


Le memorie Cache

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

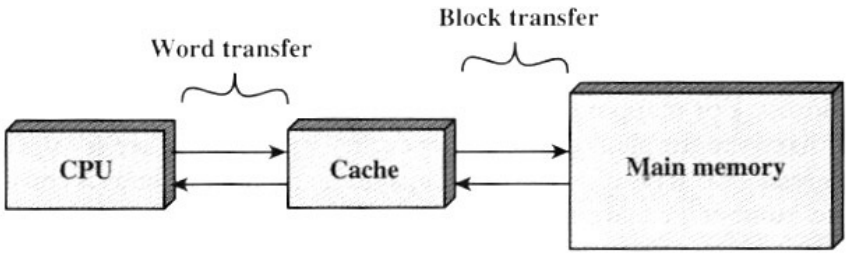
Memorie associative

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?

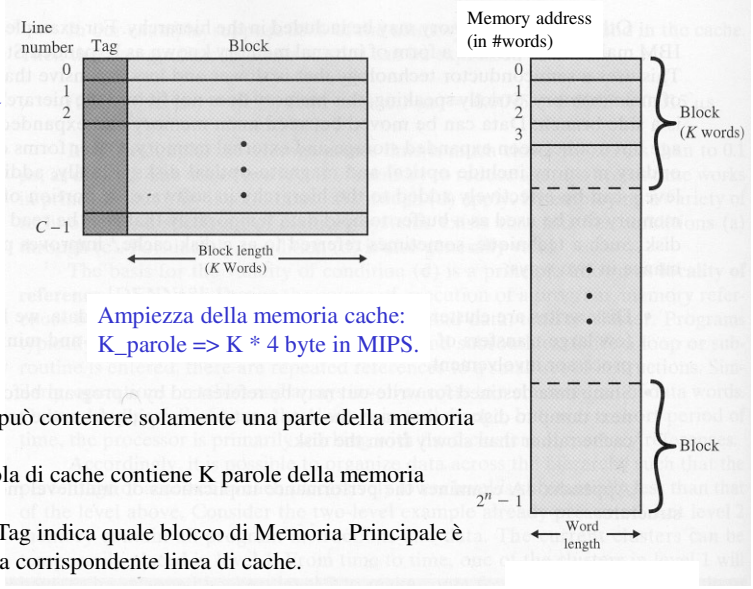
Sottosistema di memoria

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

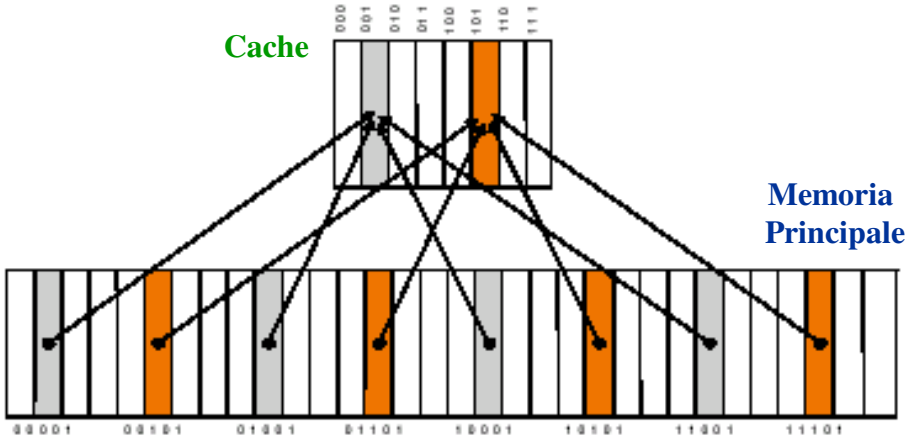


Ampiezza della memoria cache: K_parole => K * 4 byte in MIPS.

- 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.

Corrispondenza diretta (direct mapped)

