Universidad Nebrija

Memoria Cache

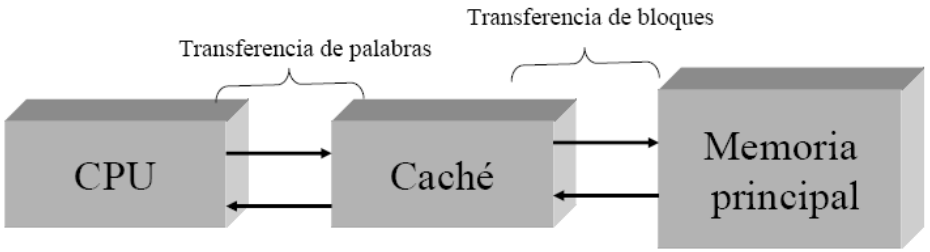
Departamento de Arquitectura de Computadores

Índice

- Introducción. Conceptos básicos
  - Características de los sistemas de memoria
  - Jerarquías de memoria
- Memoria Principal
  - Características físicas
  - Organización
- **Memoria Caché**
  - **Organización**
  - **Políticas de ubicación**
  - **Políticas de sustitución**
  - **Políticas de escritura**
  - **Políticas de búsqueda**
- Memoria Virtual

Memoria Caché

- El objetivo de la caché es lograr que la velocidad de la memoria sea lo más rápida posible.
- El esquema de la memoria que muestra como se integran ambas versiones de memoria es el siguiente:



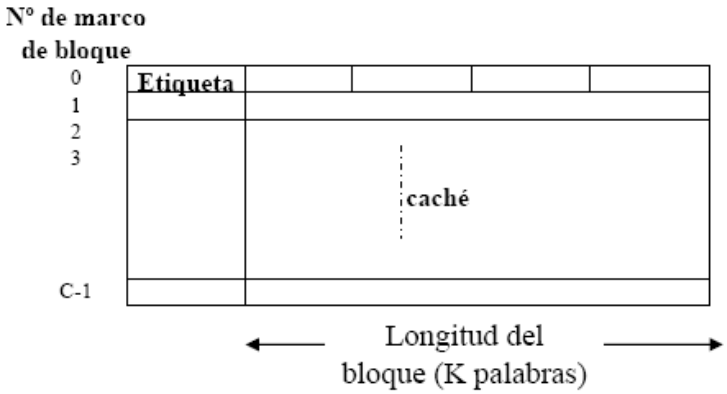
- Hay una memoria principal mas grande y relativamente mas lenta junto con una caché mas pequeña y más rápida. Esta caché tiene un copia de partes de la memoria principal. Cuando el procesador intenta leer una palabra de memoria, se comprueba si está en la caché.
  - Si dicha palabra está se entrega al procesador.
  - De lo contrario, un bloque de memoria principal, que contiene la palabra buscada, se transfiere a la caché, y más tarde la palabra es entregada al procesador.

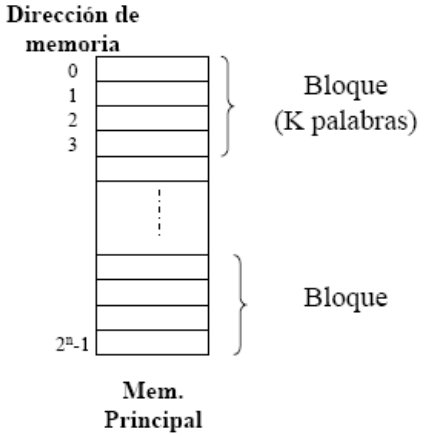
Estructura de la memoria caché/principal

- Memoria principal:
  - 2^n palabras direccionables
  - Está dividida en bloques de longitud fija (K palabras):

