Howto use uvc1.5 (webcams) on Linux (Debian / Mint / Ubuntu)

WORK IN PROGRESS /
PLEASE FILE AN ISSUE IF YOU FIND WRONG INFORMATION

Licence CC-by Author : Eric Bachard version 0.1 / 2018/09/26

1. verify your kernel can use ucv 1.5 devices.

e.g. : Logitech Brio 4K webcam

Read "How do I find out whether my camera is a UVC device or not?".

Check whether it's the case :

me@my_machine ~/linux-hwe-4.15.0/drivers/media/usb/uvc $ lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 003: ID 8087:0aa7 Intel Corp.
Bus 001 Device 002: ID 04f2:b57a Chicony Electronics Co., Ltd
Bus 001 Device 010: ID 046d:086b Logitech, Inc.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub

-> the ID 046d:086b is the Logitech Brio 4K.

Now check for true :

```
lsusb -d 046d:086b -v | grep "14 Video"
```

The answer looks ok, means my kernel is able to take over UVC devices :

me@my_machine ~/linux-hwe-4.15.0/drivers/media/usb/uvc $ lsusb -d 046d:086b -v | grep "14 Video"

Returns :

Couldn't open device, some information will be missing #(USB 2.0 builtin Chicony, Asus Zenbook)
bFunctionClass 14 Video
bInterfaceClass 14 Video
bInterfaceClass 14 Video
bInterfaceClass 14 Video
bInterfaceClass 14 Video
bInterfaceClass 14 Video
bInterfaceClass 14 Video
bInterfaceClass 14 Video
bInterfaceClass 14 Video
bInterfaceClass 14 Video
bInterfaceClass 14 Video
bInterfaceClass 14 Video
bInterfaceClass 14 Video
bInterfaceClass 14 Video
bInterfaceClass 14 Video

=> you're ready to enable the kernal traces now !

Catch the kernel traces

enable uvcvideo kernel traces

```
sudo echo 0xffff > /sys/module/uvcvideo/parameters/trace
```

capture the information :

unplug / replug the Logitech Brio 4K, to see what happens in the log

Since this is extremely verbose, better disable uvcvideo kernel traces :

```
sudo echo 0 > /sys/module/uvcvideo/parameters/trace
```

Catch the contents of the kernel log:

Simply do : dmesg > dmesg.log

Select what is usefull information (probably the last ~ 500 lines only), and finish by hand, to select only uvc relevant information.

Note: such information is extremely important, and read the log cannt be bad. If you understand nothing and cannot solve your issue, this log should very probably help the uvc maintainer to provide you a track helping you.

2. howto create a new kernel on Linux (Debian like)

Current diff using 4.15.0 sources : see

Download the right kernel sources :

1. enable the source repository with apt-get (synaptic or any

2. Download the kernel sources

The recent kernel known to provide recent uvc protocol is the 4.15 and above. I have no idea the previous kernel versions will work (please provide feedback if ever ...)
To obtain the sources, just type : apt-get source linux-source-4.15.0

Note : at the end, you'll see a linux-hwe-4.15.0 tree instead of a simple linux-4.15.0 tree. That's due to the linux kernel upgrade system.

e.g. for security reason, or in order to obtain the up-to-date version, you can use git with the following command line :

git clone git://git.launchpad.net/~ubuntu-kernel/ubuntu/+source/linux/+git/xenial -b hwe
(you can expect around a 170 MB of kernel files download).

Apply the patch

cp uvc-1.5.diff linux-hwe-4.15.0/drivers/media/usb/uvc

cd linux-hwe-4.15.0/drivers/media/usb/uvc

check the patch will apply
patch --dry-run -p1 < uvc_1.5.diff

And if nothing went wrong, apply it for true : patch -p1 < uvc_1.5.diff
On my machine, I simply obtained :

me@my_machine ~/linux-hwe-4.15.0/drivers/media/usb/uvc $ patch --dry-run -p1 < uvc_1.5.diff
checking file uvc_video.c

me@my_machine ~/linux-hwe-4.15.0/drivers/media/usb/uvc $ patch -p1 < uvc_1.5.diff
patching file uvc_video.c

And you're done for the sources modifications !

Build the kernel : complete instructions are provided using the links below.

