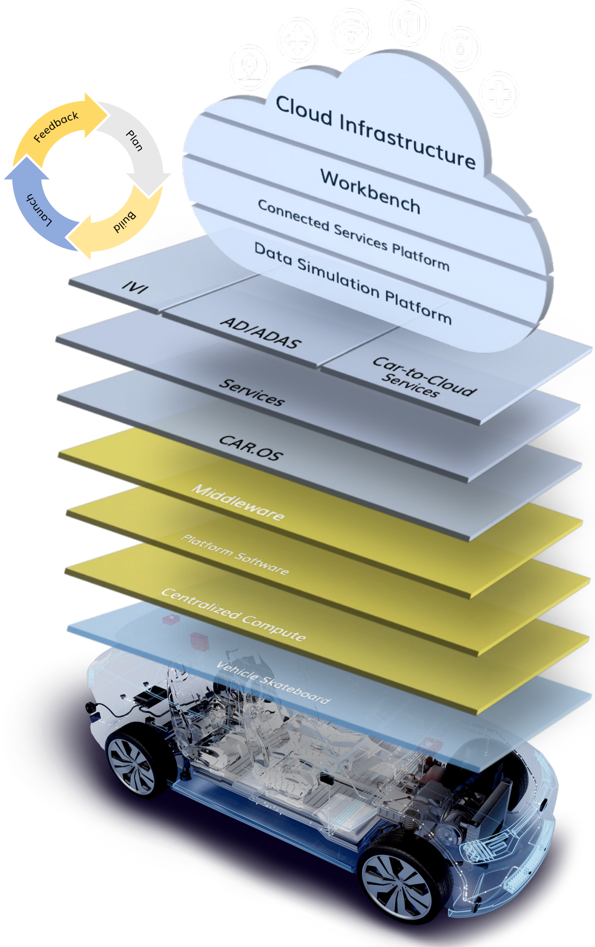
QQVP: Qualcomm's SystemC and QEMU modelling solution

Linaro Connect
MADRID 2024 | MAY 14-17 2024

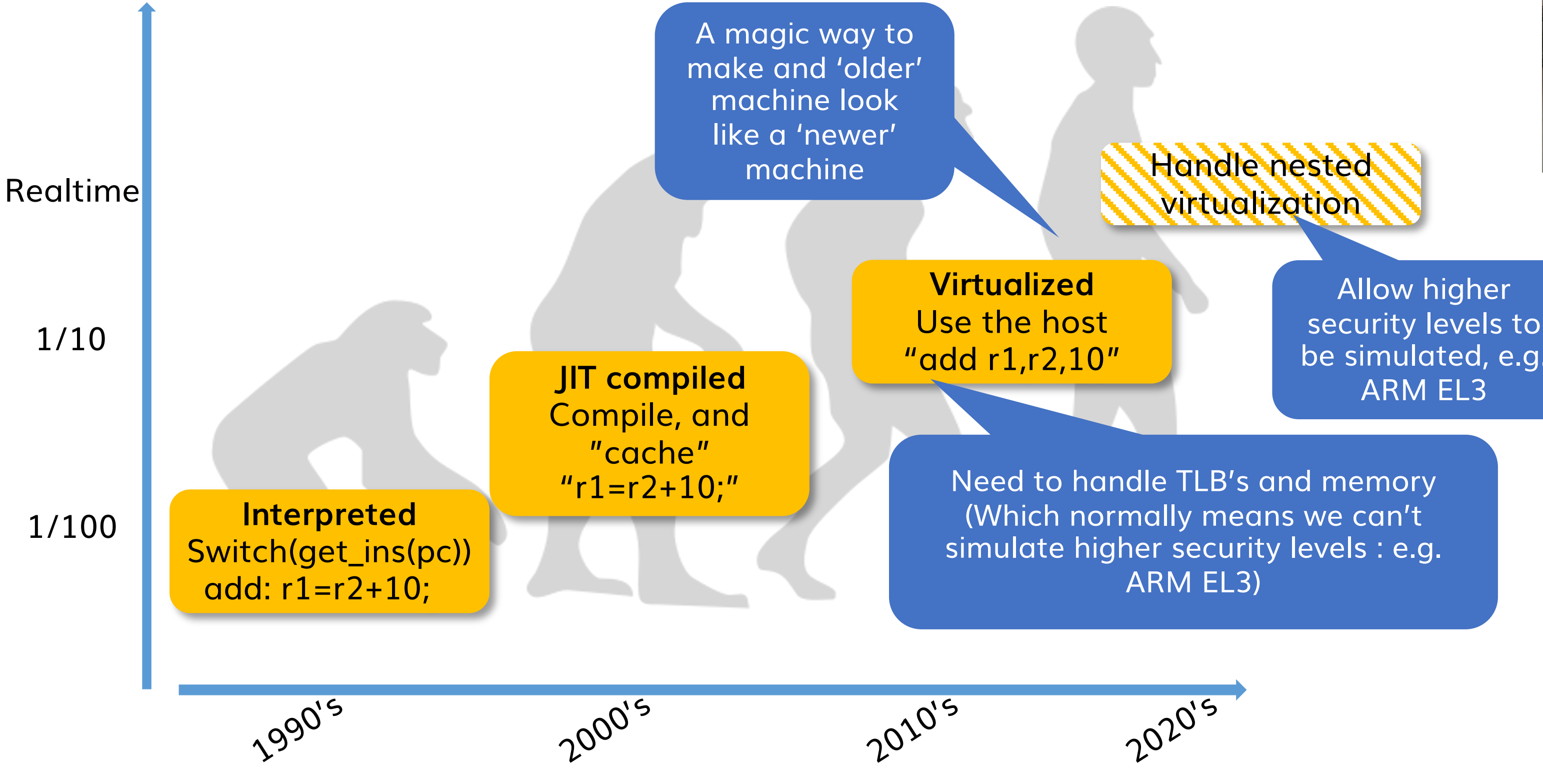
Mark Burton – Engineering Director
Antonio Caggiano – Staff Engineer

Use cases

- Software development :
  - Direct debugger connection to the kernel
  - Can see ‘hardware’ state – set breakpoints on hardware events
- Continuous Integration and test
  - Scales : runs on off the shelf machines (e.g. Amazon nodes)
  - Make use of cloud services
- Run tests that are impossible in reality
  - Can gain certification credits for extra testing performed on virtual platforms.



