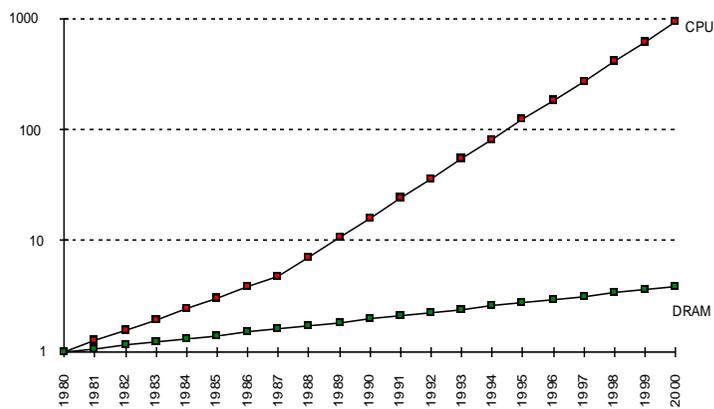


La gerarchia di Memoria

Gap delle prestazioni DRAM - CPU

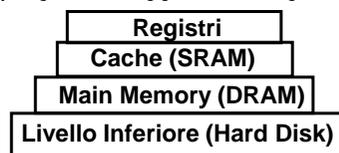


Località ed Organizzazione Gerarchica

- Le RAM statiche sono veloci ma consumano molto ed offrono densità medio/basse
- Le RAM dinamiche consumano poco ed offrono densità molto alte ma sono lente rispetto alla CPU (ed ogni anno il divario aumenta)

Località

- I programmi godono della proprietà di località sia Spaziale sia Temporale
- **Località temporale:**
 - È molto probabile che un'istruzione verrà referenziata nuovamente a breve
- **Località spaziale**
 - È molto probabile che vengano referenziate istruzioni vicine a quella attualmente in esecuzione
- La località dei programmi suggerisce una gerarchia di memoria



Gerarchia di Memoria

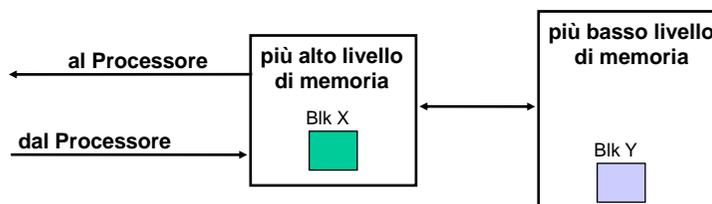
- Livelli più alti
 - Alta Velocità, Alto Costo, Piccole Dimensioni
- Livelli più bassi
 - Bassa Velocità, Basso Costo, Grandi Dimensioni
- Ordine di ricerca di un dato in memoria:
dai livelli più alti ai più bassi
- Il trasferimento di informazioni all'interno della gerarchia avviene tra livelli adiacenti.
- Ogni livello è organizzato in blocchi di n byte

Tempo medio di accesso alla memoria

$$t_{AMAT} = t_s + \text{miss rate} \cdot t_p$$

dove:

- **hit rate:** tentativi riusciti/numero di tentativi
- **miss rate:** miss rate= 1 – hit rate
- $t_s = (t_{acc} + \text{tempo per determinare se dato è nel livello attuale})$
- $t_p = (\text{tempo accesso al livello inferiore} + \text{tempo trasferimento blocco})$



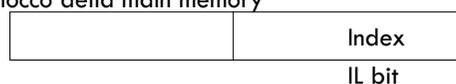
Quattro domande per chi progetta la gerarchia di memoria

- Q1: Dove piazzare un blocco ?
(Block placement)
- Q2: Come faccio a sapere se un blocco è presente?
(Block identification)
- Q3: Quale blocco devo sostituire nel caso di Miss?
(Block replacement)
- Q4: Cosa succede in scrittura ?
(Write strategy)
- Q5: Cosa succede nel caso di miss in scrittura ?
(Write Miss strategy)

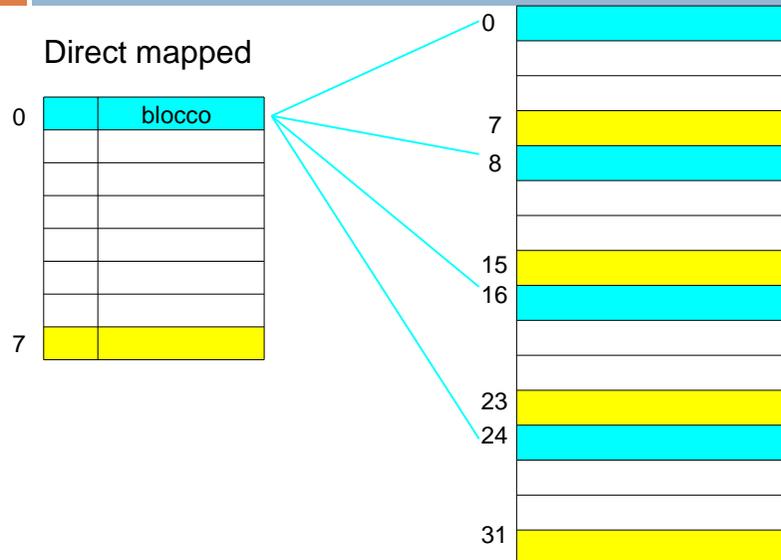
Q1: Dove piazzare un blocco ?

