



Le memorie Cache

Prof. Alberto Borghese
Dipartimento di Scienze dell'Informazione
borgnese@dsi.unimi.it

Università degli Studi di Milano

Riferimento Patterson: 7.3; 7.5



Sommario

Circuito di lettura / scrittura di una cache a mappatura diretta

Memorie associative

Memorie n-associative

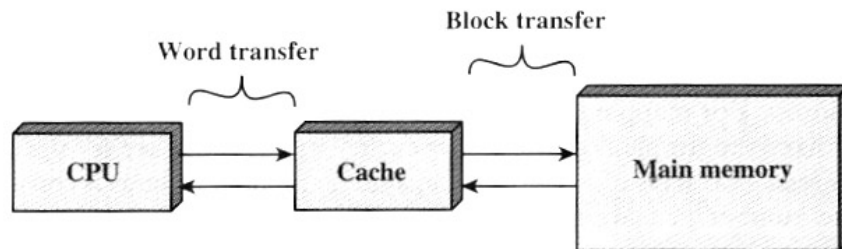


Principio di funzionamento di una cache



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 memorizzati nella Memoria Principale.



Word transfer: Data transfer or Instruction transfer. In MIPS = 1 parola.

La cache contiene una copia di parte del contenuto della memoria principale. Di che cosa?



Sottosistema di memoria



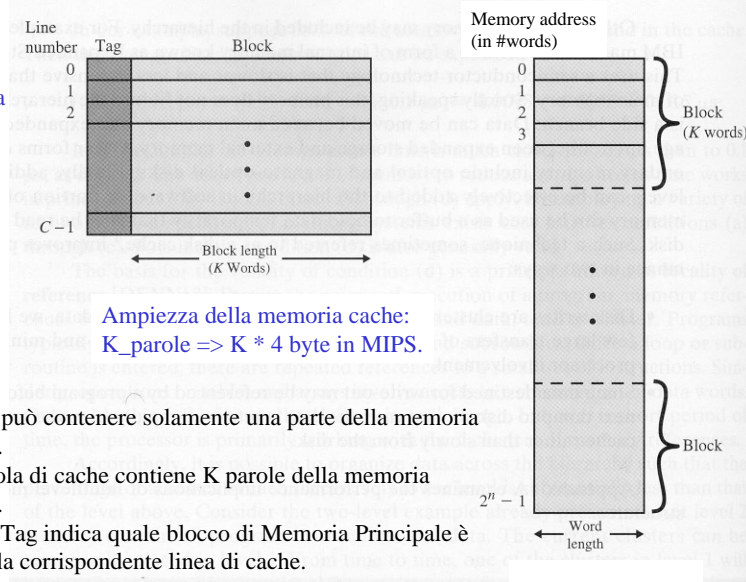
Porta nella cache primaria i dati richiesti mentre il binomio processore-memoria sta lavorando.

- 1) Controlla se una parola è in cache (Hit).
- 2) Porta una parola (e quelle vicine) in cache, prelevandole dal livello inferiore (Miss):



Mappatura diretta di una cache

Altezza della memoria cache: # di linee

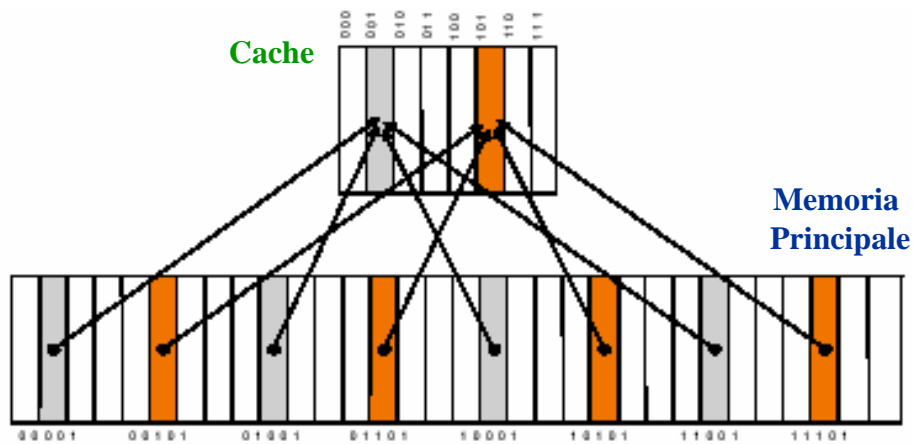


- La cache può contenere solamente una parte della memoria principale.
- Ogni parola di cache contiene K parole della memoria principale.
- Il campo Tag indica quale blocco di Memoria Principale è scritto nella corrispondente linea di cache.



