Opening and Closing Devices

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, line 2)

Unknown directive type "c:namespace".

```
.. c:namespace:: V4L
```

Controlling a hardware peripheral via V4L2

Hardware that is supported using the V4L2 uAPI often consists of multiple devices or peripherals, each of which have their own driver.

The bridge driver exposes one or more V4L2 device nodes (see :ref:`v4l2_device_naming`).

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, line 18); *backlink*

Unknown interpreted text role "ref".

There are other drivers providing support for other components of the hardware, which may also expose device nodes, called V4L2 sub-devices.

When such V4L2 sub-devices are exposed, they allow controlling those other hardware components - usually connected via a serial bus (like IÂ²C, SMBus or SPI). Depending on the bridge driver, those sub-devices can be controlled indirectly via the bridge driver or explicitly via the :ref:`Media Controller <media_controller>` and via the :ref:`V4L2 sub-devices <subdev>`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, line 24); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, line 24); *backlink*

Unknown interpreted text role "ref".

The devices that require the use of the :ref:`Media Controller <media_controller>` are called **MC-centric** devices. The devices that are fully controlled via V4L2 device nodes are called **video-node-centric**.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, line 31); *backlink*

Unknown interpreted text role "ref".

Userspace can check if a V4L2 hardware peripheral is MC-centric by calling :ref:`VIDIOC_QUERYCAP` and checking the :ref:`device_caps field <device-capabilities>`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, line 36); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, line 36); *backlink*

Unknown interpreted text role "ref".

If the device returns `V4L2_CAP_IO_MC` flag at `device_caps`, then it is MC-centric, otherwise, it is video-node-centric.

It is required for MC-centric drivers to identify the V4L2 sub-devices and to configure the pipelines via the :ref:`media controller API <media_controller>` before using the peripheral. Also, the sub-devices' configuration shall be controlled via the :ref:`sub-device API <subdev>`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, line 43); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, line 43); *backlink*

Unknown interpreted text role "ref".

Note

A video-node-centric may still provide media-controller and sub-device interfaces as well.

However, in that case the media-controller and the sub-device interfaces are read-only and just provide information about the device. The actual configuration is done via the video nodes.

V4L2 Device Node Naming

V4L2 drivers are implemented as kernel modules, loaded manually by the system administrator or automatically when a device is first discovered. The driver modules plug into the `videodev` kernel module. It provides helper functions and a common application interface specified in this document.

Each driver thus loaded registers one or more device nodes with major number 81. Minor numbers are allocated dynamically unless the kernel is compiled with the kernel option CONFIG_VIDEO_FIXED_MINOR_RANGES. In that case minor numbers are allocated in ranges depending on the device node type.

The device nodes supported by the Video4Linux subsystem are:

Default device node name	Usage
`/dev/videoX`	Video and metadata for capture/output devices
`/dev/vbiX`	Vertical blank data (i.e. closed captions, teletext)
`/dev/radioX`	Radio tuners and modulators
`/dev/swradioX`	Software Defined Radio tuners and modulators
`/dev/v4l-touchX`	Touch sensors
`/dev/v4l-subdevX`	Video sub-devices (used by sensors and other components of the hardware peripheral)[1]

Where `X` is a non-negative integer.

Note

1. The actual device node name is system-dependent, as udev rules may apply.
2. There is no guarantee that `X` will remain the same for the same device, as the number depends on the device driver's probe order. If you need an unique name, udev default rules produce `/dev/v4l/by-id/` and `/dev/v4l/by-path/` directories containing links that can be used uniquely to identify a V4L2 device node:

```
$ tree /dev/v4l
/dev/v4l
â”œâ”€â”€ by-id
â”‚Â Â  â””â”€â”€ usb-OmniVision._USB_Camera-B4.04.27.1-video-index0 -> ../../video0
â””â”€â”€ by-path
    â””â”€â”€ pci-0000:00:14.0-usb-0:2:1.0-video-index0 -> ../../video0
```

[1] **V4L2 sub-device nodes** (e. g. `/dev/v4l-subdevX`) use a different set of system calls, as covered at :ref:`subdev`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, line 107); *backlink*

Unknown interpreted text role "ref".

Many drivers support "video_nr", "radio_nr" or "vbi_nr" module options to select specific video/radio/vbi node numbers. This allows the user to request that the device node is named e.g. /dev/video5 instead of leaving it to chance. When the driver supports multiple devices of the same type more than one device node number can be assigned, separated by commas:

System Message: WARNING/2 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, line 117)

Cannot analyze code. No Pygments lexer found for "none".

```
.. code-block:: none

   # modprobe mydriver video_nr=0,1 radio_nr=0,1
```

In `/etc/modules.conf` this may be written as:

```
options mydriver video_nr=0,1 radio_nr=0,1
```

When no device node number is given as module option the driver supplies a default.

