

RELACIÓN 3 COMPLETA

Al igual que en el tema 2, al principio os dejaré todas las capturas del Campus Virtual, y después todas mis resoluciones a mano.

Ánimo, que a partir del 13 se hace muy rápido. (Se me ha bugueado el documento, empezamos en la próxima hoja jajajajaj)

Pregunta 1

Correcta

Puntúa 6,00
sobre 6,00

Marcar pregunta

Un computador tiene un bus de direcciones de 32 bits y es direccionable a nivel de byte. La cache tiene 256 Kbytes, con palabras de 1 byte y bloques de 32 bits, **asignación directa**, reemplazo LRU y post-escritura. Se utiliza un contador de 5 bits para el reemplazo y 1 bit de Dirty para los bloques que hayan sido modificados y por tanto tienen que escribirse en memoria principal.

a) Descompón en los siguientes campos la dirección física e indica su tamaño en bits:

TAG = bits

Indice/conjunto= bits

w (palabra dentro del bloque) = bits

b (byte dentro de la palabra) = bits

b) ¿Detectas algún dato en el enunciado que no te parezca consistente?

Si, con una asignación directa no hace falta implementar una política de reemplazo (no necesito el contador de 5 bits en el directorio)

¿Cuál es el tamaño, en bits, del directorio de la cache? Considera que en el directorio se encuentran todos los bits de control necesarios para el funcionamiento de la memoria cache.

Tamaño bits

Pregunta 2

Correcta

Puntúa 6,00
sobre 6,00

Marcar pregunta

Una memoria cache **asociativa por conjuntos de 2 bloques** con direcciones de 32 bits y palabras de 1 byte tiene 4 bytes en cada uno de sus bloques y una capacidad total de 128 Kbytes. El direccionamiento es a nivel de byte. Se pide:

a) El tamaño de los siguientes campos de la dirección de memoria principal:

- el campo número de línea o bloque: ✓ bits
- etiqueta (TAG): ✓ bits
- conjunto (c): ✓ bits
- desplazamiento de palabra dentro del bloque (w): ✓ bits

b) Indicar el valor de cada campo para las siguientes direcciones de memoria principal. Debes proporcionar los valores en hexadecimal, con el número de dígitos que corresponde al tamaño del campo y usando mayúsculas para las letras:

Dirección MP TAG	c	w
0284A482h	<input type="text" value="0284"/> ✓ h	<input type="text" value="2920"/> ✓ h
01148C89h	<input type="text" value="0114"/> ✓ h	<input type="text" value="2322"/> ✓ h
0038CF00h	<input type="text" value="0038"/> ✓ h	<input type="text" value="33C0"/> ✓ h
0038CF01h	<input type="text" value="0038"/> ✓ h	<input type="text" value="33C0"/> ✓ h

c) ¿Pueden todos los bloques que incluyen las referencias anteriores estar en caché al mismo tiempo? ✓

Pregunta 3

Correcta

Puntúa 8,00
sobre 8,00Marcar
pregunta

Sea un sistema de memoria de 1 MByte, con palabras de 1 byte, que incorpora una memoria caché de 64 KBytes **organizada asociativamente en cuatro conjuntos** con 8 palabras por bloque y algoritmo de reemplazo LRU. Se ejecuta un programa que referencia a la secuencia de direcciones de memoria principal (en hexadecimal) que aparece en la tabla.

a) Indica para cada referencia la dirección de bloque en memoria principal (en hexadecimal), el conjunto de la caché asignado y si se produce un acierto o un fallo. Para ello habrás tenido previamente que determinar para tus referencias:

- Tamaño del campo que identifica el byte dentro de la palabra: 0 ✓ bits
- Tamaño del campo que identifica la palabra dentro del bloque: 3 ✓ bits
- Tamaño del campo que asigna el conjunto: 2 ✓ bits
- Tamaño del campo etiqueta (TAG): 15 ✓ bits

Secuencia de direcciones

Dirección MP	Bloque en MP	Conjunto	Acierto
ABC80h	15790	✓ h 0 ✓ h	Fallo ↴ ✓
ABC81h	15790	✓ h 0 ✓ h	Acierto ↴ ✓
ABC88h	15791	✓ h 1 ✓ h	Fallo ↴ ✓
BCD90h	179B2	✓ h 2 ✓ h	Fallo ↴ ✓
BCD9Dh	179B3	✓ h 3 ✓ h	Fallo ↴ ✓
BCDA0h	179B4	✓ h 0 ✓ h	Fallo ↴ ✓
CDE00h	19BC0	✓ h 0 ✓ h	Fallo ↴ ✓
CDE18h	19BC3	✓ h 3 ✓ h	Fallo ↴ ✓
CDE20h	19BC4	✓ h 0 ✓ h	Fallo ↴ ✓

Calcula el índice de fallos (con dos decimales): ✓

b) Supón que las referencias anteriores se corresponden con una iteración de un bucle. ¿Cuál será el índice de fallos cuando el número de iteraciones tienda a infinito? ✓

Pregunta 4

Correcta

Puntúa 10,00
sobre 10,00

Marcar pregunta

Disponemos de una memoria cache de 4 Kbytes con tamaño de bloque de 256 bytes, **asociatividad 2** y *algoritmo de reemplazo LRU*. Conectamos la cache entre un procesador que trabaja con palabras de 8 bits y una memoria principal de 1 Mbyte. Para la siguiente secuencia de peticiones de direcciones a memoria principal:

319F0h, 31AF0h, 7013Ch, 77777h, 44037h, 778DEh, A5021h

a) Indica para una referencia:

- Tamaño del campo que identifica el byte dentro de la palabra: bits ✓
- Tamaño del campo que identifica la palabra dentro del bloque: bits ✓
- Tamaño del campo que asigna el conjunto: bits ✓
- Tamaño del campo etiqueta (TAG): bits ✓
- Suponiendo la cache inicialmente vacía, describir la evolución del directorio cache identificando, la etiqueta (TAG, en hexadecimal), el conjunto de la caché asignado (c, en hexadecimal) y si se produce un acierto o un fallo.

Secuencia de direcciones

Dirección MP Etiqueta (TAG) Conjunto (c) Acierto / Fallo

Dirección	MP	Etiqueta (TAG)	Conjunto (c)	Acierto / Fallo
319F0h	063	✓ h	1	✓ h Fallo ↴ ✓
31AF0h	063	✓ h	2	✓ h Fallo ↴ ✓
7013Ch	0E0	✓ h	1	✓ h Fallo ↴ ✓
77777h	0EE	✓ h	7	✓ h Fallo ↴ ✓
44037h	088	✓ h	0	✓ h Fallo ↴ ✓
778DEh	0EF	✓ h	0	✓ h Fallo ↴ ✓
A5021h	14A	✓ h	0	✓ h Fallo ↴ ✓

- Calcula el índice de fallos (con dos decimales): ✓

b) Comparar el sistema anterior con respecto a una cache **organizada de forma directa**. En este caso, indica para una referencia:

- Tamaño del campo que identifica el byte dentro de la palabra: ✓ bits
- Tamaño del campo que identifica la palabra dentro del bloque: ✓ bits
- Tamaño del campo que asigna el conjunto: ✓ bits
- Tamaño del campo etiqueta (TAG): ✓ bits
- Suponiendo la cache inicialmente vacía, describir la evolución del directorio cache identificando, la etiqueta (TAG, en hexadecimal), el conjunto de la caché asignado (c, en hexadecimal) y si se produce un acierto o un fallo.

Secuencia de direcciones

Dirección MP Etiqueta (TAG) Conjunto (c) Acierto / Fallo

319F0h	<input type="text" value="31"/> ✓ h	<input type="text" value="9"/> ✓ h	Fallo	✗	✓
31AF0h	<input type="text" value="31"/> ✓ h	<input type="text" value="A"/> ✓ h	Fallo	✗	✓
7013Ch	<input type="text" value="70"/> ✓ h	<input type="text" value="1"/> ✓ h	Fallo	✗	✓
77777h	<input type="text" value="77"/> ✓ h	<input type="text" value="7"/> ✓ h	Fallo	✗	✓
44037h	<input type="text" value="44"/> ✓ h	<input type="text" value="0"/> ✓ h	Fallo	✗	✓
778DEh	<input type="text" value="77"/> ✓ h	<input type="text" value="8"/> ✓ h	Fallo	✗	✓
A5021h	<input type="text" value="A5"/> ✓ h	<input type="text" value="0"/> ✓ h	Fallo	✗	✓

- Calcula el índice de fallos (con dos decimales): ✓

c) Comparar el sistema anterior con respecto a una cache **totalmente asociativa**. En este caso, indica para una referencia:

- Tamaño del campo que identifica el byte dentro de la palabra: ✓ bits
- Tamaño del campo que identifica la palabra dentro del bloque: ✓ bits
- Tamaño del campo que asigna el conjunto: ✓ bits
- Tamaño del campo etiqueta (TAG): ✓ bits
- Suponiendo la cache inicialmente vacía, describir la evolución del directorio cache identificando, la etiqueta (TAG, en hexadecimal), el conjunto de la caché asignado (c, en hexadecimal) y si se produce un acierto o un fallo.

