NAME

mkisofs − create an hybrid ISO9660/JOLIET/HFS filesystem with optional Rock Ridge attributes.

SYNOPSIS

mkisofs [*options*] [**−o** *filename*] *pathspec [pathspec ...]*

DESCRIPTION

mkisofs is effectively a pre-mastering program to generate an ISO9660/JOLIET/HFS hybrid filesystem.

mkisofs is capable of generating the **System Use Sharing Protocol records (SUSP)** specified by the **Rock Ridge Interchange Protocol.** This is used to further describe the files in the iso9660 filesystem to a unix host, and provides information such as longer filenames, uid/gid, posix permissions, symbolic links, block and character devices.

If Joliet or HFS hybrid command line options are specified, **mkisofs** will create additional filesystem meta data for Joliet or HFS. The file content in this case refers to the same data blocks on the media. It will generate a pure ISO9660 filesystem unless the Joliet or HFS hybrid command line options are given.

mkisofs can generate a *true* (or *shared*) HFS hybrid filesystem. The same files are seen as HFS files when accessed from a Macintosh and as ISO9660 files when accessed from other machines. HFS stands for *Hierarchical File System* and is the native file system used on Macintosh computers.

As an alternative, **mkisofs** can generate the *Apple Extensions to ISO9660* for each file. These extensions provide each file with CREATOR, TYPE and certain Finder Flags when accessed from a Macintosh. See the **HFS MACINTOSH FILE FORMATS** section below.

mkisofs takes a snapshot of a given directory tree, and generates a binary image which will correspond to an ISO9660 or HFS filesystem when written to a block device.

Each file written to the iso9660 filesystem must have a filename in the 8.3 format (8 characters, period, 3 characters, all upper case), even if Rock Ridge is in use. This filename is used on systems that are not able to make use of the Rock Ridge extensions (such as MS-DOS), and each filename in each directory must be different from the other filenames in the same directory. **mkisofs** generally tries to form correct names by forcing the unix filename to upper case and truncating as required, but often times this yields unsatisfactory results when there are cases where the truncated names are not all unique. **mkisofs** assigns weightings to each filename, and if two names that are otherwise the same are found the name with the lower priority is renamed to have a 3 digit number as an extension (where the number is guaranteed to be unique). An example of this would be the files foo.bar and foo.bar.~1~ - the file foo.bar.~1~ would be written as FOO000.BAR;1 and the file foo.bar would be written as FOO.BAR;1

When used with various HFS options, **mkisofs** will attempt to recognise files stored in a number of Apple/Unix file formats and will copy the data and resource forks as well as any relevant finder information. See the **HFS MACINTOSH FILE FORMATS** section below for more about formats **mkisofs** supports.

Note that **mkisofs** is not designed to communicate with the writer directly. Most writers have proprietary command sets which vary from one manufacturer to another, and you need a specialized tool to actually burn the disk.

The **cdrecord** utility is a utility capable of burning an actual disc. The latest version of **cdrecord** is available from ftp://ftp.berlios.de/pub/cdrecord

Also you should know that most cd writers are very particular about timing. Once you start to burn a disc, you cannot let their buffer empty before you are done, or you will end up with a corrupt disc. Thus it is critical that you be able to maintain an uninterrupted data stream to the writer for the entire time that the disc is being written.

pathspec is the path of the directory tree to be copied into the iso9660 filesystem. Multiple paths can be specified, and **mkisofs** will merge the files found in all of the specified path components to form the cdrom image.

If the option *−graft-points* has been specified, it is possible to graft the paths at points other than the root directory, and it is possible to graft files or directories onto the cdrom image with names different than what they have in the source filesystem. This is easiest to illustrate with a couple of examples. Let's start by assuming that a local file ../old.lis exists, and you wish to include it in the cdrom image.

foo/bar/=../old.lis

will include the file old.lis in the cdrom image at /foo/bar/old.lis, while

foo/bar/xxx=../old.lis

will include the file old.lis in the cdrom image at /foo/bar/xxx. The same sort of syntax can be used with directories as well. **mkisofs** will create any directories required such that the graft points exist on the cdrom image - the directories do not need to appear in one of the paths. By default, any directories that are created on the fly like this will have permissions 0555 and appear to be owned by the person running mkisofs. If you wish other permissions or owners of the intermediate directories, see −uid, −gid, −dir-mode, −file-mode and −new-dir-mode.

mkisofs will also run on Win9X/NT4 machines when compiled with Cygnus' cygwin (available from http://sourceware.cygnus.com/cygwin/). Therefore most references in this man page to *Unix* can be replaced with *Win32*.

OPTIONS

−abstract *FILE*
Specifies the abstract file name. This parameter can also be set in the file **.mkisofsrc** with ABST=filename. If specified in both places, the command line version is used.

−A *application_id*
Specifies a text string that will be written into the volume header. This should describe the application that will be on the disc. There is space on the disc for 128 characters of information. This parameter can also be set in the file **.mkisofsrc** with APPI=id. If specified in both places, the command line version is used.

−allow-lowercase
This options allows lower case characters to appear in iso9660 filenames.
This violates the ISO9660 standard, but it happens to work on some systems. Use with caution.

−allow-multidot
This options allows more than one dot to appear in iso9660 filenames. A leading dot is not affected by this option, it may be allowed separately using the **−L** option.
This violates the ISO9660 standard, but it happens to work on many systems. Use with caution.

−biblio *FILE*
Specifies the bibliographic file name. This parameter can also be set in the file **.mkisofsrc** with BIBLO=filename. If specified in both places, the command line version is used.

−cache-inodes
Cache inode and device numbers to find hard links to files. If **mkisofs** finds a hard link (a file with multiple names), then the file will only appear once on the CD. This helps to save space on the CD. The option **−cache-inodes** is default on UNIX like operating systems. Be careful when using this option on a filesystem without unique inode numbers as it may result in files containing the wrong content on CD.

−no-cache-inodes
Do not cache inode and device numbers. This option is needed whenever a filesystem does not have unique inode numbers. It is the default on **Cygwin**. As the Microsoft operating system that runs below **Cygwin** is not POSIX compliant, it does not have unique inode numbers. Cygwin creates fake inode numbers from a hash algorithm that is not 100% correct. If **mkisofs** would cache inodes on Cygwin, it would believe that some files are identical although they are not. The result

in this case are files that contain the wrong content if a significant amount of different files (> ˜5000) is in inside the tree that is to be archived. This does not happen when the **–no-cache-inodes is used, but the disadvantage is that mkisofs** cannot detect hardlinks anymore and the resulting CD image may be larger than expected.

–b *eltorito_boot_image*

Specifies the path and filename of the boot image to be used when making an "El Torito" bootable CD. The pathname must be relative to the source path specified to **mkisofs.** This option is required to make an "El Torito" bootable CD. The boot image must be exactly the size of either a 1200, 1440, or a 2880 kB floppy, and **mkisofs** will use this size when creating the output iso9660 filesystem. It is assumed that the first 512 byte sector should be read from the boot image (it is essentially emulating a normal floppy drive). This will work, for example, if the boot image is a LILO based boot floppy.

If the boot image is not an image of a floppy, you need to add one of the options: **-hard-disk-boot** or **-no-emul-boot**. If the system should not boot off the emulated disk, use **-no-boot**.

–eltorito-alt-boot

Start with a new set of "El Torito" boot parameters. This allows to have more than one El Torito boot on a CD. A maximum of 63 El Torito boot entries may be put on a single CD.

–B *img_sun4,img_sun4c,img_sun4m,img_sun4d,img_sun4e*

Specifies a comma separated list of boot images that are needed to make a bootable CD for sparc systems. There may be empty fields in the comma separated list. This option is required to make a bootable CD for Sun sparc systems. If the **–B** or **–sparc-boot** option has been specified, the first sector of the resulting image will contain a Sun disk label. This disk label specifies slice 0 for the iso9660 image and slice 1 . . . slice 7 for the boot images that have been specified with this option. Byte offset 512 . . . 8191 within each of the additional boot images must contain a primary boot that works for the appropriate sparc architecture. The rest of each of the images usually contains an ufs filesystem that is used primary kernel boot stage.

The implemented boot method is the boot method found with SunOS 4.x and SunOS 5.x. However, it does not depend on SunOS internals but only on properties of the Open Boot prom. For this reason, it should be usable for any OS that boots off a sparc system.

If the special filename **...** is used, the actual and all following boot partitions are mapped to the previous partition. If **mkisofs** is called with **–G** *image* **–B** ... all boot partitions are mapped to the partition that contains the iso9660 filesystem image and the generic boot image that is located in the first 16 sectors of the disk is used for all architectures.

