Estructura y Tecnología de Computadores

Módulo F. El subsistema de memoria

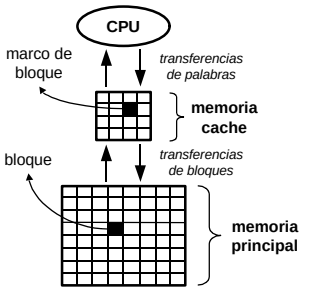
Tema 13. Memoria cache

José Manuel Mendías Cuadros
Dpto. Arquitectura de Computadores y Automática
Universidad Complutense de Madrid

contenidos

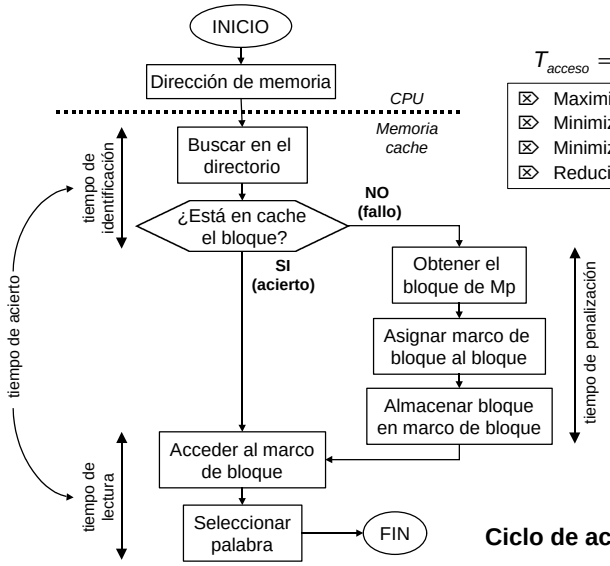
1 introducción

- Memoria pequeña y rápida situada entre el procesador y la memoria principal
  - Almacena una copia de la porción de información actualmente en uso de la memoria
- **Objetivo**: disminuir el tiempo de acceso a memoria
- **Estructura** del sistema memoria cache/principal:
  - *Mp (memoria principal):*
    - formada por 2^n palabras direccionables
    - "dividida" en nB bloques de tamaño fijo de 2^K palabras por bloque
  - *Mc (memoria cache):*
    - formada por nM marcos de bloque de 2^K palabras cada uno (nM<<nB)
  - *Directorio (en memoria cache):*
    - indica qué subconjunto de los nB bloques residen en los nM marcos de bloque



Notación:
nB: número de bloques
nM: número de marcos de bloque
B: dirección del bloque
M: dirección del marco de bloque
P: palabra dentro del bloque

1. introducción



$$T_{acceso} = T_{acierto} + (1-H)T_{penalizacion}$$

- Maximizar la tasa de aciertos
- Minimizar el tiempo de acceso
- Minimizar el tiempo de penalización
- Reducir el coste hardware

Ciclo de acceso al sistema Mc/Mp

1. introducción

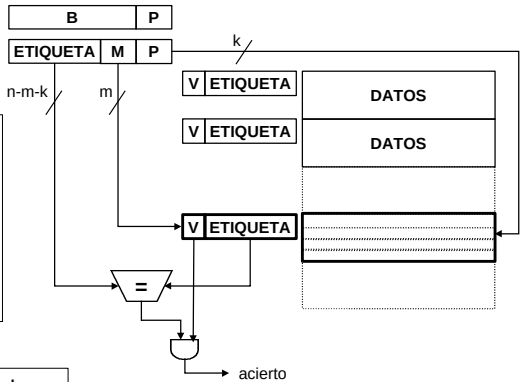
Aspectos básicos de diseño

- **Política de emplazamiento**:
  - Necesaria ya que existen menos marcos de bloque en Mc que bloques en Mp
  - Determina en que marco de bloque puede cargarse cada bloque
  - Determina que bloque entre los posibles ocupa en un cierto momento un marco de bloque dado
- **Política de reemplazamiento**:
  - Necesaria ya que cuando se carga un nuevo bloque en Mc debe reemplazarse uno de los existentes
  - Determina que bloque debe ser reemplazado
- **Política de actualización**:
  - Necesaria para mantener la coherencia entre la Mc y la Mp
  - Determina cuándo se actualiza un bloque de la Mp si éste ha sido modificado en la Mc
- **Política de búsqueda**:
  - Determina qué bloques (y en qué momento) deben cargarse en Mc
- **Organización de la cache**:
  - Tamaño
  - Tamaño de bloque
  - Niveles de cache
  - Cache unificada o cache dividida