History of VP technology



Interpreted
Switch(get_ins(pc))
add: r1=r2+10;

JIT compiled
Compile, and
"cache"
"r1=r2+10;"

A magic way to make and 'older' machine look like a 'newer' machine

Virtualized
Use the host
"add r1,r2,10"

Need to handle TLB's and memory (Which normally means we can't simulate higher security levels : e.g. ARM EL3)

Handle nested virtualization

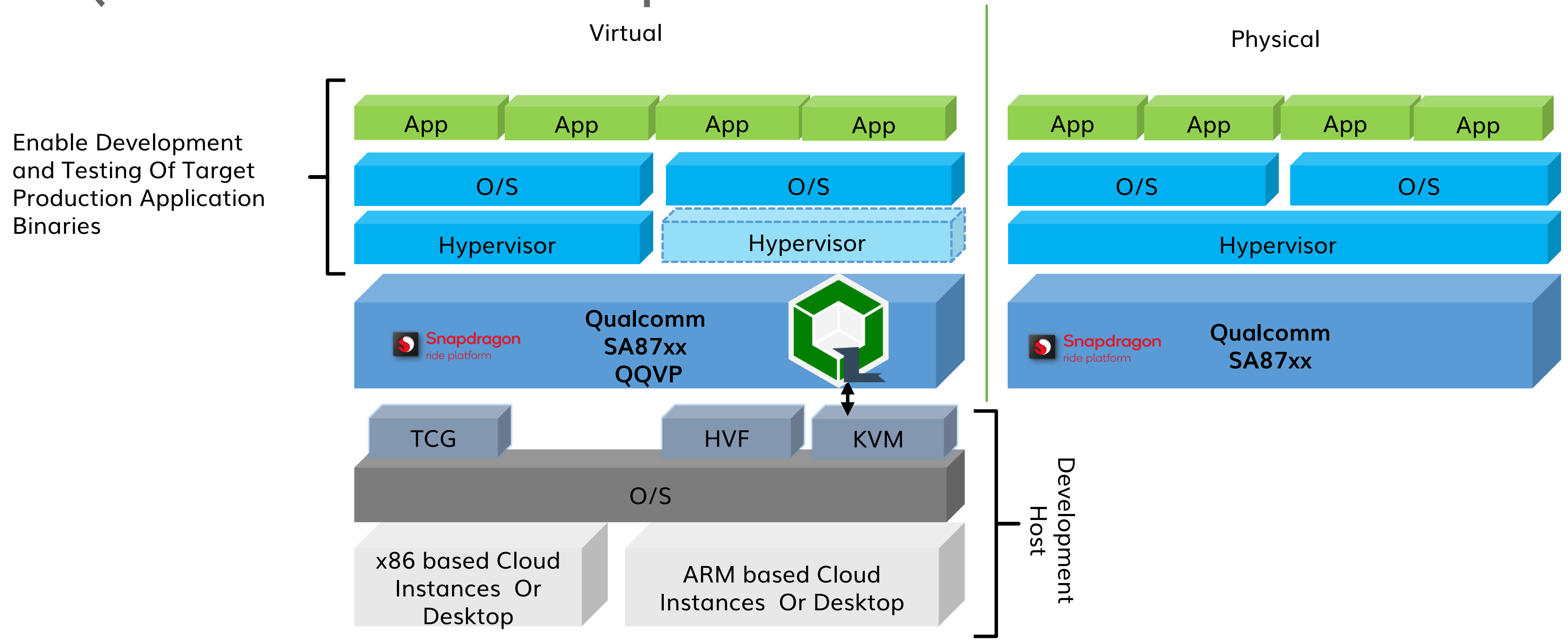
Allow higher security levels to be simulated, e.g. ARM EL3

