Organización de una caché sencilla

Esta situación es análoga a necesitar un libro de las estanterías y no disponer de más espacio sobre la mesa; alguno de los libros de la mesa debe volver a las estanterías. En una cache de correspondencia directa sólo existe un lugar donde alojar los elementos recientemente solicitados y de ahí que sólo exista una única opción para decidir qué reemplazar.

Sabemos dónde buscar en la cache cada una de las posibles direcciones: los bits menos significativos de una dirección pueden ser usados para encontrar la única entrada de la cache que se corresponde con la dirección. La figura 5.7 muestra cómo una dirección solicitada se divide en

- una etiqueta que se usa para compararla con el valor de la etiqueta almacenado en la cache
- un índice de la cache, que se utiliza para seleccionar el bloque de datos

El índice de un bloque de cache junto con el contenido de la etiqueta para este bloque, determina con precisión la dirección de memoria de la palabra almacenada en el bloque de cache. Ya que el campo índice es utilizado como una dirección para acceder a la cache y dado que un campo de n bits codifica hasta 2^n valores distintos, el número total de entradas de una cache de correspondencia directa debe ser potencia de dos. En la arquitectura MIPS, dado que las palabras están alineadas en múltiplos de 4 bytes, los 2 bits menos significativos de cada dirección determinan uno de los bytes que constituyen una palabra, y por ello se ignoran cuando se accede a la palabra de un bloque.

El número total de bits que se requiere para construir una cache está en función de la capacidad de la cache y del tamaño de la dirección debido a que la cache incluye tanto el almacenamiento para los datos como para las etiquetas. El tamaño del bloque utilizado más arriba fue de una palabra, pero normalmente es de varias palabras.

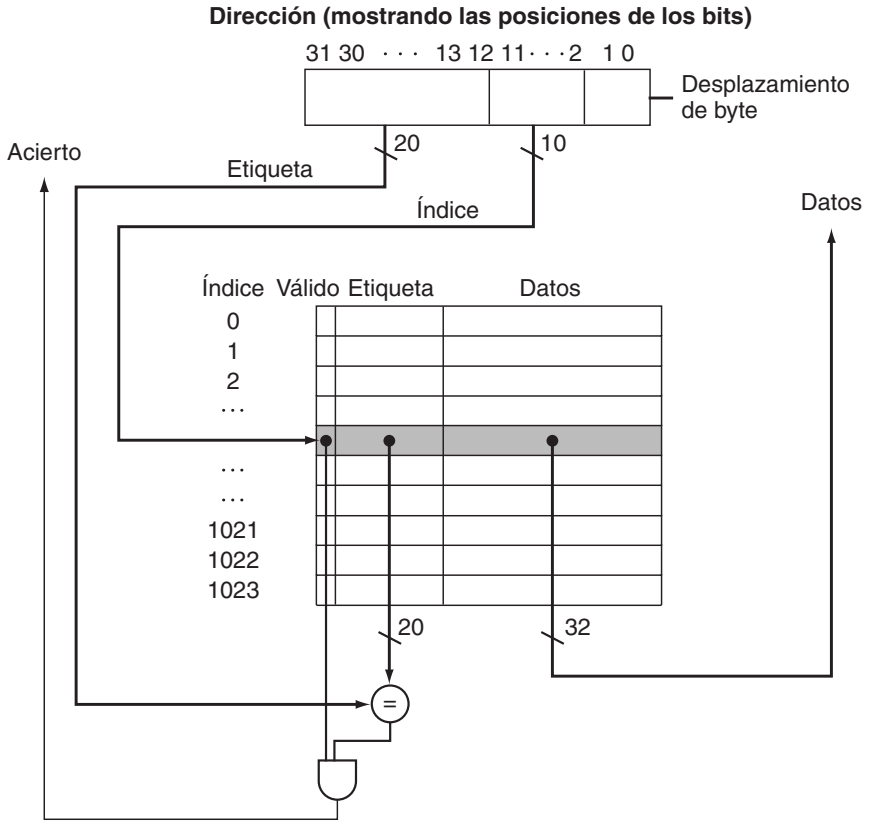

FIGURA 5.7 Para esta cache, la parte menos significativa de la dirección se utiliza para seleccionar una entrada de la cache que está formada por una palabra de datos y una etiqueta. Es una cache de 1024 palabras o 4 KB. En este capítulo supondremos que las direcciones son de 32 bits. La etiqueta almacenada en la cache se compara con la parte más significativa de la dirección para determinar si la entrada de la cache se corresponde con la dirección solicitada o no. Ya que la cache dispone de 2^{10} (o 1024) palabras y un tamaño de bloque de 1 palabra, se utilizan 10 bits para indexar la cache, dejando 32 – 10 – 2 = 20 bits para cada etiqueta. Si esta etiqueta y los 20 bits más significativos de la dirección coinciden y el bit de validez está activado, entonces la petición de memoria acierta en la cache, y la palabra es suministrada al procesador. En caso contrario, se produce un fallo.