2. políticas de emplazamiento

Emplazamiento directo

- Cada bloque B tiene asignado un marco fijo M en donde ubicarse.
  - Este marco viene dado por la expresión: **M = B mod nM**
  - Si $nM = 2^m$ entonces M = (m bits menos significativos de B)
  - El directorio almacena para cada marco una etiqueta con los n-k-m bits que completan la dirección del bloque almacenado
  - El acceso al marco y al directorio es directo. Para conocer si un bloque está cargado en Mc, basta comparar las etiquetas

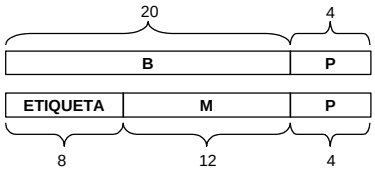


- *Ventajas*:
  - baja complejidad hardware
  - rapidez en la identificación
  - directorio pequeño
- *Desventajas*:
  - bata tasa de fallos si varios bloques **compiten** por el mismo marco

2. políticas de emplazamiento

Emplazamiento directo: ejemplo

- *Memoria principal*: 16 Mb direccionable por bytes (dirección de 24 bits)
- *Memoria cache*: 64 Kb
- *Tamaño de bloque*: 16 bytes
  - Número de bloques en Mp (nB) = 1M (dirección de 20 bits)
  - Número de marcos de bloque en Mc (nM) = 4K (dirección de 12 bits)



Direcciones de CPU	*Marco de bloque*
XXXXXXXX000000000000XXXX	000h
XXXXXXXX000000000001XXXX	001h
XXXXXXXX000000000010XXXX	002h
.....	.....
XXXXXXXX111111111110XXXX	FFEh
XXXXXXXX111111111111XXXX	FFFh

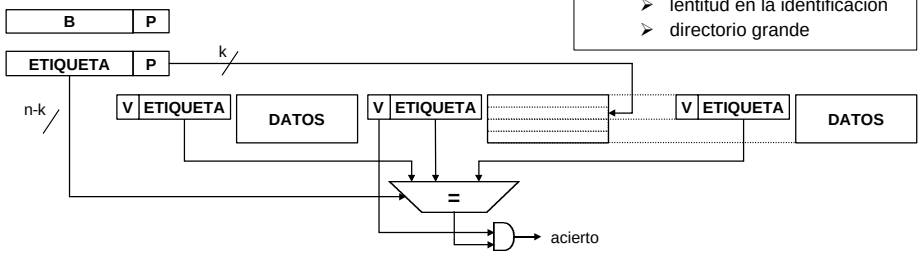
Información que se almacena en la etiqueta de marco de bloque

2. políticas de emplazamiento

Emplazamiento asociativo

- Cada bloque B puede ubicarse en **cualquier** marco M
  - El directorio guarda para cada marco una etiqueta con los n-k bits que identifican al bloque almacenado
  - El acceso al marco y al directorio es asociativo. Para conocer si un bloque está cargado, la Mc compara la etiqueta dada con todas las etiquetas del directorio

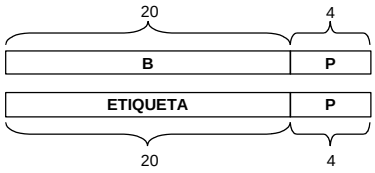
- *Ventajas*:
  - baja tasa de fallos
- *Desventajas*:
  - alta complejidad hardware
  - lentitud en la identificación
  - directorio grande

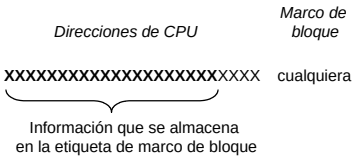


2. políticas de emplazamiento.

Emplazamiento asociativo: ejemplo

- *Memoria principal*: 16 Mb direccionable por bytes (dirección de 24 bits)
- *Memoria cache*: 64 Kb
- *Tamaño de bloque*: 16 bytes
  - Número de bloques en Mp (nB) = 1M (dirección de 20 bits)
  - Número de marcos de bloque en Mc (nM) = 4K (dirección de 12 bits)

