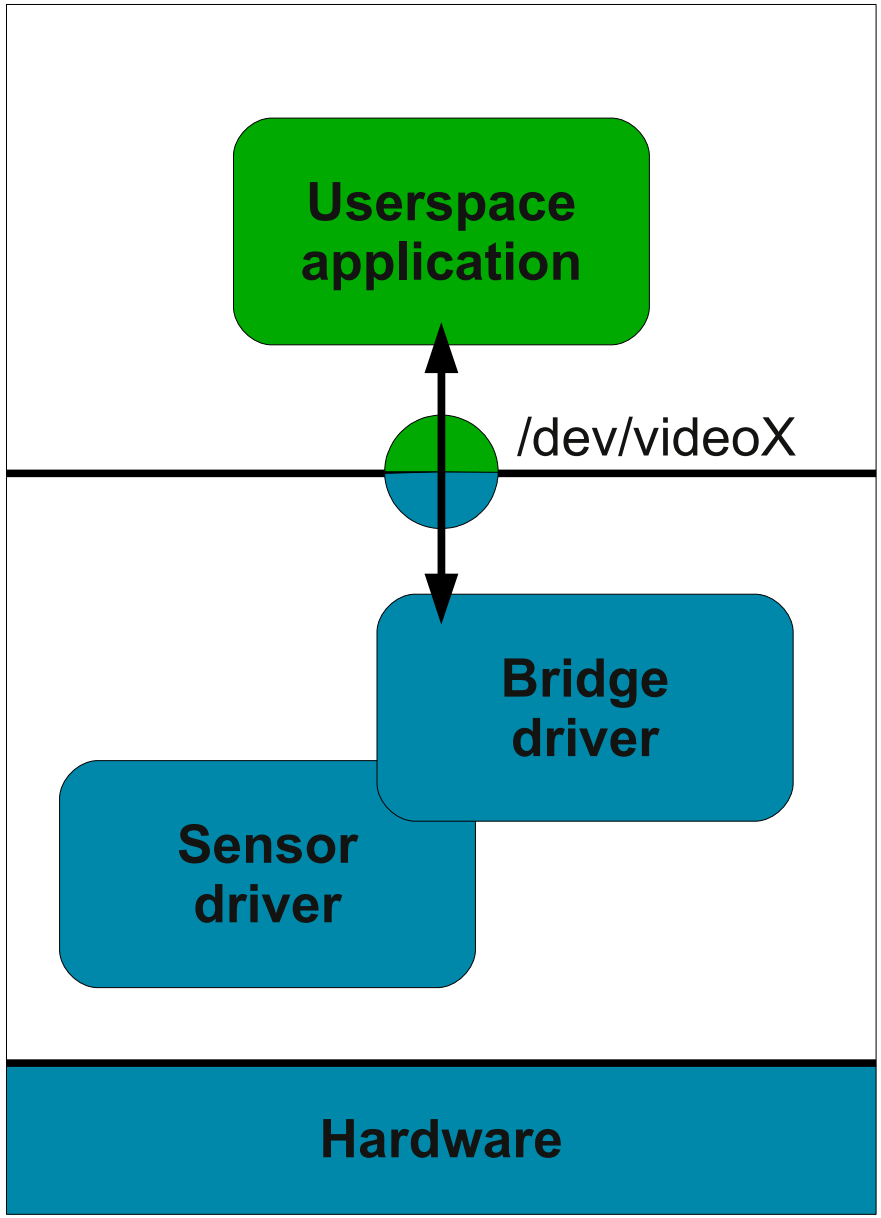
Media controller

Goals, architecture and roadmap

V4L2 Helsinki Summit 2010-06-14

Laurent Pinchart
laurent.pinchart@ideasonboard.com



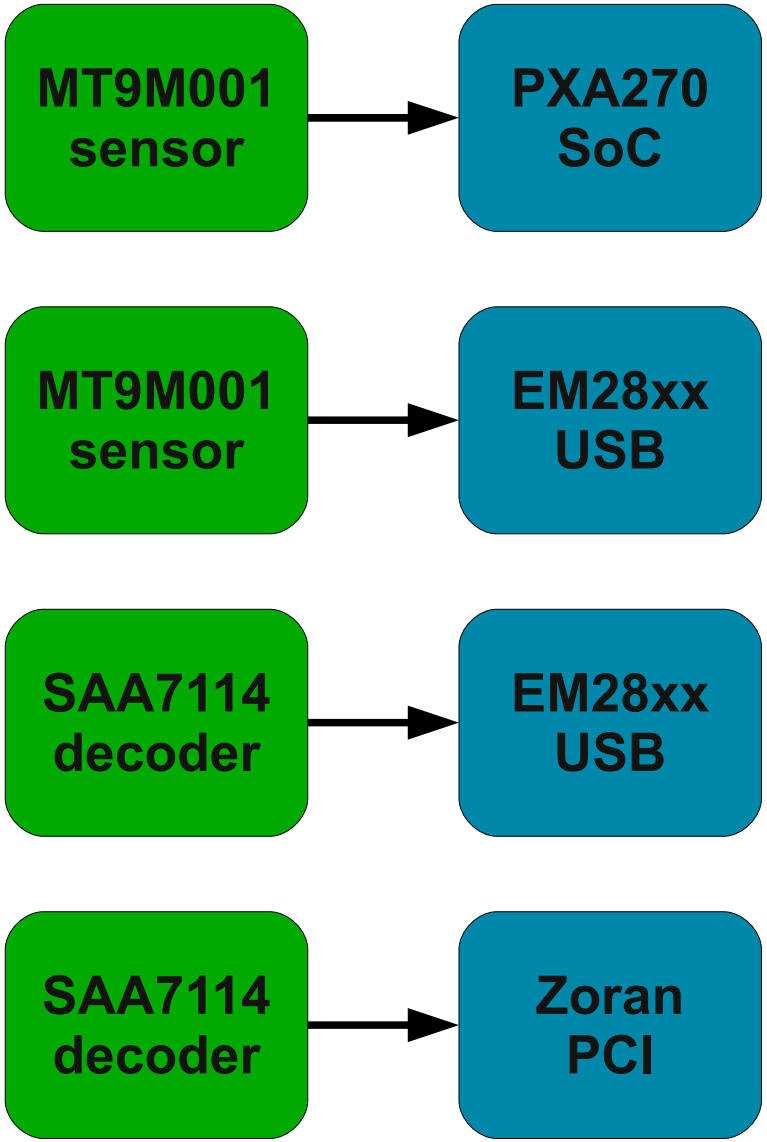
Embedded SoC camera

- soc_camera
- v4l2_device
- v4l2_subdev

Userspace library

- Format conversion
- Post-processing

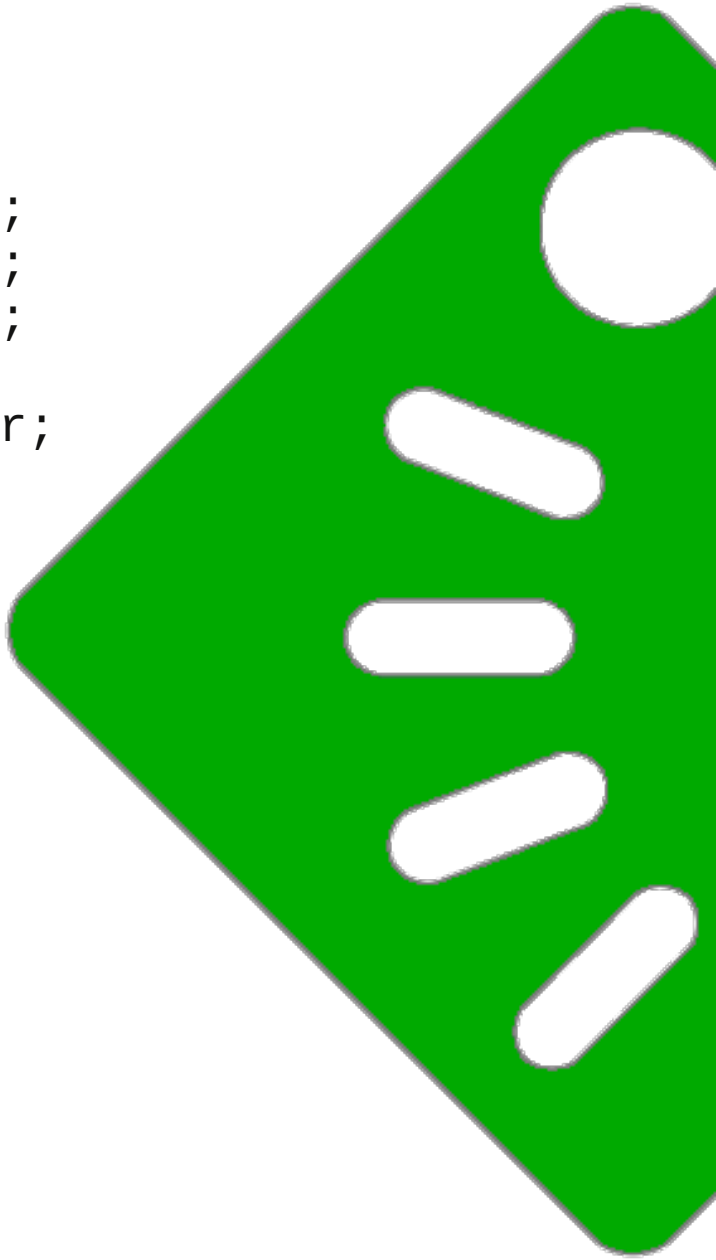
Embedded camera



- In-kernel functional abstraction layer developed by Hans Verkuil
- Designed for on-board external devices (sensors, tuners, audio codecs, …)
- Reusability, Reusability, Reusability

V4L2 subdevice

```
struct v4l2_subdev_ops {
    const struct v4l2_subdev_core_ops   *core;
    const struct v4l2_subdev_tuner_ops  *tuner;
    const struct v4l2_subdev_audio_ops  *audio;
    const struct v4l2_subdev_video_ops  *video;
    const struct v4l2_subdev_ir_ops     *ir;
    const struct v4l2_subdev_sensor_ops *sensor;
};
```

