Nº Orden	Apellido y nombre	L.U.	Cantidad de hojas

Organización del Computador 2
Segundo parcial – 28/06/2011

1 (45)	2 (45)	3 (10)	

Normas generales

- Numere las hojas entregadas. Complete en la primera hoja la cantidad total de hojas entregadas.
- Entregue esta hoja junto al examen, la misma **no** se incluye en la cantidad total de hojas entregadas.
- Está permitido tener los manuales y los apuntes con las listas de instrucciones durante el examen. Está prohibido compartir manuales o apuntes entre alumnos en el examen.
- Cada ejercicio debe realizarse en hojas separadas y numeradas. Debe identificarse cada hoja con nombre, apellido y LU.
- La devolución de los exámenes corregidos es personal. Los pedidos de revisión se realizarán por escrito, antes de retirar el examen corregido del aula.
- Los parciales tienen tres notas: I (Insuficiente): 0 a 59 pts, A- (Aprobado condicional): 60 a 64 pts y A (Aprobado): 65 a 100 pts. No se puede aprobar con A- ambos parciales. Los recuperatorios tienen dos notas: I: 0 a 64 pts y A: 65 a 100 pts.

Ej. 1. (45 puntos)

1. (10 puntos) Describa como completaría las primeras entradas de la GDT respetando los siguientes segmentos:

Comienzo	Tamaño	Uso	Atributos
0 Gb	2 Gb	código	No Lectura / nivel 0
2 Gb	1.5 Gb	datos	Escritura / nivel 3
512 Mb	4 Kb	datos	No Escritura / nivel 0
3 Gb	256 Mb	código	Lectura / nivel 3

   Indique los valores `base` y `límite` en hexadecimal. Dibuje los segmentos indicando como se solapan.

2. (15 puntos) Escriba todas las entradas de las estructuras que se requieran para construir el siguiente esquema de paginación, suponiendo que todas las entradas no mencionadas son nulas.

Rango virtual	Rango físico
`0x20000000` a `0x20004FFF`	`0x5AA0A000` a `0x5AA0EFFF`
`0xC0000000` a `0xC0002FFF`	`0x000AA000` a `0x000ACFFF`

   Todos los rangos incluyen el último valor. Se deben setear los permisos como supervisor.
3. (15 puntos) Resuelva las siguientes direcciones, desde lógica a física, pasando por lineal; utilizando las estructuras construidas en los ítems anteriores. Indique si se produce un error de protección y en qué unidad se genera.
   - `0x0008:0x2000171A`
   - `0x0010:0x40002832`
   - `0x0018:0x0000071A`
   - `0x0020:0x00000001`
4. (5 puntos) Suponiendo que se construye un esquema de paginación donde ninguna página utiliza identity mapping, ¿es posible activar paginación en estas condiciones?

Ej. 2. 45 puntos

En el sistema operativo Orga2SO, cada tarea se ejecuta durante una cantidad de ticks determinada por la constante `QUANTUM`. Cada tarea puede estar en distintos estados: `CORRIENDO`, `DURMIENDO`, `DIFUNTA` o `LISTA` (para correr). Para llevar cuenta de esto, las tareas se mantienen en un arreglo donde se almacena

el índice en la GDT donde se encuentra su descriptor de TSS, el estado de la misma y la cantidad de ticks por el cuál la tarea va a estar dormida (0 si la tarea no está dormida). Las tareas se ejecutan (solo se pueden ejecutar aquellas que están en estado `LISTA`) en orden desde el principio del arreglo (y de manera cíclica). En caso de que ninguna tarea esté en condiciones de ejecutarse se ejecuta la tarea `IDLE`.

Las estructuras de datos utilizadas son la siguientes:

```
typedef enum {
  CORRIENDO = 0, DURMIENDO = 1, DIFUNTA = 2, LISTA = 3
} estado_t;

typedef struct {
  estado_t estado;
  unsigned short indice_gdt;
  unsigned short despertar_en;
} __attribute__ ((__packed__)) tarea_t;

tarea_t tareas[CANT_TAREAS];
unsigned short indice_tarea_actual;
```

Se pide:

1. (15 puntos) Implementar en lenguaje ensamblador el código correspondiente al scheduler para que se ejecute en la interrupción del timer tick.
2. (15 puntos) Se desea poder poner a dormir a una tarea mediante la interrupción `0x60`. Implementar en lenguaje ensamblador el handler de la interrupción, en `_isrDormir`, recibiendo por `ax` la cantidad de ticks de reloj que la tarea va estar dormida. Al ser llamada la interrupción, la tarea que la llamó se debe poner a dormir, y se debe pasar a ejecutar la siguiente tarea en estado `LISTA`.
3. (15 puntos) Implementar en lenguaje ensamblador la interrupción de teclado de modo que cada vez que se presione la tecla `Del` se mate a la tarea actual, es decir, se cambie su estado a `DIFUNTA` y se pase a ejecutar la próxima tarea disponible.

Aclaraciones:

1. El tamaño del tipo de datos `estado_t` es de 4 bytes.
2. La tarea `IDLE` se encuentra en la posición 0 del arreglo de tareas y no se puede matar.
3. Se puede llamar a una función `proxima_tarea_lista` que devuelve en `ax` el índice de la próxima tarea en estado `LISTA`. Si no hay ninguna tarea en estado `LISTA`, esta función devuelve la tarea `IDLE`.
4. Se puede llamar a una función `decrementar_tick` que decrementa el campo `despertar_en` de cada tarea en el arreglo `tareas` que esté en estado `DURMIENDO`. En caso de que el campo `despertar_en` llegue a 0, esta función actualiza el campo `estado` a `LISTA`.
5. El scan code de la tecla `Del` es `0x53`.
6. Tener en cuenta que varias funcionalidades a implementar van a ser utilizadas en todos los ítems de este ejercicio.

Ej. 3. (10 puntos)

La entrada en el directorio de páginas de una página tiene permisos de Supervisor y Sólo-Lectura; la entrada en la tabla de páginas de la misma página tiene permisos de Usuario y Sólo-Lectura.

1. (5 puntos) ¿Qué sucede con los permisos efectivos de esta página? Suponga que el flag `CR0.WP` es cero, y considere los casos de acceso por parte de código corriendo con privilegio de Usuario, y código corriendo con privilegio de Supervisor.
2. (5 puntos) ¿Qué sucede si el flag `CR0.WP` es uno? Explique el comportamiento de este flag.

Nota: Revise la sección 11 del capítulo de Protección del manual de Intel.