



Memoria Caché

Sistemas Informáticos I.E.S. Virgen
de la Paloma
Carlos Barahona



Introducción (I)

- Jerarquía de memoria
 - Registros
 - Cache L1
 - Cache L2
 - Cache L3
 - Memoria principal
 - Cache de disco
 - Memoria secundaria

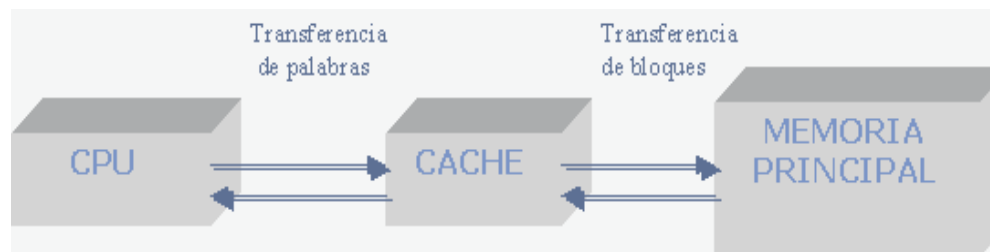


Introducción (II)

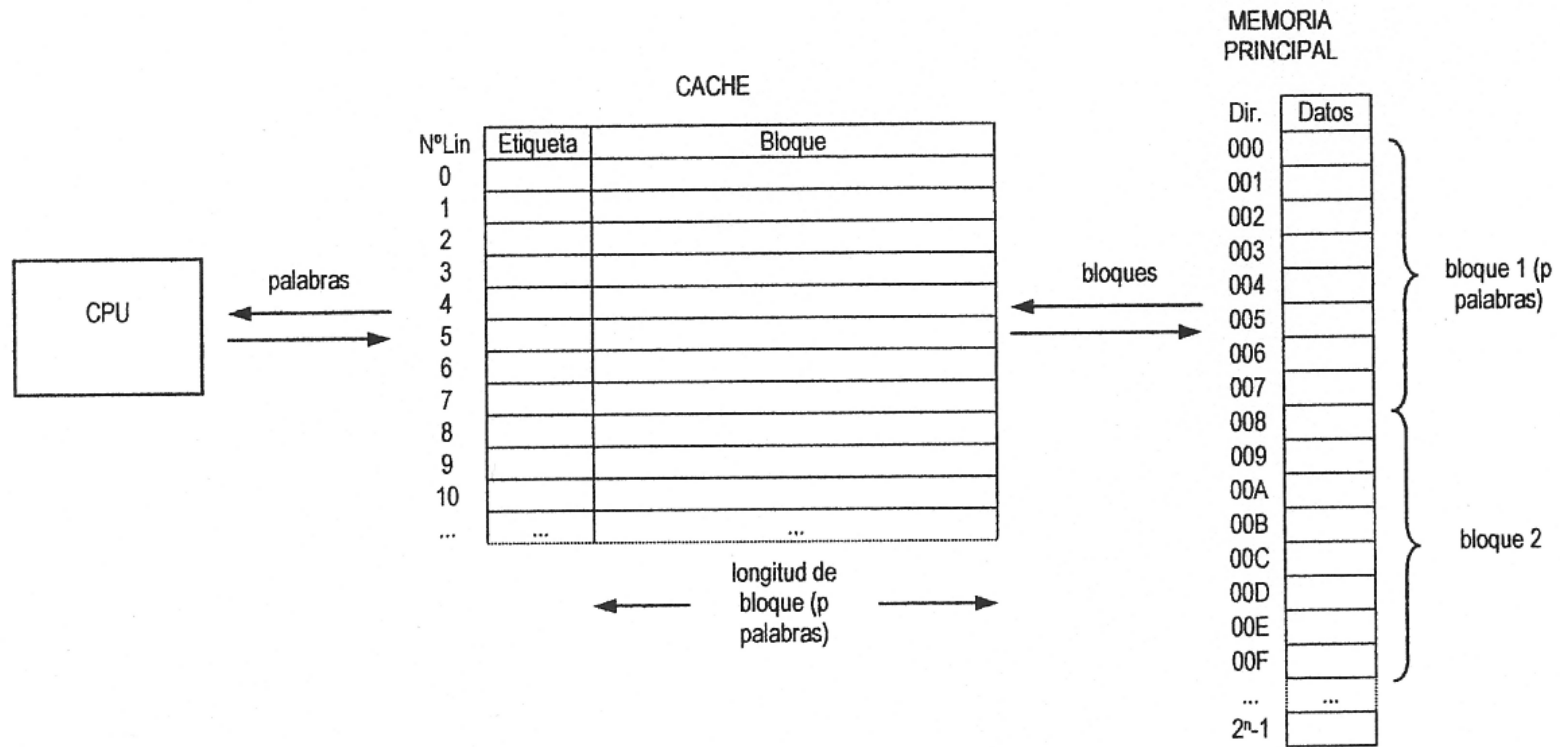
- El término “Caché” proviene de un vocablo francés que significa “oculto”.
- La razón fundamental de la aparición de la memoria caché es la alta latencia que sufrimos al acceder a la información alojada en memoria.
- Latencia: es el retraso entre el momento en que se inicia algo, y el momento en que uno de sus efectos comienza

