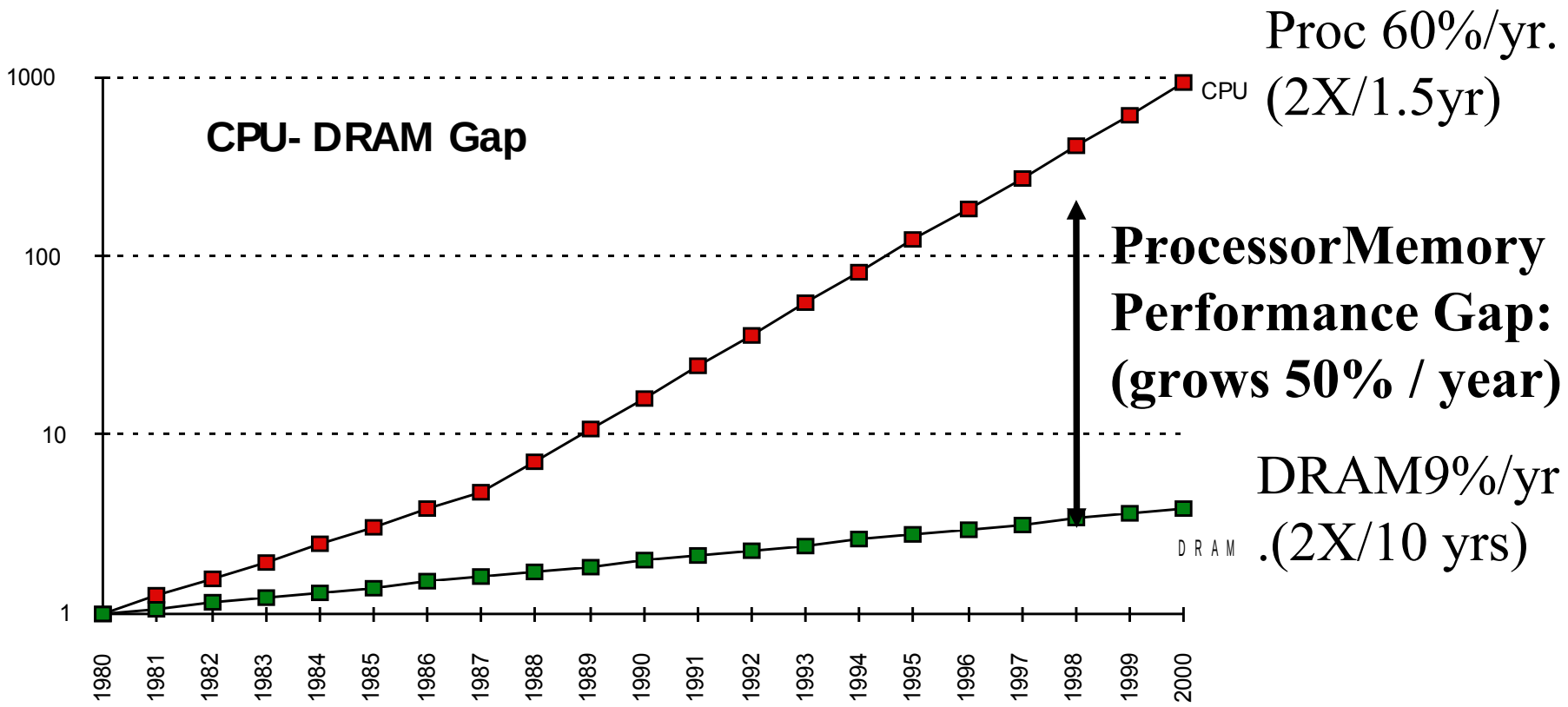
La memoria cache

Gap delle prestazioni DRAM - CPU



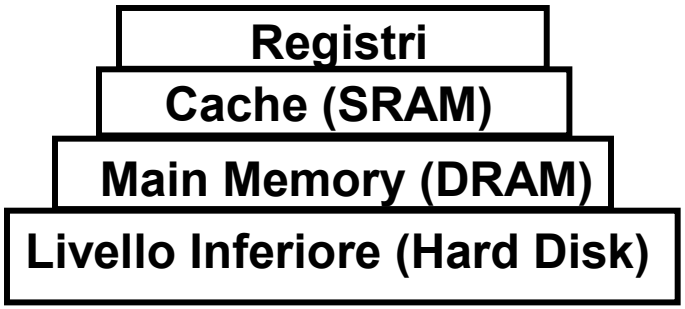
1980: no cache in µproc;
1995 2-level cache, 60% trans. on Alpha21164 µproc

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.