Secuencia de direcciones

Dirección MP Etiqueta (TAG) Conjunto (c) Acierto / Fallo

319F0h	<input type="text" value="319"/> ✓ h	<input type="text" value="0"/> ✓ h	Fallo	↔	✓
31AF0h	<input type="text" value="31A"/> ✓ h	<input type="text" value="0"/> ✓ h	Fallo	↔	✓
7013Ch	<input type="text" value="701"/> ✓ h	<input type="text" value="0"/> ✓ h	Fallo	↔	✓
77777h	<input type="text" value="777"/> ✓ h	<input type="text" value="0"/> ✓ h	Fallo	↔	✓
44037h	<input type="text" value="440"/> ✓ h	<input type="text" value="0"/> ✓ h	Fallo	↔	✓
778DEh	<input type="text" value="778"/> ✓ h	<input type="text" value="0"/> ✓ h	Fallo	↔	✓
A5021h	<input type="text" value="A50"/> ✓ h	<input type="text" value="0"/> ✓ h	Fallo	↔	✓

- Calcula el índice de fallos (con dos decimales): ✓

d) A la vista de los resultados, teniendo en cuenta criterios de rendimiento y coste, recomendarías:

Org. directa

Pregunta 5

Correcta

Puntúa 10,00
sobre 10,00Marcar
pregunta

Un adicto a los PCs pretende evaluar el comportamiento de la caché L1 de datos del procesador "Pear P3" para un famoso videojuego de moda. Dispone para ello de una arquitectura con 16 Mbytes de memoria principal direccionable a nivel de byte y con caché L1 para datos 8 Kbytes. En cuanto al videojuego (programa a ejecutar), se compone de un *bucle de 100 iteraciones*, referenciando en cada una de ellas a una secuencia de 4 datos ubicados en las siguientes direcciones de memoria principal (en hexadecimal): 000A40h, 004A40h, 008A40h, 00BA40h. Nuestro amigo observa que la caché de datos evoluciona de la siguiente manera para la primera iteración del juego (en las 99 restantes, el comportamiento es muy similar):

Al final de la primera referencia:			Al final de la segunda referencia:			Al final de la tercera referencia:			Al final de la cuarta referencia:		
C	L	Etiq	C	L	Etiq	C	L	Etiq	C	L	Etiq
0	0	-	0	0	-	0	0	-	0	0	-
	1	-		1	-		1	-		1	-
1	0	-	1	0	-	1	0	-	1	0	-
	1	-		1	-		1	-		1	-
2	0	-	2	0	-	2	0	-	2	0	-
	1	-		1	-		1	-		1	-
3	0	-	3	0	-	3	0	-	3	0	-
...	...	-	-	-	-
81	1	-	81	1	-	81	1	-	81	1	-
82	0	000h	82	0	000h	82	0	008h	82	0	008h
	1	-		1	004h		1	004h		1	00Bh
83	0	-	83	0	-	83	0	-	83	0	-
	1	-		1	-		1	-		1	-
84	0	-	84	0	-	84	0	-	84	0	-
...	...	-	-	-	-
126	1	-	126	1	-	126	1	-	126	1	-
127	0	-	127	0	-	127	0	-	127	0	-
	1	-		1	-		1	-		1	-

El significado de cada campo es el siguiente: C = Número de conjunto; L = Número de línea; Etiq = Campo etiqueta del directorio caché; “-” = Línea vacía).

a) En función del comportamiento mostrado en los diagramas anteriores, determina:

- Tamaño conjunto de los campos que identifican el byte dentro de la palabra y la palabra dentro del bloque: bits
- Tamaño del campo que asigna el conjunto: bits

- Tamaño del campo que asigna el conjunto: bits ✓
- Tamaño del campo etiqueta (TAG): bits ✓
- Qué estrategias de reemplazo de líneas pueden dar lugar al resultado anterior (excluyendo RANDOM): ✓
- Índice de aciertos total (con dos decimales): ✓

b) Nuestro amigo quiere ahora cambiar el procesador de su equipo, dudando entre adquirir un "Carrot C4" o un "Grape G5".

El "Carrot C4" dispone de 16 Kbytes para la caché de datos y las mismas características que en el "Pear P3" original, aumentando únicamente el nivel de asociatividad (esto es, **se tiene el mismo número de conjuntos, pero se dobla el número de bloques por conjunto**). Por lo tanto ahora los tamaños de los campos pasan a ser:

- Tamaño conjunto de los campos que identifican el byte dentro de la palabra y la palabra dentro del bloque: bits ✓
- Tamaño del campo que asigna el conjunto: bits ✓
- Tamaño del campo etiqueta (TAG): bits ✓

Muestra la evolución para las 2 primeras iteraciones en Carrot C4:

Dirección MP Etiqueta (TAG) Conjunto (c) Acierto / Fallo

000A40h	<input type="text" value="000"/> ✓ h	<input type="text" value="52"/> ✓ h	Fallo	✗	✓
004A40h	<input type="text" value="004"/> ✓ h	<input type="text" value="52"/> ✓ h	Fallo	✗	✓
008A40h	<input type="text" value="008"/> ✓ h	<input type="text" value="52"/> ✓ h	Fallo	✗	✓
00BA40h	<input type="text" value="00B"/> ✓ h	<input type="text" value="52"/> ✓ h	Fallo	✗	✓
000A40h	<input type="text" value="000"/> ✓ h	<input type="text" value="52"/> ✓ h	Acierto	✗	✓
004A40h	<input type="text" value="004"/> ✓ h	<input type="text" value="52"/> ✓ h	Acierto	✗	✓
008A40h	<input type="text" value="008"/> ✓ h	<input type="text" value="52"/> ✓ h	Acierto	✗	✓
00BA40h	<input type="text" value="00B"/> ✓ h	<input type="text" value="52"/> ✓ h	Acierto	✗	✓

Con lo que el índice de aciertos total (con dos decimales) sería: 0,99



Por otro lado, el "Grape G5" tiene una caché de datos de 32 Kbytes, pero mantiene el nivel de asociatividad y el resto de parámetros del "Pear P3", con la salvedad del **número de conjuntos, que es ahora cuatro veces superior**. Por lo tanto ahora los tamaños de los campos pasan a ser:

- Tamaño conjunto de los campos que identifican el byte dentro de la palabra y la palabra dentro del bloque: 5 bits
- Tamaño del campo que asigna el conjunto: 9 bits
- Tamaño del campo etiqueta (TAG): 10 bits

Muestra la evolución para las 2 primeras iteraciones en Grape G5:

Dirección MP Etiqueta (TAG) Conjunto (c) Acierto / Fallo

000A40h	000	✓ h 052	✓ h Fallo	✗	✓
004A40h	001	✓ h 052	✓ h Fallo	✗	✓
008A40h	002	✓ h 052	✓ h Fallo	✗	✓
00BA40h	002	✓ h 1D2	✓ h Fallo	✗	✓
000A40h	000	✓ h 052	✓ h Fallo	✗	✓
004A40h	001	✓ h 052	✓ h Fallo	✗	✓
008A40h	002	✓ h 052	✓ h Fallo	✗	✓
00BA40h	002	✓ h 1D2	✓ h Acierto	✗	✓

Con lo que el índice de aciertos total (con dos decimales) pasa a ser: 0,25



¿Cuál de los dos microprocesadores recomendarías a nuestro amigo atendiendo exclusivamente al índice de aciertos a caché de datos producido por las referencias a memoria efectuadas desde el mencionado bucle del videojuego?:

Carrot C4



Pregunta 6

Correcta

Puntúa 12,00
sobre 12,00▼ Marcar
pregunta

Considérese un procesador con instrucciones de 32 bits y una **caché de instrucciones de 32 bytes** con **bloques de 8 bytes**. Para los dos fragmentos de código MIPS siguientes, indicar la organización (y sus parámetros) que da lugar a una ejecución con el mínimo número de fallos (si hay varias que produzcan el mismo rendimiento, ordenar por coste hardware).
NOTA: El número a la izquierda de cada instrucción indica la dirección en memoria principal donde se encuentra.