–G *generic_boot_image*

Specifies the path and filename of the generic boot image to be used when making a generic bootable CD. The **generic_boot_image** will be placed on the first 16 sectors of the CD. The first 16 sectors are the sectors that are located before the iso9660 primary volume descriptor. If this option is used together with the **–sparc-boot** option, the Sun disk label will overlay the first 512 bytes of the generic boot image.

–hard-disk-boot

Specifies that the boot image used to create "El Torito" bootable CDs is a hard disk image. The hard disk image must begin with a master boot record that contains a single partition.

–no-emul-boot

Specifies that the boot image used to create "El Torito" bootable CDs is a ’no emulation’ image. The system will load and execute this image without performing any disk emulation.

–no-boot

Specifies that the created "El Torito" CD should be marked as not bootable. The system will provide an emulated drive for the image, but will boot off a standard boot device.

−boot-load-seg *segment_address*
Specifies the load segment address of the boot image for no-emulation "El Torito" CDs.

−boot-load-size *load_sectors*
Specifies the number of "virtual" (512-byte) sectors to load in no-emulation mode. The default is to load the entire boot file. Some BIOSes may have problems if this is not a multiple of 4.

−boot-info-table
Specifies that a 56-byte table with information of the CD-ROM layout will be patched in at offset 8 in the boot file. If this option is given, the boot file is modified in the source filesystem, so make sure to make a copy if this file cannot be easily regenerated! See the **EL TORITO BOOT INFO TABLE** section for a description of this table.

−C *last_sess_start,next_sess_start*
This option is needed when **mkisofs** is used to create a CDextra or the image of a second session or a higher level session for a multi session disk. The option **−C** takes a pair of two numbers separated by a comma. The first number is the sector number of the first sector in the last session of the disk that should be appended to. The second number is the starting sector number of the new session. The expected pair of numbers may be retrieved by calling **cdrecord −msinfo ...** If the **−C** option is used in conjunction with the **−M** option, **mkisofs** will create a filesystem image that is intended to be a continuation of the previous session. If the **−C** option is used without the **−M** option, **mkisofs** will create a filesystem image that is intended to be used for a second session on a CDextra. This is a multi session CD that holds audio data in the first session and a ISO9660 filesystem in the second session.

−c *boot_catalog*
Specifies the path and filename of the boot catalog to be used when making an "El Torito" bootable CD. The pathname must be relative to the source path specified to **mkisofs.** This option is required to make a bootable CD. This file will be inserted into the output tree and not created in the source filesystem, so be sure the specified filename does not conflict with an existing file, as it will be excluded. Usually a name like "boot.catalog" is chosen.

−check-oldnames
Check all filenames imported from old session for compliance with actual **mkisofs** iso9660 file naming rules. It his option is not present, only names with a length > 31 are checked as these files are a hard violation of the iso9660 standard.

−check-session *FILE*
Check all old sessions for compliance with actual **mkisofs** iso9660 file naming rules. This is a high level option that is a combination of the options: **−M** *FILE* **−C 0,0 −check-oldnames** For the parameter *FILE* see description of **−M** option.

−copyright *FILE*
Specifies the Copyright file name. This parameter can also be set in the file **.mkisofsrc** with COPY=filename. If specified in both places, the command line version is used.

−d
Omit trailing period from files that do not have a period.
This violates the ISO9660 standard, but it happens to work on many systems. Use with caution.

−D
Do not use deep directory relocation, and instead just pack them in the way we see them.
This violates the ISO9660 standard, but it happens to work on many systems. Use with caution.

−dir-mode *mode*
Overrides the mode of directories used to create the image to *mode*. Specifying this option automatically enables Rock Ridge extensions.

−dvd-video
Generate DVD-Video compliant UDF file system. This is done by sorting the order of the content of the appropriate files and by adding padding between the files if needed.

−f
Follow symbolic links when generating the filesystem. When this option is not in use, symbolic links will be entered using Rock Ridge if enabled, otherwise the file will be ignored.

−file-mode *mode*
Overrides the mode of regular files used to create the image to *mode*. Specifying this option automatically enables Rock Ridge extensions.

−gid *gid*
Overrides the gid read from the source files to the value of *gid*. Specifying this option automatically enables Rock Ridge extensions.

−gui Switch the behaviour for a GUI. This currently makes the output more verbose but may have other effects in future.

−graft-points
Allow to use graft points for filenames. If this option is used, all filenames are checked for graft points. The filename is divided at the first unescaped equal sign. All occurrences of '\\' and '=' characters must be escaped with '\\' if *−graft-points* has been specified.

−hide *glob*
Hide *glob* from being seen on the ISO9660 or Rock Ridge directory. *glob* is a shell wild-card-style pattern that must match any part of the filename or path. Multiple globs may be hidden. If *glob* matches a directory, then the contents of that directory will be hidden. In order to match a directory name, make sure the pathname does not include a trailing '/' character. All the hidden files will still be written to the output CD image file. Should be used with the **−hide-joliet** option. See README.hide for more details.

−hide-list *file*
A file containing a list of *globs* to be hidden as above.

−hidden *glob*
Add the hidden (existence) ISO9660 directory attribute for *glob*. This attribute will prevent *glob* from being listed on DOS based systems if the /A flag is not used for the listing. *glob* is a shell wild-card-style pattern that must match any part of the filename or path. In order to match a directory name, make sure the pathname does not include a trailing '/' character. Multiple globs may be hidden.

−hidden-list *file*
A file containing a list of *globs* to get the hidden attribute as above.

−hide-joliet *glob*
Hide *glob* from being seen on the Joliet directory. *glob* is a shell wild-card-style pattern that must match any part of the filename or path. Multiple globs may be hidden. If *glob* matches a directory, then the contents of that directory will be hidden. In order to match a directory name, make sure the pathname does not include a trailing '/' character. All the hidden files will still be written to the output CD image file. Should be used with the **−hide** option. See README.hide for more details.

−hide-joliet-list *file*
A file containing a list of *globs* to be hidden as above.

−hide-joliet-trans-tbl
Hide the **TRANS.TBL** files from the Joliet tree. These files usually don't make sense in the Joliet World as they list the real name and the ISO9660 name which may both be different from the Joliet name.

−hide-rr-moved
Rename the directory **RR_MOVED** to **.rr_moved** in the Rock Ridge tree. It seems to be impossible to completely hide the **RR_MOVED** directory from the Rock Ridge tree. This option only makes the visible tree better to understand for people who don't know what this directory is for. If you need to have no **RR_MOVED** directory at all, you should use the **−D** option. Note that in case that the **−D** option has been specified, the resulting filesystem is not ISO9660 level-1 compliant and will not be readable on MS-DOS. See also **NOTES** section for more information on the **RR_MOVED** directory.

–l Allow full 31 character filenames. Normally the ISO9660 filename will be in an 8.3 format which is compatible with MS-DOS, even though the ISO9660 standard allows filenames of up to 31 characters. If you use this option, the disc may be difficult to use on a MS-DOS system, but this comes in handy on some other systems (such as the Amiga). Use with caution.

–input-charset *charset*
Input charset that defines the characters used in local file names. To get a list of valid charset names, call **mkisofs –input-charset help.** To get a 1:1 mapping, you may use **default** as charset name. The default initial values are *cp437* on DOS based systems and *iso8859-1* on all other systems. See **CHARACTER SETS** section below for more details.

–output-charset *charset*
Output charset that defines the characters that will be used in Rock Ridge file names. Defaults to the input charset. See **CHARACTER SETS** section below for more details.

–iso-level *level*
Set the iso9660 conformance level. Valid numbers are 1..3.

With level 1, files may only consist of one section and filenames are restricted to 8.3 characters.

With level 2, files may only consist of one section.

With level 3, no restrictions apply.

With all iso9660 levels all filenames are restricted to upper case letters, numbers and the underscore (_). The maximum filename length is restricted to 31 characters, the directory nesting level is restricted to 8 and the maximum path length is limited to 255 characters.

–J Generate Joliet directory records in addition to regular iso9660 file names. This is primarily useful when the discs are to be used on Windows-NT or Windows-95 machines. The Joliet filenames are specified in Unicode and each path component can be up to 64 Unicode characters long. Note that Joliet is no standard - CD's that use only Joliet extensions but no standard Rock Ridge extensions may usually only be used on Microsoft Win32 systems. Furthermore, the fact that the filenames are limited to 64 characters and the fact that Joliet uses the UTF-16 coding for Unicode characters causes interoperability problems.

–joliet-long
Allow Joliet filenames to be up to 103 Unicode characters. This breaks the Joliet specification - but appears to work. Use with caution. The number 103 is derived from: the maximum Directory Record Length (254), minus the length of Directory Record (33), minus CD-ROM XA System Use Extension Information (14), divided by the UTF-16 character size (2).

