libcamera: Making Complex Cameras Easy

OSS Japan 2019
Tokyo, Japan

Laurent Pinchart
laurent.pinchart@ideasonboard.com

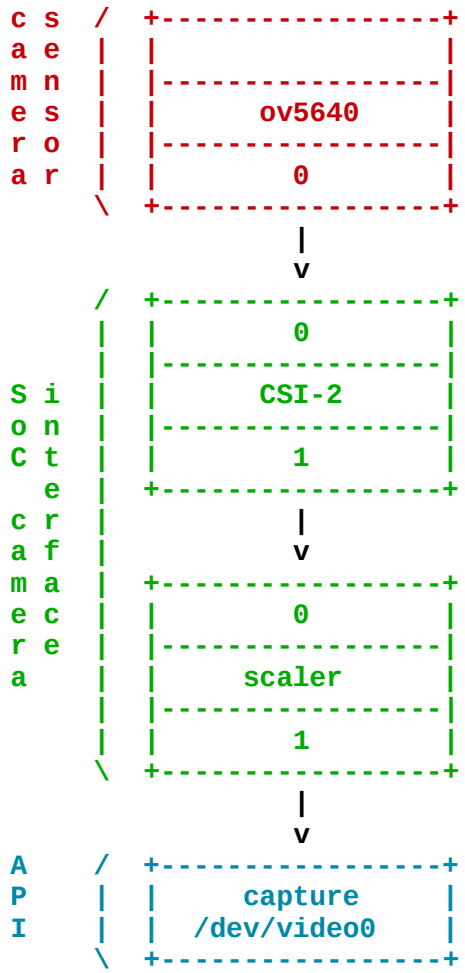
Contents

- libcamera

libcamera

Cameras are complex devices that need heavy hardware image processing operations. Control of the processing is based on advanced algorithms that must run on a programmable processor. This has traditionally been implemented in a dedicated MCU in the camera, but in embedded devices algorithms have been moved to the main CPU to save cost. Blurring the boundary between camera devices and Linux often left the user with no other option than a vendor-specific closed-source solution.

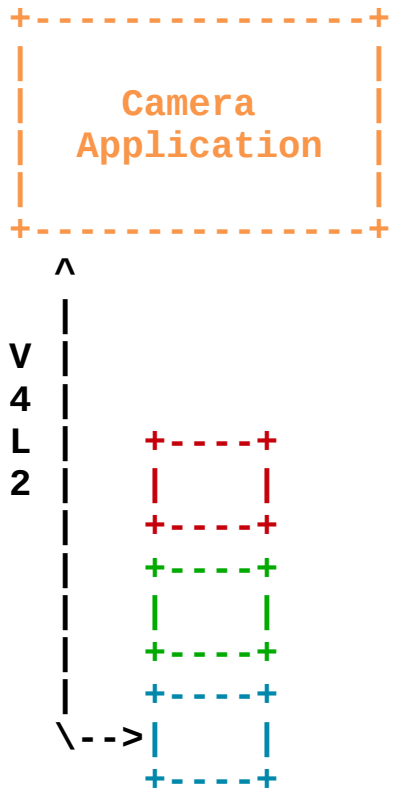
To address this problem the Linux media community has very recently started collaboration with the industry to develop a camera stack that will be open-source-friendly while still protecting vendor core IP. libcamera was born out of that collaboration and will offer modern camera support to Linux-based systems, including traditional Linux distributions, ChromeOS and Android.



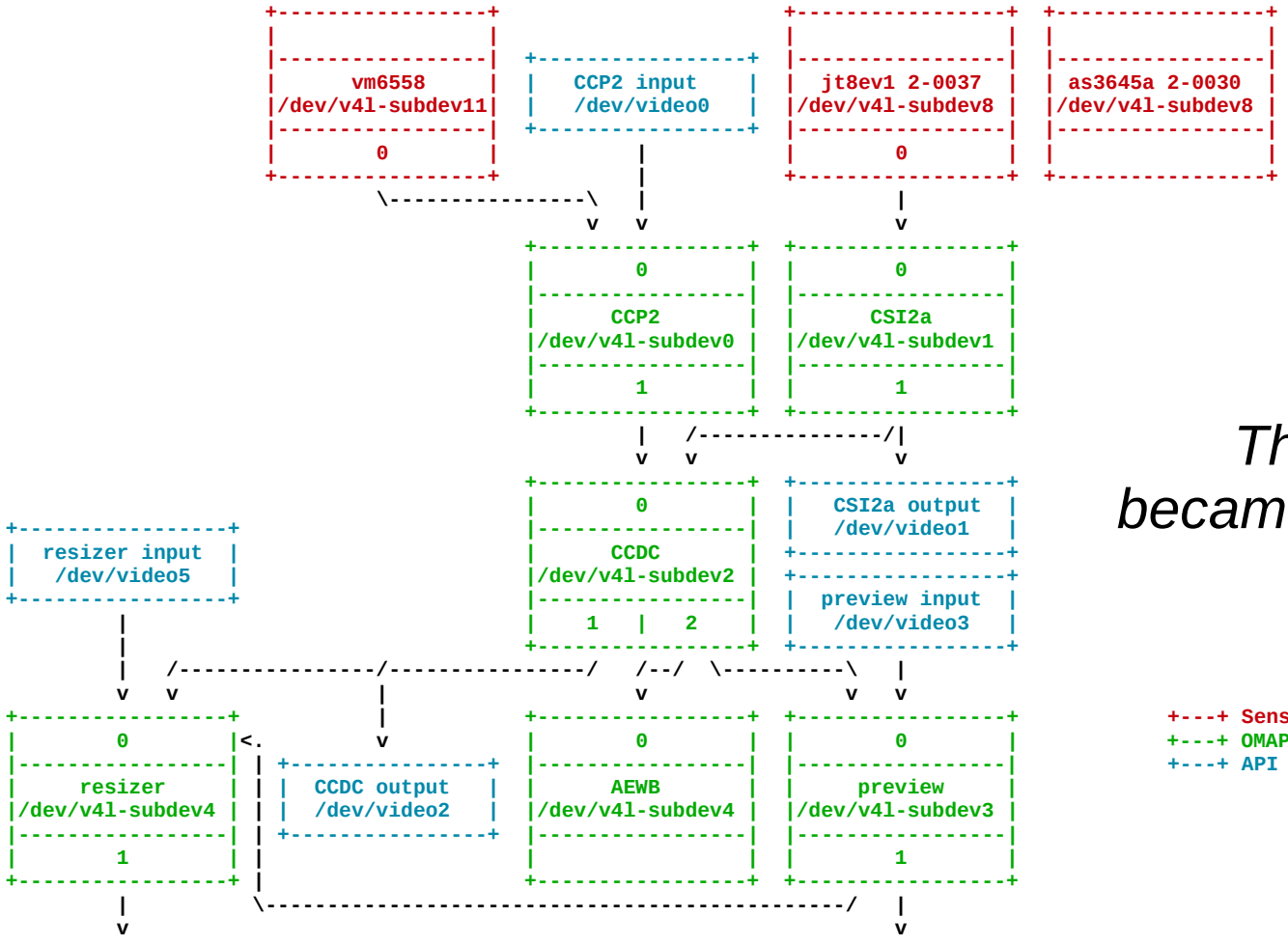
In the beginning were simple pipelines...

Why?

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



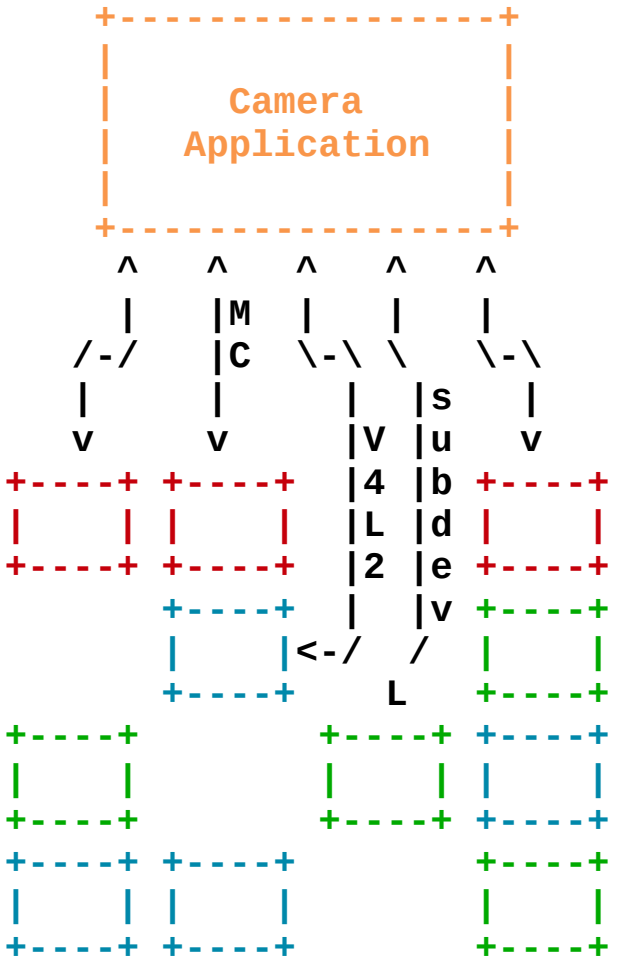
Why?



Then the world became complex ...

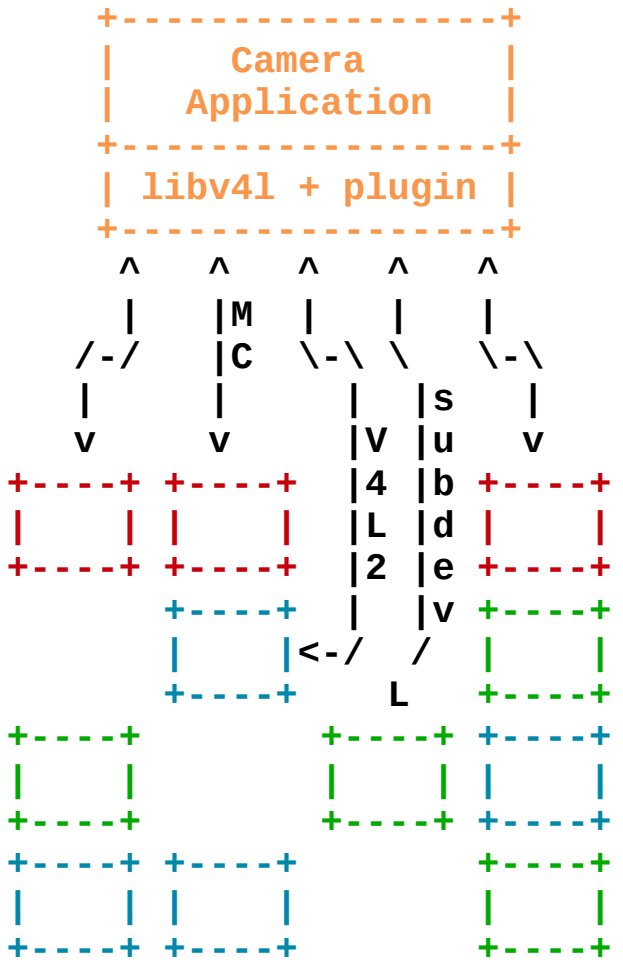
Why?

… and application developers were left suffering.



Why?

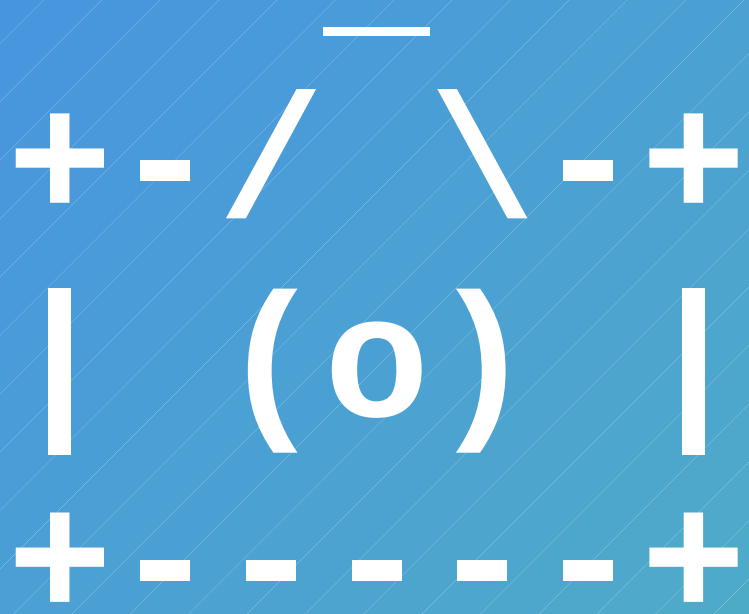
Solutions were proposed...



