EMA | Jerarquía de Memoria (Práctico - Unidad 5)

Ejercicio 1

Ejercicio 1:
Indicar si son correctas o no las siguientes afirmaciones, si se intenta realizar una comparación entre distintos sistemas de memoria tecnológicamente diferentes. Justificar su respuesta.

a. A menor tiempo de acceso, mayor coste por bit.
b. A menor capacidad, menor coste por bit.
c. A mayor capacidad, menor tiempo de acceso.
d. A mayor tiempo de acceso, mayor capacidad.
e. A mayor frecuencia de acceso, menor capacidad y mayor coste por bit.

(a). T → un ejemplo es claramente la SRAM

(b). F y V → según lo que vimos, una memoria chica implica que va a ser *muy* rápida, lo cual incrementa su costo. Sin embargo, si es chica y lenta, entonces sería súper barata

(c). F → Mientras más grande sea, más lenta va a ser la memoria

(d). T → Por lo general es así. En realidad la verdadera es la recíproca y no esta porque se podría tener una SRAM súper lenta (porque está mal implementada) pero no vamos a considerar esos casos.

(e). T → Definición básica de la caché de un micro

Ejercicio 2

Ejercicio 2:
Considerando un procesador que trabaja a 1.7GHz y los siguientes tiempos de acceso a memoria:

Memory technology	Typical access time	$ per GiB in 2012
SRAM semiconductor memory	0.5–2.5 ns	$500–$1000
DRAM semiconductor memory	50–70 ns	$10–$20
Flash semiconductor memory	5,000–50,000 ns	$0.75–$1.00
Magnetic disk	5,000,000–20,000,000 ns	$0.05–$0.10

¿Cuántos ciclos de clock implica leer un dato de caché (SRAM) y de memoria principal (DRAM)?

Primero, notemos que como trabaja a $1.7GHz$, entonces el ciclo es de $\frac{1}{1.7}ns = 0.59$. Luego, entonces tenemos que

- SRAM → Toma entre 0.85 a 4.24 clks
- DRAM → Toma entre 84.75 a 118.64 clks

Ejercicio 3

Ejercicio 3:
Calcular el tamaño total (en bits) de una caché de mapeo directo de 16KiB (16K x 8 bits) de datos y tamaño de bloque de 4 palabras de 32 bits c/u. Asuma direcciones de 64 bits.

Veamos cada parte:

- Tamaño del bloque de la caché → Son 2^2 palabras, por lo que el tamaño es de 2^(2+2) = 16 bytes = 128 bits
- Cantidad de bloques de la caché → 16KiB / (128 bits) = 1024 bits = 2^10 bits
- Tamaño del índice en bits → 10 bits
- Tamaño de la tag en bits → 64 - 10 - 2 - 2 = 50 bits

Luego, entonces, el número total de bits en esta caché es de 2^10 * (128 + 50 + 1) bits = 183296 bits

Ejercicio 4

Ejercicio 4:
Las memorias caché son fundamentales para elevar el rendimiento de un sistema de memoria jerárquico respecto del procesador. A continuación se da una lista de referencias de acceso a memoria (direcciones de 64 bits) las cuales deben ser consideradas como accesos secuenciales en ese mismo orden. El formato que se utiliza para cada dirección está reducido a sólo 16 bits, solo con fines prácticos:

```
0x000C, 0x02D0, 0x00AC, 0x0008, 0x02FC, 0x0160,
0x02F8, 0x0038, 0x02D4, 0x00AC, 0x00B0, 0x0160
```

Se debe:

a. Para cada una de estas referencias a memoria, determinar el binario de la dirección de cada palabra (cada palabra de 32 bits), la etiqueta (tag), el numero de linea (index) asignado en una cache de mapeo directo, con un tamaño de 16 bloques de 1 palabra c/u. Además liste qué referencias produjeron un acierto (hit) o un fallo (miss) de caché, suponiendo que la cache se inicializa vacía.

b. Para cada una de estas referencias a memoria, determinar el binario de la dirección de cada palabra (cada palabra de 32 bits), la etiqueta (tag), el numero de linea (index) asignado en una cache de mapeo directo, con un tamaño de 8 bloques de 2 palabra c/u. Además liste qué referencias produjeron un acierto (hit) o un fallo (miss) de caché, suponiendo que la cache se inicializa vacía.

