Requests, subdev state and media controller

Media Summit 2024
Vienna, Austria

Laurent Pinchart
laurent.pinchart@ideasonboard.com

V4L2 subdev state

Video device state

Requests

v4l2_subdev_state is a structure that

- stores formats and routing for a subdev
- is managed by the subdev core
- simplifies driver handling of the state information
- supports ‘try’ and ‘set’ semantics with a single driver operation

V4L2 subdev state

```
struct v4l2_subdev_state {
        /* lock for the struct v4l2_subdev_state fields */
        struct mutex _lock;
        struct mutex *lock;
        struct v4l2_subdev *sd;
        struct v4l2_subdev_pad_config *pads;
        struct v4l2_subdev_krouting routing;
        struct v4l2_subdev_stream_configs stream_configs;
};

struct v4l2_subdev_pad_ops {
        ...
        int (*set_fmt)(struct v4l2_subdev *sd,
                       struct v4l2_subdev_state *state,
                       struct v4l2_subdev_format *format);
        ...
};
```

struct v4l2_subdev_state

```
static const struct v4l2_subdev_pad_ops mt9m114_pa_pad_ops = {
        ...
        .get_fmt = v4l2_subdev_get_fmt,
        .set_fmt = mt9m114_pa_set_fmt,
        ...
};

static const struct v4l2_subdev_ops mt9m114_pa_ops = {
        .pad = &mt9m114_pa_pad_ops,
};

static const struct v4l2_subdev_internal_ops mt9m114_pa_internal_ops = {
        .init_state = mt9m114_pa_init_state,
};
```

struct v4l2_subdev_state

```
static int mt9m114_pa_init_state(struct v4l2_subdev *sd,
                                 struct v4l2_subdev_state *state)
{
        struct v4l2_mbus_framefmt *format;
        struct v4l2_rect *crop;

        crop = v4l2_subdev_state_get_crop(state, 0);

        crop->left = 0;
        crop->top = 0;
        crop->width = MT9M114_PIXEL_ARRAY_WIDTH;
        crop->height = MT9M114_PIXEL_ARRAY_HEIGHT;

        format = v4l2_subdev_state_get_format(state, 0);

        format->width = MT9M114_PIXEL_ARRAY_WIDTH;
        format->height = MT9M114_PIXEL_ARRAY_HEIGHT;
        format->code = MEDIA_BUS_FMT_SGRBG10_1X10;
        format->field = V4L2_FIELD_NONE;
        format->colorspace = V4L2_COLORSPACE_RAW;
        ...

        return 0;
}
```

struct v4l2_subdev_state

```
static int mt9m114_pa_set_fmt(struct v4l2_subdev *sd,
                              struct v4l2_subdev_state *state,
                              struct v4l2_subdev_format *fmt)
{
        struct mt9m114 *sensor = pa_to_mt9m114(sd);
        struct v4l2_mbus_framefmt *format;
        struct v4l2_rect *crop;
        unsigned int hscale;
        unsigned int vscale;

        crop = v4l2_subdev_state_get_crop(state, fmt->pad);
        format = v4l2_subdev_state_get_format(state, fmt->pad);

        hscale = DIV_ROUND_CLOSEST(crop->width, fmt->format.width ? : 1);
        vscale = DIV_ROUND_CLOSEST(crop->height, fmt->format.height ? : 1);
        format->width = crop->width / clamp(hscale, 1U, 2U);
        format->height = crop->height / clamp(vscale, 1U, 2U);

        fmt->format = *format;

        if (fmt->which == V4L2_SUBDEV_FORMAT_ACTIVE)
                mt9m114_pa_ctrl_update_blanking(sensor, format);

        return 0;
}
```

struct v4l2_subdev_state

- Add support for controls to store all core data
- Make the state subclassable to store driver data
- Replace .get() + .set() paradigm with .check() + .apply()
- Decouple crop bounds query from .get_selection()

V4L2 subdev state – TODO

```
static int mt9m114_pa_set_fmt(struct v4l2_subdev *sd,
                              struct v4l2_subdev_state *state,
                              struct v4l2_subdev_format *fmt)
{
        struct mt9m114 *sensor = pa_to_mt9m114(sd);
        struct v4l2_mbus_framefmt *format;
        struct v4l2_rect *crop;
        unsigned int hscale;
        unsigned int vscale;

        crop = v4l2_subdev_state_get_crop(state, fmt->pad);
        format = v4l2_subdev_state_get_format(state, fmt->pad);

        hscale = DIV_ROUND_CLOSEST(crop->width, fmt->format.width ? : 1);
        vscale = DIV_ROUND_CLOSEST(crop->height, fmt->format.height ? : 1);
        format->width = crop->width / clamp(hscale, 1U, 2U);
        format->height = crop->height / clamp(vscale, 1U, 2U);

        fmt->format = *format;

        if (fmt->which == V4L2_SUBDEV_FORMAT_ACTIVE)
                mt9m114_pa_ctrl_update_blanking(sensor, format);

        return 0;
}
```