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

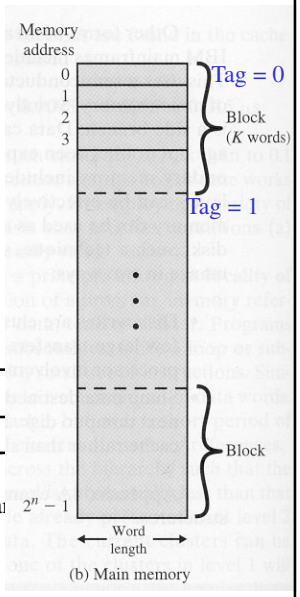


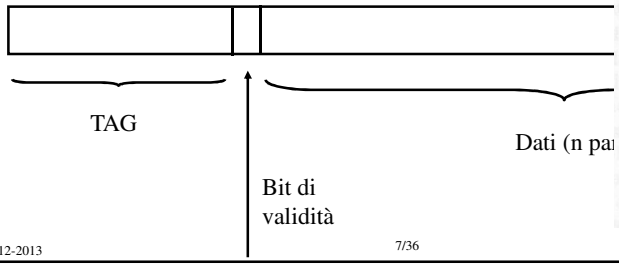
Indirizzi diversi di Memoria Principale corrispondono allo stesso indirizzo di cache.
Quali indirizzi della memoria principale si considerano?

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 0 111 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) 0....0 000000001 100 01 (00)

196 / [4 * 4 * 8] = 1 (2º 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.

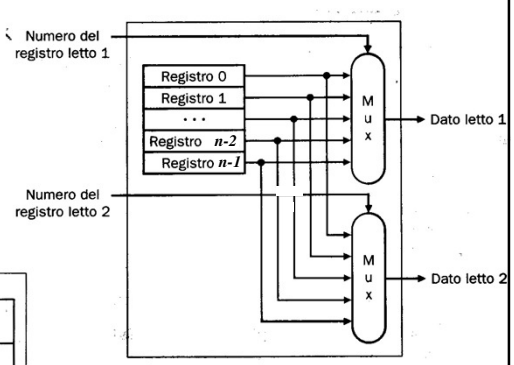
4/ 4 = 1 (2ª parola della cache) con resto R_3 = 4 – 1*4 = 0.
Il resto, R_3, rappresenta l'offset in byte all'interno della parola.

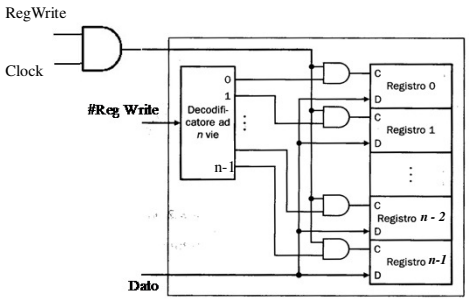
Register file

Il tempo di lettura dipende dal cammino critico dei Mux.

Il tempo di scrittura dipende dal cammino critico del Decoder.

Numero_registro = selettore.

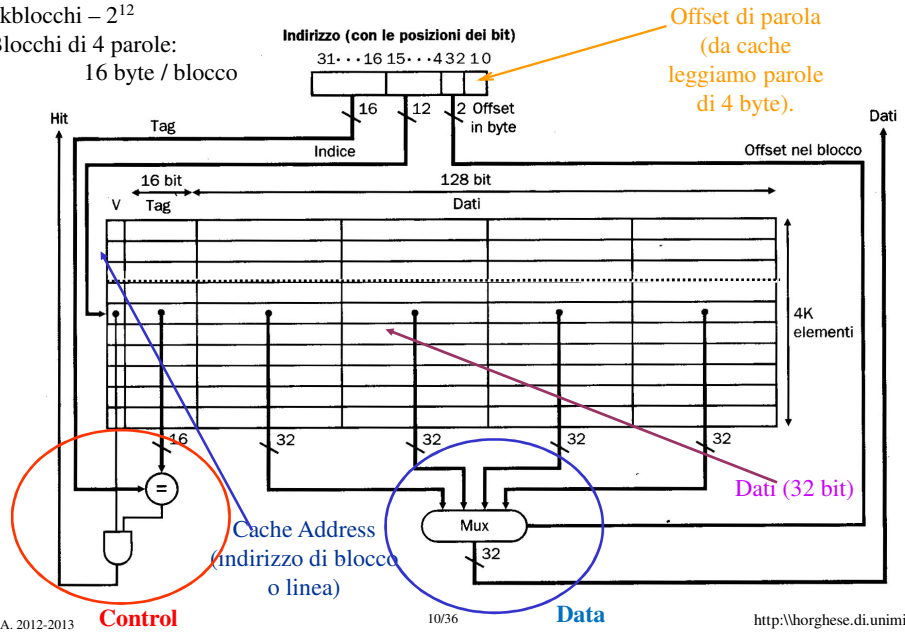




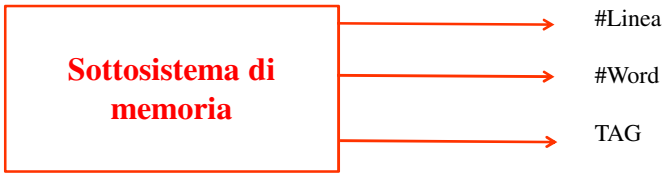
Scrittura cache = scrittura nel register file

Lettura della cache, blocchi > 1 word

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



Lettura del dato



Parte di selezione del dato: {#Linea, #Word}
Schema a 2 livelli (selezione della linea – cf. Register File + selezione della colonna)

Parte di controllo: TAG

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.

