MEMORIAS CACHE

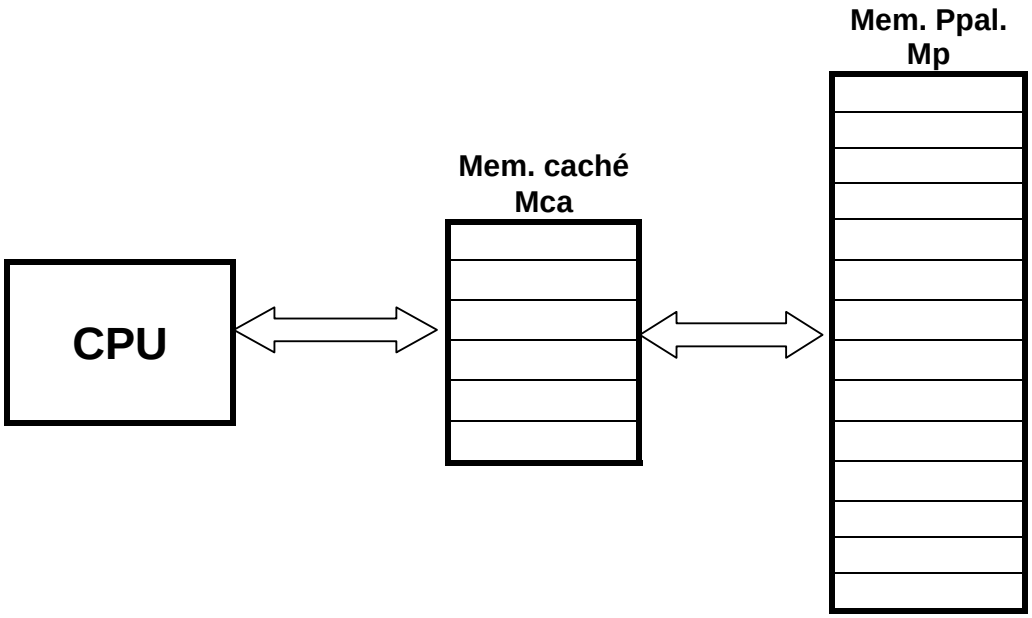
Memoria caché = memoria de tamaño pequeño y acceso rápido situada entre la CPU y la memoria principal.

Tiempo ciclo memoria > tiempo de ciclo del procesador ⇒ la CPU debe esperar a la memoria en los accesos a ésta

Mem. Caché acelera el proceso

Conceptos:

Principios de localidad de referencia { – **Espacial** – **Temporal**



Bloque = Cantidad mínima de información que puede estar presente o no en Mp y Mca
Acierto = Cuando el dato solicitado por la CPU está en la Mca
Fallo = Cuando el dato solicitado por la CPU no está en la Mca

Organización:

Mp ⇒ n bits en el bus de dir ⇒ 2^n palabras

M bloques de k palabras por bloque ⇒ $M = \frac{2^n}{k}$

Mca ⇒ c bloques de k palabras cada uno y en cada dirección una etiqueta indicativa de la dirección

M >> c

Criterios de diseño	- Capacidad	1K, 4K, 16K, 32K
	- Organización	Directa Totalmente asociativa Asociativa por conjuntos
	- Mecanismo de búsqueda	Por demanda Con anticipación Selectiva
	- Algoritmo reemplazamiento	Utilizada menos frecuentemente (LRU) Más antigua (FIFO) Utilizada menos frecuentemente (LFU) Aleatorio
	- Estrategia escritura	Escritura inmediata Post-escritura Escritura única
	- Tamaño de bloque	4, 8, 16, 32.. palabras
	- Número de cachés	Número de niveles

Rendimiento de una memoria caché:

Tasa de acierto $h = \dfrac{aciertos}{aciertos + fallos} = \dfrac{aciertos}{accesos} \Rightarrow h \geq 0{,}9$ $\Rightarrow$ ***principio de localidad***

Tasa de fallos = 1-h

Tiempo acceso medio $ta = h \times tca + (1-h) \times tp$

tca= t. Acces. Medio a Mca
tp = t. Acces. Medio a Mp

(tau) $\tau = \dfrac{tca}{tp} \Rightarrow 0{,}1 \leq \tau \leq 0{,}5$

lambda = índice de mejora = $\lambda = \dfrac{tp}{ta} = \dfrac{1}{1 - h \times (1-\tau)}$

Capacidad de la mem. Caché

Tamaño grande $\Rightarrow$ { Lógica más compleja; Más lenta; Mayor espacio físico }

Tamaño pequeño $\Rightarrow$ Mayor cantidad de fallos

→ Equilibrio $\Rightarrow$ 1K ÷512K

2002 Septiembre

2.- En una memoria caché en la que se realizan 10^{20} accesos se producen 10^{18} fallos. La tasa de aciertos correspondiente es:

A) 75% B) 99% C) 95% D) Ninguna de las anteriores

2002 Junio - 2ª semana

5.- Un sistema jerárquico de memoria tiene una memoria caché de 256 palabras, dividida en particiones de 8 palabras y con un tiempo de acceso de 20 nseg, y una memoria principal de 1024 Kpalabras con un tiempo de acceso de 200 nseg. Cuando se produce un fallo, primero se mueve el bloque completo a la memoria caché y después se lee el dato desde la caché. Si la tasa de acierto de la caché es del 90%, ¿cuál es el tiempo de acceso medio de este sistema?

A) 178 nseg B) 180 nseg C) 220 nseg D) Ninguna de las anteriores

2002 Junio - 1ª semana

8.- Un sistema jerárquico de memoria está compuesto por una memoria caché de 128 palabras, dividida en particiones de 16 palabras y con un tiempo de acceso de 10 μs, y por una memoria principal de 1024 Kpalabras con un tiempo de acceso de 200 μseg. Cuando se produce un fallo, ***primero se mueve el bloque completo a la memoria caché y después se lee el dato*** desde la caché. La política de ubicación y reemplazamiento tarda 20 μseg por término medio. Si la tasa de acierto es del 99%, decir si las siguientes afirmaciones son ciertas:

I. El tiempo de acceso medio es de 42,2 μseg.
II. Este sistema jerárquico de memoria es entre 4 y 5 veces más rápido que la memoria principal.

A) I: sí, II: sí. B) I: sí, II: no. C) I: no, II: sí. D) I: no, II: no.

