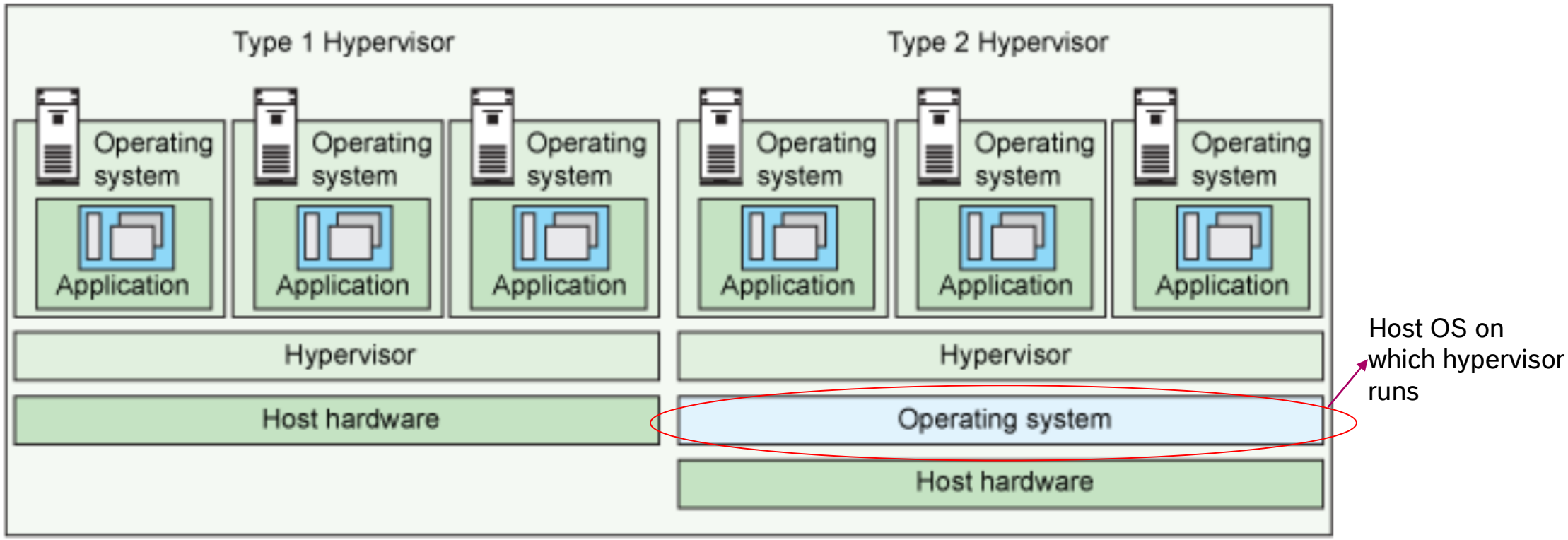
INTRODUCTION TO HYPERVISOR

Virtualization Basic
Type of Hypervisor

- Type 1 hypervisor: hypervisors run directly on the system hardware – A “bare metal” embedded hypervisor
  - QNX (QNX)
  - Multivisor (Greenhills)
  - Coqos (OpenSynergy)
- Type 2 hypervisor: hypervisors run on a host operating system that provides virtualization services, such as I/O device support and memory management.
  - VMware Server
  - Oracle VM VirtualBox
  - KVM

