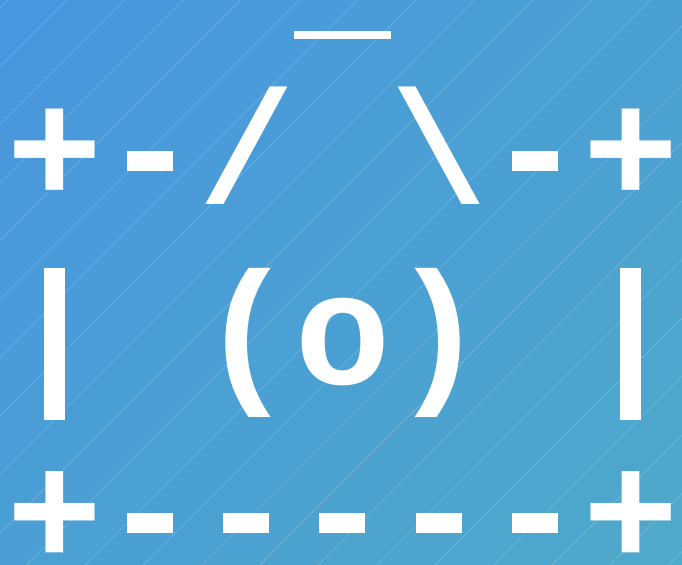
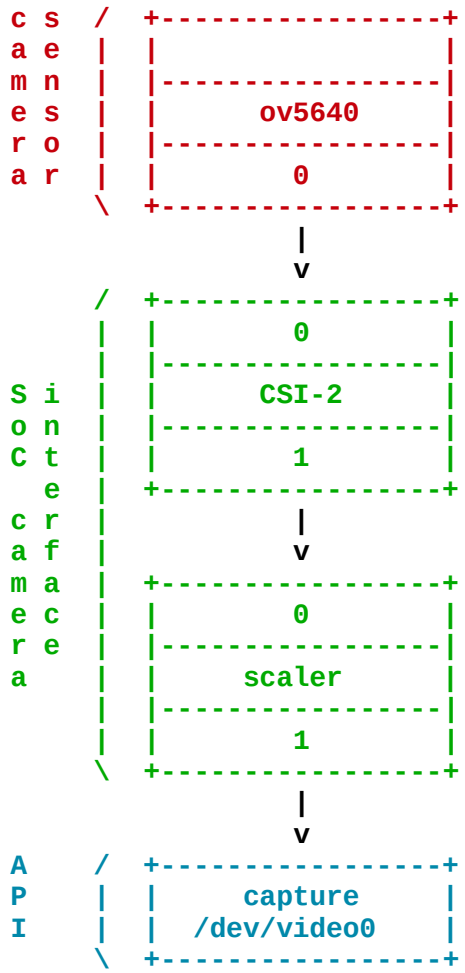


libcamera Architecture and Platform Support

Live Embedded Event
2020-12-03

Laurent Pinchart
laurent.pinchart@ideasonboard.com

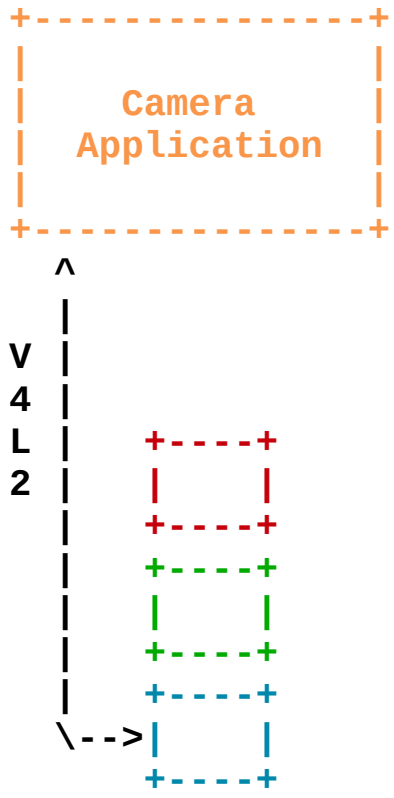
IDEAS ON BOARD



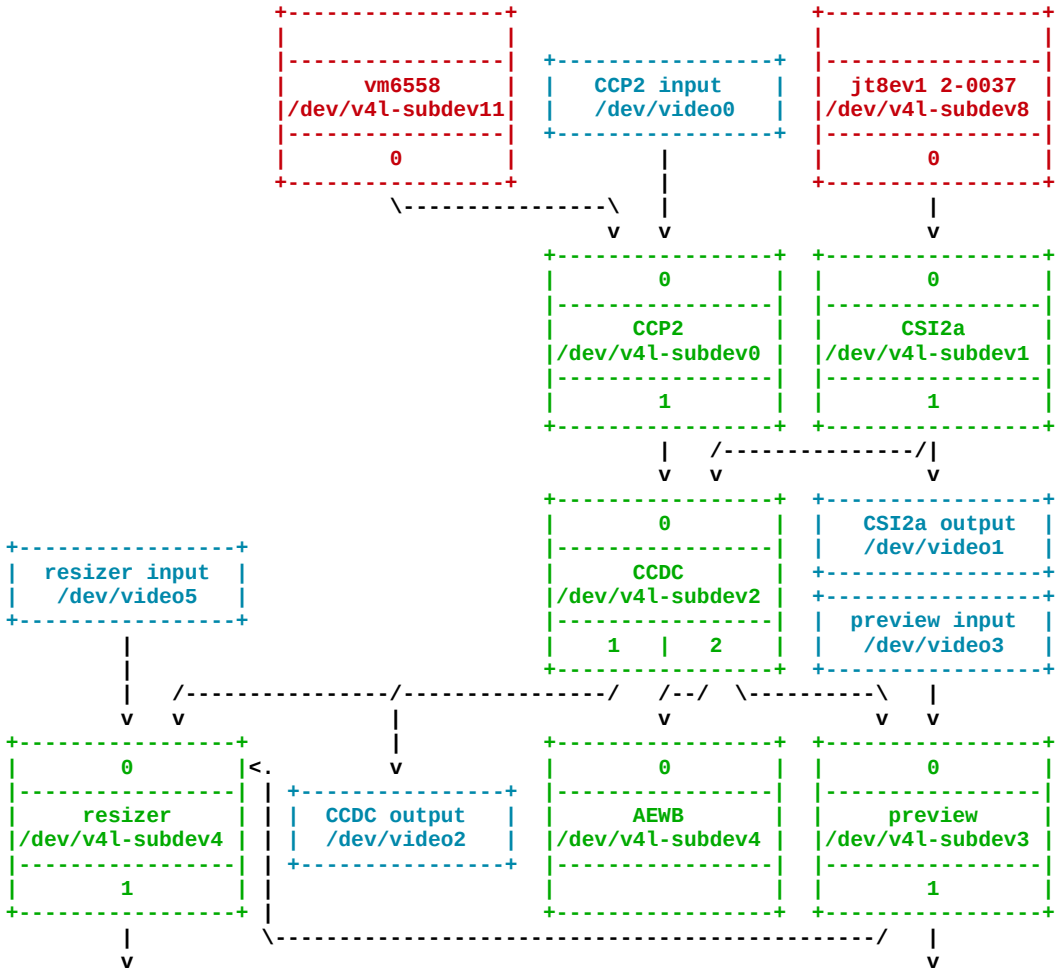
In the beginning were simple pipelines...

Why?

… and they were simple to control, with a single API.



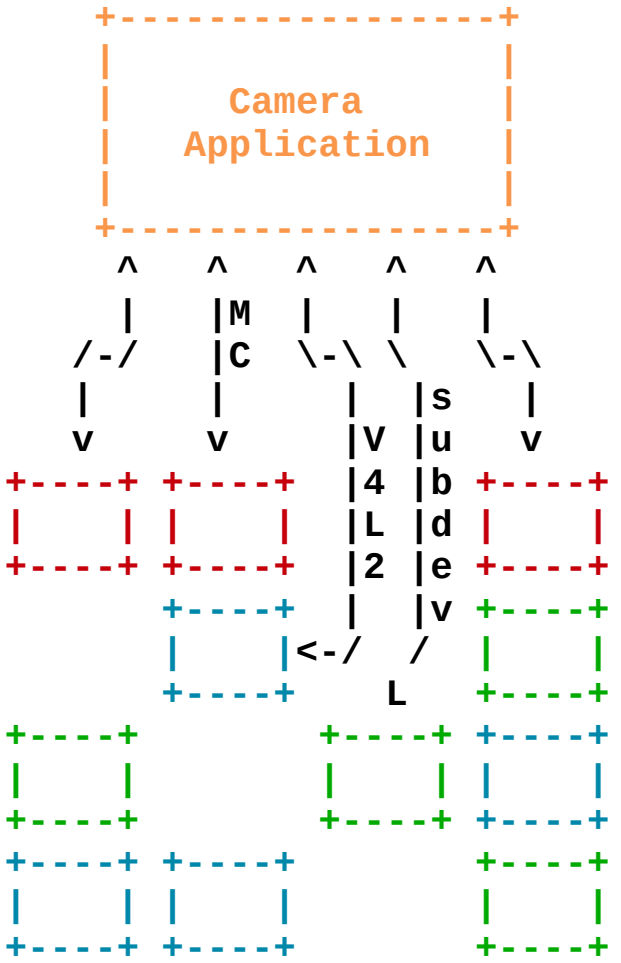
Why?



Then the world became complex ...

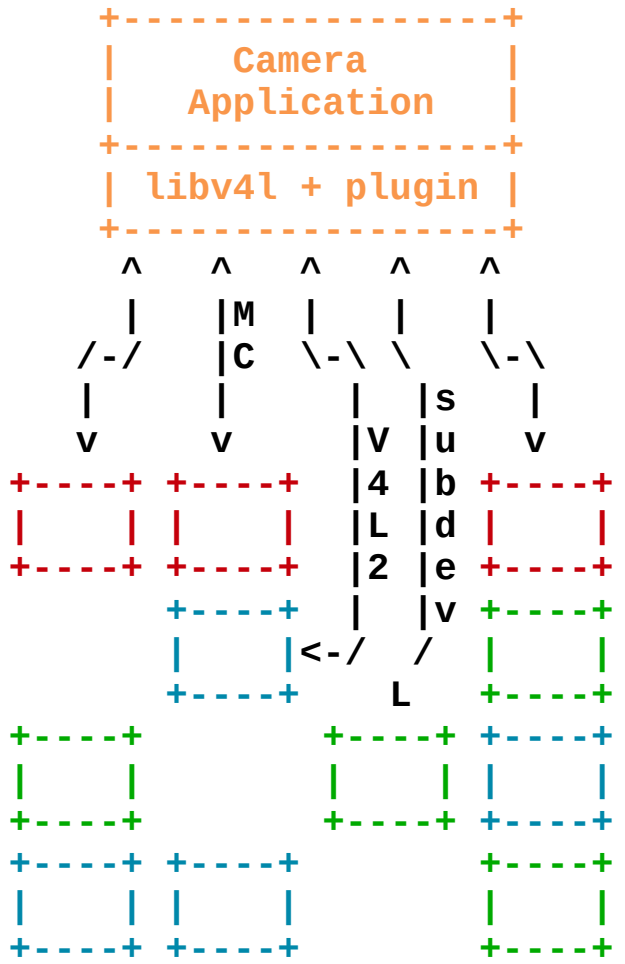
Why?