Corrispondenza diretta (direct mapped)

Ad ogni indirizzo di Memoria Principale corrisponde un indirizzo di cache.



Indirizzi diversi di Memoria Principale corrispondono allo stesso indirizzo di cache.
Quali indirizzi della memoria principale si considerano?

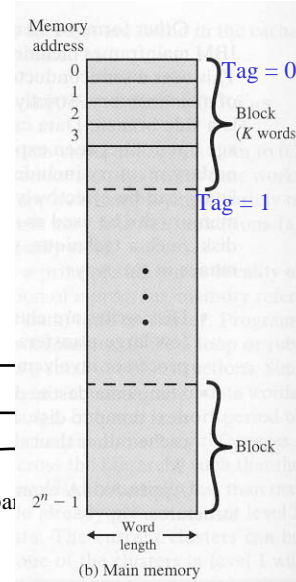
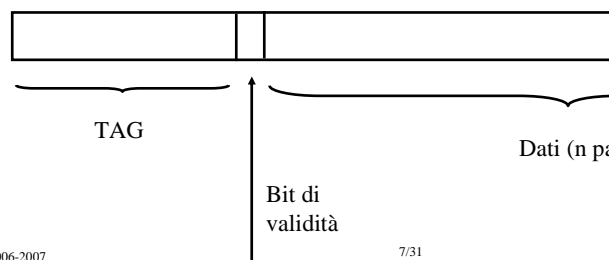


Come leggere / scrivere su cache



- Individuare la linea della cache dalla quale leggere / scrivere (operazione analoga all'indirizzamento del register file).
- Confrontare il campo tag con il blocco di Memoria Principale in cui risiede il dato.
- Controllare il bit di validità.
- Leggere (scrivere) il dato.

Per blocchi più ampi di una parola, occorre individuare una parola tra le k presenti nella linea di cache.



Esempio di parsing dell'indirizzo



0000 0000 0000 00(00) ⇒ 0000 0000 0111 11(11) 128 indirizzi diversi (32 parole di 4 byte)

La cache con linee di 4 parole (ampiezza) ed altezza di 8 linee:
 Il blocco di dati contenuto in ogni linea di cache è di dimensioni: $n = 4 * 4$ byte = 16 byte.
 La capacità della cache è di $8 * 16$ byte = 128 byte.

`lw $t0, 196($zero)`

$196 / [4 * 4 * 8] = 1$ (2° blocco di RAM, tag = 1) con resto $R_1 = 196 - 1 * 128 = 68$.
 Il resto, R_1 , rappresenta l'offset in byte all'interno della cache.

$68 / [4 * 4] = 4$ (5ª linea della cache) con resto $R_2 = 68 - 4 * 16 = 4$.
 Il resto, R_2 , rappresenta l'offset in byte all'interno della linea di cache.

$4 / 4 = 1$ (2ª parola della cache) con resto $R_3 = 4 - 1 * 4 = 0$.
 Il resto, R_3 , rappresenta l'offset in byte all'interno della parola.



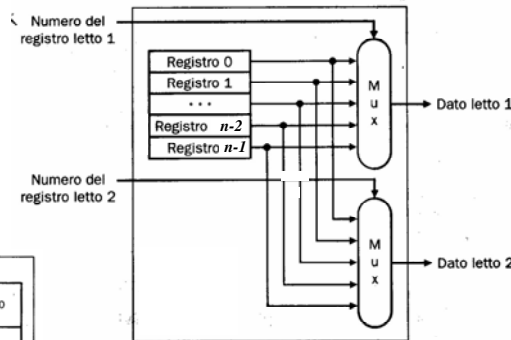
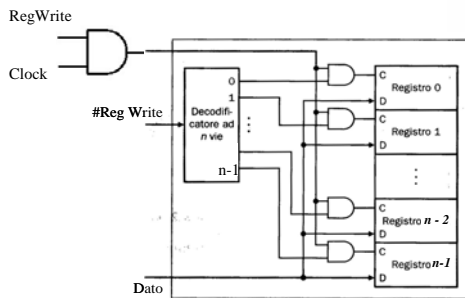
Register file



Il tempo di lettura dipende dal cammino critico dei Mux.

Il tempo di scrittura dipende dal cammino critico del Decoder.

