GESTIONE DELLE CACHE

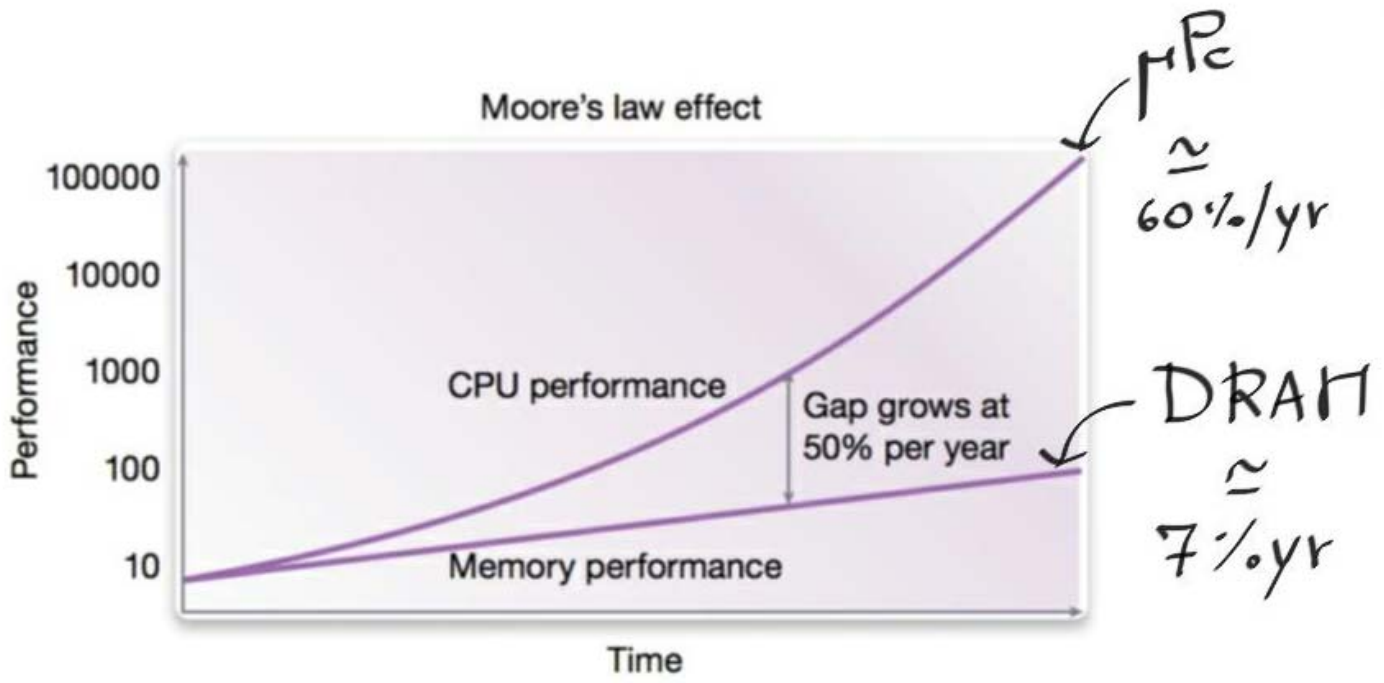
LUIGI COPPOLINO

Audience: corso di Architettura dei Sistemi a Microprocessore
Prerequisiti: concetti di base relativi alle Memorie (Spazio di Indirizzamento, indirizzo, unità di memorizzazione, ecc.)

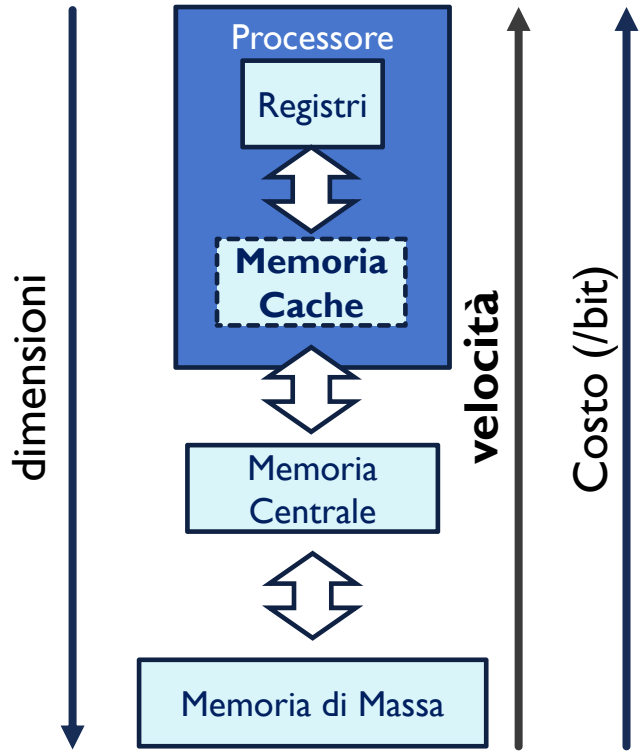
ROADMAP

- Introduzione alle cache
  - Principi di località
- Organizzazione e funzionamento di una cache
  - Politiche di Mapping
  - Politiche di Replacement
- Valutazione dei miglioramenti prestazionali
- Le cache nei processori reali: gerarchie di memoria
- Il caso Intel i7