English version : https://debian-handbook.info/browse/en-US/stable/sect.kernel-compilation.html
French version : https://debian-handbook.info/browse/fr-FR/stable/sect.kernel-compilation.html

TIP : for other locales, please look at the links, and guess how to retrieve your own ;-)

Install the new kernel :

Everything is in linux-image-4.15.18_4.15.18-1_amd64.deb (the exact version will change, following the source evolution, of course), and the final command line is :

sudo dpkg -i linux-image-4.15.18_4.15.18-1_amd64.deb

Last step : if nothing wrong, simply reboot the 4.15.18 kernel, and everything should work.

I can't promise I'll solve your issue, but if something goes wrong, feel free to file an issue, to keep a trace, and we'll be able to progress, sharing the information.

Last but not least, I'd like to say a big THANK YOU to Laurent Pinchart for his great contribution to the UVC protocol on Linux, an ddon't hesitate to visit : http://www.ideasonboard.org

Enable and use uvcvideo quirks

Just in case something goes wrong, and it appears you need quirks, the goal of this chapter is to explain you how to load uvcvideo kernel module using the right quirks.

Firstly, you'll have to define the right quirks value you need ...

Be carefull : the given values below are hexadecimal values !!

e.g. 0x00000010 means $16_{(10)}$, and 0x00000200 = 2 x 16^{2} = 2 x 256 = $512_{(10)}$

Name	Value	Description
UVC_QUIRK_STATUS_INTERVAL	0x00000001	Interpret the status endpoint bInterval value as a number of frame instead of an exponent value.
UVC_QUIRK_PROBE_MINMAX	0x00000002	Don't issue GET_MIN or GET_MAX request on the video probe and commit controls. Try to set this quirk if the device stalls the minimum and maximum video probe and comit requests.
UVC_QUIRK_PROBE_EXTRAFIELDS	0x00000004	Initialize read-only fields of the video probe and commit structure when negotiating formats with the camera. Try to set this quirk if the device stalls the set video probe and commit requests.
UVC_QUIRK_BUILTIN_ISIGHT	0x00000008	The device is an Apple iSight camera. Some versions of those cameras advertise UVC compatibility but use proprietary video data encapsulation that needs to be handled specifically.
UVC_QUIRK_STREAM_NO_FID	0x00000010	Don't use the Frame ID field from the video payload headers. UVC devices are required to toggle the FID bit at every frame, and the uvcvideo driver uses it by default to detect the start of a new frame. Try to set this quirk if the driver doesn't return any frame to applications after successfully starting the video stream.
UVC_QUIRK_IGNORE_SELECTOR_UNIT	0x00000020	Ignore UVC selector units. Selector units allow selecting the active input for multi-input video devices. Many webcams expose a selector unit with a single input, which is pointless by valid nonetheless. However, at least one camera exposes a selector unit but doesn't implement associated requests. The selector unit then has to be ingored. Try to set this quirk if VIDIOC_G_INPUT fail.
UVC_QUIRK_FIX_BANDWIDTH	0x00000080	Try to estimate the bandwidth required for uncompressed streams instead on relying on the value reported by the camera. See FAQ 7 for more information.
UVC_QUIRK_PROBE_DEF	0x00000100	Don't request the video probe and commit default values. Some cameras don't implement the GET_DEF request for the video probe and commit controls, or even completely crash when that request is received. The uvcvideo driver can then use the current values as default values.
UVC_QUIRK_RESTRICT_FRAME_RATE	0x00000200	Ignore all frame intervals reported by the device but the first one. This quirk is used for a specific device that reports buggy frame intervals, making the image severely underexposed when selected.

Source : http://www.ideasonboard.org/uvc/faq/

e.g., on my machine, I need = 0x200 | 0x008 = 2x256 + 8 means $520_{(10)}$

sudo nano /etc/modprobe.d/uvcvideo.mod

Add the line (# at the begining of the line means it is a comment) :

0x200 | 0x008 = 520
options uvcvideo quirks=520

Now you can reload the uvcvideo module :

1. verify you disconnected all uvc 1.5 devices

2. remove ucvvideo module :
sudo rmmod uvcvideo
3. reload it
sudo modprobe uvcvideo

Last but not least: connect your uvc 1.5 device, and have fun !

Important links

[1] Laurent Pinchart site, including UVC quirks values : http://www.ideasonboard.org/uvc/faq/

[2] Example about howto select the right bits in the bitfields to enable your webcam : https://stackoverflow.com/questions/25619309/how-do-i-enable-the-uvc-quirk-fix-bandwidth-quirk-in-linux-uvc-driver

[3] Linux Kernel Mailing List : https://lkml.org/lkml/2018

(to be improved, thanks in advance for other usefull links)