… and application developers were left suffering.



Why?

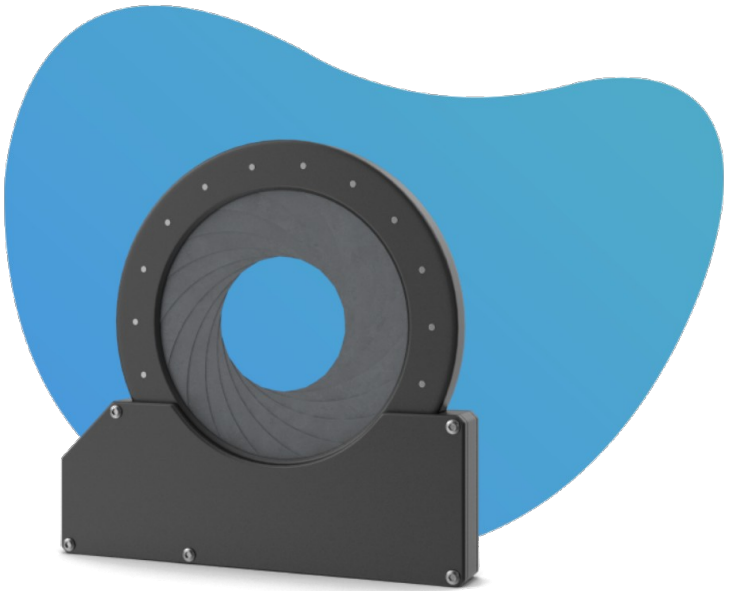
Solutions were proposed...



… but never implemented.

Why?

Then hope came back.

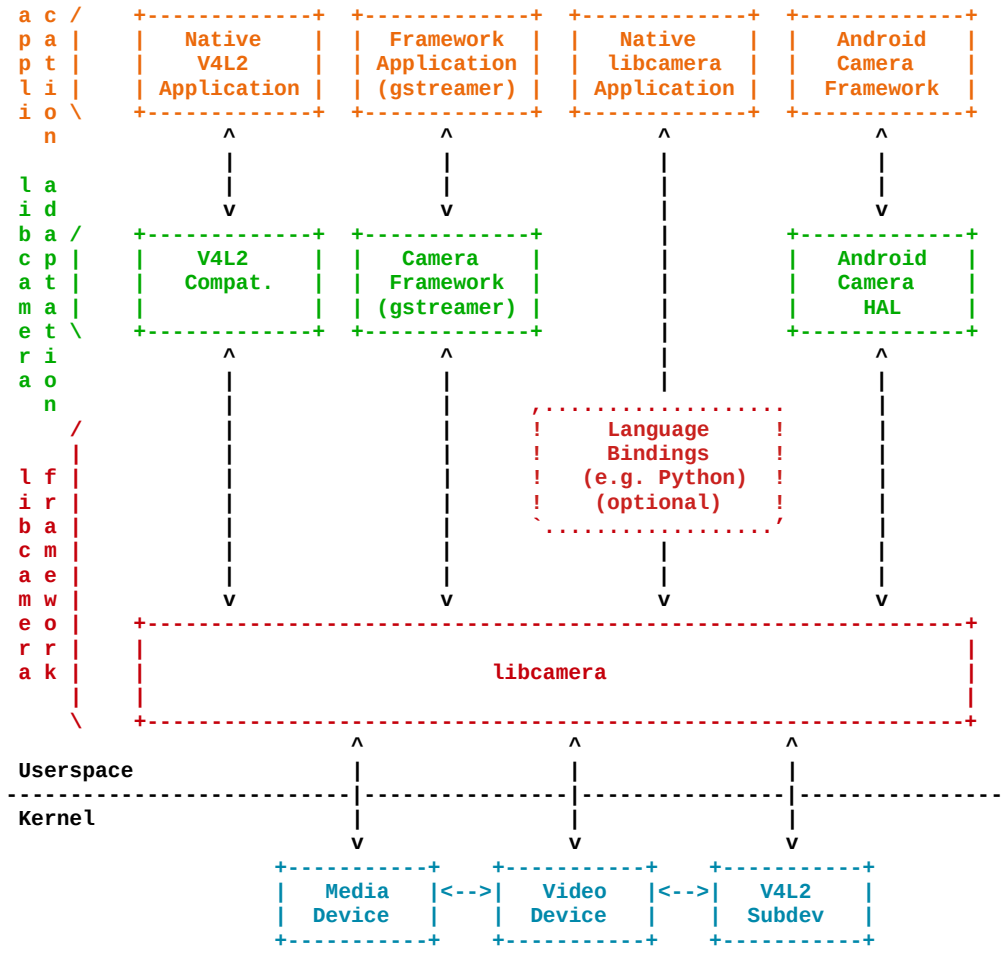


Hi, we're libcamera.

An open source camera stack and framework for Linux, Android, and ChromeOS

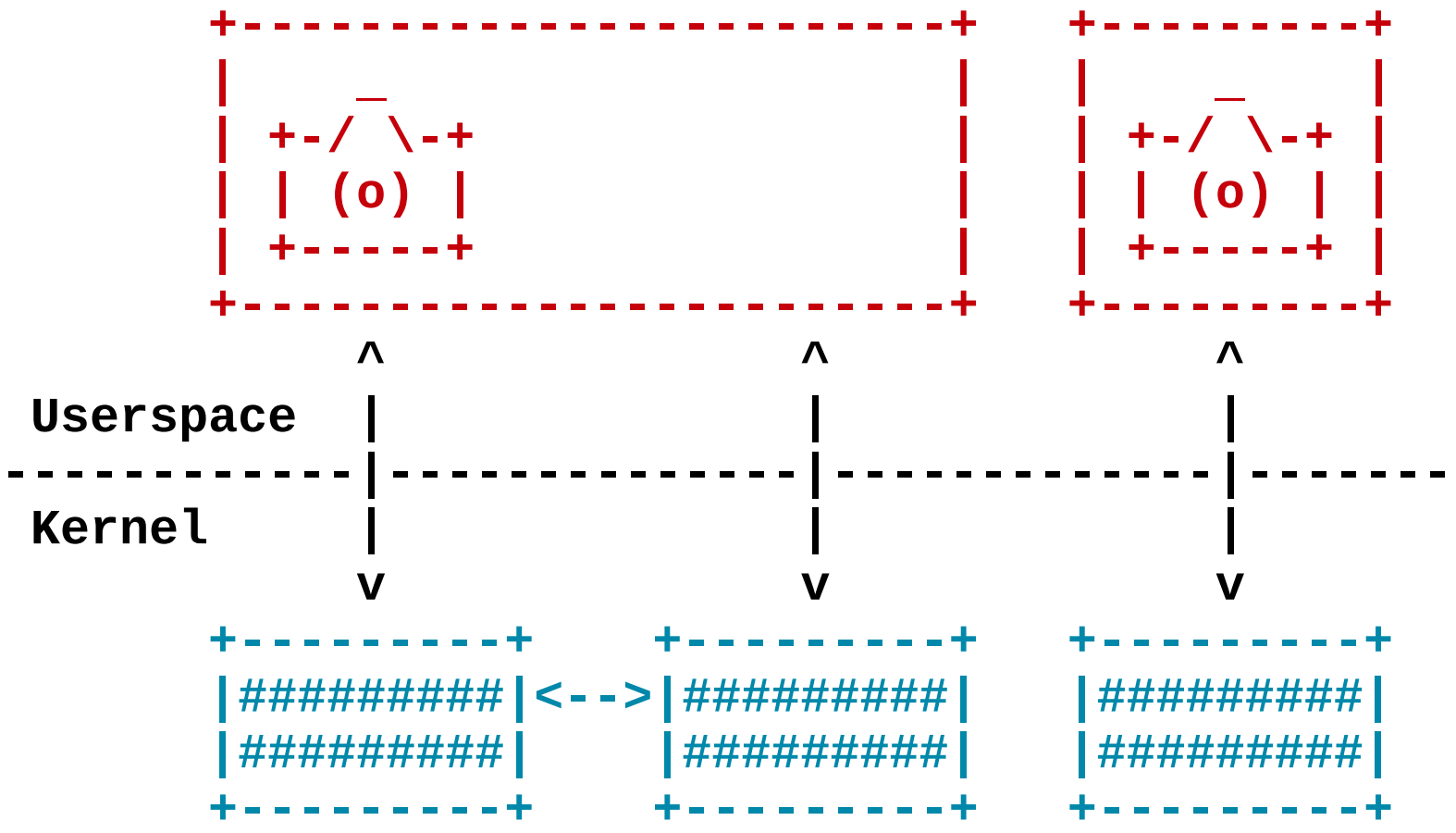
Getting Started

libcamera provides a complete userspace camera stack.



The 'mesa' of the camera world.

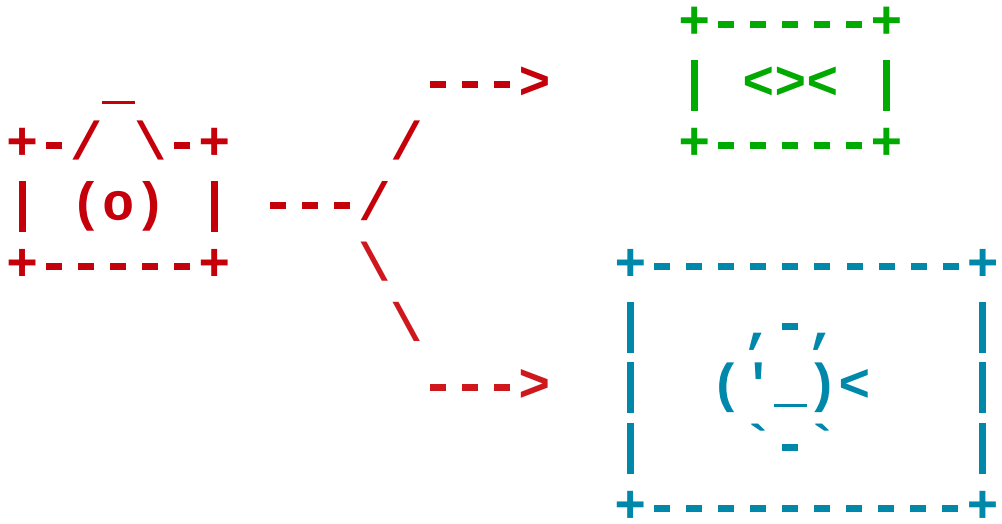
Camera Stack



libcamera enumerates cameras...

