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

IIC2343 — Arquitectura de Computadores — 1/2018
I2

Martes 8 de Mayo de 2018

La interrogación se compone de 3 preguntas. Debe responder cada una en una hoja separada, aunque puede utilizar más de una hoja para responder a una pregunta. Si no responde a una pregunta, debe entregar una hoja en blanco para ella. Recuerde poner su nombre en todas las hojas de respuesta.

Al momento de entregar, evite doblar bordes o corchetear las hojas, si no que sencillamente indique cuántas hojas son para esa pregunta en caso de utilizar más de una y procure que su hoja esté lo más lisa posible.

1. Responda brevemente las siguientes preguntas

   1.1) ¿Qué consideraciones se debe tener cuando un computador soporta más de una ISA? ¿Para qué se puede hacer esto? ¿Qué partes de la microarquitectura requieren atención? **(1 pt)**
   Solución:
   En primer lugar se debe tener en mente en qué ISA comenzará el computador, ya que usualmente el computador se encuentra con una ISA activa a la vez. Por otro lado, se debe considerar si el soporte es mediante alguna emulación o la microarquitectura soporta nativamente los elementos de esta ISA.
   Sobre el para qué se hace, es posible pensar en el caso de x86 donde la ISA fue progresando junto con los avances para permitir nuevas aplicaciones que utilizaran las nuevas características del hardware, como números más grandes, sin embargo se requería que los programas hechos para los procesadores anteriores pudieran ejecutar sin problemas.
   Finalmente, la unidad de interés en este escenario es la *Unidad de Control* del computador.

   1.2) Suponga que tiene una *caché* con mapeo directo. ¿Se demora más esta en sustituir una línea que una caché *fully associative*? ¿Por qué? **(1 pt)**
   Solución:
   No, se demora menos porque no tiene que aplicar sustitución. El mapeo directo indica de inmediato cuál línea sustituir, mientras que *fully associative* requiere de un protocolo de sustitución, lo cual exige revisar cada entrada para buscar el primero en entrar (FIFO), el menos usado (LFU) o el usado hace más tiempo (LRU).

   1.3) ¿Bajo qué condiciones puede ser conveniente un esquema de *polling* frente a uno de interrupciones, en un contexto de *Input/Output*? **(1 pt)**
   Solución:
   Frente a situaciones en que se sabe que la velocidad de entrada o salida es baja, por ejemplo, en el caso de un teclado o mouse.

   1.4) Al implementar *memory-mapped* I/O en el computador básico, ¿qué buses se deben intervenir y por qué? **(1 pt)**
   Solución:
   En principio se intervendrán los buses Memory Address, W, Data In y Data Out de la memoria, esto ya que entre todos ellos se instalará el *Address-decoder* que dependiendo de la dirección entrante por Address y el estado del bit W, eligirá si traspasar la señal y datos a la RAM o enviarlo a algún dispositivo I/O. Esto se realiza en este nivel ya que permite que el resto de la microarquitectura no tenga necesidad de ser intervenida para funcionar con este esquema de I/O.

   1.5) ¿Cómo se clasifica la ISA x86? Justifique y comente si esto tiene alguna implicancia para la microarquitectura. **(1 pt)**

Solución:
Se clasifica como una ISA *CISC*, es decir de instrucciones complejas. Sobre las implicancias para la microarquitectura, intuitivamente uno podría pensar en que requiere para una correcta implementación a lo menos registros para tomar las funciones de AX, BX, CX, DX, SI, DI, SP y BP.

1.6) ¿Por qué es deseable incluir una jerarquía de *cachés* en un computador? **(1 pt)**
Solución:
Debido a que la memoria es más lenta que los registros (y la caché). También, los principios de localidad espacial y temporal sugieren que es buena idea tener una jerarquía, ya que las mayores capacidades en los niveles superiores pueden compensar la penalidad de *miss* en los niveles inferiores.

2. Suponga que dispone de un computador compatible con la ISA clásica de x86 de 16 bits. En este computador se desea conectar una mesa de control que posee 25 botones y 10 controles que giran y tiene la particularidad de que permite que otros dispositivos compatibles, como un *joystick*, se conecten a esta. Para este dispositivo I/O:

2.1) Entregue un esquema de sus conexiones internas y conexión con el computador, usando *ports*. **(2 pts)**

2.2) Diseñe una tabla de interrupciones y mapeos adecuada. **(2 pts)**

2.3) Escriba una ISR que permita controlar este dispositivo. **(2 pts)**

3. Considere el siguiente registro de accesos de dos procesos en un computador con direcciones virtuales y con uso de **TLB** (*Translation Lookaside Buffer*):

Acceso	Proceso	Dirección Virtual	Página	TLB	Tabla de página
1	2	000101011	1	*Miss*	*Page Fault*
2	1	000111101	1	*Miss*	*Page Fault*
3	1	001101011	3	*Miss*	*Page Fault*
4	1	001011111	2	*Miss*	*Page Fault*
5	1	010011110	4	*Miss*	*Page Fault*
6	1	000110101	1	*Hit*	Página encontrada
7	1	000111011	1	*Hit*	Página encontrada
8	1	001010010	2	*Hit*	Página encontrada
9	1	001111011	3	*Hit*	Página encontrada
10	1	010010101	4	*Hit*	Página encontrada
11	1	010110100	5	*Miss*	*Page Fault*
12	1	001100101	3	*Miss*	Página encontrada
13	1	000111011	1	*Hit*	Página encontrada
14	2	000100101	1	*Miss*	*Page Fault*
15	1	001011101	2	*Miss*	Página encontrada
16	1	001111111	3	*Miss*	Página encontrada

Cuadro 1: Tabla de accesos a memoria.