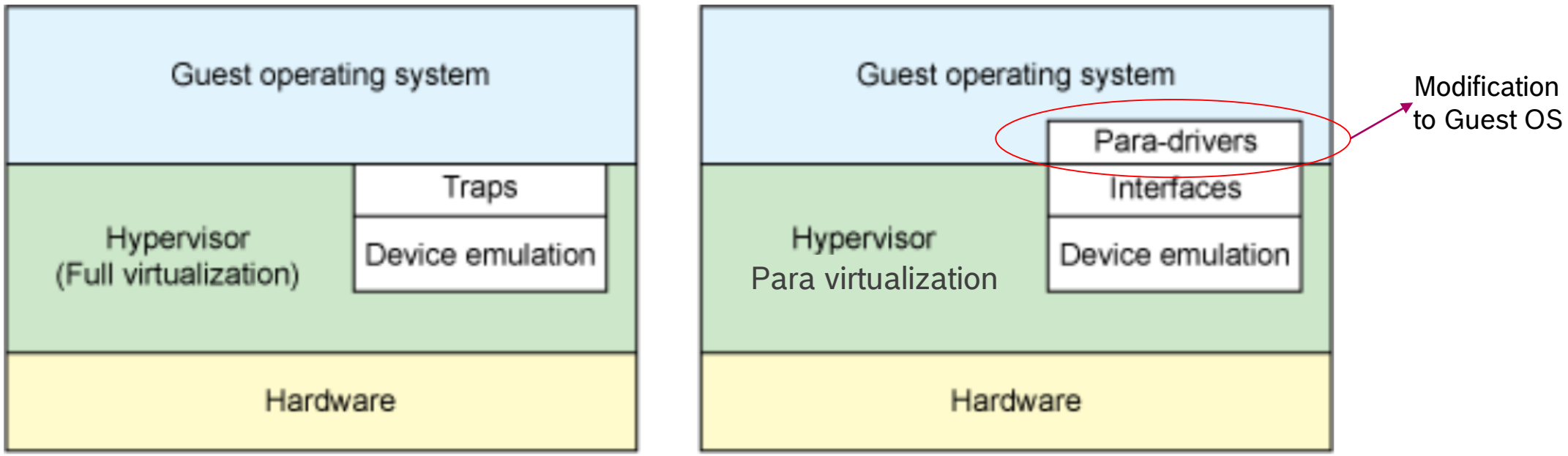
Virtualization Basic
Type of Hypervisor



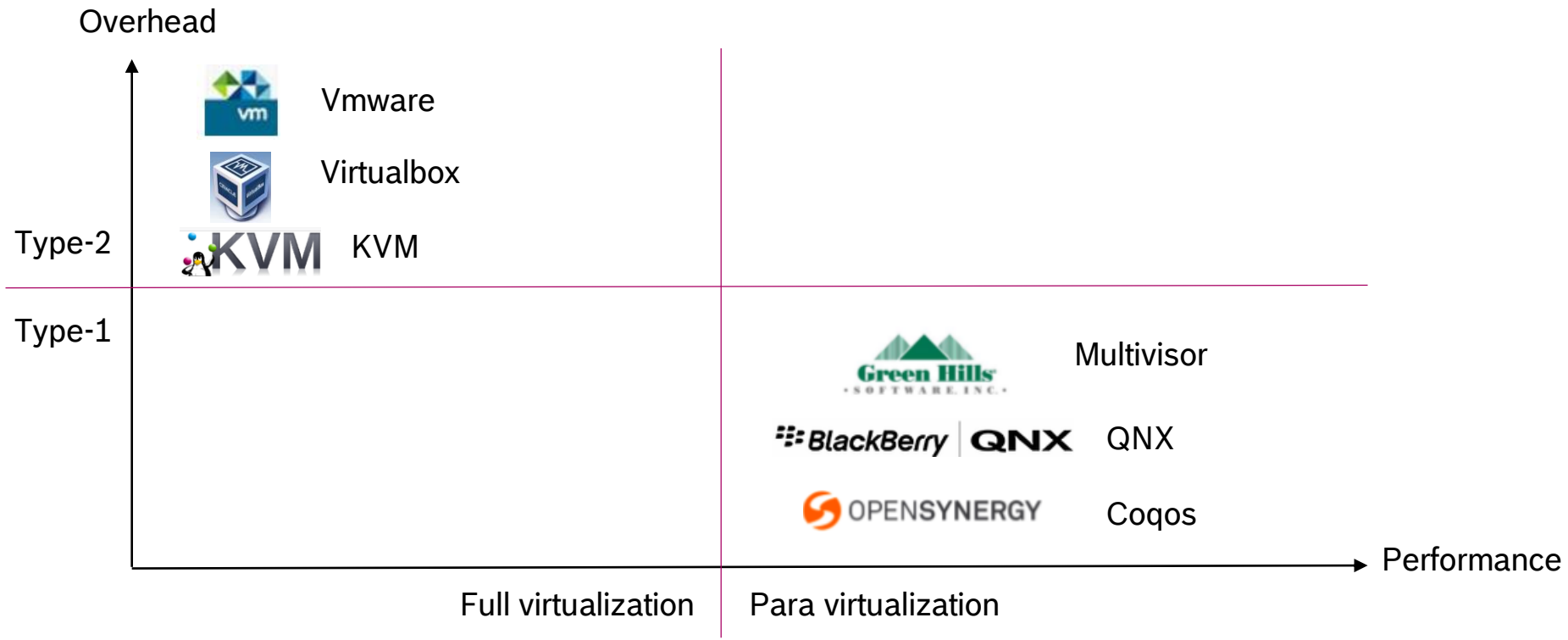
Virtualization Basic
Fullvirtualization vs. Paravirtualization

- In *paravirtualization*, the guest operating system is not only aware that it is running on a hypervisor but includes code to make guest-to-hypervisor transitions more efficient
  - QNX (QNX)
  - Multivisor (Greenhills)
  - Coqos (OpenSynergy)
- In *full virtualization*, the guest operating system runs on top of a hypervisor that sits on the bare metal. The guest is unaware that it is being virtualized and requires no changes to work in this configuration
  - VMware Server
  - Oracle VM VirtualBox
  - KVM

Virtualization Basic
Fullvirtualization vs. Paravirtualization



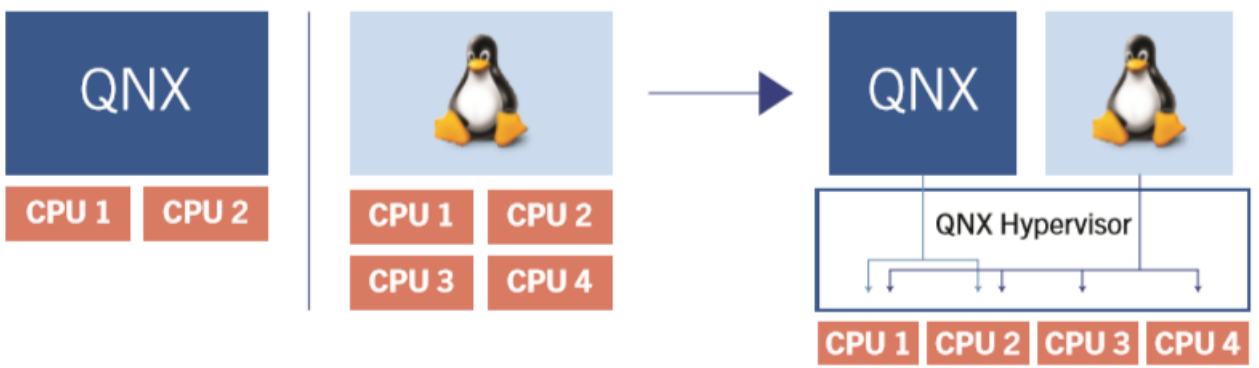
Virtualization Basic
Comparison and Conclusion



