Read/Write HPFS 2.09

1998-2004, Mikulas Patocka

email: mikulas@artax.karlin.mff.cuni.cz
homepage: https://artax.karlin.mff.cuni.cz/~mikulas/vyplody/hpfs/index-e.cgi

Credits

Chris Smith, 1993, original read-only HPFS, some code and hpfs structures file
is taken from it

Jacques Gelinas, MSDos mmap, Inspired by fs/nfs/mmap.c (Jon Tombs 15 Aug 1993)

Werner Almesberger, 1992, 1993, MSDos option parser & CR/LF conversion

Mount options

uid=xxx,gid=xxx,umask=xxx (default uid=gid=0 umask=default_system_umask)
: Set owner/group/mode for files that do not have it specified in extended attributes. Mode is inverted umask - for example umask 027 gives owner all permission, group read permission and anybody else no access. Note that for files mode is anded with 0666. If you want files to have 'x' rights, you must use extended attributes.

case=lower,asis (default asis)
: File name lowercasing in readdir.

conv=binary,text,auto (default binary)
: CR/LF -> LF conversion, if auto, decision is made according to extension - there is a list of text extensions (I thing it's better to not convert text file than to damage binary file). If you want to change that list, change it in the source. Original readonly HPFS contained some strange heuristic algorithm that I removed. I thing it's danger to let the computer decide whether file is text or binary. For example, DJGPP binaries contain small text message at the beginning and they could be misidentified and damaged under some circumstances.

check=none,normal,strict (default normal)
: Check level. Selecting none will cause only little speedup and big danger. I tried to write it so that it won't crash if check=normal on corrupted filesystems. check=strict means many superfluous checks - used for debugging (for example it checks if file is allocated in bitmaps when accessing it).

errors=continue,remount-ro,panic (default remount-ro)
: Behaviour when filesystem errors found.

chkdsk=no,errors,always (default errors)
: When to mark filesystem dirty so that OS/2 checks it.

eas=no,ro,rw (default rw)
: What to do with extended attributes. 'no' - ignore them and use always values specified in uid/gid/mode options. 'ro' - read extended attributes but do not create them. 'rw' - create extended attributes when you use chmod/chown/chgrp/mknod/ln -s on the filesystem.

timeshift=(-)nnn (default 0)
: Shifts the time by nnn seconds. For example, if you see under linux one hour more, than under os/2, use timeshift=-3600.

File names

As in OS/2, filenames are case insensitive. However, shell thinks that names are case sensitive, so for example when you create a file FOO, you can use 'cat FOO', 'cat Foo', 'cat foo' or 'cat F*' but not 'cat f*'. Note, that you also won't be able to compile linux kernel (and maybe other things) on HPFS because kernel creates different files with names like bootsect.S and bootsect.s. When searching for file thats name has characters >= 128, codepages are used - see below. OS/2 ignores dots and spaces at the end of file name, so this driver does as well. If you create 'a. ...', the file 'a' will be created, but you can still access it under names 'a.', 'a..', 'a . . . ' etc.

Extended attributes

On HPFS partitions, OS/2 can associate to each file a special information called extended attributes. Extended attributes are pairs of (key,value) where key is an ascii string identifying that attribute and value is any string of bytes of variable length. OS/2 stores window and icon positions and file types there. So why not use it for unix-specific info like file owner or access rights? This driver can do it. If you chown/chgrp/chmod on a hpfs partition, extended attributes with keys "UID", "GID" or "MODE" and 2-byte values are created. Only that extended attributes those value differs from defaults specified in mount options are created. Once created, the extended attributes are never deleted, they're just changed. It means that when your default uid=0 and you type something like 'chown luser file; chown root file' the file will contain extended attribute UID=0. And when you umount the fs and mount it again with uid=luser_uid, the file will be still owned by root! If you chmod file to 444, extended attribute "MODE" will not be set, this special case is done by setting read-only flag. When you mknod a block or char device, besides "MODE", the special 4-byte extended attribute "DEV" will be created containing the device number. Currently this driver cannot resize extended attributes - it means that if somebody (I don't know who?) has set "UID", "GID", "MODE" or "DEV" attributes with different sizes, they won't be rewritten and changing these

values doesn't work.