–jcharset *charset*
Same as using **–input-charset** *charset* and **–J** options. See **CHARACTER SETS** section below for more details.

–L Allow ISO9660 filenames to begin with a period. Usually, a leading dot is replaced with an underscore in order to maintain MS-DOS compatibility.
This violates the ISO9660 standard, but it happens to work on many systems. Use with caution.

–log-file *log_file*
Redirect all error, warning and informational messages to *log_file* instead of the standard error.

–m *glob*
Exclude *glob* from being written to CDROM. *glob* is a shell wild-card-style pattern that must match part of the filename (not the path as with option **–x**). Technically *glob* is matched against the *d->d_name* part of the directory entry. Multiple globs may be excluded. Example:

mkisofs –o rom –m '*.o' –m core –m foobar would exclude all files ending in ".o", called "core" or "foobar" to be copied to CDROM. Note that if you had a directory called "foobar" it too (and of course all its descendants) would be excluded.

NOTE: The **−m** and **−x** option description should both be updated, they are wrong. Both now work identical and use filename globbing. A file is excluded if either the last component matches or the whole path matches.

−exclude-list *file*
A file containing a list of *globs* to be exclude as above.

−max-iso9660-filenames
Allow 37 chars in iso9660 filenames. This option forces the **−N** option as the extra name space is taken from the space reserved for ISO-9660 version numbers.
This violates the ISO9660 standard, but it happens to work on many systems. Although a conforming application needs to provide a buffer space of at least 37 characters, disks created with this option may cause a buffer overflow in the reading operating system. Use with extreme care.

−M *path*
or

−M *device*
Specifies path to existing iso9660 image to be merged. The alternate form takes a SCSI device specifier that uses the same syntax as the **dev=** parameter of **cdrecord.** The output of **mkisofs** will be a new session which should get written to the end of the image specified in −M. Typically this requires multi-session capability for the recorder and cdrom drive that you are attempting to write this image to. This option may only be used in conjunction with the **−C** option.

−N Omit version numbers from ISO9660 file names.
This violates the ISO9660 standard, but no one really uses the version numbers anyway. Use with caution.

−new-dir-mode *mode*
Mode to use when creating new directories in the iso fs image. The default mode is 0555.

−nobak

−no-bak
Do not include backup files files on the iso9660 filesystem. If the **−no-bak** option is specified, files that contain the characters '~' or '#' or end in '.bak' will not be included (these are typically backup files for editors under unix).

−force-rr
Do not use the automatic Rock Ridge attributes recognition for previous sessions. This helps to show rotten iso9660 extension records as e.g. created by NERO burning ROM.

−no-rr Do not use the Rock Ridge attributes from previous sessions. This may help to avoid getting into trouble when **mkisofs** finds illegal Rock Ridge signatures on an old session.

−no-split-symlink-components
Don't split the SL components, but begin a new Continuation Area (CE) instead. This may waste some space, but the SunOS 4.1.4 cdrom driver has a bug in reading split SL components (link_size = component_size instead of link_size += component_size).

−no-split-symlink-fields
Don't split the SL fields, but begin a new Continuation Area (CE) instead. This may waste some space, but the SunOS 4.1.4 and Solaris 2.5.1 cdrom driver have a bug in reading split SL fields (a '/' can be dropped).

−o *filename*
is the name of the file to which the iso9660 filesystem image should be written. This can be a disk file, a tape drive, or it can correspond directly to the device name of the optical disc writer. If not specified, stdout is used. Note that the output can also be a block special device for a regular disk

drive, in which case the disk partition can be mounted and examined to ensure that the premastering was done correctly.

−pad Pad the end of the ISO9660 by 16 sectors (32kB). If the total size then is not a multiple of 16 sectors, the needed number of sectors is added. If the option **−B** is used, then there is a second padding at the end of the boot partitions.

The padding is needed as many operating systems (e.g. Linux) implement read ahead bugs in their filesystem I/O. These bugs result in read errors on one or more files that are located at the end of a track. They are usually present when the CD is written in Track at Once mode or when the disk is written as mixed mode CD where an audio track follows the data track.

To avoid problems with I/O error on the last file on the filesystem, the **−pad** option has been made the default.

−no−pad
Do not Pad the end of the ISO9660 by 16 sectors (32kB).

−path-list *file*
A file containing a list of *pathspec* directories and filenames to be added to the ISO9660 filesystem. This list of pathspecs are processed after any that appear on the command line. If the argument is −, then the list is read from the standard input.

−P *publisher_id*
Specifies a text string that will be written into the volume header. This should describe the publisher of the CDROM, usually with a mailing address and phone number. There is space on the disc for 128 characters of information. This parameter can also be set in the file **.mkisofsrc** with PUBL=. If specified in both places, the command line version is used.

−p *preparer_id*
Specifies a text string that will be written into the volume header. This should describe the preparer of the CDROM, usually with a mailing address and phone number. There is space on the disc for 128 characters of information. This parameter can also be set in the file **.mkisofsrc** with PREP=. If specified in both places, the command line version is used.

−print-size
Print estimated filesystem size in multiples of the sector size (2048 bytes) and exit. This option is needed for Disk At Once mode and with some CD-R drives when piping directly into **cdrecord.** In this case it is needed to know the size of the filesystem before the actual CD-creation is done. The option −print-size allows to get this size from a "dry-run" before the CD is actually written. Old versions of **mkisofs** did write this information (among other information) to *stderr*. As this turns out to be hard to parse, the number without any other information is now printed on **stdout** too. If you like to write a simple shell script, redirect **stderr** and catch the number from **stdout**. This may be done with:

cdblocks=‘ mkisofs -print-size -quiet . . . ‘

mkisofs . . . | cdrecord . . . tsize=${cdblocks}s -

−quiet This makes **mkisofs** even less verbose. No progress output will be provided.

−R Generate SUSP and RR records using the Rock Ridge protocol to further describe the files on the iso9660 filesystem.

−r This is like the −R option, but file ownership and modes are set to more useful values. The uid and gid are set to zero, because they are usually only useful on the author's system, and not useful to the client. All the file read bits are set true, so that files and directories are globally readable on the client. If any execute bit is set for a file, set all of the execute bits, so that executables are globally executable on the client. If any search bit is set for a directory, set all of the search bits, so that directories are globally searchable on the client. All write bits are cleared, because the CD-

Rom will be mounted read-only in any case. If any of the special mode bits are set, clear them, because file locks are not useful on a read-only file system, and set-id bits are not desirable for uid 0 or gid 0. When used on Win32, the execute bit is set on *all* files. This is a result of the lack of file permissions on Win32 and the Cygwin POSIX emulation layer. See also −uid −gid, −dir-mode, −file-mode and −new-dir-mode.

−relaxed-filenames

The option **−relaxed-filenames** allows ISO9660 filenames to include digits, uppercase characters and all other 7 bit ASCII characters (resp. anything except lowercase characters).
This violates the ISO9660 standard, but it happens to work on many systems. Use with caution.

−sort *sort file*

Sort file locations on the media. Sorting is controlled by a file that contains pairs of filenames and sorting offset weighting. If the weighting is higher, the file will be located closer to the beginning of the media, if the weighting is lower, the file will be located closer to the end of the media. There must be only one space or tabs character between the filename and the weight and the weight must be the last characters on a line. The filename is taken to include all the characters up to, but not including the last space or tab character on a line. This is to allow for space characters to be in, or at the end of a filename. This option does **not** sort the order of the file names that appear in the ISO9660 directory. It sorts the order in which the file data is written to the CD image - which may be useful in order to optimize the data layout on a CD. See README.sort for more details.

−split-output

Split the output image into several files of approximately 1 GB. This helps to create DVD sized iso9660 images on operating systems without large file support. Cdrecord will concatenate more than one file into a single track if writing to a DVD. To make **−split-output** work, the **−o** *filename* option must be specified. The resulting outout images will be named: *filename_00*,*filename_01,*filename_02...

−sysid *ID*

Specifies the system ID. This parameter can also be set in the file **.mkisofsrc** with SYSI=system_id. If specified in both places, the command line version is used.

−T Generate a file TRANS.TBL in each directory on the CDROM, which can be used on non-Rock Ridge capable systems to help establish the correct file names. There is also information present in the file that indicates the major and minor numbers for block and character devices, and each symlink has the name of the link file given.

−table-name *TABLE_NAME*

Alternative translation table file name (see above). Implies the **−T** option. If you are creating a multi-session image you must use the same name as in the previous session.

−ucs-level *level*

Set Unicode conformance level in the Joliet SVD. The default level is 3. It may be set to 1..3 using this option.