… but never implemented.

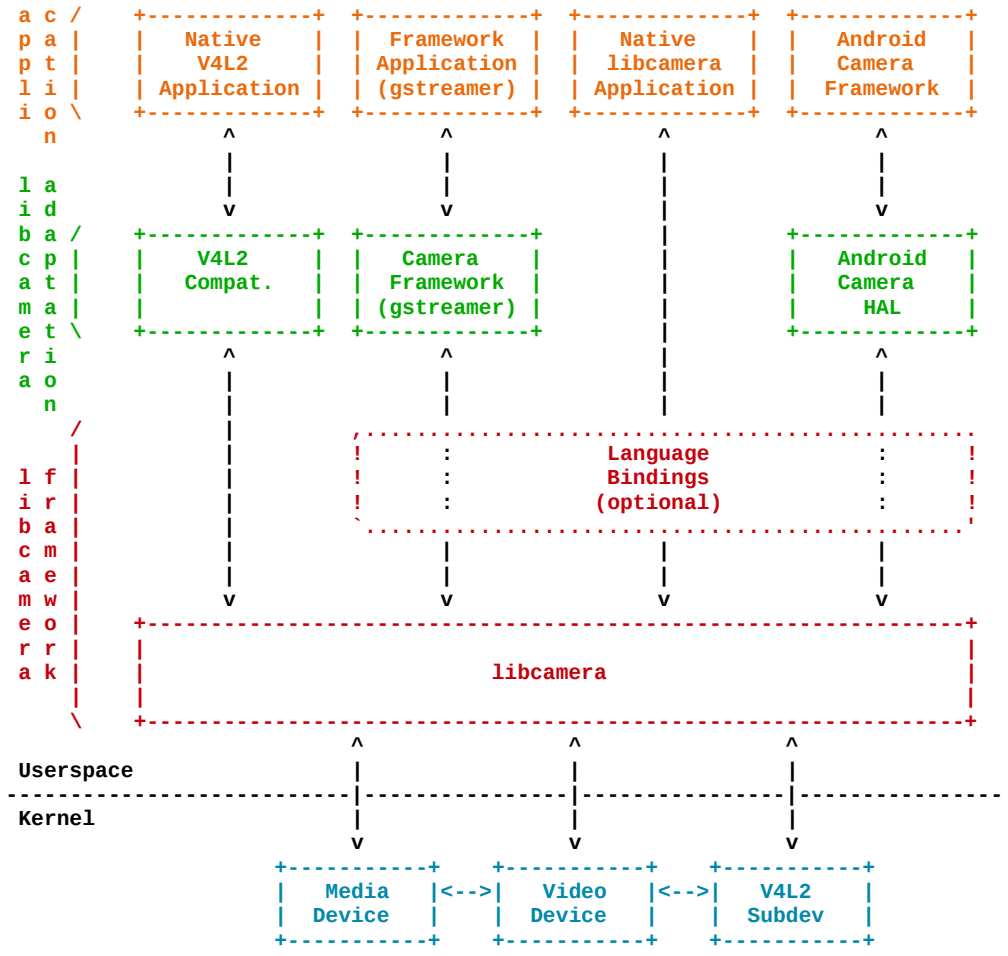
Why?

Then hope came back.



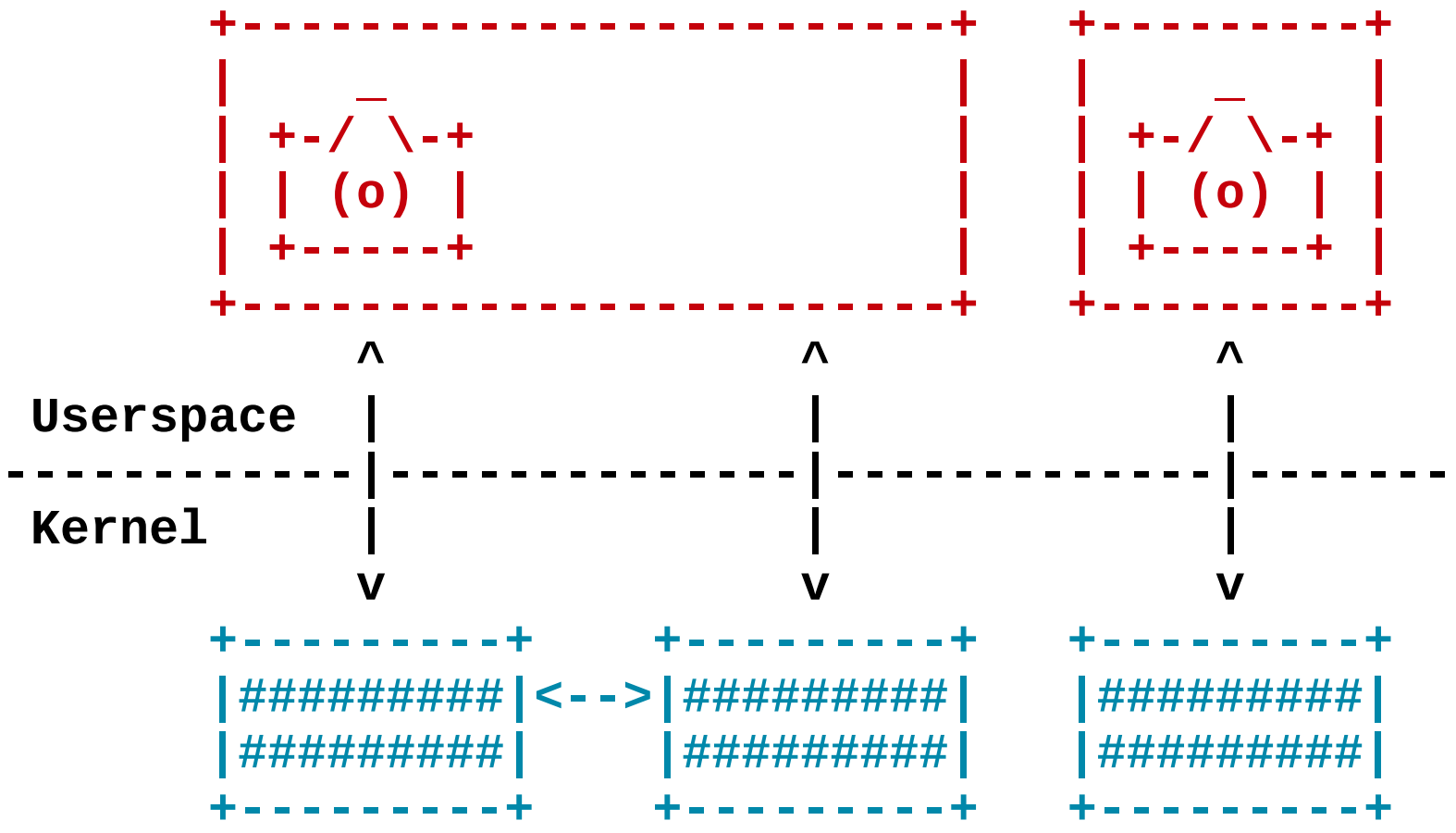
libcamera

libcamera provides a complete userspace camera stack.



The 'mesa' of the camera world.

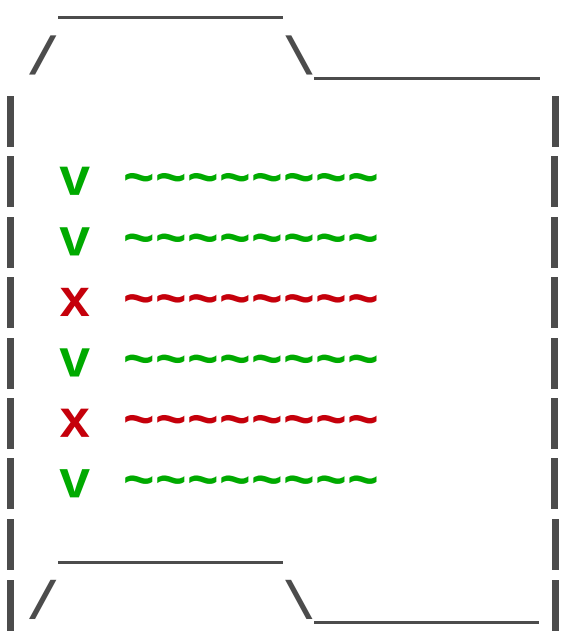
Camera Stack



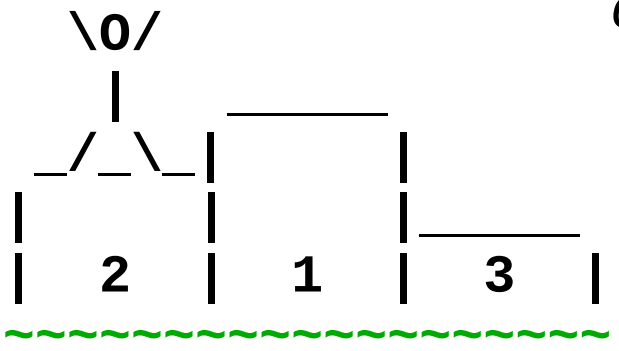
libcamera enumerates cameras...

Camera Devices & Enumeration

… and exposes their capabilities.



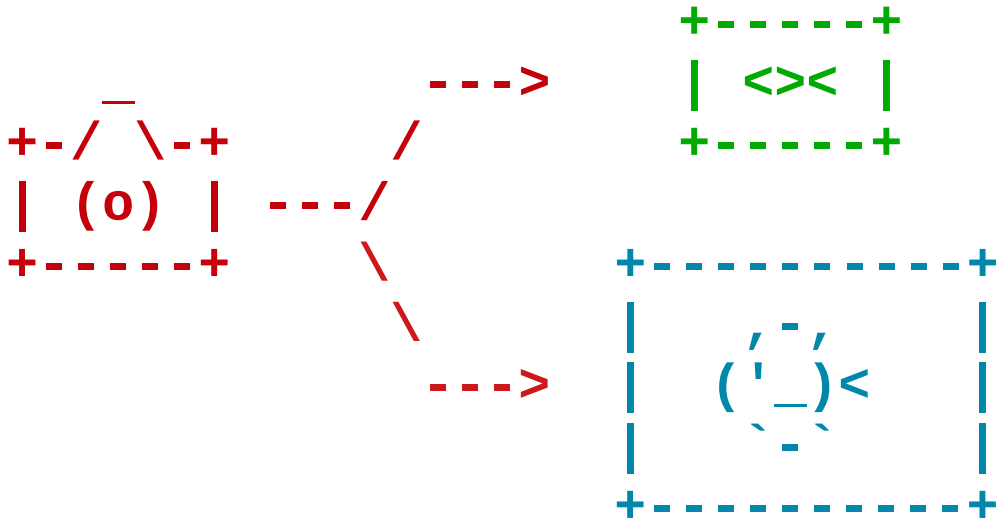
Capabilities



Profiles

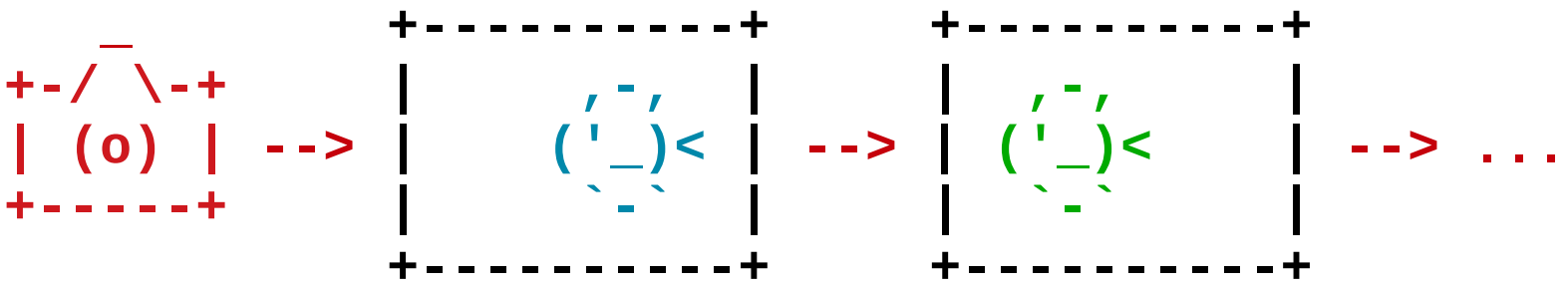
Capabilities & Profiles

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.

