Segundo Parcial

Segundo Cuatrimestre 2023

Normas generales

- El parcial es INDIVIDUAL
- Puede disponer de la bibliografía de la materia y acceder al repositorio de código del taller de system programming, desarrollado durante la cursada
- Las resoluciones que incluyan código, pueden usar assembly o C. No es necesario que el código compile correctamente, pero debe tener un nivel de detalle adecuado para lo pedido por el ejercicio.
- Numere las hojas entregadas. Complete en la primera hoja la cantidad de hojas entregadas
- Entregue esta hoja junto al examen. La misma no se incluye en el total de hojas entregadas.
- Luego de la entrega habrá una instancia coloquial de defensa del examen

Régimen de Aprobación

- Para aprobar el examen es necesario obtener cómo mínimo **60 puntos**.
- Para promocionar es condición suficiente y necesaria obtener como mínimo **80 puntos** tanto en este examen como en el primer parcial

NOTA: Lea el enunciado del parcial hasta el final, antes de comenzar a resolverlo.

Enunciado

Ejercicio 1 - (50 puntos)

Se desea construir un sistema similar al del taller que debe ejecutar concurrentemente 5 tareas independientes. Las tareas de este sistema utilizarán el registro `ecx` como reservado el cual al ser un registro de la tarea, este puede ser modificado y seteado en cualquier momento. El registro `ecx` contendrá en todo momento un número de ticks del reloj denominado UTC (Unreal Time Clock) el cual será actualizado por el sistema, incrementándolo cada vez que la tarea vuelva a ser ejecutada luego de una interrupción de reloj. Tener en cuenta que este valor solo esta guardado en un único lugar que se corresponde siempre con el espacio dedicado a los `ecx` de cada tarea.

Por otro lado, el sistema cuenta además con el servicio *fuiLlamadaMasVeces* que permite que una tarea pregunte si el UTC de otra tarea es menor que el suyo. Este servicio espera en `edi` el ID de la tarea por la que se esta preguntando (los IDs van del 0 al 4) y devuelve el resultado en `eax`. El resultado será 0 si la tarea llamadora tiene un UTC menor o igual que la tarea por la que preguntó y 1 en caso contrario.

(a) Describir qué entradas están presentes en la GDT indicando los campos que consideren relevantes.

(b) Describir qué deben modificar respecto del sistema del taller para que el valor UTC se actualice correctamente en los `ecx` de cada tarea.

(c) Describir qué y cómo deben modificar el sistema del taller para poder implementar el servicio *fuiLlamadaMasVeces*.

(d) Implementar el servicio *fuiLlamadaMasVeces* usando asm.

(e) Pregunta extra: ¿tiene sentido a nivel sistema tener un registro de propósito general reservado para guardar el UTC? ¿De qué otra manera podría solucionarse?

Ejercicio 2 - (50 puntos)

Por un lado tienen un sistema similar al utilizado en el taller de la materia y por el otro una colección de software originalmente diseñado para correr en nivel cero. Los programas de dicha colección de software utilizaban la instrucción `HLT` para apagar la computadora una vez habían terminado su trabajo. Por razones obvias no podemos permitir esto en un sistema multitarea.

Se solicita adaptar el sistema implementado en los talleres para que se puedan utilizar estos programas **en nivel de usuario** y **sin modificaciones**. Cuando éstos intenten ejecutar `HLT` el sistema operativo debe interpretar esa acción como una solicitud de "fin de la tarea".

a. ¿Qué excepción ocurrirá cuando un proceso no privilegiado intente ejecutar `HLT`?

b. ¿Cómo puede determinar que la instrucción que se quiso ejecutar es `HLT`?

c. ¿Qué pasos debe seguir para "finalizar" un proceso?

d. ¿Cómo determinará el próximo proceso a ejecutar?

e. Describa los cambios que debe realizar a las estructuras del sistema para poder agregar el mecanismo solicitado.

f. Escriba el pseudocódigo necesario para implementar el mecanismo.

Su pseudocódigo debe describir el detalle de las siguientes acciones:

- Atender la excepción y determinar si es necesario usar el nuevo mecanismo
- Finalizar el proceso que generó la excepción
- Abandonar el proceso finalizado saltando a uno todavía vivo
- Modificar el scheduler (de ser necesario) para que nunca ejecute procesos finalizados

A tener en cuenta para la entrega (para todos los ejercicios):

- Está permitido utilizar las funciones desarrolladas en los talleres.
- Es necesario que se incluya una explicación con sus palabras de la idea general de las soluciones.
- Es necesario escribir todas las asunciones que haga sobre el sistema.
- Es necesaria la entrega de código o pseudocódigo que implemente las soluciones.