

When Embedded Codecs Meet Graphics

@ndufresne
Nicolas Dufresne
nicolas@collabora.com

Open First





COLLABORA

Who am I?

Open-source Linux developer

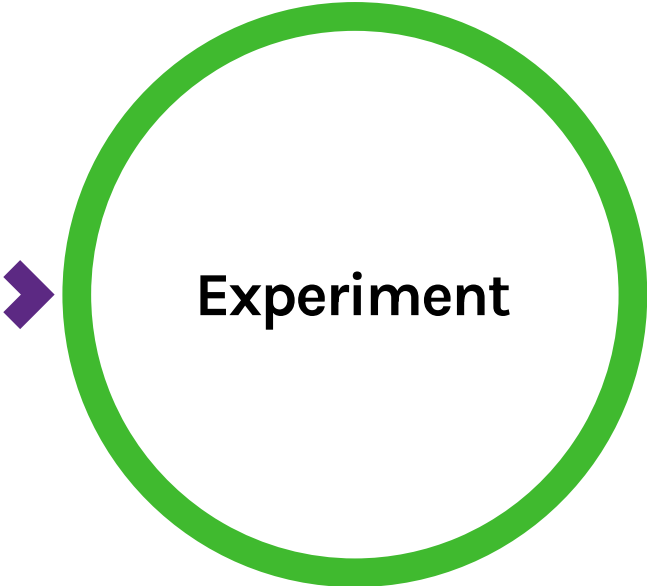
- Maintains **CODECs** drivers and their APIs in **Linux Media**
- All-around contributor to **GStreamer** (with a Linux focus)
- Helps with the GStreamer project **maintenance**
- Consultant at **Collabora** since **2009**





COLLABORA

What am I up to?