			TAG	#L
• lb $t0, 0($t1)	$t1 = 22 (10 110)	MISS	10	6
• lb $t0, 0($t1)	$t1 = 26 (11 010)	MISS	10	4
• lb $t0, 0($t1)	$t1 = 22 (10 110)	HIT	10	6
• lb $t0, 0($t1)	$t1 = 26 (11 010)	HIT	11	4
• lb $t0, 0($t1)	$t1 = 16 (10 000)	MISS	10	0
• lb $t0, 0($t1)	$t1 = 10 (00 010)	MISS	00	2
• lb $t0, 0($t1)	$t1 = 7 (00 111)	MISS	00	7
• lb $t0, 0($t1)	$t1 = 16 (10 000)	HIT	10	0
• lb $t0, 0($t1)	$t1 = 18 (10 010)	MISS	10	2

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.

			TAG	#L	#W
• lb $t0, 0($t1)	$t1 = 22 (1 011 0)	MISS	1	3	0
• lb $t0, 0($t1)	$t1 = 26 (1 101 0)	MISS	1	5	0
• lb $t0, 0($t1)	$t1 = 22 (1 011 0)	HIT	1	3	0
• lb $t0, 0($t1)	$t1 = 27 (1 101 0)	HIT	1	5	1
• lb $t0, 0($t1)	$t1 = 16 (1 000 0)	MISS	1	0	0
• lb $t0, 0($t1)	$t1 = 2 (0 001 0)	MISS	0	1	0
• lb $t0, 0($t1)	$t1 = 7 (0 011 1)	MISS	0	3	1
• sb $t0,0($t1)	$t1 = 8 (0 100 0)	n/a	0	4	0
• lb $t0, 0($t1)	$t1 = 16 (1 000 0)	HIT	1	0	0
• lb $t0, 0($t1)	$t1 = 18 (1 001 0)	MISS	1	1	0

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

Costruire la porta di scrittura della Cache di primo livello.

Sommario

Circuito di lettura / scrittura di una cache a mappatura diretta

Memorie associative

Memorie n-associative

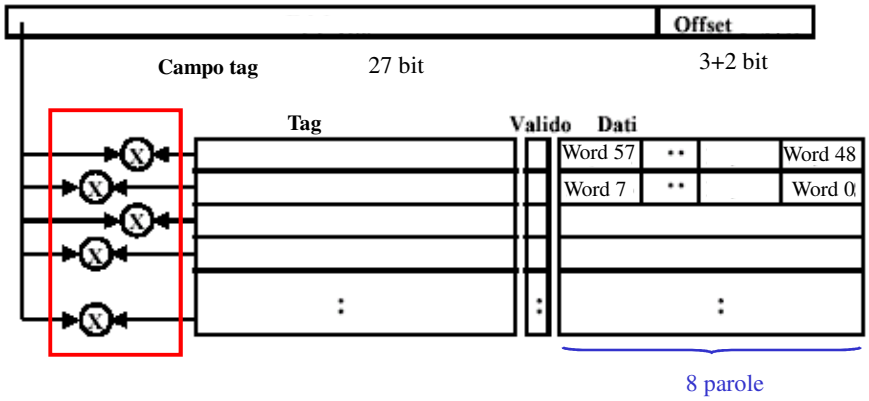
Problemi con le cache a mappatura diretta

- 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.

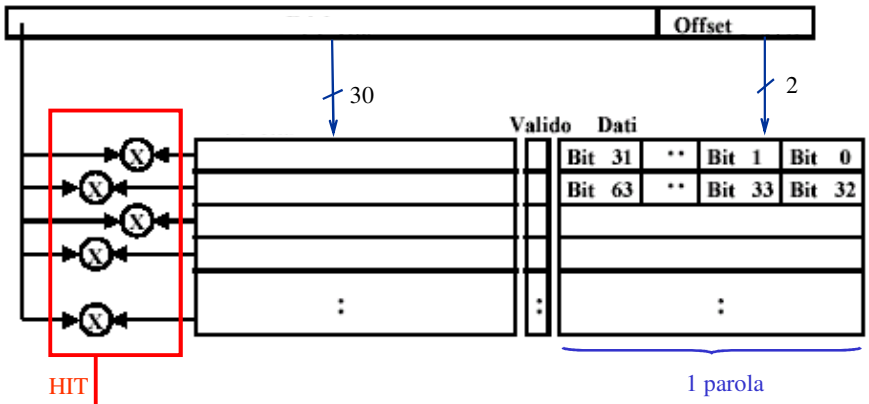
Memorie associative



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?