- Full Associative:
 - Un blocco della main memory può essere mappato in un blocco qualsiasi della cache
- Direct Mapped:
 - Data una memoria cache di NB blocchi, il blocco della memoria principale di indice j può essere mappato solo nel blocco della memoria cache di indice
$$\text{Index} = j \text{ modulo } \text{NB}$$
 - L'indice del blocco in cache è ottenuto considerando gli $\text{IL} = \log_2 \text{NB}$ bit meno significativi dall'indice del blocco della memoria principale

Indirizzo di blocco della main memory



Q2: Come faccio a sapere se un blocco è presente?



Q2: Come faccio a sapere se un blocco è presente?

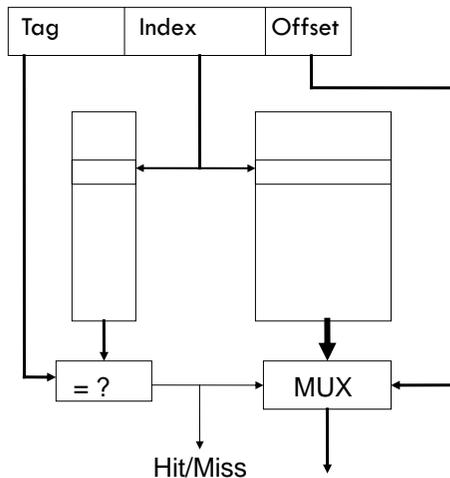
- Data una cache direct mapped di NB blocchi e un main memory di NM blocchi, nello stesso blocco della cache possono essere mappati $NM \div NB$ blocchi.
- Per sapere se il blocco presente in cache è quello effettivamente cercato, oltre a memorizzare il blocco, viene memorizzata una informazione supplementare, il tag del blocco ovvero la parte più significativa dell'indirizzo del blocco che si sta cercando.
- La memorizzazione del Tag richiede $IB = \log_2 (NM \div NB)$ bit

Indirizzo di blocco generato dal processore

Tag	Index	Offset
-----	-------	--------

- Mediante l'index viene individuato il blocco in cache, mediante il Tag viene verificato se il blocco presente è quello cercato
- L'offset individua la word all'interno del blocco

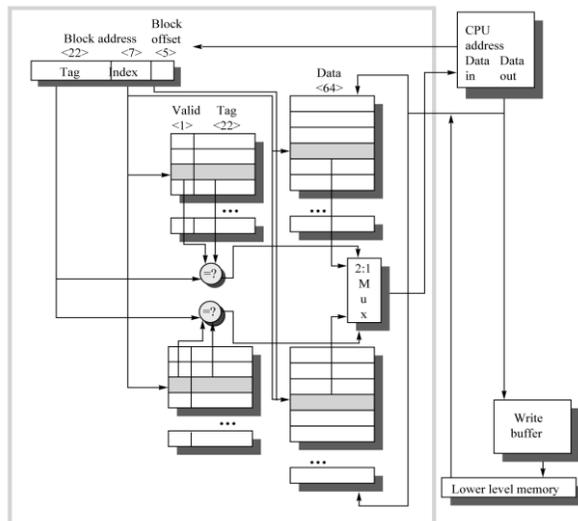
Q2: Come faccio a sapere se un blocco è presente?



Q2: Come faccio a sapere se un blocco è presente?

- Data una cache Set Associative a N vie di NS set e un main memory di NM blocchi, nello stesso set della cache possono essere mappati $NM \div NS$ blocchi.
- La memorizzazione del Tag richiede $IB = \log_2(NM \div NS)$ bit
- Fissata la dimensione della memoria cache, all'aumentare del numero di blocchi di ciascun set diminuisce il numero di set presenti (diminuisce il numero di bit dell'index e aumenta quello del tag)
- Nel caso di memoria full associative possiamo pensare la cache con un unico set e pertanto il tag coincide con l'indirizzo del blocco

Q2: Come faccio a sapere se un blocco è presente?



Q3: Quale blocco devo sostituire nel caso di Miss?

