Lezione 18
Introduzione alla memoria cache (2)

http://www.dii.unisi.it/~giorgi/didattica/arcal1

Cache Associative

- Posso fare in modo che sullo stesso indirizzo di cache insistano piu' blocchi. Il numero di blocchi associati ad un dato indirizzo di cache e' fisso per una data cache ed e' chiamato:

  A = "grado di Associativita'" della cache o "numero di vie"

- Detti inoltre
  - B = dimensione del Blocco di cache in byte
  - C = Capacita' della cache in byte

  dove:

  $A \in \{1,2,\ldots,A_{MAX}\}$

  $B = 2^b$ con $b \in \{0,1,2,\ldots,b_{MAX}\}$

  $C = 2^c$ con $c \in \{0,1,2,\ldots,c_{MAX}\}$

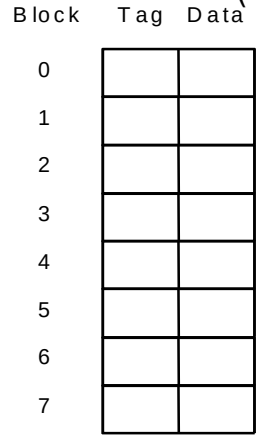
- Sussistono le seguenti relazioni

  $N_C = C / B = 2^{c-b}$ → Numero di blocchi della Cache

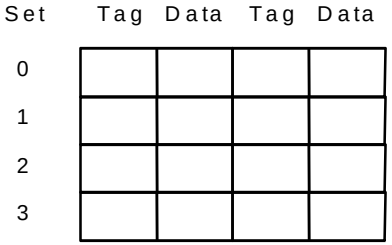
  $N_M = M / B = 2^{m-b}$ → Numero di blocchi della Memoria (essendo M la dimensione della memoria in byte e $m=\log_2(M)$)

Cache Set Associative

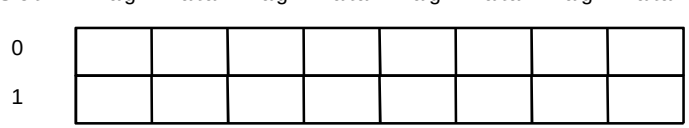
Cache ad 1 via (direct mapped o direct access)



Cache Set-Associativa a 2 vie (A=2)



Cache Set-Associativa a 4 vie (A=4)



- Data una cache ad 1 via ad 8 blocchi, posso organizzare gli stessi blocchi in modo diverso
- Raggruppo i blocchi in insiemi o "set": il numero di blocchi in uno stesso set e' detto A="numero di vie" o associativita' della cache

  $N_S = C / (B \cdot A)$

  → Numero di Set della cache

- L'indirizzo del set sara'

  $X_S = F_{S\text{-}SA}(X_M) = X_M \bmod N_S$

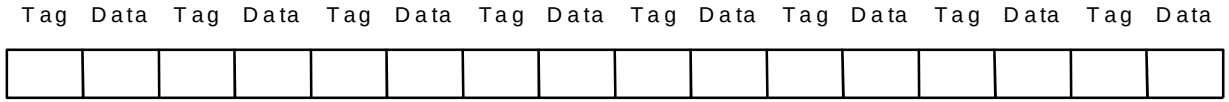
- ...e il tag

  $X_T = F_{T\text{-}SA}(X_M) = X_M \operatorname{div} N_S$

Nota: come caso particolare queste relazioni valgono anche per le cache ad accesso diretto (A=1→N_S=N_C)

Cache Full Associative

Cache Set-Associativa a 8 vie (A=8) → Completamente Associativa (o "Full Associative")



$N_S=1 \rightarrow X_T = X_M$ e $X_S = 0$ (cioe' non occorre indirizzare il set)

- Come individuo il blocco all'interno dello stesso set?
  - Il problema si ha in generale in tutte le cache associative
  - Devo confrontare in parallelo il tag X_T coi tag memorizzati $Y_{T,k}$
  - La funzione di hit sara' piu' complessa
    - $F_H : (\forall k=1...A,\ X_T == Y_{T,k}\ \&\&\ Y_{V,k} == 1\ ?\ 1\ :\ 0)$

