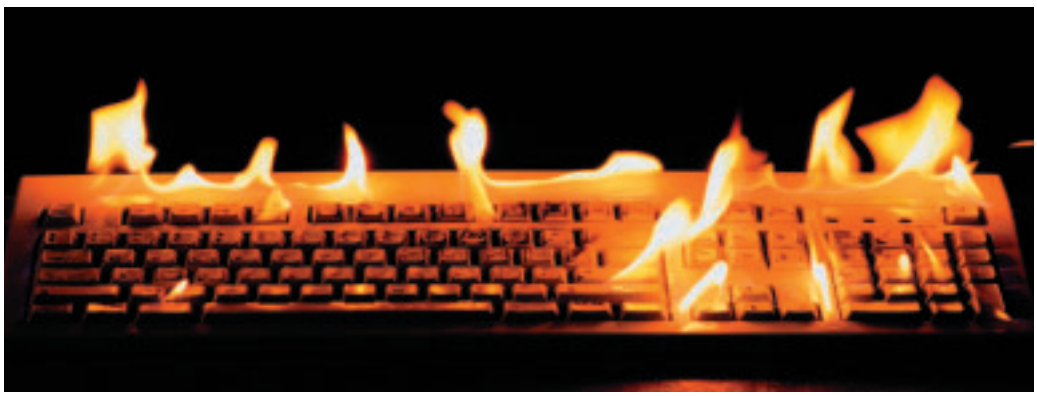
CDs written fast on the command line

BURNING BY COMMAND

HANS-GEORG ESSER



The graphical user interface may be the user friendly way of burning audio and data to disc, but for pure speed the command line simply can't be beaten.

Why wrestle with CD-burning command-line tools when you can simply use one of the graphical applications already available? Well, there are several benefits to writing discs at the command line:

- **Speed** You can burn all the files in an existing directory onto a data CD with one or two brief commands using mkisofs and cdrecord, or a series of WAV files onto an audio CD. Try it out – no GUI tool can do it that fast.
- **Control** Command line tools give you complete control over the task to be performed. You tell the program what it should do in detail via options, meaning that there will be no surprises.
- **Resources** Even an ancient computer, if it has been equipped with a new ATAPI or SCSI CD burner, can be used as a burn computer. There is no need to use an up-to-date graphics card to do so. You can use text mode throughout. It even works without a monitor if you log in from another computer via a network connection.
- **Principle** Dedicated users of the shell will always employ it as a first option for performing tasks.

Even if you don't intend to use the command line for burning, it still makes sense to understand this tool, as the programs described here are working away busily behind the scenes of your GUI. If things go wrong with your graphical application you'll still need to know your way around mkisofs and cdrecord

Data CDs

The simplest way of burning a data CD is to back up a complete sub-directory, providing the content is under 650MB in the case of a standard CD/R. To burn something like the entire */tmp/cddata/* directory onto CD, you could use a variant of the two following commands:

```
mkisofs -r -J -o /tmp/cddata.iso /tmp/cddata/
cdrecord -v dev=0,4,0 speed=4 /tmp/cddata.iso
```

In the two commands above *mkisofs* – make ISO9660 filesystem – creates an ISO file system (where file system type ISO9660 is the default for both Windows and UNIX). *mkisofs -o /tmp/cddata.iso /tmp/cddata/* creates a **CD image** called */tmp/cddata.iso* according to strict ISO standard. This contains all the files from the

CD Image *A CD image is an identical copy of the raw content of a data CD. Such an image can be created under Linux from a data CD using the command dd if=/dev/cdrom of=/tmp/cd.iso. Here, dd simply reads out the CD, byte for byte, and writes the content in the output file cd.iso. This CD image can then be re-written, using cdrecord, onto a blank CD or even be mounted in the file system with mount, like a proper CD.*