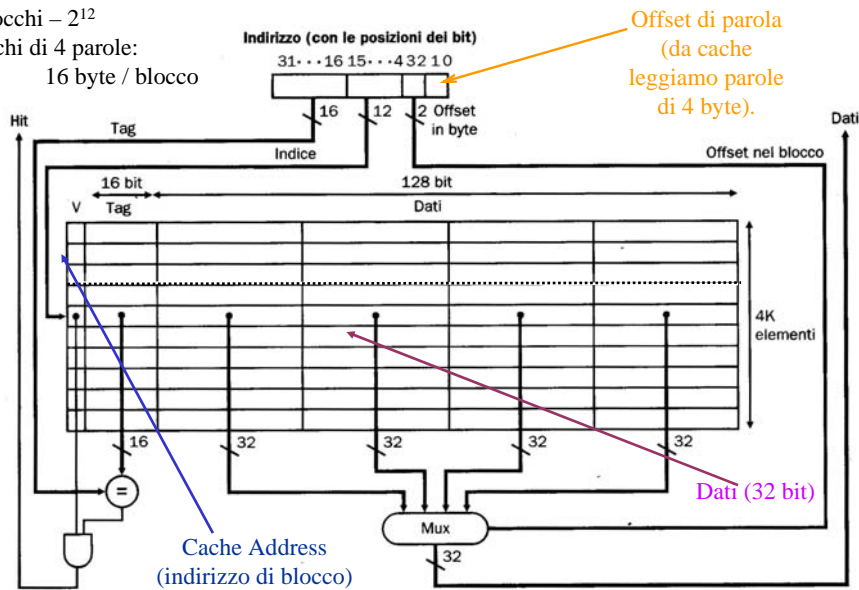
Numero_registro = selettore.



Controller della cache, blocchi > 1 word



4kblocchi - 2^{12}
Blocchi di 4 parole:
16 byte / blocco





Esempio di operazioni su cache



All'inizio il bit di `data_valid` = False. Suppongo cache di 8 linee di 1 byte.
Non importa quello che è contenuto. Cambio il contenuto di `$t1`.

- `lw $t0, 0($t1)` `$t1 = 22 (10 110)` MISS TAG = 10
- `lw $t0, 0($t1)` `$t1 = 26 (11 010)` MISS TAG = 11
- `lw $t0, 0($t1)` `$t1 = 22 (10 110)` HIT TAG = 10
- `lw $t0, 0($t1)` `$t1 = 26 (11 010)` HIT TAG = 11
- `lw $t0, 0($t1)` `$t1 = 16 (10 000)` MISS TAG = 10
- `lw $t0, 0($t1)` `$t1 = 2 (00 010)` MISS TAG = 00
- `lw $t0, 0($t1)` `$t1 = 7 (00 111)` MISS TAG = 0
- `lw $t0, 0($t1)` `$t1 = 16 (10 000)` HIT TAG = 10
- `lw $t0, 0($t1)` `$t1 = 18 (10 010)` MISS TAG = 10



Esempio di operazioni su cache



All'inizio il bit di `data_valid` = False. Suppongo cache di 8 linee di 2 byte.
Non importa quello che è contenuto. Cambio il contenuto di `$t1`.

- `lw $t0, 0($t1)` `$t1 = 22 (1 011 0)` MISS TAG = 1
- `lw $t0, 0($t1)` `$t1 = 26 (1 101 0)` MISS TAG = 1
- `lw $t0, 0($t1)` `$t1 = 22 (1 011 0)` HIT TAG = 1
- `lw $t0, 0($t1)` `$t1 = 26 (1 101 0)` HIT TAG = 1
- `lw $t0, 0($t1)` `$t1 = 16 (1 000 0)` MISS TAG = 1
- `lw $t0, 0($t1)` `$t1 = 2 (0 001 0)` MISS TAG = 0
- `lw $t0, 0($t1)` `$t1 = 7 (0 011 1)` MISS TAG = 0
- `lw $t0, 0($t1)` `$t1 = 16 (1 000 0)` HIT TAG = 1
- `lw $t0, 0($t1)` `$t1 = 18 (1 001 0)` MISS TAG = 1



Esercizi



Sia data una cache a corrispondenza diretta contenente 64Kbyte di dati e avente blocchi di 1 parola. Assumendo che gli indirizzi siano di 32 bit quale è il numero totale di bit richiesto per l'implementazione della cache?