Symlinks

You can do symlinks on HPFS partition, symlinks are achieved by setting extended attribute named "SYMLINK" with symlink value. Like on ext2, you can chown and chgrp symlinks but I don't know what is it good for. chmoding symlink results in chmoding file where symlink points. These symlinks are just for Linux use and incompatible with OS/2. OS/2 PmShell symlinks are not supported because they are stored in very crazy way. They tried to do it so that link changes when file is moved ... sometimes it works. But the link is partly stored in directory extended attributes and partly in OS2SYS.INI. I don't want (and don't know how) to analyze or change OS2SYS.INI.

Codepages

HPFS can contain several uppercasing tables for several codepages and each file has a pointer to codepage its name is in. However OS/2 was created in America where people don't care much about codepages and so multiple codepages support is quite buggy. I have Czech OS/2 working in codepage 852 on my disk. Once I booted English OS/2 working in cp 850 and I created a file on my 852 partition. It marked file name codepage as 850 - good. But when I again booted Czech OS/2, the file was completely inaccessible under any name. It seems that OS/2 uppercases the search pattern with its system code page (852) and file name it's comparing to with its code page (850). These could never match. Is it really what IBM developers wanted? But problems continued. When I created in Czech OS/2 another file in that directory, that file was inaccessible too. OS/2 probably uses different uppercasing method when searching where to place a file (note, that files in HPFS directory must be sorted) and when searching for a file. Finally when I opened this directory in PmShell, PmShell crashed (the funny thing was that, when rebooted, PmShell tried to reopen this directory again :-). chkdsk happily ignores these errors and only low-level disk modification saved me. Never mix different language versions of OS/2 on one system although HPFS was designed to allow that. OK, I could implement complex codepage support to this driver but I think it would cause more problems than benefit with such buggy implementation in OS/2. So this driver simply uses first codepage it finds for uppercasing and lowercasing no matter what's file codepage index. Usually all file names are in this codepage - if you don't try to do what I described above :-)

Known bugs

HPFS386 on OS/2 server is not supported. HPFS386 installed on normal OS/2 client should work. If you have OS/2 server, use only read-only mode. I don't know how to handle some HPFS386 structures like access control list or extended perm list, I don't know how to delete them when file is deleted and how to not overwrite them with extended attributes. Send me some info on these structures and I'll make it. However, this driver should detect presence of HPFS386 structures, remount read-only and not destroy them (I hope).

When there's not enough space for extended attributes, they will be truncated and no error is returned.

OS/2 can't access files if the path is longer than about 256 chars but this driver allows you to do it. chkdsk ignores such errors.

Sometimes you won't be able to delete some files on a very full filesystem (returning error ENOSPC). That's because file in non-leaf node in directory tree (one directory, if it's large, has dirents in tree on HPFS) must be replaced with another node when deleted. And that new file might have larger name than the old one so the new name doesn't fit in directory node (dnode). And that would result in directory tree splitting, that takes disk space. Workaround is to delete other files that are leaf (probability that the file is non-leaf is about 1/50) or to truncate file first to make some space. You encounter this problem only if you have many directories so that preallocated directory band is full i.e.:

```
number_of_directories / size_of_filesystem_in_mb > 4.
```

You can't delete open directories.

You can't rename over directories (what is it good for?).

Renaming files so that only case changes doesn't work. This driver supports it but vfs doesn't. Something like 'mv file FILE' won't work.

All atimes and directory mtimes are not updated. That's because of performance reasons. If you extremely wish to update them, let me know, I'll write it (but it will be slow).

When the system is out of memory and swap, it may slightly corrupt filesystem (lost files, unbalanced directories). (I guess all filesystem may do it).

When compiled, you get warning: function declaration isn't a prototype. Does anybody know what does it mean?

What does "unbalanced tree" message mean?

Old versions of this driver created sometimes unbalanced dnode trees. OS/2 chkdsk doesn't scream if the tree is unbalanced (and sometimes creates unbalanced trees too :-) but both HPFS and HPFS386 contain bug that it rarely crashes when the tree is not balanced. This driver handles unbalanced trees correctly and writes warning if it finds them. If you see this message, this is probably because of directories created with old version of this driver. Workaround is to move all files from that directory to another and then

back again. Do it in Linux, not OS/2! If you see this message in directory that is whole created by this driver, it is BUG - let me know about it.

Bugs in OS/2

When you have two (or more) lost directories pointing each to other, chkdsk locks up when repairing filesystem.

