Memoria Caché

Una de las limitaciones más importantes a la hora de diseñar computadores es la velocidad.

CPU → Frecuencia de Ck (clock) muy alta } Ejecuta un gran número de instrucciones por ciclo de CK

Memoria, buses de datos, periféricos, etc → Más lentos, la CPU debe esperar para tener disponibles los datos

Ej: Si un microprocesador trabaja a una frecuencia de 1Ghz (1×10^9 Hz)

$$\text{Frecuencia} = \frac{\text{Ciclos de Reloj}}{1\text{Seg}} \Rightarrow$$

1 seg . Frec = Ciclos de Reloj

1 seg . $1 .10^9$ ciclo/seg = Ciclos de Reloj

1.10^9 = Ciclos de Reloj

1 seg_____ 1.10^9 ciclos

X _____ 1 ciclo

$$\frac{1 \text{ ciclo} \cdot 1 \text{ seg}}{1.10^9 \text{ ciclos}} \Rightarrow$$

1.10^{-9} seg = 1ns

Tiempo de período procesador = 1ns (para una frecuencia de 1Ghz)

Si por otra parte la memoria DRAM, trabaja con un tiempo de acceso de 50ns, y de acuerdo con las etapas definidas para un procesador (Pentium) el tiempo total para ejecutar una instrucción sería:

Etapas

1-Fetch → acceso a memoria **50ns**

2- Decodificación de la instrucción → la hace la CPU **1ns**

3- Búsqueda de operandos → acceso a memoria para lectura **50 ns**

4- Ejecución de la instrucción → EU de la CPU **1ns**

5- Escritura de Resultados → Se almacenan en la memoria (se escribe en la misma)

Tiempo total = 152ns

De estos 152ns, el tiempo total de acceso a memoria es de 150ns y el tiempo total de la CPU es de 2ns

De estos resultados se desprende un notable desequilibrio (en el tiempo) entre etapas. Para obtener un óptimo rendimiento todas las etapas deberían durar lo mismo.

Una solución para el problema planteado es el uso de la memoria caché.

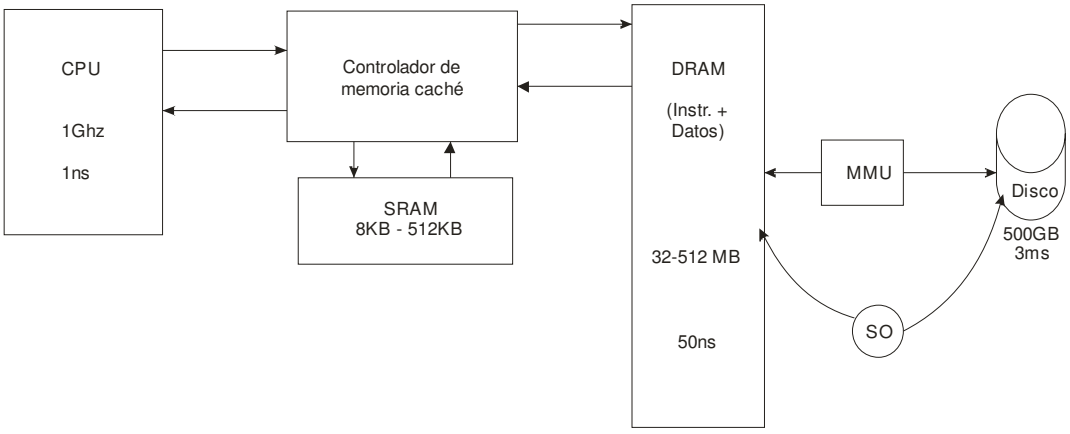
Son memorias ultrarrápidas pero muy caras, por lo que no reemplazan a la DRAM (tienen poca capacidad)

Se aplica la jerarquía de memoria que consiste en interponer una pequeña memoria ultrarrápida (caché) entre la CPU y la DRAM.

El procedimiento que se utiliza con la caché es similar al empleado con la MMU (Memory Managemet Unit) entre la Memoria Principal y la Memoria Virtual.