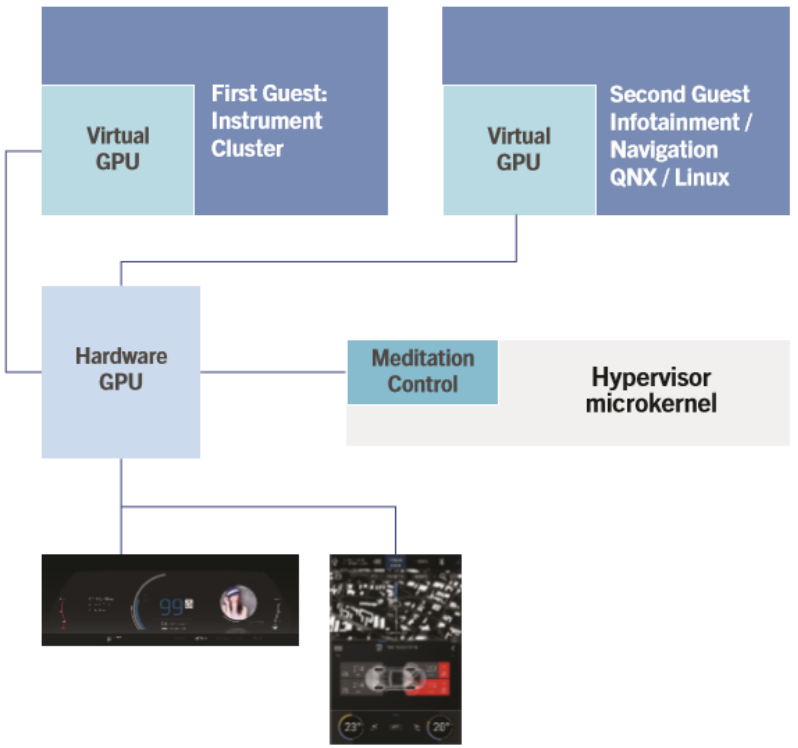
Device Handling
Sharing by HW Virtualization

- Single physical device is shared by multi-guest OSes
- The physical device support HW virtualization
- Hypervisor control the sharing of device by taking advantage of HW capacity