QQVP components: open source

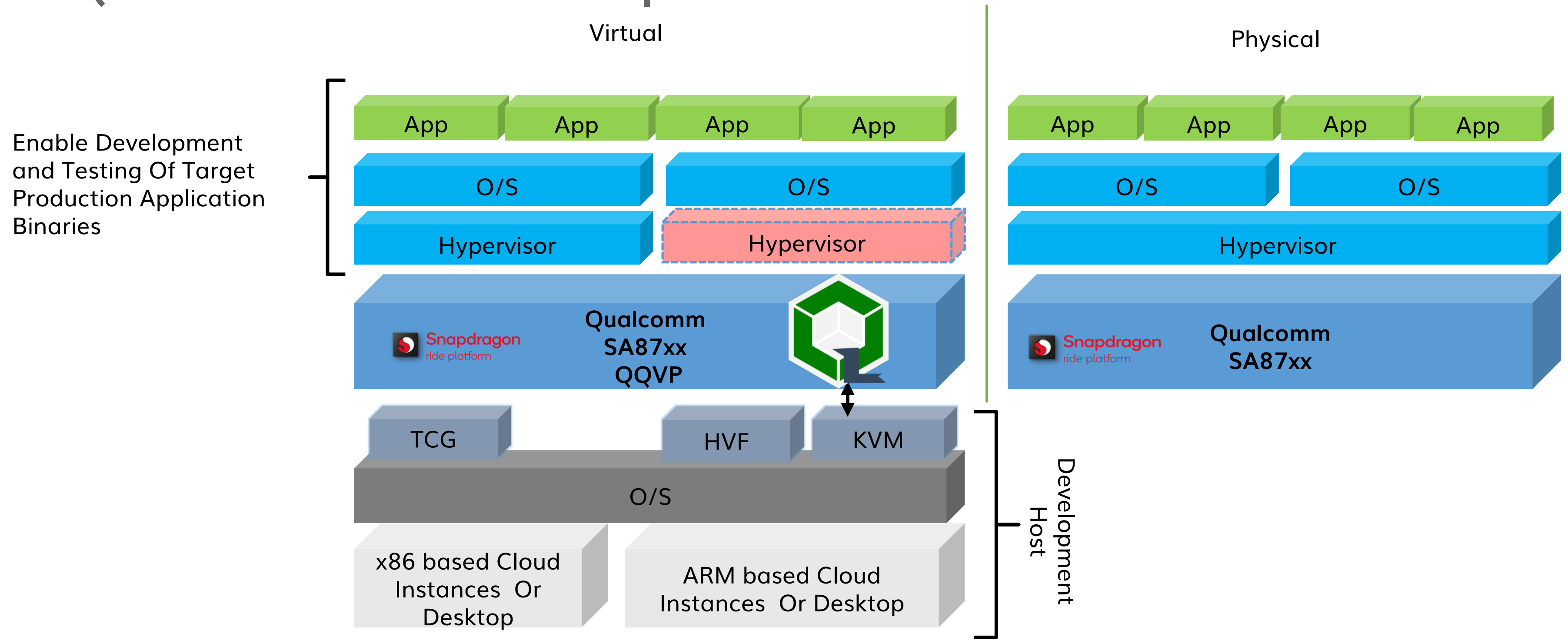
https://en.m.wikipedia.org/wiki/File:Qemu_logo.svg

- SystemC: Open Source Event Based Simulator
  - IEEE 1666 standard
  - github: accellera-official/systemc
- QEMU: Open Source Emulator/Virtualiser
  - Covers many architectures (ARM, Hexagon, Xtensa, RiscV...)
  - github: qemu/qemu
- Qualcomm's Open Source SystemC library
  - Contains synchronisation, configuration, basic components (registers, routers, memory...)
  - Github: quic/qbox

