Video device' s internal representation

The actual device nodes in the `/dev` directory are created using the :c:type:`video_device` struct (`v4l2-dev.h`). This struct can either be allocated dynamically or embedded in a larger struct.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, line 6); *backlink*

Unknown interpreted text role "c:type".

To allocate it dynamically use :c:func:`video_device_alloc`:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, line 10); *backlink*

Unknown interpreted text role "c:func".

```
struct video_device *vdev = video_device_alloc();

if (vdev == NULL)
        return -ENOMEM;

vdev->release = video_device_release;
```

If you embed it in a larger struct, then you must set the `release()` callback to your own function:

```
struct video_device *vdev = &my_vdev->vdev;

vdev->release = my_vdev_release;
```

The `release()` callback must be set and it is called when the last user of the video device exits.

The default :c:func:`video_device_release` callback currently just calls `kfree` to free the allocated memory.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, line 33); *backlink*

Unknown interpreted text role "c:func".

There is also a :c:func:`video_device_release_empty` function that does nothing (is empty) and should be used if the struct is embedded and there is nothing to do when it is released.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, line 36); *backlink*

Unknown interpreted text role "c:func".

You should also set these fields of :c:type:`video_device`:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, line 40); *backlink*

Unknown interpreted text role "c:type".

- :c:type:`video_device`->v4l2_dev: must be set to the :c:type:`v4l2_device` parent device.

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, line 42); *backlink*

  Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 42); *backlink*

Unknown interpreted text role "c:type".

- :c:type:\`video_device\`->name: set to something descriptive and unique.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 45); *backlink*

Unknown interpreted text role "c:type".

- :c:type:\`video_device\`->vfl_dir: set this to `VFL_DIR_RX` for capture devices (`VFL_DIR_RX` has value 0, so this is normally already the default), set to `VFL_DIR_TX` for output devices and `VFL_DIR_M2M` for mem2mem (codec) devices.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 47); *backlink*

Unknown interpreted text role "c:type".

- :c:type:\`video_device\`->fops: set to the :c:type:\`v4l2_file_operations\` struct.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 51); *backlink*

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 51); *backlink*

Unknown interpreted text role "c:type".

- :c:type:\`video_device\`->ioctl_ops: if you use the :c:type:\`v4l2_ioctl_ops\` to simplify ioctl maintenance (highly recommended to use this and it might become compulsory in the future!), then set this to your :c:type:\`v4l2_ioctl_ops\` struct. The :c:type:\`video_device\`->vfl_type and :c:type:\`video_device\`->vfl_dir fields are used to disable ops that do not match the type/dir combination. E.g. VBI ops are disabled for non-VBI nodes, and output ops are disabled for a capture device. This makes it possible to provide just one :c:type:\`v4l2_ioctl_ops\` struct for both vbi and video nodes.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 54); *backlink*

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 54); *backlink*

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 54); *backlink*

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 54); *backlink*

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 54); *backlink*

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 54); *backlink*

Unknown interpreted text role "c:type".

- :c:type:`video_device`->lock: leave to `NULL` if you want to do all the locking in the driver. Otherwise you give it a pointer to a struct `mutex_lock` and before the :c:type:`video_device`->unlocked_ioctl file operation is called this lock will be taken by the core and released afterwards. See the next section for more details.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 64); *backlink*

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 64); *backlink*

Unknown interpreted text role "c:type".

- :c:type:`video_device`->queue: a pointer to the struct vb2_queue associated with this device node. If queue is not `NULL`, and queue->lock is not `NULL`, then queue->lock is used for the queuing ioctls (`VIDIOC_REQBUFS`, `CREATE_BUFS`, `QBUF`, `DQBUF`, `QUERYBUF`, `PREPARE_BUF`, `STREAMON` and `STREAMOFF`) instead of the lock above. That way the :ref:`vb2 <vb2_framework>` queuing framework does not have to wait for other ioctls. This queue pointer is also used by the :ref:`vb2 <vb2_framework>` helper functions to check for queuing ownership (i.e. is the filehandle calling it allowed to do the operation).