Ítem a

Primero, notemos que:

- Como tenemos 16 bloques, entonces notemos que nuestro índice es de 4 bits
- Además, como los bloques son de 1 palabra, no se tiene bit de desplazamiento
- Luego, entonces, la tag tiene 64 - 4 - 0 - 2 = 58 bits

Por ello, entonces tenemos que:

Dirección a memoria	Dirección de cada palabra	Offset	Index	Tag	Resultado
0x000C	0000_0000_0000_1100	00.	0011.	0000_0000_00	miss
0x02D0	0000_0010_1101_0000	00.	0100.	0000_0010_11	miss
0x00AC	0000_0000_1010_1100	00.	1011.	0000_0000_10	miss
0x0008	0000_0000_0000_1000	00.	0010.	0000_0000_00	miss
0x02FC	0000_0010_1111_1100	00.	1111.	0000_0010_11	miss
0x0160	0000_0001_0110_0000	00.	1000.	0000_0001_01	miss
0x02F8	0000_0001_1111_1000	00.	1110.	0000_0001_11	miss
0x0038	0000_0000_0011_1000	00.	1110.	0000_0000_00	miss
0x02D4	0000_0001_1101_0100	00.	0101.	0000_0001_11	miss
0x00AC	0000_0000_1010_1100	00.	1011.	0000_0000_10	hit
0x00B0	0000_0000_1011_0000	00.	1100.	0000_0000_10	miss
0x0160	0000_0001_0110_0000	00.	1000.	0000_0001_01	hit

Ítem b

Del mismo modo que en el ejercicio anterior, notemos que:

- Como tenemos 8 bloques, entonces nuestro índice es de 3 bits
- Como tenemos 2 palabras por cada bloque, entonces tenemos un bit para desplazamiento
- Luego, entonces, tenemos que la tag tiene 64 - 3 - 1 - 2 = 58 bits

Dado esto, la tabla nos queda en este caso como:

Dirección a memoria	Dirección de cada palabra	Offset	Desplazamiento	Index	Tag	Resultado
0x000C	0000_0000_0000_1100	00.	1	001.	0000_0000_00	miss
0x02D0	0000_0010_1101_0000	00.	0	010.	0000_0010_11	miss
0x00AC	0000_0000_1010_1100	00.	1	101.	0000_0000_10	miss
0x0008	0000_0000_0000_1000	00.	0	001.	0000_0000_00	miss
0x02FC	0000_0010_1111_1100	00.	1	111.	0000_0010_11	miss
0x0160	0000_0001_0110_0000	00.	0	100.	0000_0001_01	miss
0x02F8	0000_0001_1111_1000	00.	0	111.	0000_0001_11	miss
0x0038	0000_0000_0011_1000	00.	0	111.	0000_0000_00	miss
0x02D4	0000_0001_1101_0100	00.	1	010.	0000_0001_11	miss
0x00AC	0000_0000_1010_1100	00.	1	101.	0000_0000_10	hit
0x00B0	0000_0000_1011_0000	00.	0	110.	0000_0000_10	miss
0x0160	0000_0001_0110_0000	00.	0	100.	0000_0001_01	hit

Ejercicio 5

Ejercicio 5:
Un computador tiene una memoria principal de 64K bytes y una memoria caché de 1K bits para datos. La memoria caché utiliza correspondencia (mapeo) directa, con un tamaño de línea de 4 palabras de 16 bits c/u. Obtener:

a. ¿Cuántos bits hay en los diferentes campos del formato de dirección de memoria principal?
b. ¿Cuántas líneas contiene la memoria caché?
c. ¿Cuántos bits hay en cada línea de la memoria caché y cómo se dividen según su función?

Ítem a

- Consideramos que tenemos 64K palabras de 1 bytes (i.e., 8 bits) en la memoria principal
- Luego, como tenemos 2^16 palabras, la dirección de la memoria principal es de 16 bits
- Ahora bien, como en memoria tenemos palabras de 1 byte mientras que en caché son de 2 bytes, el *offset byte* contiene 1 bit
- Del mismo modo, como en caché vamos a almacenar 4 palabras, el *offset word* va a ser de 2 bits
- Respecto al índice, como tenemos en caché $\frac{2^{10}}{4 \times 16} = 16 = 2^4$ bloques, entonces va a ser de 4 bits
- Por último, entonces, la *tag* va a contener 16 - 1 - 2 - 4 = 9 bits

tag	index	ow	ob
9	4	2	1

Ítem b

- La caché va a tener $\frac{2^{10}}{4 \times 16} = 16$ bloques / líneas

Ítem c

Respecto a lo mencionado en (a) y (b), llegamos a que la caché tiene:

valid bit	tag	—————	datos	—————	—————
—————	—————	w3	w2	w1	w0
1	9	16	16	16	16

Por lo que la cantidad total de bits es de 74.