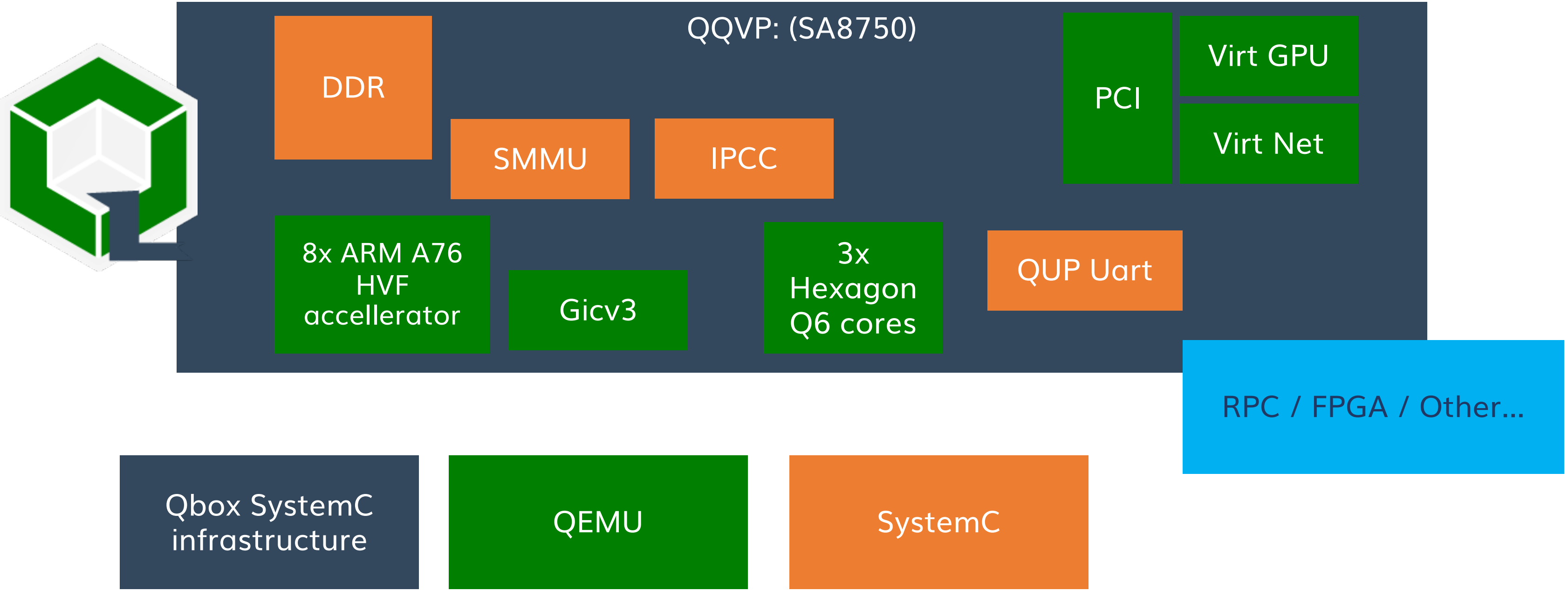
Qualcomm virtual platforms



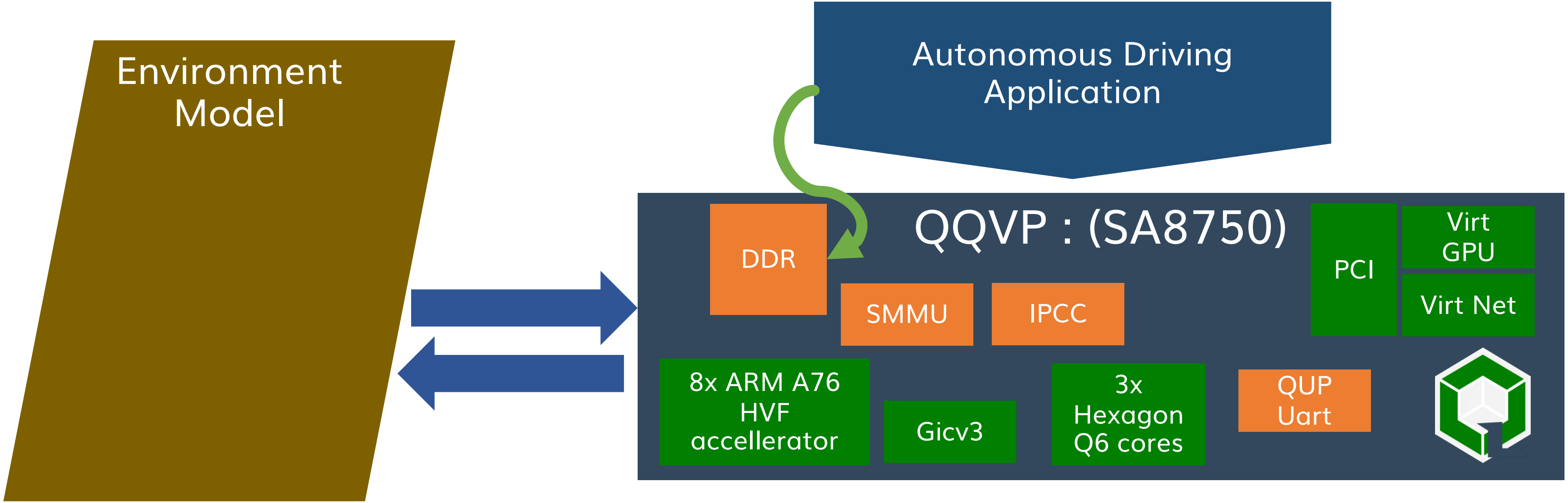
Qualcomm virtual platforms



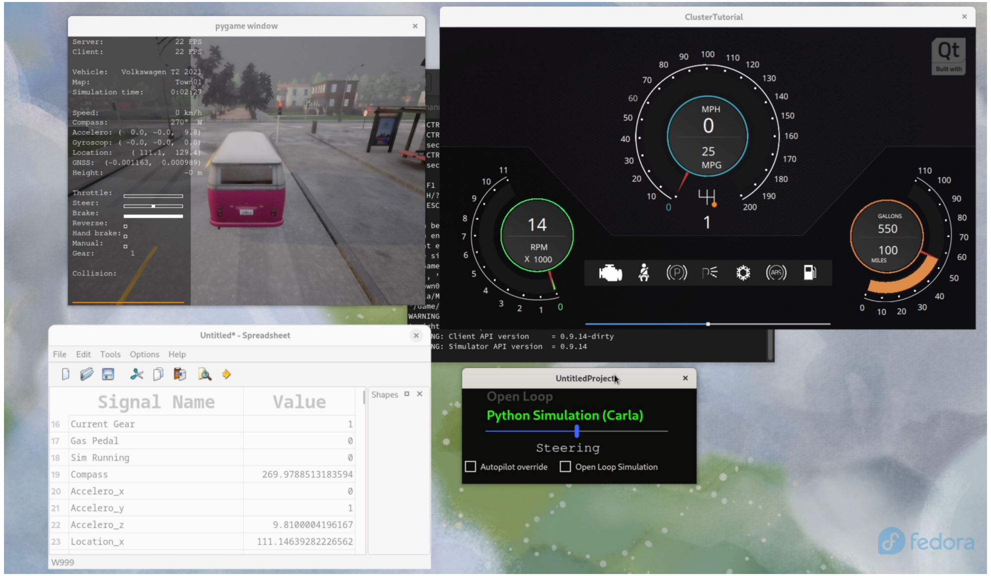
QQVP components



Car simulation: environment and application



Carla example



```
Server:                22 FPS
Client:                22 FPS

Vehicle:   Volkswagen T2 2021
Map:                   Town01
Simulation time:      0:02:27

Speed:                 0 km/h
Compass:             270°   W
Accelero: (  0.0, -0.0,  9.8)
Gyroscop: ( -0.0, -0.0,  0.0)
Location:     ( 111.1,  129.4)
GNSS:  (-0.001163,  0.000989)
Height:                  -0 m

Throttle:
Steer:
Brake:
Reverse:    □
Hand brake: □
Manual:     □
Gear:        1

Collision:
```

```
NG: Client API version     = 0.9.14-dirty
NG: Simulator API version  = 0.9.14
```

Untitled* - Spreadsheet

File Edit Tools Options Help

	Signal Name	Value
16	Current Gear	1
17	Gas Pedal	0
18	Sim Running	0
19	Compass	269.9788513183594
20	Accelero_x	0
21	Accelero_y	1
22	Accelero_z	9.8100004196167
23	Location_x	111.14639282226562

W999

UntitledProject

Open Loop

Python Simulation (Carla)