Test 1ª semana de Junio de 2000

7.- Un sistema jerárquico de memoria está compuesto por una memoria caché de 256 palabras, dividida en particiones de 8 palabras y con un tiempo de acceso de 10 nseg, y por una memoria principal de 1024 Kpalabras con un tiempo de acceso de 100 nseg. Cuando se produce un fallo, se mueve el dato a la UCP y, ***simultáneamente, se mueve el bloque a la memoria caché***. Si la tasa de acierto de la caché es del 90%, ¿cuál será el tiempo de acceso medio del conjunto?

A) 20 nseg B) 19 nseg C) 89 nseg D) Ninguna de las anteriores

ORGANIZACIÓN DE LA MEMORIA CACHÉ

Establecer la función de correspondencia que asigna a los bloques de la memoria principal en las posiciones definidas en la memoria caché

Técnicas { Directa
Totalmente asociativa
Asociativa por conjuntos

Parámetros del ejemplo a utilizar en las descripciones:

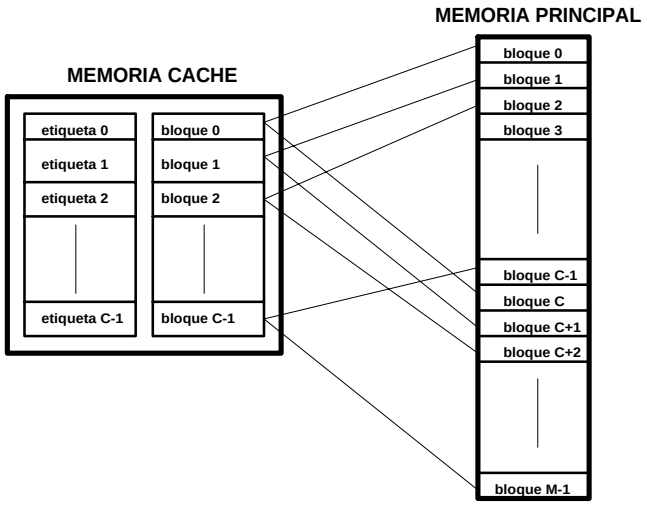
a) Ancho de palabra de datos 16 bits
b) Memoria caché 512 B = 2^9 Bytes
c) Tamaño de bloque k = 8
d) Memoria principal = 32 KB

Consecuencias:

- 32 KB = 2^{15} $\Rightarrow$ Bus de direcciones = 15 bits $\Rightarrow$ A0 a A14
- 512B = 2^9 $\Rightarrow$ Bus direcciones de la caché $\Rightarrow$ 9 bits
- 512B y k = 8 $\Rightarrow$ Nº bloques = 512/8 = 64 bloques
- k = 8 $\Rightarrow$ 2^3 $\Rightarrow$ 3 bits
- 64 bloques $\Rightarrow$ 2^6 $\Rightarrow$ 6 bits

CORRESPONDENCIA DIRECTA

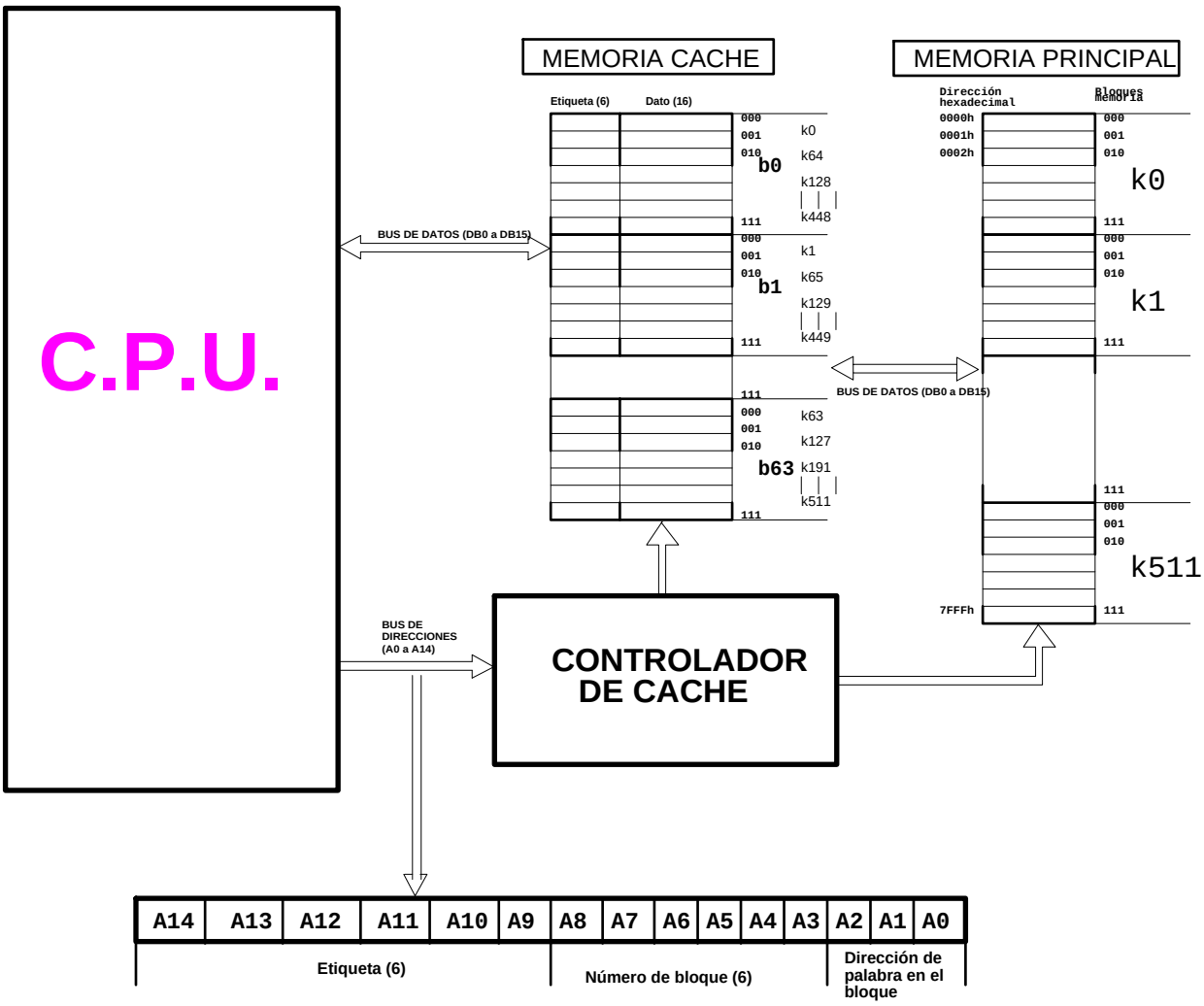
Cada bloque de la memoria principal tiene su posición en la caché y ***SIEMPRE*** en el mismo sitio



