Examen de Periféricos Interfaces y Buses

Septiembre 2008

1) (2 puntos)

Escribir la función de instalación del módulo de un driver (Linux) y proponer las estructuras de datos necesarias para un dispositivo que permite dos modos de funcionamiento. Ambos modos tienen las operaciones de apertura y cierre del dispositivo, así como las de lectura, escritura e ioctl. Los datos que necesita el driver para los diferentes modos son los siguientes:

- Modo 1: dos números enteros, y un string de hasta 50 caracteres
- Modo 2: cinco números reales

Explicar cómo se debe instalar el driver para su uso en los diferentes modos desde un programa de aplicación.

2) (2 puntos)

Escribir la función `ioctl` de un driver de Linux para un puerto serie que permita modificar o establecer los siguientes parámetros de funcionamiento:

- velocidad de transmisión: se permiten los valores 6400, 9600 y 19200 bps
- tamaño máximo del mensaje que se transmite (en bytes): se guarda en una variable estática accesible desde todas las funciones del driver que se llama `tam_max` (su valor debe estar entre 50 y 500)

Diseñar la estructura de datos necesaria para el intercambio de información con la función ioctl y la codificación del comando o comandos. Si alguno de los valores pasados como argumento no está en el rango permitido, la llamada debe retornar un error.

3) (2 puntos)

Se tiene una aplicación del control de nivel de un depósito de agua en la que el programa necesita conocer periódicamente el valor del nivel suministrado por un sensor. La especificación determina que la lectura del sensor debe obtener un nuevo dato, pero si transcurridos 100 milisegundos no se obtiene un nuevo valor, se debe devolver el valor anterior.

El acceso al sensor se realiza a través de dos registros:

- *Valor del Sensor*: 16 bits en la posición 6A0h de la memoria de I/O, que mantiene el valor de la última lectura del sensor.
- *Control del Sensor*: La escritura de cualquier valor en este registro hace que el sensor comience la medida del nivel. Cuando tiene un valor estable se modifica *Valor del Sensor* y se produce un interrupción (IRQ5). La lectura de cualquier valor en el registro anula la medida pendiente y no se modifica *Valor del Sensor.*

Escribir la función de lectura y la función manejadora de la interrupción del driver Linux que permite a la aplicación obtener el valor del nivel del agua del depósito de acuerdo a las especificaciones. Diseñar las estructuras de datos necesarias y utilizar los mecanismos de sincronización más adecuados. Justificar la propuesta realizada.

4) (2 puntos)

Se desea programar una interfaz para un dispositivo contador de las personas que pasan por un determinado lugar en el que está instalado. El dispositivo anota el número de personas detectadas correctamente, así como también el número de detecciones que no son fiables (errores).

Para el acceso al hardware se dispone de la siguiente función:

```
void lee_contador (struct contador *cuenta);
// Devuelve en el parámetro cuenta los valores de los dos
// contadores: personas detectadas, y errores
```

La `struct contador` se define como sigue:

```
struct contador {
   long personas;
   long errores;
};
```

El objetivo es permitir que el driver Linux suministre a la aplicación el valor de los contadores (personas y errores) a través de la operación de lectura (que no hay que hacer). El driver debe leer los valores del dispositivo cada segundo y anotarlos en una variable interna del driver, que es la que la aplicación podrá consultar.

Proponer el código del driver que permita realizar la actualización de los contadores. Se deben realizar las estructuras de datos y de programa necesarias para cumplir la especificación (también el código que pudiera ir en la inicialización del driver).

5) (2 puntos)

Se tiene un dispositivo conectado al bus PCI cuya identificación viene dada por las constantes:

```
#define FABRICANTE  345    // identificador del fabricante
#define DISPOSITIVO 12     // identificador del dispositivo
```

La región de memoria válida para este dispositivo en su espacio de configuración es la número 3 y pertenece al espacio de memoria de I/O. Desarrollar una función que sea capaz de escribir el string "Acceso Posible" en la dirección de memoria 0x10 del dispositivo, y devuelva el valor del byte de la posición de memoria 0x00 (ambas posiciones relativas a la dirección base). La función debe realizarse teniendo en cuenta que puede formar parte del código de un driver Linux.