Summary of CDROM ioctl calls

- Edward A. Falk <efalk@google.com>

November, 2004

This document attempts to describe the ioctl(2) calls supported by the CDROM layer. These are by-and-large implemented (as of Linux 2.6) in drivers/cdrom/cdrom.c and drivers/block/scsi_ioctl.c

ioctl values are listed in <linux/cdrom.h>. As of this writing, they are as follows:

CDROMPAUSE	Pause Audio Operation
CDROMRESUME	Resume paused Audio Operation
CDROMPLAYMSF	Play Audio MSF (struct cdrom_msf)
CDROMPLAYTRKIND	Play Audio Track/index (struct cdrom_ti)
CDROMREADTOCHDR	Read TOC header (struct cdrom_tochdr)
CDROMREADTOCENTRY	Read TOC entry (struct cdrom_tocentry)
CDROMSTOP	Stop the cdrom drive
CDROMSTART	Start the cdrom drive
CDROMEJECT	Ejects the cdrom media
CDROMVOLCTRL	Control output volume (struct cdrom_volctrl)
CDROMSUBCHNL	Read subchannel data (struct cdrom_subchnl)
CDROMREADMODE2	Read CDROM mode 2 data (2336 Bytes) (struct cdrom_read)
CDROMREADMODE1	Read CDROM mode 1 data (2048 Bytes) (struct cdrom_read)
CDROMREADAUDIO	(struct cdrom_read_audio)
CDROMEJECT_SW	enable(1)/disable(0) auto-ejecting
CDROMMULTISESSION	Obtain the start-of-last-session address of multi session disks (struct cdrom_multisession)
CDROM_GET_MCN	Obtain the "Universal Product Code" if available (struct cdrom_mcn)
CDROM_GET_UPC	Deprecated, use CDROM_GET_MCN instead.
CDROMRESET	hard-reset the drive
CDROMVOLREAD	Get the drive's volume setting (struct cdrom_volctrl)
CDROMREADRAW	read data in raw mode (2352 Bytes) (struct cdrom_read)
CDROMREADCOOKED	read data in cooked mode
CDROMSEEK	seek msf address
CDROMPLAYBLK	scsi-cd only, (struct cdrom_blk)
CDROMREADALL	read all 2646 bytes
CDROMGETSPINDOWN	return 4-bit spindown value
CDROMSETSPINDOWN	set 4-bit spindown value
CDROMCLOSETRAY	pendant of CDROMEJECT
CDROM_SET_OPTIONS	Set behavior options
CDROM_CLEAR_OPTIONS	Clear behavior options
CDROM_SELECT_SPEED	Set the CD-ROM speed
CDROM_SELECT_DISC	Select disc (for juke-boxes)
CDROM_MEDIA_CHANGED	Check is media changed
CDROM_TIMED_MEDIA_CHANGE	Check if media changed since given time (struct cdrom_timed_media_change_info)
CDROM_DRIVE_STATUS	Get tray position, etc.
CDROM_DISC_STATUS	Get disc type, etc.
CDROM_CHANGER_NSLOTS	Get number of slots
CDROM_LOCKDOOR	lock or unlock door
CDROM_DEBUG	Turn debug messages on/off
CDROM_GET_CAPABILITY	get capabilities
CDROMAUDIOBUFSIZ	set the audio buffer size
DVD_READ_STRUCT	Read structure
DVD_WRITE_STRUCT	Write structure
DVD_AUTH	Authentication
CDROM_SEND_PACKET	send a packet to the drive
CDROM_NEXT_WRITABLE	get next writable block
CDROM_LAST_WRITTEN	get last block written on disc

The information that follows was determined from reading kernel source code. It is likely that some corrections will be made over

time.

General:

Unless otherwise specified, all ioctl calls return 0 on success and -1 with errno set to an appropriate value on error. (Some ioctls return non-negative data values.)

Unless otherwise specified, all ioctl calls return -1 and set errno to EFAULT on a failed attempt to copy data to or from user address space.

Individual drivers may return error codes not listed here.

Unless otherwise specified, all data structures and constants are defined in <linux/cdrom.h>

CDROMPAUSE

Pause Audio Operation

usage:

```
ioctl(fd, CDROMPAUSE, 0);
```

inputs:
none
outputs:
none
error return:
- ENOSYS cd drive not audio-capable.

CDROMRESUME

Resume paused Audio Operation

usage:

```
ioctl(fd, CDROMRESUME, 0);
```

inputs:
none
outputs:
none
error return:
- ENOSYS cd drive not audio-capable.

