**Memoria cache**

La memoria cache è una memoria in cui vengono memorizzate le parole che vengono utilizzate dai processi durante la loro esecuzione. Il suo utilizzo è dovuto al fatto che è una memoria molto veloce, il cui tempo di accesso è molto minore rispetto al tempo di accesso alla memoria centrale. Per velocizzare maggiormente l’esecuzione di un processo si sono introdotte due cache contemporaneamente, una solo per le istruzioni, una solo per i dati. La cache per svolgere al meglio i suoi compiti si basa su due principi fondamentali: **la località temporale** e la **località spaziale.** La prima riflette la considerazione che è molto probabile che si acceda ad indirizzi di memoria numericamente vicini a quello a cui si è appena accesso. La seconda si verifica quando si accede nuovamente a locazioni di memoria già utilizzate in precedenza.

Nella **cache a corrispondenza diretta** ciascun elemento della riga può memorizzare esattamente una linea di cache dalla memoria centrale.

|  |  |  |
| --- | --- | --- |
| Valid | Tag | Data |
|  |  |  |
|  |  |  |
|  |  |  |

Il bit **Valid** indica se il dato nell’elemento è valido oppure no. All’avvio del sistema tutti gli elementi della cache sono marcati come non validi. Il campo **Tag** è un valore univoco a 16 bit corrispondente alla linea di memoria da cui provengono i dati. Invece il capo **Data** contiene una copia del dato in memoria.

Nella cache a corrispondenza diretta una data parola in memoria può essere memorizzata in un’unica posizione della cache. Per ogni indirizzo di memoria esiste un solo posto all’interno della cache in cui cercare il dato. Per memorizzare e prelevare dati all’interno della cache l’indirizzo è diviso in quattro campi:

* Il campo **TAG** corrisponde ai bit Tag memorizzati in un elenco della cache
* Il campo **LINE** indica l’elemento della cache contenente i dati corrispondenti, se sono presenti
* Il campo **WORD** indica a quale parola si fa riferimento all’interno della linea
* Il campo **BYTE** non viene generalmente utilizzato, ma se richiede un solo byte, esso indica quale byte esso indica quale byte è richiesto all’interno della parola.

|  |  |  |  |
| --- | --- | --- | --- |
| TAG | LINE | WORD | BYTE |

Una cache con n possibili elementi per ciascun indirizzo è chiamata cache set-associativa a n vie. Questa cache è più complessa dell’altra. Infatti, anche se l’insieme di elementi della cache da esaminare viene calcolato a partire dall’indirizzo di memoria, è necessario controllare un insieme di n elementi per vedere se la linea richiesta è presente nella cache. Questa verifica deve essere effettuata rapidamente. Per inserire una nuova riga di memoria si utilizzano diverse strategie:

* LRU
* Write-through
* Write-allocation

L’algoritmo LRU mantiene un ordinamento temporale di ciascun insieme di elementi ai quali si può far riferimento in base a una certa locazione di memoria. Quando si accede a una delle linee presenti nella cache l’algoritmo aggiorna la lista marcando l’elemento utilizzato più recentemente. Quando giunge il momento di sostituire un elemento, l’algoritmo scarta quello che si trova alla fine della lista, cioè quello al quale è stato effettuato un accesso meno recente.

L’algoritmo write-trhough serve per aggiornare immediatamente l’elemento della cache. Questo approccio è generalmente più semplice e il più affidabile, dato che il contenuto della memoria rimane sempre aggiornato. Ciò è utile se si verifica un errore e si vuole recuperare lo stato della memoria. In questo algoritmo, quando il processore scrive una parola, e questo è presente nella cache, ovviamente questo deve aggiornare la parola o scaricare l’elemento dalla cache. In quasi tutte le partizioni si sceglie di aggiornare la cache.

Nell’algoritmo write-allocation consiste nello scrivere nella cache il dato da aggiornare quando si verifica un fallimento in scrittura. Al contrario, quando si impiega la scrittura diretta, si tende generalmente a non allocare un elemento della cache a ogni operazione di scrittura dato che ciò si complicherebbe un’organizzazione altrimenti semplice. L’allocazione in scrittura risulta vincente solo nel caso si ripetute scritture della stessa parola o di diverse parole facenti parte di una stessa linea di cache.