Schema logico di una cache 4-way set associative

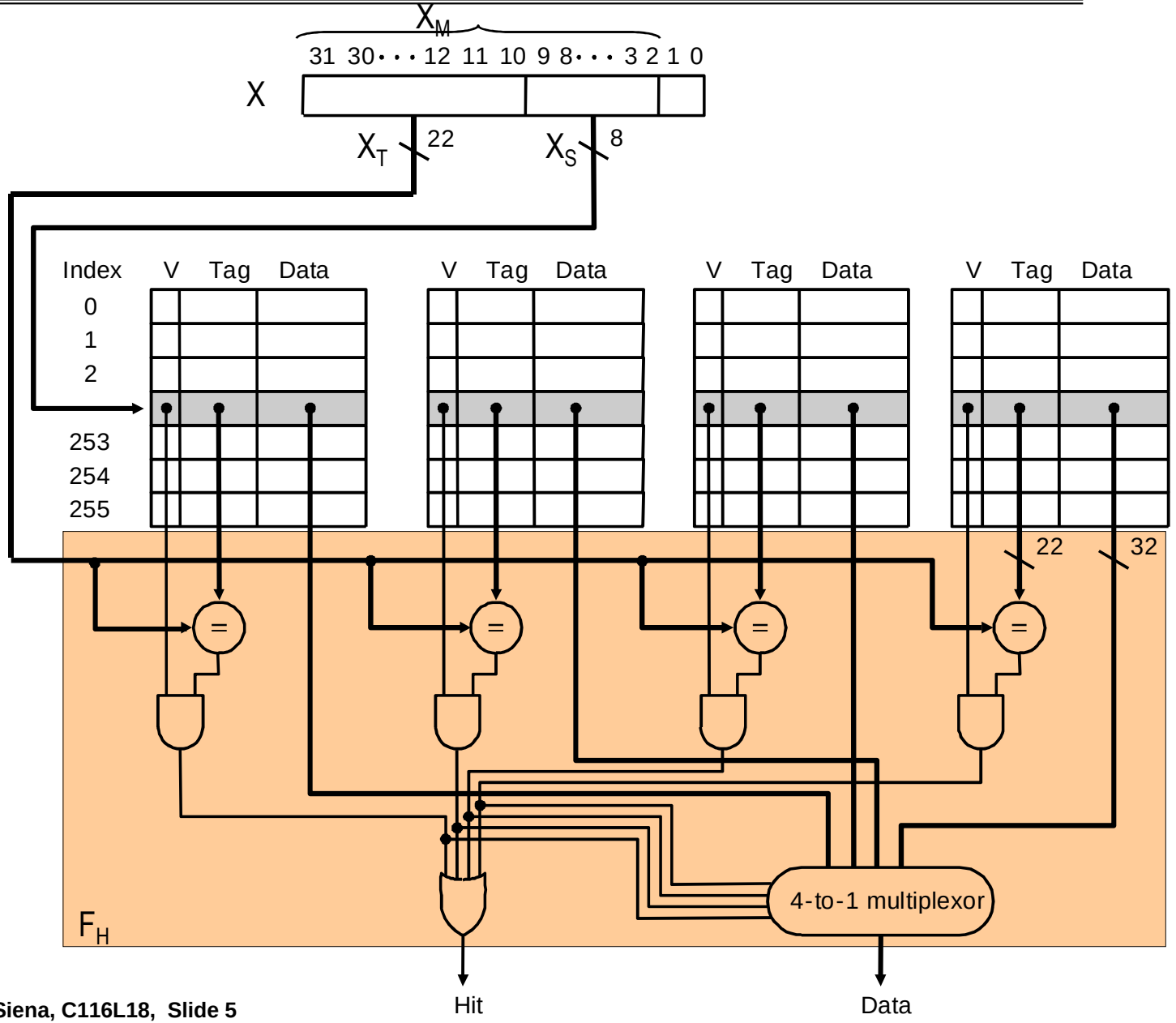
- A = 4
- B = 4 byte
- C = 4 Kbyte
- M = 4 Gbyte