CPU sharing

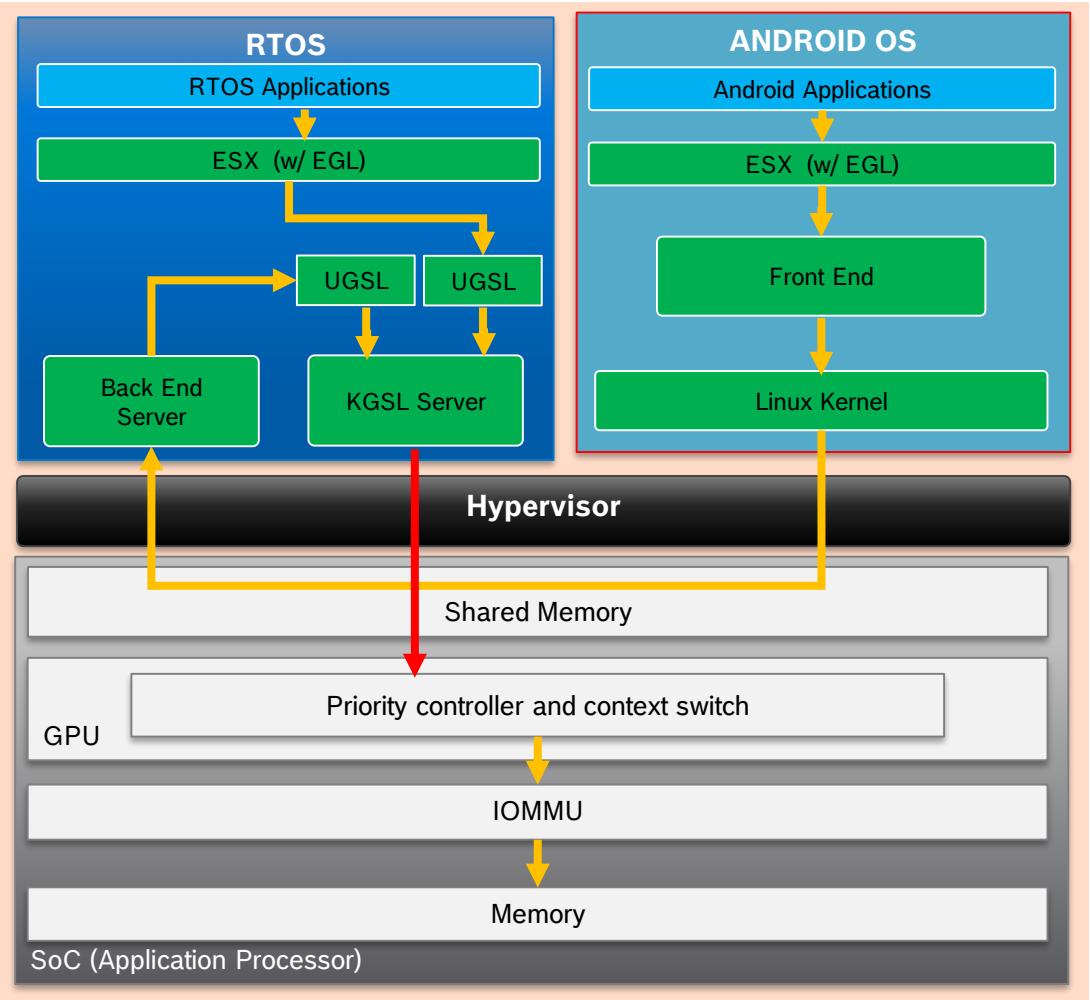


GPU sharing



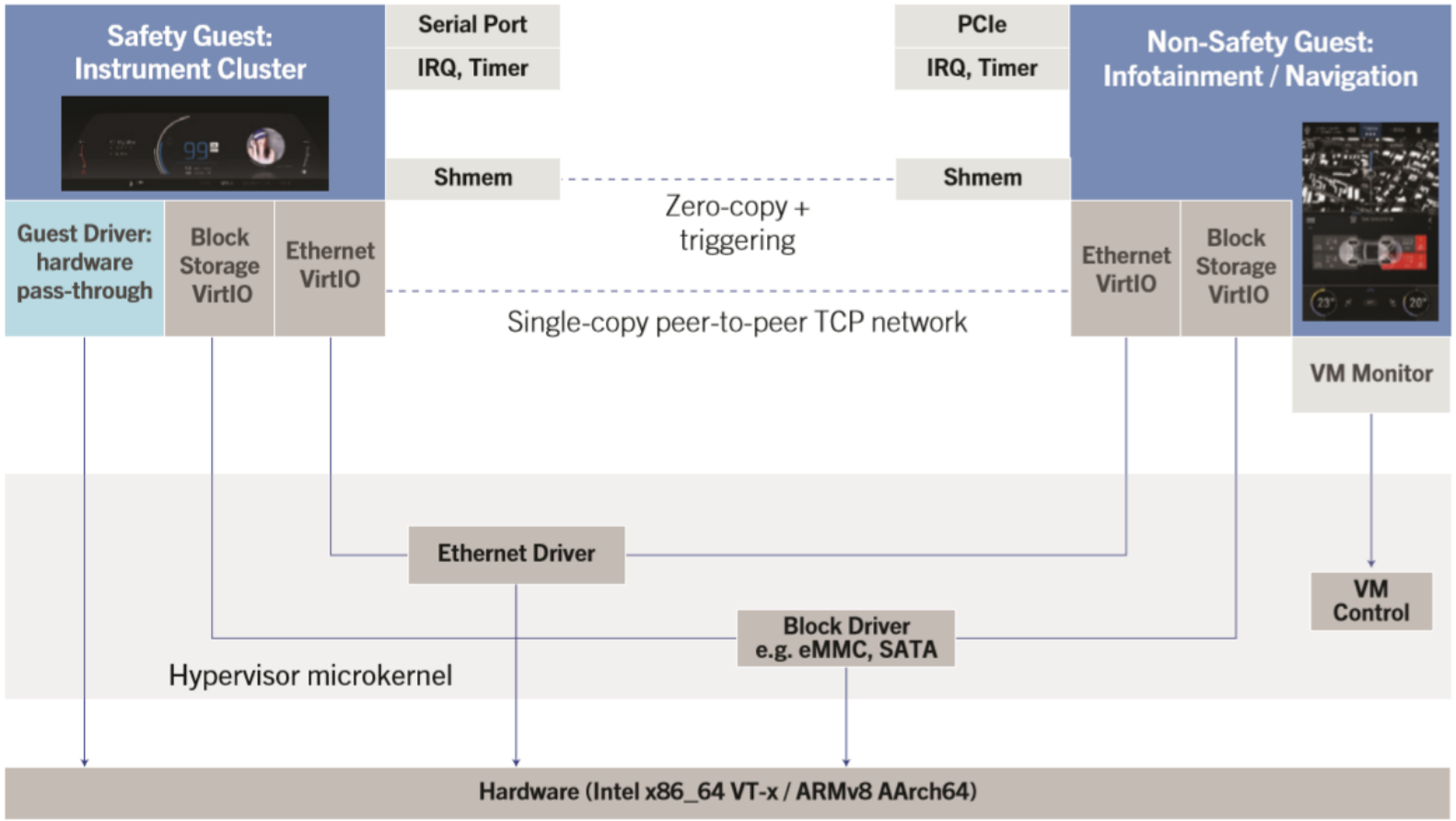
Device Handling
Sharing by Front-End/Back-End

- Graphic commands from Android OS are send by the FE (Front End) to BE server (Back End)
  - Shared memory is used to store Graphic command chunk (no VM context switch needed ⇨ no performance overhead)
  - Events used to draw prepared chunk. (VM context switch => performance overhead)
  - Modified GLES/EGL is provided by Qualcomm
- KGSL server receives the Graphic commands and sends on behalf of other GPU contexts to GPU core.
- UGSL - User Graphic Shared Library
- KGSL- Kernel Graphic Shared Library
- ESX – Enhanced Shared Graphix



