Video4Linux soc-camera subsystem

Guennadi Liakhovetski

(Contact: linux-media@vger.kernel.org)

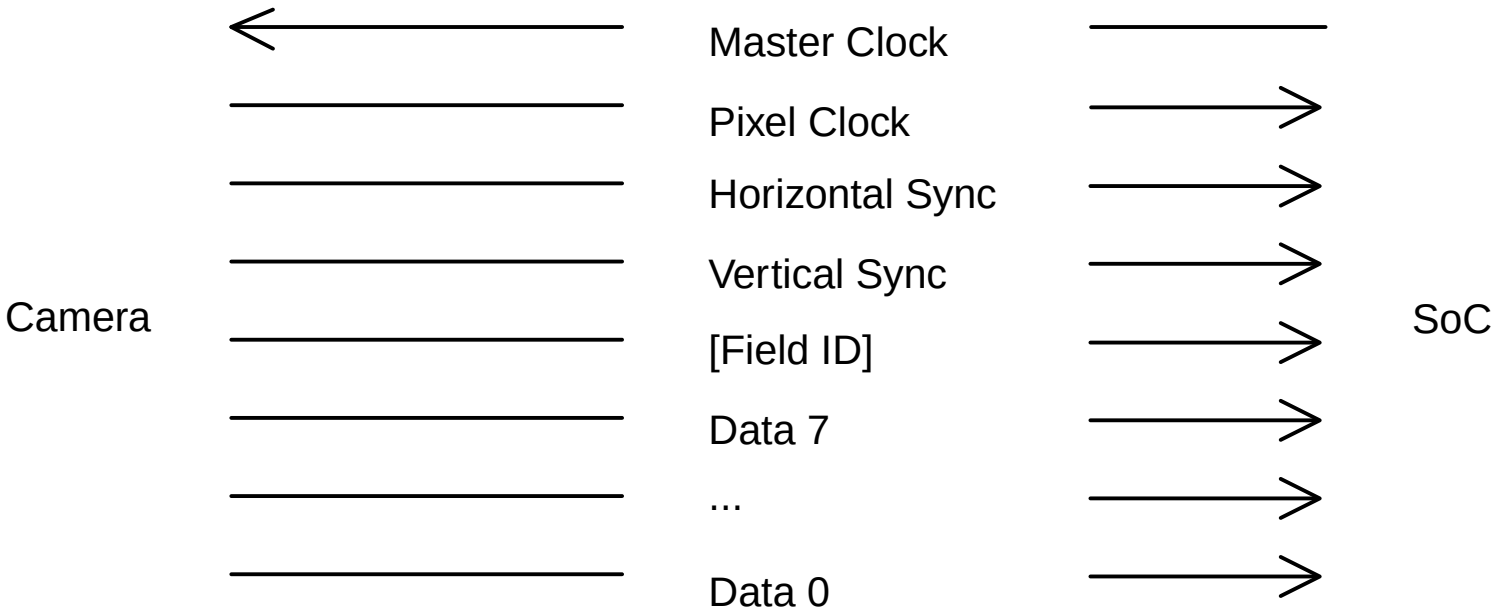
October 12, 2009

Definitions

- Soc-camera is a set of drivers and a core module, that implement v4l functionality on embedded devices.
- Typical video-enabled embedded device: SoC with a capture interface and video data sources.
- What are video hosts and video clients.

Example system

SuperH mobile host with OV7725 CMOS camera sensor and TW9910 TV-decoder from Techwell.



Purpose

- Soc-camera provides a standard API for host drivers on one side, and for client drivers on the other side.
- The client-driver API is currently being ported over to the new v4l2-subdev API.
- Driver re-use.

General functionality

- Creates a new bus-type.
- Registers a platform driver.
- Presents a standard API for video host drivers.
- Manages video_device objects and their user-space interface.

Host-client linking

- Platform code must register one platform devices for each video client.
- soc-camera core probes all those platform devices and puts them on an internal list.
- Upon host driver registration the core looks through clients on the list and finds any matches.
- For each match a device instance is registered for the client on the soc-camera bus with the host device as its parent.
- The camera interface on the SoC is activated, and then the client's (i2c) driver is loaded and probes the video client hardware.
- After successful probing a v4l2 subdevice and a video-device are created and a bus-parameter- and data-format-negotiation is performed.

Run-time

- All system calls to video-device nodes get dispatched to the soc-camera core.
- The core either handles the call internally or dispatches it to the host driver.
- Host-drivers either handle calls themselves, or pass them further to clients.

Supported hardware

- Host drivers: PXA270, SuperH-mobile, i.MX31, i.MX1 / i.MXL.
- Client drivers: Micron / Aptina MT9M001, MT9M111, MT9T031, MT9V022, OmniVision OV7720 (OV7725), OV9640, Sharp RJ54N1CB0C, Techwell TW9910.

New APIs

- v4l2-subdev: currently in the mainline.
- New RFCs:
  - Media controller proposal (Hans Verkuil).
  - Bus and data format negotiation (Hans Verkuil).
  - Global video buffers pool (Laurent Pinchart).
  - Video events (Laurent Pinchart).
- The completion of the soc-camera to v4l2-subdev conversion will depend on these RFCs.

Media controller proposal

- Author: Hans Verkuil, version: 2.1 (13 Sep 2009), URL: http://www.spinics.net/lists/linux-media/msg09971.html.
- Published on the linux-media mailing list, discussed at a privately held informal developer meeting in August 2009, and then at the Linux Plumbers Conference in September 2009.
- Introduces a new /dev/v4l/mcX device node, one per board.
- Used to access advanced features of the underlying video hardware (board):
  - Enumeration of available components.
  - Connection of various component inputs and outputs.
  - Configure single components.
- Doesn't aim to modify the existing V4L2 API. All existing applications should still work.
- Example implementation is available at http://www.linuxtv.org/hg/~hverkuil/v4l-dvb-mc.

Bus and data format negotiation

- Author: Hans Verkuil, version: 1.0 (13 Sep 2009), URL: http://www.spinics.net/lists/linux-media/msg09979.html.
- Bus configuration: specifies, which physical lines comprise the bus, and what signalling is used on those lines.
- Video data format negotiation: in what format the data should be sent over the video bus.

Global video buffers pool

- Author: Laurent Pinchart, version: 1.0 (16 Sep 2009), URL: http://www.spinics.net/lists/linux-media/msg10145.html.
- Allocating large contiguous memory buffers for video data is problematic due to memory fragmentation, preparation of such buffers often requires time-consuming cache-operations.
- Creation of a global system-wide video-buffer pool.

Video events

- Author: Laurent Pinchart, version: 1.0 (18 Sep 2009), URL: http://www.spinics.net/lists/linux-media/msg10217.html.
- Handles asynchronous notification of user-space applications. Event examples: button pressed, frame size changed, exposure changed.
- Currently several custom video event implementation exist in the kernel.
- The API should allow to select which events to listen to, and to get details of occurring events.
- Currently not implemented by soc-camera, but certain event types can become interesting to the user.

Future development

- Finish conversion to the new APIs, thus make soc-camera 100% v4l2-subdev compliant.
- Handle originally soc-camera client driver use for other set ups, and generic subdev driver re-use with soc-camera.
- Several new drivers have to be handled, that are being held back due to API instability.
- New features and improvements to existing drivers.