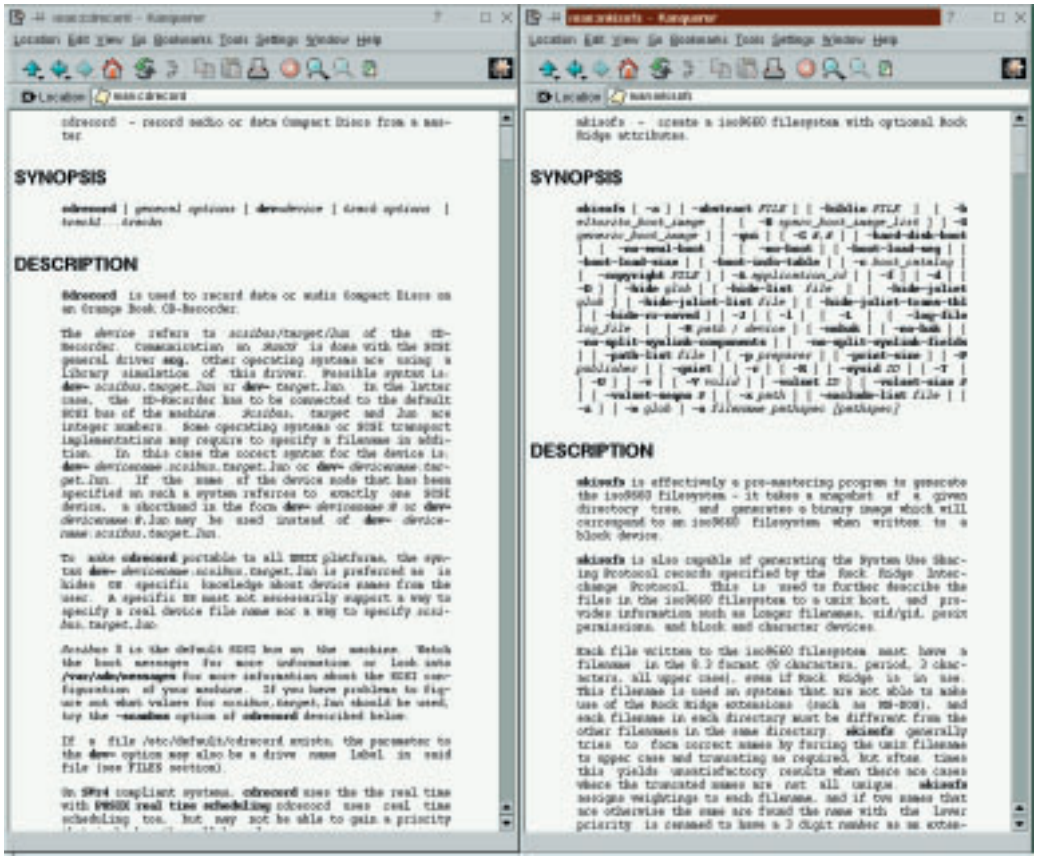


Figure 1: The man pages for cdrecord and mkisofs are exhaustive and very helpful – a look at the more esoteric options can be very entertaining.

/tmp/cddata/ sub-directory. However, when it comes to naming and other file system features (such as UNIX access rights and symbolic links) the ISO standard is limited. An ordinary file system, easily legible under Linux/UNIX and Windows is only obtained by extending the ISO-conform content directory using Rockridge extensions (UNIX) and Joliet extensions (Windows). This is precisely what the *-r* and *-J* options do.

In the second step, the CD image is written onto a blank CD using the *cdrecord* command. The two parameters without "-" define which device (the CD burner) should be used and what burn rate is required. In practice this means:

- *dev=0,4,0*: The burner links up with the first SCSI controller (counting from 0), has the SCSI ID 4 and the LUN (Logical Unit Number) 0. The LUN is always 0 for burners. If you do not know which SCSI ID your burner has, use the *cdrecord -scanbus* command. We'll be explaining more about using ATAPI burners later.
- *speed=4*: Burn at 4x speed.

The additional parameter -v stands for verbose (wordy) and issues a status message at regular intervals during the burn procedure on what percentage of the CD has so far been completed. The file name of the CD image should come at the end of the command.

On the fly

It is also possible to burn a directory directly onto the CD, without taking up 650MB of disk space first. Output from *mkisofs* is simply transferred via a pipe to *cdrecord*; *mkisofs* is called up without specifying an output file (*-o file*) and simply writes the ISO image in the standard output:

```
mkisofs -r /tmp/cddata | cdrecord -v fs=6m
 speed=4 dev=0,4,0 -
```

The minus sign at the end of the command is necessary so that cdrecord reads the data from the standard input and does not attempt to read from a file. A buffer of 6MB (option fs=6m) should prevent buffer underrun. But such buffer underruns (which result in the destruction of the blank) are highly unlikely under Linux anyway, as the operating system has a good multitasking function. Even with the most vigorous activities being performed by other programs, we have never known cdrecord to miss a stroke. In operating systems with less powerful multitasking, burnt out blanks are commonplace – even the screensaver kicking in can cause problems. It is only on very low-powered or very heavily overloaded computers that this type of failure would be conceivable under Linux, and the industry has created a remedy for this scenario: devices with BURN-Proof technology reliably prevent buffer underruns. cdrecord supports this technology.

