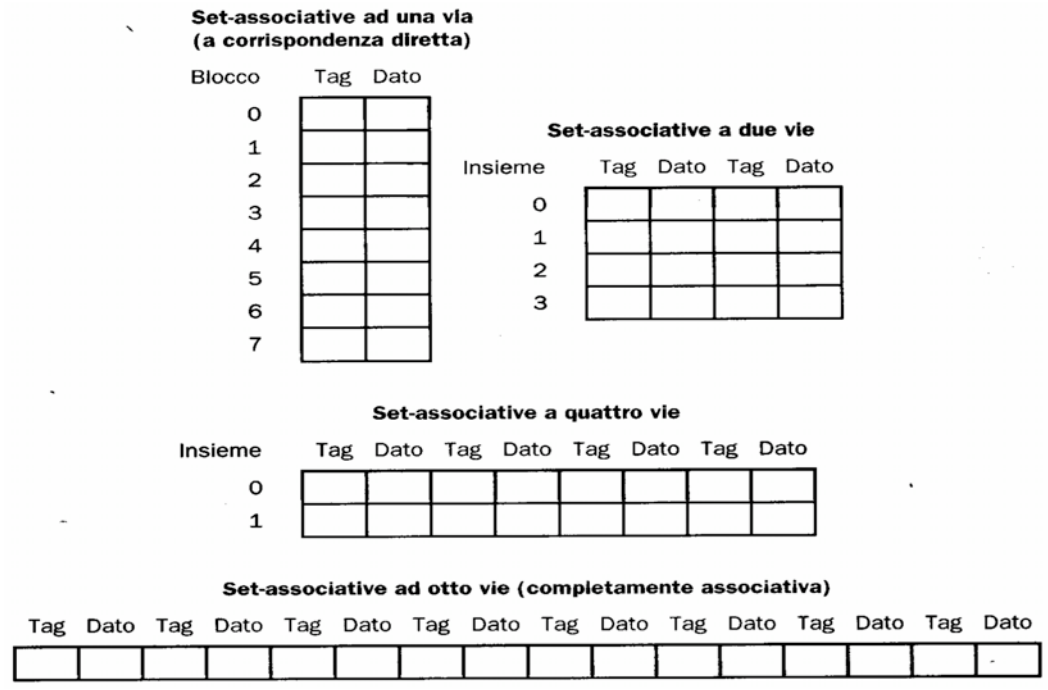
- La memoria è suddivisa in **N** insiemi, o banchi, ciascuno di **k** linee, posti in parallelo
  - ***Blocco/Linea (linea di cache):***
    #parole(byte) lette/scritte contemporaneamente in cache.
  - ***Insieme (banco):*** cache elementare (a mappatura diretta).
  - ***Capacità della cache:***