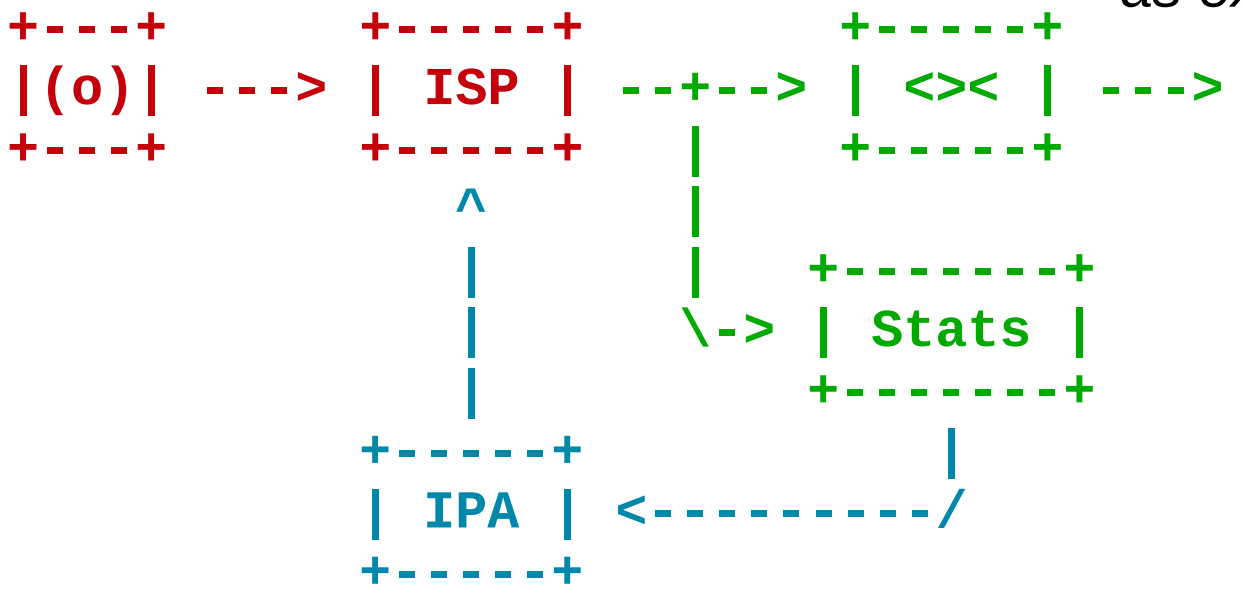
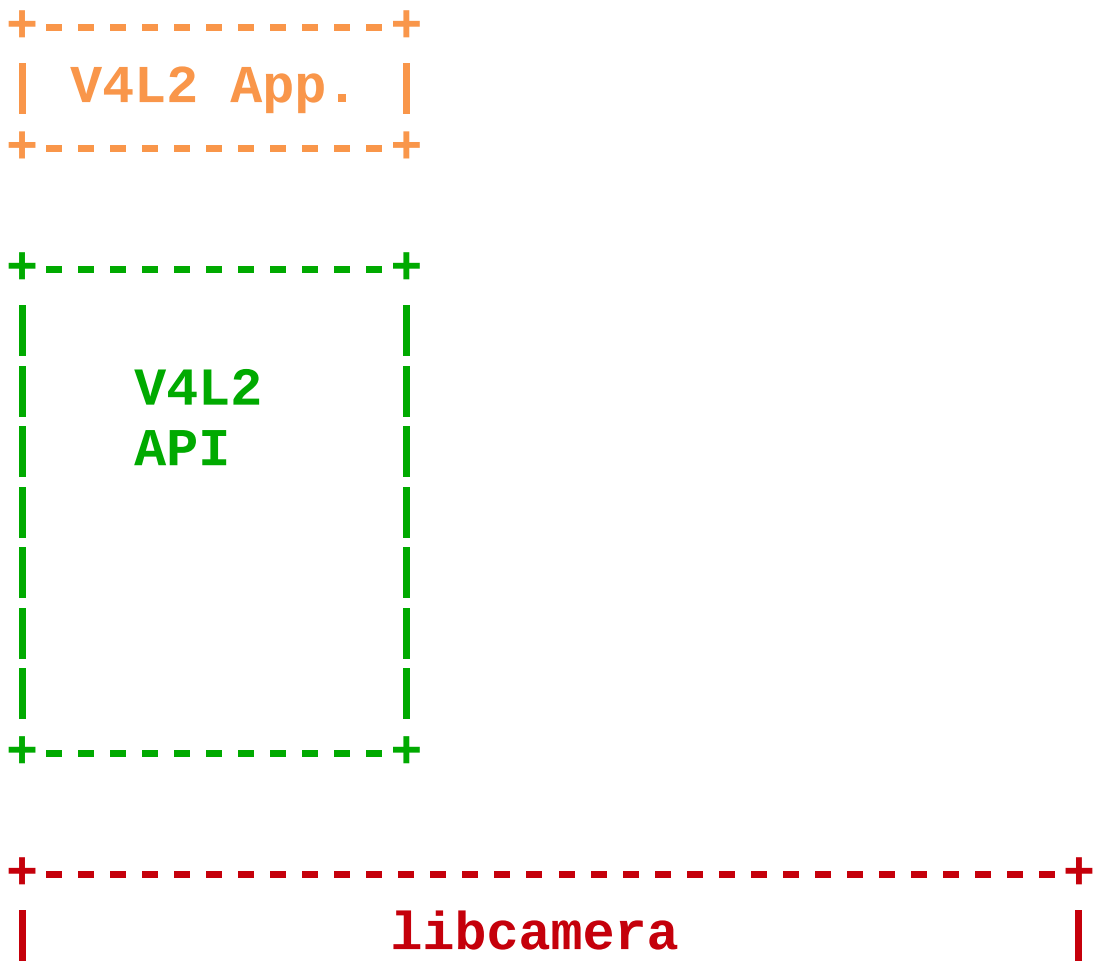
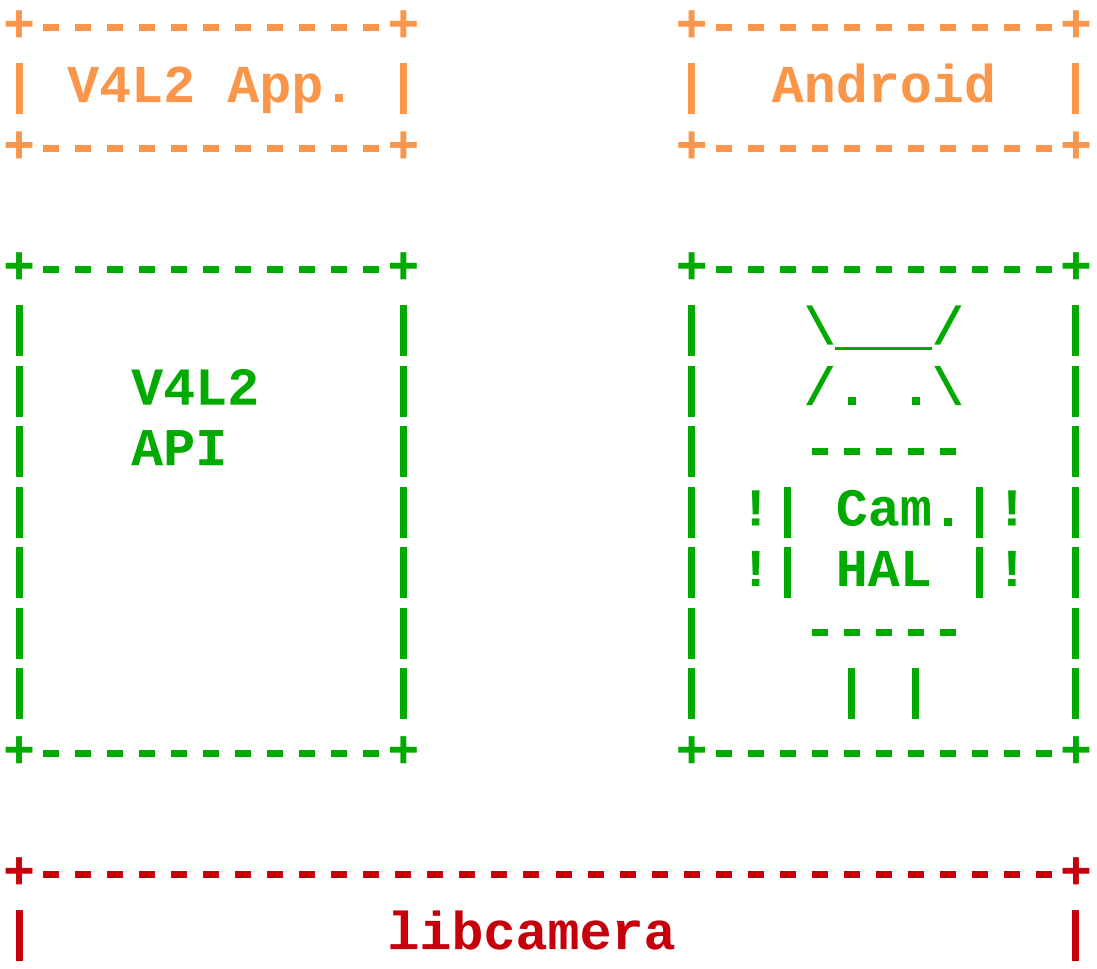


Image Processing Algorithms (3A)



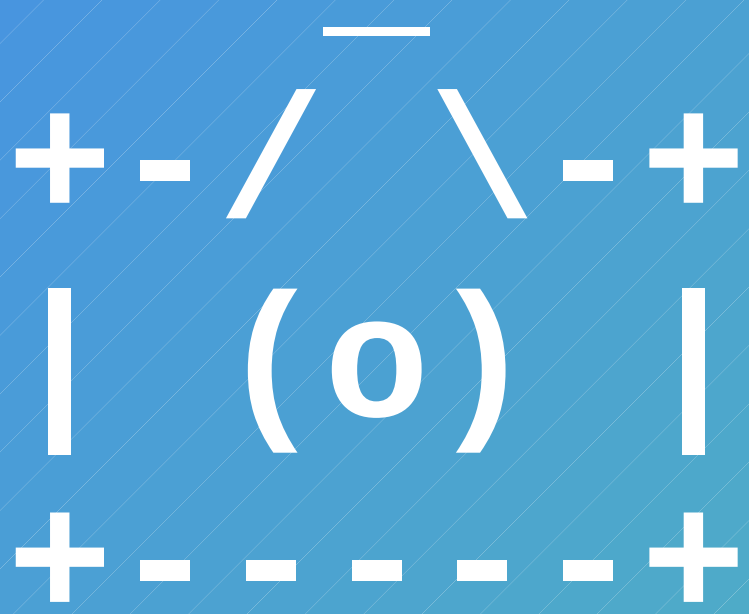
Adaptation layers offer backward compatibility with existing APIs...