−udf Include **UDF** support in the generated filesystem image. **UDF** support is currently in alpha status and for this reason, it is not possible to create UDF only images. **UDF** data structures are currently coupled to the Joliet structures, so there are many pitfalls with the current implementation. There is no UID/GID support, there is no POSIX permission support, there is no support for symlinks. Note that **UDF** wastes the space from sector ~20 to sector 256 at the beginning of the disk in addition to the spcae needed for real **UDF** data structures.

−uid *uid*

Overrides the uid read from the source files to the value of *uid*. Specifying this option automatically enables Rock Ridge extensions.

−use-fileversion

The option **−use-fileversion** allows mkisofs to use file version numbers from the filesystem. If the option is not specified, **mkisofs** creates a version if 1 for all files. File versions are strings in the range *;1* to *;32767* This option is the default on VMS.

−U Allows "Untranslated" filenames, completely violating the iso9660 standards described above. Forces on the −d, −l, −L, −N, −relaxed-filenames, −allow-lowercase, −allow-multidot and −no-iso-translate flags. It allows more than one '.' character in the filename, as well as mixed case filenames. This is useful on HP-UX system, where the built-in CDFS filesystem does not recognize ANY extensions. Use with extreme caution.

−no-iso-translate
Do not translate the characters '#' and '~' which are invalid for iso9660 filenames. These characters are though invalid often used by Microsoft systems.
This violates the ISO9660 standard, but it happens to work on many systems. Use with caution.

−V *volid*
Specifies the volume ID (volume name or label) to be written into the master block. This parameter can also be set in the file **.mkisofsrc** with VOLI=id. If specified in both places, the command line version is used. Note that if you assign a volume ID, this is the name that will be used as the mount point used by the Solaris volume management system and the name that is assigned to the disc on a Microsoft Win32 or Apple Mac platform.

−volset *ID*
Specifies the volset ID. This parameter can also be set in the file **.mkisofsrc** with VOLS=volset_id. If specified in both places, the command line version is used.

−volset-size *#*
Sets the volume set size to #. The volume set size is the number of CD's that are in a CD set. The **−volset-size** option may be used to create CD's that are part of e.g. a Operation System installation set of CD's. The option **−volset-size** must be specified before **−volset-seqno** on each command line.

−volset-seqno *#*
Sets the volume set sequence number to #. The volume set sequence number is the index number of the current CD in a CD set. The option **−volset-size** must be specified before **−volset-seqno** on each command line.

−v Verbose execution. If given twice on the command line, extra debug information will be printed.

−x *path* Exclude *path* from being written to CDROM. *path* must be the complete pathname that results from concatenating the pathname given as command line argument and the path relative to this directory. Multiple paths may be excluded. Example:

mkisofs −o cd −x /local/dir1 −x /local/dir2 /local

NOTE: The **−m** and **−x** option description should both be updated, they are wrong. Both now work identical and use filename globbing. A file is excluded if either the last component matches or the whole path matches.

−z Generate special RRIP records for transparently compressed files. This is only of use and interest for hosts that support transparent decompression, such as Linux 2.4.14 or later. You must specify the **−R** or **−r** options to enable RockRidge, and generate compressed files using the **mkzftree** utility before running **mkisofs**. Note that transparent compression is a nonstandard Rock Ridge extension. The resulting disks are only transparently readable if used on Linux. On other operating systems you will need to call **mkzftree** by hand to decompress the files.

HFS OPTIONS

−hfs Create an ISO9660/HFS hybrid CD. This option should be used in conjunction with the **−map**, **−magic** and/or the various *double dash* options given below.

−apple Create an ISO9660 CD with Apple's extensions. Similar to the **−hfs** option, except that the Apple Extensions to ISO9660 are added instead of creating an HFS hybrid volume.

–map *mapping_file*
Use the *mapping_file* to set the CREATOR and TYPE information for a file based on the filename's extension. A filename is mapped only if it is not one of the know Apple/Unix file formats. See the **HFS CREATOR/TYPE** section below.

–magic *magic_file*
The CREATOR and TYPE information is set by using a file's *magic number* (usually the first few bytes of a file). The *magic_file* is only used if a file is not one of the known Apple/Unix file formats, or the filename extension has not been mapped using the **–map** option. See the **HFS CREATOR/TYPE** section below for more details.

–hfs-creator *CREATOR*
Set the default CREATOR for all files. Must be exactly 4 characters. See the **HFS CREATOR/TYPE** section below for more details.

–hfs-type *TYPE*
Set the default TYPE for all files. Must be exactly 4 characters. See the **HFS CREATOR/TYPE** section below for more details.

–probe Search the contents of files for all the known Apple/Unix file formats. See the **HFS MACINTOSH FILE FORMATS** section below for more about these formats. However, the only way to check for *MacBinary* and *AppleSingle* files is to open and read them. Therefore this option *may* increase processing time. It is better to use one or more *double dash* options given below if the Apple/Unix formats in use are known.

–no-desktop
Do not create (empty) Desktop files. New HFS Desktop files will be created when the CD is used on a Macintosh (and stored in the System Folder). By default, empty Desktop files are added to the HFS volume.

–mac-name
Use the HFS filename as the starting point for the ISO9660, Joliet and Rock Ridge file names. See the **HFS MACINTOSH FILE NAMES** section below for more information.

–boot-hfs-file *driver_file*
Installs the *driver_file* that *may* make the CD bootable on a Macintosh. See the **HFS BOOT DRIVER** section below. (Alpha).

–part Generate an HFS partition table. By default, no partition table is generated, but some older Macintosh CDROM drivers need an HFS partition table on the CDROM to be able to recognize a hybrid CDROM.

–auto *AutoStart_file*
Make the HFS CD use the QuickTime 2.0 Autostart feature to launch an application or document. The given filename must be the name of a document or application located at the top level of the CD. The filename must be less than 12 characters. (Alpha).

–cluster-size *size*
Set the size in bytes of the cluster or allocation units of PC Exchange files. Implies the **––exchange** option. See the **HFS MACINTOSH FILE FORMATS** section below.

–hide-hfs *glob*
Hide *glob* from the HFS volume. The file or directory will still exist in the ISO9660 and/or Joliet directory. *glob* is a shell wild-card-style pattern that must match any part of the filename Multiple globs may be excluded. Example:

mkisofs –o rom –hfs –hide-hfs '*.o' –hide-hfs foobar

would exclude all files ending in ".o" or called "foobar" from the HFS volume. Note that if you had a directory called "foobar" it too (and of course all its descendants) would be excluded. The *glob* can also be a path name relative to the source directories given on the command line.

Example:

mkisofs −o rom −hfs −hide-hfs src/html src

would exclude just the file or directory called "html" from the "src" directory. Any other file or directory called "html" in the tree will not be excluded. Should be used with the **−hide** and/or **−hide-joliet** options. In order to match a directory name, make sure the pathname does not include a trailing '/' character. See README.hide for more details.

−hide-hfs-list *file*
A file containing a list of *globs* to be hidden as above.

−hfs-volid *hfs_volid*
Volume name for the HFS partition. This is the name that is assigned to the disc on a Macintosh and replaces the *volid* used with the **−V** option

−icon-position
Use the icon position information, if it exists, from the Apple/Unix file. The icons will appear in the same position as they would on a Macintosh desktop. Folder location and size on screen, its scroll positions, folder View (view as Icons, Small Icons, etc.) are also preserved. This option may become set by default in the future. (Alpha).

−root-info *file*
Set the location, size on screen, scroll positions, folder View etc. for the root folder of an HFS volume. See README.rootinfo for more information. (Alpha)

−prep-boot *FILE*
PReP boot image file. Up to 4 are allowed. See README.prep_boot (Alpha)

−input-hfs-charset *charset*
Input charset that defines the characters used in HFS file names when used with the *−mac-name* option. The default charset is cp10000 (Mac Roman) *cp10000* (Mac Roman) See **CHARACTER SETS** and **HFS MACINTOSH FILE NAMES** sections below for more details.

−output-hfs-charset *charset*
Output charset that defines the characters that will be used in the HFS file names. Defaults to the input charset. See **CHARACTER SETS** section below for more details.

−hfs-unlock
By default, **mkisofs** will create an HFS volume that is *locked*. This option leaves the volume unlocked so that other applications (e.g. hfsutils) can modify the volume. See the **HFS PROBLEMS/LIMITATIONS** section below for warnings about using this option.

−hfs-bless *folder_name*
"Bless" the given directory (folder). This is usually the **System Folder** and is used in creating HFS bootable CDs. The name of the directory must be the whole path name as **mkisofs** sees it. e.g. if the given pathspec is ./cddata and the required folder is called System Folder, then the whole path name is "./cddata/System Folder" (remember to use quotes if the name contains spaces).