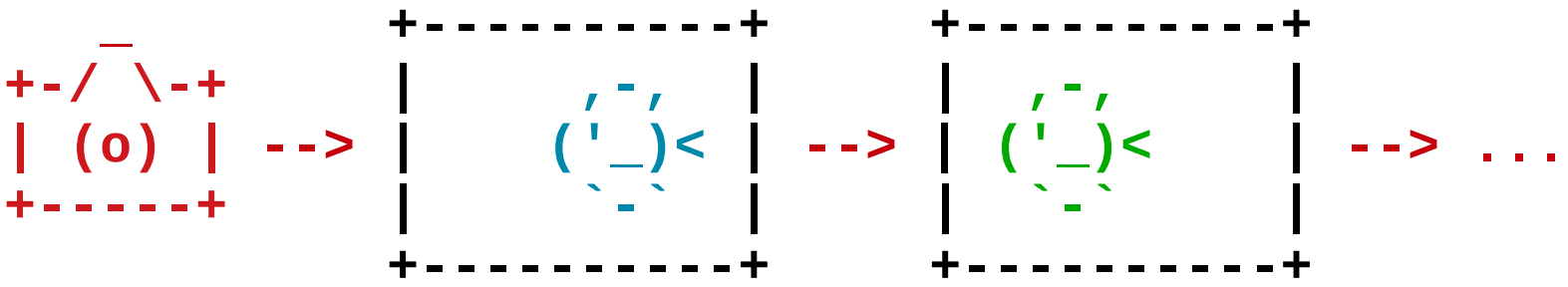
Camera Devices & Enumeration

It supports multiple concurrent streams for the same camera...



Streams

… and per-frame controls.



Per-Frame Controls

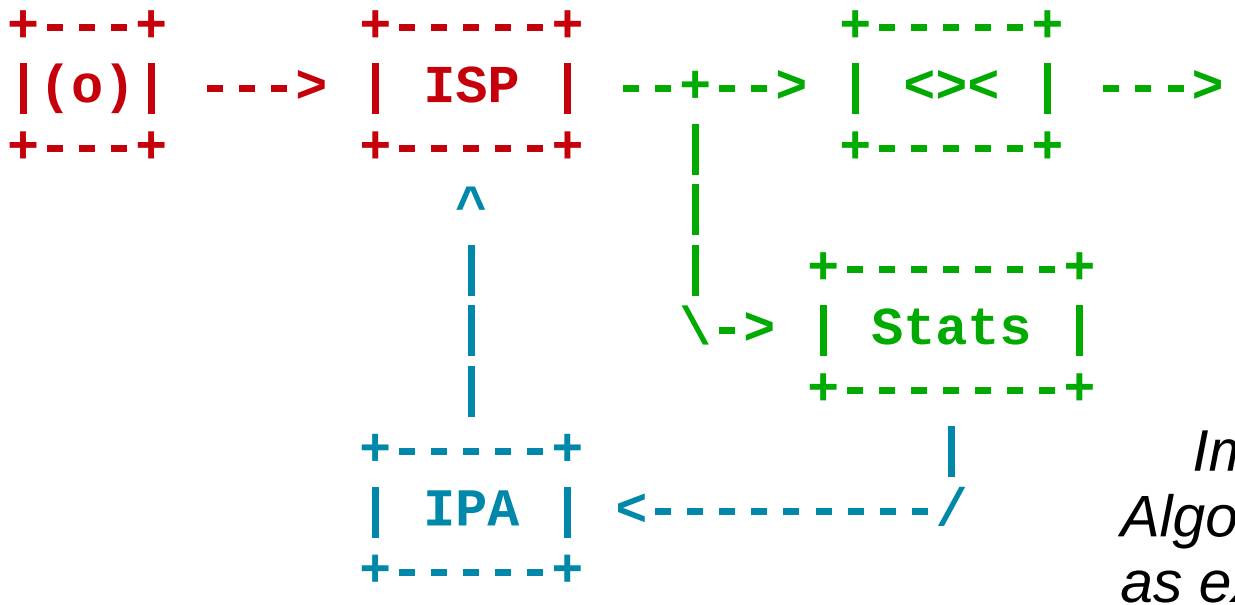
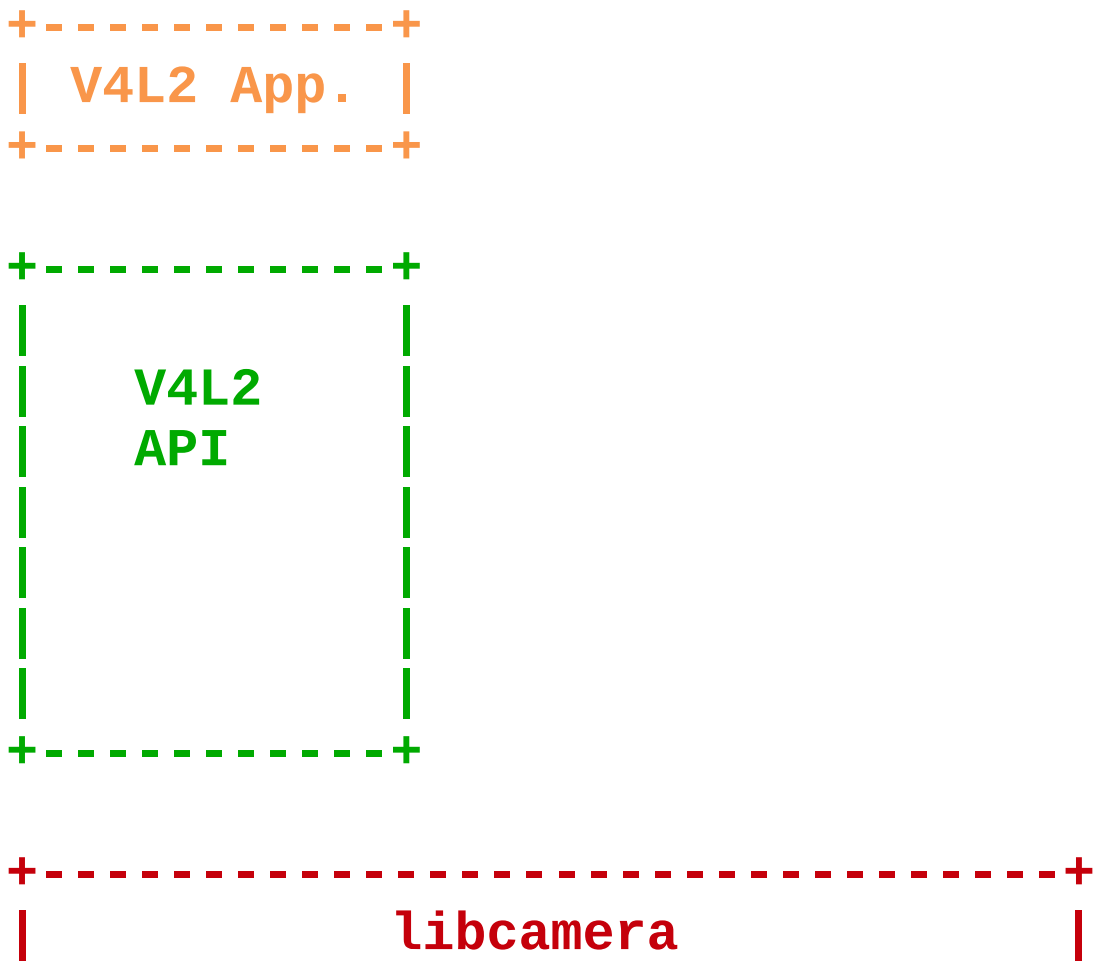


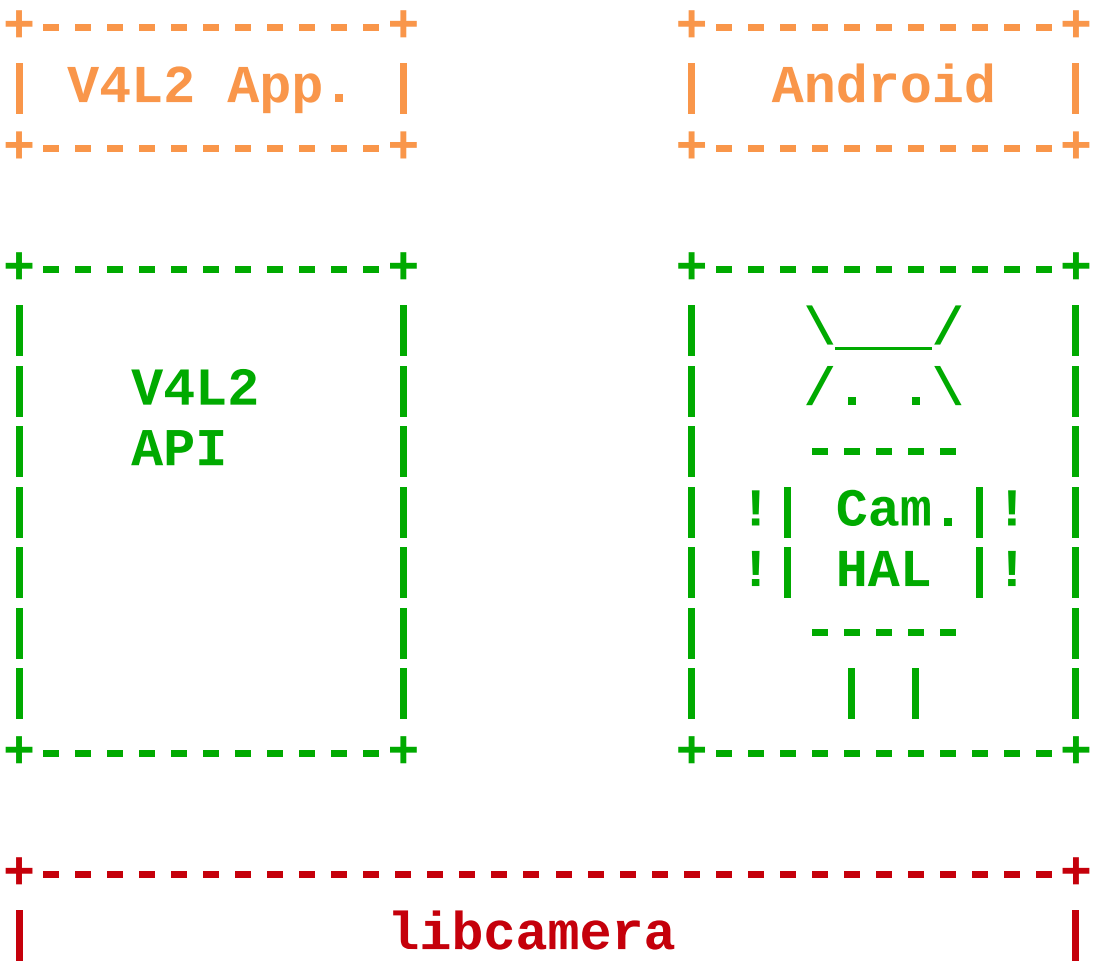
Image Processing Algorithm are loaded as external modules.