Código A:

0 lw \$1, 100(\$2)	0 lw \$1, 100(\$2)
4 add \$1, \$1, \$3	4 add \$1, \$1, \$3
8 sub \$4, \$5, \$1	8 sub \$4, \$5, \$1
12 j 24	12 j 32
...	...
24 mul \$2, \$2, \$8	24 mul \$2, \$2, \$8
28 sub \$2, \$2, \$9	28 sub \$2, \$2, \$3
32 div \$8, \$1, \$4	32 add \$8,
36 j 0	36 j 24

Código B:

a) Indica para una referencia:

- Para la **organización directa**:

- Tamaño del campo que identifica el byte dentro de la palabra: ✓ bits
- Tamaño del campo que identifica la palabra dentro del bloque: ✓ bits
- Tamaño del campo que asigna el conjunto: ✓ bits
- Tamaño del campo etiqueta (TAG): ✓ bits

- Para la **organización totalmente asociativa**:

- Tamaño del campo que identifica el byte dentro de la palabra: ✓ bits
- Tamaño del campo que identifica la palabra dentro del bloque: ✓ bits
- Tamaño del campo que asigna el conjunto: ✓ bits
- Tamaño del campo etiqueta (TAG): ✓ bits

- Para la **organización asociativa por conjuntos**:

- Tamaño del campo que identifica el byte dentro de la palabra: ✓ bits
- Tamaño del campo que identifica la palabra dentro del bloque: ✓ bits
- Tamaño del campo que asigna el conjunto: ✓ bits
- Tamaño del campo etiqueta (TAG): ✓ bits

b) Suponiendo que el código se ejecuta un número de iteraciones muy grande (infinito), calcula para la ejecución completa:

Código A:

- Para la **organización directa**:

- El índice de fallos (con dos decimales): ✓

- Para la **organización totalmente asociativa**:
 - El índice de fallos (con dos decimales): ✓
- Para la **organización asociativa por conjuntos**:
 - El índice de fallos (con dos decimales): ✓
- A la vista del rendimiento, teniendo en cuenta el coste hardware recomendaría: Org. asociativa por conjuntos ✓

Código B:

- Para la **organización directa**:
 - El índice de fallos (con dos decimales): ✓
- Para la **organización totalmente asociativa**:
 - El índice de fallos (con dos decimales): ✓
- Para la **organización asociativa por conjuntos**:
 - El índice de fallos (con dos decimales): ✓
- A la vista del rendimiento, teniendo en cuenta el coste hardware recomendaría: Org. directa ✓

Pregunta 7

Correcta

Puntúa 12,00
sobre 12,00Marcar
pregunta

Sea el siguiente programa en MIPS:

```
addi $1, $0, 1
j lab1
lab0: lw $1, 16($0)
sw $1, 80($0)
j lab1
...
lab1: sw $0, 16($0)
lw $2, 208($0)
lw $3, 336($0)
sw $2, 400($0)
bne $1, $0, lab0
```

donde la dirección lab0 es igual a 8 d y lab1 es igual a 140 d. Implementamos el primer nivel de la jerarquía de memoria con una cache de instrucciones, con asignación directa, y otra cache de datos, asociativa de 4 conjuntos y reemplazo LRU. Ambas caches tienen bloques de 8 bytes y un tamaño de 128 bytes cada una. Para cada cache se pide:

a) Mostrar la secuencia de referencias (direcciones, a nivel de byte) que hace el programa durante su ejecución:

a.1) Secuencia de referencias a instrucciones de MP (en decimal): 0 ✓ d, 4 ✓ d, 140 ✓ d, 144 ✓ d, 148 ✓ d, 152 ✓ d, 156 ✓ d
✓ d, 8 ✓ d, 12 ✓ d, 16 ✓ d, 140 ✓ d, 144 ✓ d, 148 ✓ d, 152 ✓ d, 156 ✓ d

a.2) Secuencia de referencias a datos de MP (en decimal): 16 ✓ d, 208 ✓ d, 336 ✓ d, 400 ✓ d, 16 ✓ d, 80 ✓ d, 16 ✓ d,
208 ✓ d, 336 ✓ d, 400 ✓ d

b) Muestra la evolución de cada una de las caches, indicando para cada referencia a un bloque de MP el número de conjunto de caché asignado y si se produce un fallo o un acierto.

b.1) Caché de instrucciones:

En primer lugar, indica para una referencia:

- Tamaño del campo que identifica el byte dentro de la palabra: 2 ✓ bits
- Tamaño del campo que identifica la palabra dentro del bloque: 1 ✓ bits
- Tamaño del campo que asigna el conjunto: 4 ✓ bits

- Tamaño del campo etiqueta (TAG): 25 ✓ bits

Para las referencias que anotaste en el apartado a.1 indica:

Bloque de MP Conjunto Acierto?

0	✓ d	0	✓ d	Fallo ↴	✓
0	✓ d	0	✓ d	Acierto ↴	✓
17	✓ d	1	✓ d	Fallo ↴	✓
18	✓ d	2	✓ d	Fallo ↴	✓
18	✓ d	2	✓ d	Acierto ↴	✓
19	✓ d	3	✓ d	Fallo ↴	✓
19	✓ d	3	✓ d	Acierto ↴	✓
1	✓ d	1	✓ d	Fallo ↴	✓
1	✓ d	1	✓ d	Acierto ↴	✓
2	✓ d	2	✓ d	Fallo ↴	✓
17	✓ d	1	✓ d	Fallo ↴	✓
18	✓ d	2	✓ d	Fallo ↴	✓
18	✓ d	2	✓ d	Acierto ↴	✓
19	✓ d	3	✓ d	Acierto ↴	✓
19	✓ d	3	✓ d	Acierto ↴	✓

- Tamaño del campo que identifica el byte dentro de la palabra: ✓ bits
- Tamaño del campo que identifica la palabra dentro del bloque: ✓ bits
- Tamaño del campo que asigna el conjunto: ✓ bits
- Tamaño del campo etiqueta (TAG): ✓ bits

Para las referencias que anotaste en el apartado a.2 indica:

Bloque de MP Conjunto Acierto?

<input type="text" value="2"/>	<input checked="" type="checkbox"/> d	<input type="text" value="2"/>	<input checked="" type="checkbox"/> d	Fallo	✗	✓
<input type="text" value="26"/>	<input checked="" type="checkbox"/> d	<input type="text" value="2"/>	<input checked="" type="checkbox"/> d	Fallo	✗	✓
<input type="text" value="42"/>	<input checked="" type="checkbox"/> d	<input type="text" value="2"/>	<input checked="" type="checkbox"/> d	Fallo	✗	✓
<input type="text" value="50"/>	<input checked="" type="checkbox"/> d	<input type="text" value="2"/>	<input checked="" type="checkbox"/> d	Fallo	✗	✓
<input type="text" value="2"/>	<input checked="" type="checkbox"/> d	<input type="text" value="2"/>	<input checked="" type="checkbox"/> d	Acierto	✗	✓
<input type="text" value="10"/>	<input checked="" type="checkbox"/> d	<input type="text" value="2"/>	<input checked="" type="checkbox"/> d	Fallo	✗	✓
<input type="text" value="2"/>	<input checked="" type="checkbox"/> d	<input type="text" value="2"/>	<input checked="" type="checkbox"/> d	Acierto	✗	✓
<input type="text" value="26"/>	<input checked="" type="checkbox"/> d	<input type="text" value="2"/>	<input checked="" type="checkbox"/> d	Fallo	✗	✓
<input type="text" value="42"/>	<input checked="" type="checkbox"/> d	<input type="text" value="2"/>	<input checked="" type="checkbox"/> d	Fallo	✗	✓
<input type="text" value="50"/>	<input checked="" type="checkbox"/> d	<input type="text" value="2"/>	<input checked="" type="checkbox"/> d	Fallo	✗	✓

c) Calcula las tasas de fallo (deja indicado numerador y denominador):

Tasa de fallos para la caché de instrucciones: ✓ / ✓

Tasa de fallos para la caché de datos: ✓ / ✓

Pregunta 8

Correcta

Puntúa 8,00
sobre 8,00

Marcar pregunta

Sea el siguiente programa en MIPS:

```
ori $2, $0, 1000h
loop: lw $1, 2800h($2)
      sub $4, $1, $0
      jal rotar
      sw $7, 7800h($2)
      sw $1, C800h($2)
      subi $2, $2, 4
      bne $2, $0, loop
      ...
rotar: add $10, $4, $4
      muli $7, $10, 2
      jr $31
```

Se desea usar una memoria cache de tamaño 4 Kbytes, con asignación directa, para resolver las referencias a datos. Se pide:

a) Vamos a analizar primero cual es el tamaño de bloque óptimo teniendo en cuenta la tasa de fallos. Prueba distintos tamaños de bloque, empezando con 1 palabra por bloque hasta llegar al máximo número de palabras por bloque, y calcula sus tasas de fallo.