X indirizzo al byte

$X_M = X / B$
$= X / 4$

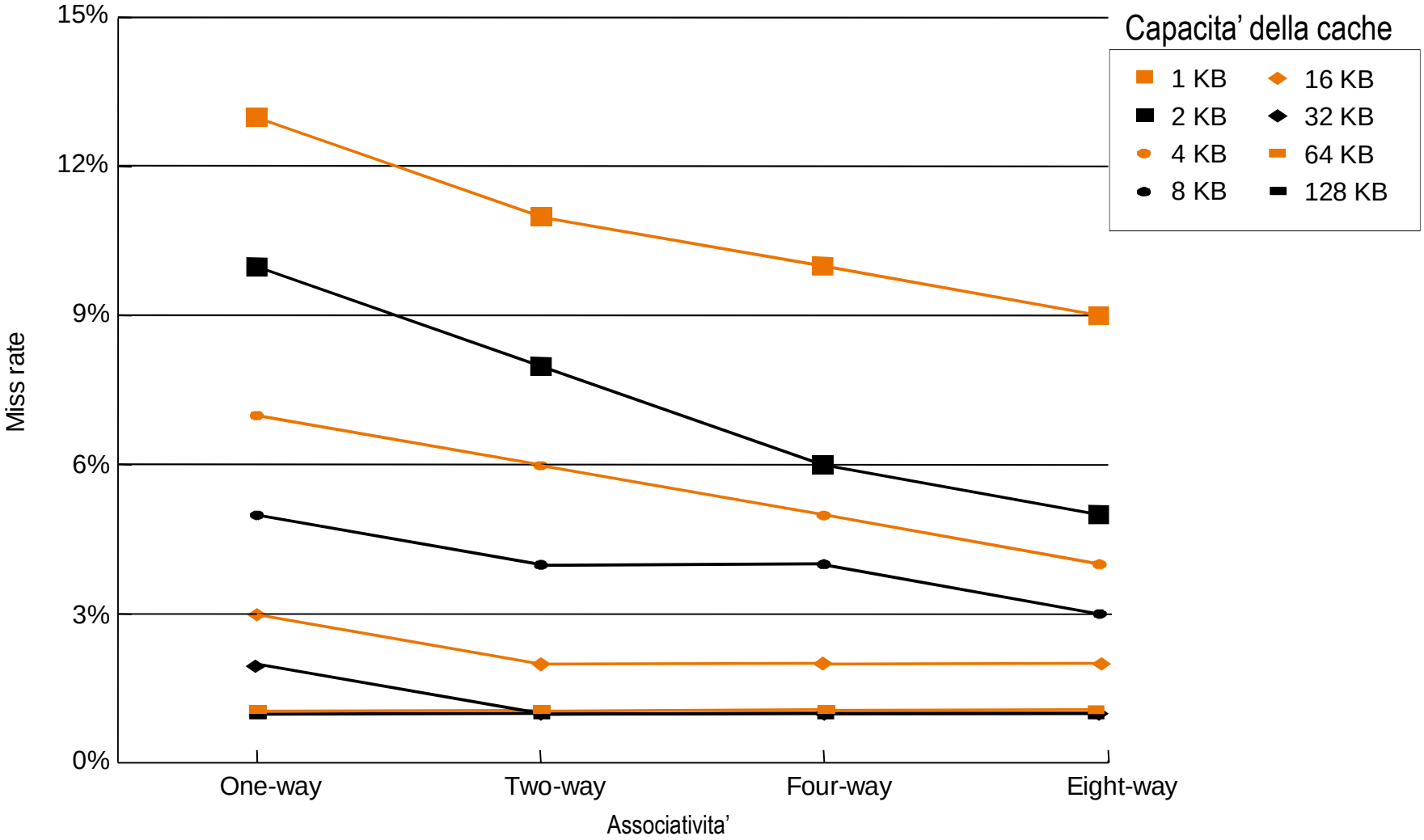
$X_S = X_M \bmod N_S$
$= X_M \bmod C/B/A$
$= X_M \bmod 2^8$