Ejercicio 6

Ejercicio 6:
Una caché asociativa por conjuntos consta de 64 líneas, dividida en 4 vías. La memoria principal contiene 4K bloques de 128 palabras cada uno. Muestre el formato de dirección de memoria principal suponiendo que cada palabra es direccionable directamente en memoria.

En este problema tenemos los siguientes datos:

- La memoria principal tiene 4K bloques de 128 palabras cada uno ⇒ 4K * 128 = 2^19 palabras ⇒ 19 bits de address
- Como cada palabra es direccionable directamente en memoria, el *offset byte* es de 0 bits (no tiene)
- Ahora, como tenemos 64 líneas divididas en 4 vías, entonces el *index* va a ser de 4 bits (dado que aplica para una vía en particular)
- Además, como para cada bloque de la memoria principal hay 2^7 palabras, entonces nuestro *offset word* va a tener 7 bits
- Luego, finalmente, tenemos que el *tag* va a contener 19 - 0 - 4 - 7 = 8 bits

El esquema, entonces, nos queda:

tag	index	offset word	offset byte

8	4	7	0

Ejercicio 7

Ejercicio 7:
Considere una CACHE con los siguientes parámetros:

- Criterio de correspondencia: Asociativa por conjuntos
- Asociatividad (N-vias): 2
- Tamaño de bloque: 2 words
- Tamaño de palabra (word): 32 bits
- Tamaño de la cache: 32K words
- Tamaño de dirección: 32 bits
- Cada palabra es directamente direccionable en memoria

Responder:

a. Muestre el formato de dirección de memoria principal.
b. ¿Cuál es el tamaño de toda el área de Tag de la cache, expresada en bits?
c. Suponga que cada LINEA de la cache contiene además un bit de validación (V) y un tiempo de vida de 8 bits. Cual es el tamaño completo (expresado en bits) de un CONJUNTO de la cache, considerando datos, tags y los bits de status antes mencionados?

Ítem a

Notemos que:

- Como cada palabra es directamente direccionable en memoria, entonces *offset byte* tiene 0 bits
- Como cada bloque de la caché tiene 2 palabras, entonces *offset word* tiene 1 bit
- Como la caché tiene 32K palabras y los bloques son de 2 palabras, entonces tenemos en total 16K bloques. Ahora, como consideramos 2 vías, cada “set” va a contener 8K líneas ⇒ El *índice* va a contener 13 bits
- Finalmente, como el tamaño de la dirección es de 32 bits, entonces la *tag* va a tener 32 - 0 - 1 - 13 = 18 bits

Por ello, entonces, tenemos el siguiente esquema:

tag	index	offset word	offset byte
18	13	1	0

Ítem b

- Notemos que el campo *tag* tiene 18 bits en su campo ⇒ Son 18 bits por línea en la caché
- Notemos que como tenemos 16K bloques, entonces vamos a tener 16K líneas en total ⇒ **294912 bits en total**

Ítem c

- Dado que queremos considerar
  - Bit de validación (1 bits)
  - Tiempo de vida (8 bits)

  entonces el **tamaño completo de un CONJUNTO de la caché** es de:

  2 * [1 (valid) + 8 (tiempoVida) + 18 (tag) + 2*32 (cntWords, tamWords)] (cntSets, bitsPorSet) = 2 * 91 bits = 182 bits

Ejercicio 8 (Falta hacer)

Ejercicio 8:
Sea un sistema con una memoria principal de 1M palabras divididas en 4K bloques, donde cada palabra es direccionable directamente en memoria. Definir el formato de la dirección de memoria principal en los siguientes casos, sabiendo que la memoria caché posee 64 líneas:

- **a.** Memoria caché con función de correspondencia directa.
- **b.** Memoria caché con función de correspondencia full-asociativa.
- **c.** Memoria caché con función de correspondencia asociativa de 8 vías.

Ejercicio 9 (Falta hacer)

Ejercicio 9:
Considere que un sistema tiene una CACHE para DATOS de correspondencia ASOCIATIVA POR CONJUNTOS de 2 VÍAS de 256Kbyte y 4 palabras por línea, sobre un procesador de 32bits, CPI = 1, que resuelve todos los data y control hazard sin necesidad de stalls, tiene una memoria principal de 4G palabras de 1 byte cada una.