    Capacità [byte] = N.bytes/banco x N.banchi

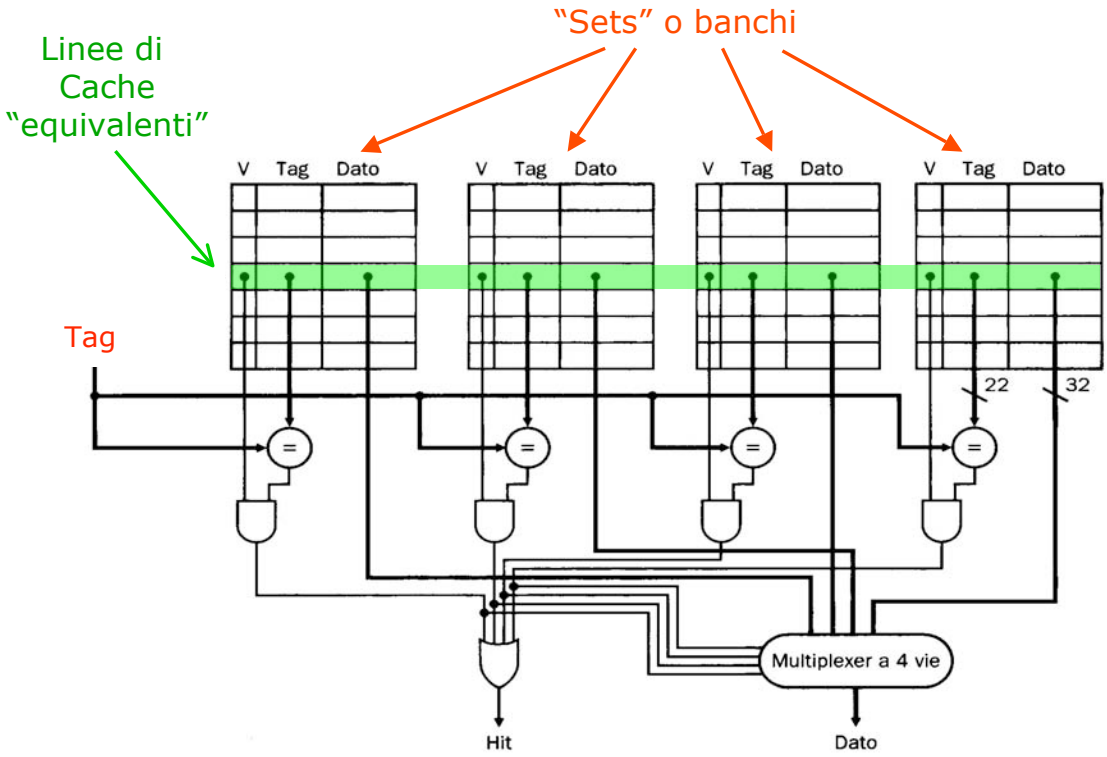
- La corrispondenza tra MP e linea di un banco è a mappatura diretta
- La corrispondenza tra MP e banco è associativa
  - Per cercare un dato devo analizzare tutti i banchi della linea indirizzata.

Dalle cache *direct map* alle cache *associative*

- N-associativa: N canali (vie) di cache direct-map, in parallelo



Memorie *N-associative*



Accesso a cache N-associative

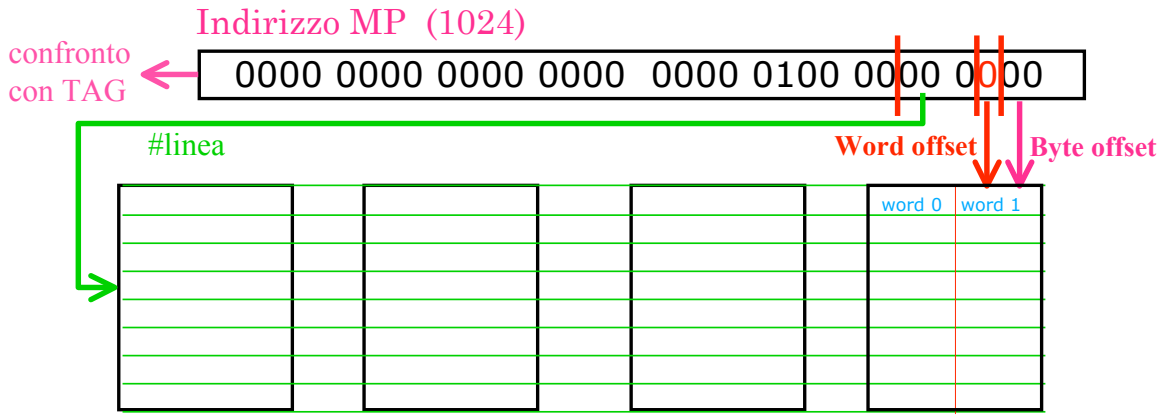
- **INDICE –** seleziona il *banco (insieme)* che potrebbe contenere il blocco desiderato.
  - La parola di MP memorizzata in cache si trova in una particolare linea di uno dei banchi → linea individuata dall' indice
  - L'indice è costituito da **k** bit, dove: $k = log_2(\#linee)$ analogo al numero di linea nelle cache a mappatura diretta

- **TAG –** contiene il blocco della MP da cercare.
  - Cerca il tag di MP all'interno dei TAG associati alla linea individuata in ciascun banco.
  - Nelle memorie N–associative, ho N possibili linee da controllare

