**Universidad Tecnológica Nacional**
**Facultad Regional Buenos Aires**
**Departamento de Ingeniería Electrónica**

**Técnicas Digitales III**
23 de Septiembre de 2010
Examen Final

| Apellido y Nombres | Legajo | Calificación |
|---|---|---|
| | | |

**Parte Teórica : Tiempo Límite 45 minutos**

1) Unidad de Paginación:
   - Cómo se activa y configura?
   - Qué funciones cumple en el sistema?
   - Cómo realiza la paginación de la memoria?

2) Memoria Cache:
   - Qué función cumple en el sistema?
   - Diagrama de conexión y estructura de la lógica de control.
   - Describa una situación dónde se den problemas de coherencia con la memoria principal.

3) Linux:
   - Qué es un proceso?
   - Qué es un thread?
   - En qué se diferencian?
   - Cómo implementa Linux ambos?
   - Cómo se pueden comunicar diferentes entre si diferentes procesos, y qué condiciones tienen que darse para ello?
   - Cómo se comunican entre si diferentes threads y qué condiciones tienen que darse para ello?

4) Linux Device Drivers:
   - Qué es un módulo?
   - Cómo se instala en el sistema (en un kernel 2.6) y que requisitos debe cumplir?
   - Cuál es la API que debe implementar, y cómo hace el sistema para vincularla con el código?
   - Cómo hace el código de un usuario para acceder a las funciones del driver, y qué requisitos debe cumplir?

**Universidad Tecnológica Nacional**
**Facultad Regional Buenos Aires**
**Departamento de Ingeniería Electrónica**

**Técnicas Digitales III**
23 de Septiembre de 2010
Examen Final

| Apellido y Nombres | Legajo | Calificación |
|---|---|---|
| | | |

**Parte Práctica : Tiempo Límite 1hora 30 minutos**

Sobre la base de un sistema embedded IA32 que tiene las siguientes características:

- Implementa multitasking por hardware, pero cooperativo, mediante una llamada al sistema ejecutando INT 0x80.
- El scheduler usa para implementar su lógica solo la GDT.
- Implementa el resto de las llamadas al sistema mediante una llamada a INT 0x81
- Maneja un máximo de 100 tareas de DPL=3, y las llamadas al sistema ejecutan código de DPL=0.
- Tiene un segmento de datos de DPL=0 dedicado para almacenar la información de todos los TSS (KERNEL_TSS_DATA)
- Tiene un segmendo de datos de DPL=3 dedicado para almacenar los stacks de todas las tareas (USER_STACK_DATA), de forma continua, que tendrán una longitud máxima de USER_STACK_LEN.
- Tiene un segmendo de datos de DPL=0 dedicado para el stack del scheduler.

Se pide:

a) Desarrolle una rutina de control que se ejecutará con cada IRQ0, y que controlará que ninguna tarea exceda 1000 los ticks de ejecución, en cuyo caso forzará la conmutación a la siguiente tarea.

b) Desarrolle una system call que clone la tarea actual, y devuelva:
   - 1 en caso de éxito, es decir, que se pudo clonar la tarea.
   - 0 en caso de que la tarea sea la clonada, y no la original.
   - (-1) si no se pudo crear por exceder el número máximo de tareas.

c) El contenido y posición de las entradas de la IDT para implementar las system calls, la rutina de la IRQ0 y explique cómo se pasan los parámetros en los casos que fueran necesarios.