La CPU realiza una petición de información (datos o instrucciones) al controlador de caché, que se encarga de trasladar esa petición a la memoria caché. Si ésta contiene la información solicitada (lo indica con un bit de presencia que cuando está en "1" indica la presencia de la información solicitada) se produce un acierto, y entrega la información a la CPU. En caso de no encontrarse la información (bit de presencia = 0, es decir ausencia) se produce un fallo. Entonces se debe obtener la información de la Memoria Principal. Si tampoco se encuentra en la MP, la MMU activará el mecanismo para traerla desde la Memoria Virtual.

Jerarquías de Memoria



La petición a la MP puede darse, bien simultáneamente a la petición que se hace a la memoria caché (sin esperar a conocer el resultado) o bien después de esa petición y una vez que se conoce el resultado de la misma ha sido un fallo. Esto depende del tipo de conexionado de la caché e influye en los tiempos de retardo y en el rendimiento.

- ✓ Si la caché tiene el dato, sólo penaliza el tiempo de acceso a la misma. Sin embargo cuando la caché no dispone del dato solicitado, el tiempo empleado se incrementa, debido al acceso a la memoria principal.

- ✓ La memoria caché permite mejorar la productividad de los sistemas informáticos sin elevar el costo en forma significativa.

El movimiento de los datos se realiza de forma que, al producirse un fallo por ausencia, la memoria caché recibe de la MP no solo el dato pedido, sino también otros contiguos que previsiblemente va a pedir la CPU. De este modo, se consigue optimizar la transferencia de bloques, siempre que la programación cumpla las reglas clásicas de la vecindad espacial y temporal (simplemente poner a mano las cosas de uso más frecuente)

La localidad espacial se refiere a las direcciones físicas de memoria en que se alojan las instrucciones o datos. La localidad temporal se refiere al instante en que se van a necesitar esas instrucciones o datos. Es fácil prever la siguiente información que va a solicitar la CPU, debido a que tiende a requerir datos que estén en posiciones cercanas físicamente en instantes próximos en el tiempo, y a utilizar las mismas instrucciones repetidamente.

- ✓ Hay dos factores destacables en la memoria que proporciona la caché:
  - o Factor de Velocidad: Es la relación entre el tiempo de acceso a la MP y el tiempo de acceso a la Memoria Caché.

$$\gamma = Tp/Tc$$

  - o Factor de Eficacia: Depende en gran medida, del programa que se está ejecutando, es decir de cómo está escrito y estructurado el software.

$$Eficacia = Tc/T$$

Donde, Tp: Tiempo de acceso a la Memoria Principal.

Tc: Tiempo de acceso a la Caché.

T: Tiempo promedio de acceso.

Ejemplo: Si el tiempo de acceso a la MP es de 50 ns y el tiempo de acceso a la memoria caché es de 5 ns

El factor de velocidad se calcula como $\gamma = Tp/Tc = 50/5 = 10$

El factor de velocidad de la caché respecto a la MP es de 10 lo que significa que el tiempo de acceso a caché es 10 veces menor al de la MP.

Si además el tiempo medio de acceso, para un programa dado es de 20ns, ¿qué eficacia tiene la caché?

Eficacia = Tc/T = 5/20 = 0,25

El tiempo medio de acceso depende en gran medida, de los tiempos de acceso a las memorias empleadas y del tipo de interconexión, si emplea jerarquía de niveles, etc.

Fórmula para calcular el tiempo medio de acceso $\overline{T} = \alpha \cdot Tc + (1-\alpha) \cdot (Tc + Tp)$

Donde, Tc: Tiempo de acceso a Memoria Caché

Tp: Tiempo de acceso a MP

α: Tasa de acierto

Ejemplo: Si el Tc = 5 ns ; el Tp = 50 ns y la tasa de acierto (α) es del 90%

$\overline{T} = 0{,}9 \cdot 5\text{ ns} + (1 - 0{,}9) \cdot (5\text{ ns} + 50\text{ ns}) = 4{,}5\text{ ns} + 0{,}1 \cdot 55\text{ ns} =$ **10 ns**

El tiempo medio de acceso es de 10 ns

↑ Tiempo medio de acceso

Principio de Funcionamiento de la Caché:

- ✓ La memoria caché es de tipo SRAM (RAM estática), reside muy cerca de la CPU.
- ✓ Los tiempos de acceso a la caché se encuentran entre los 3 y los 10 ns aproximadamente.
- ✓ Está diseñada para proporcionar a la CPU los datos e instrucciones que se solicitan con más frecuencia.
- ✓ Se fundamenta en la regla "80/20" que establece que aproximadamente el 20% de todos los programas y datos en la computadora se utilizan el 80% del tiempo. Del mismo modo, el 80% restante de los datos e información se utiliza el 20% del tiempo.
- ✓ El proceso de gestión de memoria caché es transparente para la CPU. La única diferencia que debe encontrar una CPU con memoria caché respecto a una memoria RAM es la velocidad con que recibe los datos.