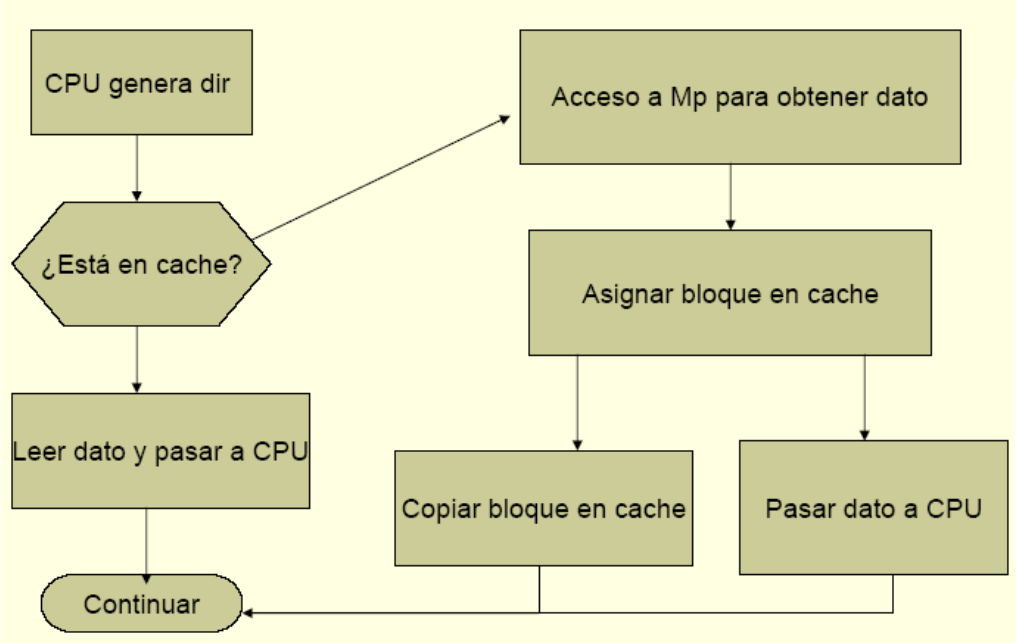
$$M = 2^n / K$$

- Memoria caché:
  - Consta de C líneas (marcos de bloque) de K palabras cada una.
  - El número de líneas es mucho menor que el número de bloques: C<<M.
  - Cada línea incluye una etiqueta que identifica el bloque de memoria principal que está almacenado en cada momento.





Operación de lectura de caché



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

donde:

$T_{acierto} \equiv$ Tiempo requerido para leer una palabra ubicada en Cache

$H \equiv$ Tasa de Fallos

$T_{penalización} \equiv$ Tiempo necesario para trasladar un bloque de MP a MC

Elementos de diseño de la caché

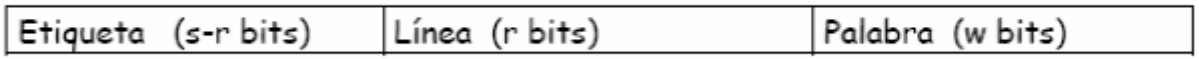
- **Organización:**
  - *Tamaño*
  - Tamaño de las líneas de caché
  - Número de cachés
- **Políticas de ubicación:** ya que hay menos líneas de caché que bloques de MP, se necesitan algoritmos que hagan corresponder bloques de memoria a líneas de caché: ***correspondencia directa, asociativa, asociativa por conjuntos***.
- **Políticas de sustitución:** cuando se introduce un nuevo bloque en la caché, debe sustituirse uno de los bloques existentes. En determinadas ocasiones habrá que determinar cual debe dejar de figurar en la caché.
- **Política de escritura:** antes de reemplazar un bloque que está en caché, es necesario comprobar si ha sido modificado en caché pero no en memoria principal. De ser así la MP debe actualizarse para mantener la coherencia.
- **Política de búsqueda:** se utilizan para decidir cuando se transfiere un bloque y qué bloque o bloques se van a transferir desde MP a caché.