¿Cuál es la mejor tasa de fallos que obtienes?

b) Si ahora fijamos el tamaño de bloque en 256 bytes,

¿es posible reducir la tasa de fallos aumentando el nivel de asociatividad?

¿A partir de qué nivel de asociatividad no se obtiene mejora?

Menor número de fallos totales obtenidos

c) Para la configuración elegida en el apartado anterior, ¿Qué política de reemplazo es mejor, LRU o FIFO?

Política de reemplazo óptima

Pregunta 9

Correcta

Puntúa 6,00
sobre 6,00

Marcar pregunta

En la figura se proponen dos códigos en alto nivel que inicializan ambas una matriz de 10×10 números reales a cero. El código generado por el compilador almacena la matriz en la memoria en posiciones consecutivas por filas. Las variables i, j son ubicadas en dos registros diferentes del procesador (no están en memoria).

<pre>i=1; while (i <=10) { j=1; while (j <=10) { A[i][j]=0.0; j=j+1; } i=i+1; }</pre>	<pre>i=1; while (i <=10) { j=1; while (j <=10) { A[j][i]=0.0; j=j+1; } i=i+1; }</pre>
<i>Código A</i>	<i>Código B</i>

Se dispone de un sistema de memoria cuya **caché de datos es completamente asociativa** y con reemplazo FIFO. El tamaño de la caché es equivalente a diez números reales en la representación usada por el procesador y permite almacenar dos números por bloque. Ambos códigos realizan la misma función, pero desde el punto de vista del sistema de memoria el rendimiento es diferente. Analiza cuál de ellos daría más fallos de caché. Para ello contesta a las siguientes preguntas:

• CÓDIGO A:

- En el código A, la secuencia de elementos referenciados es:

A(1,1),A(1,2),A(1,3),A(1,4),...,A(1,10),A(2,1),A(2,2),A(2,3),...,A(10,10)

por lo tanto el acceso a los elementos de la matriz se realiza por Filas

El índice de fallos (expresado como un porcentaje) resultante será: IF= %

• CÓDIGO B:

- En el código B, la secuencia de elementos referenciados es:

A(1,1),A(2,1),A(3,1),A(4,1),...,A(1,2),A(2,2),A(3,2),A(4,2),...,A(10,10)

por lo tanto el acceso a los elementos de la matriz se realiza por Columnas

El índice de fallos (expresado como un porcentaje) resultante será: IF= %

Pregunta 10

Correcta

Puntúa 12,00
sobre 12,00▼ Marcar
pregunta

Queremos ejecutar un bucle simple como éste:

```
for (i=0; i<=9; i++)  
    A[i] = 0;
```

en un procesador con una memoria caché de 8 bytes y **tamaño de bloque 2 bytes**. Sabiendo que `A` es un array de 30 números almacenados en memoria a partir de la dirección 000h y que cada elemento del array ocupa 1 byte, muestra la evolución en la caché de datos de ese código para las siguientes organizaciones:

- Organización directa

Dato	Dirección MP	Dir. Bloque	Conjunto (c)	Acierto / Fallo
A[0]	000h	0	0	Fallo ↴ ✓
A[1]	001	0	0	Acierto ↴ ✓
A[2]	002	1	1	Fallo ↴ ✓
A[3]	003	1	1	Acierto ↴ ✓
A[4]	004	2	2	Fallo ↴ ✓
A[5]	005	2	2	Acierto ↴ ✓
A[6]	006	3	3	Fallo ↴ ✓
A[7]	007	3	3	Acierto ↴ ✓
A[8]	008	4	0	Fallo ↴ ✓
A[9]	009	4	0	Acierto ↴ ✓

- Calcula el índice de aciertos (con dos decimales): ✓

- Organización totalmente asociativa

Dato	Dirección MP	Dir. Bloque	Conjunto (c)	Acierto / Fallo
A[0]	000h	0 ✓ h 0	✓ h	Fallo ↴ ✓
A[1]	001	✓ h 0 ✓ h 0	✓ h	Acierto ↴ ✓
A[2]	002	✓ h 1 ✓ h 0	✓ h	Fallo ↴ ✓
A[3]	003	✓ h 1 ✓ h 0	✓ h	Acierto ↴ ✓
A[4]	004	✓ h 2 ✓ h 0	✓ h	Fallo ↴ ✓
A[5]	005	✓ h 2 ✓ h 0	✓ h	Acierto ↴ ✓
A[6]	006	✓ h 3 ✓ h 0	✓ h	Fallo ↴ ✓
A[7]	007	✓ h 3 ✓ h 0	✓ h	Acierto ↴ ✓
A[8]	008	✓ h 4 ✓ h 0	✓ h	Fallo ↴ ✓
A[9]	009	✓ h 4 ✓ h 0	✓ h	Acierto ↴ ✓

◦ Calcula el índice de aciertos (con dos decimales): 0,50 ✓

- Organización asociativa por conjuntos

Dato	Dirección MP	Dir. Bloque	Conjunto (c)	Acierto / Fallo
A[0]	000h	0 ✓ h 0	✓ h	Fallo ↴ ✓
A[1]	001	✓ h 0 ✓ h 0	✓ h	Acierto ↴ ✓
A[2]	002	✓ h 1 ✓ h 1	✓ h	Fallo ↴ ✓
A[3]	003	✓ h 1 ✓ h 1	✓ h	Acierto ↴ ✓
A[4]	004	✓ h 2 ✓ h 0	✓ h	Fallo ↴ ✓
A[5]	005	✓ h 2 ✓ h 0	✓ h	Acierto ↴ ✓
A[6]	006	✓ h 3 ✓ h 1	✓ h	Fallo ↴ ✓
A[7]	007	✓ h 3 ✓ h 1	✓ h	Acierto ↴ ✓
A[8]	008	✓ h 4 ✓ h 0	✓ h	Fallo ↴ ✓
A[9]	009	✓ h 4 ✓ h 0	✓ h	Acierto ↴ ✓

◦ Calcula el índice de aciertos (con dos decimales): 0,50 ✓

a) ¿Qué tipo de organización de memoria cache (directa, por conjuntos o totalmente asociativa) elegirías en función del porcentaje de aciertos? Nota: a igualdad de rendimiento se debe elegir la menos costosa desde el punto de vista HW: Org. directa ✓

b) ¿Se podría decir lo mismo independientemente de la posición de comienzo del array A en memoria?: Si, porque todos los fallos son obligatorios ✓

Imaginemos ahora que el bucle que estamos procesando es este otro (en la sentencia $A[i]=B[9-i]$ el acceso de lectura ocurre antes que el de escritura):

```
for (i=0; i<=9; i++)
    A[i] = B[9-i];
```

donde **A** es el mismo array anterior y **B** es otro array de 45 números de 1 byte, almacenado a partir de la posición 60 de memoria. Realiza un esquema de la evolución en la caché de datos de este código, similar al código anterior, y responde a las siguientes preguntas:

- Organización directa:

- Calcula el índice de aciertos (con dos decimales): 0,3 ✓

- Organización totalmente asociativa:

- Calcula el índice de aciertos (con dos decimales): 0,5 ✓

- Organización asociativa por conjuntos:

- Calcula el índice de aciertos (con dos decimales): 0,5 ✓

c) ¿Qué tipo de organización de memoria cache elegirías ahora?: Org. asociativa por conjuntos ✓

d) ¿Es independiente la respuesta anterior de la posición de comienzo del array B en memoria?