Adaptation

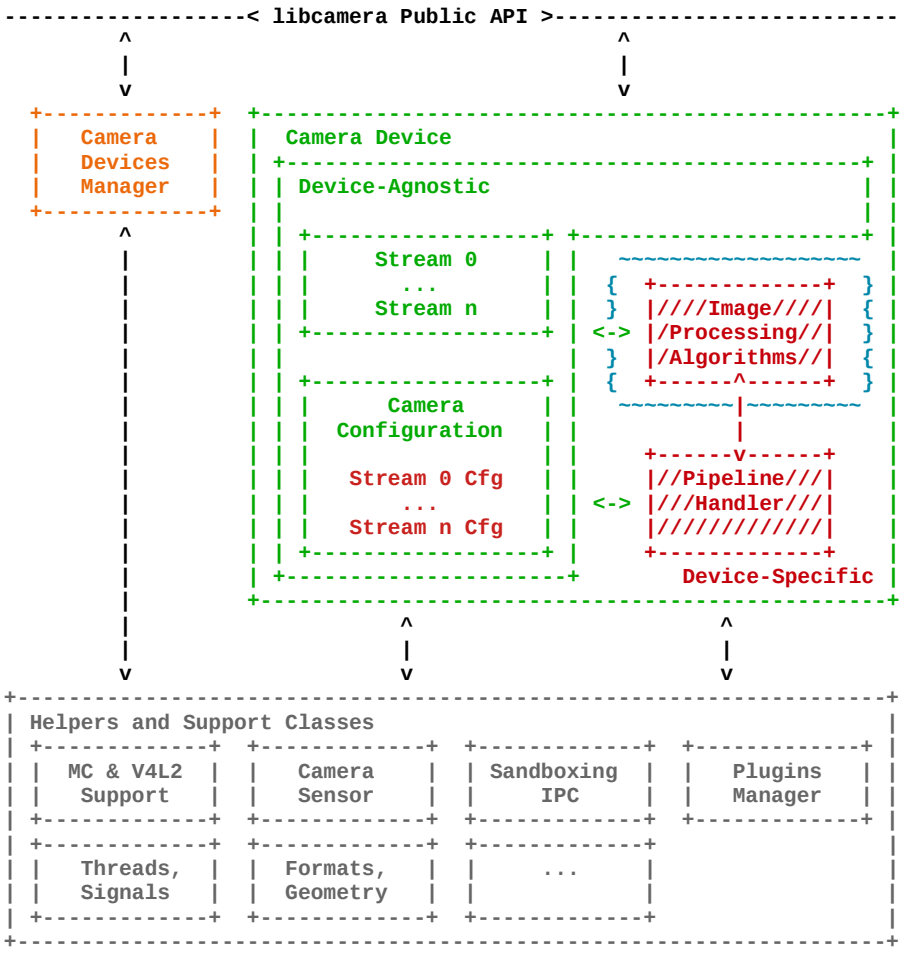


… and integrate libcamera with other operating systems.

Adaptation

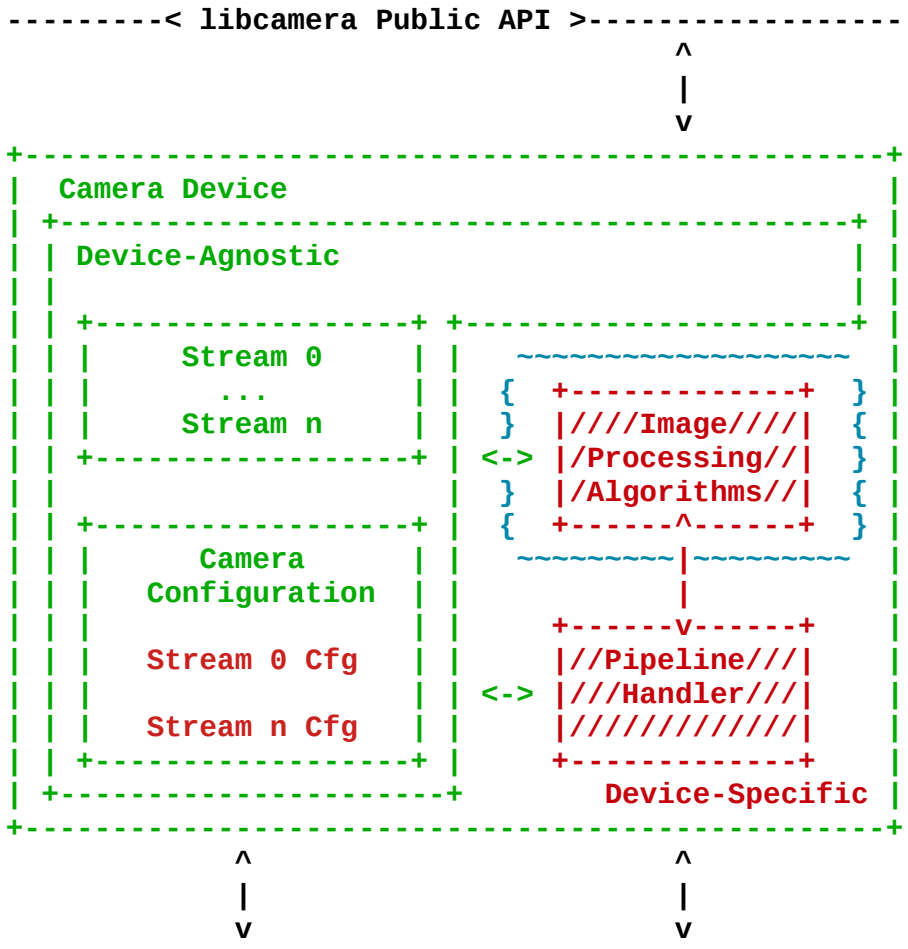


libcamera



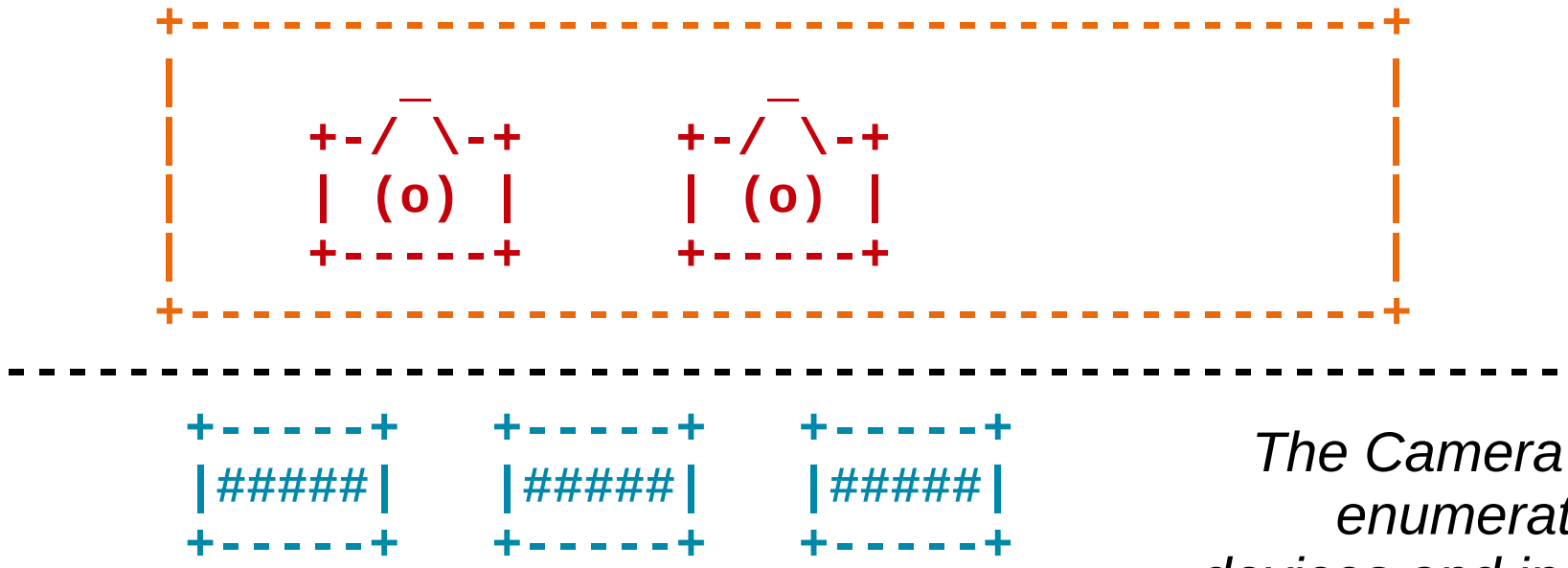
Central to the stack is the Camera object, interfacing to device-specific pipeline handlers.

libcamera architecture



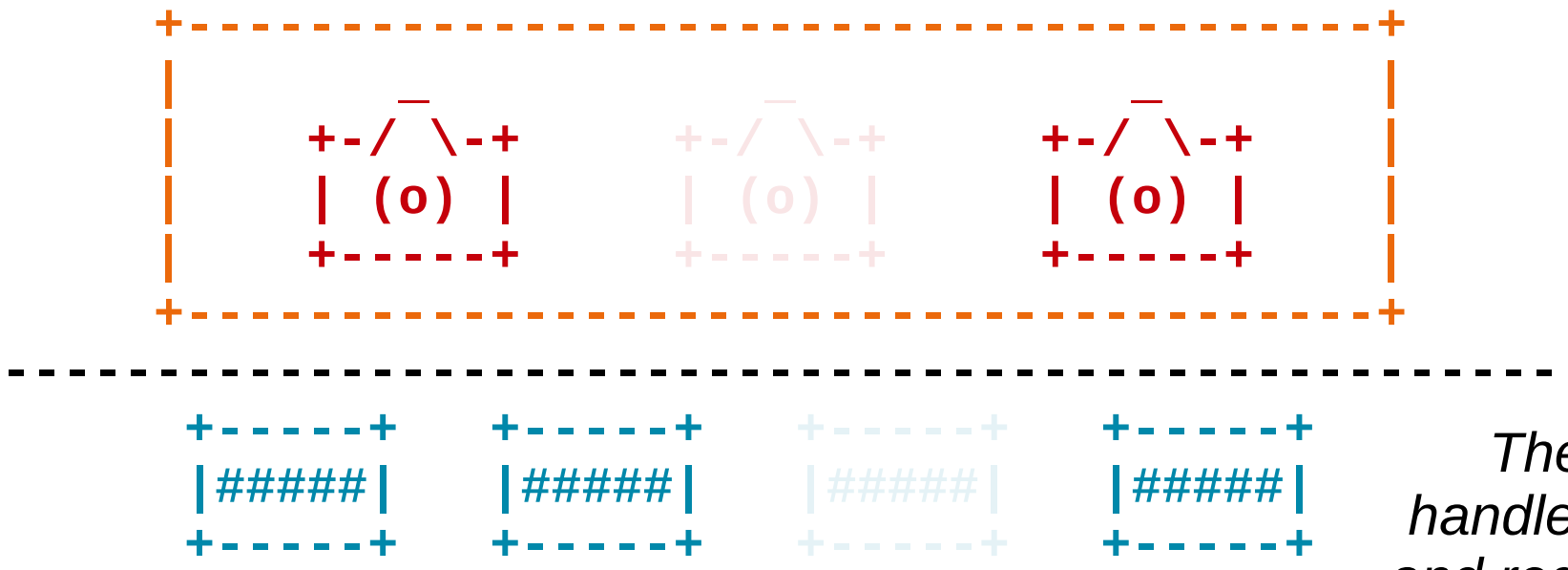
The camera configuration is backed by device-specific validation from the pipeline handler.

Camera Device



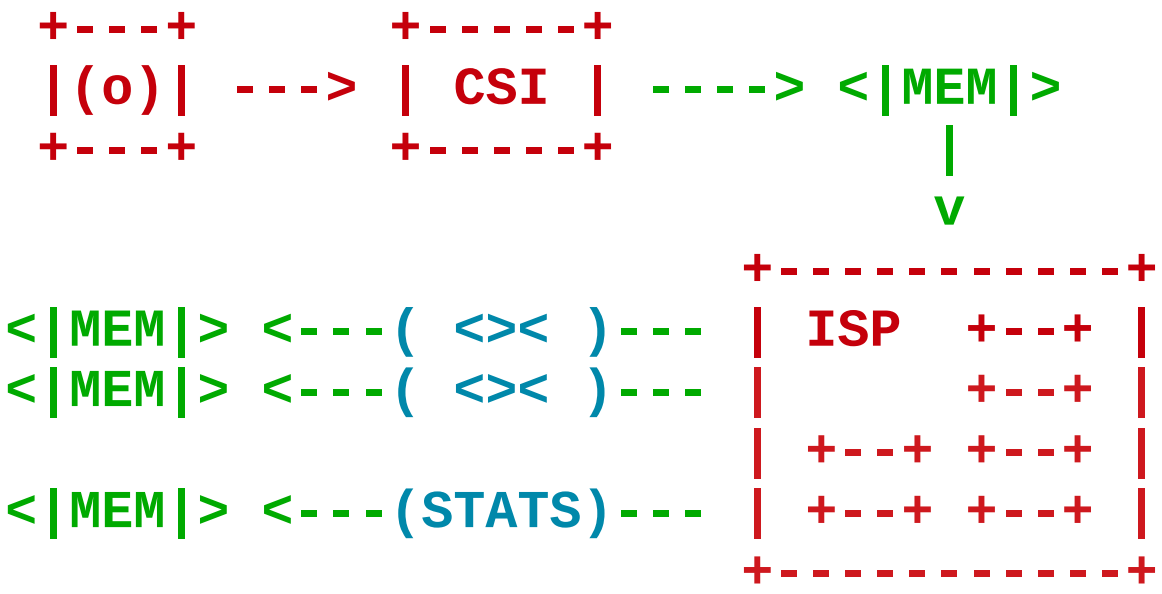
The Camera Manager enumerates media devices and instantiates corresponding pipeline handlers.

