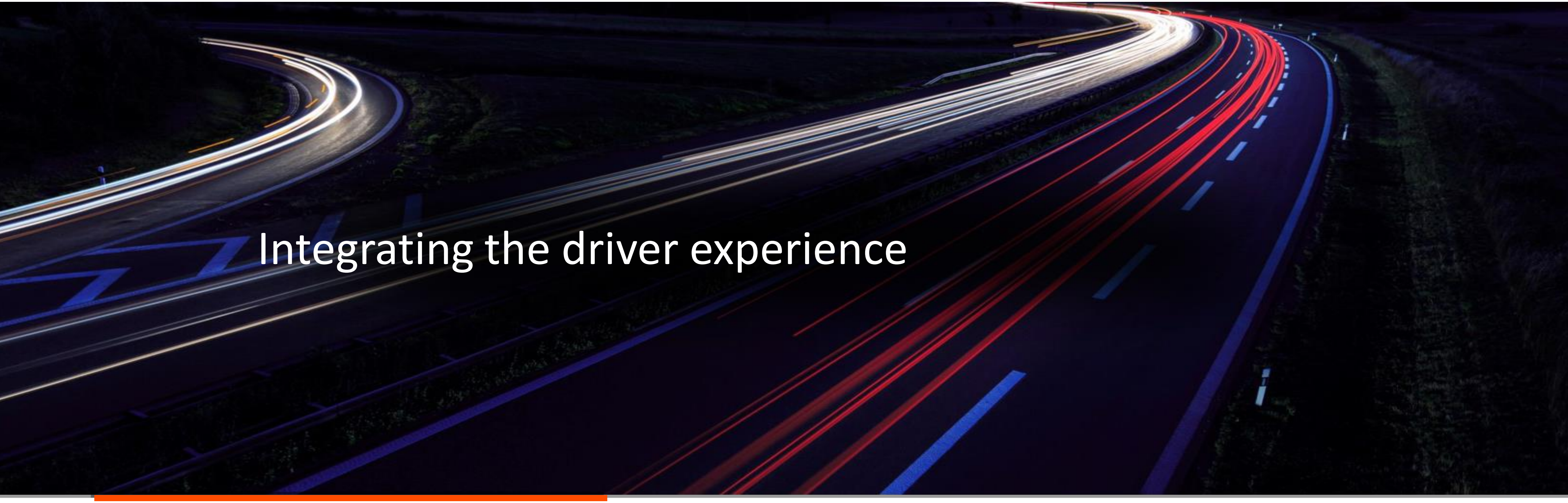


Automotive Virtual Platform Using VIRTIO

Mikhail Golubev @ Automotive Linux Summit 2019

public

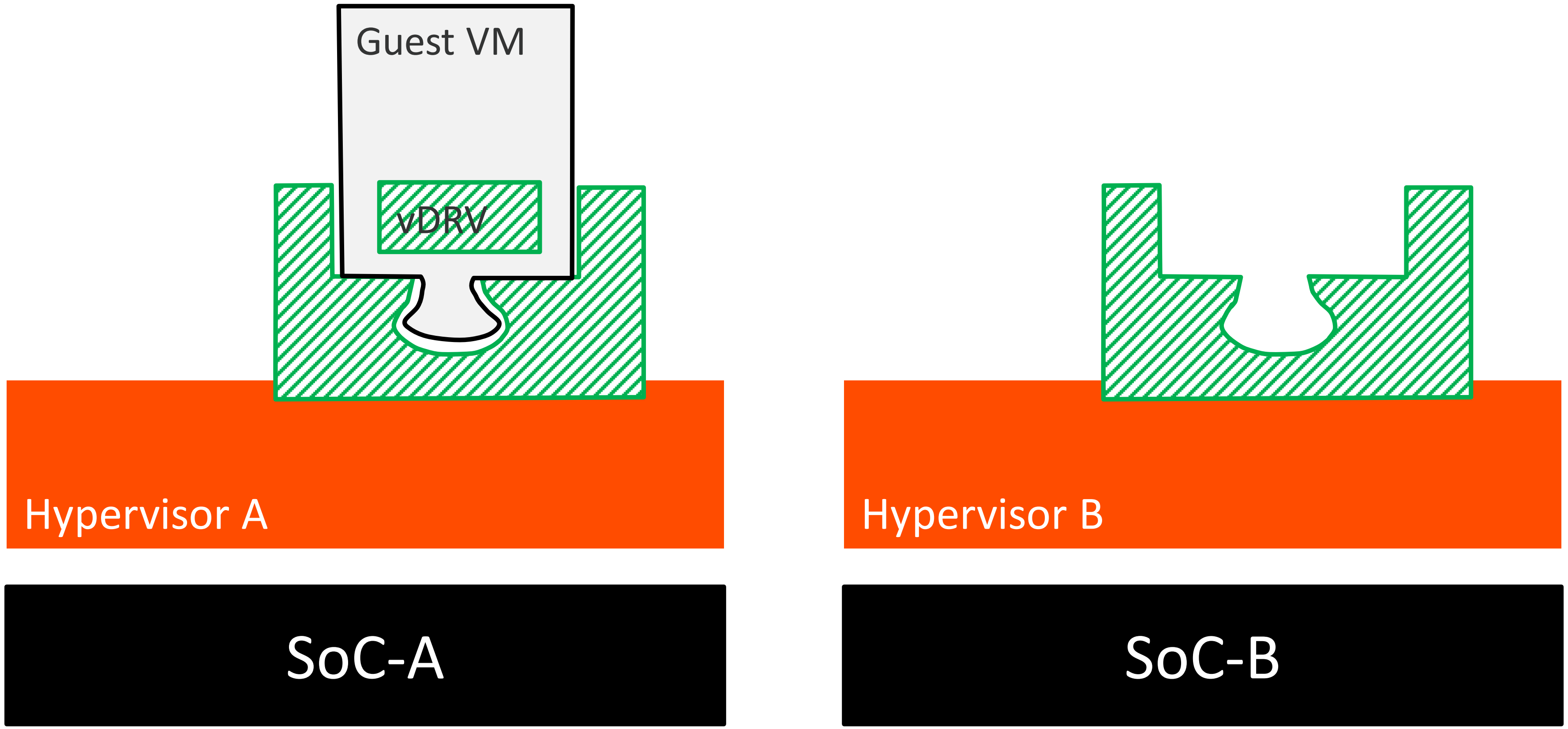
Agenda

- “Paving the Path to Standardization of Virtualization” Dr. Ralph Sasse @ ALS Tokyo 2018
- VIRTIO in a nutshell
- virtio-video
- virtio-snd
- Demo

