Interfaz de Programación del Controlador MCA

Alan Cox

alan@redhat.com

David Weinehall

Chris Beauregard

Interfaz de Programación del Controlador MCA

por Alan Cox, David Weinehall, y Chris Beauregard

Tabla de contenidos

Capítulo 1. Introducción

Las funciones del bus MCA suministran una interfaz generalizada para encontrar tarjetas del bus MCA, pedirles un controlador, y para leer y manipular los registros POS sin ser conscientes de los entresijos de la placa madre o de cierta magia profunda específica de los dispositivos de la placa.

La interfaz básica de los dispositivos del bus MCA es el slot. Cada slot está numerado y los números de slots virtuales son asignados a los dispositivos internos. Realmente el usar pci_dev como con otros buses no tiene sentido en los contextos MCA, ya que los recursos del bus MCA requieren una interpretación específica de la tarjeta.

Finalmente, las funciones del bus MCA suministran un conjunto paralelo de funciones DMA parecidas a las funciones DMA del bus ISA, tan cerradamente como es posible, aunque también soportan las funcionalidades adicionales DMA en las controladoras del bus MCA.

Capítulo 2. Bugs Conocidos Y Suposiciones

Ninguno.

Capítulo 3. Funciones Públicas Suministradas

mca_find_adapter

Nombre

`mca_find_adapter` — busca adaptadores

Sinopsis

```
int mca_find_adapter (int id, int start);
```

Argumentos

`id`

Identificación del bus MCA en el que buscar

`start`

slot de comienzo

Descripción

Busca en la configuración del MCA adaptadores que se correspondan con la identificación de 16 bits. La primera vez debería de ser llamado para que empiece en cero, y entonces las llamadas posteriores que se harán pasando el valor de retorno de la llamada previa hasta que sea retornado `MCA_NOTFOUND`.

Los adaptadores deshabilitados no son reportados.

mca_find_unused_adapter

Nombre

`mca_find_unused_adapter` — busca adaptadores sin usar

Sinopsis

```
int mca_find_unused_adapter (int id, int start);
```

Argumentos

id

Identificación del bus MCA en el que buscar

start

slot de comienzo

Descripción

Busca en la configuración del MCA adaptadores que se correspondan con la identificación de 16 bits. La primera vez debería de ser llamado para que empiece en cero, y entonces las llamadas posteriores que se harán pasando el valor de retorno de la llamada previa hasta que sea retornado `MCA_NOTFOUND`.

Los adaptadores que han sido reclamados por controladores o que están deshabilitados no son reportados. Esta función permite a un controlador buscar tarjetas adicionales cuando quizás alguien ya las controle.

mca_read_stored_pos

Nombre

`mca_read_stored_pos` — lee los registros POS de los datos de arranque

Sinopsis

```
unsigned char mca_read_stored_pos (int slot, int reg);
```

Argumentos

slot

número de slot desde donde leer

reg

registro desde donde leer

Descripción

Obtiene un valor POS que fue almacenado en tiempo de arranque por el núcleo cuando este buscó en el espacio del MCA. Es retornado el valor del registro. Registros perdidos o inválidos devuelven un 0.

mca_read_pos

Nombre

`mca_read_pos` — lee el registro POS de la tarjeta

Sinopsis

```
unsigned char mca_read_pos (int slot, int reg);
```

Argumentos

slot

número de slot desde donde leer

reg

registro de donde leer

Descripción

Obtiene un valor POS directamente del hardware para obtener el valor actual. Esto es mucho más lento que mca_read_stored_pos y quizás no sea llamado desde el contexto de interrupciones. Maneja la magia profunda requerida para los dispositivos en la placa de forma transparente.

mca_write_pos

Nombre

`mca_write_pos` — lee el registro POS de la tarjeta

Sinopsis

```
void mca_write_pos (int slot, int reg, unsigned char byte);
```

Argumentos

`slot`

número de slot desde donde leer

`reg`

registro de donde leer

`byte`

byte a escrigir en los registros POS

Descripción

Almacena un valor POS directamente al hardware. Normalmente no necesitarías usar esta función y deberías de tener un muy buen conocimiento del bus MCA antes de hacerlo. Realizando esto de forma incorrecta puedes dañar el hardware.