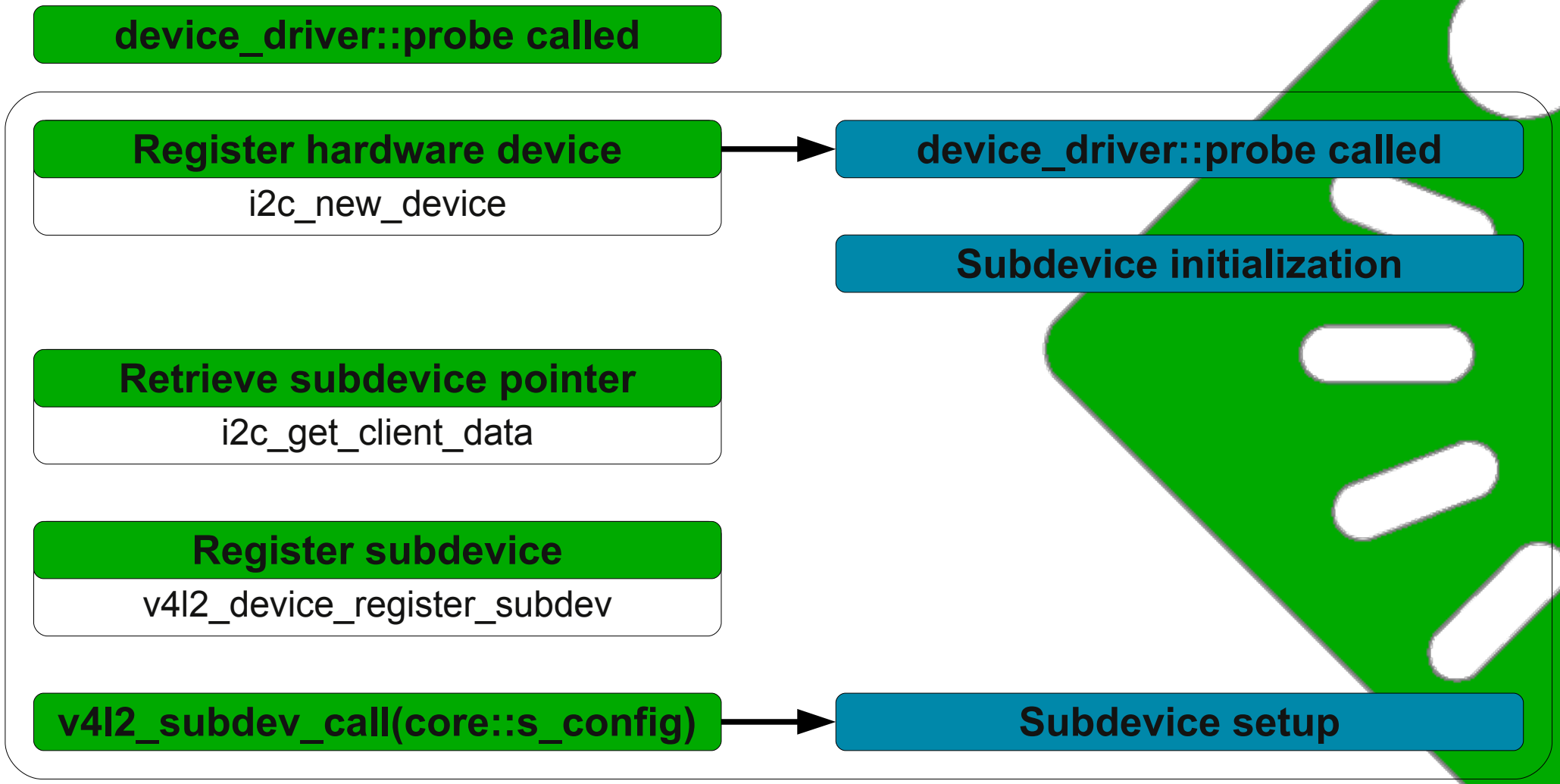


- Hardware independent
- Bus type independent

V4L2 subdevice operations

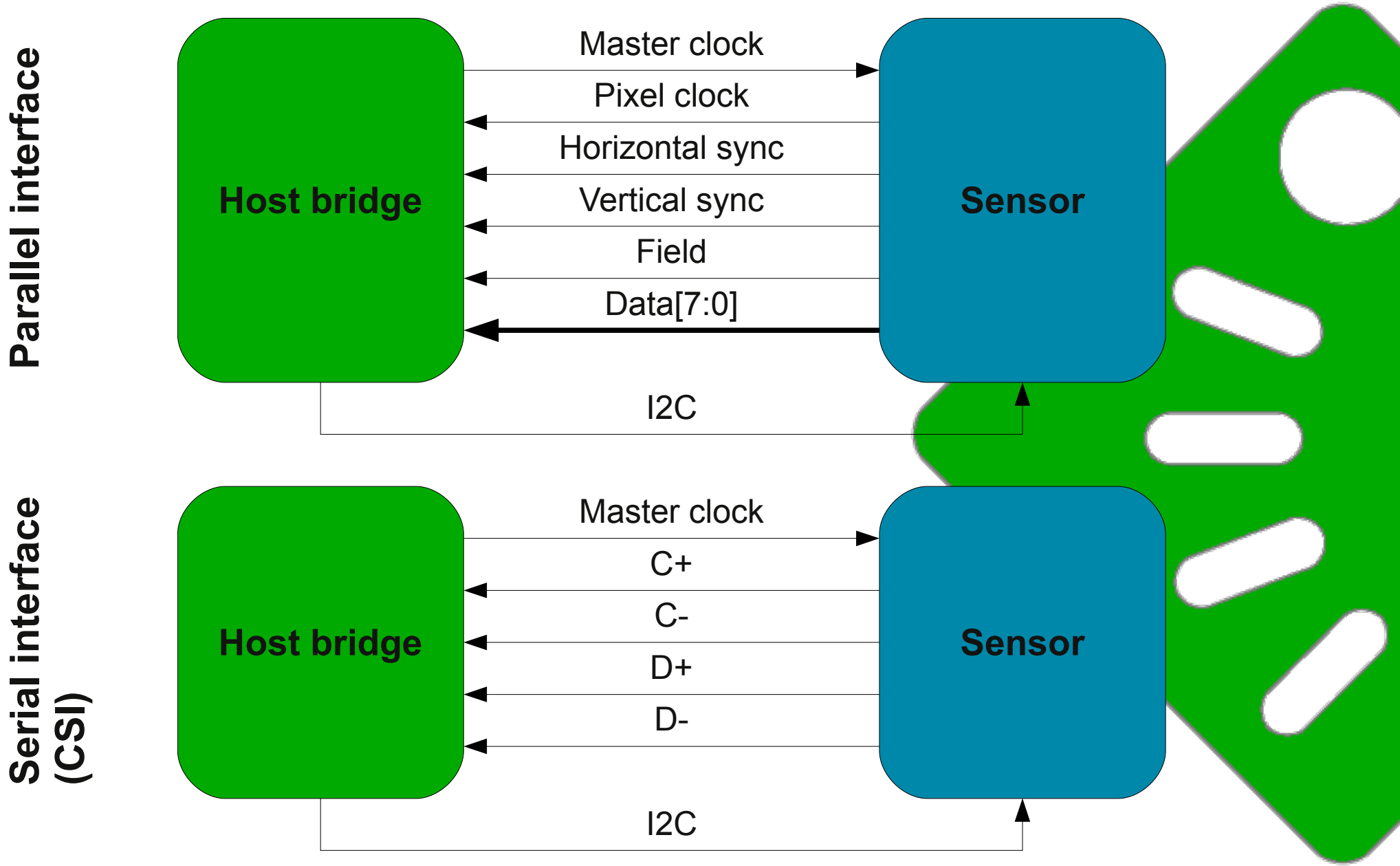
V4L2 device

V4L2 subdevice

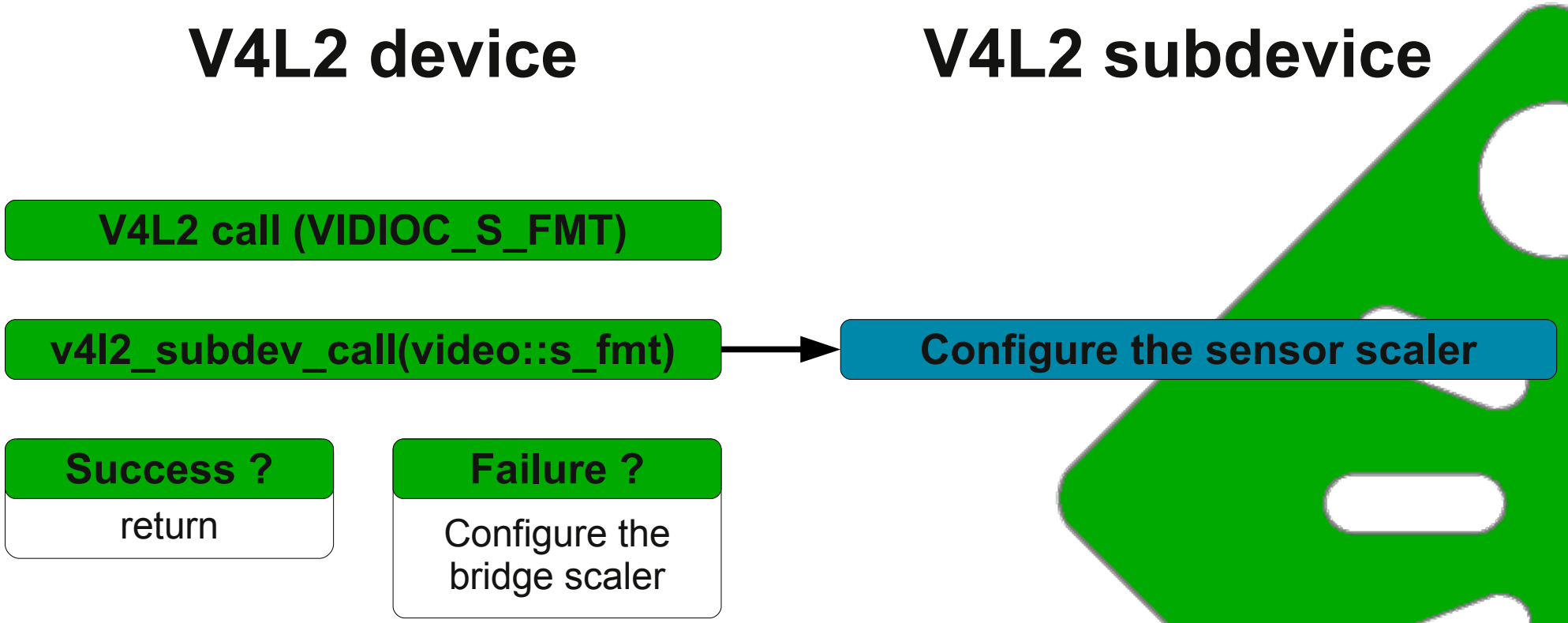


v4l2_i2c_new_subdev_board

V4L2 subdevice registration

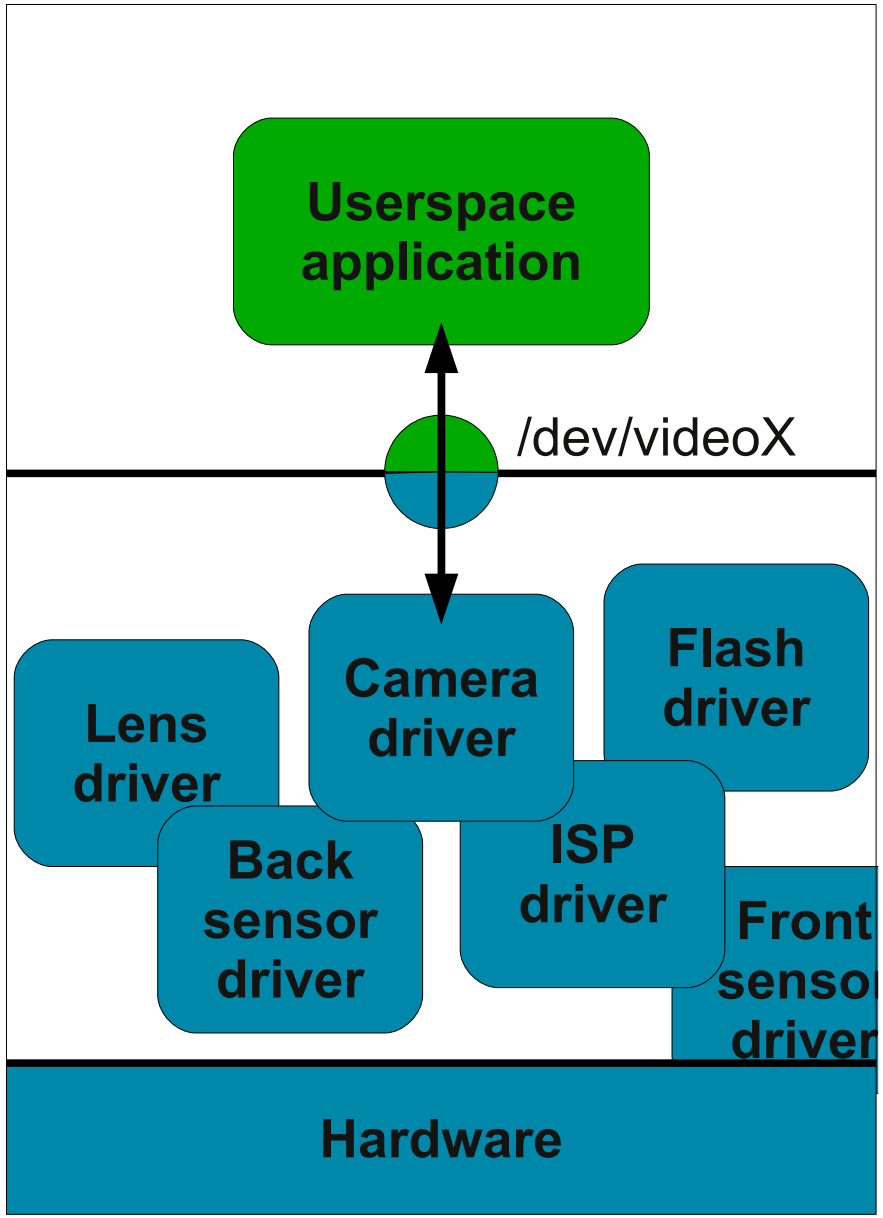

Parallel interface
Host bridge
Master clock
Pixel clock
Horizontal sync
Vertical sync
Field
Data[7:0]
Sensor
I2C
Serial interface (CSI)
Host bridge
Master clock
C+
C-
D+
D-
Sensor
I2C

SoC Camera



- **How do we decide where to perform scaling ?**
