SIMULADOR DE MEMORIA CACHE 2003

INTRODUCCION

La memoria de los computadores, aunque parezcan conceptualmente sencillas, presentan talvez la más amplia diversidad de tipos, tecnología, estructura, prestaciones y coste, comparados con todos los componentes de un computador. Ninguna tecnología es óptima para satisfacer las necesidades de memoria de un computador. En consecuencia, un computador convencional está equipado con una jerarquía de subsistemas de memoria, algunos internos (accesibles por el procesador), y otros externos (accesibles por el procesador mediante módulos de entrada y salida).

Las características claves de los sistemas de memoria de computadores son:

Ubicación: CPU, Interna (principal), Externa (secundaria).

Capacidad: Tamaño de palabra, Número de palabras.

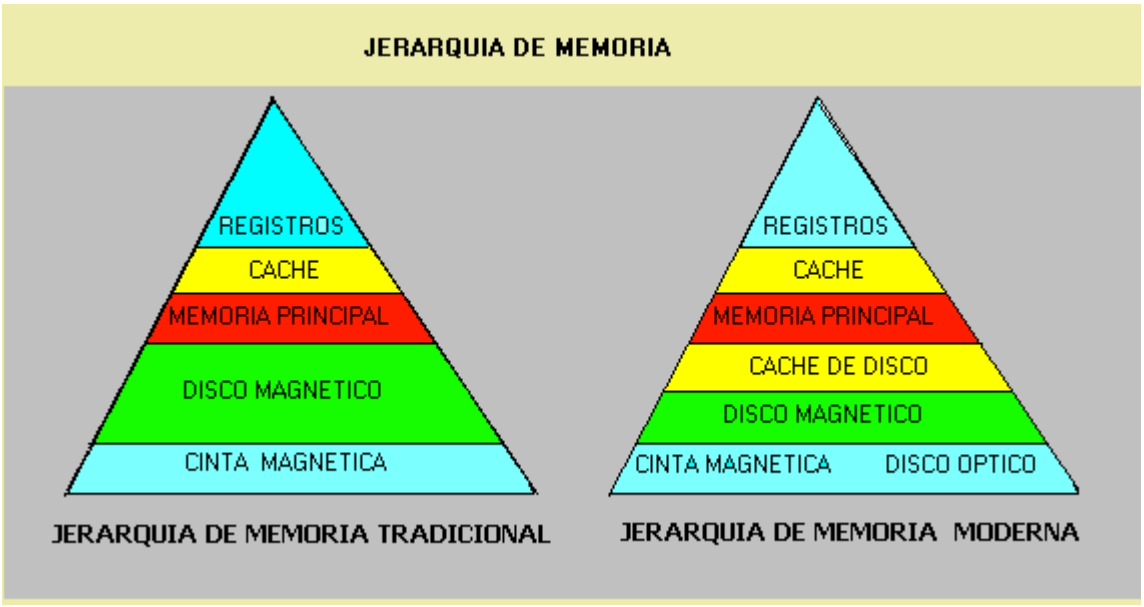
Unidad de transferencia: Palabra, Bloque.

Método de acceso: Secuencial, Directo, Aleatorio, Asociativo.

Prestaciones: Tiempo de acceso, Tiempo de ciclo, Velocidad de transferencia.

Dispositivo físico: Semiconductores, soporte magnético.

Características físicas: Volátil, No volátil, Borrable, No borrable.



MEMORIA PRINCIPAL

La tecnología ha afectado de diferente manera al procesador y a la memoria. Los avances tecnológicos han supuesto un aumento importante de la densidad de las memorias pasando de memorias de 16 bits en 1965 a memorias de 16 Mbits en 1995, cumpliéndose así las previsiones de la ley de Moore. Este aumento en la densidad de componentes, no se ha visto correspondido con incrementos espectaculares en los tiempos de acceso.
Los tiempos de acceso se ven multiplicados por un factor de 1,07 cada año, mientras que las velocidades de los procesadores se ven multiplicados por un factor de 1,55. Esto provoca que cada vez sea mayor la barrera que separa los rendimientos del procesador de las velocidades de la memoria, llegando a ser el acceso a éstas uno de los principales cuellos de botella con los que se puede encontrar un diseñador.
Una de las técnicas habituales para minimizar este problema es utilizar memorias principales entrelazadas. Sin embargo la densidad de las memorias RAM está creciendo más rápidamente que la necesidad de memoria de los usuarios. Esto lleva aparejado una disminución de módulos de memoria en los computadores y por lo tanto disminuye la capacidad de entrelazamiento. Otra técnica posible es aumentar el ancho de bus, pero esto encarece mucho los sistemas.
En lugar de mejorar las interfaces de la memoria por medio de una organización externa de los módulos DRAM, en la actualidad se busca mejorar la organización interna. A continuación se comenta alguna de las organizaciones ya comercializadas:

1. EDRAM (Enhanced DRAM) Incluye una pequeña memoria SRAM que almacena la última fila seleccionada de modo que si el siguiente acceso se realiza en la misma fila, solo se debe acceder a la rápida SRAM. Además permite realizar una lectura simultáneamente con el refresco o con una escritura.

2. CDRAM (Cache DRAM): similar a la anterior pero con una memoria caché SRAM que almacena varias filas, siendo más efectiva para los accesos aleatorios de memoria.

3. SDRAM (Synchronus DRAM) en lugar de ser una memoria asíncrona como el resto, esta intercambia datos con el procesador sincronizado por una señal de reloj externa. Incluye un módulo SRAM que recoge la dirección y la orden y responde después de un cierto número de ciclos de reloj. Entre tanto el procesador puede ir realizando otra tarea.

4. RDRAM (Rambus DRAM) A diferencia de las anteriores en este caso se cambia la interfaz entre la DRAM y el procesador, sustituyendo las líneas de selección de fila y de columna por un bus que permite otros accesos mientras se da servicio a un acceso, usando transferencias de ciclo partido.

Dado que todas las estrategias decrementan los tiempos de acceso pero no reducen las diferencias entre procesador y memoria principal de manera significativa, se suele aprovechar el principio de localidad de los programas para introducir una memoria SRAM entre el procesador y la memoria DRAM lo que produce bueno resultados. A esta memoria se le llama caché y se comenta más adelante.

En computadores antiguos, la forma más común de acceso aleatorio para la memoria principal consistía en una matriz de pequeños anillos ferromagnéticos denominados núcleos. El advenimiento de la microeléctronica y sus ventajas acabó con las memorias de núcleos. Hoy en día es universal el uso de memorias semiconductoras para la memoria principal.
Memorias semiconductoras; la mas común es la denominada memoria de acceso aleatorio (RAM, ""Random Acces Memory"). Características de las RAM es que es

posible tanto leer datos como escribir rápidamente nuevos datos en ella, es volátil, tienen que estar continuamente alimentadas eléctricamente.

Las tecnologías de las RAM se dividen en dos variantes: estáticas y dinámicas. Una RAM dinámica está hecha con celdas que almacenan los datos como cargas en condensadores.
En una RAM estática los valores binarios se almacenan utilizando configuraciones de puertas formando biestables("flip_flop").
En otras palabras podemos decir que las memorias semiconductoras pueden ser de los siguientes tipos:
RAM, ROM, PROM, EPROM, EEPROM y las memorias flash.

PROCESO DE GESTION DE MEMORIA

El proceso de gestión de memoria.....

RENDIMIENTO DEL SISTEMA DE MEMORIA

Fórmula básica:

Tacceso = Tacierto + (1-H)T transf_bloque

Escritura directa con asignación en escritura:

Tacceso = Tacierto + (1-H)T transf_bloque + Porcentaje_escrituras(tm - tc)*

Donde:
tm = tiempo de ciclo de memoria
tc = tiempo de ciclo caché

Escritura directa sin asignación en escritura:

Tacceso = Tacierto + (1-H) (1- H) (1 - Porcentaje_escrituras) Ttransf_bloque +*

Porcentaje_escrituras(tm - tc)*

Postescritura:

*Tacceso = Tacierto + (1-H)T transf_bloque + Prob_modificado * (1-H)T*

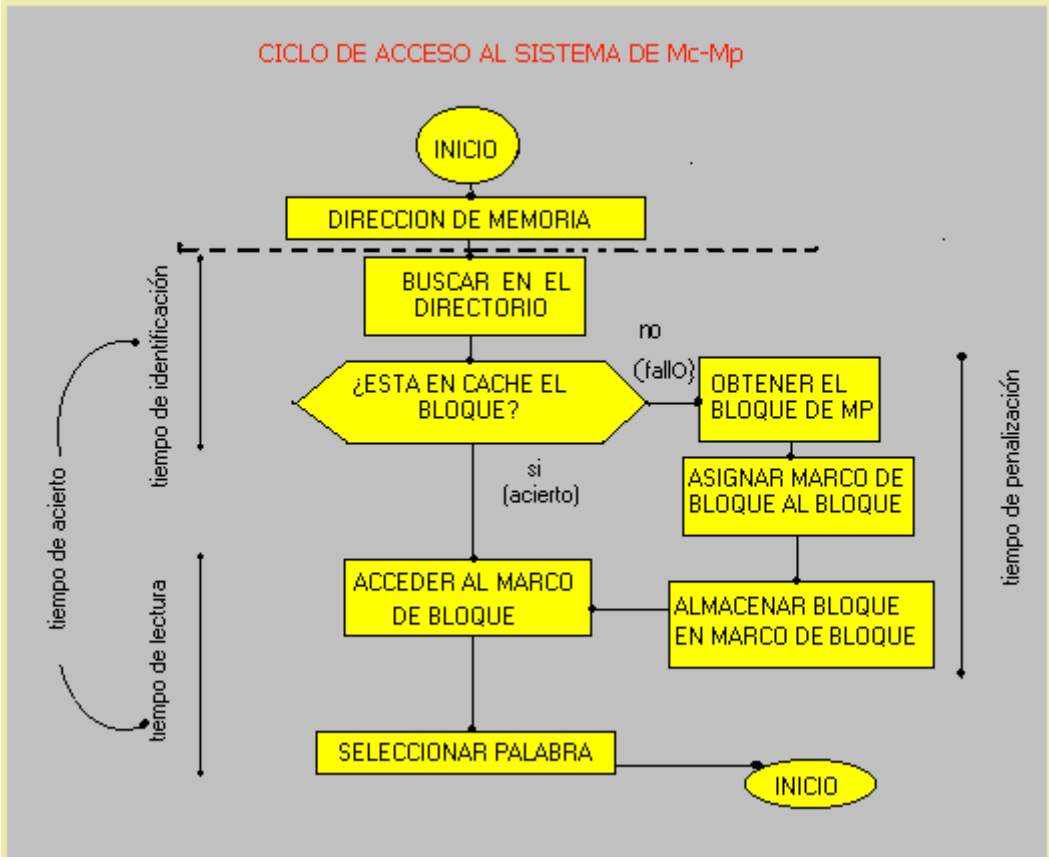
transf_bloque

Tamaño de la MC

- Es un aspecto muy dependiente de la tecnología
- Al aumentar el tamaño de la MC se disminuya 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 bit0. 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 al tamaño del bloque disminuye el número de marcos de bloque, capturando ineficientemente la localidad temporal ya que un bloque es reemplazado al poco tiempo de ser cagado.
- Al aumentar el tamaño de bloque aumenta al tiempo de transferencia de bloque entre MP y MC; aumentando el tiempo de penalización.
- Tamaños óptimos: entre 4 y 8 unidades direccionables.



LA MEMORIA CACHE (MC)

Memoria pequeña y rápida, situada entre el procesador y la memoria principal.
Función: Almacena una copia de la porción de información actualmente en uso de la memoria.
Objetivo: Disminuir el tiempo de acceso de la memoria.
Principio básico: Localidad espacial y temporal.

Estructura del sistema memoria Caché/Principal.

MP (memoria principal):
formada por 2 a la n palabras direccionables, "dividida" en nB bloques de tamaño fijo de 2 a la k palabras por bloque.

MC (memoria caché):
formada por nM marcos de bloque de 2 a la k palabras cada uno (nM mucho menor que nB)

DIRECTORIO (memoria caché):
Indica que subconjunto de los nB bloques residen en los nM marcos de bloque.

