Trabajo Práctico Especial

Primer Cuatrimestre 2019

Objetivo

Implementar un *kernel* que administre los recursos de hardware de una computadora y muestre características del modo protegido de Intel.

Enunciado

Implementar un *kernel* booteable por Pure64, modificado y provisto por la cátedra. El mismo debe administrar los recursos de hardware y proveer una API para que aplicaciones de usuarios puedan utilizar estos recursos.

Se deben definir dos espacios claramente separados, uno *kernel space* y el otro *user space*. El espacio del *kernel* interactuará directamente con el *hardware*mediante *drivers,* mientras al mismo tiempo proveerá funciones al *user space*. El espacio de usuario no debe acceder por ninguna razón directamente al hardware, sólo a través del espacio de *kernel*. En la **Figura 1**se puede ver un ejemplo de los módulos que conforman el espacio de usuario y los módulos que conforman el espacio de kernel.

Se deberá definir una API con la cual el espacio de usuario puede acceder a las funciones del *kernel*. Este acceso deberá ser implementado a través de la interrupción de software `80h`, ya que estos dos módulos se encuentran en distintos espacios de memoria. Dicha API deberá basarse en la API de Linux, que está comprendida por funciones como ***read*** y***write.***

Además se deberá definir un set de funciones para interactuar con dicha API, deberá formar parte al equivalente en Linux a la biblioteca estándar de C. En la cual, se deberá estar basado. Por ejemplo, deberá contar con funciones como ***scanf***, ***printf***, ***putChar***, ***getChar.***

La implementación del sistema deberá contar con los siguiente:

- Intérprete de comandos: que demuestren el funcionamiento del *kernel.* Deberá tomar dichos comandos por entrada estándar y mostrar el resultado de la ejecución por salida estándar. El intérprete deberá interactuar con el sistema mediante un modelo de Terminal. Este intérprete de comandos deberá siempre despegar su prompt para ingreso de nuevos comandos en la línea inferior de la pantalla y cada vez que se muestre una nueva línea se deberá desplazar las líneas de información ya existentes hacia arriba eliminando la línea más antigua si fuera necesario.

- Funcionalidades:

- Una función de ayuda, que muestre los distintos programas disponibles.
- Intérprete de comandos, tipo *Shell* que desencadene las distintas funcionalidades.
- El kernel deberá poder manejar dos tipos de excepciones : división por cero y código de operación inválido. Ante estas excepciones se deberá desplegar información sobre el tipo de error, instruction pointer y registros en el momento del error.
- Módulos/comandos para verificar el funcionamiento de las rutinas de excepción antes descritas.
- Módulo/comandos para desplegar la hora del sistema
- Cuando el sistema arranca, debe preguntar qué módulo se desea correr
  Módulo /comando que permita graficar en modo video el juego Snake https://es.wikipedia.org/wiki/La_serpiente_(videojuego)usando un conjunto de teclas para el movimiento de la víbora y manteniendo en pantalla el tiempo transcurrido .Cada 15 segundos se deberá incrementar la velocidad de la vìbora e incrementar su largo emitiendo un sonido de aviso,.
  Al finalizar el juego se deberá desplegar el tiempo de vida.y también un sonido de finalización.

  Cualquier otra función que el grupo considere apropiada para mostrar el funcionamiento interno del sistema.

