KVM Platform Device Passthrough

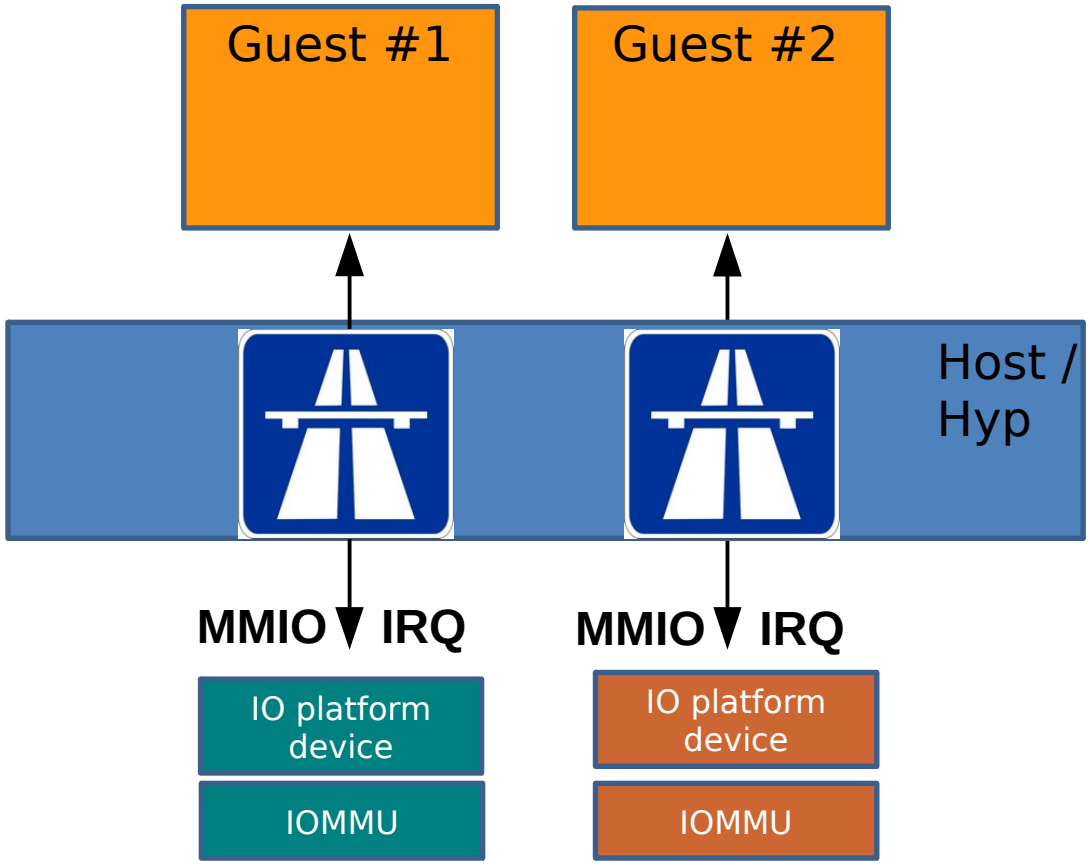
Eric Auger, Linaro

KVM Forum

Oct 14, 2014

Linaro

Goal: efficiently assign platform devices to KVM guests



Linaro

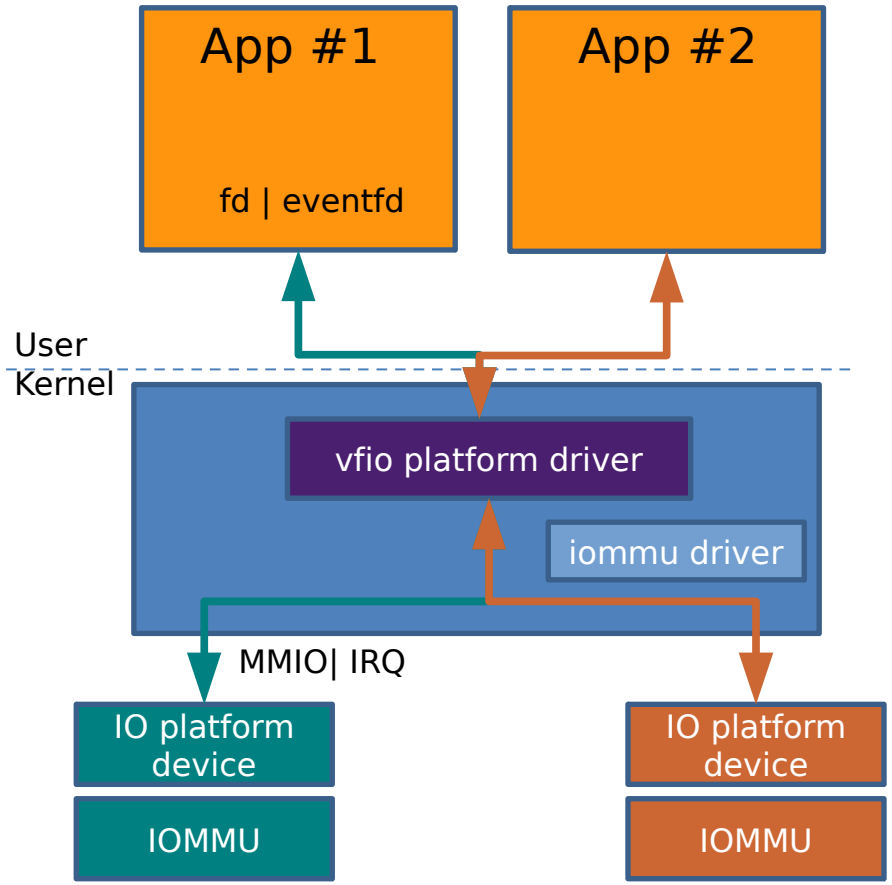
Agenda

- VFIO Framework
- Focus on IRQ assignment
  - Understand legacy frameworks
  - Why hardware-assisted IRQ forwarding is crucial?
- Forwarded IRQ Integration with KVM/VFIO
- Experimental Results

