Caché

* Justifique el uso de dos niveles de caché
* Caché: mencione algoritmos de reemplazo y políticas de escritura.
* Describa las funciones de correspondencia entre memoria principal y caché. Analice las políticas de escritura desde el punto de vista de la coherencia de datos.
* Mencionar los tipos de correspondencia de la memoria caché. Describir las políticas de escritura (en acierto y en fallo)
* ¿Por qué funciona una jerarquía de memoria? Describa las políticas de ubicación y de reemplazo de bloques en memoria caché.
* Describa las técnicas de ubicación de bloques y las politicas de escritura en Cache.
* Describa las funciones que se utilizan en la política de ubicación de bloques en memoria caché. Analice las políticas de escritura de datos desde el punto de vista de la coherencia de los mismos en la jerarquía.
* Describa las técnicas de reemplazo de bloque, correspondencia y políticas de escritura en memoria cache.

# Dos niveles de caché

La caché de dos niveles consta de:

* la caché interna, el nivel 1 (L1)
* la caché externa, el nivel 2 (L2)

Si no hay caché L2 y el procesador hace una petición de acceso a una posición de memoria que no está en la caché L1, el procesador debe acceder a la DRAM o la ROM a través del bus. Debido a la lentitud usual del bus y a los tiempos de acceso de las memorias, se obtienen bajas prestaciones.Pero si se usa una caché L2 SRAM, entonces con frecuencia la información que falta puede recuperarse fácilmente. Si la SRAM es rápida, los datos pueden accederse con cero estados de espera, el tipo de transferencia de bus más rápido.

Actualmente se estila separar la cache en dos: una dedicada a instrucciones y otra a datos. La ventaja de caché unificada es tener una tasa de aciertos mayor que una partida, ya que nivela automáticamente la carga entre captación de instrucciones y de datos. Además, solo se necesita diseñar e implementar una caché.

La ventaja de caché partida es que elimina la competición por la caché entre el procesador de instrucciones y la unidad de ejecución, importante en diseños que cuentan con segmentación de cauce de instrucciones (pipelining)

En resumen: el aumento de densidad de integración permitió tener una caché en el mismo chip del procesador. Esto reduce la actividad del bus externo del procesador y por lo tanto reduce los tiempos de ejecución, e incrementa las prestaciones globales del sistema. Actualmente se incluyen tanto cache on-chip como internos. La estructura más sencilla se denomina caché de dos niveles. En general, el uso de caché multinivel mejora las prestaciones, pero aumenta la complejidad del diseño.

# Algoritmos de reemplazo

Algoritmos de reemplazo / políticas de reemplazo / algoritmos de sustitución:

Cuando se introduce un nuevo bloque en la caché, debe sustituirse uno de los bloques existentes.

Para correspondencia directa, solo hay una posible línea para cada bloque particular, por lo que no hay elección y se sustituye por el que ocupa el lugar del nuevo.

Para las técnicas asociativas se requieren algoritmos de sustitución que deben implementarse en hardware para conseguir alta velocidad. Los más comunes:

* LRU: Sustituye el bloque que se ha mantenido en la cache por mas tiempo sin haber sido referenciado (debería dar la mejor tasa de aciertos). Requiere controles de acceso (Cada línea tiene un flag que se pone en 1 al ser referenciada)
* FIFO: El que ha estado más tiempo en la caché. Requiere controles de tiempo (se puede implementar con una técnica de round robin)
* LFU: Sustituye aquel bloque del conjunto que ha experimentado menos referencias. Requiere controles de uso (se implementan contadores a cada línea)
* Aleatoria: Agarra una línea al azar entre las posibles candidatas.

# Políticas de escritura

Antes de que pueda ser reemplazado un bloque que está en una línea de caché, es necesario comprobar si ha sido alterado en caché pero no en memoria principal.