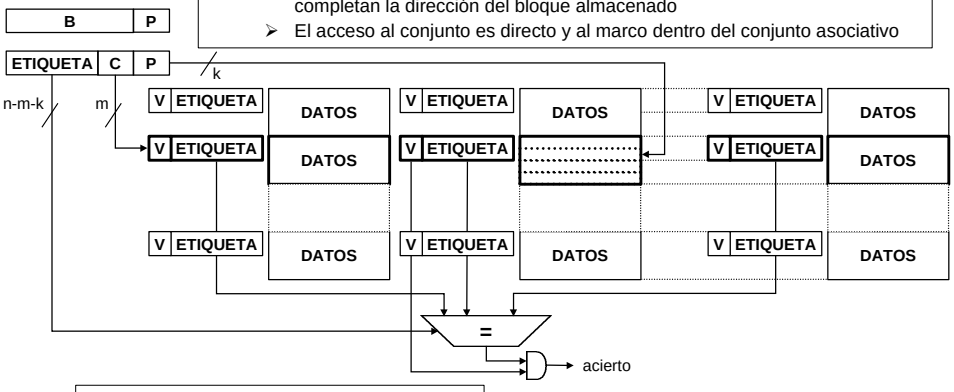




2. políticas de emplazamiento

Emplazamiento asociativo por conjuntos

- La Mc está dividida en nC conjuntos de nM/nC marcos cada uno.
- Cada bloque B tiene asignado un conjunto fijo C y puede ubicarse en cualquiera de los marcos que componen dicho conjunto
  - Este conjunto viene dado por la expresión: **C = B mod nC**
  - Si nC = 2^m entonces C = (m bits menos significativos de B)
  - El directorio almacena para cada marco una etiqueta con los n-k-m bits que completan la dirección del bloque almacenado
  - El acceso al conjunto es directo y al marco dentro del conjunto asociativo



2. políticas de emplazamiento

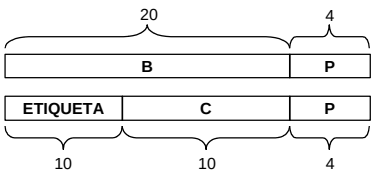
Emplazamiento asociativo por conjuntos (cont.)

- *Ventajas*:
  - Es un enfoque intermedio entre emplazamiento directo y asociativo
- El valor nM/nC se denomina **grado de asociatividad** o **número de vías** de la Mc
  - Grado de asociatividad = 1, equivale a emplazamiento directo
  - Grado de asociatividad = nM, equivale a emplazamiento asociativo
- El grado de asociatividad afecta al rendimiento de esta política de emplazamiento
  - Al aumentar el grado de asociatividad disminuyen los fallos por competencia por un marco
  - Al aumentar el grado de asociatividad aumenta el tiempo de acceso y el coste hardware
  - *Grado óptimo*: entre 2 y 16
  - *Grado más común*: 2

2. políticas de emplazamiento

Emplazamiento asociativo por conjuntos: ejemplo

- Memoria principal: 16 Mb direccionable por bytes (dirección de 24 bits)
- Memoria cache: 64 Kb
- Tamaño de bloque: 16 bytes
  - Número de bloques en Mp (nB) = 1M (dirección de 20 bits)
  - Número de marcos de bloque en Mc (nM) = 4K (dirección de 12 bits)
- 4 vías
  - Número de marcos por conjunto = 4
  - Número de conjuntos en Mc (nC) = 1K (dirección de 10 bits)



Direcciones de CPU	*Conjunto*	*Marco de bloque*
XXXXXXXXXX0000000000XXXX	000h	cualquiera de 4
XXXXXXXXXX0000000001XXXX	001h	cualquiera de 4
XXXXXXXXXX0000000010XXXX	002h	cualquiera de 4
.....	.....	.....
XXXXXXXXXX1111111110XXXX	3FEh	cualquiera de 4
XXXXXXXXXX1111111111XXXX	3FFh	cualquiera de 4

Información que se almacena en la etiqueta de marco de bloque

3. políticas de reemplazamiento

- *¿Qué sucede cuando se produce un fallo y todos los marcos de bloque están ocupados?*
  - Es necesario elegir uno y sobreescribir el nuevo bloque sobre el ya existente
- **Espacio de reemplazamiento**: conjunto de posibles bloques que pueden ser reemplazados por el nuevo bloque
  - *Directo*: el bloque que reside en marco que el nuevo bloque tiene asignado. Al no existir alternativas no se requieren algoritmos de reemplazamiento
  - *Asociativo*: cualquier bloque que resida en de la cache
  - *Asociativo por conjuntos*: cualquier bloque que resida en el conjunto que el nuevo bloque tiene asignado
- **Algoritmos** (implementados en hardware):
  - **Aleatorio**: se escoge un bloque del espacio de reemplazamiento al azar
  - **FIFO**: se sustituye el bloque del espacio de reemplazamiento que lleve más tiempo cargado
  - **LRU *(last recently used)***: se sustituye el bloque del espacio de reemplazamiento que lleve más tiempo sin haber sido referenciado
  - **LFU (*last frequently used*):** se sustituye el bloque del espacio de reemplazamiento que haya sido menos referenciado

