


Le memorie Cache
a mappatura diretta e associative

Prof. Alberto Borghese
Dipartimento di Scienze dell'Informazione
alberto.borghese@unimi.it

Università degli Studi di Milano

Riferimento Patterson: 5.2, 5.3


Sommario

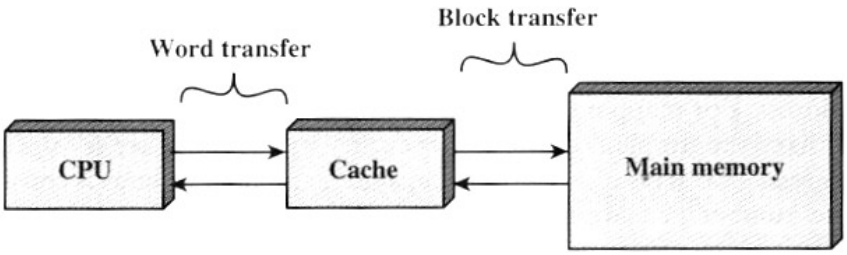
Circuito di lettura / scrittura di una cache a mappatura diretta

Cache 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?

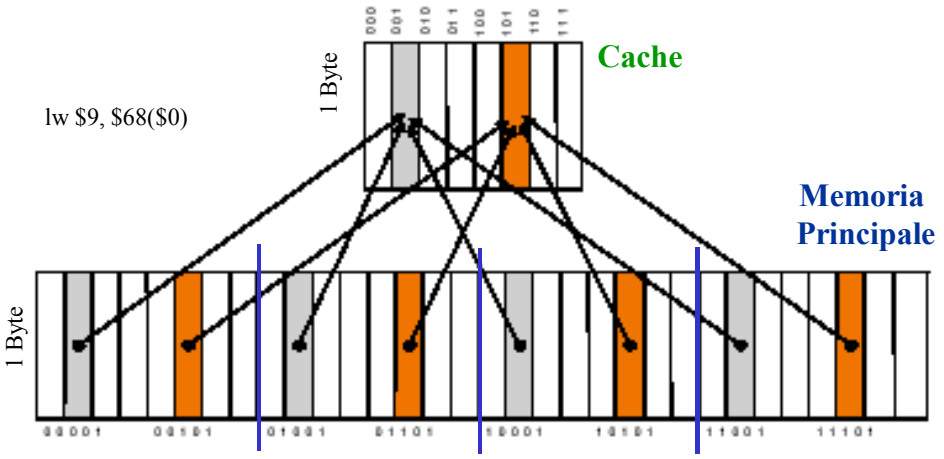
Sottosistema di memoria

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):

Corrispondenza diretta (direct mapped)

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

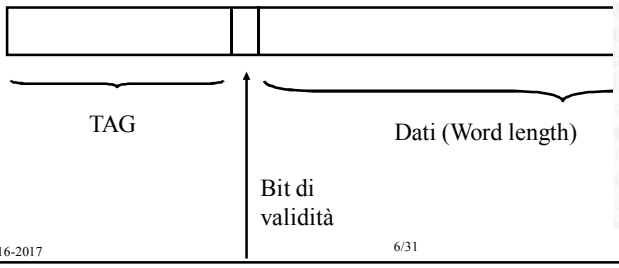


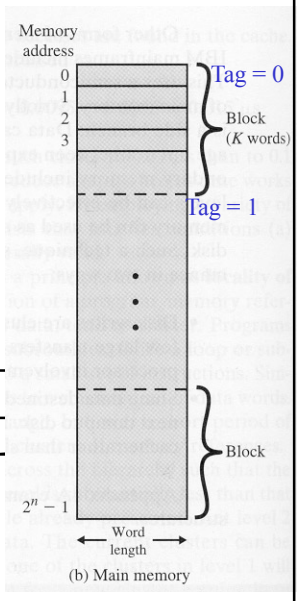
Indirizzi diversi di Memoria Principale corrispondono allo stesso indirizzo di cache.
Quali indirizzi della memoria principale si considerano?