Camera Devices Manager



The pipeline handlers create and register one or more cameras.

Camera Devices Manager



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

Pipeline Handler

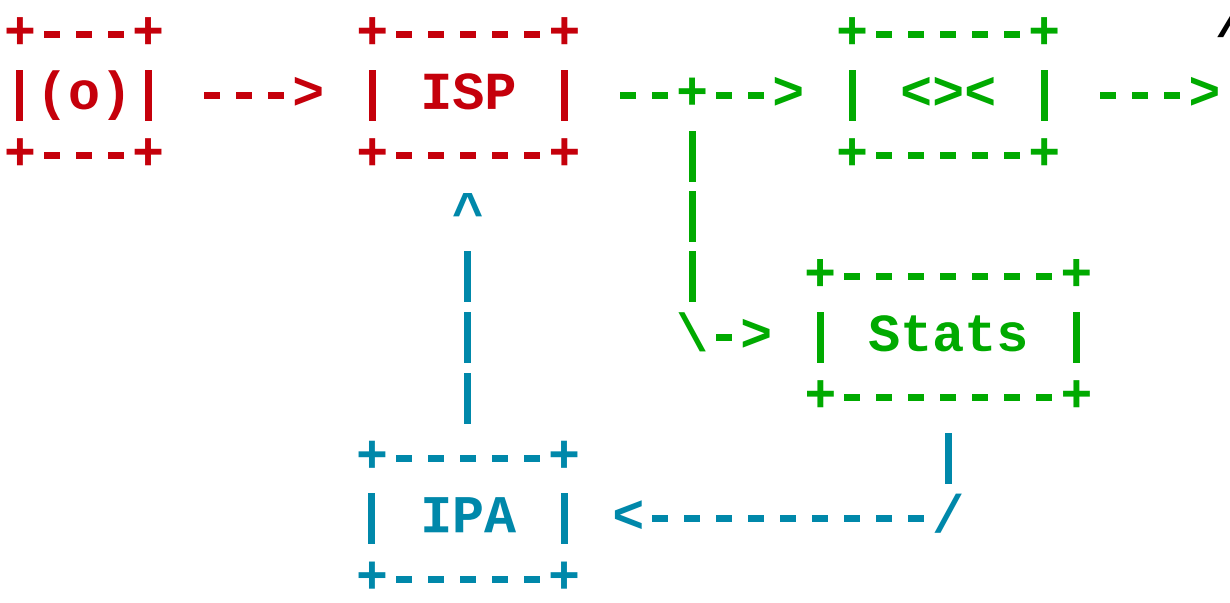
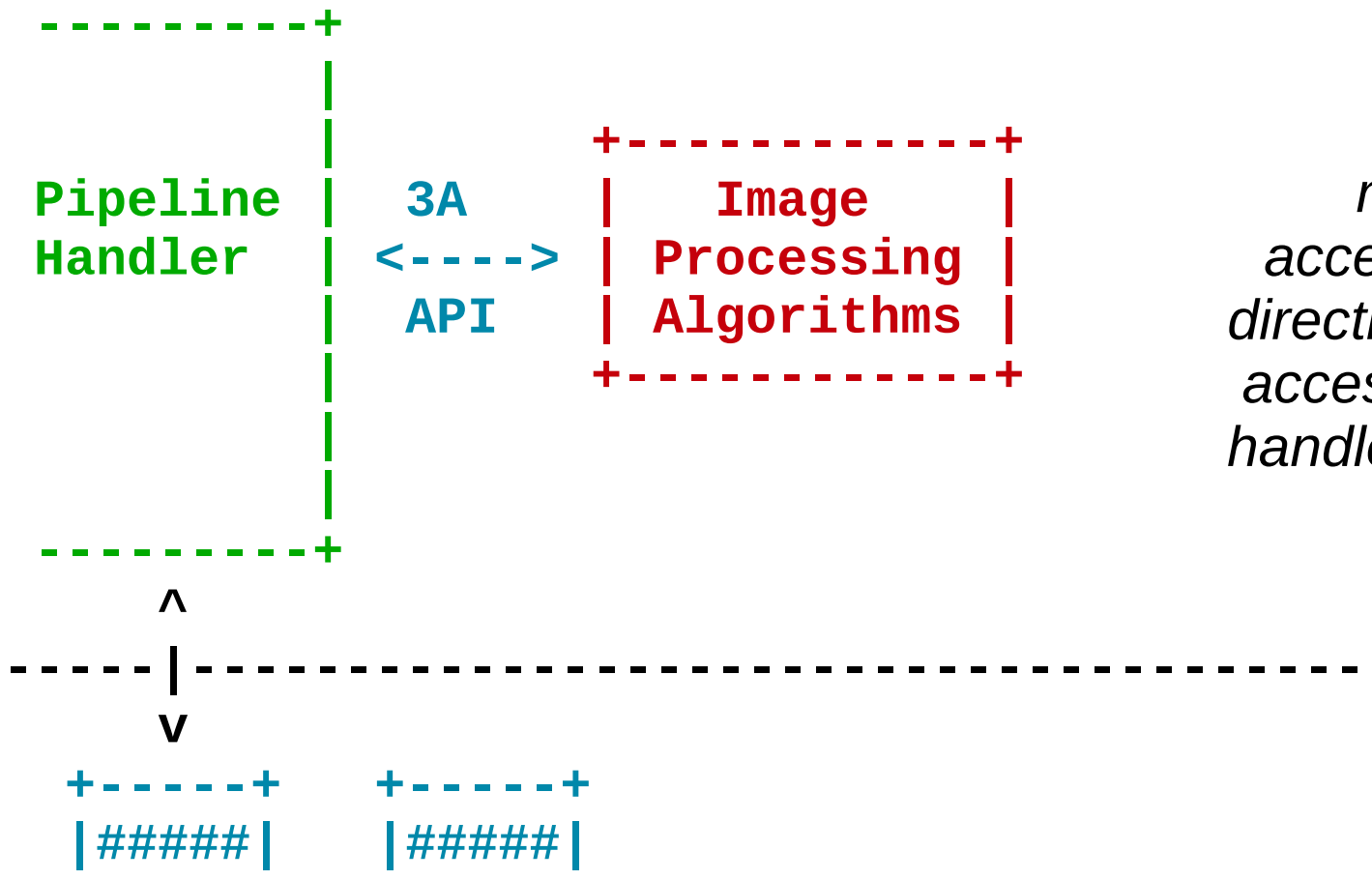


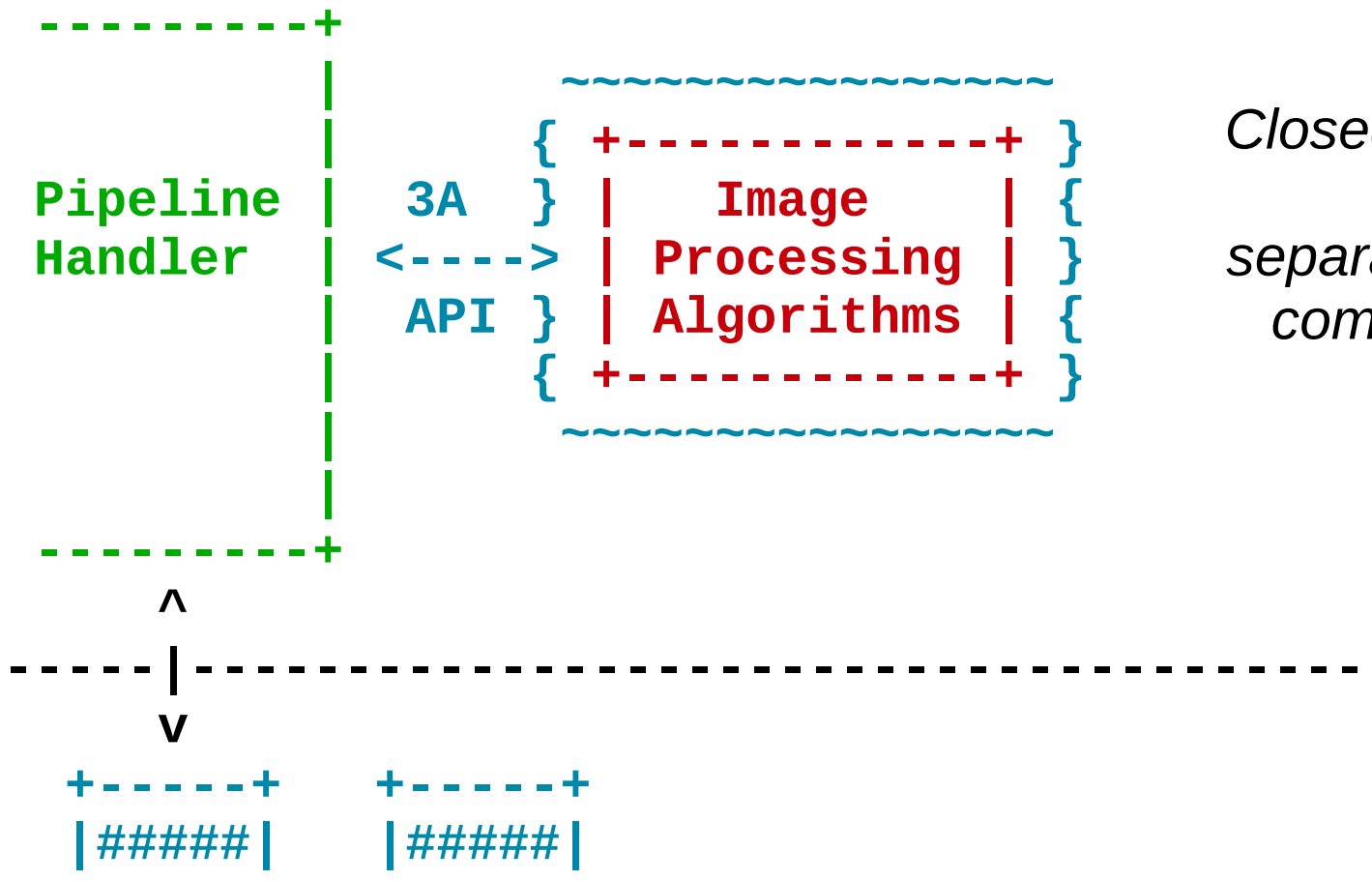
Image Processing Algorithms (IPA) receive statistics from the hardware and compute optimal image parameters.

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



Closed-source IPAs are sandboxed in a separate process. They communicate with the pipeline handler through IPC.

