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

IIC2343 – Arquitectura de Computadores (I/2015)

Solución Interrogación 3

Pregunta 1

a) ¿Cómo es el rendimiento de una memoria caché, si el patrón de accesos a memoria distribuye de manera uniforme sobre todas las posibles direcciones? Ejemplifique el o los posibles casos. **(1 pto.)**
Solución: El rendimiento dependerá del patrón temporal de accesos. Si los accesos se realizan linealmente, el *hit-rate* será muy alto, ya que sólo existirán *misses* al comenzar a leer un bloque. Por otro lado, si los accesos se realizan de manera que se elija una palabra de un bloque que no está en la caché, el *hit-rate* será cercano a cero.

b) El *orden de booteo* de un computador es una preferencia que se modifica generalmente en el BIOS. Sin embargo, estas propiedades pueden modificarse, lo que implica que no se guardan en la ROM donde está el BIOS, y se mantienen incluso cuando el computador se apaga. ¿Cómo y donde se pueden almacenar estos datos? **(1 pto.)**
Solución: En general, para guardar estas preferencias, se utilizan una pequeña cantidad de memoria volátil, que es alimentada por una pila de bajo consumo, que se encuentra en la tarjeta madre. Otra posible solución es almacenarlas en un sector predeterminado de un dispositivo de almacenamiento no volátil, como un disco duro.

c) Un computador de 64 bits tiene una memoria caché de 32KB, con 1024 líneas de 32 palabras. ¿Cuanto espacio de esta caché es usado por información distinta de los datos? **(1 pto.)**
Solución: La solución de este ejercicio depende de que función de correspondencia se ocupe. Si usamos *4-way associative*, necesitamos 5 bits para el offset y 8 bits para el conjunto. Luego, se requieren 49 bits para el tag. Si asumimos que hay un tag y un bit de validez por línea, necesitamos en total $50 \times 1024 bits = 50Kb = 6,25KB$, lo que equivale a casi un 20 % de la capacidad de la caché para almacenar datos.

d) ¿Qué es y cuál es la función del *Address Decoder*? **(1 pto.)**
Solución: El *Address Decoder* es una pieza de *hardware* que permite el mapeo a memoria de dispositivos de I/O. Transforma las direcciones mapeadas, a las direcciones de los dispositivos.

e) Describa una política de reemplazo óptima para memorias caché. **(1 pto.)**
Solución: El criterio óptimo para sustituir un bloque, es elegir aquel que no vuelva a ser requerido en la mayor cantidad de tiempo. Esta política es llamada el algoritmo de Bélády. Cualquier solución que implique la selección de un bloque de esas características, es correcta.

f) Un disco duro SSD puede alcanzar velocidades de transferencia de hasta 600MB/s. Donde debería conectarse idealmente un disco de este tipo en un computador x86? **(1 pto.)**
Solución: Debería conectarse al bus PCI Express, idealmente a una entrada mayor o igual a 3x.

Pregunta 2

Un computador compatible con el assembly x86, tiene conectado un controlador de DMA mediante los siguientes puertos:

Puerto	Función
33	Comandos
34-35	Origen
36-37	Destino
38-39	Cont. palabras

Este controlador inicia la copia cuando recibe su registro de comandos la palabra 0x01 y puede utilizarse sólo mediante la interrupción de software 22, que está mapeada a la cuarta palabra del vector de interrupciones.

1. Asumiendo que el vector de interrupciones comienza en la dirección 0x1000, escriba un programa que registre en el sistema una subrutina con *label* ISR_DMA, como la ISR de la interrupción de software para llamar al controlador de DMA. Antes de realizar este proceso, respalde la ISR anterior. **(1 pto.)**
   Solución:

```
MOV AX, [0x1003]
PUSH AX
MOV [0x1003], ISR_DMA
```

2. Escriba el código de la subrutina ISR_DMA. Defina claramente la convención de llamada que usará la ISR. **(2 ptos.)**
   Solución: Asumimos que la convención de llamada para esta interrupción de software, consiste en almacenar en AX la dirección de origen, en BX la dirección de destino y en CX la cantidad de palabras.

```
OUT 34, AL
OUT 35, AH
OUT 36, BL
OUT 37, BH
OUT 38, CL
OUT 39, CH
OUT 33, 0x01
```

3. Asuma ahora que al computador se conecta una unidad de almacenamiento externo de sólo lectura, a la cual se accede mediante su buffer interno, *i.e.*, el dispositivo copia en su buffer de manera instantánea, la cantidad de palabras indicada en su registro de parámetros, desde su dirección interna indicada en su registro de direcciones, cuando se ingresa el comando 0x10 en su registro de comandos. Teniendo esto en consideración, el computador expone el siguiente mapa de memoria:

Dirección	**Función asociada**
0-6	Exception handlers
7	Registro de comandos de dispositivo de almacenamiento externo
8	Registro de direcciones de dispositivo de almacenamiento externo
9	Registro de parámetros de dispositivo de almacenamiento externo
10-2057	Buffer del dispositivo de almacenamiento externo
2058-5119	Espacio reservado del sistema
5120-65535	Espacio de memoria de libre disposición