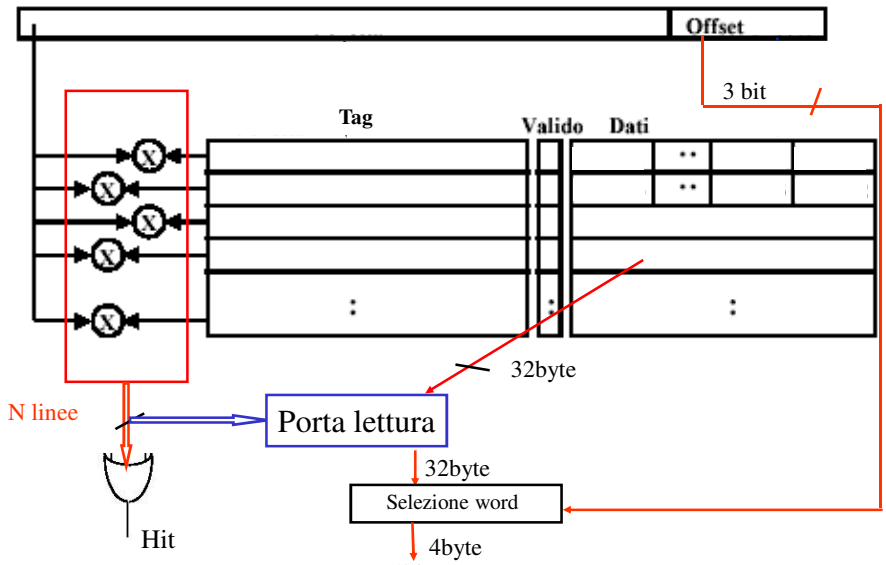
Memorie associative



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.

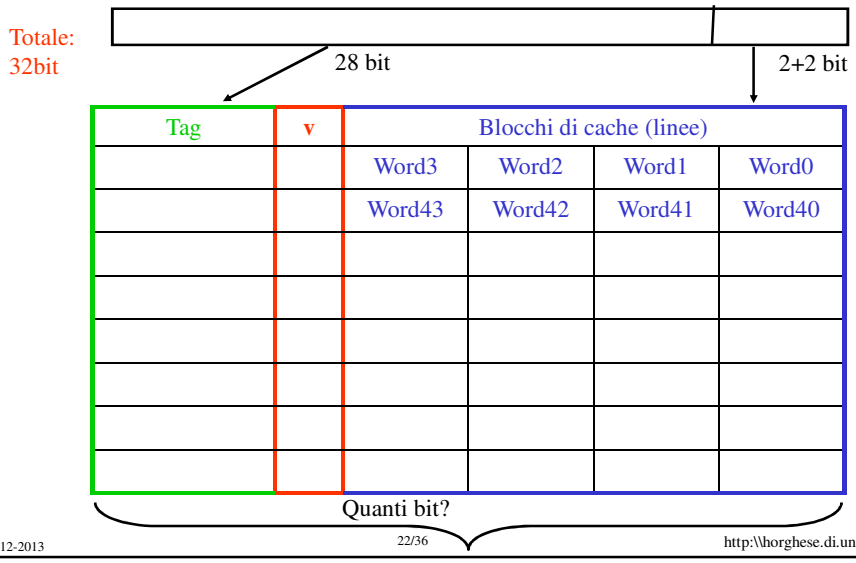
Lettura del dato

Parte di selezione del dato: #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.