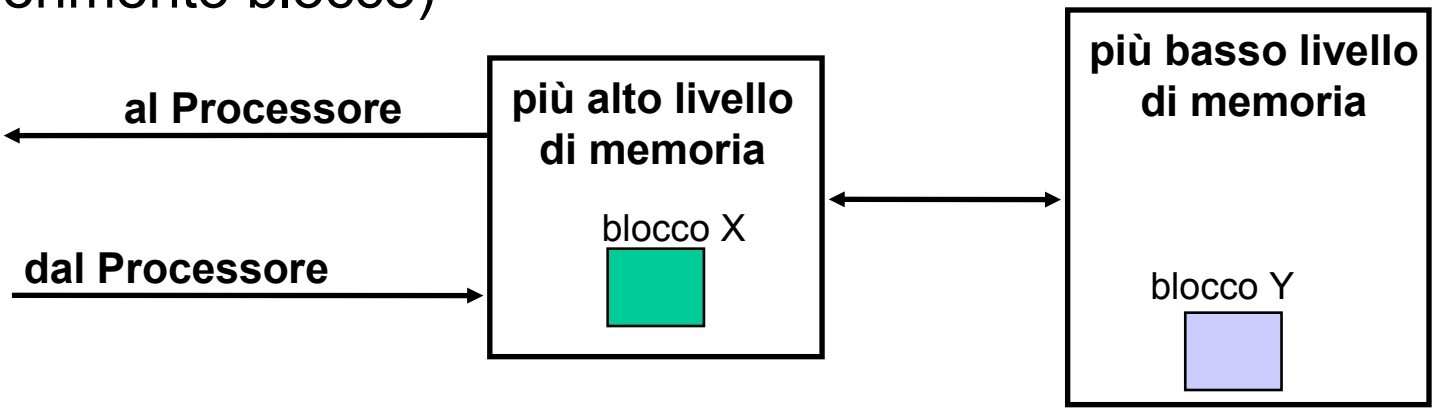
- Ogni livello è organizzato in blocchi di n byte

Tempo medio di accesso alla memoria

t_{AMAT} = hit rate•hit time + miss rate•miss penalty

dove:

- **hit rate:** tentativi riusciti/numero di tentativi
- **miss rate:** miss rate= 1 – hit rate
- Hit time=(t_{acc} + tempo per determinare se dato è nel livello attuale)
- Miss penalty = (tempo accesso al livello inferiore + tempo trasferimento blocco)



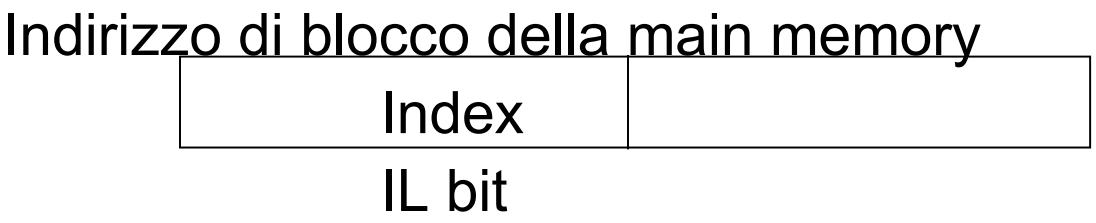
Quattro domande per chi progetta la gerarchia di memoria

- 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)

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

    Index = j modulo NB
  - L'indice del blocco in cache è ottenuto considerando gli IL=$\log_2$NB bit meno significativi dall'indice del blocco della memoria principale