3. políticas de reemplazamiento

- **Implementación de algoritmos**:
  - Se utilizan **registros de edad** que se actualizan convenientemente
  - El tamaño de los registros dependerá del tamaño del espacio de reemplazamiento:
    - ⇨ *ejemplo*: Mc asociativa por conjuntos de 2 vías: basta con 1 bit por marco

- *Algoritmo LRU:*
  - Si hay un **acierto** al acceder a un bloque:
    - ⇨ Se lee el valor de su contador de edad
    - ⇨ Todos los contadores del espacio de reemplazamiento que tengan un valor inferior al valor leído se incrementan
    - ⇨ Se pone su contador de edad a 0
  - Si hay un **fallo** al acceder a un bloque:
    - ⇨ Si el espacio de reemplazamiento no está completo, se pone el contador del nuevo bloque a 0 y los demás se incrementan
    - ⇨ Si el espacio de reemplazamiento está completo, se reemplaza el bloque cuyo contador tenga el valor máximo.
  - Observar como el valor de contador nunca supera (*tamaño del espacio de remplazamiento*) -1

4. políticas de actualización

- *¿Qué sucede cuando la CPU escribe una palabra?*
  - Si se escribe sobre uno de los bloques cargados en la cache, cuando este bloque sea reemplazado deberá actualizarse el bloque correspondiente de Mp
- **Escritura inmediata (*write-through*):** cada vez que se hace una escritura en la Mc se actualiza inmediatamente la Mp.
  - *Variantes* en función de lo que sucede en caso de fallo de escritura:
    - **Con asignación en escritura**: un bloque se carga en Mc como consecuencia de fallos de lectura y escritura
    - **Sin asignación en escritura**: un bloque se carga en Mc solamente como consecuencia de fallos de lectura, lo fallos de escritura no cargan bloque y se hacen siempre con Mp
  - *Ventajas:* bajo coste hardware y consistencia en todo momento
  - *Desventajas:* aumenta el tráfico entre Mc-Mp
  - Para evitar que el procesador tenga que esperar a que la actualización se realice, se utiliza un buffer intermedio (4 referencias)
- **Post-escritura (*copy-back*):** la Mp se actualiza sólo cuando se reemplaza el bloque
  - En Mc se cargan bloques tanto por fallos de lectura como de escritura
  - Se utiliza un **bit de actualización** por bloque que indica si debe actualizarse en Mp.
    - Cuando se realiza una escritura se activa
    - Cuando se realiza un reemplazamiento se comprueba si está activado o no.
  - *Ventajas:* disminuye el tráfico entre Mc y Mp y disminuye el tiempo de acceso para escritura
  - *Desventajas:* inconsistencia
  - Para evitar retrasos en los reemplazamientos se utiliza un buffer intermedio (1 bloque)

5. políticas de búsqueda

- **Búsqueda por demanda:** un bloque se trae a Mc cuando se necesita, es decir, como consecuencia de un fallo
- **Búsqueda anticipativa:** un bloque se trae a Mc antes de que se necesite para reducir la tasa de fallos.
  - Lo más común es elegir el bloque siguiente al referenciado (***one block lookahead***).
  - *Variantes*:
    - **Prebúsqueda siempre**: la primera vez que se referencia un bloque se prebusca el siguiente
    - **Prebúsqueda por fallo**: si se produce un fallo al acceder a un bloque se buscan dicho bloque y el siguiente

17

6. organización de la cache

⌦ **Tamaño de la Mc**:
- Es un aspecto muy dependiente de la tecnología
- Al aumentar el tamaño de la Mc se disminuye la tasa de fallos
- Al aumentar el tamaño de la Mc aumenta su tiempo de acceso
- Debe ser lo suficientemente pequeña para que el coste por bit del sistema Mc/Mp sea próximo al de la Mp
- *Tamaños óptimos*: entre 1 y 512 Kb

⌦ **Tamaño de bloque**:
- No debe ser ni muy grande ni muy pequeño.
- Al aumentar el tamaño de bloque se captura mejor la localidad espacial. Si el tamaño de bloque es demasiado grande los datos de un bloque pueden estar demasiado alejados.
- Al aumentar el tamaño de bloque disminuye el número de marcos de bloque, capturando peor la localidad temporal ya que un bloque es reemplazado al poco tiempo de ser cargado
- Al aumentar el tamaño de bloque aumenta el tiempo de transferencia de bloque entre Mp y Mc, aumentando el tiempo de penalización
- *Tamaños óptimos*: entre 4 y 8 unidades direccionables