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 70); *backlink*

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 70); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 70); *backlink*

Unknown interpreted text role "ref".

- :c:type:`video_device`->prio: keeps track of the priorities. Used to implement `VIDIOC_G_PRIORITY` and `VIDIOC_S_PRIORITY`. If left to `NULL`, then it will use the struct v4l2_prio_state in :c:type:`v4l2_device`. If you want to have a separate priority state per (group of) device node(s), then you can point it to your own struct :c:type:`v4l2_prio_state`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 82); *backlink*

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, **line 82);** ***backlink***

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, **line 82);** ***backlink***

Unknown interpreted text role "c:type".

- :c:type:`video_device`->dev_parent: you only set this if v4l2_device was registered with `NULL` as the parent `device` struct. This only happens in cases where one hardware device has multiple PCI devices that all share the same :c:type:`v4l2_device` core.

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, **line 89);** ***backlink***

  Unknown interpreted text role "c:type".

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, **line 89);** ***backlink***

  Unknown interpreted text role "c:type".

  The cx88 driver is an example of this: one core :c:type:`v4l2_device` struct, but it is used by both a raw video PCI device (cx8800) and a MPEG PCI device (cx8802). Since the :c:type:`v4l2_device` cannot be associated with two PCI devices at the same time it is setup without a parent device. But when the struct video_device is initialized you **do** know which parent PCI device to use and so you set `dev_device` to the correct PCI device.

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, **line 94);** ***backlink***

  Unknown interpreted text role "c:type".

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, **line 94);** ***backlink***

  Unknown interpreted text role "c:type".

If you use :c:type:`v4l2_ioctl_ops`, then you should set :c:type:`video_device`->unlocked_ioctl to :c:func:`video_ioctl2` in your :c:type:`v4l2_file_operations` struct.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, **line 101);** ***backlink***

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, **line 101);** ***backlink***

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, **line 101);** ***backlink***

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, **line 101**); ***backlink***

Unknown interpreted text role "c:type".

In some cases you want to tell the core that a function you had specified in your :c:type:`v4l2_ioctl_ops` should be ignored. You can mark such ioctls by calling this function before :c:func:`video_register_device` is called:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, **line 105**); ***backlink***

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, **line 105**); ***backlink***

Unknown interpreted text role "c:func".

:c:func:`v4l2_disable_ioctl <v4l2_disable_ioctl>` (:c:type:`vdev <video_device>`, cmd).

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, **line 109**); ***backlink***

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, **line 109**); ***backlink***

Unknown interpreted text role "c:type".

This tends to be needed if based on external factors (e.g. which card is being used) you want to turns off certain features in :c:type:`v4l2_ioctl_ops` without having to make a new struct.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, **line 112**); ***backlink***

Unknown interpreted text role "c:type".

The :c:type:`v4l2_file_operations` struct is a subset of file_operations. The main difference is that the inode argument is omitted since it is never used.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, **line 116**); ***backlink***

Unknown interpreted text role "c:type".

If integration with the media framework is needed, you must initialize the :c:type:`media_entity` struct embedded in the :c:type:`video_device` struct (entity field) by calling :c:func:`media_entity_pads_init`:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, **line 120**); ***backlink***

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst, line 120); *backlink*

Unknown interpreted text role "c:type".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst, line 120); *backlink*

Unknown interpreted text role "c:func".

```
struct media_pad *pad = &my_vdev->pad;
int err;

err = media_entity_pads_init(&vdev->entity, 1, pad);
```

The pads array must have been previously initialized. There is no need to manually set the struct media_entity type and name fields.

A reference to the entity will be automatically acquired/released when the video device is opened/closed.

ioctls and locking

The V4L core provides optional locking services. The main service is the lock field in struct video_device, which is a pointer to a mutex. If you set this pointer, then that will be used by unlocked_ioctl to serialize all ioctls.