Image Processing Algorithms (3A)



Adaptation layers offer backward compatibility with existing APIs...

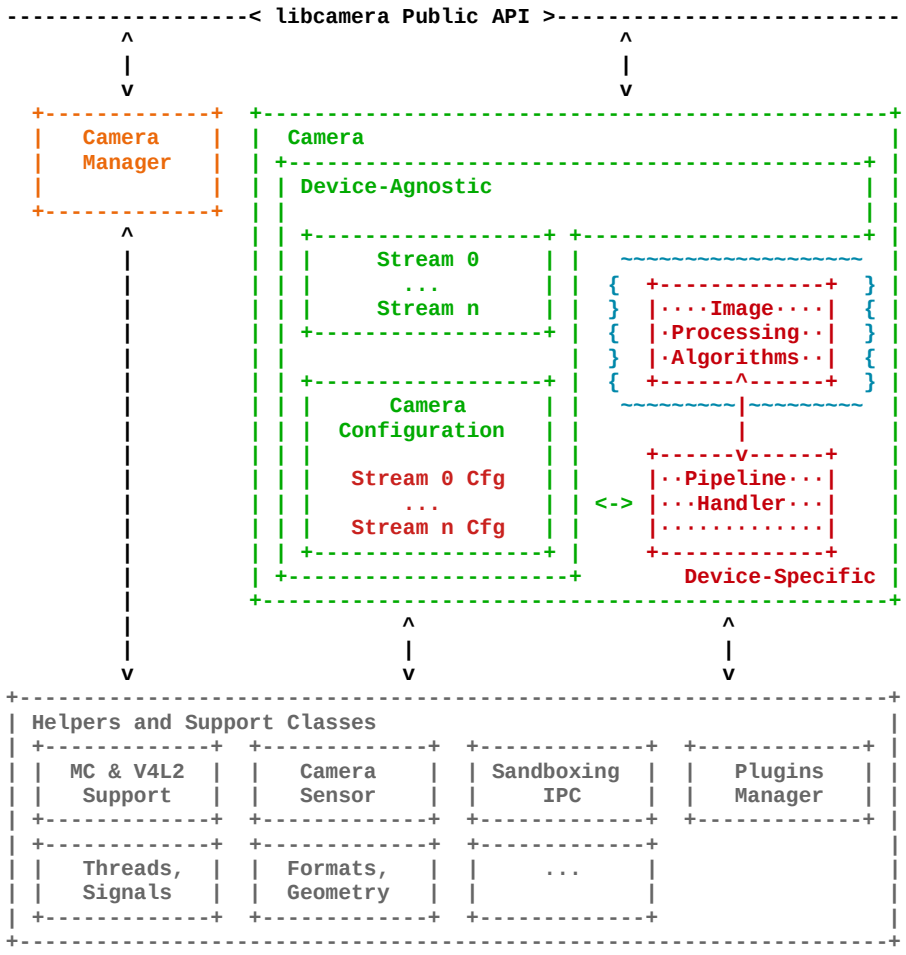
Adaptation



… and integrate libcamera with other operating systems.

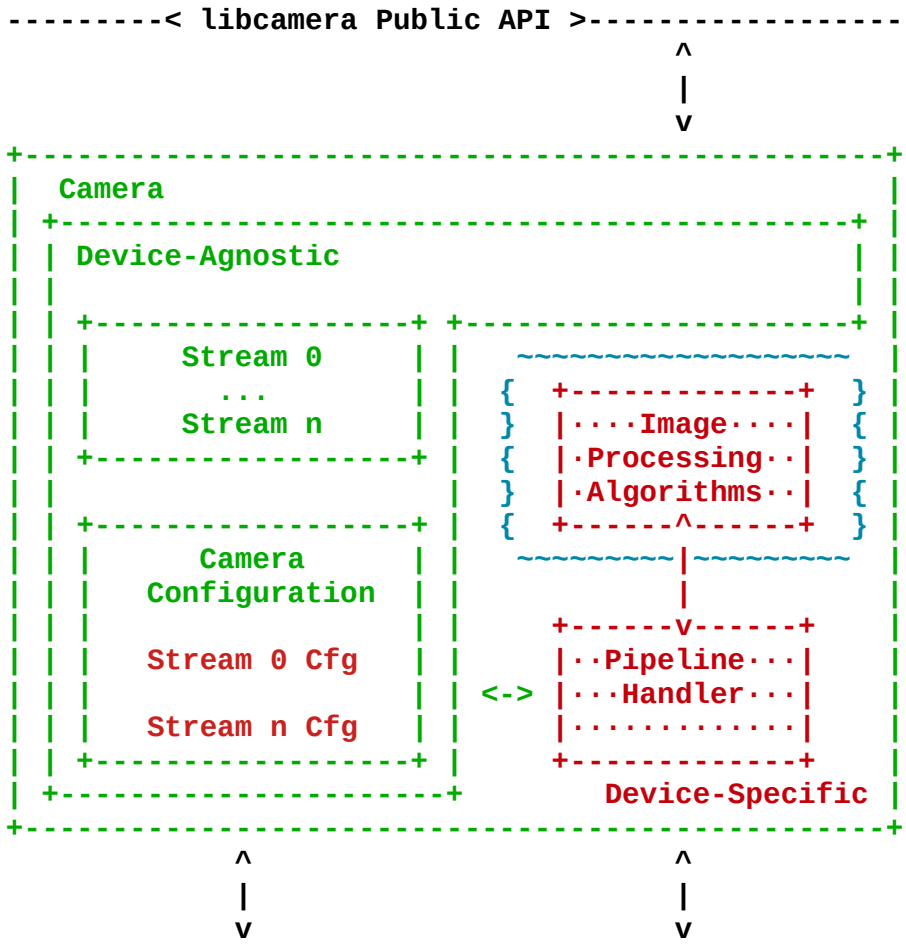
Adaptation

Architecture



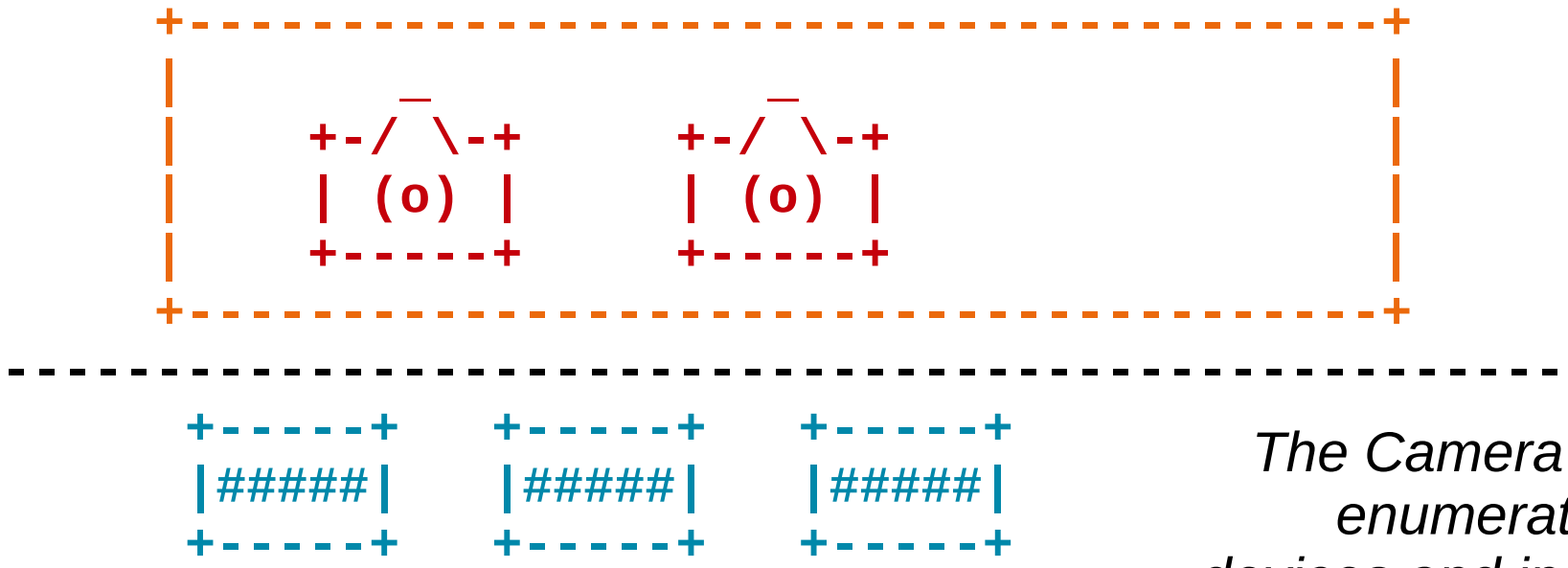
Central to the stack is the Camera object, interfacing to device-specific pipeline handlers.

