NAME

readcd − read or write data Compact Discs

SYNOPSIS

readcd dev=*device* [*options*]

DESCRIPTION

Readcd is used to read or write Compact Discs.

The *device* refers to *scsibus*/*target*/*lun* of the drive. Communication on *SunOS* is done with the SCSI general driver **scg.** Other operating systems are using a library simulation of this driver. Possible syntax is: **dev=** *scsibus*,*target*,*lun* or **dev=** *target*,*lun*. In the latter case, the drive has to be connected to the default SCSI bus of the machine. *Scsibus*, *target* and *lun* are integer numbers. Some operating systems or SCSI transport implementations may require to specify a filename in addition. In this case the correct syntax for the device is: **dev=** *devicename*:*scsibus*,*target*,*lun* or **dev=** *devicename*:*target*,*lun*. If the name of the device node that has been specified on such a system refers to exactly one SCSI device, a shorthand in the form **dev=** *devicename*:@ or **dev=** *devicename*:@,*lun* may be used instead of **dev=** *devicename*:*scsibus*,*target*,*lun*.

To access remote SCSI devices, you need to prepend the SCSI device name by a remote device indicator. The remote device indicator is either **REMOTE:***user@host:* or **REMOTE:***host:*
A valid remote SCSI device name may be: **REMOTE:***user@host:* to allow remote SCSI bus scanning or **REMOTE:***user@host:1,0,0* to access the SCSI device at *host* connected to SCSI bus # 1,target 0 lun 0.

To access SCSI devices via alternate transport layers, you need to prepend the SCSI device name by a transport layer indicator. The transport layer indicator may be something like **USCSI:** or **ATAPI:**. To get a list of supported transport layers for your platform, use **dev=** *HELP*:

To make **readcd** portable to all UNIX platforms, the syntax **dev=** *devicename*:*scsibus*,*target*,*lun* is preferred as is hides OS specific knowledge about device names from the user. A specific OS must not necessarily support a way to specify a real device file name nor a way to specify *scsibus*,*target*,*lun*.

Scsibus 0 is the default SCSI bus on the machine. Watch the boot messages for more information or look into **/var/adm/messages** for more information about the SCSI configuration of your machine. If you have problems to figure out what values for *scsibus*,*target*,*lun* should be used, try the **−scanbus** option of **cdrecord**.

OPTIONS

If no options except the *dev*= option have been specified, **readcd** goes into interactive mode. Select a primary function and then follow the instructions.

−version
Print version information and exit.

dev=*target*
Sets the SCSI target for the drive, see notes above. A typical device specification is **dev=***6,0* . If a filename must be provided together with the numerical target specification, the filename is implementation specific. The correct filename in this case can be found in the system specific manuals of the target operating system. On a *FreeBSD* system without *CAM* support, you need to use the control device (e.g. */dev/rcd0.ctl*). A correct device specification in this case may be **dev=***/dev/rcd0.ctl:@* .

On Linux, drives connected to a parallel port adapter are mapped to a virtual SCSI bus. Different adapters are mapped to different targets on this virtual SCSI bus.

If no *dev* option is present, **cdrecord** will try to get the device from the **CDR_DEVICE** environment.

If the argument to the **dev=** option does not contain the characters ',', '/', '@' or ':', it is interpreted as an label name that may be found in the file /etc/default/cdrecord (see FILES section).

timeout=#
Set the default SCSI command timeout value to # seconds. The default SCSI command timeout is the minimum timeout used for sending SCSI commands. If a SCSI command fails due to a timeout, you may try to raise the default SCSI command timeout above the timeout value of the failed command. If the command runs correctly with a raised command timeout, please report the better timeout value and the corresponding command to the author of the program. If no *timeout* option is present, a default timeout of 40 seconds is used.

debug=#, **-d**
Set the misc debug value to # (with debug=#) or increment the misc debug level by one (with -d). If you specify *-dd,* this equals to **debug=***2*. This may help to find problems while opening a driver for libscg. as well as with sector sizes and sector types. Using **−debug** slows down the process and may be the reason for a buffer underrun.

kdebug=#, **kd=**#
Tell the **scg**-driver to modify the kernel debug value while SCSI commands are running.

−silent, **−s**
Do not print out a status report for failed SCSI commands.