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

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 ineficientemente 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

Niveles de caché:
Se amplía el paradigma de la jerarquía de memoria aumentando el número de niveles de memoria caché.
Lo más habitual es tener 2 niveles de caché ubicados en lugares diferentes
Una caché 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 caché 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 caché unificada para ambos tipos de referencias, puede tenerse:

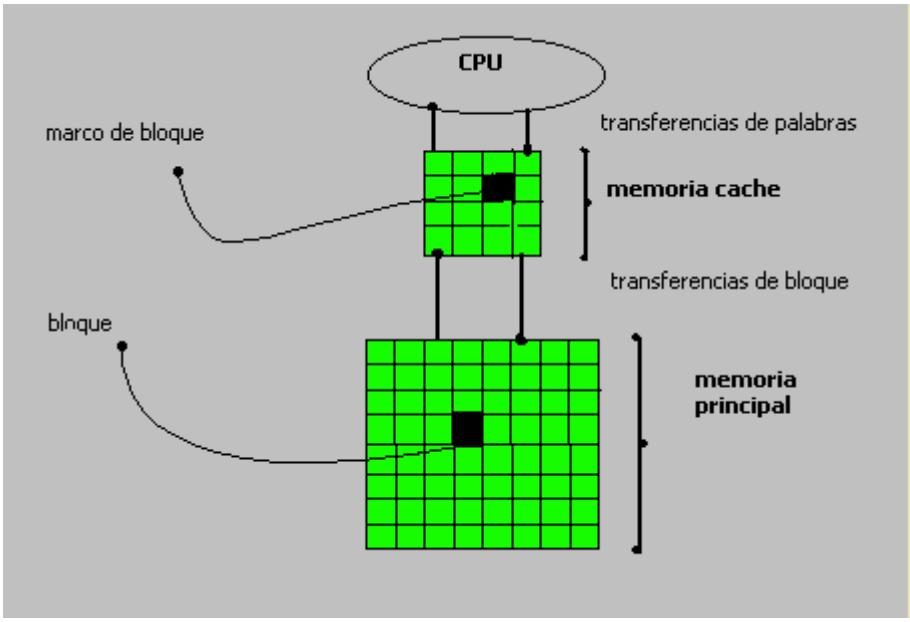
- Una caché de lectura para instrucciones
- Una caché de lectura/escritura para datos

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



Ejemplo: memoria caché en un pentium

- 2 niveles de caché

- 2 cachés internas, una de datos y otra de instrucciones
- 1 caché externa

- Caché 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)

- Caché 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 caché:

- En el registro de estado existen 2 bits para controlar la caché interna

- CD (caché disable): para inhabilitar la caché
- NW (not write throught): para seleccionar la política de actualización

- En el repertorio de instrucciones existen 2 instrucciones:

- INDV: limpia la memoria caché
- WBINVD: limpia la memoria caché, actualizando previamente la memoria principal

FACTORES QUE INFLUYEN EL RENDIMIENTO DE LA MEMORIA CACHE (MC)

El tiempo de acceso durante la ejecución del programa será:
*Tacceso = Na * Tc + Nf * Tp*

Donde:

Na: número de referencias con acierto.
Nf: número de referencias con fallo.
Tc: tiempo de acceso a una palabra de memoria caché.
Ta: tiempo de acceso a una palabra de memoria principal.

El tiempo de acceso medio durante la ejecución del programa valdrá:

Tacceso_medio = Tacceso/Nr

El cual a su vez es igual a:

*Tacceso_medio = Ta * Tc + Tf * Tp*

Donde:
Ta = Na/Nr = Tasa de aciertos
Tf = Nf/Nr = Tasa de fallos
Nr = el número total de referencias a memoria.

El tiempo de acceso a un bloque de memoria principal (MP) constituye la componente principal del tiempo total de penalización de un fallo. El tiempo de penalización también se le conoce como tiempo de transferencia. En la siguiente expresión hemos considerado ambos términos equivalentes:

*Tacceso_medio = Tacierto + Tf * Penalización_fallo*

Donde:
Tacierto = Ta * Tc , válido si frente a un fallo, el bloque se lleva a memoria principal al tiempo que la palabra referenciada del bloque se lleva en paralelo a

memoria caché. Si estas dos acciones en vez de realizarse en paralelo se realizan en forma secuencial entonces = Tacierto = Tc, ya que toda referencia implicaría un acceso a memoria caché y por tanto la expresión anterior se transformaría en:

*Tacceso_medio = Nr * Tc + Nf * Tp = Tacceso/Nr = Tc + Tf * Tp*

Las fórmulas anteriores son las fórmulas básicas para el cálculo de tiempo de acceso. Éstas no toman en cuenta las políticas de escritura de las memorias.

Las fórmulas de tiempo de acceso tomando en cuenta las políticas de escritura se definen a continuación:

Escritura directa con asignación de escritura:

*Tacceso_medio = Tacierto + Tf * Penalización_fallo + Porcentaje_escrituras *(Ta- Tc)*

Donde:

Porcentaje_escrituras: es el porcentaje de datos de tipo escritura que se encuentran en la memoria principal.