−hfs-parms *PARAMETERS*
Override certain parameters used to create the HFS file system. Unlikely to be used in normal circumstances. See the libhfs_iso/hybrid.h source file for details.

−−cap Look for AUFS CAP Macintosh files. Search for CAP Apple/Unix file formats only. Searching for the other possible Apple/Unix file formats is disabled, unless other *double dash* options are given.

−−netatalk
Look for NETATALK Macintosh files

−−double
Look for AppleDouble Macintosh files

−−ethershare
Look for Helios EtherShare Macintosh files

−−ushare
Look for IPT UShare Macintosh files

−−exchange
Look for PC Exchange Macintosh files

−−sgi Look for SGI Macintosh files

−−xinet
Look for XINET Macintosh files

−−macbin
Look for MacBinary Macintosh files

−−single
Look for AppleSingle Macintosh files

−−dave
Look for Thursby Software Systems DAVE Macintosh files

−−sfm Look for Microsoft's Services for Macintosh files (NT only) (Alpha)

−−osx−double
Look for MacOS X AppleDouble Macintosh files

−−osx−hfs
Look for MacOS X HFS Macintosh files

CHARACTER SETS

mkisofs processes file names in a POSIX compliant way as strings of 8-bit characters. To represent all codings for all languages, 8-bit characters are not sufficient. Unicode or **ISO-10646** define character codings that need at least 21 bits to represent all known languages. They may be represented with **UTF-32**, **UTF-16** or **UTF-8** coding. **UTF-32** uses a plain 32-bit coding but seems to be uncommon. **UTF-16** is used by Microsoft with Win32 with the disadvantage that it only supports a subset of all codes and that 16-bit characters are not compliant with the POSIX filesystem interface.

Modern UNIX operating systems may use **UTF-8** coding for filenames. This coding allows to use the complete Unicode code set. Each 32-bit character is represented by one or more 8-bit characters. If a character is coded in **ISO-8859-1** (used in Central Europe and North America) is maps 1:1 to a **UTF-32** or **UTF-16** coded Unicode character. If a character is coded in **7-Bit ASCII** (used in USA and other countries with limted character set) is maps 1:1 to a **UTF-32**, **UTF-16** or **UTF-8** coded Unicode character. Character codes that cannot be represented as a single byte in UTF-8 (typically if the value is > 0x7F) use escape sequences that map to more than one 8-bit character.

If all operating systems would use **UTF-8** coding, **mkisofs** would not need to recode characters in file names. Unfortunately, Apple uses completely nonstandard codings and Microsoft uses a Unicode coding that is not compatible with the POSIX filename interface.

For all non **UTF-8** coded operating systems, the actual character that each byte represents depends on the *character set* or *codepage* (which is the name used by Microsoft) used by the local operating system in use - the characters in a character set will reflect the region or natural language used by the user.

Usually character codes 0x00-0x1f are control characters, codes 0x20-0x7f are the 7 bit ASCII characters and (on PC's and Mac's) 0x80-0xff are used for other characters. Unfortunately even this does not follow ISO standards that reserve the range 0x80-0x9f for control characters and only allow 0xa0-0xff for other characters.

As there is a lot more than 256 characters/symbols in use, only a small subset are represented in a character set. Therefore the same character code may represent a different character in different character sets. So a file name generated, say in central Europe, may not display the same character when viewed on a machine in, say eastern Europe.

To make matters more complicated, different operating systems use different character sets for the region or language. For example the character code for "small e with acute accent" may be character code 0x82 on a PC, code 0x8e on a Macintosh and code 0xe9 on a UNIX system. Note while the codings used on a PC or Mac are nonstandard, Unicode codes this character as 0x00000000e9 which is basically the same value as the value used by most UNIX systems.

As long as not all operating systems and applications will use the Unicode character set as the basis for file names in a unique way, it may be necessary to specify which character set your file names use in and which character set the file names should appear on the CD.

There are four options to specify the character sets you want to use:

−input-charset
: Defines the local character set you are using on your host machine. Any character set conversions that take place will use this character set as the staring point. The default input character sets are *cp437* on DOS based systems and *iso8859-1* on all other systems.

  If the *−J* option is given, then the Unicode equivalents of the input character set will be used in the Joliet directory. Using the *−jcharset* option is the same as using the *−input-charset* and *−J* options.

−output-charset
: Defines the character set that will be used with for the Rock Ridge names on the CD. Defaults to the input character set. Only likely to be useful if used on a non-Unix platform. e.g. using **mkisofs** on a Microsoft Win32 machine to create Rock Ridge CDs. If you are using **mkisofs** on a Unix machine, it is likely that the output character set will be the same as the input character set.

−input-hfs-charset
: Defines the HFS character set used for HFS file names decoded from any of the various Apple/Unix file formats. Only useful when used with *−mac-name* option. See the **HFS MACINTOSH FILE NAMES** for more information. Defaults to *cp10000* (Mac Roman).

−output-hfs-charset
: Defines the HFS character set used to create HFS file names from the input character set in use. In most cases this will be from the character set given with the *−input-charset* option. Defaults to the input HFS character set.

There are a number of character sets built in to *mkisofs*. To get a listing, use **mkisofs −input-charset help.**

Additional character sets can be read from file for any of the character set options by giving a filename as the argument to the options. The given file will only be read if its name does not match one of the built in character sets.

The format of the character set files is the same as the mapping files available from http://www.unicode.org/Public/MAPPINGS The format of these files is:

    Column #1 is the input byte code (in hex as 0xXX)
    Column #2 is the Unicode (in hex as 0xXXXX)
    Rest of the line is ignored.

Any blank line, line without two (or more) columns in the above format or comments lines (starting with the # character) are ignored without any warnings. Any missing input code is mapped to Unicode character 0x0000.

Note that there is no support for 16 bit UNICODE (UTF-16) or 32 bit UNICODE (UTF-32) coding because this coding is not POSIX compliant. There should be support for UTF-8 UNICODE coding which is compatible to POSIX filenames and supported by moder UNIX implementations such as Solaris.

A 1:1 character set mapping can be defined by using the keyword *default* as the argument to any of the character set options. This is the behaviour of older (v1.12) versions of **mkisofs**.

The ISO9660 file names generated from the input filenames are not converted from the input character set. The ISO9660 character set is a very limited subset of the ASCII characters, so any conversion would be pointless.

Any character that **mkisofs** can not convert will be replaced with a '_' character.

HFS CREATOR/TYPE

A Macintosh file has two properties associated with it which define which application created the file, the *CREATOR* and what data the file contains, the *TYPE*. Both are (exactly) 4 letter strings. Usually this allows a Macintosh user to double-click on a file and launch the correct application etc. The CREATOR and TYPE of a particular file can be found by using something like ResEdit (or similar) on a Macintosh.

The CREATOR and TYPE information is stored in all the various Apple/Unix encoded files. For other files it is possible to base the CREATOR and TYPE on the filename's extension using a *mapping* file (the **–map** option) and/or using the *magic number* (usually a *signature* in the first few bytes) of a file (the **–magic** option). If both these options are given, then their order on the command line is important. If the **–map** option is given first, then a filename extension match is attempted before a magic number match. However, if the **–magic** option is given first, then a magic number match is attempted before a filename extension match.

If a mapping or magic file is not used, or no match is found then the default CREATOR and TYPE for all regular files can be set by using entries in the **.mkisofsrc** file or using the **–hfs-creator** and/or **–hfs-type** options, otherwise the default CREATOR and TYPE are 'unix' and 'TEXT'.

The format of the *mapping* file is the same *afpfile* format as used by *aufs*. This file has five columns for the *extension*, *file translation*, *CREATOR*, *TYPE* and *Comment*. Lines starting with the '#' character are comment lines and are ignored. An example file would be like:

```
# Example filename mapping file
#
# EXTN   XLate   CREATOR   TYPE     Comment
.tif     Raw     '8BIM'    'TIFF'   "Photoshop TIFF image"
.hqx     Ascii   'BnHq'    'TEXT'   "BinHex file"
.doc     Raw     'MSWD'    'WDBN'   "Word file"
.mov     Raw     'TVOD'    'MooV'   "QuickTime Movie"
*        Ascii   'ttxt'    'TEXT'   "Text file"
```

Where:

The first column *EXTN* defines the Unix filename extension to be mapped. The default mapping for any filename extension that doesn't match is defined with the "*" character.

The *Xlate* column defines the type of text translation between the Unix and Macintosh file it is ignored by **mkisofs**, but is kept to be compatible with **aufs**(1). Although **mkisofs** does not alter the contents of a file, if a binary file has it's TYPE set as 'TEXT', it *may* be read incorrectly on a Macintosh. Therefore a better choice for the default TYPE may be '????'

The *CREATOR* and *TYPE* keywords must be 4 characters long and enclosed in single quotes.

