



Full credit is given to the above companies including the Operating System (OS) that this PDF file was generated!

Rocky Enterprise Linux 9.2 Manual Pages on command 'sane-canon_pp.5'

\$ man sane-canon_pp.5

sane-canon_pp(5) SANE Scanner Access Now Easy sane-canon_pp(5)

NAME

sane-canon_pp - SANE backend for Canon CanoScan Parallel Port flatbed scanners

DESCRIPTION

The sane-canon_pp library implements a SANE (Scanner Access Now Easy) backend that provides access to the following Canon flatbed scanners:

- CanoScan FB320P
- CanoScan FB620P
- CanoScan FB330P
- CanoScan FB630P
- CanoScan N340P
- CanoScan N640P
- CanoScan N640P ex

No USB scanners are supported and there are no plans to support them in the future. Other projects are working on support for USB scanners. See the PROJECTS file for more detail.

The FB310P and FB610P are re-badged Avison scanners which use a different command set, so are unlikely to be supported by this backend in the future.

IMPORTANT: this is alpha code. While we have made every effort to make it as reliable as possible, it will not always work as expected. Feedback is still appreciated. Please send any bug reports to the maintainers as listed on the web page (listed in SEE ALSO below).

DEVICE NAMES

This backend expects device names of the form presented by libieee1284(3). These names

are highly dependent on operating system and version.

On Linux 2.4 kernels this will be of the form `parport0` or older (2.2 and before) kernels may produce names like `0x378` (the base address of your port) or simply `0` depending on your module configuration. Check the contents of `/proc/parport` if it exists. If you don't want to specify a default port (or don't know its name), the backend should be able to detect which port your scanner is on.

CONFIGURATION

The contents of the `canon_pp.conf` file is a list of options for the driver to use. Empty lines and lines starting with a hash mark (`#`) are ignored.

The supported options are currently `ieee1284`, `calibrate`, `init_mode`, and `force_nibble`. Option `ieee1284` `port-name` defines which port to use. The format of `port-name` is OS dependent, based on the names presented by `libieee1284`. Please only have one of these lines, or all but one will be ignored.

Option `calibrate` `cal-file` [`port-name`] defines which calibration file to use on a per-port basis. If you only have one `parport`, the `port-name` argument may be omitted - but be careful as this will cause problems on multi-scanner systems. You may have as many of these lines as you like, as long as each has a unique port name. The tilde (`~`) character is acceptable and will be expanded to the value of the `HOME` environment.

Option `init_mode` `<AUTO|FB620P|FB630P>` [`portname`] defines which initialisation (wake-up) mode to use on a per-port basis. If you only have one `parport`, the `portname` argument may be omitted - but be careful as this may cause problems on multi-scanner systems. You may have as many of these lines as you like, as long as each has a unique port name. The valid initialisation modes are `FB620P` (which strobes `10101010` and `01010101` on the data pins), `FB630P` (which strobes `11001100` and `00110011` on the data pins) and `AUTO`, which will try `FB630P` mode first then `FB620P` mode second. The `FB620P` mode is also used by the `FB320P`. The `FB630P` mode is used by the `FB330P`, `N340P`, and `N640P`.

Option `force_nibble` forces the driver to use nibble mode even if ECP mode is reported to work by `libieee1284`. This works-around the rare issue of ECP mode being reported to work by the library, then not working.

TIPS

Hit the "Calibrate" button before scanning. It vastly improves the quality of scans.

To enable automatic detection of your scanner, uncomment the "`canon_pp`" line from `/etc/sane.d/dll.conf`

FILES

`/etc/sane.d/canon_pp.conf`

The backend configuration file (see also description of `SANE_CONFIG_DIR` below).

`/usr/lib/x86_64-linux-gnu/sane/libsane-canon_pp.a`

The static library implementing this backend.

`/usr/lib/x86_64-linux-gnu/sane/libsane-canon_pp.so`

The shared library implementing this backend (present on systems that support dynamic loading).

ENVIRONMENT

`SANE_CONFIG_DIR`

This environment variable specifies the list of directories that may contain the configuration file. On *NIX systems, the directories are separated by a colon (`:`), under OS/2, they are separated by a semi-colon (`;`). If this variable is not set, the configuration file is searched in two default directories: first, the current working directory (`."`) and then in `/etc/sane.d`. If the value of the environment variable ends with the directory separator character, then the default directories are searched after the explicitly specified directories. For example, setting `SANE_CONFIG_DIR` to `"/tmp/config:"` would result in directories `tmp/config`, `.`, and `/etc/sane.d` being searched (in this order).

`SANE_DEBUG_CANON_PP`

If the library was compiled with debug support enabled, this environment variable controls the debug level for this backend. Higher debug levels increase the verbosity of the output.

Example: `export SANE_DEBUG_CANON_PP=4`

NOTES

Features available in the Windows interface

Brightness and Contrast

These are not implemented, and probably never will be. These appear to be implemented entirely in software. Use GIMP or a similar program if you need these features.

Descreen Mode

This appears on our first analysis to be just oversampling with an anti-aliasing filter. Again, it seems to be implemented entirely in software, so GIMP is your

best bet for now.

Gamma Tables

This is under investigation, but for now only a simple gamma profile (ie: the one returned during calibration) will be loaded.

Communication Problems

ECP mode in libieee1284 doesn't always work properly, even with new hardware. We believe that this is a ppdev problem. If you change the configuration file to include force_nibble, the problem will go away, but you will only be able to scan in nibble mode.

Sometimes the scanner can be left in a state where our code cannot revive it. If the backend reports no scanner present, try unplugging the power and plugging it back in. Also try unplugging printers from the pass-through port.

The scanner will not respond correctly to our commands when you first plug in the power. You may find if you try a scan very soon after plugging in the power that the backend will incorrectly report that you have no scanner present. To avoid this, give it about 10 seconds to reset itself before attempting any scans.

Repeated Lines

Sometimes at high resolutions (ie. 600dpi) you will notice lines which appear twice. These lines correspond to points where the scanner head has stopped during the scan (it stops every time the internal 64kb buffer is full). Basically it's a mechanical problem inside the scanner, that the tolerance of movement for a start/stop event is greater than 1/600 inches. I've never tried the windows driver so I'm not sure how (or if) it works around this problem, but as we don't know how to rewind the scanner head to do these bits again, there's currently no nice way to deal with the problem.

Grey-scale Scans

Be aware that the scanner uses the green LEDs to read grey-scale scans, meaning green coloured things will appear lighter than normal, and red and blue coloured items will appear darker than normal. For high-accuracy grey-scale scans of colour items, it's best just to scan in colour and convert to grey-scale in graphics software such as the GIMP.

FB620P/FB320P Caveats

These models can not be reset in the same way as the others. The windows driver doesn't know how to reset them either - when left with an inconsistent scanner, it will start scanning half way down the page!

Aborting is known to work correctly on the FB*30P models, and is known to be broken on the

FB*20P models. The FB620P which I tested on simply returns garbage after a scan has been aborted using the method we know. Aborting is able to leave the scanner in a state where it can be shut down, but not where another scan can be made.

SEE ALSO

sane(7), sane-dll(5), libieee1284(3),

<http://canon-fb330p.sourceforge.net/>

AUTHOR

This backend is primarily the work of Simon Krix (Reverse Engineering), and Matthew Duggan (SANE interface).

Many thanks to Kevin Easton for his comments and help, and Kent A. Signorini for his help with the N340P.

11 Jul 2008

sane-canon_pp(5)