$X_T = X_M / N_S$
$= X_M / (C/B/A)$
$= X_M / 2^8$



Prestazioni: Direct-access vs. Set-associative cache

- L'associativita' tipicamente riduce il miss rate



Politica di Rimpiazzamento

- Come si procede quando la cache e' piena?
  1) Scelgo il blocco da rimpiazzare
  2) Carico il nuovo blocco nella locazione liberata
  3) Leggo o scrivo in quel blocco
- La scelta del blocco puo' avvenire secondo una delle seguenti politiche
  - random: scelgo un blocco a caso
    - Veloce, economica, prestazioni scarse...
  - LRU (Least Recently Used): scelgo il blocco usato meno recentemente
    - Ottime prestazioni, costosa da implementare
    - Spesso viene usato una versione semplificata (es. Intel usa pseudo-LRU) che ha prestazioni un poco inferiori
  - FIFO: scelgo il blocco caricato meno recentemente
    - Non molto usata
    - Ha prestazioni intermedie fra random e LRU

Politica di Rimpiazzamento LRU

- LRU (Least Recently Used)
  - Idea: butto il blocco usato (per una lettura o una scrittura) meno recentemente
  - Vantaggi: sfrutta la localita' temporale → aver usato il blocco poche istruzioni fa, comporta che probabilmente lo riusero'…
  - Svantaggi: con una cache a 2 vie e' facile tenere traccia del blocco LRU (basta un bit); con una cache a 4 vie l'hardware necessario e' abbastanza piu' complesso

	2-vie		4-vie		8-vie	
Capacita'	LRU	Random	LRU	Random	LRU	Random
16 KB	5.2%	5.7%	4.7%	5.3%	4.4%	5.0%
64 KB	1.9%	2.0%	1.5%	1.7%	1.4%	1.5%
256 KB	1.15%	1.17%	1.13%	1.13%	1.12%	1.12%

Politica di Rimpiazzamento LRU: esempio (1)

- Consideriamo una cache set-associative a due vie, con una capacita' totale di 4 word e blocchi da 1 word:

  A=2
  B=4
  C=16
  ⇨ N_S=2

- Supponiamo che i riferimenti X_M (al blocco) generati dal processore siano:

  R 0
  W 2
  R 0
  R 1
  W 4
  R 0
  R 2
  R 3
  W 5
  W 4

- Quanti hit e quanti miss saranno generati se la politica di rimpiazzamento e' LRU?

Politica di Rimpiazzamento LRU: esempio (2)

- Indirizzi 0, 2, 0, 1, 4, 0, ...

0: (X_S=0,X_T=0) → miss, carico nel set 0 (loc 0)

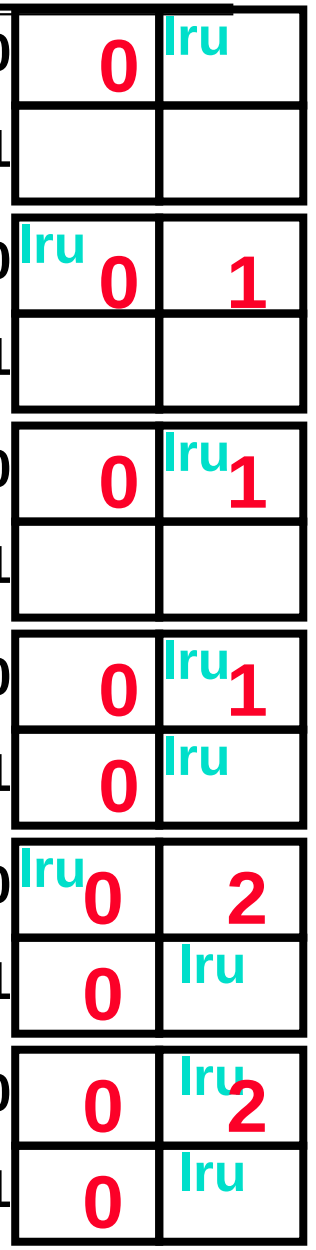
2: (X_S=0,X_T=1) → miss, carico nel set 0 (loc 1)