Organización

- Tamaño: cuanto más grande se hace el sistema menor tasa de fallos pero por contra, más lentas y caras se manifiestan. El tamaño estándar suele estar entre 1K y 512K palabras.
- *Tamaño de la líneas*: a medida que aumenta el tamaño de bloque la tasa de aciertos primero aumenta debido al principio de localidad, pero más tarde decrecerá ya que cada palabra adicional estará más lejos de la palabra requerida, y por tanto es más improbable que sea necesaria a corto plazo. La relación entre tamaño de bloque y tasa de aciertos es compleja, dependiendo de las características de localidad de cada programa particular (entre 4 y 8 unidades direccionables).
- *Número de cachés*: con el aumento de densidad de integración (ley de Moore) ha sido posible tener una caché en el mismo chip del procesador: **caché on-chip**. Esto permite tener un sistema basado en dos niveles de caché:
  - Interna (L1): reduce el tiempo de acceso pues elimina el acceso al bus. Es pequeña ya que debe caber en la CPU. Suele utilizar emplazamiento directo.
  - Externa (L2): grande (2MB) y suele utilizar emplazamiento asociativo por conjuntos.
- Con respecto a su contenido suele separarse la cache en dos:
  - Cache de instrucciones (solo lectura)
  - Cache de datos (lectura/escritura)
  - Ventaja: duplica el ancho de banda.
  - Desventaja: ofrece mayor tasa de fallos que la cache unificada.

Políticas ubicación “Directa”

- La correspondencia directa es la técnica más simple y consiste en hacer corresponder cada bloque de memoria principal a sólo una línea posible de cache. La correspondencia se expresa como: **i = j módulo m**
  - i = número de línea de memoria cache.
  - j = número de bloque de memoria principal.
  - m = número de líneas de cache.



  - **Palabra**: codifica el nº de palabras de cada bloque de memoria.
  - **Línea**: codifica el número de línea de cache donde se realiza la búsqueda
  - **Etiqueta**: codifica el bloque de memoria asociado a esa línea de cache

Implementación de la función de correspondencia

- Se implementa fácilmente utilizando la dirección:
  - Desde el punto de vista de la MP la dirección física generada por la CPU está dividida en dos campos: *nº de bloque* y *palabra* dentro del bloque
    - Si la memoria principal se compone de 2^S bloques se necesitarán S bits para identificar el bloque.
    - Si cada bloque tiene 2^W palabras serán necesarios W bits para identificar la palabra dentro del bloque.

Nº Bloque (S bits)	Palabra (W bits)

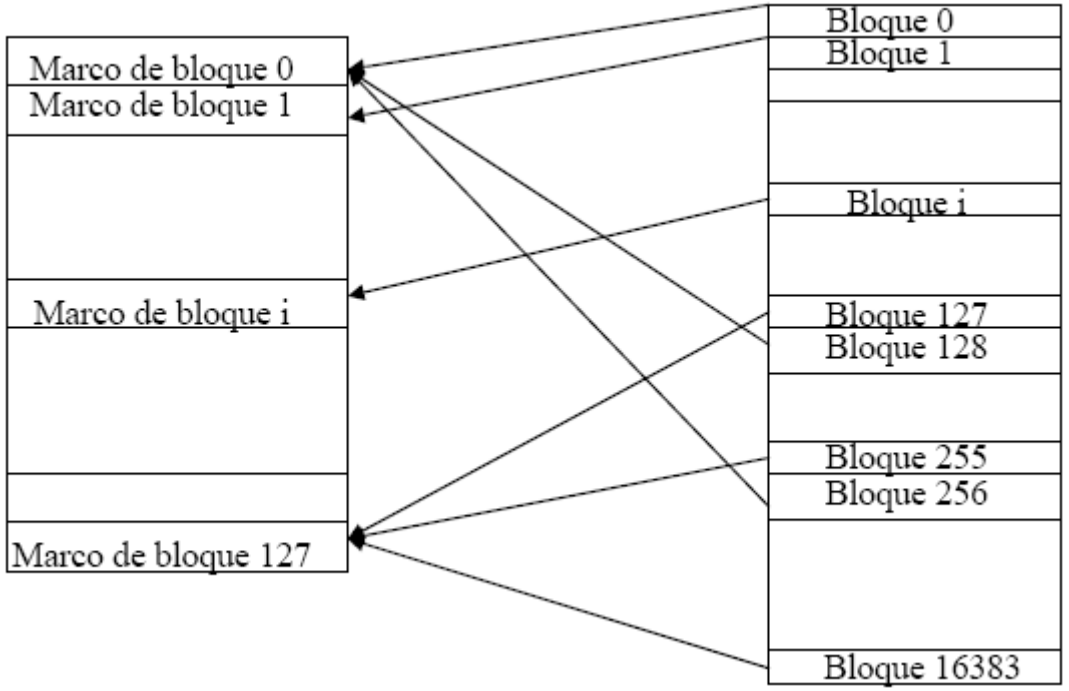
  - Desde el punto de vista de la memoria cache cada dirección se ve dividida en 3 campos: el campo de "*nº de bloque*" de S bits se divide a su vez en una *etiqueta* de S-r bits y un campo de *línea* de r bits.