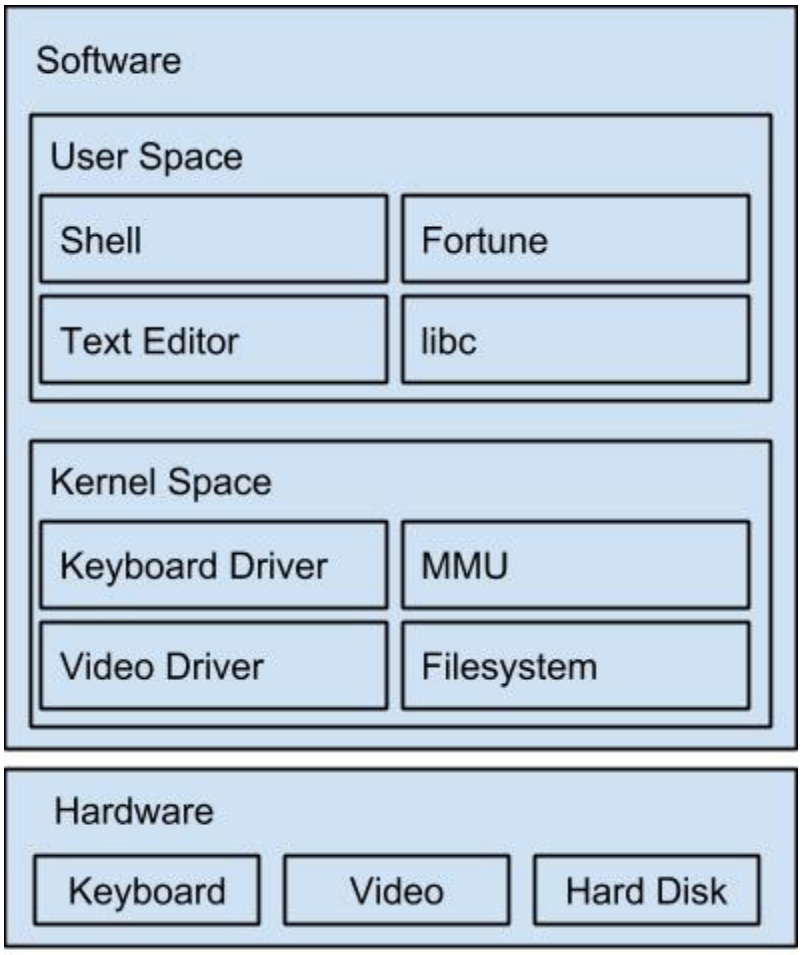

Figura 1 - Ejemplo de Módulos

Los requerimientos implican la implementación de drivers de teclado y video.

El sistema deberá ser implementado para una Arquitectura Intel de 64 bits en**Long Mode.** La entrada a ese modo será provista por un módulo implementado por la cátedra y debe correr en una PC física real.

Fuentes

Cómo base del trabajo práctico, descargar los archivos del siguiente repositorio:

https://bitbucket.org/RowDaBoat/x64barebones/wiki/Home

Integrantes

El grupo de trabajo podrá contar con un máximo de cuatro (4) integrantes.

Calificación

La nota del trabajo práctico consta de la evaluación de los entregables más la ***demostración*** de su funcionamiento en un coloquio oral. En dicho coloquio se podrá consultar a cualquier integrante cualquier porción de código cual es su implementación y funcionamiento. Además se podrán consultar conceptos teóricos de la materia.

A la hora de la presentación, todos los miembros deberán estar presentes, salvo causa de fuerza mayor, en la cual, el integrante faltante deberá acreditar la documentación que justifique la falta. Dicho integrante luego deberá rendir el coloquio oral en la fecha que se acuerde con la cátedra.

En el caso de que se deba re-entregar el TP por no estar aprobado en primera instancia, se penalizará a los integrantes del grupo con una disminución de 2 (dos) puntos sobre la nota final. Es decir, será necesario un 6 (seis) para aprobar el trabajo práctico en instancia de recuperación.

Entregables

Cada grupo deberá entregar:

- Por email, un archivo ZIP con el código fuente (sin los binarios), informe (que explique el diseño elegido y sus justificaciones , pros y contras) y manual de usuario. Además, su correspondiente hash md5, en el cuerpo del mail. El archivo debe estar nombrado cómo X-Y-Z.zip, donde X, Y, Z son los legajos de los participantes. Tener en cuenta que el **hash md5** es la firma del grupo sobre el entregable. En caso de tener problemas con el envió del TPE, enviar el hash directamente.

Todo el material deberá ser enviado en forma digital el dia de la entrega vía email el día de la entrega a la casilla arq-catedra@googlegroups.com

Consideraciones

- El tipo de diseño y la forma de implementación serán discutidos entre el grupo y la cátedra durante las clases de laboratorio, dejando la posibilidad de modificar éste enunciado escrito, previo acuerdo entre el docente y los integrantes del grupo.

- Para la evaluación se tendrá en cuenta no sólo el funcionamiento del programa sino también la calidad del software y la documentación escrita pedida.

- Cualquier funcionalidad agregada al TP por sobre lo pedido deberá estar documentada apropiadamente.
- Cualquier código encontrado en Internet u otra fuente debe ser consultado con la cátedra previo a la inclusión en el TPE. Toda inclusión de código de terceros deberá ser documentada con los enlaces o referencias correspondientes a su origen. Deben estar sumarizados en el informe de la cátedra.
- Cualquier aclaración oral a cargo de la cátedra con respecto al enunciado del TP tiene la misma validez que el enunciado escrito.
- Los distintos grupos podrán consultar y discutir diseños entre sí, pero la implementación deberá ser propia de cada uno. Cualquier detección de plagio es susceptible de sanciones académicas de acuerdo al reglamento de la Universidad.
- Todos los trabajos corren sobre la misma arquitectura y cuentan con el mismo *template*, por lo tanto existirán muchos códigos equivalentes o similares. La cátedra tendrá esta situación en cuenta.

Fechas de Entrega y Defensa

La entrega del TPE es hasta el domingo 23 a las 23:59 hs. por correo electrónico dirigido al correo de la cátedra y en el Laboratorio de Informática el manual de usuario y el informe. La defensa será el 24 de junio a las 16 hs. en el aula 26.

La fecha de recuperatorio será el domingo 30 de junio , y la defensa será al día siguiente 1 de julio a las 16 hs en el aula 26.