0: (X_S=0,X_T=0) → hit

1: (X_S=1,X_T=0) → miss, carico nel set 1 (loc 0)

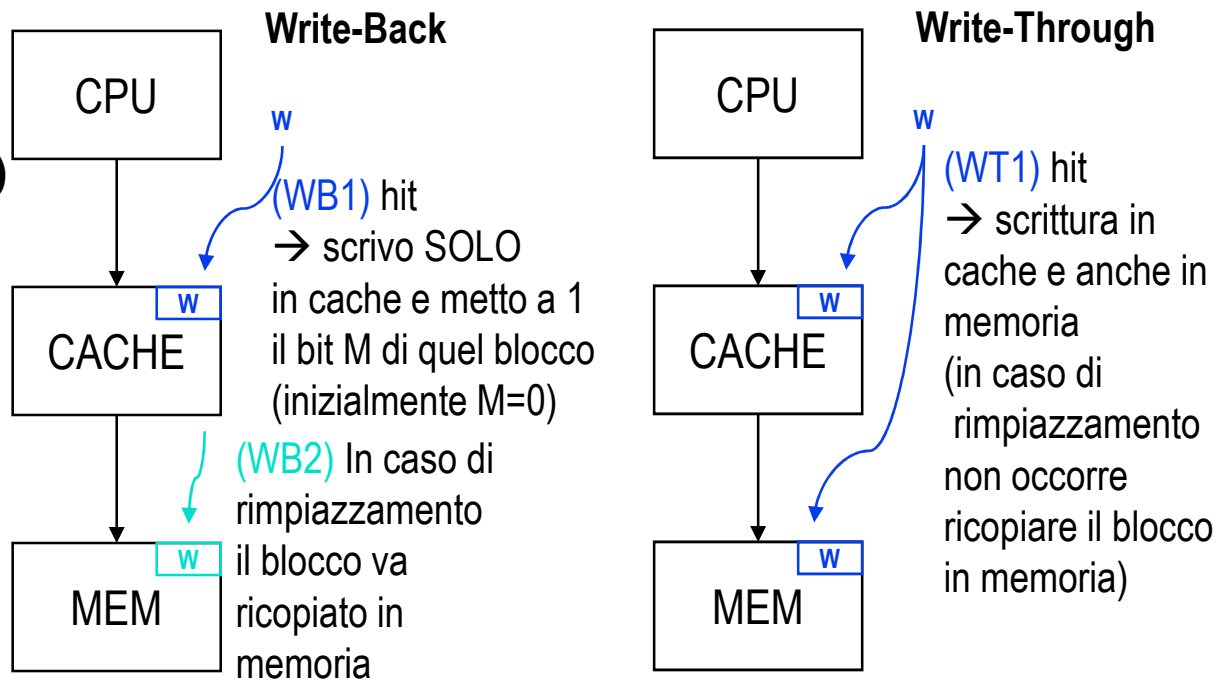
4: (X_S=0,X_T=2) → miss, carico nel set 0 (loc 1, rimpiazzo l'1)

