

Índice

1. Ejercicio 1	2
2. Ejercicio 2	2

1. Ejercicio 1

El primer ejercicio consistió en dos elementos, el primero fue sobre llenar la Global Descriptor Table (GDT), con ciertos segmentos y el segundo, en pasar a modo protegido.

Inicializar la GDT

En esta parte lo que hicimos primero fue crear un arreglo de `gdt_entry`, en el archivo `gdt.c` de 41 entradas y luego completamos 6 de las mismas. En la primer posición del arreglo, fue seteado el descriptor Nulo por restricciones de Intel. Y los otros 5 descriptors de segmento fueron completados empezando desde la posición 18, por restricciones de T_p, hasta la 22 inclusive.

Los dos descriptors de segmento de las posiciones 18 y 19 fueron seteados como segmentos de código nivel 0 y 3 respectivamente. Empezando la base de ambos desde la posición 0x0000, y con un límite de 0x6ffff con el bit de Granularity activado representando así 1.75 GB. Al type de ambos segmentos se les puso el valor de 0xA (Execute/Read), y los atributos de Sistema y Precencia en Uno.

Los dos descriptors de segmento de las posiciones 20 y 21 fueron seteados como segmentos de datos nivel 0 y 3 respectivamente. Empezando la base de ambos desde la posición 0x0000, y con un límite de 0x6ffff con el bit de Granularity activado. Al type de ambos segmentos se les puso el valor de 0x02 (Read/Write), y a los atributos de Sistema y Precencia en Uno.

Por último al descriptor seteado en la posición 22 del arreglo, fue colocado como un descriptor de video, con la base a partir de 0xb8000, y límite 0xf9f, con el bit de Granularity en 0. A este segmento también se le asignaron en los atributos de Sistema y Precencia el valor de 1.

Pasar a Modo Protegido

Luego de cargar la GDT, creándola con los requisitos ya especificados y cargando el GDT register, con `lgdt`. Habilitamos A20 para poder acceder a las posiciones mayores a 2^{20} , y seteamos el bit de PE de CR0 en 1.

Así ya con todo el contexto armado, ejecutamos la instrucción `jmp 18 << 3:mp` para hacer un `jmp` far a modo protegido donde, ponemos dentro del selector de segmento de código, el valor 18 que es la posición dentro de nuestro arreglo `gdt`, donde está el descriptor de código nivel 0.

2. Ejercicio 2

Inicialización de la IDT

En este ejercicio el tema que se va a tratar es el de inicializar la IDT. Para ello, primero se declaró la estructura de la IDT en `idt.c`, como un arreglo de 256 entradas de `idt_entry`.

En este punto lo que haremos va a ser simplemente llenar las primeras 31 entradas de este arreglo, con las interrupciones definidas por Intel. Para ello se utilizó una macro, ya definida, en la cual asignamos a todas las entradas por igual, como segmento de código en el cual se van a ejecutar, el número 18 dentro de la GDT, por ser este de nivel 0. Y en los atributos, el valor de 0x8E00, lo cual especifica, que cada excepción tiene el bit de precencia encendido, nivel de privilegio 0, y que es una interrupción de 32 bits. Y luego a cada entrada dentro del campo `offset`, le escribimos la dirección del handler de la misma.

Los handler, de las interrupciones, fueron declarados en `isr.h` e implementados en `isr.asm`. Donde a cada uno, simplemente se le pasó la tarea de imprimir por pantalla el número de interrupción que representa. Para ello en el área de memoria de la pantalla imprimimos el texto que según Intel, representa dicha interrupción.