NAME

sane−apple − SANE backend for Apple flatbed scanners

DESCRIPTION

The **sane−apple** library implements a SANE (Scanner Access Now Easy) backend that provides access to Apple flatbed scanners. At present, the following scanners are supported from this backend:

```
--------------- ----- ------------------ ------
AppleScanner    4bit  16 Shades of Gray
OneScanner      8bit  256 Shades of Gray
ColorOneScanner 24bit RGB color          3-pass
```

If you own a Apple scanner other than the ones listed above that works with this backend, please let us know by sending the scanner's model name, SCSI id, and firmware revision to *sane−devel@lists.alioth.debian.org*. See http://www.sane−project.org/mailing−lists.html for details on how to subscribe to sane−devel.

DEVICE NAMES

This backend expects device names of the form:

special

Where *special* is either the path-name for the special device that corresponds to a SCSI scanner. For SCSI scanners, the special device name must be a generic SCSI device or a symlink to such a device. Under Linux, such a device name could be */dev/sga* or */dev/sge*, for example. See sane−scsi(5) for details.

CONFIGURATION

The *apple.conf* file is a list of options and device names that correspond to Apple scanners. Empty lines and lines starting with a hash mark (#) are ignored. See sane−scsi(5) on details of what constitutes a valid device name.

Options come in two flavors: global and positional ones. Global options apply to all devices managed by the backend, whereas positional options apply just to the most recently mentioned device. Note that this means that the order in which the options appear matters!

SCSI ADAPTER TIPS

SCSI scanners are typically delivered with an ISA SCSI adapter. Unfortunately, that adapter is not worth much since it is not interrupt driven. It is sometimes possible to get the supplied card to work, but without an interrupt line, scanning will put so much load on the system that it becomes almost unusable for other tasks.

FILES

@CONFIGDIR@/apple.conf
: The backend configuration file (see also description of **SANE_CONFIG_DIR** below).

@LIBDIR@/libsane−apple.a
: The static library implementing this backend.

@LIBDIR@/libsane−apple.so
: The shared library implementing this backend (present on systems that support dynamic loading).

ENVIRONMENT

SANE_CONFIG_DIR
: This environment variable is list of directories where SANE looks for the configuration file. Under UNIX directory names are separated by a colon (':'), under OS/2 by a semi-colon (';'). If SANE_CONFIG_DIR is not set, SANE defaults to searching the current working directory (".") and then /etc/sane.d. If the value of $SANE_CONFIG_DIR ends with the separator character, the default directories are searched after the directory list. For example, setting **SANE_CONFIG_DIR** to "/tmp/config:" would result in directories "tmp/config", ".", and "@CONFIGDIR@"

being searched (in that order).

SANE_DEBUG_APPLE
Controls the debug level. A value of 255 prints all debug output. Smaller values reduce verbosity. Requires a library compiled with debug support.

CURRENT STATUS

The apple backend is now in version 0.3 (Tue Jul 21 1998). Since I only have the AppleScanner and not the other models (OneScanner, ColorOneScanner) I can only develop/test for the AppleScanner effectively. However with this release I almost completed the gui part of all scanners. Most of the functionality is there. At least OneScanner should scan at the AppleScanner's compatible modes (LineArt, HalfTone, Gray16). My personal belief is that with a slight touch of debugging the OneScanner could be actually usable. The ColorOneScanner needs more work. AppleScanner is of course almost fully supported.

MISSING FUNCTIONALITY

Currently all three models lack upload/download support.

AppleScanner
Cannot up/download a halftone pattern.

OneScanner
Cannot up/download halftone patterns or calibration vectors.

ColorOneScanner
Cannot up/download halftone patterns, calibration vectors, custom Color Correction Tables (CCT) and of course custom gamma tables.

Park/UnPark (OneScanner, ColorOneScanner)
Some capabilities are missing.

The above functionalities are missing because I don't have the hardware to experiment on. Another reason is my lack of understanding as to how or if the SANE API provide means to describe any array type besides gamma.

UNSUPPORTED FEATURES

The following "features" will never be supported, at least while I maintain the sane−apple backend.