RATIONALE


Moore's law effect
Performance
100000
10000
1000
100
10
CPU performance
Gap grows at
50% per year
Memory performance
Time
μPc
≃
60%/yr
DRAM
≃
7%/yr

MEMORIA CACHE



- Ha dimensioni ridotte (KB-MB)
- È veloce (SRAM)
- È invisibile al processore
  - Che continua a generare indirizzi in maniera trasparente
- Non può ospitare l'intero contenuto della RAM per cui sarà necessario
  - Definire quale porzione della RAM caricare
  - Come sostituire il contenuto della cache

Il tutto funziona se la maggior parte degli accessi alla memoria trova effettivamente i dati in cache

PRINCIPI DI LOCALITÀ

Dalle evidenze sperimentali:

PRINCIPIO DI LOCALITA' SPAZIALE: se al tempo T si verifica un accesso all'indirizzo A, è altamente probabile che l'indirizzo acceduto al tempo T+1 sia in un intorno di A

PRINCIPIO DI LOCALITA' TEMPORALE: se al tempo T si verifica un accesso all'indirizzo A, è altamente probabile che l'indirizzo A sia nuovamente acceduto nell'immediato futuro di T

Le evidenze suggeriscono che:

1) conviene lavorare su **blocchi di dati** anziché singole parole
2) conviene tenere in cache i dati usati di recente

INTERAZIONI PROCESSORE-CACHE

- Il processore, durante una **lettura** o una **scrittura**, genererà un indirizzo come farebbe in assenza di cache
- Il dato corrispondente all'indirizzo può essere presente in cache oppure no
  - Cache hit
  - Cache miss

CACHE HIT: IL DATO RICHIESTO È IN CACHE

- **Caso LETTURA**:
  - Il dato viene prelevato dalla cache e inviato al processore: nessuna interazione con la memoria principale
- **Caso SCRITTURA**:
  - Scrittura in cache => perdita di consistenza
  - Due possibili policy (Write policy):
    1. **Write-Through**: la scrittura viene effettuata sulla cache e contemporaneamente sulla memoria principale
       - Semplice, ma il tempo di scrittura è dettato dalla velocità della memoria principale
    2. **Write-Back**: la scrittura avviene in cache, ma il blocco modificato è invalidato (valid bit), quando il blocco sarà cacciato dalla cache, se non valido, sarà ricopiato sulla memoria principale
       - Necessario aggiungere un bit per ciascun blocco della cache, con il vantaggio che più scritture sullo stesso blocco generano un solo accesso alla memoria (quando sarà scaricato il blocco).

CACHE MISS: IL DATO RICHIESTO NON E' PRESENTE IN CACHE

- CASO LETTURA: dato trasferito in cache
- CASO SCRITTURA:
  - **Write-Through**: dato modificato nella Memoria Principale (eventualmente portato in cache)
  - **Write-Back**: dato trasferito in cache e poi modificato

DATO TRASFERITO IN CACHE:

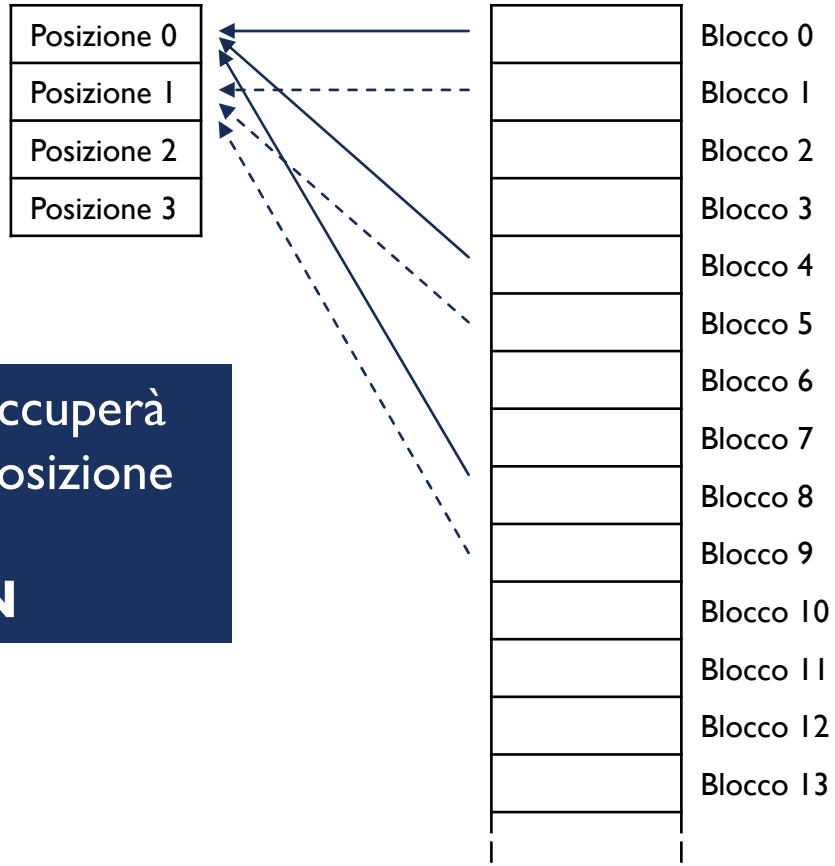
- Dove? => Politica di Mapping
- Se la cache è piena, quale blocco dovrà essere liberato per far posto al blocco entrante? => Politica di Sostituzione