No, porque se puede escoger una posición inicial de B con la que se evitan los conflictos en la asignación directa ✓

Pregunta 11

Correcta

Puntúa 10,00
sobre 10,00

Marcar pregunta

Sea un sistema de memoria de 1 MByte, con palabras de 1 byte, que incorpora una memoria caché de 64 KBytes, con 4096 palabras por bloque y algoritmo de reemplazo FIFO. Se ejecuta un programa que referencia a la siguiente secuencia de direcciones (en hexadecimal):

00000h, 01000h, 01001h, 0F000h, 10000h, 00000h, 40000h, 20000h, 08000h

a) Compara las tasas de fallo cuando se considera una organización con asignación directa, totalmente asociativa y asociativa con 4 conjuntos.

- Organización directa

Dato	Etiqueta (TAG)	Dir. Bloque	Conjunto (c)	Acierto / Fallo
00000h	0	✓ h	00	✓ h 0 ✓ h Fallo ↴ ✓
01000h	0	✓ h	01	✓ h 1 ✓ h Fallo ↴ ✓
01001h	0	✓ h	01	✓ h 1 ✓ h Acierto ↴ ✓
0F000h	0	✓ h	0F	✓ h F ✓ h Fallo ↴ ✓
10000h	1	✓ h	10	✓ h 0 ✓ h Fallo ↴ ✓
00000h	0	✓ h	00	✓ h 0 ✓ h Fallo ↴ ✓
40000h	4	✓ h	40	✓ h 0 ✓ h Fallo ↴ ✓
20000h	2	✓ h	20	✓ h 0 ✓ h Fallo ↴ ✓
08000h	0	✓ h	08	✓ h 8 ✓ h Fallo ↴ ✓

Calcula el índice de aciertos (con dos decimales): ✓

- Organización totalmente asociativa

Dato	Etiqueta (TAG)	Dir. Bloque	Vía dentro del conjunto	Acierto / Fallo
00000h	00	✓ h	00	✓ h Fallo ↴ ✓
01000h	01	✓ h	01	✓ h 1 ✓ h Fallo ↴ ✓

01001h	01	✓ h	01	✓ h	1	✓ h	Acierto	✓
0F000h	0F	✓ h	0F	✓ h	2	✓ h	Fallo	✓
10000h	10	✓ h	10	✓ h	3	✓ h	Fallo	✓
00000h	00	✓ h	00	✓ h	0	✓ h	Acierto	✓
40000h	40	✓ h	40	✓ h	4	✓ h	Fallo	✓
20000h	20	✓ h	20	✓ h	5	✓ h	Fallo	✓
08000h	08	✓ h	08	✓ h	6	✓ h	Fallo	✓

Calcula el índice de aciertos (con dos decimales): 0,22 ✓

- Organización asociativa por conjuntos

Dato	Etiqueta (TAG)	Dir. Bloque	Conjunto (c)	Acierto / Fallo
00000h	00	✓ h	00	✓ h Fallo
01000h	00	✓ h	01	✓ h Fallo
01001h	00	✓ h	01	✓ h Acierto
0F000h	03	✓ h	0F	✓ h Fallo
10000h	04	✓ h	10	✓ h Fallo
00000h	00	✓ h	00	✓ h Acierto
40000h	10	✓ h	40	✓ h Fallo
20000h	08	✓ h	20	✓ h Fallo

08000h 02 ✓ h 08 ✓ h 0 ✓ h Fallo ↴ ✓

Calcula el índice de aciertos (con dos decimales): 0,22 ✓

b) Calcula el tamaño de cada una de las caches suponiendo que en la zona de directorio solo se almacena la etiqueta TAG.

- Organización directa:

Tamaño de la cache en bits: 524352 ✓ bits

- Organización totalmente asociativa:

Tamaño de la cache en bits: 524416 ✓ bits

- Organización asociativa por conjuntos:

Tamaño de la cache en bits: 524384 ✓ bits

Pregunta 12

Correcta

Puntúa 10,00
sobre 10,00

Marcar pregunta

Considerar una memoria principal de 64 Kbytes direccionable a nivel de byte y con palabras de este mismo tamaño. Sea la siguiente secuencia de direcciones de acceso a memoria principal:

04F5h, 11E0h, 1500h, 2000h, 241Fh, 16FFh, 1233h, 21F0h

Mostrar la evolución, así como los fallos que se producen para cada una de las caches descritas en la siguiente tabla:

	Tamaño Cache	Tamaño Bloque	Organización	Reemplazo
C1	4 Kb	256 bytes	Directa	---
C2	2 Kb	256 bytes	Totalmente asociativa	FIFO
C3	2 Kb	256 bytes	Asociativa por conjuntos de 4 vías	LRU

Cache C1

- Tamaño del campo que identifica el byte dentro de la palabra: 0 ✓ bits
- Tamaño del campo que identifica la palabra dentro del bloque: 8 ✓ bits
- Tamaño del campo que asigna el conjunto: 4 ✓ bits
- Tamaño del campo etiqueta (TAG): 4 ✓ bits

Dirección MP Etiqueta (TAG) Conjunto (c) Acierto / Fallo

04F5h	0	✓ h	4	✓ h	Fallo	↔	✓
11E0h	1	✓ h	1	✓ h	Fallo	↔	✓
1500h	1	✓ h	5	✓ h	Fallo	↔	✓
2000h	2	✓ h	0	✓ h	Fallo	↔	✓
241Fh	2	✓ h	4	✓ h	Fallo	↔	✓

16FFh	1 ✓ h	6 ✓ h	Fallo ↕ ✓
1233h	1 ✓ h	2 ✓ h	Fallo ↕ ✓
21F0h	2 ✓ h	1 ✓ h	Fallo ↕ ✓

- Calcula el índice de fallos (con dos decimales): 1 ✓

Cache C2

- Tamaño del campo que identifica el byte dentro de la palabra: 0 ✓ bits
- Tamaño del campo que identifica la palabra dentro del bloque: 8 ✓ bits
- Tamaño del campo que asigna el conjunto: 0 ✓ bits
- Tamaño del campo etiqueta (TAG): 8 ✓ bits

Dirección MP Etiqueta (TAG) Conjunto (c) Acierto / Fallo

04F5h	04 ✓ h	0 ✓ h	Fallo ↕ ✓
11E0h	11 ✓ h	0 ✓ h	Fallo ↕ ✓
1500h	15 ✓ h	0 ✓ h	Fallo ↕ ✓
2000h	20 ✓ h	0 ✓ h	Fallo ↕ ✓
241Fh	24 ✓ h	0 ✓ h	Fallo ↕ ✓
16FFh	16 ✓ h	0 ✓ h	Fallo ↕ ✓
1233h	12 ✓ h	0 ✓ h	Fallo ↕ ✓
21F0h	21 ✓ h	0 ✓ h	Fallo ↕ ✓

- Calcula el índice de fallos (con dos decimales): 1 ✓

Cache C3

- Tamaño del campo que identifica el byte dentro de la palabra: 0 ✓ bits
- Tamaño del campo que identifica la palabra dentro del bloque: 8 ✓ bits
- Tamaño del campo que asigna el conjunto: 1 ✓ bits
- Tamaño del campo etiqueta (TAG): 7 ✓ bits

Dirección MP Etiqueta (TAG) Conjunto (c) Acierto / Fallo

04F5h	02 ✓ h	0 ✓ h	Fallo ↕ ✓
11E0h	08 ✓ h	1 ✓ h	Fallo ↕ ✓
1500h	0A ✓ h	1 ✓ h	Fallo ↕ ✓
2000h	10 ✓ h	0 ✓ h	Fallo ↕ ✓
241Fh	12 ✓ h	0 ✓ h	Fallo ↕ ✓
16FFh	0B ✓ h	0 ✓ h	Fallo ↕ ✓
1233h	09 ✓ h	0 ✓ h	Fallo ↕ ✓
21F0h	10 ✓ h	1 ✓ h	Fallo ↕ ✓

- Calcula el índice de fallos (con dos decimales): 1 ✓

Pregunta 13

Correcta

Puntúa 5,00
sobre 5,00Marcar
pregunta

Supongamos un sistema de memoria compuesto por 2 bancos de memoria principal entrelazada donde el tiempo de direccionamiento es de 1 ciclo de reloj y un tiempo de transferencia de memoria (tiempo de bus) de 2 ciclos. El tiempo de acceso a los bancos de memoria DRAM es de 20 ciclos para el primer acceso y 8 ciclos para los siguientes. Calcula la penalización por fallo y el ancho de banda de la memoria para los siguientes casos:

a) Tamaño de bus: 8 bytes y tamaño de bloque: 32 bytes

Penalización por fallo: ✓ ciclos

Ancho de banda de la memoria: ✓ bytes/ciclo (redondea a 2 decimales)

b) Tamaño de bus: 8 bytes y tamaño de bloque: 64 bytes

Penalización por fallo: ✓ ciclos

Ancho de banda de la memoria: ✓ bytes/ciclo (redondea a 2 decimales)

Pregunta 14

Correcta

Puntúa 2,00
sobre 2,00Marcar
pregunta

Calcula el tiempo medio de acceso a memoria (AMAT) en nseg. de un sistema cache donde la frecuencia de reloj es de 500 Mhz, el **tiempo de acierto es de 1 ciclo** de reloj, la **penalización por fallo** de 20 ciclos, y donde el **porcentaje de aciertos** es del 95%.