CDROMPLAYMSF

Play Audio MSF

(struct cdrom_msf)

usage:

```
struct cdrom_msf msf;

ioctl(fd, CDROMPLAYMSF, &msf);
```

inputs:
cdrom_msf structure, describing a segment of music to play
outputs:
none
error return:
- ENOSYS cd drive not audio-capable.

notes:
- MSF stands for minutes-seconds-frames
- LBA stands for logical block address
- Segment is described as start and end times, where each time is described as minutes:seconds:frames. A frame is 1/75 of a second.

CDROMPLAYTRKIND

Play Audio Track/index

(struct cdrom_ti)

usage:

```
struct cdrom_ti ti;
```

```
ioctl(fd, CDROMPLAYTRKIND, &ti);
```

inputs:
cdrom_ti structure, describing a segment of music to play
outputs:
none
error return:
- ENOSYS cd drive not audio-capable.

notes:
- Segment is described as start and end times, where each time is described as a track and an index.

CDROMREADTOCHDR

Read TOC header

(struct cdrom_tochdr)

usage:

```
cdrom_tochdr header;

ioctl(fd, CDROMREADTOCHDR, &header);
```

inputs:
cdrom_tochdr structure
outputs:
cdrom_tochdr structure
error return:
- ENOSYS cd drive not audio-capable.

CDROMREADTOCENTRY

Read TOC entry

(struct cdrom_tocentry)

usage:

```
struct cdrom_tocentry entry;

ioctl(fd, CDROMREADTOCENTRY, &entry);
```

inputs:
cdrom_tocentry structure
outputs:
cdrom_tocentry structure
error return:
- ENOSYS cd drive not audio-capable.
- EINVAL entry.cdte_format not CDROM_MSF or CDROM_LBA
- EINVAL requested track out of bounds
- EIO I/O error reading TOC

notes:
- TOC stands for Table Of Contents
- MSF stands for minutes-seconds-frames
- LBA stands for logical block address

CDROMSTOP

Stop the cdrom drive

usage:

```
ioctl(fd, CDROMSTOP, 0);
```

inputs:
none
outputs:
none
error return:
- ENOSYS cd drive not audio-capable.

notes:
- Exact interpretation of this ioctl depends on the device, but most seem to spin the drive down.

CDROMSTART

Start the cdrom drive

usage:

```
ioctl(fd, CDROMSTART, 0);
```

inputs:
none
outputs:
none
error return:
- ENOSYS cd drive not audio-capable.

notes:
- Exact interpretation of this ioctl depends on the device, but most seem to spin the drive up and/or close the tray. Other devices ignore the ioctl completely.

CDROMEJECT

- Ejects the cdrom media

usage:

```
ioctl(fd, CDROMEJECT, 0);
```

inputs:
none
outputs:
none
error returns:
- ENOSYS cd drive not capable of ejecting
- EBUSY other processes are accessing drive, or door is locked

notes:
- See CDROM_LOCKDOOR, below.

CDROMCLOSETRAY

pendant of CDROMEJECT

usage:

```
ioctl(fd, CDROMCLOSETRAY, 0);
```

inputs:
none
outputs:
none
error returns:
- ENOSYS cd drive not capable of closing the tray
- EBUSY other processes are accessing drive, or door is locked

notes:
- See CDROM_LOCKDOOR, below.

CDROMVOLCTRL

Control output volume (struct cdrom_volctrl)

usage:

```
struct cdrom_volctrl volume;

ioctl(fd, CDROMVOLCTRL, &volume);
```

inputs:
cdrom_volctrl structure containing volumes for up to 4 channels.
outputs:
none
error return:
- ENOSYS cd drive not audio-capable.

CDROMVOLREAD

Get the drive's volume setting

(struct cdrom_volctrl)

usage:

```
struct cdrom_volctrl volume;

ioctl(fd, CDROMVOLREAD, &volume);
```

inputs:

none

outputs:

The current volume settings.

error return:

- ENOSYS cd drive not audio-capable.

CDROMSUBCHNL

Read subchannel data

(struct cdrom_subchnl)

usage:

```
struct cdrom_subchnl q;

ioctl(fd, CDROMSUBCHNL, &q);
```

inputs:

cdrom_subchnl structure

outputs:

cdrom_subchnl structure

error return:

- ENOSYS cd drive not audio-capable.
- EINVAL format not CDROM_MSF or CDROM_LBA

notes:

- Format is converted to CDROM_MSF or CDROM_LBA as per user request on return

CDROMREADRAW

read data in raw mode (2352 Bytes)

(struct cdrom_read)

usage:

```
union {

  struct cdrom_msf msf;               /* input */
  char buffer[CD_FRAMESIZE_RAW];      /* return */
} arg;
ioctl(fd, CDROMREADRAW, &arg);
```

inputs:

cdrom_msf structure indicating an address to read.