Etiqueta (S-r)	Nº Línea (r bits)	Palabra (W bits)

  - *No hay dos bloques de MP a los que se les asigne la misma línea y que tengan el mismo número de etiqueta*

Ejemplo emplazamiento directo (I)

- Memoria cache: 2K palabras y líneas de 16 palabras
  - Líneas de cache =128
- Memoria principal: 256K palabras
  - Nº de bloques = 16384



Ejemplo emplazamiento directo (II)

- Continuación del ejemplo
  - MP

Nº Bloque (14 bits)	Palabra (4 bits)

  - Cache

Etiqueta (7 bits)	Nº Línea (7 bits)	Palabra (4 bits)

    - Comprobar que bloques que ocupan la misma línea de cache por ubicación directa se diferencian en la etiqueta

Palabras	Nº de bloque	Etiqueta	Marco de bloque
00000000000000XXXX	00000000000000=0	0000000	0000000
00000010000000XXXX	00000010000000=128	0000001	0000000
00000100000000XXXX	00000100000000=256	0000010	0000000
00000110000000XXXX	00000110000000=384	0000011	0000000
00001000000000XXXX	00001000000000=512	0000100	0000000
11111110000000XXXX	11111110000000=16256	1111111	0000000

Operación de lectura de caché

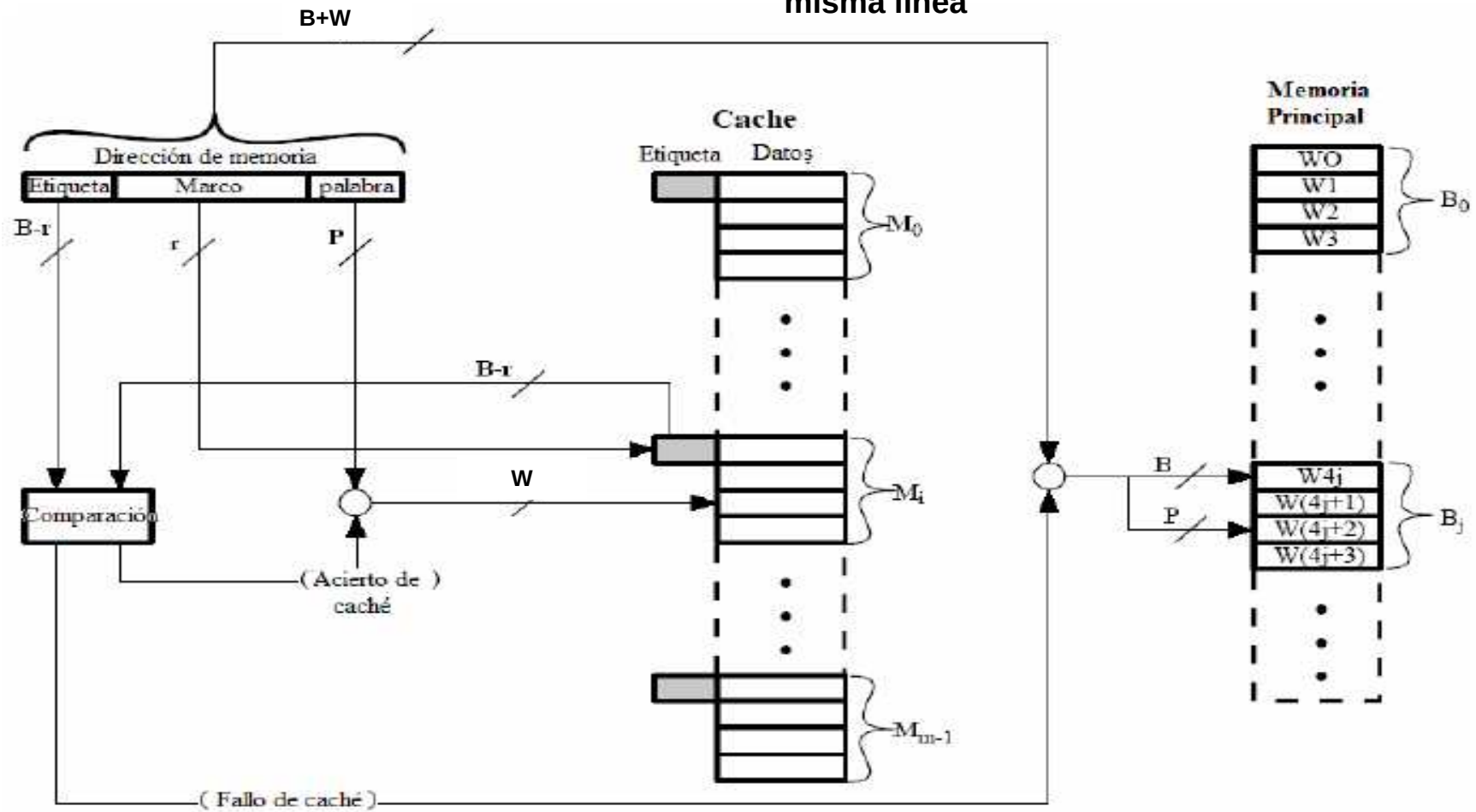
- Cuando la CPU referencia una palabra y proporciona su dirección:
  - Se localiza la línea de cache en la que debe estar situada.
  - Se compara el valor de la etiqueta almacenada en dicha línea con el de la palabra referenciada.
  - Si ambos valores coinciden, se selección la palabra utilizando los bits de palabras
  - Si no hay coincidencia, se produce **fallo de caché**. En este caso habrá que buscar el bloque en MP que contenga la palabra y llevarlo a memoria cache sustituyendo un bloque según la *política de sustitución* que se haya implementado.

- Ventajas:
  - Es simple y poco costosa de implementar.
- Desventaja:
  - Como consecuencia de haber una posición concreta de cache para cada bloque dado, si un programa referencia repetidas veces a palabras de dos bloques diferentes asignados en la misma línea, dichos bloques estarían alternándose continuamente en la cache haciendo que la **tasa de aciertos sea baja**.