Linaro

VFIO Platform Driver

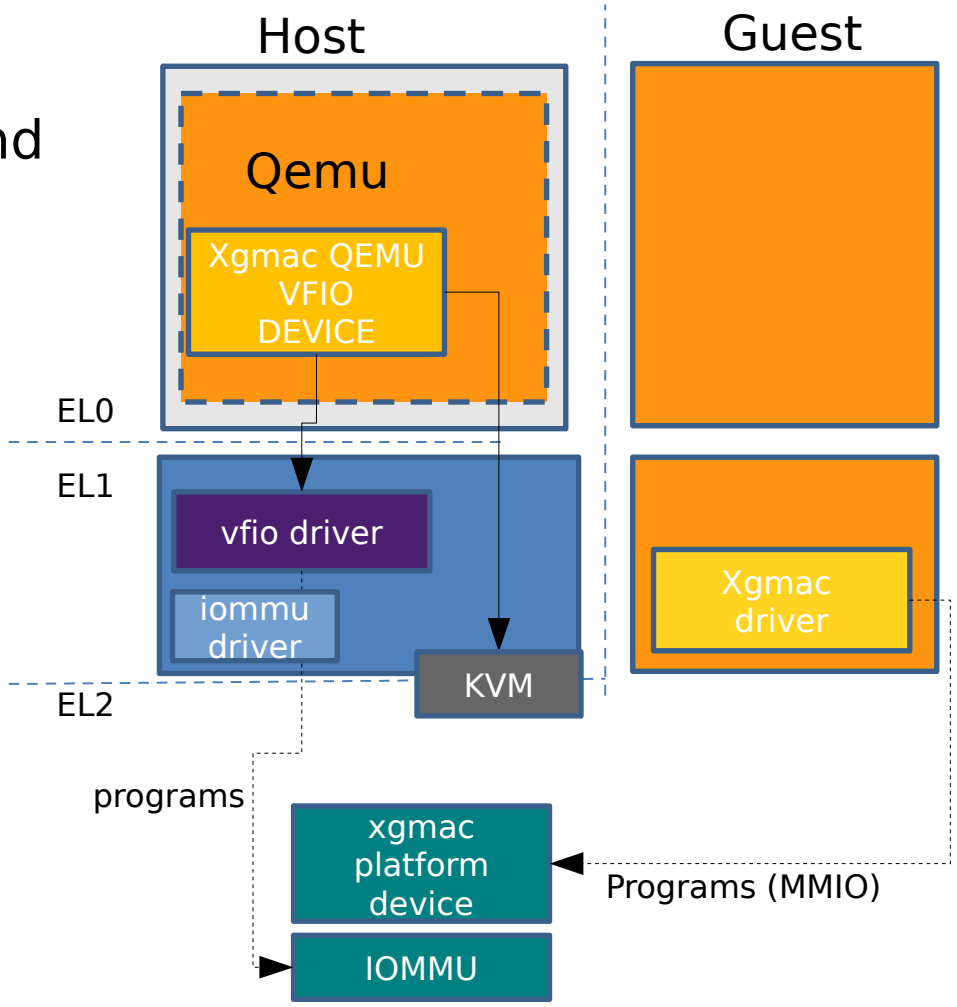
- allows user-side to
  - mmap device MMIO regions
  - route physical IRQ to eventfd
  - Dma map buffers on iommu



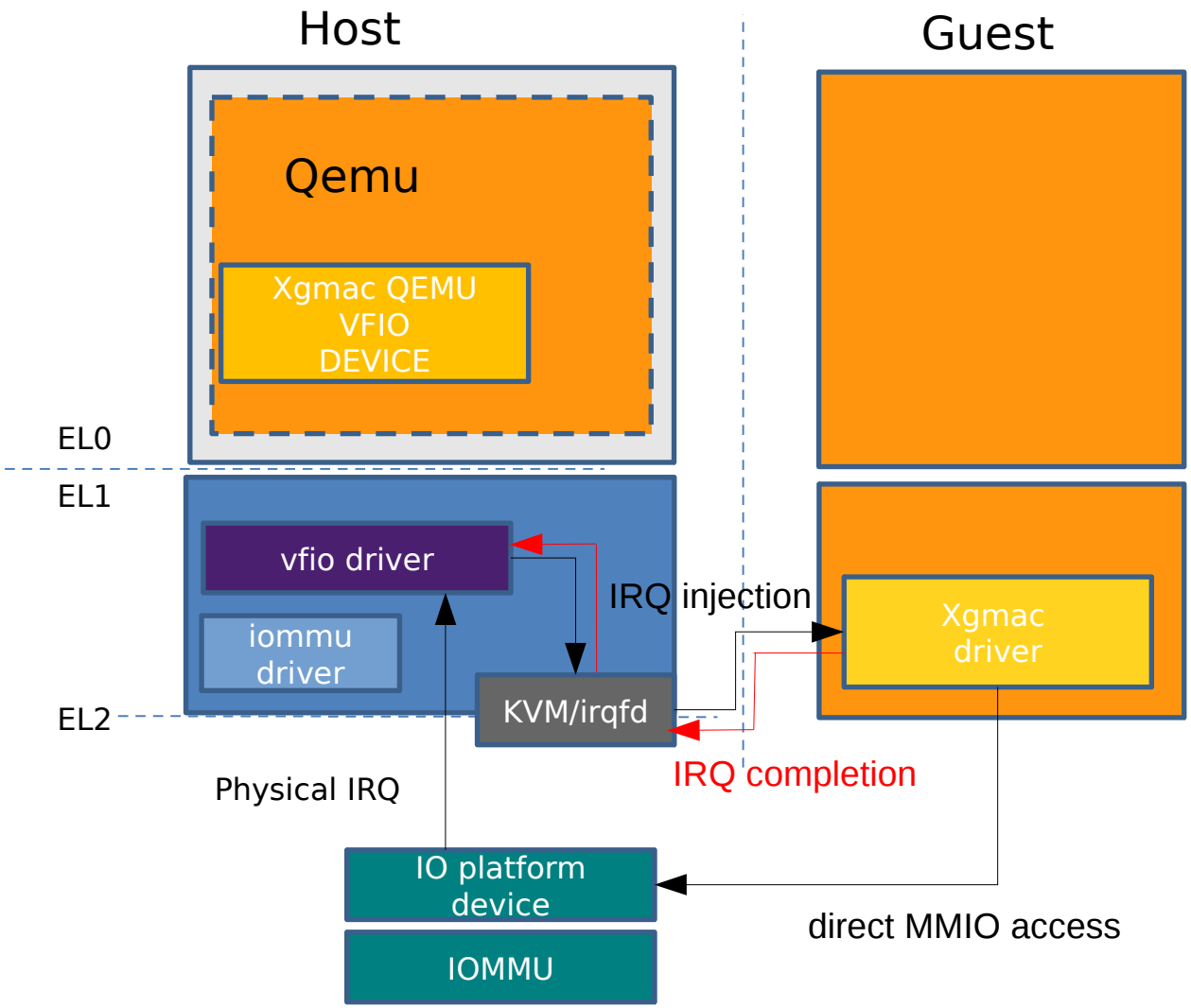
Linaro

QEMU VFIO device

- Setup routes between guest and assigned device
  - MMU
  - IOMMU
  - IRQ injection path
- Generate guest device device tree node