Si no se modificó, puede escribirse sobre la línea de caché. Si se modificó, significa que se ha realizado al menos una operación de escritura sobre una palabra de la línea correspondiente de la caché. Hay dos cosas que hay que tener en cuenta:

* Más de un dispositivo puede tener acceso a la memoria principal
  + Si un módulo de E/S puede escribir/leer directamente en/de memoria, si una palabra ha sido modificada solo en la caché, la correspondiente palabra de memoria no es válida. Y si el dispositivo ha alterado la memoria principal, entonces la palabra de caché no es válida.
* Varios procesadores se conectan al mismo bus y cada uno de ellos tiene su propia caché local
  + Si se modifica una palabra en una de la caché, podría presumiblemente invalidar una palabra de otras cache

Las técnicas en acierto:

* Escritura inmediata (write-through): Es la más sencilla, todas las operaciones de escritura se hacen tanto en cache como en memoria principal, asegurando que el contenido de la memoria principal siempre es válido. Se actualizan simultáneamente la posición de la caché y de la memoria principal. Desventaja: con múltiples CPU, se genera un tráfico sustancial a memoria que puede originar un cuello de botella y retrasa la escritura. Se debe observar el tráfico a memoria principal para mantener actualizada la caché local
* Post-escritura (write-back): minimiza las escrituras en memoria, ya que las actualizaciones se hacen solo en la caché. Cuando tiene lugar una actualización, se activa un bit ACTUALIZAR (bit de “sucio”) asociado a la línea. Después, cuando el bloque es sustituido, es (post-)escrito en memoria principal si y sólo si el bit ACTUALIZAR está activo. Desventaja: A veces, porciones de memoria principal no son válidas y los accesos por parte de los módulos de E/S sólo podrían hacerse a través de la caché, complicando la circuitería y generando un potencial cuello de botella (la memoria principal puede contener información errónea en algún momento).

Las técnicas en fallo:

* Write allocate: La información se lleva de la memoria principal a la caché. Se sobreescribe en la caché
  + Habitual con write-back
* No-write allocate: El bloque no se lleva a la memoria caché. Se escribe directamente en la memoria principal
  + Habitual con write-through

# Jerarquía de memoria

# 

# Políticas de ubicación

Como hay menos líneas de caché que bloques de memoria principal, se necesita un algoritmo que haga corresponder bloques de memoria principal a líneas de caché.

También se requiere algún medio para determinar qué bloque de memoria principal ocupa actualmente una línea dada de caché. La elección de la función de correspondencia determina cómo se organiza la caché. Pueden usarse tres técnicas:

* Correspondencia directa: Hace corresponder cada bloque de memoria principal a sólo una línea posible de caché. Se usa una parte de la dirección como numero de línea, proporcionando así una correspondencia o asignación única. Es simple y poco costosa de implementar. Desventaja: Hay una posición concreta de caché para cada bloque dado y si un programa referencia repetidas veces a palabras de dos bloques diferentes asignados en la misma línea, los bloques se estarían intercambiando continuamente en la caché y la tasa de aciertos sería baja.
  + N° línea caché = N° bloque ref. MOD N° líneas caché
* Correspondencia (totalmente) asociativa: Permite que cada bloque de memoria principal pueda cargarse en cualquier línea de la caché. La lógica de control de la caché interpreta una dirección de memoria como una etiqueta y un campo de palabra. El campo de etiqueta identifica unívocamente un bloque de memoria principal. En esta técnica hay flexibilidad para que cualquier bloque sea reemplazado cuando se va a escribir uno nuevo en la caché. Desventaja: se necesita una circuitería compleja para examinar en paralelo las etiquetas de todas las líneas de caché (para determinar si un bloque determinado está en la caché) y es costoso en tiempo
* Correspondencia asociativa por conjuntos: Un bloque puede almacenarse en un conjunto restringido de lugares en la caché. Un conjunto es un grupo de líneas en la caché. Se llama correspondencia asociativa por conjuntos de k vías. N° conjunto = N° bloque ref. MOD N° conjuntos caché.

**Fuentes**:

* “Clase 07”
* “Capítulo 4.3 - Memoria caché” (Stalling 5ta ed. Pág 111, 115, 121)
* Material de Organización de Computadoras