



# Le memorie Cache associative

Prof. Alberto Borghese  
Dipartimento di Informatica  
[alberto.borghese@unimi.it](mailto:alberto.borghese@unimi.it)

Università degli Studi di Milano

Riferimento Patterson: 5.3, 5.4, 5.8



## Sommario

**Principio di funzionamento di una cache**

Circuito di lettura / scrittura di una cache a mappatura diretta

Cache associative

Osservazioni



## Gerarchia di memorie



Livelli multipli di memorie con diverse dimensioni e velocità.

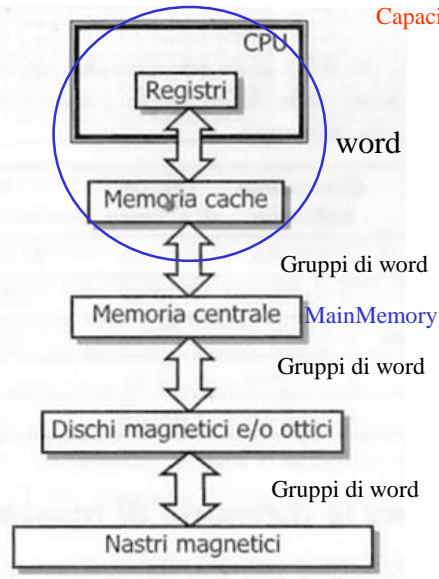
*Nel livello superiore troviamo un sottoinsieme dei dati del livello inferiore.*

*Ciascun livello vede il livello inferiore e viceversa.*

*Cache (memoria nascosta)*

Costo per bit  
& Velocità

Capacità

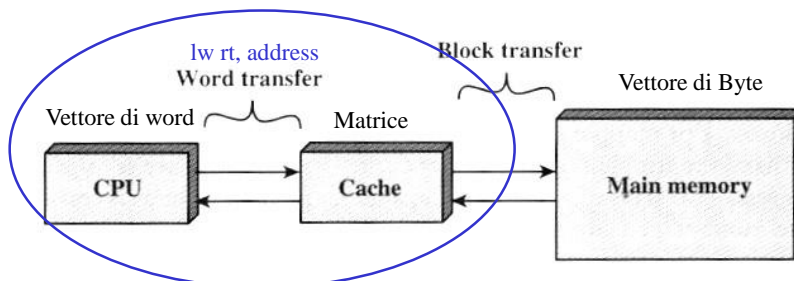


## Principio di funzionamento di una cache



**Scopo:** fornire alla CPU una velocità di trasferimento pari a quella della memoria più veloce con una capacità pari a quella della memoria più grande.

Una cache “disaccoppia” i dati utilizzati dal processore da quelli letti/scritti nella Memoria Principale.



Word transfer (dato o istruzione). In MIPS32 = 1 parola = 4 Byte.

Block transfer (più parole consecutive in MM – micro-blocco)

**La cache contiene una copia di parte del contenuto della memoria principale.**



# MMU, CPU e Cache



La MMU porta nella cache primaria i dati richiesti mentre il binomio processore-memoria sta lavorando.

- 1) Controlla se una parola è in cache (Hit).
- 2) Se si verifica una miss, porta una parola (e quelle vicine) in cache, prelevandole dal livello inferiore della gerarchia.

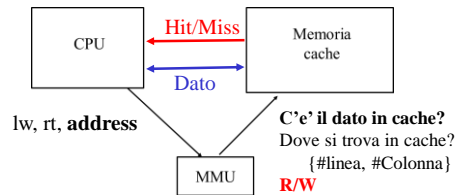
- **Linea di cache** = 4 word (16 Byte)
- **Capacità cache** = 128 Byte (32 word)

`lw $t0, 220($zero)`

`20410 = 0000 0000 0000 0000 0000 0000 1101 11002`

E gli altri bit?

#linea (5)    #colonna (3)



# Determinazione di #linea, #colonna



La cache con linee di **ampiezza pari a 4 parole** (micro-blocco = 4 parole) da 32 bit (4 byte), e **altezza di 8 linee**:

Il micro-blocco di dati della memoria principale che può essere contenuto in ogni linea di cache, ha dimensioni  $\text{dim\_linea} = 4 \text{ parole} * 4 \text{ byte} = 16 \text{ byte}$ .

La capacità della cache sarà  $C = 8 \text{ linee} * \text{dim\_linea} = 8 * 16 = 128 \text{ Byte}$  (macro-blocco di MM).

`lw $t0, 220($zero)`

Indirizzo\_cache = Indirizzo\_Memoria principale *modulo* dimensione\_macro-blocco\_MM

`220 / 128 Byte = 1` → mappiamo il 2° macro-blocco di MM sulla cache.

resto = 92 è l'indirizzo all'interno della cache (da trasformare in #linea, #colonna)

Numero linea = resto *modulo* dim\_linea (capacità\_micro-blocco)

`92 / 16 = 5` → La word è contenuta nella linea #5, => 6ª linea della cache.

resto = 12 è l'indirizzo all'interno della linea della cache (numero di byte nella linea, da trasformare in #colonna)

`12 / 4 = 3` → La word è la 4ª parola nella 6ª linea di cache. Resto = 0 è il numero di byte intra-word.

Il dato viene letto (trasferito nella CPU) assieme ai byte 221, 222, 223 della stessa 6ª linea della cache.

NB `220 / 16 = 13` → Riempio interamente 8 linee di una prima cache virtuale + 5 linee della cache reale.

**Le diverse linee possono provenire da macro-blocchi diversi della MM.**



## Determinazione della posizione mediante shift



.....0000 0000 1101 11(00)

lw \$t0, 220(\$zero)

220=128+64+16+8+4

Cache di 8 linee x 4 parole di 4 Byte ciascuna => Capacità della cache =  $8 \times 4 \times 4$  Byte = 128 Byte  
 $\log_2 128 = 7$  Numero di bit per indirizzare la cache.

Indirizzo / capacità\_cache (dimensione macro-blocco)  $220 / 128 = 1$  TAG

**R = 92 (indirizzo intra-cache)**

Resto / dim\_linea (capacità micro-blocco)  $92 / 16 = 5$  #Linea

$\log_2 8 = 3$  Numero di bit per indirizzare la linea (indice)

I 3 bit più significativi dell'indirizzo intra-cache indicano il numero di linea della cache indirizzata (per lettura / scrittura)

**R = 12 (numero di byte nella linea)**

Resto / dimensione della word (numero di Byte per word)  $12 / 4 = 3$  #Numero word

$\log_2 4 = 2$  Numero di bit per indirizzare la parola nella linea

Questi bit indicano il numero di colonna (parola) all'interno della linea della cache.

I 25 bit in verde identificano il campo TAG. E' il numero di macro-blocco di MM associato all'indirizzo 220.



## Cosa rappresentano gli altri bit?



.....0000 0000 1101 11(00)

lw \$t0, 220(\$zero)

220=128+64+16+8+4

I 25 leading bit in verde identificano il campo TAG. E' il numero di macro-blocco di MM associato all'indirizzo 220.

I 2 bit meno significativi identificano i byte intra-word, e non interessano perché il trasferimento da cache a CPU è di parole intere.

lw \$t0, 220(\$zero)

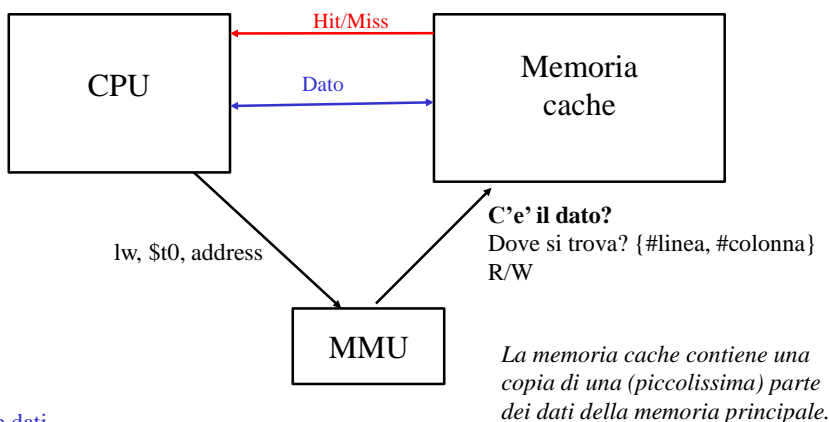
- Numero di linea = 5
- Numero di colonna = 3

Questa coppia, {numero linea, numero di colonna} è comune a tanti indirizzi, e.g.