0: (X_S=0,X_T=0) → hit



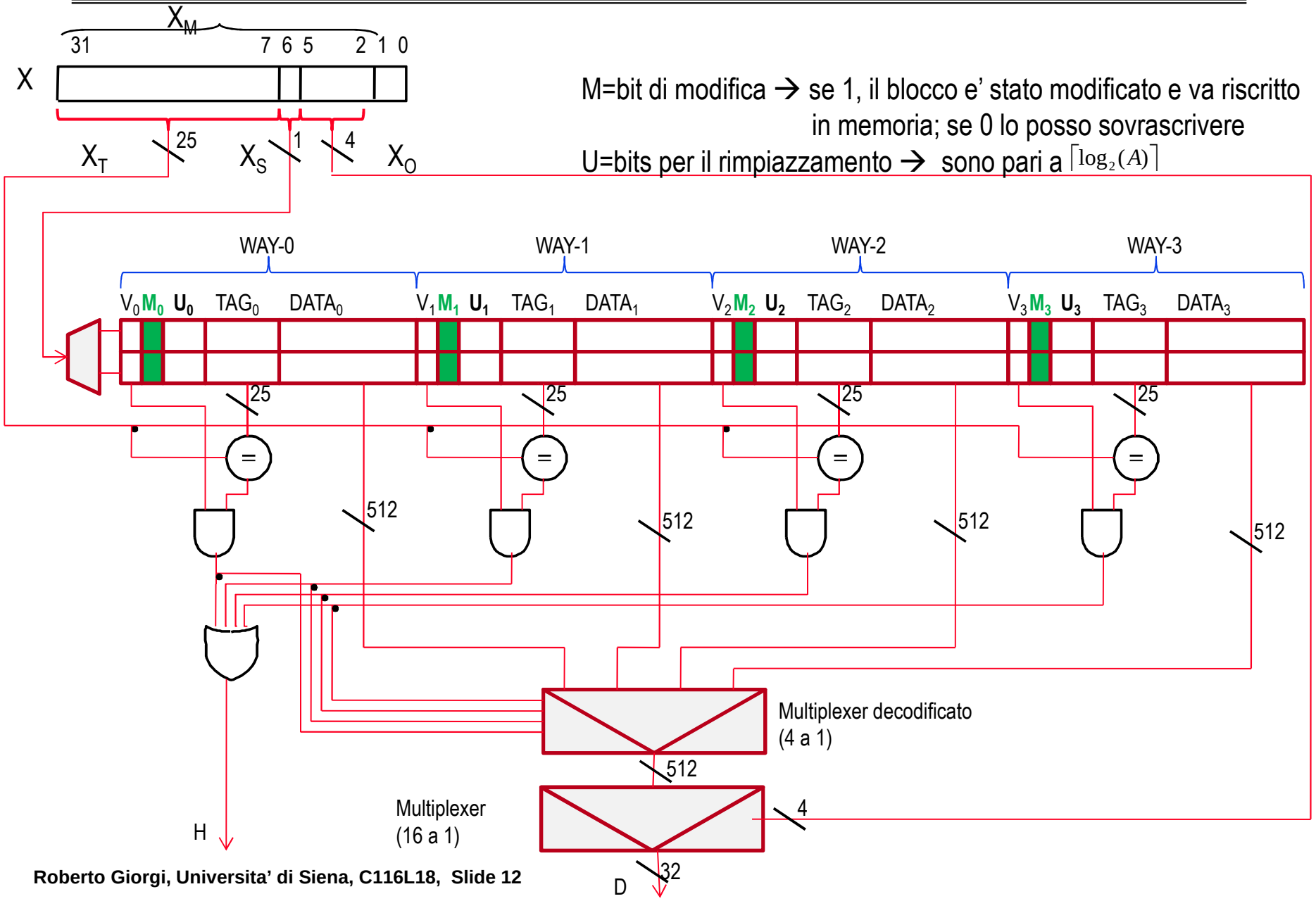
Politica di gestione delle scritture su hit

- Ho due tecniche:
  - Write-Back (WB)
  - Write-Through (WT)



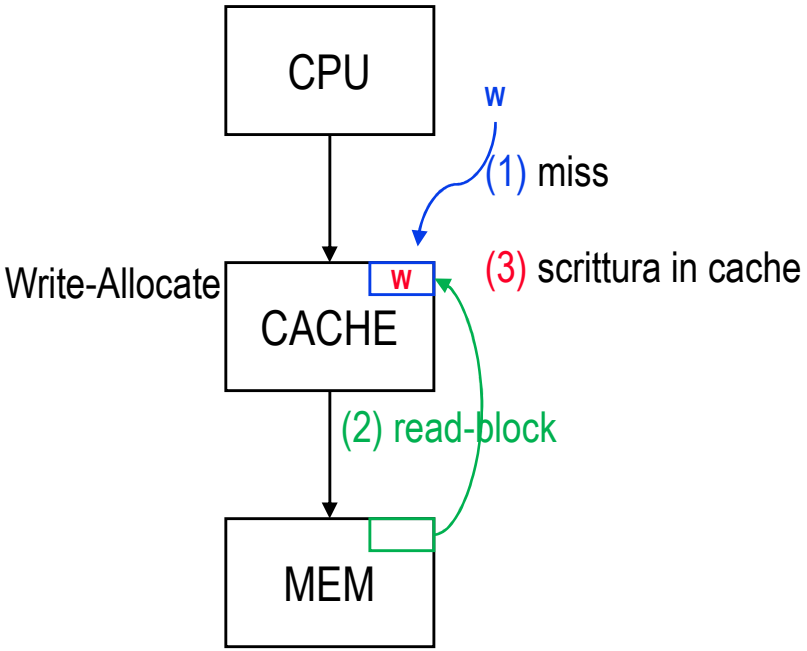
- Confronto Write-Back (WB) / Write-Through (WT):
  - \- WB: occorre un bit M per ogni blocco
  - \+ WB: riduco il traffico sul bus di memoria (write), rispetto a WT
  - \- WB: si introduce un po' di traffico di aggiornamento della memoria

