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

IIC2343 – Arquitectura de Computadores (I/2012)

Interrogación 3

Pregunta 1

a) ¿Que problema de consistencia en la escritura en memoria podría ocurrir si la memoria caché se ubicara antes de la MMU, i.e. la memoria caché trabaja con direcciones virtuales? **(1 pto.)**

Solución: Si la caché se ubica antes de la MMU, entonces este trabaja con direcciones virtuales. De esta manera, en un instante dado, en la caché pueden existir dos bloques asociados a paginas virtuales diferentes, pero que apuntan al mismo bloque. Así, si se escribe en uno de estos bloques, las páginas tendrán distinto contenido, a pesar de estar mapeadas al mismo bloque.

b) Si dos procesos comparten una porción de memoria menor al tamaño de un marco, ¿existe algún problema si uno de los procesos escribe en un porción no compartida del mismo marco? **(1 pto.)**

Solución: Dado que un marco no puede ser subdividido, cualquier escritura en el será visible para los dos procesos, sin importar si esta fue dentro del área compartido o no.

c) ¿Es posible reutilizar el contenido de la TLB cuando ocurre un cambio de contexto? **(1 pto.)**

Solución: Dado que las asignaciones en la TLB, al hacer un cambio de contexto, no necesariamente corresponden a la tabla de páginas del proceso que se encuentra actualmente ejecutándose, la TLB debe ser borrada para que se llene con mapeos válidos.

d) Asumiendo que la caché en el primer nivel de una jerarquía de memoria es muy rápida y muy pequeña, ¿que función de correspondencia recomendaría en este caso? **(1 pto.)**

Solución: Se recomendaría usar una caché fully associative.

e) Un programa necesita copiar a otra posición de memoria (duplicar) una matriz que pesa 1 GB. ¿Como debería realizar este proceso el programa? **(1 pto.)**

Solución: El proceso debería utilizar DMA, ya que la memoria, por defecto, está mapeada a memoria. De esta manera, la CPU no debe preocuparse de la transferencia.

f) Un computador con microarquitectura avanzada realiza la búsqueda en la TLB al mismo tiempo que la búsqueda en el caché, en vez de hacerlo de manera secuencial, i.e., primero en la TLB y

luego en la caché. ¿Que relación debe cumplirse, con respecto al tamaño de página y la cantidad de bits necesaria para encontrar un bloque en la caché, para que este esquema presente una mejora en el rendimiento? **(1 pto.)**

Solución: Debe cumplirse que la cantidad de bits necesarias para buscar una palabra dentro de una página sea la misma que la cantidad de bits necesaria para realizar una búsqueda en la caché (sin considerar el tag). De esta manera, al hacer la transformación virtual-física, el resultado de esta que no corresponde al offset, se utiliza para verificar si el tag del bloque obtenido en el caché coincide.

Pregunta 2

Considere un computador de 32 bits, con páginas de memoria de 4K palabras y memoria virtual de dos niveles. El primer nivel consiste en un diccionario de tablas de páginas, mientras que el segundo contiene las tablas. Este esquema permite disminuir el espacio usado por las tablas de páginas, ya que la única que debe estar siempre en memoria es la correspondiente al diccionario.

a) Describa la subdivisión de una dirección de memoria virtual con este esquema. **(2 ptos.)**

Solución: Dado que el diccionario siempre debe estar en memoria, este deberá estar almacenado en una página de memoria. Luego, este puede tener un tamaño máximo de 4K entradas por lo que se necesitan 12 bits para especificar una entrada en el diccionario. Cómo los márcos son de 4K palabras, el offset también debe ser de 12 bits. Finalmente, dado que el computador es de 32 bits, la tabla de páginas sólo puede tener 2^8 entradas. Gráficamente, la división queda de la siguiente forma:

Índice en Diccionario	Índice en Tabla de Páginas (Página)	Offset dentro del Marco
12 bits	8 bits	12 bits

b) Explique los pasos que deben realizarse para traducir una dirección de memoria virtual a una física (no tome en cuenta los posibles accesos a disco). **(2 ptos.)**

