DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
ESCUELA DE INGENIERÍA
PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

IIC2343 - Arquitectura de Computadores (II/2025)

Guía de Ejercicios: Jerarquía de Memoria

Ayudantes: Daniela Ríos (danielaarp@uc.cl), Alberto Maturana (alberto.maturana@uc.cl), Fernanda Escobar (fer_jez2002@uc.cl), Ignacio Gajardo (gajardo.ignacio@uc.cl)

Pregunta 1: I3-2024-1 (Pregunta 1)

(a) En la jerarquía de memoria de un computador, cuenta con una memoria caché de 64KiB, líneas de 16 palabras y tiempo de acceso de 10ns. Por otra parte, cuenta con un *miss penalty* de 100ns. ¿Qué *hit-rate* debe tener la caché para obtener un tiempo de acceso promedio de 20ns en el computador?

Solución: Para obtener el tiempo de acceso promedio en una memoria, se utiliza la siguiente fórmula:

$$TP = HR \times HT + (1 - HR) \times (HT + MP)$$

Siendo TP el tiempo de acceso promedio; HT el *hit-time* promedio; HR el *hit-rate* promedio; y MP el *miss penalty*. Luego, reemplazando valores:

$$20\text{ns} = HR \times 10\text{ns} + (1 - HR) \times (10 + 100)$$
$$20\text{ns} = 110 - 100HR$$
$$HR = \frac{90}{100}$$
$$HR = 0{,}9$$

Se otorgan **0.5 ptos.** por la aplicación correcta de la fórmula y **0.5 ptos.** por el cálculo correcto del *hit-rate* promedio.

(b) Suponga que posee dos computadores con arquitecturas similares, salvo por sus memorias caché: poseen las mismas dimensiones, pero una es 16-*way associative* y la otra es 8-*way associative*. ¿Cuál presenta un *hit-time* promedio menor? Justifique su respuesta.

Solución: Considerando que ambas memorias caché poseen las mismas dimensiones, la que presenta un menor *hit-time* promedio corresponde a la 8-*way associative*. Esto se debe a que la búsqueda en memorias caché N-*way associative* se realiza solo sobre las líneas del conjunto al que se mapea el bloque de la palabra buscada. Como N equivale a la cantidad de líneas por conjunto, en una caché 8-*way associative* la búsqueda se realiza sobre la mitad de las líneas de una 16-*way associative*, lo que implica un menor tiempo para determinar si hay *hit* o no. Se otorgan **0.5 ptos.** por indicar la respuesta correcta y **0.5 ptos.** por justificación.

(c) Suponga que, para una memoria principal de 256 palabras de 1 byte, posee una caché de 16 líneas y 4 palabras por línea con el siguiente estado intermedio:

Línea	Validez	*Tag*	*Timestamp*
0	1	20	1
1	1	5	11
2	1	10	4
3	1	25	9
4	1	31	8
5	1	16	3
6	1	1	6
7	1	8	2

Línea	Validez	*Tag*	*Timestamp*
8	0	10	-
9	0	20	-
10	0	30	-
11	0	5	-
12	0	15	-
13	0	2	-
14	0	1	-
15	0	0	-

La columna "Línea" representa el índice de línea; "Validez" representa el *valid bit*; "*Tag*" representa el valor decimal del *tag* que indica el bloque de memoria almacenado en la línea; y "*Timestamp*" representa el tiempo **desde el último acceso a la línea**. A partir de este estado, se realiza el acceso a memoria de las siguientes direcciones: 0x54, 0xE4, 0xE3, 0xCA, 0xCC, 0xA1. Indique, para cada acceso, si existe un *hit* o *miss*; si corresponde, la línea de la caché que es modificada; y el *hit-rate* para la función de correspondencia *8-way associative*. Asuma una la política de reemplazo LRU. Para responder, complete las tablas adjuntas al enunciado. El *tag* de cada línea lo puede escribir en base binaria o decimal. No necesita indicar el *timestamp* final por línea, pero sí debe considerar su valor en caso de reemplazos.

Solución: Al tener una memoria de 256 palabras, se necesitan $\log_2(256) = 8$ bits para cada dirección. Por otra parte, al ser las líneas de 4 palabras, se requiere $\log_2(4) = 2$ bits de *offset* para ubicar una palabra dentro de una línea. Por otra parte, en una caché 8-*way associative* se tienen conjuntos de 8 líneas, lo que deriva en un total de $\frac{16}{8} = 2$ conjuntos y, de esta forma, $\log_2(2) = 1$ bit de índice de conjunto. Esto, finalmente, resulta en $8 - 1 - 2 = 5$ bits de *tag*. En este caso, el conjunto 0 corresponde a las líneas 0-7 y el conjunto 1 a las líneas 8-15.