Introducción (III)

- El objetivo de la memoria caché es reducir el tiempo de latencia en los accesos a memoria.
- La memoria caché es una memoria intermedia de pequeño tamaño y muy rápido acceso, gestionada por hardware, que se encuentra entre la CPU y la memoria principal y que se aprovecha del concepto de “localidad”.



Introducción (IV)





Introducción (V)

■ Localidad

□ Puede ser de dos tipos:

- Temporal: tendencia a usar en breve los mismos datos que se acaban de utilizar.
- Espacial: tendencia a usar datos ubicados muy cerca de los que se acaba de acceder.



Introducción (VI)

- Operación de la caché
 - La CPU solicita contenidos de la localización de memoria.
 - Comprueba la caché para estos datos.
 - Si está, la obtiene de la cache (rápidamente).
 - Si no está, lee el bloque requerido a partir de la memoria principal hasta la caché.
 - Después, de la caché los entrega a la CPU.
 - La caché incluye etiquetas para identificar qué bloque de la memoria principal está en cada ranura de la cache.



Diseño (I)

- Memoria principal:

- Según la figura anterior, la memoria principal dispone de 2^n palabras direccionables, cada una con una dirección de n bits. La memoria se la considera dividida en bloques iguales compuestos por p palabras (número de bloques = m)

- Memoria caché:

- Dispone de l líneas de p palabras cada una, cumpliéndose que $l \ll m$, por lo que en la caché solo cabrá un pequeño subconjunto de la memoria principal.
- De este modo, no es posible que cada línea de la caché se asigne a un bloque de manera fija, por ello, y para conocer que bloque se encuentra en la línea es necesario etiquetar cada línea con algún valor que permita identificar al bloque.



Diseño (II)

- Conceptos a tener en cuenta a la hora de diseñar una caché.
 - Tamaño de la caché.
 - Función de correspondencia.
 - Algoritmo de sustitución.
 - Política de escritura.



Diseño (III)

- El tamaño de la caché viene limitado por el coste de este tipo de memorias, por lo que suelen ser de un tamaño reducido.
- Existen estudios que indican que un aumento desmesurado del tamaño de la caché redundaría en una disminución de su eficacia.



Función de Correspondencia (I)

- Para poder determinar a qué bloque de memoria principal corresponde una línea de memoria caché se utiliza una función de correspondencia.
- Dependiendo de esta función de correspondencia hablamos de tres tipos de correspondencia:
 - Directa
 - Asociativa
 - Asociativa por conjuntos