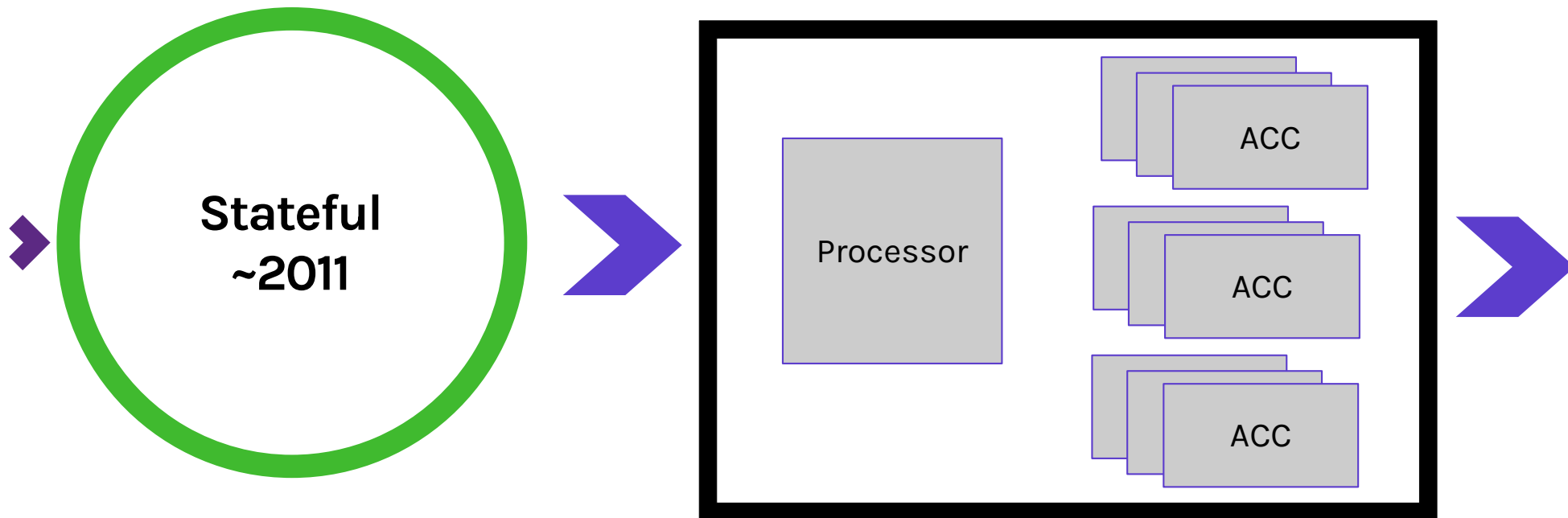
Experiment

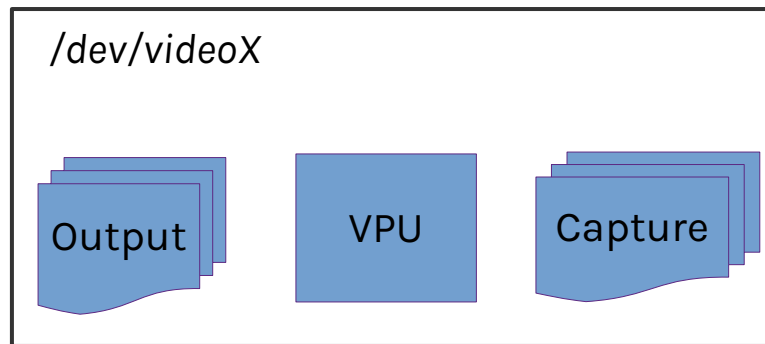
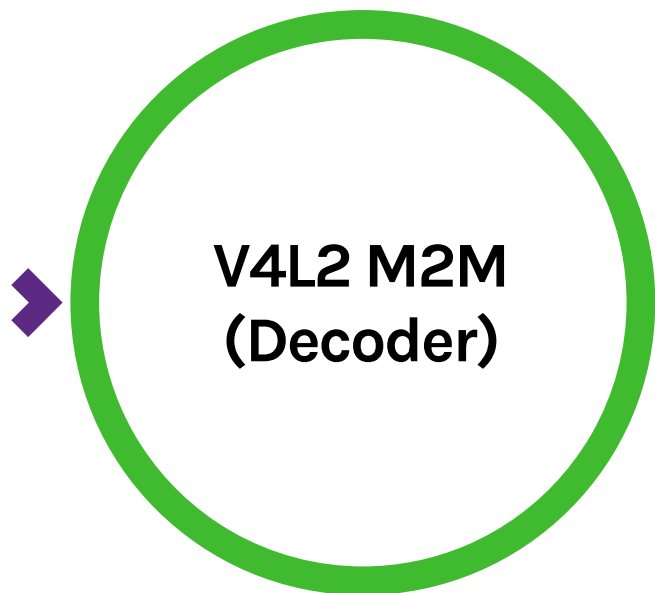
- **Vulkan Video** as a replacement API
- **Thin** hardware specific drivers with uAPI that covers the entire IP
- User space driver in **Mesa**



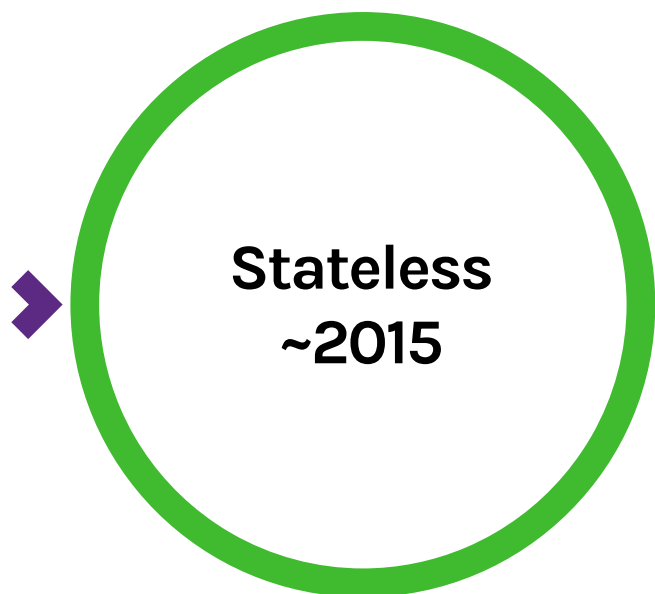
COLLABORA

A little bit of history





- A V4L2 output queue is used for the bitstream
- A V4L2 capture queue is used for the decoded pictures
- Additional control flows are added to support draining and flushing
- Inter-queue configuration flow is needed