Audio CDs

Since audio CDs do not contain a file system, mkisofs is not needed for them. The tracks are simply written one after another onto the CD. As

with data CDs, cdrecord is used for the burn procedure, with a few modifications. The sample command from the start of the article now becomes

```
cdrecord -v -audio -pad dev=0,4,0 speed=4 /tmp/track*.wav
```

The *dev* and *speed* options have not been altered, but two new ones have been added:

- -audio tells the burn software that an audio CD is to be burned.
- -pad is less obvious. Audio CDs include tracks in accordance with the CD-DA (Compact-Disc Digital-Audio) standard, and tracks with this specification need a few special characteristics. They need a sampling rate of 44,100 samples per second and their file size must be a multiple of 2352 bytes. Since the wav or au files available for burning do not, as a rule, comply with this file size requirement, the -pad option is needed to add an appropriate number of zeros to the end of the file.

It is also possible to create audio CDs incrementally. The additional option -nofix is used to do so. The following three commands each burn two tracks onto the CD and only fix the blank on the third write procedure, thus finishing it off completely:

```
cdrecord -v -audio -pad -nofix dev=0,4,0 speed=4 t1.wav t2.wav
cdrecord -v -audio -pad -nofix dev=0,4,0 speed=4 t3.wav t4.wav
cdrecord -v -audio -pad dev=0,4,0 speed=4 t5.wav t6.wav
```

A CD which has not been fixed can only be played back in CD burners and not in simple CD-ROM drives or audio CD players. You can finish these discs off at any time using the *-fix* option:

```
cdrecord -v -fix dev=0,4,0
```

Sometimes the resultant CD is missing some audio tracks – with noise taking their place. This is due to the structure of your *wav* files – the byte sequence of the audio coding is wrong; The sequence can be **Little Endian** or **Big Endian**. To get rid of this very strange phenomenon (and to correct the byte sequence), use the *-swab* (SWAp Bytes) option.

Reading back audio tracks

The read-out of an audio CD is a slightly different subject to that of burning but since when copying such CDs, the read-out comes before the burn, we will also briefly introduce the appropriate command line tool here. Unlike data CDs, audio CDs do not simply contain a normal file system, which can be read out using *dd*. Instead, a special CDDA grabber has to be used. The standard tool for this task is *cdda2wav*. MP3 files can also be burned directly onto CD (with at least ten albums usually fitting onto one blank). However, these are then normal data CDs, which cannot be played back using a CD player. To be played on a CD player, they have to be converted into normal *wav* files.

ATAPI burner

We've only covered SCSI burners so far. But ATAPI owners are not excluded from the joy of burning. Linux has an SCSI emulation for ATAPI devices which allows an ATAPI burner to be recognised as a SCSI device and controlled with the usual control commands under SCSI.

This SCSI-ATAPI emulation is *not* usually mounted in the kernel, since it is seldom needed. To load the corresponding kernel module, enter, as *root* administrator, the command

```
modprobe -v ide-scsi
```

The module should now be loaded, and using the command

```
cat /proc/scsi/scsi
```

you will receive a summary of the newly added SCSI devices in your computer. If a proper SCSI controller is present, the SCSI-ATAPI emulation is added to the system as a second SCSI bus (No. 1). The dev-option necessary for the *cdrecord* command is *dev=1,X,0*. ■

Little / Big Endian: *Little and Big Endian are terms which tell us something about the processor architecture. Values which cannot be stored completely in one memory cell are to be distributed over several such cells: So something like the value 43981 (hexadecimal ABCD) is split into AB and CD. With Big Endian, AB is now stored in the first memory cell and CD is stored in the second; in the case of Little Endian it is exactly the opposite.* ■

Problem-solving

- *Access rights: To scan the bus and/or to perform the actual burn procedure, the device files /dev/sg* must be provided for you with read-privileges (the file belonging to the burner should also have write-privileges). Alternatively, you can perform the bus scan as root administrator with chmod a+r /dev/sg* and then just set the necessary device file to "a+rw" to release the read privileges for all users.*
- *Generic Devices: cdrecord uses direct access to the CD burner for burning – for which the "Generic Devices", addressed via the device files /dev/sg* (SCSI generic), have to be recognised. As a rule, this is done by an sg module which is loaded automatically when one of these files is accessed. If this does not happen, load it in manually with modprobe -v sg.*