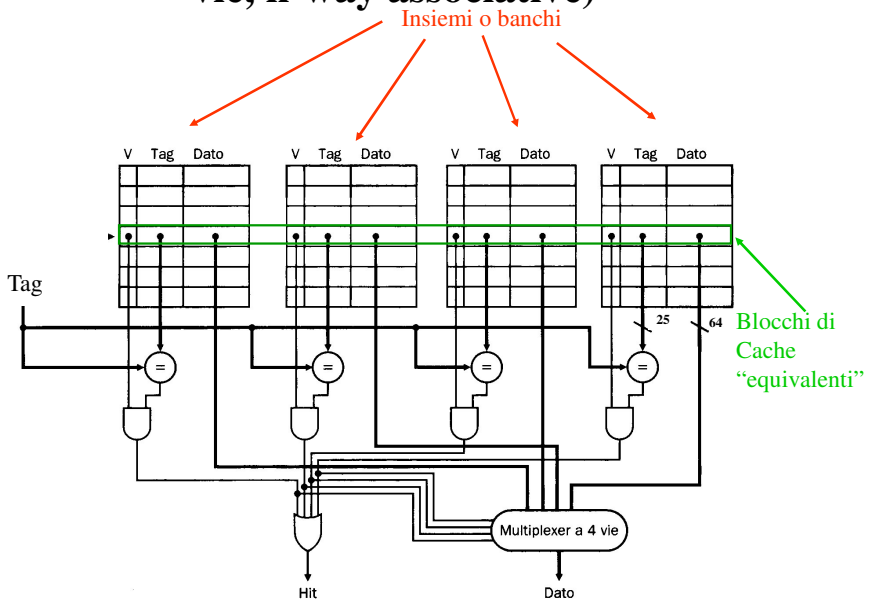
Sommario

Circuito di lettura / scrittura di una cache a mappatura diretta

Memorie associative

Memorie n-associative

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



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.

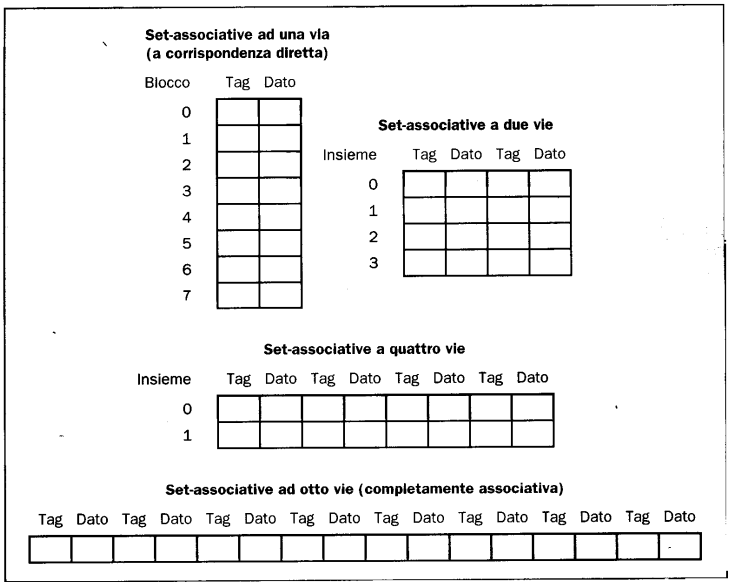
Cache: *è* l'insieme dei banchi più i circuiti che li gestiscono.

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.

Dalle cache a mappatura diretta alle cache associative



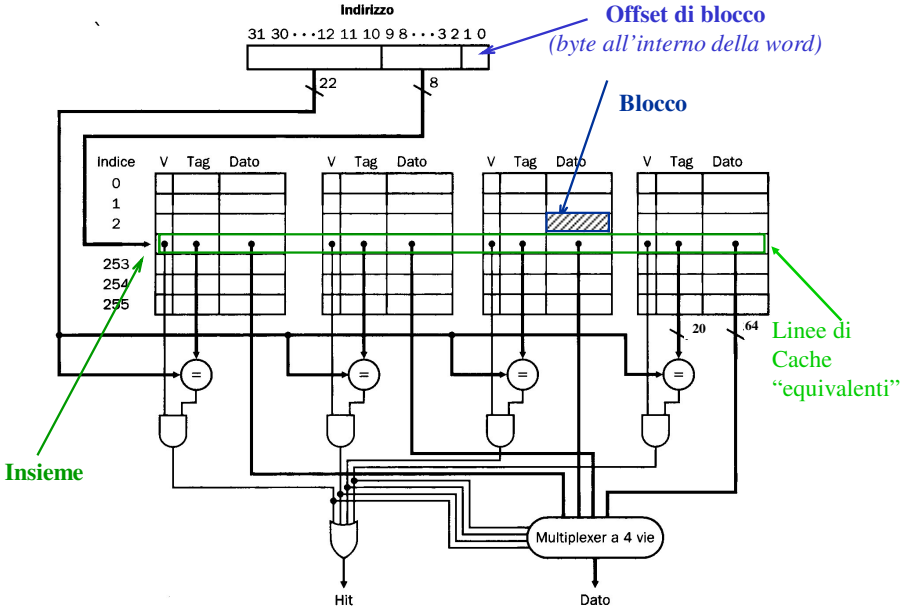
Accesso a cache ad n-vie

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.