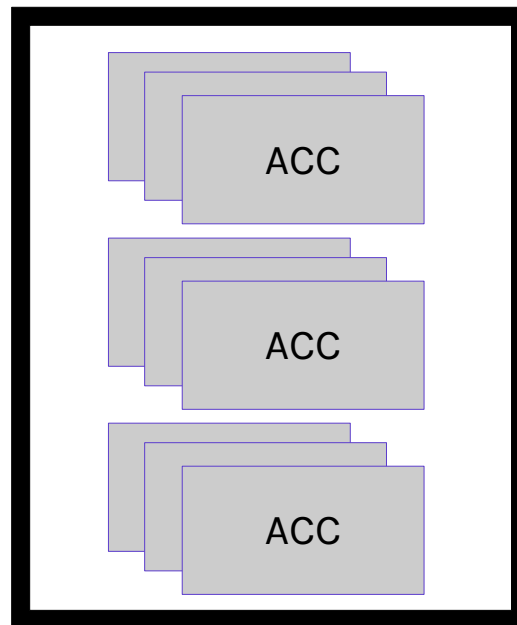
Reference(s)



Bitstream



Parameters



Picture

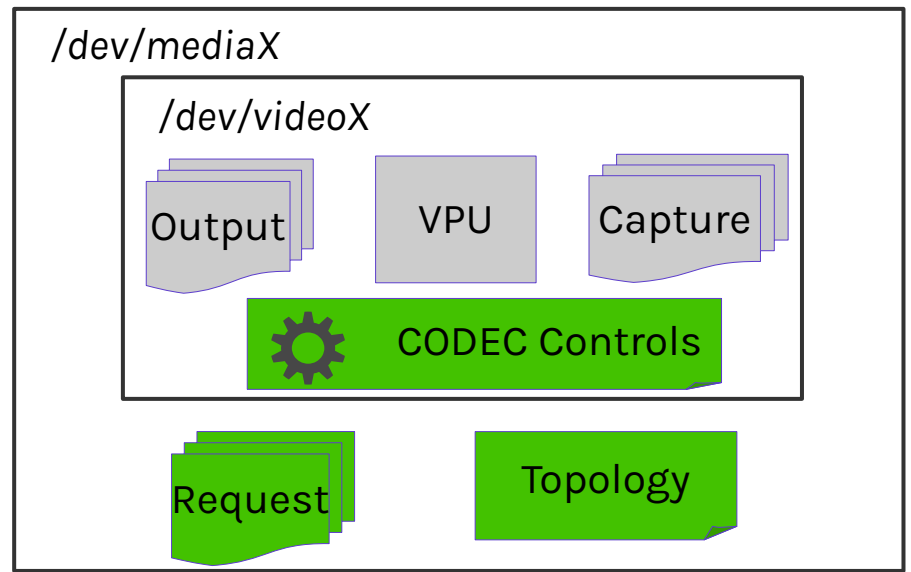


COLLABORA

Open First

➤

**V4L2 M2M
+
Request**

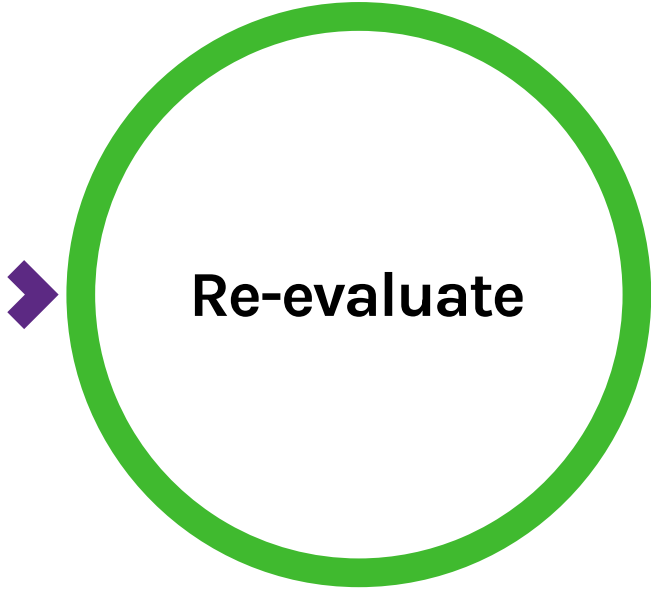


- Per-frame (or slice) controls are associated with bitstream buffers using requests
- References are signaled using user-defined timestamps
- Better identification can be done using the Media Controller topology



V4L2 is the standard

- Takes **years** to develop
- Getting **feedback** is difficult
- V4L2 has **limitations** that have not yet been resolved, despite my years of working on it
- Especially challenging for embedded stateless **encoders**



Re-evaluate

- **Paul Kocialkowski** developed an **H.264** driver for **Hantro VC8000E** (I.MX8M+)
