Proposal of subdev pool for V4L2 API

Tomasz Stanislawski

Samsung SPRC

March 17, 2011

Agenda

1. Subdev registration methods:
   - v4l2_i2c_new_subdev
   - v4l2_spi_new_subdev
   - subdev pointer kept in device private fields
2. Issues
   - keeping bus configuration in host driver
   - no general framework for subdev access

Requirements

1. accessing subdev by name[1] like in other kernel frameworks
2. handling multiple subdevs from single device
3. orthogonal to existing mechanism for subdev management
4. deal with drivers dependency
   - request module
   - wait until probe is finished
   - return errors if probing device failed or it is not possible

[1] optional numeric id for multiple instances of the same device

API proposal

1. Subdev registration to the pool

```
int v4l2_subdev_pool_register(
struct v4l2_subdev *sd, struct module *owner);
```

2. Subdev unregistration

```
void v4l2_subdev_pool_unregister(
struct v4l2_subdev *sd);
```

API proposal

1. acquiring subdev from the pool

   ```
   struct v4l2_subdev *v4l2_subdev_pool_get(char *name);
   ```

   - name is taken from v4l2_subdev.name

2. releasing subdev acquired from the pool

   ```
   void v4l2_subdev_pool_put(struct v4l2_subdev *sd);
   ```

Usage scenario - slave driver's probe

```
sd = kzalloc(sizeof(*sd), GFP_KERNEL);
/* ... setting up subdevice ... */
v4l2_i2c_subdev_init(sd, client, &sensor_driver_ops);
/* setting subdev's name */
sprintf(sd->name, "sensor_device");
ret = v4l2_subdev_named_register(sd, THIS_MODULE);
/* error testing, further initalization, etc.*/
```

Usage scenario - slave driver's remove

```
struct v4l2_subdev *sd = i2c_get_clientdata(client);
v4l2_subdev_named_unregister(sd);
kfree(sd);
return 0;
```

Usage scenario - master driver's probe

```
struct v4l2_subdev *sd;
sd = v4l2_subdev_named_get("sensor_device");
if(sd == NULL) {
/* ... error info and cleanup ... */
}
/* ... storing sd in a device's private data ... */
```

Usage scenario - master driver's remove

```
struct v4l2_subdev *sd;
/* ... obtaining variable sd from device private data ... */
v4l2_subdev_named_put(sd);
/* ... cleanup .... */
```