- **a.** Muestre el formato de memoria principal.
- **b.** Considere ahora la ejecución en este sistema de los siguientes segmentos de código LEGv8 equivalentes para N > 0, donde N→ X3, start = 0x000203C y X6 contiene la dirección base del arreglo A[] del tipo uint32_t. Considerar que el resto de los registros están inicializados en 0 y la CACHÉ vacía al inicio de la ejecución de cada segmento.

Segmento #1:

```
start: ADD X9,X3,XZR
 loop: CBZ X9, end
       LDURH X10, [X6,#0]
       LDURH X11, [X6,#8]
       ADD X12, X10, X11
       LSR X12, X12, #1
       ADDI X6, X6, #4
       SUBI X9, X9, #1
       B loop
  end: …
```

Segmento #2:

```
start: LDURH X10, [X6,#0]
       LDURH X11, [X6,#8]
       ADD X12, X10, X11
       LSR X12, X12, #1
       ADDI X6, X6, #4
       ADDI X9, X9, #1
       SUBS XZR, X9, X3
       B.NE start
  end: ...
```

¿Cuantos MISS de CACHE se produjeron en 5 iteraciones del lazo para el segmento #1 si la dirección base del arreglo A es 0x00001008?

- **c.** Ahora considere que se incorpora una CACHE exclusiva para INSTRUCCIONES de correspondencia FULL ASOCIATIVA de una sola línea de 8 palabras de 32 bits c/u. Considerando que cada MISS de caché supone una penalidad de 5 ciclos de clk. Determine qué segmento de código se ejecuta en menor tiempo para N = 4. Argumente su respuesta indicando los ciclos totales en cada caso. Suponer que el pipeline ya se encuentra en régimen y que la CACHE de DATOS solo produce aciertos en los accesos, por lo que no se tiene penalidades por acceso a datos.

Ejercicio 10 (Falta hacer)

Ejercicio 10:
Encuentre el AMAT (Average Memory Access Time) para un procesador con un tiempo de ciclo de 1 ns, un miss penalty de 20 ciclos de reloj, un miss rate de 5% fallos por instrucción y un tiempo de acceso a caché (incluida la detección de acierto) de 1 ciclo de reloj. Suponga que las penalizaciones por falla de lectura y escritura son las mismas e ignore otros stalls en la escritura.

Ejercicio 11 (Falta hacer)

Ejercicio 11:
Una computadora posee un sistema de memoria jerárquica que consiste en dos cachés separadas (una de instrucciones y una de datos), seguidas por la memoria principal y un procesador ARM que trabaja a 1 GHz, con hit time = 1 ciclo de clock y CPI = 1.

a. La caché de instrucciones es perfecta (siempre acierta), pero la caché de datos tiene un 15% de miss rate. En una falla de caché, el procesador se detiene durante 200 ns para acceder a la memoria principal y luego reanuda el funcionamiento normal. Teniendo en cuenta los errores de caché, ¿cuál es el tiempo promedio de acceso a la memoria?

b. ¿Cuántos ciclos de reloj por instrucción (CPI) se requieren en promedio para instrucciones load y store, considerando las condiciones del punto a)?

c. Asumiendo que la distribución de instrucciones se divide en las siguientes categorías: 25% loads, 10% stores, 13% branches y 52% instrucciones tipo R, ¿cuál es el CPI promedio?.

d. Suponiendo ahora que la caché de instrucciones tampoco es ideal y tiene un miss rate del 10%, ¿cuál es el CPI promedio considerando los fallos de ambas cachés?. Asuma la misma distribución de instrucciones de programa del punto c).

Ejercicio 12 (Falta hacer)

Ejercicio 12:
La siguiente tabla muestra los datos de la caché de un solo nivel (L1) para dos procesadores distintos (P1 y P2). En ambos procesadores el tiempo de acceso a memoria principal es de 70ns y el 36% de las instrucciones acceden a la memoria de datos.

	L1 Size	L1 Miss Rate	L1 Hit Time
P1	2 KiB	8.0%	0.66 ns
P2	4 KiB	6.0%	0.90 ns

a. Asumiendo que el *hit time* de la caché L1 determina el tiempo de ciclo de ambos procesadores, calcule sus respectivas frecuencias de CLK.

b. ¿Cual es el AMAT para ambos procesadores?

c. Asumiendo un CPI de 1 (sin memory stalls) y considerando las penalidades derivadas de instrucciones y datos (mismo Miss Rate), ¿cuál es el CPI total para ambos procesadores? ¿Cuál de los dos procesadores es más rápido?