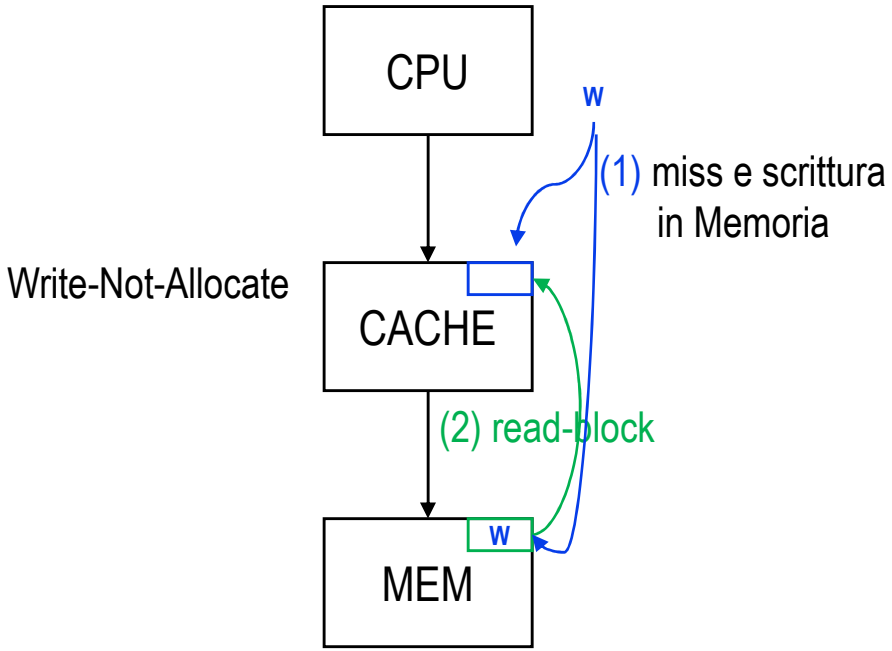
Cache set associativa a 4 vie di tipo Write-Back



Politica di gestione delle scritture su miss: WA/WNA

- Esistono due tecniche
  - Write Allocate (WA)
  - Write Not-Allocate (WNA)