Un ejemplo de cache: el procesador FastMATH de Intrinsity

El FastMATH de Intrinsity es un microprocesador empotrado rápido que utiliza la arquitectura MIPS y una implementación sencilla de la cache. Cerca del final del capítulo, examinaremos el diseño de cache más complejo del AMD Opteron X4 (Barcelona), pero por razones pedagógicas comenzaremos con este ejemplo que es simple pero real. La figura 5.9 muestra la organización de la cache de datos del FastMATH de Intrinsity.

Este procesador tiene un camino de datos segmentado de 12 etapas, similar al descrito en el capítulo 4. Cuando opera a velocidad pico, el procesador puede solicitar una palabra de instrucción y otra palabra de datos en cada ciclo de reloj. Para satisfacer las demandas del camino de datos segmentado sin que se produzcan paradas, las caches de instrucciones y de datos están separadas. Cada cache es de 16 KB, o 4K palabras, con bloques de 16 palabras.

Las solicitudes de lectura para la cache son simples. Debido a que existen caches de datos e instrucciones separadas, se necesitarán distintas señales de control para leer y escribir cada cache. (Recuerde que se necesita actualizar la cache de instrucciones cuando se produce un fallo). De este modo, los pasos que se siguen en cada cache para una solicitud de lectura son los siguientes:

1. Se envía la dirección a la cache apropiada. La dirección proviene, bien desde el PC (para una instrucción), bien de la ALU (para datos).

