A fully open source stack for MIPI cameras.

FOSDEM24

Bryan O'Donoghue - Linaro
Hans de Goede - Red Hat
Introductions

- **Hans**
  - Principal sw engineer @ Red Hat (Laptop HW enablement)
  - Kernel subsystem maintainer for drivers/platform/x86
  - [https://git.kernel.org/pub/scm/linux/kernel/git/hansg/linux.git/](https://git.kernel.org/pub/scm/linux/kernel/git/hansg/linux.git/)
  - hansg @ #linux-media #libcamera #fedora-workstation

- **Bryan**
  - Kernel engineer @ Linaro (Qualcomm Landing Team)
  - Maintainer of Qualcomm CAMSS driver
  - [https://git.code-linaro.org/bryan.odonoghue/kernel](https://git.code-linaro.org/bryan.odonoghue/kernel)
  - [https://github.com/0xB0D](https://github.com/0xB0D)
  - bryanodonoghue @ #linux-media #linux #aarch64-laptops #libcamera #linux-msm
What this talk is about

1. Bayer Encoding
2. What is a Hard ISP - what are the 3As?
3. SoftISP in Libcamera - what’s the problem we are trying to solve?
4. Pipewire and Libcamera
5. Future plans
6. Demos
Bayer Encoding

- Human eye is composed of “rods” and “cones”
  - Rods Scotopic
  - Cones Photopic - sensitive to light around 550 nm
  - Most sensitive to “greenish yellow” colours
- Most RGB sensors don’t capture RGB
  - Contain monochrome sensors
- Sensors overlaid with a “mosaic” pattern of Red, Green, Blue
  - Bruce Bayer Eastman Kodak 1974
  - Bayer encoding
- Problem - each pixel contains only one colour
  - Approximate based on proximate pixels
  - Called “interpolation”
- Methods
  - Label
  - Nearest
  - Bilinear
  - Malar-He-Cutler
Bayer encoding / decoding visualised

Source: McGuire, Morgan “Efficient, High-Quality Bayer Demosaic Filtering on GPUs”
What is a HardISP - what are the 3As?

- **HardISP**
  - Hardware Image Signal Processor
  - A specialised silicon or firmware block - usually both
  - Principle of data locality - process close to sensor input

- **Debayers**

- **Implements 3As**
  - Autofocus (AF)
  - Auto White Balance (AWB)
  - Auto Exposure / Gain (AG)
    - Usually MIPI sensors have an analogue and digital gain

- **More advanced “secret sauce” algorithms**
  - Sensor tunings (joking but not joking)
  - Skin tones
  - Low light noise reductions
  - Contrast
  - Lens flare reductions
  - Black mirror universe bridal poses (again not joking)
SoftISP in Libcamera - what problem are we solving?

- Reluctance/refusal of some vendors to disclose “secret sauce”
- Raw Bayer data delivered to userspace -> IPU6 Intel, CAMSS Qcom, others
- We receive bayer encoded data to userspace and that’s it.
- Desire to have a generic open source implementation that solves the same problem across platforms.

Libcamera
- Defacto OSS standard - RPI4 and RPI5 as example
- Pipewire integration
- Hide away details of V4L camera pipelines

Linaro initiated project to upstream a CPU SoftISP
- Recipes Paris
- Guidance from libcamera people on how
- Andrey Konovalov
- V1

Red Hat engineers joined 2-3 months later
- Hans de Goede
- V2 onwards
Pipewire - New way for applications to access cameras

- MIPI cameras require complex pipeline configuration and extra processing (3A-s) in userspace
- libcamera takes care of this for applications
- Directly opening /dev/video0 no longer works, instead applications need to go through libcamera somehow
- Some distributions are moving to a model where applications run from a sandbox (flatpaks)
- Using pipewire to access cameras solves both the sandboxing and go to through libcamera issues
- Upstream libwebRTC has full support for pipewire cameras
- Jan Grulich (Red Hat) has landed support for pipewire cameras in Firefox 122
Future plans - Better, Faster, Cheaper

- **GPU acceleration - OpenGL**
  - Faster and uses less energy
  - Reuse upstream GLSL fragment/vertex shaders already in libcamera
  - [https://gitlab.freedesktop.org/camera/libcamera-softisp/-/tree/SoftwareISP-v05-opengl-v1-bod](https://gitlab.freedesktop.org/camera/libcamera-softisp/-/tree/SoftwareISP-v05-opengl-v1-bod)
  - Reuse IPA / 3A from CPU if possible

- **GPU acceleration - OpenCL/Vulkan**
  - Some GPUs - Imagination are reported moving Vulcan only
  - An OpenCL or Vulkan compute shader to debayer may be required

- **Image quality enhancements:**
  - Contrast enhancement (CPU and GPU, e.g. Y histogram equalization)
  - Flicker controls (CPU and GPU)
  - Color Saturation enhancement (GPU)
  - Lens shade correction (GPU)
  - Lens shape correction (GPU)
  - Noise reduction (GPU)
  - Defect pixel correction (GPU)
References

- McGuire, Morgan “Efficient, High-Quality Bayer Demosaic Filtering on GPUs”
- Mosiac pattern graphic By Amada44 - Own work, Public Domain,
  https://commons.wikimedia.org/w/index.php?curid=3483669
- Scotopic / Photopic graphic
- 3A description  https://gimoonnam.github.io/imageprocessing/3A-initiation/
- Adorno, José “This iPhone photo of a bride shows a crazy Matrix glitch – but it’s easy to explain”
- Jan Grulich’s blog posts on Firefox pipewire camera integration
  https://jgrulich.cz/
Thank you

Slides?  
Visit www.linaro.org