Submitting Drivers For The Linux Kernel

This document is intended to explain how to submit device drivers to the various kernel trees. Note that if you are interested in video card drivers you should probably talk to XFree86 (http://www.xfree86.org/) and/or X.Org (http://x.org/) instead.

> ***Note***
>
> This document is old and has seen little maintenance in recent years; it should probably be updated or, perhaps better, just deleted. Most of what is here can be found in the other development documents anyway.
>
> Oh, and we don't really recommend submitting changes to XFree86 :)

Also read the Documentation/process/submitting-patches.rst document.

Allocating Device Numbers

Major and minor numbers for block and character devices are allocated by the Linux assigned name and number authority (currently this is Torben Mathiasen). The site is http://www.lanana.org/. This also deals with allocating numbers for devices that are not going to be submitted to the mainstream kernel. See Documentation/admin-guide/devices.rst for more information on this.

If you don't use assigned numbers then when your device is submitted it will be given an assigned number even if that is different from values you may have shipped to customers before.

Who To Submit Drivers To

Linux 2.0:

No new drivers are accepted for this kernel tree.

Linux 2.2:

No new drivers are accepted for this kernel tree.

Linux 2.4:

If the code area has a general maintainer then please submit it to the maintainer listed in MAINTAINERS in the kernel file. If the maintainer does not respond or you cannot find the appropriate maintainer then please contact Willy Tarreau <w@1wt.eu>.

Linux 2.6 and upper:

The same rules apply as 2.4 except that you should follow linux-kernel to track changes in API's. The final contact point for Linux 2.6+ submissions is Andrew Morton.

What Criteria Determine Acceptance

Licensing:

The code must be released to us under the GNU General Public License. We don't insist on any kind of exclusive GPL licensing, and if you wish the driver to be useful to other communities such as BSD you may well wish to release under multiple licenses. See accepted licenses at include/linux/module.h

Copyright:

The copyright owner must agree to use of GPL. It's best if the submitter and copyright owner are the same person/entity. If not, the name of the person/entity authorizing use of GPL should be listed in case it's necessary to verify the will of the copyright owner.

Interfaces:

If your driver uses existing interfaces and behaves like other drivers in the same class it will be much more likely to be accepted than if it invents gratuitous new ones. If you need to implement a common API over Linux and NT drivers do it in userspace.

Code:

Please use the Linux style of code formatting as documented in :ref:`Documentation/process/coding-style.rst <codingStyle>`. If you have sections of code that need to be in other formats, for example because they are shared with a windows driver kit and you want to maintain them just once separate them out nicely and note this fact.

Portability:

Pointers are not always 32bits, not all computers are little endian, people do not all have floating point and you shouldn't use inline x86 assembler in your driver without careful thought. Pure x86 drivers generally are not popular. If you only have x86 hardware it is hard to test portability but it is easy to make sure the code can easily be made portable.

Clarity:

It helps if anyone can see how to fix the driver. It helps you because you get patches not bug reports. If you submit a driver that intentionally obfuscates how the hardware works it will go in the bitbucket.

PM support:

Since Linux is used on many portable and desktop systems, your driver is likely to be used on such a system and therefore it should support basic power management by implementing, if necessary, the .suspend and .resume methods used during the system-wide suspend and resume transitions. You should verify that your driver correctly handles the suspend and resume, but if you are unable to ensure that, please at least define the .suspend method returning the -ENOSYS ("Function not implemented") error. You should also try to make sure that your driver uses as little power as possible when it's not doing anything. For the driver testing instructions see Documentation/power/drivers-testing.txt and for a relatively complete overview of the power management issues related to drivers see Documentation/power/admin-guide/devices.rst .

Control:

In general if there is active maintenance of a driver by the author then patches will be redirected to them unless they are totally obvious and without need of checking. If you want to be the contact and update point for the driver it is a good idea to state this in the comments, and include an entry in MAINTAINERS for your driver.

What Criteria Do Not Determine Acceptance

Vendor:

Being the hardware vendor and maintaining the driver is often a good thing. If there is a stable working driver from other people already in the tree don't expect 'we are the vendor' to get your driver chosen. Ideally work with the existing driver author to build a single perfect driver.

Author:

It doesn't matter if a large Linux company wrote the driver, or you did. Nobody has any special access to the kernel tree. Anyone who tells you otherwise isn't telling the whole story.

Resources

Linux kernel master tree:

ftp.*country_code*.kernel.org:/pub/linux/kernel/...

where *country_code* == your country code, such as **us**, **uk**, **fr**, etc.

http://git.kernel.org/?p=linux/kernel/git/torvalds/linux.git

Linux kernel mailing list:

linux-kernel@vger.kernel.org [mail majordomo@vger.kernel.org to subscribe]

Linux Device Drivers, Third Edition (covers 2.6.10):

http://lwn.net/Kernel/LDD3/ (free version)

LWN.net:

Weekly summary of kernel development activity - http://lwn.net/

2.6 API changes:

http://lwn.net/Articles/2.6-kernel-api/

Porting drivers from prior kernels to 2.6:

http://lwn.net/Articles/driver-porting/

KernelNewbies:

Documentation and assistance for new kernel programmers

http://kernelnewbies.org/

Linux USB project:

http://www.linux-usb.org/

How to NOT write kernel driver by Arjan van de Ven:

http://www.fenrus.org/how-to-not-write-a-device-driver-paper.pdf

Kernel Janitor:

http://kernelnewbies.org/KernelJanitors

GIT, Fast Version Control System:

http://git-scm.com/