- My **team's energy** is no longer there
- **Customers** are not ready to sponsors
- **Every** new codec requires **new user API**
- We believe the kernel should manage the **resources** and **schedule** work



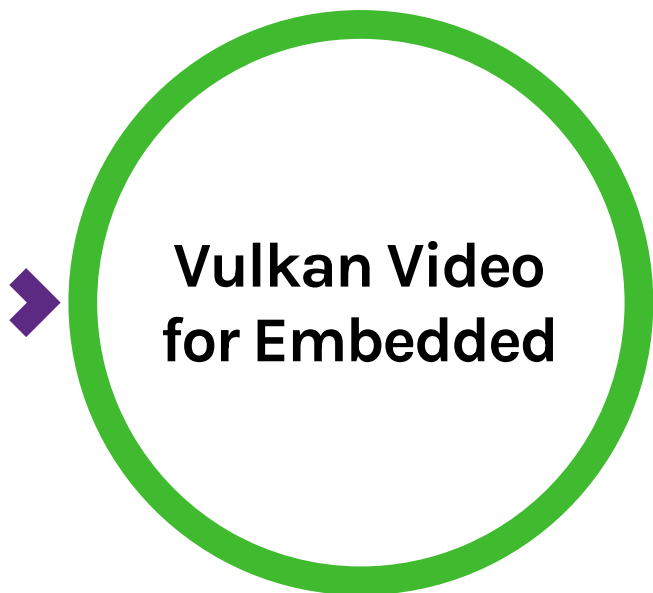


Solving these issues



- **Actively developed** by major industry players (AMD, Intel, NVIDIA, Amazon, etc.)
- Supports the most important codecs
- More **advanced features** are supported, making it more **future-proof**
- **You can join** and contribute

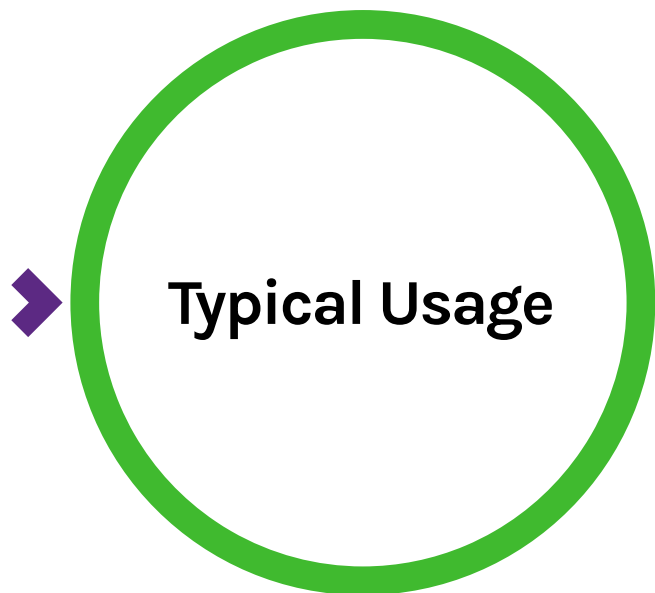




- First introduced in **March 2026**
- Supported in **FFMpeg** and **GStreamer**
- **Firefox** decode adoption
- **Chromium** encode adoption
- And more to come, including why I'm here