VISION: Run Guest VMs without modifications

OPENSYNERGY

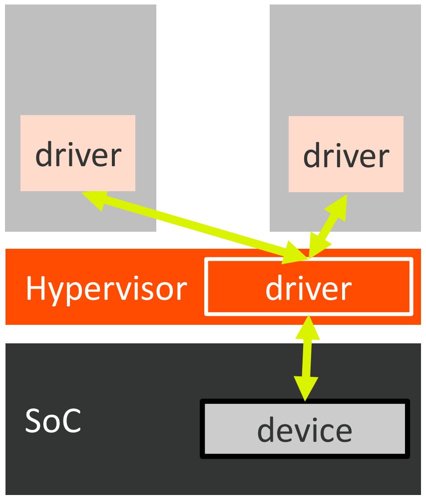
- Guest Virtual Machines that could be moved among different hypervisor systems and/or HW platforms **without further modification** through establishing an industry standard / de-facto standard



Mechanisms for device sharing (in COQOS)

OPENSYNERGY

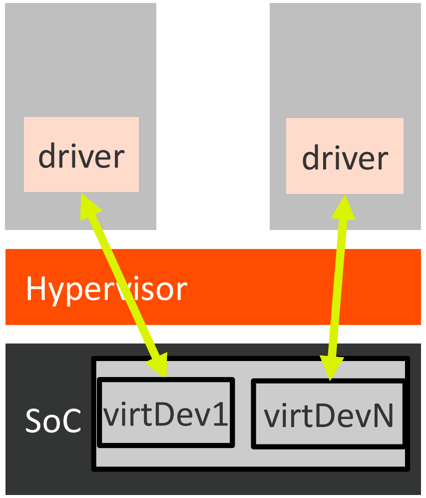
in Hypervisor



- Only used for UART (optionally)
- Not recommended for other devices as the Hypervisor is minimalistic

Example: UART

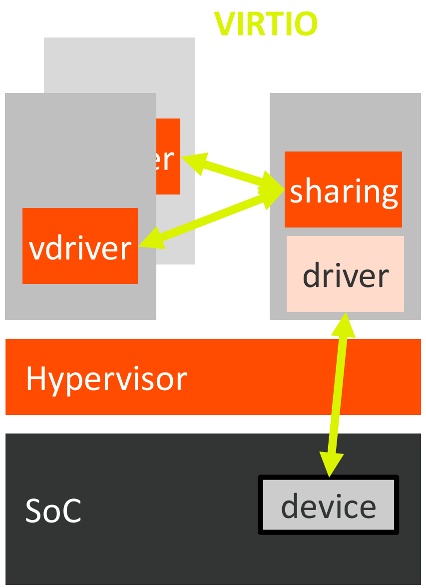
device with virtualization support



- COQOS supports this when the SoC hardware supports virtualized devices
- Recommended wherever the hardware supports it, as it tends to give the best performance and separation

Example: GPU on RCAR-H3

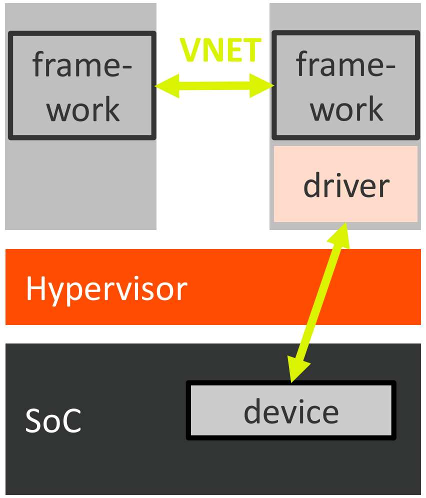
virtual driver



- Single driver in VM that acts as "server"
- Driver-specific sharing logic is needed
- Other VMs use "virtual driver"
- Compromise between performance and flexibility
- Can use VIRTIO

Example: shared block device

distributed frameworks over virtual network



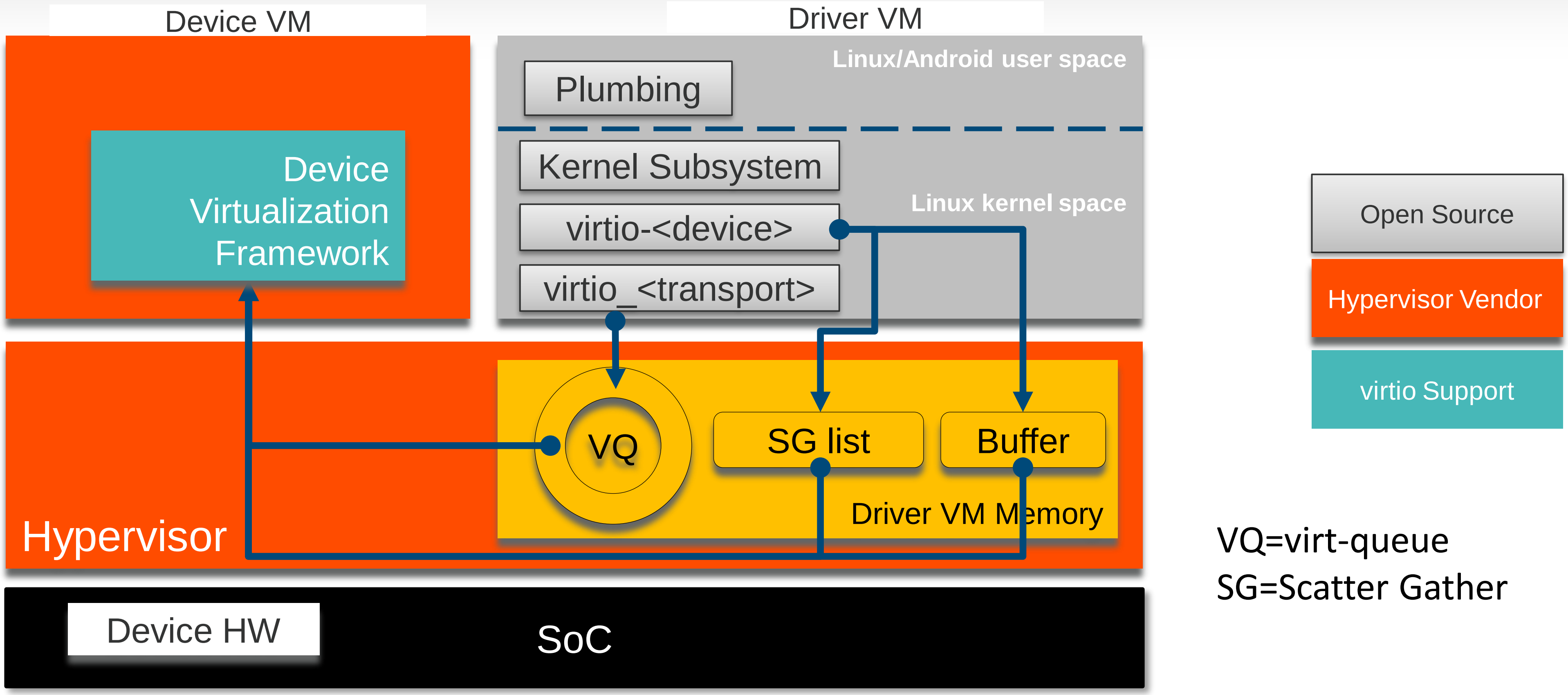
- Allows reuse of existing frameworks for distributed applications over virtual network
- Supports complex sharing semantics at the cost of more overhead

