DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
ESCUELA DE INGENIERÍA
PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

IIC2343 - Arquitectura de Computadores (I/2020)

Pregunta 9

Fecha: 24 de junio, 2020.

Instrucciones

I. Esta evaluación es estrictamente **INDIVIDUAL**.

II. Lee el enunciado completo de manera atenta y detallada.

III. Incluye tu procedimiento ordenado.

IV. Referencia de manera clara y ordenada en APA en caso de utilizar material que esté público en internet.

V. Pruebas de otros semestres no son una fuente válida que referenciar.

VI. Para responder preguntar dudas de enunciado, usa las issues del Syllabus en GitHub, pero si darías pistas de la solución, escríbenos a `iic2343.puc@gmail.com`.

Entrega

La entrega será mediante este formulario de google.

- Tienes hasta las 12:00 horas del día 25 de junio de 2020 para entregar.
- Entrega un documento PDF con tus respuestas **para cada pregunta** de manera independiente.
- Es preferible que tus respuestas estén escritas en digital, para evitar que el ayudante no entienda tu letra. Queda a tú responsabilidad el decidir hacerlo a mano.

9. Entrada y Salida

a) En el caso de I/O programado, con *busy-waiting*, considera que el controlador de una impresora tiene 32 bytes direccionables individualmente, agrupados en 8 registros:

   i) Suponiendo que el primer registro de la impresora tiene dirección 1000 y que cumple con la función de registro *status*. Diseña una asignación de los registros de modo que sea posible saber si:

      - La impresora está encendida o no.
      - Ordenar que cargue una hoja de papel.
      - Saber si cargó la hoja.
      - Especificar la dirección de memoria del string que hay que imprimir.
      - Hacer que lea el string que está en esa dirección.
      - Saber si lo leyó.
      - Ordenar que inicie la impresión de una línea.
      - Saber si la imprimió.
      - Hacer que avance la hoja a la próxima línea.
      - Saber si lo hizo.
      - Y saber si la impresora está ocupada o no.

   ii) ¿Cuáles de estas operaciones corresponden a operaciones *fetch*, cuáles a *store*, y por qué?

   iii) Escribe el código que debe ejecutar la CPU para controlar el funcionamiento de la impresora, en lo más parecido que puedas a un lenguaje de programación.

b) En el caso de interrupciones, considera un computador con tres dispositivos de I/O: una impresora, un disco, y un modem, con prioridades 1, 2 y 4 (más alta), respectivamente; y considera que procesar una interrupción (de cualquier tipo) toma 10 unidades de tiempo. En $t = 0$, hay un programa corriendo. En $t = 10$ ocurre una interrupción del modem; en $t = 15$, la impresora genera su propia interrupción; y en $t = 17$, el disco produce su propia interrupción.

   i) Dibuja una línea de tiempo y especifica cuál interrupción está siendo procesada cuándo, desde $t = 0$ y hasta que vuelve a ejecutarse el programa original. Justifica brevemente.

   ii) ¿Qué hay en el stack en cada uno de los intervalos relevantes de tiempo? Justifica brevemente.

Soluciones

IMPORTANTE: nos reservamos el derecho a no restringirnos del todo a los formatos exigidos por los enunciados al momento de mostrar una solución.

9. Entrada y salida.

9.1. Parte A.

I. El registro del controlador con dirección 1000 es el registro de *status* de la impresora ⇒ sus diferentes bits nos dicen en qué estado está la impresora (la signación específica de bits no importa).

Los otros registros del controlador son para darle instrucciones o transmitirle información, según la especificación en el enunciado; sus direcciones son 1004, 1008, etc. P.ej., una posibilidad es:

Direcciones	Tipo	Significado
1000-1003	fetch	0 ⇒ apagada; otro valor ⇒ encendida (y así según lo que se esté chequeando).
1004-1007	store	$\neq 0$ ⇒ cargue una hoja.
1008-1011	store	Dirección de memoria del string.
1012-1015	store	$\neq 0$ ⇒ mover el string al buffer interno.
1016-1019	store	Imprimir una línea del string.
1020-1023	store	Avanzar la hoja hasta la próxima línea.
etc.		

II. Los tipos de las operaciones aparecen en la respuesta anterior: son fetch cuando la CPU quiere sa-ber algo sobre la impresora (su status); son store cuando la CPU quiere decirle algo a la impresora: enviarle información o darle una orden.

III. Supongamos que la CPU usa su propio registro p para referirse a los registros del controlador; es decir, en p tiene la dirección del registro del controlador. Voy a representar por reg[p] el contenido del registro del controlador que tiene asignada la dirección p.

```
1  p = 1000
2  if reg[p] == 0:
3          ***reclamar para que alguien encienda la impresora***
4  p = p + 4
5  reg[p] = 1      -ordenar a la impresora cargar una hoja
6  p = 1000
7  while reg[p] == 0:  -y esperar hasta que haya cargado una hoja
8  p = 1008
9  reg[p] = dirección del string en la memoria              -poner en el controlador la dirección del string
10 p = 1012
11 reg[p] = 1      -ordenar a la impresora leer el string
12 p = 1000
13 while reg[p] == 0:      -esperar hasta que la impresora haya leído el string
14 p = 1016
15 reg[p] = 1      -ordenar a la impresora imprimir una línea
16 p = 1000
17 while reg[p] ==0:       -esperar hasta que haya impreso la línea
18 p = 1020
19 reg[p] = 1      -ordenar a la impresora avanzar a la próxima línea
20 p = 1000
21 while reg[p] ==0:       -esperar hasta que haya avanzado a la próxima línea
```

Los while's no tienen una instrucción supeditada, de modo que parece que fueran loops infinitos; pero lo que pasa en que el estado de la impresora cambia cuando termina de ejecutar la orden que se le dio, y entonces cambia el contenido del registro de status (que tiene dirección 1000).

Estrictamente hablando, el código debiera usar sólo desplazamientos relativos a la dirección del primer registro del controlador, ya que el que escribe este código no sabe qué dirección se le va a asignar a ese registro.

9.2. Parte B.

I. Entre t = 0 y t = 10, se está ejecutando el programa, no hay interrupciones.

   En t = 10 se empieza a procesar la interrupción del modem.

   En t = 15 llega la interrupción de la impresora, pero como tiene menos prioridad que la del modem, queda en espera.

   En t = 17 llega la interrupción del disco, pero como tiene menos prioridad que la del modem, también queda en espera.

   En t = 20 termina de procesarse la interrupción del modem y empieza a procesarse la interrupción del disco, ya que tiene mayor prioridad que la de la impresora.

   En t = 30 termina de procesarse la interrupción del disco y empieza a procesarse la interrupción de la impresora.

   En t = 40 termina de procesarse la interrupción de la impresora y se reanuda la ejecución del programa.

II. Hasta t = 10 y desde t = 40, nada.

   Entre t = 10 y t = 40, los datos relevantes de la ejecución del programa; básicamente, los contendidos de los registros de la CPU: PC, PSW, los registros generales de la CPU.