Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2343 – Arquitectura de Computadores (II/2016)

Solución Interrogación 3

Pregunta 1

a) ¿Cuál es la función del bit de validez en una tabla de páginas? **(1 pto.)**
Solución: El bit de validez en la tabla de páginas es utilizado para determinar si una página está mapeada correctamente a un marco físico. Cuando el bit de validez tiene valor 1 entonces esa entrada en la tabla de páginas es válida, mientras que si su valor es 0, no es válida.

b) Indique bajo que condiciones el uso de memoria virtual podría hacer más lenta la ejecución de los procesos en un computador. **(1 pto.)**
Solución: El uso de memoria virtual puede hacer más lenta la ejecución de procesos cuando ocurren page faults en los siguientes casos:

- No hay marcos físicos disponibles para alguna página de un proceso, se guarda alguna otra página de algún proceso en el disco duro para liberar un marco físico (swap out).
- Se quiere ocupar alguna dirección de memoria cuya página está guardada en disco. Se debe hacer swap out de alguna página para luego hacer swap in de la página que tiene la dirección de memoria solicitada.

c) Analice y comente desde los conceptos de justicia e inanición, el algoritmo *fixed priority pre-emptive scheduling*. **(1 pto.)**
Solución: El algoritmo fixed priority pre-emptive scheduling le otorga a cada proceso una prioridad. Como este algoritmo es pre-emptive, cada proceso es interrumpido para pasar el control al sistema operativo y así realizar un cambio de contexto a otro proceso. Cuando se acaba el tiempo de un proceso o llega otro proceso con mayor prioridad, el proceso actual es interrumpido. Al utilizar la prioridad para ordenar la cola de procesos, se intenta asegurar la justicia sobre el uso de recursos para los procesos en el sistema. Por otro lado, puede ocurrir que los procesos con las prioridades más bajas nunca se lleguen a ejecutar, lo que se conoce como inanición.

d) ¿Por qué no es enmascarable la IRQ0 en un computador x86? **(1 pto.)**
Solución: La IRQ0 corresponde al timer del sistema, que permite realizar cambios de contexto entre procesos (pre-emptive scheduling), por lo que no está permitido enmascarar esta IRQ.

e) ¿Cómo se relaciona el tamaño de las páginas con la efectividad de la TLB? **(1 pto.)**
Solución: Mientras mayor sea el tamaño de la página, mayor porcentaje de los marcos totales del proceso podrá ser almacenado en la TLB, ya que el proceso tendrá menos páginas en total. Por otro lado, el tamaño de cada entrada de la TLB no es afectado por el cambio en el tamaño de la página (la TLB solo guarda marcos asociados a páginas de un proceso). Finalmente, aumentará el hit-rate de la TLB.

f) Durante la mayoría de sus tiempo de ejecución, un proceso tiene el 51 % de sus páginas en el *swap file*. Otro proceso que utiliza la misma cantidad de memoria total, tiene el 49 % de sus páginas en el *swap file*, también durante la mayor parte de su tiempo de ejecución. ¿Asumiendo que ambos se ejecutan durante la misma cantidad de tiempo, cuál de los dos procesos generó más *page faults*? **(1 pto.)**
Solución: Dada la información entregada por el enunciado, no se puede saber cual de los 2 procesos generó más page faults, ya que no se conoce las direcciones de memoria a las que accedió cada proceso. Por ejemplo, puede que el proceso con más páginas en el swap file solo haya utilizado direcciones de memoria que están en la memoria principal, o bien que solo haya utilizado direcciones de memoria cuyas páginas están en el swap file. Se puede apreciar que la respuesta depende exclusivamente de las direcciones de memoria utilizadas por cada proceso.