Steering

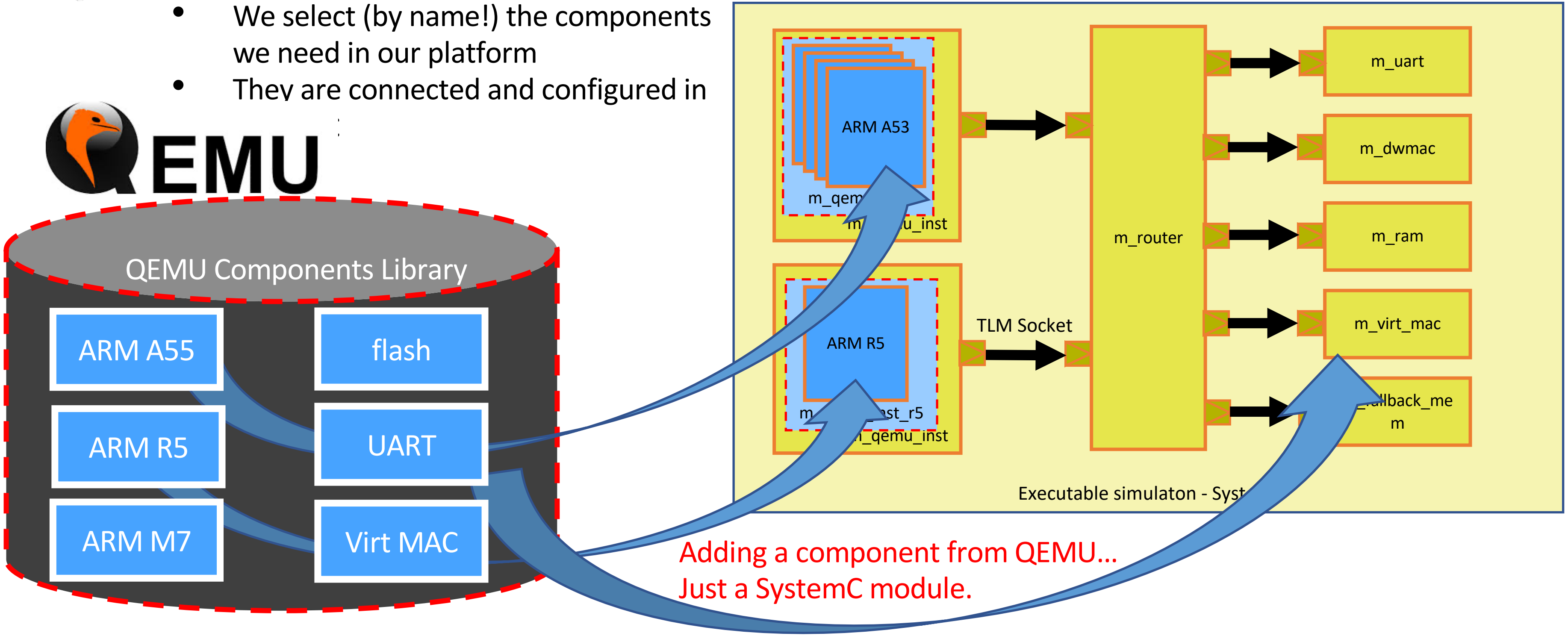
□ Autopilot override □ Open Loop Simulation

- github: carla-simulator/carla
- Open source autonomous driving system
  - Development
  - Training
  - Validation
- Based on Unreal game engine
- Provides digital assets (road layouts etc)
- Provides an ‘example’ application