Ventajas { Simple
Económica

Inconveniente: Cada bloque tiene asignada una posición fija en la memoria caché $\Rightarrow$ ante continuas referencias a palabras de dos bloques con la misma localización en caché, continuos fallos habiendo sitio libre en la caché

Esquema de la caché



Funcionamiento:

Principios:

- Palabras por bloque 8 ⇒ 3 bits (b0 a b2)
- Nº bloques 64 ⇒ 6 bits
- Etiqueta = (bus direcc) – (bits de bloques) -- (bits palab/bloque) ⇒ 15 – 6 – 3 = 6
- Ancho de la mem. caché = Ancho palabra + ancho etiqueta ⇒ 16 + 6 = 22 bits

1. La CPU entrega la dirección de mem. al controlador de caché.
2. El ctrl. de caché aísla de los bits b3 a b8 y con ellos apunta al bloque correspondiente.
3. Comprueba si la etiqueta en caché = bits correspondientes a etiqueta en bus de direcciones (b9 a b14)
   - 3.a Si iguales ⇒ acierto, con lo que coge de los bits b15 a b0 (dato) de la caché y los saca al bus de datos de la CPU.
   - 4.a Fin de acceso.
   - 4.b Si distintos ⇒ Fallo
   - 5.b El ctrl. de caché transmite bloque completo desde mem. ppal. a mem. caché
     - 5.b.1 El ctrl. Caché pone a 000 los bits de dir. de palabra (b2 a b0).
     - 5.b.2 El ctrl. Caché apunta al bloque indicado por bus dir. (b3 a b8).
     - 5.b.3 El ctrl. Caché lleva de la dir. b14 b13 b12 .....b3 0 0 0 de mem. ppal. a la caché al bloque formado por los bits b8 a b3 en la dir. de palabra del bloque 0 0 0 (palabra 000 del bloque). Incrementa un contador y lleva de la dir. b14 b13 b12 .....b3 0 0 1 de mem. ppal. a la caché al bloque formado por los bits b8 a b3 en la dir. de palabra del bloque 0 0 1 (palabra 001 del bloque). Así sucesivamente hasta la palabra b14 b13 b12 .....b3 1 1 1 (palabra 7 del bloque) . Escribiendo en la zona de la etiqueta el valor de los bits b14 a b9.
     - 5.b.4 Una vez ha cargado el bloque completo (8 palabras) ⇒ Hay acierto y se salta al paso 4-a.

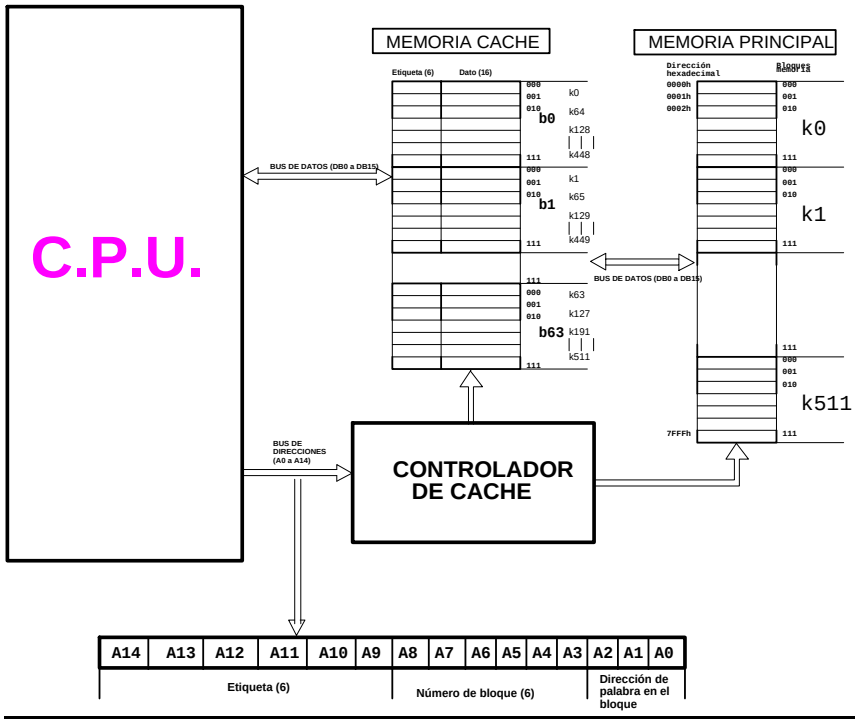
Ejemplo con los parámetros indicados

- Se supone que todavía no ha habido ningún acceso ⇒ Mem. caché vacía ⇒ todo con 0
- Se quiere acceder al contenido de la posición mem. $025F_H$

$025F_H$ =

b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
0	0	0	0	0	1	0	0	1	0	1	1	1	1	1
etiqueta						**Nº bloque**						**Dir. palabra**		

Bits dir. palabra = 3	Bit bloque = 6	Bits etiqueta = 6	Ancho palabra = 16



- El ctrl. de la caché mira si la etiqueta del bloque:

0	0	1	0	1	1

Es:

0	0	0	0	0	1

Como es el primer acceso la mem. caché contiene todo 0 ⇒ hay primer fallo.

- El ctrl. de caché va a direccionar a la mem. ppal. a la dirección

0	0	0	0	0	1	0	0	1	0	1	1	0	0	0

y llevará el contenido al bloque

0	0	1	0	1	1

dirección

0	0	0

escribiendo la etiqueta

0	0	0	0	0	1

Repitiendo el proceso hasta:

- El ctrl. de caché va a direccionar a la mem. ppal. a la dirección

0	0	0	0	0	1	0	0	1	0	1	1	1	1	1

y llevará el contenido al bloque

0	0	1	0	1	1

dirección

1	1	1

escribiendo la etiqueta

0	0	0	0	0	1

- Con lo que el bloque completo estará escrito en la mem. caché , existiendo ahora acierto y presentando en el bus de datos el contenido de la mem. caché correspondiente al bloque

0	0	1	0	1	1

Y la dirección de palabra

1	1	1

- Si posteriormente se quiere acceder al contenido de la dirección de mem. $085B_H$

$085B_H$

b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
0	0	0	1	0	0	0	0	1	0	1	1	0	1	1
etiqueta						**Nº bloque**						**Dir. palabra**		

Irá al bloque

0	0	1	0	1	1

Que estás ocupado con el dato anterior de la dirección $025F_H$

Por lo tanto la comparación de la etiqueta de la caché con la parte correspondiente a la etiqueta del bus de direcciones presentará desigualdad ⇒ Fallo

Etiqueta de la caché

0	0	0	0	0	1

bits correspondientes a etiqueta del bus de direcciones

0	0	0	1	0	0

Por lo que llevará el bloque desde la dirección de mem. ppal.