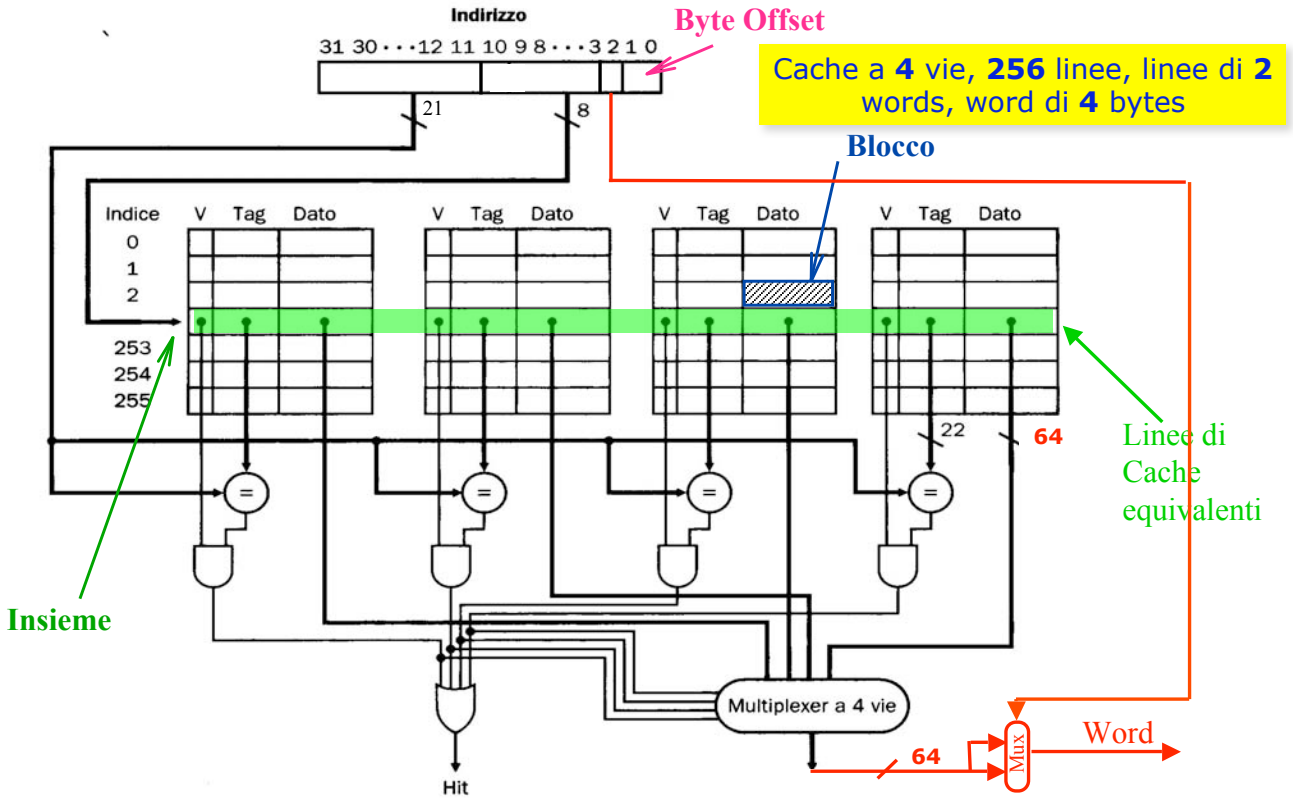
- Il comparatore pilota anche il MUX che trasferisce in uscita il contenuto del banco opportuno della cache.

