Prácticas de Periféricos Interfaces y Buses

3er Curso de Ingeniería Informática

UNIVERSIDAD DE CANTABRIA

Práctica 1: Gestión de drivers de dispositivos - cola circular

Objetivos:

- Practicar la instalación, desinstalación y uso de drivers de dispositivos en Linux

Descripción:

- Modificar el ejemplo de buffer virtual visto en clase para que se comporte como un ***cola circular*** de caracteres en que se puede escribir hasta que se llene y se puede leer hasta que se vacíe.
- Se pueden realizar operaciones de lectura y escritura con tamaños arbitrarios.
- Realizar las pruebas de instalación y desinstalación del driver comprobando los resultados.

Práctica 1: Gestión de drivers de dispositivos - cola circular (cont.)

- Realizar dos programas de prueba:
  - uno para enviar al driver los datos introducidos por el usuario (en un lazo hasta que decida terminar)
  - el otro para obtenerlos y pintarlos en pantalla pidiendo al usuario la cantidad de caracteres que quiere leer.
- Ejecutar los programas desde dos terminales diferentes

Práctica 2: Dispositivos o modos de funcionamiento múltiples

Objetivos:

- Practicar el desarrollo de drivers de dispositivos en los que existen diferentes subunidades o subfunciones.

Descripción:

- Añadir a la cola circular de la práctica 1 un segundo modo de funcionamiento en el que la operación de lectura devuelva siempre un número de caracteres fijo (por ejemplo 5).
- Para ello utilizar dos números menores, uno asociado a cada comportamiento de la cola.
- Realizar las dos posibles implementaciones descritas a continuación:

Práctica 2: Dispositivos o modos de funcionamiento múltiples (cont.)

1. Modificar el comportamiento del driver eligiendo el modo de funcionamiento según el número menor en la operación `open`
2. Hacer dos conjuntos de operaciones e instalarlas como dispositivos de caracteres diferentes (desde la misma función de instalación del módulo)

- Realizar pruebas con los programas de la práctica 1 en los que haya un escritor y dos lectores (uno en cada modo)

Práctica 3: Control de la concurrencia en los drivers

Objetivos

- Experimentar con la concurrencia en los drivers mediante el control del acceso y mediante el uso de mecanismos de sincronización en los datos.

Descripción

- Modificar el driver de la práctica 1 para controlar la concurrencia de dos modos:
  - controlando el acceso en el `open` no permitiendo la apertura por más de un proceso
  - controlando las estructuras de datos internas con mecanismos de acceso a las secciones críticas (la cola será un recurso compartido)

Práctica 3: Control de la concurrencia en los drivers

- Realizar dos programas de prueba:
  - uno que escriba periodicamente los datos pasados por el usuario al inicio
  - otro que lea datos periodicamente y los muestre en pantalla
- Realizar pruebas con varios procesos leyendo y varios escribiendo (desde diferentes terminales):
  - probar que el driver con control de la concurrencia no falla
  - verificar que los programas de prueba fallan con el driver de la práctica 1 (forzar si es necesario artificialmente las condiciones de fallo)

Práctica 4: Programación del puerto serie

Objetivos

- Experimentar con la programación del puerto serie y el uso de interrupciones

Descripción

- Modificar el driver del puerto serie visto en clase para que utilice la cola circular implementada en la práctica 1 con las siguientes características:
  - en las colas sólo se gestionarán mensajes completos identificados por cada llamada individual a ***read*** o ***write***
  - hay que tener en cuenta que ahora habrá dos colas cuyo uso se debe sincronizar con el manejador de interrupción
  - por otro lado, también hay que tener en cuenta el acceso concurrente a las operaciones de lectura y escritura del driver

Práctica 4: Programación del puerto serie (cont.)

- Para la identificación de un mensaje se deben añadir caracteres especiales de comienzo y finalización en el momento de la escritura; usar el siguiente protocolo:
  - STX (0x02) para el comienzo
  - ETX (0x03) para la finalización
- Realizar programas de prueba en los que se pueda comprobar:
  - el correcto funcionamiento de la transmisión y recepción de los mensajes
  - el correcto funcionamiento de los accesos concurrentes a las operaciones de lectura y escritura
  - utilizar velocidades de transmisión bajas para hacer pruebas en lazo cerrado
- Realizar pruebas por parejas con los drivers y programas de cada cual para comprobar que todo sigue funcionando bien

Práctica 5: Gestión de dispositivos en sistemas mínimos

UNIVERSIDAD DE CANTABRIA

Objetivos

- Practicar la instalación, desinstalación y uso de drivers de dispositivos en MaRTE OS

Descripción

- Adaptar la cola circular de caracteres desarrollada en la práctica 1 al entorno de programación de drivers del sistema operativo MaRTE
- Realizar un programa de prueba que en un lazo
  - escriba un mensaje dado por el usuario
  - y después lea el número de caracteres indicado por el mismo

Práctica 5: Gestión de dispositivos en sistemas mínimos (cont.)

UNIVERSIDAD DE CANTABRIA

- Opcionalmente, adaptar los programas de prueba de la práctica 1 para que funcionen en el entorno monoproceso de MaRTE OS:
  - integrar el código de los programas de prueba en threads de MaRTE OS, creando posteriormente los que sean necesarios

Práctica 6: Programación de drivers en sistemas heterogéneos

Objetivos

- Experimentar la programación de drivers para la interconexión de dos sistemas diferentes

Descripción

- Adaptar el driver del puerto serie realizado en la práctica 4 al sistema operativo MaRTE
- Utilizar los programas de prueba realizados en la práctica 4 para realizar las pruebas de integración
  - comunicación de los dos sistemas Linux/MaRTE OS con transmisión y recepción simultánea

Práctica 7: Programación de dispositivos PCI

Objetivos

- Experimentar con la programación de dispositivos conectados al bus PCI, con interfaces de entradas y salidas analógicas y digitales y con la operación *ioctl*

Descripción

- Realizar un sistema de muestreo y reconstrucción de una señal analógica utilizando la tarjeta PCI-9111
- Escribir un driver sencillo:
  - capaz de digitalizar el valor de la señal de entrada como un comando de la operación *ioctl*
  - capaz de establecer el valor de la señal analógica de salida también como un comando de la operación *ioctl*

Práctica 7: Programación de dispositivos PCI (cont.)

- Escribir un programa de aplicación que a petición del usuario:
  - tome un número determinado de muestras de una señal analógica de audio
  - reproduzca la señal almacenada que previamente se ha digitalizado

Práctica 8: Digitalización y reproducción remota de señales

Objetivos

- Realizar una aplicación completa en la que se experimente con los drivers programados en el desarrollo de la asignatura, para determinar su posibilidad de uso y detectar posible deficiencias de diseño, implementación, etc.

Descripción

Realizar una aplicación capaz de llevar a cabo las siguientes actividades:

- Muestreo de una señal analógica utilizando la tarjeta PCI-9111 (en el sistema Linux)
- Envío de cada dato muestreado al sistema MaRTE a través del puerto serie

Práctica 8: Digitalización y reproducción remota de señales (cont.)

- Reproducción del dato recibido en el sistema MaRTE utilizando la tarjeta de salidas analógicas PCM-3712

La aplicación debe permitir ajustar la frecuencia a la que se quiere muestrear la señal

Utilizar el programa para muestrear/reproducir señales de audio

Verificar la frecuencia máxima de operación así como la calidad de la reproducción conseguida a diferentes frecuencias de operación