Supponendo che il MIPS abbia una cache di 512byte, indicare cosa succede nei campi della cache quando vengono eseguite le seguenti istruzioni:

lw \$t1, 0x0000(\$t0) \$t0 = 1kbyte = 1,024 byte

lw \$t1, 0x0000(\$t0) \$t0 = 0

lw \$t1, 0x0202(\$t0) \$t0 = 1kbyte = 1,024 byte

lw \$t1, 0x0001(\$t0) \$t0 = 0

lw \$t1, 0x0201(\$t0) \$t0 = 1kbyte = 1,024 byte



Sommario



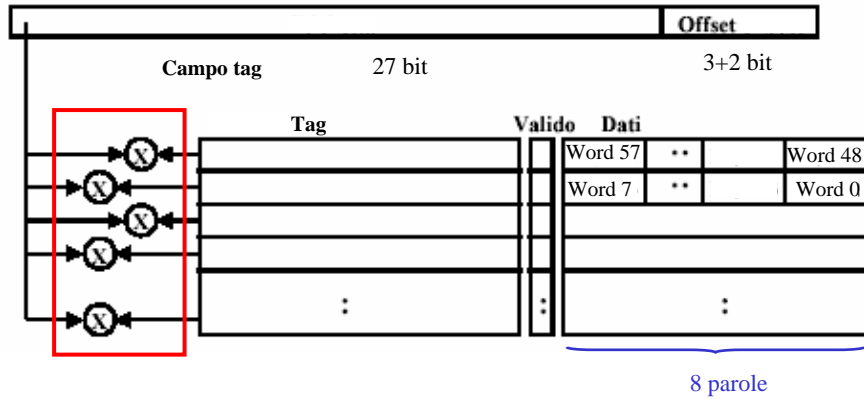
Circuito di lettura / scrittura di una cache a mappatura diretta

Memorie associative

Memorie n-associative



Memorie associative



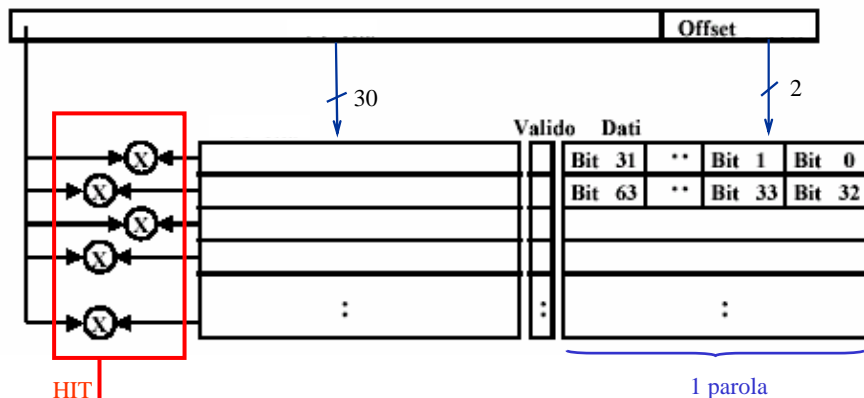
Consentono di caricare un blocco di Memoria Principale in una qualsiasi linea di cache.
E' una memoria completamente associativa.

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?