Escritura directa sin asignación en escritura:

*Tacceso_medio = Tacierto + Tf * Penalización_fallo + (Porcentaje_lecturas) * Penalización_fallo + Porcentaje_escrituras *(Ta- Tc)*

Donde:

Porcentaje_lecturas: es el porcentaje de datos de tipo lectura que se encuentran en la memoria principal.

Post-Escritura:

*Tacceso_medio = Tacierto + Tf * Penalizacion_fallo + Prob_modificado * Penalizacion_fallo * Tf*

Donde:

Prob_modificado = es la probabilidad de que u bloque en memoria principal se halla modificado. Es decir: 1/(No. De bloques de memoria principal)

POLITICAS DE EMPLAZAMIENTO

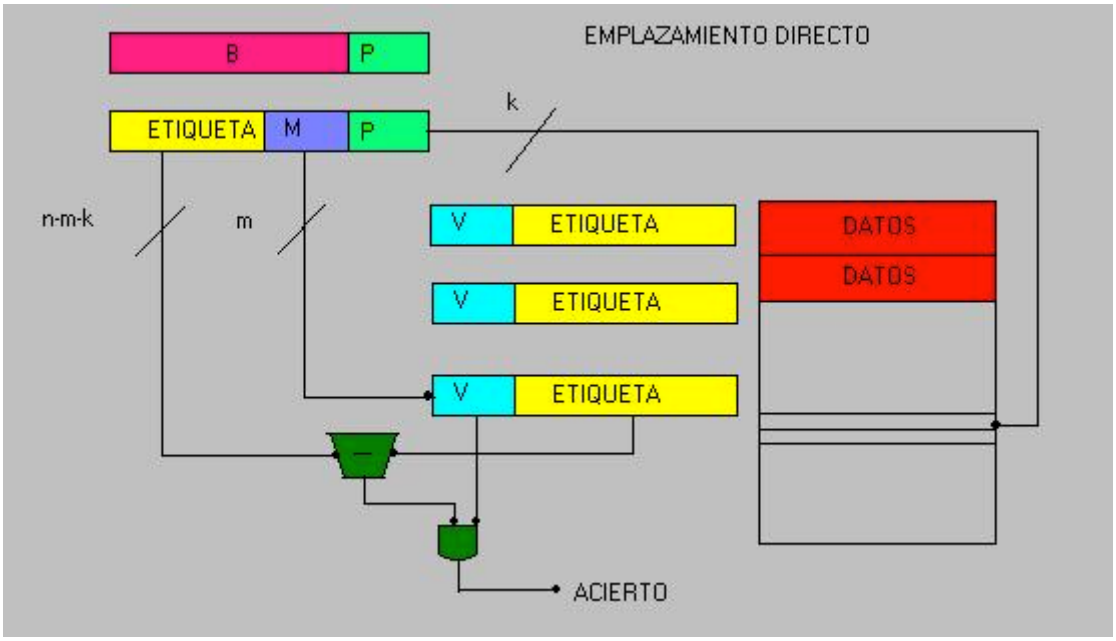
Como la memoria caché es mucho más pequeña que la memoria principal, un marco de bloque de caché no puede estar ocupado permanentemente por el mismo bloque. Esto da lugar a las políticas de:

- Emplazamiento
- Reemplazamiento
- Actualización

El emplazamiento consiste en determinar dónde se coloca un bloque de información traído de la memoria principal cuando hay espacio en la memoria caché. El reemplazamiento selecciona el bloque que se sustituye cuando la caché está llena. En cuanto a las políticas de actualización determinan como se actualizarán los datos en la jerarquía de memoria cuando la Unidad Central de Proceso (UPC) quiere realizar una operación de escritura en memoria. En los siguientes apartados estudiaremos cada una de estas políticas, indicando como afectan sus parámetros de diseño al coste rendimiento del sistema.