0	0	0	1	0	0	0	0	1	0	1	1	0	0	0

a la dirección

0	0	0	1	0	0	0	0	1	0	1	1	1	1	1

al bloque

0	0	1	0	1	1

de la caché, existiendo actualmente un acierto y presentando en el bus de datos el contenido de la mem. caché correspondiente al bloque

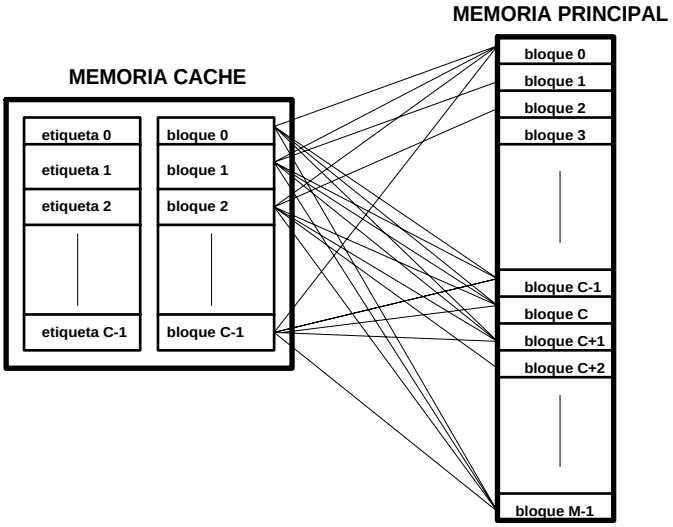
0	0	1	0	1	1

Y la dirección de palabra

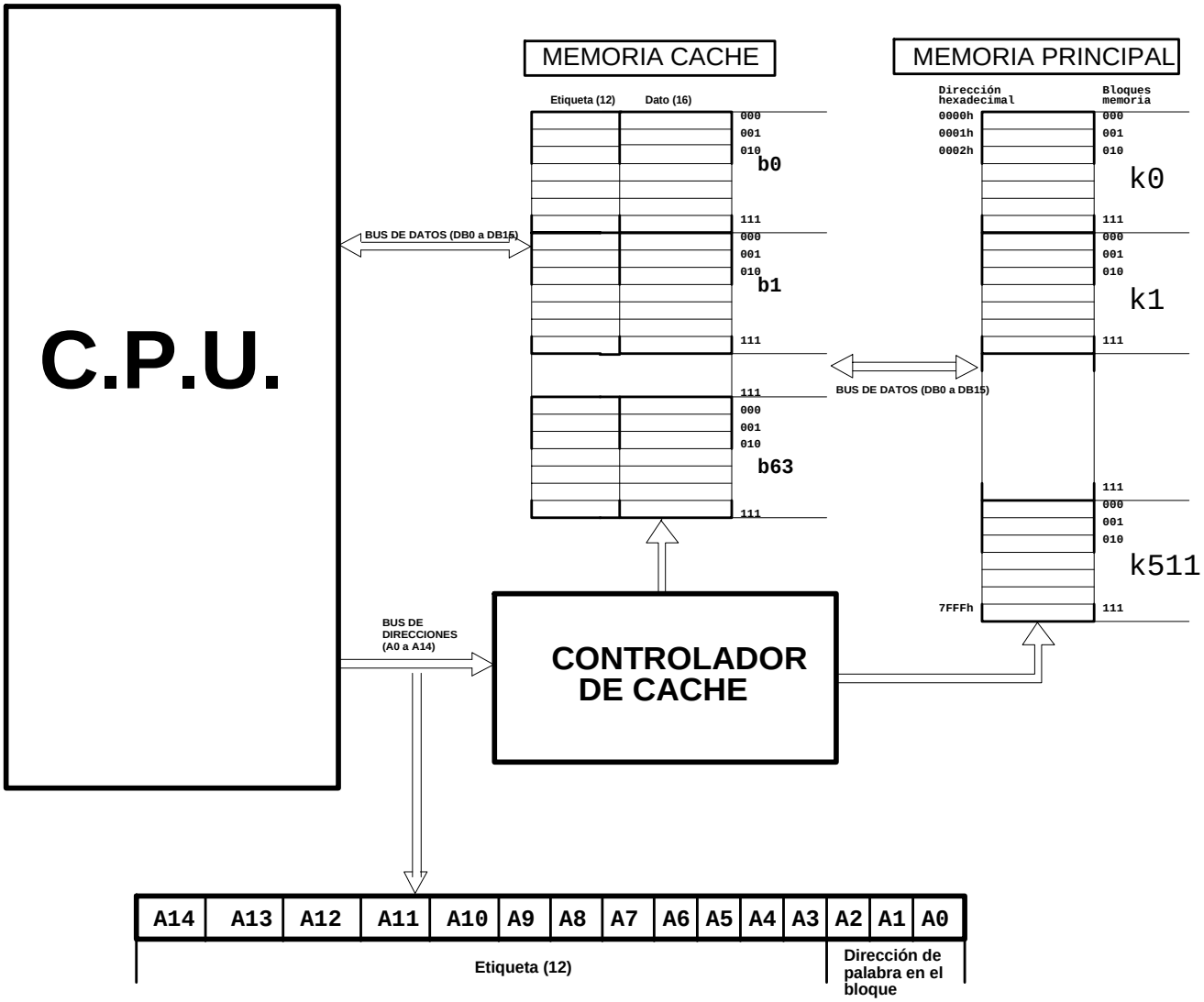
0	1	1

CORRESPONDENCIA TOTALMENTE ASOCIATIVA

Los bloques de la mem principal se alojan en cualquier bloque de la mem caché, comprobando solamente la etiqueta de todos y cada uno de los bloques para verificar acierto.
El principal inconveniente es que precisa una circuitería compleja para hacer la comparación paralelo de todos los campos de etiqueta.



Esquema de la caché



Funcionamiento:

Principios:

- Palabras por bloque 8 ⇒ 3 bits (b0 a b2)
- Nº bloques 64
- Etiqueta = (bus direcc) – (bits palab/bloque) ⇒ 15 –3 = 12
- Ancho de la mem. caché = Ancho palabra + ancho etiqueta ⇒ 16 + 12 = 28 bits

1. La CPU entrega la dirección de mem. al controlador de caché.
2. El ctrl. de caché busca en todos y cada uno de los bloques coincidencia entre los bits b15 a b3 del bus de direcciones con la etiqueta.