Esta función quizás no sea usada desde un contexto de interrupciones.

Nota que esto es técnicamente Algo Malo, como dice el equipo técnico de IBM, sólo deberías de establecer valores POS a través de sus utilidades. En todo caso, algunos dispositivos como el 3c523 recomiendan que les escribas algunos datos para asegurarte de que la configuración es consistente. Yo diría que IBM está en lo cierto, pero me gusta que mis controladores funcionen.

Esta función no puede chequear para ver si múltiples dispositivos están con los mismos recursos, por lo tanto quizás veas humo mágico si alguien se lía.

mca_set_adapter_name

Nombre

`mca_set_adapter_name` — Establece la descripción de la tarjeta

Sinopsis

```
void mca_set_adapter_name (int slot, char* name);
```

Argumentos

`slot`

slot a nombrar

`name`

cadena de texto para el nombre

Descripción

Esta función establece el nombre reportado a través de /proc para este slot adaptador. Esto es únicamente para información de usuario. Estableciendo un nombre borra cualquier nombre previo.

mca_set_adapter_procfn

Nombre

`mca_set_adapter_procfn` — Establece la retrollamada /proc

Sinopsis

```
void mca_set_adapter_procfn (int slot, MCA_ProcFn procfn, void* dev);
```

Argumentos

slot

slot a configurar

procfn

función de retrollamada a llamar para /proc

dev

información del dispositivo pasada a la retrollamada

Descripción

Establece una retrollamada de información para /proc/mca/slot?. La función es llamada con el buffer, slot y puntero al dispositivo (o alguna otra información igualmente informativa del contexto, o nada, si lo prefieres), y se espera que ponga información útil en el buffer. El nombre del adaptador, ID, y registros POS fueron imprimidos antes de esta llamada, por lo tanto no lo hagas otra vez.

Debería de ser llamada con un *procfn* `NULL` cuando un módulo se desregistra, entonces preveniendo que el núcleo rompa y otras semejantes cosas feas.

mca_is_adapter_used

Nombre

`mca_is_adapter_used` — chequea si fue pedido por un controlador

Sinopsis

```
int mca_is_adapter_used (int slot);
```

Argumentos

slot

slot a chequear

Descripción

Retorna 1 si el slot ha sido pedido por un controlador

mca_mark_as_used

Nombre

`mca_mark_as_used` — pide un dispositivo MCA

Sinopsis

```
int mca_mark_as_used (int slot);
```

Argumentos

`slot`

slot a pedir

FIXME

deberíamos hacer esto seguro para los threads

Pide un slot MCA para un controlador de dispositivo. Si el slot ya está tomado la función retorna 1, si no está tomado es pedido y es devuelto un 0.

mca_mark_as_unused

Nombre

`mca_mark_as_unused` — libera un dispositivo MCA

Sinopsis

```
void mca_mark_as_unused (int slot);
```

Argumentos

slot

slot a pedir

Descripción

Libera el slot para que otros controladoras lo usen.

mca_get_adapter_name

Nombre

`mca_get_adapter_name` — obtiene la descripción del adaptador

Sinopsis

```
char * mca_get_adapter_name (int slot);
```

Argumentos

slot

slot al que preguntar

Descripción

Retorna la descripción del adaptador si está establecida. Si no ha sido establecida o el slot está fuera de rango entonces devuelve NULL.

mca_isadapter

Nombre

`mca_isadapter` — chequea si el slot mantiene un adaptador

Sinopsis

```
int mca_isadapter (int slot);
```

Argumentos

`slot`

slot al que preguntar

Descripción

Retorna cero si el slot no mantiene un adaptador, distinto de cero si lo mantiene.

mca_isenabled

Nombre

`mca_isenabled` — chequa si el slot mantiene un adaptador

Sinopsis

```
int mca_isenabled (int slot);
```

Argumentos

`slot`

slot al que preguntar

Descripción

Retorna un valor distinto de cero si el slot mantiene un adaptador habilitado y cero para cualquier otro caso.