If you are using the :ref:`videobuf2 framework <vb2_framework>`, then there is a second lock that you can set: :c:type:`video_device`->queue->lock. If set, then this lock will be used instead of :c:type:`video_device`->lock to serialize all queuing ioctls (see the previous section for the full list of those ioctls).

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst, line 145); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst, line 145); *backlink*

Unknown interpreted text role "c:type".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst, line 145); *backlink*

Unknown interpreted text role "c:type".

The advantage of using a different lock for the queuing ioctls is that for some drivers (particularly USB drivers) certain commands such as setting controls can take a long time, so you want to use a separate lock for the buffer queuing ioctls. That way your `VIDIOC_DQBUF` doesn't stall because the driver is busy changing the e.g. exposure of the webcam.

Of course, you can always do all the locking yourself by leaving both lock pointers at `NULL`.

If you use the old :ref:`videobuf framework <vb_framework>` then you must pass the :c:type:`video_device`->lock to the videobuf queue initialize function: if videobuf has to wait for a frame to arrive, then it will temporarily unlock the lock and relock it afterwards. If your driver also waits in the code, then you should do the same to allow other processes to access the device node while the first process is waiting for something.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst, line 160); *backlink*

Unknown interpreted text role "ref".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst, line 160); *backlink*

Unknown interpreted text role "c:type".

In the case of :ref:`videobuf2 <vb2_framework>` you will need to implement the `wait_prepare()` and `wait_finish()` callbacks to unlock/lock if applicable. If you use the `queue->lock` pointer, then you can use the helper functions :c:func:`vb2_ops_wait_prepare` and :c:func:`vb2_ops_wait_finish`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media) v4l2-dev.rst`, **line 168**); ***backlink***

Unknown interpreted text role "ref".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media) v4l2-dev.rst`, **line 168**); ***backlink***

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media) v4l2-dev.rst`, **line 168**); ***backlink***

Unknown interpreted text role "c:func".

The implementation of a hotplug disconnect should also take the lock from :c:type:`video_device` before calling v4l2_device_disconnect. If you are also using :c:type:`video_device`->queue->lock, then you have to first lock :c:type:`video_device`->queue->lock followed by :c:type:`video_device`->lock. That way you can be sure no ioctl is running when you call :c:func:`v4l2_device_disconnect`.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media) v4l2-dev.rst`, **line 173**); ***backlink***

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media) v4l2-dev.rst`, **line 173**); ***backlink***

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media) v4l2-dev.rst`, **line 173**); ***backlink***

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media) v4l2-dev.rst`, **line 173**); ***backlink***

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media) v4l2-dev.rst`, **line 173**); ***backlink***

Unknown interpreted text role "c:func".

Video device registration

Next you register the video device with :c:func:`video_register_device`. This will create the character device for you.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media) v4l2-dev.rst`, **line 183**); ***backlink***

Unknown interpreted text role "c:func".

```
err = video_register_device(vdev, VFL_TYPE_VIDEO, -1);
if (err) {
        video_device_release(vdev); /* or kfree(my_vdev); */
        return err;
}
```

If the :c:type:`v4l2_device` parent device has a not `NULL` mdev field, the video device entity will be automatically registered with the media device.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 194); *backlink*

Unknown interpreted text role "c:type".

Which device is registered depends on the type argument. The following types exist:

:c:type:`vfl_devnode_type` **System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 203); *backlink*** **Unknown interpreted text role "c:type".**	**Device name**	**Usage**
`VFL_TYPE_VIDEO`	`/dev/videoX`	for video input/output devices
`VFL_TYPE_VBI`	`/dev/vbiX`	for vertical blank data (i.e. closed captions, teletext)
`VFL_TYPE_RADIO`	`/dev/radioX`	for radio tuners
`VFL_TYPE_SUBDEV`	`/dev/v4l-subdevX`	for V4L2 subdevices
`VFL_TYPE_SDR`	`/dev/swradioX`	for Software Defined Radio (SDR) tuners
`VFL_TYPE_TOUCH`	`/dev/v4l-touchX`	for touch sensors

The last argument gives you a certain amount of control over the device node number used (i.e. the X in `videoX`). Normally you will pass -1 to let the v4l2 framework pick the first free number. But sometimes users want to select a specific node number. It is common that drivers allow the user to select a specific device node number through a driver module option. That number is then passed to this function and video_register_device will attempt to select that device node number. If that number was already in use, then the next free device node number will be selected and it will send a warning to the kernel log.