−v Increment the level of general verbosity by one. This is used e.g. to display the progress of the process.

−V Increment the verbose level with respect of SCSI command transport by one. This helps to debug problems during the process, that occur in the CD-Recorder. If you get incomprehensible error messages you should use this flag to get more detailed output. **−VV** will show data buffer content in addition. Using **−V** or **−VV** slows down the process.

f=*file* Specify the filename where the output should be written or the inout should be taken from. Using '-' as filename will cause **readcd** to use **stdout** resp. **stdin**.

−w Switch to write mode. If this option is not present, **readcd** reads from the specified device.

−c2scan
Scans the whole CD or the range specified by the **sectors=***range* for C2 errors. C2 errors are errors that are uncorrectable after the second stage of the 24/28 + 28/32 Reed Solomon correction system at audio level (2352 bytes sector size). If an audio CD has C2 errors, interpolation is needed to hide the errors. If a data CD has C2 errors, these errors are in most cases corrected by the ECC/EDC code that makes 2352 bytes out of 2048 data bytes. The ECC/EDC code should be able to correct about 100 C2 error bytes per sector.

If you find C2 errors you may want to reduce the speed using the **speed=** option as C2 errors may be a result of dynamic unbalance on the medium.

sectors=*range*
Specify a sector range that should be read. The range is specified by the starting sector number, a minus sign and the ending sector number. The end sector is not included in the list, so **sectors=**0-0 will not read anything and may be used to check for a CD in the drive.

speed=#
Set the speed factor of the read or write process to #. # is an integer, representing a multiple of the audio speed. This is about 150 KB/s for CD-ROM and about 172 KB/s for CD-Audio. If no *speed* option is present, **readcd** will use maximum speed. Only MMC compliant drives will benefit from this option. The speed of non MMC drives is not changed.

Using a lower speed may increase the readability of a CD or DVD.

−notrunc
Do not truncate the outputfile when opening it.

−fulltoc
Retrieve a full TOC from the current disk and print it in hex.

−clone Do a clone read. Read the CD with all sub-channel data and a full TOC. The full TOC data will be but into a file with similar name as with the **f=** option but the suffix **.toc** added.

−noerror
Do not abort if the high level error checking in **readcd** found an uncorrectable error in the data stream.

−nocorr
Switch the drive into a mode where it ignores read errors in data sectors that are a result of uncorrectable ECC/EDC errors before reading. If **readcd** completes, the error recovery mode of the drive is switched back to the remembered old mode.

retries=#
Set the retry count for high level retries in **readcd** to #. The default is to do 128 retries which may be too much if you like to read a CD with many unreadable sectors.

−overhead
Meter the SCSI command overhead time. This is done by executing several commands 1000 times and printing the total time used. If you divide the displayed times by 1000, you get the average overhead time for a single command.

EXAMPLES

For all examples below, it will be assumed that the drive is connected to the primary SCSI bus of the machine. The SCSI target id is set to 2.

To read the complete media from a CD-ROM writing the data to the file *cdimage.raw*:

readcd dev=2,0 f=cdimage.raw

To read sectors from range 150 ... 10000 from a CD-ROM writing the data to the file *cdimage.raw*:

readcd dev=2,0 sectors=150-10000 f=cdimage.raw

To write the data from the file *cdimage.raw* (e.g. a filesystem image from **mkisofs**) to a DVD-RAM, call:

readcd dev=2,0 -w f=cdimage.raw

ENVIRONMENT

RSH If the **RSH** environment is present, the remote connection will not be created via **rcmd**(3) but by calling the program pointed to by **RSH**. Use e.g. **RSH=**/usr/bin/ssh to create a secure shell connection.

Note that this forces **cdrecord** to create a pipe to the **rsh(1)** program and disallows **cdrecord** to directly access the network socket to the remote server. This makes it impossible to set up performance parameters and slows down the connection compared to a **root** initiated **rcmd(3)** connection.

RSCSI If the **RSCSI** environment is present, the remote SCSI server will not be the program **/opt/schily/sbin/rscsi** but the program pointed to by **RSCSI**. Note that the remote SCSI server program name will be ignored if you log in using an account that has been created with a remote SCSI server program as login shell.