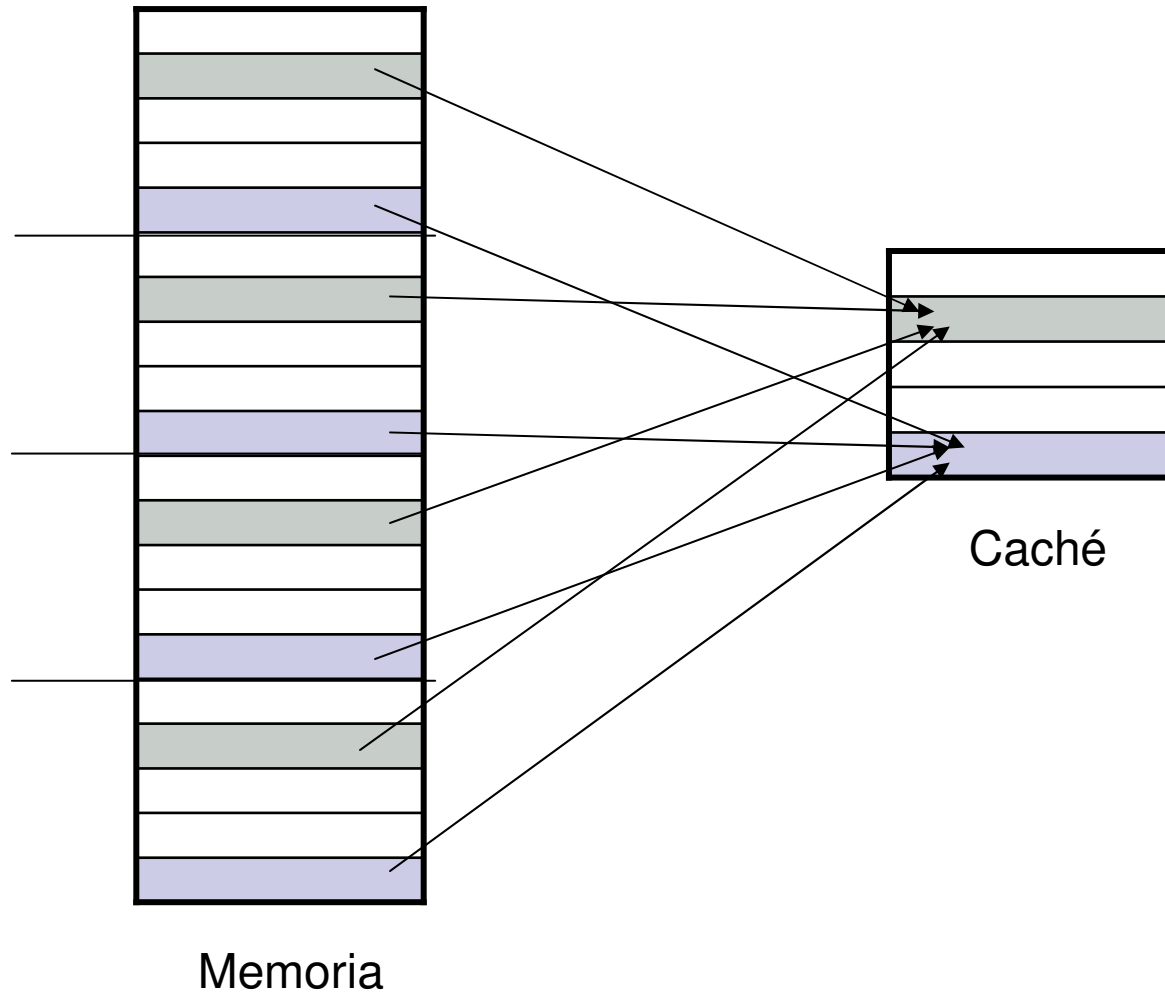


Función de Correspondencia (II)

■ Directa

- Cada bloque de la memoria tiene una línea de la caché asignada, de manera que cuando ha de cargarse lo hace en esta línea y no en otra.
- Ventajas:
 - La localización de un bloque en la memoria es muy rápida.
- Inconvenientes:
 - Si en un momento dado un programa hace referencia a bloques de memoria cuya línea de referencia de la caché es la misma se producirán conflictos de ubicación.

Función de Correspondencia (III)





Función de Correspondencia (IV)

■ Asociativa

- Permite que cada bloque de memoria se cargue en cualquier línea de la caché.