esempio (cont.) – blocchi di 2 parole

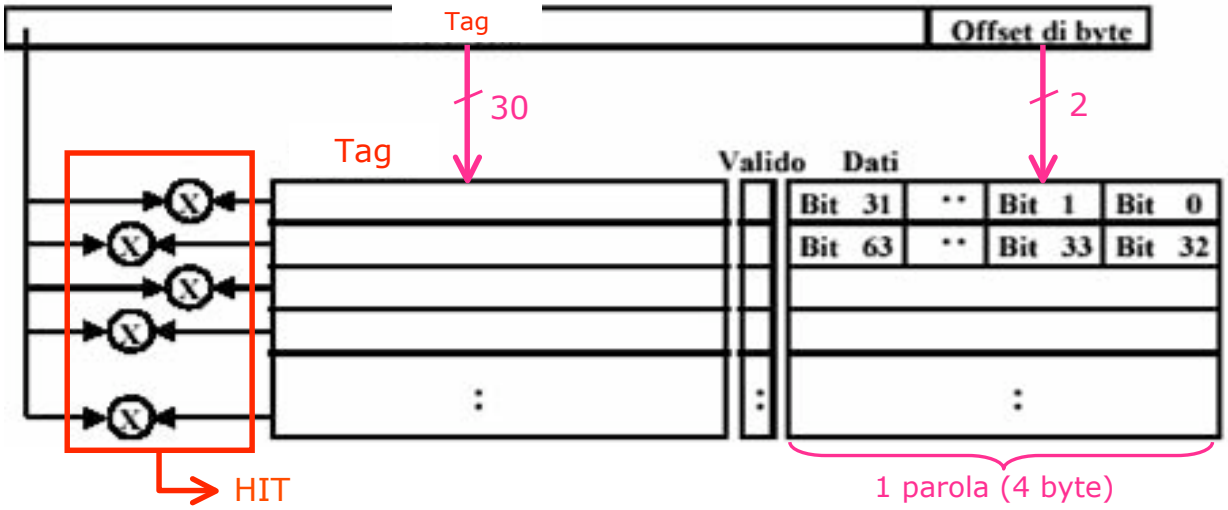
- Es: Cache a **4** vie, **8** linee, linee di **2** words, word di **4** bytes
  - 4 banchi, ciascuno di 8 linee.
  - Linea di cache = 2 word → byte offset: 2 byte, word offset: 1 byte
- Come viene elaborato l'indirizzo: `lw 0($s0)` ? ($s0 = 1024)



Memorie set-associative



Memorie associative



- Memoria completamente associativa.
  - Consente di caricare un blocco di RAM in una qualsiasi linea di cache.
  - Tramite comparatori individuo in quale blocco si trova il mio dato.
  - Il segnale di Hit si genera come AND (comparatore_output, Valido)
  - *Dove scrivo il blocco?*

Posizionamento di un blocco di MP in cache

- Corrispondenza diretta: in un'unica posizione.
  - Memoria ad 1 via.
    N. di banchi (vie) = N. di blocchi (linee)
  - Controllo del tag del blocco → 1 comparazione

- Completamente associative: in **C** posizioni (C blocchi).
  - Ciascun banco è costituito da 1 blocco.
  - C insiemi o banchi.
  - C comparazioni: controllo di tutti i tag

- N-associative: in **N** posizioni (N: grado di associatività).
  - Ho **N** insiemi (banchi)
  - Ciascun insieme è costituito da: **m = C/N** blocchi.
  - N comparazioni

Sommario

- Memorie cache n-associative
- Gestione della cache

Strategie di sostituzione di un blocco

Se devo scrivere un blocco di cache, quale banco scelgo?

- Ho libertà di scelta: N vie = N alternative

- Dove inserisco il blocco letto dalla memoria?
  - Necessità: soluzione hardware, algoritmo semplice

- **LRU** – Least Recently Used
  - Viene associato ad ogni blocco un bit di USE.
- **LFU** – Least frequently Used
  - Associa un contatore ad ogni blocco di cache.
- **FIFO**
  - Implementazione tramite buffer circolare.
- **Random**
  - Scelgo semplicemente a caso.

- Random non funziona molto peggio degli altri!

Cache: *meccanismi di scrittura*

- **Write-through:**
  Scrittura contemporanea in cache e in memoria principale
  - *Write_buffer* per liberare la CPU (DEC 3100)
  - Necessaria una gestione della sincronizzazione tra contenuto della Memoria Principale (che può essere letta anche da I/O e da altri processori) e Cache.
  - Svantaggio: traffico intenso sul bus per trasferimenti di dati in memoria.