Memorie associative

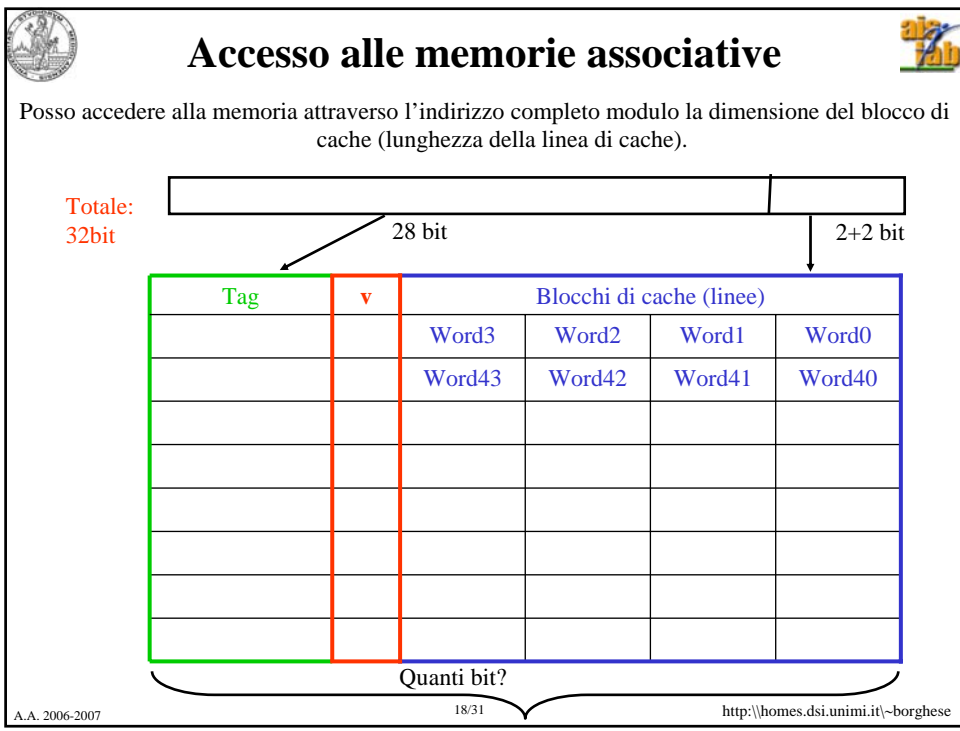
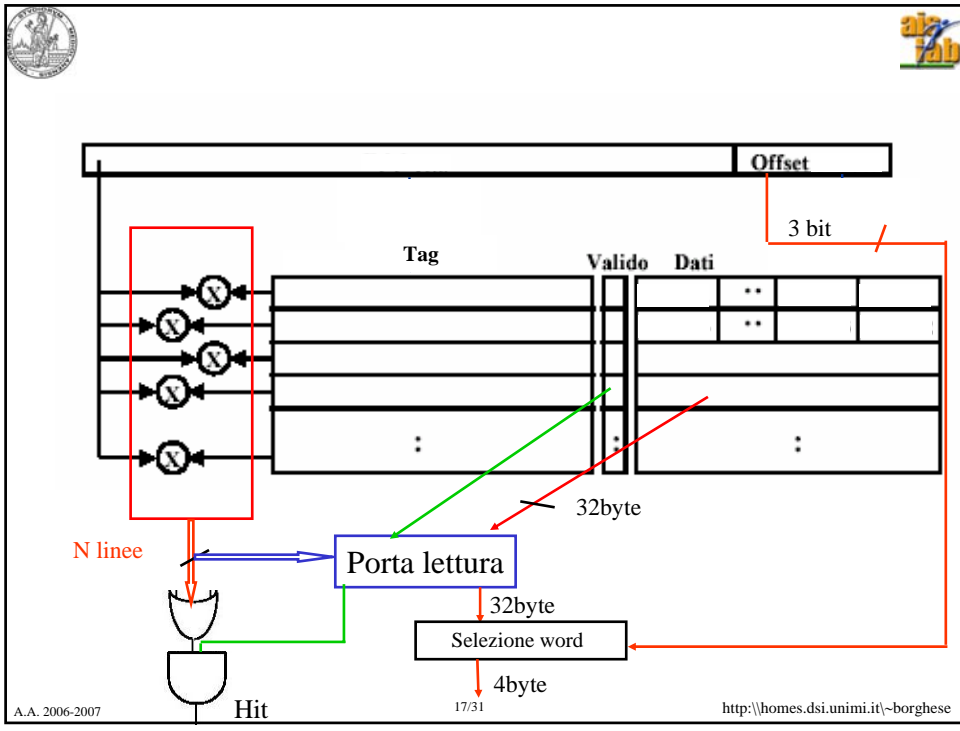


HIT

Consentono di caricare un blocco di Memoria Principale in una qualsiasi linea di cache.
E' una memoria completamente associativa.

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?





Tassonomia



Spazio di indirzzamento: $(s + w)$ bit: somma della dimensione del campo tag + somma della dimensione dell'offset all'interno della parola. Spazio misurato in word o byte (come nel caso del MIPS).

Numero di unità indirzzabili: $2^{(s+w)}$ unità ($2^{(s+w)}$ byte in MIPS).

Dimensione del blocco = dimensione della linea di cache = 2^w parole o byte.

Numero totale di macro-blocchi della memoria principale: 2^s .

Dimensioni del campo tag: s bit.

Viene aumentato il numero di Hit ma con un appesantimento notevole della circuiteria.