Only the start values are significant.

outputs:

Data written to address provided by user.

error return:

- EINVAL address less than 0, or msf less than 0:2:0
- ENOMEM out of memory

notes:

- As of 2.6.8.1, comments in <linux/cdrom.h> indicate that this ioctl accepts a cdrom_read structure, but actual source code reads a cdrom_msf structure and writes a buffer of data to the same address.
- MSF values are converted to LBA values via this formula:

```
lba = (((m * CD_SECS) + s) * CD_FRAMES + f) - CD_MSF_OFFSET;
```

CDROMREADMODE1

Read CDROM mode 1 data (2048 Bytes)

(struct cdrom_read)

notes:

Identical to CDROMREADRAW except that block size is CD_FRAMESIZE (2048) bytes

CDROMREADMODE2

Read CDROM mode 2 data (2336 Bytes)

(struct cdrom_read)

notes:

Identical to CDROMREADRAW except that block size is CD_FRAMESIZE_RAW0 (2336) bytes

CDROMREADAUDIO

(struct cdrom_read_audio)

usage:

```
struct cdrom_read_audio ra;

ioctl(fd, CDROMREADAUDIO, &ra);
```

inputs:

cdrom_read_audio structure containing read start point and length

outputs:

audio data, returned to buffer indicated by ra

error return:

- EINVAL format not CDROM_MSF or CDROM_LBA
- EINVAL nframes not in range [1 75]
- ENXIO drive has no queue (probably means invalid fd)
- ENOMEM out of memory

CDROMEJECT_SW

enable(1)/disable(0) auto-ejecting

usage:

```
int val;

ioctl(fd, CDROMEJECT_SW, val);
```

inputs:

Flag specifying auto-eject flag.

outputs:

none

error return:

- ENOSYS Drive is not capable of ejecting.
- EBUSY Door is locked

CDROMMULTISESSION

Obtain the start-of-last-session address of multi session disks

(struct cdrom_multisession)

usage:

```
struct cdrom_multisession ms_info;

ioctl(fd, CDROMMULTISESSION, &ms_info);
```

inputs:

cdrom_multisession structure containing desired

format.

outputs:

cdrom_multisession structure is filled with last_session information.

error return:

- EINVAL format not CDROM_MSF or CDROM_LBA

CDROM_GET_MCN

Obtain the "Universal Product Code" if available

(struct cdrom_mcn)

usage:

```
struct cdrom_mcn mcn;

ioctl(fd, CDROM_GET_MCN, &mcn);
```

inputs:

none

outputs:

Universal Product Code

error return:

- ENOSYS Drive is not capable of reading MCN data.

notes:

- Source code comments state:

```
The following function is implemented, although very few
audio discs give Universal Product Code information, which
should just be the Medium Catalog Number on the box.  Note,
that the way the code is written on the CD is /not/ uniform
across all discs!
```

CDROM_GET_UPC

CDROM_GET_MCN (deprecated)

Not implemented, as of 2.6.8.1

CDROMRESET

hard-reset the drive

usage:

```
ioctl(fd, CDROMRESET, 0);
```

inputs:

none

outputs:

none

error return:

- EACCES Access denied: requires CAP_SYS_ADMIN
- ENOSYS Drive is not capable of resetting.

CDROMREADCOOKED

read data in cooked mode

usage:

```
u8 buffer[CD_FRAMESIZE]

ioctl(fd, CDROMREADCOOKED, buffer);
```

inputs:

none

outputs:

2048 bytes of data, "cooked" mode.

notes:

Not implemented on all drives.

CDROMREADALL

read all 2646 bytes

Same as CDROMREADCOOKED, but reads 2646 bytes.

CDROMSEEK

seek msf address

usage:

```
struct cdrom_msf msf;

ioctl(fd, CDROMSEEK, &msf);
```

inputs:

MSF address to seek to.

outputs:

none

CDROMPLAYBLK

scsi-cd only

(struct cdrom_blk)

usage:

```
struct cdrom_blk blk;

ioctl(fd, CDROMPLAYBLK, &blk);
```

inputs:

Region to play

outputs:

none

CDROMGETSPINDOWN

usage:

```
char spindown;

ioctl(fd, CDROMGETSPINDOWN, &spindown);
```

inputs:

none

outputs:

The value of the current 4-bit spindown value.