- **Write-back:**
  Scrittura ritardata: scrivo solo quando devo scaricare il blocco di cache e riportarlo in memoria principale
  - Utilizzo un bit di flag: UPDATE, che viene settato quando altero il contenuto del blocco
    - Alla MP trasferisco il blocco
  - Vantaggiosa con cache n-associative: minimizzo il traffico su bus

Sistemi *multiprocessore*

CACHE COHERENCE: Mantenimento dell'informazione di cache coerente tra varie cache (**sistemi multi-processore**)

- **Bus watching** with **write-through**
  - Il controller della cache monitora il bus indirizzi + il segnale di controllo write della memoria
  - Invalida il contenuto di un blocco se il suo corrispondente in memoria principale viene scritto.
  - Funziona se tutti i processori adottano il meccanismo di write-through
- **Hardware trasparency**.
  - Circuito addizionale attivato ad ogni scrittura della Memoria Principale.
  - Copia la parola aggiornata in tutte le cache che contengono quella parola.
- **Non-cacheable memory**.
  - Viene definita un'area di memoria condivisa, che non viene copiata in cache.

Gestione dei fallimenti di una cache

- **HIT**: → il funzionamento della CPU non viene alterato.
- **MISS:** → devo aspettare che il dato sia pronto in cache
  → stallo! : il programma non può continuare!!

- Operazioni da eseguire in caso di MISS:
  1. Ricaricare l'indirizzo dell'istruzione (PC→ PC–4)
  2. Leggere il blocco di memoria dalla memoria principale.
  3. Trasferire il blocco in cache, aggiornare i campi validità e tag
  4. Riavviare la fase di fetch dell'istruzione.

È quindi opportuno:

- Diminuire il tasso di fallimento (miss_rate)
- massimizzare la velocità delle operazioni di gestione di MISS.
  - Massimizzare la velocità di trasferimento tra cache e memoria principale

Miglioramento prestazioni del sistema di memoria

- La durata totale della gestione di MISS dipende dai tempi di accesso a memoria:
  - **1 ciclo** di clock per inviare l'indirizzo.
  - **15 cicli** per ciascuna attivazione della MP (lettura di parola).
  - **1 ciclo** per trasferire una parola al livello superiore (cache).

- ESEMPIO: Blocco di cache: 4 parole; Bus dati ↔ RAM: 1 parola

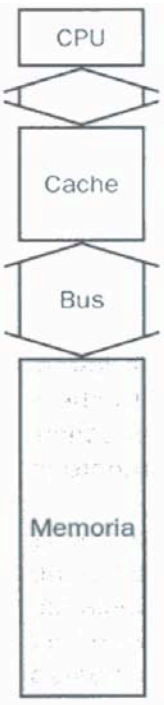
⇓

Miss_Penalty:

1 + 15 * 4 *(parole)* + 1 * 4 *(parole)* = **65 *cicli_clock***

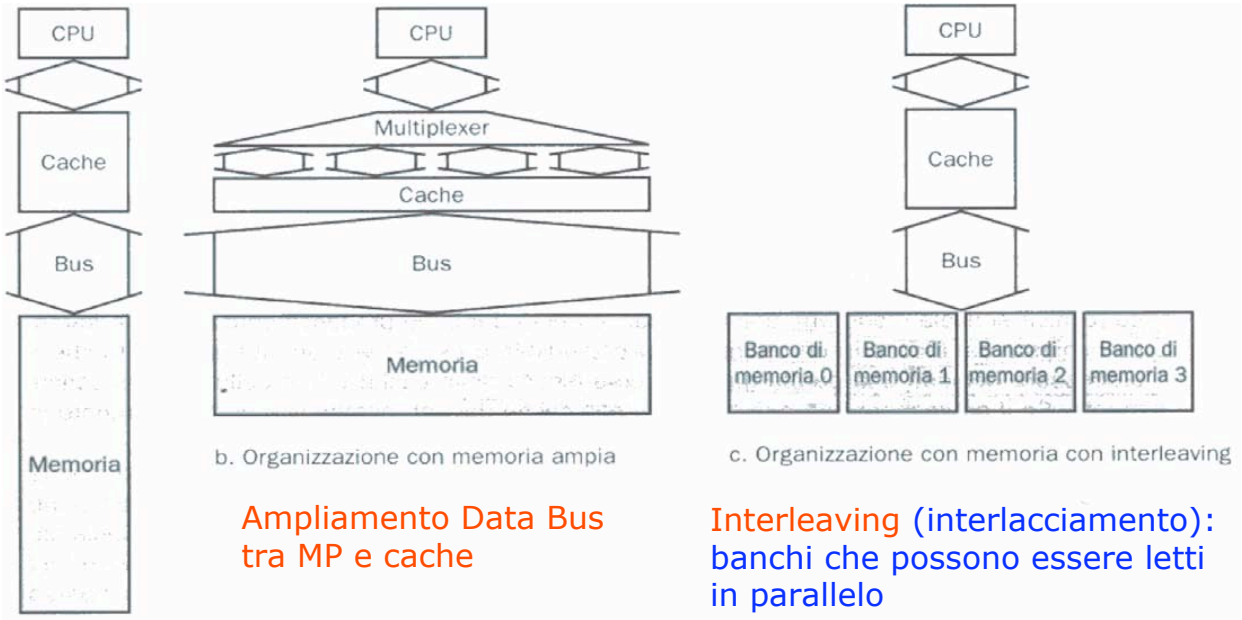
Transfer Rate:

#byte/ciclo_clock = 4(*parole*) * 4(*byte/parola*) / 65(*cicli_clock*) =
≅ **0,25 *byte/ciclo_clock***



Diminuire la penalità di fallimento (**miss_penalty**).

Strutture RAM per ridurre la **miss penalty**



Struttura tradizionale

Ampliamento Data Bus tra MP e cache

Interleaving (interlacciamento): banchi che possono essere letti in parallelo

Valutazione riduzione di "miss_penalty"

- Architettura standard: penalità di miss è di **65** cicli_clock.

- **Maggiore ampiezza della memoria:**
  - Organizzazione della Memoria Principale per blocchi.
  - Bus più ampio (bus dati largo un blocco, 4 parole).

  Per blocchi di MP di 4 parole, blocchi di cache di 4 parole:
  - **Miss_penalty** = 1 + 15 * 1 + 1 * 1 = **17 cicli_clock**
  - ***Transfer_rate*** = 4(*parole*) * 4(*byte/parola*) / 17(*cicli_clock*)
    = **0,94** (*byte/ciclo_clock*)

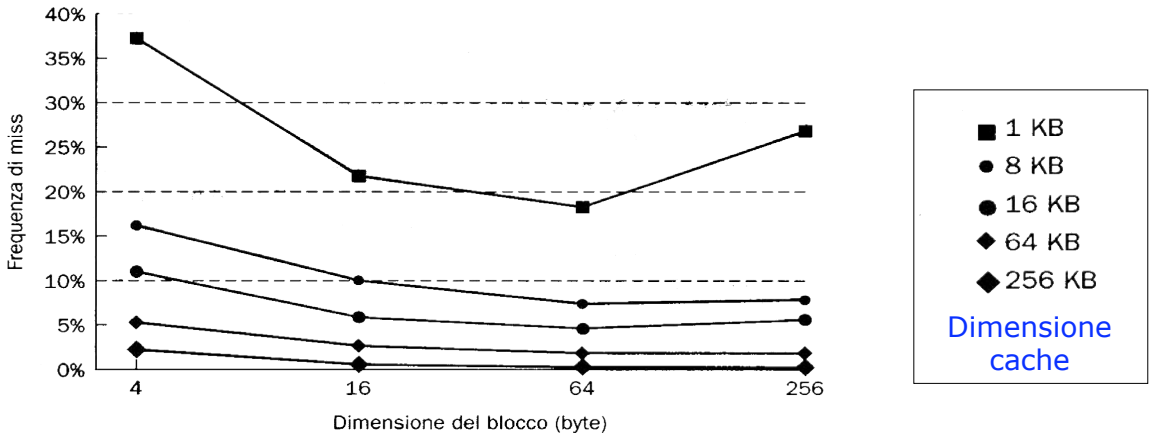
- **Interleaving:**
  - Organizzazione della Memoria Principale per banchi con accesso indipendente alla memoria (interleaving)
  - L'accesso alla parola da leggere in MP è fatto in parallelo su ogni banco
  - Bus standard ➜ trasferimento di 1 parola alla volta