18

6. organización de la cache

⌦ **Niveles de cache**:
- Se amplía el paradigma de la jerarquía de memoria aumentando el número de niveles de memoria cache.
- Lo más habitual es tener 2 niveles de cache ubicados en lugares diferentes
- Una **cache interna (*on-chip*)**: para reducir los tiempos de acceso
  - ⇨ *Rápida*: no se accede a través del bus
  - ⇨ *Pequeña*: debe caber dentro de la CPU
  - ⇨ *Sencilla*: poco hardware de gestión (emplazamiento directo)
- Una **cache externa (*off-chip*)**: para reducir la tasa de fallos
  - ⇨ Grande y con cierto grado de asociatividad (1-4)

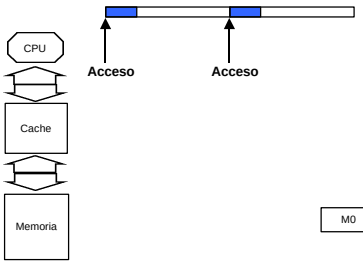
⌦ **Caches separadas**:
- La CPU procesa instrucciones y datos por ello en lugar de tener una cache unificada para ambos tipos de referencias, puede tenerse:
  - ⇨ Una cache de lectura para **instrucciones**
  - ⇨ Una cache de lectura/escritura para **datos**
- *Ventajas*:
  - ⇨ Pueden diseñarse con parámetros de diseño diferentes
  - ⇨ Duplica el ancho de banda (dos accesos en paralelo)
- *Desventajas*:
  - ⇨ Por separado tienen mayores tasas de fallo que una unificada

7. Ejemplo: memoria cache en un pentium

- **2 niveles de cache**
  - 2 caches internas, una de datos y otra de instrucciones
  - 1 cache externa
- **Cache interna:**
  - *tamaño*: 8 Kb
  - *tamaño de bloque*: 32 bytes
  - *política de emplazamiento*: asociativa por conjuntos de 2 vías
  - *política de reemplazamiento*: LRU (1 bit)
  - *política de actualización (para la de datos)*: post-escritura o inmediata (seleccionable)
- **Cache externa:**
  - *tamaño*: 256 o 512 Kb
  - *tamaño de bloque*: 32, 64 o 128 bytes
  - *política de emplazamiento*: asociativa por conjuntos de 2 vías
  - *política de reemplazamiento*: LRU (1 bit)
  - *política de actualización*: post-escritura
- **Control de cache:**
  - En el registro de estado existen 2 bits para controlar la cache interna
    - **CD** (*cache disable*): para inhabilitar la cache
    - **NW** (*not write throught*): para seleccionar la política de actualización
  - En el repertorio de instrucciones existen 2 instrucciones:
    - **INDV**: limpia la memoria cache
    - **WBINVD**: limpia la memoria cache, actualizando previamente la memoria principal

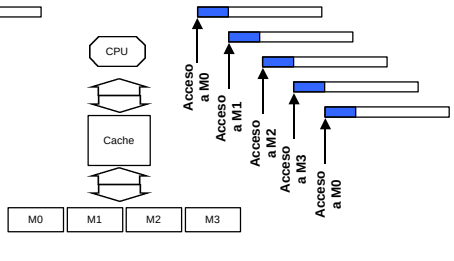
8. memoria entrelazada

- La memoria entrelazada es un método de reducir el tiempo de acceso efectivo a Mp
  - Dividiendo la Mp en módulos independientes haciendo posible un acceso paralelo a los datos.
- *Ejemplo*:
  - Mp: 1 ciclo en enviar la dirección, 10 ciclos en el acceso y 1 ciclo en el envío del dato
  - Mc: Bloques de tamaño 4 palabras de 4 bytes cada uno



BUS Y MEMORIA DE 1 PALABRA
ACCESO SECUENCIAL

1+4·(10+1)=45 => ***0,35 bytes/ciclo***



ANCHURA DE BUS DE 1 PALABRA
ANCHURA DE MEMORIA DE 4 PALABRAS
ACCESO SOLAPADO A LOS MÓDULOS

1+10+4x1=15 => ***1 byte/ciclo***

8. memoria entrelazada

De orden bajo

- La Mp de 2^n palabras se divide en 2^m módulos de 2^{n-m} palabras cada uno
  - direcciones consecutivas se ubican en módulos consecutivos

De orden alto

- La Mp de 2^n palabras se divide en 2^m módulos de 2^{n-m} palabras cada uno
  - cada módulo almacena 2^{n-m} palabras consecutivas