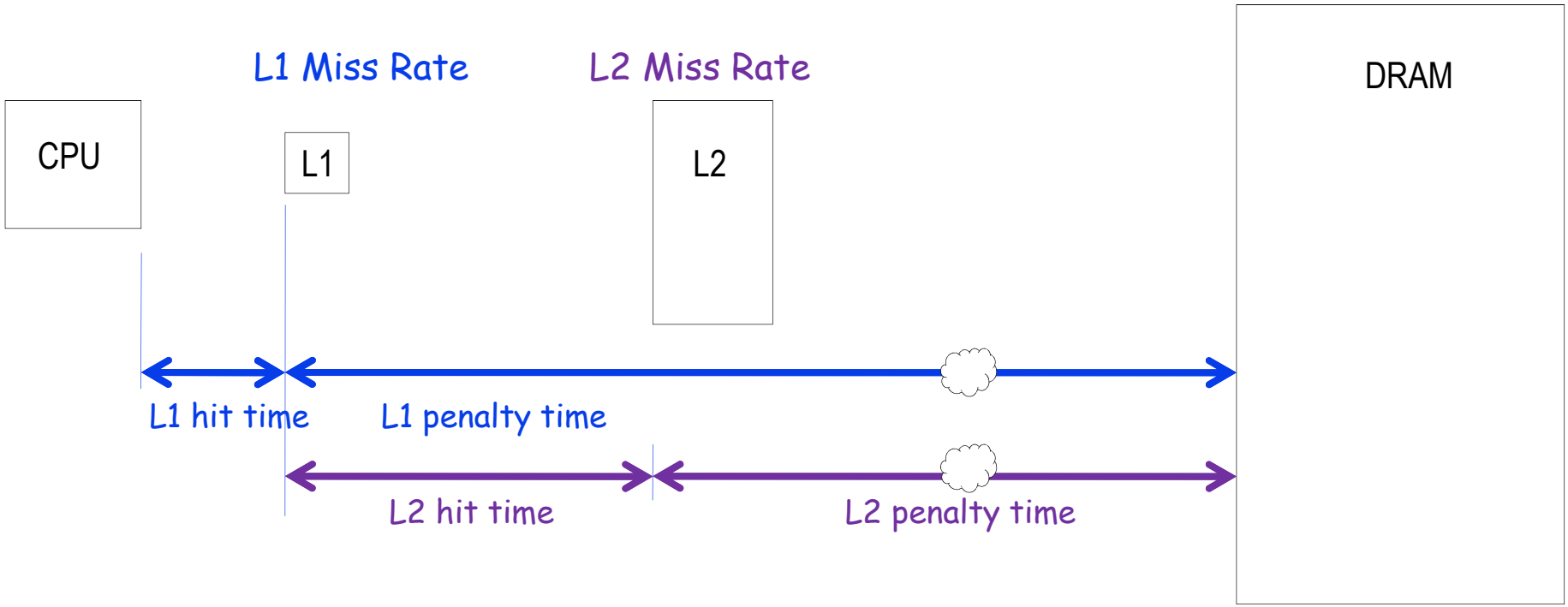




WA, NOTA: Le operazioni 2 e 3 vengono compiute come una singola azione

WNA, NOTA: Normalmente l'operazione 2, NON viene compiuta (il beneficio effettivo di prelevare il blocco o meno puo' dipendere fortamente dal tipo di applicazione)

AMAT per gerarchia di memoria con 2 livelli di cache



AMAT = L1 Hit Time + L1 Miss Rate * L1 Penalty Time

L1 Penalty Time = L2 Hit Time + L2 Miss Rate * L2 Penalty Time

AMAT = L1 Hit Time + L1 Miss Rate * (L2 Hit Time + L2 Miss Rate * L2 Penalty Time)

Valori tipici

- L1
  - Capacita': decine di KB
  - Hit-time: uno (o due) cicli di clock
  - Miss-rate tipici: 1-5%
- L2:
  - Capacita': centinaia di KB
  - Hit-time: qualche ciclo di clock
  - Miss-rate tipici: 10-20%

- I miss di L2 sono una frazione dei miss di L1
  - Perche' allora il miss_rate di L2 e' cosi' alto?

Cache a piu' livelli: esempio

- Supponiamo di avere L1 e L2 con i seguenti dati:
  - L1 Hit-Time = 1 ciclo
  - L1 Miss-rate = 5%
  - L2 Hit-Time = 5 cicli
  - L2 Miss-rate = 15% (% dei miss di L1 che fa miss in L2)
  - L2 Penalty-Time = 100 cicli
- L1 miss-penalty = 5 + 0.15 * 100 = 20
- AMAT = 1 + 0.05 x 20 = 2 cicli

- Supponiamo di avere solo L1 con:
  - L1 Hit-Time = 1 ciclo
  - L1 Miss-rate = 5%
  - L1 Penalty-Time = 100 cicli
- AMAT = 1 + 0.05 x 100 = 6 cicli

→ La cache L2 migliora il tempo medio di accesso di un fattore 3