Example of VirtIO I/F Use in the native & virtual AGL

Apr 22, 2021

Future Steps

Virt-EG Scope

SAT Scope

SAT/ Profile-Specified EG's Scope

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..

Userspace

IVI PR / IC / Telematics

Use VirtIO driver I/F as lower I/F makes userspace Implementation (such as IVI PR, IC) independent from lower level device driver implementation!

Kernel

VirtIO driver I/F
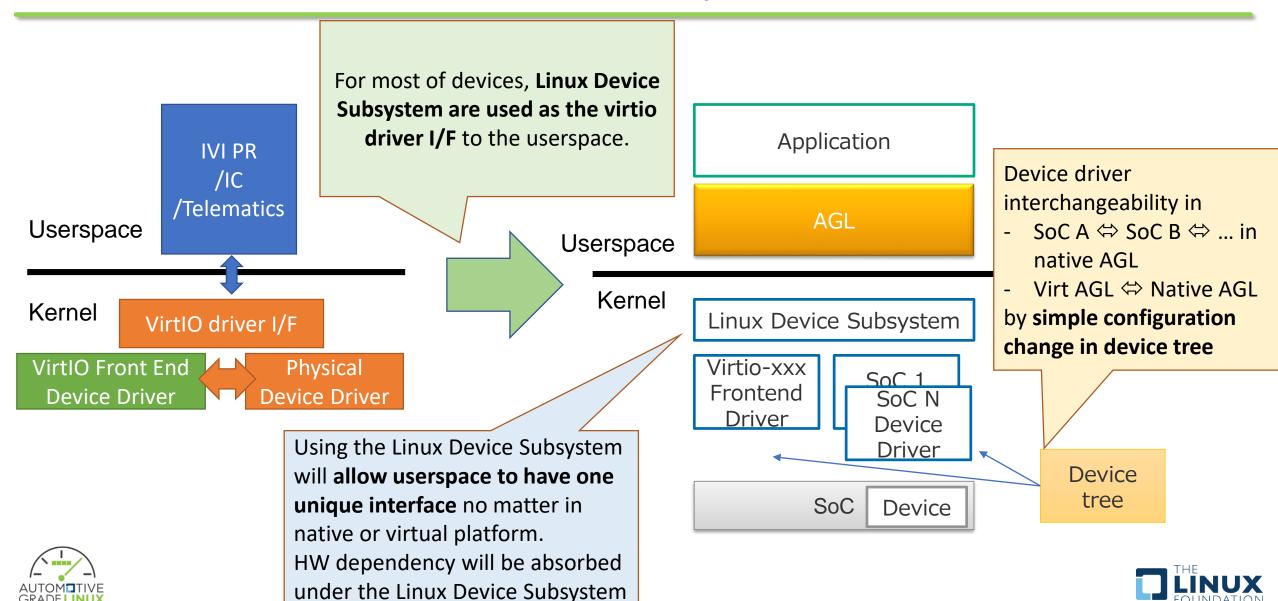
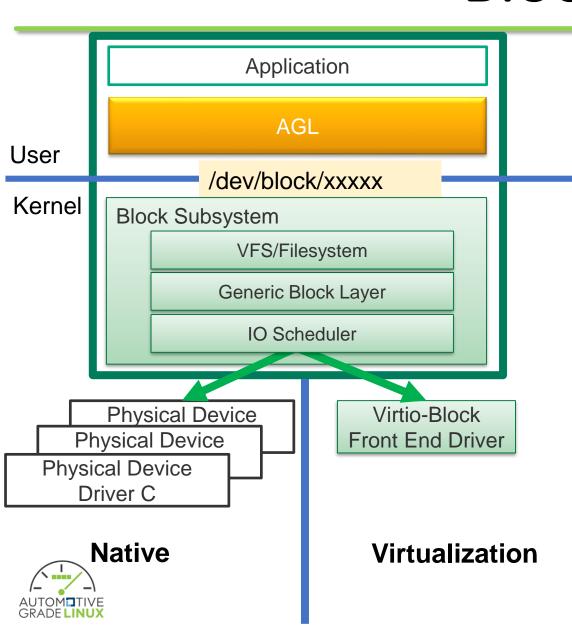VirtIO Front End Device Driver

Physical Device Driver

AUTOMOTIVE GRADE LINUX

THE LINUX FOUNDATION

Concept

For most of devices, **Linux Device Subsystem are used as the virtio driver I/F** to the userspace.

