Shifting Media app development into high gear

Using virtual drivers to speed up development

Helen Koike
helen.koike@collabora.com

Creative Commons Attribution-ShareAlike 4.0
Summary

- Classic V4L2 API → Vivid Driver
- Media API → Vimc Driver
- Codecs → Vicodec Driver

Vimc:
  - Submodules
  - Configfs API
  - Future work
Summary

- Classic V4L2 API → Vivid Driver
- Media API → VIMC Driver
- Codecs → Vicodec Driver

VIMC:
- Submodules
- Configure API
- Future work

Oct 2018
User space

/dev/video*

Kernel space

Driver

Physical device
User space

Kernel space

Physical device

APP

Read/Write/MMAP

/dev/video*

Video stream
User space

Kernel space

Physical device

------------------- /dev/video* -------------------

**Driver**

- Video stream
- General configs
  * Img fmt
  * Buffers
  * Video std
  * Frame rate

**Standard Controls**

- Contrast
- Brightness
- Gamma

**Custom Controls**

- DRV DEF 1
- DRV DEF 2
- DRV DEF 3

Read/Write/MMAP

IOCTLS
User space

/dev/video*

Kernel space

Driver

Physical device
User space

Kernel space

Driver

Physical device

/dev/video*

APP

User space

Kernel space

Driver

Physical device

Oct 2018
The Virtual Video Test Driver (VIVID)
The Virtual Video Test Driver (VIVID)

User space
/dev/video*

Kernel space

Physical device

---
Vivid driver
Vivid driver: current state

- Merged in 3.17 by Hans Verkuil
- Good coverage of the API
- Error injection
- Up to 4K resolution
- ...

More info:
Summary

- Classic V4L2 API → Vivid Driver
- Media API → Vimc Driver
- Codecs → Vicodec Driver

- Vимc:
  - Submodules
  - Configfs API
  - Future work
Media API - Motivation

- Embedded systems: great variety of devices
- Hardware complexity
- Similar configuration in different components of the device
What is part of the device? What is the topology?

/dev/v4l-subdev0
/dev/v4l-subdev14
/dev/video0
/dev/video42
/dev/v4l-subdev31
/dev/v4l-subdev98
/dev/v4l-subdev6
/dev/video2
/dev/v4l-subdev11
User space

/dev/media*

Kernel space

Driver

Physical device
The Virtual Media Controller Driver (VIMC)
The Virtual Media Controller Driver (VIMC)
Vimc Driver

- Proposed by Laurent Pinchart for Outreachy in 2015
- Merged in Kernel 4.12
- Moving slowly
Vimc Driver: current state

• Basic set of emulation units:
  - capture (no output e.g. HDMI/S-Video)
  - sensor (generates images in different formats)
  - debayer (converts bayer to non-bayer format)
  - scaler (scaling down is not supported)
• Several optimizations and controls are still missing
• Hard-coded topology (re-compilation required)
Vimc Driver
Summary

- Classic V4L2 API → Vivid Driver
- Media API → Vimc Driver
- Codecs → Vicodec Driver

- Vimc:
  - Submodules
  - Configfs API
  - Future work
Codecs (encoder/decoder)

• Encoder:
  - raw → compressed (e.g. RGB → H.264)

• Decoder:
  - compressed → raw (e.g. H.264 → RGB)
Stateful vs Stateless codecs

- Stateful codecs:
  - Driver keeps the current state
    (definition of state depends on the image format)
  - Requires /dev/video*

- Stateless codecs:
  - Userspace keeps the state sending it in every frame
  - Requires /dev/video* /dev/media*
The Virtual Codec Driver (Vicodec)

User space
/dev/video* (/dev/media*)

Kernel space

Physical device

Encoder  Decoder  Encoder  Decoder

Encoder  Decoder
Vicodec Driver: current state

- Relatively new (4.18 by Tom aan de Wiel)
- FWHT codec (Fast Walsh Hadamard Transform)
- Supports stateful only
- Stateless support: proposed as an Outreachy project
Use cases for virtual drivers

- Hardware is not ready / Board bring-up phase
- Convenient to make tests on a PC instead of an embedded device
- Test application in different hardware
  - e.g. Gstreamer
Use cases for virtual drivers (cont)

• Loop back device
  – Adding layers of effects
  – Test streams
• Test APIs
• Reference code
• Improve compliance tests
Summary

- Classic V4L2 API → Vivid Driver
- Media API → Vimc Driver
- Codecs → Vicodec Driver

- Vimc:
  - Submodules
  - Configfs API
  - Future work
VIMC Configfs API
---------------------
\hspace{2cm} \texttt{/dev/media0}
\hspace{2cm} \texttt{/dev/media1}

\hspace{2cm} User space
\hspace{2cm} \textbf{Developer}
\hspace{2cm} \textbf{APP}

\hspace{2cm} Kernel space

create

\hspace{2cm} create

Oct 2018
Vimc: submodules

Implementation of subdevice drivers are modularized and doesn’t need to alter Vimc’s core code.
APP Developer

Emulate different topologies to test the app in several scenarios
Summary

- Classic V4L2 API → Vivid Driver
- Media API → VImc Driver
- Codecs → Vicodec Driver

- VImc:
  - Submodules
  - Configfs API
  - Future work

Oct 2018
Configfs API: Topology

- Entities
  - Name
  - Submodule
- Pads
  - Source
  - Sink
- Links
mkdir "MEDIA_NAME"
mkdir "SUBMOD:NAME"
mkdir "NAME1:PAD->NAME2:PAD"
Summary

- Classic V4L2 API → Vivid Driver
- Media API → Vimc Driver
- Codecs → Vicodec Driver

- Vimc:
  - Submodules
  - Configfs API
  - Future work
Vimc Driver: future work

- API in Configfs (WIP)
- More controls:
  - LKCAMP – Linux kernel study group
- Image generation directly from capture node (WIP)
  - LKCAMP
- Frame rate control

Oct 2018
Vimc Driver: future work

- Optimizations of img processing in the pipeline
  - Calculate final format based on the full pipeline configuration
  - Multi-threaded processing
- Submodules for output: HDMI / S-Video
- More standard submodules
- Add more V4L2 mechanisms / controls / options
Thank you!

Helen Koike
helen.koike@collabora.com

This work is licensed under https://creativecommons.org/licenses/by-sa/4.0/
Userspace tools

• v4l-utils
  media-ctl -d /dev/media0 --print-dot > /tmp/out.dot && xdot /tmp/out.dot
  media-ctl -d /dev/media0 -V "Sensor A':0[fmt:RGB888_1X24/600x600]"
  media-ctl -v -d /dev/media0 --links "'Debayer A':1->'Scaler':0 [0]"

• Yavta (Yet Another V4L2 Test Application)
  yavta --format RGB24 --size 600x600 /dev/video0
Summary

- Classic V4L2 API → Vivid Driver
- Media API → Vimb Driver
- Codecs → Vicodec Driver

- Vimb:
  - Submodules
  - Configfs API
  - Future work

Oct 2018