- lw \$t0 92(\$zero) - TAG = 0
- lw \$t0, 348(\$zero) - TAG = 2
- lw \$t0, 476(\$zero) - TAG = 3



## Le domande alla MMU sulla memoria cache



### Parte dati.

Selezione del dato gerarchica: {#linea, #colonna (word)}

Schema a 2 livelli (selezione della linea – cf. Register File + selezione della colonna)

### Parte di controllo.

Hit/miss



## Come si può sapere se un dato è presente in cache?



Aggiungiamo a ciascuna delle linee della cache un campo **TAG**.

Il tag contiene i bit che costituiscono la parte più significativa dell'indirizzo e rappresenta il numero di macro-blocco di MM associato alla linea di cache in cui il dato è contenuto. Esso è costituito da K bit:

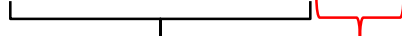
$$K = M - \text{sup}(\log_2 \text{Capacità\_cache (Byte)})$$

Nell'esempio precedente:  $K = 32 - \text{sup}(\log_2 128) = 25$  bit.

I bit per l'indirizzamento intra-cache sono  $32 - 25 = 7$  bit (128 Byte)

lw \$t0, 220(\$zero)

$220_{10} = 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 1101\ 1100_2$



Macro-blocco MM = TAG - etichetta associata a una linea



## Come si può sapere se un dato è presente in cache?



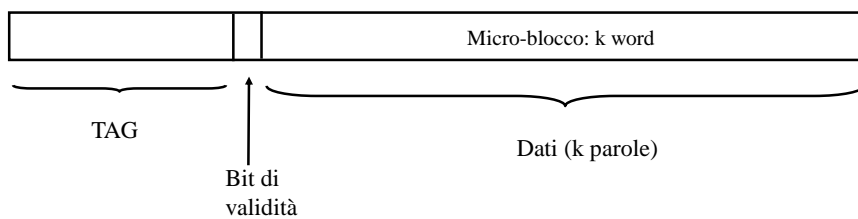
E all'accensione?

Come possiamo sapere che nessun dato è ancora stato caricato in cache?

Viene introdotto un bit di validità, associate a ogni linea. La validità è della linea intera.



## Struttura di una linea di cache

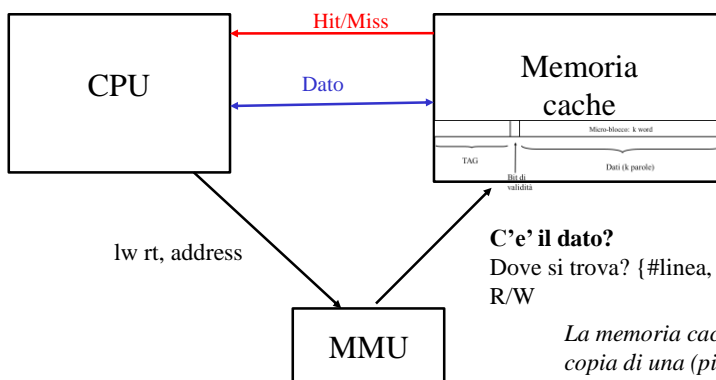


Nel caso precedente, avremo linee di cache di lunghezza:

$$25 (\text{lunghezza\_campo\_TAG}) + 1 (\text{bit di validità}) + 4 (\text{parole}) * 4 (\text{Byte/parola}) * 8 (\text{bit/Byte}) = 156 \text{ bit.}$$



## Le domande alle memorie cache



**C'è il dato?**  
Dove si trova? {#linea, #colonna}  
R/W

*La memoria cache contiene una copia di una (piccolissima) parte dei dati della memoria principale.*

`lw $t0, 220($zero)`  
`22010 = 0000 0000 0000 0000 0000 0000 1 101 11002`

Hit in cache se: `"TAG_address = TAG_lineaCache & validità_lineaCache == 1"`



## Sommario



Principio di funzionamento di una cache

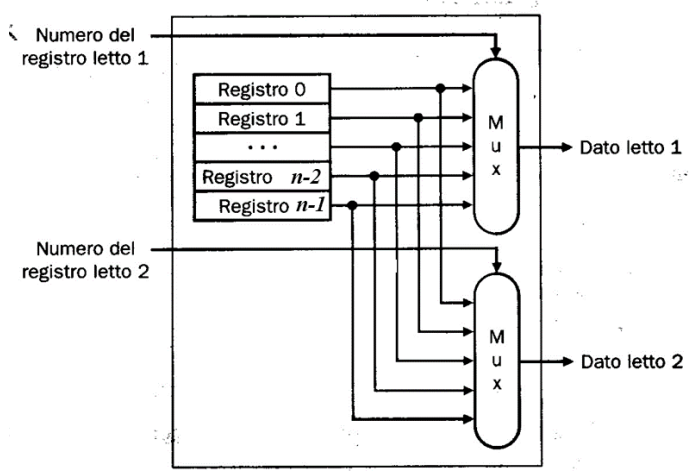
**Circuito di lettura / scrittura di una cache a mappatura diretta**

Cache associative

Osservazioni

# Porta di lettura di un dato (ampiezza 1 parola)

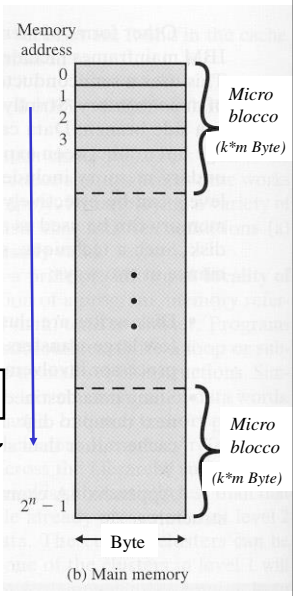
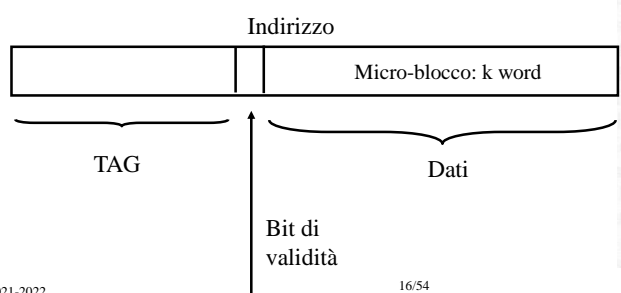
**Register file**  
 Pochi registri,  
 2 porte di lettura