Componentes más importantes de la Caché

- Controlador de Caché: Encargado de gobernar cada uno de los elementos de los que consta la memoria caché y controla los movimientos de información entre los dispositivos.
- Directorio Caché o bloque de etiquetas, RAM-CAM: Es una memoria RAM de acceso por contenido, no por dirección. Contiene una lista de etiquetas que hacen referencia a las direcciones de la memoria principal, cuyos datos están almacenados en la memoria caché. En lugar de utilizar una dirección específica para acceder a una posición de la memoria caché, se emplea parte de la dirección referida a la MP. Este valor se compara con cada etiqueta. Si se produce una coincidencia, entonces el dato buscado está en la caché.
- Memoria Caché o bloque de datos asociados, (SRAM): Es una memoria pequeña y rápida usada para almacenar réplicas de instrucciones y datos.

  La información se agrupa en conjuntos de datos (líneas) Cada elemento de la caché está formado por una etiqueta y un conjunto de datos (líneas)

  Cuando se habla del tamaño de la memoria caché se refiere al tamaño de este componente, no de otros.

- Lógica de Control: Son comparadores de tantos bits como tenga la etiqueta.

Organización de la Memoria Caché: (Traducción de la dirección física)

Como el tamaño de la DRAM no coincide con el de la caché, sus respectivos espacios de direccionamiento son distintos, por lo tanto una dirección física deberá ser traducida o mapeada por el controlador, que comprueba que exista tal referencia en la caché.

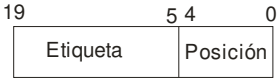
Clasificación según la organización:

- ✓ Totalmente asociativa.
- ✓ Asociativa de 1 vía (o de correspondencia directa)
- ✓ Asociativa de conjunto o de n vías

Mapeo Totalmente Asociativo:

En este caso a cada línea le corresponde una etiqueta.

Esquema de cómo se analiza la dirección física:



- Se la considera la mejor organización.
- Ante un fracaso la palabra se obtiene de la memoria principal, siguiendo dos caminos. Hacia la CPU y hacia la Caché (para agregarla a la misma)
- Se puede organizar como un anillo.

Desventaja: Debido a que hay una gran cantidad de etiquetas la lógica de comparación es cara.

Mapeo Asociativo de una Vía o de Correspondencia Directa:

- Es la organización interna menos utilizada actualmente.
- Es muy simple.

Ejemplo: Suponiendo una MP de 1MB y una Caché de 4 KB, la caché se puede organizar en: 256 líneas de 16 bytes cada una (256 L . 16 B = 4096 Bytes = 4KB)

A su vez, para representar las "megaposiciones" se necesitan 20 bits (1 Mega = 2^{20})

Representados en hexadecimal → XX YY Z ← Identifican el nro de Byte (desde 0 a 2^4-1)

Etiqueta — Identifican el nro de bloque o línea de caché, que equivale al nro de sector de RAM (desde 0 a 2^8-1)

Aclaraciones:

Z → sale de 16 Bytes = 2^4 tamaño en bits de Z

YY → sale de 256 líneas = 2^8 tamaño en bits de YY

XX → el resto de los bits de orden superior (del total de 20 bits) se asignan a la etiqueta.

De este modo hay una etiqueta por número sector RAM o línea caché (en el ejemplo son 256 etiquetas)

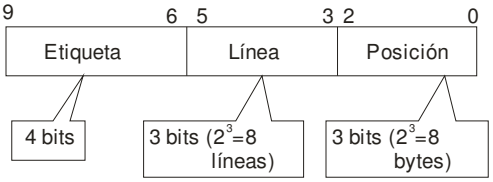
YYZ se considera como la dirección de acceso (para el ejemplo 12 bits 2^{12}= 4K) La información buscada se encuentra en la caché si para YYZ la etiqueta asociada coincide con XX.