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 = 2m 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:

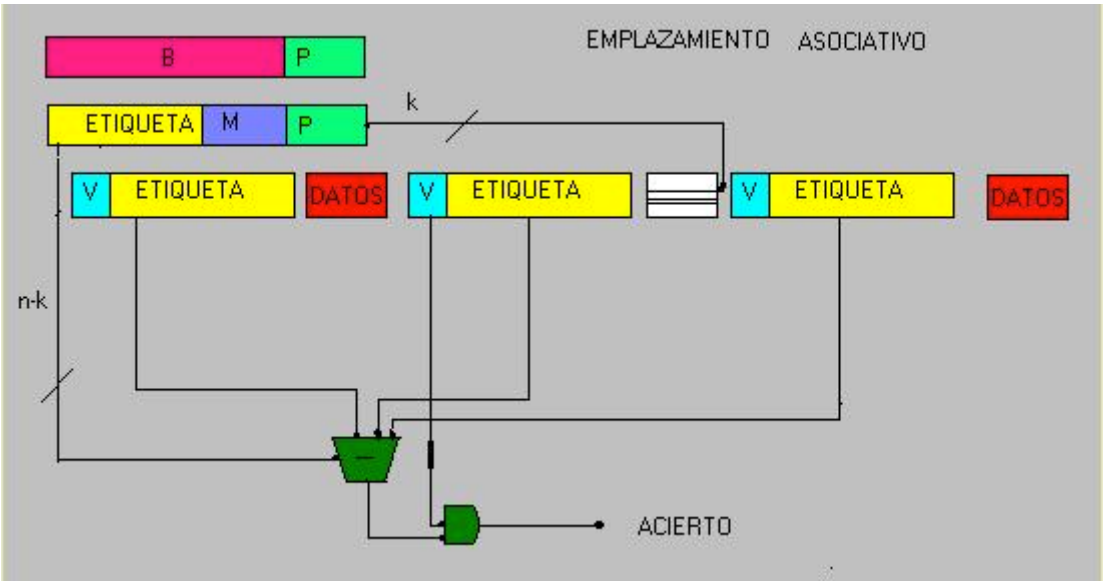
- baja complejidad hardware
- rápidez en la identificación
- directorio pequeño

Desventajas:

- baja tasa de fallos si varios bloques compiten por el mismo marco.

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

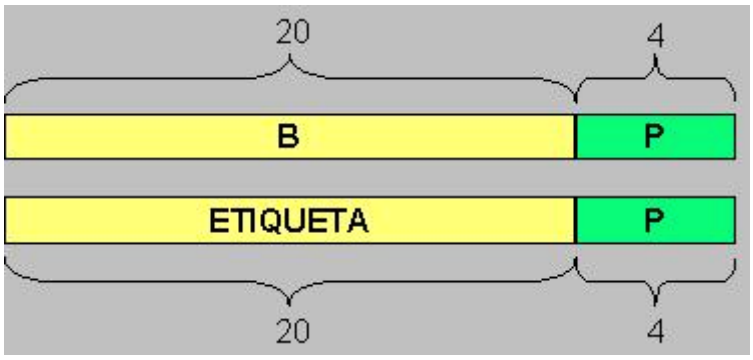


Ventajas:

- baja tasa de fallos

Desventajas:

- alta complejidad hardware
- lentitud en la identificación
- directorio grande



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 Marcode

bloque

XXXXXXXXXXXXXXXXXXXX
XXXX cualquiera

_____

|

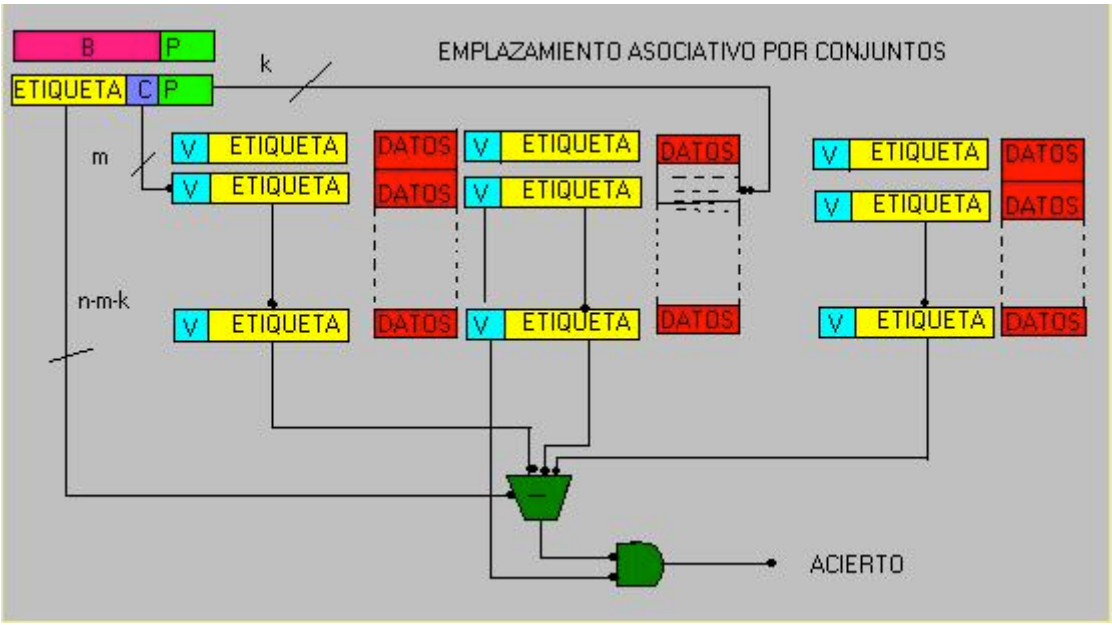
V

Información que se almacena

en la etiqueta de marco de bloque

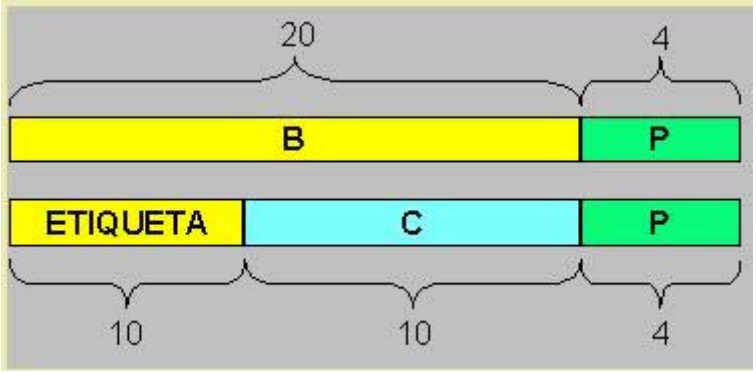
EMPLAZAMIENTO ASOCIATIVO POR CONJUNTOS

- 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 = 2m 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.



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



Ejemplo:

Direcciones de CPU	Conjunto bloque	Marco de
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

El emplazamiento asociativo por CONJUNTOS

POLITICAS 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

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:

1. Se lee el valor de su contador de edad
2. Todos los contadores del espacio de reemplazamiento que tengan un valor inferior al valor leído se incrementan
3. Se pone su contador de edad a 0

Si hay un fallo al acceder a un bloque:

1. Si el espacio de reemplazamiento no está completo, se pone el contador del nuevo bloque a 0 y los demás se incrementan
2. 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

POLITICAS DE ACTUALIZACION

¿Qué sucede cuando la CPU escribe una palabra?
Si se escribe sobre uno de los bloques cargados en la caché, 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)

POLITICAS DE BUSQUEDA

- 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

POLITICAS DE ACTUALIZACION

ESCRITURA INMEDIATA

Cuando se produce un acierto de escritura (es decir cuando el dato que queremos modificar está en la caché), se actualiza el dato tanto en la memoria principal como en la memoria caché. En los fallos de lectura, independientemente de que el bloque de caché a sustituir haya sido modificado o no siempre se sobreescribe, porque existe coherencia.
La principal ventaja de esta política es que no existe inconsistencia nunca, esto tiene como efecto que los fallos de lectura sean más rápidos porque no necesitan escribir en el nivel inferior. Es más sencilla de diseñar y de manejar. Su principal desventaja es que no produce buenos rendimientos de escritura puesto que tiene que estar realizando constantemente accesos a la memoria principal.
Una solución a este problema consiste en añadir un buffer de escritura que almacena el dato mientras que éste espera ser escrito en la memoria principal. Una vez escrito el dato en la caché y en el buffer, el procesador puede seguir la ejecución desentendiéndose de la relación entre el buffer y la memoria principal.
El buffer de escritura tiene un número fijo de palabras que suele oscilar entre 1 y 10. Si el buffer está lleno, cuando se realiza una escritura desde el procesador éste debe detenerse hasta que haya una posición vacía. Lógicamente si la frecuencia a que la memoria principal completa escrituras es inferior a la que las genera el procesador el buffer no sirve de nada.