Sommario



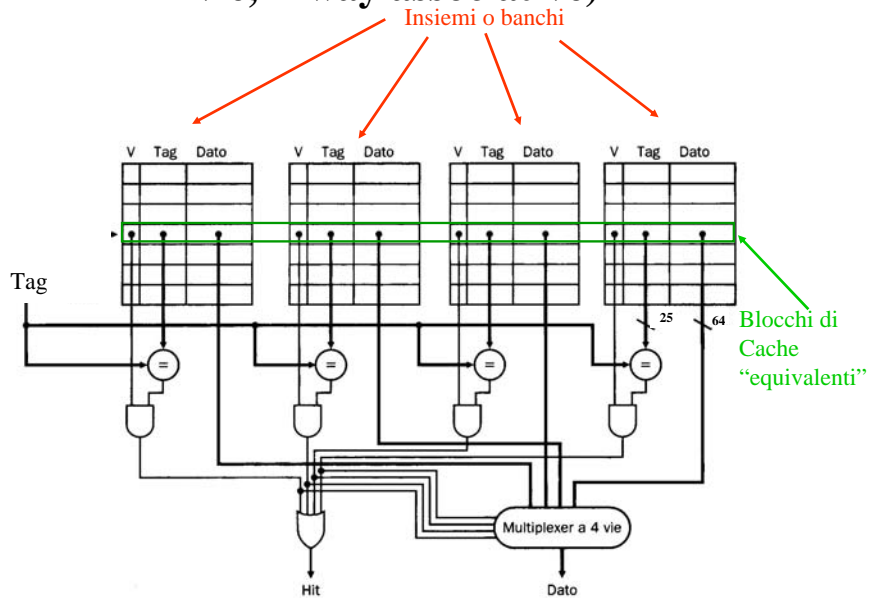
Circuito di lettura / scrittura di una cache a mappatura diretta

Memorie associative

Memorie n-associative



Memorie n-associative (o associative a n-vie, n-way associative)



A.A. 2006-2007

21/31

<http://homes.dsi.unimi.it/~borgnese>



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 di cache): #parole (byte) lette/scritte contemporaneamente in cache, "parola" della cache.

Insieme (banco): cache elementare.

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

Capacità della cache: #parole = #Insiemi * (#blocchi / insieme) * (#parole / blocco).

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

Per cercare un dato non devo più analizzare tutte le linee di una cache, ma un'unica linea per ogni banco.