POLITICA DI MAPPING

- Definisce la corrispondenza tra indirizzi nella memoria principale e indirizzi nella cache
- Tre politiche:
  - Mapping Diretto (Direct Mapping)
  - Mapping completamente Associativo (Full Associative)
  - Mapping Set-Associativo

DIRECT MAPPING

Cache di N=4 blocchi



Il blocco X occuperà in Cache la posizione

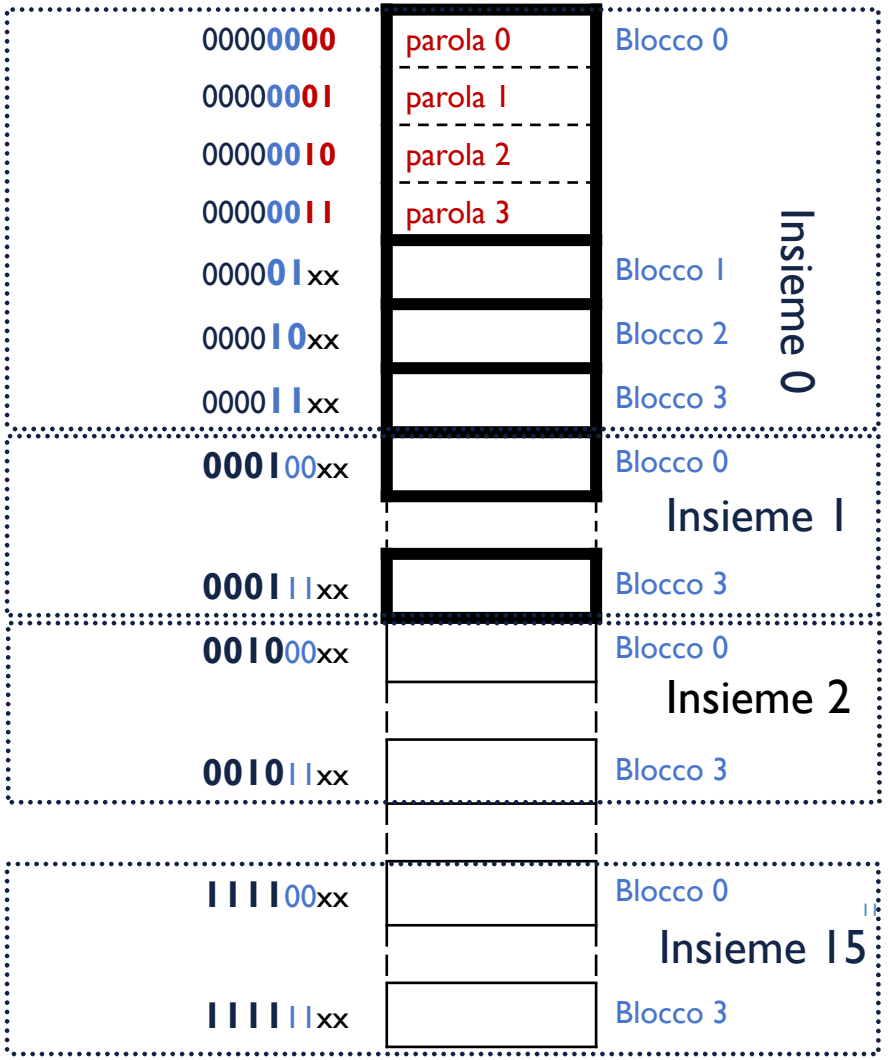
POS = X%N

DIRECT MAPPING: ACCESSO AL DATO
MEMORIA DA 256 (2^8) PAROLE
CACHE DA 4 (2^2) BLOCCHI DA 4 (2^2) PAROLE CIASCUNO

ETICHETTA (4 bit)	BLOCCO (2 bit)	SPIAZZAMENTO (2 bit)
1011	10	11

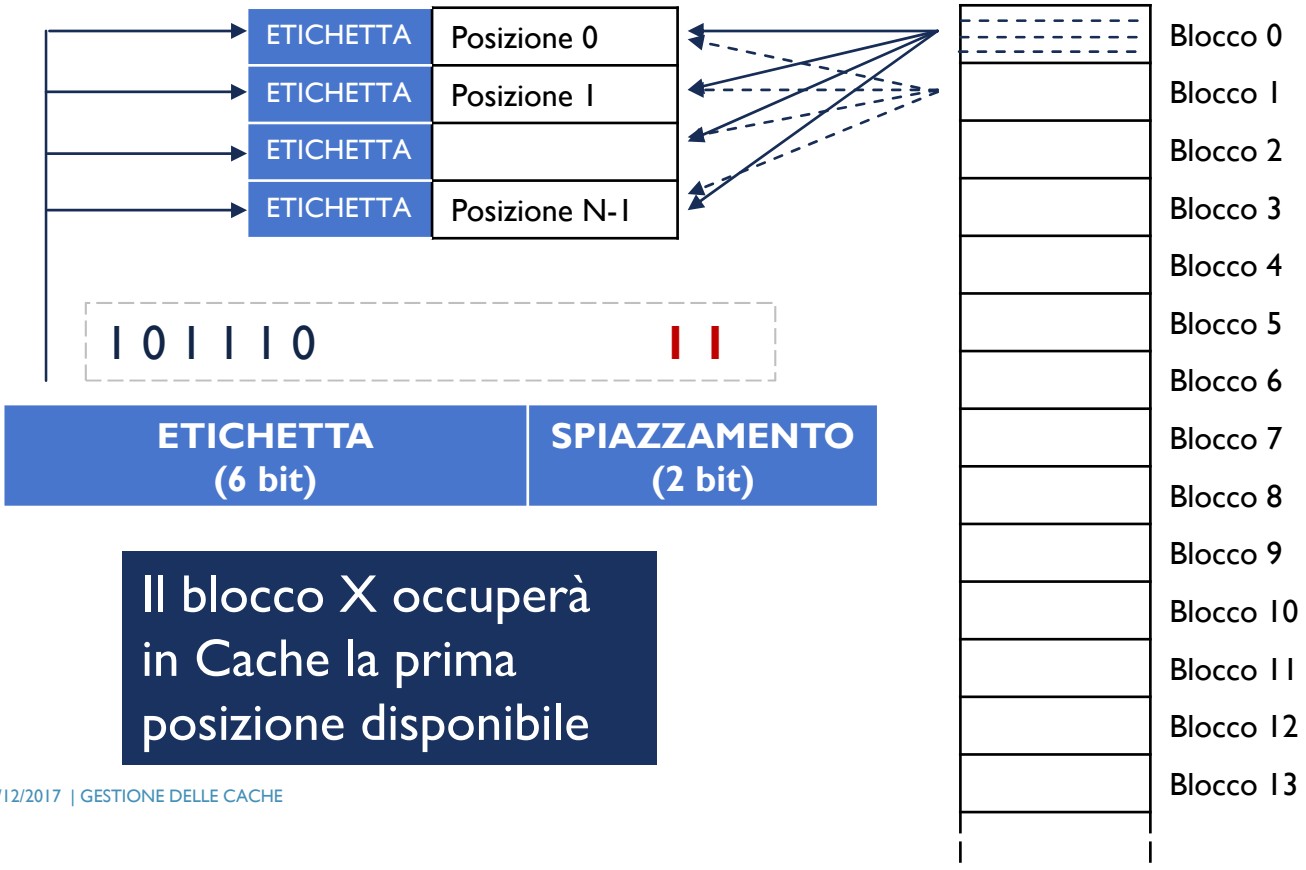
ETICHETTA	Posizione 0
ETICHETTA	Posizione 1
ETICHETTA	Posizione 2
ETICHETTA	Posizione 3 (N-1)