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.

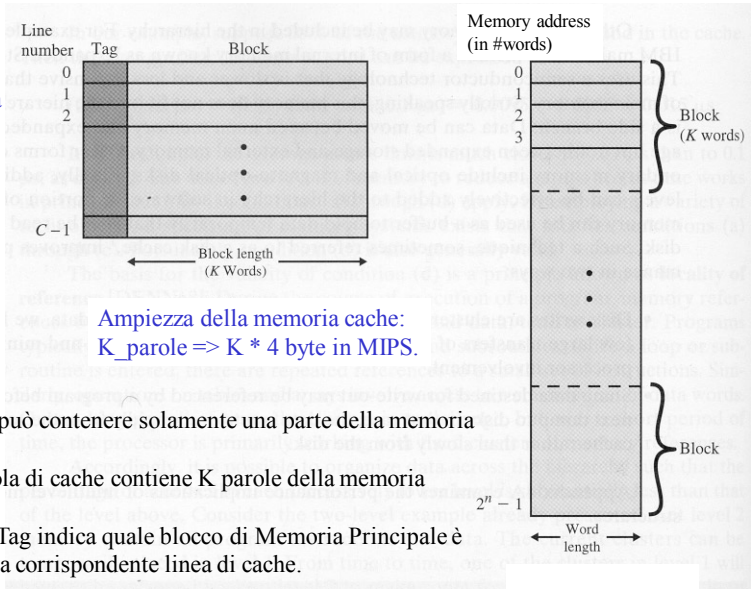
Per blocchi più ampi di una parola, occorre individuare una parola tra le k presenti nella linea di cache.





Mappatura diretta di una cache

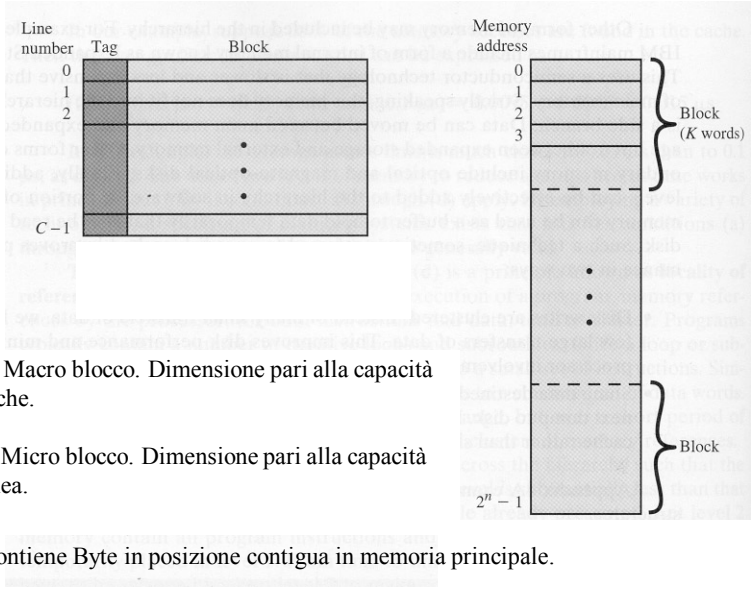
Altezza della memoria cache: # di linee



Ampiezza della memoria cache: K_parole => K * 4 byte in MIPS.

- 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.

Osservazioni



Block – Macro blocco. Dimensione pari alla capacità della cache.

Linea – Micro blocco. Dimensione pari alla capacità della linea.

La linea contiene Byte in posizione contigua in memoria principale.

Come si ottiene l'indirizzo di cache? (Metodo delle divisioni successive)

lw $t0, 14($t1) - carico in cache tutto il blocco associato all'indirizzo $t1 + 14.

Identifico il blocco di Memoria principale a cui appartiene il byte da leggere / scrivere.
Associo il blocco di Memoria principale ad una linea di cache mediante operazione di modulo.