Con esto en consideración, se obtiene el siguiente resultado de la secuencia de accesos a memoria:

1. Acceso a dirección 0x54 = 84 = 01010|1|00b. Resulta en *miss* al no existir línea válida con *tag* 01010b = 10 en el conjunto 1. Se copia el bloque sobre la línea 8.
2. Acceso a dirección 0xE4 = 228 = 11100|1|00b. Resulta en *miss* al no existir línea válida con *tag* 11100b = 28 en el conjunto 1. Se copia el bloque sobre la línea 9.

3. Acceso a dirección 0xE3 = 227 = 11100|0|11b. Resulta en *miss* al no existir línea válida con *tag* 11100b = 28 en el conjunto 0. Se copia el bloque sobre la línea 1, al ser esta la de mayor *timestamp*. Ahora, la línea 3 es la que posee mayor *timestamp*, en caso de requerir un reemplazo.

4. Acceso a dirección 0xCA = 202 = 11001|0|10b. Resulta en *hit*, ya que la línea 3 posee *tag* igual a 11001b = 25 en el conjunto 0. Ahora, la línea 4 es la que posee mayor *timestamp*, en caso de requerir un reemplazo.

5. Acceso a dirección 0xCC = 204 = 11001|1|00b. Resulta en *miss* al no existir línea válida con *tag* 11001b = 25 en el conjunto 1. Se copia el bloque sobre la línea 10.

6. Acceso a dirección 0xA1 = 161 = 10100|0|01b. Resulta en *hit*, ya que la línea 0 posee *tag* igual a 10100b = 20 en el conjunto 0.

A continuación, se muestra el estado final de la caché:

Línea	Validez	*Tag*
0	1	20
1	1	28
2	1	10
3	1	25
4	1	31
5	1	16
6	1	1
7	1	8

Línea	Validez	*Tag*
8	1	10
9	1	28
10	1	25
11	0	-
12	0	-
13	0	-
14	0	-
15	0	-

Hit-rate: $\frac{2}{6} = 0.\overline{3} = 33,\overline{3}\,\%$

Se asignan **0.4 ptos.** por obtener el *hit-rate* correcto y **0.6 ptos** por acceso correctamente descrito (*i.e.* obtención de *tag*; determinación de *hit* o *miss* y selección de línea a reemplazar). Si existen errores de arraste, se otorgan como máximo **0.3 ptos.** por acceso si en el desarrollo se evidencia una correcta obtención de bits de *offset*, índice y *tag* por dirección.

Pregunta 2: I3-2024-2 (Pregunta 1)

(a) ¿Qué incidencia tiene la política de reemplazo en una caché con función de correspondencia *directly mapped*? Justifique.

Solución: No tiene ninguna incidencia. En la función de correspondencia *directly mapped*, los reemplazos solo se realizan sobre una línea que ya se encuentra ocupada contenido con un bloque de memoria distinto al que se busca acceder. Por este motivo, no se utiliza ningún criterio de elección de línea a ser reeemplazada. Se otorgan **0.5 ptos.** por responder correctamente; y **0.5 ptos.** por justificación.

(b) ¿Cómo incide el parámetro "N" en el *hit-time* de una memoria caché *N-way associative*?

Solución: El parámetro N corresponde a la cantidad de líneas por conjunto en una caché *N-way associative*. Luego, el *hit-time* en este tipo de caché corresponde al tiempo de búsqueda de un bloque en las líneas del conjunto al que se mapea. Por ende, a mayor N, mayor será el *hit-time* al tener que revisar una mayor cantidad de líneas en el conjunto. Se otorgan **0.5 ptos.** por señalar correctamente la incidencia; y **0.5 ptos.** por la explicación.

