Universidad de Valladolid
Departamento de Informática
Escuela Universitaria Politécnica c/Francisco Mendizábal,1 47014 Valladolid Tel.:(983) 423501-02 Fax:(983) 423490

Ordenadores - I. T. Telecomunicación

Examen extraordinario - 11 de julio de 2.000
Algunas de las soluciones

Problema 1

Completar

Problema 2

La dirección de esta máquina se divide en tres campos, ya que la caché es asociativa por conjuntos. El tamaño de cada campo es el siguiente:

Bloque : 2 bits, ya que hay cuatro palabras por bloque.

Conjunto : 9 bits, ya que hay 512 ($= 2^9$) conjuntos en la cache.

Etiqueta : 13 bits, ya que 16Mpalabras $= 2^{24}$ palabras, y 24-9-2=13 bits.

La traza de los accesos necesarios para ejecutar el fragmento de programa es la siguiente:

1. Se accede para traer la instrucción `ADD [R1], [R2]`. Dirección = 0x10. Etiqueta = 0. Conjunto = 4. Palabra = 0. Lectura. No se produce fallo, ya que el bloque 0 está en el conjunto 4 de la cache. No se introduce bloque.
2. Se accede al dato situado en `[R2]`. Dirección = 0x814. Etiqueta = 1. Conjunto = 5. Palabra = 0. Lectura. No se produce fallo, ya que el bloque 1 está en el conjunto 5. No se introduce bloque.
3. Se accede al dato situado en `[R1]`. Dirección = 0x808. Etiqueta = 1. Conjunto = 2. Palabra = 0. Lectura. No se produce fallo, ya que el bloque 1 está en el conjunto 2. No se introduce bloque.
4. Se escribe el resultado de la suma en `[R1]`. Dirección = 0x808. Etiqueta = 1. Conjunto = 2. Palabra = 0. Escritura. No se produce fallo, ya que el bloque 1 está en el conjunto 2. No se introduce bloque. Como la cache es write-through, se actualizan a la vez la memoria y la cache.
5. Se accede para traer la instrucción `INC R2`. Dirección = 0x11. Etiqueta = 0. Conjunto = 4. Palabra = 1. Lectura. No se produce fallo, ya que el bloque 0 está en el conjunto 4 de la cache. No se introduce bloque. No se producen más accesos debidos a esta instrucción.
6. Se accede para traer la instrucción `ADD [R1], [R2]`. Dirección = 0x12. Etiqueta = 0. Conjunto = 4. Palabra = 2. Lectura. No se produce fallo, ya que el bloque 0 está en el conjunto 4 de la cache. No se introduce bloque.
7. Se accede al dato situado en `[R2]`. Dirección = 0x815. Etiqueta = 1. Conjunto = 5. Palabra = 1. Lectura. No se produce fallo, ya que el bloque 1 está en el conjunto 5. No se introduce bloque.
8. Se accede al dato situado en `[R1]`. Dirección = 0x808. Etiqueta = 1. Conjunto = 2. Palabra = 0. Lectura. No se produce fallo, ya que el bloque 1 está en el conjunto 2. No se introduce bloque.
9. Se escribe el resultado de la suma en `[R1]`. Dirección = 0x808. Etiqueta = 1. Conjunto = 2. Palabra = 0. Escritura. No se produce fallo, ya que el bloque 1 está en el conjunto 2. No se introduce bloque. Como la cache es write-through, se actualizan a la vez la memoria y la cache.
10. Se accede para traer la instrucción `CALL RUTINA`. Dirección = 0x13. Etiqueta = 0. Conjunto = 4. Palabra = 3. Lectura. No se produce fallo, ya que el bloque 0 está en el conjunto 4 de la cache. No se introduce bloque.
11. Se decrementa el puntero de pila (suponemos que la pila crece hacia posiciones decrecientes). No se produce acceso a memoria. Nuevo puntero de pila: 0x180F.
12. Se guarda el PC en la dirección apuntada por el puntero de pila. Dirección = 0x180F. Etiqueta = 3. Conjunto = 3. Palabra = 3. Escritura. Se produce un fallo en el conjunto 3, ya que el bloque con etiqueta 3 no está en él. Se desaloja el bloque cuya etiqueta es 2, al ser el más antiguo del conjunto (política FIFO). Como la cache es write-through, se actualizan a la vez la memoria y la cache.
13. Se carga en el PC la dirección de RUTINA. Esta dirección forma parte de la instrucción `CALL RUTINA`, por lo que no se produce acceso a memoria.
14. Se accede para traer la instrucción `PUSH R3`. Dirección = 0x1808. Etiqueta = 3. Conjunto = 2. Palabra = 0. Lectura. Se produce un fallo, ya que la etiqueta 3 no está en el conjunto 2 de la caché. Se desaloja el bloque más antiguo (el 1) y se trae el bloque 3.
15. Se decrementa el puntero de pila (suponemos que la pila crece hacia posiciones decrecientes). No se produce acceso a memoria. Nuevo puntero de pila: 0x180E.
16. Se guarda el registro R3 en la dirección apuntada por el puntero de pila. Dirección = 0x180E. Etiqueta = 3. Conjunto = 3. Palabra = 2. Escritura. No se produce fallo.