Image Processing Algorithms

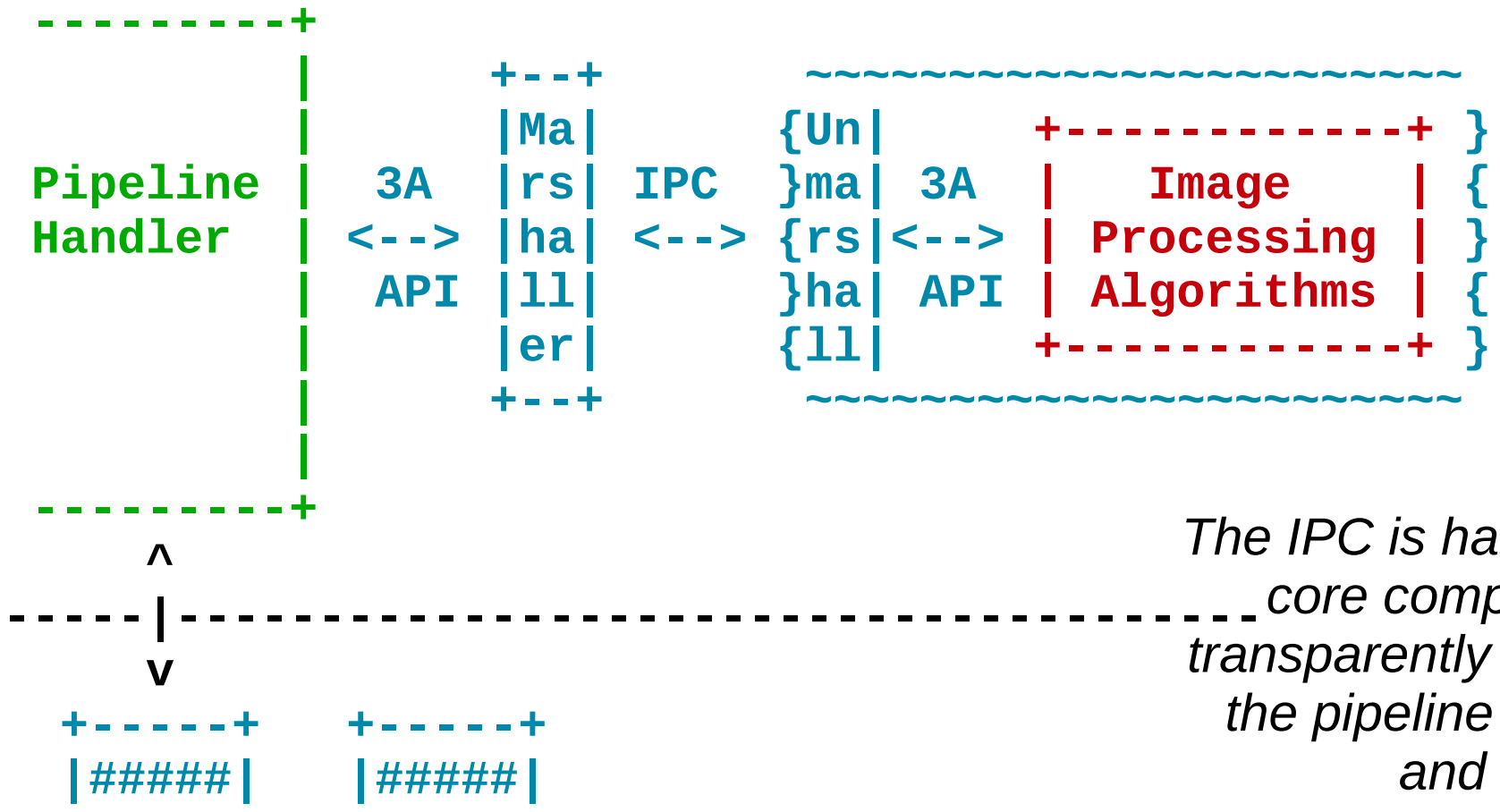
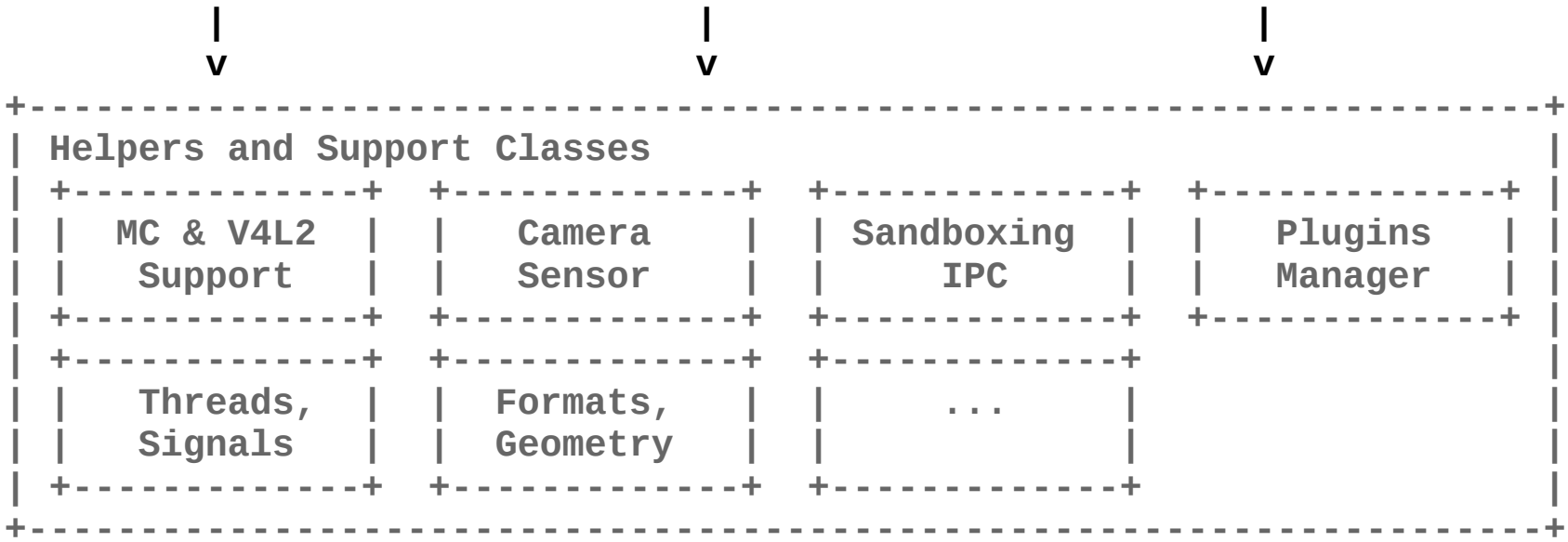
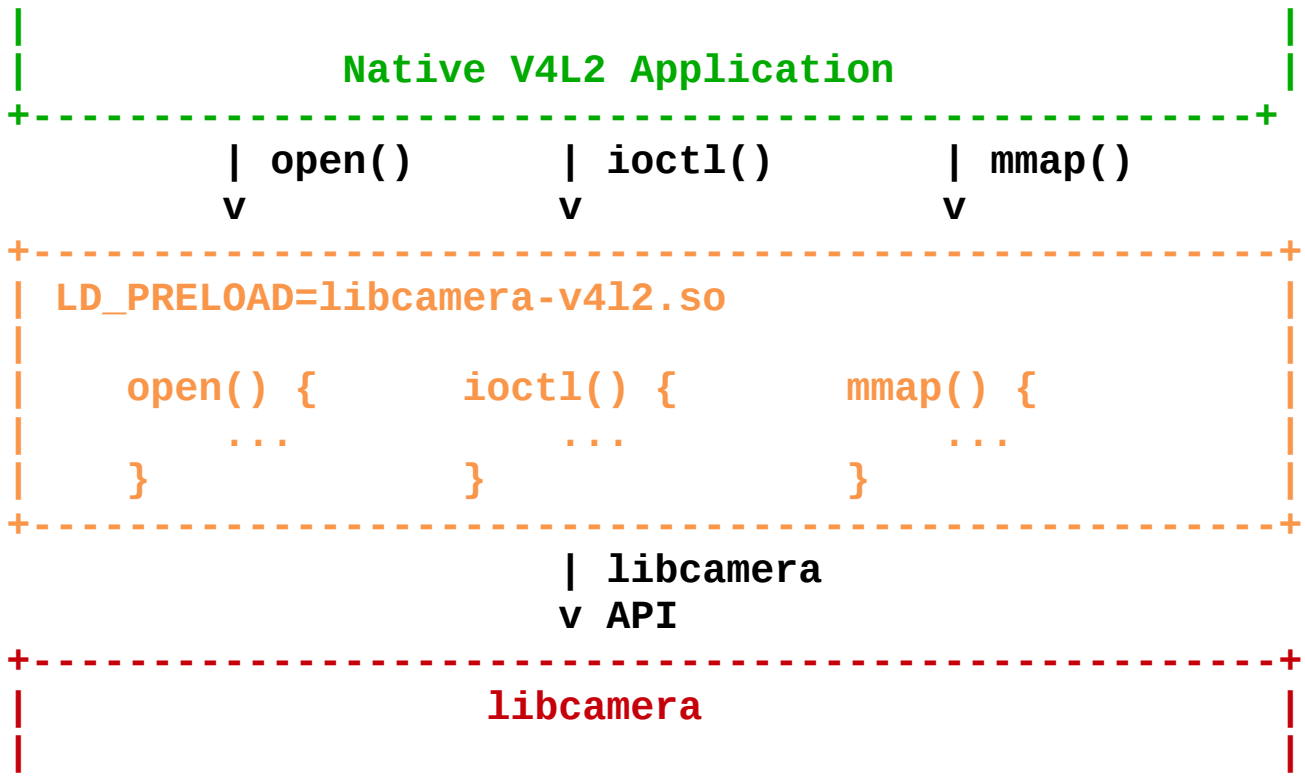


Image Processing Algorithms



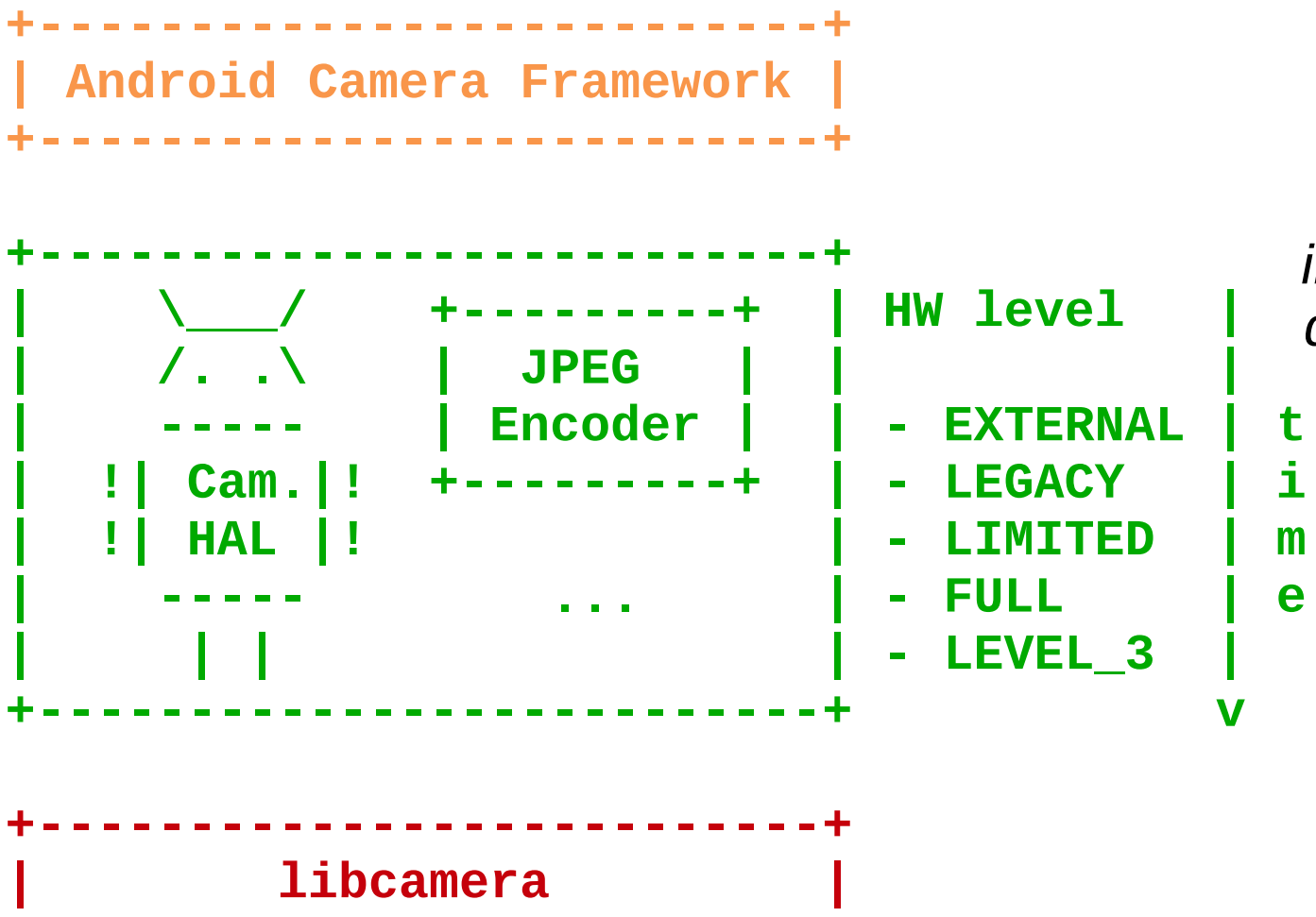
Many helper classes ease the implementation of pipeline handlers for device vendors.