FILES

SEE ALSO

cdrecord(1), **mkisofs**(1), **scg**(7), **fbk**(7), **rcmd**(3), **ssh**(1).

NOTES

If you don't want to allow users to become root on your system, **readcd** may safely be installed suid root. This allows all users or a group of users with no root privileges to use **readcd. Readcd** in this case will only allow access to CD-ROM type drives- To give all user access to use **readcd,** enter:

```
chown root /usr/local/bin/readcd
chmod 4711 /usr/local/bin/readcd
```

To give a restricted group of users access to **readcd** enter:

```
chown root /usr/local/bin/readcd
chgrp cdburners /usr/local/bin/readcd
chmod 4710 /usr/local/bin/readcd
```

and add a group *cdburners* on your system.

Never give write permissions for non root users to the */dev/scg?* devices unless you would allow anybody to read/write/format all your disks.

You should not connect old drives that do not support disconnect/reconnect to either the SCSI bus that is connected to the CD-Recorder or the source disk.

When using **readcd** with the broken **Linux SCSI generic driver.** You should note that **readcd** uses a hack, that tries to emulate the functionality of the scg driver. Unfortunately, the sg driver on **Linux** has several severe bugs:

- It cannot see if a SCSI command could not be sent at all.
- It cannot get the SCSI status byte. **Readcd** for that reason cannot report failing SCSI commands in some situations.
- It cannot get real DMA count of transfer. **Readcd** cannot tell you if there is an DMA residual count.
- It cannot get number of bytes valid in auto sense data. **Readcd** cannot tell you if device transfers no sense data at all.
- It fetches to few data in auto request sense (CCS/SCSI-2/SCSI-3 needs >= 18).

DIAGNOSTICS

A typical error message for a SCSI command looks like:

```
readcd: I/O error. test unit ready: scsi sendcmd: no error
CDB:  00 20 00 00 00 00
status: 0x2 (CHECK CONDITION)
Sense Bytes: 70 00 05 00 00 00 00 0A 00 00 00 00 25 00 00 00 00 00
Sense Key: 0x5 Illegal Request, Segment 0
Sense Code: 0x25 Qual 0x00 (logical unit not supported) Fru 0x0
Sense flags: Blk 0 (not valid)
cmd finished after 0.002s timeout 40s
```

The first line gives information about the transport of the command. The text after the first colon gives the error text for the system call from the view of the kernel. It usually is: **I/O error** unless other problems happen. The next words contain a short description for the SCSI command that fails. The rest of the line tells you if there were any problems for the transport of the command over the SCSI bus. **fatal error** means that it was not possible to transport the command (i.e. no device present at the requested SCSI address).

The second line prints the SCSI command descriptor block for the failed command.

The third line gives information on the SCSI status code returned by the command, if the transport of the command succeeds. This is error information from the SCSI device.

The fourth line is a hex dump of the auto request sense information for the command.

The fifth line is the error text for the sense key if available, followed by the segment number that is only valid if the command was a *copy* command. If the error message is not directly related to the current command, the text *deferred error* is appended.

The sixth line is the error text for the sense code and the sense qualifier if available. If the type of the device is known, the sense data is decoded from tables in *scsierrs.c* . The text is followed by the error value for a field replaceable unit.

The seventh line prints the block number that is related to the failed command and text for several error flags. The block number may not be valid.

The eight line reports the timeout set up for this command and the time that the command really needed to complete.

BUGS
CREDITS
MAILING LISTS

If you want to actively take part on the development of cdrecord, you may join the cdwriting mailing list by sending mail to:

other-cdwrite-request@lists.debian.org

and include the word *subscribe* in the body. The mail address of the list is:

cdwrite@lists.debian.org

AUTHOR

Jörg Schilling
Seestr. 110
D-13353 Berlin
Germany

Additional information can be found on:
http://www.fokus.fhg.de/usr/schilling/cdrecord.html

If you have support questions, send them to:

cdrecord-support@berlios.de
or **other-cdwrite@lists.debian.org**

Of you have definitely found a bug, send a mail to:

cdrecord-developers@berlios.de
or **schilling@fokus.fhg.de**

To subscribe, use:

http://lists.berlios.de/mailman/listinfo/cdrecord-developers
or **http://lists.berlios.de/mailman/listinfo/cdrecord-support**