- Random
- Least Recently Used

Associativity:	2-way		4-way		8-way	
	LRU	Random	LRU	Random	LRU	Random
Size						
16 KB	5.18%	5.69%	4.67%	5.29%	4.39%	4.96%
64 KB	1.88%	2.01%	1.54%	1.66%	1.39%	1.53%
256 KB	1.15%	1.17%	1.13%	1.13%	1.12%	1.12%

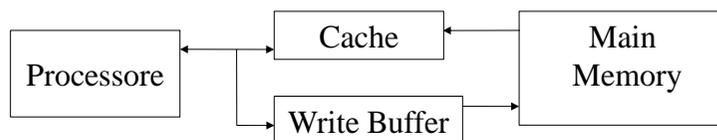
Miss Rate

Q4: Cosa succede in scrittura ?

- **Write through:** L'informazione e' scritta sia in cache sia nel livello inferiore di memoria.
- **Write back:** L'informazione e' scritta solo in cache. Il blocco della cache modificato è scritto in main memory solo quando è sostituito. Ad ogni blocco è associato un dirty bit che viene settato a 1 se il blocco viene modificato, altrimenti resta a 0
- Pro e contro di entrambi
 - WT: un read miss non richiede scritte (in main memory)
 - WB: più scritte sullo stesso blocco (in cache) comportano una sola scrittura in main memory

Q4: Cosa succede in scrittura ?

- WT è combinato con un write buffers in modo da non aspettare il livello inferiore di memoria.



Q5: Write-miss Policy: Write Allocate versus Not Allocate

- ❑ Write allocate: in caso di miss in scrittura il blocco viene caricato in cache e poi viene fatta la scrittura sul blocco
- ❑ Write Not allocate: in caso di miss non viene fatto il caricamento del blocco in cache

- ❑ Tipicamente abbiamo le seguenti combinazioni:
 - Write back con Write allocate
 - Write through con Write Not allocate

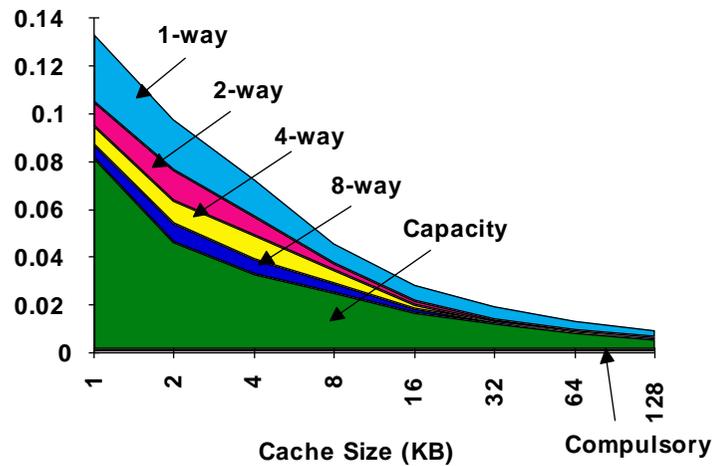
Cause Miss Rate

- ❑ Compulsary

- ❑ Capacity

- ❑ Conflict

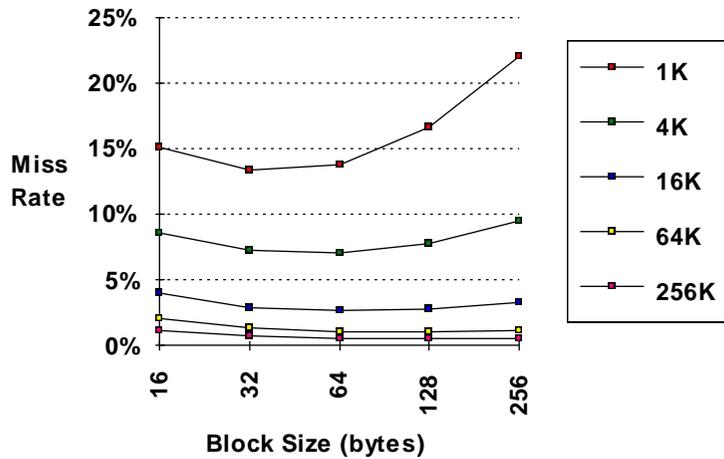
Cause Miss Rate



Elementi che influenzano il miss

- ❑ Dimensione della cache
- ❑ Dimensione del blocco
- ❑ Associatività

DIMENSIONE DEL BLOCCO



Dimensione cache e associatività

Miss rate vs A.M.A.T

Cache Size (KB)	Associativity			
	1-way	2-way	4-way	8-way
1	2.33	2.15	2.07	2.01
2	1.98	1.86	1.76	1.68
4	1.72	1.67	1.61	1.53
8	1.46	1.48	1.47	1.43
16	1.29	1.32	1.32	1.32
32	1.20	1.24	1.25	1.27
64	1.14	1.20	1.21	1.23
128	1.10	1.17	1.18	1.20

Ridurre il miss rate aumentando l'associatività non necessariamente riduce l'AMAT