Cache di N=4 blocchi
da 4 parole



FULL ASSOCIATIVE: ANALISI DELL'INDIRIZZO

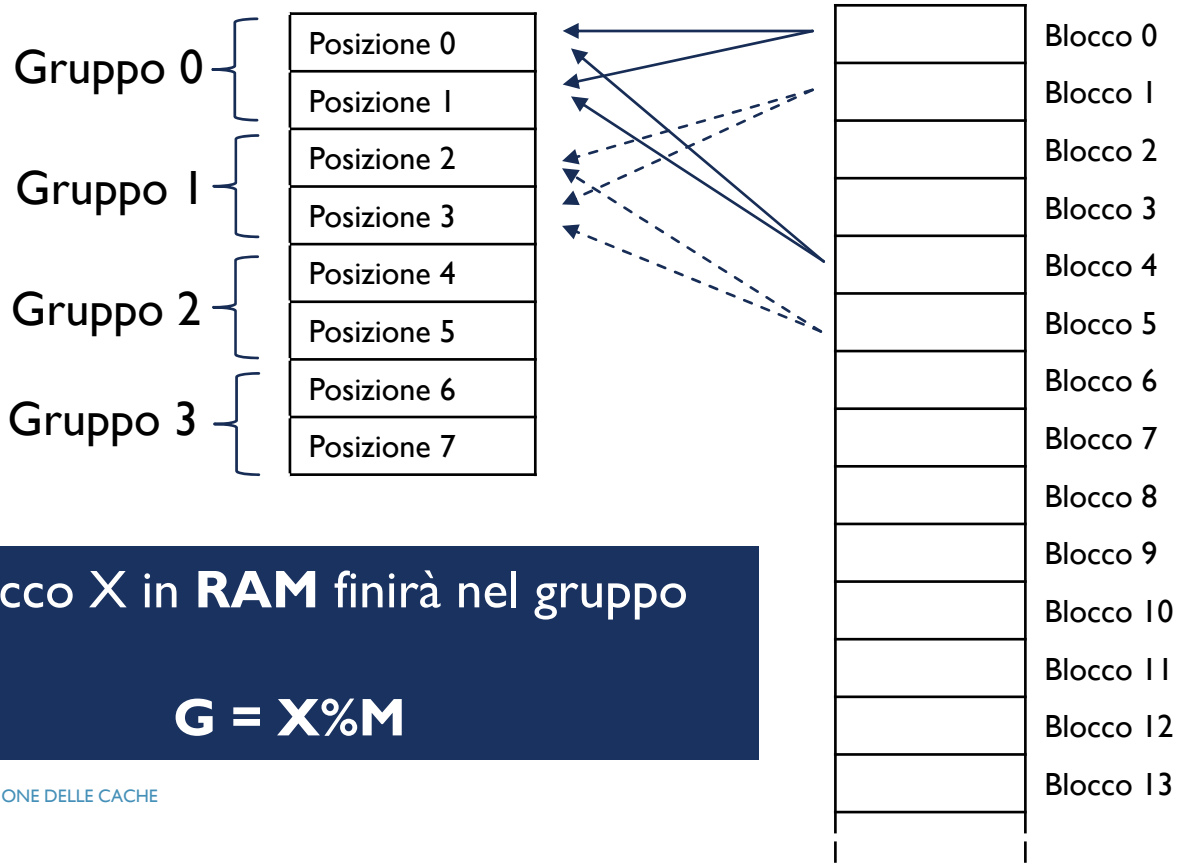
Cache di N=4 blocchi da 4 parole



Il blocco X occuperà in Cache la prima posizione disponibile

SET ASSOCIATIVA

Cache di N = 8 blocchi in M = 4 gruppi

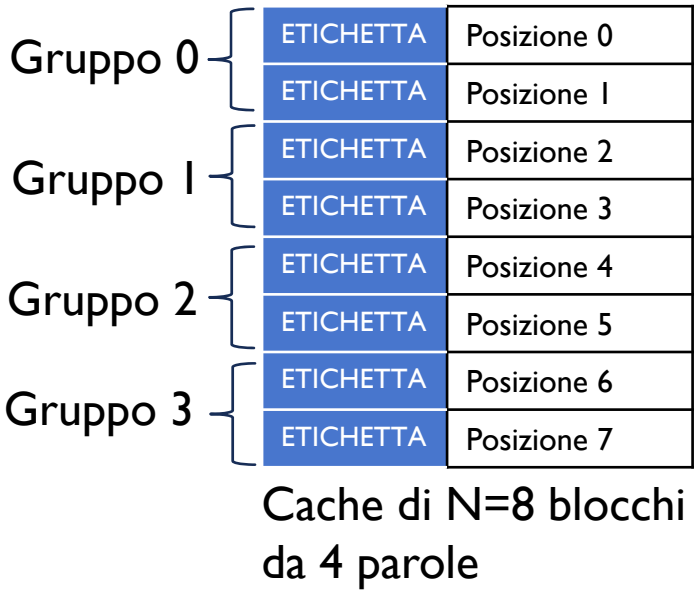


Il blocco X in **RAM** finirà nel gruppo

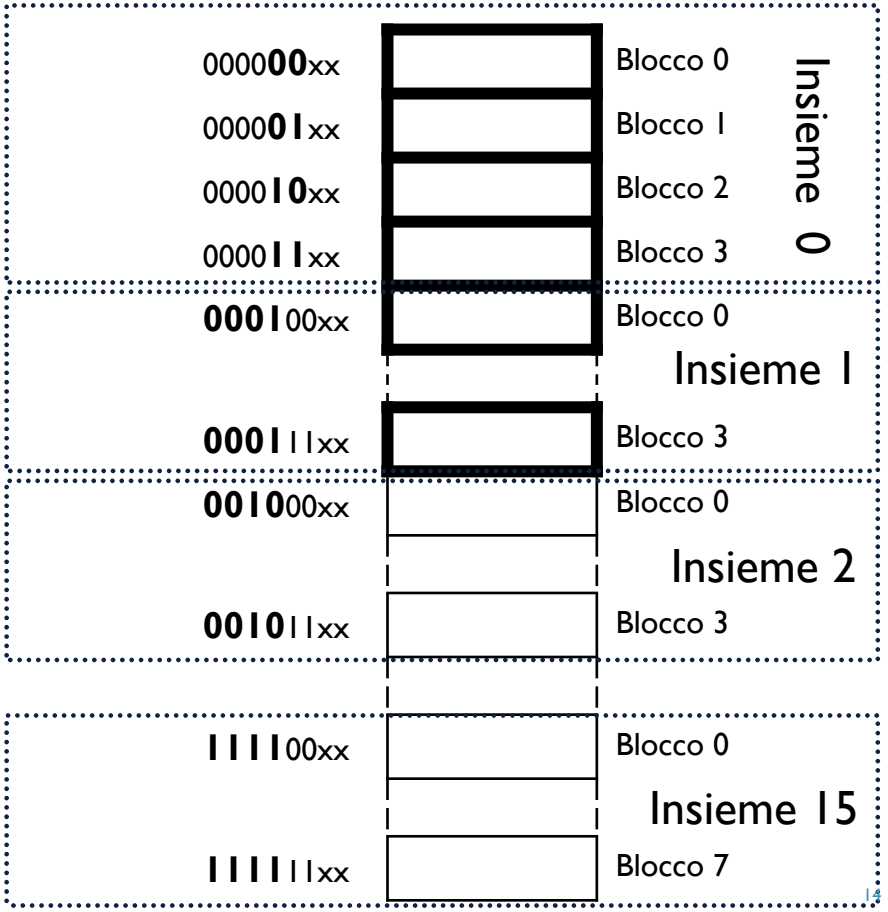
G = X%M

SET ASSOCIATIVA: ACCESSO AL DATO

ETICHETTA (4 bit)	GRUPPO (2 bit)	SPIAZZAMENTO (2 bit)