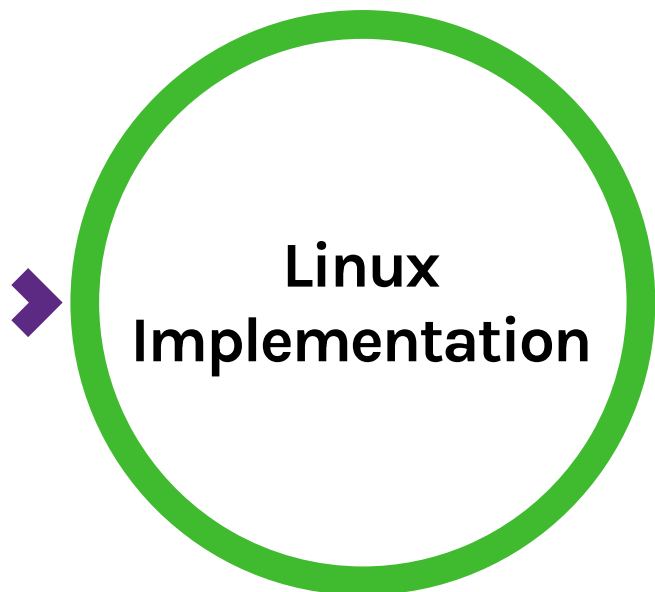
How it works for user?



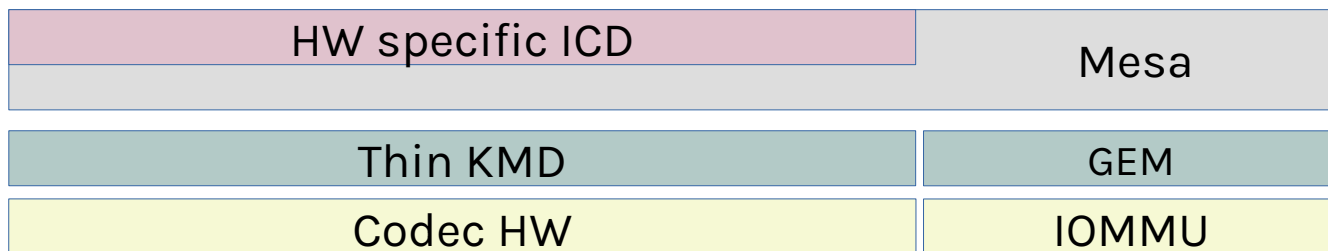
- **Enumerate** Devices
- Query Encode/Decoder **capabilities**
- Create and configure a **Video Session**
- Submit **commands** (begin/end style)
- You can create **synchronization objects**
- You can submit work that **depends** on synchronization objects
- Repeat