  - On the sensor side for higher frame rates ?
  - On the host side for better quality ?

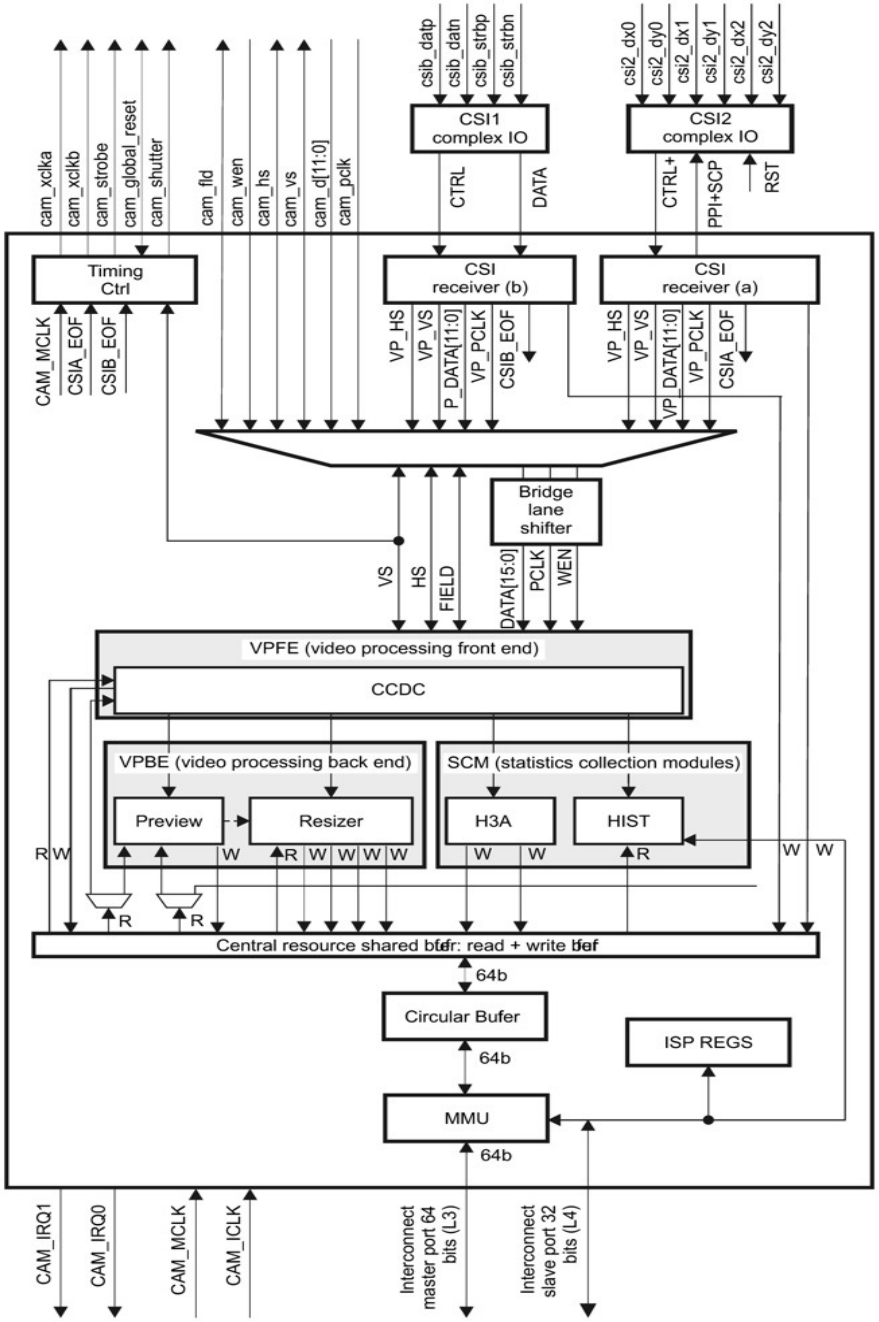
V4L2 subdevice usage



Highly complex devices

- Multiple inputs
- Multiple streams
- Configurable pipeline

Embedded mess

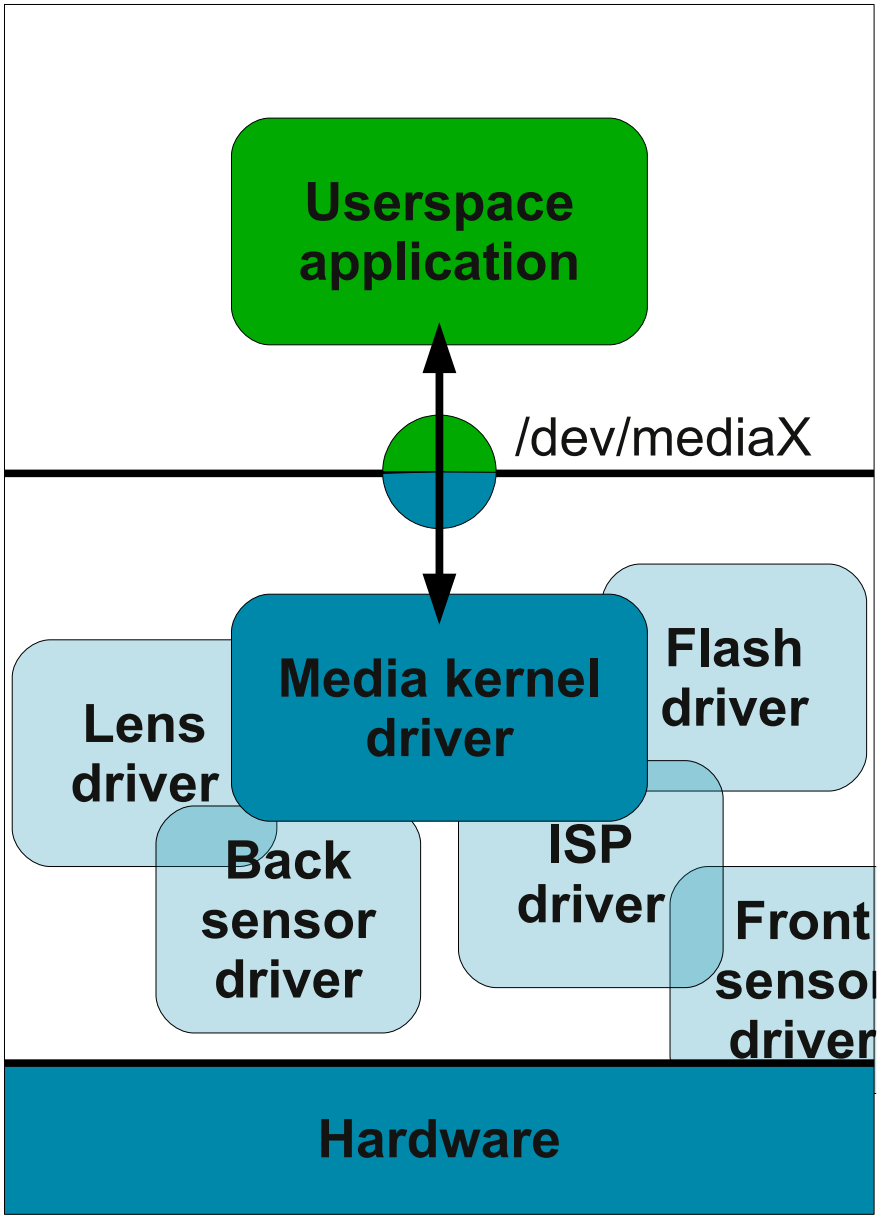


OMAP3430 ISP

- Reconfigurable pipeline
- Parallel processing
- Memory-to-memory paths
- Fine-grain parameters

How do we handle the zillion configuration options through a single video device ?

OMAP3430 ISP



What ?

- Just a device. Similar to v4l2_device, but more abstract.

Why ?