Another use-case is if a driver creates many devices. In that case it can be useful to place different video devices in separate ranges. For example, video capture devices start at 0, video output devices start at 16. So you can use the last argument to specify a minimum device node number and the v4l2 framework will try to pick the first free number that is equal or higher to what you passed. If that fails, then it will just pick the first free number.

Since in this case you do not care about a warning about not being able to select the specified device node number, you can call the function :c:func:`video_register_device_no_warn` instead.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 232); *backlink*

Unknown interpreted text role "c:func".

Whenever a device node is created some attributes are also created for you. If you look in `/sys/class/video4linux` you see the devices. Go into e.g. `video0` and you will see 'name', 'dev_debug' and 'index' attributes. The 'name' attribute is the 'name' field of the video_device struct. The 'dev_debug' attribute can be used to enable core debugging. See the next section for more detailed information on this.

The 'index' attribute is the index of the device node: for each call to :c:func:`video_register_device()` the index is just increased by 1. The first video device node you register always starts with index 0.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, line 243); *backlink*

Unknown interpreted text role "c:func".

Users can setup udev rules that utilize the index attribute to make fancy device names (e.g. '`mpegX`' for MPEG video capture device nodes).

After the device was successfully registered, then you can use these fields:

- :c:type:`video_device`->vfl_type: the device type passed to :c:func:`video_register_device`.

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, line 252); *backlink*

  Unknown interpreted text role "c:type".

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, line 252); *backlink*

  Unknown interpreted text role "c:func".

- :c:type:`video_device`->minor: the assigned device minor number.

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, line 254); *backlink*

  Unknown interpreted text role "c:type".

- :c:type:`video_device`->num: the device node number (i.e. the X in `videoX`).

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, line 255); *backlink*

  Unknown interpreted text role "c:type".

- :c:type:`video_device`->index: the device index number.

  System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, line 257); *backlink*

  Unknown interpreted text role "c:type".

If the registration failed, then you need to call :c:func:`video_device_release` to free the allocated :c:type:`video_device` struct, or free your own struct if the :c:type:`video_device` was embedded in it. The `vdev->release()` callback will never be called if the registration failed, nor should you ever attempt to unregister the device if the registration failed.

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, line 259); *backlink*

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master) (Documentation) (driver-api) (media)v4l2-dev.rst`, line 259); *backlink*

Unknown interpreted text role "c:type".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\ (linux-master) (Documentation) (driver-api) (media) v4l2-dev.rst, line 259); *backlink*

Unknown interpreted text role "c:type".

video device debugging

The 'dev_debug' attribute that is created for each video, vbi, radio or swradio device in `/sys/class/video4linux/<devX>/` allows you to enable logging of file operations.

It is a bitmask and the following bits can be set:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\ (linux-master) (Documentation) (driver-api) (media) v4l2-dev.rst, line 275)

Unknown directive type "tabularcolumns".

```
.. tabularcolumns:: |p{5ex}|L|
```

Mask	Description
0x01	Log the ioctl name and error code. VIDIOC_(D)QBUF ioctls are only logged if bit 0x08 is also set.
0x02	Log the ioctl name arguments and error code. VIDIOC_(D)QBUF ioctls are only logged if bit 0x08 is also set.
0x04	Log the file operations open, release, read, write, mmap and get_unmapped_area. The read and write operations are only logged if bit 0x08 is also set.
0x08	Log the read and write file operations and the VIDIOC_QBUF and VIDIOC_DQBUF ioctls.
0x10	Log the poll file operation.
0x20	Log error and messages in the control operations.