Posizione_byte_cache = indirizzo_Memoria principale (in #byte) *modulo* #byte_cache.
Utilizzo come range di conteggio in RAM, il blocco di cache (# TAG).
Resto1 = posizione in cache.

Posizione_linea = Resto1 (in #byte) *modulo* #byte_linea.
Utilizzo come range di conteggio in Cache, la linea di cache (# linea).
Resto2 = posizione all'interno della linea.

Posizione_byte_linea = Resto 2 (in # byte) *modulo* #byte_word.
Utilizzo come range di conteggio nella linea la dimensione della parola (# parola).
Resto3 = posizione del byte all'interno della parola.

Esempio

La cache con linee di ampiezza pari a 4 parole (blocco = 4 parole) ed altezza di 8 linee:
Il blocco di dati della memoria principale che può essere contenuto in ogni linea di cache, ha dimensioni:
n_byte_x_linea = 4 parole * 4 byte = 16 byte.
La capacità della cache sarà C = **8 linee** * n = **8** * 16 = 128 byte.

•lw $t0, 72($zero) – 72 è il 9° byte della 5ª linea (72 / 16) = 4, **∈ 5ª linea della cache**.

Indirizzo_cache = Indirizzo_Memoria principale *modulo* blocco_cache
Indirizzo_cache = 72 / 16 = 4 → resto 8. {Il resto indica la posizione del primo byte della parola all'interno della linea di cache (blocco) ➔ 9° byte = 1° byte della 3ª parola.}
La word letta è costituita dal byte 72, e dai byte 73, 74, 75, contenuti nella 3ª parola della 5ª linea della cache.

•lw $t0, 204($zero) – 204 / 128 byte = 1 (resto = 76) => mappiamo il 2° blocco di RAM sulla cache.
204 è il 13° byte della 13ª linea (204 / 16) = 12, **∈ 13ª linea (8 + 5)**. La memoria cache ha solo 8 linee.

Indirizzo_cache = Indirizzo_memoria principale (**relativo all'inizio del blocco**) *modulo* capacità_blocco
→ 76 / 16 = 4 → resto 12.{Il resto indica la posizione del primo byte della parola all'interno della linea di cache ➔ 13° byte.}
Il dato viene letto (trasferito nella CPU) assieme ai byte 205, 206, 207) nella stessa linea 5ª della cache.
Il dato viene recuperato (dalla RAM) assieme alle word di indirizzo 200, 196, 192) nella stessa linea 5ª della cache.

Operazione di modulo

L'operazione di modulo se la dimensione del blocco è multiplo della base di conteggio si ottiene scartando alcune delle cifre meno significative.

La divisione per un multiplo di 2 si ottiene come shift a sx, le cifre che "escono" rappresentano il resto.

Parsing dell'indirizzo

0000 0000 0000 00(00) ⟹ 0000 0000 0111 11(11)

128 indirizzi diversi di RAM (32 parole sequenziali di 4 byte)

•Questi 128 byte vengono messi in corrispondeza con i 128 byte di una cache che è organizzata come 8 x 16: 4 parole per linea.

•Gli ultimi 2 bit a destra indicano i byte interni alla parola. Non vengono considerati nei trasferimenti da / per la memoria. Viene considerata la *Word*.

•I 2 bit seguenti indicano la posizione della parola ricercata all'interno della linea della cache.

•I 3 bit seguenti indicano la linea della cache nella / dalla quale si scrive / legge.

•I bit seguenti indicano il numero del blocco della memoria principale messo in corrispondenza con la cache.

Metodo di parsing dell'indirizzo

0000 0000 0000 00(00) ⟹ 0000 0000 0111 11(11)

128 indirizzi diversi di RAM (32 parole sequenziali di 4 byte)

Questo vale per:
- Cache di 8 linee.
- Linee di 4 parole.
- Parole di 4 byte.

