Manual de Usuario - Driver IIE-PCI

Tabla de contenido

Características del driver

Las principales características del driver son las siguientes:

- licencia de libre distribución
- compatible con Kernel Linux 2.4.x
- mapea recursos de la placa PCI como dispositivos de caracteres
- simple y fácil de utilizar
- simple y fácil de adaptar
- diseño modular, permite ser cargado en tiempo de ejecución, no es necesario compilarlo con el kernel.
- provee información de performance, cantidad de bytes transferidos, tasas de transferencia, etc.
- provee una generosa cantidad de información sobre valores y datos durante la ejecución, facilitando el debugging de aplicaciones.

Utilización del driver

Compilación del driver

Para compilar el driver utilizando (**rw_bar.c**) se deben utilizar los siguientes parámetros y símbolos:

```
CFLAGS = -D__KERNEL__ -DMODULE -Wall -O
```

El símbolo __KERNEL__ habilita muchas funciones útiles dentro de los cabezales de kernel. El símbolo MODULE debe ser incluido para todos aquellos drivers que no sean compilados dentro del kernel. El parámetro -Wall habilita el despliegue de todos los mensajes de advertencia del compilador. El parámetro -O es necesario porque muchas funciones están declaradas *inline* en los cabezales y el el compilador *gcc* no expande las funciones inline a menos que la optimización esté habilitada.

Junto con el código fuente del driver se incluye un archivo *Makefile* para facilitar el proceso de compilación.

Instalación y desinstalación del driver

Para instalar el módulo se utiliza el programa *insmod* , al que se le pasan como parámetros el nombre del módulo, y los parámetros extra que este requiera. En nuestro caso es necesario especificar la cantidad de espacios de memoria PCI utilizados (cantidad de BAR).
Un ejemplo de la carga del módulo utilizando tres espacios de memoria PCI sería:

```
/sbin/insmod -f ./rw_bar.o "used_bars=3"
```

Para simplificar la tarea de determinar la cantidad de espacios de memoria PCI utilizados por el dispositivo se utiliza un script (*load_rw_bar.sh*), que hace uso de la herramienta *lspci* explicada más adelante. También en este script se crean los nodos en el sistema de archivos para acceder a cada espacio de memoria PCI.

Deben de completarse los siguientes parámetros en el script:

- vendor_id="nnnn". *nnnn* es un número de 4 dígitos que identifica al fabricante del dispositivo PCI. Por ejemplo a Intel le corresponde 8086, en el core PCI desarrollado este valor es configurable.

- device_id="dddd". *dddd* es una cadena de 4 caracteres que identifica los dispositivos del fabricante.
- module="modulo". *modulo* debe ser el nombre con el cual fue compilado el driver.
- device="nodo". *nodo* es el nombre con el que se crearán los nodos en /dev
- mode="xxx". *xxx* permisos asignados a los nodos creados.

Por ejemplo:
vendor_id="1172"
device_id="abba"
module="rw_bar"
device="rw_bar"
mode="664"

El vendor_id="1172" corresponde a ALTERA.

En caso de que la aplicación utilice 3 BARs, se crean los nodos:
/dev/rw_bar0, correspondiente a BAR0
/dev/rw_bar1, correspondiente a BAR1
/dev/rw_bar2, correspondiente a BAR2

Al instalarlo se muestra información sobre las direcciones del espacio PCI asignadas a cada BAR y el tamaño de estos.

La descarga del driver utiliza el programa *rmmod* pasándole como parámetro el nombre del módulo. Para simplificar la tarea se utiliza un script(*unload_rw_bar.sh*)
El mismo desinstala el módulo y elimina los nodos creados con *load_rw_bar.sh*.
Los parámetros a completar son los mismos que los del script de instalación.

Lecturas y escrituras a BARs

Una vez instalado el driver, cada uno de los BAR estará mapeado como un nodo en el sistema de archivos especiales /dev. Por lo que realizar lecturas y escrituras a cada uno de los BAR es como realizar lecturas y escrituras de un archivo.

Se desarrollaron un par de scripts basados en el lenguaje Perl, uno de lectura y otro de escritura. Ambos scripts permiten manipular un byte en particular, ubicado en cualquier posición dentro del espacio de memoria, utilizando un cierto *offset* o desplazamiento a partir del primer byte.

roff

El script *roff* permite leer desde un archivo una cierta cantidad de bytes especificada, a partir de un cierto desplazamiento deseado. El comando posee ayuda en linea.

Uso:
roff.pl [-C] [-x] [-b] [-o offset] [-c count] [-s buffsize] archivo

Parámetros requeridos:
- archivo de lectura

Parámetros opcionales:
- -C muestra un dump con el programa "hexdump -C"
- -b utiliza la lectura buffereada (read). Por defecto se utiliza sysread, que no es buffereada
- -o byte de offset de inicio de la lectura. Por defecto = 0
- -c cantidad de bytes a leer a partir del offset. Por defecto = 128
- -x convierte la salida a representación ASCII hexa (un byte -> dos caracteres entre 0-9,A-F)
- -s tamaño del buffer de lectura a utilizar, en bytes

Un ejemplo de utilización sería:

```
./roff.pl -C -o 2 -c 30 /dev/rw_bar1
```

Se leen 30 bytes (parámetro -c) a partir del byte 2 (parámetro -o) del archivo /dev/rw_bar1, y la salida se presenta en pantalla en forma hexadecimal y ascii en forma de columnas (parámetro -C).

woff

El script *woff* permite escribir a un archivo una cierta cantidad de bytes, a partir de un cierto desplazamiento deseado. Los datos a escribir son recibidos por la entrada estándar del programa.

