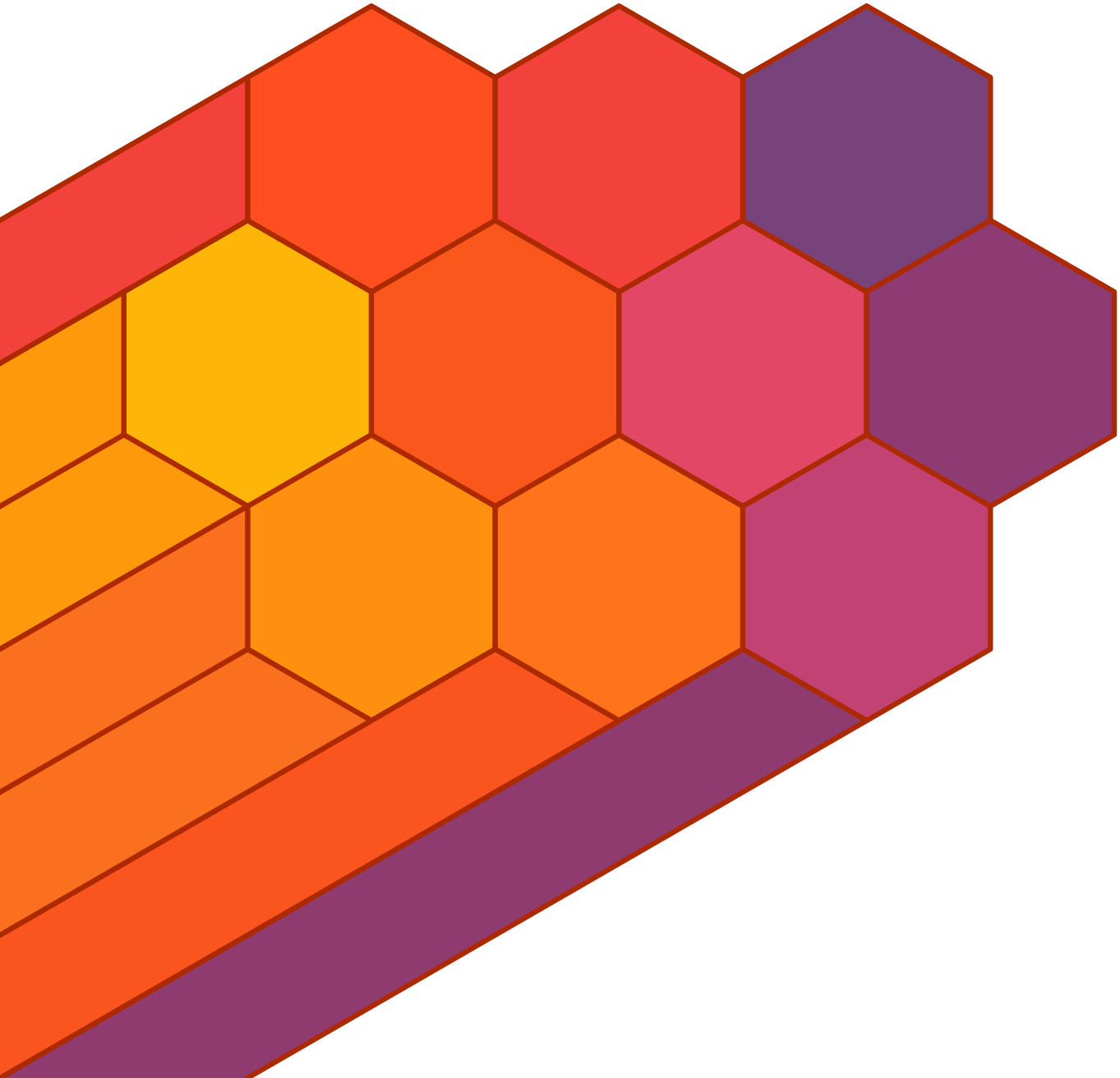
ucsd-psystem-fs
UCSD p-System Filesystem

Reference Manual

Peter Miller
pmiller@opensource.org.au



.

This document describes ucsd-psystem-fs version 1.22
and was prepared 21 July 2013.

Table of Contents(ucsd-psystem-fs)

Table of Contents(ucsd-psystem-fs)

Table of Contents(ucsd-psystem-fs)

NAME

ucsd-psystem-fs − UCSD p-System file system

DESCRIPTION

The *ucsd-psystem-fs* package is a collection of tools for manipulating and mounting UCSD p-System disk images.

ARCHIVE SITE

The latest version of *ucsd-psystem-fs* is available on the Web from:

URL:	http://ucsd-psystem-fs.sourceforge.net/	
File:	ucsd-psystem-fs-1.22.README	# Description, from the tar file
File:	ucsd-psystem-fs-1.22.lsm	# Description, LSM format
File:	ucsd-psystem-fs-1.22.tar.gz	# the complete source
File:	ucsd-psystem-fs-1.22.pdf	# Reference Manual

BUILDING ucsd-psystem-fs

Full instructions for building *ucsd-psystem-fs* may be found in the *BUILDING* file included in this distribution.

COPYRIGHT

ucsd-psystem-fs version 1.22

AUTHOR

Peter Miller	E-Mail:	pmiller@opensource.org.au
/\/*	WWW:	http://miller.emu.id.au/pmiller/

RELEASE NOTES

This section details the various features and bug fixes of the various releases.

Version 1.22 (2012-Jul-08)

- The web site was not updating the version number correctly, due to a missing build dependency rule.
- A false positive has been fixed in test 20.

Version 1.21 (2011-May-24)

• There is a new *ucsdpsys_disk*(1) −−**system−volume** option, which may be used to test whether or not a disk image looks like a system disk. It is intended for use by shell scripts.

Version 1.20 (2010-Oct-11)

- The *ucsdpsys_charset*(1) command has been moved to the ucsd-psystem-xc project.
- There is a new *ucsdpsys_disk*(1) −−**wipe−unused** option, used to reset unused blocks to zero.

Version 1.19 (2010-Sep-18)

- There is now a pre-built package in the LaunchPad PPA, and a link from the ucsd-psystem-fs web site.
- An Ubuntu Maverick build problem was fixed.
- Several test false negatives were foxed.

Version 1.18 (2010-Sep-12)

- The debian packaging Build-Depends was missing libboost-dev

Version 1.17 (2010-Sep-09)

- A build problem on i386 Linux has been fixed. The large file mechanism is fragile.

Version 1.16 (2010-Sep-08)

- A build problem on 32-bit Linux machines has been fixed, a necessary include file had been omitted in several places.
- All of the commands now ask libexplain for a four column hanging indent on each error message. This makes it easier to see when one ends and the next begins.

Version 1.15 (2010-Aug-24)

- The package now depends on libexplain (http://libexplain.sourceforge/net/) for its error messages and error handling.
- The *ucsdpsys_disk*(1) and *ucsdpsys_mkfs*(1) commands now understand how to write the boot blocks.

Version 1.14 (2010-Jun-22)

- The *ucsdpsys_disk*(1) command has a new −−**all** option, that can be used to request that dot-files also be transferred when a while directory is being added to a disk image. By default dot-files are ignored.
- The *ucsdpsys_mkfs*(1) command now defaults its disk size based on the −−**architecture** given.

Version 1.13 (2010-Jun-20)

- The *ucsdpsys_mkfs*(1) command has a new **−−architecture** option, that can be used to set the byte sex of the new file system. See *ucsdpsys_mkfs*(1) for more information.
- There is a new *ucsdpsys_rt11*(1) command, that may be used to extract files from an RT-11 disk image.

Version 1.12 (2010-Jun-18)

Some verbose, annoying, left-over debugging code has been removed from the sort-by-name directory listing.

Version 1.11 (2010-Jun-16)

- The *ucsdpsys_disk*(1) command now allows you to specify the sort criteria for the file listing. See the *ucsdpsys_disk*(1) man page for more information.
- The *ucsdpsys_disk*(1) command can now move all of the files to the beginning of the disk image, maximizing the space available for the UCSD p-System to work with. See the *ucsdpsys_disk*(1) man page for more information.

Version 1.9 (2010-May-30)

- It is now possible to specify a third interleave type on the *ucsdpsys_mkfs*(1) command line, see the man page for details.
- The file system now silently translates shash (/) characters in file names into underscore (_) characters. This was causing a very mysterious and uninformative error message to be reported.
- The *ucsdpsys_mkfs*(1) command's **−−label** option now converts the label to upper case before writing it to the disk image/

Version 1.8 (2010-Apr-09)

- The commands now all understand long option names.
- A bug has been fixed in the file-kind guessing code, it no longer tries to make SYSTEM.PASCAL a text file.
- It is now possible to add a size suffix when specifying the size to *ucsdpsys_mkfs*(1).

Version 1.7 (2010-Apr-03)

- The code that tries to divine the file type from the file's name, has been extended to understand more of the types of the system files.
- The `.imd` format code is now able to cope with broken sector maps.
- There is a new *ucsdpsys_interleave −Tguess* option, for guessing the interleave of a disk image.
- The code has been made more robust around volume header records with incorrect dnumfiles fields. This is now detected, and can be repaired with the *ucsdpsys_fsck*(1) command.

Version 1.6 (2008-Jan-20)

- There is a new *ucsdpsys_mount −t* option, which may be used to have text files converted text files between Unix and UCSD formats on-the-fly.
- The *ucsdpsys_umount*(1) command now exits with a non-zero exit status if it runs out of retries. It also emits a comforting message if it succeeds after more than one attempt.
- The FUSE behaviour has changed slightly, and it is now sometimes necessary to retry umount attempts is within (about) a second of the last access. We use *usleep*(2) if available to minimize the time spent sleeping.

Version 1.5 (2008-Jan-10)

- The text file decoding can now cope with text block padding in the first block.
- The Teledisk TD0 format is now understood for reading. See http://www.classiccmp.org/dunfield/img/td0notes.txt for a description.

Version 1.4 (2007-Sep-10)

- The IMageDisk (IMD) format is now supported for reading.
- The license has been changed to GNU GPL version 3.

Version 1.3 (2007-Apr-04)

- A couple of build problems have been fixed.
- Files ending in `.pas` are now considered text files.
- The file names in the system are now converted to upper case automatically. This is bacuse the p-System performs case *sensitive* file name comparisons some of the time, and case *in*sensitive at other times. The only way to work consistently is to always convert the names of new files to upper case.
- The *ucsdpsys_disk*(1) command now automagically converts text files during put (**−p**) and get (**−p**) operations unless specifically requested to berform binary transfers (**−B**).
- The *ucsdpsys_disk*(1) command now preserves the modification date (as far as is possible) across gets (**−g**) and puts (**−p**).
- The *ucsdpsys_disk*(1) command is now able to get and put whole directories from and to UCSD p-System disk images.

Version 1.2 (2006-Apr-16)

- A bug has been fixed in the *ucsdpsys_text*(1) program. It no longer inserts tab characters in the text of a line when a tab is in the Unix file at that position − space characters are inserted instead. A warning is issued for all other cases of non-printing characters.
- A bug has been fixed in the date-last-modified file meta-data. The bit layout being used for dates was wrong.
- The *ucsdpsys_disk*(1) command now understands how to manipulate disk files which have differing Unix and p-System file names.
- The *ucsdpsys_disk*(1) directory listings now resemble the original more closely.
- The *write*(2) handling in the file system has been improved. It now uses the existing gap beyond the current file if that will serve immediate needs. This saves expensive block shuufling to make a gap we didn't need. This situation can arrise, for example, as a result of an open with O_TRUNC specified.

Version 1.1 (2006-Apr-11)

- The *ucsdpsys_mount*(1) is used to mount a UCSD p-System filesystem disk image as a Linux file system. See *ucsdpsys_mount*(1) for more information.
- The *ucsdpsys_umount*(1) is used to unmount a filesystem mounted by the *ucsdpsys_mount*(1) command. See *ucsdpsys_umount*(1) for more information.
- The *ucsdpsys_mkfs*(1) is used to create a new empty UCSD p-System filesystem disk image. See *ucsdpsys_mkfs*(1) for more information.
- The *ucsdpsys_fsck*(1) is used to verify and repair a UCSD p-System filesystem disk image. See *ucsdpsys_fsck*(1) for more information.
- The *ucsdpsys_disk*(1) is used to list, extract, insert and remove files from a UCSD p-System filesystem disk image, without mounting it. See *ucsdpsys_disk*(1) for more information.
- The *ucsdpsys_text*(1) is used to convert text files from the UCSD p-System format to a Unix text file, and back again. See *ucsdpsys_text*(1) for more information.

NAME

How to build ucsd-psystem-fs

BEFORE YOU START

There are a few pieces of software you may want to fetch and install before you proceed with your installation of ucsd-psystem-fs.

FUSE
: The *ucsd-psystem-fs* package depends on the FUSE (file system in user space) package. If it is not available on your system, ucsd-psystem-fs will not work on your system. At the moment, that means Linux, BSD, MacOS X, Hurd and OpenSolaris only.
http://fuse.sourceforge.net/

Boost
: The *ucsd-psystem-fs* package depends on the Boost C++ library.
http://boost.org/

libexplain
: The *ucsd-psystem-fs* package depends on libexplain (>= 0.33), a library of system-call-specific strerror replacements, for most of its error messages.
http://libexplain.sourceforge.net/

GNU Groff
: The documentation for the *ucsd-psystem-fs* package was prepared using the GNU Groff package (version 1.14 or later). This distribution includes full documentation, which may be processed into PostScript or DVI files at install time − if GNU Groff has been installed.

SITE CONFIGURATION

The **ucsd-psystem-fs** package is configured using the *configure* program included in this distribution.

The *configure* shell script attempts to guess correct values for various system-dependent variables used during compilation, and creates the *Makefile* and *lib/config.h* files. It also creates a shell script *config.status* that you can run in the future to recreate the current configuration.

Normally, you just *cd* to the directory containing *ucsd-psystem-fs*'s source code and then type

```
% ./configure
...lots of output...
%
```

Running *configure* takes a minute or two. While it is running, it prints some messages that tell what it is doing. If you don't want to see the messages, run *configure* using the quiet option; for example,

```
% ./configure --quiet
%
```

To compile the **ucsd-psystem-fs** package in a different directory from the one containing the source code, you must use a version of *make* that supports the *VPATH* variable, such as *GNU make*. Change directory to the directory where you want the object files and executables to go and run the *configure* script. The *configure* script automatically checks for the source code in the directory that *configure* is in and in .. (the parent directory). If for some reason *configure* is not in the source code directory that you are configuring, then it will report that it can't find the source code. In that case, run *configure* with the option `--srcdir=`*DIR*, where *DIR* is the directory that contains the source code.

By default, *configure* will arrange for the *make install* command to install the **ucsd-psystem-fs** package's files in */usr/local/bin*, and */usr/local/man*. There are options which allow you to control the placement of these files.

`--prefix=`*PATH*
: This specifies the path prefix to be used in the installation. Defaults to */usr/local* unless otherwise specified.

`--exec-prefix=`*PATH*
: You can specify separate installation prefixes for architecture-specific files files. Defaults to *${prefix}* unless otherwise specified.

`--bindir=`*PATH*
: This directory contains executable programs. On a network, this directory may be shared between machines with identical hardware and operating systems; it may be mounted read-only. Defaults to *${exec_prefix}/bin* unless otherwise specified.

`--mandir=`*PATH*
: This directory contains the on-line manual entries. On a network, this directory may be shared between all machines; it may be mounted read-only. Defaults to *${prefix}/man* unless otherwise specified.

The *configure* script ignores most other arguments that you give it; use the `--help` option for a complete list.

On systems that require unusual options for compilation or linking that the *ucsd-psystem-fs* package's *configure* script does not know about, you can give *configure* initial values for variables by setting them in the environment. In Bourne-compatible shells, you can do that on the command line like this:

```
$ CXX='g++ -traditional' LIBS=-lposix ./configure
...lots of output...
$
```

Here are the *make* variables that you might want to override with environment variables when running *configure*.

Variable: CXX
: C++ compiler program. The default is *c++*.

Variable: CPPFLAGS
: Preprocessor flags, commonly defines and include search paths. Defaults to empty. It is common to use `CPPFLAGS=-I/usr/local/include` to access other installed packages.

Variable: INSTALL
: Program to use to install files. The default is *install* if you have it, *cp* otherwise.

Variable: LIBS
: Libraries to link with, in the form `-l`*foo* `-l`*bar*. The *configure* script will append to this, rather than replace it. It is common to use `LIBS=-L/usr/local/lib` to access other installed packages.

If you need to do unusual things to compile the package, the author encourages you to figure out how *configure* could check whether to do them, and mail diffs or instructions to the author so that they can be included in the next release.

BUILDING UCSD-PSYSTEM-FS

All you should need to do is use the

```
% make
...lots of output...
%
```

command and wait. When this finishes you should see a directory called *bin* containing several programs.

If you have GNU Groff installed, the build will also create a *etc/reference.ps* file. This contains the README file, this BUILDING file, and all of the man pages.

You can remove the program binaries and object files from the source directory by using the

```
% make clean
...lots of output...
%
```

command. To remove all of the above files, and also remove the *Makefile* and *lib/config.h* and *config.status* files, use the

```
% make distclean
...lots of output...
%
```

command.

The file *etc/configure.in* is used to create *configure* by a GNU program called *autoconf*. You only need to know this if you want to regenerate *configure* using a newer version of *autoconf*.

TESTING UCSD-PSYSTEM-FS

The *ucsd-psystem-fs* package comes with a test suite. To run this test suite, use the command

```
% make sure
...lots of output...
Passed All Tests
%
```

The tests take a few seconds each, with a few very fast, and a couple very slow, but it varies greatly depending on your CPU.

If all went well, the message

```
Passed All Tests
```

should appear at the end of the make.

If a test fails, make will stop. The make **-k** tells it to keep going.

If you see an error message like “Software caused connection abort” or “Transport endpoint is not connected”, it may indicate that your FUSE kernel module and your FUSE user-space library have a version mismatch. Or it could be a bug in *ucsdpsys_mount*(1).

INSTALLING UCSD-PSYSTEM-FS

As explained in the *SITE CONFIGURATION* section, above, the *ucsd-psystem-fs* package is installed under the */usr/local* tree by default. Use the `--prefix=`*PATH* option to *configure* if you want some other path. More specific installation locations are assignable, use the `--help` option to *configure* for details.

All that is required to install the *ucsd-psystem-fs* package is to use the

```
% make install
...lots of output...
%
```

command. Control of the directories used may be found in the first few lines of the *Makefile* file and the other files written by the *configure* script; it is best to reconfigure using the *configure* script, rather than attempting to do this by hand.

GETTING HELP

If you need assistance with the *ucsd-psystem-fs* package, please do not hesitate to contact the author at

```
Peter Miller <pmiller@opensource.org.au>
```

Any and all feedback is welcome.

When reporting problems, please include the version number given by the

```
% ucsdpsys_mount -V
ucsdpsys_mount version 1.22.D005
...warranty disclaimer...
%
```

command. Please do not send this example; run the program for the exact version number.

COPYRIGHT

ucsd-psystem-fs version 1.22

AUTHOR

Peter Miller E-Mail: pmiller@opensource.org.au
/\/* WWW: http://miller.emu.id.au/pmiller/

NAME

ucsdpsys_disk − manipulate files on a UCSD p-System filesystem image

SYNOPSIS

ucsdpsys_disk −f *disk-image* **−l**
ucsdpsys_disk −f *disk-image* **−g** *files-to-get...*
ucsdpsys_disk −f *disk-image* **−p** *files-to-put...*
ucsdpsys_disk −f *disk-image* **−r** *files-to-remove...*
ucsdpsys_disk −f *disk-image* **−k**
ucsdpsys_disk −f *disk-image* **−−system−volume**
ucsdpsys_disk −V

DESCRIPTION

The *ucsdpsys_disk* program is used to manipulate the contents of a UCSD p-System filesystem disk image.

Features include

- Get single files from the disk image, with automatic text file translation.
- Put single files into the disk image, with automatic text file translation.
- Get all files from a disk image into a directory, with a single command, with automatic text file translation.
- Put all files from a directory into a disk image, with a single command, with automatic text file translation.
- Remove files from a disk image.
- You can crunch a disk image; that is, you can move all of the files as close to the start of the disk image as possible. (Also known as **−−squeeze** or **−−defragment**.)
- List all of the files in a disk image. You can select the sort criterion.

This program understands a variety of sector interleave patterns, and detects them automatically. It also automatically recongises ImageDisk (`.imd`) and Teledisk (`.td0`) files, but access is read-only.

OPTIONS

The following options are understood:

−A

−−all By default, files with names that start with a dot (“`.`”) are ignored when you **−−put** a whole directory. This option says to include files with names that start with a dot.

−B

−−all-binary
This option requests that all file transfers be binary, without text encoding or decoding. If you happen to transfer a text file this way, you can use the *ucsdpsys_text*(1) command to translate the text files afterwards.

−b *filename*

−−boot=*filename*
This option may be used to obtain the boot blocks (with the **−−get** option) or set the boot blocks (with the **−−put** option). The named file is expected to be raw binary (exactly 1 KiB).

−D

−−debug
Increase debug level. Only of interest to developers.

−f *filename*

−−file=*filename*
: The name of the file containing the UCSD p-System filesystem disk image.

−g *filename...*

−−get *filename...*
: Get the named files from the disk image and write them to Unix, using the same file name. Naming a directory will result in the whole directory being transferred. Note that text file formats will *not* be translated.

−k

−−crunch
: Move all of the files as far towards the start of the disk image as possible. This will maximize the space available for writing by the (much dumber) UCSD p-System. You can use this option in combination with the **−−put** or **−−remove** options. It is common to combine this option with the **−−wipe-unused** option, see below.

−l

−−list
: Obtain a listing of the volume's files. (Used twice, it will print the block numbers as well.)

  By default, files are sorted by start block (the order they appear in the disk image). To sort by a different criterion, use the **−−sort** option; see below.

−p *filename...*

−−put *filename...*
: Put the named files into the disk image, reading from the Unix file of the same name. Naming a directory will result in the whole directory being transferred. Note that text file formats will *not* be translated.

−r *filename...*

−−remove=*filename...*
: Remove the named files from the filesystem image.

  Actually, this just removes the directory entry. To completely erase the file contents as well, use the **−−wipe-unused** option; see below.

−S

−−system−volume
: This option may be used to test whether or not a disk image can be considered a system volume, by testing for the presence of the "`SYSTEM.COMPILER`", "`SYSTEM.EDITOR`", "`SYSTEM.FILER`" and "`SYSTEM.PASCAL`" files. If all are present, it is probably a system volume.

  This option prints no output. It is silent, because it is intended for use in shell scripts and the like. The results are in the exit status: EXIT_SUCCESS (zero) if it is a system volume, or EXIT_FAILURE (non-zero) if it is not.

−s *name*

−−sort=*name*
: This option may be used to change the criterion by which the files are sorted in a directory listing.

  block
  : Sort the files according to their initial block number, the order in which they appear on the disk. This is the default, for compatibility with the original UCSD system.

  name
  : Sort the directory entries by the file name.

date
: Sort the directory entries by the date last modified, or by name if that is not sufficient.

kind
: Sort the directory entries by the kind of file they are, or by name if that is not sufficient.

Any other sort name is an error.

−t

−−auto-text
: Convert text files between Unix and UCSD text formats automatically, in the same way as the *ucsdpsys_mount*(1) command.

−V

−−version
: Print the version of the *ucsdpsys_disk* program being executed.

−w

−−wipe-unused
: This option may be used to make sure that all blocks not accounted for in the directory are reset to zero, wiping any “left over” content. Not only is this more secure (things you didn’t intent to stay on the disk don’t) but the disk images compress better, too. The boot blocks are unaffected.

  If there are any files that are not exact multiples of 512 bytes long, the unused portions of their last blocks are also reset to zero. Code files and text files are always multiples of 512 bytes long, but other data files can have short last blocks.

  When combined with other disk-altering options, this option is the last applied to the disk image. This is useful when combined with the **−−crunch** option.

All other options will produce a diagnostic error.

EXIT STATUS

The *ucsdpsys_disk* command will exit with a status of 1 on any error. The *ucsdpsys_disk* command will only exit with a status of 0 if there are no errors.

SEE ALSO

ucsdpsys_mkfs(1)
: create a new UCSD p-System disk image

ucsdpsys_mount(1)
: mount a UCSD p-System disk image as a Linux file system

ucsdpsys_text(1)
: translate to and from the UCSD p-System text file format

COPYRIGHT

AUTHOR

Peter Miller E-Mail: pmiller@opensource.org.au
/\/* WWW: http://miller.emu.id.au/pmiller/

NAME

ucsdpsys_fsck − verify and repair UCSD p-System filesystem images

SYNOPSIS

ucsdpsys_fsck [*option*...] *disk-image*
ucsdpsys_fsck −V

DESCRIPTION

The *ucsdpsys_fsck* program is used to verify and repair UCSD p-System filesystem disk images.

OPTIONS

The following options are understood:

−D

−debug Inclease debug level. Probably only of interest to file system developers.

−f

−−fix This option causes the file system to be fixed, without this the file system will be checked but not repaired.

−r

−−read-only
Open the disk image read-only. It will be checked but not repaired.

−V

−−version
Print the version of the *ucsdpsys_fsck* program being executed.

All other options will produce a diagnostic error.

EXIT STATUS

The *ucsdpsys_fsck* command will exit with a status of 1 on any error. The *ucsdpsys_fsck* command will only exit with a status of 0 if there are no errors.

SEE ALSO

ucsdpsys_disk(1)
manipulate a disk image

ucsdpsys_mount(1)
mount a disk image

COPYRIGHT

ucsdpsys_fsck version 1.22

AUTHOR

Peter Miller E-Mail: pmiller@opensource.org.au
/\/* WWW: http://miller.emu.id.au/pmiller/

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program -- to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work’s System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation’s users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product,

and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party’s predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing

the Program or any portion of it.

11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor’s “contributor version”.

A contributor’s “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor’s essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient’s use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to

infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF

SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

> *one line to give the program’s name and a brief idea of what it does.*
> Copyright (C) *year name of author*
>
> This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.
>
> This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
>
> You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

> <program> Copyright (C) <year> <name of author>
> This program comes with ABSOLUTELY NO WARRANTY; for details type “show w”. This is free software, and you are welcome to redistribute it under certain conditions; type “show c” for details.

The hypothetical commands “show w” and “show c” should show the appropriate parts of the General Public License. Of course, your program’s commands might be different; for a GUI interface, you would use an “about box”.

You should also get your employer (if you work as a programmer) or school, if any, to sign a “copyright disclaimer” for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>.

NAME

ucsdpsys_interleave − decode interleaved UCSD p-System filesystem image

SYNOPSIS

ucsdpsys_interleave −d −T*name infile outfile*
ucsdpsys_interleave −e −T*name infile outfile*
ucsdpsys_interleave −V

DESCRIPTION

The *ucsdpsys_interleave* program is used to read a UCSD p-System filesystem image and decode it into a new uninterleaved filesystem image file. It is also possible to do the reverse.

OPTIONS

The following options are understood:

−D

−−debug
Increase debug level. Only of interest to developers.

−d

−−decode
read a UCSD p-System filesystem image and decode it into a new **un**interleaved filesystem image file.

−e

−−encode
read a UCSD p-System filesystem image and encode it into a new interleaved filesystem image file.

−T *name*

−−type=*name*
This option is used to specify the type of interleaving in question. Known formats are:

apple The symmetric interleaving used by the Apple][Pascal system.

pdp The offset and asymmetric interleaved format used by the PDP11 (?) system.

guess For decode *only*, it is possible to have the *ucsdpsys_interleave* command attempt to guess the interleaving, using the same method as the *ucsdpsys_mount*(1) and *ucsdpsys_disk*(1) programs. This is particularly useful for decoding weird disk images into uninterleaved raw disk images. This only works if the disk image is of a UCSD p-System disk, it can't guess the interleave for any other type of disk image.

−V

−−version
Print the version of the *ucsdpsys_interleave* program being executed.

All other options will produce a diagnostic error.

EXIT STATUS

The *ucsdpsys_interleave* command will exit with a status of 1 on any error. The *ucsdpsys_interleave* command will only exit with a status of 0 if there are no errors.

COPYRIGHT

ucsdpsys_interleave version 1.22

command.

AUTHOR

Peter Miller E-Mail: pmiller@opensource.org.au
/\/* WWW: http://miller.emu.id.au/pmiller/

NAME

ucsdpsys_mkfs − create new UCSD p-System filesystem disk images

SYNOPSIS

ucsdpsys_mkfs [*option*...] *filename*
ucsdpsys_mkfs −V

DESCRIPTION

The *ucsdpsys_mkfs* program is used to create new UCSD p-System disk image files.

While the *ucsdpsys_disk*(1) command understands IMD and TD0 disk image formats for reading, it is not possible at this time to for *ucsdpsys_mkfs*(1) to create or *ucsdpsys_disk*(1) to modify these disk image formats.

OPTIONS

The following options are understood:

−A *name*

−−architecture=*name*
: This option is used to specify the microprocessor of the host system. This, in turn, implies the byte sex of the file system to be created. The usual names are accepted, *e.g.* “pdp11”, “6502”, “6800”, *etc*

−B *number*

−−size=*number*
: This option may be used to specify the size of the disk image, in kilobytes (KB = 1024 bytes). Defaults to 140 if not specified, the size of an Apple Pascal floppy.

  Other common sizes are 800KB for 5.25" disks, and 1440KB for PC 3.5" floppies.

  You may specify B, K and M suffixes to indicate bytes, kilobytes (2**10) and megabytes (2**20).

  The maximum possible addressable size is 16MB (however, the limit of 77 files, and no subdirectories, makes it unlikely that a 16MB disk image would be useful). It is an error to specify a size greater than 16MB. The size must be a multiple of 4KB.

−b *filename*

−−boot=*filename*
: This option may be used to set the boot blocks. The named file is expected to be raw binary (1KiB).

−D

−−debug
: Increase debugging level. Only of interest to developers.

−i This is a synonym for `−−interleave=apple`

−−interleave=*name*
: Interleave the sectors in the style given. Known interleave patterns include:

  apple
  : The interleave pattern that Apple][Pascal used.

  pdp
  : The interleave pattern that PDP-11 used.

  none
  : No interleave. This is the default.

help
Print a list of known interleave pattern names

Name are case-**in**sensitive. All other names will result in a diagnostic error being issued.

−L *string*

−−label=*string*
This option may be used to specify the name of the volume. Defaults to something random, but probably unique, starting with "V". There is a size limit of 7 characters, the label will be truncated if it is longer than this.

−t

−−twin This option may be used to ask for two copies of the directory meta-data to be stored, not just one. This will not be used by *ucsdpsys_fsck*(1), but it will be kept up-to-date by all of the disk access methods.

−V

−−version
Print the version of the *ucsdpsys_mkfs* program being executed.

All other options will produce a diagnostic error.

EXIT STATUS

The *ucsdpsys_mkfs* command will exit with a status of 1 on any error. The *ucsdpsys_mkfs* command will only exit with a status of 0 if there are no errors.

SEE ALSO

ucsdpsys_fsck(1)
check a disk image

ucsdpsys_disk(1)
Manipulate a disk image. Is able to get, put and remove individual files, or whole directories.

ucsdpsys_mount(1)
mount a disk image

COPYRIGHT

ucsdpsys_mkfs version 1.22

AUTHOR

Peter Miller E-Mail: pmiller@opensource.org.au
/\/* WWW: http://miller.emu.id.au/pmiller/

NAME

ucsdpsys_mount − mount a UCSD p-System filesystem

SYNOPSIS

ucsdpsys_mount [*option*...] *filename directory*
ucsdpsys_mount −V

DESCRIPTION

All files accessible in a Unix system are arranged in one big tree, the file hierarchy, rooted at /. These files can be spread out over several devices. The *ucsdpsys_mount* command serves to attach a UCSD p-System disk image to the big file tree.

Disk Formats

At present, only the Apple][Pascal disk format is understood for reading and writing, however it is simple to add more formats in future.

This program understands a variety of sector interleave patterns, and detects them automatically. It also automatically recongises ImageDisk (`.imd`) and Teledisk (`.td0`) files, but access is read-only.

Umount When Finish

To umount the file system when you are done with it, use the

`ucsdpsys_umount` *directory*

command.

Concurrent Writes

The original p-System had difficulty writing to more than on file at time. This file system uses the Buffer Gap algorithm (a common implementation for text editors) to establish a gap for write to be performed within, compacting file automatically when necessary. While you have a single file open for writing, this is very efficient.

If you have two files open for writing, this file system can cope, but the constant block shuffling to obtain gaps in which to write two (or more) file simultaneously will affect performance.

OPTIONS

The following options are understood:

−D

−−debug
Turn on internal debugging. Specifying this option more than once increases the verbosity.

−d

−−fuse−debug
Turn on FUSE (libfuse) debugging. Only interesting to *ucsdpsys_mount*(1) developers. Implies the **−f** options.

−f

−−foreground
Execute the filesystem in the foreground. Usually a daemon process is spawned, and the *ucsdpsys_mount*(1) command returns immediately.

−o *string*

−−options=*string*
One or *mount*(1) options, separated by commas. This option may be given more than once.

−r

−−read−only
Mount the file system read-only.

−t

−−text Convert text files between Unix and UCSD formats on-the-fly.

−V

−−version
Print the version of the *ucsdpsys_mount* program being executed.

All other options will produce a diagnostic error.

EXIT STATUS

The *ucsdpsys_mount* command will exit with a status of 1 on any error. The *ucsdpsys_mount* command will only exit with a status of 0 if there are no errors.

SEE ALSO

fusermount(1)
mount FUSE file systems

ucsdpsys_fsck(1)
Check the integrity of UCSD p-System filesystem disk images.

ucsdpsys_mkfs(1)
create new empty UCSD p-System filesystem disk images.

ucsdpsys_umount(1)
unmount UCSD p-System filesystems

COPYRIGHT

ucsdpsys_mount version 1.22

AUTHOR

Peter Miller E-Mail: pmiller@opensource.org.au
/\/* WWW: http://miller.emu.id.au/pmiller/

NAME

ucsdpsys_rt11 − extract files from RT-11 disk images

SYNOPSIS

ucsdpsys_rt11 −f *disk-image* −g *filename*
ucsdpsys_rt11 −f *disk-image* −l
ucsdpsys_rt11 −−version

DESCRIPTION

The *ucsdpsys_rt11* program is used to extract fiels from RT-11 disk images.

OPTIONS

The following options are understood:

−f *filename*

−−file=*filename*
: The name of the file containing the disk image (usually coms as a pair of `.dir` and `.files` files, name either half).

−g *filename*

−−get=*filename*
: This option may be used to extract a file from the disk image. The filename is case-**in**sensitive; the file in the disk image is always upper-case, the Unix file it writes is always lower-case. When possible, the extracted file's modification time will be set to match the date stamp in the disk image.

−l

−−list This option may be used to request a directory listing.

−V

−−version
: Print the version of the *ucsdpsys_rt11* program being executed.

−x

−−extract-all
: Extract all of the files into the currect directory. All extracted file's names with be lower-case.

All other options will produce a diagnostic error.

EXIT STATUS

The *ucsdpsys_rt11* command will exit with a status of 1 on any error. The *ucsdpsys_rt11* command will only exit with a status of 0 if there are no errors.

COPYRIGHT

AUTHOR

Peter Miller E-Mail: pmiller@opensource.org.au
`/\/\*` WWW: http://miller.emu.id.au/pmiller/

NAME

ucsdpsys_text − translate UCSD p-System text files

SYNOPSIS

ucsdpsys_text −d [−*option*...][*filename*...]
ucsdpsys_text −e [−*option*...][*filename*...]
ucsdpsys_text −V

DESCRIPTION

The *ucsdpsys_text* program is used to translate UCSD p-System text files to and from Unix text files.

If no files are named on the command line, the standard input will be translated and written on the standard output.

Any files named on the command line will be translated *in situ*. A temporary output file will be in the same directory as each file being translated; the file system will need enough extra space to be able to hold the temporary files, until they are moved back over the input files. No backup copies of the inputs are kept.

The results are undefined if you attempt to use this command on binary data files.

OPTIONS

The following options are understood:

−d

−−decode
: This option is used to translate files from UCSD p-System text format to Unix text format.

−e

−−encode
: This option is used to translate files from Unix text format to UCSD p-System text format.

−N

−−nul
: There is a bug in the UCSD compiler. It will report error 400 (invalid character on line) if a line of text ends *exactly* at the end of the block.

  Interestingly, the editor is capable of reading text files with completely filled 1KB blocks, but never writes them out.

  By default, the encode (**−e**) option guarantees that there will be at least one NUL (0x00) character at the end of each 1KB block. The **−N** option says not to bother.

−t

−−tabs
: When decoding the default is to use tabs to replace leading spaces on a line (tabs are assumed to be 8 characters wide). When used with **−d**, this option says not to use tabs.

  When encoding the default is to use two byte sequences (0x10 *nn*) to replace leading spaces on a line. When used with **−e**, this option says not to use tabs.

−V

−−version
: Print the version of the *ucsdpsys_text* program being executed.

All other options will produce a diagnostic error.

EXIT STATUS

The *ucsdpsys_text* command will exit with a status of 1 on any error. The *ucsdpsys_text* command will only exit with a status of 0 if there are no errors.

SEE ALSO

ucsdpsys_disk(1)
: manipulate UCSD p-System disk images

ucsdpsys_text(5)
: format of UCSD p-System text files

COPYRIGHT

AUTHOR

Peter Miller E-Mail: pmiller@opensource.org.au
/\/* WWW: http://miller.emu.id.au/pmiller/

NAME

ucsdpsys_umount − unmount UCSD p-System filesystems

SYNOPSIS

ucsdpsys_umount *mount-point*
ucsdpsys_umount −V

DESCRIPTION

The *ucsdpsys_umount* program is used to unmount a file system mounted by the *ucsdpsys_mount*(1) command.

OPTIONS

The following options are understood:

−V

−−version
Print the version of the *ucsdpsys_umount* program being executed.

All other options will produce a diagnostic error.

EXIT STATUS

The *ucsdpsys_umount* command will exit with a status of 1 on any error. The *ucsdpsys_umount* command will only exit with a status of 0 if there are no errors.

SEE ALSO

fusermount(1)
umount FUSE file systems

ucsdpsys_mount(1)
mount UCSD p-System filesystem disk images.

COPYRIGHT

ucsdpsys_umount version 1.22

AUTHOR

Peter Miller E-Mail: pmiller@opensource.org.au
/\/* WWW: http://miller.emu.id.au/pmiller/

NAME

ucsdpsys_fs − UCSD p-System filesystem format

DESCRIPTION

The UCSD-pSystem file system format is very simple, compared to modern file systems. It consists of a single directory, containing at most 77 files. The file system does not support hierarchical directories.

The directory description, and the directory contents, assume the disk is composed of 512 byte blocks. While this is common in modern machines, it was uncommon in the 1970s, when 128 byte and 256 byte sectors were more common. If there is sector interleaving, this is done transparently from the perspective of the logical blocks access to the disks by the p-Machine.

In most cases, the directory resides at block 2 on the disk and extends for 4 blocks if it is a single directory, 8 blocks if it is a duplicated (backed-up) directory. In most cases, blocks 0 and 1 are reserved for the bootstrap.

In some cases, the entire disk contents are offset by several blocks, and this is also transparent. One trick, used when first developing the p-System, before it was self-hosting, was to have an RT-11 disk also contain a UCSD p-System directory, and have both file systems reference the same files. The offset avoided stomping on the RT-11 directory blocks.

Byte Sex

Byte sex is an issue for disk images. They have the same byte sex as the host microprocessor architecture. You can detect the byte sex by looking that the FIRSTBLK field of the first directory entry (the volume label). It will always be six, so the high byte will be zero.

Once the byte sex has been determined, it should be used for all 16-byte word accesses. When extracting a bit field, it is always specified as bits within such a value, never the raw bytes.

Strings (file names and volume names) are unaffected by the byte sex.

Directory Entry Format

The Pascal declaration of a directory entry is as follows:

```
TYPE
 DIRENTRY =
  PACKED RECORD
   FIRSTBLK: INTEGER;
   DLASTBLK: INTEGER;
   CASE DFKIND: FILEKIND OF
   SECUREDIR,
   UNTYPEDFILE: (
    FILLER1 : 0..2048;
    DVID: VID;
    DEOVBLK: INTEGER;
    DNUMFILES: DIRRANGE;
    DLOADTIME: INTEGER;
    DLASTBOOT: DATEREC);
   XDSKFILE,CODEFILE,TEXTFILE,INFOFILE,
   DATAFILE,GRAFFILE,FOTOFILE: (
    FILLER2 : 0..1024;
    STATUS: BOOLEAN;
    DTID: TID;
    DLASTBYTE: integer;
    DACCESS: DATEREC)
  END;
```

This is not useful unless you know how UCSD Pascal lays out it records.

The directory entry layout for regular file looks like this:

bytes 0, 1:
: The number of the first block of the file, counting from zero, relative to the start of the disk.

bytes 2, 3:
: The number of the first block past the end of the file, counting from zero, relative to the start of the disk.

bytes 4, 5:
: The bottom four bits of the 16-bit word (*i.e.* in a different byte depending on byte sex) describe the file kind.

  UNTYPEDFILE, 0:
  : Not used (but see Volume Label, below).

  XDSKFILE, 1:
  : Not used.

  CODEFILE, 2:
  : Created by the Compiler, Assembler, or Linker. Contains code segments, see *ucsdpsys_codefile*(5) from the ucsd-psystem-xc project, for more information.

  TEXTFILE, 3:
  : All text file created by the Editor, or Listings by the Compiler or Linker, are this file kind.

  INFOFILE, 4:
  : Not used.

  DATAFILE, 5:
  : Any file created by a program, that is not a text file, will be marked as a datafile.

  GRAFFILE, 6:
  : Not used.

  FOTOFILE, 7:
  : Not used.

  SECUREDIR, 8:
  : Not used.

  The file's extension is usually the first 4 bytes of the file kind. For example, ".TEXT" and ".CODE".

bytes 6..21:
: The file's name. The first byte is the length, remaining bytes are the name, in 7-bit ASCII, no control characters. Always upper case on disk, case insensitive for searching. A length of zero is invalid, a length greater than 15 is invalid.

bytes 22, 23:
: How many bytes of the last block are actually used.

bytes 24, 25:
: The date the file was last modified. In the 16-bit word, the lower 5 bits are the day (1..32), the next 4 bits are them month (1..12), the high 7 bits are the year (0..99).

Directory Format

Each directory entry is 26 bytes long. Thus, 78 directory entries fit into 2kB (4 blocks). The remaining bytes are zero padded. The first directory entry describes the directory itself, the remaining 77 entries are used for regular files.

The first two blocks are reserved for the boot loader.

Volume label.

The layout of the first directory entry (the volume label) is slightly different:

bytes 0, 1:
: The first physical block on the disk (*i.e.* zero).

bytes 2, 3:
: The first block after the directory (*i.e.* six or ten).

bytes 4, 5:
: Ignore, set to zero, corresponding to a UNTYPEDFILE file kind value, see above.

bytes 6..13:
: The volume's name. The first byte is the length, remaining bytes are the name, in 7-bit ASCII, no control characters. Always upper case on disk, case insensitive for searching. A length of zero is invalid, a length greater than 7 is invalid.

bytes 14, 15:
: The physical number of blocks on the volume (disk).

bytes 16, 17:
: The number of files on the disk.

bytes 18, 19:
: Ignore. Set to zero.

bytes 20, 21:
: The date the volume was last mounted or modified. Same date format as above.

bytes 22..25:
: Ignore. Set to zero.

COPYRIGHT

AUTHOR

Peter Miller E-Mail: pmiller@opensource.org.au
/\/* WWW: http://miller.emu.id.au/pmiller/

NAME

ucsdpsys_text − UCSD p-System text file format

DESCRIPTION

The format of a textfile is as follows:

- There are two blocks (1024 bytes) of header information at the beginning of the file. This information is used by the Pascal Editor. The Pascal system creates the header page when a user program opens a textfile. The header page is transferred only during disk-to-disk transfers; transfers to character devices, such as the console or printer, always omit the header page.
- The rest of the file consists of two-block pages. Each page contains lines of text, separated from each other by RETURN characters (ASCII 0x0D). No line ever crosses a page boundary; thus a page contains only whole lines. After the last line on a page, the remainder of the page is filled with NUL characters (ASCII 0x00). READ and READLN skip the NUL characters, and WRITE and WRITELN provide them automatically. Thus this page formatting is normally invisible to a Pascal program.
- A sequence of leading spaces in a line may be compressed to a DLE-blank code. This code consists of a DLE control character (ASCII 0x10) followed by one byte containing the number of spaces to indent plus 32 (decimal). Using this code saves a considerable amount of space in files where indentation occurs frequently. The Editor is the main creator of DLE-blank codes; it usually outputs a DLE-blank code where a sequence of spaces occurs at the beginning of a line. However, the DLE-blank code is optional; some lines may have it, and others may have space characters instead. Also, a line with no indentation may or may not be preceded by a DLE character and an indent code value of 32 (meaning O indentation).

Limitations

- The smallest text file is 2kB. There is no way to avoid the 1kB editor information at the start of a text file.
- You must guarantee that every 1KB text block ends with at least one NUL. This is to cope with a bug in the native compiler.

OPTIONS

The following options are understood:

−Help
: Provide some help with using the *ucsdpsys_text* program.

−VERSion
: Print the version of the *ucsdpsys_text* program being executed.

All other options will produce a diagnostic error.

SEE ALSO

ucsdpsys_text(1)
: convert UCSD p-System text files

COPYRIGHT

ucsdpsys_text version 1.22

AUTHOR

Peter Miller E-Mail: pmiller@opensource.org.au
/\/* WWW: http://miller.emu.id.au/pmiller/