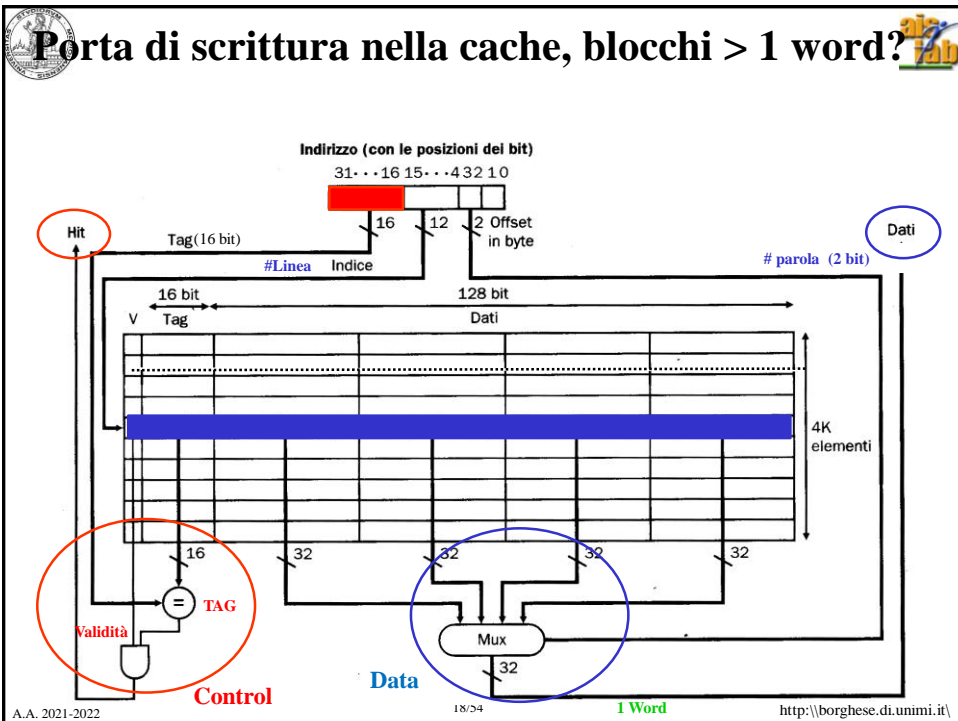
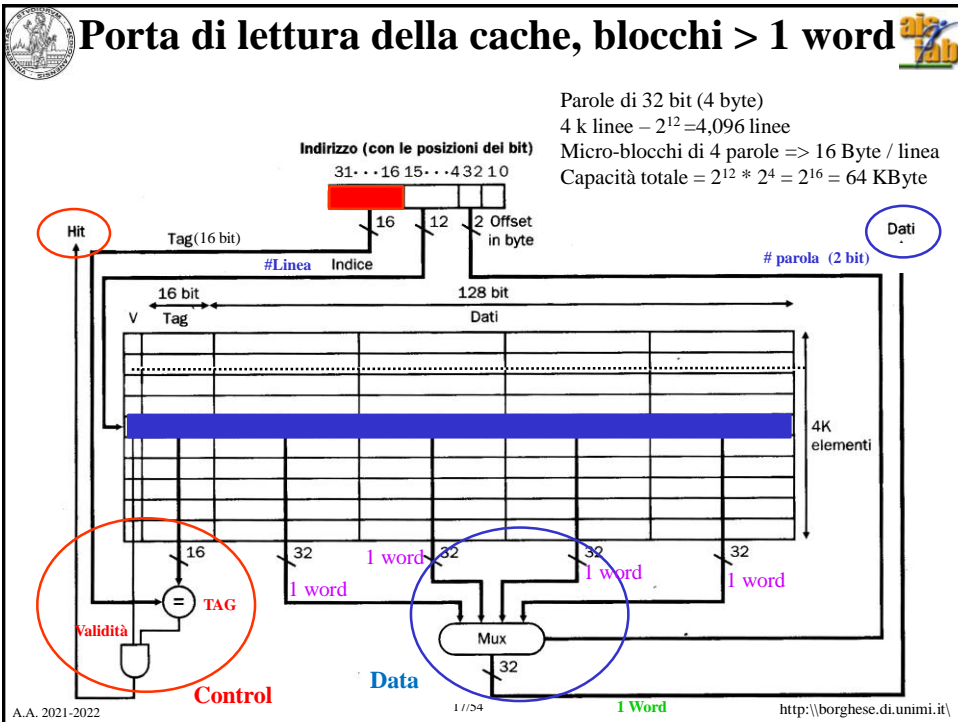
Cache: numero maggiore di celle di memoria. 1 sola porta di lettura. **Accesso a 2 livelli.**

# Come leggere dalla cache

- 1) Individuare la linea della cache dalla quale leggere.  
 Operazione analoga all'indirizzamento del register file.
- 2) Leggere la linea.
- 3) Confrontare il campo **tag dell'indirizzo** con il **campo tag della linea di cache** (tag è il numero di macro-blocco di MM)
- 4) Controllare il **bit di validità della linea di cache**.
- 5) Rispondere con hit/miss
- 6) *Selezionare la parola all'interno della linea* (per linee più ampie di una parola, occorre individuare una delle parole tra le k presenti nella linea di cache – micro-blocco).









## Sommario



Principio di funzionamento di una cache

Circuito di lettura / scrittura di una cache a mappatura diretta

Cache associative

Osservazioni



## Problemi con le cache a mappatura diretta



- Riempimento non ottimale (a macchia di leopardo): sostituzione del contenuto dell'intera linea della cache ogni volta che scrivo un singolo dato che appartiene a un blocco diverso di MM (anche nel caso di cache quasi vuota).
- **Memoria associativa**: il contenuto viene recuperato fornendo degli elementi dei dati, parte del contenuto, detti **chiavi** (e.g. ricerca nei data-base, ricerca attraverso ontologie WEB).
- Nelle memorie associative delle architetture si utilizza una **parte dell'indirizzo** come **chiave**, per recuperare il dato. Viene recuperato il dato che ha quella particolare chiave.

Occorre quindi sostituire il meccanismo di accesso diretto (parte dell'indirizzo -> numero di linea) con un meccanismo associativo che identifichi la linea associata alla chiave (parte dell'indirizzo = chiave di ricerca della linea).

Occorre provare la chiave fornita dall'indirizzo su tutte le "serrature" della cache.

Non esiste più un numero d'ordine delle linee, una numerazione.



## Meccanismo di accesso

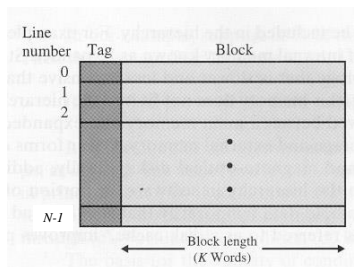
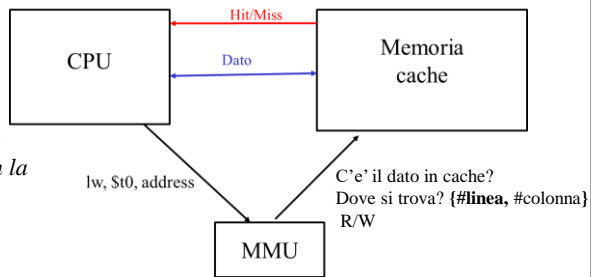
lw \$t0, 220(\$zero)

Come ricavo il #Linea?

Identifico la linea, confrontando la **chiave** contenuta nell'indirizzo con la chiave di tutte le linee di cache.

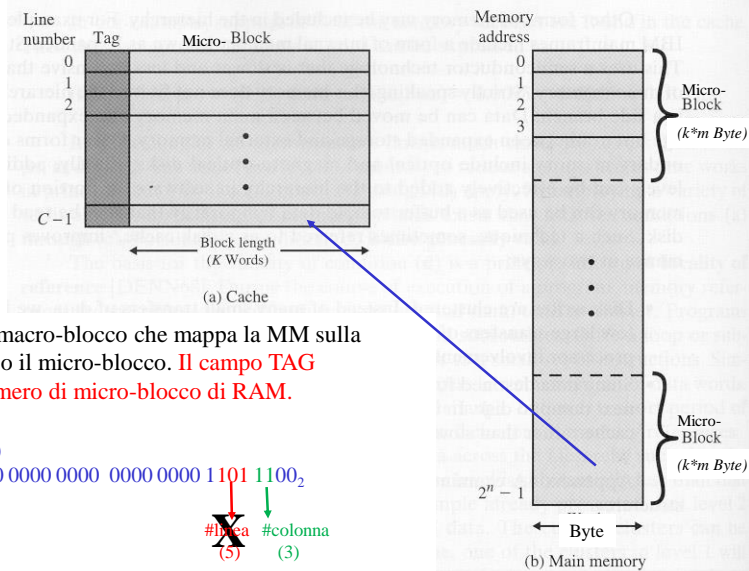
Come ricavo la chiave?

Come ricavo il numero di parola (word) nella linea?



