NAME

ch - SCSI media-changer (juke box) driver

SYNOPSIS

device ch

DESCRIPTION

The **ch** driver provides support for a *SCSI* media changer. It allows many slots of media to be multiplexed between a number of drives. The changer device may optionally be equipped with a bar code reader, which reads label information attached to the media.

A SCSI adapter must also be separately configured into the system before a SCSI changer can be configured.

As the SCSI adapter is probed during boot, the *SCSI* bus is scanned for devices. Any devices found which answer as 'Changer' type devices will be 'attached' to the **ch** driver. In FreeBSD releases prior to 2.1, the first found will be attached as *ch0* and the next, *ch1* etc. Beginning in 2.1 it is possible to specify what ch unit a device should come on line as; refer to scsi(4) for details on kernel configuration.

KERNEL CONFIGURATION

It is only necessary to explicitly configure one **ch** device; data structures are dynamically allocated as media changes are found on the SCSI bus.

IOCTLS

User mode programs communicate with the changer driver through a number of ioctls which are described below. Changer element addresses used in the communication between the kernel and the changer device are mapped to zero-based logical addresses. Element types are specified as follows:

CHET_MT	Medium transport element (picker).
CHET_ST	Storage element (slot).
CHET_IE	Import/export element (portal).
CHET_DT	Data transfer element (drive).

The following ioctl(2) calls apply to the changer. They are defined in the header file <*sys/chio.h*>.

CHIOMOVE (*struct changer_move*) Move a medium from one element to another (**MOVE MEDIUM**) using the current picker. The source and destination elements are specified in a changer_move structure, which includes at least the following fields:

```
u_int cm_fromtype; /* element type to move from */
u_int cm_fromunit; /* logical unit of from element */
u_int cm_totype;   /* element type to move to */
u_int cm_tounit;   /* logical unit of to element */
u_int cm_flags;         /* misc. flags */
```

If the CM_INVERT in the *cm_flags* field is set, the medium changer is instructed to flip the medium while moving it.

CHIOEXCHANGE (*struct changer_exchange*) Move the medium located in the source element to the first destination element, and move the medium that had been in the first destination element to the second destination element. In case of a simple exchange, the source and second destination elements should be the same. The current picker is used to perform the operation. The addresses of the affected elements is specified to the ioctl in a *changer_exchange* structure which includes at least the following fields:

```
u_int ce_srctype;     /* element type of source */
u_int ce_srcunit;     /* logical unit of source */
u_int ce_fdsttype; /* element type of first destination */
u_int ce_fdstunit; /* logical unit of first destination */
u_int ce_sdsttype; /* element type of second destination */
u_int ce_sdstunit; /* logical unit of second destination */
u_int ce_flags;        /* misc. flags */
```

In *ce_flags*, CM_INVERT1 and/or CM_INVERT2 may be set to flip the first or second medium during the exchange operation, respectively.

This operation is untested.

CHIOPOSITION (*struct changer_position*) Position the current picker in front of the specified element. The element is specified with a changer_position structure, which includes at least the following elements:

```
u_int cp_type;  /* element type */
u_int cp_unit;  /* logical unit of element */
u_int cp_flags; /* misc. flags */
```

The *cp_flags* field may be set to CP_INVERT to invert the picker during the operation.

CHIOGPICKER (*int*) Return the logical address of the current picker.

CHIOSPICKER (*int*) Select the picker specified by the given logical address.

CHIOGPARAMS (*struct changer_params*) Return the configuration parameters for the media changer. This ioctl fills the changer_params structure passed by the user with at least the following fields:

```
u_int cp_npickers; /* number of pickers */
u_int cp_nslots;   /* number of slots */
u_int cp_nportals; /* number of import/export portals */
u_int cp_ndrives;  /* number of drives */
```

This call can be used by applications to query the dimensions of the jukebox before using the CHIGSTATUS ioctl to query the jukebox status.

CHIOIELEM Perform the **INITIALIZE ELEMENT STATUS** call on the media changer device. This forces the media changer to update its internal status information with respect to loaded media. It also scans any barcode labels provided that it has a label reader. The **ch** driver's status is not affected by this call.