• $t_{AMAT} = 2$ ✓ ciclos

• $t_{AMAT} = 4$ ✓ nseg.

Pregunta 15

Correcta

Puntúa 6,00
sobre 6,00Marcar
pregunta

Considera un benchmark que ejecuta 1000 instrucciones en un procesador a 1.2 GHz, del cual 330 instrucciones son load/store. En este programa, el CPI para una cache perfecta es 1.8 (cache perfecta: no hay ningún fallo). Ahora consideremos una cache real con política write-through, en la que la tasa de fallos para la cache de instrucciones (I\$) es del 2% y la de la cache de datos (D\$) es del 8%. La penalización por fallo es de 10 ciclos para I\$ y de 15 ciclos para D\$. Calcula (redondea a 2 decimales los resultados):

a) Los ciclos de detención (stalls) debidos a los fallos: ✓ ciclos

b) CPI efectivo: ✓

c) Tiempo de ejecución del programa: ✓ nseg

d) Tiempo medio de acceso a memoria (AMAT) (hit time: 1 cc): ✓ nseg

Pregunta 16

Correcta

Puntúa 4,00
sobre 4,00Marcar
pregunta

Considera un procesador con caches separadas para instrucciones (I\$) y datos (D\$) cuya I\$ tiene una **tasa de fallos** del 0.5%, mientras que para la D\$ es del 1.5%, la **penalización por fallo** de I\$ es de 10 ciclos y de 15 para la D\$. El CPI base (el de una cache perfecta) es 1.7 y en el código hay un 32% de load&stores. Calcula (redondea a dos decimales si hace falta):

• CPI efectivo= ✓ ciclos

Pregunta 17

Correcta

Puntúa 8,00
sobre 8,00

Marcar pregunta

Considera un programa que ejecuta 2150 instrucciones en un MIPS a 1.5 GHz en que se han implementado todos los cortocircuitos posibles. Sabemos que el 20% de las instrucciones son saltos condicionales (el 30% de los mismos se toman), el 5% son saltos incondicionales, el 15% son una instrucción de carga seguida de una instrucción con una dependencia de datos verdadera con ella, el 10% son loads sin esa dependencia y el 8% son stores (con política write-through). Adicionalmente la tasa de aciertos para I\$ es del 99,4% (con 12 cc de penalización por fallo) y para la D\$ del 98% (con una penalización por fallo de 15 cc).

El procesador implementa la técnica de predicción estática de salto no tomado, actualizándose el PC en la etapa MEM para los altos condicionales y en la etapa ID para los incondicionales. **Calcula el CPI resultante.**

Para ello empieza por calcular las penalizaciones (ciclos por instrucción) a añadir sobre el CPI ideal de 1. Redondea a dos decimales donde sea necesario:

- Penalizaciones debidas a riesgos:
 - Penalización causada por los saltos condicionales: 0,18 ✓
 - Penalización causada por los saltos incondicionales: 0,05 ✓
 - Penalización debida a los loads-uso: 0,15 ✓
- Penalizaciones debidas a fallos en acceso a cache:
 - Penalización en las referencias a instrucciones: 0,072 ✓
 - Penalización en las referencias a datos: 0,1 ✓

Con lo que al sumarle todas las penalizaciones, el CPI efectivo resultante es: CPIef= 1,55 ✓

Pregunta 18

Correcta

Puntúa 5,00
sobre 5,00Marcar
pregunta

Considera un sistema cache con dos niveles L1, L2. La tasa de fallos para L1 es del 3% y para L2 es del 5%. Calcula (redondea a dos decimales si te hace falta):

a) Tasa de fallos global del sistema de memoria: 0,15 ✓ %

b) Si el número total de referencias es de 12.000, calcula el número de:

i) Fallos en L1: 360 ✓

ii) Referencias a L2: 360 ✓

iii) Fallos en L2: 18 ✓

c) Asumiendo, como usualmente, que el "**hit time**" para L1 es de 1 cc, el "**hit time**" para L2 es de 10 cc y la **penalización por fallos** de L2 es de 50 cc, calcula el AMAT.

$t_{AMAT} =$ 1,375 ✓ ciclos

Pregunta 19

Correcta

Puntúa 4,00
sobre 4,00Marcar
pregunta

Considera un procesador que trabaja a 1,7 GHz y tiene un sistema cache de dos niveles para el que la tasa de aciertos para L1 es del 90% y para L2 es del 75%. Teniendo en cuenta que el AMAT es de 5 cc y el tiempo de acceso para L2 es de 15 cc, calcula (redondeando a 2 decimales donde sea necesario):

• la penalización por fallo de L1: 40 ✓ ciclos, que equivale a 23,53 ✓ ns

• la penalización por fallo de L2: 100 ✓ ciclos, que equivale a 58,82 ✓ ns

Relación del Tema 3

① Cache $256 \text{ bytes} = 2^8 \cdot 2^{10} \text{ bytes} = 2^{18} \text{ bytes}$.

Blöques de $32 \text{ bits} = 4 \text{ bytes} = 2^2 \text{ bytes} \Rightarrow w=2$
Palabras de $2^0 \text{ bytes} \Rightarrow b=0$.

Índice = $18 - 2 - 0 = 16$.

TAG = $32 - 18 = 14$.

b) En el directorio tenemos el TAG (14 bits), el bit de Dirty y el de validez, y eso para cada bloque de la cache:

$$\# \text{blöques} = \frac{2^{18}}{2^2} = 2^{16}$$

$$\Rightarrow \text{Directorio: } 2^{16} \cdot (14+1+1) = 2^{20} = 1048576 \text{ bits}$$

② Asociatividad 2. (2 vías / conjunto).

Blöques de $2^2 \text{ bytes} \Rightarrow w=2$.

Capacidad cache = $2^7 \cdot 2^{10} \text{ bytes} = 2^{17} \text{ bytes}$.

$$\# \text{blöques} = \frac{2^{17}}{2^2} = 2^{15}.$$

$$\# \text{conjuntos} = \frac{\# \text{blöques}}{\text{asociatividad}} = 2^4.$$

TAG = $32 - 14 - 2 - 0 = 16$.

El número de línea es la dirección del blöque, $32 - 2 = 30$.

$w=2$

TAG	sets	w
-----	------	---

TAG = 16

sets = 14

w = 0

Dirección

Las 4 primeras son el TAG. El resto los pasare a bits para dividir

TAG	SETS	WORD
0284h 10100100100000b	$= 2920h$	$10b = \underline{2h}$
0114h 10001100100010b	$= 2322h$	$01b = \underline{1h}$
0038h 11001111000000b	$= 33C0h$	$00b = \underline{0h}$
0038h 11001111000000	$= 33C0h$	$01b = \underline{1h}$

c) Si pueden estar al mismo tiempo todas, porque caben hasta dos a la vez en el mismo conjunto. \rightarrow distintas

③ 1MByte \Rightarrow Direcciones de 20 bits.