3.a Si encuentra coincidencia ⇒ acierto, con lo que coge de los bits b15 a b0 (dato) del bloque de la caché con coincidencia y los saca al bus de datos de la CPU.

4.a Fin de acceso.

4.b Si no encuentra coincidencia ⇒ Fallo

5.b El ctrl. de caché transmite bloque completo desde mem. ppal. a mem. caché

5.b.1 El ctrl. Caché busca un bloque de caché para librarlo (algoritmo de reemplazamiento).

5.b.2 Una vez liberado el ctrl. Caché pone a 000 los bits de dir. de palabra (b2 a b0).

5.b.2 El ctrl. Caché apunta al bloque liberado.

5.b.3 El ctrl. Caché lleva de la dir. b14 b13 b12 .....b3 0 0 0 de mem. ppal. a la caché al bloque liberado en la dir. de palabra del bloque 0 0 0 (palabra 000 del bloque). Incrementa un contador y lleva de la dir. b14 b13 b12 .....b3 0 0 1 de mem. ppal. a la caché al bloque liberado en la dir. de palabra del bloque 0 0 1 (palabra 001 del bloque). Así sucesivamente hasta la palabra b14 b13 b12 .....b3 1 1 1 (palabra 7 del bloque) . Escribiendo en la zona de la etiqueta el valor de los bits b14 a b3.

5.b.4 Una vez ha cargado el bloque completo (8 palabras) ⇒ Hay acierto y se salta al paso 3.a.

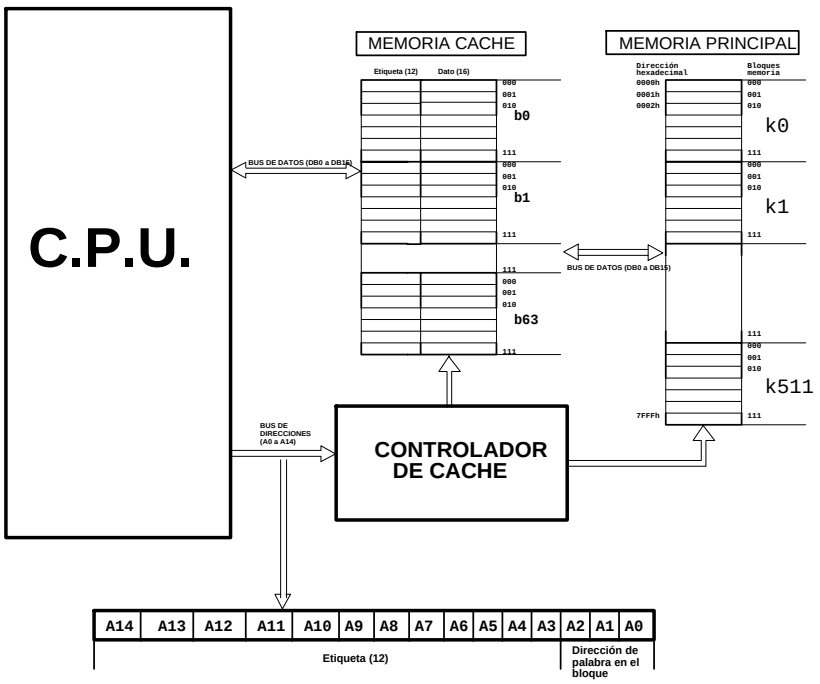
Ejemplo con los parámetros indicados

- Se supone que todavía no ha habido ningún acceso ⇒ Mem. caché vacía ⇒ todo con 0
- Se quiere acceder al contenido de la posición mem. $025F_H$

$025F_H$ =

b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
0	0	0	0	0	1	0	0	1	0	1	1	1	1	1
etiqueta												**Dir. palabra**		

Bits dir. palabra = 3	Bits etiqueta = 12	Ancho palabra = 16



- El ctrl. de la caché mira si encuentra la etiqueta en alguno de los bloques:

0	0	0	0	0	1	0	0	1	0	1	1

Como es el primer acceso la mem. caché contiene todo 0 ⇒ hay primer fallo.

- El ctrl. de caché va a direccionar a la mem. ppal. a la dirección

0	0	0	0	0	1	0	0	1	0	1	1	0	0	0

y llevará el contenido al bloque primero

0	0	0	0	0	0

dirección

0	0	0

escribiendo la etiqueta

0	0	0	0	0	1	0	0	1	0	1	1

Repitiendo el proceso hasta:

El ctrl. de caché va a direccionar a la mem. ppal. a la dirección

0	0	0	0	0	1	0	0	1	0	1	1	1	1	1

y llevará el contenido al bloque

0	0	0	0	0	0

dirección

1	1	1

escribiendo la etiqueta

0	0	0	0	0	1	0	0	1	0	1	1

- Con lo que el bloque completo estará escrito en la mem. caché , existiendo ahora acierto y presentando en el bus de datos el contenido de la mem. caché correspondiente al bloque

0	0	0	0	0	0

Y la dirección de palabra

1	1	1

- Si posteriormente se quiere acceder al contenido de la dirección de mem. $085B_H$

$085B_H$

b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
0	0	0	1	0	0	0	0	1	0	1	1	0	1	1
etiqueta												**Dir. palabra**		

El ctrl. de la caché mira si encuentra la etiqueta en alguno de los bloques, que no encontrará porque todavía no ha sido cargado ⇒ Fallo

- El ctrl. de caché busca un bloque libre

0	0	0	0	0	1

- El ctrl. de caché va a direccionar a la mem. ppal. a la dirección

0	0	0	1	0	0	0	0	1	0	1	1	0	0	0

y llevará el contenido al bloque segundo

0	0	0	0	0	1

dirección

0	0	0

escribiendo la etiqueta

0	0	0	1	0	0	0	0	1	0	1	1

Repitiendo el proceso hasta:

El ctrl. de caché va a direccionar a la mem. ppal. a la dirección

0	0	0	1	0	0	0	0	1	0	1	1	1	1	1

y llevará el contenido al bloque

0	0	0	0	0	1

dirección

1	1	1

escribiendo la etiqueta

0	0	0	1	0	0	0	0	1	0	1	1

- Con lo que el bloque completo estará escrito en la mem. caché , existiendo ahora acierto y presentando en el bus de datos el contenido de la mem. caché correspondiente al bloque