Prendiamo un indirizzo ed operiamo per divisioni successive nel caso più generale:

Capacità della cache

M / [(#byte / parola) * (#parole / linea) * #linee] = Numero_blocco di RAM.
Il resto, R_1, rappresenta l'offset in byte all'interno della cache a partire dal byte 0 di cache.

R_1/ [(#byte / parola) * (#parole / linea)] = Numero di linea (di blocco) di cache.
Il resto, R_2, rappresenta l'offset in byte all'interno della linea di cache.

R_2/ [(#byte / parola)] = Numero di parola all'interno della linea di cache.
Il resto, R_3, rappresenta l'offset in byte all'interno della parola.

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) 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.

Esempio di parsing dell'indirizzo

0000 0000 0000 00(00) ⟹ 0000 0001 1111 11(11) 512 indirizzi diversi (128 parole, 4 byte)

La cache con linee di 8 parole (ampiezza) ed altezza di 16 linee:
Il blocco di dati contenuto in ogni linea di cache è di dimensioni: n = 8 * 4 byte = 32 byte.
La capacità della cache è di 16 * 32 byte = 512 byte.

lw $t0, 600($zero)

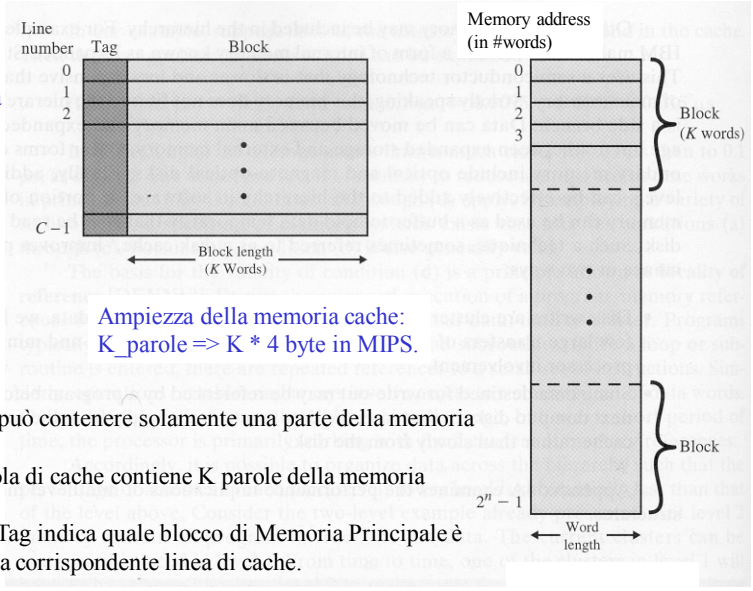
600 / [16 * 8 * 4] = 1 (2° blocco di RAM) con resto R_1 = 600 – 1 * 512 = 88.
Il resto, R_1, rappresenta l'offset in byte all'interno della cache.

88 / [8 * 4] = 2 (3ª linea della cache) con resto R_2 = 88 – 2 * 32 = 24.
Il resto, R_2, rappresenta l'offset in byte all'interno della linea di cache.

24/ 4 = 6 (7ª parola della linea di cache) con resto R_3 = 24 – 6*4 = 0.
Il resto, R_3, rappresenta l'offset in byte all'interno della parola.

Mappatura diretta di una cache

Altezza della memoria cache: # di linee



Ampiezza della memoria cache: K_parole => K * 4 byte in MIPS.

- 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.

Come leggere / scrivere su cache

Individuare la linea della cache dalla quale leggere / scrivere (operazione analoga all'indirizzamento del register file).