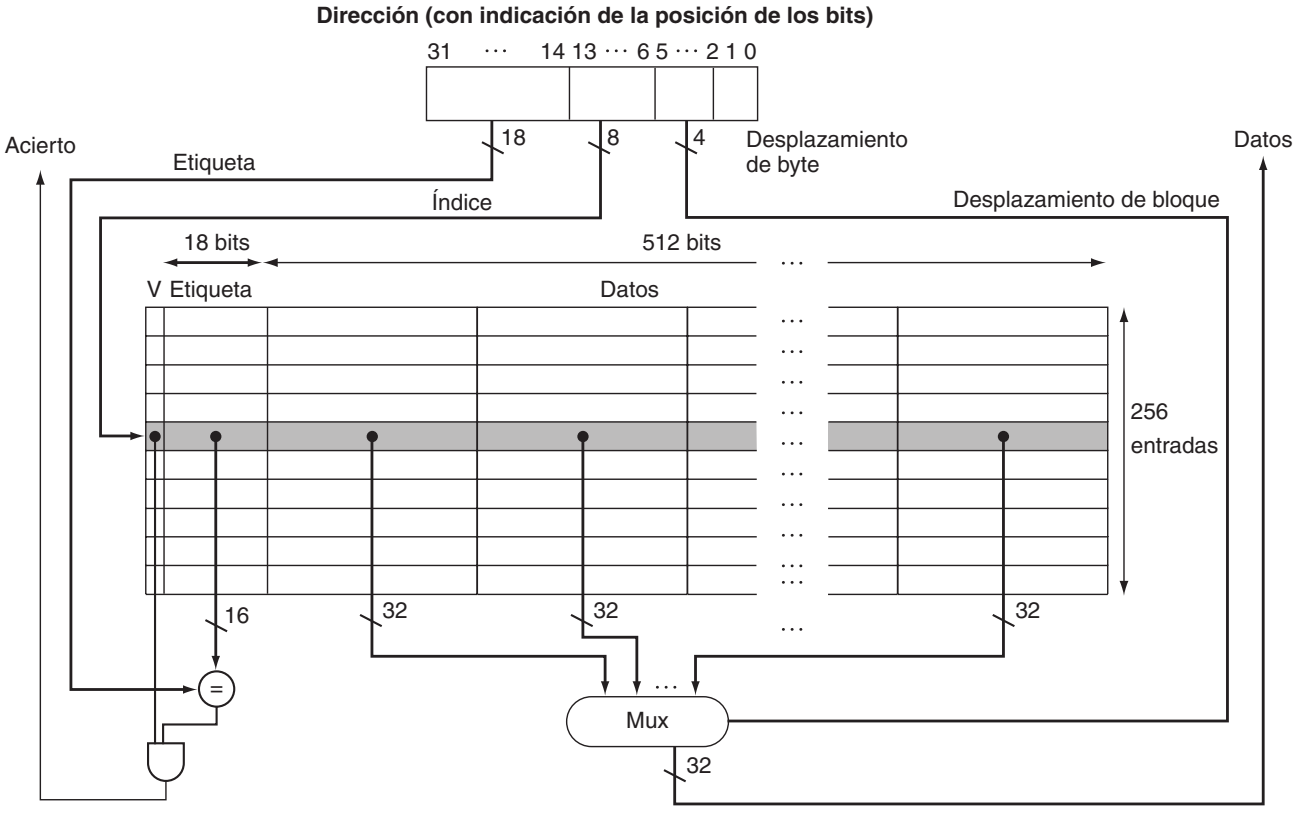

FIGURA 5.9 Las caches de 16 KB en el FastMATH de Intrinsity, cada una de ellas contiene 256 bloques con 16 palabras por bloque. El campo etiqueta ocupa 18 bits y el campo índice ocupa 8 bits, mientras que un campo de 4 bits (bits 5-2) se usa para indexar el bloque y seleccionar la palabra del bloque por medio de un multiplexor 16-a-1. En la práctica, para eliminar el multiplexor, las caches combinan una RAM con mayor rango de direccionamiento para los datos y una RAM con menor rango de direccionamiento para las etiquetas, con los bits adicionales de la dirección que determinan el desplazamiento dentro del bloque en la RAM de datos. En este caso, la RAM más grande tiene un ancho de palabra de 32 bits y debe disponer de un número de palabras que es 16 veces el número de bloques de la cache.

2. Si se acierta en la cache, la palabra solicitada se encuentra disponible en las líneas de datos. Ya que existen 16 palabras en el bloque deseado, necesitamos seleccionar la palabra que se pide. Un campo del índice del bloque se usa para controlar el multiplexor (que aparece en la parte inferior de la figura), el cual selecciona la palabra solicitada de entre las 16 palabras del bloque indexado.
3. Si se falla en la cache, la dirección se envía a la memoria principal. Cuando la memoria devuelve el dato, se escribe en la cache y luego éste se lee para completar la operación solicitada.

Para los almacenamientos, el FathMATH de Intrinsity ofrece tanto escrituras directas como escrituras retardadas, dejando al sistema operativo que decida qué estrategia usar para una aplicación. El microprocesador dispone de un búfer de escritura de una sola entrada.

¿Cuál es la frecuencia de fallos que se alcanza con una cache cuya estructura es como la que usa el FastMATH de Intrinsity? La figura 5.10 muestra las frecuencias de fallos de las caches de instrucciones y de datos. La frecuencia de fallos combinada es la frecuencia de fallos efectiva por acceso que muestra cada programa después de considerar las distintas frecuencias de accesos a instrucciones y datos.

Aunque la frecuencia de fallos es una característica importante del diseño de caches, la medida final reflejará el efecto del sistema de memoria sobre el tiempo de ejecución de los programas. Dentro de poco veremos cómo se relacionan la frecuencia de fallos y el tiempo de ejecución.

Cache separada: técnica en la cual un nivel de la jerarquía de memoria se compone de dos caches independientes que funcionan en paralelo, una de ellas destinada a manejar instrucciones y la otra a manejar datos.

Extensión: Una cache combinada con una capacidad total igual a la suma de dos **caches separadas** tendrá normalmente una mayor frecuencia de aciertos. Ellos se debe a que la cache combinada no diferencia estrictamente las entradas que pueden ser usadas por las instrucciones de las que son usadas por los datos. Aún así, muchos procesadores utilizan caches separadas para instrucciones y datos con el objetivo de aumentar el *ritmo de transferencia o ancho de banda* (*bandwidth*). (Puede haber, también, menos fallos de conflicto; véase sección 5.5.)

A continuación se dan las frecuencias de fallos para caches del tamaño que se encuentra en el procesador FastMATH de Intrinsity y para una cache combinada cuya capacidad es igual al total de las dos caches.

- Capacidad total de la cache: 32 KB
- Frecuencia de fallos efectiva de la cache separada: 3.24%
- Frecuencia de fallos de la cache combinada: 3.18%

La frecuencia de fallos de la cache separada resulta ser sólo ligeramente peor.

Frecuencia de fallos para las instrucciones	Frecuencia de fallos para los datos	Frecuencia de fallos combinada
0.4%	11.4%	3.2%

FIGURA 5.10 Frecuencias aproximadas de fallos para instrucciones y datos en el procesador FastMATH de Instrinsity que se obtienen con los programas de prueba SPEC2000. La frecuencia de fallos combinada es la frecuencia de fallos efectiva que experimenta una combinación formada por una cache de instrucciones de 16 KB y una cache datos de 16 KB. Se obtiene factorizando las frecuencias de fallos individuales por la frecuencia de accesos a instrucciones y datos.