V4L2 subdev state – TODO

V4L2 subdev state

Video device state

Requests

- Implement the same for video_device
- Implement .get() + .try() + .set() in core based on the state
- Make the transition smooth

Video device state – TODO

V4L2 subdev state

Video device state

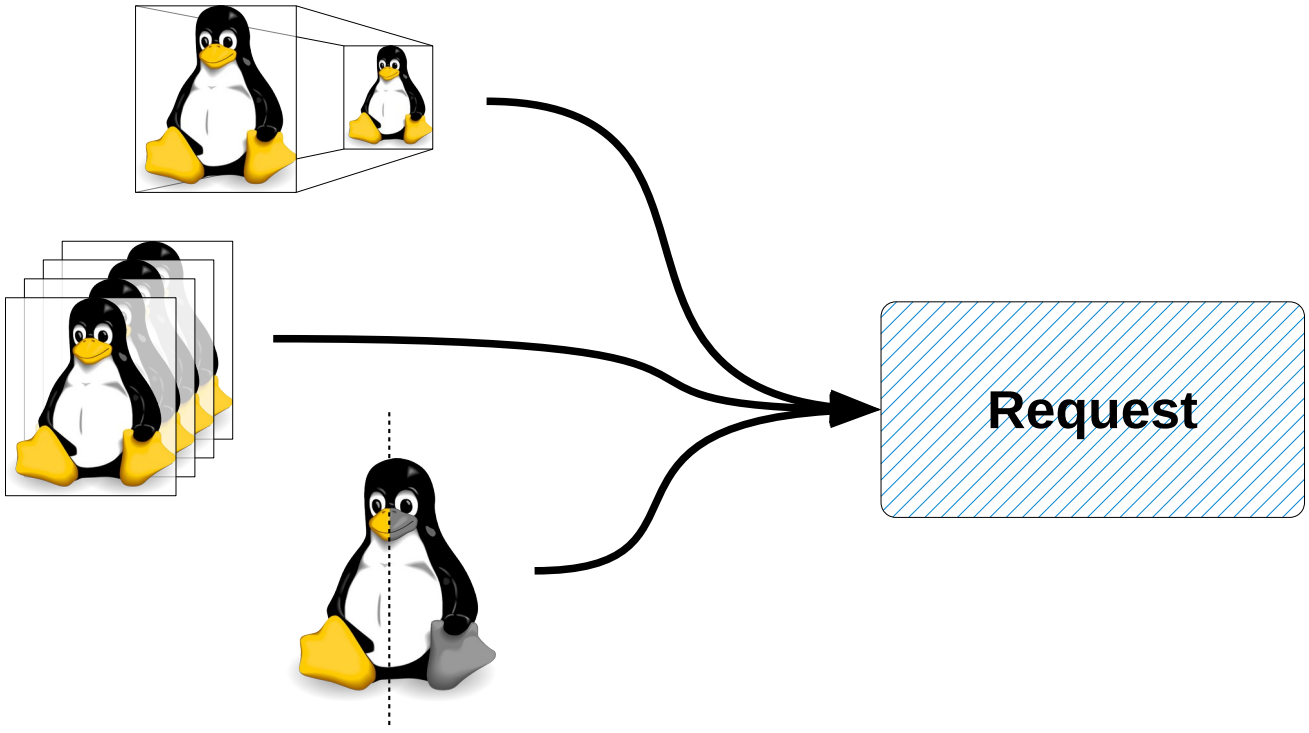
Requests

V4L2 on Steroids
The Request API

Embedded Linux Conference 2016
San Diego

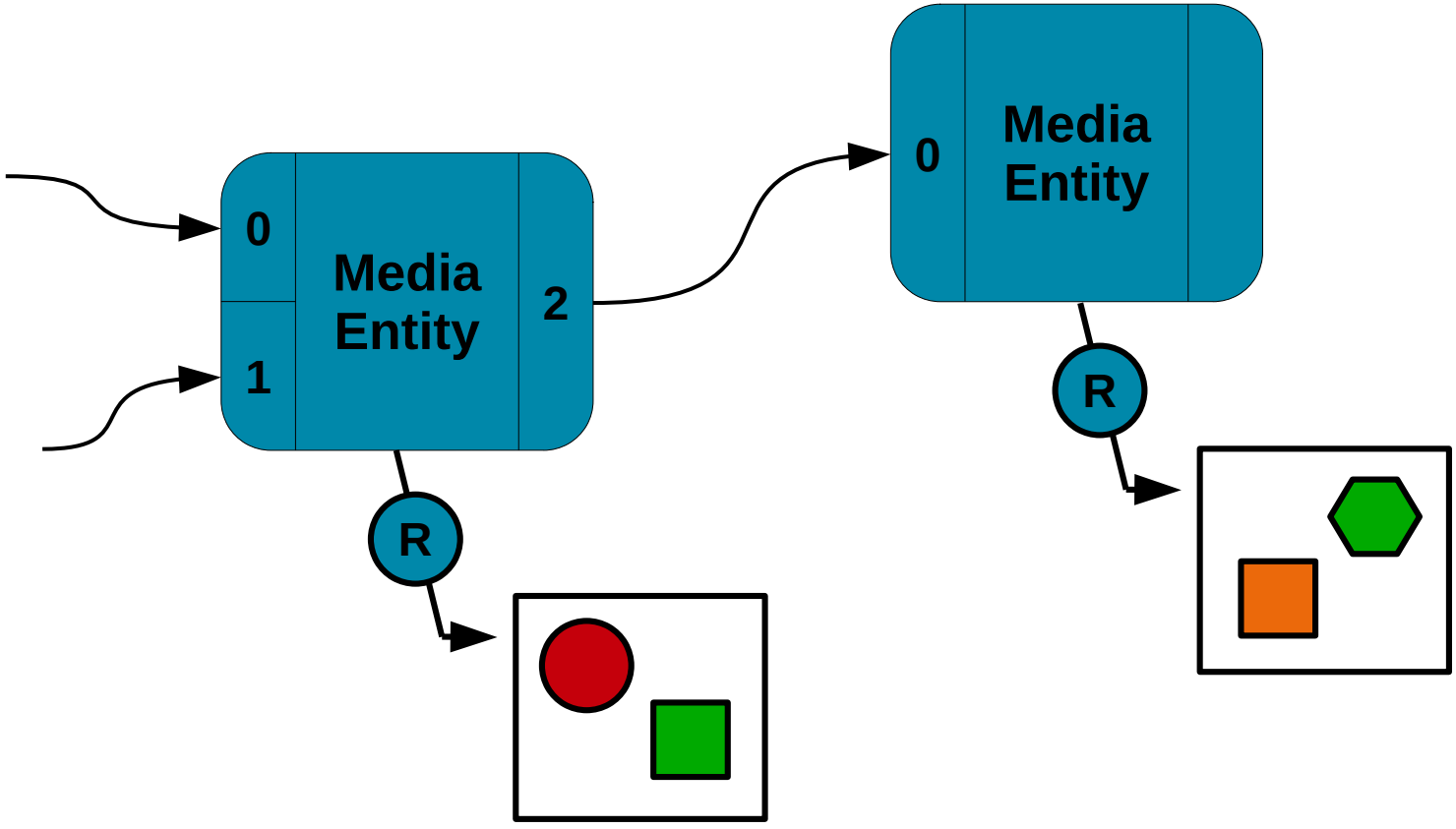
Laurent Pinchart
laurent.pinchart@ideasonboard.com