libcamera architecture



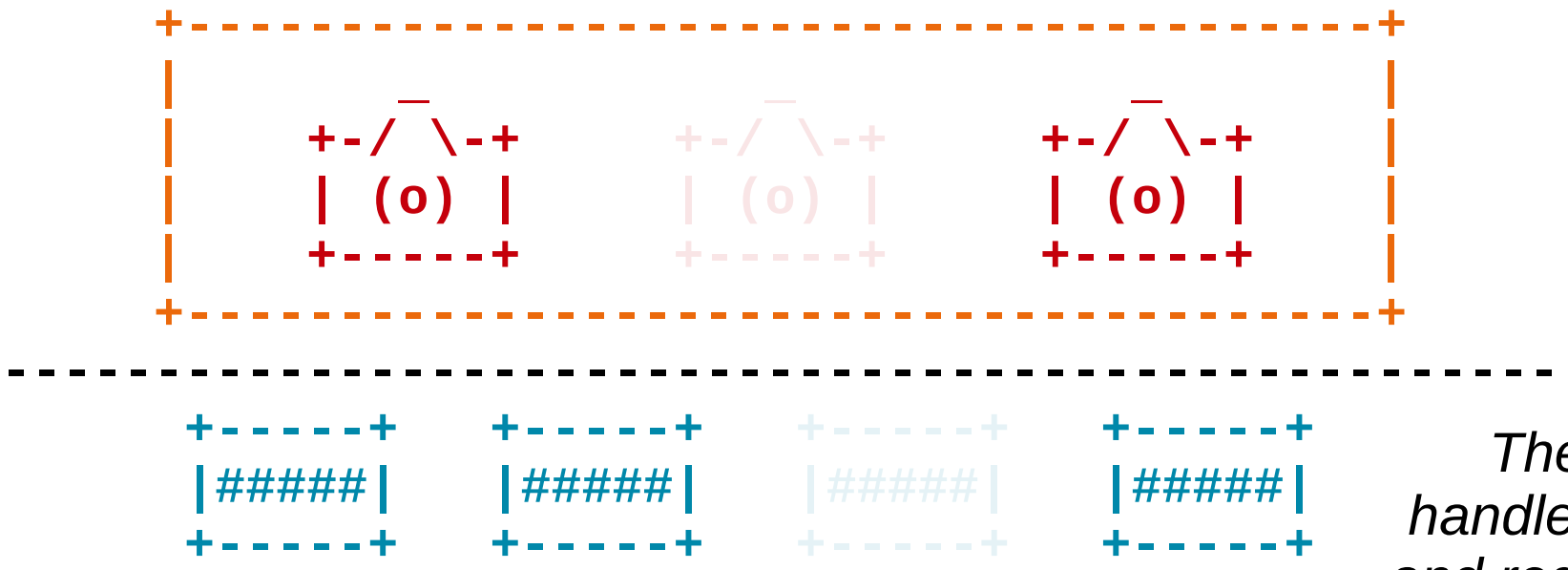
The camera configuration is backed by device-specific validation from the pipeline handler.

Camera Device



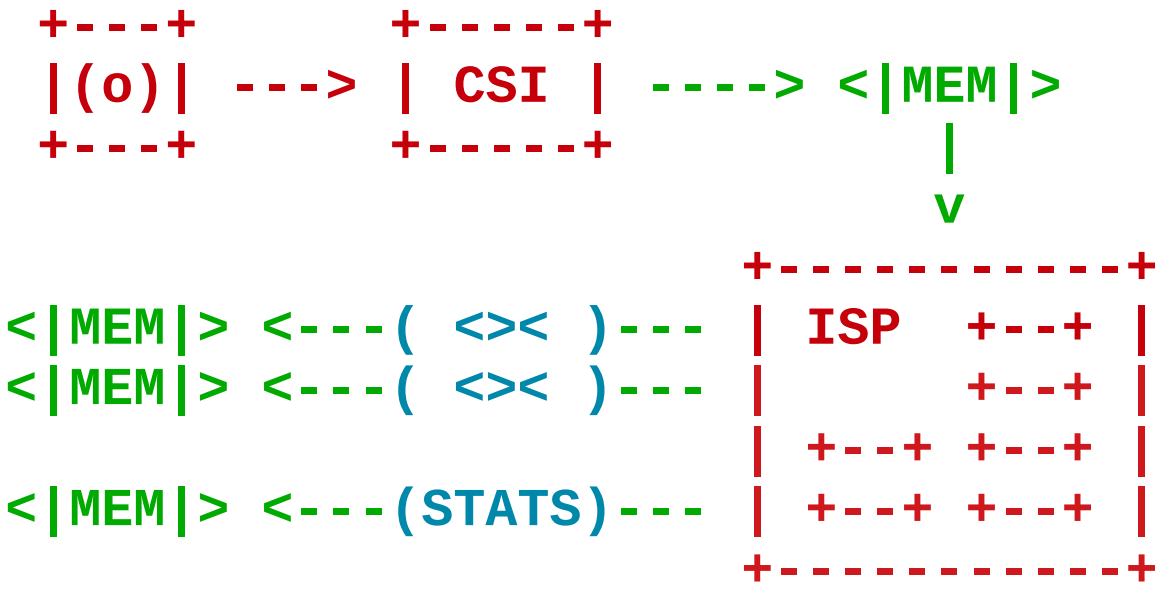
The Camera Manager enumerates media devices and instantiates corresponding pipeline handlers.

Camera Manager



The pipeline handlers create and register one or more cameras.

Camera Manager



The pipeline handler interfaces with all kernel devices. It abstracts them and exposes video streams to upper layers.

Pipeline Handler

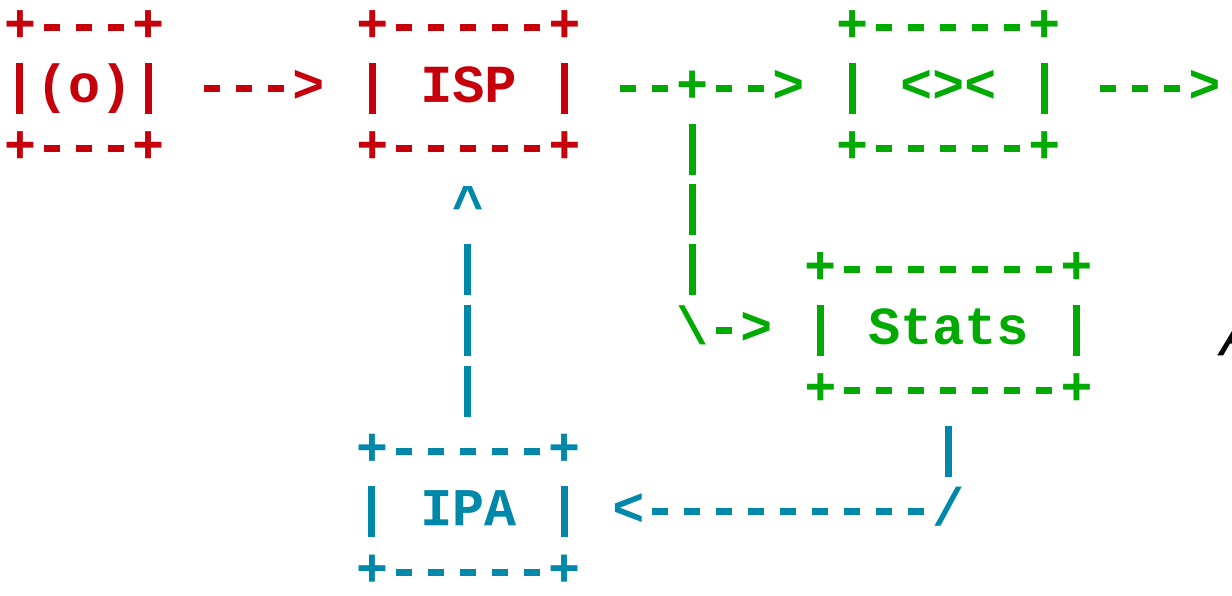
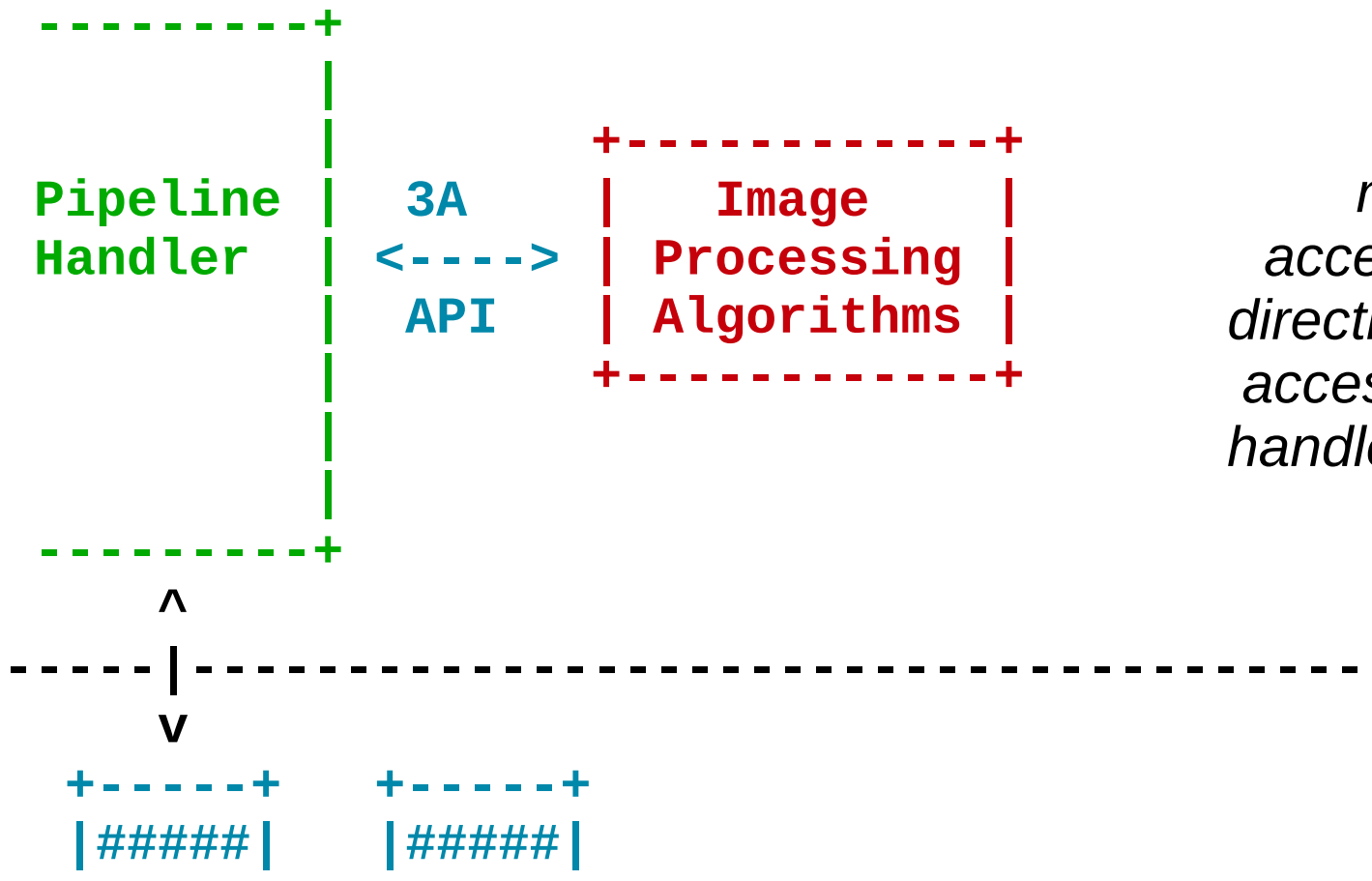