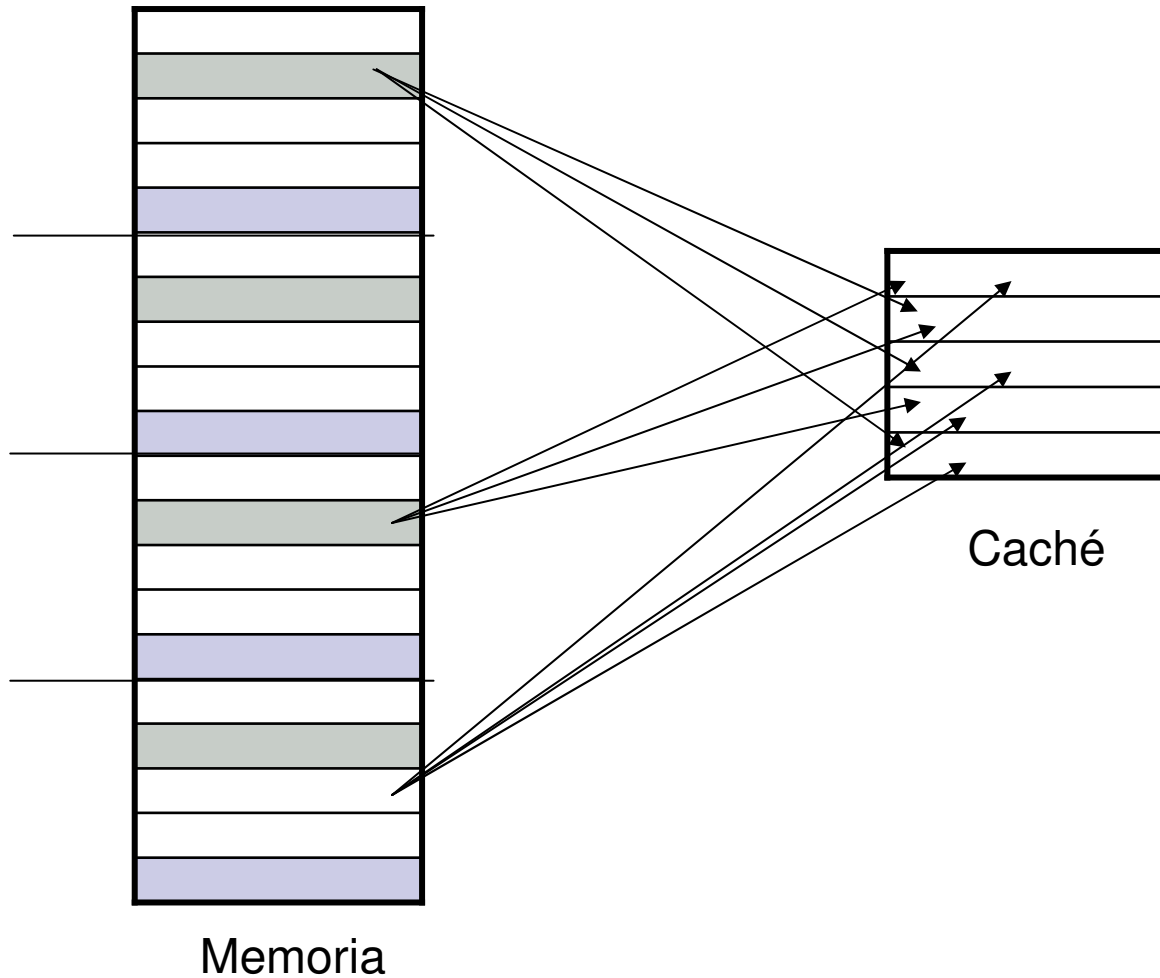
■ Ventajas

- Se evitan conflictos de ubicación (a priori)

■ Desventajas

- Circuitería compleja.
- Búsqueda mucho más lenta.

Función de Correspondencia (V)