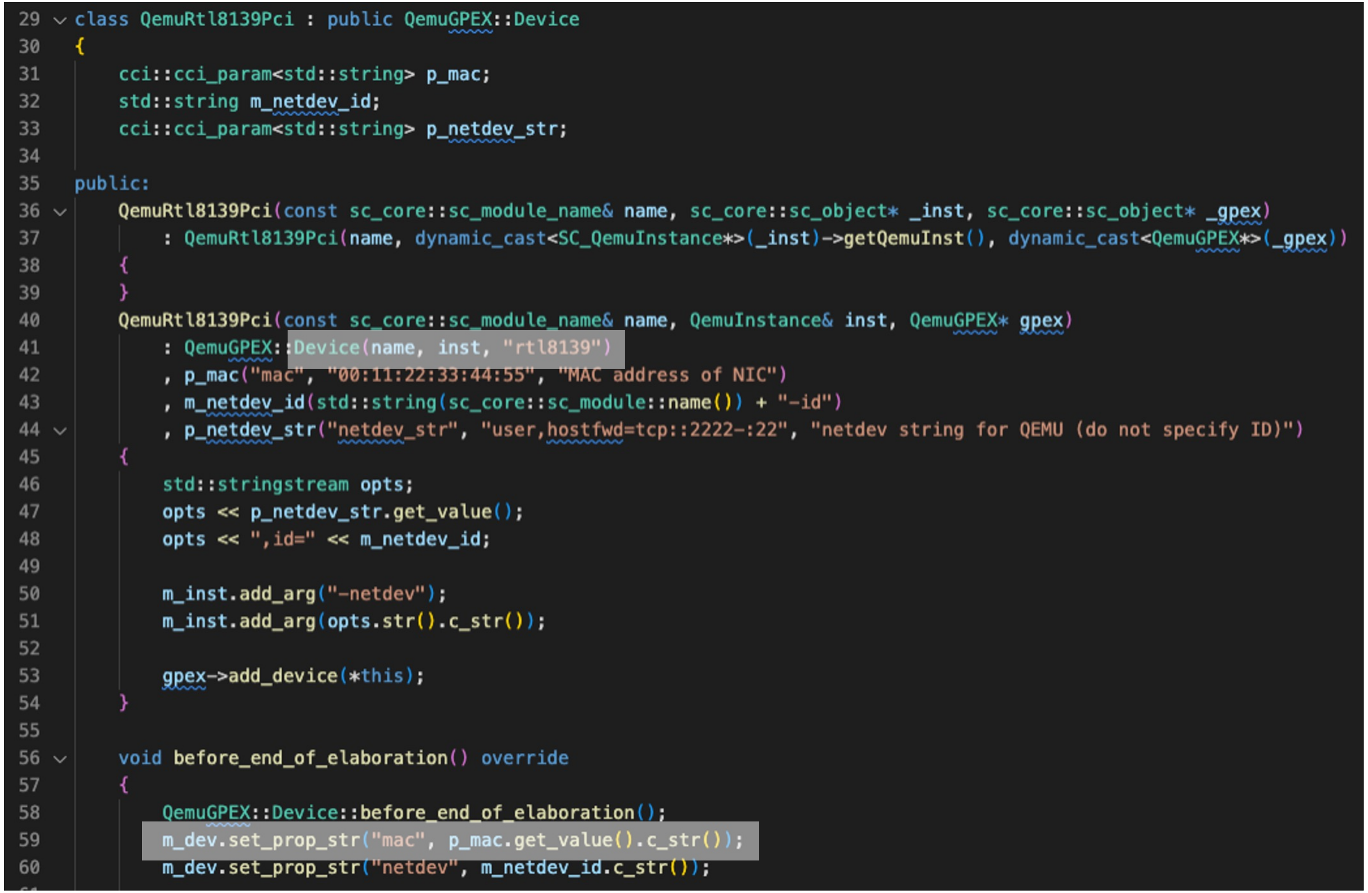
DEMO

Qbox under the hood

- QEMU is a ‘library’ of components
- We select (by name!) the components we need in our platform
- They are connected and configured in

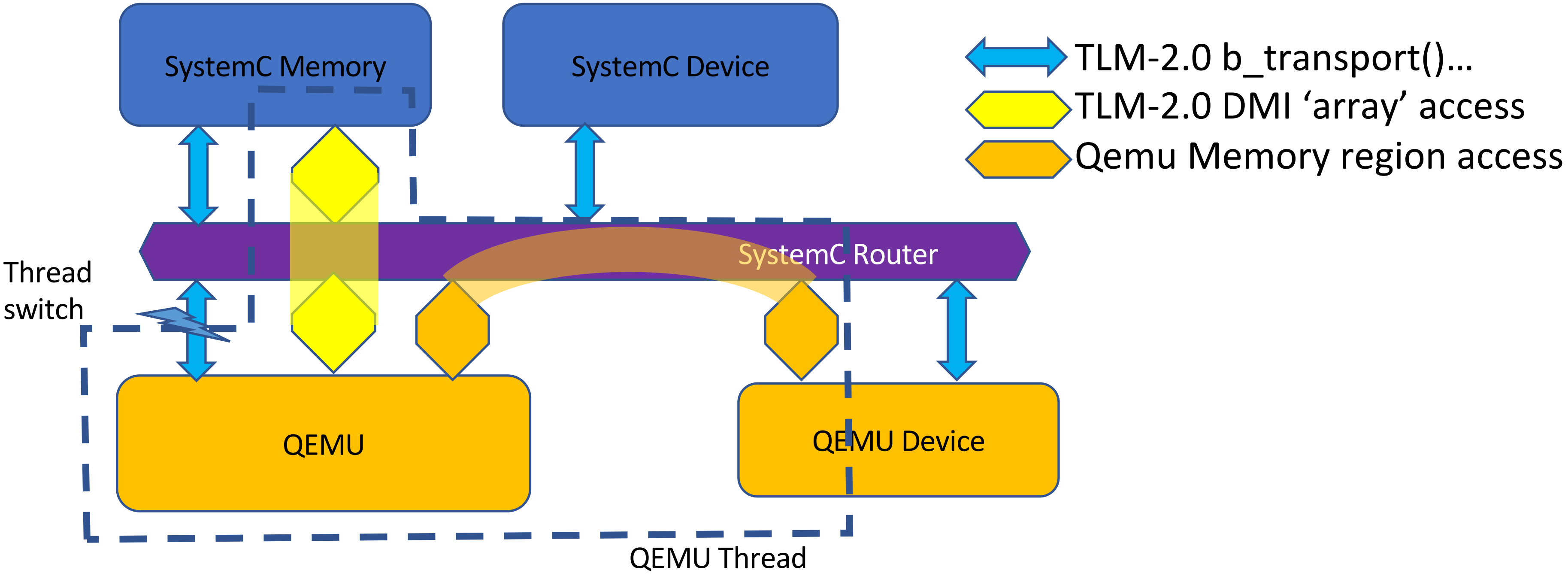


Adding a component from QEMU…
Just a SystemC module.



```cpp
class QemuRtl8139Pci : public QemuGPEX::Device
{
    cci::cci_param<std::string> p_mac;
    std::string m_netdev_id;
    cci::cci_param<std::string> p_netdev_str;

public:
    QemuRtl8139Pci(const sc_core::sc_module_name& name, sc_core::sc_object* _inst, sc_core::sc_object* _gpex)
        : QemuRtl8139Pci(name, dynamic_cast<SC_QemuInstance*>(_inst)->getQemuInst(), dynamic_cast<QemuGPEX*>(_gpex))
    {
    }
    QemuRtl8139Pci(const sc_core::sc_module_name& name, QemuInstance& inst, QemuGPEX* gpex)
        : QemuGPEX::Device(name, inst, "rtl8139")
        , p_mac("mac", "00:11:22:33:44:55", "MAC address of NIC")
        , m_netdev_id(std::string(sc_core::sc_module::name()) + "-id")
        , p_netdev_str("netdev_str", "user,hostfwd=tcp::2222-:22", "netdev string for QEMU (do not specify ID)")
    {
        std::stringstream opts;
        opts << p_netdev_str.get_value();
        opts << ",id=" << m_netdev_id;

        m_inst.add_arg("-netdev");
        m_inst.add_arg(opts.str().c_str());

        gpex->add_device(*this);
    }

    void before_end_of_elaboration() override
    {
        QemuGPEX::Device::before_end_of_elaboration();
        m_dev.set_prop_str("mac", p_mac.get_value().c_str());
        m_dev.set_prop_str("netdev", m_netdev_id.c_str());
```