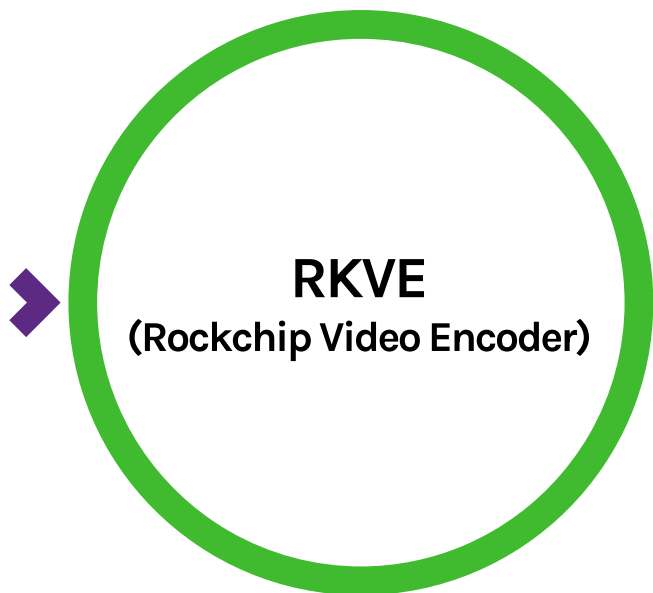


It is a good fit for DRM-style drivers

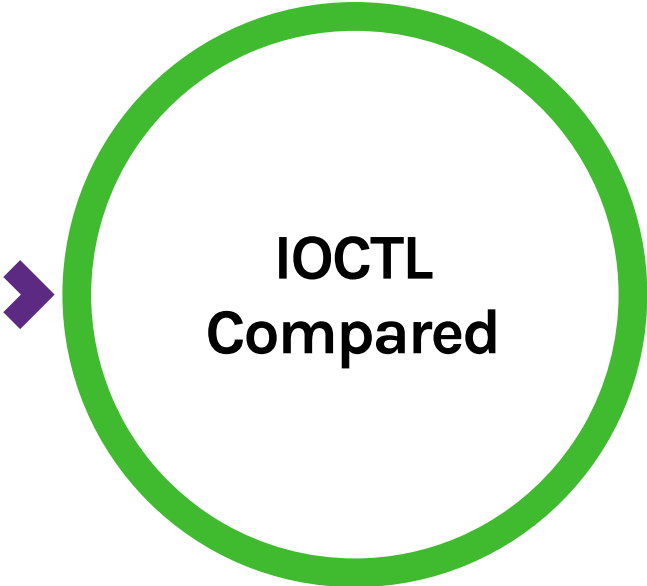


- ICD – Installable Client Driver
- KMD – Kernel Mode Driver





- At the **prototype** level
 - Encodes H.264 **I** and **P** frames on **RK3588**
 - The pre-code block is unconfigured
 - Rate control is not functional
-
- Early Kernel Code 
 - Early Mesa Code
 - WIP GStreamer Changes



IOCTL Compared

- DRM_IOCTL_RKVE_CTX_CREATE
- DRM_IOCTL_RKVE_CTX_DESTROY
- DRM_IOCTL_RKVE_SUBMIT
- DRM_IOCTL_RKVE_BO_CREATE
- DRM_IOCTL_RKVE_BO_MMAP_OFFSET

- VIDIOC_QUERYCAP
- VIDIOC_ENUM_FMT
- VIDIOC_ENUM_FRAMESIZES
- VIDIOC_G_FMT
- VIDIOC_S_FMT
- VIDIOC_REQBUFS
- VIDIOC_CREATE_BUFS
- VIDIOC_REMOVE_BUFS
- VIDIOC_QUERYBUF
- VIDIOC_QBUF
- VIDIOC_DQBUF
- VIDIOC_EXPBUF
- VIDIOC_STREAMON
- VIDIOC_STREAMOFF
- VIDIOC_QUERY_EXT_CTRL
- VIDIOC_G_EXT_CTRLS
- VIDIOC_S_EXT_CTRLS



- `#define RKVE_CMD_ENCODE_H264 1`
- `#define RKVE_CMD_SET_H264_ENC_STATE 2`
- `#define RKVE_CMD_SET_H264_RC 3`
- ```
struct rkve_cmd_encode_h264 {
 __u32 type;
 __u32 size;
 __u32 ctx;
 __u8 flags;
 __u8 qp;
 ...
```