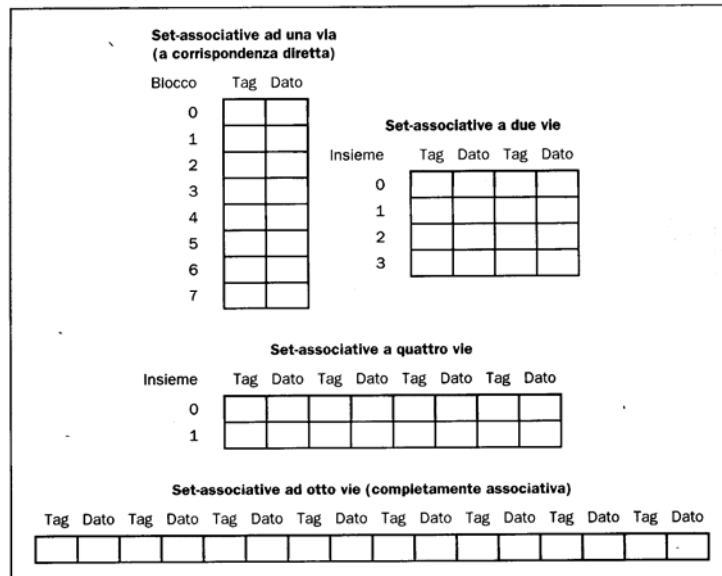
A.A. 2006-2007

22/31

<http://homes.dsi.unimi.it/~borgnese>



Dalle cache a mappatura diretta alle cache associative



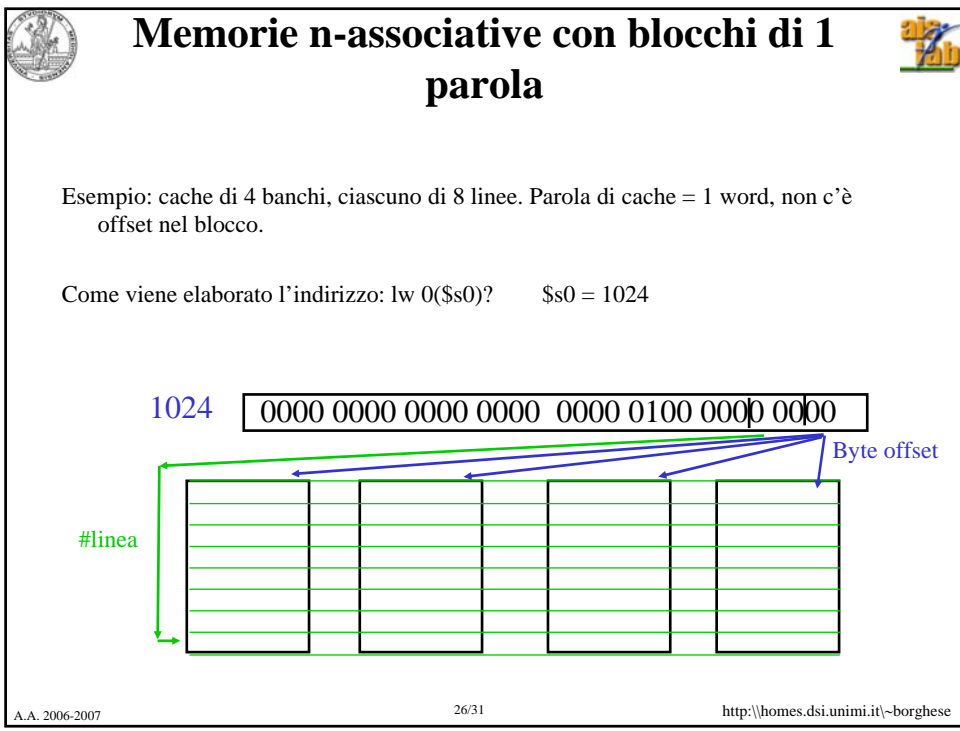
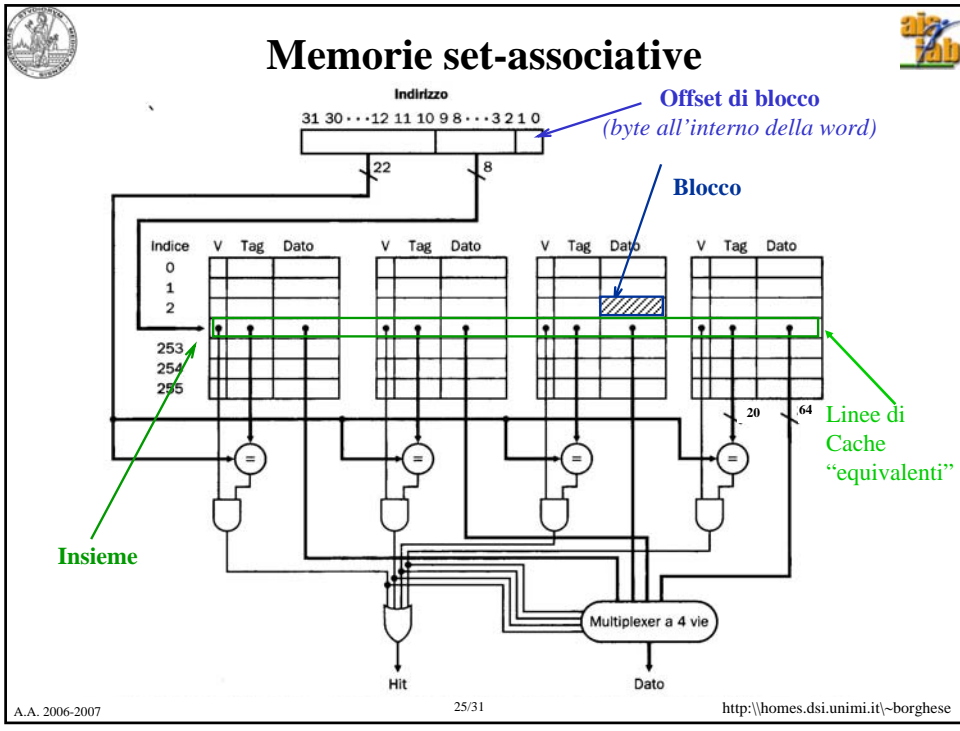
Accesso a cache ad n-way



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. L'indice è costituito da k bit, dove $k = \log_2(\#linee)$. E' analogo al numero di linea nelle cache a mappatura diretta.

TAG – contiene il blocco della RAM a cui appartiene il dato. Cerca il tag di Memoria Principale all'interno dei TAG associati alla linea individuata in ciascun banco.