## Memoria associativa

Numero linee: 8  
Block length: 4



Non esiste più il macro-blocco che mappa la MM sulla cache, rimane solo il micro-blocco. Il campo TAG rappresenta il numero di micro-blocco di RAM.

lw \$t0, 220(\$zero)

$220_{10} = 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 1101\ 1100_2$

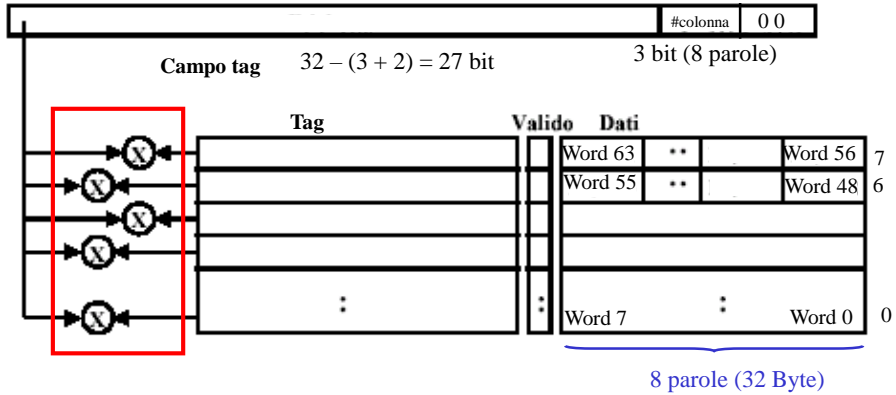
#linea (5) #colonna (3)

Non posso più ricavare il numero di linea direttamente dall'indirizzo. Posso utilizzare il campo TAG come chiave.



# Memorie associative

Memoria associativa con linee di 8 parole => Dimensione della linea: 8 x 4 Byte = 32 Byte



Numero di linee = 16 (la dimensione del campo TAG non dipende dal numero di linee)

Capacità della cache: 32 x 16 = 512 Byte (128 word)

Il numero di linee non entra nell'indirizzamento

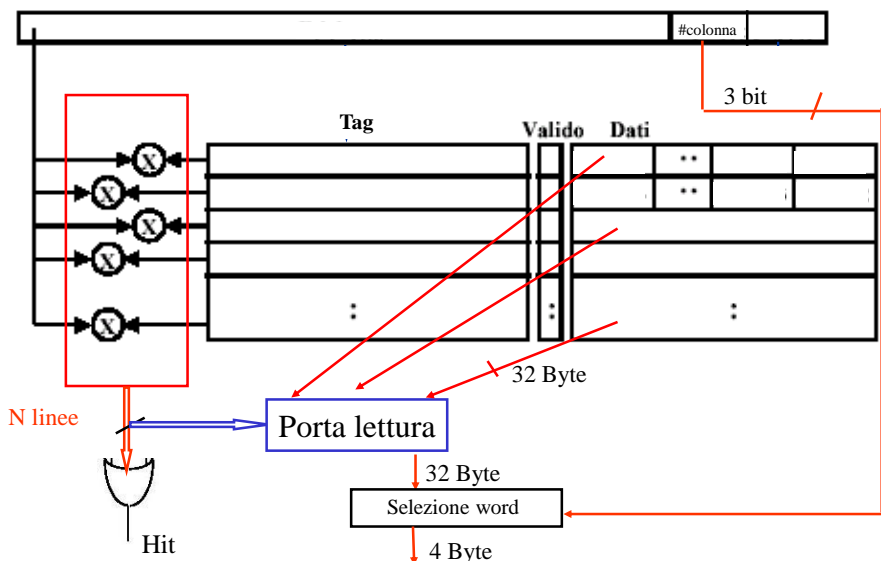
Consentono di caricare un blocco di Memoria Principale in una qualsiasi linea di cache.

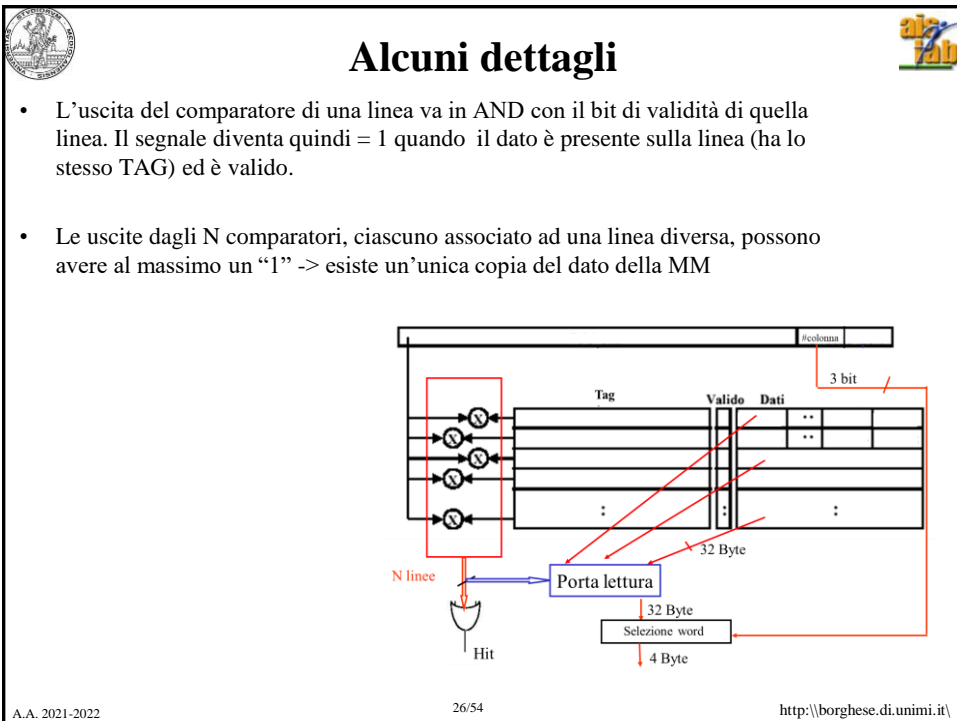
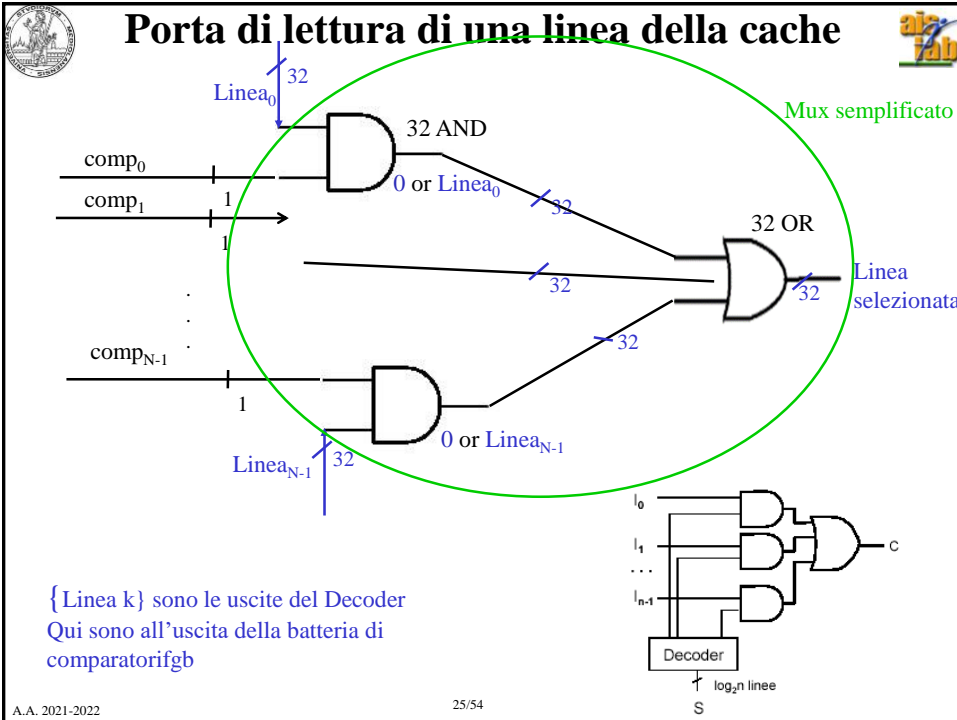
E' una memoria completamente associativa.

Tramite una schiera di comparatori individuo in quale linea si trova il mio dato.