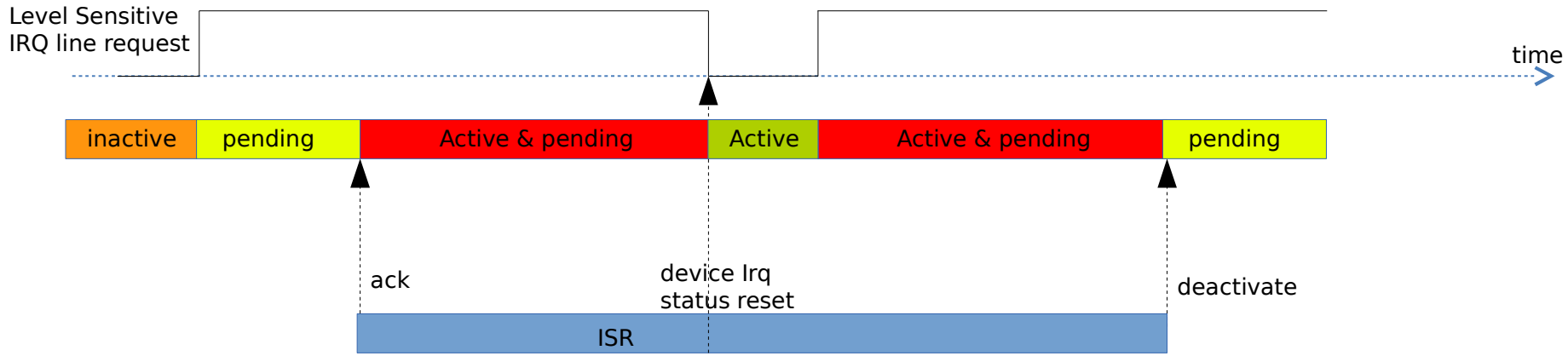


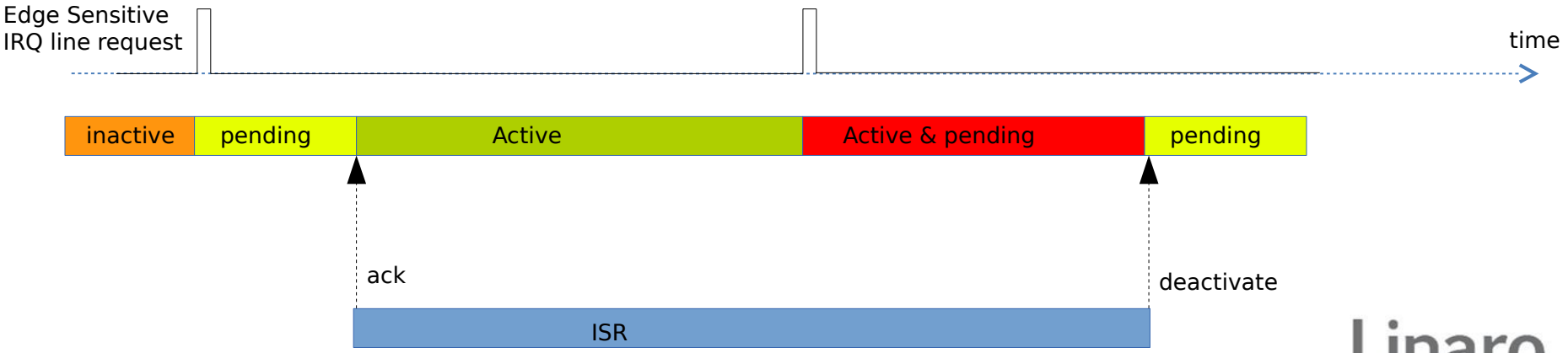
MMIO & IRQ Paths



Linaro

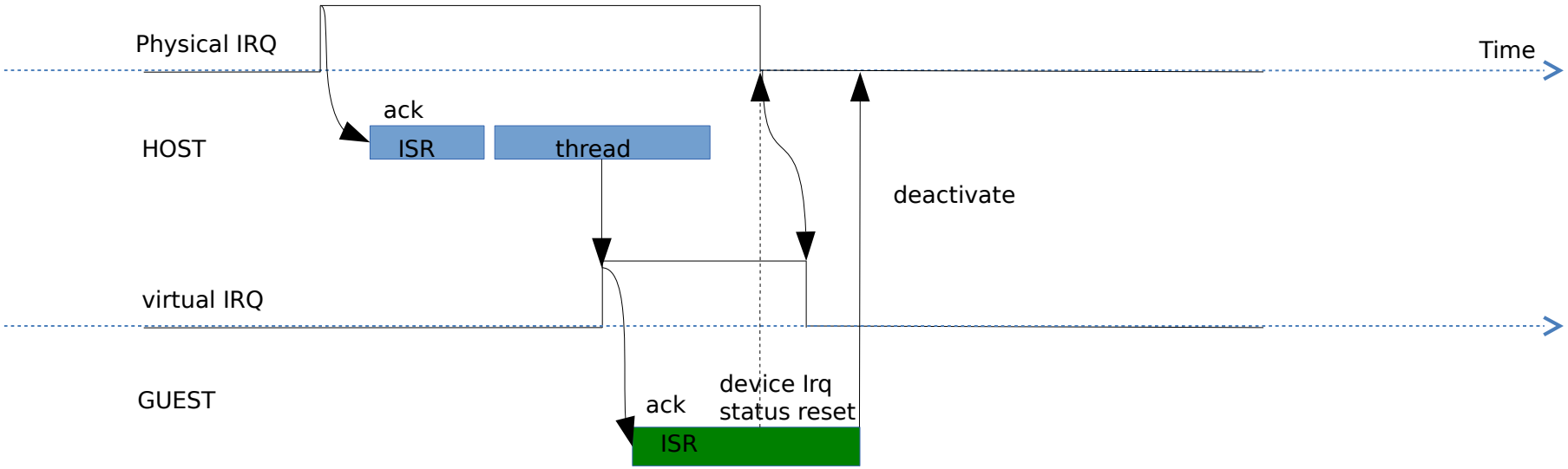
ARM IRQ Handling





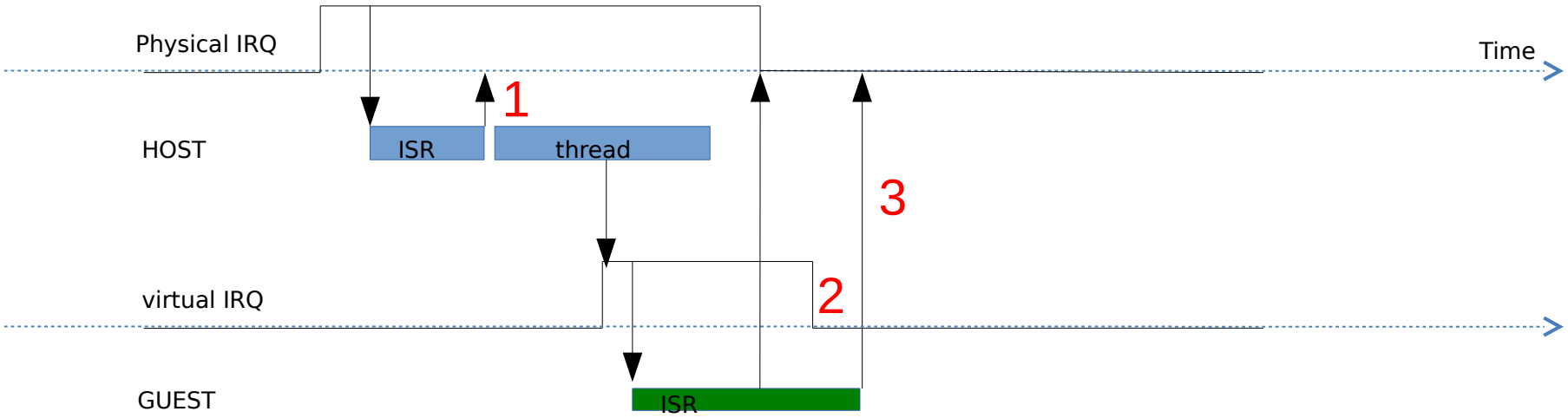
Linaro

Assigned Level Sensitive IRQ Model



Linaro

Level Sensitive IRQ Implementation Challenges



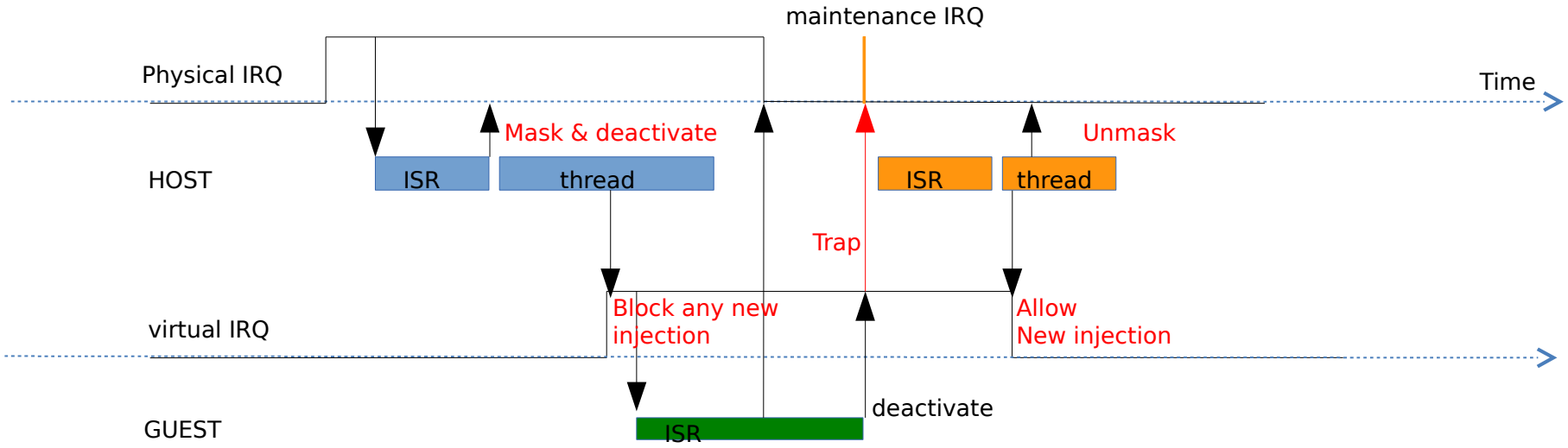
1)Physical IRQ completion

2)Virtual IRQ modeling

3)Virtual IRQ completion propagation

Linaro

Basic vfio/irqfd ARM porting



- VFIO Mask/unmask
- Trap on completion

Linaro

Performance Challenges on ARM

- 1 VM switch when injecting
- 1 VM switch when completing
- VM Switch really costly on ARM