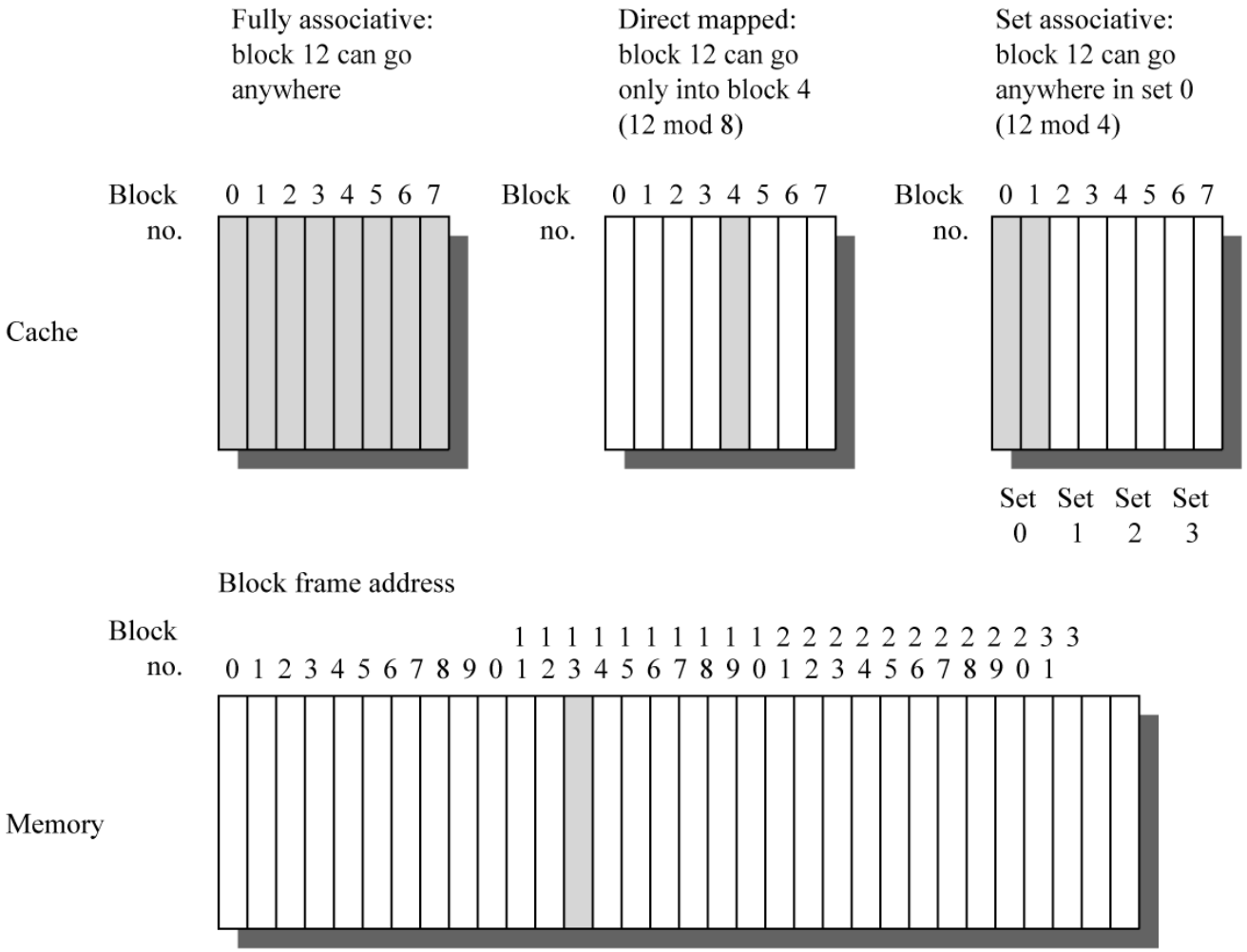


Q1: Dove piazzare un blocco ?

