V4L2 device instance

Each device instance is represented by a struct v4l2_device. Very simple devices can just allocate this struct, but most of the time you would embed this struct inside a larger struct.

You must register the device instance by calling:

:c:func:`v4l2_device_register <v4l2_device_register>` (dev, :c:type:`v4l2_dev <v4l2_device>`).

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\[linux-master][Documentation][driver-api][media]v4l2-device.rst, line 12); *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-device.rst, line 12); *backlink*

Unknown interpreted text role "c:type".

Registration will initialize the :c:type:`v4l2_device` struct. If the dev->driver_data field is `NULL`, it will be linked to :c:type:`v4l2_dev <v4l2_device>` argument.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\[linux-master][Documentation][driver-api][media]v4l2-device.rst, line 15); *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-device.rst, line 15); *backlink*

Unknown interpreted text role "c:type".

Drivers that want integration with the media device framework need to set dev->driver_data manually to point to the driver-specific device structure that embed the struct v4l2_device instance. This is achieved by a `dev_set_drvdata()` call before registering the V4L2 device instance. They must also set the struct v4l2_device mdev field to point to a properly initialized and registered :c:type:`media_device` instance.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\[linux-master][Documentation][driver-api][media]v4l2-device.rst, line 19); *backlink*

Unknown interpreted text role "c:type".

If :c:type:`v4l2_dev <v4l2_device>`->name is empty then it will be set to a value derived from dev (driver name followed by the bus_id, to be precise). If you set it up before calling :c:func:`v4l2_device_register` then it will be untouched. If dev is `NULL`, then you **must** setup :c:type:`v4l2_dev <v4l2_device>`->name before calling :c:func:`v4l2_device_register`.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\[linux-master][Documentation][driver-api][media]v4l2-device.rst, line 26); *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-device.rst, line 26); *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-device.rst, line 26); *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-device.rst, line 26); *backlink*

Unknown interpreted text role "c:func".

You can use :c:func:`v4l2_device_set_name` to set the name based on a driver name and a driver-global atomic_t instance. This will generate names like `ivtv0`, `ivtv1`, etc. If the name ends with a digit, then it will insert a dash: `cx18-0`, `cx18-1`, etc. This function returns the instance number.

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

Unknown interpreted text role "c:func".

The first `dev` argument is normally the `struct device` pointer of a `pci_dev`, `usb_interface` or `platform_device`. It is rare for dev to be `NULL`, but it happens with ISA devices or when one device creates multiple PCI devices, thus making it impossible to associate :c:type:`v4l2_dev <v4l2_device>` with a particular parent.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\[linux-master][Documentation][driver-api][media]v4l2-device.rst, line 38); *backlink*

Unknown interpreted text role "c:type".

You can also supply a `notify()` callback that can be called by sub-devices to notify you of events. Whether you need to set this depends on the sub-device. Any notifications a sub-device supports must be defined in a header in `include/media/subdevice.h`.

V4L2 devices are unregistered by calling:

:c:func:`v4l2_device_unregister` (:c:type:`v4l2_dev <v4l2_device>`).

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

Unknown interpreted text role "c:type".

If the dev->driver_data field points to :c:type:`v4l2_dev <v4l2_device>`, it will be reset to `NULL`. Unregistering will also automatically unregister all subdevs from the device.

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

Unknown interpreted text role "c:type".

If you have a hotpluggable device (e.g. a USB device), then when a disconnect happens the parent device becomes invalid. Since :c:type:`v4l2_device` has a pointer to that parent device it has to be cleared as well to mark that the parent is gone. To do this call:

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\[linux-master][Documentation][driver-api][media]v4l2-device.rst, line 58); *backlink*

Unknown interpreted text role "c:type".

:c:func:`v4l2_device_disconnect` (:c:type:`v4l2_dev <v4l2_device>`).

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\[linux-master][Documentation][driver-api][media]v4l2-device.rst, line 63); *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-device.rst, line 63); *backlink*

Unknown interpreted text role "c:type".

This does *not* unregister the subdevs, so you still need to call the :c:func:`v4l2_device_unregister` function for that. If your driver is not hotpluggable, then there is no need to 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-device.rst, line 66); *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-device.rst, line 66); *backlink*

Unknown interpreted text role "c:func".

Sometimes you need to iterate over all devices registered by a specific driver. This is usually the case if multiple device drivers use the same hardware. E.g. the ivtvfb driver is a framebuffer driver that uses the ivtv hardware. The same is true for alsa drivers for example.

You can iterate over all registered devices as follows:

```
static int callback(struct device *dev, void *p)
{
        struct v4l2_device *v4l2_dev = dev_get_drvdata(dev);

        /* test if this device was inited */
        if (v4l2_dev == NULL)
                return 0;
        ...
        return 0;
}

int iterate(void *p)
{
        struct device_driver *drv;
        int err;

        /* Find driver 'ivtv' on the PCI bus.
        pci_bus_type is a global. For USB buses use usb_bus_type. */
        drv = driver_find("ivtv", &pci_bus_type);
        /* iterate over all ivtv device instances */
        err = driver_for_each_device(drv, NULL, p, callback);
        put_driver(drv);
        return err;
}
```

Sometimes you need to keep a running counter of the device instance. This is commonly used to map a device instance to an index of a module option array.

The recommended approach is as follows:

```
static atomic_t drv_instance = ATOMIC_INIT(0);

static int drv_probe(struct pci_dev *pdev, const struct pci_device_id *pci_id)
{
        ...
        state->instance = atomic_inc_return(&drv_instance) - 1;
}
```

If you have multiple device nodes then it can be difficult to know when it is safe to unregister :c:type:`v4l2_device` for hotpluggable devices. For this purpose :c:type:`v4l2_device` has refcounting support. The refcount is increased whenever :c:func:`video_register_device` is called and it is decreased whenever that device node is released. When the refcount reaches zero, then the :c:type:`v4l2_device` release() callback is called. You can do your final cleanup there.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\[linux-master][Documentation][driver-api][media]v4l2-device.rst, **line 119**); *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-device.rst, **line 119**); *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-device.rst, **line 119**); *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-device.rst, **line 119**); *backlink*

Unknown interpreted text role "c:type".

If other device nodes (e.g. ALSA) are created, then you can increase and decrease the refcount manually as well by calling:

:c:func:`v4l2_device_get` (:c:type:`v4l2_dev <v4l2_device>`).

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\[linux-master][Documentation][driver-api][media]v4l2-device.rst, **line 130**); *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-device.rst, **line 130**); *backlink*

Unknown interpreted text role "c:type".

or:

:c:func:`v4l2_device_put` (:c:type:`v4l2_dev <v4l2_device>`).

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-master\Documentation\driver-api\media\[linux-master][Documentation][driver-api][media]v4l2-device.rst, **line 135**); *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-device.rst, **line 135**); *backlink*

Unknown interpreted text role "c:type".

Since the initial refcount is 1 you also need to call :c:func:`v4l2_device_put` in the `disconnect()` callback (for USB devices) or in the `remove()` callback (for e.g. PCI devices), otherwise the refcount will never reach 0.

System Message: ERROR/3 (D:\onboarding-resources\sample-onboarding-resources\linux-

master\Documentation\driver-api\media\[linux-master][Documentation][driver-api][media]v4l2-device.rst, line 138); *backlink*

Unknown interpreted text role "c:func".

v4l2_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-device.rst, line 146)

Unknown directive type "kernel-doc".

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