- Goal: Propose a new method to save completion VM switch using ARM GIC virtualization features

Linaro

GIC Forwarding Feature

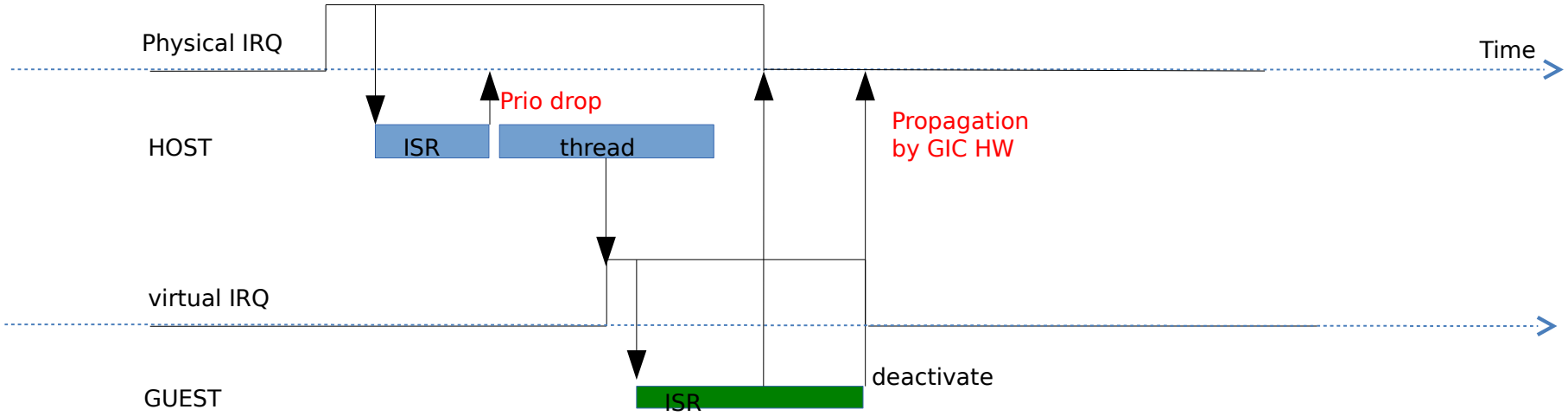
- GIC can automatically complete physical IRQ on virtual IRQ completion
- Host only drops the running priority of the CPU I/F to allow other physical IRQs to be signaled
- Same IRQ cannot be signaled before its deactivation by GIC HW

Forwarded IRQ Patch

- “ARM: forwarding physical interrupts to a guest VM” from M. Zyngier
  - Enable mode where priority drop and deactivate are separated, Linux wide
    - Current used mode is simultaneous prio drop & deactivate
  - Provides separate operations to program IRQ forwarding at
    - IRQCHIP
    - VGIC