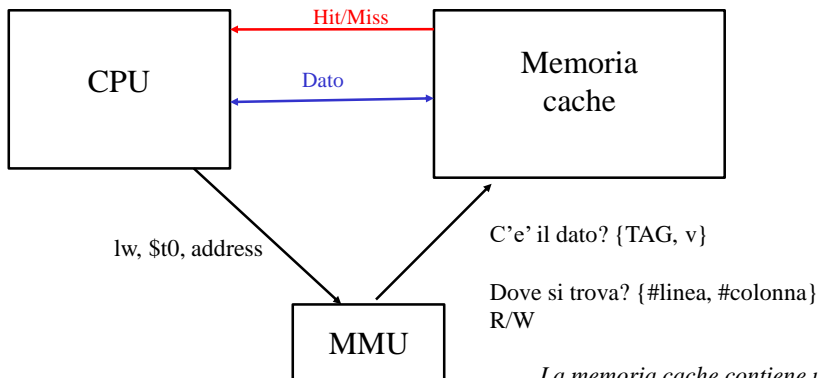
# Letture di una memoria associativa







## Le domande alla MMU sulla memoria cache



C'e' il dato? {TAG, v}

Dove si trova? {#linea, #colonna}  
R/W

*La memoria cache contiene una copia di una (piccolissima) parte dei dati della memoria principale.*

### Parte dati:

Schema a 2 livelli (selezione della linea mediante chiave + selezione della colonna mediante indice)

Parte di selezione del dato gerarchica: {#linea, #word (colonna)}

**Parte di controllo:** {TAG,v}



## 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.

**Cache:** è l'insieme dei banchi più i circuiti che li gestiscono.

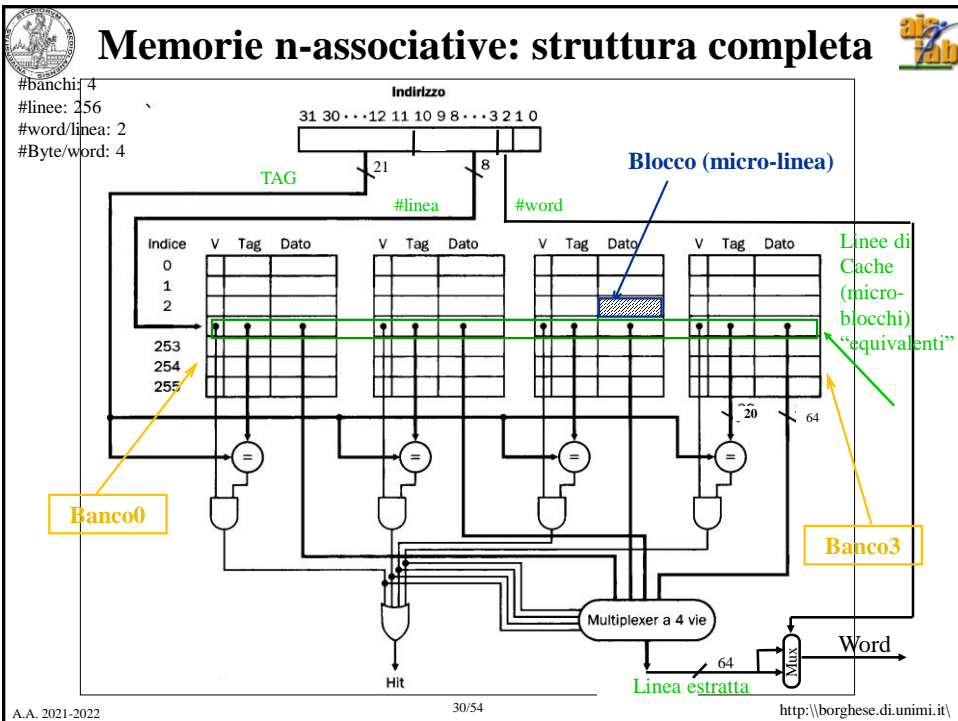
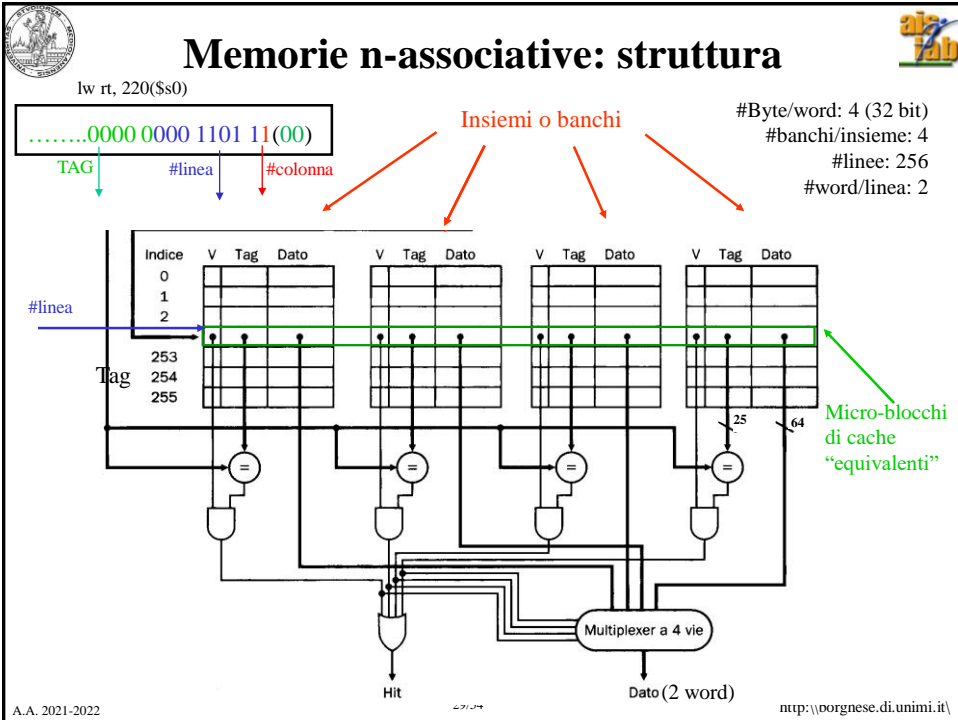
**Insieme (banco):** cache elementare ad accesso diretto.

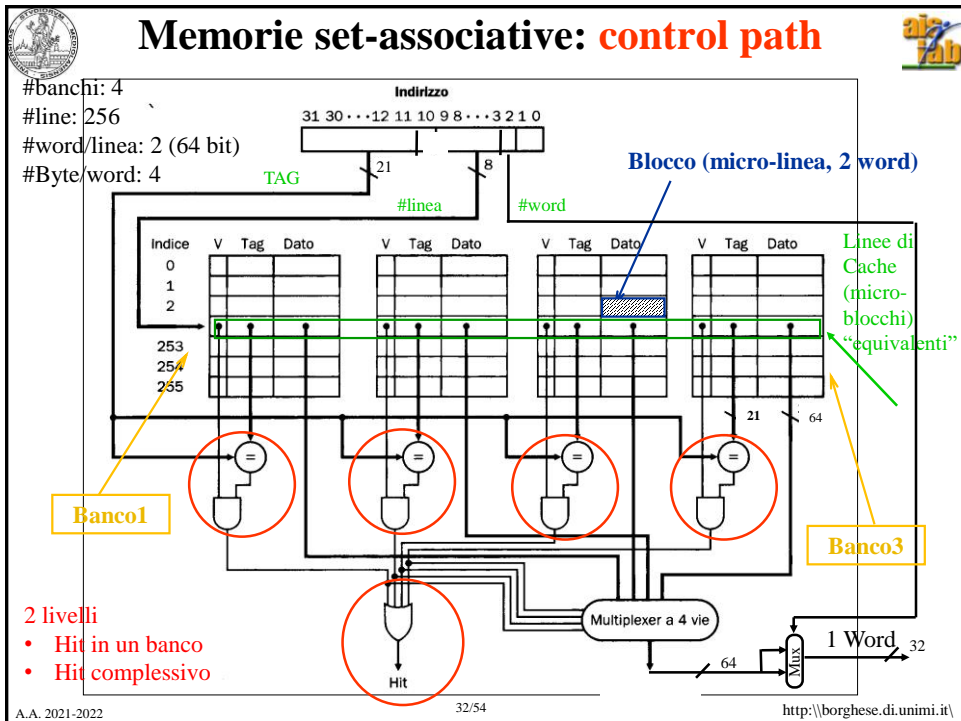
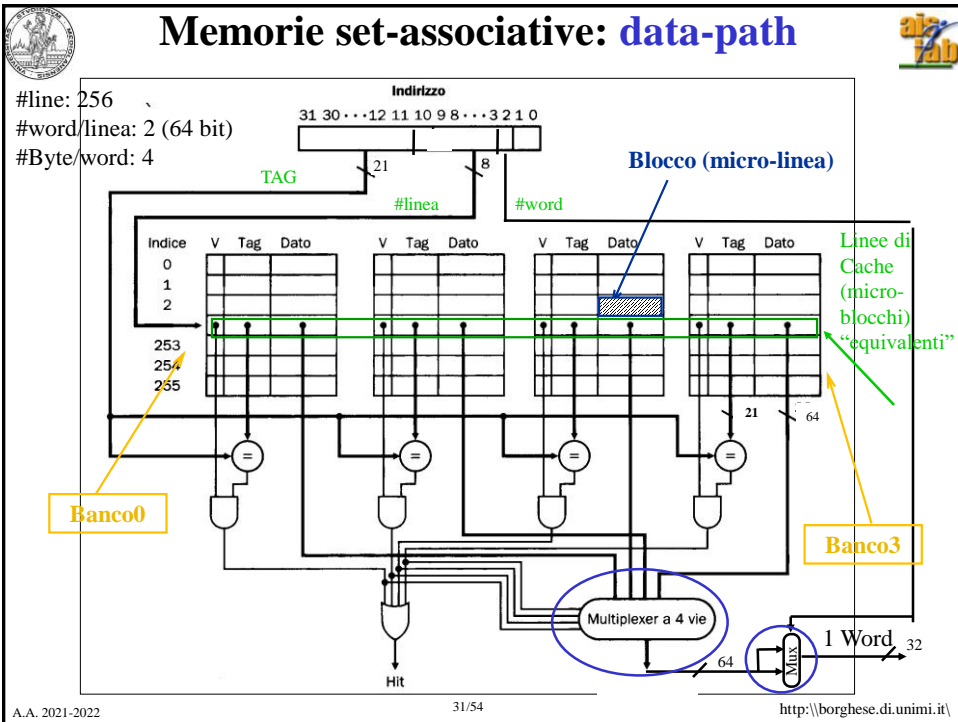
**Capacità della cache:** #parole = #banchi \* (#linee / banco) \* (#parole / linea).