Example: NFS, PulseAudio

Introduction to VIRTIO

- “virtio: Towards a de-Facto Standard For Virtual I/O Devices” [Rusty Russell 2008]
- Formally standardized since March 2016 (OASIS VIRTIO-v1.0)
- VIRTIO provides the transport layer and device models for many devices (OASIS VIRTIO-v1.1 approved 2019)
  - **Block Storage, SCSI**
  - **Network**
  - **Console**
  - **Entropy (rng)**
  - **Memory balloon**
  - **Crypto**
  - **GPU 2D**
  - **Input (hid)**
  - **Socket (vsock)**
  - **Many more in development (vIOMMU, etc.)**
- For the Automotive domain there is work in progress
  - **Audio**
  - **Sensors**
  - **Media Acceleration (VPU, IPU, CODEC)**
  - **...**

VIRTIO in a nutshell



VQ=virt-queue
SG=Scatter Gather

Bulk data transport via DMA-like memory model

- Buffer **allocations** handled by „Driver" part (client)
- **Direct** R/W access to allocated buffers in the „Device" part (server)

Metadata transport via virt-queues (ring buffers, asynchronous pipeline)

virtio-video

Virtual video streaming device concept

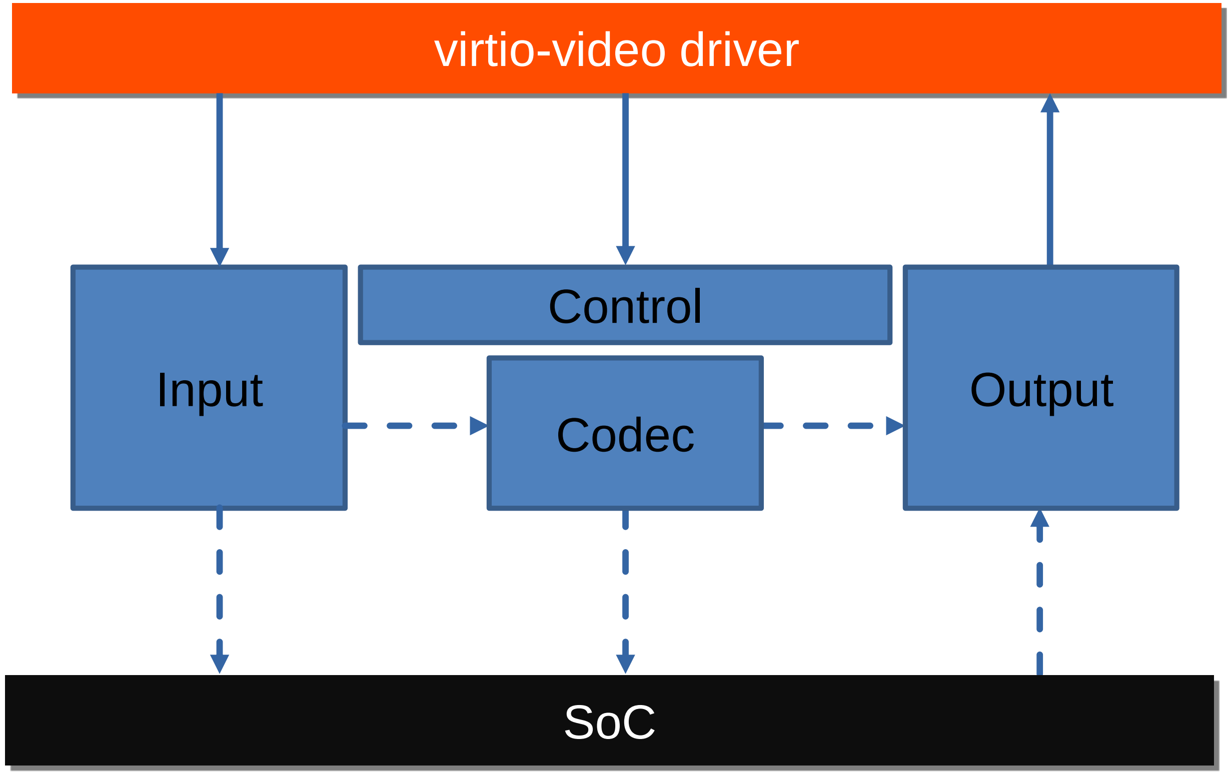
- Paravirtualised guests require **video streaming devices**, including video cameras, streaming capture and output devices, codec devices
- **Hardware video acceleration** offloads the CPU, increases performance, and saves power
- An abstract video streaming device that operates input and/or output data buffers is used to **share video devices** with several guests
- Buffers are essentially scatter-gather lists used for **DMA** operations (similar to virtio-gpu)
- The buffers are used to organize **data streams**, e.g. from a camera (output stream) or from a decoder (input stream with data and output stream with decoded frames)

Virtual video required functional

OPENSYNERGY

The virtio-video device performs operations on **video streams**

- Decoding
- Encoding
- Input/output
- Control



APIs to access video devices

Several major APIs exist at the moment:

- **OpenMAX**
  - A royalty-free, cross-platform API that provides comprehensive streaming media codec and application portability
- **VA-API**
  - Provides access to graphics hardware acceleration capabilities for video processing. It consists of a main library and driver-specific acceleration backends for each supported hardware vendor
- **V4L2**
  - API that has been designed to control media devices in Linux. Supports the DMABUF framework, which provides a generic method for sharing buffers between multiple devices

Codec device types

Two codec device types exist:

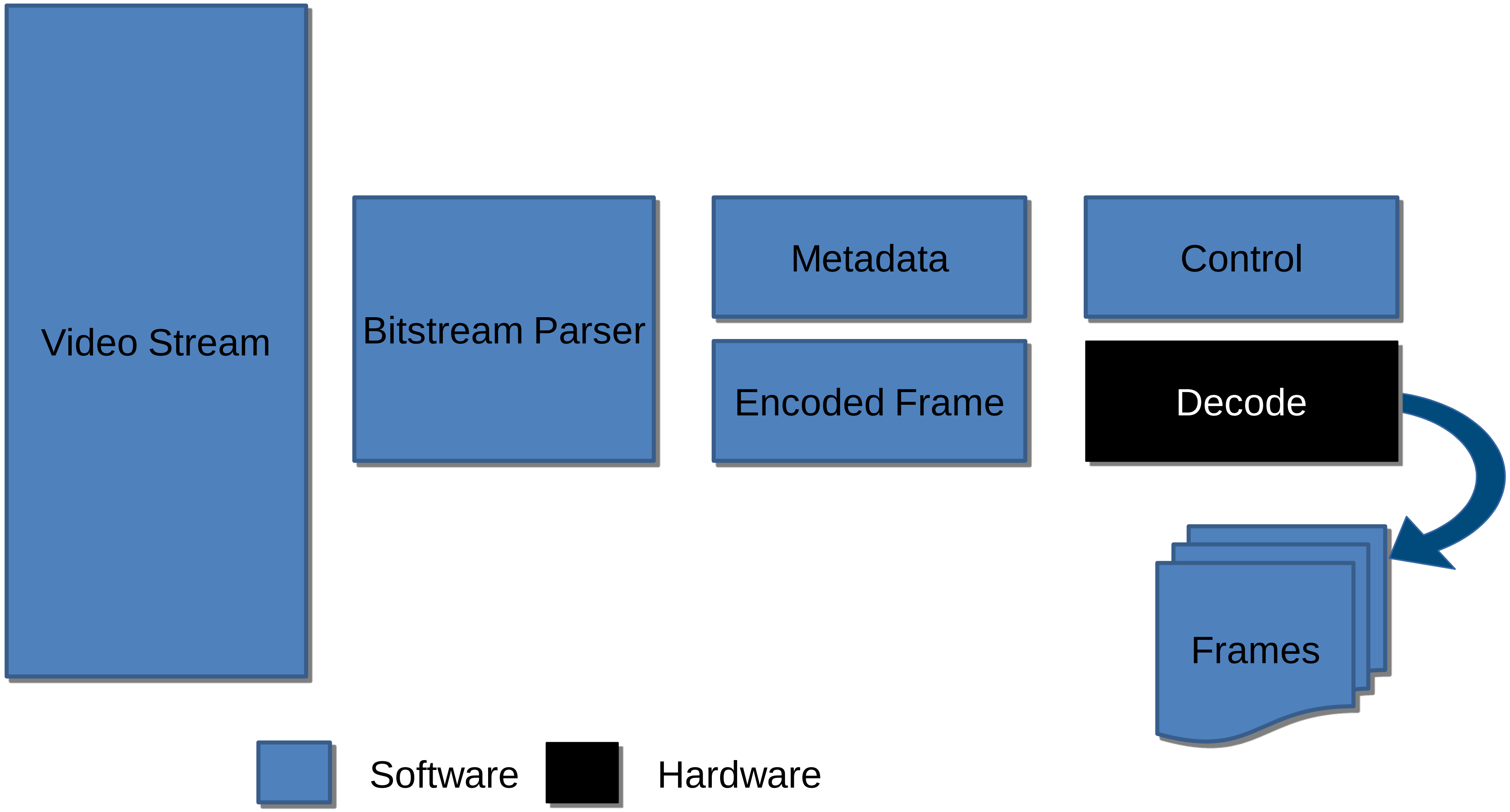
- **Stateful Video Codec***
  - *Decoder* takes complete chunks of the bitstream and decodes them into raw video frames in display order. The decoder is expected not to require any additional information from the client to process these buffers
  - *Encoder* takes raw video frames in display order and encodes them into a bitstream. It generates complete chunks of the bitstream, including all metadata, headers, etc. The resulting bitstream does not require any further post-processing by the client

- **Stateless Video Codec***
  - Is a *decoder* that works without retaining any kind of state between processed frames. This means that each frame is decoded independently of any previous and future frames, and that the client is responsible for maintaining the decoding state and providing it to the decoder with each decoding request

* from the LKML: https://lkml.org/lkml/2019/1/24/246

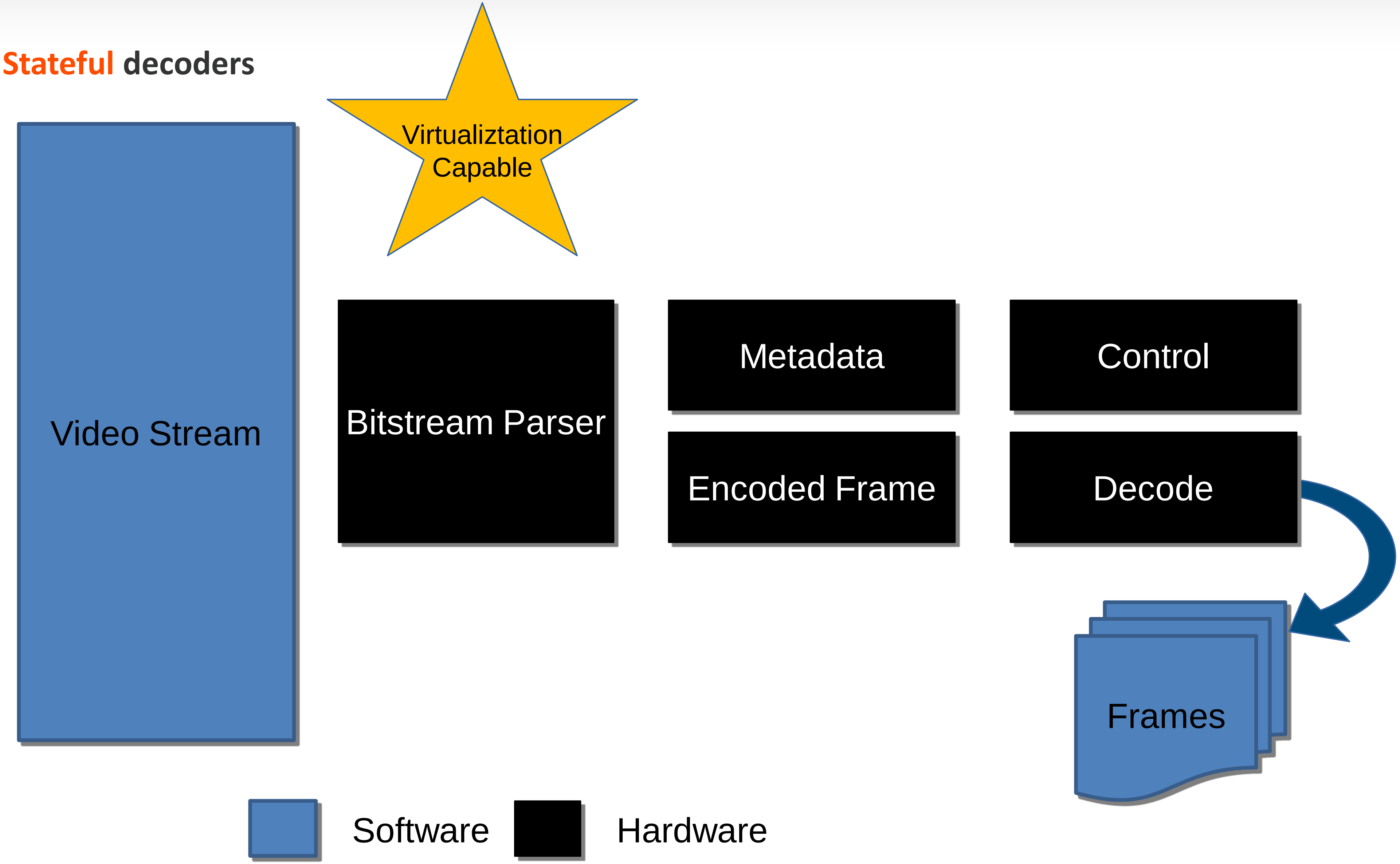
Codec device types

Stateless decoders



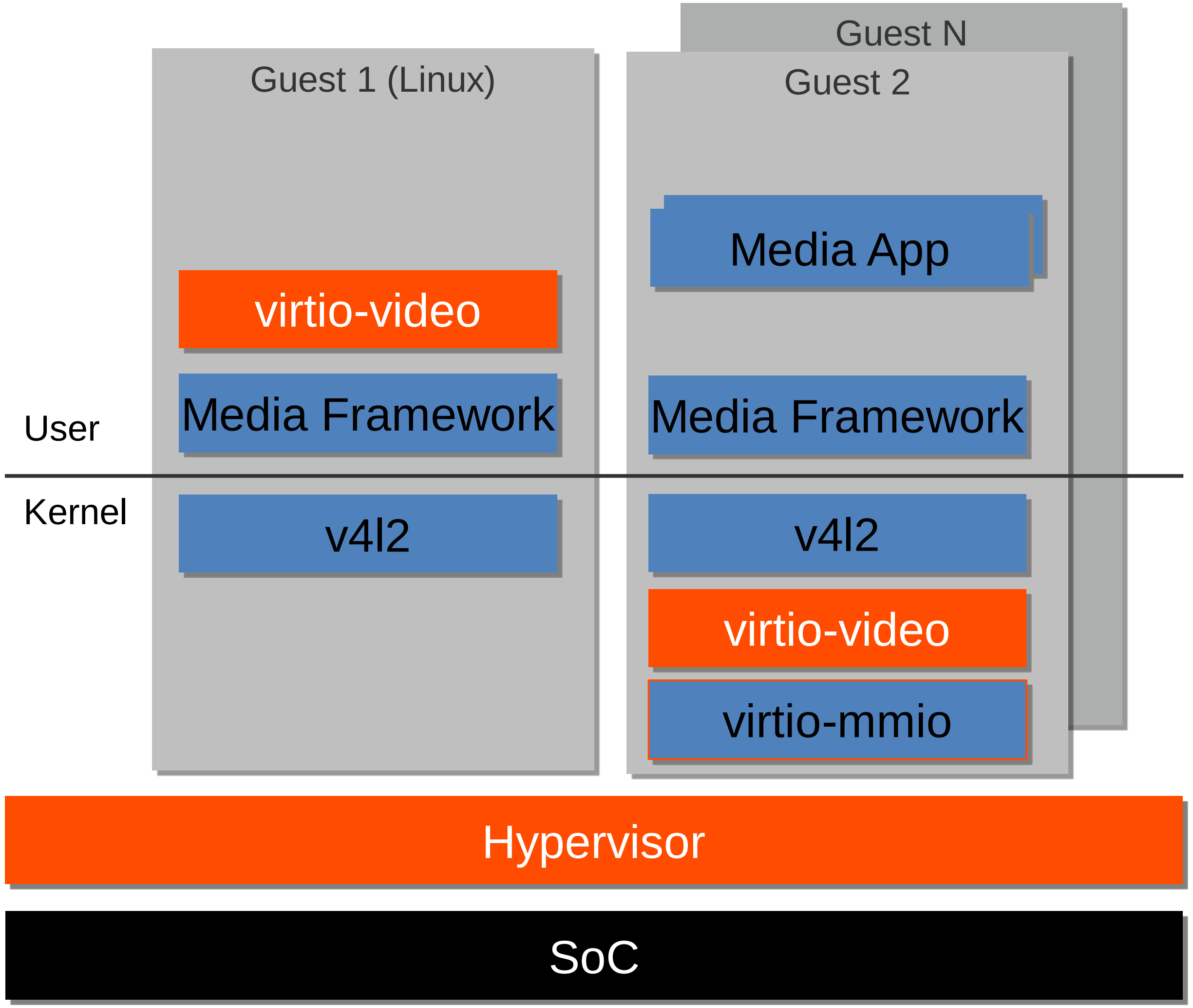
Codec device types

Stateful decoders



virtio-video on Linux based systems

OPENSYNERGY



- V4l2 based driver
- Stateful interface
- Supports:
  - Hardware video codec virtualization
  - Camera input
- Memory-to-memory or device-to-device by use of DMABUF
- Memory model same as virtio-gpu
- virtio-gpu and virtio-video can share buffers

All of the above is WIP !

virtio-video challenges

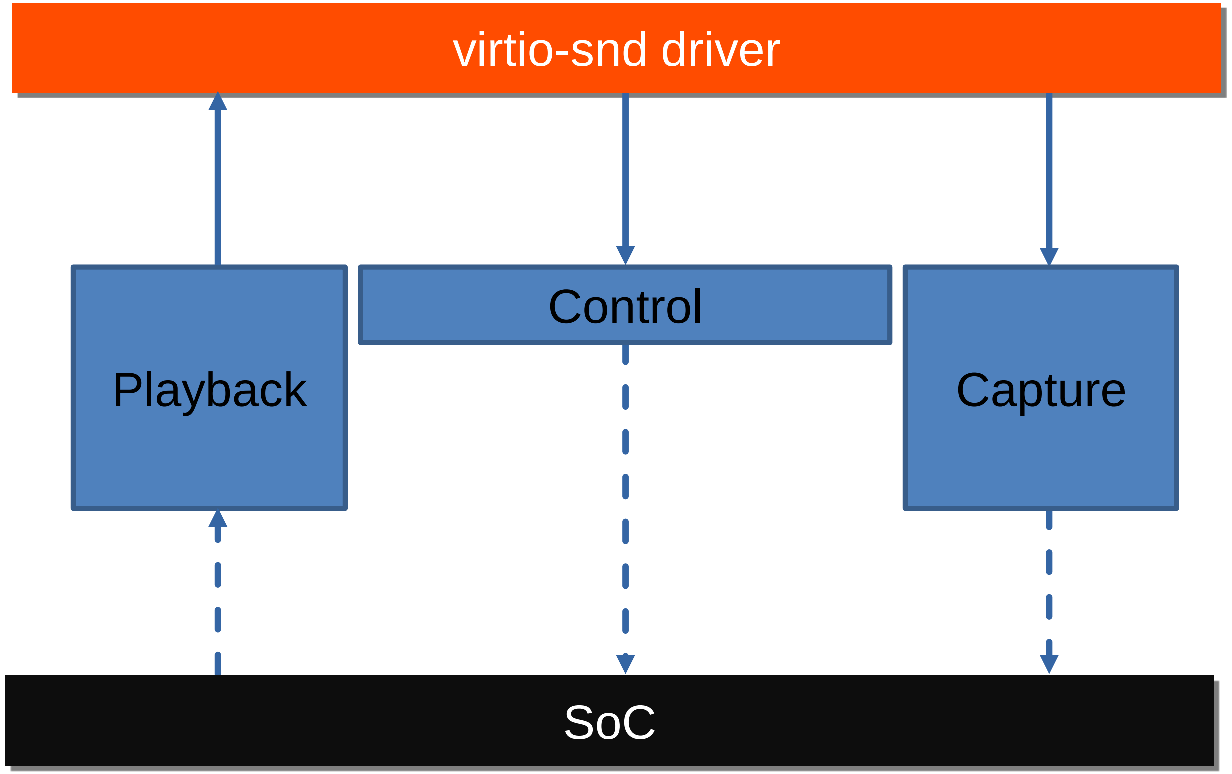
- **Configuration**
  - No APIs really make it possible to get all supported hardware features. Datasheet is the only source of reliable information
  - Virtio device configuration layout can be very complex, especially for devices with many customizable controls
- **BSP versions**
  - The media subsystem in the upstream kernel is evolving rapidly. E.g. the 4.14 kernel does not contain a definition of the H265/HEVC video format
- **Android integration**
  - Currently OMX is ubiquitous. Codec2 is the new HAL for v4l2, but does not work out of the box

virtio-snd

Virtual audio required functional

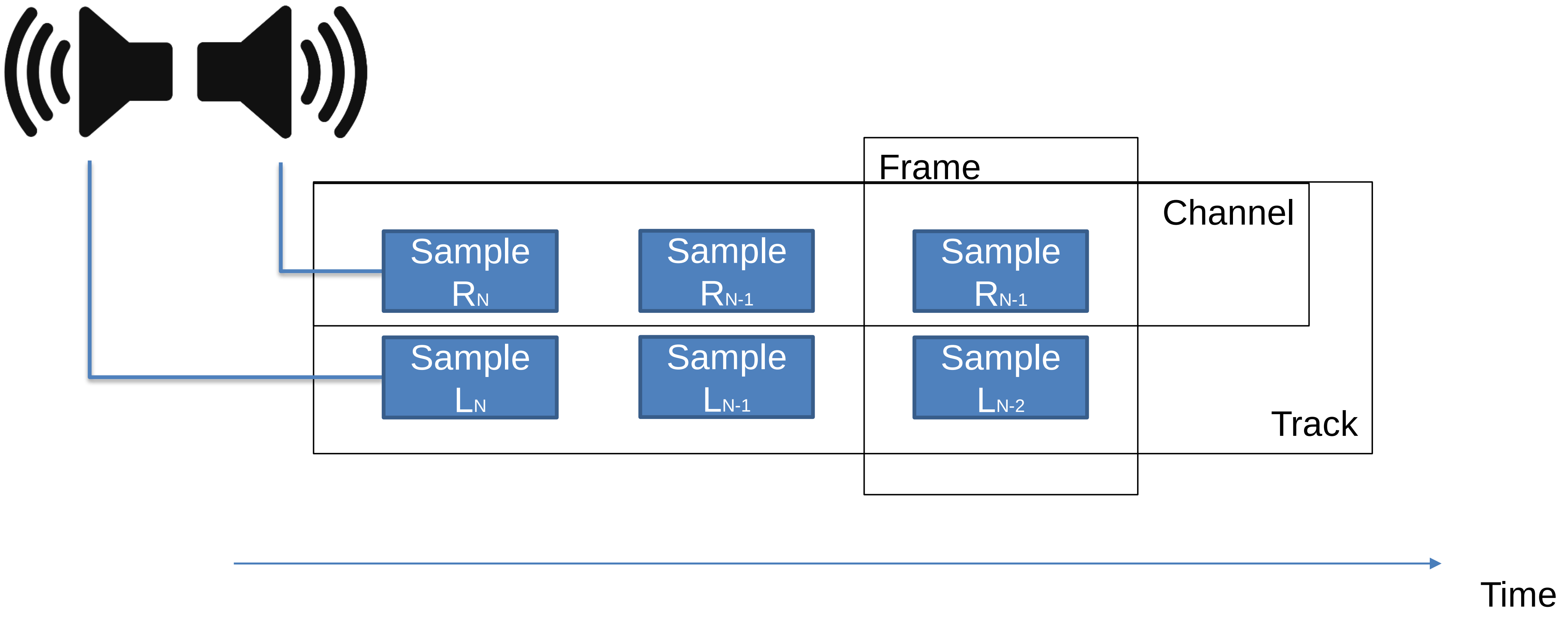
The virtio-snd device performs operations on **PCM audio streams**

- Playback
- Record
- Controls



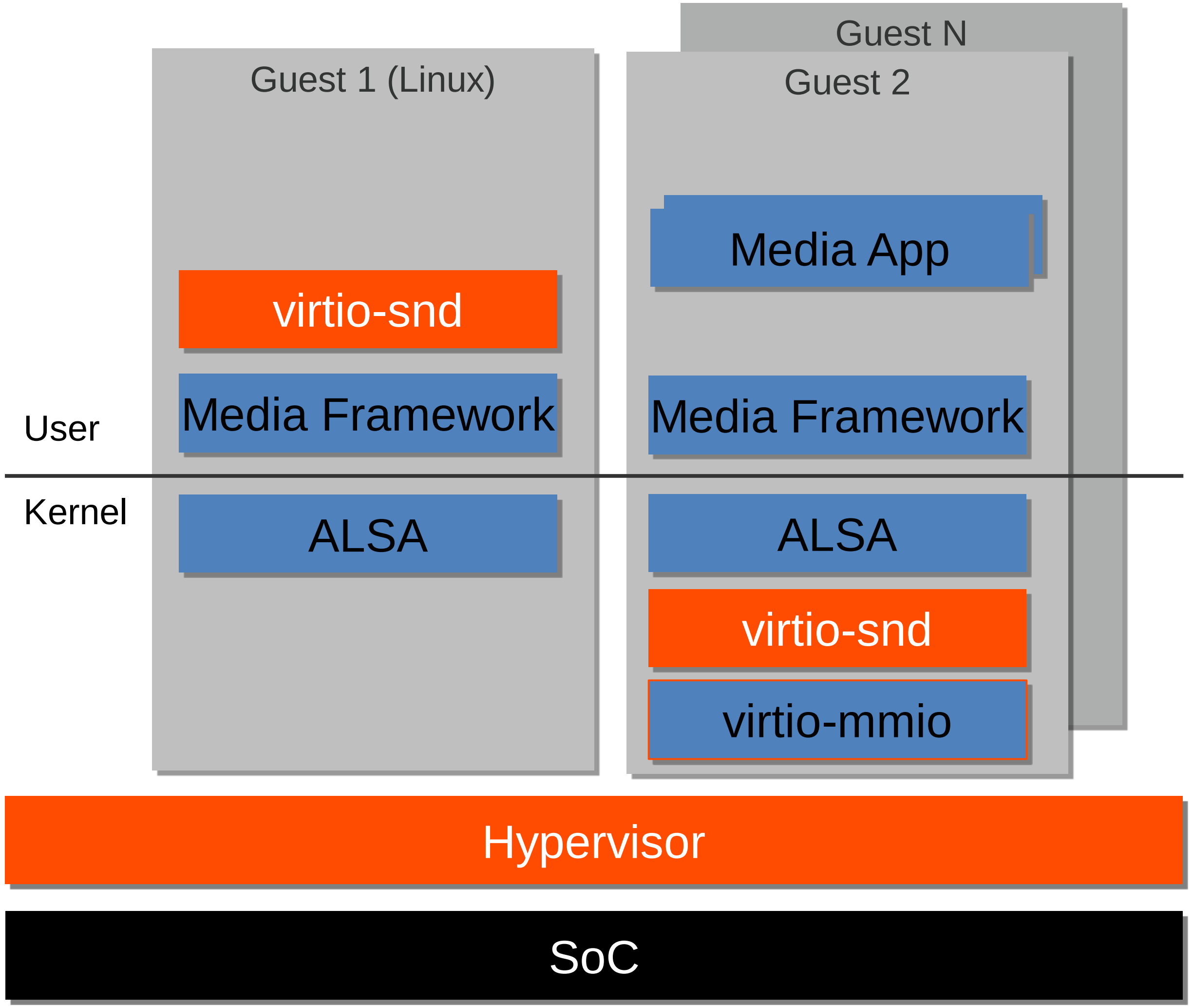
PCM audio stream

OPENSYNERGY



virtio-snd on Linux based systems

OPENSYNERGY



- ALSA based driver
- Supports
  - Audio playback
  - Audio capture
- Mmap capable
- Only PCM stream leaves/enters guests
- Playback command flow:
  - Set format
  - Prepare
  - Playback pre-buffer
  - Start
  - Pause/unpause
  - Stop

virtio-snd challanges

- **Broken BSP**
  - Stream stopped playing by it‘s own. BSP debugging was needed
- **Playback stream underruns**
  - Hypervisor scheduler should be carefuly configured
  - Driver‘s optimal design couldn‘t be foreseen – experiments needed
- **Playback stream seeking**
  - Seek back in a buffer that was already played in device‘s HW?
- **Latency requirement**
  - Should be low
- **Stream start-up time requirement**
  - Should be low

virtio-video and virtio-snd outlook

OPENSYNERGY

virtio-video

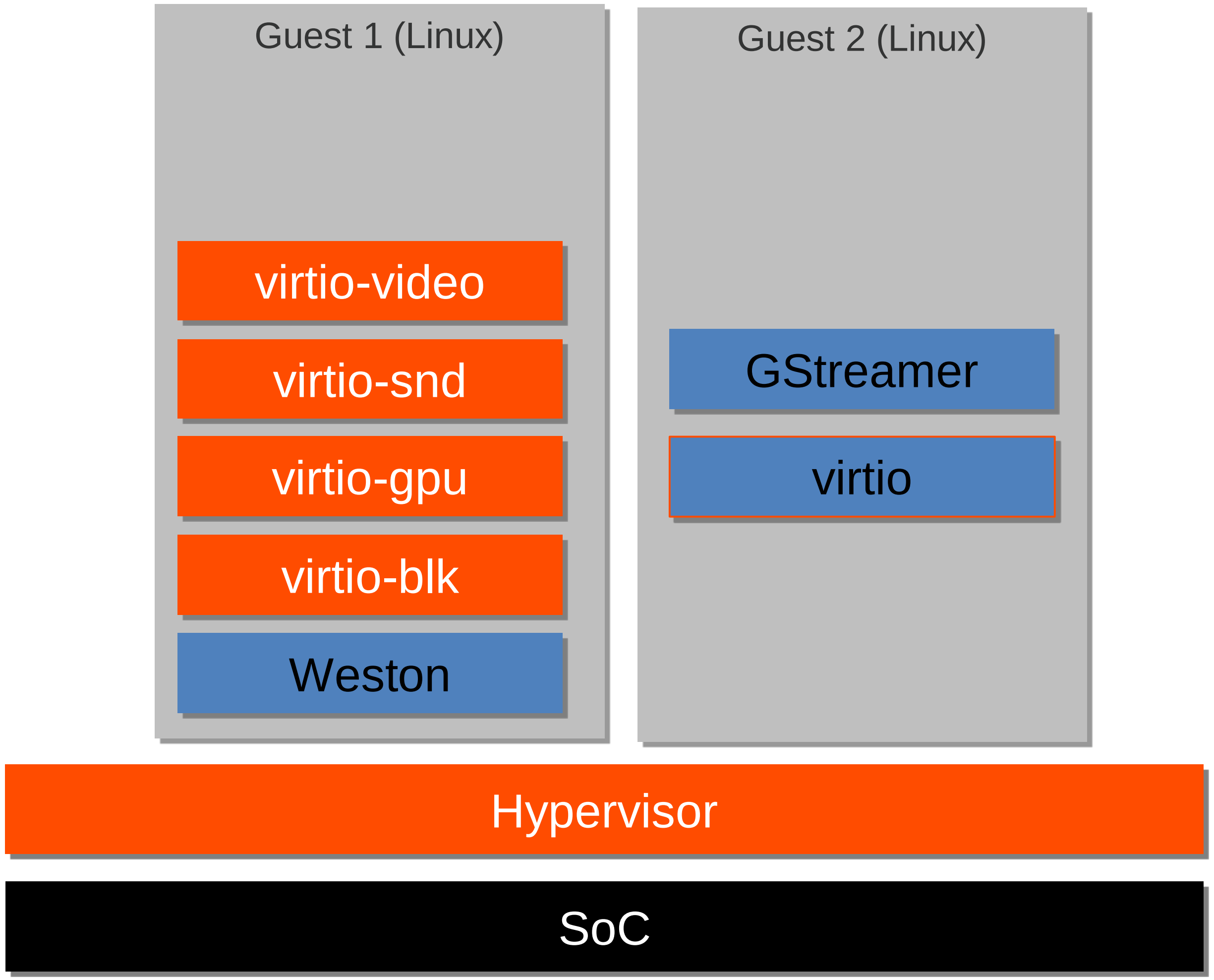
- **POC**
  - With COOQS hypervision on Linux Renesas RCar H3

virtio-snd

- Proposed specification is on **virtio-devel** mailing list
  - Dicussion still ongoing. Updated specification to follow
- **POC**
  - With COOQS hypervision on Linux Renesas RCar H3
  - QEmu-KVM running Linux
  - QEmu-KVM running Windows 10
  - QEmu-ARM running Android
- Linux kernel **driver RFC** patchset is ready and will be posted shortly
- **QEmu** reference implementation to follow

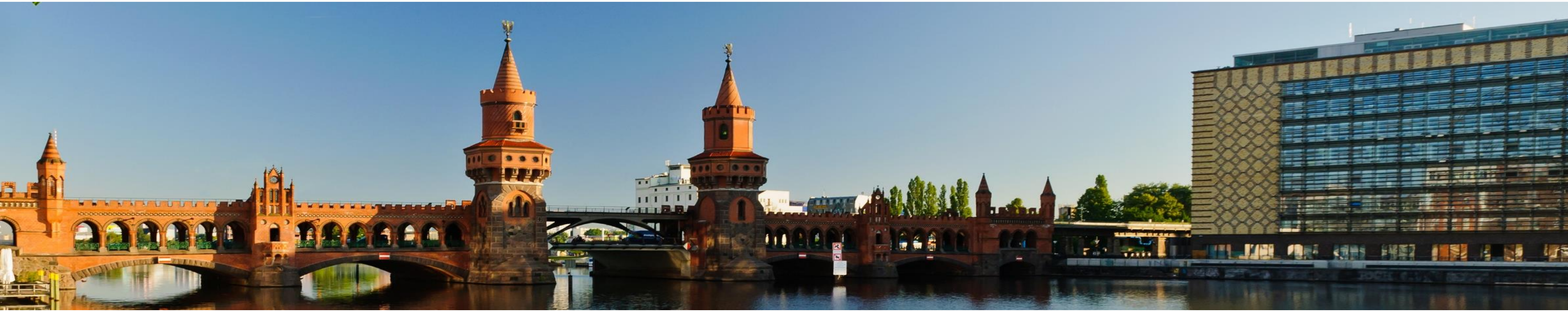
Demo setup

OPENSYNERGY



- Renesas RCar H3 Salvator XS
- Two Poky Linux guests
- Virtio devices
  - Video
  - Audio
  - Gpu 2D
  - Block device
- 720p H264 encoded sample video


THANK
YOU
GRACIAS
ARIGATO
SHUKURIA
DANKSCHEEN
BIYAN
SHUKRIA
TINGKI
YAQHANYELAY
TASHAKKUR ATU
SUKSAMA
EKHMET
GRAZIE
MEHRBANI
PALDIES
BOLZÏN
MERCI
GOZAIMASHITA
EFCHARISTO
JUSPAXAR
KOMAPSUMNIDA
MAAKE
SPASSIBO
SNACHALHUYA
CHALTU
NUHUN
WABEEJA
MAITEKA
SPASIBO
DENKAUJA
NENACHALHYA
MERASTAWHY
GAEJTHO
TAVTAPUCH
MEDAWAGSE
AGUYJE
FAKAAUE
SIKOMO
MINMONCHAR



Headquarter

Berlin

OpenSynergy GmbH

Rotherstraße 20
D-10245 Berlin
Germany
Phone +49 30 / 6098 5400

Further Locations

Utah

OpenSynergy, Inc. (USA)

765 East 340 South
Suite 106
American Fork, Utah 84003
USA

California

OpenSynergy, Inc. (USA)

501 W. Broadway, Suite 832
San Diego, California 92101
USA
Phone +1 619 962 1725

Munich

OpenSynergy GmbH

Starnberger Str. 22
D-82131 Gauting / Munich
Germany
Phone: + 49 89 / 8934 1333

E-Mail info@opensynergy.com
Web www.opensynergy.com