Ejemplo 2: RAM de 1K y una caché de o líneas o bloques de o Bytes cada uno (es decir 64 Bytes)

RAM de 1K (2^{10}) → que se divide en 8 líneas (que es lo que tiene la caché)

$8 = 2^3 \Rightarrow 2^{10}/2^3 = 2^{10} \cdot 2^{-3} = 2^{10-3} = 2^7 = 128$ Grupos

Para representar las "Kilo posiciones" se necesitan 10 bits distribuidos de la siguiente manera:



Aclaración:

Línea y posición identifican a la cantidad de líneas y de bytes respectivamente. En este ejemplo se dijo que hay 8 líneas de 8 bytes cada una.

La dirección anterior se armó de la siguiente manera

Teniendo en cuenta que se debe establecer una correspondencia entre la dirección física de la RAM u la dirección de la caché, en donde:

- La dirección se forma con un total de 10 bits (son 10 bits porque se debe poder representar cualquier dirección de la RAM, que en este ejemplo es de 1KB y como 1K = 1024 = 2^{10} Entonces se necesitan 10 bits para representar las direcciones desde 0 a 2^{10})
- Los diez bits calculados en el punto anterior se dividen de la siguiente manera:
  - Los 3 bits de orden inferior (bits 0, 1 y 2) representan a la posición (son 3 bits porque la caché tiene 8 líneas de 8 bytes cada una. 8 bytes = 2^3 (por lo que se necesitan 3 bits para representar al nro byte)
  - Los tres bits de orden inferior siguientes (bits 3, 4 y 5) representan a la línea de caché (que en este ejemplo son 8 líneas)
  - Los 4 bits restantes (que son los de orden superior, es decir los bits 6, 7, 8 y 9) se asignan a la etiqueta

Equivalencia con la dirección física de RAM

- ✓ Los tres bits correspondientes a "línea" identifican a la línea de la caché y al sector de la RAM. Ejemplo si los 3 bits tienen valor 000 identifican a la línea 0 de la caché y al sector 0 de la RAM.
- ✓ Los 3 bits correspondientes a "posición" identifican al número de byte dentro de la caché (para este ejemplo van desde 0 a 7) Pero este valor no identifica al número de grupo o bloque de la RAM. Para calcular el número de grupo o sector de la RAM se debe hacer lo siguiente:
  - Como se dijo anteriormente en este ejemplo la RAM tiene un tamaño de 1KB. Este 1 KB se debe dividir entre las líneas de caché que son 8. Por lo tanto:
    RAM → 1KB = 2^{10}
    Caché → 8 líneas = 2^3
    Divido $2^{10}/2^3 = 2^{10}. 2^{-3} = 2^7$ = 128 bloques

    Por lo tanto la RAM dividida en las 8 líneas de caché da un total de 128 bloques. Como $128 = 2^7$ se necesitan 7 bits (de orden superior) para identificar al nro de grupo o bloque de la RAM.

    Por ejemplo para la dirección: etiqueta 1001 | línea 000 | byte 111

    Identifican al nro de grupo o bloque de RAM.

    Pasados a hexadecimal: 4 8 → 0100 1000 = 48h (nro de bloque de la RAM)

Al existir esta correspondencia entre las direcciones físicas de la RAM y la caché, se pude saber si un dato que se desea acceder está guardado o no en la caché.
Dada una dirección, basta con verificar si para el nro de línea de la misma, la etiqueta coincide con la de la caché. Si coincide entonces hay un acierto (el dato está en la caché) Si no coincide hay un fallo, y se debe acceder a la RAM a buscar el dato. Al mismo tiempo este dato accedido se almacena en la caché (en la línea correspondiente) Es decir se actualiza la memoria de datos. Y se actualiza la etiqueta (para esa línea) Es decir que se actualiza la memoria de etiquetas también.

Ejemplo: Se quiere acceder al dato correspondiente a la dirección 1F2h. Se desea saber si el dato está guardado o no en la caché, y en caso de que no lo esté actualizar la caché para que lo contenga.