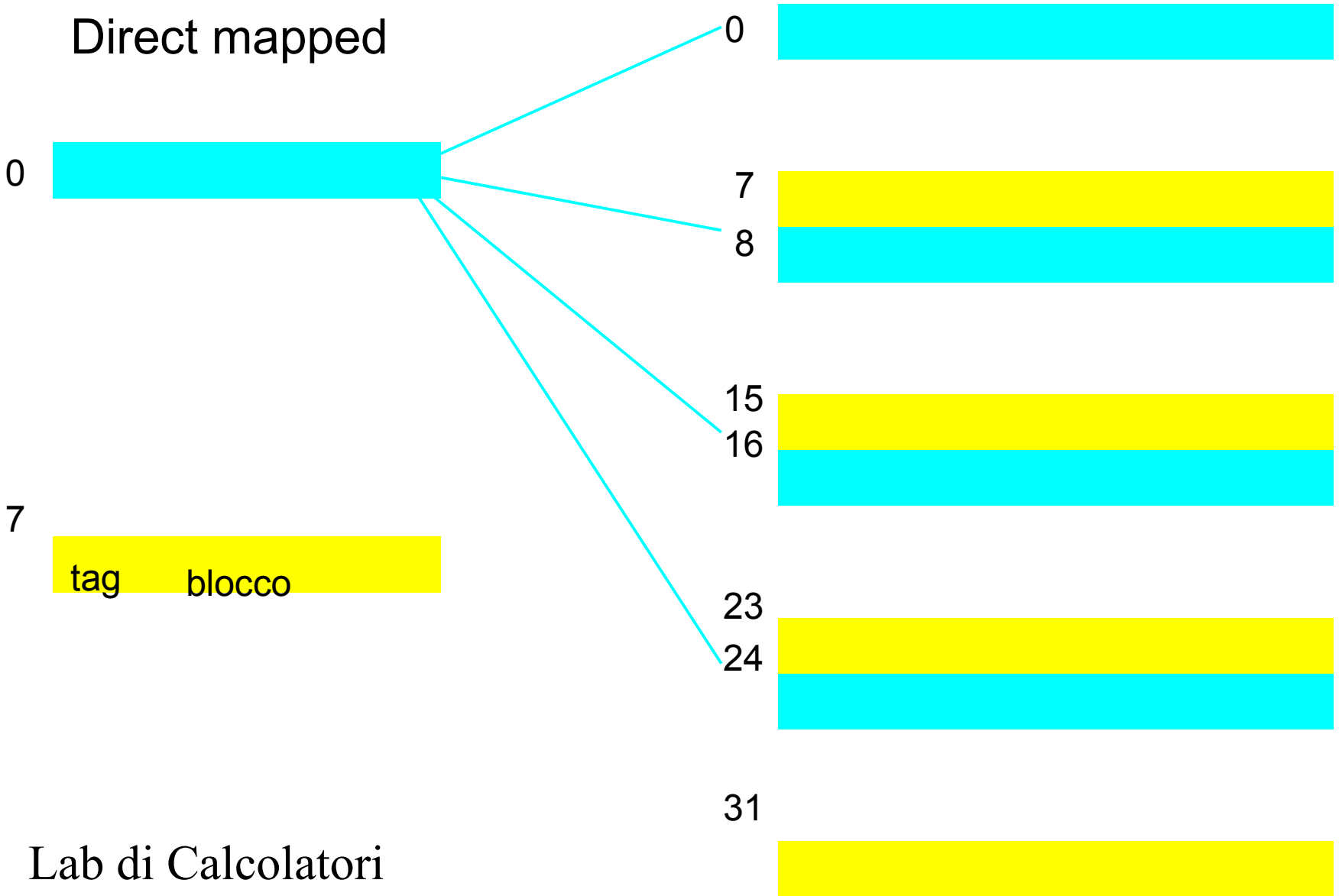
- N-way Set Associative
  - Data una memoria cache di NS set, ciascuno di N blocchi, il blocco della memoria principale di indice j può essere mappato nel set della memoria cache di indice

    Index = j modulo NS
  - All'interno del set un blocco può essere piazzato in una posizione qualsiasi.
  - L'indice del set in cache è ottenuto considerando gli $IL=\log_2 NS$ bit meno significativi dall'indice del blocco della memoria principale.

Q1: Dove piazzare un blocco ?



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



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

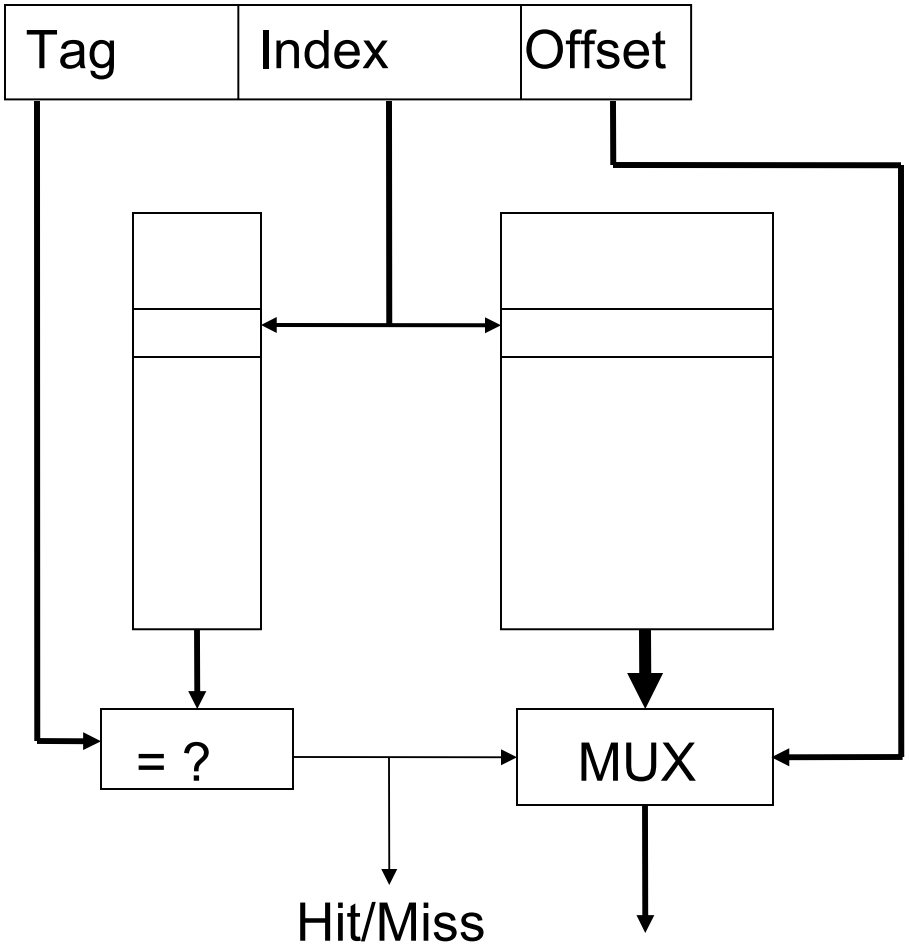
- Data una cache direct mapped di NB blocchi e un main memory di NM blocchi, nello stesso blocco della cache possono essere mappati NM div NB blocchi.
- Per sapere se il blocco presente in cache è quello effettivamente cercato, oltre a memorizzare il blocco, viene memorizzata una informazione supplementare, 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

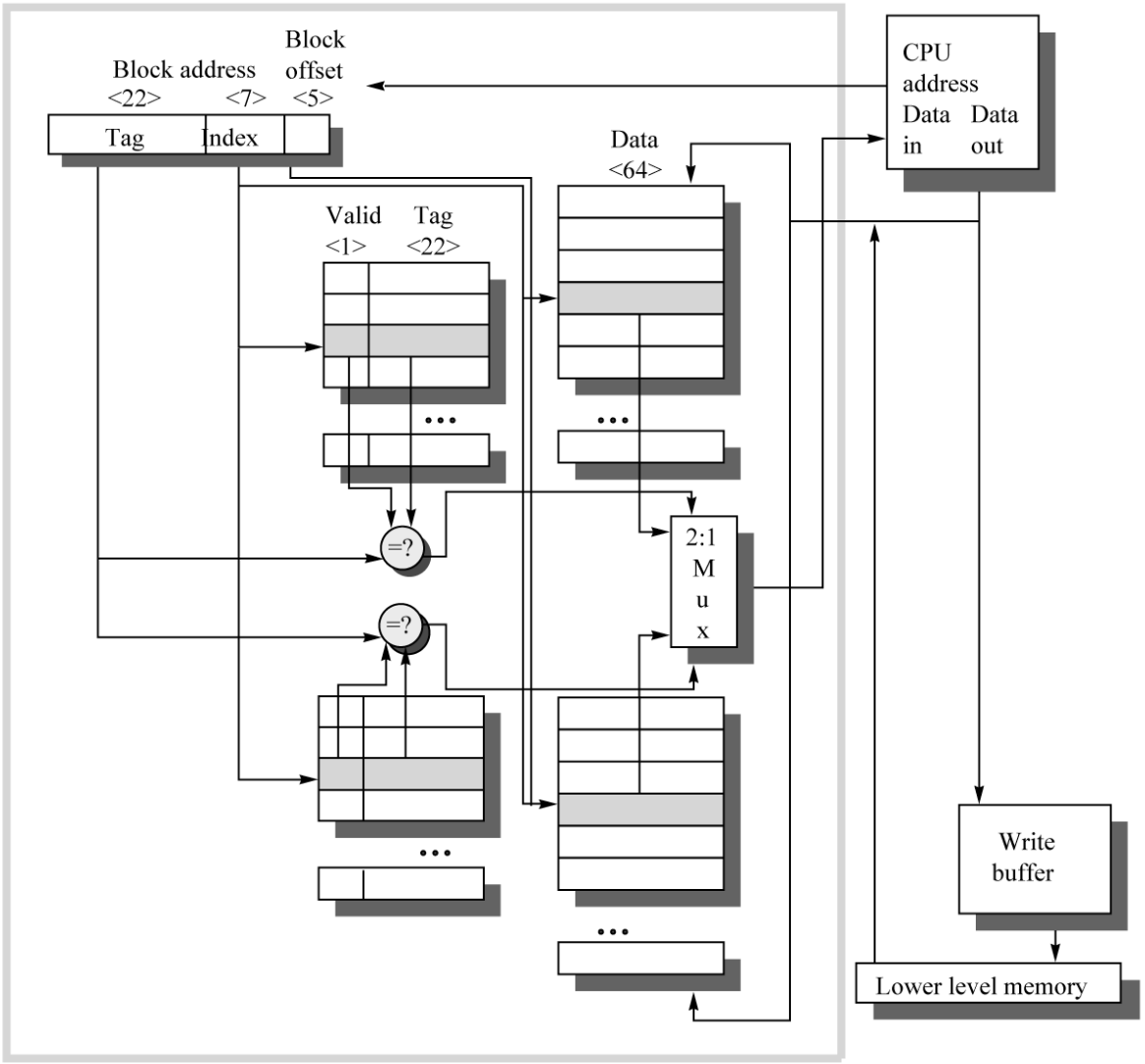
Q2: Come faccio a sapere se un blocco è presente?



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

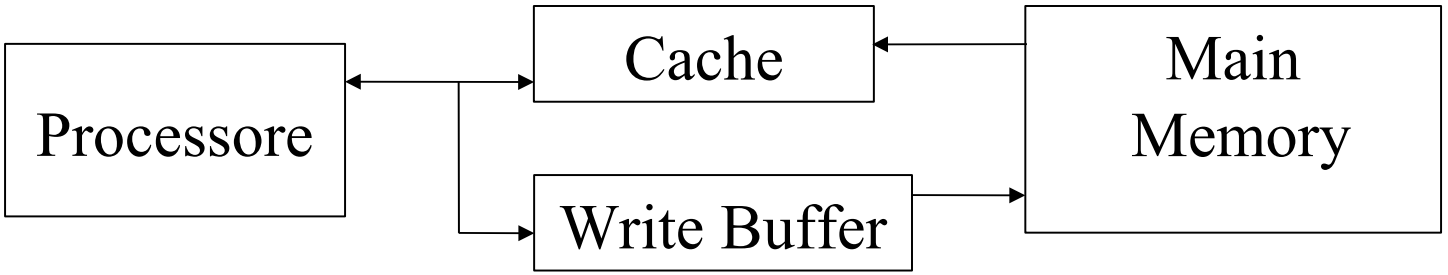
Associativity:	2-way		4-way		8-way	
Size	LRU	Random	LRU	Random	LRU	Random
16 KB	5.18%	5.69%	4.67%	5.29%	4.39%	4.96%
64 KB	1.88%	2.01%	1.54%	1.66%	1.39%	1.53%
256 KB	1.15%	1.17%	1.13%	1.13%	1.12%	1.12%

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 scritture (in main memory)
  - WB: più scritture sullo stesso blocco (in cache) comportano una sola scrittura in main memory

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

Write buffer



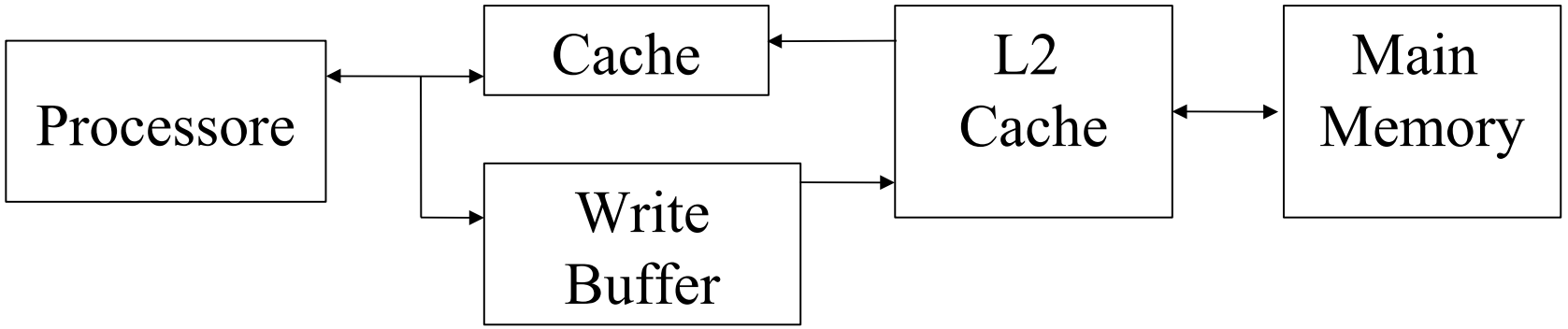
- A Write Buffer is needed between the Cache and Memory
  - –Processor: writes data into the cache and the write buffer
  - –Memory controller: write contents of the buffer to memory
- Write buffer is just a FIFO:
  - –Typical number of entries: 4
  - –Works fine if: Store frequency (w.r.t. time) << 1 / DRAM write cycle
- Memory system designer's nightmare:
  - –Store frequency (w.r.t. time) > 1 / DRAM write cycle
  - –Write buffer saturation

Write buffer

Solution for write buffer saturation:

–Use a write back cache

–Install a second level (L2) cache



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

Structural Hazard: Instruction and Data?

Size	Separate Instruction Instruction Cache	Data Cache	Unified Cache
1 KB	3.06%	24.61%	13.34%
2 KB	2.26%	20.57%	9.78%
4 KB	1.78%	15.94%	7.24%
8 KB	1.10%	10.19%	4.57%
16 KB	0.64%	6.47%	2.87%
32 KB	0.39%	4.82%	1.99%
64 KB	0.15%	3.77%	1.35%
128 KB	0.02%	2.88%	0.95%

Cache Performance

CPU time = (CPU execution clock cycles + Memory stall clock cycles) x clock cycle time

Memory stall clock cycles = (Reads x Read miss rate x Read miss penalty + Writes x Write miss rate x Write miss penalty)

Memory stall clock cycles = Memory accesses x Miss rate x Miss penalty

Cache Performance

CPUtime = IC x ($CPI_{execution}$ + Mem accesses per instruction x Miss rate x Miss penalty) x Clock cycle time

Misses per instruction = Memory accesses per instruction x Miss rate

CPUtime = IC x ($CPI_{execution}$ + Misses per instruction x Miss penalty) x Clock cycle time

Improving Cache Performance

Average memory-access time = Hit time +
Miss rate x Miss penalty (ns or clocks)

- Improve performance by:
  1. Reduce the miss rate,
  2. Reduce the miss penalty, or
  3. Reduce the time to hit in the cache