Linaro

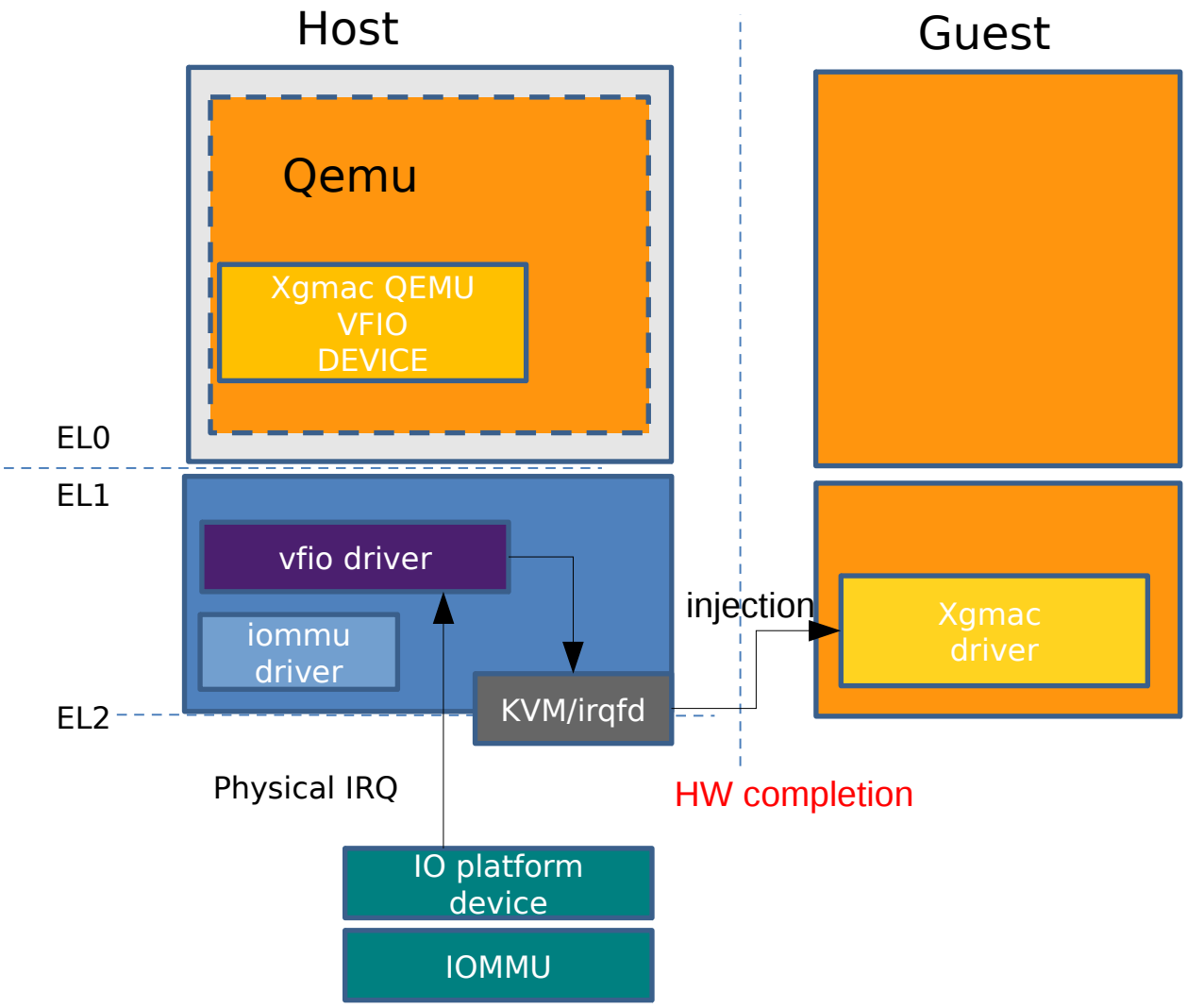
vfio/irqfd/forward



- No mask/unmask anymore
- Guest completion propagated by GIC HW
- No VM switch at completion
- Natural and optimized implementation

Linaro

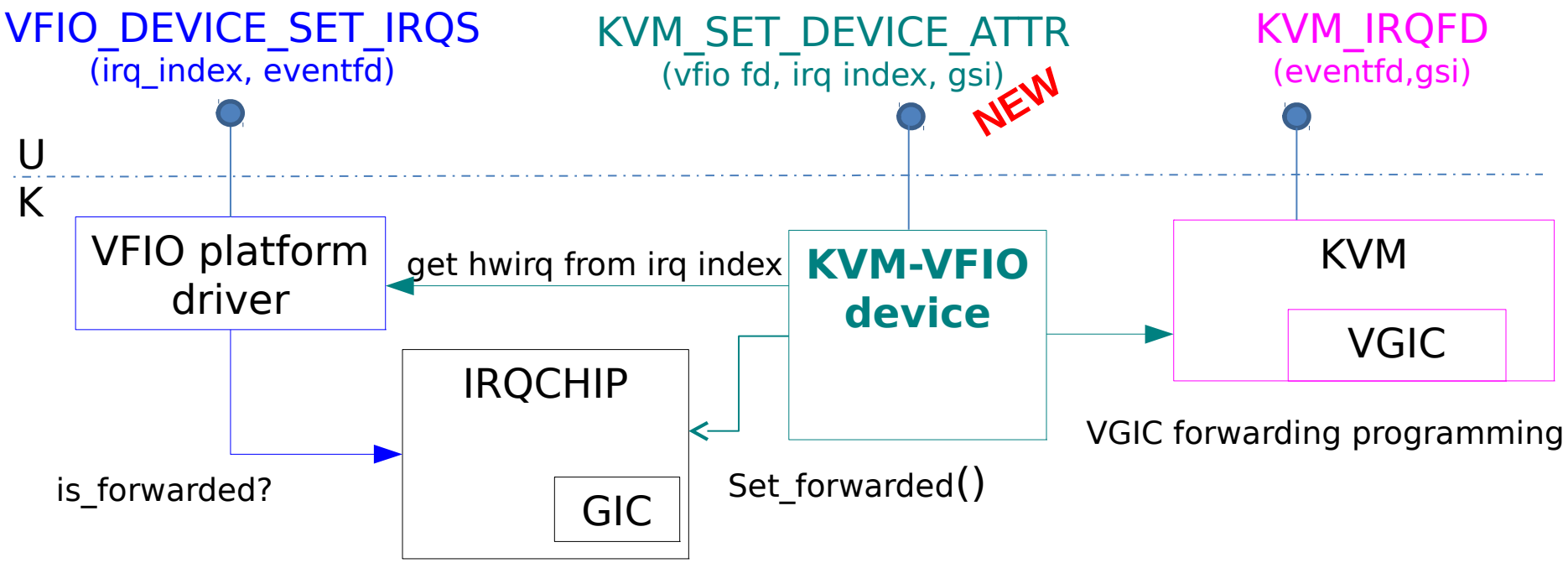
IRQ Path with KVM (irqfd/forward)