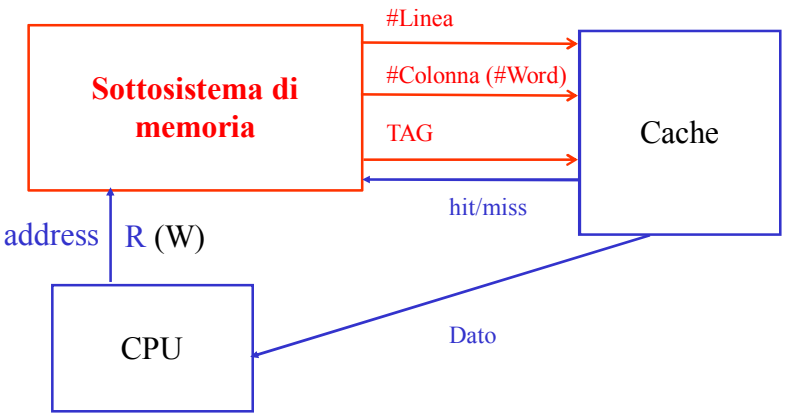
Confrontare il campo tag con il blocco RAM in cui risiede il dato.

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.

Lettura del dato



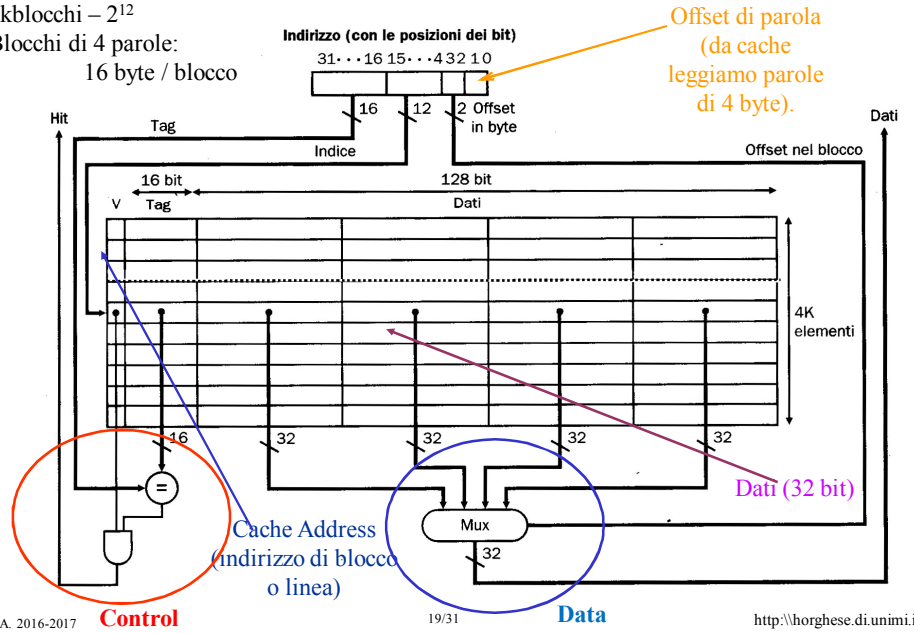
Parte di selezione del dato gerarchica: {#Linea, #Word}

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

Parte di controllo: TAG

Lettura della cache, blocchi > 1 word

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



Scrittura del dato

Parte di selezione del dato gerarchica mediante decoder: {#Linea, #Word}

Parte di controllo: TAG.

Cosa succede se il TAG presente è diverso dal TAG dell'istruzione di `sw` ed il bit di validità è uguale ad 1?

Sommario

Circuito di lettura / scrittura di una cache a mappatura diretta

Cache associative

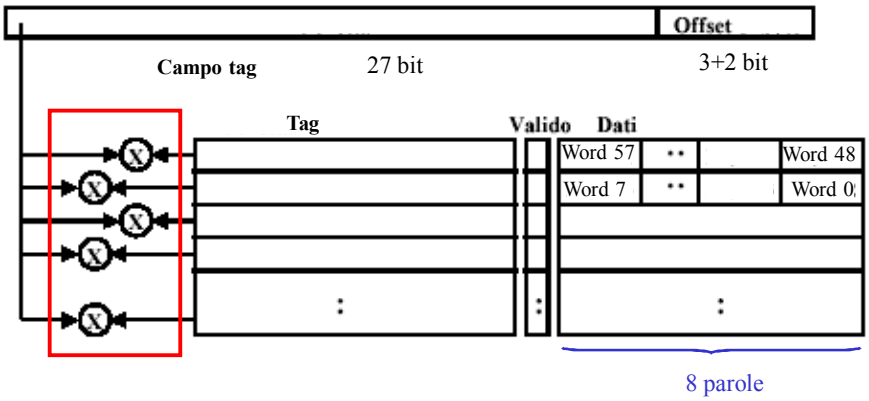
Problemi con le cache a mappatura diretta