(c) Suponga que, para una memoria principal de 4096 palabras de 1 byte, posee una caché de 32 líneas y 16 palabras por línea con el siguiente estado intermedio:

Línea	*Valid*	*Tag*	*Time*
0	1	0	7
1	1	7	11
2	1	10	17
3	0	12	-
4	1	12	15
5	1	15	4
6	0	1	-
7	0	8	-

Línea	*Valid*	*Tag*	*Time*
8	1	5	13
9	1	9	5
10	1	13	9
11	1	11	2
12	1	15	10
13	1	0	12
14	1	1	21
15	1	5	14

Línea	*Valid*	*Tag*	*Time*
16	0	10	-
17	0	15	-
18	0	2	-
19	0	7	-
20	0	9	-
21	0	14	-
22	0	1	-
23	0	2	-

Línea	*Valid*	*Tag*	*Time*
24	1	8	6
25	1	9	16
26	1	14	18
27	1	7	8
28	1	3	3
29	1	6	19
30	1	2	20
31	1	15	1

La columna "Línea" representa el índice de línea; "*Valid*" representa el *valid bit*; "*Tag*" representa el valor decimal del *tag* que indica el bloque de memoria almacenado en la línea; y "*Time*" representa el tiempo **desde el último acceso a la línea**. A partir de este estado, se realiza el acceso a memoria de las siguientes direcciones: 0xC27, 0xFF0, 0x123, 0x782. Indique, para cada acceso, si existe un *hit* o *miss*; si corresponde, la línea de la caché que es modificada; y el *hit-rate* para la función de correspondencia *2-way associative*. Asuma una la política de reemplazo LRU. Para responder, complete las tablas adjuntas al enunciado. El *tag* de cada línea lo puede escribir en base binaria o decimal. No necesita indicar el *timestamp* final por línea, pero sí debe considerar su valor en caso de reemplazos.

Solución: Al tener una memoria de 4096 palabras, se necesitan $\log_2(4096) = 12$ bits para cada dirección. Por otra parte, al ser las líneas de 16 palabras, se requiere $\log_2(16) = 4$ bits de *offset* para ubicar una palabra dentro de una línea. Por otra parte, en una caché *2-way associative* se tienen conjuntos de 2 líneas, lo que deriva en un total de $\frac{32}{2} = 16$ conjuntos y, de esta forma, $\log_2(16) = 4$ bits de índice de conjunto. Esto, finalmente, resulta en $12 - 4 - 4 = 4$ bits de *tag*. Dados estos valores, se puede observar que el dígito hexadecimal menos significativo de cada dirección será el *offset*; el dígito del medio será el identificador de conjunto; y el dígito más significativo será el *tag*.

Con esto en consideración, se obtiene el siguiente resultado de la secuencia de accesos a memoria:

1. Acceso a dirección 0xC27 = 3111 = 1100|0010|0111b. Resulta en *hit*, ya que la línea 4 posee *tag* igual a 1100b = 12 = 0xC en el conjunto 0x2 = 2 (líneas 4 y 5). Ahora, la línea 5 es la que posee mayor *timestamp*, en caso de requerir un reemplazo.

2. Acceso a dirección 0xFF0 = 4080 = 1111|1111|0000b. Resulta en *hit*, ya que la línea 31 posee *tag* igual a 1111b = 15 = 0xF en el conjunto 0xF = 15 (líneas 30

y 31). Ahora, la línea 30 es la que posee mayor *timestamp*, en caso de requerir un reemplazo.

3. Acceso a dirección 0x123 = 291 = 0001|0010|0011b. Resulta en *miss* al no existir línea válida con *tag* 0001b = 1 en el conjunto 0x2 = 2 (líneas 4 y 5). Se copia el bloque sobre la línea 5, al ser esta la de mayor *timestamp*. Ahora, la línea 4 es la que posee mayor *timestamp*, en caso de requerir un reemplazo.

4. Acceso a dirección 0x782 = 1922 = 0111|1000|0010b. Resulta en *miss* al no existir línea válida con *tag* 0111b = 7 en el conjunto 0x8 = 8 (líneas 16 y 17). Se copia el bloque sobre la línea 16, al ser esta la primera disponible.

A continuación, se muestra el estado final de la caché:

Línea	*Valid*	*Tag*
0	1	0
1	1	7
2	1	10
3	0	12
4	1	12
5	1	1
6	0	1
7	0	8

Línea	*Valid*	*Tag*
8	1	5
9	1	9
10	1	13
11	1	11
12	1	15
13	1	0
14	1	1
15	1	5

Línea	*Valid*	*Tag*
16	1	7
17	0	15
18	0	2
19	0	7
20	0	9
21	0	14
22	0	1
23	0	2

Línea	*Valid*	*Tag*
24	1	8
25	1	9
26	1	14
27	1	7
28	1	3
29	1	6
30	1	2
31	1	15

Hit-rate: $\frac{2}{4} = 0{,}5 = 50\,\%$

Se asignan **0.4 ptos.** por obtener el *hit-rate* correcto (ya sea explicitado u obtenido correctamente por cantidad de *hits* y *misses*) y **0.9 ptos** por acceso correctamente descrito (*i.e.* obtención de *tag*; determinación de *hit* o *miss* y selección de línea a reemplazar). Si se reemplaza la línea incorrecta por *timestamp*, se otorgan **0.5 ptos.** en el acceso correspondiente.