1011	10	11

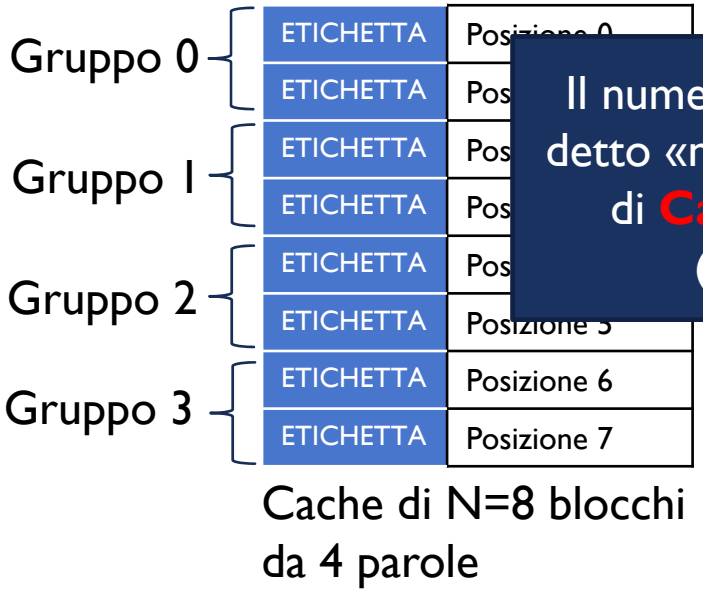


Cache di N=8 blocchi da 4 parole



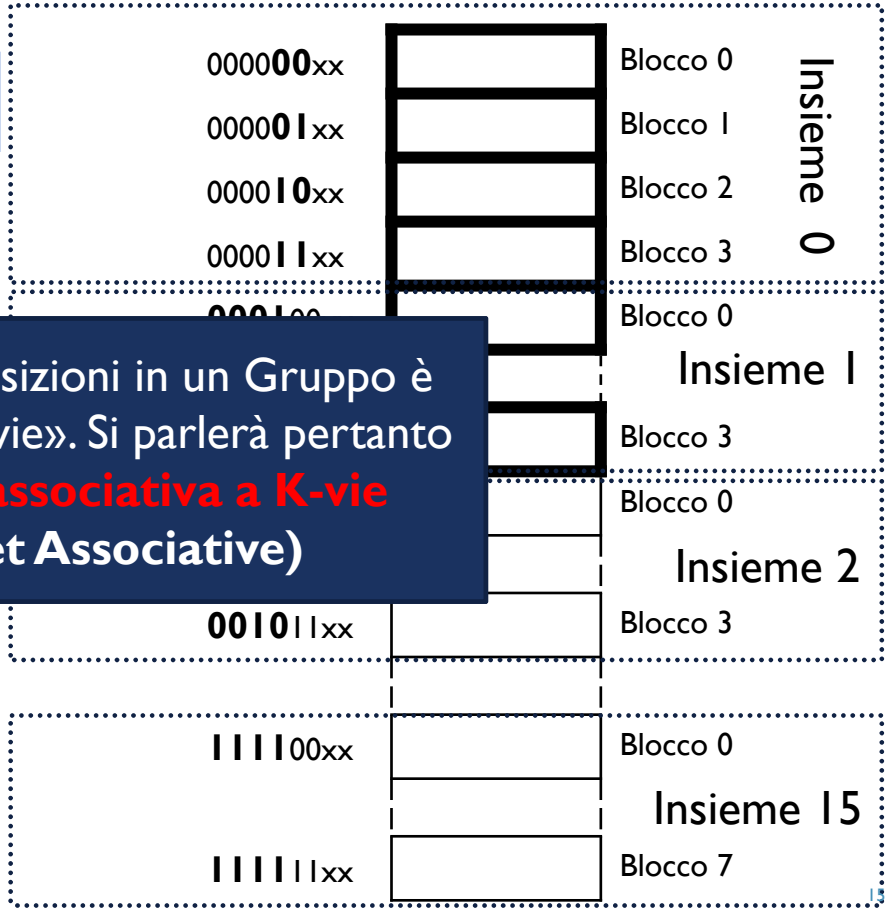
SET ASSOCIATIVA: ACCESSO AL DATO

ETICHETTA (4 bit)	GRUPPO (2 bit)	SPIAZZAMENTO (2 bit)
1011	10	11



Cache di N=8 blocchi da 4 parole

Il numero **K** di Posizioni in un Gruppo è detto «numero di vie». Si parlerà pertanto di **Cache set-associativa a K-vie (K-way Set Associative)**



POLITICA DI SOSTITUZIONE: QUALE BLOCCO LIBERARE?