Memoria cache con $64 \cdot 2^{10}$ bytes = 2^{16} bytes.
Bloques de 8 palabras = 2^3 bytes.

$$\# \text{ bloques} = \frac{2^{16}}{2^3} = 2^{13}$$

$$\text{Asociatividad} = \frac{\# \text{ bloques}}{\# \text{ sets}} = \frac{2^{13}}{2^2} = 2^{11}. \text{ LRU.}$$

15	2	3
TAG	sets	W
19	5 4 3 2	0

$$b = 0$$

ABC80h = 1010 1011 1100 1000 0000b
 TAG set block offset.
 Dirección de bloque
 en Mem. princ.

Todos se dividen de esa forma, y se pasa cada campo a hexadecimal. A continuación si ese bloque se haya metido en la caché anteriormente. Al principio del documento están las respuestas finales. Procedimiento:

$$ABC8h = 1010 \ 1011 \ 1100 \ 0000 \ 0001b$$

$$ABC8h = 1010 \ 1011 \ 1100 \ 1000 \ 1000b$$

$$BCD90h = 1011 \ 1100 \ 1101 \ 1001 \ 0000b$$

$$BCD90h = 1011 \ 1100 \ 1101 \ 1001 \ 1010b$$

$$BCD40h = 1011 \ 1100 \ 1101 \ 1010 \ 0000b$$

$$CDE00h = 1100 \ 1101 \ 1110 \ 0000 \ 0000b$$

$$CDE18h = 1100 \ 1101 \ 1110 \ 0001 \ 1000b$$

$$CDE20h = 1100 \ 1101 \ 1110 \ 0010 \ 0000b$$

$$\text{Índice de fallas} = \frac{8}{9} = 0'8.$$

b) Será cero, pues caben hasta 2^8 bloques distintos en cada conjunto, luego todos caben en la caché sin problema.

$$\textcircled{4} \quad NC \ 4kb = 2^2 \cdot 2^{10}b = 2^{12}b.$$

Block offset = 8 bits (2^8 bytes / bloque).

$$\# \text{bloques} = \frac{2^{12}}{2^8} = 2^4.$$

$$\# \text{sets} = \frac{\# \text{bloques}}{\text{asociatividad}} = \frac{2^4}{2^1} = 2^3.$$

Memoria principal con direcciones de 20 bits.

a)	9	3	8	
	TAG	adr	w	
	19	11 10 8 7	0	$b=0$, pues palabras de 1 byte = 8 bits.

En todos los ejercicios será igual, aquí el procedimiento, y a veces el resultado. Si no es así el resultado, es porque está seguro en las capturas del campus virtual.

$$319F0h = 01011\ 01001\ 10011\ 11111\ 00000\ b$$

$$31AFC0h = 01011\ 01001\ 10110\ 11111\ 00000\ b \rightarrow \text{Falla porque va a otro conjunto.}$$

$$7013Ch = 01111\ 01000\ 00011\ 00111\ 11000\ b$$

$$777777h = 01111\ 01111\ 01111\ 10111\ 01111\ b$$

$$44037h = 01000\ 01100\ 00000\ 10011\ 01111\ b$$

$$778DEh = 01111\ 01111\ 10000\ 11101\ 11100\ b$$

$$45021h = 01100\ 01101\ 00000\ 00010\ 00010\ b$$

b) Asociación directa:

8	4	8
TAG	index	w

$b=0$.

No hace falta pasar a binario.

Fallan porque el conjunto y el TAG deben coincidir con alguno que esté en ese momento en la MC (memoria caché), y empieza vacía.

c) Totalmente asociativa:

TAG	w
12	8

index=0, $b=0$.

Todo falla porque no coincide ningún TAG.

d) Organización directa es más rápida porque solo se busca en una línea de la MC.

⑤ $16M\text{bytes} = 2^{24} \text{ bytes} \Rightarrow$ direcciones de 24 bits.

Espacio L1: $2^3 \cdot 2^{16} \text{ bytes} = 2^{19} \text{ b.}$

#sets = 2^7 , asociatividad = 2.

Nº de bloques = #sets · asociatividad = 2^8 .

TAG = Etiq	sets	Block off.
------------	------	------------

$$12 \quad 7 \quad 5 = 24 - 12 - 7.$$

Todos fallan porque el TAG es distinto. ^{y se} Superponer claramente verifica las condiciones de FIFO, pero también podría ser LRU, porque ninguno es usado.

b) Espacio MC: 2^{14} bytes . #sets = 2^7 , asociatividad 4,
Nº de bloques = 2^9 .

El dibujo es igual que en a).

000440h = 0000 0000 0000 | 1010 0100 0000b

004440h = 0000 0000 0100 | 1010 0100 0000b

008440h = 0000 0000 1000 | 1010 0100 0000b

00B440h = 0000 0000 1011 | 1010 0100 0000b

TAG Set

Como la asociatividad es 4, tener 4 vias en el mismo conjunto, luego solo fallarán en la primera iteración.

Índice de acierto total: $\frac{396}{400} = 0'99$.

Ejerc 6.5:

4 veces más ytos (conjuntos) \Rightarrow 2 bits más en el direcccionamiento. Luego:

TAG	sets	bloco
10	9	5

Mirando la conversion a bits del anterior, sacar los resultados.

Averta solo la ultima instrucción porque se queda en un yto aparte. Las otras tres se solapan.

Como averta una de las cuatro en todos las iteraciones menor es primera: indice de avertos = $99/400 = 0.2475$.

⑥ MC: 2^5 bytes , bloques: 2^3 bytes.

$$\# \text{ bloques} = \frac{2^5}{2^3} = 2^2.$$

La linea va de 4 en 4 bytes, luego en cada palabra hay 2^2 bytes.

Directa:

TAG	index	W	b
27	2	1	2

Totalmente
asociativa:

TAG	W	b
29	1	2

Asociativa por
ytos
(estos
pueden ser
 2^2 ytos)

TAG	sets	W	b
28	1	1	2

b) Represento los 8 últimos bits de cada una:

			①	②	①	②	①	②
	0d =	0000	0 000	6	F	F	F	A
El d	4d =	0000	0 100	6	A	A	A	A
es de decimal	8d =	0000	1 000	6	F	A	F	A
	12d =	0000	1 100	6	A	A	A	A
Columna ①: 1 ^{ra} iteración	24d =	0001	1 000	6	F	A	F	A
Columna ②: 2 ^{da} iteración	28d =	0001	1 100	6	A	A	A	A
	32d =	0010	0 000	6	F	F	F	A
	36d =	0010	0 100	6	A	A	A	A

Código A) En la directa, fallos ≈ 0.25 (literaciones $\rightarrow \infty$)

En la de gtes, fallos ≈ 0

En la totalmente asociativa, fallos ≈ 0 .

El mejor código es el de menor asociatividad \Rightarrow

\Rightarrow Asociativa por conjuntos.

Código B) El orden de ejecución sería:

0d =	0000	0 000	6	F	F	F
4d =	0000	0 100	6	A	A	A
8d =	0000	1 000	6	F	F	F
12d =	0000	1 100	6	A	A	A
32d =	0010	0 000	6	F	F	F
36d =	0010	0 100	6	A	A	A
24d =	0001	1 000	6	F	F	F
28d =	0001	1 100	6	A	A	A
32d				A	A	A
36d				A	A	A
24d				A	A	A
28d				A	A	A
32d				A	A	A
:				:	:	:

Donde los colores representan las mismas cachés de antes. Todas tienden a un índice de fallos nulo, luego la mejor es la directa.

$$\textcircled{7} \quad b) \text{ Bloques} = 2^3 \text{ bytes} = 2 \text{ palabras.}$$

$$\text{Cachés: } 128 \text{ bytes} = 2^7 \text{ bytes.}$$

$$\# \text{ Bloques} = \frac{2^7}{2^3} = 2^4.$$

Instrucciones:	TAG	index	w	b
	25	4	1	2

Datos:	TAG	sets	w	b
	27	2	1	2

Instrucciones :	0d =	0....0	0 000	0000b
	4d =		0 000	0100b
	8d =		0 000	1000b
	12d =		0 000	1100b
	16d =		0 001	0000b
	140d =		1 000	1100b
	144d =		1 001	0000b
	148d =		1 001	0100b
	152d =		1 001	1000b
	156d =		1 001	1100b

Datos:	16d =	0...0	0000	0001	0 000b
	208d =		0000	1101	0 000b
	336d =		0001	0101	0 000b
	400d =		0001	1001	0 000b
	80d =		0000	0101	0 000b

⑧ a) $MC = 2^{12}$ bytes, palabras de 4 bytes (MIPS)

Si empiezo con 1 palabra por bloque, en la MC caben 2^{10} bloques. Quedaría:

TAG	index	b
20	10	2

Al ser asignación directa, los últimos 12 bits siempre determinarán a qué bloque de la MC voy, independientemente del tamaño de bloque. Otro ejemplo: con bloques de 2^8 palabras sería:

TAG	w	b
20	10	2

Como los tres últimos dígitos en hexadecimal de las direcciones siempre van a coincidir, se van a ir solapando entre ellos en la MC, luego la tasa de fallos será de 1.

b) Tamaño de bloque: 2^8 bytes. Veamos cuántas iteraciones hace el bucle:

$1000h = 2^{12}d$, como resta de 4 en 4,

$2^{12} - 4n = 0 \Rightarrow n = 2^{10} = 1024$ iteraciones. Como los bloques que se van solapando son 3, con asimetría de 4 se arregla este conflicto.