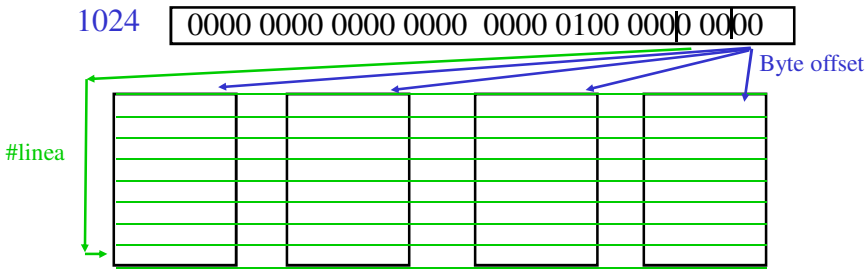
Memorie set-associative



Memorie n-associative con blocchi di 1 parola

Esempio: cache di 4 banchi, ciascuno di 8 linee. Parola di cache = 1 word, non c'è offset nel blocco.

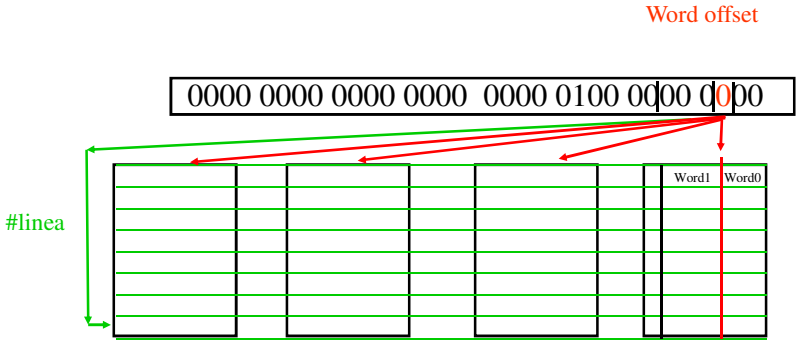
Come viene elaborato l'indirizzo: lw 0($s0)? $s0 = 1024



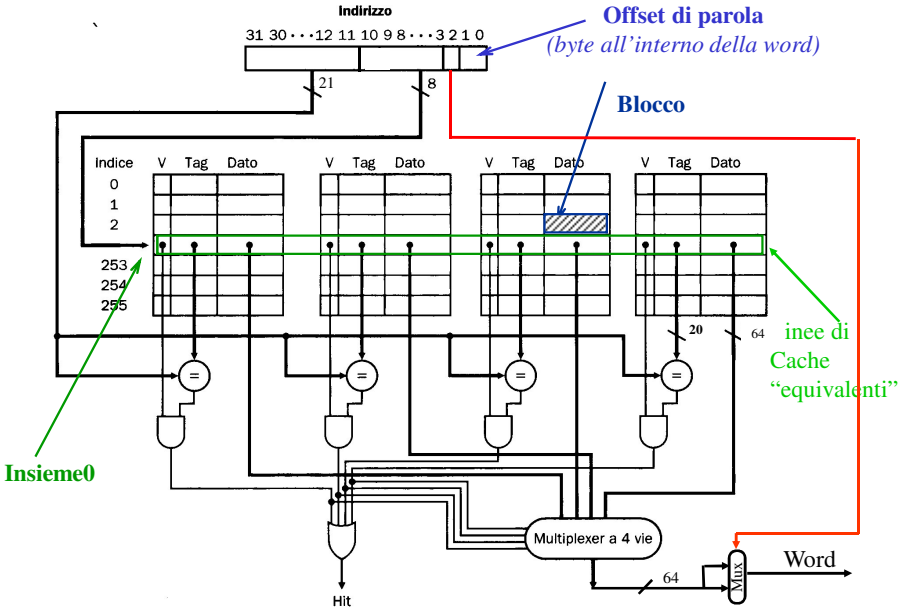
Memorie n-associative con blocchi di 2 parole

Esempio: cache di 4 banchi, 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 RAM?

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.

Sommario

Circuito di lettura / scrittura di una cache a mappatura diretta

Memorie associative

Memorie n-associative