Universidad de Valladolid
Departamento de Informática
Escuela Universitaria Politécnica c/Francisco Mendizábal,1 47014 Valladolid Tel.:(983) 423501-02 Fax:(983) 423490

Cuestión 1

Los algoritmos utilizados más comúnmente para planificación son los siguientes:

First come, first served (FCFS) : El primer proceso en solicitar la CPU es el primero en recibir la asignación de la misma. Se implementa a través de una cola FIFO. Problema: si se ejecuta primero el más largo, el tiempo de retorno medio es muy alto.

Shortest Job First (SJF) : El tiempo de CPU se divide en ráfagas. Cuando la CPU queda libre, se asigna al trabajo con la siguiente ráfaga más pequeña. Requiere conocer la duración de cada ráfaga futura para cada trabajo, por lo cual sólo es útil en trabajos grandes del que se conoce su tiempo de ejecución.

Prioridad : Cada trabajo tiene asociada una prioridad, y la CPU se asigna al trabajo con prioridad más alta. Si dos trabajos tienen la misma prioridad, se planifican por FCFS. Cada S.O. define sus propios niveles de prioridad.

El peligro de la prioridad es la *inanición* o *bloqueo indefinido*. Sucede cuando un proceso tiene una prioridad tan baja que no consigue ejecutarse nunca. Normalmente estos trabajos acaban ejecutándose cuando no hay procesos de más prioridad (de noche, etc). Se resuelve introduciendo el concepto de *envejecimiento*: a medida que el proceso va llevando más tiempo, va subiendo su prioridad, por ejemplo cada 15 minutos.

Estos algoritmos son apropiativos: hasta que no se solicita una operación de E/S la CPU continúa ejecutando el trabajo.

Round Robin : Se divide el tiempo de CPU en *quantum* de tiempo, asignando un quantum a cada proceso. El proceso puede consumirlo en su totalidad o solicitar antes una operación de E/S. Si lo consume, al cabo del quantum el planificador de la CPU lo desaloja: *cambio de contexto*.

Si el quantum es muy largo, sus prestaciones son iguales a las del FCFS. Si es muy corto, el tiempo de cambio de contexto empieza a ser relevante, degradándose las prestaciones.

Cuestión 2

- Un *trap* o trampa es una transferencia del control del programa a una rutina de servicio. Dicha transferencia es iniciada por el programador. También se las denomina "interrupción software". Su objetivo es solicitar al sistema operativo la realización de alguna tarea. En los x86 se consigue a través de la instrucción `int`.
- Una *excepción* es un *trap* generado automáticamente en respuesta a una condición excepcional, por ejemplo intentar ejecutar una instrucción con un código ilegal, división por cero o un error de protección de memoria (intentar acceder a una zona de memoria inválida). La CPU invoca a una rutina de tratamiento que decide qué hacer: normalmente escribe un mensaje de error y suspende la ejecución del programa. Las excepciones, por lo tanto, no son invocadas por el programador.
- Una *interrupción hardware*, o *interrupción* propiamente dicha, es una solicitud de atención por parte de un dispositivo externo, provocado a través de una línea de solicitud de interrupción específica.