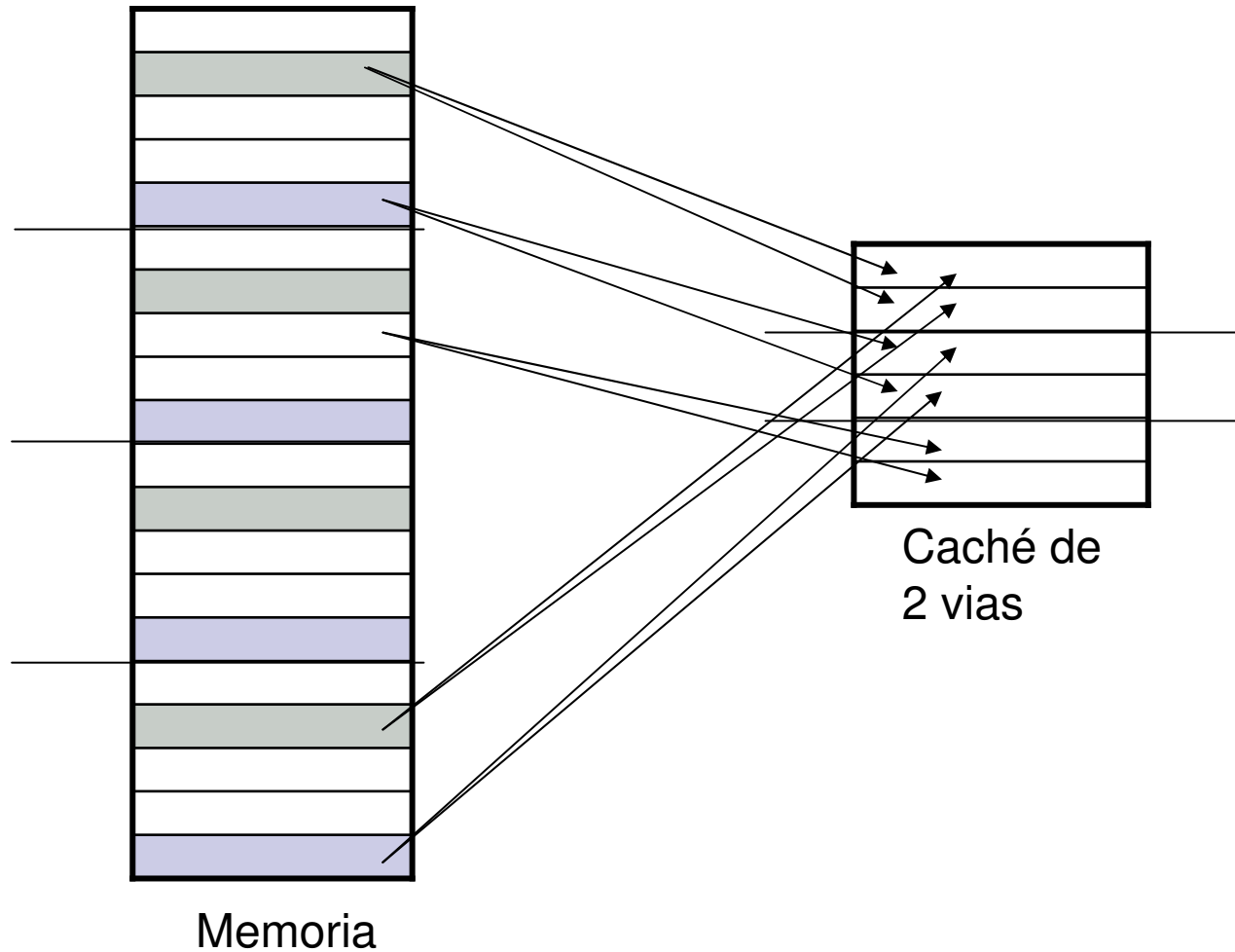


Función de Correspondencia (VI)

■ Asociativa por conjuntos

- Es una mezcla de las dos anteriores.
- La caché se divide en c conjuntos cada uno con k líneas de modo que el número total de líneas es de $l = c \cdot k$.
- Cada bloque de la memoria tienen un conjunto asignado, pero dentro del conjunto puede ocupar cualquier línea.
- De este modo el acceso al conjunto es directo, mientras que dentro de cada conjunto cada bloque es buscado siguiendo técnicas asociativas.
- Al número de líneas por conjunto se le denomina vía, por ejemplo, una caché asociativa por conjuntos de ocho vías tendrá c conjuntos de ocho líneas cada uno.

Función de Correspondencia (VII)





Algoritmos de Reemplazo (I)

- Para el esquema directo no hay elección, ya que cada bloque de memoria solo puede estar en un sitio.
- Para los otros esquemas:
 - LRU (Less Recently Used)
 - FIFO (First In First Out)
 - LFU (Less Frequently Used)
 - Random



Algoritmos de Reemplazo (II)

■ LRU

- El algoritmo LRU toma su nombre de Less Recently Used (Menos recientemente usada) y parte de la suposición de que los bloques más usados seguirán usándose en un futuro cercano.
- Esta suposición implica que los bloques menos usados tienen menos probabilidad de usarse en un futuro cercano.
- El algoritmo se encarga de desalojar de la memoria caché aquel bloque que lleve más tiempo sin usarse.
- Para poder utilizar este algoritmo la caché ha de almacenar para cada bloque una referencia de tiempo indicando su último uso.