- Riempimento non ottimale (a macchia di leopardo).
- MISS per accesso alla stessa linea di cache con dati appartenenti a blocchi diversi di RAM

- Memoria associativa: il contenuto viene recuperato fornendo degli elementi associati al contenuto (e.g. ricerca di testo, ricerca attraverso ontologie WEB).
- Nelle memorie associative si utilizza una parte dell'indirizzo per recuperare il dato.

Occorre quindi sostituire il meccanismo di accesso diretto tramite numero di linea, mediante un meccanismo associativo che determini il numero della linea.

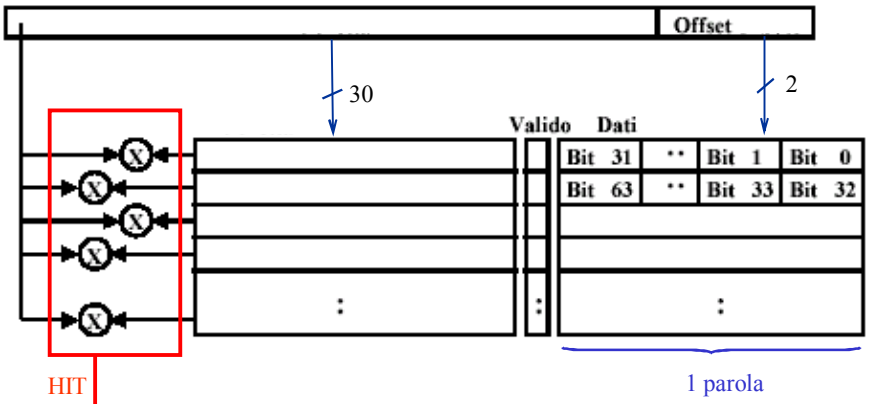
Memorie associative



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?

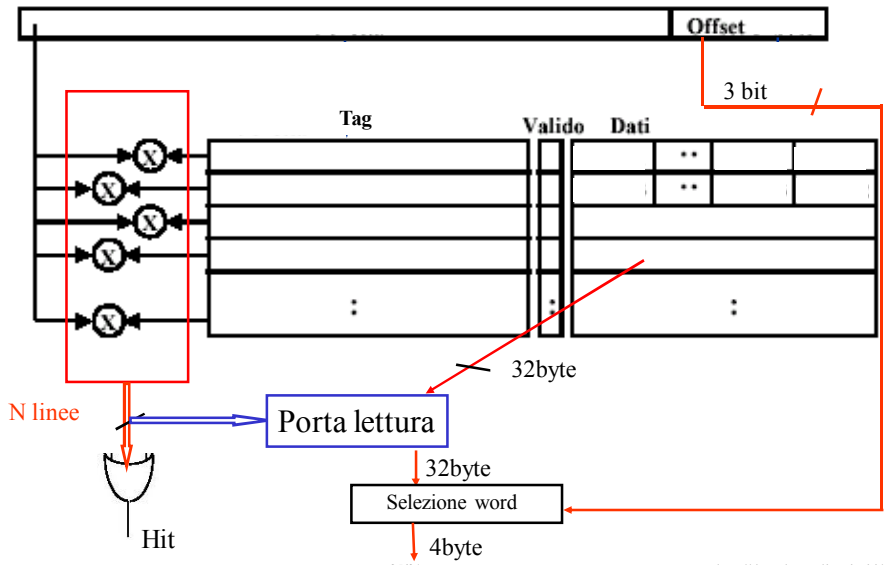
Memorie associative



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?