Linaro

Forwarded IRQ Integration

Allow userspace to configure forwarding of a VFIO device IRQ



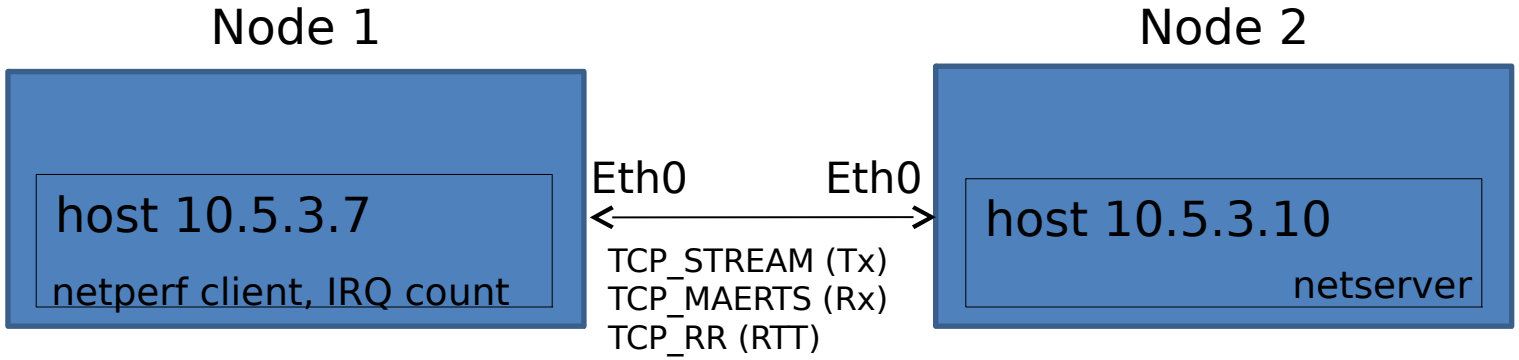
Linaro

Performance Measures

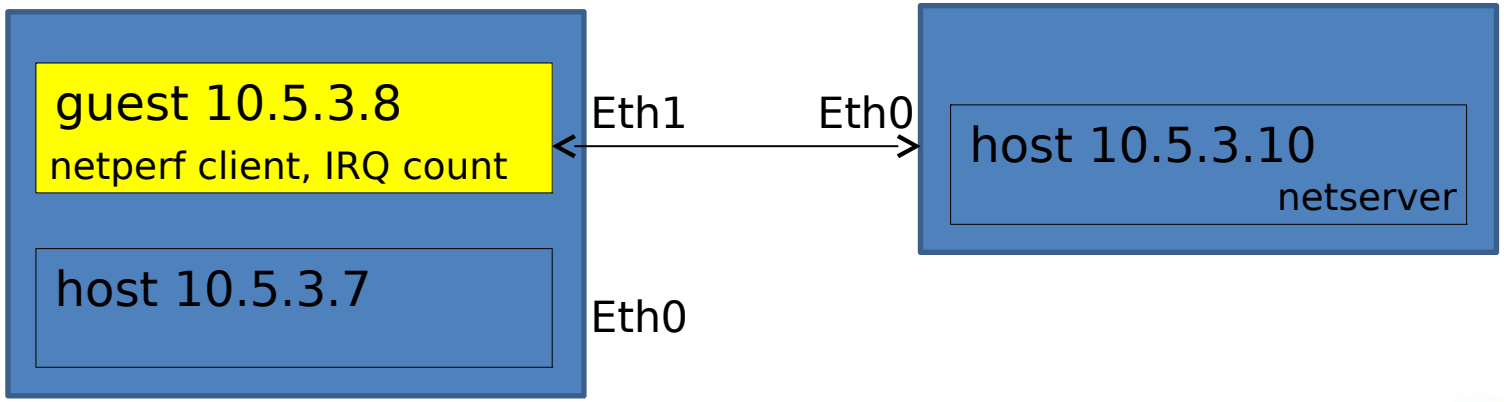
- Calxeda Midway
  - Communication between 2 nodes
  - 1Gb/s switch
- 2 xgmacs
  - eth0 assigned to host
  - eth1 assigned to guest if any
- Versions:
  - All kernels are 3.17rc3
  - QEMU is 2.1.0