Algoritmos de Reemplazo (III)

■ FIFO

- El algoritmo FIFO toma su nombre de First In First Out (El primero en entrar es el primero en salir) .
- El algoritmo crea una lista de la secuencia de la entrada de los bloques a la memoria caché, desalojando la entrada más antigua de esta lista cuando haya que realizar un reemplazo.
- Cuidado, no se reemplaza aquella entrada cuyo uso sea el más antiguo, eso sería LRU, si no aquella que lleva mas tiempo en la caché.



Algoritmos de Reemplazo (IV)

■ LFU

- El algoritmo LFU toma su nombre de Less Frequently Used (Menos frecuentemente usada) .
- Este algoritmo sustituye aquella entrada que haya experimentado menos referencias.
- Para utilizar este algoritmo la caché ha de guardar un contador por cada bloque que indique las veces que ha sido referenciado, sustituyendo la entrada cuyo contador es el más bajo.



Algoritmos de Reemplazo (V)

■ Random

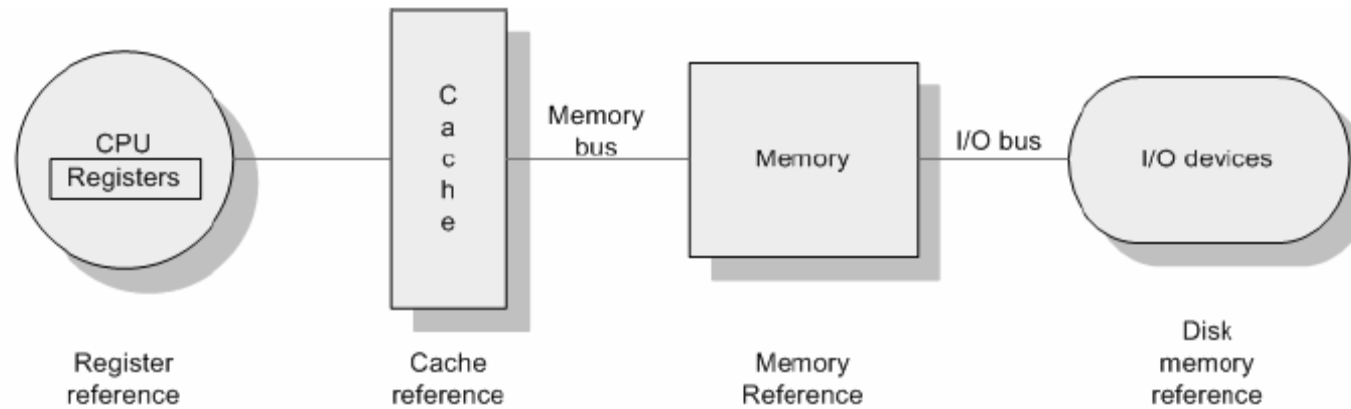
- El algoritmo Random toma su nombre de la función aleatoria que utiliza para realizar la selección de la entrada a sustituir.
- Diferentes estudios han mostrado que la sustitución aleatoria proporciona un rendimiento ligeramente menor a cualquiera de los algoritmos de reemplazo vistos con anterioridad.



Políticas de escritura (I)

- Antes de que pueda ser reemplazado un bloque de la caché es necesario comprobar si ha sido alterado en la caché y no en la memoria principal.
- Si la memoria principal se encuentra actualizada, el bloque puede ser sobrescrito.
- En caso contrario habrá que actualizar la memoria principal antes de sobrescribir el bloque.

Políticas de escritura (II)



Problemas de diseño: ¿qué lee el dispositivo de E/S si se sobrescribe una entrada de caché modificada que no ha sido volcada a memoria?



Políticas de escritura (III)

- ¿Cuándo escribir (de la caché a la memoria principal)?
- Existen dos técnicas:
 - Escritura inmediata
 - Escritura demorada



Políticas de escritura (IV)

■ Escritura inmediata

- Todas las operaciones de escritura se hacen tanto en la caché como en la memoria principal inmediatamente.
- Así se asegura que el contenido de la memoria principal sea siempre válido.
- Desventaja: se genera un tráfico de sustancial a la memoria principal que puede disminuir el rendimiento.
- Estudios señalan que el porcentaje de referencias a memoria para escritura es del orden del 15%.



Políticas de escritura (V)

■ Escritura demorada

- Cada bloque de la caché posee un bit de actualización que se inicializa en '0' cuando se carga un bloque nuevo en la caché.
- Cada vez que se escriba en el bloque el bit de actualización se pone en '1'.
- Cuando se desee reemplazar el bloque, el bloque se copia a la memoria principal sólo si el bit de actualización es '1'.