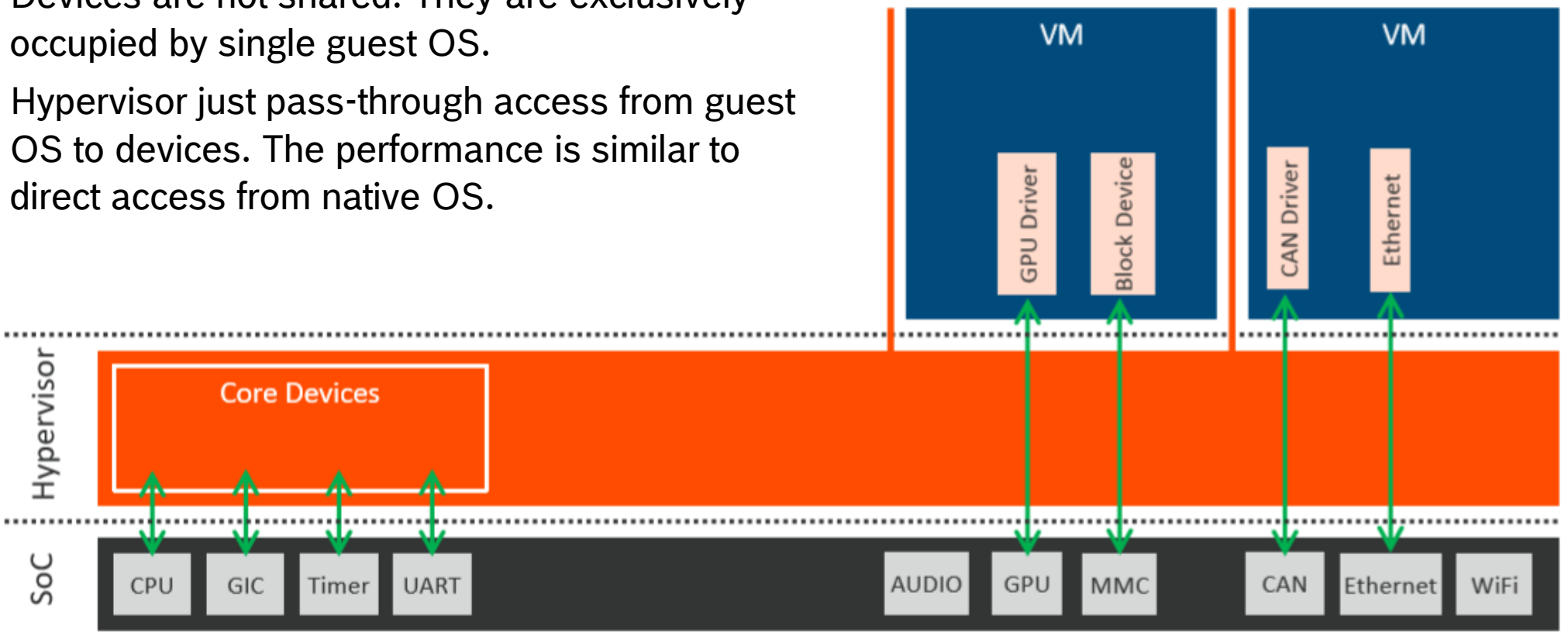
Device Handling
Sharing by VirtIO (Special type of Front End/Back End)

- Single physical device is shared by multi-guest OSes
- The physical device doesn't support HW virtualization
- Hypervisor simulates virtual device that can be shared by mulit-guest OSes with virtIO technology
- The virtual device can be emulated in hypervisor, in dedicated guest OS, or in specified guest OS



Device Handling
Pass-through

- Devices are not shared. They are exclusively occupied by single guest OS.
- Hypervisor just pass-through access from guest OS to devices. The performance is similar to direct access from native OS.