Emplazamiento directo

Simple y poco costosa de implementar

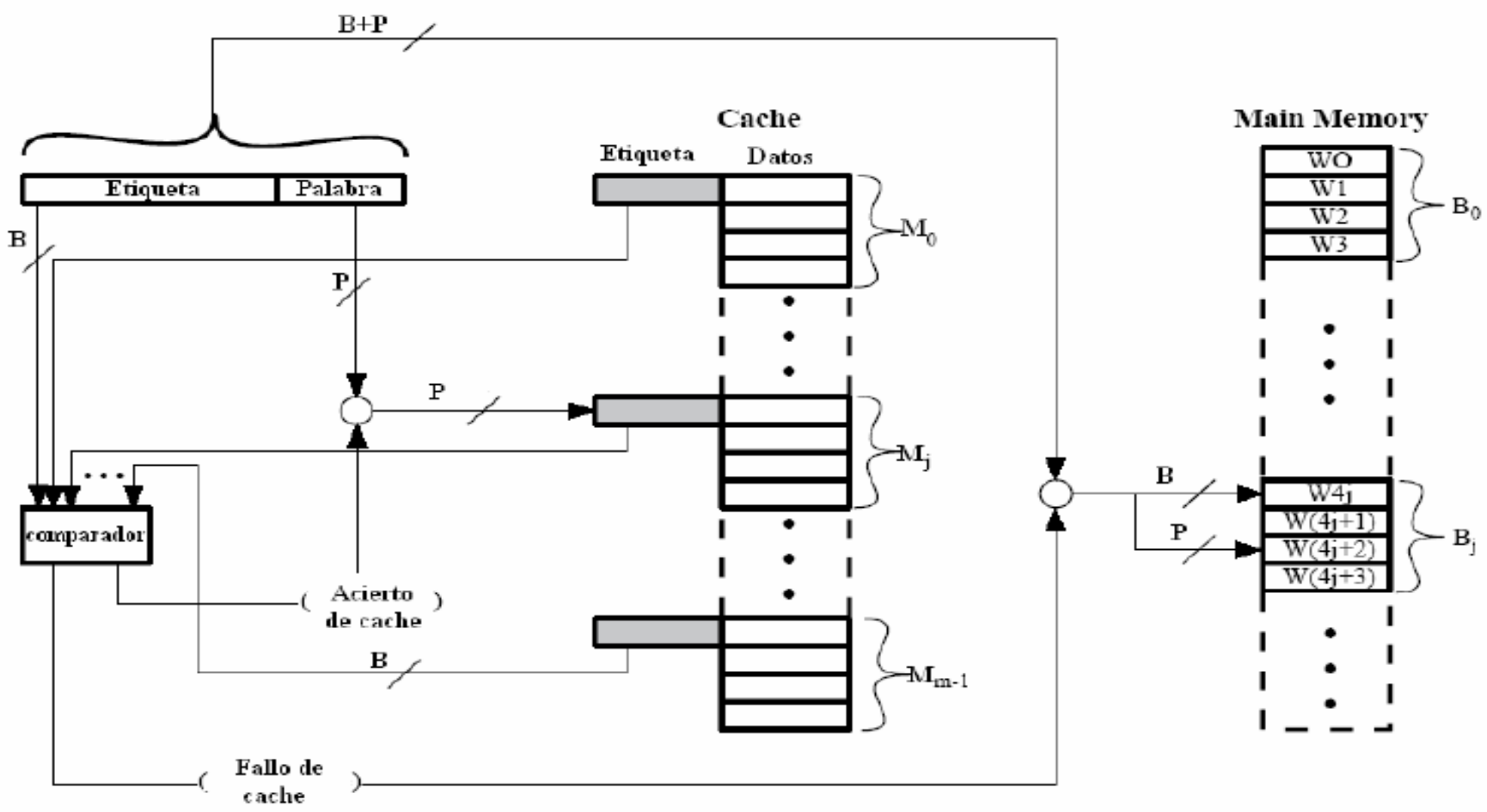
Alta tasa de fallos si dos bloques compiten por la misma línea



Políticas ubicación “Asociativa”

- Permite que cada bloque de memoria principal pueda cargarse en cualquier línea de cache evitando la principal desventaja de la correspondencia directa.
- En este casi la lógica de control de la cache interpreta una dirección de memoria simplemente como una *etiqueta* y un *campo de palabra*.
- Para determinar si el bloque está en cache, su lógica de control debe examinar simultáneamente todas las etiquetas de líneas para buscar una coincidencia. La etiqueta por tanto identifica unívocamente un bloque de memoria principal
- Ventajas:
  - Flexibilidad para que cualquier bloque pueda ser reemplazado por uno nuevo en la cache.
  - Alta tasa de aciertos.
- Desventajas:
  - Circuitería necesaria para examinar en paralelo las etiquetas de todas las líneas de caché es muy compleja.
    - Proceso de identificación lento
    - Alto coste

Organización de cache asociativa



Políticas ubicación "Asociativa por conjuntos"