Helpers and Support Classes



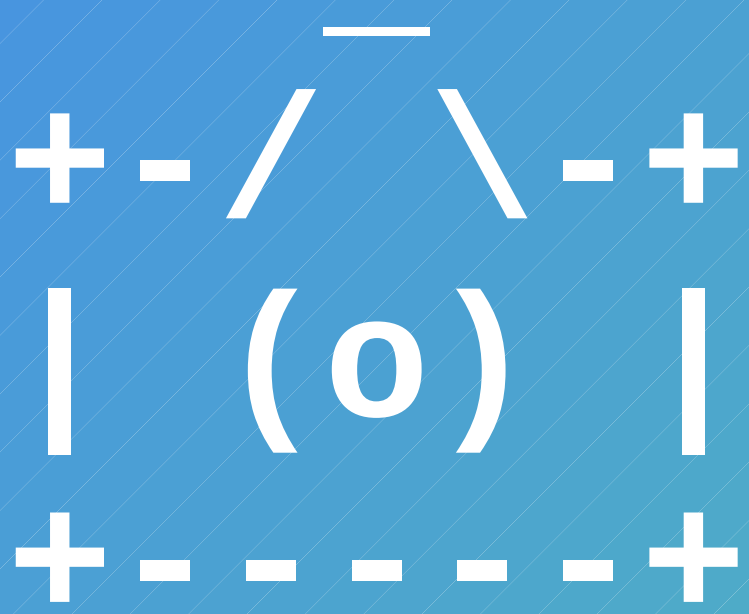
Native V4L2 applications are supported through a transparent compatibility layer.

V4L2 Compatibility

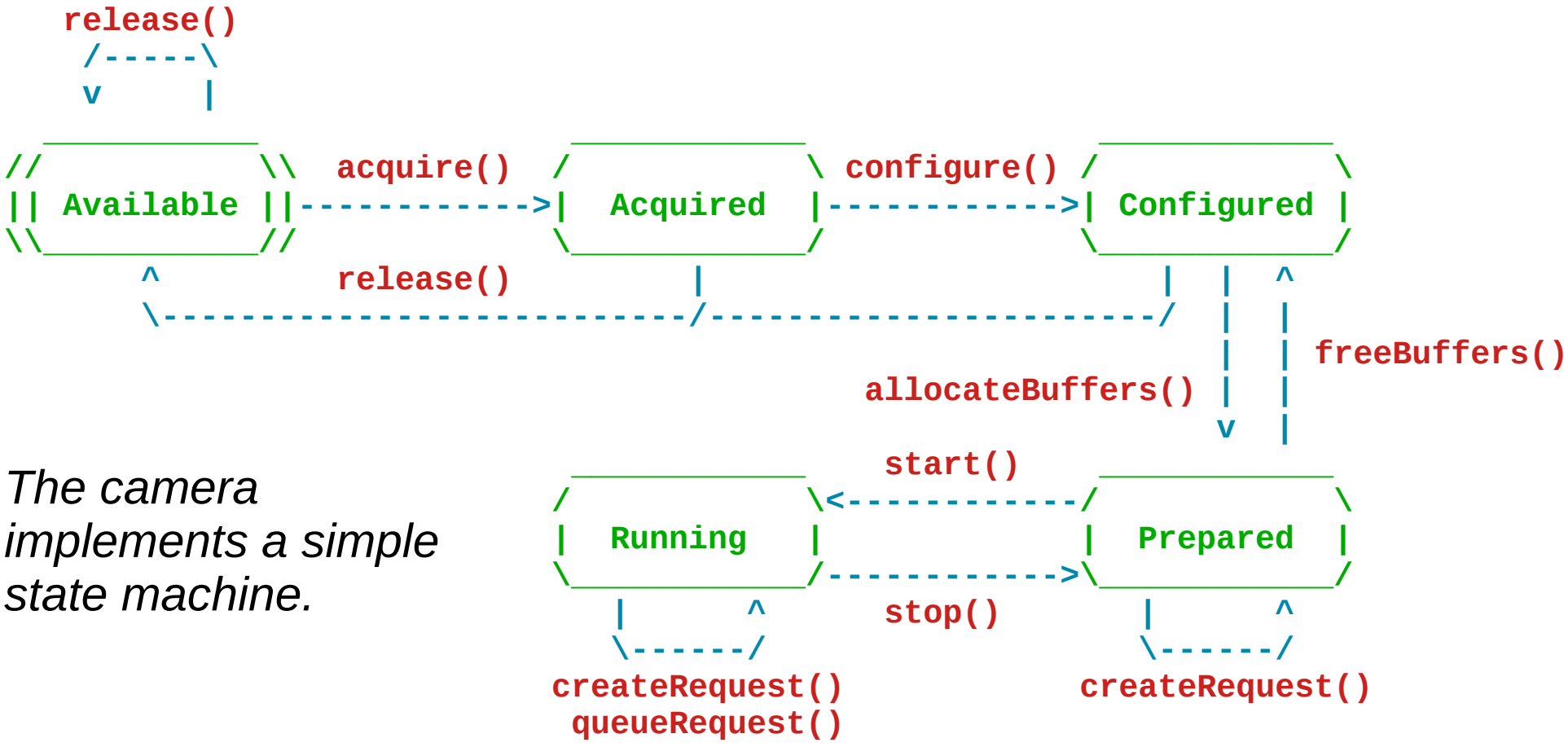


A single Android camera HAL module implementation for all devices supported by libcamera.

Android Camera HAL

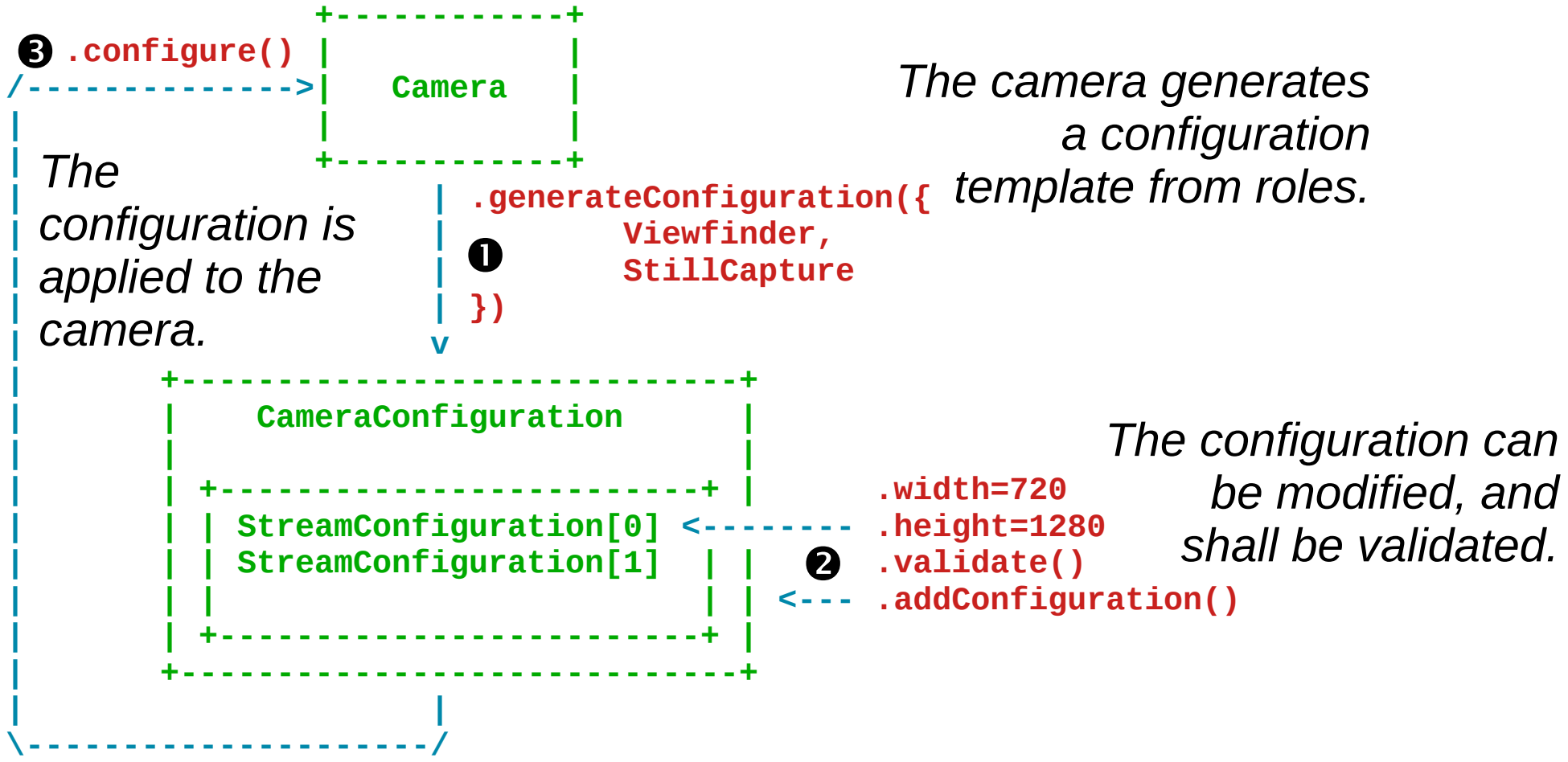


libcamera

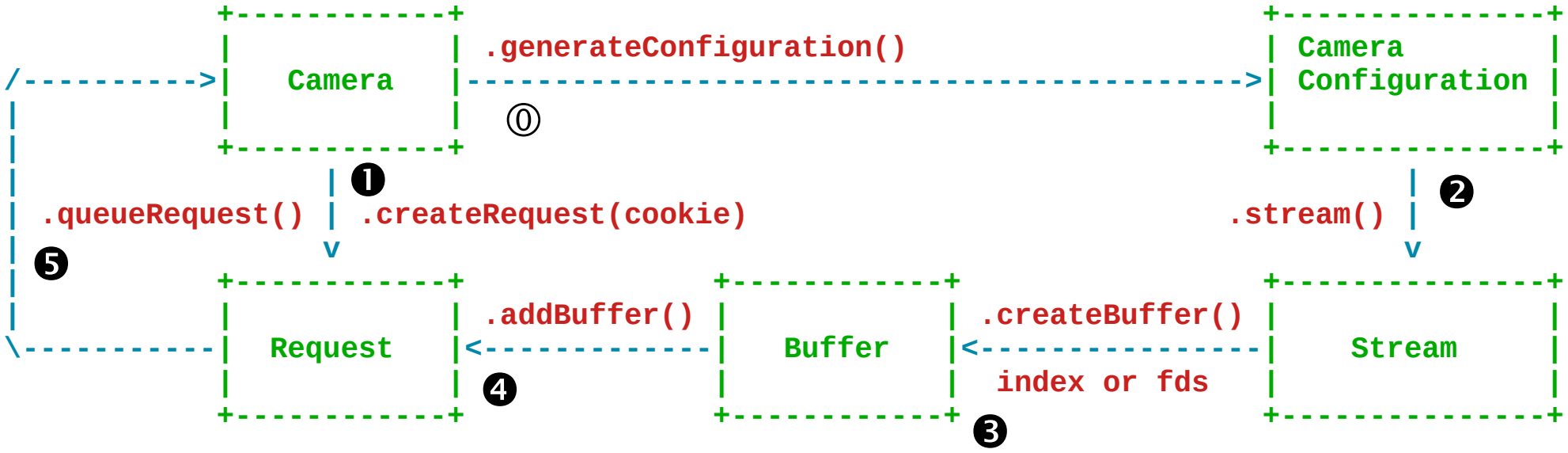


The camera implements a simple state machine.