  Per blocchi di MP di 1 parola, blocchi di cache di 4 parole:
  - **Miss_penalty** = 1 + 15 * 1 + 1*4 = **20 cicli_clock**
  - ***Transfer_rate*** = 4 (*parole*) * 4 (*byte/parola*) / 20 (*cicli_clock*)
    = **0,80** (*byte/ciclo_clock*)

Miss-rate e dimensione dei blocchi

- Dimensionamento del blocco(linea) di cache:
  - La dimensione della linea di cache è multipla della parola della macchina.
  - La dimensione ottima della linea dipende dalla parola del processore.

- All'aumentare della dimensione della linea…
  - **Per la località spaziale ➔ diminuisce la frequenza di MISS.**
  - **Per le dimensioni del blocco rispetto alla capacità della cache ➔ aumenta la penalità di MISS (competizione per le poche linee di cache)**



Gerarchie di cache

- Cache primaria nel processore.
- Cache secondaria con un bus dedicato per il trasferimento al processore.
  - Problemi: complessità nel circuito che deve assicurare la cache coherence.

- Split-cache: Cache Dati / Cache Istruzioni.
- Vantaggi
  - Possibilità di analizzare le istruzioni in coda (contenute nella cache istruzioni) mentre si eseguono altre istruzioni (che lavorano su dati contenuti nella cache dati), senza dover competere per l'accesso alla cache.
  - Efficiente per le architetture superscalari.
- Svantaggi
  - Minore hit-rate, perchè non si sfrutta al meglio la memoria cache. Si potrebbe riempire un'unica cache maggiormente con dati o istruzioni a seconda del frammento di codice correntemente in esecuzione.

Le dimensioni di alcune cache

Processor	Type	Year of Introduction	L1 cache[a]	L2 cache	L3 cache
IBM 360/85	Mainframe	1968	16 to32 KB	—	—
PDP-11/70	Minicomputer	1975	1KB	—	—
VAX 11/780	Minicompterer	1978	16 KB	—	—
IBM 3033	Mainframe	1978	64 KB	—	—
IBM 3090	Mainframe	1985	128 to 256 KB	—	—
Intel 80486	PC	1989	8 KB	—	—
Pentium	PC	1993	8 KB/8 KB	256 to 512 KB	—
PowerPC 601	PC	1993	32 KB	—	—
PowerPC 620	PC	1996	32 KB/32 KB	—	—
PowerPC G4	PC/server	1999	32 KB/32 KB	256 KB to 1 MB	2 MB
IBM S/390 G4	Mainframe	1997	32 KB	256 KB	2 MB
IBM S/390 G6	Mainframe	1999	256 KB	8 MB	—
Pentium 4	PC/server	2000	8 KB/8 KB	256 KB	—
IBM SP	High-end server/ supercomputer	2000	64 KB/32 KB	8 MB	—
CRAY MTA[b]	PC/server	2001	16 KB/16 KB	96 KB	4 MB
Itanium	PC/server	2001	16 KB/16 KB	96 KB	4 MB
SGI Origin 2001	High-end server	2001	32 KB/32 KB	4 MB	—

Memoria Virtuale

- **Memoria virtuale**: estensione (cache) della Memoria principale (RAM) su memoria di massa (hard disk)
  - Estensione della memoria fisica. Maggiore quantità di memoria.
  - Gestione del **multi-tasking**. Anni '90: ***overlay*** definito nel linker. Oggi: trasparente, tramite il gestore della memoria virtuale.
- Concettualmente analoga alla cache.
  - Blocco/linea di memoria → **Pagina**
  - MISS → **Page Fault**

Struttura:

- Ogni programma ha il suo spazio di indirizzamento.
- Mappatura della spazio di indirizzamento nella memoria fisica
  - (*memory mapping* tramite la *page table*)