- Es una solución de compromiso que recoge lo positivo de las correspondencias directa y asociativa, sin presentar sus desventajas.
- La cache se divide $C=2^d$ conjuntos cada uno con E líneas:

$$M = C * E \qquad i = j \bmod C$$

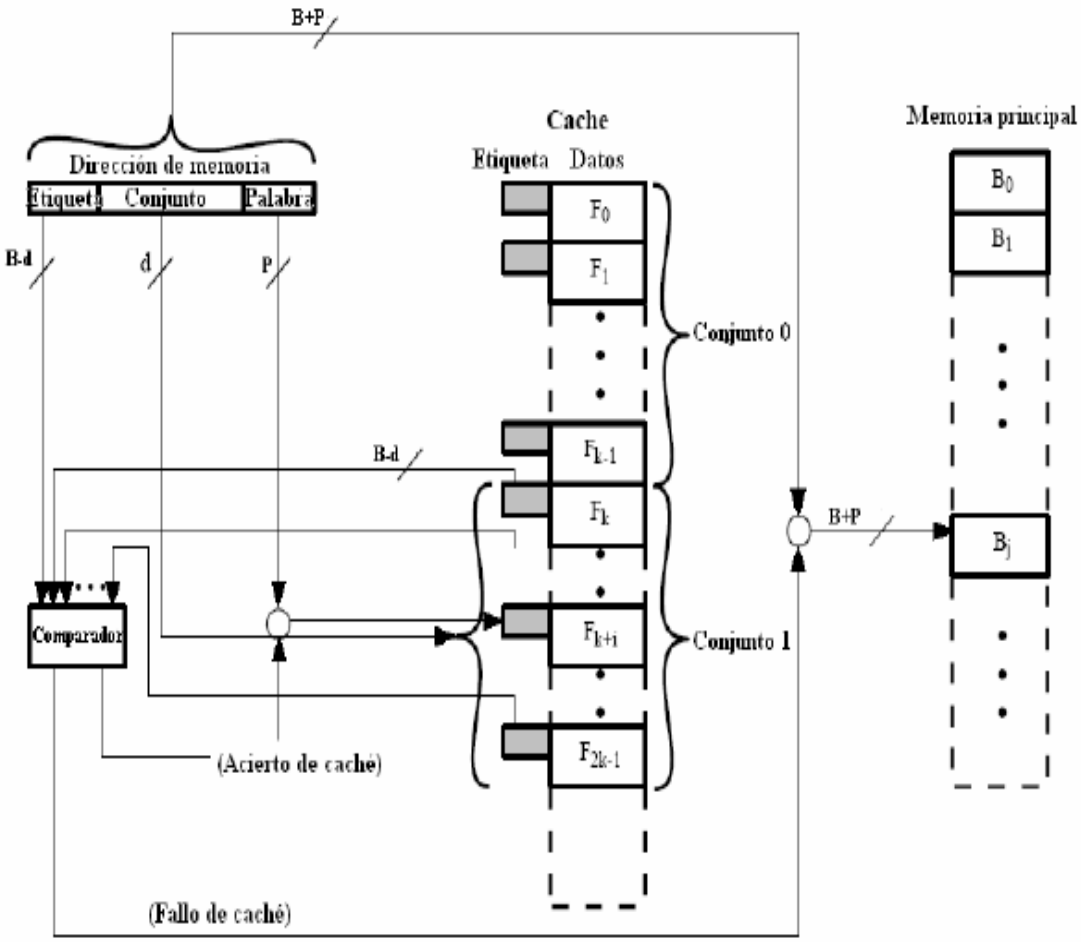
  - i = conjunto de caché en el que se ubica el bloque j de MP.
  - j = número de bloque de memoria principal.
  - C = número de conjuntos de la caché.
  - M = número total de líneas en la caché.
- En este caso, el bloque B_j puede ubicarse en cualquier de las E líneas del conjunto i.
- La lógica de control de la cache interpreta la dirección de memoria como tres campos: *etiqueta*, *conjunto* y *palabra*.
  - Los d bits de conjunto especifican uno de los $C = 2^d$ conjuntos.
  - Los B bits de los campos etiqueta y conjunto especifican uno de los 2^B bloques de MP.