Comparison

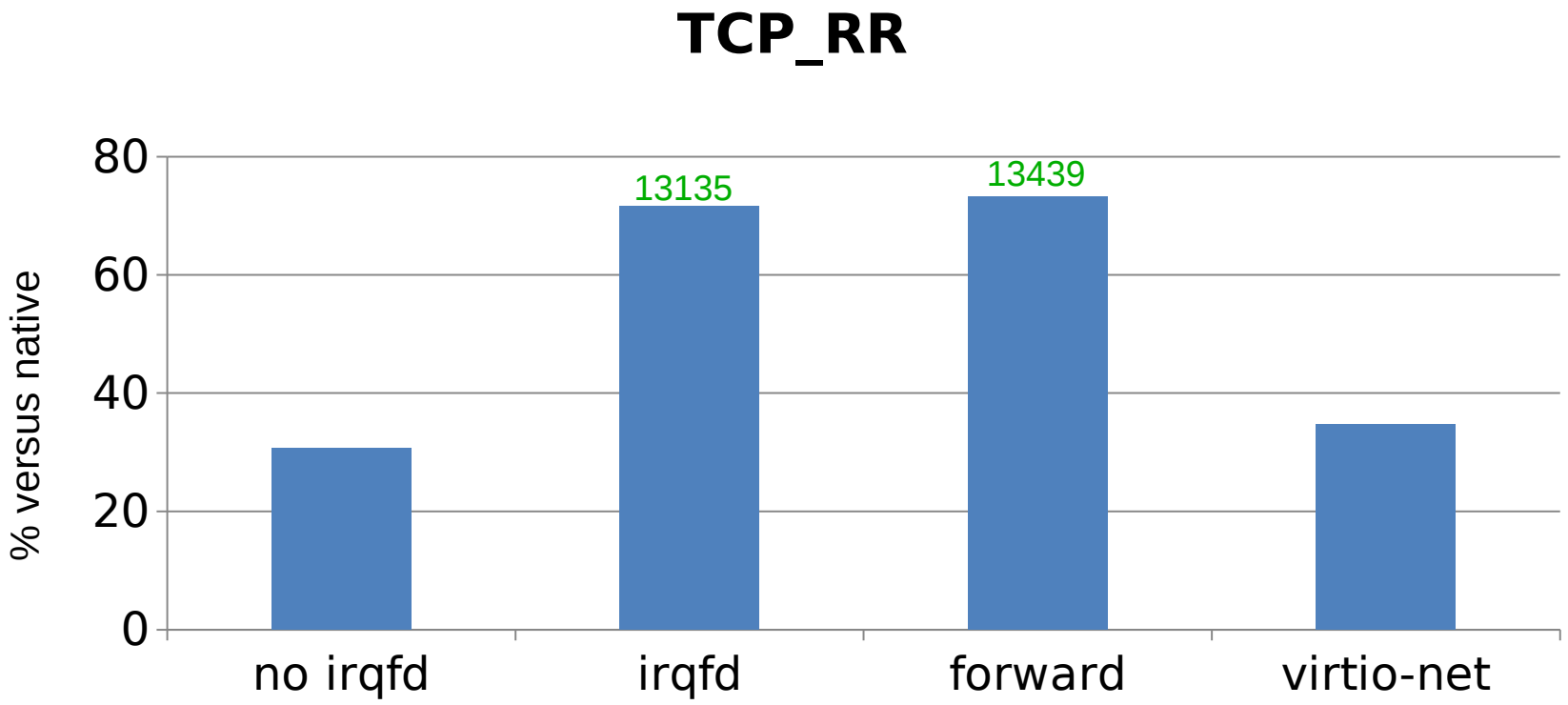
- Native Performance



- Guest Performance



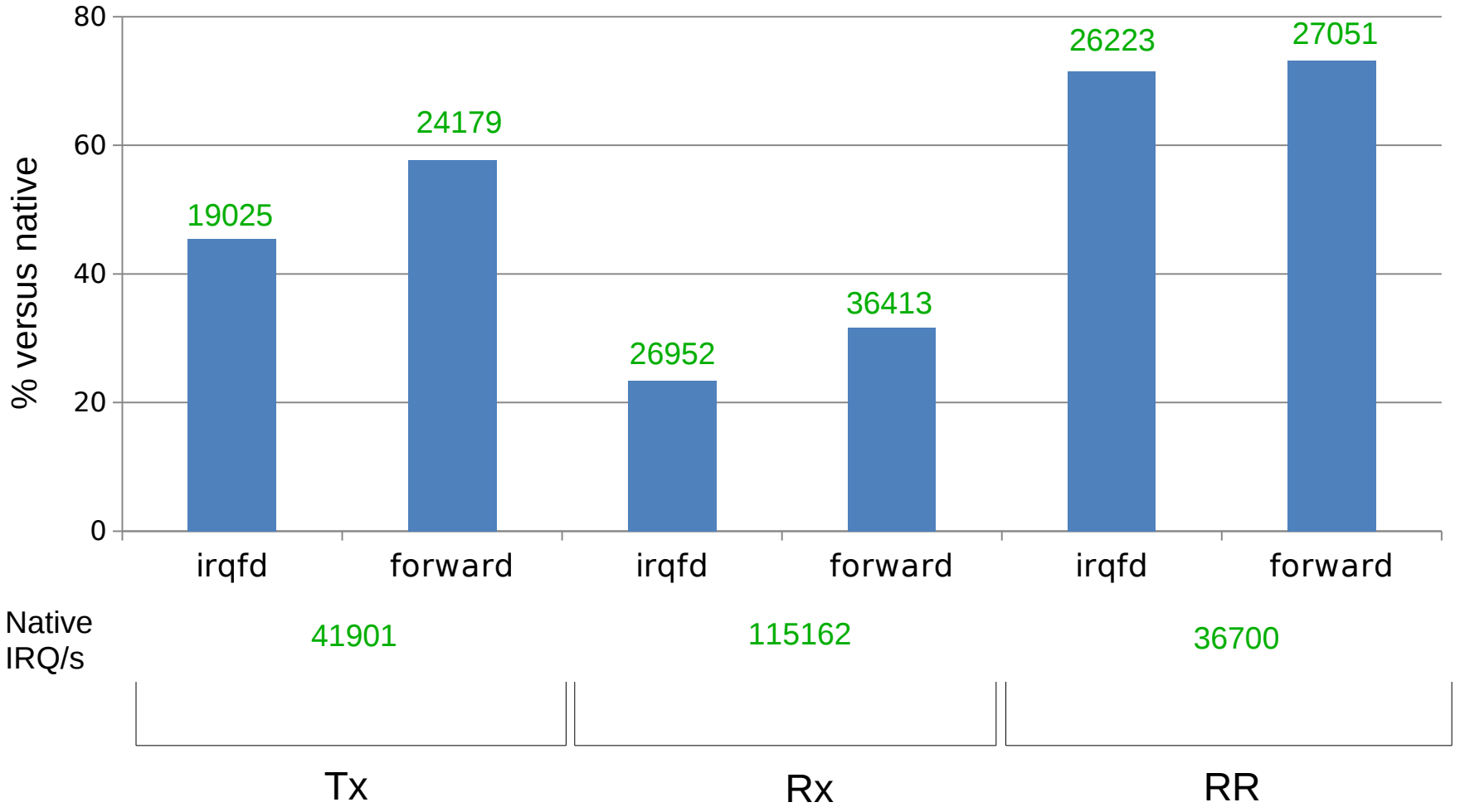
Round Trip Time



Native Perf: 18350 trans/s

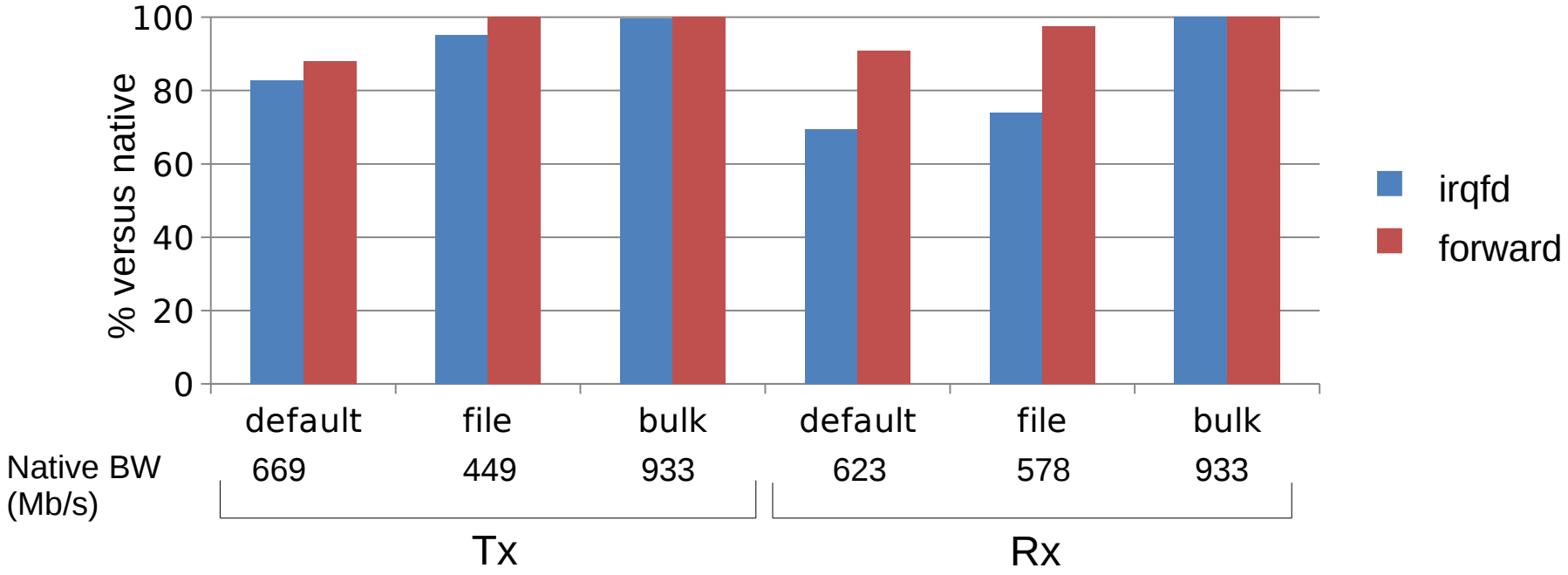
Linaro

Xgmac IRQ rate on guest (IRQ/s)



Linaro

Throughput with 3 TCP/IP patterns



	default	file	bulk
Local Tx & Rx socket buffer size (-s)	8kB	8kB	64kB
Remote Tx & Rx socket buffer size (-S)	8kB	8kB	64kB
Local send size (-m)	8kB	4kB	8kB
Remote received size (-M)	8kB	4kB	8kB

Status & Next

QEMU patches & dependencies

#	QEMU Patches	Author
0	KVM platform device passthrough	E. Auger
1	Dynamic sysbus device allocation support	A. Graf
2	machvirt dynamic sysbus device instantiation	E. Auger