IVI PR /IC /Telematics

Userspace

Kernel

VirtIO driver I/F

VirtIO Front End Device Driver

Physical Device Driver

Using the Linux Device Subsystem will **allow userspace to have one unique interface** no matter in native or virtual platform.
HW dependency will be absorbed under the Linux Device Subsystem

Application

AGL

Userspace

Kernel

Linux Device Subsystem

Virtio-xxx Frontend Driver

SoC 1

SoC N Device Driver

SoC   Device

Device driver interchangeability in
- SoC A ⇔ SoC B ⇔ … in native AGL
- Virt AGL ⇔ Native AGL

by **simple configuration change in device tree**
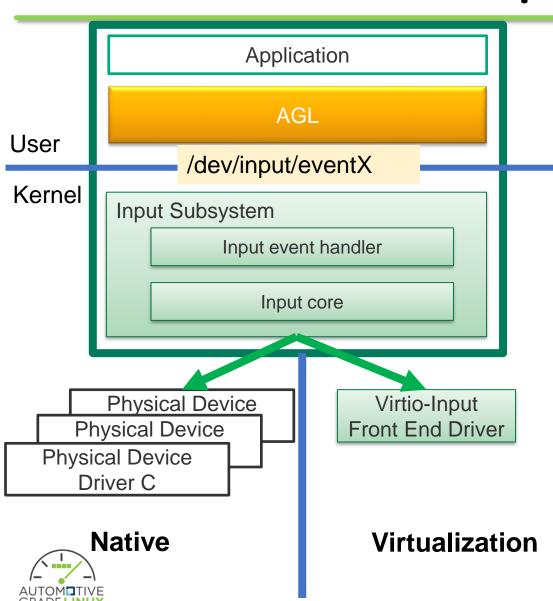
Device tree

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

Application

AGL

User
/dev/input/eventX

Kernel

Input Subsystem

Input event handler

Input core

Physical Device
Physical Device
Physical Device
Driver C

Virtio-Input
Front End Driver

Native

Virtualization

- 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.
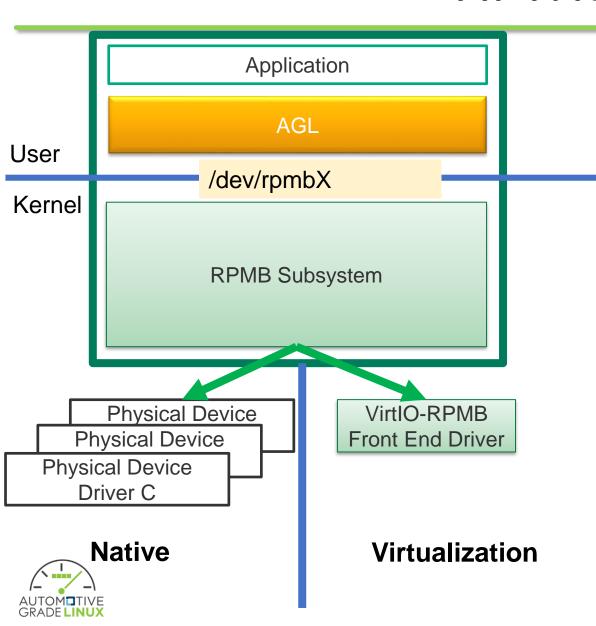
RPMB Device

- 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
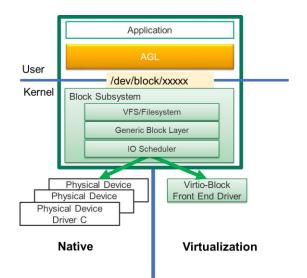
Summary

- 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.
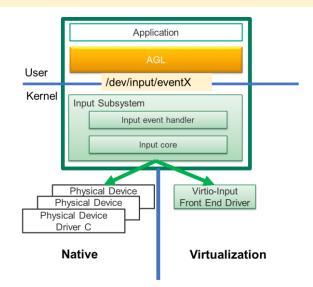
Block Device	**Input Device**	**RPMB Device**
Standard & Mature	Standard but Need Extension for Automotive Production Use Case	Under standardization