Nº Bloque (B bits)	Palabra (P bits)

Etiqueta (B-d bits)	Conjunto (d bits)	Palabra (P bits)

Organización de cache asociativa por conjuntos

- En una operación de lectura, el número de conjunto se utiliza para determinar qué conjunto va a examinar.
- Las etiquetas de las líneas que integran este conjunto se comparan con la etiqueta de la dirección a la que se quiere acceder.



Emplazamiento asociativo por conjuntos

- El número de líneas por conjunto, E, se denomina número de vías o grado de asociatividad de la cache:
  - Si E=1, el emplazamiento es directo.
  - Si E=M (número de marcos) el emplazamiento es asociativo.

- Al aumentar el número de líneas:
  - Se reduce el número de fallos
  - Aumenta la complejidad
  - Grado óptimo entre 2 y 16 vías

- El caso más común es usar 2 vías. Con este diseño:
  - Se reduce el número de fallos respecto al emplazamiento directo.
  - El grado de complejidad del circuito comparador es muy bajo en contraposición con la técnica asociativa por conjuntos.

Políticas de sustitución

- Cuando se introduce un nuevo bloque en la cache, debe sustituirse uno de los bloques existentes (cuando la cache está llena).
  - En el caso de la correspondencia directa, sólo hay una posible línea para cada bloque particular, y no hay elección posible con lo que no se hace necesario un algoritmo de sustitución.
  - Para las técnicas asociativas es necesario un algoritmo de sustitución que nos señale que bloque de la cache debe ser eliminado.
- Algoritmos de reemplazamiento:
  - **LRU** (least-recently used): se sustituye el bloque que se ha mantenido en la cache por más tiempo sin haber sido referenciado.
  - **FIFO** (first-in first-out): se sustituye el que más tiempo lleva ubicado en la cache
  - **LFU** (least-frecuently used): se sustituye el que me menos referencias haya recibido. Suele ser el que mejores resultados ofrece.
  - **Aleatoria**: no está basado en el grado de utilización y consiste en coger un línea al azar entre los posibles candidatos.

Política de escritura

- Antes de que un bloque sea reemplazado de la cache, es necesario comprobar si ha sido alterado en cache pero no en memoria principal:
  - Si no lo ha sido puede escribirse sobre esta línea.
  - De lo contrario, la memoria principal debe ser actualizada de acuerdo a mantener el ***principio de coherencia***.
- Como políticas de escritura con distintos compromisos entre prestaciones y coste aparecen:
  - **Escritura inmediata o Write-Through**: todas las operaciones de escritura se realizan tanto en cache como en MP, asegurando que la memoria principal siempre es válido.
    - Ventaja: bajo coste y consistencia garantizada.
    - La desventaja es que genera un tráfico sustancial a memoria que puede originar un cuello de botella.
  - **Post-escritura o Write Back**: es una técnica alternativa que minimiza las escrituras en memoria haciendo las actualizaciones únicamente en cache. Cuando tiene lugar una actualización, se activa un bit ACTUALIZAR asociado a la línea. Mas tarde cuando es sustituido, es post escrito en MP si y solo si , el bit ACTUALIZAR está activo.
    - La desventaja principal es que cualquier módulo de E/S debe hacer su acceso a través de cache ya que puede haber bloques en memoria principal que no estén actualizados.

Política de búsqueda

- Dictan cuando transferir un bloque de MP a memoria cache:
  - **Búsqueda bajo demanda**: cuando se produce un fallo de cache, se busca el bloque deseado y se transfiere a cache.
  - **Búsqueda anticipada**: en este caso el bloque se lleva a cache antes de ser demandado. Se busca con esto reducir la tasa de fallos y la política normal es trasladar un bloque (i el referenciado) y el siguiente (i +1 principio de localidad).