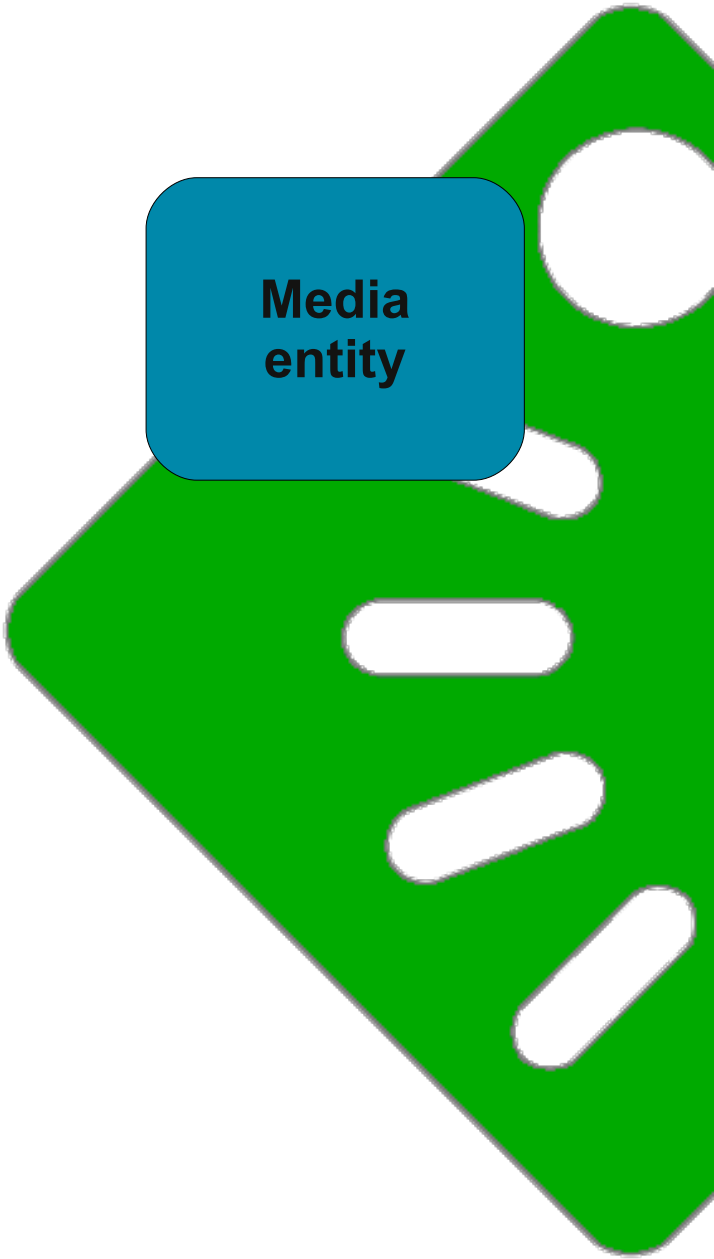
- To let userspace applications have fine grain control over all the media device parameters.

Media controller – What and why

How ?

- Expose the media device topology to userspace as a graph of building blocks called **entities** connected through **pads**.
- Activate/deactivate **links** from userspace.
- Give access to entities **internal parameters** through read/write/ioctl calls.
- Configure image **streaming parameters** at each pad.

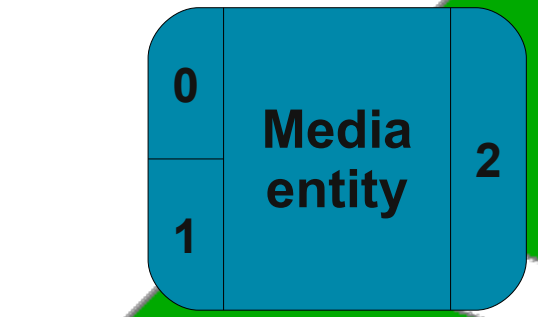
```
struct media_entity
{
    u32 id;
    const char *name;
    u32 type;
    u32 subtype;
    ...
};
```



- media_entity::type
  - MEDIA_ENTITY_TYPE_NODE
  - MEDIA_ENTITY_TYPE_SUBDEV

Media entity

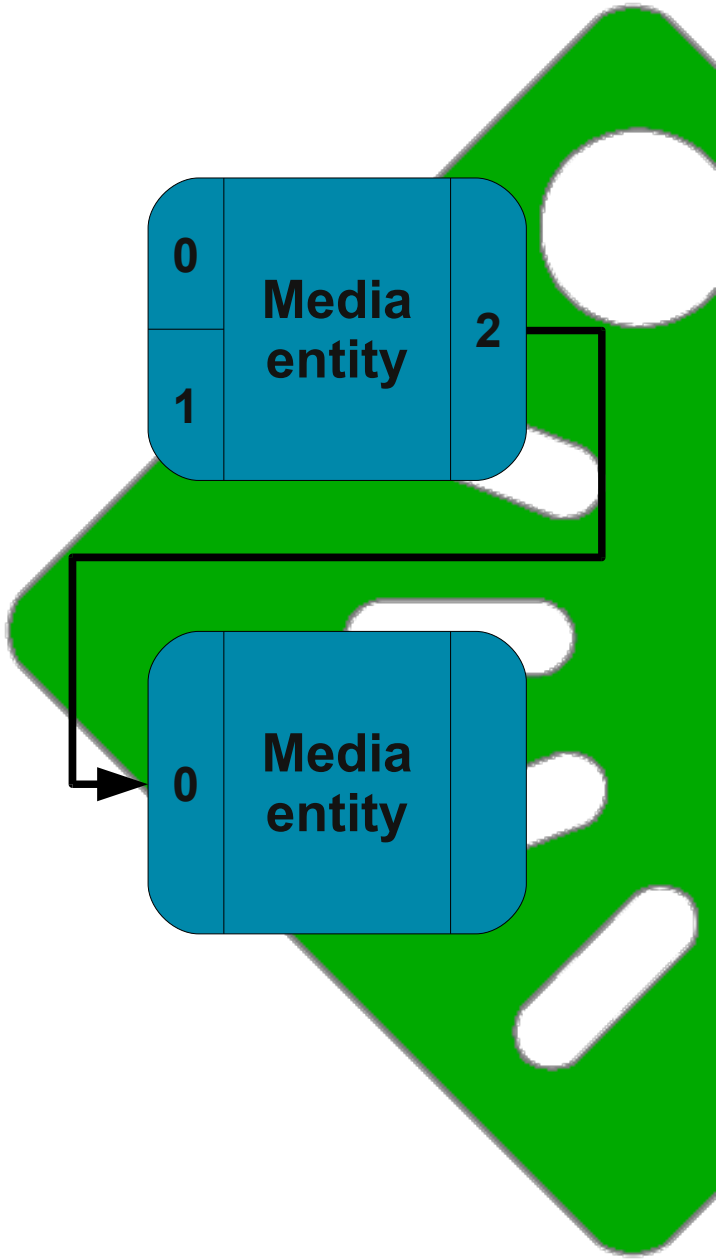
```
struct media_entity
{
    ...
    u8 num_pads;
    struct media_entity_pad *pads;
    ...
};

struct media_entity_pad
{
    u32 type;
    u32 index;
};
```



- media_entity_pad::type
  - MEDIA_PAD_TYPE_INPUT
  - MEDIA_PAD_TYPE_OUTPUT

Media entity - Pads

```
struct media_entity
{
    ...
    u32 num_links;
    struct media_entity_link *links;
    ...
};

struct media_entity_link
{
    struct media_entity_pad *source;
    struct media_entity_pad *sink;
    u32 flags;
};
```



- media_entity_link::flags
  - MEDIA_LINK_FLAG_ACTIVE
  - MEDIA_LINK_FLAG_IMMUTABLE