Solución: El primer paso consiste en buscar en el diccionario, utilizando los 12 bits más significativos de la dirección virtual, la tabla de páginas asignada a ese índice. Una vez obtenida identificada la página, se obtiene esta desde la memoria. Cabe destacar que en cada marco de memoria caben 2^4 tablas de páginas. Luego, se busca dentro de la página el marco correspondiente a la página indicada por el índice ubicado en los bits [12-19] de la dirección virtual. Finalmente, a la dirección del marco obtenido en el paso anterior se le concatena el offset que se ubica en los 12 bits menos significativos de la dirección virtual, con lo que se obtiene la dirección física.

c) ¿Cuantas operaciones de memoria son necesarias para leer o escribir una palabra? **(1 pto.)**

Solución: Se necesitan 3 operaciones. Una para leer el diccionario, otra para la tabla de paginas y una final para el marco.

d) ¿Cuanta memoria física necesita un proceso que utiliza una página de memoria virtual? **(1 pto.)**

Solución: Si se necesita una sola página de memoria, se requiere tener un diccionario, una tabla y un marco. Luego, para el diccionario se necesitan 4KB (1 marco-página), mientras que para una tabla se necesitan 2^8B= 256B y para un marco 4KB. Así, para un proceso de una sóla página, se necesitan 4096B+4096B+256B= 8448B.

Pregunta 3

a) Un computador tiene 16 MB de memoria direccionable y una caché de 128 KB, con función de correspondencia 2-way associative y política de sustitución LRU, y con bloques de 16 palabras de 1 byte. Asuma que la caché está vacia y que se ejecuta un programa con la siguiente secuencia de accesos a direcciones de memoria: 3C2B20h, 3C2324h, 172B24h, 3C2B28h, 182B27h, 258032h, B22328h, 172328h. Escriba para cada uno de los accesos su correspondiente tag, conjunto y bloque en la caché. **(3 ptos.)**

Solución:
Tamaño de la caché = 128 KB = $16 \times 2 \times$N, luego son N = 4096 sets conjuntos.
Bits por dirección = $\log_2(16\text{MB}) = \log_2(2^{24}) = 24$
Bits de direccionamiento por bloque = $\log_2(16) = 4$, luego se usan los 4 bits menos significativos para direccionar una palabra dentro de un bloque.
Bits de direccionamiento para conjuntos = $\log_2(4096) = 12$, por lo tanto se utilizan los 12 bits a continuación de los 4 bits menos significativos para direccionar un conjunto.
Bits de direccionamiento para tag = 24–4–12 = 8
Accesos:

1. **3C2B20h**: Se ubica en el bloque 0 del conjunto 2B2h, con tag 3Ch.
2. **3C2324h**: Se ubica en el bloque 0 del conjunto 232h, con tag 3Ch.
3. **172B24h**: Se ubica en el bloque 1 del conjunto 2B2h, con tag 17h (bloque 0 del conjunto ya estaba ocupado).
4. **3C2B28h**: Hit en el bloque 0 del conjunto 2B2h con tag 3Ch.
5. **182B27h**: Se ubica en el bloque 1 del conjunto 2B2h, con tag 18h (se saca el bloque con tag 17h, por política LRU).
6. **258032h**: Se ubica en el bloque 0 del conjunto 803h, con tag 25h.
7. **B22328h**: Se ubica en el bloque 1 del conjunto 232h, con tag B2h.
8. **172328h**: Se ubica en el bloque 0 del conjunto 232h, con tag 17h (se saca el bloque con tag 2Ch, por política LRU).

b) Un computador con soporte para memoria virtual tiene una jerarquía de memoria consistente en una memoria caché, memoria RAM y disco duro. La obtención de una palabra de la caché toma 20ns, mientras que si la palabra no está en la caché pero si en la memoria RAM, se necesitan 60ns para cargarla en la caché desde la memoria. Finalmente, si la palabra no se encuentra en la memoria RAM, se necesitan 60ms para obtenerla desde el disco y cargarla en la memoria RAM. Si el hit-rate promedio de la caché es 0.9 y el de la memoria RAM es de 0.99, ¿cuál es el tiempo promedio que toma el obtener una palabra en este sistema? **(3 ptos.)**

Solución: Luego:

- Tiempo acceso primedio = Hit-Time(caché) + Miss-Rate(caché) × Miss-Penalty(caché)
- Hit-Time(caché) = 20ns, Miss-Rate(caché) = 1 - 0.9 = 0.1
- Miss-Penalty(caché) = Hit-Time(memoria) + Miss-Rate(memoria) × Miss-Penalty(memoria)
- Hit-Time(memoria) = 60ns, Miss-Rate(memoria) = 1 - 0.99 = 0.01
- Miss-Penalty(memoria) = Hit-Time(disco) + Miss-Rate(disco) × Miss-Penalty(disco)
- Hit-Time(disco) = 12ms = 12×10^6ns, Miss-Rate(disco) = 0