Video device cleanup

When the video device nodes have to be removed, either during the unload of the driver or because the USB device was disconnected, then you should unregister them with:

:c:func:`video_unregister_device` (:c:type:`vdev <video_device>`);

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\ (linux-master) (Documentation) (driver-api) (media) v4l2-dev.rst, line 301); *backlink*

Unknown interpreted text role "c:func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\ (linux-master) (Documentation) (driver-api) (media) v4l2-dev.rst, line 301); *backlink*

Unknown interpreted text role "c:type".

This will remove the device nodes from sysfs (causing udev to remove them from `/dev`).

After :c:func:`video_unregister_device` returns no new opens can be done. However, in the case of USB devices some application might still have one of these device nodes open. So after the unregister all file operations (except release, of course) will return an error as well.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\ (linux-master) (Documentation) (driver-api) (media) v4l2-dev.rst, line 307); *backlink*

Unknown interpreted text role "c:func".

When the last user of the video device node exits, then the `vdev->release()` callback is called and you can do the final cleanup there.

Don't forget to cleanup the media entity associated with the video device if it has been initialized:

:c:func:`media_entity_cleanup <media_entity_cleanup>` (&vdev->entity);

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst, line 318); *backlink*

Unknown interpreted text role "c:func".

This can be done from the release callback.

helper functions

There are a few useful helper functions:

- file and :c:type:`video_device` private data

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst, line 329); *backlink*

Unknown interpreted text role "c:type".

You can set/get driver private data in the video_device struct using:

:c:func:`video_get_drvdata <video_get_drvdata>` (:c:type:`vdev <video_device>`);

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst, line 333); *backlink*

Unknown interpreted text role "c:func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst, line 333); *backlink*

Unknown interpreted text role "c:type".

:c:func:`video_set_drvdata <video_set_drvdata>` (:c:type:`vdev <video_device>`);

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst, line 336); *backlink*

Unknown interpreted text role "c:func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst, line 336); *backlink*

Unknown interpreted text role "c:type".

Note that you can safely call :c:func:`video_set_drvdata` before calling :c:func:`video_register_device`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst, line 339); *backlink*

Unknown interpreted text role "c:func".

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst, line 339); *backlink*

Unknown interpreted text role "c:func".

And this function:

:c:func:`video_devdata <video_devdata>` (struct file *file);

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 344); *backlink*

Unknown interpreted text role "c:func".

returns the video_device belonging to the file struct.

The :c:func:`video_devdata` function combines :c:func:`video_get_drvdata` with :c:func:`video_devdata`:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 349); *backlink*

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 349); *backlink*

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 349); *backlink*

Unknown interpreted text role "c:func".

:c:func:`video_drvdata <video_drvdata>` (struct file *file);

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 352); *backlink*

Unknown interpreted text role "c:func".

You can go from a :c:type:`video_device` struct to the v4l2_device struct using:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 355); *backlink*

Unknown interpreted text role "c:type".

```
struct v4l2_device *v4l2_dev = vdev->v4l2_dev;
```

- Device node name

The :c:type:`video_device` node kernel name can be retrieved using:

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 363); *backlink*

Unknown interpreted text role "c:type".

:c:func:`video_device_node_name <video_device_node_name>` (:c:type:`vdev <video_device>`);

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 365); *backlink*

Unknown interpreted text role "c:func".

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 365); *backlink*

Unknown interpreted text role "c:type".

The name is used as a hint by userspace tools such as udev. The function should be used where possible instead of accessing the video_device::num and video_device::minor fields.

video_device functions and data structures

System Message: ERROR/3 (`D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\(linux-master)(Documentation)(driver-api)(media)v4l2-dev.rst`, line 375)

Unknown directive type "kernel-doc".

```
.. kernel-doc:: include/media/v4l2-dev.h
```