Cuestión 3

Los 8086 permiten hasta 256 vectores de interrupción diferentes. Para ello disponen de una tabla de 256 entradas que comienza a partir de la dirección `0:0`, con entradas de 4 bytes. El tratamiento de las interrupciones es el siguiente:

1. Para localizar la dirección de la rutina de servicio de interrupción, se multiplica por 4 el vector de interrupción y se mira la entrada correspondiente en la tabla, cargándose las direcciones de segmento y desplazamiento.
2. La CPU guarda el estado, el CS y el IP en la pila y transfiere el control a la rutina de servicio.
3. La rutina de servicio debe encargarse, ante todo, de guardar *absolutamente* todos los registros de la CPU en la pila, para no modificar ninguno. Cuando termine, deberá recuperarlos y acabar con la instrucción `iret`, para recuperar el estado del procesador.
4. No se permite el anidamiento de interrupciones: dentro de la rutina de servicio, se ejecuta `cti` para limpiar el bit que permite las interrupciones. Antes de volver, se ejecuta `sti` para dejarlo a 1. Los traps y las excepciones no hacen esto.

Cuestión 4

- Sistemas cableados (hardwired control unit): Se utilizan metodos de diseño de circuitos secuenciales síncronos para construir una unidad de control.
- Sistemas microprogramados (microprogrammed control unit): se agrupan todas las funciones que pueden activarse simultáneamente en palabras de control y se almacenan en una memoria independiente (memoria de control). Los campos de control individuales se dirigen a las distintas unidades funcionales.
- El diseño de una unidad microprogramada es más elaborado que el de una cableada: inclusión de una ROM adicional, etc. Además es más lenta. Ventaja: proporciona una organización de control bien estructurada. Se

Universidad de Valladolid
Departamento de Informática
Escuela Universitaria Politécnica c/Francisco Mendizábal,1 47014 Valladolid Tel.:(983) 423501-02 Fax:(983) 423490

pueden realizar cambios sólo modificando el microprograma.

Cuestión 5

Escritura inmediata (*Write through*): Consiste en actualizar a la vez la memoria caché y la memoria principal. De esta manera, si la CPU solicita de nuevo el dato en lectura, la copia de la caché sigue siendo válida. Si hay que desalojar el bloque, dicho bloque puede descartarse directamente. Sin embargo, un mismo bloque puede sobreescribirse varias veces consecutivas, lo que obliga a realizar varias escrituras a memoria.

Escritura diferida (*Write back*): Sólo actualiza la caché. Es mas rápida, pero obliga a copiar el bloque entero a la memoria cuando se lo desaloje. Las cachés con este sistema utilizan un bit por cada bloque que indica "limpio" (no modificado) o "sucio" (modificado). Si el bloque está sucio, se desaloja a memoria; en caso contrario puede descartarse.

Cuestión 6

Tiempo de acceso Se distingue entre el de lectura y el de escritura. Se mide en ns. Inverso: frecuencia de acceso.

Tiempo de ciclo Tiempo mínimo entre dos lecturas consecutivas. Inverso: tasa de transferencia, ancho de banda o caudal (bits/seg).

Volatilidad Hace referencia a la capacidad de almacenar o no los datos en ausencia de energía.

Destructibilidad Si al leer un dato éste se pierde: memoria destructiva. Ejemplo: memorias DRAM, construidas con condensadores.

Acoplamiento Término utilizado en multiprocesadores.

- Si sólo hay un módulo de memoria para varios procesadores, se dice que la memoria está "centralizada": sistemas fuertemente acoplados.
- Si hay un módulo de memoria por cada procesador, se dice que la memoria está "distribuida": sistemas débilmente acoplados.