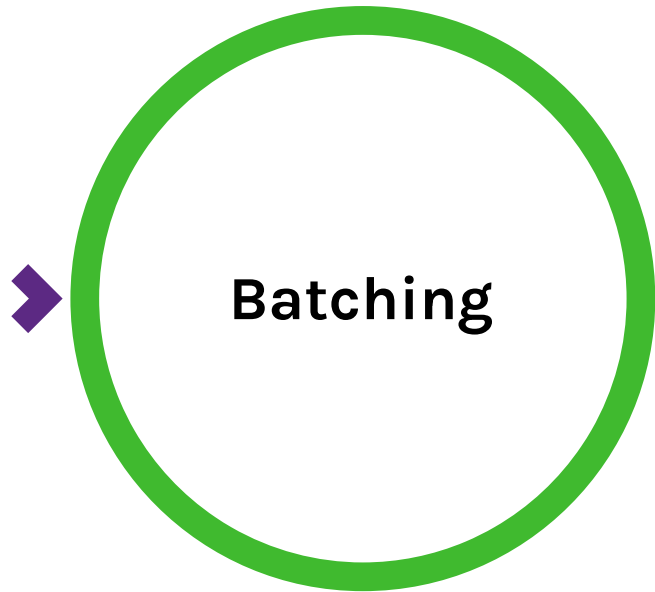


- Write **register** in their **hardware format** (create **partitions**)
- **Read** feedback registers in hardware format
- Prefer **DMABuf** file descriptors over exposing page tables (drivers can decide what is best)
- Define **rules** so this is not blob interface

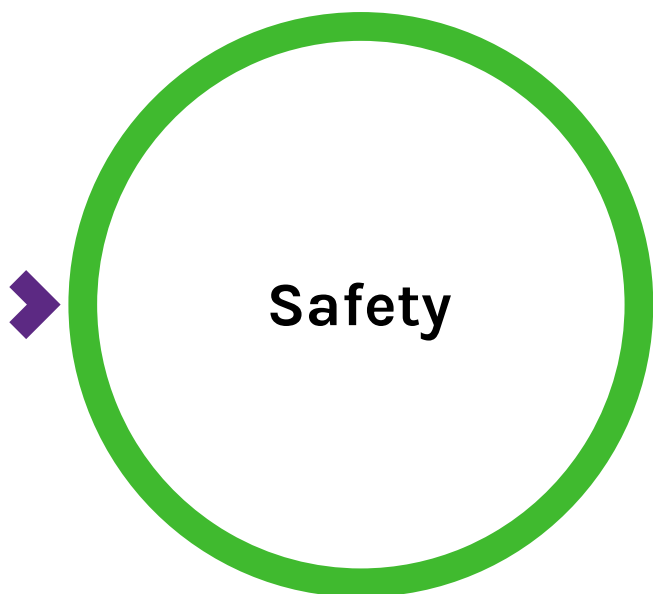


COLLABORA

# The real discussions begins



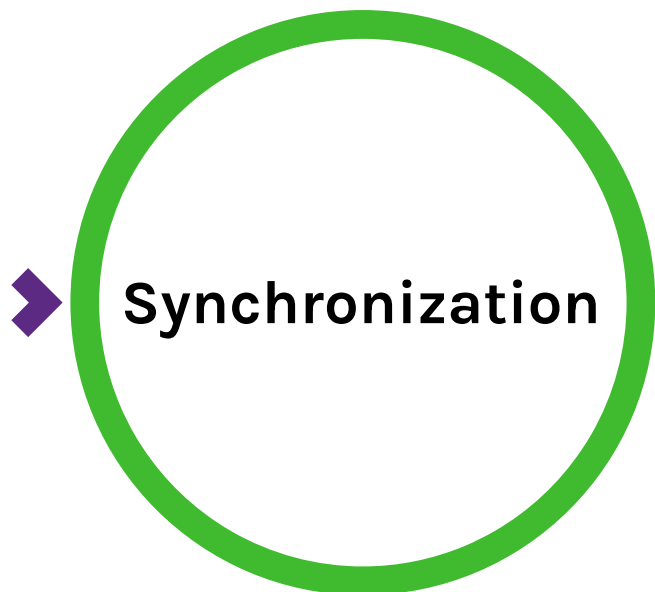
- Make use of Rockchip **Linked-Table**
- **Multiple** encode **registers** set into an array
- The hardware **program itself** from this array and fires an IRQ between each frame
- **Results** are **resources**, and every commands have its own results buffer



- The **IOMMU** provides most of the safety  
(Hardware without IOMMU must use a different approach)
- **Per-process** address space\*\*
- IOMMU and encode **errors are handled** and recovered by the driver