The comment field is enclosed in double quotes - it is ignored by **mkisofs**, but is kept to be compatible with **aufs**.

The format of the *magic* file is almost identical to the **magic**(4) file used by the Linux **file**(1) command - the routines for reading and decoding the *magic* file are based on the Linux **file**(1) command.

This file has four tab separated columns for the *byte offset*, *type*, *test* and *message*. Lines starting with the '#' character are comment lines and are ignored. An example file would be like:

```
# Example magic file
#
# off   type   test   message
```

0	string	GIF8	8BIM GIFf GIF image
0	beshort	0xffd8	8BIM JPEG image data
0	string	SIT!	SIT! SIT! StuffIt Archive
0	string	\037\235	LZIV ZIVU standard unix compress
0	string	\037\213	GNUz ZIVU gzip compressed data
0	string	%!	ASPS TEXT Postscript
0	string	\004%!	ASPS TEXT PC Postscript with a ^D to start
4	string	moov	txtt MooV QuickTime movie file (moov)
4	string	mdat	txtt MooV QuickTime movie file (mdat)

The format of the file is described in the **magic**(4) man page. The only difference here is that for each entry in the magic file, the *message* for the initial offset **must** be 4 characters for the CREATOR followed by 4 characters for the TYPE - white space is optional between them. Any other characters on this line are ignored. Continuation lines (starting with a '>') are also ignored i.e. only the initial offset lines are used.

Using the **–magic** option may significantly increase processing time as each file has to opened and read to find it's magic number.

In summary, for all files, the default CREATOR is 'unix' and the default TYPE is 'TEXT'. These can be changed by using entries in the *.mkisofsrc* file or by using the **–hfs-creator** and/or **–hfs-type** options.

If the a file is in one of the known Apple/Unix formats (and the format has been selected), then the CREATOR and TYPE are taken from the values stored in the Apple/Unix file.

Other files can have their CREATOR and TYPE set from their file name extension (the **–map** option), or their magic number (the **–magic** option). If the default match is used in the *mapping* file, then these values override the default CREATOR and TYPE.

A full CREATOR/TYPE database can be found at http://www.angelfire.com/il/szekely/index.html

HFS MACINTOSH FILE FORMATS

Macintosh files have two parts called the *Data* and *Resource* fork. Either may be empty. Unix (and many other OSs) can only cope with files having one part (or fork). To add to this, Macintosh files have a number of attributes associated with them - probably the most important are the TYPE and CREATOR. Again Unix has no concept of these types of attributes.

e.g. a Macintosh file may be a JPEG image where the image is stored in the Data fork and a desktop thumbnail stored in the Resource fork. It is usually the information in the data fork that is useful across platforms.

Therefore to store a Macintosh file on a Unix filesystem, a way has to be found to cope with the two forks and the extra attributes (which are referred to as the *finder info*). Unfortunately, it seems that every software package that stores Macintosh files on Unix has chosen a completely different storage method.

The Apple/Unix formats that *mkisofs* (partially) supports are:

CAP AUFS format
: Data fork stored in a file. Resource fork in subdirectory .resource with same filename as data fork. Finder info in .finderinfo subdirectory with same filename.

AppleDouble/Netatalk
: Data fork stored in a file. Resource fork stored in a file with same name prefixed with "%". Finder info also stored in same "%" file. Netatalk uses the same format, but the resource fork/finderinfo stored in subdirectory .AppleDouble with same name as data fork.

AppleSingle
: Data structures similar to above, except both forks and finder info are stored in one file.

Helios EtherShare
: Data fork stored in a file. Resource fork and finder info together in subdirectory .rsrc with same filename as data fork.

IPT UShare

Very similar to the EtherShare format, but the finder info is stored slightly differently.

MacBinary

Both forks and finder info stored in one file.

Apple PC Exchange

Used by Macintoshes to store Apple files on DOS (FAT) disks. Data fork stored in a file. Resource fork in subdirectory resource.frk (or RESOURCE.FRK). Finder info as one record in file finder.dat (or FINDER.DAT). Separate finder.dat for each data fork directory.

Note: *mkisofs* needs to know the native FAT cluster size of the disk that the PC Exchange files are on (or have been copied from). This size is given by the **−cluster-size** option. The cluster or allocation size can be found by using the DOS utility **CHKDSK**.

May not work with PC Exchange v2.2 or higher files (available with MacOS 8.1). DOS media containing PC Exchange files should be mounted as type **msdos** (not **vfat**) when using Linux.

SGI/XINET

Used by SGI machines when they mount HFS disks. Data fork stored in a file. Resource fork in subdirectory .HSResource with same name. Finder info as one record in file .HSancillary. Separate .HSancillary for each data fork directory.

Thursby Software Systems DAVE

Allows Macintoshes to store Apple files on SMB servers. Data fork stored in a file. Resource fork in subdirectory resource.frk. Uses the AppleDouble format to store resource fork.

Services for Macintosh

Format of files stored by NT Servers on NTFS filesystems. Data fork is stored as "filename". Resource fork stored as a NTFS *stream* called "filename:AFP_Resource". The finder info is stored as a NTFS *stream* called "filename:Afp_AfpInfo". These streams are normally invisible to the user.

Warning: mkisofs only partially supports the SFM format. If an HFS file or folder stored on the NT server contains an *illegal* NT character in its name, then NT converts these characters to *Private Use Unicode* characters. The characters are: " * / < > ? | also a space or period if it is the last character of the file name, character codes 0x01 to 0x1f (control characters) and Apple’ apple logo.

Unfortunately, these private Unicode characters are not readable by the mkisofs NT executable. Therefore any file or directory name containing these characters will be ignored - including the contents of any such directory.

MacOS X AppleDouble

When HFS/HFS+ files are copied or saved by MacOS X on to a non-HFS file system (e.g. UFS, NFS etc.), the files are stored in AppleDouble format. Data fork stored in a file. Resource fork stored in a file with same name prefixed with "._". Finder info also stored in same "._" file.

MacOS X HFS (Alpha)

Not really an Apple/Unix encoding, but actual HFS/HFS+ files on a MacOS X system. Data fork stored in a file. Resource fork stored in a pseudo file with the same name with the suffix ’/rsrc’. The finderinfo is only available via a MacOS X library call.

Notes: (also see README.macosx)

Only works when used on MacOS X.

If a file is found with a zero length resource fork and empty finderinfo, it is assumed not to have any Apple/Unix encoding - therefore a TYPE and CREATOR can be set using other methods.

mkisofs will attempt to set the CREATOR, TYPE, date and possibly other flags from the finder info. Additionally, if it exists, the Macintosh filename is set from the finder info, otherwise the Macintosh name is based on the Unix filename - see the **HFS MACINTOSH FILE NAMES** section below.

When using the **−apple** option, the TYPE and CREATOR are stored in the optional System Use or SUSP field in the ISO9660 Directory Record - in much the same way as the Rock Ridge attributes are. In fact to make life easy, the Apple extensions are added at the beginning of the existing Rock Ridge attributes (i.e. to get the Apple extensions you get the Rock Ridge extensions as well).

The Apple extensions require the resource fork to be stored as an ISO9660 *associated* file. This is just like any normal file stored in the ISO9660 filesystem except that the associated file flag is set in the Directory Record (bit 2). This file has the same name as the data fork (the file seen by non-Apple machines). Associated files are normally ignored by other OSs

When using the **−hfs** option, the TYPE and CREATOR plus other finder info, are stored in a separate HFS directory, not visible on the ISO9660 volume. The HFS directory references the same data and resource fork files described above.

In most cases, it is better to use the **−hfs** option instead of the **−apple** option, as the latter imposes the limited ISO9660 characters allowed in filenames. However, the Apple extensions do give the advantage that the files are packed on the disk more efficiently and it may be possible to fit more files on a CD - important when the total size of the source files is approaching 650MB.

HFS MACINTOSH FILE NAMES

Where possible, the HFS filename that is stored with an Apple/Unix file is used for the HFS part of the CD. However, not all the Apple/Unix encodings store the HFS filename with the finderinfo. In these cases, the Unix filename is used - with escaped special characters. Special characters include '/' and characters with codes over 127.

Aufs escapes these characters by using ":" followed by the character code as two hex digits. Netatalk and EtherShare have a similar scheme, but uses "%" instead of a ":".

If mkisofs can't find an HFS filename, then it uses the Unix name, with any %xx or :xx characters (xx == two hex digits) converted to a single character code. If "xx" are not hex digits ([0-9a-fA-F]), then they are left alone - although any remaining ":" is converted to "%" as colon is the HFS directory separator. Care must be taken, as an ordinary Unix file with %xx or :xx will also be converted. e.g.