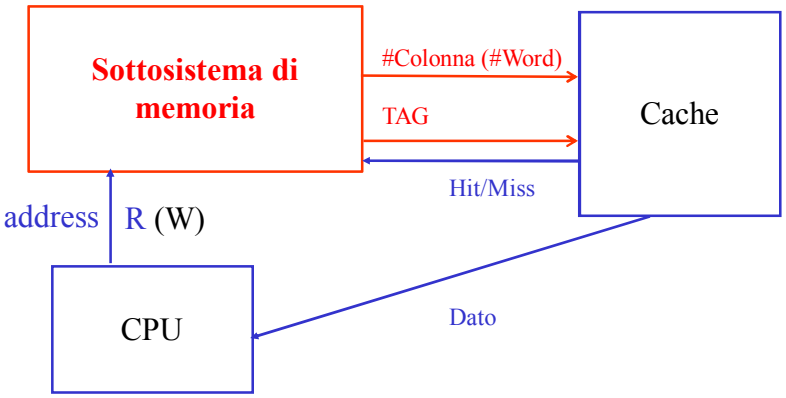
Lettura di una memoria associativa



Alcuni dettagli

- L'uscita del comparatore di una linea va in AND con il bit di validità di quella linea. Il segnale diventa quindi = 1 quando il dato è presente (stesso TAG) ed è valido.
- Le uscite dagli N comparatori, ciascuno associato ad una linea diversa, possono avere al massimo un "1".
- Se colleghiamo le uscite degli N comparatori ad un encoder, otteniamo il #Linea contenende il dato che è il segnale di selezione che cercavamo.

Lettura del dato



Parte di selezione del dato gerarchica: {#Linea, #Word} – la parola è cercata in tutte le linee

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

Parte di controllo: TAG (più ampio che nel caso a mappatura diretta)

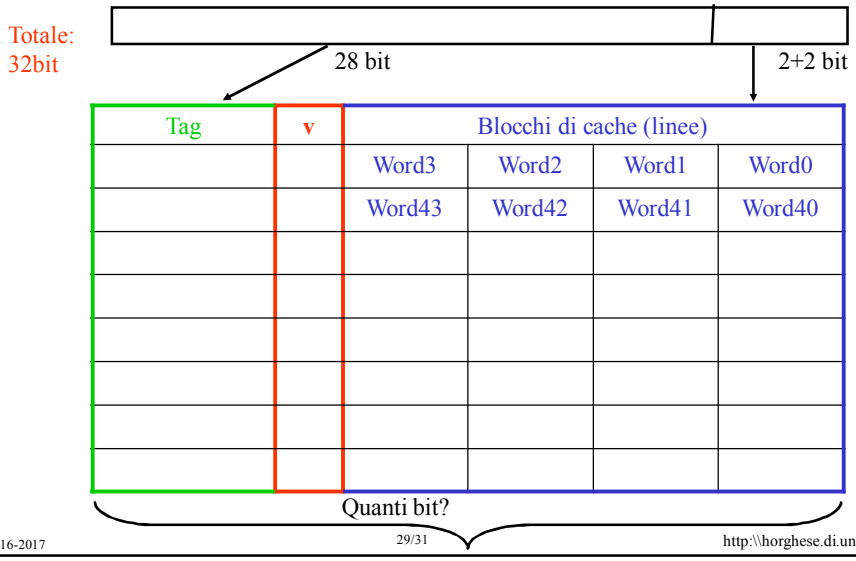
Lettura del dato

Parte di selezione del dato: #Word – La parola è cercata in tutte le linee

Parte di controllo: TAG

Accesso alle memorie associative

Posso accedere alla memoria attraverso l'indirizzo completo modulo la dimensione del blocco di cache (lunghezza della linea di cache).



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 indirizzabili: $2^{(s+w)}$ 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

Cache associative