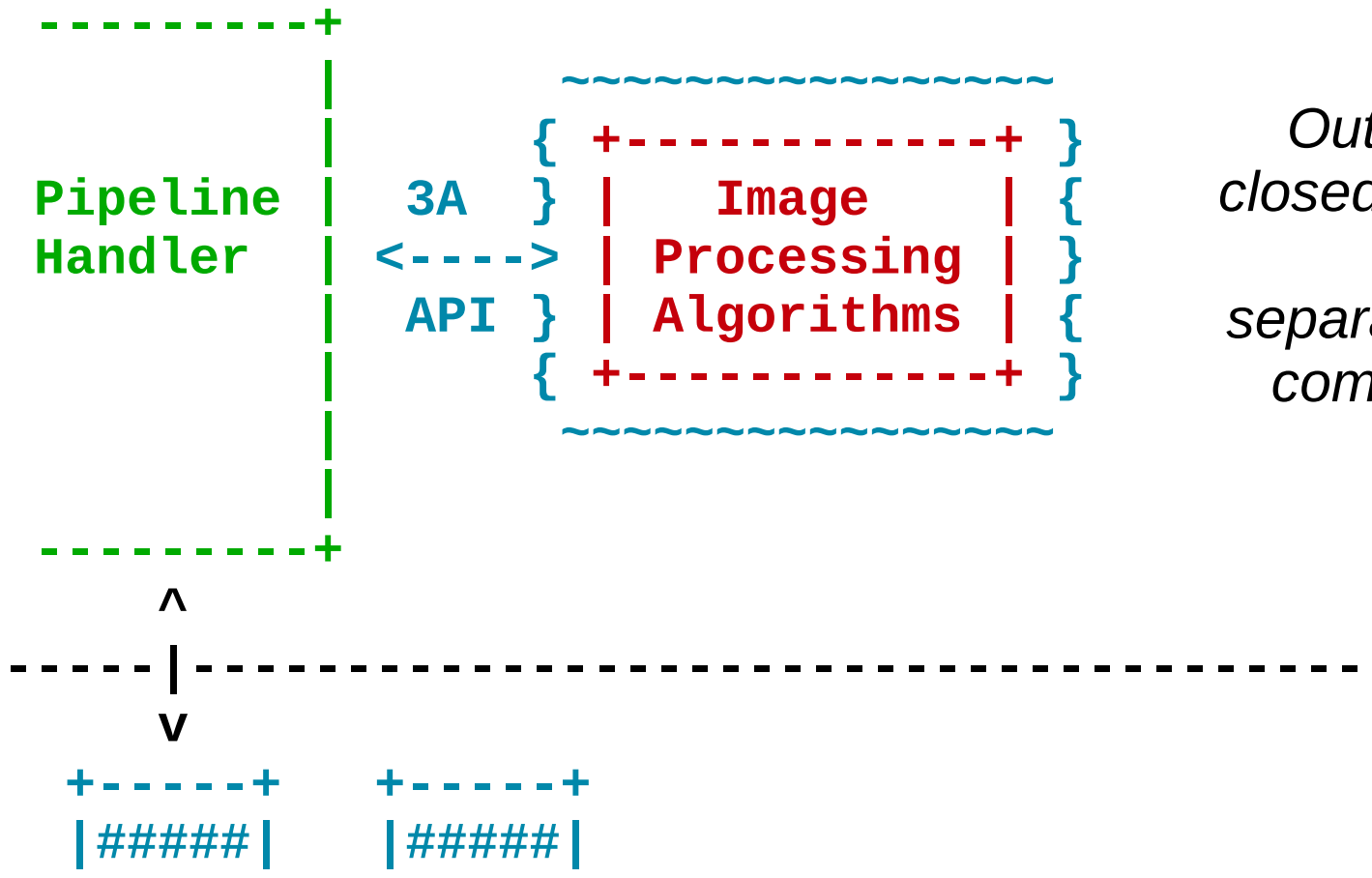
Image Processing Algorithms (IPA) receive statistics from the hardware and compute optimal image parameters.

Image Processing Algorithms



IPAs are separate modules that don't access kernel devices directly. They only have access to their pipeline handler through the IPA API.

Image Processing Algorithms



Out-of-tree (including closed-source) IPAs are sandboxed in a separate process. They communicate with the pipeline handler through IPC.

Image Processing Algorithms

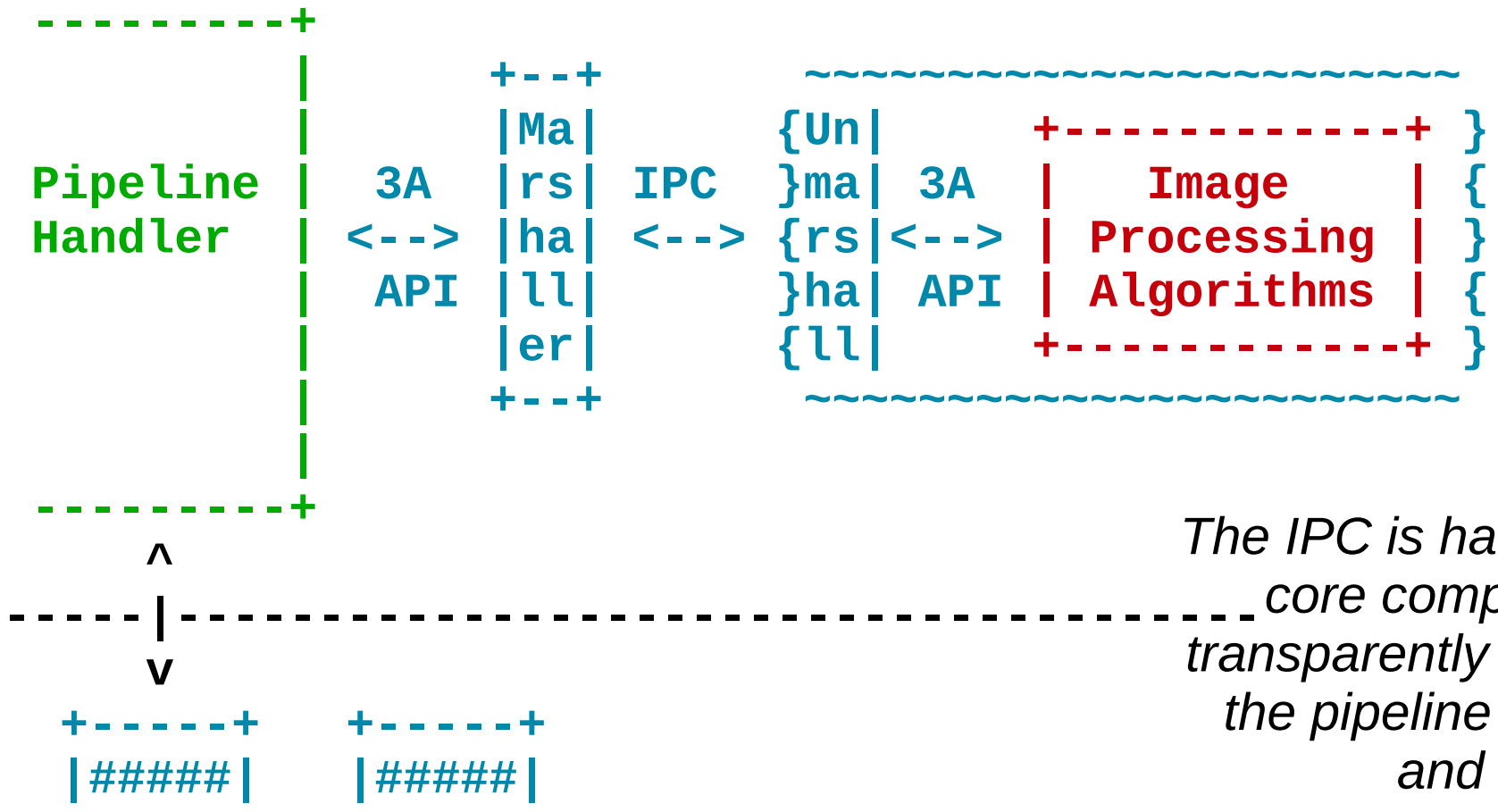
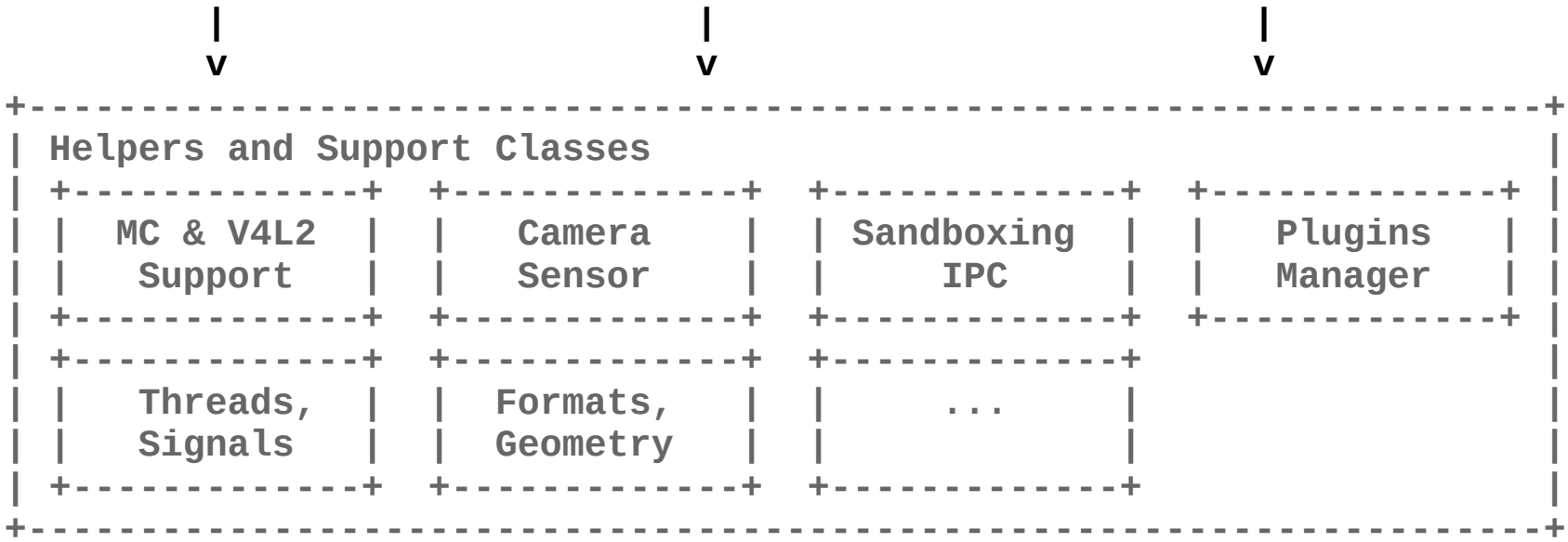
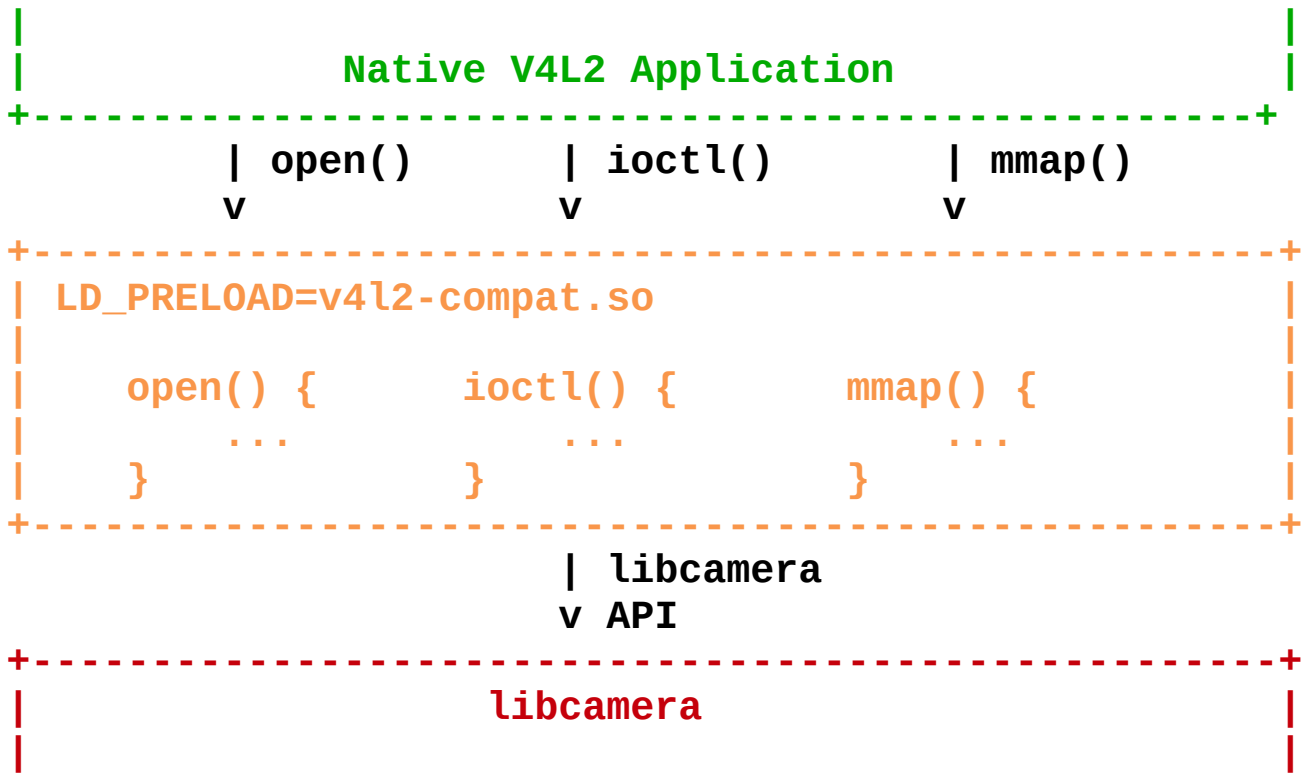


