

## Re: Digital-tv card drivers and API discussion

*Source:* <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/arch/2004-04/0038.html>

---

*Teemu.Parkkinen\_at\_patria.fi*

*Date:* 04/15/04

To: freebsd-arch@freebsd.org

Date: Thu, 15 Apr 2004 12:36:38 +0300

Implementation in userspace would be quite interesting, as it does not need any modifications to the kernel. I have no idea if this is feasible with PCI-devices though. Linux-compatibility using libraries could probably be made, but needs some extra work when compared to direct support of linux-api. If you have ideas on this, I'm interested to hear them.

As most digital-tv receivers are PCI-cards, I think it's important to have some kind of elegant mechanism to use them that is:

- a) sufficient for most (all) PCI, USB, Firewire etc digital tv-cards to use them for watching, recording and using other dvb-services.
- b) easy to add new drivers to it
- c) compatible to linux (as it is desired)

The cards vary greatly with different options, some have decoders, some not, some use satellite antenna motors, ci-modules etc, etc. To be sufficient for all dvb-cards, the API is very important. Therefore I prefer linux-api as it seems to be rather complete. Designing completely new api takes some time and I don't know if it will be any better than linux-api, because they have been working on it for some time.

I don't have enough experience to decide if it is ok for FreeBSD, as Jari said that it may not be OS-independent enough. I would appreciate some feedback on this matter, especially which is the best way to go when considering the long-term dvb-support in FreeBSD.

-Teemu

Jari Kirma  
<kirma@cs.hut.fi> To: freebsd-arch@freebsd.org  
Sent by: cc:  
owner-freebsd-arch@ Subject: Digital-tv card drivers and API discussion  
freebsd.org

04/15/04 10:11 AM

- > *I am about to write a digital tv-driver for my dvb-c -card. Because*
- > *FreeBSD does not yet have any dvb-devices and I don't have any prior*
- > *driver development experience, I have a couple of questions for you.*
- >
- > *1) Should we use Linux-DVB API as a reference, or should we consider*
- > *some changes to it? The API seems to be constantly changing and*
- > *improving. Version 3 is available here:*
- > *<http://www.linuxtv.org/download/dvb/linux-dvb-api-1.0.0.pdf>*
- > *but they are currently working on version 4. In my opinion, the API*
- > *should be minimal, but complete, so there is no need to constantly add*
- > *new features to it.*
- >
- > *2) As linux kernel is GPL-licensed, I cannot just port the linux driver*
- > *to FreeBSD, right? In other words, we have to write the driver from*
- > *scratch. In this case we don't have to stick with the Linux DVB-API and*
- > *therefore I suggest that we give think the api through before*
- > *deciding how we implement it (do we follow linux api or not).*
- >
- > *3) Do you have any pointers to good books or other documentation on how*
- > *to write device drivers for UNIX (BSD)? I already have read those from*
- > *FreeBSD documentation, but a decent book would be handy.*

I received my TechnoTrend USB DVB-C box couple weeks ago and pondered the same issues. I got some suggestions in freebsd-multimedia (like, compability with Linux is nice). I was also suggested to take a look at early work at VideoBSD <<http://people.freebsd.org/~jmg/videobsd.html>>. I haven't made any decisions on the interface yet, but I have had some thoughts.

The biggest problem with Linux DVB API is that its interface is at device/ioctl level. I consider that a poison for reasonable OS independence. There's also libdvb <<http://freshmeat.net/projects/libdvb/>> which supposedly abstracts away some OS-specific parts, but based on a very quick glance, it might not cover whole set of operations required to operate DVB sensibly.

freebsd-arch: Re: Digital-tv card drivers and API discussion

In my case, I've been able to prototype my "driver" completely in userspace as it is a USB device. I use ugen device bulk pipe for device control, ugen isochronous pipe provides stream transport. I have had some problems because ugen isn't really tried and tested on (relatively) high-bandwidth isochronous transfers, but most of those have been solved. I'm able to watch DVB programs converted from MPEG TS substreams to MPEG PS stream and piped to mplayer, completely in userland. Of course, this isn't really doable with devices in PCI bus, but in USB or Firewire, it should be pretty easy. It can also nicely separate the parts that may be "poisoned" by GPL outside the kernel.

Even with this design, it would be possible to design "fake" devices emulating Linux DVB interface and actually redirect the operations to userland daemon. Extra copying of the MPEG TS stream can be avoided (I'm planning to write a mmaped, properly synchronised, zero-copy usb isochronous device driver), and testing is reasonably easy. Biggest problem with this approach is obviously that you have to choose either long latency or extra context switches...

Please note that I have not given thought on MPEG decoding, because my device exports only MPEG stream. MPEG decoders should be handled in some intelligent, modular way, because those two parts may well be operated separately although they may be bundled on the same card.

-kirma

---

freebsd-arch@freebsd.org mailing list

<http://lists.freebsd.org/mailman/listinfo/freebsd-arch>

To unsubscribe, send any mail to "freebsd-arch-unsubscribe@freebsd.org"

---

freebsd-arch@freebsd.org mailing list

<http://lists.freebsd.org/mailman/listinfo/freebsd-arch>

To unsubscribe, send any mail to "freebsd-arch-unsubscribe@freebsd.org"