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

IIC2343 – Arquitectura de Computadores (II/2018)

Tarea 11

Fecha de entrega: jueves 29 de noviembre de 2018 a las 23:59 horas

Pregunta 1

En un computador x86 de 16 bits se desea implementar un mecanismo que permita al sistema operativo terminar programas que llevan mucho tiempo en ejecución, e informarle al usuario de esto mediante un mensaje en la pantalla. Para esto, se considera la siguiente tabla que indica los vectores de interrupción:

Dirección del vector	Tipo	Función asociada
00-01	Excepción	*Exception handlers*
02	Excepción	Usada para errores críticos del sistema, no enmascarable
03-07	Excepción	*Exception handlers*
08	IRQ0	*Timer del sistema*
09	IRQ1	Puerto PS/2: Teclado
0A	IRQ2	Conectada al `PIC` esclavo
0B	IRQ3	Puerto serial
0C	IRQ4	Puerto serial
0D	IRQ5	Puerto paralelo
0E	IRQ6	*Floppy disk*
0F	IRQ7	Puerto paralelo
10	Int. de *Software*	Funciones de video
11-6F	Int. de Software	Funciones varias
70	IRQ8	*Real time clock* (`RTC`)
71 - 73	IRQ9-11	No tienen asociación estándar, libre uso
74	IRQ12	Puerto PS/2: *Mouse*
75	IRQ13	Coprocesador matemático
76	IRQ14	Controlador de disco 1
77	IRQ15	Controlador de disco 2
78-FF	Int. de *Software*	Funciones varias

a) ¿Qué dispositivos de I/O de los descritos en la tabla utilizaría para implementar el mecanismo? Indique qué función cumpliría cada uno. **(2 ptos.)**

b) Describa a alto nivel el procedimiento completo que realiza el mecanismo. **(2 ptos.)**

c) Para cada uno de los dispositivos utilizados, indique en detalle cómo sería la interacción con él (tipo de comunicación, instrucciones, modelo de interacción, función de interrupción), desde que comienza su participación en el proceso. **(2 ptos.)**

Pregunta 2

a) ¿Tiene sentido utilizar una memoria *caché* de tamaño superior al de la memoria principal? ¿Cambia el análisis si el tamaño es igual al de la memoria direccionable? **(1.5 ptos.)**

b) Considere una memoria *caché fully-associative*, con *hit-time* igual a $12L^3 - 81L + 68$ [ns], donde L es la cantidad de líneas de la *caché*. Si el *hit-rate* de esta memoria es de 0.95, ¿cuál es la cantidad de líneas que genera el tiempo de acceso promedio mínimo? **(1.5 ptos.)**

c) La contención de bloques es un problema del esquema de mapeo directo, donde dos o más bloques pelean por la misma línea, existiendo otras líneas no utilizadas en la *caché*. ¿Existe un problema similar en el esquema *N-way*? Si su respuesta es negativa, justifiquela, y si es positiva, indique detalladamente un caso en que esto se de. **(1.5 ptos.)**

d) Suponga que está escribiendo una subrutina que realiza el producto punto entre dos matrices de $N \times N$. Explique detalladamente qué consideraciones se deben tomar para maximizar el *hit-rate*. **(1.5 ptos.)**
Nota: Con detalladamente, se espera que otra persona sea capaz de replicar los resultados.

Pregunta 3

¿Qué problema podría existir en un sistema operativo con soporte para memoria virtual, si **muchos procesos** se encuentran en ejecución **simultánea**? En caso de existir uno, indique cómo solucionarlo desde el punto de vista del uso de memoria, detallando claramente los elementos y estructuras involucradas. En caso de no existir problema, justifique el por qué. **(6 ptos.)**

Pregunta 4

a) ¿Qué desventajas tiene el uso de un *pipeline* muy profundo en un computador? **(1 pto.)**

b) Comente brevemente en qué consisten las vulnerabilidades *Spectre* y *Meltdown* y por qué no se pueden solucionar en los procesadores actuales. **(1 pto.)**

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

```
DATA
    n 2
CODE
    main:
        MOV A,0
        MOV B,(n)
        JEQ end
        SUB B,1
        MOV (n),B
        JMP main
    end:
```