CHIOGSTATUS (*struct changer_element_status_request*) Perform the **READ ELEMENT STATUS** call on the media changer device. This call reads the element status information of the media changer and converts it to an array of *changer_element_status* structures.

With each call to CHIOGSTATUS, the status of one or more elements of one type may be queried.

The application passes a *changer_element_status_request* structure to the **ch** driver which contains the following fields:

```
u_int                  cesr_element_type;
u_int                  cesr_element_base;
u_int                  cesr_element_count;
u_int                  cesr_flags;
struct changer_element_status *cesr_element_status;
```

This structure is read by the driver to determine the type, logical base address and number of elements for which information is to be returned in the array of *changer_element_status* structures pointed to by the *cesr_element_status* field. The application must allocate enough memory for *cesr_element_count* status structures (see below). The *cesr_flags* can optionally be set to CESR_VOLTAGS to indicate that volume tag (bar code) information is to be read from the jukebox and returned.

The *cesr_element_base* and *cesr_element_count* fields must be valid with respect to the physical configuration of the changer. If they are not, the CHIOGSTATUS ioctl returns the EINVAL error code.

The information about the elements is returned in an array of *changer_element_status* structures. This structure include at least the following fields:

```
u_int           ces_addr;      /* element address in media changer */
u_char          ces_flags;     /* see CESTATUS definitions below */
u_char          ces_sensecode; /* additional sense code for element */
u_char          ces_sensequal; /* additional sense code qualifier */
u_char          ces_invert;    /* invert bit */
u_char          ces_svalid;    /* source address (ces_source) valid */
u_short         ces_source;    /* source address of medium */
changer_voltag_t ces_pvoltag;   /* primary volume tag */
changer_voltag_t ces_avoltag;   /* alternate volume tag */
u_char          ces_idvalid;   /* ces_scsi_id is valid */
u_char          ces_scsi_id;   /* SCSI id of element (if ces_idvalid is nonzero) */
u_char          ces_lunvalid;  /* ces_scsi_lun is valid */
u_char          ces_scsi_lun;  /* SCSI lun of element (if ces_lunvalid is nonzero) */
```

The *ces_addr* field contains the address of the element in the coordinate system of the media changer. It is not used by the driver, and should be used for diagnostic purposes only.

The following flags are defined for the *ces_flags* field:

CESTATUS_FULL	A medium is present.
CESTATUS_IMPEXP	The medium has been deposited by the operator (and not by a picker).
CESTATUS_EXCEPT	The element is in an exceptional state (e.g. invalid barcode label, barcode not yet scanned).

CESTATUS_ACCESS
: The element is accessible by the picker.

CESTATUS_EXENAB
: The element supports medium export.

CESTATUS_INENAB
: The element supports medium import.

Note that not all flags are valid for all element types.

NOTES

This version of the **ch** driver has been tested with a DEC TZ875 (5 slot, one DLT drive) and a Breece Hill Q47 (60 slot, four DLT drives, barcode reader).

Many of the features the **ch** driver supports are not thoroughly tested due to the fact that the devices available for testing do not support the necessary commands. This is true for alternate volume tags, media flipping, import/export element handling, multiple picker operation and other things.

FILES

/dev/ch[0-9] device entries

DIAGNOSTICS

If the media changer does not support features requested by the **ch** driver, it will produce both console error messages and failure return codes to the ioctls described here.

SEE ALSO

chio(1), cam(4), cd(4), da(4), sa(4)

HISTORY

The **ch** driver appeared in 386BSD-0.1.

AUTHORS

The **ch** driver was written by Jason R. Thorpe <*thorpej@and.com*> for And Communications, *http://www.and.com/*. It was added to the system by Stefan Grefen <*grefen@goofy.zdv.uni-mainz.de*> who apparently had such a device. It was ported to CAM by Kenneth Merry <*ken@FreeBSD.org*>. It was updated to support volume tags by Hans Huebner <*hans@artcom.de*>.