Basándose en los siguientes datos:

- La memoria física posee 4 bloques.
- El *offset* utiliza 5 bits.
- La TLB y memoria física parten vacías.

Describa la TLB y la memoria virtual del computador. En particular, rellene con la página asociada a cada dirección virtual **(0.5 pts)** y luego responda las siguientes preguntas:

3.1) ¿Cuántas líneas posee la TLB? Justifique **(1 pt)**.
Respuesta:
Tiene 4 líneas la TLB, porque las primeras 4 páginas del programa 1 son *miss*, pero luego, esas 4 páginas son *HIT*. Esto significa que esas 4 líneas están en la TLB, pero cuando se solicita la página 5, es *miss* y después pido la 3 que estaba dentro y fue miss, es decir, fue sustituida esta entrada. Si fue sustituida significa que estaba llena con 4 páginas.

3.2) ¿Cúal es la política de sustitución de la TLB? Justifique **(1.5 pts)**
Respuesta:
Si vemos los hit y miss, en el acceso N°5, la TLB tiene las páginas 1, 2, 3 y 4. Luego llega la página 5, miss y cuando volvemos a preguntar por la página 3, ya no está. Esto implica que hubo un reemplazo de la página 5 por la 3. Si analizamos cada sustitución:

- LRU elimina el 1, porque las últimas páginas vistas son la 2 y 3
- FIFO elimina el 1, porque fue la primera página en entrar en la TLB
- LFU elimina 2, 3 o 4 porque tienen la misma cantidad de accesos

Lo importante es que tanto LRU y FIFO sustituyen la página 1, pero esa página fue Hit en el acceso 13, por lo tanto, se descartan esas 2 sustituciones y queda LFU como política de sustitución.

3.3) ¿Cuántas páginas soporta cada programa? **(0.5pts)**
Respuesta:

Con el offset se puede determinar el tamaño de cada página. Con la dirección más grande virtual se puede determinar el tamaño de la memoria virtual. La cantidad de páginas es la división entre ambos valores. Cada página tiene tamaño 2^5 y la dirección virtual tiene 2^9. Hay $2^9/2^5$ páginas = 16 páginas

3.4) ¿Cúal es el tamaño de la memoria física? Indique en bits o bytes. **(0.5 pts)**
Respuesta:
Con el offset, se puede determinar el tamaño de cada página y la cantidad de bloques de la memoria física ya está dicha. El tamaño de la memoria física es el tamaño de cada página por la cantidad de bloques. Es $2^5 * 4$ = 128 bytes

3.5) ¿Cúal es la política de sustitución en la memoria física? Justifique **(2 pts)**
Respuesta:
LFU.
Si vemos los page fault que provocan sustitución:

- Llega la página 4 del programa 2. El estado de la RAM en ese momento es:
  * El número aumenta el LRU (por cada acceso) para decir que el mínimo es el usado hace más tiempo.

Cuadro 2: RAM cuando llega pág.4 del prog. 2

Programa	Página	FIFO	LRU*	LFU
2	1	0	0	1
1	1	1	1	1
1	3	2	2	1
1	2	3	3	1

  Por los accesos 7,8,9 dicen que los bloque 1, 2 y 3 del programa 1 están en la memoria física. Por lo tanto se fue el bloque 1 del programa 2. Esto no aporta información porque es el primer bloque en llegar, el usado hace más tiempo y presenta misma cantidad de usos que los demás.
- Llega la página 5 del programa 1. El estado de la RAM en ese momento (luego de los accesos 6, 7, 8, 9 y 10) es:

Cuadro 3: RAM cuando llega pág. 5 del prog. 1

Programa	Página	FIFO	LRU*	LFU
1	4	4	9	2
1	1	1	6	3
1	3	2	7	2
1	2	3	8	2

  * El número aumenta el LRU para decir que el mínimo es el usado hace más tiempo.
  Por los accesos 12, 13 y 15, se dice que los bloque 1, 2 y 3 del programa 1 están en la memoria física. Por lo tanto se fue el bloque 4 del programa 1. Es decir, se fue el último bloque que entró y el recién usado. En otras palabras no puede ser FIFO o LRU. Además presenta la mínima cantidad de usos, por lo tanto LFU todavía sigue funcionando, solo que su desempate no es FIFO.
- Llega la página 1 del programa 1. El estado de la RAM en ese momento (después de los accesos 12 y 13) es:
  * El número aumenta el LRU para decir que el mínimo es el usado hace más tiempo.
  Por los accesos 15 y 16 dicen que los bloque 2 y 3 del programa 1 están en la memoria física. Por lo tanto se fue el bloque 1 o 5 del programa 1. Justamente el bloque 5 es el menos usado, por lo tanto LFU es el único algoritmo de sustitución posible en este problema.

Cuadro 4: RAM cuando llega pág. 1 del prog. 1

Programa	bloque	FIFO	LRU*	LFU
1	5	5	10	1
1	1	1	11	4
1	3	2	11	3
1	2	3	8	2

***Hint*:** Se recomienda entender qué pasa en cada acceso.