Pregunta 2

a) Un computador tiene una memoria caché 2-way de 16 líneas de 2 bytes cada una, con política de sustitución LRU. Indique para cada acceso de la siguiente lista, si corresponde a un *hit* o un *miss*: 2, 3, 11, 16, 21, 10, 80, 48, 39, 11, 3, 80. Justifique su respuesta diagramando el estado de la memoria caché al finalizar la secuencia. **(2 ptos.)**
Solución: La secuencia tiene los siguiente hits/misses: M, H, M, M, M, H, M, M, M, H, H, H. El estado de la caché al finalizar la secuencia es el siguiente:

Conj.	Lín. A	Lín. B
0	48 49	80 81
1	2 3	- -
2	20 21	- -
3	38 39	- -
4	- -	- -
5	10 11	- -
6	- -	- -
7	- -	- -

b) ¿Tiene sentido tener una memoria caché más grande que la memoria principal (física)? De ejemplos y justifique su respuesta. **(2 ptos.)**
Solución: No, no tiene sentido si se utiliza un esquema donde la memoria caché se ubique después de la MMU, ya que todas las peticiones de memoria tendrán como límite el tamaño de la memoria principal (CPU sólo conoce tamaño de la memoria principal, no de la caché). Tampoco tiene sentido sacar la memoria principal y dejar la caché como si fuera ahora la principal. Una caché no puede funcionar sin la noción de espacio direccionable virtual o espacio direccionable físico, de la manera en que está construida una CPU y la caché dentro de ella (la CPU no sabe que existe la caché, sólo los espacios direccionables).

Por otro lado, si la caché se ubica antes de la MMU, entonces una caché mayor a la memoria principal traería ventajas, ya que podría almacenar el contenido de marcos que se encuentran en el swap-file.

c) Un computador con espacio direccionable de 128 bits tiene una caché *fully-associative* con líneas de 16 bytes. ¿A qué porcentaje de la capacidad total de la caché para almacenar datos corresponde el espacio utilizado por los tags? **(2 ptos.)**
Solución: Por cada línea se necesita unicamente un tag, si se calcula el porcentaje para una línea, este es válido para toda la caché. Si cada línea tiene 16 palabras, el tag tendrá un tamaño de $128 - 4 = 124$ bits. Dado esto, el porcentaje utilizado por los tags sera $\frac{124}{16 \times 8} = \frac{124}{128} = \frac{31}{32} \approx 0{,}97$.

Pregunta 3

a) En promedio, ¿cuántos ciclos por salto pierde un computador con un pipeline de 12 etapas, si su unidad predictora acierta el 75 % de las veces? Asuma que los saltos se realizan en la penúltima etapa. **(1 pto.)**
Solución: Dado que los saltos se realizan en la etapa 11, cada vez que haya un error en la predicción, ser perderán 10 ciclos. Luego, en promedio, el computador perderá $10 \times 0{,}25 = 2{,}5$ ciclos por salto.

b) En caso que la única solución para solucionar un *hazard* de datos sea introducir una burbuja, ¿cuál es el momento más adecuado para hacerlo? **(1 pto.)**
Solución: La idea central es introducir lo antes posible la burbuja para evitar posibles modificaciones a la memoria, que la dejen en un estado inconsistente y que después haya que eliminar. El momento ideal es apenas se conocen las señales de control, o sea, en la etapa ID, justo después de pasar por la unidad de control.

c) Considere el siguiente código escrito para el assembly del computador básico con *pipeline*:

```
ADD A, B
MOV (var0), A
MOV (var1), B
```

a) Indique con un diagrama los *hazards* que se generan al ejecutar el código anterior. **(1 pto.)**
Solución: Hay un hazard entre las líneas 1 y 2 ya que el valor de A no ha alcanzado a hacer WB cuando se requeire su valor para la instrucción siguiente.

b) Indique que sucede al intercambiar el orden de la segunda y tercera línea del código. ¿Cambia el resultado final? En caso de existir *hazards*, indíquelos con un diagrama. **(1 pto.)**
Solución: Hay un hazard entre las líneas 1 y 3, ya que se requiere el valor de A que aún no ha hecho WB cuando pasa la línea 3 por EX.

c) Para los dos casos anteriores, en caso de existir *hazards*, indique como los detectan las *forwarding units* correspondientes, especificando las señales de control participantes. **(2 ptos.)**
Solución: En el primer caso, la FU1 considera las señales ID/EX.AluA, MEM/WB.LoadB que indican que se requiere el valor de A antes que haya hecho WB. Para el segundo caso, la FU2 deberá leer MEM/WB.LoadA, EX/MEM.MemIn y MEM/WB.RegIn. El primer par de señales identifica al Hazard y la tercera indica que la FU2 puede subir la señal FowardDin para hacer el forward que corresponde.