Image Processing Algorithms



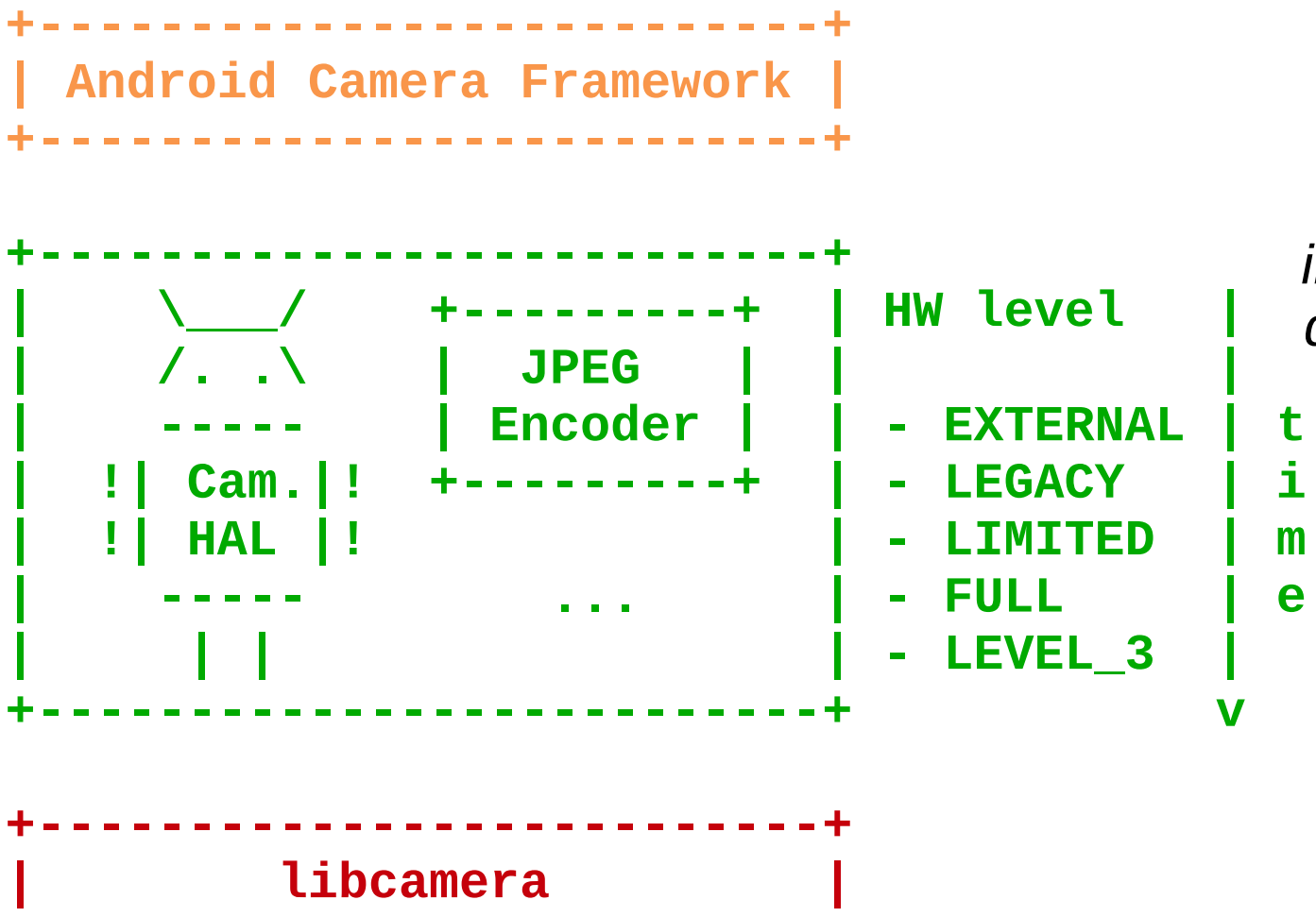
Many helper classes ease the implementation of pipeline handlers for device vendors.

Helpers and Support Classes



Native V4L2 applications are supported through a transparent compatibility layer.

V4L2 Compatibility



A single Android camera HAL module implementation for all devices supported by libcamera.