0	0	0	0	0	1

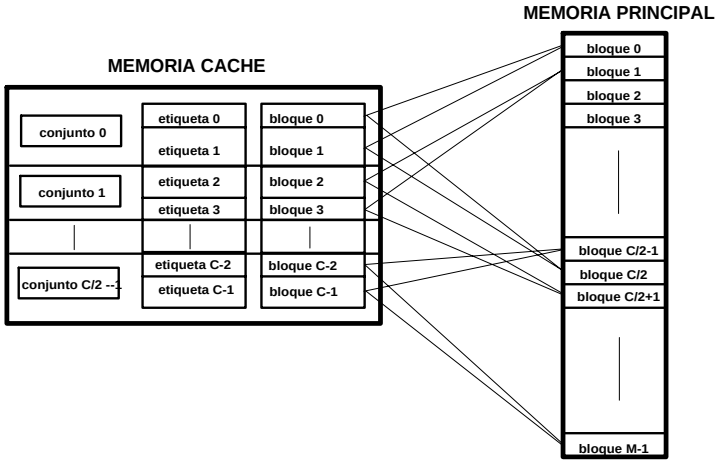
Y la dirección de palabra

0	1	1

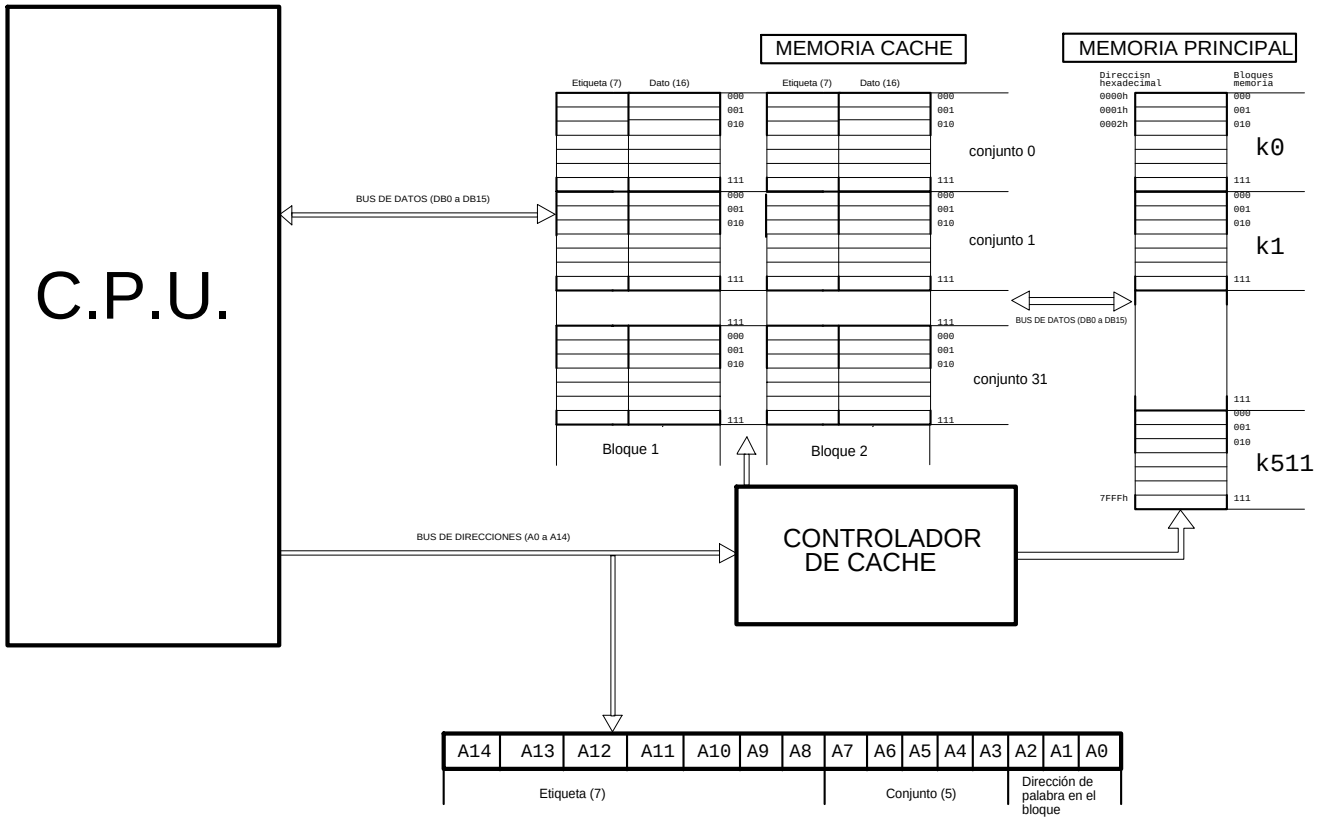
Este proceso se repetirá hasta ocupar los 64 bloques, momento en el cual el próximo bloque antes de entrar deberá liberar uno ya existente, según el algoritmo de reemplazamiento utilizado por el controlador de la memoria caché.

CORRESPONDENCIA ASOCIATIVA POR CONJUNTOS

- Auna las ventajas de los dos métodos anteriores.
- Está compuesta por “r” bloques y “q” conjuntos de modo que C = q×r , siendo C el nº de bloques de la mem. caché.
- El funcionamiento consiste en que cada bloque de la mem. ppal. tiene asignado un conjunto de la caché, pero se puede ubicar en cualquiera de los bloques que pertenecen a dicho conjunto. Ello permite mayor flexibilidad que la correspondencia directa y menor cantidad de comparaciones que la totalmente asociativa.



Esquema de la caché



Funcionamiento:

Principios:

- Palabras por bloque 8 $\Rightarrow$ 3 bits (b0 a b2)
- Nº bloques/conjunto =2 $\Rightarrow$ Nº conjuntos 64 /2 = 32 $\Rightarrow$ 2^5 $\Rightarrow$ 5 bits
- Etiqueta = (bus direcc) – (bits palab/bloque) -- (bits conjuntos) $\Rightarrow$ 15 –3 – 5 = 7
- Ancho de la mem. caché = Ancho palabra + ancho etiqueta $\Rightarrow$ 16 + 7 = 23 bits

1. La CPU entrega la dirección de mem. al controlador de caché.
2. El ctrl. de caché aísla de los bits b3 a b7 y con ellos apunta al conjunto correspondiente.
3. El ctrl. de caché busca en todos y cada uno de los bloques del conjunto coincidencia entre los bits b14 a b8 del bus de direcciones con la etiqueta.

   3.a Si iguales $\Rightarrow$ acierto, con lo que coge de los bits b15 a b0 (dato) de la caché y los saca al bus de datos de la CPU.

   4.a Fin de acceso.