\*\*

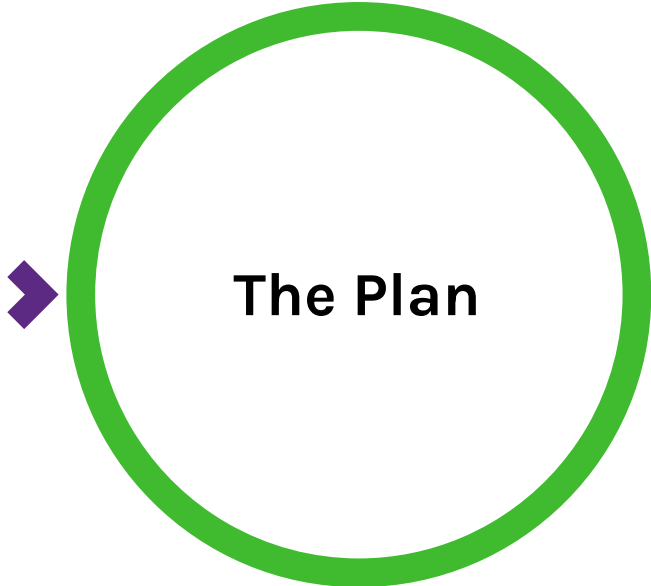
Rockchip Link-Table does not support IOMMU directly.,  
Perhaps we can change domain upon IRQ or maybe we  
need a safety trade-off while scheduling



- A command **can depend** on **fences**
- A command **can export** fences
- Application can submit **ahead of time**
- Fences **errors are handled** and propagated
- Mostly generic and handled by the **scheduler**  
(we are currently sharing a new scheduler with Tyr)



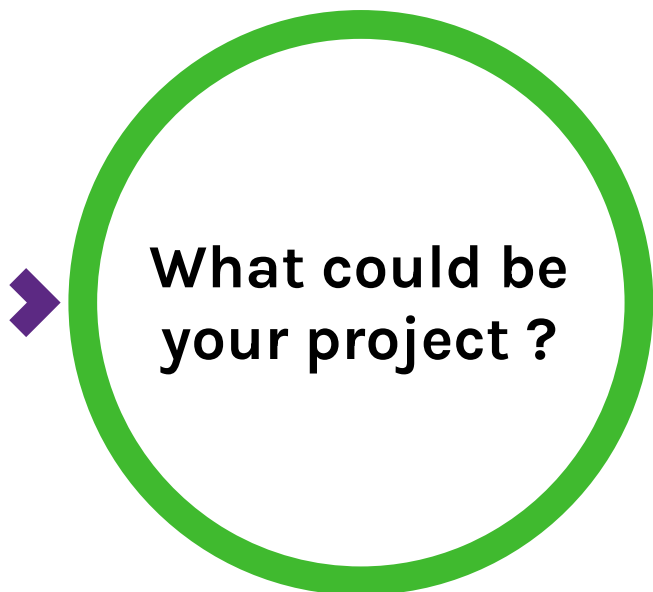
# Acceptability and community engagement wanted!



## The Plan

- On-board **Vulkan Video TSG** (Done)
- Introduce this to **Linux Media** devs (Done)
- Talk about it at **Plumbers** (I'm on it)
- Get feedback from **Mesa** developers
- Find a **home** (DRM, Media, something new?)
- **Vulkan CTS** for **Vulkan Video-only** drivers
- **Extend** Vulkan Video for embedded **decoders**
- Make great drivers !





- More **Rockchip** Generation
- Newer VeriSilicon **Hantro**-based designs  
(ideally with IOMMU, possibly with VCMD)
- **Apple** Video codecs
- Newer **Allwinner** VDEs
- **MediaTek** (needs specific firmware)
- **Qualcomm** (using Windows DXVA firmware?)
- **Raspberry Pi** (Argon) codecs
- **Amlogic** Video Codecs
- Any hardware with a **firmware** can be converted!



Question?



COLLABORA

Open First



Thank you!



COLLABORA

Open First