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



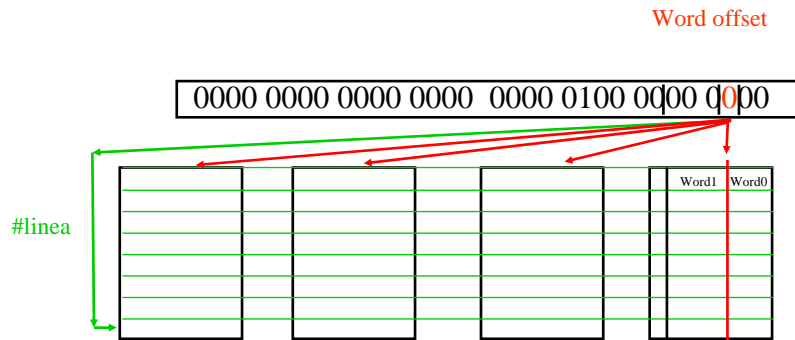


Memorie n-associative con blocchi di 2 parole

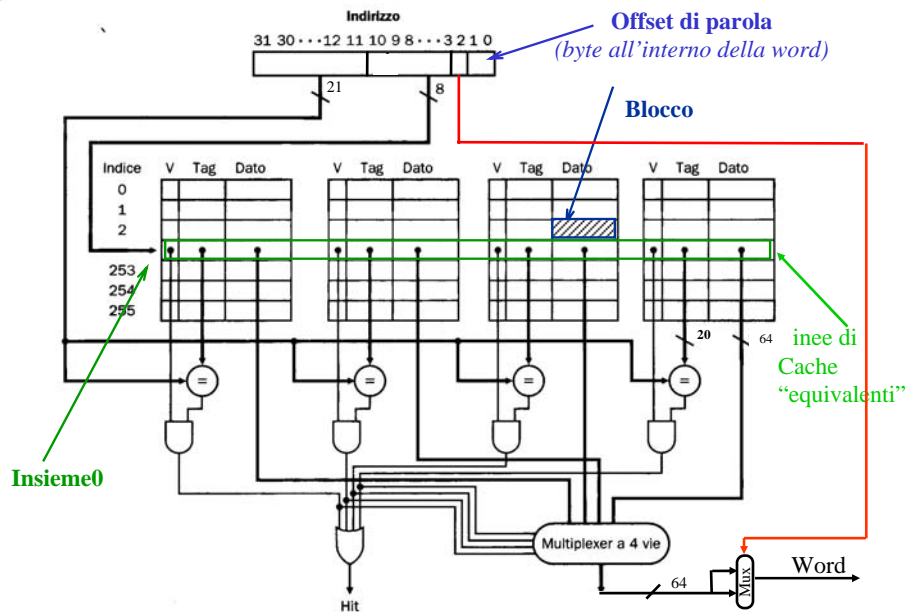


Esempio: cache di 4 banche, ciascuno di 8 linee. Parola di cache = 2 word.

Come viene elaborato l'indirizzo: lw 0(\$s0)? \$s0 = 1024



Memorie set-associative





Criteri di sostituzione di un blocco



Dove inserisco il blocco letto dalla memoria?

Soluzione hardware, algoritmo semplice.

LRU – Least recently Used. Viene associato ad ogni blocco un bit di USE.
Efficiente per memorie a 2 vie.

FIFO – Implementazione tramite buffer circolare.

LFU – Least frequently Used. Associa un contatore ad ogni blocco di cache.

RANDOM – Non funziona molto peggio!!



Dove si può posizionare un blocco di RAM in cache?



Corrispondenza diretta: in un'unica posizione.

Memoria ad 1 via.
 $\#posizioni = \#linee$.

Completamente associative: in n posizioni (n banchi).

Ciascun banco è costituito da 1 linea.
n insiemi o banchi.

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

Ho m insiemi (banchi)
Ciascun insieme è costituito da n linee.



Come si trova un blocco di RAM in cache?



Corrispondenza diretta: indicizzazione.

Controllo del tag del blocco (1 comparazione).

Associativa: ricerca in tutti gli elementi della cache.

n comparazioni: controllo di tutti i tag.

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

N-associativa: ricerca negli m insiemi,

m comparazioni.



Come funziona la scrittura?



Write-through. Scrittura in cache e contemporaneamente in RAM.

Write_buffer per liberare la CPU (DEC 3100)

Sincronizzazione tra contenuto della Memoria Principale (che può essere letto 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 quando devo scaricare il blocco di cache.

Utilizzo un bit di flag: UPDATE, che viene settato quando altero il contenuto del blocco.

Vantaggiosa con cache n-associative.

Alla Memoria Principale trasferisco il blocco.



Cache coherence



Mantenimento dell'informazione di cache coerente tra varie cache (sistemi multi-processori).

Bus watching with write through.

Il controller della cache monitora il bus indirizzi + segnale di controllo write della memoria.
Invalida il contenuto di un blocco se il suo corrispondente in memoria viene scritto.
Quando funziona? Quando tutti i dispositivi utilizzano un meccanismo write-through.

Hardware transparency.

Circuito addizionale attivato ad ogni scrittura della Memoria Principale.
Copia la parola aggiornata in tutte le cache che contengono quella parola.

Noncachable memory.

Viene definita un'area di memoria condivisa, che non deve passare per la cache.



Sommario



Circuito di lettura / scrittura di una cache a mappatura diretta

Memorie associative

Memorie n-associative