**Micro-blocco (linea di cache):** #parole (byte) della MM adiacenti, contenute in una linea di uno dei banchi della cache.

**La corrispondenza tra Memoria Principale e linea di un banco è a mappatura diretta.**  
**La corrispondenza tra Memoria Principale e i diversi banchi è associativa.**

**Associatività.** Per cercare un dato non devo più analizzare tutte le linee di una cache, ma un'unica linea, ma devo analizzare quella linea sui diversi banchi.









## Accesso a cache ad n-vie

- 1) **INDICE.** Se la parola richiesta è memorizzata in cache, si trova in una particolare linea di uno dei banchi. Questa linea è individuata dall'**indice (di linea)**.
- 2) **NUMERO COLONNA.** Estrae la parola dalla linea.
- 3) **TAG** – contiene numero del macro-blocco della MM a cui appartiene il dato. Cerca il tag dell'indirizzo all'interno del **TAG della linea** identificata a mappatura diretta, nei **diversi banchi**.

Indice (numero di linea) e colonna (numero di parola) sono equivalenti alla mappatura diretta.

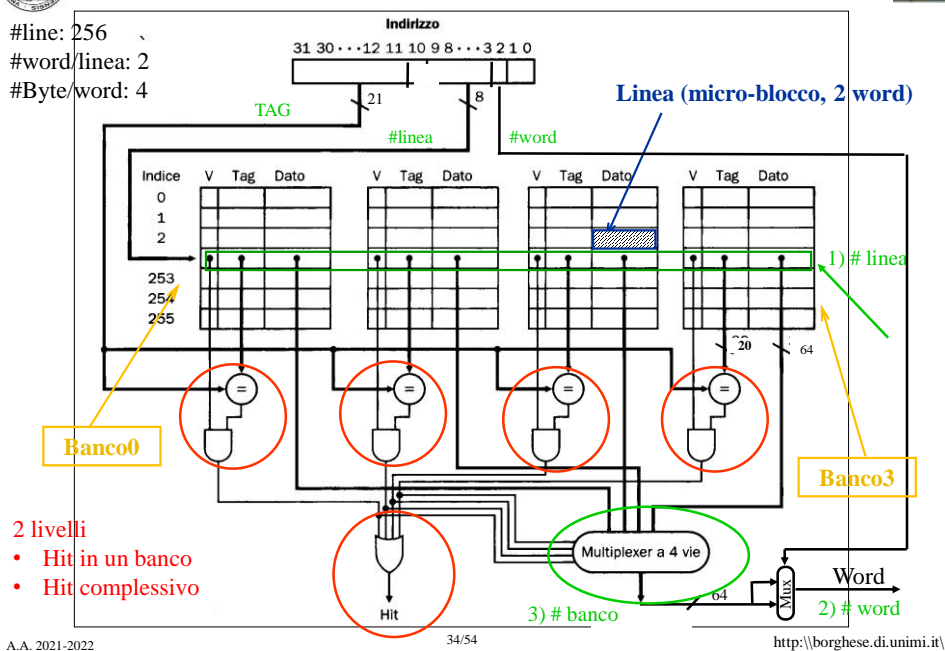
Qui si introduce un terzo livello di indirizzamento attraverso l'**associatività tra banchi** (accesso per chiave).

L'insieme dei segnali di HIT pilotano anche il «MUX» che trasferisce in uscita il contenuto del banco opportuno della cache.



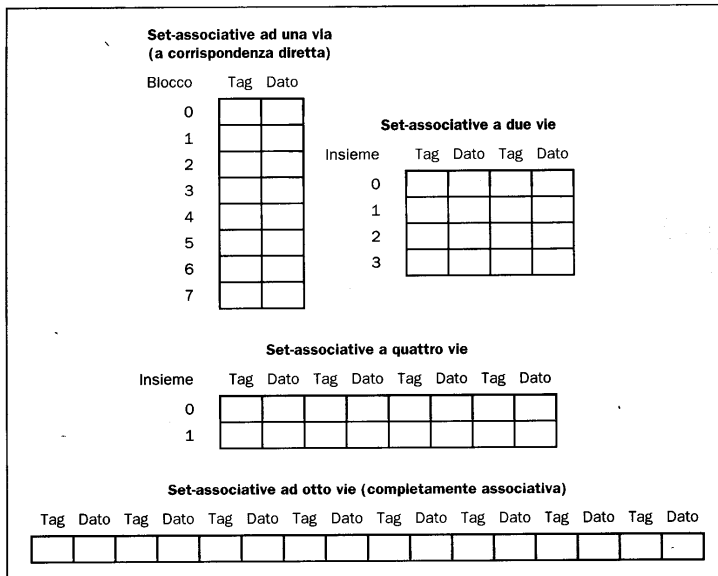
## Memorie set-associative

#line: 256  
#word/linea: 2  
#Byte/word: 4





# Dalle cache a mappatura diretta alle cache associative



## Sommario



Principio di funzionamento di una cache

Circuito di lettura / scrittura di una cache a mappatura diretta

Cache associative

**Osservazioni**



## Criteri di progettazione



Parametri di definizione della struttura di una cache:

- Capacità
- Grado di associatività

**Cache primaria:** massimizzo Hit rate.

**Cache secondaria:** minimizzo Miss penalty (massimizzo transfer rate).



## Tassonomia del funzionamento



**HIT** Successo nel tentativo di accesso ad un dato: è presente al livello superiore della gerarchia.

**MISS** Fallimento del tentativo di accesso al livello superiore della gerarchia => il dato o l'indirizzo devono essere cercati al livello inferiore.