   4.b Si no encuentra coincidencia $\Rightarrow$ Fallo

   5.b El ctrl. de caché transmite bloque completo desde mem. ppal. a mem. caché

   5.b.1 El ctrl. Caché busca un bloque dentro del conjunto apuntado de caché para librarlo (algoritmo de reemplazamiento).

   5.b.2 Una vez liberado el ctrl. Caché pone a 000 los bits de dir. de palabra (b2 a b0).

   5.b.2 El ctrl. Caché apunta al bloque liberado.

   5.b.3 El ctrl. Caché lleva de la dir. b14 b13 b12 .....b3 0 0 0 de mem. ppal. a la caché al bloque liberado en el conjunto apuntado por los bits b7 a b3 la dir. de palabra del bloque 0 0 0 (palabra 000 del bloque). Incrementa un contador y lleva de la dir. b14 b13 b12 .....b3 0 0 1 de mem. ppal. a la caché al bloque liberado en el conjunto apuntado por los bits b7 a b3 en la dir. de palabra del bloque 0 0 1 (palabra 001 del bloque). Así sucesivamente hasta la palabra b14 b13 b12 .....b3 1 1 1 (palabra 7 del bloque) . Escribiendo en la zona de la etiqueta el valor de los bits b14 a b8.

   5.b.4 Una vez ha cargado el bloque completo (8 palabras) $\Rightarrow$ Hay acierto y se salta al paso 3.a.

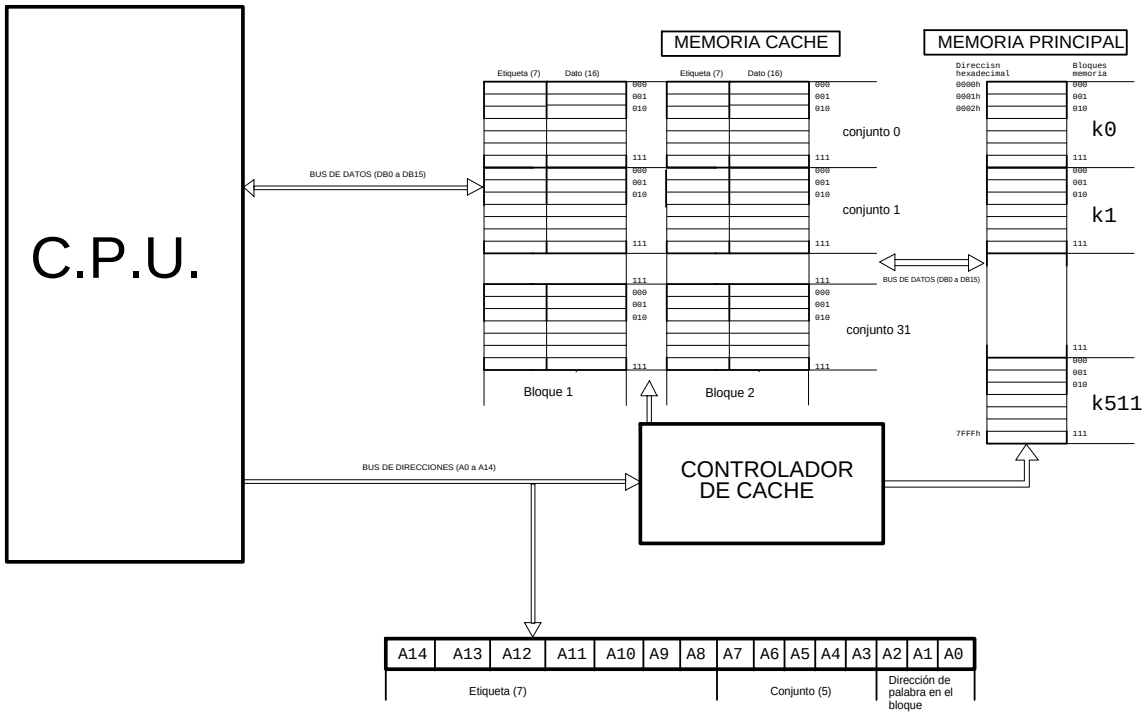
Ejemplo con los parámetros indicados

- Se supone que todavía no ha habido ningún acceso ⇒ Mem. caché vacía ⇒ todo con 0
- Se quiere acceder al contenido de la posición mem. $025F_H$

$025F_H$ =

b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
0	0	0	0	0	1	0	0	1	0	1	1	1	1	1
etiqueta							**conjunto**					**Dir. palabra**		

Bits dir. palabra = 3	Bits etiqueta = 7	Bits de conjunto = 5	Ancho palabra = 16



- El ctrl. de la caché mira si en el conjunto :

0	1	0	1	1

Hay una etiqueta que contenga

0	0	0	0	0	1	0

Como es el primer acceso la mem. caché contiene todo 0 ⇒ hay primer fallo.

- El ctrl. de caché va a direccionar a la mem. ppal. a la dirección

0	0	0	0	0	1	0	0	1	0	1	1	0	0	0

y llevará el contenido al bloque 1 del conjunto seleccionado

0	1	0	1	1

dirección

0	0	0

escribiendo la etiqueta

0	0	0	0	0	1	0

Repitiendo el proceso hasta:

- El ctrl. de caché va a direccionar a la mem. ppal. a la dirección

0	0	0	0	0	1	0	0	1	0	1	1	1	1	1

y llevará el contenido al bloque 1 del conjunto seleccionado

0	1	0	1	1

dirección

1	1	1

escribiendo la etiqueta

0	0	0	0	0	1	0

- Con lo que el bloque completo estará escrito en la mem. caché , existiendo ahora acierto y presentando en el bus de datos el contenido de la mem. caché correspondiente al bloque 1 del conjunto

0	1	0	1	1

Y la dirección de palabra

1	1	1

- Si posteriormente se quiere acceder al contenido de la dirección de mem. $085B_H$

$085B_H$

b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
0	0	0	1	0	0	0	0	1	0	1	1	0	1	1
etiqueta							**conjunto**					**Dir. palabra**		

- El ctrl. de la caché mira si en el conjunto :

0	1	0	1	1

Hay una etiqueta que contenga

0	0	0	0	0	1	0

Como no existe porque todavía no ha sido cargada $\Rightarrow$ hay fallo.