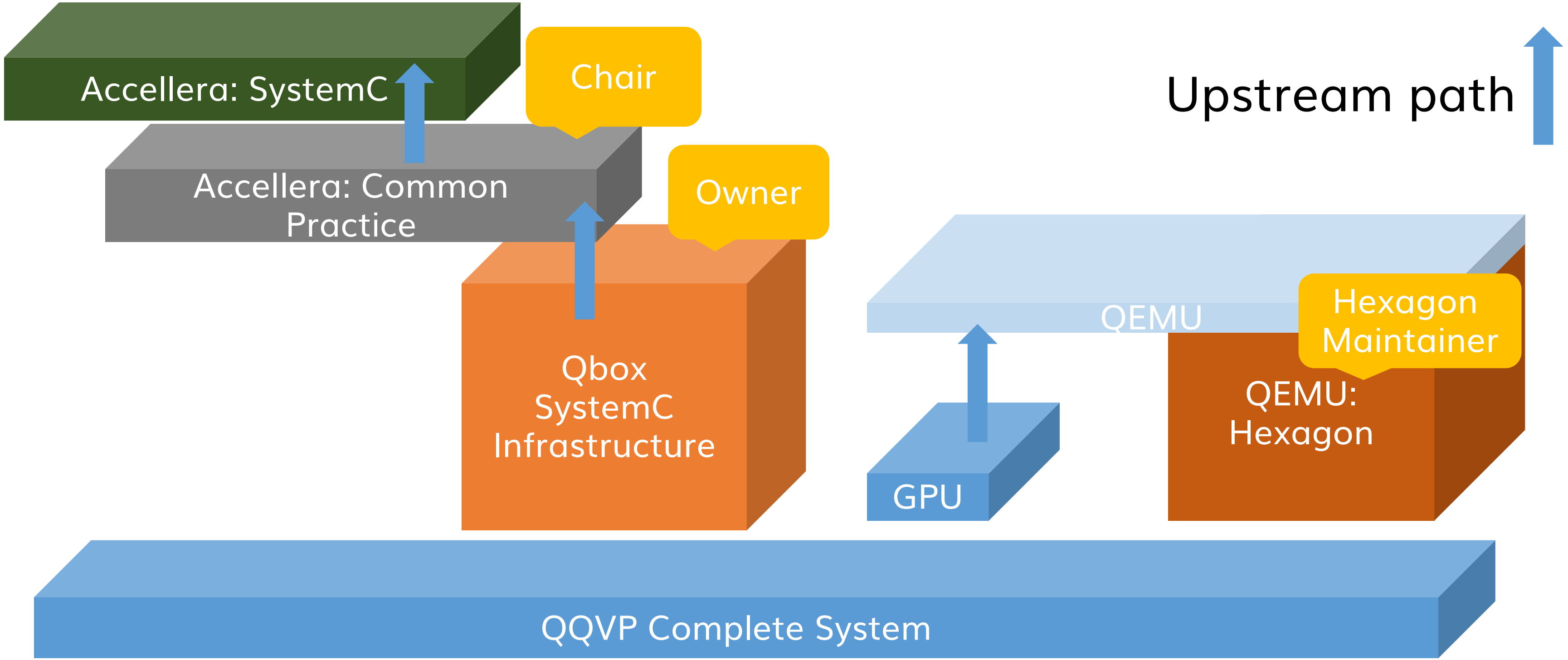
Partitioning between QEMU and SystemC



Advantages of open source for customers

- Familiarity : People have used open-source tools before, they are well known.
- Quality : Many eyes on the code, means the quality is very high. Many members of the community, many use cases are covered.
- Flexibility : With the source code, customers re-configure, add features, fix issues…
- Security : With the source code, you can always maintain your version yourself.