Capítulo 4. Funciones DMA Suministradas

mca_enable_dma

Nombre

`mca_enable_dma` — canal en el que habilitar el DMA

Sinopsis

```
void mca_enable_dma (unsigned int dmanr);
```

Argumentos

`dmanr`

canal DMA

Descripción

Habilita en el bus MCA el DMA en un canal. Puede ser llamado desde un contexto IRQ.

mca_disable_dma

Nombre

`mca_disable_dma` — canal en el que deshabilitar el DMA

Sinopsis

```
void mca_disable_dma (unsigned int dmanr);
```

Argumentos

dmanr

canal DMA

Descripción

Habilita en el bus MCA el DMA en un canal. Puede ser llamado desde un contexto IRQ.

mca_set_dma_addr

Nombre

mca_set_dma_addr — carga una dirección DMA de 24 bits

Sinopsis

```
void mca_set_dma_addr (unsigned int dmanr, unsigned int a);
```

Argumentos

dmanr

canal DMA

a

direcciones de bus de 24 bits

Descripción

Carga los registros de direcciones en el controlador DMA. Esto tiene una limitación de 24 bits (16Mb).

mca_get_dma_addr

Nombre

`mca_get_dma_addr` — carga una dirección DMA de 24 bits

Sinopsis

```
unsigned int mca_get_dma_addr (unsigned int dmanr);
```

Argumentos

`dmanr`

canal DMA

Descripción

Lee los registros de direcciones en el controlador DMA. Esto tenía una limitación de 24 bits (16Mb). El retorno es una dirección del bus.

mca_set_dma_count

Nombre

`mca_set_dma_count` — carga una cuenta de transferencia de 16 bits

Sinopsis

```
void mca_set_dma_count (unsigned int dmanr, unsigned int count);
```

Argumentos

`dmanr`

canal DMA

count

count

Descripción

Establece la cuenta DMA para este canal. Esto puede ser hasta 64Kbytes. Estableciendo una cuenta a cero no hará lo que esperas.

mca_get_dma_residue

Nombre

mca_get_dma_residue — obtiene los bytes que quedan por transferir

Sinopsis

```
unsigned int mca_get_dma_residue (unsigned int dmanr);
```

Argumentos

dmanr

canal DMA

Descripción

Esta función retorna el número de bytes que quedan por transferir en este canal DMA.

mca_set_dma_io

Nombre

mca_set_dma_io — establece el puerto para una transferencia de E/S

Sinopsis

```
void mca_set_dma_io (unsigned int dmanr, unsigned int io_addr);
```

Argumentos

dmanr

canal DMA

io_addr

un número de puerto de E/S

Descripción

A diferencia de los controladores DMA del bus ISA un bus MCA puede transferir con un puerto de E/S de destino.

mca_set_dma_mode

Nombre

mca_set_dma_mode — establece el modo de DMA

Sinopsis

```
void mca_set_dma_mode (unsigned int dmanr, unsigned int mode);
```

Argumentos

dmanr

canal DMA

mode

modo a establecer

Descripción

El controlador DMA soporta varios modos. Los modos que puedes tener son

set are

`MCA_DMA_MODE_READ` para leer de un dispositivo DMA.

`MCA_DMA_MODE_WRITE` para escribir al dispositivo DMA.

`MCA_DMA_MODE_IO` para hacer DMA a o desde un puerto de E/S.

`MCA_DMA_MODE_16` para hacer transferencias de 16 bits.

Capítulo 5. Sobre la Traducción

Este documento es la traducción de "MCA Driver Programming Interface", documento que acompaña al código del núcleo de Linux, versión 2.4.18.

Este documento ha sido traducido por Rubén Melcón <melkon@terra.es>; y es publicado por el Proyecto Lucas (http://lucas.hispalinux.es)

Versión de la tradución 0.04 (Julio de 2002).

Si tienes comentarios sobre la traducción, ponte en contacto con Rubén Melcón <melkon@terra.es>