Once upon a time



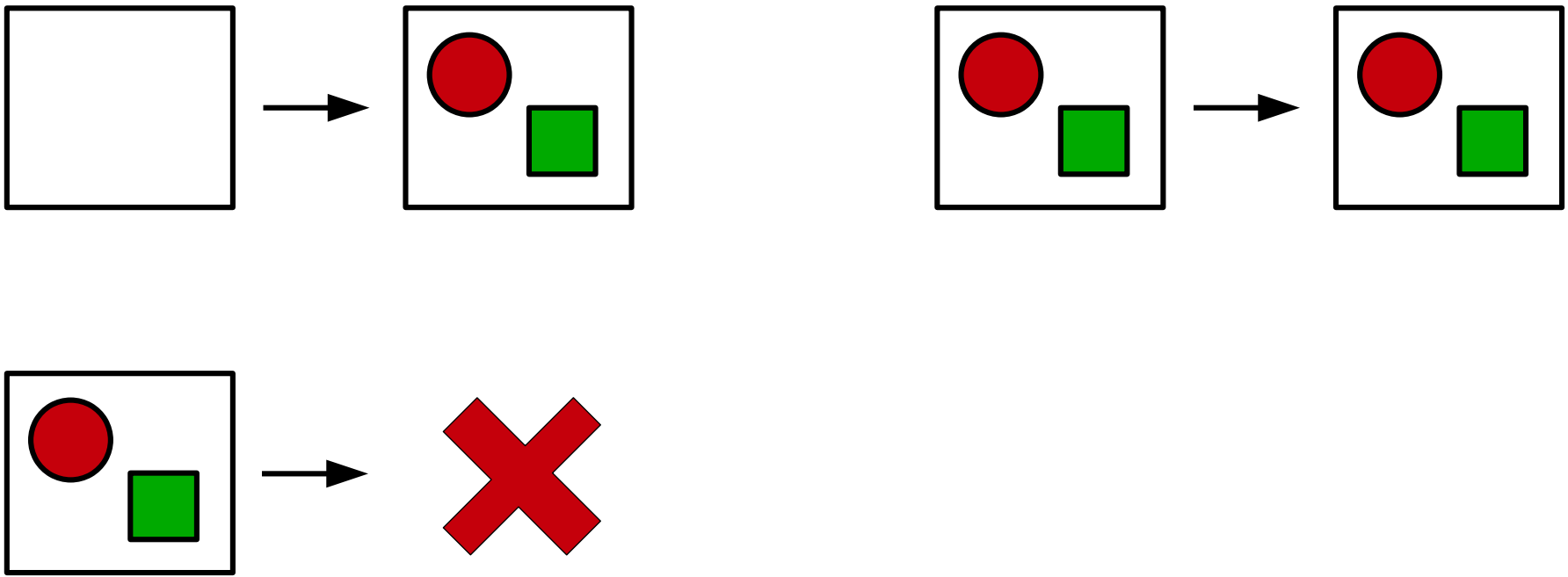
IDEAS ON BOARD

Requests

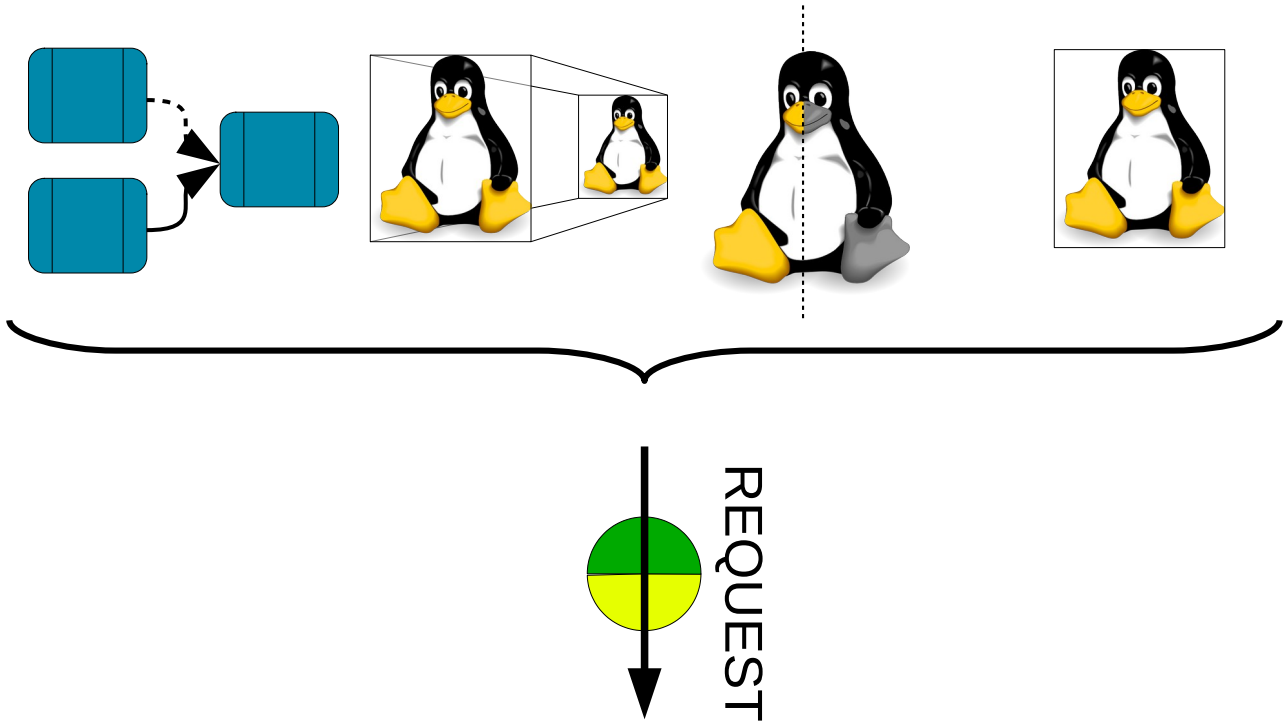


IDEAS ON BOARD

Entity State



Entity State



Request API

- Implement the video device state
- Design the atomic request uAPI
- Map requests to states in kernel space
- Get inspiration from KMS

Request API – TODO

✅ Pieces are falling into place

😨 There’s lots of work left

Request API – TODO

laurent.pinchart@ideasonboard.com

Contact

Vielen Danke