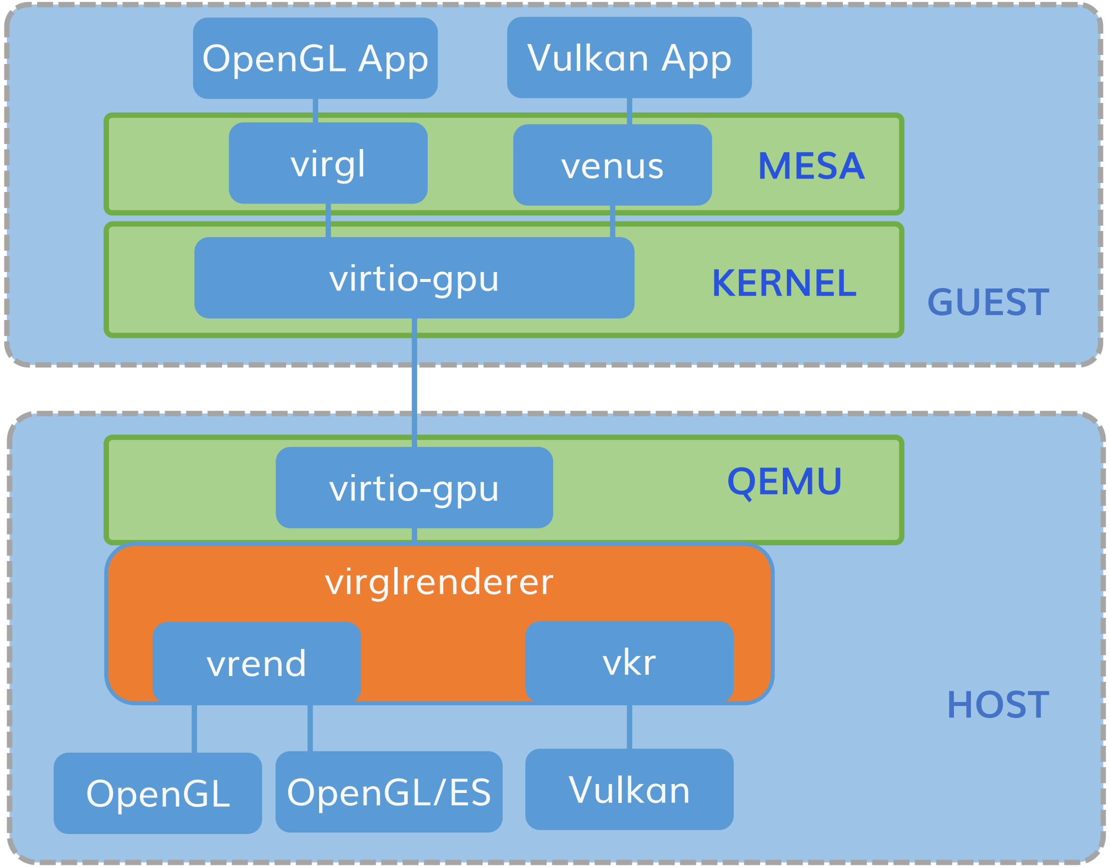
QQVP is built on open source projects



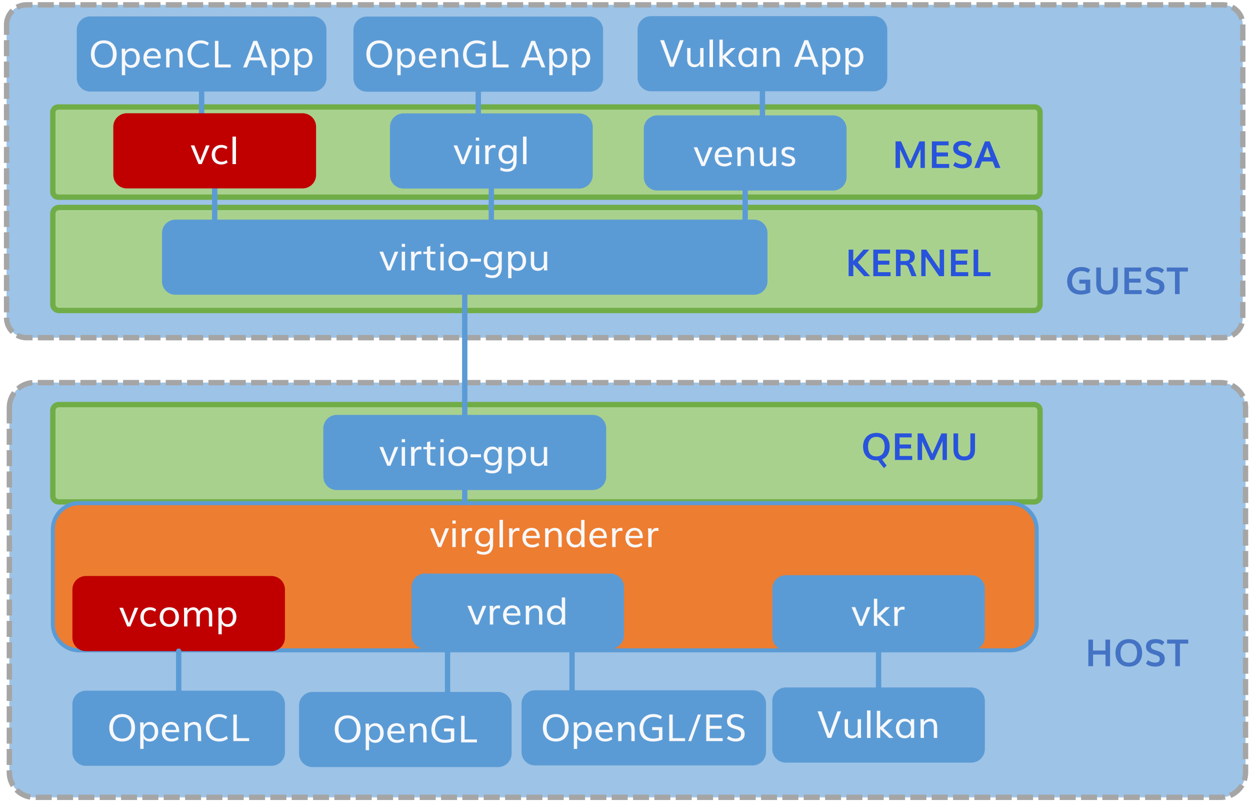
Open CL

- Issues with current situation . . .

VirtIO-GPU


OpenGL App
Vulkan App
virgl
venus
MESA
virtio-gpu
KERNEL
GUEST
virtio-gpu
QEMU
virglrenderer
vrend
vkr
HOST
OpenGL
OpenGL/ES
Vulkan

OpenCL



We are planning to contribute to both MESA and Virglrenderer

Linaro Connect

MADRID 2024 | MAY 12-17 2024

Thank you