Normally udev will create the device nodes in /dev automatically for you. If udev is not installed, then you need to enable the CONFIG_VIDEO_FIXED_MINOR_RANGES kernel option in order to be able to correctly relate a minor number to a device node number. I.e., you need to be certain that minor number 5 maps to device node name video5. With this kernel option different device types have different minor number ranges. These ranges are listed in :ref:`devices`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, line 130); *backlink*

Unknown interpreted text role "ref".

The creation of character special files (with mknod) is a privileged operation and devices cannot be opened by major and minor number. That means applications cannot *reliably* scan for loaded or installed drivers. The user must enter a device name, or the application can try the conventional device names.

Related Devices

Devices can support several functions. For example video capturing, VBI capturing and radio support.

The V4L2 API creates different V4L2 device nodes for each of these functions.

The V4L2 API was designed with the idea that one device node could support all functions. However, in practice this never worked: this 'feature' was never used by applications and many drivers did not support it and if they did it was certainly never tested. In addition, switching a device node between different functions only works when using the streaming I/O API, not with the :c:func:`read()`/:c:func:`write()` API.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, line 154); *backlink*

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, line 154); *backlink*

Unknown interpreted text role "c:func".

Today each V4L2 device node supports just one function.

Besides video input or output the hardware may also support audio sampling or playback. If so, these functions are implemented as ALSA PCM devices with optional ALSA audio mixer devices.

One problem with all these devices is that the V4L2 API makes no provisions to find these related V4L2 device nodes. Some really complex hardware use the Media Controller (see :ref:`media_controller`) which can be used for this purpose. But several drivers do not use it, and while some code exists that uses sysfs to discover related V4L2 device nodes (see libmedia_dev in the v4l-utils git repository), there is no library yet that can provide a single API towards both Media Controller-based devices and devices that do not use the Media Controller. If you want to work on this please write to the linux-media mailing list: https://linuxtv.org/lists.php.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, **line 168**); ***backlink***

Unknown interpreted text role "ref".

Multiple Opens

V4L2 devices can be opened more than once. [2] When this is supported by the driver, users can for example start a "panel" application to change controls like brightness or audio volume, while another application captures video and audio. In other words, panel applications are comparable to an ALSA audio mixer application. Just opening a V4L2 device should not change the state of the device. [3]

Once an application has allocated the memory buffers needed for streaming data (by calling the :ref:`VIDIOC_REQBUFS` or :ref:`VIDIOC_CREATE_BUFS` ioctls, or implicitly by calling the :c:func:`read()` or :c:func:`write()` functions) that application (filehandle) becomes the owner of the device. It is no longer allowed to make changes that would affect the buffer sizes (e.g. by calling the :ref:`VIDIOC_S_FMT <VIDIOC_G_FMT>` ioctl) and other applications are no longer allowed to allocate buffers or start or stop streaming. The EBUSY error code will be returned instead.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, **line 191**); ***backlink***

Unknown interpreted text role "ref".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, **line 191**); ***backlink***

Unknown interpreted text role "ref".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, **line 191**); ***backlink***

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, **line 191**); ***backlink***

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, **line 191**); ***backlink***

Unknown interpreted text role "ref".

Merely opening a V4L2 device does not grant exclusive access. [4] Initiating data exchange however assigns the right to read or write the requested type of data, and to change related properties, to this file descriptor. Applications can request additional access privileges using the priority mechanism described in :ref:`app-pri`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, **line 202**); ***backlink***

Unknown interpreted text role "ref".

Shared Data Streams

V4L2 drivers should not support multiple applications reading or writing the same data stream on a device by copying buffers, time multiplexing or similar means. This is better handled by a proxy application in user space.

Functions

To open and close V4L2 devices applications use the :c:func:`open()` and :c:func:`close()` function, respectively. Devices are programmed using the :ref:`ioctl() <func-ioctl>` function as explained in the following sections.

> **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, **line 219**); ***backlink***
>
> Unknown interpreted text role "c:func".

> **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, **line 219**); ***backlink***
>
> Unknown interpreted text role "c:func".

> **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, **line 219**); ***backlink***
>
> Unknown interpreted text role "ref".

[2] There are still some old and obscure drivers that have not been updated to allow for multiple opens. This implies that for such drivers :c:func:`open()` can return an `EBUSY` error code when the device is already in use.

> **System Message: ERROR/3** (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\userspace-api\media\v4l\[linux-master][Documentation][userspace-api][media][v4l]open.rst`, **line 226**); ***backlink***
>
> Unknown interpreted text role "c:func".

[3] Unfortunately, opening a radio device often switches the state of the device to radio mode in many drivers. This behavior should be fixed eventually as it violates the V4L2 specification.

[4] Drivers could recognize the `O_EXCL` open flag. Presently this is not required, so applications cannot know if it really works.