CDROMSETSPINDOWN

usage:

```
char spindown

ioctl(fd, CDROMSETSPINDOWN, &spindown);
```

inputs:

4-bit value used to control spindown (TODO: more detail here)

outputs:

none

CDROM_SET_OPTIONS

Set behavior options

usage:

```
int options;

ioctl(fd, CDROM_SET_OPTIONS, options);
```

inputs:

New values for drive options. The logical 'or' of:

CDO_AUTO_CLOSE	close tray on first open(2)
CDO_AUTO_EJECT	open tray on last release
CDO_USE_FFLAGS	use O_NONBLOCK information on open
CDO_LOCK	lock tray on open files
CDO_CHECK_TYPE	check type on open for data

outputs:

Returns the resulting options settings in the ioctl return value. Returns -1 on error.

error return:

- ENOSYS selected option(s) not supported by drive.

CDROM_CLEAR_OPTIONS

Clear behavior options

Same as CDROM_SET_OPTIONS, except that selected options are turned off.

CDROM_SELECT_SPEED

Set the CD-ROM speed

usage:

```
int speed;

ioctl(fd, CDROM_SELECT_SPEED, speed);
```

inputs:

New drive speed.

outputs:

none

error return:

- ENOSYS speed selection not supported by drive.

CDROM_SELECT_DISC

Select disc (for juke-boxes)

usage:

```
int disk;

ioctl(fd, CDROM_SELECT_DISC, disk);
```

inputs:

Disk to load into drive.

outputs:

none

error return:

- EINVAL Disk number beyond capacity of drive

CDROM_MEDIA_CHANGED

Check is media changed

usage:

```
int slot;

ioctl(fd, CDROM_MEDIA_CHANGED, slot);
```

inputs:

Slot number to be tested, always zero except for jukeboxes.

May also be special values CDSL_NONE or CDSL_CURRENT

outputs:

Ioctl return value is 0 or 1 depending on whether the media

has been changed, or -1 on error.

error returns:

- ENOSYS Drive can't detect media change
- EINVAL Slot number beyond capacity of drive
- ENOMEM Out of memory

CDROM_DRIVE_STATUS

Get tray position, etc.

usage:

```
int slot;

ioctl(fd, CDROM_DRIVE_STATUS, slot);
```

inputs:

Slot number to be tested, always zero except for jukeboxes.

May also be special values CDSL_NONE or CDSL_CURRENT

outputs:

Ioctl return value will be one of the following values

from <linux/cdrom.h>:

CDS_NO_INFO	Information not available.
CDS_NO_DISC	
CDS_TRAY_OPEN	
CDS_DRIVE_NOT_READY	
CDS_DISC_OK	
-1	error

error returns:

- ENOSYS Drive can't detect drive status
- EINVAL Slot number beyond capacity of drive
- ENOMEM Out of memory

CDROM_DISC_STATUS

Get disc type, etc.

usage:

```
ioctl(fd, CDROM_DISC_STATUS, 0);
```

inputs:

none

outputs:

Ioctl return value will be one of the following values

from <linux/cdrom.h>:

- CDS_NO_INFO
- CDS_AUDIO
- CDS_MIXED
- CDS_XA_2_2
- CDS_XA_2_1
- CDS_DATA_1

error returns:

none at present

notes:

- Source code comments state:

```
Ok, this is where problems start.  The current interface for
the CDROM_DISC_STATUS ioctl is flawed.  It makes the false
assumption that CDs are all CDS_DATA_1 or all CDS_AUDIO, etc.
Unfortunately, while this is often the case, it is also
very common for CDs to have some tracks with data, and some
tracks with audio.      Just because I feel like it, I declare
the following to be the best way to cope.  If the CD has
ANY data tracks on it, it will be returned as a data CD.
If it has any XA tracks, I will return it as that.      Now I
could simplify this interface by combining these returns with
the above, but this more clearly demonstrates the problem
with the current interface.  Too bad this wasn't designed
to use bitmasks...             -Erik

Well, now we have the option CDS_MIXED: a mixed-type CD.
User level programmers might feel the ioctl is not very
useful.
                ---david
```

CDROM_CHANGER_NSLOTS

Get number of slots

usage:

```
ioctl(fd, CDROM_CHANGER_NSLOTS, 0);
```

inputs:

none

outputs:

The ioctl return value will be the number of slots in a CD changer. Typically 1 for non-multi-disk devices.

error returns:

none

CDROM_LOCKDOOR

lock or unlock door

usage:

```
int lock;
```

```
ioctl(fd, CDROM_LOCKDOOR, lock);
```

inputs:

Door lock flag, 1=lock, 0=unlock

outputs:

none

error returns:

- EDRIVE_CANT_DO_THIS

  Door lock function not supported.