- Nel caso Direct Mapping la soluzione è banale
- Nel caso di cache associative:
  - Idealmente: sostituire il blocco che **non sarà usato** nel prossimo futuro
    - Corollario Località Temporale: Il dato usato meno recentemente è quello che meno probabilmente sarà usato nel futuro
    - Policy LRU (Least Recently Used): ad ogni blocco associato un contatore azzerato ad ogni accesso al blocco
  - LRU efficace ma costoso
  - Random: si mostra essere molto efficace ed è sicuramente poco costoso

PERFORMANCE ANALYSIS

- **HIT RATE** (**h**): rapporto tra gli hit in cache e il totale degli accessi in memoria (ipotizziamo 95%)
- **MISS RATE (1-h):** rapporto tra i miss in cache e il totale degli accessi in memoria (5%)
- **C** = tempo di risposta della cache
- **R** = tempo di accesso alla RAM (inclusa una penalty) (ipotizziamo 10*C)
- **N** = Numero di accessi in memoria
- $\mathbf{T_{NO\_CACHE} \approx N*R}$ (tempo per l'accesso ai dati in assenza di cache)
- $\mathbf{T_{CACHE} = N*h*C+N*(1-h)*R}$ (tempo per l'accesso ai dati in presenza di cache)

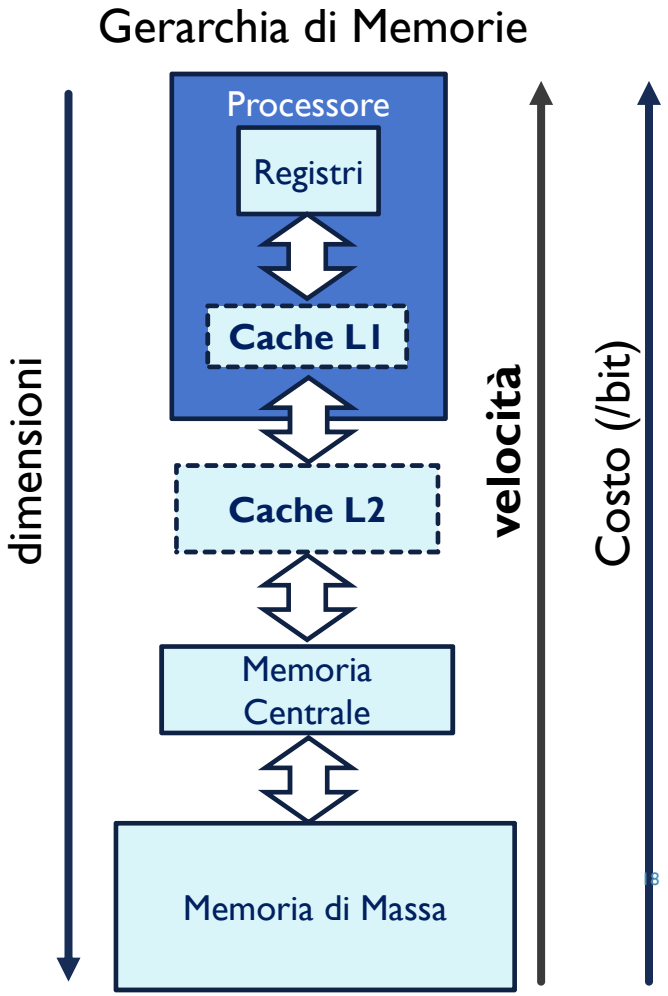
- $\text{SPEED\_UP} = \frac{T_{NO\_CACHE}}{T_{CACHE}} = \frac{N*R}{N*h*C+N*(1-h)*R} = \frac{10*C}{0,95*C+0,05*10*C} = \frac{10}{0,95+0,05*10} = 6,90$