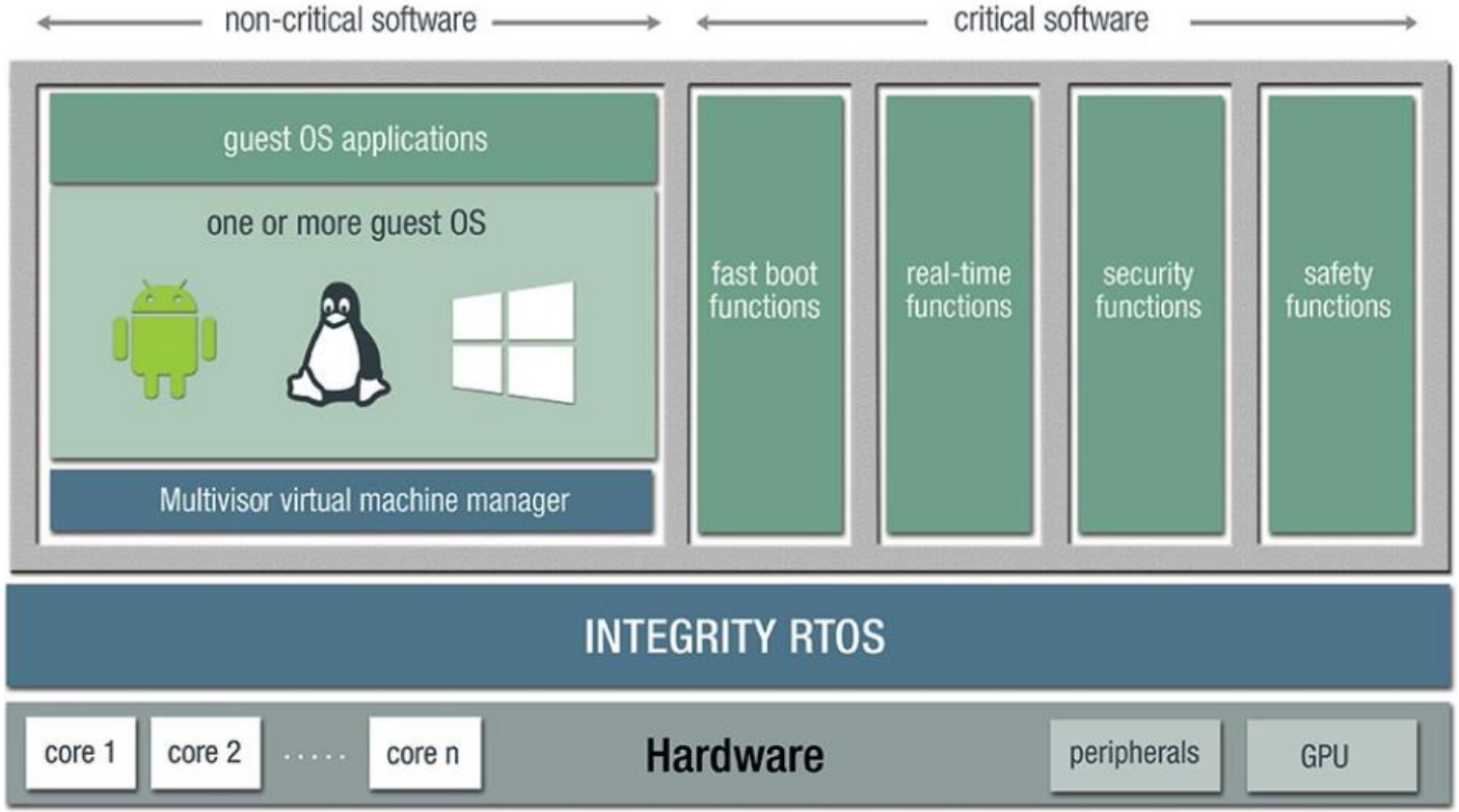


Device Handling
Comparison and Conclusion

	Performance	HW Support	Sharable	Hypervisor
HW Virtualization	☆☆☆☆	required	yes	QNX, Multivisor, Coqos
Fron tEnd/Back End	☆☆☆	no	yes	QNX, Multivisor, Coqos
VirtIO	☆☆☆	no	yes	QNX, Multivisor, Coqos
Pass-through	☆☆☆☆☆	no	no	QNX, Multivisor, Coqos

Guest RTOS
Hypervisor as RTOS: QNX and Multivisor

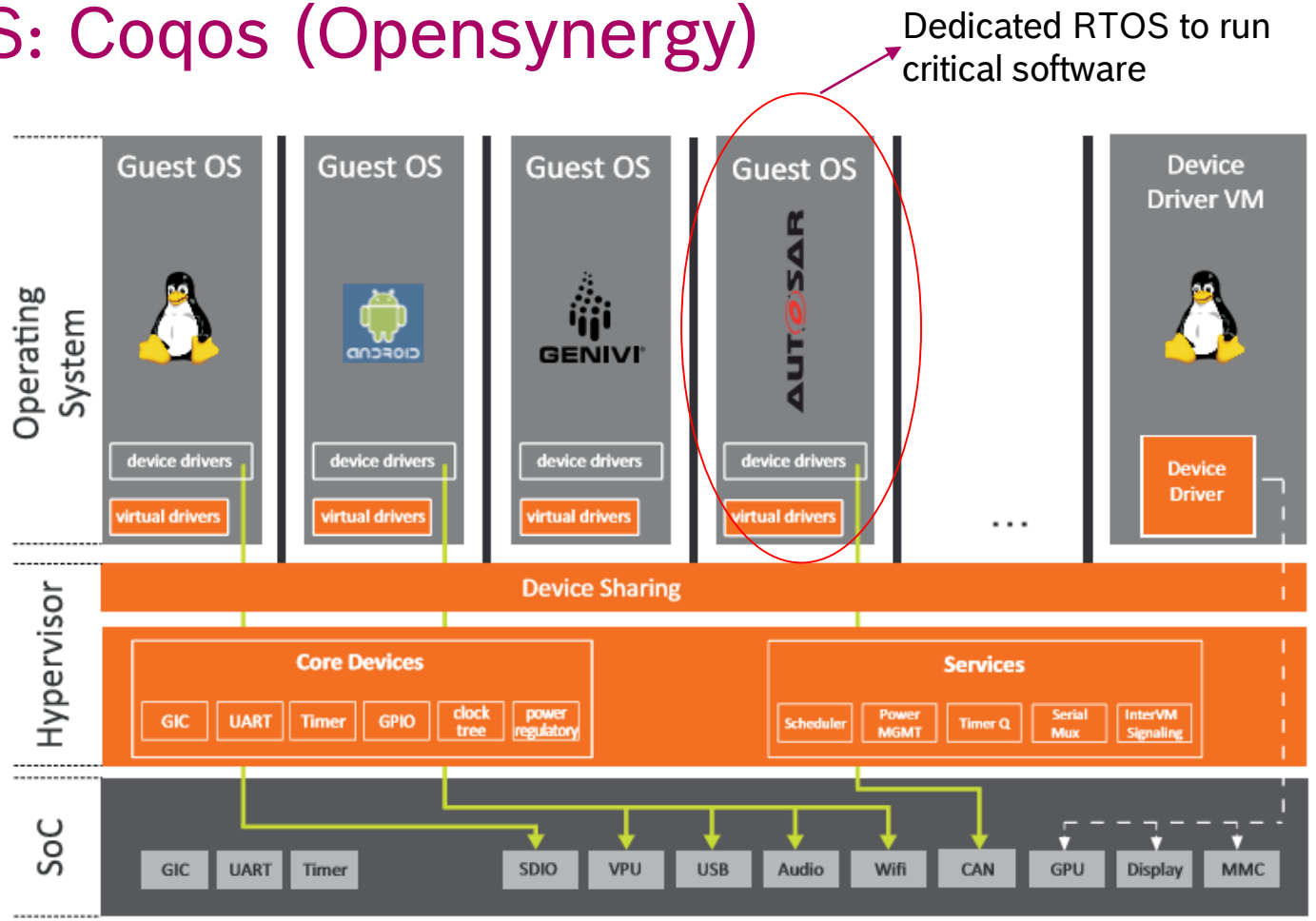
- The hypervisor itself is a RTOS
- It combines general purpose guest OS with critical software running at the RTOS
- No extra guest OS is required to run the critical software



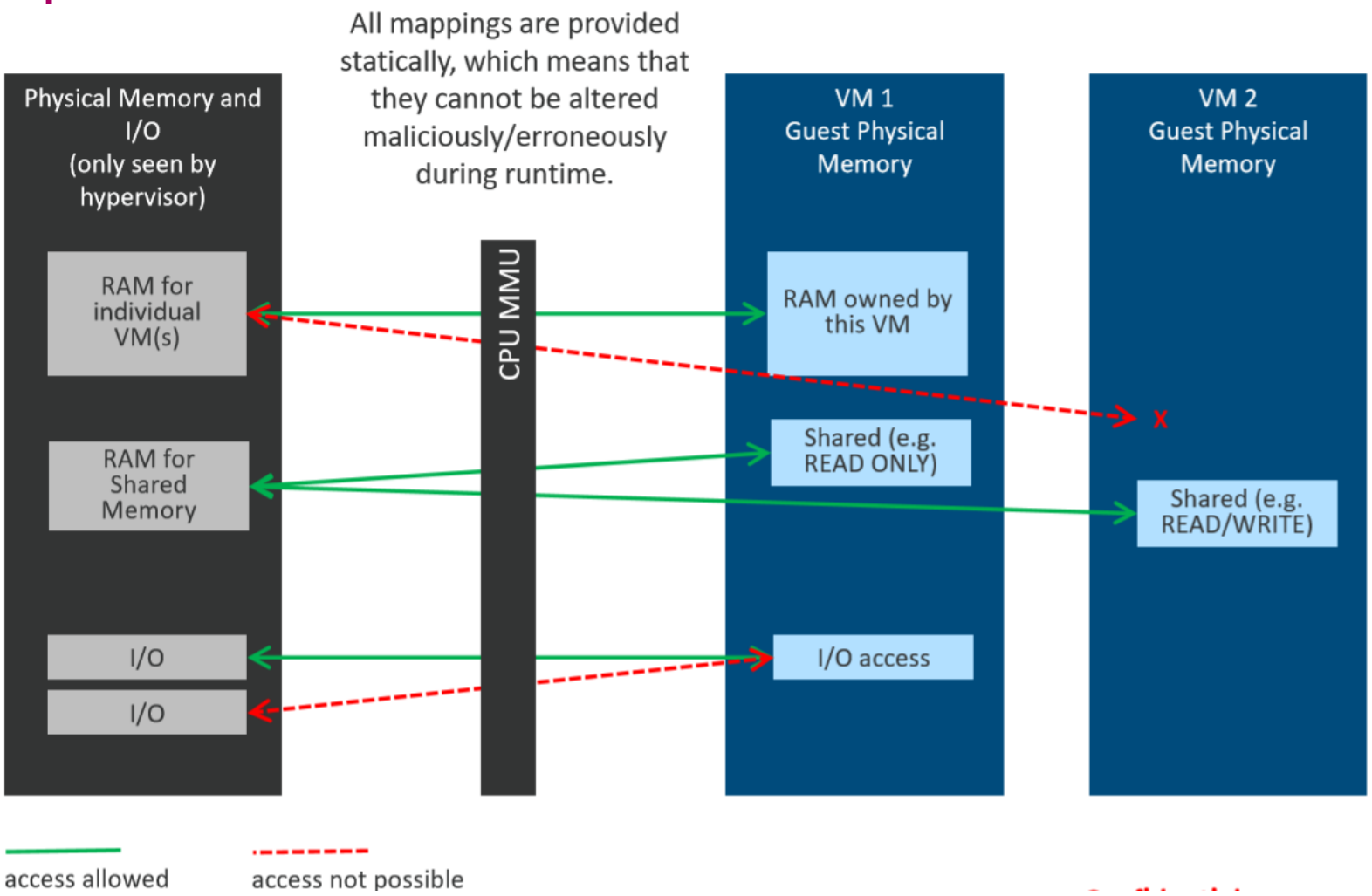
Guest RTOS

Hypervisor without RTOS: Coqos (Opensynergy)

- Hypervisor is only responsible for virtualization; no RTOS is provided
- All functions runs in guest OS. This means to run critical software, an extra guest RTOS is required.
- Coqos doesn't support Integrity OS and QNX. But there are other RTOS running as guest OS for critical functions.

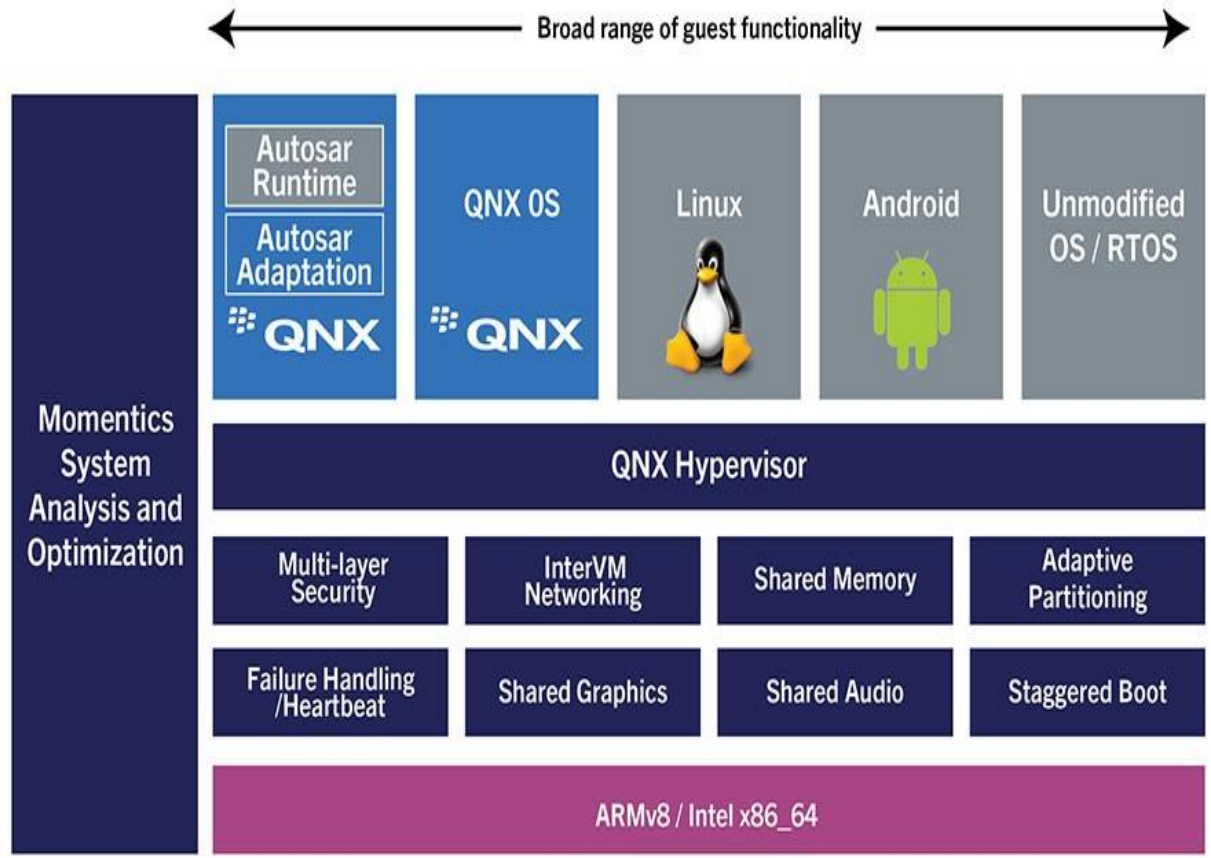


Virtualization Security
Memory Separation



QNX
Features

- Type 1 Hypervisor
- Safety certification pedigree
- Virtual CPU model
- Pin to cores or share cores based on priority
- Adaptive partitioning. Allows for CPU guarantees of guest runtime
- 64-bit and 32-bit guests: QNX, Linux, Android, RTOS
- Shared memory with triggering
- VirtIO (1.0) device sharing
- TAP(Test Access Point) and peer-to-peer networking with bridging
- Failure detection and restart of guests
- Virtual watchdog for guest integrity checking
- Low overhead (typical < 2%)
- Graphical tools for analysis and debug



QNX
Support Platform

- The QNX Hypervisor makes full use of all virtualization capabilities offered by the hardware
- Support X86_64 and Arm64
- Platform: Qualcomm 820/8155/61xx, Renesas H3/M3, Intel Apollo Lake, NXP i.MX8QM
- If we have new SoCs requirements, we may discuss how to support them case by case with QNX

Hypervisor
Conclusion

- Similarity of QNX, Multivisor and Coqos
  - Type-1
  - Paravirtualization
  - Small in footprint, high performance, very low overhead (< 2%-5%)
  - Similar technologies for device handling
  - ISO 26262 ASIL B or higher
- Difference
  - Multivisor and QNX is RTOS in nature; no dedicated guest OS for critical software
  - Coqos is a pure hypervisor; critical software is running in guest OS as other software in generic purpose guest OS
  - Multivisor doesn't support QNX; QNX doesn't support Integrity OS; Coqos doesn't support both
  - Other details such schedule policy, failure detection, SOC support…
- There is no clear criteria to decide which hypervisor has advantage other the others. It is up to individual project and tier1 to select the most suitable hypervisor.