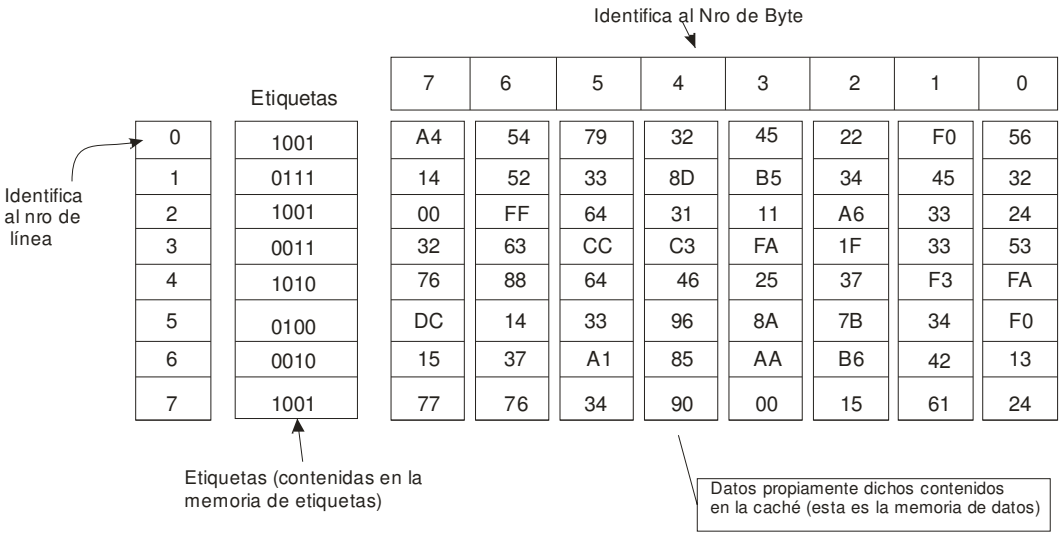
- ✓ En primer lugar se pasa la dirección dada en hexadecimal a binario:
  0001 1111 0010
  1 F 2

- ✓ Segundo, en el paso anterior se obtuvieron 12 bits, pero en este ejemplo las direcciones de caché son de 10 bits. Por lo tanto se toman los 10 bits de orden inferior, es decir: 01 1111 0010
- ✓ Tercero, se mapean los 10 bits obtenidos en el punto anterior, al formato de dirección de la caché de la siguiente forma:

0111 (Etiqueta, 4 bits) 110 (Línea, 3 bits) 010 (Byte, 3bits)

- ✓ Cuarto, se procede a buscar en la caché el dato deseado. Para esto se mira el valor correspondiente a la línea obtenida en el paso anterior (110 = 6) Para esta línea el valor de la etiqueta debe coincidir con el de la etiqueta de la caché (para que exista un acierto) Supóngase para ello la siguiente caché

Memoria caché de ejemplo:



En esta caché de ejemplo para la línea 6, la etiqueta es 0010 que no coincide con la etiqueta de la dirección (que es 0111)

En este caso hubo un fallo y se debe acceder a la RAM a buscar el dato solicitado y a su vez actualizar la caché.

- El valor actual de la caché en la línea 6 es:

6 (Línea) 0010 (Etiqueta) 010 (Byte) → valor actual: B6

- Después de la sustitución por el valor leído de la RAM (suponiendo que este sea A1) quedaría así:

6 (Línea) 0111 (Etiqueta) 010 (Byte) → valor actualizado: A1

- Es decir que para la línea 6 se actualiza la etiqueta y el valor contenido en la memoria para el byte 2.

Ejemplo 2: Ídem ejemplo anterior, pero para la dirección 3CD. Aplico los pasos vistos en el ejemplo anterior:

- ✓ En primer lugar se pasa la dirección dada en hexadecimal a binario:

  0011 1100 1101
  3 C D

- ✓ Segundo, se toman los 10 bits de orden inferior, es decir: 11 1100 1101
- ✓ Tercero, se mapean los 10 bits obtenidos en el punto anterior, al formato de dirección de la caché de la siguiente forma:

  1111 (Etiqueta 4 bits) 001 (Línea 3 bits) 101 (Byte 3bits)

- ✓ Cuarto, se procede a buscar en la caché el dato deseado. Para esto se mira el valor correspondiente a la línea obtenida en el paso anterior (001 = 1) Para esta línea el valor de la etiqueta debe ser 1111 para que exista un acierto (mirar gráfico de caché de ejemplo dado en el ejemplo anterior)

  - El valor actual de la caché en la línea 1 es:

  1 (Línea) 0111 (Etiqueta) 101 (Byte) → valor actual: 33

  - Pero como la etiqueta buscada es la 1111, hubo un fallo y se debe actualizar la caché con el valor que se leerá de la RAM
  - Después de la sustitución por el valor leído de la RAM (suponiendo que este sea B2) la línea 1 de la caché queda de este modo:

  1 (Línea) 1111 (Etiqueta) 101 (Byte) → valor actualizado: B2

Mapeo Asociativo de n Vías o n Conjuntos:

- Es similar al mapeo directo, pero cada línea admite n etiquetas o matrices de datos. Esto implica una caché n veces más grande y menos posibilidad de fallos (al buscar un dato en caché)
- Las n etiquetas y las n matrices de una misma línea constituyen un conjunto.

- Para buscar un dato en caché, dad una dirección, primero se busca la línea correspondiente, y luego se verifica la etiqueta. Que en este caso serán n etiquetas para una misma línea.

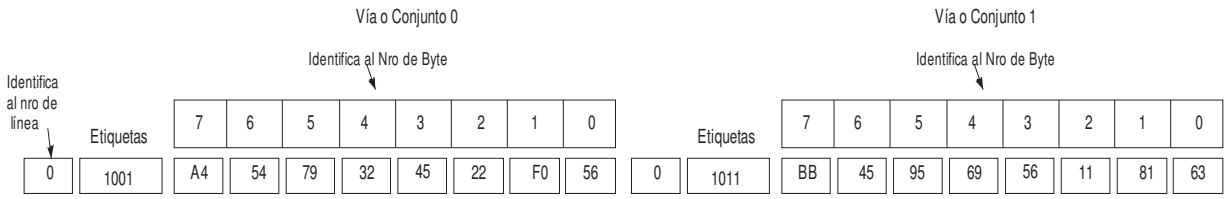
  Ejemplo: Para n=2

  - ✓ Con respecto al mapeo directo, en este caso (donde se sigue tomando una caché de 8 líneas con 8 bytes por cada una de ellas) Se tiene que se duplica (porque n=2) la cantidad de información que la caché guarda por cada línea (En el ejemplo anterior eran 64 bytes por matriz y aquí son 2 matrices de 64 bytes lo que totaliza 128 bytes de información)

    Memoria caché de 2 vías de ejemplo:

(Ver gráfico en página 12)

Para la línea 0 de dicha caché se tiene



¿Cómo se arman las direcciones en esta caché?

Para la línea 0

(Qué es la del ejemplo)

Vía 0: 1001 000 XXX
Etiqueta Línea Byte (desde 0 a 7)

Vía 1: 1011 000 XXX
Etiqueta Línea Byte (desde 0 a 7)

El número de línea, también identifica al número de sector de la RAM (como en el mapeo directo o de una vía) Aquí se comprueba que se guarda el doble de información de cada sector.

Actualización de Caché (ver unidad 9 del libro)

Políticas de sustitución: Cuando hay un fallo, es decir la palabra buscada no está en la caché se la debe incorporar a la misma. Para esto existen diferentes métodos para determinar el elemento que se reemplazará con el nuevo valor a incorporar.

- LRU: (Least Recently Used) se reemplaza la palabra de uso menos reciente.
- FIFO: (First In First Out) se reemplaza la palabra que se escribió primero.
- RND: (Random) se reemplaza una palabra en forma aleatoria.

Niveles de Caché (ver unidad 9 del libro)

- ✓ Caché de nivel 1, caché L1, caché primaria: Es una memoria integrada a la CPU. Es la memoria más rápida de un ordenador, suele funcionar a la misma velocidad que el microprocesador y es de tamaño pequeño.
- ✓ Caché de nivel 2, caché L2, caché secundaria: Es ligeramente más lenta que la caché L1, y de mayor tamaño. Se encarga, generalmente de almacenar aquellos datos e instrucciones muy usados recientemente pero que no han sido guardados por la caché L1. Se la puede encontrar en la placa base o integrada en la CPU. Aún en este caso se la considera diferente a la L1 de la que permanece separada.
- ✓ Caché de nivel 3, caché L3: como muchos microprocesadores integran parte de la caché L2 en la CPU, a la parte que permanece en la placa base a menudo se la llama así. En la actualidad en el Itanium el nivel llega a 3.

Memoria caché de 2 vías de ejemplo: