Virt-EG Virtual F2F SAT

June 16, 2021

THE LINUX FOUNDATION

Agenda

- Agenda
- [VirtIO in Virtualization] Virt-EG Work for AGL LL 40 min
- [VirtIO in Virt/Non-virt AGL] Device Common I/F 20 min

Future Steps

Virt-EG Scope

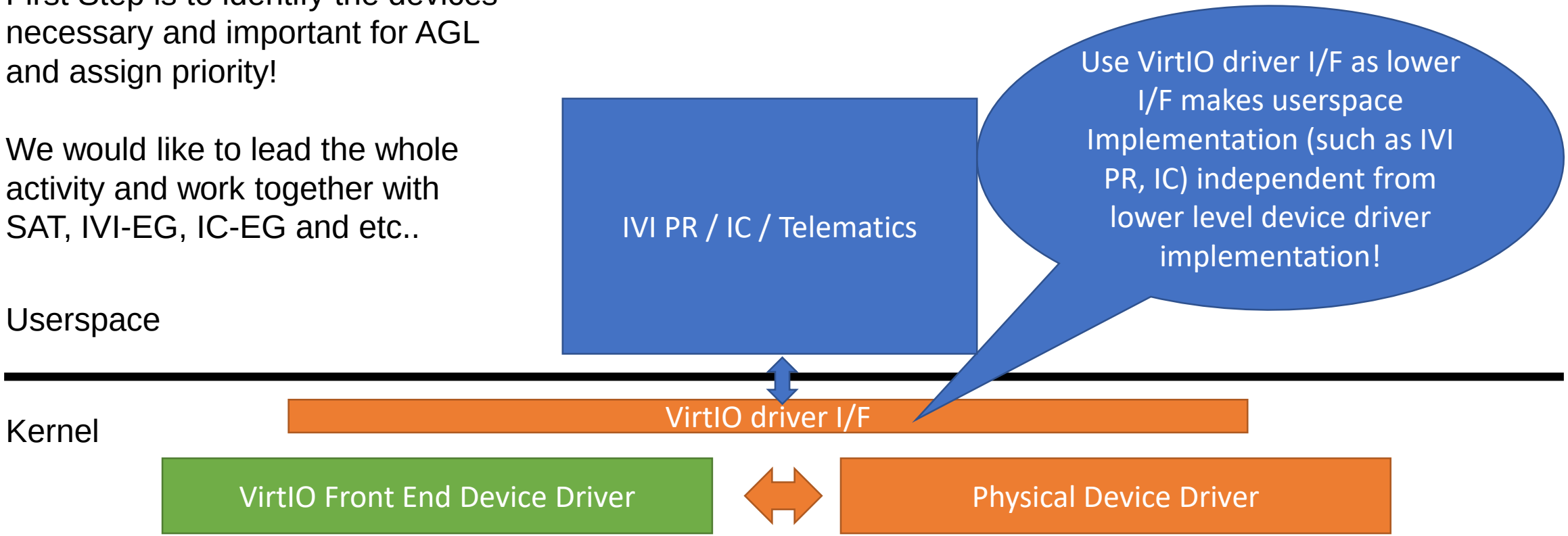
SAT Scope

SAT/ Profile-Specified EG's Scope

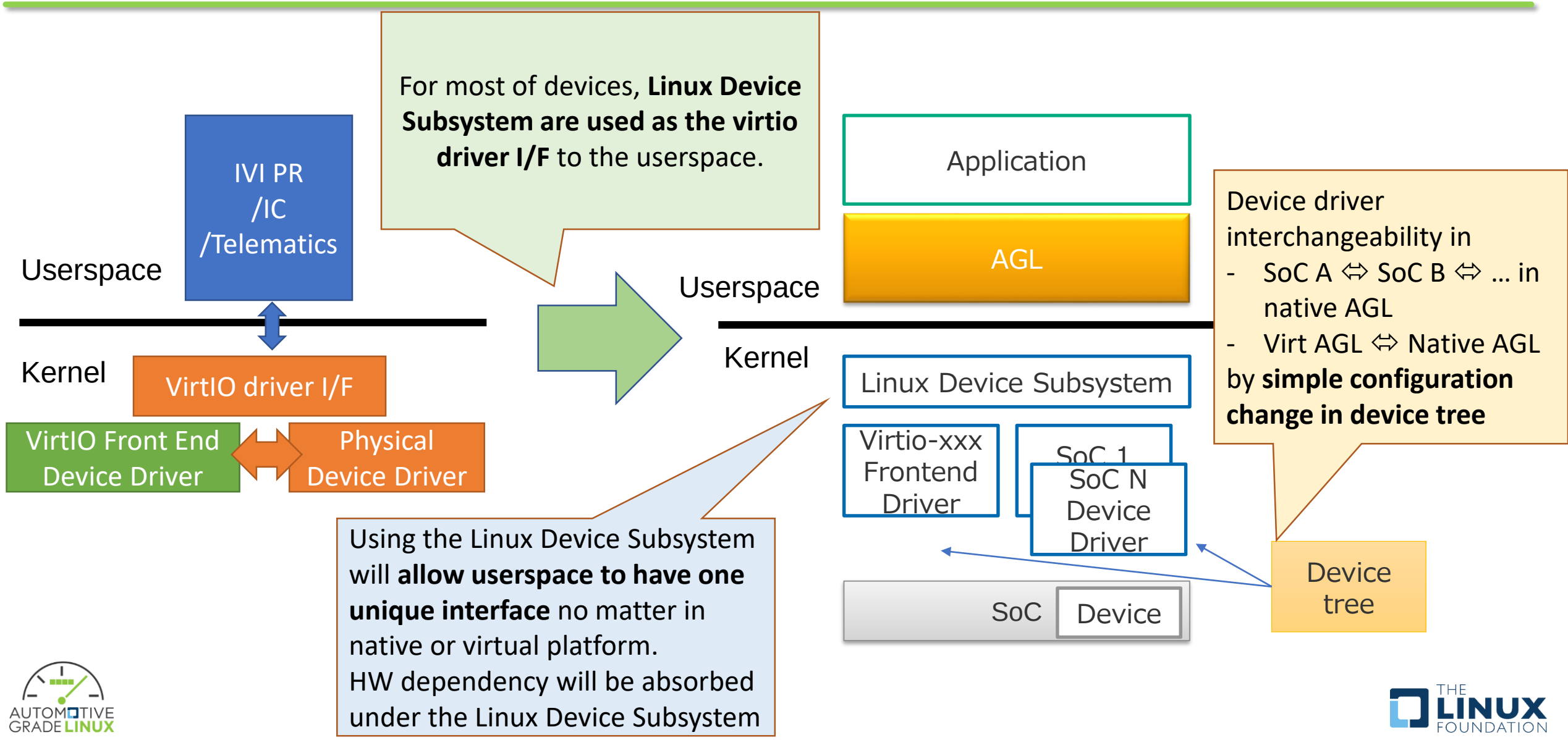
Showed in previous F2F SAT meeting in Mar

First Step is to identify the devices necessary and important for AGL and assign priority!

We would like to lead the whole activity and work together with SAT, IVI-EG, IC-EG and etc..



Concept



AUTOMOTIVE GRADE LINUX

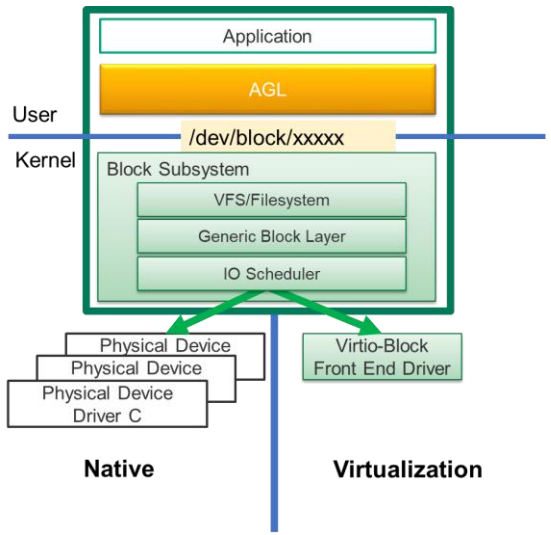
THE LINUX FOUNDATION

Kernel Level Common Device I/F

- VirtIO utilize the standard Linux Device Subsystem to provide unique interface to userspace independent from HW. (Linux Device Subsystem can be seen as VirtIO driver I/F to userspace)
- Benefiting from the VirtIO standardization, Linux subsystem is also growing to a more mature common interface that different vendors can take advantage of it.
- Same idea can be applied to native case to absorb HW difference under Linux Device Subsystem and use common user-kernel interface.

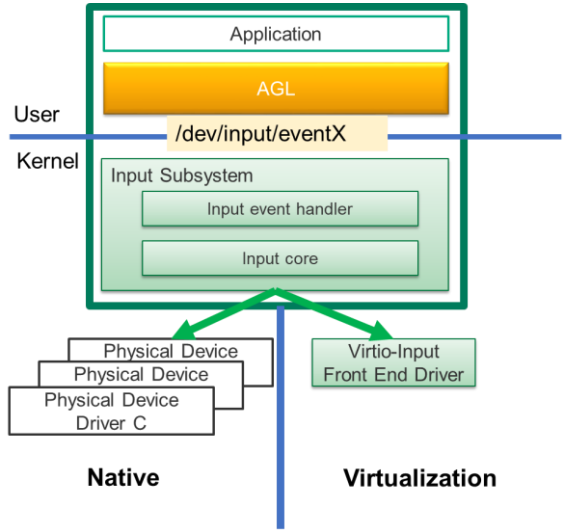
Block Device

Standard & Mature



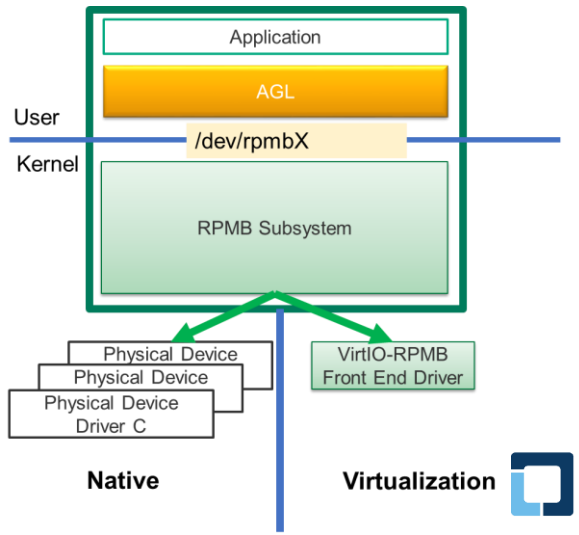
Input Device

Standard but Need Extension for Automotive Production Use Case

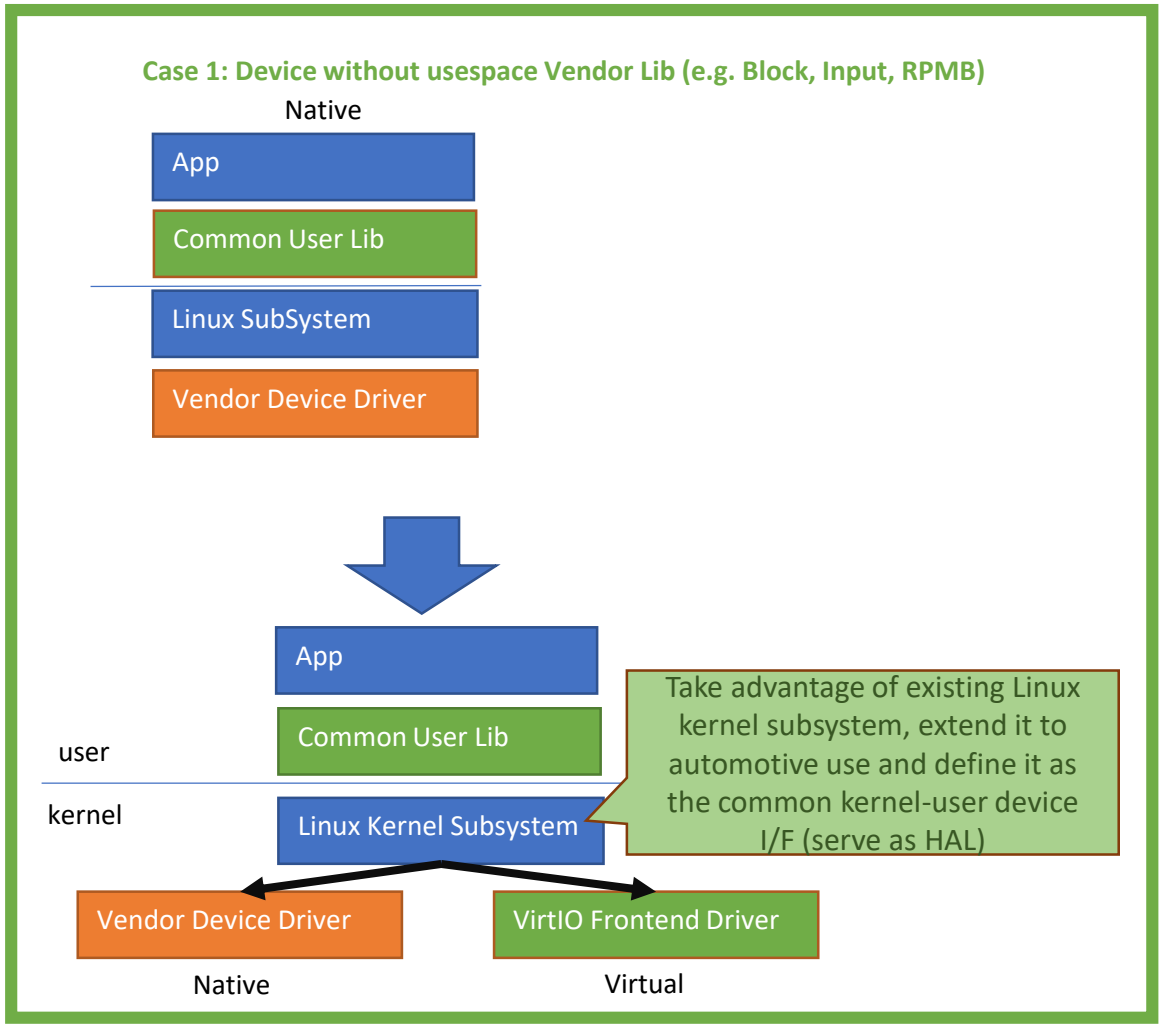


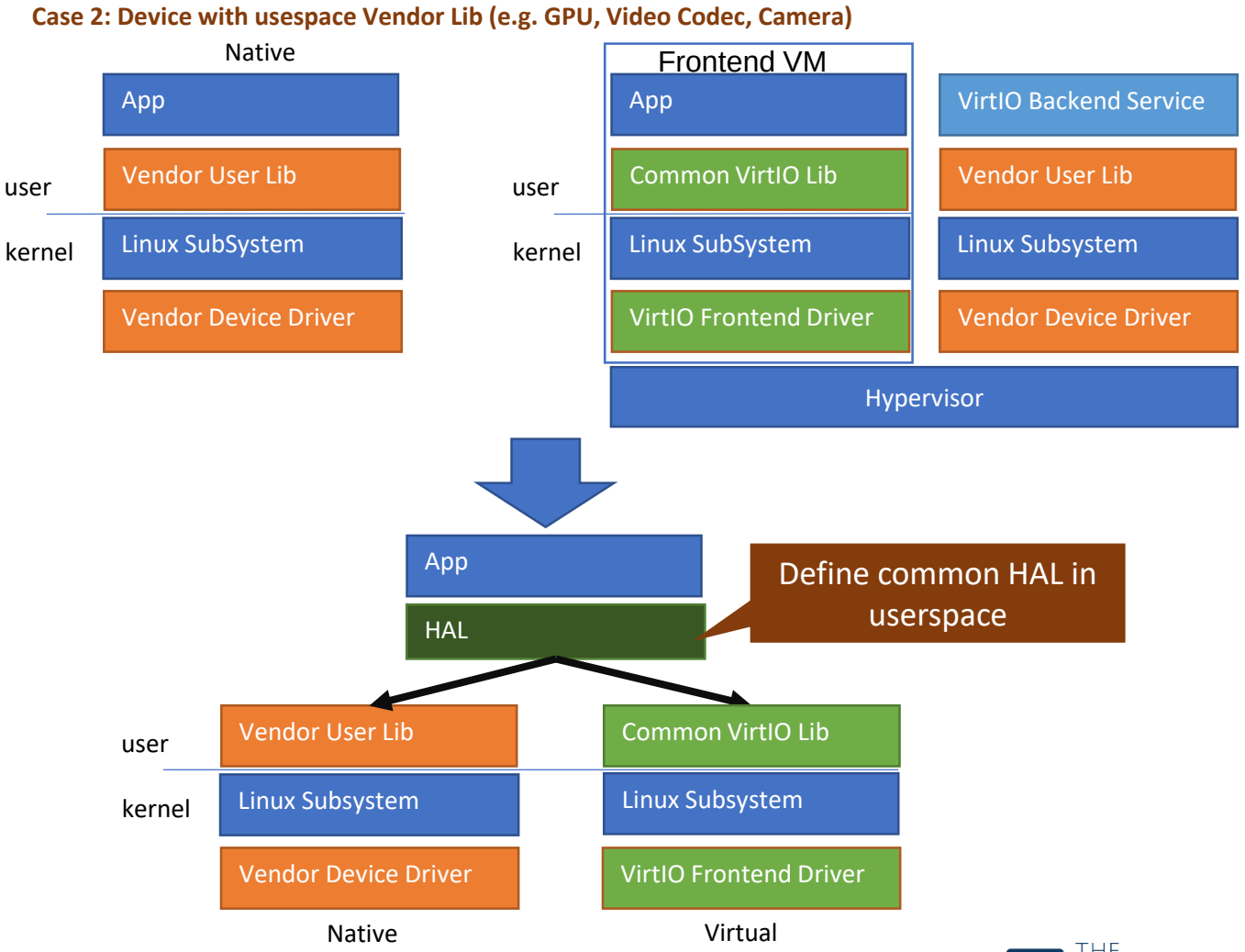
RPMB Device

Under standardization



How we achieve HAL?





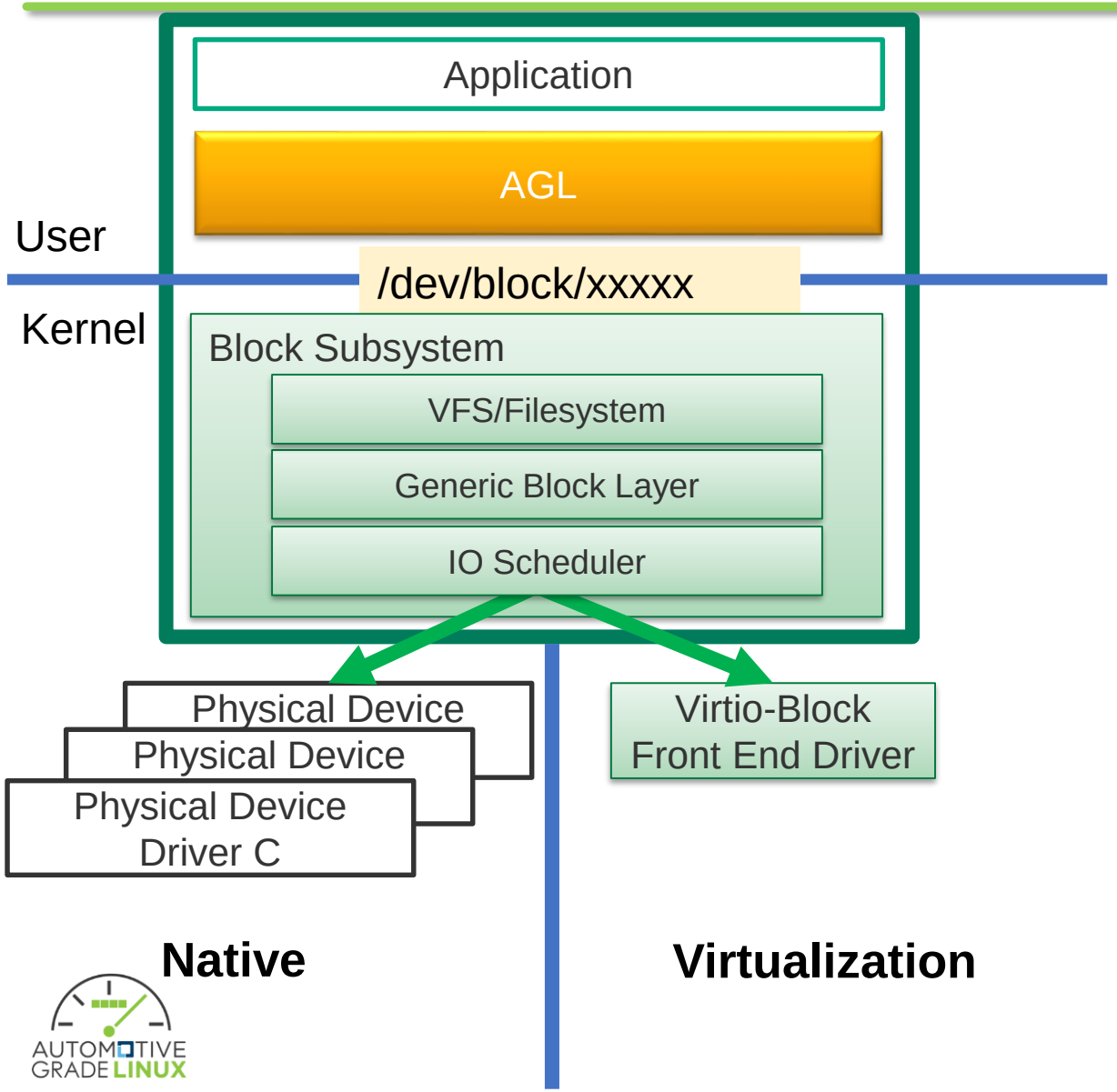
Our Focus

AGL Priority

https://docs.google.com/spreadsheets/d/1jpLNUBKz19LOdtGyqan5Wk4OgZFFxUNcSpMrFMPFCKI/

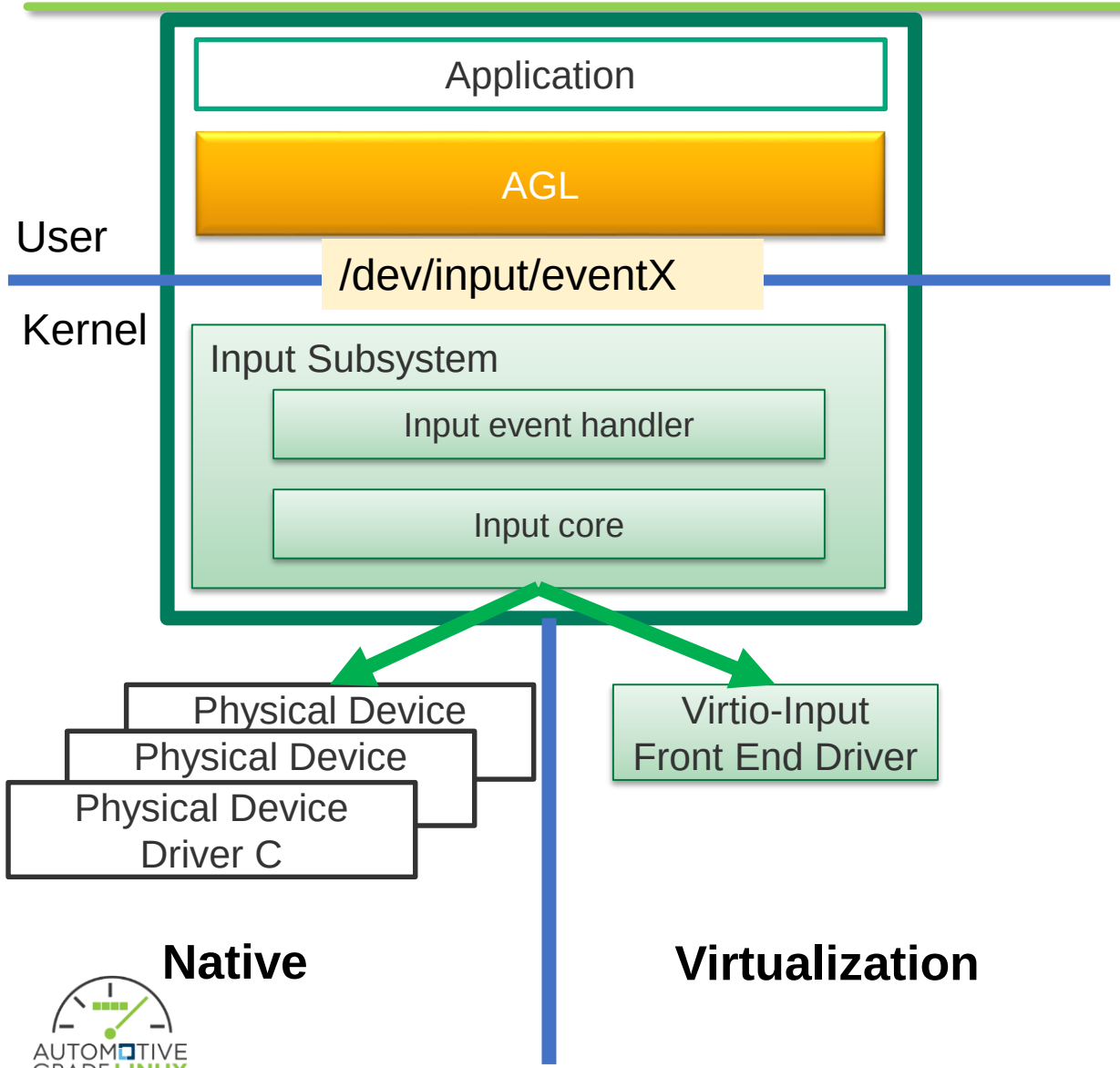
Device	VirtIO Device	Linux Kernel Version	OASIS Specification	Linux Kernel Device Subsystem	Total Score	AGL Overall Priority
Input Device (e.g. touch)	virtio-input	v4.0-rc4	v1.1	evdev (Input Subsystem)	29	1
Display (Video Display Controller)	virtio-gpu(2d)	v4.1-rc4 (2d)	v1.1	DRM & KMS	27	2
GPU	virtio-gpu(3d)	v4.3-rc5 (3d)	v1.2	DRM & KMS	26	3
CAN bus	virtio-can	-	Spec RFC in virtio-comment ML	socket CAN	20	4
Block Device	virtio-blk	v2.6.23	v1.0	block subsystem (/dev/block)	19	5
Audio (microphone & speaker)	virtio-snd	v5.13	v1.2	ALSA	18	6
Ethernet	virtio-net	v2.6.23	v1.0	network subsystem	11	7
Bluetooth	virtio-bluetooth	-	-	Bluetooth subsystem (vitrio-bt has HCI IF)	9	8
SPI	virtio-console	v.2.6.23	v1.0	SPI subsystem	8	9
Serial console	virtio-console	v.2.6.23	v1.0	tty/serial interface	8	9
SCMI (Sensors, Clocks/Regulators, Performace ...)	virtio-scmi	Upstreamed but under review (RFC v2)	v1.2	no specific interface to userspace at the moment (maybe can linked to industrial IO)	8	9

Block Device



- Mature Standard Linux Block Subsystem are commonly used by virtualization world and native world without hardware dependency.
- Data read/write/trim operation enabled by block subsystem have already covered basic use cases in automotive
- With the existing I/F, abstraction of hardware has been already achieved and few work need to be done.

Input Device



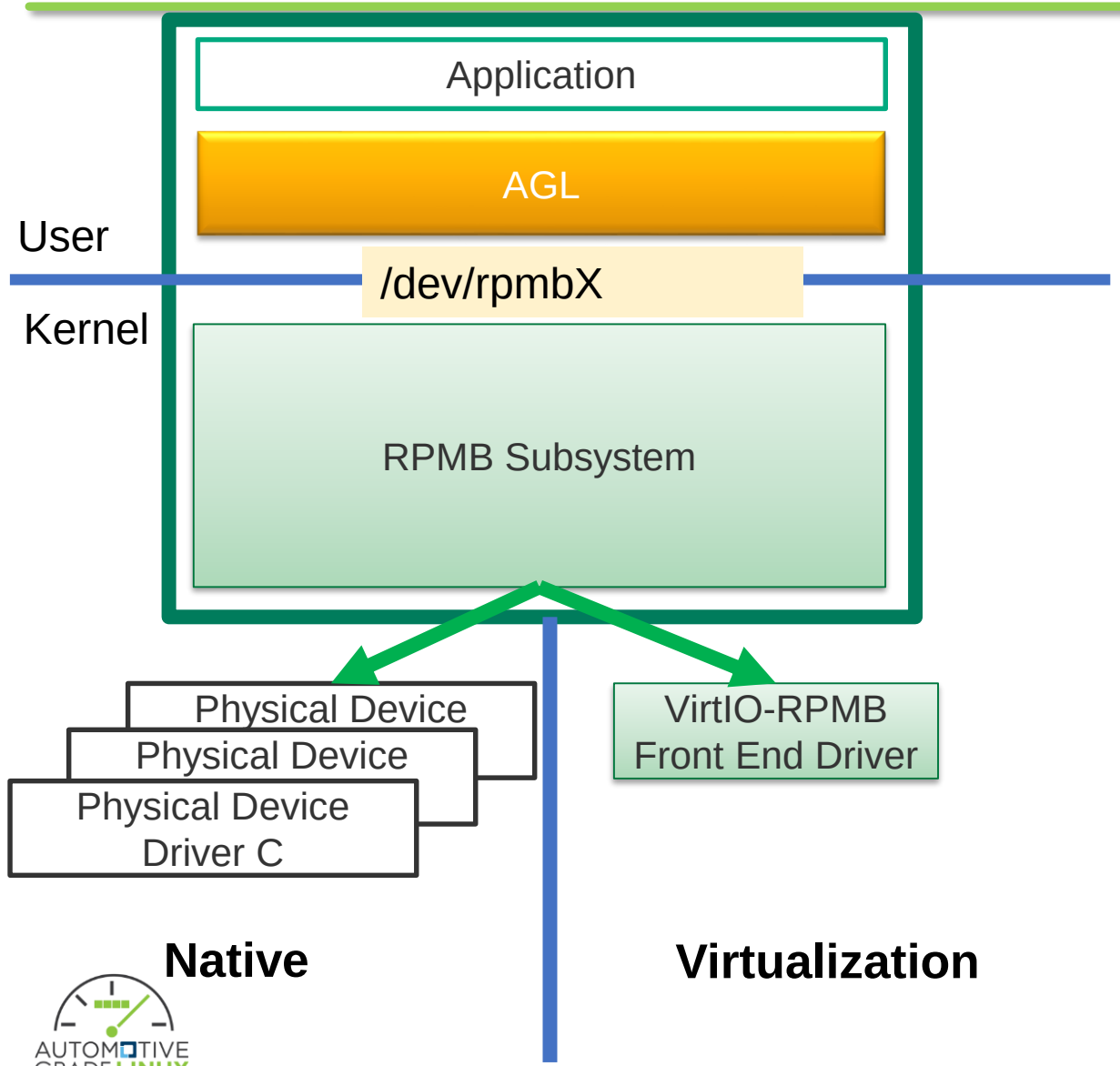
- Standard "*evdev*" generic input event interface
  - passed events generated in kernel straight to the program with same event codes on all architectures and HW-independent.
- Additional Extension is needed for automotive use
  - Multi-touch protocol has been supported in input subsystem but extension of virtio-input front end is needed (planned in Virt-EG activities).
  - Current input subsystem doesn't cover the calibration/sensitivity setting and need to be extended to support the use case.

RPMB Device

- What is RPMB
  - RPMB is Replay Protected Memory Block
  - A write protected region on certain flash devices such as eMMC and UFS.
  - Fixed size partition (128KB ~ 16MB) with counter and can only be accessed by Trustzone

- Use Case: Anti roll-back and replay attack protection
  - Protect from downgrading software
  - Protect from unauthorized device unlocking (times of attempts to unlock is recorded in RPMB)
  - Secure boot (partitions write protection)

RPMB Device



- Fragmented I/F for rpmb
  - MMC: MMC_IOC_CMD ioctl
  - UFS: SG_IO ioctl
- Along with standardization of VirtIO-RPMB, standardization of Linux RPMB subsystem is progressing
  - Common RPMB subsystem with one ioctl (+simulator) /dev/rpmbX
- Apply the same RPMB subsystem to native case will help the device abstraction in the way that one unique interface is used from userspace