**HIT\_RATE** Percentuale dei tentativi di accesso ai livelli superiori della gerarchia che hanno avuto successo.  
 $HIT\_RATE = \text{Numero\_successi} / \text{Numero\_accessi\_memoria}$

**MISS\_RATE** Percentuale dei tentativi di accesso ai livelli superiori della gerarchia che sono falliti  
 $MISS\_RATE = \text{Numero\_fall.} / \text{Numero\_accessi\_memoria}$

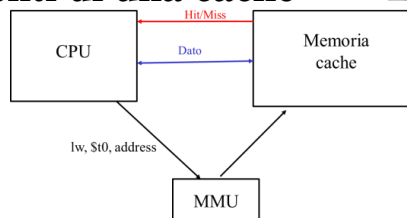
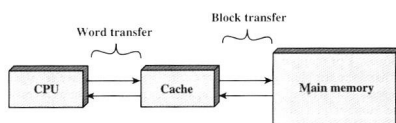
$$HIT\_RATE + MISS\_RATE = 1$$

**HIT TIME** Tempo richiesto per verificare se il micro-blocco contenente il dato è presente al livello attuale della memoria.

**MISS\_PENALTY** Tempo richiesto per sostituire il micro-blocco di memoria mancante al livello superiore.



## Gestione dei fallimenti di una cache



La gestione avviene tra CPU e MMU.

**Hit** – è quello che vorremmo ottenere, il funzionamento della CPU non viene alterato.

**Miss** – **in lettura** devo aspettare che il dato sia pronto in cache -> eccezione particolare della CPU che crea uno **stallo** della CPU. **Nelle CPU super-scalari si sfrutta l'esecuzione fuori ordine per nascondere questa latenza.**

**Passi da eseguire in caso di Miss (miss penalty):**

- 1) Bloccare tutte le istruzioni nella pipeline (blocco dei registri di pipeline per uno o più cicli di clock)
- 2) Richiedere che la MM legga e porti fuori il micro-blocco contenente il dato da leggere.
- 3) Trasferire il micro-blocco in cache, aggiornare i campi validita' e tag.
- 4) Riavviare l'esecuzione della pipeline.

NB Il programma non può continuare!!

ni.it\



## Tipi di miss di una cache



**3-C miss model: compulsory, capacity and conflict.**

**Miss obbligate (compulsory).** Quando vengono caricati dei dati **per la prima volta** in una linea. Sono chiamate anche miss da partenza a freddo (**cold-start**).

**Miss per capacità (capacity).** Inevitabili. La cache non può contenere tutti i micro-blocchi di dati e/o istruzioni che servono per l'esecuzione di un programma: il micro-blocco viene caricato in cache, poi scaricato e poi caricato ancora.

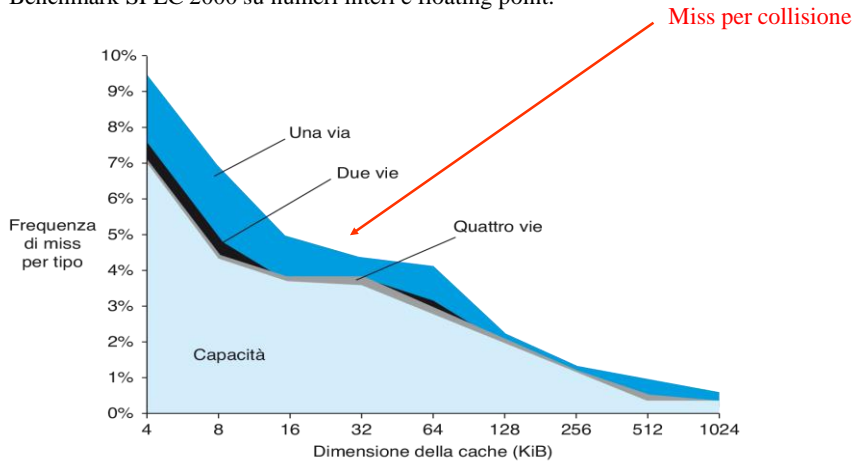
**Miss per collisione (conflict or collision).** Più micro-blocchi devono essere trasferiti nella stessa linea. In questo caso i trasferimenti devono essere accodati e sui micro-blocchi successive si verificano miss.

Quale pesa di più?



## Valutazione su benchmark

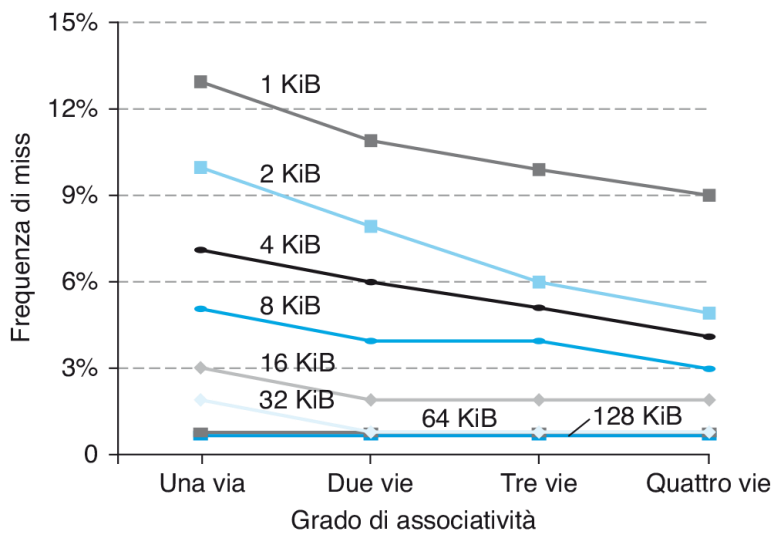
Benchmark SPEC 2000 su numeri interi e floating point.



Cold-start miss: 0,006%. Incremento oltre le 4-vie non è apprezzabile.



## Quanta associatività?



All'aumentare dell'associatività, aumenta la complessità e aumenta il tempo di accesso.



## Dove si può posizionare un blocco di MM in cache?



Corrispondenza diretta: in un'unica posizione.

Memoria a 1 via. Un unico banco.  
n linee (posizione individuata dall'indice).

Completamente associative: in n posizioni (n banchi).

Ciascun banco è costituito da 1 linea.  
n insiemi o banchi (equivalenti a n memorie indipendenti, banco individuato mediante chiave)

m-associative: in m posizioni (m grado di associatività).

Ho m insiemi (banchi)  
Ciascun insieme è costituito da n linee (posizione individuata dall'indice)



## Mappatura diretta



4 linee di 1 word → Capacità della cache = 16 Byte

lw \$t0, 0(\$0)  
lw \$t1, 32(\$0)  
lw \$t2, 0(\$0)  
lw \$t3, 24(\$0)  
lw \$t4, 32(\$0)

$0/16 = 0$  R = 0  
 $0/4 = 0$  (linea)

$32/16 = 2$  R = 0  
 $0/4 = 0$  (linea)

$24/16 = 1$  R = 8  
 $8/4 = 2$  (linea)

Indirizzo MM	Hit o Miss	Contenuto delle linee di cache dopo ogni lw			
		0	1	2	3
0	Miss	Mem[0]			
32	Miss	Mem[32]			
0	Miss	Mem[0]			
24	Miss	Mem[0]		Mem[24]	
32	Miss	Mem[32]			

← 4 Linee di 1 parola →

5 Miss (2 miss per cold-start, 3 miss per collisione)



## A 2-vie – prima possibilità



2 Banchi da 2 linee di 1 word ciascuna → Capacità della linea = 8 Byte; capacità della cache = 16 Byte.

lw \$t0, 0(\$0)  
lw \$t0, 32(\$0)  
lw \$t2, 0(\$0)  
lw \$t3, 24(\$0)  
lw \$t4, 32(\$0)

Indirizzo MM	Hit o Miss	Contenuto delle linee di cache dopo ogni lw			
		Banco 0 <sub>lin0</sub>	Banco 0 <sub>lin1</sub>	Banco 1 <sub>lin0</sub>	Banco 1 <sub>lin1</sub>
0	Miss	Mem[0]			
32	Miss			Mem[32]	
0	Hit	Mem[0]			
24	Miss	Mem[0]		Mem[24]	
32	Miss	Mem[32]		Mem[24]	

0/8 = 0 R = 0  
0/4 = 0 (linea 0)  
→ banco 0)

32/8 = 4 R = 0  
0/4 = 0 (linea 0)  
→ banco 1 (la linea 0 del banco 0 è già occupata)

24/8 = 3 R = 0  
0/4 = 0 (linea0)  
→ banco 0 o banco 1

32/8 = 4 R = 0  
0/4 = 0 (linea 0)  
→ banco 0 (la linea 0 del banco 1 è già occupata e di recente)