Media entity - Links

Media entity

Initialize entity

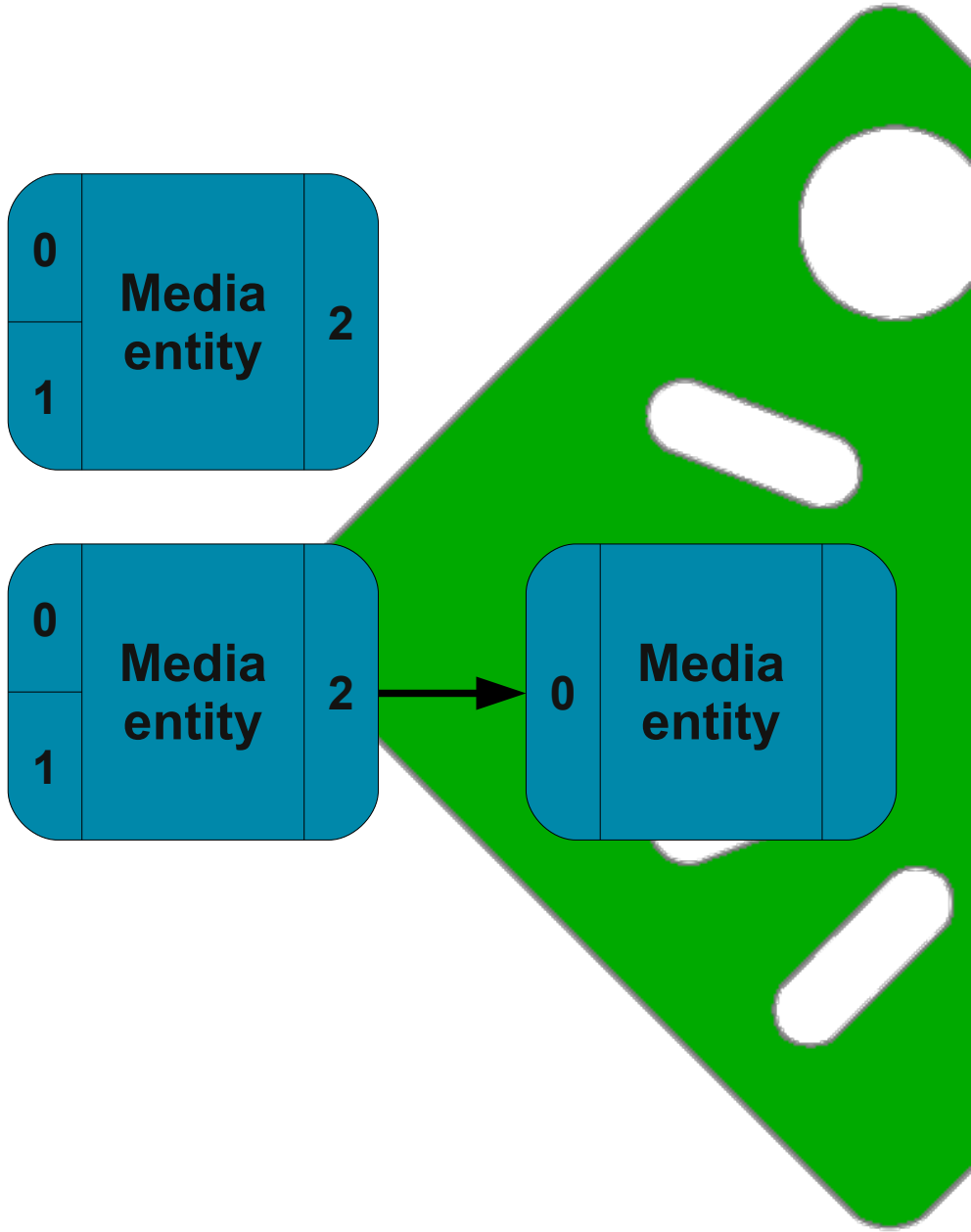
media_entity_init

Create links

media_entity_create_link

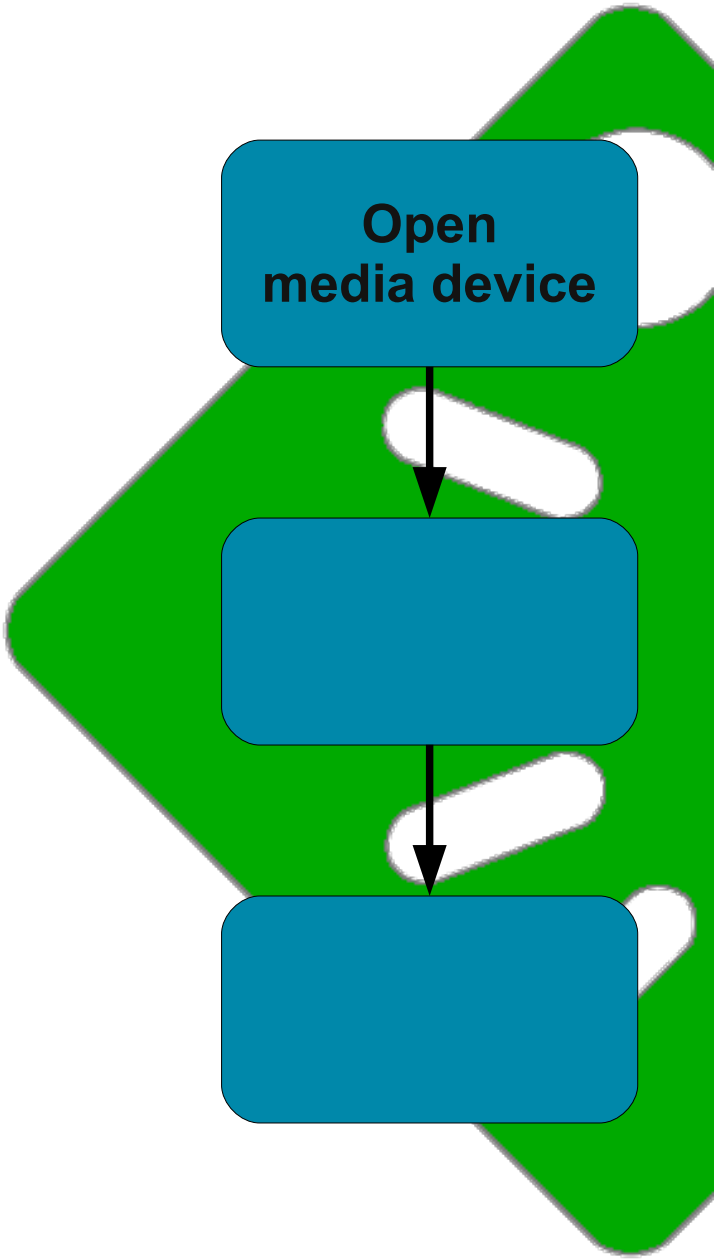
Register entity

media_device_register_entity



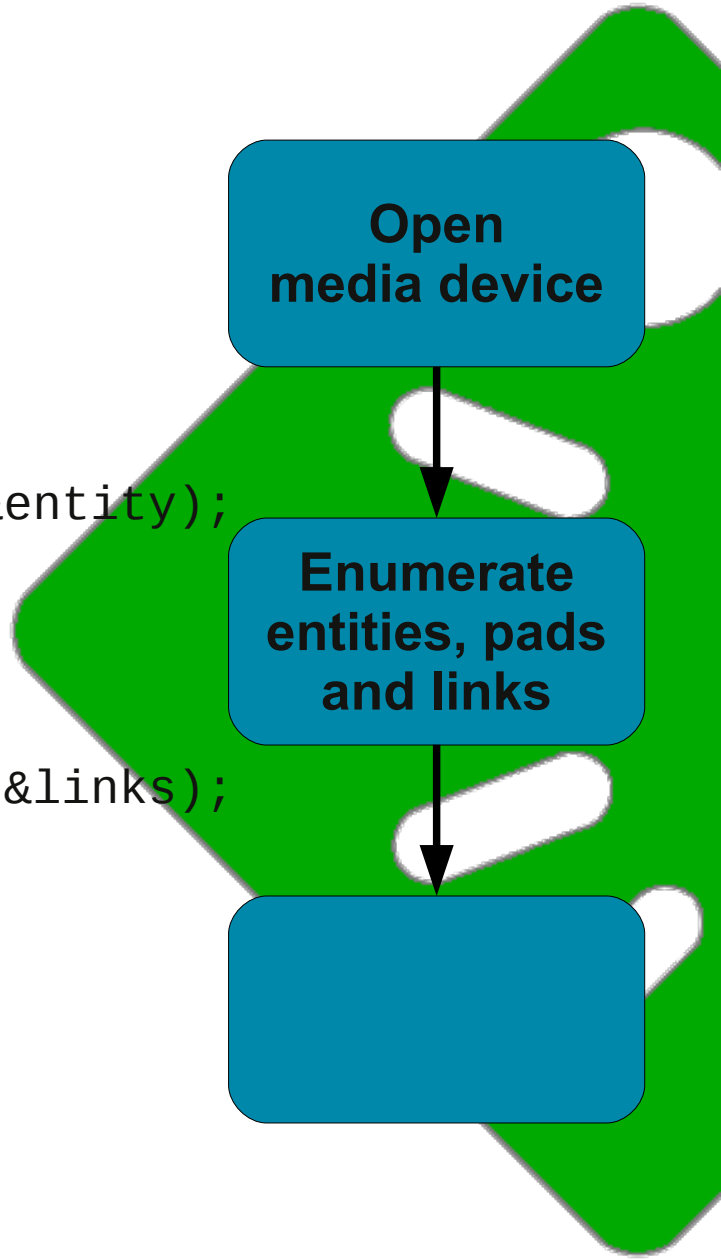
Media entity registration

```
int fd;

fd = open(“/dev/media0”, O_RDWR);
```