LE CACHE NEI SISTEMI MODERNI

Livelli Multipli di Cache

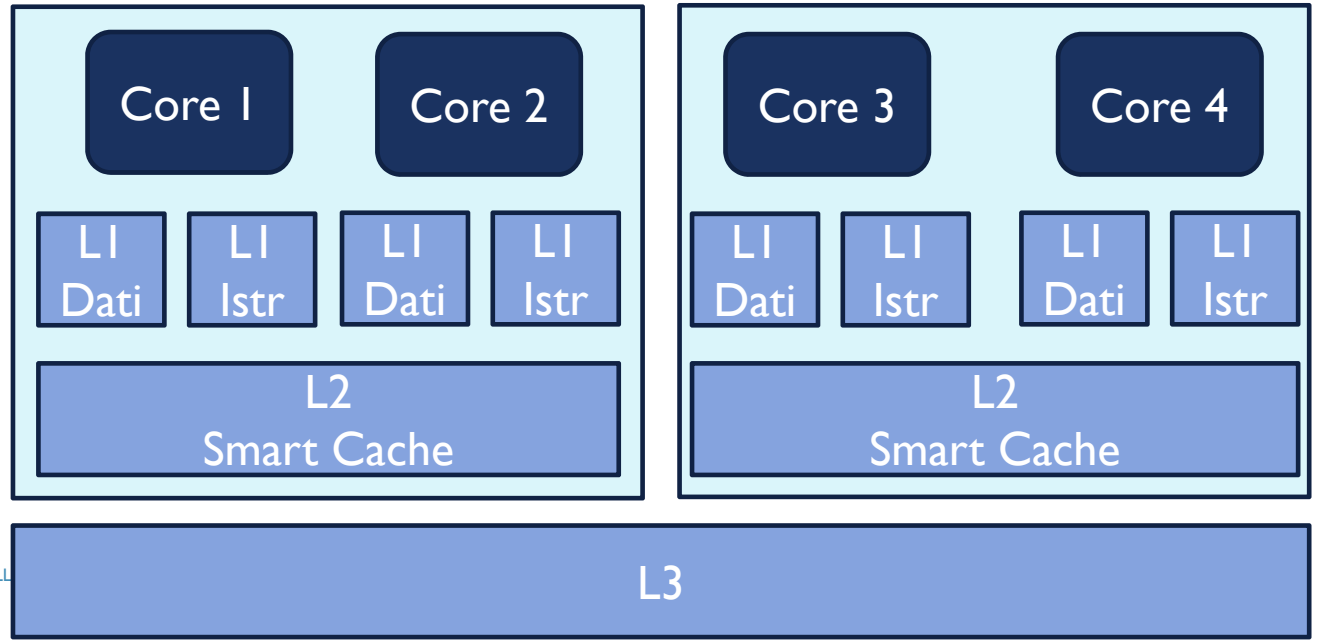
- la L1 serve gli HIT, la L2 serve i MISS.

MISS a due livelli = MISS RATE L1*MISS RATE L2



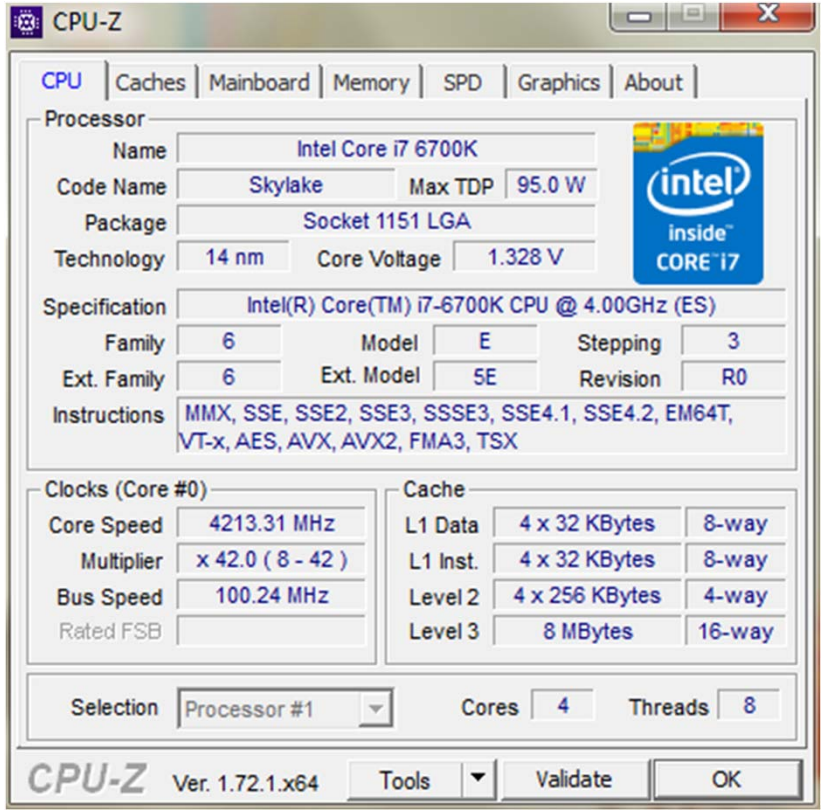
INTEL CACHE LEVELS

- Intel introduce Cache con 486Dx
- Dal Pentium II introdotta L2 Cache
  - Con il Pentium III la L2 passa sul Chip del processore
- Con il Pentium IV introdotta la L3
- Intel I7:



INTEL I7-6700 (SKYLAKE), 4.0 GHZ (TURBO BOOST), 14 NM

- Configurazione
  - L1 Data cache = 32 KB, 8-WAY.
  - L1 Instruction cache = 32 KB, 8-WAY.
  - L2 cache = 256 KB, 4-WAY
  - L3 cache = 8 MB,16-WAY
- Latenza
  - L1 Data Cache Latency = 4/5 cycles
  - L2 Cache Latency = 12 cycles
  - L3 Cache Latency = 38 cycles
  - RAM Latency = 42 cycles



RIFERIMENTI

- Computer Organization and Embedded Systems, 6th Ed., Carl Hamacher, Chap. 8
- Computer Architecture, A Quantitative Approach, John L. Hennessy and David A. Patterson, Chap. 2 and App. B

- https://www.cpuid.com/softwares/cpu-z.html
- http://www.7-cpu.com/cpu/Skylake.html

DOMANDE?

LUIGI.COPPOLINO@UNIPARTHENOPE.IT