Asumiendo que puede realizar llamados del tipo CALL Reg, donde Reg es un registro que almacena una dirección de memoria, escriba un programa que ejecute un archivo de 10KB, ubicado en la dirección 0x7A12 del dispositivo de almacenamiento. **(3 ptos.)**

Solución: En estricto rigor, el enunciado tiene un error al asignar sólo una palabra del mapeo para el registro de direcciones y una para el registro de parámetros. Como solución, se asume que es posible asignar palabras de 16 bits a esas direcciones.

```
    MOV DX, 0x0800 ; 0x0800 = 2048
    MOV [0x0008], 0x7A12
    MOV [0x0009], DX
    MOV AX, 0x000A
    MOV BX, 0x1400 ; 0x1400 = 5120
    MOV CX, DX
    MOV SI, 0
  start:
    CMP SI, 5
    JE end
    MOV [0x0007], 0x10
    INT 22
    ADD [0x0008], DX
    ADD BX, DX
    ADD SI, 1
    JMP start
  end:
    MOV AX, 0x1400
    CALL AX
```

Pregunta 3

a) Una memoria caché es inclusiva, cuando todos los bloque contenidos en un nivel, también están contenidos en los inferiores. Por otro lado, una caché es exclusiva, cuando un bloque no se repite en ninguno de sus niveles.

I. En una CPU con 2 niveles de caché, indique cuándo es conveniente tener una caché inclusiva y cuando una exclusiva. **(1 pto.)**
Solución: Si el tamaño de los dos niveles de caché es similar, conviene tener una caché exclusiva, para no tener bloques duplicados y desaprovechar espacio. Por otro lado, si el segundo nivel es bastante más grande que el primer nivel, conviene tener una caché inclusiva.

II. Asuma ahora que se tiene un sistema con 2 procesadores, cada uno con una caché independiente. Una de los procesadores quiere saber si el otro procesador tiene un dato en la caché ¿Cuál de estos dos tipos de caché permite que esta búsqueda sea más rápida? **(1 pto.)**
Solución: La caché inclusiva permite una búsqueda más rápida, ya que sólo es necesario buscar en el segundo nivel, debido a que si no está ahí, tampoco puede estar en el primer nivel. Por otro lado, si la caché es exclusiva, es necesario buscar en ambos niveles.

b) Considere un computador con microarquitectura Von Neumann, donde la tasa de ciclos de clock por instrucción es igual a N, cuando todos los accesos a memoria producen hits en la caché. La memoria caché tiene miss-rate de 4 % y miss-penalty de 25*N ciclos de clock. Si en un programa de K instrucciones, el 50 % de éstas realizan lectura de un dato en memoria, ¿cuántos ciclos de clock menos tomaría la ejecución del programa, si todas las instrucciones produjeran hits en la memoria caché? **(2 ptos.)**
Solución: Como el programa tiene K instrucciones, sabemos que cuando el hit-rate es 100 %, el programa demorará K*N ciclos de clock. Dado que en la arquitectura Von Neumann todas las instrucciones requieren al menos un acceso a memoria (fetch), si nos ponemos en el caso donde el miss-rate es 4 %, se tiene que 0,04*K instrucciones generarán un miss en la memoria caché en la etapa fetch. Además de esto, se sabe que el 50 % de las instrucciones son de escritura o lectura, por lo tanto, otras 0,5*0,04*K instrucciones generarán un miss cuando quieran leer datos. Sumando, tenemos 0,06*K instrucciones que generan miss, por lo que el tiempo total será de K*N+0,06*25*K*N =2,5*K*N ciclos de clock. De esta manera, la ejecución cuando el hit-rate es 100 % tomará 1,5*K*N ciclos de clock menos.

c) Considere el siguiente código que almacena en el vector c el producto entre una matriz A y un vector b:

```
for (i=0; i<96; i++ )
{
  for (j=0; j<96; j++ )
  {
    c[i] = c[i]+ A[i][j] * b[j];
  }
}
```

Asuma que la matriz A está almacenada en orden de filas y que utiliza una memoria caché de 4KB con mapeo directo para ella sola, *i.e.*, b y c usan una memoria caché distinta a esta. Asumiendo que A[0][0] es mapeado a la primer línea de la caché, y que el tamaño de un bloque es de 64 bytes, ¿Cuáles son los valores para **i** y **j**, para los cuales A[i][j] sustituirá a A[0][0] en la primera línea de la caché? **(2 ptos.)**
Solución: La respuesta para esta pregunta depende de el tipo de dato que se asuma para la matriz A. En este caso, si asumimos que A es del tipo double, *i.e.* 64 bits por elemento, en cada línea de la caché podrán almacenarse 8 elementos de la matriz A, por ejemplo, A[0][0], A[0][1] ... A[0][7] estarán en la misma línea. Esto significa que una fila de A utiliza 96/8 = 12 líneas de caché. Como la caché para A tiene 4096 / 64 = 64 líneas, 5 filas completas más 4*8 elementos de la fila siguiente de A la llenarán completamente. Luego, el elemento que sustituye a A[0][0] es A[5][32].