- El ctrl. de caché va a direccionar a la mem. ppal. a la dirección

0	0	0	1	0	0	0	0	1	0	1	1	0	1	1

y llevará el contenido al bloque 2 del conjunto seleccionado

0	1	0	1	1

dirección

0	0	0

escribiendo la etiqueta

0	0	0	1	0	0	0

Repitiendo el proceso hasta:

- El ctrl. de caché va a direccionar a la mem. ppal. a la dirección

0	0	0	1	0	0	0	0	1	0	1	1	1	1	1

y llevará el contenido al bloque 1 del conjunto seleccionado

0	1	0	1	1

dirección

1	1	1

escribiendo la etiqueta

0	0	0	1	0	0	0

- Con lo que el bloque completo estará escrito en la mem. caché , existiendo ahora acierto y presentando en el bus de datos el contenido de la mem. caché correspondiente al bloque 1 del conjunto

0	1	0	1	1

Y la dirección de palabra

1	1	1

- Si posteriormente se quiere acceder al contenido de la dirección de mem. $005B_H$

$005B_H$

b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
0	0	0	0	0	0	0	0	1	0	1	1	0	1	1
etiqueta							**conjunto**					**Dir. palabra**		

- El ctrl. de la caché mira si en el conjunto :

0	1	0	1	1

Hay una etiqueta que contenga

0	0	0	0	0	0	0

Como no existe $\Rightarrow$ hay fallo.

- El ctrl. de caché liberará uno de los dos bloques del mencionado conjunto según algoritmo de reemplazamiento y procederá según los pasos indicados anteriormente al cargar el nuevo bloque.

CORRESPONDENCIA DIRECTA

6.- Una memoria caché por correspondencia directa utiliza particiones de 64 palabras y su capacidad total son 1024 palabras. La memoria principal tiene capacidad para 2^{20} palabras. Decir si las siguientes afirmaciones son ciertas:

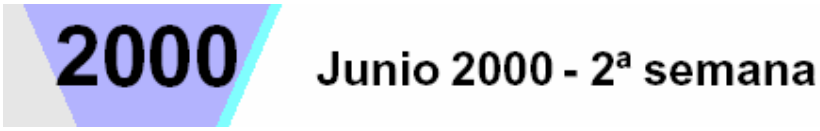
I. Las direcciones de memoria principal **1D45F** y **02075**, expresadas en hexadecimal, se corresponden con la partición 4 de la memoria caché.
II. La dirección de memoria **2A23B**, expresada en hexadecimal, se corresponde con la partición 8 de la memoria caché.

A) I: sí, II: sí. B) I: sí, II: no. C) I: no, II: sí. D) I: no, II: no.

8.- Un computador tiene una unidad de memoria de 4096 palabras y una memoria caché de 64 palabras. La memoria caché utiliza correspondencia directa, con un tamaño de partición de 16 palabras. Suponer que inicialmente la memoria caché está vacía y que se leen sucesivamente las direcciones de memoria principal 000000010000, 000100010100, 000001001000 y 000001111000. Indicar si al finalizar estas cuatro operaciones de lectura las afirmaciones siguientes son ciertas:

I. La palabra de dirección 000001111000 se encuentra almacenada en el bloque 3 de la caché.
II. El bloque 1 de la caché tiene asociada la etiqueta 000100.

A) I: sí, II: sí. B) I: sí, II: no. C) I: no, II: sí. D) I: no, II: no.

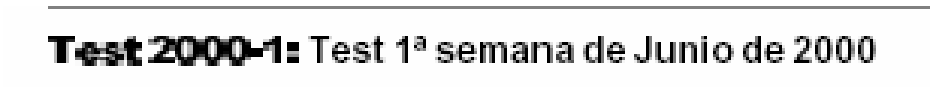


6.- Un computador tiene una unidad de memoria de 4096 palabras y una memoria caché de 64 palabras. La memoria caché utiliza correspondencia directa, con un tamaño de partición de 16 palabras. Suponer que inicialmente la memoria caché está vacía y que se leen sucesivamente las direcciones de memoria principal 000000010000, 000100010100, 000001001000 y 000001111000. Indicar si al finalizar estas cuatro operaciones de lectura las afirmaciones siguientes son ciertas:

I. La palabra de dirección 000001111000 se encuentra almacenada en el conjunto 0 de la caché.
II. El conjunto 1 de la caché tiene asociada la etiqueta 000000.

A) I: sí, II: sí. B) I: sí, II: no. C) I: no, II: sí. D) I: no, II: no.

TOTALMENTE ASOCIATIVA



5.- Un computador tiene una unidad de memoria de 256 palabras y una memoria caché de 32 palabras. La memoria caché es totalmente asociativa, con un tamaño de partición de 8 palabras. Cuando se produce un fallo en la caché se reemplaza la partición más antigua. Suponer que inicialmente la memoria caché está vacía y que se leen sucesivamente las direcciones de memoria principal: 00000000, 00000001, 00000011, 00100001, 00100101, 00010000, 00010010 y 00000000. Si se leyera la dirección 00100111.

I. Se produciría un acierto en la memoria caché.
II. Se produciría un fallo en la memoria caché y sería necesario reemplazar uno de los bloques existentes en la caché.

A) I: sí, II: sí. B) I: sí, II: no. C) I: no, II: sí. D) I: no, II: no.

ASOCIATIVA POR CONJUNTOS

2001 Junio 2ª SEMANA

3.- Un computador tiene una unidad de memoria de 8192 Kpalabras y una memoria caché de 2 Kpalabras. La memoria caché utiliza correspondencia asociativa por conjuntos, con un tamaño de partición de 64 palabras y 4 particiones por conjunto. Suponer que inicialmente la memoria caché está **llena** con la ejecución de un programa anterior **Prog1** y se carga en memoria principal un nuevo programa **Prog2**. **Prog2** efectúa la siguiente secuencia de referencias en la ejecución de su código: se leen secuencialmente las direcciones *128, 129, 130 hasta la 143*, posteriormente se repite un bucle de lectura de las direcciones *131,132 y 133* **20 veces** y finalmente se leen secuencialmente las direcciones *134, 135, 136 hasta la 168*. ¿Cuál es la tasa de aciertos obtenida para **Prog2**?

A) 87%. B) 13 %. C) 99%. D) Ninguna de las anteriores.

2002 Septiembre

3.- Desde el punto de vista de una caché una dirección de memoria principal se divide en tres campos: ETIQUETA 20 bits, CONJUNTO 7 bits y PALABRA 5 bits. La caché tiene 2 bloques por conjunto. Sabiendo que la longitud de palabra es de 1 byte, ¿cuál es el tamaño de la memoria caché?

A) 1 Kbyte B) 2 Kbytes C) 4 Kbytes D) Ninguna de las anteriores.

PROBLEMAS

En la sección de problemas se han puesto los correspondientes a los resueltos en el libro de problemas:

2.8 ..... Septiembre del 2000

2.12 ..... Junio del 2000 1ª semana

2.13 ..... Junio del 2001 1ª semana