#	Kernel Patches	Author
0	VFIO support for platform devices	A. Motakis
1	ARM: KVM: add irqfd support	E. Auger
2	KVM-VFIO IRQ forward control	E. Auger
3	ARM: Forwarding physical interrupts to a guest VM	M. Zyngier

Linaro

Conclusion

- Main functional bricks are available for efficient KVM platform device passthrough
- Forwarded IRQ usage shows improvements on
  - Sustained IRQ rate
  - Latency
  - Bandwidth, on some patterns
- Please test and use VFIO platform
  - Start integrating your devices
  - Share issues with complex device tree nodes
  - Work ongoing on AArch64 too

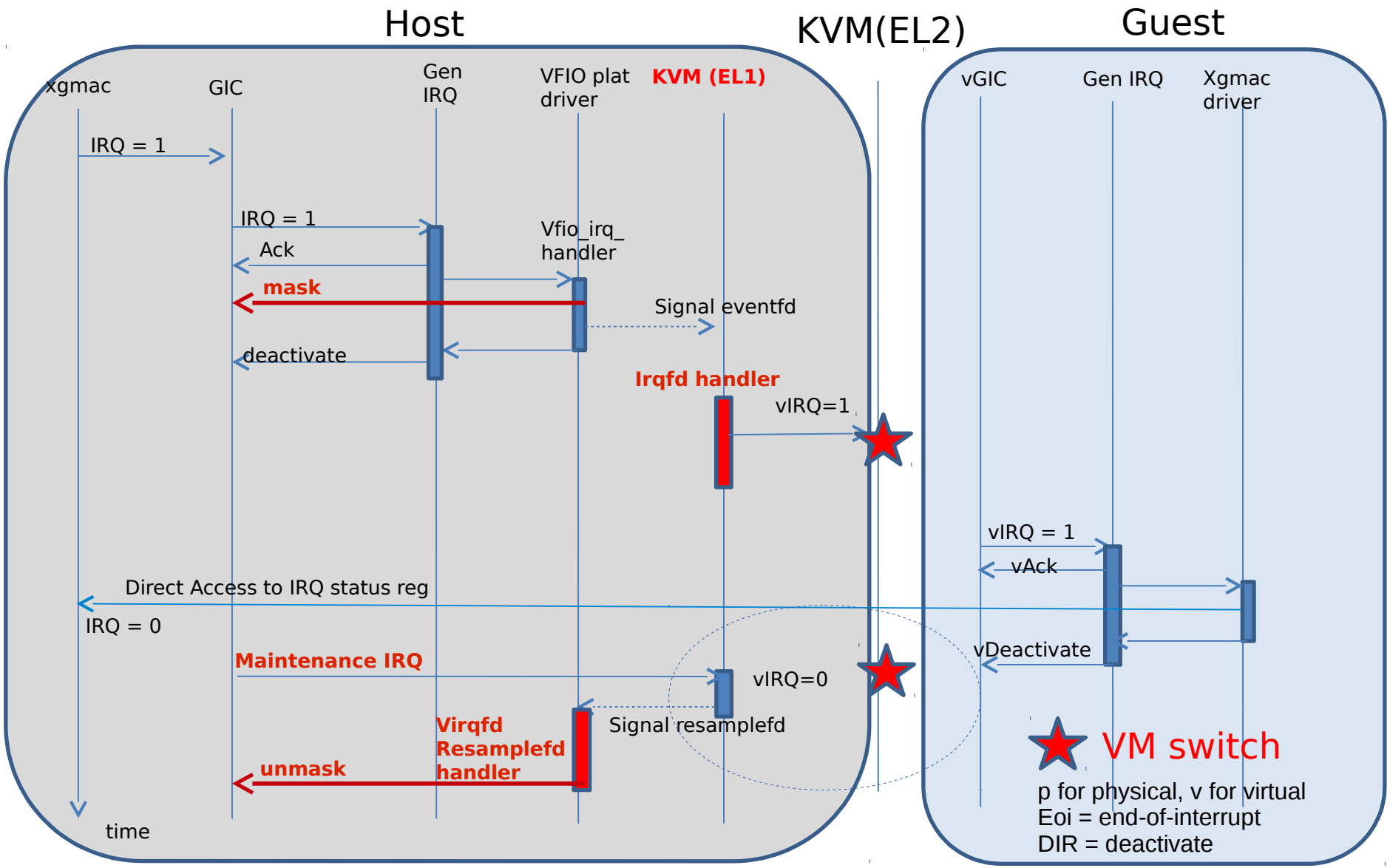
Linaro

Questions?

Thanks!

Backup Slides

Irqfd Standard ARM Porting



Forwarded IRQ