Indique con un diagrama, si y cuando ocurre *forwarding*, *stalling* y *flushing* al ejecutar el código anterior y cómo los detectan las *forwarding units* correspondientes, especificando las señales de control participantes. Considere que el *pipeline* tiene *forwarding* entre todas sus etapas, el manejo de *stalling* es por *software* (instrucción `NOP`) y predicción de salto asumiendo que no ocurre. **(4 ptos.)**

Pregunta 5

A partir del protocolo `MESI`, haga una extensión para crear el protocolo `MOESIF`. En este protocolo, el estado `O` significa "*Owned*", es decir, que una *caché* es "dueña" de la línea. Por otra parte, `F` significa *Forward*, el que indica que uno, y solo uno de los controladores de *caché* será el encargado de compartir una línea compartida con el resto de las *cachés* **siempre que posea el mismo valor que en la memoria principal**. Para ello, debe tener las siguientes consideraciones:

- Para este protocolo, cuando una línea en estado `E` es solicitada por otra *caché*, la original la marca en estado `F` y las copias se realizan con estado `S`. La copia se hace entre *cachés*.
- Cuando una línea en estado `F` o `S` es cambiada, la *caché* que realiza el cambio solicita cambiar el estado a `O` e invalida a todas las otras. Luego, en el siguiente acceso, la *caché* que tiene la línea en estado `O` envía directamente la línea a las otras *cachés*, que la marcarán con estado `S`.
- Si una *caché* tiene una línea en estado `M` y esta es solicitada por otra *caché*, esta marca su línea en estado `O` y envía ella directamente el dato a la nueva *caché* que lo marcará como `S`.

Sea lo más claro posible con el funcionamiento del protocolo a partir de la descripción anterior, incluyendo las acciones realizadas por todas las partes involucradas. **(6 ptos.)**

Entrega y evaluación

La tarea se debe realizar de **manera individual** y la entrega se realizará a través de GitHub. El formato de entrega debe consistir en un archivo llamado `tarea11.tex`, junto con los recursos necesarios para que este pueda ser compilado correctamente. Archivos que no compilen o no cumplir este formato de entrega implicará nota **1.0** en la tarea. En caso de atraso, se aplicará un descuento de **1.0** punto por cada 6 horas o fracción.

Política de Integridad Académica

Los alumnos de la Escuela de Ingeniería deben mantener un comportamiento acorde al Código de Honor de la Universidad:

"Como miembro de la comunidad de la Pontificia Universidad Católica de Chile me comprometo a respetar los principios y normativas que la rigen. Asimismo, prometo actuar con rectitud y honestidad en las relaciones con los demás integrantes de la comunidad y en la realización de todo trabajo, particularmente en aquellas actividades vinculadas a la docencia, el aprendizaje y la creación, difusión y transferencia del conocimiento. Además, velaré por la integridad de las personas y cuidaré los bienes de la Universidad."

En particular, se espera que mantengan altos estándares de honestidad académica. Cualquier acto deshonesto o fraude académico está prohibido; los alumnos que incurran en este tipo de acciones se exponen a un procedimiento sumario. Específicamente, para los cursos del Departamento de Ciencia de la Computación, rige obligatoriamente la siguiente política de integridad académica. Todo trabajo presentado por un alumno (grupo) para los efectos de la evaluación de un curso debe ser hecho individualmente por el alumno (grupo), sin apoyo en material de terceros. Por "trabajo" se entiende en general las interrogaciones escritas, las tareas de programación u otras, los trabajos de laboratorio, los proyectos, el examen, entre otros. Si un alumno (grupo) copia un trabajo, los antecedentes serán enviados a la Dirección de Docencia de la Escuela de Ingeniería para evaluar posteriores sanciones en conjunto con la Universidad, las que pueden incluir reprobación del curso y un procedimiento sumario. Por "copia" se entiende incluir en el trabajo presentado como propio partes hechas por otra persona. Está permitido usar material disponible públicamente, por ejemplo, libros o contenidos tomados de Internet, siempre y cuando se incluya la cita correspondiente.