This:2fFile converted to This/File

This:File converted to This%File

This:t7File converted to This%t7File

Although HFS filenames appear to support upper and lower case letters, the filesystem is case insensitive. i.e. the filenames "aBc" and "AbC" are the same. If a file is found in a directory with the same HFS name, then *mkisofs* will attempt, where possible, to make a unique name by adding '_' characters to one of the filenames.

If an HFS filename exists for a file, then mkisofs can use this name as the starting point for the ISO9660, Joliet and Rock Ridge filenames using the **−mac-name** option. Normal Unix files without an HFS name will still use their Unix name. e.g.

If a *MacBinary* (or *PC Exchange*) file is stored as *someimage.gif.bin* on the Unix filesystem, but contains a HFS file called *someimage.gif*, then this is the name that would appear on the HFS part of the CD. However, as mkisofs uses the Unix name as the starting point for the other names, then the ISO9660 name generated will probably be *SOMEIMAG.BIN* and the Joliet/Rock Ridge would be *someimage.gif.bin*. Although the actual data (in this case) is a GIF image. This option will use the HFS filename as the starting point and the ISO9660 name will probably be *SOMEIMAG.GIF* and the Joliet/Rock Ridge would be *someimage.gif*.

Using the **−mac-name** option will not currently work with the **−T** option - the Unix name will be used in the TRANS.TBL file, not the Macintosh name.

The character set used to convert any HFS file name to a Joliet/Rock Ridge file name defaults to *cp10000* (Mac Roman). The character set used can be specified using the *-input-hfs-charset* option. Other built in HFS character sets are: cp10006 (MacGreek), cp10007 (MacCyrillic), cp10029 (MacLatin2), cp10079

(MacIcelandandic) and cp10081 (MacTurkish).

Note: the character codes used by HFS file names taken from the various Apple/Unix formats will not be converted as they are assumed to be in the correct Apple character set. Only the Joliet/Rock Ridge names derived from the HFS file names will be converted.

The existing mkisofs code will filter out any illegal characters for the ISO9660 and Joliet filenames, but as mkisofs expects to be dealing directly with Unix names, it leaves the Rock Ridge names as is. But as '/' is a legal HFS filename character, the **−mac-name** option converts '/' to a '_' in Rock Ridge filenames.

If the Apple extensions are used, then only the ISO9660 filenames will appear on the Macintosh. However, as the Macintosh ISO9660 drivers can use *Level 2* filenames, then you can use options like **−allow-multidot** without problems on a Macintosh - still take care over the names, for example *this.file.name* will be converted to *THIS.FILE* i.e. only have one '.', also filename *abcdefgh* will be seen as *ABCDEFGH* but *abcdefghi* will be seen as *ABCDEFGHI.* i.e. with a '.' at the end - don't know if this is a Macintosh problem or mkisofs/mkhybrid problem. All filenames will be in uppercase when viewed on a Macintosh. Of course, DOS/Win3.X machines will not be able to see Level 2 filenames...

HFS CUSTOM VOLUME/FOLDER ICONS

To give a HFS CD a custom icon, make sure the root (top level) folder includes a standard Macintosh volume icon file. To give a volume a custom icon on a Macintosh, an icon has to be pasted over the volume's icon in the "Get Info" box of the volume. This creates an invisible file called 'Icon\r' ('\r' is the 'carriage return' character) in the root folder.

A custom folder icon is very similar - an invisible file called 'Icon\r' exits in the folder itself.

Probably the easiest way to create a custom icon that mkisofs can use, is to format a blank HFS floppy disk on a Mac, paste an icon to its "Get Info" box. If using Linux with the HFS module installed, mount the floppy using something like:

```
mount −t hfs /dev/fd0 /mnt/floppy
```

The floppy will be mounted as a CAP file system by default. Then run mkisofs using something like:

```
mkisofs −−cap −o output source_dir /mnt/floppy
```

If you are not using Linux, then you can use the hfsutils to copy the icon file from the floppy. However, care has to be taken, as the icon file contains a control character. e.g.

```
hmount /dev/fd0
hdir −a
hcopy −m Icon^V^M icon_dir/icon
```

Where '^V^M' is control-V followed by control-M. Then run **mkisofs** by using something like:

```
mkisofs −−macbin −o output source_dir icon_dir
```

The procedure for creating/using custom folder icons is very similar - paste an icon to folder's "Get Info" box and transfer the resulting 'Icon\r' file to the relevant directory in the mkisofs source tree.

You may want to hide the icon files from the ISO9660 and Joliet trees.

To give a custom icon to a Joliet CD, follow the instructions found at: http://www.fadden.com/cdrfaq/faq03.html#[3-21]

HFS BOOT DRIVER

It *may* be possible to make the hybrid CD bootable on a Macintosh.

A bootable HFS CD requires an Apple CD-ROM (or compatible) driver, a bootable HFS partition and the necessary System, Finder, etc. files.

A driver can be obtained from any other Macintosh bootable CD-ROM using the *apple_driver* utility. This file can then be used with the **−boot-hfs-file** option.

The HFS partition (i.e. the hybrid disk in our case) must contain a suitable System Folder, again from another CD-ROM or disk.

For a partition to be bootable, it must have it's *boot block* set. The boot block is in the first two blocks of a partition. For a non-bootable partition the boot block is full of zeros. Normally, when a System file is copied to partition on a Macintosh disk, the boot block is filled with a number of required settings - unfortunately I don't know the full spec for the boot block, so I'm guessing that the following will work OK.

Therefore, the utility *apple_driver* also extracts the boot block from the first HFS partition it finds on the given CD-ROM and this is used for the HFS partition created by **mkisofs**.

PLEASE NOTE

By using a driver from an Apple CD and copying Apple software to your CD, you become liable to obey Apple Computer, Inc. Software License Agreements.

EL TORITO BOOT INFORMATION TABLE

When the **−boot-info-table** option is given, **mkisofs** will modify the boot file specified by the **−b** option by inserting a 56-byte "boot information table" at offset 8 in the file. This modification is done in the source filesystem, so make sure you use a copy if this file is not easily recreated! This file contains pointers which may not be easily or reliably obtained at boot time.

The format of this table is as follows; all integers are in section 7.3.1 ("little endian") format.

Offset	Name	Size	Meaning
8	bi_pvd	4 bytes	LBA of primary volume descriptor
12	bi_file	4 bytes	LBA of boot file
16	bi_length	4 bytes	Boot file length in bytes
20	bi_csum	4 bytes	32-bit checksum
24	bi_reserved	40 bytes	Reserved

The 32-bit checksum is the sum of all the 32-bit words in the boot file starting at byte offset 64. All linear block addresses (LBAs) are given in CD sectors (normally 2048 bytes).

CONFIGURATION

mkisofs looks for the **.mkisofsrc** file, first in the current working directory, then in the user's home directory, and then in the directory in which the **mkisofs** binary is stored. This file is assumed to contain a series of lines of the form **TAG=***value,* and in this way you can specify certain options. The case of the tag is not significant. Some fields in the volume header are not settable on the command line, but can be altered through this facility. Comments may be placed in this file, using lines which start with a hash (#) character.

APPI The application identifier should describe the application that will be on the disc. There is space on the disc for 128 characters of information. May be overridden using the **−A** command line option.

COPY The copyright information, often the name of a file on the disc containing the copyright notice. There is space in the disc for 37 characters of information. May be overridden using the **−copyright** command line option.

ABST The abstract information, often the name of a file on the disc containing an abstract. There is space in the disc for 37 characters of information. May be overridden using the **−abstract** command line option.

BIBL The bibliographic information, often the name of a file on the disc containing a bibliography. There is space in the disc for 37 characters of information. May be overridden using the **−bilio** command line option.

PREP This should describe the preparer of the CDROM, usually with a mailing address and phone number. There is space on the disc for 128 characters of information. May be overridden using the **−p** command line option.

PUBL This should describe the publisher of the CDROM, usually with a mailing address and phone number. There is space on the disc for 128 characters of information. May be overridden using the **−P** command line option.

SYSI The System Identifier. There is space on the disc for 32 characters of information. May be overridden using the **−sysid** command line option.

VOLI The Volume Identifier. There is space on the disc for 32 characters of information. May be overridden using the **−V** command line option.

VOLS The Volume Set Name. There is space on the disc for 128 characters of information. May be overridden using the **−volset** command line option.

HFS_TYPE
The default TYPE for Macintosh files. Must be exactly 4 characters. May be overridden using the **−hfs-type** command line option.

HFS_CREATOR
The default CREATOR for Macintosh files. Must be exactly 4 characters. May be overridden using the **−hfs-creator** command line option.

mkisofs can also be configured at compile time with defaults for many of these fields. See the file defaults.h.

