Instituto Tecnológico de Buenos Aires

TPE Arquitectura de Computadoras (72.08)

INFORME

Integrantes:

Axel Castro Benza: 62358 - acastrobenza@itba.edu.ar

Lautaro Farias: 60505 - lfarias@itba.edu.ar

Simón Marabi: 61626 - smarabi@itba.edu.ar

Índice

Separación Kernel Space - User Space

El Kernel es el responsable de administrar y controlar los recursos del sistema como la memoria, dispositivos de entrada y salida, entre otros. Además, este otorga una capa de abstracción y un conjunto de funciones y/o servicios que pueden ser utilizados por las aplicaciones y procesos que se ejecutan en el User Space.

Es decir, el Kernel actúa como un intermediario entre el hardware y el software. Este se relaciona de manera directa con componentes propios del hardware a través de controladores o drivers como lo son el keyboard.c, naiveGraphicsConsole.c, entre otros que en este caso interactúan con el teclado y la pantalla respectivamente.

Por otro lado, el User Space, a diferencia del Kernel Space no posee un acceso directo al hardware. Este debe utilizar las interfaces proporcionadas por Kernel para poder acceder a los recursos y servicios del sistema. Dicho acceso es implementado mediante la instrucción de interrupción 80h ya que los dos módulos se encuentran en espacios de memoria distintos. Donde dentro del Kernel se encuentra un dispatcher de syscalls que se encarga de determinar que syscall se desea llamar y cada syscall tiene su handler, que se encarga de ejecutar la syscall.

Manejo de las syscalls implementadas

A continuación se detallarán las syscalls implementadas:

sys_write_handler: se ocupa de imprimir una cadena de caracteres solamente en caso de que el parámetro “fd” (file descriptor) sea igual a la pantalla y/o salida estándar. Esta impresión se realiza con ayuda de la función “ngc_printChar(...)” que se encarga de la impresión pixel por pixel.

sys_read_handler: se encarga de leer los datos del teclado y/o de entrada estándar. En caso de que el parámetro “fd” (file descriptor) no sea ninguno de los previamente mencionados entonces la función retorna con valor -1. Las cantidad de letras leídas se definen gracias al parámetro “bytes”. Luego, estas se almacenan en un buffer con la ayuda de la función “getFirstChar()” que obtiene el primer carácter disponible de un buffer.

sys_time_handler: se ocupa de para obtener la hora actual del sistema. Lo logra mediante el llamado a las funciones “_NRTCGetHours()” (obtiene las horas), “_NRTCGetMinutes()” (obtiene los minutos) y “_NRTCGetSeconds()” (obtiene los segundos).

sys_font_handler: se ocupa de alterar el tamaño de los caracteres que se imprimen por pantalla y/o salida estándar. El parámetro level determina el tamaño que tendrán las letras (1, 2 o 3 siendo 3 el mayor). Logra esto en cooperación con la función changeFontSize(...) que actualiza el nivel de tamaño de los caracteres.

sys_printColor_handler: se dedica a imprimir una cadena de caracteres de un color en específico. Lo logra llamando a la función “ngc_printColor(...)” que se encarga de alterar el color de la pluma y luego a imprimir la cadena.

sys_clear_screen_handler: se encarga de limpiar la pantalla con un color. Si bien no era requisito incluir un comando que limpie la pantalla, fue incluida con el fin de facilitar la visualización de la terminal.

sys_screenData_handler: se ocupa de obtener el ancho y el largo de la pantalla. Estos valores se almacenan en el vector pasado como parámetro. Siendo la primera posición del vector el ancho y la segunda el largo.

sys_paint_rect_handler: se destina a la impresión de un rectángulo en la pantalla de un color determinado. Donde “fromX” y “fromY” representan las coordenadas donde se iniciará la impresión, y “width” y “length” el ancho y largo que tendrá este rectángulo respectivamente.

sys_ticks_handler: se encarga de obtener el número de ticks transcurridos del sistema en un momento dado.

sys_beeper_handler: se ocupa de generar un sonido o beep en el sistema durante un periodo de tiempo determinado por el parámetro “interval”.

Manejo de interrupciones

Se utiliza la IDT (Interrupt Descriptor Table) para gestionar las interrupciones. Se trata de una tabla de descriptores de interrupción que contiene información sobre cómo manejar diferentes tipos de interrupciones y excepciones, cuya carga se realiza en el archivo "idtLoader.c". Las dos principales interrupciones manejadas en este trabajo pertenecen al teclado y al timer tick.

El manejo de la primera interrupción mencionada implica capturar el código de la tecla presionada, procesarlo y realizar las acciones correspondientes, como manejar combinaciones de teclas especiales o realizar acciones específicas según la tecla, a medida que se carga el buffer.

La interrupción del timer tick es relevante a la hora de mantener un control del tiempo. Por lo que el manejo de esta interrupción implica la actualización de un contador de ticks.

Por otro lado, las rutinas de interrupción propiamente dichas se definen en el archivo "interrupts.asm", las cuales se ocupan de manejar y atender las interrupciones específicas.

Modo Gráfico VESA

Se eligió el modo gráfico para tener un control más sencillo sobre los píxeles en la pantalla, a pesar de la necesidad de dibujar píxel por píxel. Esto llevó a la creación de un sistema de representación de formas y texto para mostrar la información en la pantalla. Un componente clave para esto es el archivo "font.h", que define el formato de cada carácter. Además, se desarrollaron varias funciones en el archivo "naiveGraphicsConsole.c" para manejar la impresión de caracteres, como "ngc_printChar(...)", así como funciones para ajustar el tamaño de la fuente, como "ngc_fontSize1(...)", "ngc_fontSize2(...)" y "ngc_fontSize3(...)".

Además, se implementaron funciones destinadas a garantizar el funcionamiento adecuado del intérprete de comandos. Entre estas funciones se encuentran "ngc_printNewLine()", que se encarga de actualizar las líneas de comando, así como "ngc_paint_screen(...)" y "delete_last_char()", las cuales desempeñan roles específicos en la gestión y el manejo de la interfaz de línea de comandos.