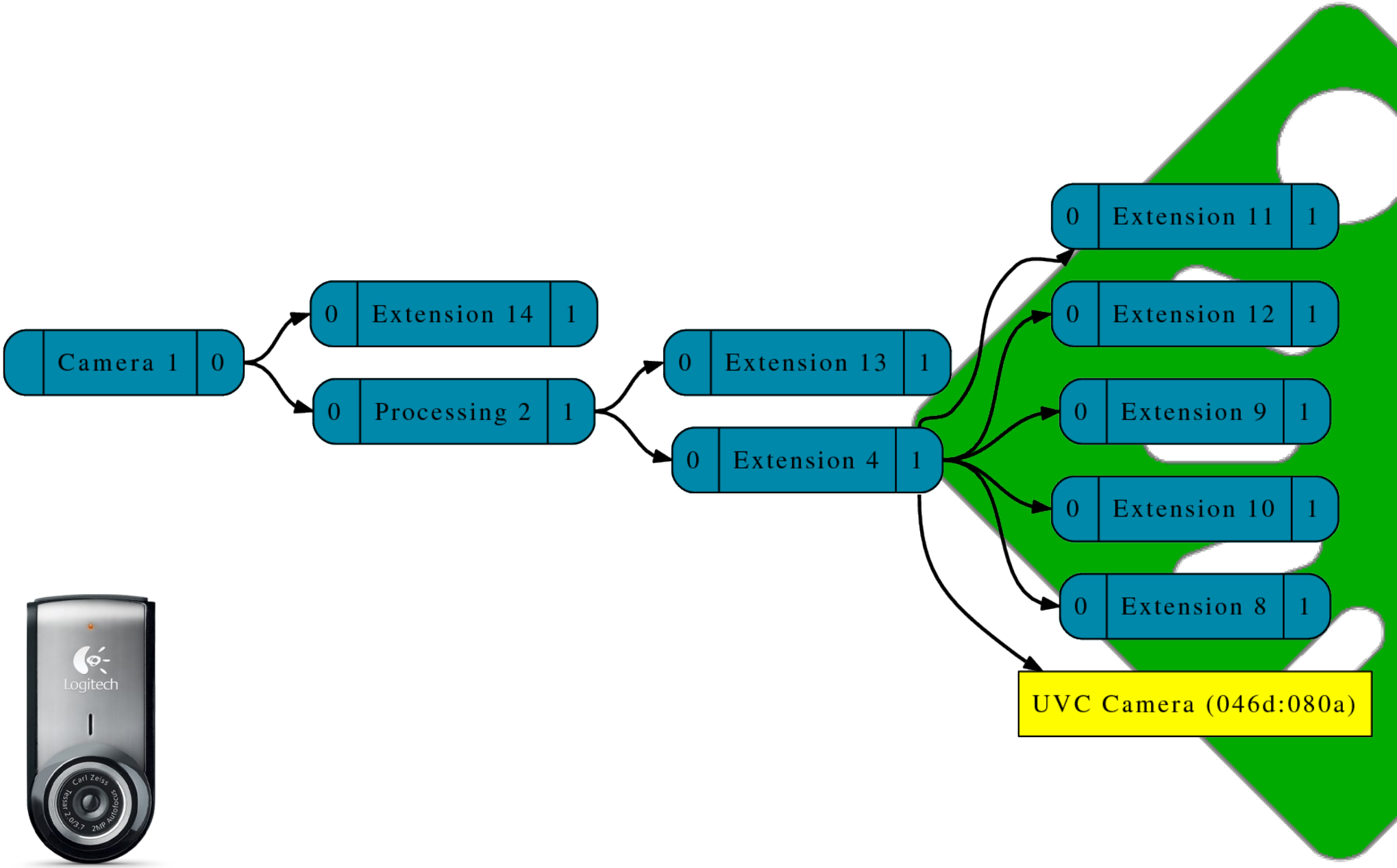


Media controller – Userspace API

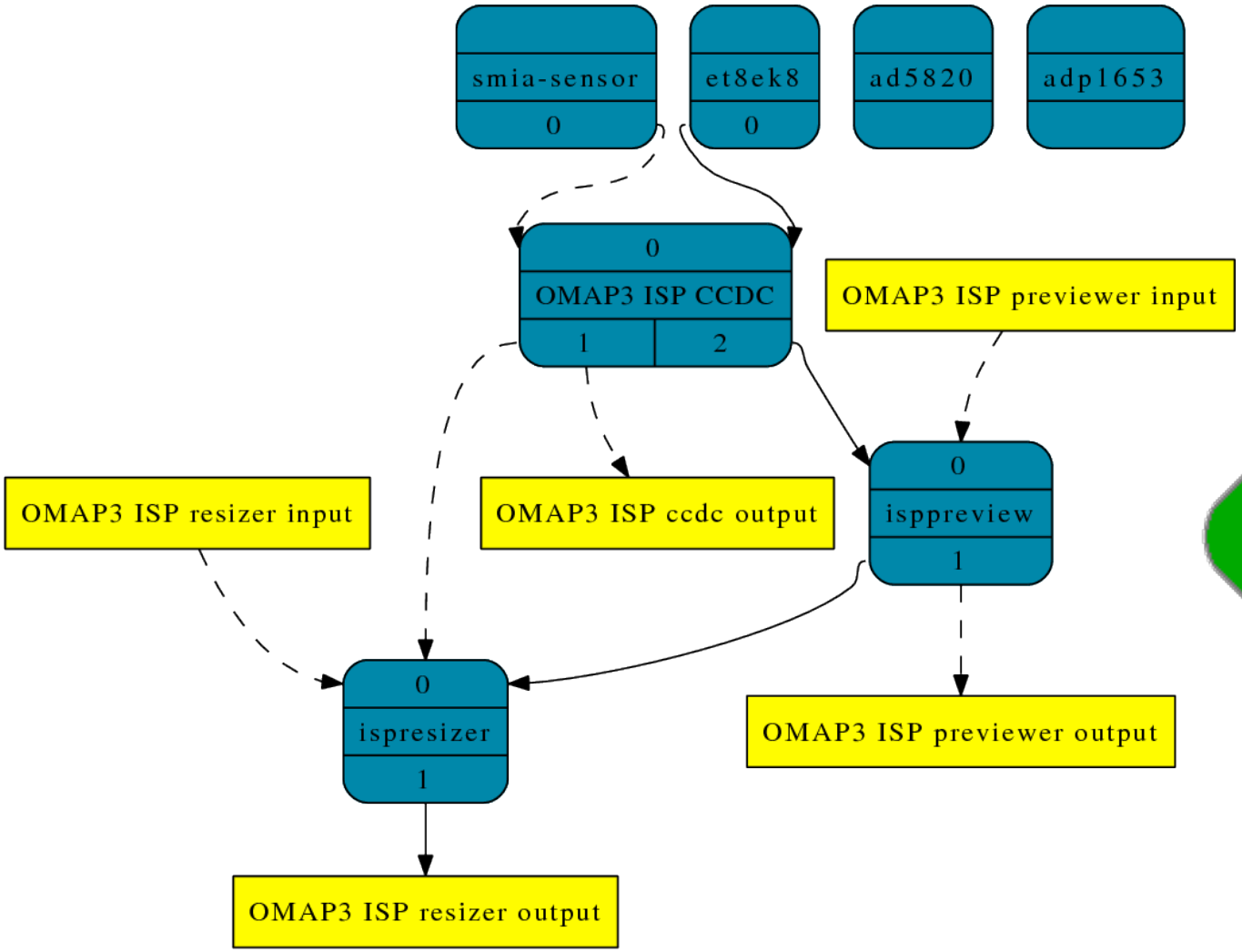
```
int fd;

fd = open(“/dev/media0”, O_RDWR);

while (1) {
    struct media_user_entity entity;
    struct media_user_links links;

    ret = ioctl(fd, MEDIA_IOC_ENUM_ENTITIES, &entity);
    if (ret < 0)
        break;

    while (1) {
        ret = ioctl(fd, MEDIA_IOC_ENUM_LINKS, &links);
        if (ret < 0)
            break;
    }
}
```



Media controller – Userspace API



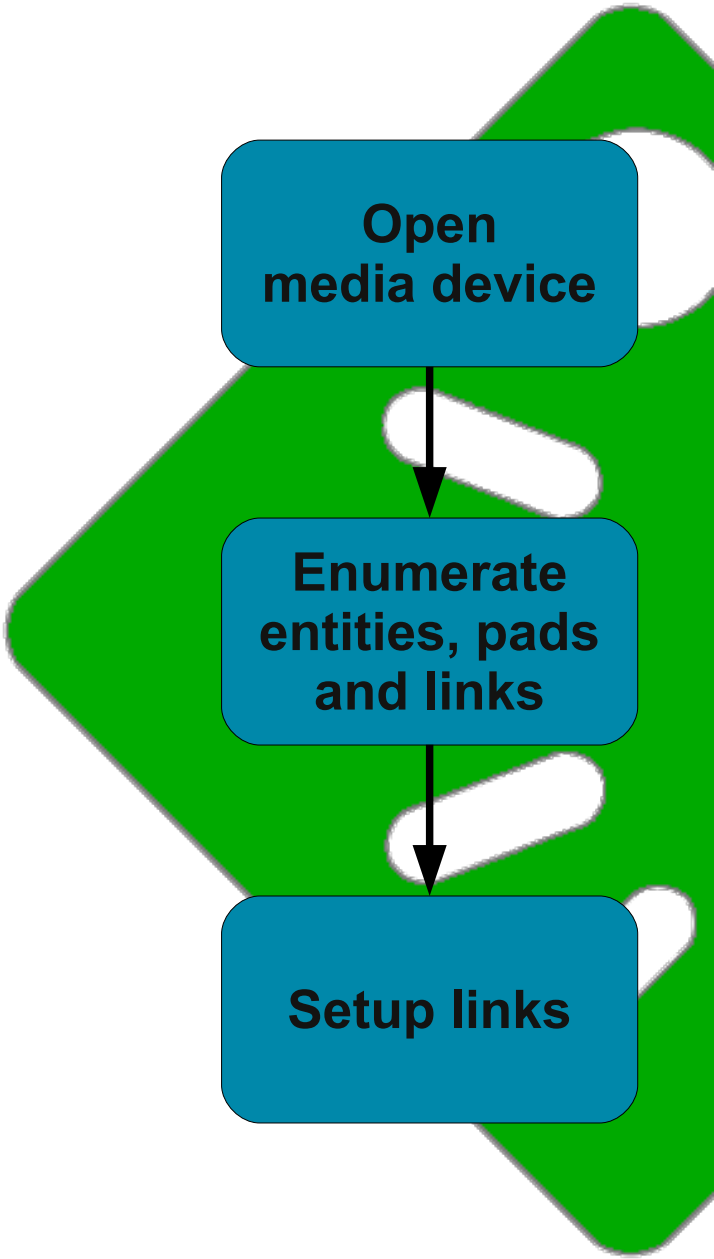
Logitech Portable Webcam C905


smia-sensor
0
et8ek8
0
ad5820
adp1653
0
OMAP3 ISP CCDC
1
2
OMAP3 ISP previewer input
OMAP3 ISP resizer input
OMAP3 ISP ccdc output
0
isppreview
1
0
ispresizer
1
OMAP3 ISP previewer output
OMAP3 ISP resizer output

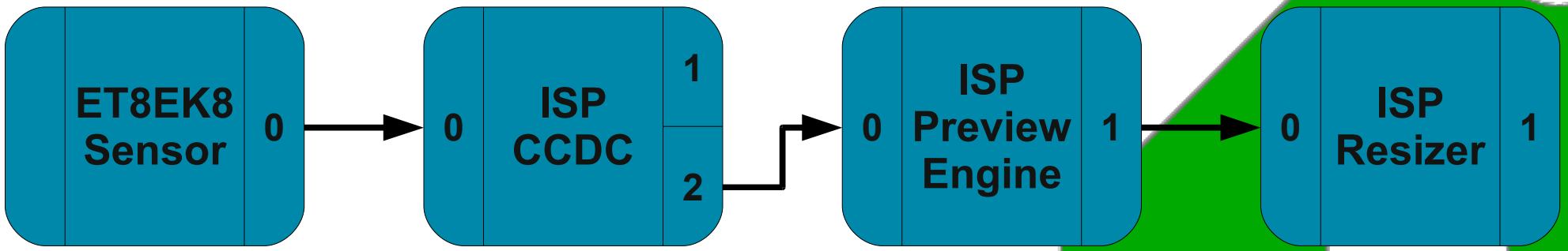

NOKIA
Operator
Marina
Maemo

Nokia N900

```
struct media_user_link link;

link.source.entity = OMAP3_ISP_ENTITY_CCDC;
link.source.index = 2;
link.sink.entity = OMAP3_ISP_ENTITY_PREVIEW;
link.sink.index = 0;
link.flags = 0;

ioctl(fd, MEDIA_IOC_SETUP_LINK, &link);

link.source.entity = OMAP3_ISP_ENTITY_CCDC;
link.source.index = 1;
link.sink.entity = OMAP3_ISP_ENTITY_CCDC_OUT;
link.sink.index = 0;
link.flags = MEDIA_LINK_FLAG_ACTIVE;


ioctl(fd, MEDIA_IOC_SETUP_LINK, &link);
```



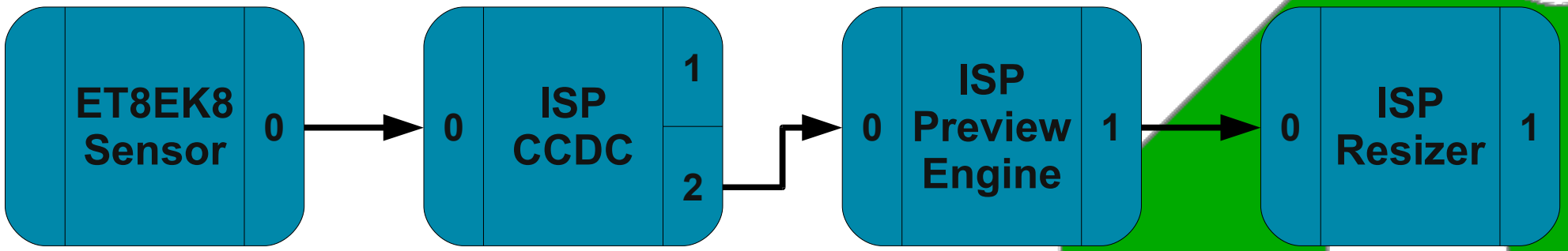
Media controller – Userspace API



- White balance
- Faulty pixels correction
- White balance
- Faulty pixels correction

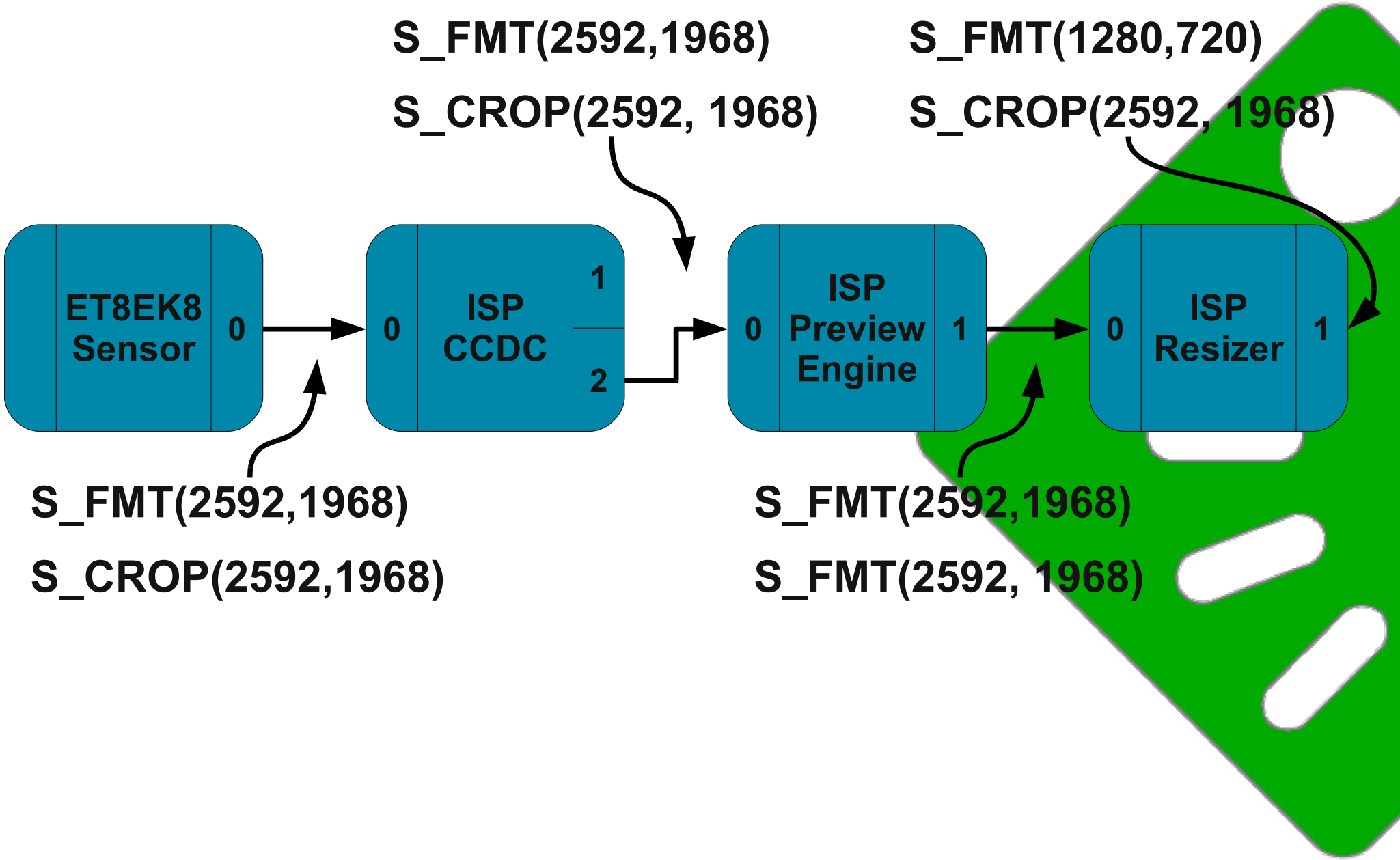
The S_CTRL API is not up to the job.