EXAMPLES

To create a vanilla ISO-9660 filesystem image in the file *cd.iso*, where the directory *cd_dir* will become the root directory if the CD, call:

% mkisofs −o cd.iso cd_dir

To create a CD with Rock Ridge extensions of the source directory *cd_dir*:

% mkisofs −o cd.iso −R cd_dir

To create a CD with Rock Ridge extensions of the source directory *cd_dir* where all files have at least read permission and all files are owned by *root*, call:

% mkisofs −o cd.iso −r cd_dir

To create a HFS hybrid CD with the Joliet and Rock Ridge extensions of the source directory *cd_dir*:

% mkisofs −o cd.iso −R −J −hfs cd_dir

To create a HFS hybrid CD from the source directory *cd_dir* that contains Netatalk Apple/Unix files:

% mkisofs −o cd.iso −−netatalk cd_dir

To create a HFS hybrid CD from the source directory *cd_dir*, giving all files CREATOR and TYPES based on just their filename extensions listed in the file "mapping".:

% mkisofs −o cd.iso −map mapping cd_dir

To create a CD with the ’Apple Extensions to ISO9660’, from the source directories *cd_dir* and *another_dir.* Files in all the known Apple/Unix format are decoded and any other files are given CREATOR and TYPE based on their magic number given in the file "magic":

```
% mkisofs −o cd.iso −apple −magic magic −probe \
    cd_dir another_dir
```

The following example puts different files on the CD that all have the name README, but have different contents when seen as a ISO9660/RockRidge, Joliet or HFS CD.

Current directory contains:

```
% ls −F
README.hfs     README.joliet  README.unix    cd_dir/
```

The following command puts the contents of the directory *cd_dir* on the CD along with the three README files - but only one will be seen from each of the three filesystems:

```
% mkisofs −o cd.iso −hfs −J −r −graft-points \
    −hide README.hfs −hide README.joliet \
    −hide-joliet README.hfs −hide-joliet README.unix \
    −hide-hfs README.joliet −hide-hfs README.unix \
    README=README.hfs README=README.joliet \
    README=README.unix cd_dir
```

i.e. the file README.hfs will be seen as README on the HFS CD and the other two README files will be hidden. Similarly for the Joliet and ISO9660/RockRidge CD.

There are probably all sorts of strange results possible with combinations of the hide options ...

AUTHOR

mkisofs is not based on the standard mk*fs tools for unix, because we must generate a complete copy of an existing filesystem on a disk in the iso9660 filesystem. The name mkisofs is probably a bit of a misnomer, since it not only creates the filesystem, but it also populates it as well. However, the appropriate tool name for a UNIX tool that creates populated filesystems - mkproto - is not well known.

Eric Youngdale <ericy@gnu.ai.mit.edu> or <eric@andante.org> wrote the first versions (1993 . . . 1998) of the mkisofs utility. The copyright for old versions of the mkisofs utility is held by Yggdrasil Computing, Incorporated. Jörg Schilling wrote the SCSI transport library and it's adaptation layer to **mkisofs** and newer parts (starting from 1999) of the utility, this makes **mkisofs** Copyright (C) 1999, 2000, 2001 Jörg Schilling.

HFS hybrid code Copyright (C) James Pearson 1997, 1998, 1999, 2000, 2001
libhfs code Copyright (C) 1996, 1997 Robert Leslie
libfile code Copyright (C) Ian F. Darwin 1986, 1987, 1989, 1990, 1991, 1992, 1994, 1995.

NOTES

Mkisofs may safely be installed suid root. This may be needed to allow **mkisofs** to read the previous session when creating a multi session image.

If **mkisofs** is creating a filesystem image with Rock Ridge attributes and the directory nesting level of the source directory tree is too much for ISO-9660, **mkisofs** will do deep directory relocation. This results in a directory called **RR_MOVED** in the root directory of the CD. You cannot avoid this directory.

BUGS

- Any files that have hard links to files not in the tree being copied to the iso9660 filesystem will have an incorrect file reference count.
- Does not check for SUSP record(s) in "." entry of the root directory to verify the existence of Rock Ridge enhancements.

  This problem is present when reading old sessions while adding data in multi-session mode.
- Does not properly read relocated directories in multi-session mode when adding data.

  Any relocated deep directory is lost if the new session does not include the deep directory.

  Repeat by: create first session with deep directory relocation then add new session with a single dir that differs from the old deep path.
- Does not re-use RR_MOVED when doing multi-session from TRANS.TBL
- Does not create whole_name entry for RR_MOVED in multi-session mode.

There may be some other ones. Please, report them to the author.

HFS PROBLEMS/LIMITATIONS

I have had to make several assumptions on how I expect the modified libhfs routines to work, however there may be situations that either I haven't thought of, or come across when these assumptions fail. Therefore I can't guarantee that mkisofs will work as expected (although I haven't had a major problem yet). Most of the HFS features work fine, however, some are not fully tested. These are marked as *Alpha* above.

Although HFS filenames appear to support upper and lower case letters, the filesystem is case insensitive. i.e. the filenames "aBc" and "AbC" are the same. If a file is found in a directory with the same HFS name, then *mkisofs* will attempt, where possible, to make a unique name by adding '_' characters to one of the filenames.

HFS file/directory names that share the first 31 characters have _N' (N == decimal number) substituted for the last few characters to generate unique names.

Care must be taken when "grafting" Apple/Unix files or directories (see above for the method and syntax involved). It is not possible to use a new name for an Apple/Unix encoded file/directory. e.g. If a Apple/Unix encoded file called "oldname" is to added to the CD, then you can not use the command line:

mkisofs −o output.raw −hfs −graft-points newname=oldname cd_dir

mkisofs will be unable to decode "oldname". However, you can graft Apple/Unix encoded files or directories as long as you do not attempt to give them new names as above.

When creating an HFS volume with the multisession options, **−M** and **−C**, only files in the last session will be in the HFS volume. i.e. mkisofs can not *add* existing files from previous sessions to the HFS volume.

However, if each session is created with the **−part** option, then each session will appear as separate volumes when mounted on a Mac. In this case, it is worth using the **−V** or **−hfs-volid** option to give each session a unique volume name, otherwise each "volume" will appear on the Desktop with the same name.

Symbolic links (as with all other non-regular files) are not added to the HFS directory.

Hybrid volumes may be larger than pure ISO9660 volumes containing the same data. In some cases (e.g. DVD sized volumes) the hybrid volume may be significantly larger. As an HFS volume gets bigger, so does the allocation block size (the smallest amount of space a file can occupy). For a 650Mb CD, the allocation block is 10Kb, for a 4.7Gb DVD it will be about 70Kb.

The maximum number of files in an HFS volume is about 65500 - although the real limit will be somewhat less than this.

The resulting hybrid volume can be accessed on a Unix machine by using the hfsutils routines. However, no changes can be made to the volume as it is set as **locked.** The option **−hfs-unlock** will create an output image that is unlocked - however no changes should be made to the contents of the volume (unless you really know what you are doing) as it's not a "real" HFS volume.

Using the **−mac-name** option will not currently work with the **−T** option - the Unix name will be used in the TRANS.TBL file, not the Macintosh name.

Although **mkisofs** does not alter the contents of a file, if a binary file has it's TYPE set as 'TEXT', it *may* be read incorrectly on a Macintosh. Therefore a better choice for the default TYPE may be '????'

The **−mac-boot-file** option may not work at all...

May not work with PC Exchange v2.2 or higher files (available with MacOS 8.1). DOS media containing PC Exchange files should be mounted as type **msdos** (not **vfat**) when using Linux.

The SFM format is only partially supported - see **HFS MACINTOSH FILE FORMATS** section above.

It is not possible to use the the **−sparc-boot** or **−generic-boot** options with the **−boot-hfs-file** or **−prep-boot** options.

mkisofs should be able to create HFS hybrid images over 4Gb, although this has not been fully tested.

SEE ALSO

cdrecord(1), **mkzftree**(1), **magic**(5), **apple_driver**(8).

FUTURE IMPROVEMENTS

Some sort of gui interface.

AVAILABILITY

mkisofs is available as part of the cdrecord package from ftp://ftp.berlios.de/pub/cdrecord/

hfsutils from ftp://ftp.mars.org/pub/hfs

mkzftree is available as part of the zisofs-tools package from ftp://ftp.kernel.org/pub/linux/utils/fs/zisofs/

MAILING LISTS

If you want to actively take part on the development of mkisofs, and/or mkhybrid, 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

MAINTAINER

Jörg Schilling
Seestr. 110
D-13353 Berlin
Germany

HFS MKHYBRID MAINTAINER

James Pearson

j.pearson@ge.ucl.ac.uk

If you have support questions, send them to:

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

Of you definitly 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**