Android Camera HAL

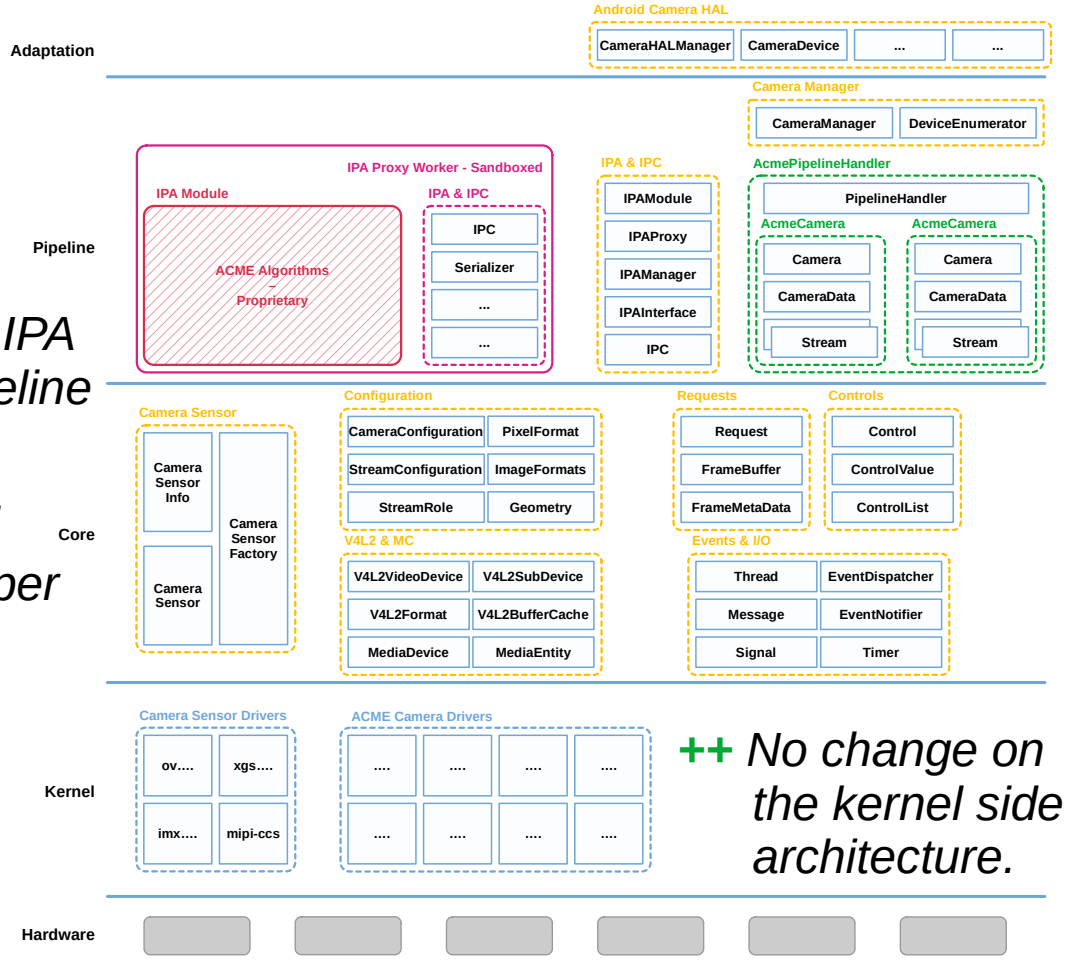
Platform Enablement

IDEAS ON BOARD

++ *IPA module sandboxing for Chrome-OS compliance.*

/!\ *Custom API for IPA module <-> pipeline handler communication.*

++ *libcamera wrapper classes reduce custom code.*



++ *Standard Android Camera HAL Implementation.*

++ *GStreamer, V4L2, ...*

/!\ *Pipeline handler is ACME-specific development.*

++ *Development support available.*

++ *No change on the kernel side architecture.*

/!\ *Implementation changes may be required to mainline drivers.*

Platform Enablement

We drive MC and V4L2 standardization and extensions development according to our needs.

Kernel APIs

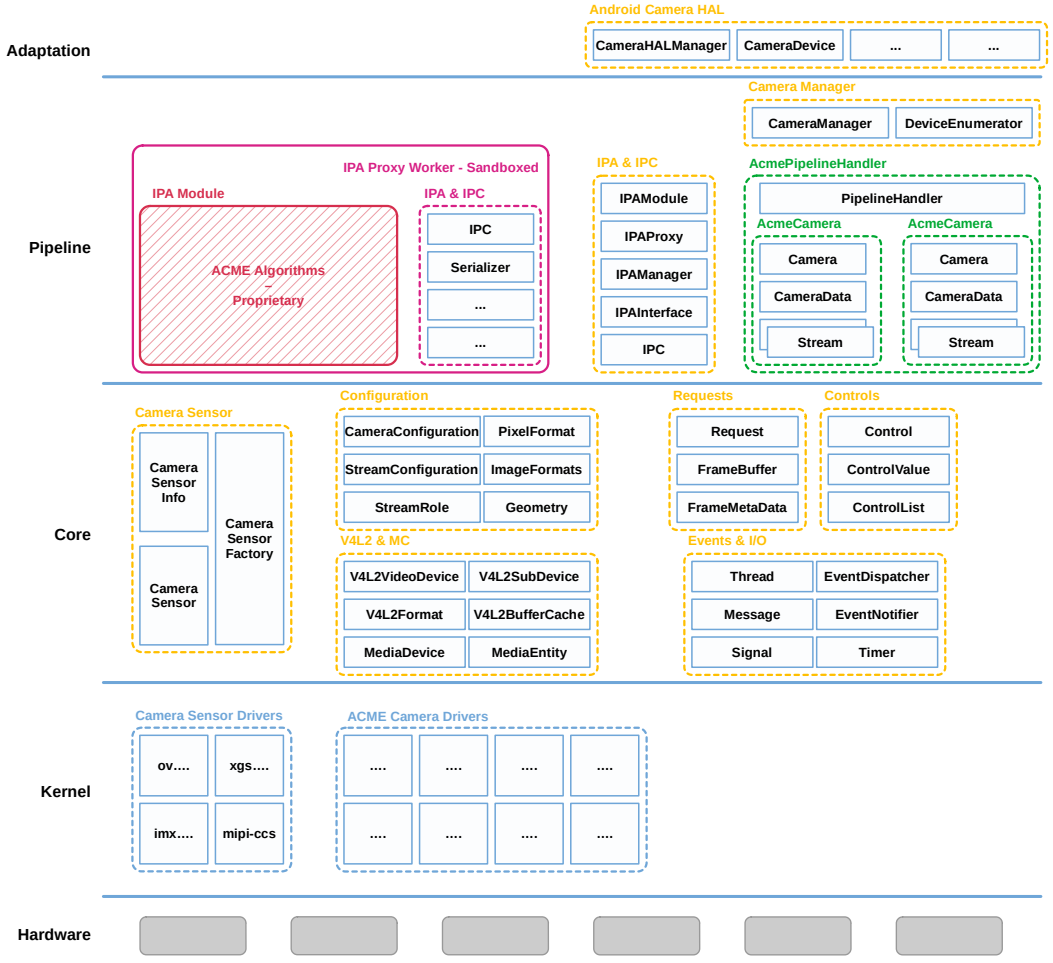
We drive MC and V4L2 standardization and extensions development according to our needs.

libcamera is however a userspace framework, not a hostile takeover of kernel development.

Kernel APIs

Closed-source IPA modules are fully supported.

Pipeline handlers and IPA modules can link to third-party libraries.



The libcamera core, pipeline handlers and adaptation are licensed under LGPL v2.1 or later

Kernel code is licensed under GPL v2.0.

Licensing

The libcamera core is licensed under the LGPL v2.1 or later.

Licensing

The libcamera core is licensed under the LGPL v2.1 or later.

Changes need to be published according to the license. This includes pipeline handlers.

Licensing

Licensing

The libcamera core is licensed under the LGPL v2.1 or later.

Changes need to be published according to the license. This includes pipeline handlers.

Upstreaming is not mandatory but highly recommended (forks are costly to maintain).

Status

IDEAS ON BOARD

Features	Status
Core	Multi-camera, multi-stream, per-frame control, hotplug.
Documentation	Extensive API documentation and high-level tutorials and guides available.
Supported platforms	Raspberry Pi 3&4, Intel Kaby Lake, Rockchip RK3399, UVC, NXP i.MX7, Allwinner A31
IPA modules	Raspberry Pi
Adaptation layers	GStreamer source element, Android camera HAL v3.3, V4L2
Tooling	Camera Tuning Tool (Rasperry Pi), tracing infrastructure and analysis script.
Applications	cam (command line Swiss army knife), qcam (GUI), simple-cam (tutorial)

Completed

Features	Status
Reprocessing API	Design and prototype finalization. Will include abstract pipeline model.
IPA module isolation with IPC	Infrastructure under development, based on Chrome mojom IDL.
Supported platforms	MediaTek MT8183 (with YUV sensors)
Open-source IPA modules	Initial development of Rockchip rkisp1 and Intel IPU3 support.
Controls and properties	New controls and properties are continuously added on a per-need basis.
Public API freeze	Minor API changes will remain backward-compatible (extensible API design, d-pointer design pattern, …).

Ongoing – Core

Features	Status
LIMITED and FULL CTS compliance	Core infrastructure ready, controls and properties (static, control and dynamic metadata) being developed incrementally.
HAL v3.5(+)	On the roadmap, on hold due to lack of development and test platform. Future Android camera HAL API extensions will be implemented (including extensions to the libcamera core if needed).
RAW/YUV reprocessing API	Will be started once the feature is available in the libcamera core.
Zero Shutter Lag	HAL-based ZSL implemented using the libcamera reprocessing API.

Ongoing – Android HAL

Features	Status
Zero shutter lag	Will be possible through the reprocessing API. We are considering a high-level “use cases” library on top of libcamera for ZSL and similar features.
Integration	Native support in Chromium web browser available at https://github.com/libcamera-org/chromium.
Language bindings	Python
Community development	Expand the user base by supporting the Windows-based Intel Kaby Lake machines.

Ongoing – Extensions

Chromium on MS Surface Go 2

Roadmap

IDEAS ON BOARD

Features	Status
Logical camera devices (W+T zoom, power saving, ...)	Not planned yet, missing development and test platform.
Still image trigger sequence (focus & flash)	Not planned yet, missing development and test platform.
Per-stream controls	Concept approved, will be scheduled in the future.
GPU-based processing	Proof of concept shader code in qcam test application, should be leveraged to create GPU-based ISP.
Exposure bracketing HDR	Similarly to ZSL, could be implemented in a “use cases” library. A solution for device-assisted HDR (hardware merging or software merging based on hardware-generated metadata) is needed.

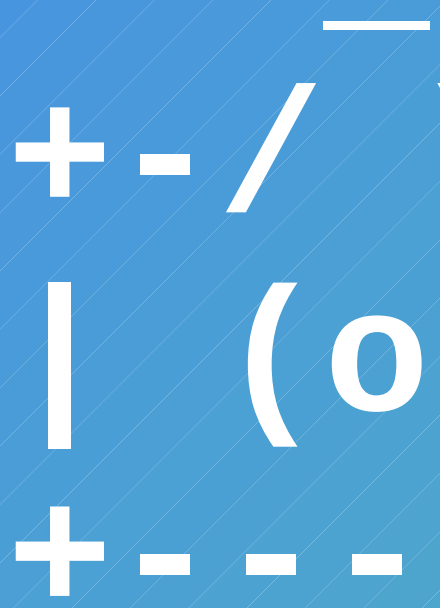
Future Features – Core

Features	Status
Frameworks	Multi-stream GStreamer support, PipeWire, Firefox, Qt Multimedia, Electron, <insert your framework here>, ...
Applications	Firefox, <insert your application here>, ...
Operating Systems	Chrome OS, Android, Linux distributions, Buildroot, OpenEmbedded, ...

Future Features – Integration

Features	Status
New devices support	Ongoing discussions with three SoC/system vendors, informal commitment from two of them.
Open-source IPA modules	Cross-platform core library, community effort for platform-specific support (any volunteer for the N900/N9 ?), long term work to convince device vendors.

Future Features – Devices



libcamera

libcamera-devel@lists.libcamera.org
irc://chat.freenode.net/#libcamera

laurent.pinchart@ideasonboard.com

Contact

Merci.

IDEAS ON BOARD