NoHome (AppleScanner)
The scanner lamp stays on and the carriage assembly remains where it stops at the end of the scan. After two minutes, if the scanner does not receive another SCAN command, the lamp goes off and the carriage returns to the home position.

Compression (AppleScanner)
The Scanner can compress data with CCITT Group III one dimensional algorithm (fax) and the Skip White Line algorithm.

Multiple Windows (AppleScanner)
AppleScanner may support multiple windows. It would be a cool feature and a challenge for me to code if it could intermix different options for different windows (scan areas). This way it could scan a document in LineArt mode but the figures in it in Gray and at a different resolution. Unfortunately this is impossible.

Scan Direction (OneScanner)
It controls the scan direction. (?)

Status/Reset Button (OneScanner)
This option controls the status of the button on the OneScanner model. You can also reset the button status by software.

BUGS

SANE backend bugs are divided in two classes. We have **GUI** bugs and **scanner specific** bugs.

We know we have a GUI bug when a parameter is not showing up when it should (active) or vice versa. Finding out which parameters are active across various Apple modes and models from the documentation **ftp://ftpdev.info.apple.com/devworld/Technical_Documentation/Peripherals_Documentation/** is an interesting exercise. I may have missed some dependencies. For example of the threshold parameter the Apple Scanners Programming Guide says nothing. I had to assume it is valid only in LineArt mode.

Scanner specific bugs are mostly due to mandatory round-offs in order to scan. In the documentation in one place states that the width of the scan area should be a byte multiple. In another place it says that the width of the scan area should be an even byte multiple. Go figure...

Other sources of bugs are due to scsi communication, scsi connects and disconnects. However the classical bugs are still there. So you may encounter buffer overruns, null pointers, memory corruption and **SANE** API violations.

SIGSEGV on SliceBars

When you try to modify the scan area from the slice bar you have a nice little cute core dump. I don't know why. If you select the scan area from the preview window or by hand typing the numbers everything is fine. The SIGSEGV happens deep in gtk library (gdk). I really cannot debug it.

Options too much

It is possible, especially for the ColorOneScanner, for the backend's options panel to extend beyond your screen. It happens with mine and I am running my X Server at 1024x768. What can I say? Try smaller fonts in the X server, or virtual screens.

Weird SCSI behaviour

I am quoting David Myers Here...

>> OS: FreeBSD 2.2.6
>> CC: egcs-1.02
Just wanted to follow up on this... I recently changed my SCSI card from the Adaptec 2940UW to a dual-channel Symbios 786 chipset. When I started up SANE with your driver, I managed to scan line art drawings okay, but Gray16 scans led to a stream of SCSI error messages on the console, ultimately hanging with a message saying the scanner wasn't releasing the SCSI bus. This may be that the Symbios is simply less tolerant of ancient hardware, or may be bugs in your driver or in SANE itself...

DEBUG

If you encounter a GUI bug please set the environmental variable SANE_DEBUG_APPLE to 255 and rerun the exact sequence of keystrokes and menu selections to reproduce it. Then send me a report with the log attached.

If you have an Apple Macintosh with the AppleScanners driver installed, reporting to me which options are grayed out (inactive) in what modes would be very helpful.

If you want to offer some help but you don't have a scanner, or you don't have the model you would like to help with, or you are a SANE developer and you just want to take a look at how the apple backend looks like, goto to apple.h and #define the NEUTRALIZE_BACKEND macro. You can select the scanner model through the APPLE_MODEL_SELECT macro. Available options are APPLESCANNER, ONESCANNER, COLORONESCANNER.

If you encounter a SCSI bus error or trimmed and/or displaced images please set the environment variable SANE_DEBUG_SANEI_SCSI to 255 before sending me the report.

TODO

Non Blocking Support
Make sane−apple a non blocking backend. Properly support **sane_set_io_mode** and **sane_get_select_fd**

Scan Make scanning possible for all models in all supported modes.

Add other missing functionality

SEE ALSO

sane(7), sane−scsi(5)

AUTHOR

The sane−apple backend was written not entirely from scratch by Milon Firikis. It is mostly based on the mustek backend from David Mosberger and Andreas Czechanowski