Sin embargo, siempre fallaremos cuando accedamos a un bloque por primera vez.

TAG	index	block offset
22	2	8

Los dos primeros dígitos en hexadecimal determinan el block offset, luego el bloque en sí, la dirección de bloque, el número de bloque, lo determinan el resto de bits.

La instrucción IEE 51, 2800h (f2) accede a 17 bloques distintos: primera dirección: $2800h + 1000h = 3800h \Rightarrow$ bloque 38. Accede a:

38, 37, 36, 35, 34, 33, 32, 31, 30, 2F, 2E,
2D, 2C, 2B, 2A, 29, 28, que son 17 bloques.

Análogamente, las otras dos instrucciones acceden a 17 bloques cada una también.

Como se comete un fallo por cada vez que accedemos a un bloque (solo la primera vez), entonces el número de fallas sería $17 \cdot 3 = 51$.

⑨ Caber 5 bloques en la MC. (10/2).

Por fijas fallaría la mitad: la primera vez que lee el bloque falla, y la siguiente vez que hace una lectura es para el mismo número del bloque.

⑩ MC: caben 4 bloques. Cada dos palabras, se aumenta en uno la dirección de bloque.

Direcfa:

	index	block o.
	2	1

Por conjuntos: coger la dirección de bloque, y hacer módulo 2 para saber el íte.

Por comodidad, denotare $A[i] \equiv A_i$.

Leyenda: Directa, por bytes, associatividad total.

Dato	dirección (h)	Dir. bloque (h)	yto 4/F					
B9	045	22	2	F	0	F	0	F
A0	000	0	0	F	0	F	0	F
B8	044	22	2	A	0	A	0	A
A1	001	0	0	A	0	A	0	A
B7	043	21	1	F	1	F	0	F
A2	002	1	1	F	1	F	0	F
B6	042	21	1	F	1	A	0	A
A3	003	1	1	F	1	A	0	A
B5	041	20	0	F	0	F	0	F
A4	004	2	2	F	0	F	0	F
B4	040	20	0	A	0	A	0	A
A5	005	2	2	A	0	A	0	A
B3	03F	1F	3	F	1	F	0	F
A6	006	3	3	F	1	F	0	F
B2	03E	1F	3	F	1	A	0	A
A7	007	3	3	F	1	A	0	A
B1	03D	1E	2	F	0	F	0	F
A8	008	4	0	F	0	F	0	F
B0	03C	1E	2	A	0	A	0	A
A9	009	4	0	A	0	A	0	A

⑪ Direcciones de 20 bits.

MC: 2^{16} bytes.

Tamaño de bloque: 2^{12} bytes. } 2^4 bloques.

Directa:

TAG	index	bloco.
-----	-------	--------

4 4 12

Totalmente
asociativa

TAG	blocks
8	12

Asociativa
por 4 bytes

TAG	sets	blocks
6	2	12

a) Está hecho en clase.

b) • Directa: $2^4 \cdot 2^{12} \cdot 8 + 2^4 \cdot 4 = 524352$

\downarrow \downarrow \downarrow
#bloques bytes/byte #bloques · TAG
bytes/ bloque

- Totalmente asociativa: $2^4 \cdot 2^{12} \cdot 8 + 2^4 \cdot 8 = 524416$.
- Asociativa por bytes: $2^4 \cdot 2^{12} \cdot 8 + 2^4 \cdot 6 = 524384$.

(12) Direcciones de 16 bits.

c1: 2^{12} bytes, bloques de 2^8 bytes $\Rightarrow 2^4$ bloques.

c2: 2^{10} bytes, " , 2^3 bloques.

c3: 2^3 bloques también.

c1:

TAG	index	block offset
4	4	8

c2:

TAG	b. o.
8	8

c3:

TAG	set	b. o.
7	1	8

$\#sets = \frac{\#blocks}{ways} = 2$

Los hago directamente porque ya de por si
es larga la relación :)

Los de solo encunciado no los voy a hacer por ahora. Si los hago, estaré subido el día 12/11, así que puedes volver ese día, y mirar al final de este documento.

$$(13) \text{ a) Penalización: } 1 + 20 + 8 \cdot 3 + 2 = 47.$$

$$\text{Ancho de banda: } 32 / 47 = 0'68.$$

$$\text{b) Penalización: } 1 + 20 + 8 \cdot 7 + 2 = 79.$$

$$\text{Ancho de banda: } 64 / 79 = 0'81.$$

$$(14) f = 500 \text{ MHz} = 500 \cdot 10^6 \text{ Hz} \Rightarrow T_c = \frac{1}{f} = 2 \text{ ns}$$

$$\text{AMAT} = \text{Hutime} + \text{Penalty Miss} \cdot \text{Miss Rate} =$$

$$= 1 + 20 \cdot 0'05 = 2 \text{ ns} = 4 \text{ ns}.$$

$$(15) f = 1'2 \text{ GHz} \Rightarrow T_c = \frac{1}{1'2 \cdot 10^9} = 0'833 \text{ ns}.$$

$$\text{CPI}_{\text{base}} = 1'8.$$

$$\text{Ciclos detendr por instrucciones} = 0'02 \cdot 10 \cdot 1000 = 200.$$

$$\text{Ciclos detendr por datos: } 330 \cdot 0'08 \cdot 15 = 396.$$

$$\text{CPI}_y = 1'8 + \frac{596}{1000} = 2'396.$$

$$T_f = \frac{\text{CPI}_y \cdot IC}{f} = 1996'6 \text{ ns}.$$

$$\text{AMAT} = \frac{1000}{1330} (1 + 10 \cdot 0'02) + \frac{330}{1330} (1 + 15 \cdot 0'08) =$$

$$= 1'44812 \text{ ciclos} = 1'2067 \text{ ns}.$$

$$\textcircled{16} \quad \text{CPI}_{\text{efectivo}} = 1'7 + 0'005 \cdot 10 + \\ + 0'015 \cdot 15 + 0'32 = 1'82.$$

- \textcircled{17}
- Saltos condicionales : $3 \cdot 0'3 \cdot 0'2 = 0'18$.
 - Saltos incondicionales : $1 \cdot 0'05 = 0'05$.
 - Load-use : $1 \cdot 0'15 = 0'15$.
 - Penalización IF : $0'006 \cdot 12 = 0'072$.
 - Penalización OF : $0'33 \cdot 0'02 \cdot 15 = 0'099$.

$$\text{Total : CPI}_{\text{ej}} = 1'551.$$

$$\textcircled{18} \quad \text{a) Tasa de fallos global} = 0'08 \cdot 0'05 = \\ = 0'0015 = 0'15\%.$$

$$\text{b) Fallos L1} : 12000 \cdot 0'03 = 360 = \text{ref. L2}.$$

$$\text{Fallos L2} : 360 \cdot 0'05 = 18.$$

$$\text{IAAT} = 1 + 0'03 \cdot (10 + 0'05 \cdot 50) = 1'375.$$

$$\textcircled{19} \quad f = 1'7 \text{ GHz} \Rightarrow T_c = \frac{1}{1'7 \cdot 10^9 \text{ Hz}} = 0'58823 \text{ ns}$$

$$S = 1 + 0'1 \cdot PM_{L1} \Rightarrow PM_{L1} = 40 \text{ ciclos} = 23'53 \text{ ns}$$

$$PM_{L1} = 15 + 0'25 \cdot PM_{L2} \Rightarrow PM_{L2} = 100 \text{ ciclos} = 58'8 \text{ ns}$$