Luego, el tiempo de acceso promedio es: $20ns + 0{,}1 \times (60ns + 0{,}01 \times (12 \times 10^6 ns + 0)) = 12026ns$

Pregunta 4

Suponga que se tiene un computador con arquitectura x86, que tiene una memoria principal de 1024 bytes, con el siguiente mapa de memoria:

Dirección	Función asociada
0-7	Exception handlers
8-15	Vectores de interrupciones de hardware
16	Vector de interrupción de dibujo en pantalla
17-31	Vectores de interrupciones de software
32-123	Memoria de uso libre
124-1023	frame buffer (lo que se dibuja en pantalla) de la tarjeta de video (memory mapped)

Se desea escribir una subrutina que permita dibujar líneas horizontales o verticales en una pantalla de 30x30 pixeles, mediante el uso de interrupciones de software. Cada pixel de la pantalla puede tomar 256 tonalidades de gris, donde 0 representa el negro y 255 el blanco. Se asume que el pixel (0,0) se encuentra en la esquina superior izquierda de la pantalla.

a) Escriba la ISR asociada al vector de interrupción 10h, que permita dibujar una serie de pixeles en pantalla. Para esto, la subrutina asume que cuando fue llamada se cumple lo siguiente: **(4 ptos.)**

- La columna y fila de inicio de la línea se encuentra almacenada en el registro BX, donde la fila se ubica en BL y la columna en BH.
- El largo de la línea, en pixeles, se encuentra almacenado en los 7 bits menos significativos del registro CL, mientras que la dirección de la línea se encuentra almacenada en el bit más significativo de CL. Si es 1 la línea es horizontal, si es 0 es vertical.
- El color de la línea, se encuentra almacenado en el registro CH.

Solución: Se asume que el frame buffer se organiza en orden de filas

```
ISR10h:
  MOV AL, 30        ; Se calcula el inicio de la linea
  MUL BH            ; y se almacena en BX
  MOV DX, AX
  ADD DX, BL
  ADD DX, 124
  MOV BX, DX

  MOV DI, 1         ; Calculamos el salto por pixel en
  MOV AL, CL        ; memoria (1 o 30), dependiendo si
  SHR AL, 7         ; es vertical o horizontal. Se almacena
  CMP AL, 1         ; en DI
  JE end_if
  MOV DI, 30

end_if:
  SHL CL, 1         ; Se ajusta CL para que represente
  SHR CL, 1         ; el largo.
  MOV SI, 0         ; Se inicializan los contadores
  MOV AL, 0
```

```
for:
  CMP AL, CL       ; Se verifica la condicion de fin
  JE end
  MOV [BX+SI], CH ; Se dibuja el pixel
  ADD SI, DI
  INC AL
end:
  IRET
```

b) Escriba un programa que dibuje un cuadrado, cuyos lados miden 10 pixeles, en el centro de la pantalla. Además, el color de cada lado deberá diferenciarse del de su vecino por lo menos en 20 tonalidades de gris. **(2 ptos.)**

Solución:

```
  org 100h

  MOV BH, 10       ; Fila
  MOV BL, 10       ; Columna
  MOV CH, 20       ; Color
  MOV CL, -118     ; Largo diez y horizontal (1000 1010)
  INT 10h          ; Llamamos a la ISR de dibujo

  MOV BH, 19       ; Fila
  MOV BL, 10       ; Columna
  MOV CH, 60       ; color
  MOV CL, -118     ; Largo diez y horizontal (1000 1010)
  INT 10h          ; Llamamos a la ISR de dibujo

  MOV BH, 10       ; File
  MOV BL, 10       ; Columna
  MOV CH, 80       ; Color
  MOV CL, 10       ; Largo diez y vertical (000 1010)
  INT 10h          ; Llamamos a la ISR de dibujo

  MOV BH, 10       ; Fila
  MOV BL, 19       ; Columna
  MOV CH, 40       ; Color
  MOV CL, 10       ; Largo diez y vertical (000 1010)
  INT 10h          ; Llamamos a la ISR de dibujo

  RET
```