TP3 Orga-2

Ballan Paulo
Sansone Agustín
Rios Cuba Kennedy

Descripción de los items realizados:

I. Segmentación

• GDT

Completamos la tabla de descriptores globales dos para código y dos para datos, uno de nivel 0 y otro de nivel 3 respectivamente.

Inicializamos los descriptores de la GDT apartir del incide 22 (inclusive) en adelante.

Pasamos a modo protegido y seteamos la pila del kernel en la dirrección 0x27000 .

Declaramos un segmento adicional para la pantalla, que solo pueda ser usado por el kernel.

Pintamos en pantalla el tablero junto con las barras de los jugadores.

• IDT

Completamos las entradas de la IDT para poder responder a las interrupciones del sistema y que imprima por pantalla el numero y la interrupción de la misma.

Testeamos estas interrupciones con una división por cero.

Asociamos las entradas necesarias para el reloj, teclado y para que atienda los otros servicios de las tareas.

Completamos la rutina de atención del reloj para que por cada tick llame a la función nextClock.

Completamos la rutina de atención del teclado para que al precionar una de las teclas numericas, imprima en pantalla el numero presionado.

II. Paginación

Inicializamos cr3 para el kernel y dos variable globales $NEXT - PAGE - FREE - TASK - FISICA$ y $NEXT - PAGE - FREE - KERNEL$ para saber en todo momento cuál página voy a pedir cuando una tarea necesite pedir 2 páginas para su código, dato y pila (nivel 3) dentro del espacio libre de tareas. Otra página para la pila nivel 0 de la tarea (en el espacio libre de kernel).

En el cr3 del kernel mappeamos el area de kernel y libre de kernel con identity mapping.

Escribimos una función $MMU - InitTaskDir$ la cual, dado un puntero a TSS, crea su Directorio y tabla correspondiente a la tarea y pedir las dos páginas para copiar código y dato de la tarea.

III. Tareas

La tarea Idle comparte el directorio y la pila nivel 0 con el kernel, esta tarea a diferencia de las otras solamente pide una página para código y dato.

Para inicializar las estructuras para las tareas, hicimos dos funciones. Una se encarga de inicializar los descriptores en la GDT y la otra función se encarga de inicializar las TSS con valores de TSS validos.

Para cada tarea, se piden tres páginas en el área libre de tareas para la dato, código y pila lvl 3, y una página del area libre de kernel para su pila lvl 0. Además copiamos las dos páginas de código y dato, y las mappeamos en la dirección virtual 0x80000000 (con su cr3) para que una tarea no pueda acceder a otra.

IV. Scheduler

Definimos las estructuras mínimas para poder responder a las interrupciones que atienden los servicios de las tareas.

V. Preguntas o Faltantes

El descriptor del $video - kernel$ tiene g = 1, db =1 y dpl =3, están bien?

El mapeo del kernel a cada tarea, esta en modo supervisor, ¿podría ser también usuario?, ya que está protegido por segmentación.

¿El atributo de las TSS's, iomap con qué es seteado? (nosotros seteamos con 0).

Para la función divide, no llegamos a entender el retorno 1 si es la original y 0 si es la copia. Nosotros entendemos que esta función hace la copia y salta a la tarea Idle, no llegamos a ver donde se ve reflejado este 1 y 0 (o siempre retorna 1 porque la copia corre en el siguiente turno).

¿Puede haber mas de una tarea del mismo jugador en una casilla?, si es asi, entonces no vemos como podria no haber lugar para una tarea. Caso contrario, ¿que pasa cuando muchas tareas se mueven a un casillero, gana el conjunto de tareas que tiene mayor puntaje y quien toma la casilla?