- EBUSY

  Attempt to unlock when multiple users have the drive open and not CAP_SYS_ADMIN

notes:

As of 2.6.8.1, the lock flag is a global lock, meaning that all CD drives will be locked or unlocked together. This is probably a bug.

The EDRIVE_CANT_DO_THIS value is defined in <linux/cdrom.h> and is currently (2.6.8.1) the same as EOPNOTSUPP

CDROM_DEBUG

Turn debug messages on/off

usage:

```
int debug;

ioctl(fd, CDROM_DEBUG, debug);
```

inputs:

Cdrom debug flag, 0=disable, 1=enable

outputs:

The ioctl return value will be the new debug flag.

error return:

- EACCES Access denied: requires CAP_SYS_ADMIN

CDROM_GET_CAPABILITY

get capabilities

usage:

```
ioctl(fd, CDROM_GET_CAPABILITY, 0);
```

inputs:

none

outputs:

The ioctl return value is the current device capability flags. See CDC_CLOSE_TRAY, CDC_OPEN_TRAY, etc.

CDROMAUDIOBUFSIZ

set the audio buffer size

usage:

```
int arg;

ioctl(fd, CDROMAUDIOBUFSIZ, val);
```

inputs:

New audio buffer size

outputs:

The ioctl return value is the new audio buffer size, or -1 on error.

error return:

- ENOSYS Not supported by this driver.

notes:

Not supported by all drivers.

DVD_READ_STRUCT Read structure

usage:

```
dvd_struct s;

ioctl(fd, DVD_READ_STRUCT, &s);
```

inputs:

dvd_struct structure, containing:

type	specifies the information desired, one of DVD_STRUCT_PHYSICAL, DVD_STRUCT_COPYRIGHT, DVD_STRUCT_DISCKEY, DVD_STRUCT_BCA, DVD_STRUCT_MANUFACT
physical.layer_num	desired layer, indexed from 0
copyright.layer_num	desired layer, indexed from 0
disckey.agid	

outputs:

dvd_struct structure, containing:

physical	for type == DVD_STRUCT_PHYSICAL
copyright	for type == DVD_STRUCT_COPYRIGHT
disckey.value	for type == DVD_STRUCT_DISCKEY
bca.{len,value}	for type == DVD_STRUCT_BCA
manufact.{len,valu}	for type == DVD_STRUCT_MANUFACT

error returns:

- EINVAL physical.layer_num exceeds number of layers
- EIO Received invalid response from drive

DVD_WRITE_STRUCT Write structure

Not implemented, as of 2.6.8.1

DVD_AUTH Authentication

usage:

```
dvd_authinfo ai;

ioctl(fd, DVD_AUTH, &ai);
```

inputs:
dvd_authinfo structure. See <linux/cdrom.h>
outputs:
dvd_authinfo structure.
error return:

- ENOTTY ai.type not recognized.

CDROM_SEND_PACKET

send a packet to the drive

usage:

```
struct cdrom_generic_command cgc;

ioctl(fd, CDROM_SEND_PACKET, &cgc);
```

inputs:

cdrom_generic_command structure containing the packet to send.

outputs:

none

cdrom_generic_command structure containing results.

error return:

- EIO

command failed.

- EPERM

  Operation not permitted, either because a write command was attempted on a drive which is opened read-only, or because the command requires CAP_SYS_RAWIO

- EINVAL

  cgc.data_direction not set

CDROM_NEXT_WRITABLE

get next writable block

usage:

```
long next;

ioctl(fd, CDROM_NEXT_WRITABLE, &next);
```

inputs:

none

outputs:

The next writable block.

notes:

If the device does not support this ioctl directly, the

ioctl will return CDROM_LAST_WRITTEN + 7.

CDROM_LAST_WRITTEN

get last block written on disc

usage:

```
long last;

ioctl(fd, CDROM_LAST_WRITTEN, &last);
```

inputs:

none

outputs:

The last block written on disc

notes:

If the device does not support this ioctl directly, the result is derived from the disc's table of contents. If the table of contents can't be read, this ioctl returns an error.