Camera State Machine

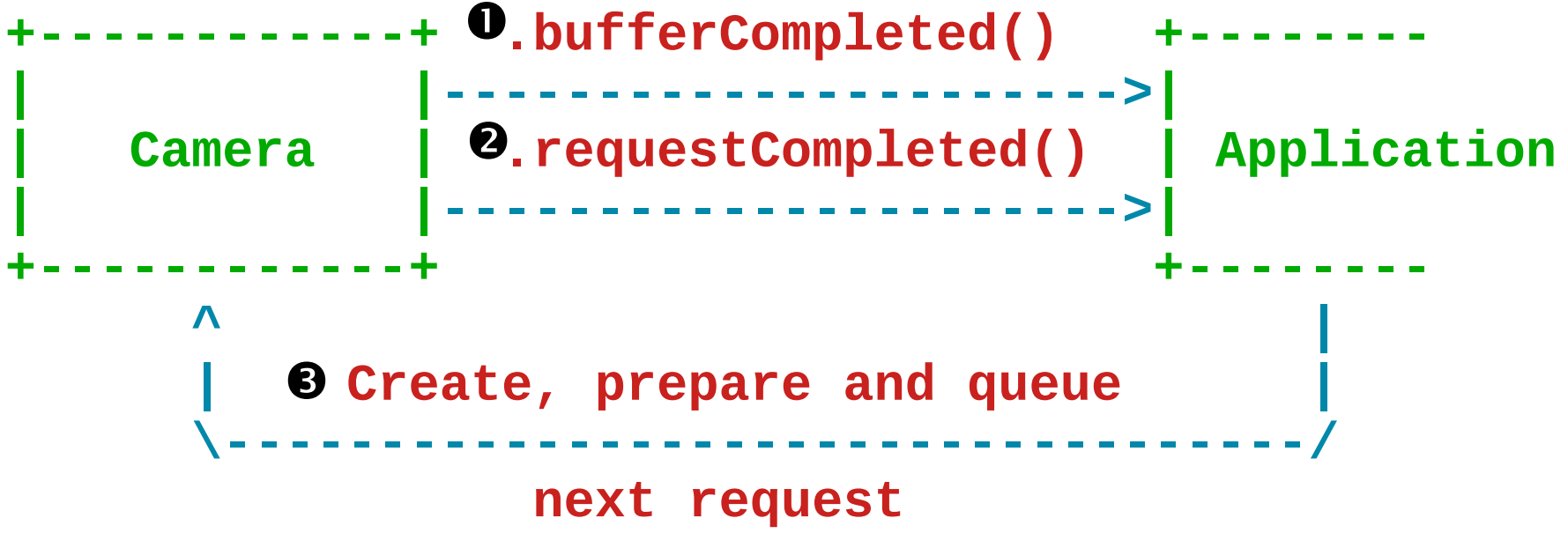


Camera Configuration



A request is created on the Camera, populated with a Buffer for each Stream, and queued for capture.

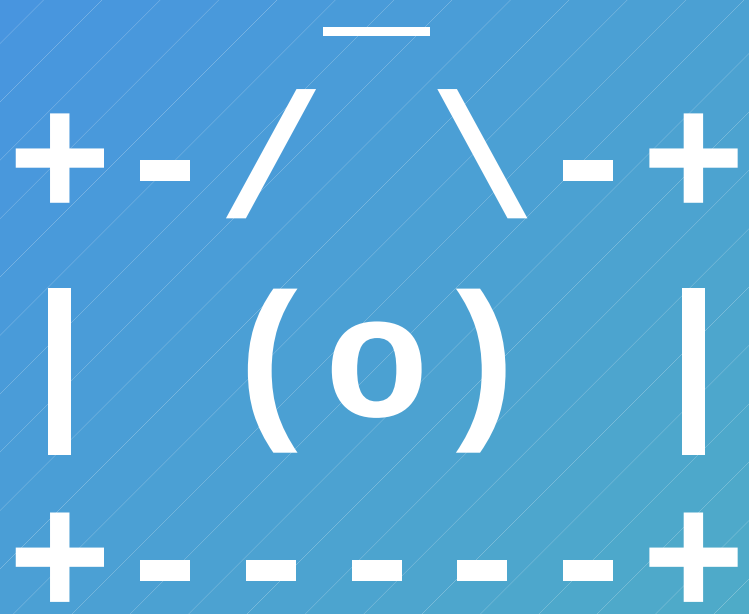
Request Queuing



Buffer and request completion are notified separately.

Applications submit new requests to keep the streams going.

Request Completion



libcamera

Contents

- Contributing
  - Mailing List
  - IRC Channel
  - Source Code
  - Documentation
  - Submitting Patches

Contributing

libcamera is developed as a free software project and welcomes contributors. Whether you would like to help with coding, documentation, testing, proposing new features, or just discussing the project with the community, you can join our official public communication channels, or simply check out the code.

Mailing List

We use a public mailing list as our main means of communication. You can find subscription information and the messages archive on the **libcamera-devel** list information page.

IRC Channel

For informal and real time discussions, our IRC channel on Freenode is open to the public. Point your IRC client to #libcamera to say hello, or use the **WebChat**.

Source Code

libcamera is in early stages of development, and no releases are available yet. The source code is available from the project's **git tree**, hosted by **LinuxTV**.

```
$ git clone git://linuxtv.org/libcamera.git
```

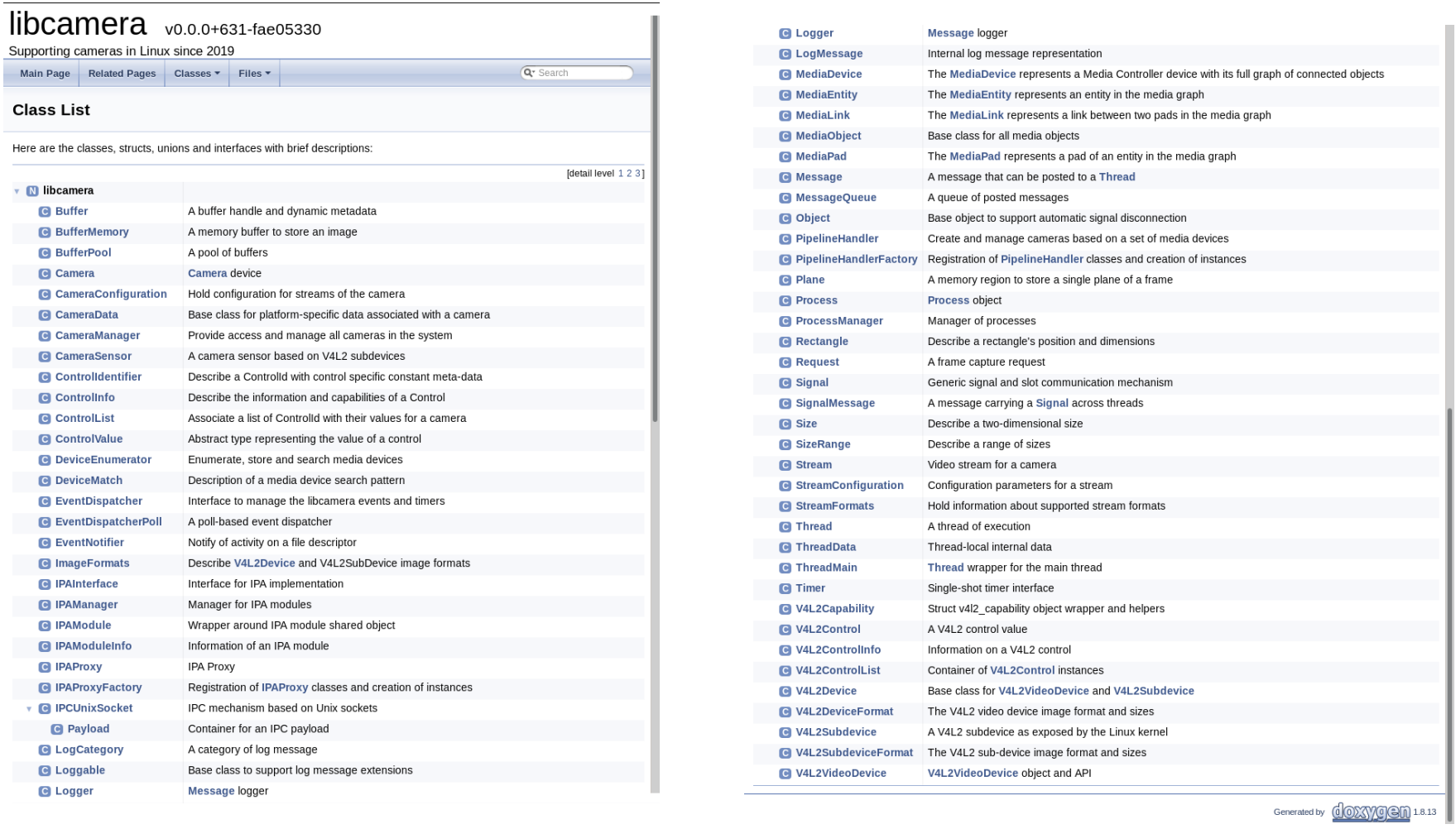
Documentation

Project documentation is created using **Sphinx**. Source level documentation uses **Doxygen**. Please make sure to document all code during development.

Sphinx integration with Doxygen is planned, likely using **Breathe** and **Exhale**.

Submitting Patches

Contribute



libcamera v0.0.0+631-fae05330

Supporting cameras in Linux since 2019

Main Page | Related Pages | Classes | Files | Search

Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

[detail level 1 2 3]

libcamera	
Buffer	A buffer handle and dynamic metadata
BufferMemory	A memory buffer to store an image
BufferPool	A pool of buffers
Camera	Camera device
CameraConfiguration	Hold configuration for streams of the camera
CameraData	Base class for platform-specific data associated with a camera
CameraManager	Provide access and manage all cameras in the system
CameraSensor	A camera sensor based on V4L2 subdevices
ControlIdentifier	Describe a ControlId with control specific constant meta-data
ControlInfo	Describe the information and capabilities of a Control
ControlList	Associate a list of ControlId with their values for a camera
ControlValue	Abstract type representing the value of a control
DeviceEnumerator	Enumerate, store and search media devices
DeviceMatch	Description of a media device search pattern
EventDispatcher	Interface to manage the libcamera events and timers
EventDispatcherPoll	A poll-based event dispatcher
EventNotifier	Notify of activity on a file descriptor
ImageFormats	Describe V4L2Device and V4L2SubDevice image formats
IPAInterface	Interface for IPA implementation
IPAManager	Manager for IPA modules
IPAModule	Wrapper around IPA module shared object
IPAModuleInfo	Information of an IPA module
IPAProxy	IPA Proxy
IPAProxyFactory	Registration of IPAProxy classes and creation of instances
IPCUnixSocket	IPC mechanism based on Unix sockets
Payload	Container for an IPC payload
LogCategory	A category of log message
Loggable	Base class to support log message extensions
Logger	Message logger
LogMessage	Internal log message representation
MediaDevice	The MediaDevice represents a Media Controller device with its full graph of connected objects
MediaEntity	The MediaEntity represents an entity in the media graph
MediaLink	The MediaLink represents a link between two pads in the media graph
MediaObject	Base class for all media objects
MediaPad	The MediaPad represents a pad of an entity in the media graph
Message	A message that can be posted to a Thread
MessageQueue	A queue of posted messages
Object	Base object to support automatic signal disconnection
PipelineHandler	Create and manage cameras based on a set of media devices
PipelineHandlerFactory	Registration of PipelineHandler classes and creation of instances
Plane	A memory region to store a single plane of a frame
Process	Process object
ProcessManager	Manager of processes
Rectangle	Describe a rectangle's position and dimensions
Request	A frame capture request
Signal	Generic signal and slot communication mechanism
SignalMessage	A message carrying a Signal across threads
Size	Describe a two-dimensional size
SizeRange	Describe a range of sizes
Stream	Video stream for a camera
StreamConfiguration	Configuration parameters for a stream
StreamFormats	Hold information about supported stream formats
Thread	A thread of execution
ThreadData	Thread-local internal data
ThreadMain	Thread wrapper for the main thread
Timer	Single-shot timer interface
V4L2Capability	Struct v4l2_capability object wrapper and helpers
V4L2Control	A V4L2 control value
V4L2ControlInfo	Information on a V4L2 control
V4L2ControlList	Container of V4L2Control instances
V4L2Device	Base class for V4L2VideoDevice and V4L2Subdevice
V4L2DeviceFormat	The V4L2 video device image format and sizes
V4L2Subdevice	A V4L2 subdevice as exposed by the Linux kernel
V4L2SubdeviceFormat	The V4L2 sub-device image format and sizes
V4L2VideoDevice	V4L2VideoDevice object and API

Generated by doxygen 1.8.13

API Documentation

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

laurent.pinchart@ideasonboard.com

Contact

ご清聴
ありがとう
ございました！

IDEAS ON BOARD