4 Miss (2 miss di cold-start, 2 per collisione)



## A 2-vie ottimizzata



2 Banchi da 2 linee di 1 word ciascuno → Capacità del banco = 8 Byte; capacità della cache = 16 Byte.

lw \$t0, 0(\$0)  
lw \$t1, 32(\$0)  
lw \$t2, 0(\$0)  
lw \$t3, 24(\$0)  
lw \$t4, 32(\$0)

Indirizzo MM	Hit o Miss	Contenuto delle linee di cache dopo ogni lw			
		Banco 0 <sub>lin0</sub>	Banco 0 <sub>lin1</sub>	Banco 1 <sub>lin0</sub>	Banco 1 <sub>lin1</sub>
0	Miss	Mem[0]			
32	Miss			Mem[32]	
0	Hit	Mem[0]			
24	Miss	Mem[24]		Mem[32]	
32	Hit	Mem[24]		Mem[32]	

0/8 = 0 R = 0  
0/4 = 0 (linea 0, banco 0)

32/8 = 4 R = 0  
0/4 = 0 (linea 0, banco 0)

24/8 = 3 R = 0  
0/4 = 0 (linea0, banco 0)

3 Miss (2 Miss da cold-start, 1 miss da collisione)

Ottimizzazione della scelta del banco



## A 4-vie (completamente associative)



4 Banche da 1 linea di 1 word ciascuno → Capacità della linea = 4 Byte; capacità della cache = 16 Byte.

lw \$t0, 0(\$0)  
lw \$t0, 32(\$0)  
lw \$t2, 0(\$0)  
lw \$t3, 24(\$0)  
lw \$t4, 32(\$0)

Indirizzo MM	Hit o Miss	Contenuto delle linee di cache dopo ogni lw			
		Banco 0	Banco 1	Banco 2	Banco 3
0	Miss	Mem[0]			
32	Miss		Mem[32]		
0	Hit	Mem[0]			
24	Miss	Mem[0]	Mem[32]	Mem[24]	
32	Hit	Mem[0]	Mem[32]	Mem[24]	

0/4 = 0 R = 0

0/4 = 0 (linea 0, banco 0)

32/4 = 4 R = 0

0/4 = 0 (linea 0, banco 1)

24/4 = 3 R = 0

0/4 = 0 (linea 0, banco 2)

3 Miss per cold-start



## Effetto della modifica della struttura



Cambiamento nel progetto	Effetto sulla frequenza di miss	Eventuale effetto negativo sulle prestazioni
Aumento della dimensione della cache	Diminuiscono le miss di capacità	Può aumentare il tempo di accesso
Aumento del grado di associatività	Diminuisce la frequenza delle miss causate da conflitti	Può aumentare il tempo di accesso
Aumento della dimensione del blocco	Diminuisce la frequenza delle miss per un'ampia gamma di dimensioni dei blocchi a causa della località spaziale	Aumenta la penalità di miss. Blocchi molto grandi potrebbero aumentare la frequenza di miss

La capacità della cache L2 e L3 e l'ampiezza della loro linea, aumenta con continuità. Aumenta di poco la latenza, ma diminuiscono le miss.

La capacità della cache L1 è rimasta pressochè costante sia in dimensioni che in struttura (32 Kbyte, 1, 2, 4 vie).





## Come si trova un blocco di MM in cache?



Corrispondenza diretta:

Indicizzazione mediante {#riga, #colonna}.

Controllo del tag + bit validità del blocco (1 comparazione).

Associativa: ricerca in tutte le n linee della cache.

n comparazioni: controllo di tutti i tag + bit di validità + indicizzazione colonna.

La memoria virtuale è di questo tipo (tramite la *Page Table*).

m-associativa: ricerca negli m insiemi (banchi),

Indicizzazione mediante {#riga, #colonna}.

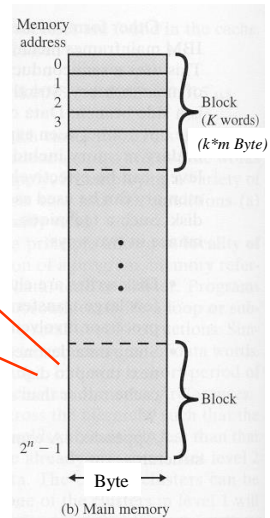
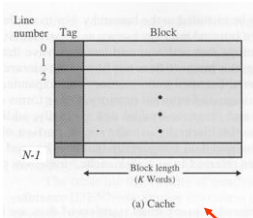
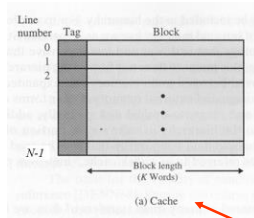
Controllo del tag + bit validità della linea sugli m insiemi (m comparazioni).



## Politiche di sostituzione di una linea di cache



**In quale banco inserisco in cache il micro-blocco letto dalla MM?**



Banco 0?

Banco 1?

Corrispondenza diretta: 1 solo posto possibile =>  
Non c'è scelta.

Associativa: ricerca in tutti le n linee della cache =>  
n possibili scelte.

M-associative: ricerca in tutti gli m banchi della cache =>  
m possibili scelte.



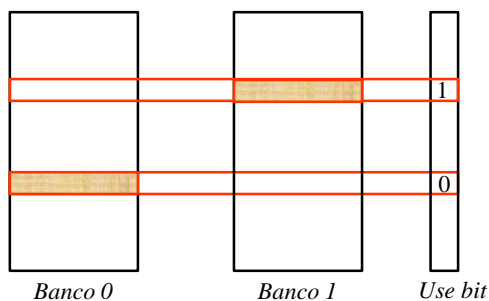
## Politiche di sostituzione di una linea di cache



### Quando posso scegliere il banco, quale banco sovra-scrivo?

LRU – Least recently Used (linea del banco utilizzato meno di recente).

Cache a 2-vie. 1 unico bit (*use bit*) di utilizzo che viene impostato a 0 o a 1 ogni volta che viene letta/scritta la linea di uno dei due banchi.

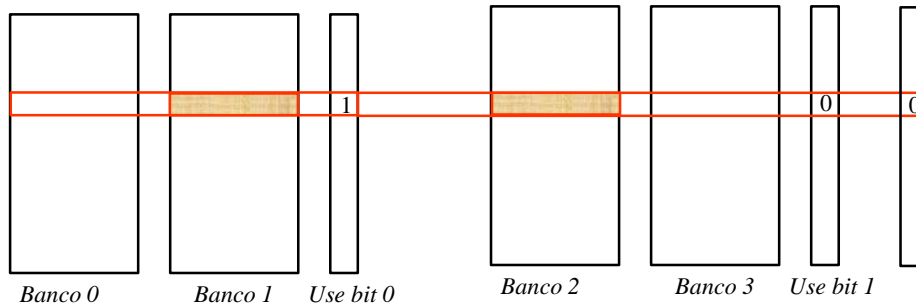


## Use bit in cache a 4-vie



LRU – Least recently Used gerarchico

Cache a 4-vie. Una gerarchia di *use bit*.



Il “Global use bit” viene impostato a 0, quando leggo/scrivo nel banco 0/1 e a 1 quando leggo/scrivo nel banco 2/3.

Identifico il banco da cui scaricare la linea percorrendo la gerarchia degli use bit:

Global use bit = 1 (coppia di banchi più vecchi è la coppia #1: banco #2 – banco #3)

Use bit 1 = 0 (il banco più vecchio è il primo della coppia: banco #2).



## Altre politiche di sostituzione



- LRU approssimato. Use bit che viene periodicamente impostato a 0 su tutte le linee (reference bit della memoria virtuale).
- LFU – Least frequently Used. Associa un contatore ad ogni linea di cache. Efficiente per memorie a 2 vie.
- FIFO – Implementazione tramite buffer circolare (cache a n-vie)
- Scelta random della linea da scaricare. Non così cattiva! Nelle cache a 2-vie, la miss rate è di circa 1.1 volte quella della politica LRU. Spesso la soluzione per cache con grado di associatività  $> 4$ .



## Sommario



- Principio di funzionamento di una cache
- Circuito di lettura / scrittura di una cache a mappatura diretta
- Cache associative
- Osservazioni