Caché Multinivel (I)

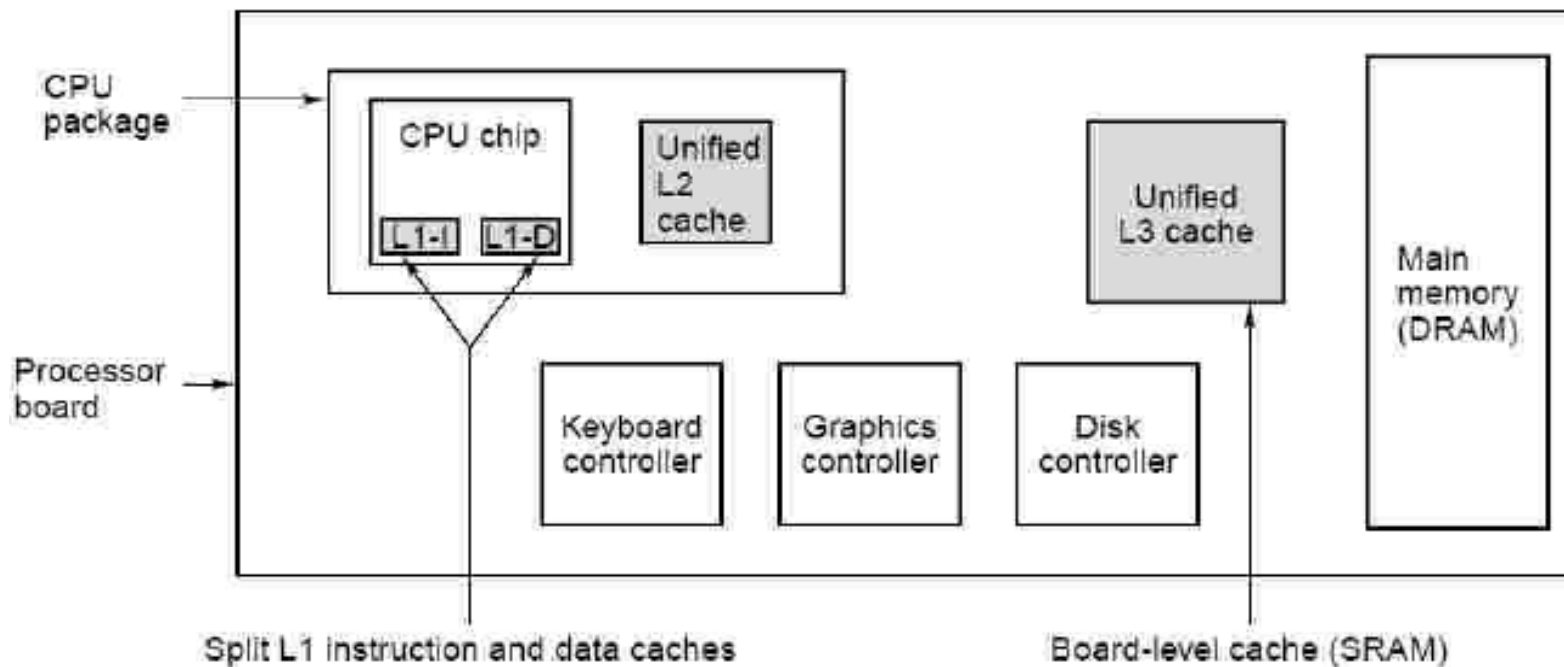
- Inicialmente, se usaba sólo una caché externa (off-chip) a la CPU.
 - Si había un fallo de caché se accedía directo a memoria
- Luego se desarrollaron cachés on-chip (mas rápidas).
- Idea: Colocar una caché más grande detrás de la caché, antes de la memoria principal
- Actualmente se tienen sistemas con cachés on-chip y off-chip.



Caché Multinivel (II)

- Además, existe una clasificación de cachés unificadas y otras partidas:
 - Las unificadas tienen instrucciones y datos.
 - Las partidas tienen una caché dedicada a instrucciones y otra dedicada a datos.
 - Las cachés 'partidas' tiene la ventaja de la paralelización ya que mientras se lee una instrucción se puede estar leyendo un dato.

Caché Multinivel (III)



Estructura actual de caché multinivel