Uso:
woff.pl [-x] [-b] [-v] [-o offset] [-c count] [-s buffsize] archivo

Parámetros requeridos:
* archivo de lectura * datos por entrada estándar

Parámetros opcionales
- -x interpreta la entrada estándar representada como ASCII hexa
- -b utiliza la escritura buffereada (print). Por defecto se utiliza syswrite, que no es buffereada
- -o byte de offset de inicio de la lectura. Por defecto = 0
- -c cantidad de bytes a leer de la entrada. Ver explicación más adelante de su uso junto con -x.
- -s tamaño del buffer de lectura (no de escritura) a utilizar, en caracteres o bytes, dependiendo de si se utiliza el modo -x o no.

El uso del modo -x hace que se interprete la entrada como ASCII hexadecimal. Esto significa que los caracteres de la entrada se juntan de a pares para generar un byte, que es lo que que finalmente se escribe en el archivo de salida. Por ejemplo, el par de caracteres ASCII hexa "FF" corresponden a un byte cuyo valor es 255.
El uso de este modo afecta a los valores de -c y -s. Ambos pasan a interpretarse en caracteres y no en bytes. Si se quiere escribir con un buffer de tamaño 4, el valor de -s debe ser 8 cuando se lo usa junto con -x.

Un ejemplo sería el siguiente:

```
echo -n "testing" | ./woff.pl -o 2 /dev/rw_bar1
```

En este caso se escribe el texto "testing" al archivo /dev/rw_bar1, tomando como posición inicial de escritura el byte 2 dentro del archivo (parámetro -o).
El uso del parámetro -x permite escribir cualquier valor deseado, representado en forma hexadecimial. El siguiente ejemplo escribe el numero 255 en el primer byte del archivo /dev/rw_bar1:

```
echo -n ff | ./woff.pl -x /dev/rw_bar1
```

NOTA: la variable de entorno LANG no debe estar configurada para utilizar caracteres UTF8. Se recomienda LANG=en_US.

Estadística de transferencias

El driver permite conocer estadísticas de las transferencias realizadas a cada BAR.
Sus valores se obtienen desplegando el contenido del archivo especial /proc/driver/iiepci, como se muestra a continuación:

Uso:

```
#cat /proc/driver/iiepci

--------- IEEPCI Stats - HZ :  512 - loops_per_jiffy:   778240 -------

--------- W R I T E  --  M E M C O P Y -------------------------------
          |   KB/sec     |     bytes    | microseconds | tsc loops
----------+--------------+--------------+--------------+--------------
TOTAL     |         32231|         26624|           806|        321425
BAR1 TOTAL|         32231|         26624|           806|        321425
BAR2 TOTAL|             0|             0|             0|             1
BAR3 TOTAL|             0|             0|             0|             1
LAST WRITE|         32468|           512|            15|          6136
--------- R E A D  --  M E M C O P Y ---------------------------------
          |   KB/sec     |     bytes    | microseconds | tsc loops
----------+--------------+--------------+--------------+--------------
TOTAL     |          5472|      10058788|       1795103|     715274956
BAR1 TOTAL|          5472|      10058788|       1795103|     715274956
BAR2 TOTAL|             0|             0|             0|             1
BAR3 TOTAL|             0|             0|             0|             1
LAST READ |          2829|             4|             1|           550
```

También se recomienda el uso de la utilidad *watch* de Linux (parte del paquete procps de RedHat) que reitera un comando cada una cierta cantidad de segundos, especificad por el parámetro -n.
Ejemplo:

```
# watch -n 1 cat /proc/driver/iiepci
```

Este comando despliega automáticamente las estadísticas en pantalla, cada 1 segundo.

Herramientas para dispositivos PCI del sistema operativo Linux

Linux posee abundantes herramientas para el desarrollo, testeo y debugging de hardware PCI. Mencionamos a continuación algunas de las más utilizadas en nuestro proyecto.

/proc/pci

El archivo especial /proc/pci contiene un listado de todos los dispositivos PCI presentes en el sistema e información específica de cada uno. Entre la información desplegada se encuentra:

- número de bus en el que se encuentra el dispositivo
- tipo de dispositivo
- fabricante y modelo del dispositivo
- irq utilizada
- master o target, latencia, etc.
- información sobre direcciones de memoria utilizadas y su tipo.

Es muy útil para obtener información rápidamente.

/proc/bus/pci

El sistema de archivos especial /proc/bus/pci permite obtener y modificar las configuraciones de los dispositivos PCI presentes. Su uso directo no es sencillo y es recomendable utilizar herramientas para modificar estos valores, como ser *setpci* (mencionado más adelante).

pciutils

El paquete pciutils (que forma parte de la mayoría de las distribuciones de kernel 2.4) contiene dos aplicaciones extremadamente útiles: *lspci* y *setpci* .
Dichas aplicaciones permiten inspeccionar y configurar los dispositivos conectados al bus PCI, y realizan la mayoría de sus operaciones a través del sistema de archivos /proc/bus/pci

La aplicación *lspci* despliega información sobre todos los buses PCI presentes en el sistema, y los dispositivos que estén conectados en ellos.

La aplicación *setpci* permite obtener y modificar valores de configuración de los dispositivos PCI conectados. Esta aplicación permite hacer cambios en los registros de configuración de un dispositivo en forma muy sencilla. Fue utilizada extensivamente en las primeras etapas de desarrollo del core PCI.