Sometimes (I think it's random) when you create a file with one-char name under OS/2, OS/2 marks it as 'long'. chkdsk then removes this flag saying "Minor fs error corrected".

File names like "a .b" are marked as 'long' by OS/2 but chkdsk "corrects" it and marks them as short (and writes "minor fs error corrected"). This bug is not in HPFS386.

Codepage bugs described above

If you don't install fixpacks, there are many, many more...

History

0.90	First public release
0.91	Fixed bug that caused shooting to memory when write_inode was called on open inode (rarely happened)
0.92	Fixed a little memory leak in freeing directory inodes
0.93	Fixed bug that locked up the machine when there were too many filenames with first 15 characters same Fixed write_file to zero file when writing behind file end
0.94	Fixed a little memory leak when trying to delete busy file or directory
0.95	Fixed a bug that i_hpfs_parent_dir was not updated when moving files
1.90	First version for 2.1.1xx kernels
1.91	Fixed a bug that chk_sectors failed when sectors were at the end of disk Fixed a race-condition when write_inode is called while deleting file Fixed a bug that could possibly happen (with very low probability) when using 0xff in filenames. Rewritten locking to avoid race-conditions Mount option 'eas' now works Fsync no longer returns error Files beginning with '.' are marked hidden Remount support added Alloc is not so slow when filesystem becomes full Atimes are no more updated because it slows down operation Code cleanup (removed all commented debug prints)
1.92	Corrected a bug when sync was called just before closing file
1.93	Modified, so that it works with kernels >= 2.1.131, I don't know if it works with previous versions Fixed a possible problem with disks > 64G (but I don't have one, so I can't test it) Fixed a file overflow at 2G Added new option 'timeshift' Changed behaviour on HPFS386: It is now possible to operate on HPFS386 in read-only mode Fixed a bug that slowed down alloc and prevented allocating 100% space (this bug was not destructive)
1.94	Added workaround for one bug in Linux Fixed one buffer leak Fixed some incompatibilities with large extended attributes (but it's still not 100% ok, I have no info on it and OS/2 doesn't want to create them) Rewritten allocation Fixed a bug with i_blocks (du sometimes didn't display correct values) Directories have no longer archive attribute set (some programs don't like it) Fixed a bug that it set badly one flag in large anode tree (it was not destructive)
1.95	Fixed one buffer leak, that could happen on corrupted filesystem Fixed one bug in allocation in 1.94

1.96	Added workaround for one bug in OS/2 (HPFS locked up, HPFS386 reported error sometimes when opening directories in PMSHELL) Fixed a possible bitmap race Fixed possible problem on large disks You can now delete open files Fixed a nondestructive race in rename
1.97	Support for HPFS v3 (on large partitions) ZFixed a bug that it didn't allow creation of files > 128M (it should be 2G)
1.97.1	Changed names of global symbols Fixed a bug when chmoding or chowning root directory
1.98	Fixed a deadlock when using old_readdir Better directory handling; workaround for "unbalanced tree" bug in OS/2
1.99	Corrected a possible problem when there's not enough space while deleting file Now it tries to truncate the file if there's not enough space when deleting Removed a lot of redundant code
2.00	Fixed a bug in rename (it was there since 1.96) Better anti-fragmentation strategy
2.01	Fixed problem with directory listing over NFS Directory lseek now checks for proper parameters Fixed race-condition in buffer code - it is in all filesystems in Linux; when reading device (cat /dev/hda) while creating files on it, files could be damaged
2.02	Workaround for bug in breada in Linux. breada could cause accesses beyond end of partition
2.03	Char, block devices and pipes are correctly created Fixed non-crashing race in unlink (Alexander Viro) Now it works with Japanese version of OS/2
2.04	Fixed error when ftruncate used to extend file
2.05	Fixed crash when got mount parameters without = Fixed crash when allocation of anode failed due to full disk Fixed some crashes when block io or inode allocation failed
2.06	Fixed some crash on corrupted disk structures Better allocation strategy Reschedule points added so that it doesn't lock CPU long time It should work in read-only mode on Warp Server
2.07	More fixes for Warp Server. Now it really works
2.08	Creating new files is not so slow on large disks An attempt to sync deleted file does not generate filesystem error
2.09	Fixed error on extremely fragmented files