OMAP3430 ISP - Controls

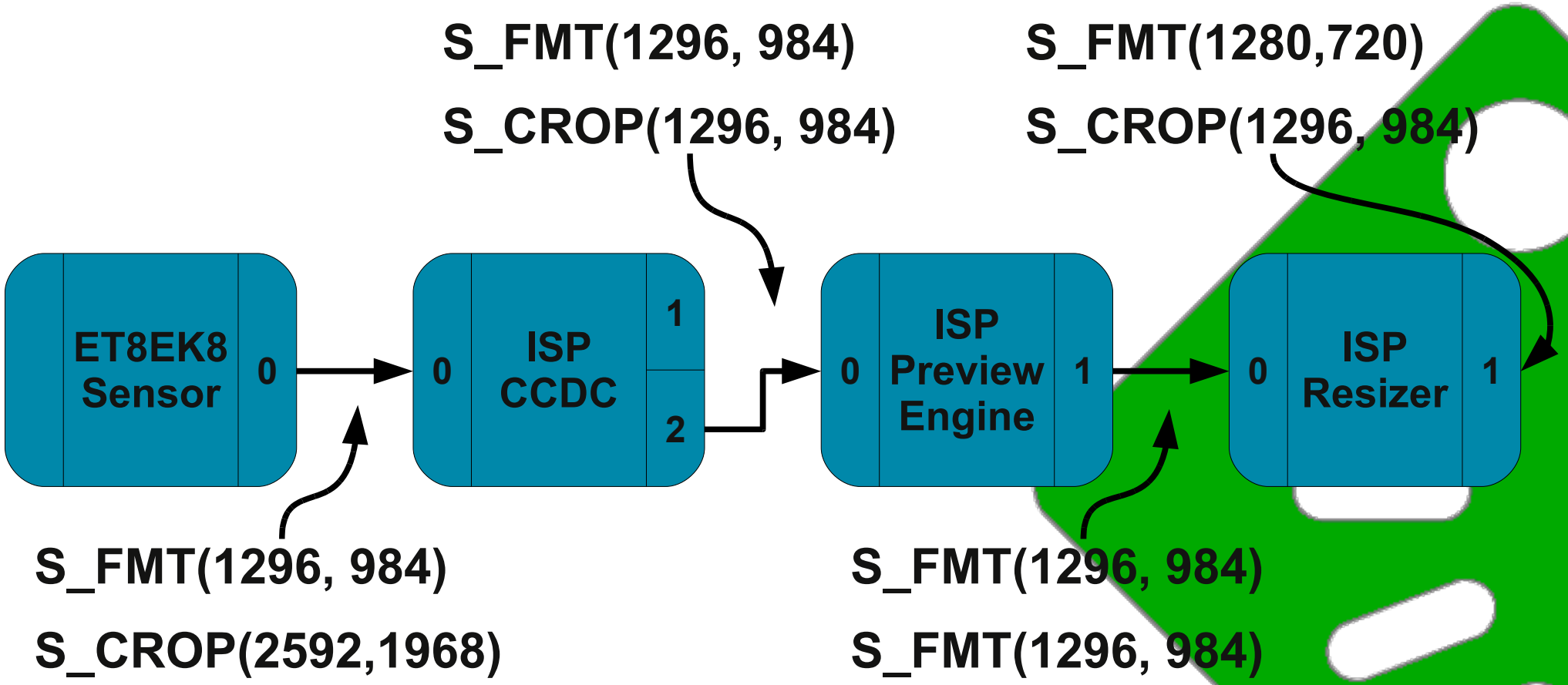


- Binning
- Horizontal averaging
- Polyphase filter

OMAP3430 ISP - Resizing

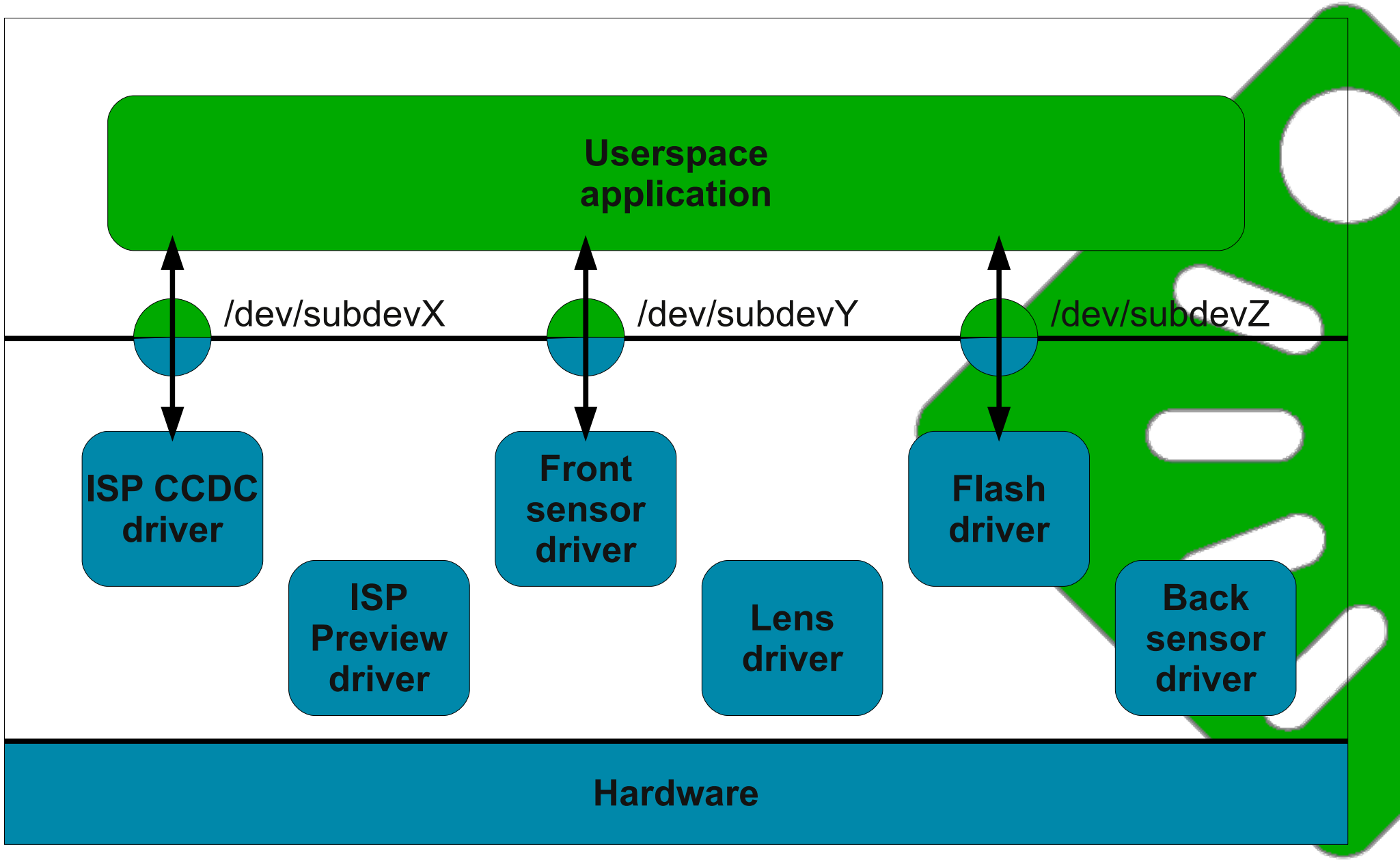


OMAP3430 ISP – High quality



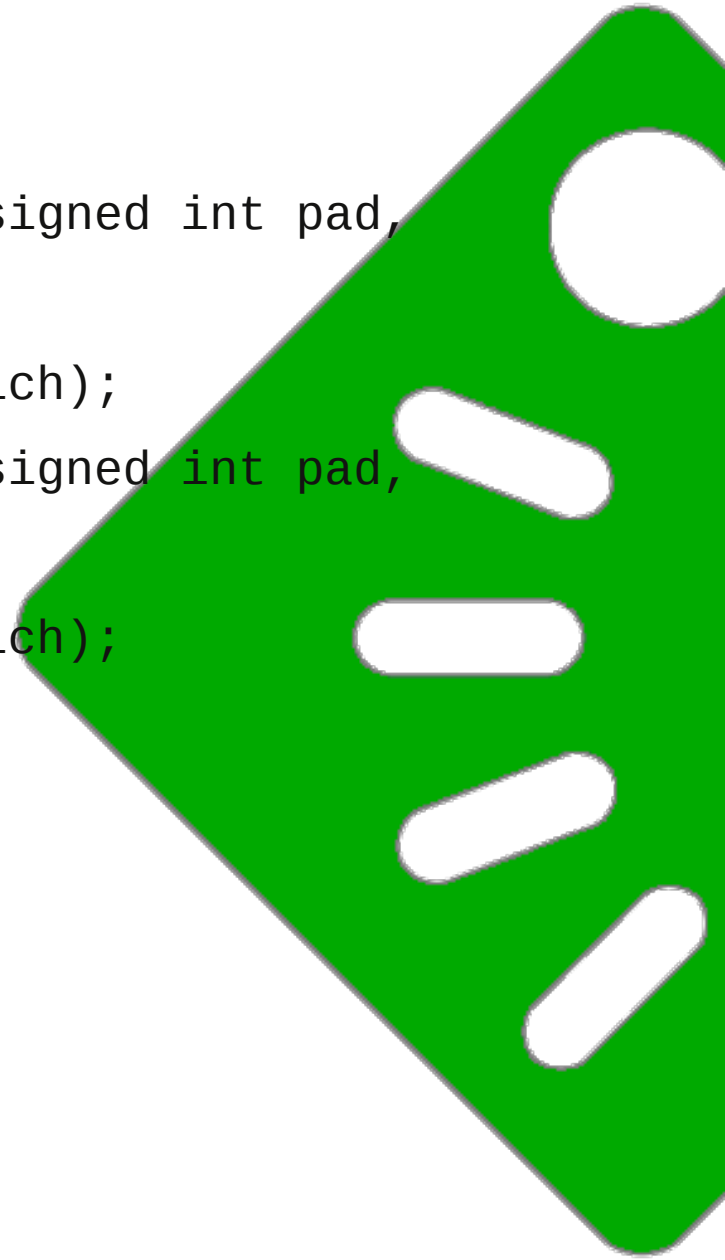
The S_FMT/S_CROP API is not up to the job.

OMAP3430 ISP – High speed



V4L2 subdev userspace API

```
struct v4l2_subdev_pad_ops {
    ...
    int (*get_fmt)(struct v4l2_subdev *sd, unsigned int pad,
                   struct v4l2_format *fmt,
                   enum v4l2_subdev_format which);
    int (*set_fmt)(struct v4l2_subdev *sd, unsigned int pad,
                   struct v4l2_format *fmt,
                   enum v4l2_subdev_format which);
    ....
};

struct v4l2_subdev_ops {
    ...
    const struct v4l2_subdev_pad_ops  *pad;
};
```



V4L2 subdevice pad operations

Is this V4L3 ?

- No, V4L2 is still alive and well
- Best effort to provide V4L2-only compatibility for existing applications (API and ABI)
- Advanced features will require Media Controller

OMAP3430 ISP

- Default pipeline through /dev/video0 ?
- Limited set of resolutions, limited set of controls

V4L2 plain API

Retrieve statistics

V4L2 events API

Compute parameters

Host-side software algorithm

Set parameters

VIDIOC_S_CTRL

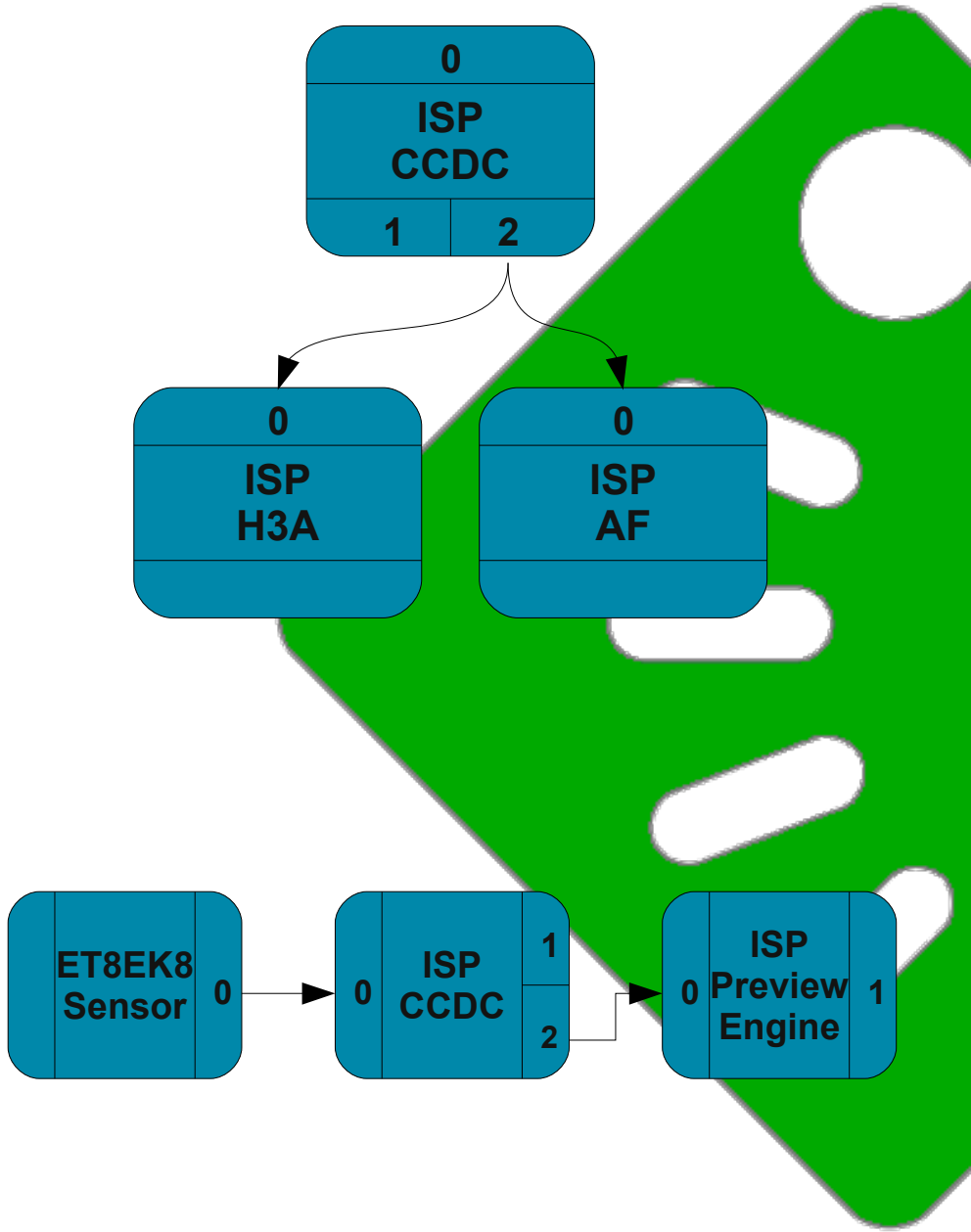
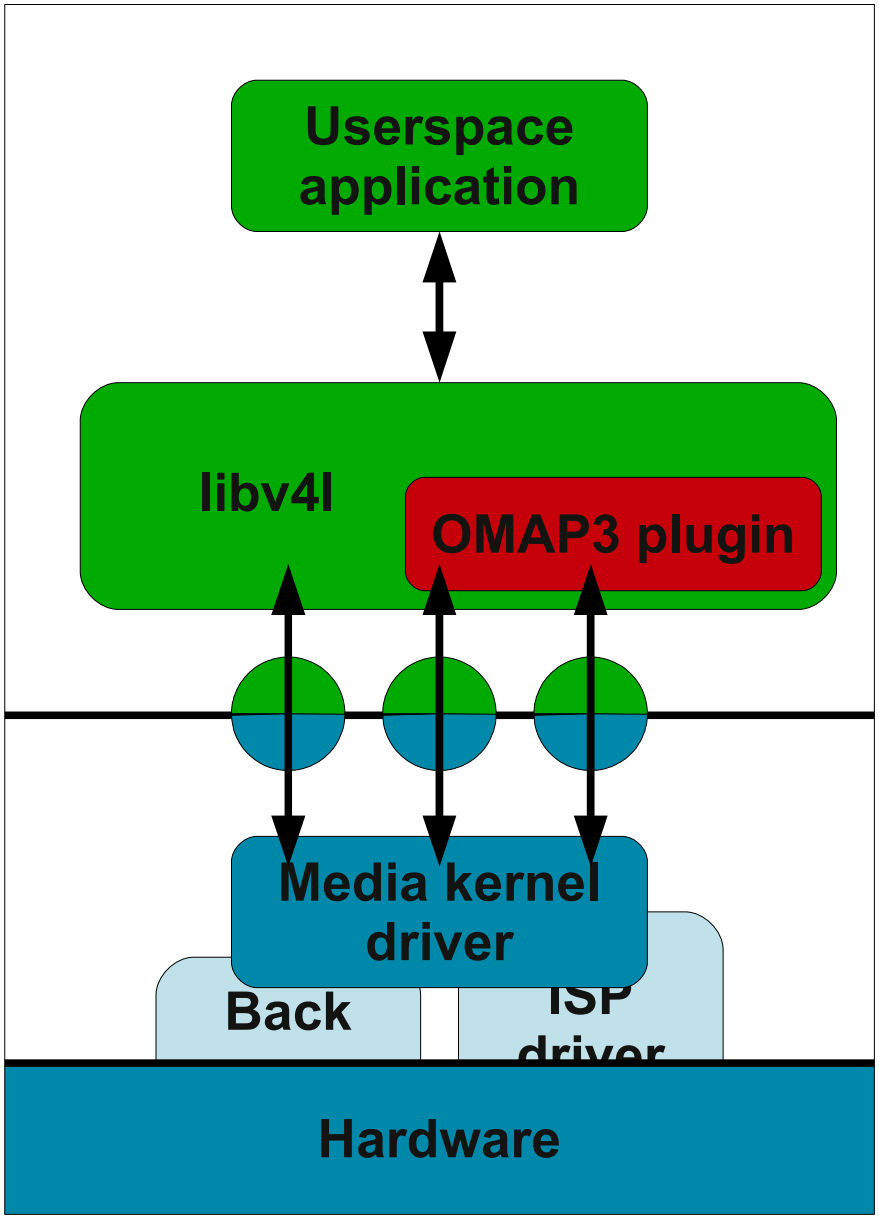


Image quality



- Image enhancement algorithms
- Hardware-specific acceleration
- Transparent for applications

libv4l

- http://gitorious.org/omap3camera
- http://git.ideasonboard.org/?p=media-ctl.git

Source code