Excepcionalmente, se otorgan **3 ptos.** del total si se realiza **todo** el procedimiento y cálculo de *hit-rate* de forma correcta, pero con error de arrastre en el cálculo de bits de *offset*, índice de conjunto o *tag*; o bien si se parte contando del conjunto 1 y no del conjunto 0 al momento de seleccionar las líneas.

Pregunta 3: I2-2022-2 (Pregunta 1)

Considera un computador con una memoria de 1 KiB, organizada en palabras de 4 bytes cada una. Las direcciones generadas por el programa son siempre direcciones de palabras, es decir, la dirección del byte (de la palabra) con la dirección numéricamente menor. Para este computador se está diseñando el sistema de caché, con líneas de cuatro palabras, y se quiere decidir si es mejor un sistema 2-*way set associative* o uno 4-*way set associative*, considerando que en la caché caben 8 líneas en total y con política de escritura *write-through*.

(a) ¿Cuántos bits adicionales—para tags, validez, y offset—necesita cada una de las dos versiones del sistema de caché? Justifica (No consideres la implementación del sistema de reemplazo de líneas).

Solución:

- **2-way set associative:**
  - Se requiere un bit de validez.
  - Como se tienen líneas de 4 palabras, se requieren 2 bits para el offset.
  - Como se tiene una caché de 8 líneas, ordenada en sets de 2 líneas cada uno, tenemos 4 sets. Se necesitan 2 bits para el indicar el set, por lo que se necesitan 4 para el tag.

  La descomposición de una dirección se vería como:

$$\underbrace{0001}_{\text{tag}}\ \underbrace{10}_{\text{índice}}\ \underbrace{01\,00}_{\text{offset}}$$

- **4-way set associative:**
  - Se requiere un bit de validez.
  - Como se tienen líneas de 4 palabras, se requieren 2 bits para el offset.
  - Como se tiene una caché de 8 líneas, ordenada en sets de 4 líneas cada uno, tenemos 2 sets. Se necesita 1 bit para el indicar el set, por lo que se necesitan 5 para el tag.

  La descomposición de una dirección se vería como:

$$\underbrace{00011}_{\text{tag}}\ \underbrace{0}_{\text{índice}}\ \underbrace{01\,00}_{\text{offset}}$$

Considera la siguiente secuencia de direcciones de memoria generadas como consecuencia de la ejecución de un programa:

1. 0x028 (40)	7. 0x084 (132)	13. 0x044 (68)
2. 0x048 (72)	8. 0x008 (8)	14. 0x098 (152)
3. 0x014 (20)	9. 0x064 (100)	15. 0x060 (96)
4. 0x078 (120)	10. 0x058 (88)	16. 0x094 (148)
5. 0x010 (16)	11. 0x0B4 (180)	17. 0x018 (24)
6. 0x070 (112)	12. 0x024 (36)	18. 0x004 (4)

(b) Suponiendo que la estrategia de reemplazo de líneas fuera la ideal (Bélády), es decir, que el computador contara con un dispositivo que pudiera determinar cuál de las líneas en juego en la caché es la que se va a demorar más en volver a ser usada en el futuro, ¿cuál es el *hit rate* para cada una de las dos versiones del sistema de caché?

Solución:

2-way set associative

Acceso (Decimal)	Acceso (Binario)	Indice	Tag	Set 0		Set 1		Set 2		Set 3		Hit	Comentario
				Tag 0	Tag 1	Tag 0	Tag 1	Tag 0	Tag 1	Tag 0	Tag 1		
40	0000101000	10	0000	-	-	-	-	0000	-	-	-	0	Como tiene indice 10, se debe guardar en el set 2.
72	0001001000	00	0001	0001	-	-	-	0000	-	-	-	0	Como tiene indice 00, se debe guardar en el set 0.
20	0000010100	01	0000	0001	-	0000	-	0000	-	-	-	0	Como tiene indice 01, se debe guardar en el set 1.
120	0001111000	11	0001	0001	-	0000	-	0000	-	0001	-	0	Como tiene indice 11, se debe guardar en el set 3.
16	0000010000	01	0000	0001	-	0000	-	0000	-	0001	-	1	Hit!
112	0001110000	11	0001	0001	-	0000	-	0000	-	0001	-	1	Hit!
132	0010000100	00	0010	0001	0010	0000	-	0000	-	0001	-	0	Como tiene indice 00, se debe guardar en el set 0. Toma el segundo espacio.
8	0000001000	00	0000	0001	0000	0000	-	0000	-	0001	-	0	Se reemplaza la linea con tag 0010 porque no se vuelve a usar en el futuro.
100	0001100100	10	0001	0001	0000	0000	-	0000	0001	0001	-	0	Como tiene indice 10, se debe guardar en el set 2. Toma el segundo espacio.
88	0001011000	01	0001	0001	0000	0000	0001	0000	0001	0001	-	0	Como tiene indice 01, se debe guardar en el set 1. Toma el segundo espacio.
180	0010110100	11	0010	0001	0000	0000	0001	0000	0001	0001	0010	0	Como tiene indice 11, se debe guardar en el set 3. Toma el segundo espacio.
36	0000100100	10	0000	0001	0000	0000	0001	0000	0001	0001	0010	1	Hit!
68	0001000100	00	0001	0001	0000	0000	0001	0000	0001	0001	0010	1	Hit!
152	0010011000	01	0010	0001	0000	0000	0010	0000	0001	0001	0010	0	Se reemplaza la linea con tag 0001 porque no se vuelve a usar en el futuro.
96	0001100000	10	0001	0001	0000	0000	0010	0000	0001	0001	0010	1	Hit!
148	0010010100	01	0010	0001	0000	0000	0010	0000	0001	0001	0010	1	Hit!
24	0000011000	01	0000	0001	0000	0000	0010	0000	0001	0001	0010	1	Hit!
4	0000000100	00	0000	0001	0000	0000	0010	0000	0001	0001	0010	1	Hit!

El hit-rate es de $\frac{8}{18}$.

4-way set associative

Acceso (Decimal)	Acceso (Binario	Indice	Tag	Set 0				Set 1				Hit	Comentario
				Tag 0	Tag 1	Tag 2	Tag 3	Tag 0	Tag 1	Tag 2	Tag 3		
40	0000101000	0	00001	00001	-	-	-	-	-	-	-	0	Como tiene indice 0, se debe guardar en el set 0.
72	0001001000	0	00010	00001	00010	-	-	-	-	-	-	0	Como tiene indice 0, se debe guardar en el set 0. En la segunda posicion.
20	0000010100	1	00000	00001	00010	-	-	00000	-	-	-	0	Como tiene indice 1, se debe guardar en el set 1.
120	0001111000	1	00011	00001	00010	-	-	00000	00011	-	-	0	Como tiene indice 1, se debe guardar en el set 1. En la segunda posicion.
16	0000010000	1	00000	00001	00010	-	-	00000	00011	-	-	1	Hit!
112	0001110000	1	00011	00001	00010	-	-	00000	00011	-	-	1	Hit!
132	0010000100	0	00100	00001	00010	00100	-	00000	00011	-	-	0	Como tiene indice 0, se debe guardar en el set 0. En la tercera posicion.
8	0000001000	0	00000	00001	00010	00100	00000	00000	00011	-	-	0	Como tiene indice 0, se debe guardar en el set 0. En la cuarta posicion.
100	0001100100	0	00011	00001	00010	00011	00000	00000	00011	-	-	0	Se reemplaza la linea con tag 00100 porque no se vuelve a usar en el futuro.
88	0001011000	1	00010	00001	00010	00011	00000	00000	00011	00010	-	0	Como tiene indice 1, se debe guardar en el set 1. En la tercera posicion.
180	0010110100	1	00101	00001	00010	00011	00000	00000	00011	00010	00101	0	Como tiene indice 1, se debe guardar en el set 1. En la cuarta posicion.
36	0000100100	0	00001	00001	00010	00011	00000	00000	00011	00010	00101	1	Hit!
68	0001000100	0	00010	00001	00010	00011	00000	00000	00011	00010	00101	1	Hit!
152	0010011000	1	00100	00001	00010	00011	00000	00000	00100	00010	00101	0	Se reemplaza la linea con tag 00011 porque no se vuelve a usar en el futuro. Tambien se pudo haber cambiado la con tag 00010 o 00101
96	0001100000	0	00011	00001	00010	00011	00000	00000	00100	00010	00101	1	Hit!
148	0010010100	1	00100	00001	00010	00011	00000	00000	00100	00010	00101	1	Hit!
24	0000011000	1	00000	00001	00010	00011	00000	00000	00100	00010	00101	1	Hit!
4	0000000100	0	00000	00001	00010	00011	00000	00000	00100	00010	00101	1	Hit

El hit-rate es de $\frac{8}{18}$.