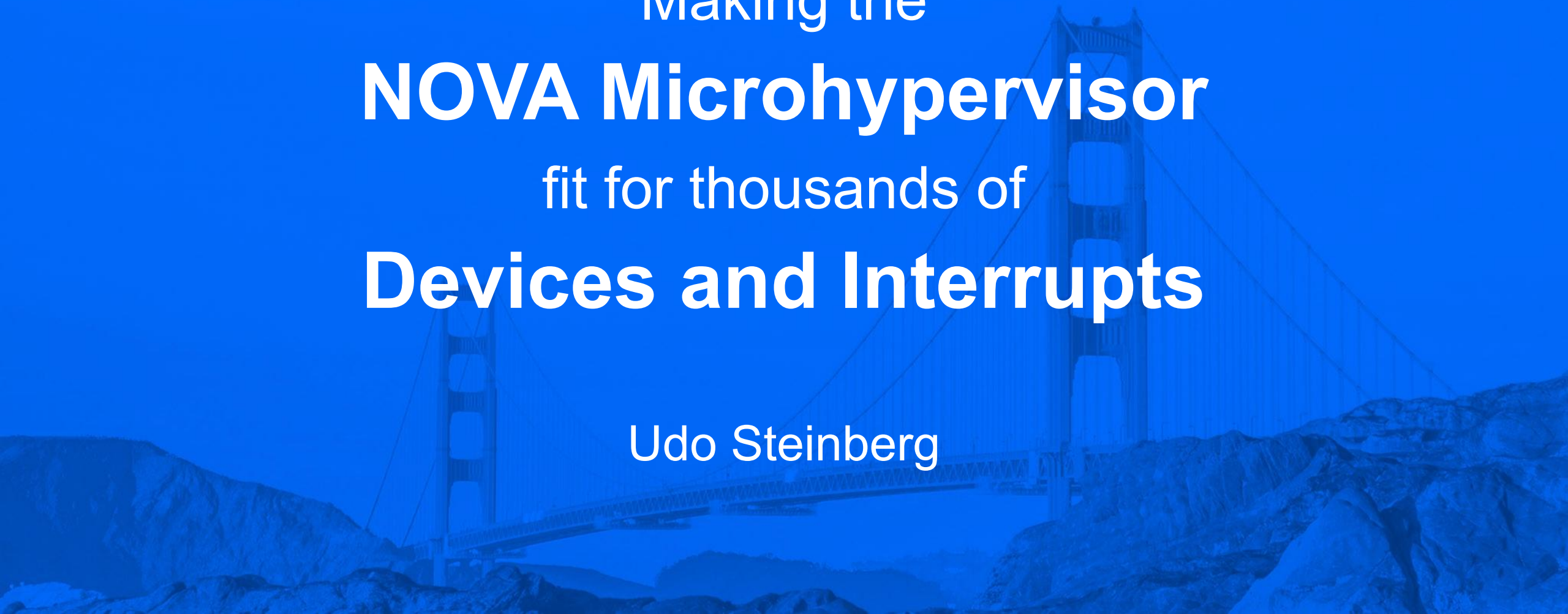
BlueRock

Making the NOVA Microhypervisor fit for thousands of Devices and Interrupts

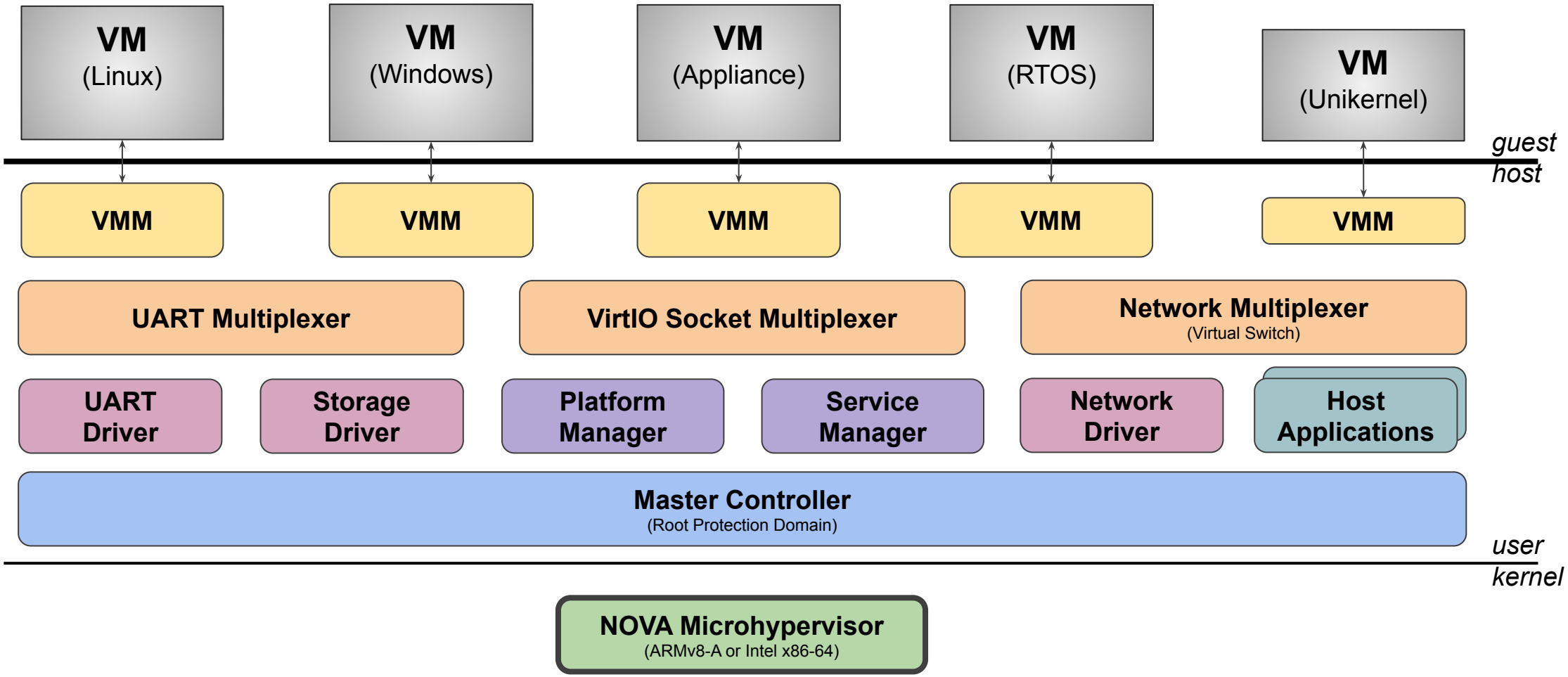


Udo Steinberg

Agenda

- Architecture Overview
- Formal Verification Update
- Scalability Improvements
  - More Devices (Arm SMMUv3)
  - More Interrupts (Arm GITS and Intel IOMMU)
  - New APIs (Device Contexts)
- Q & A

Architecture Overview



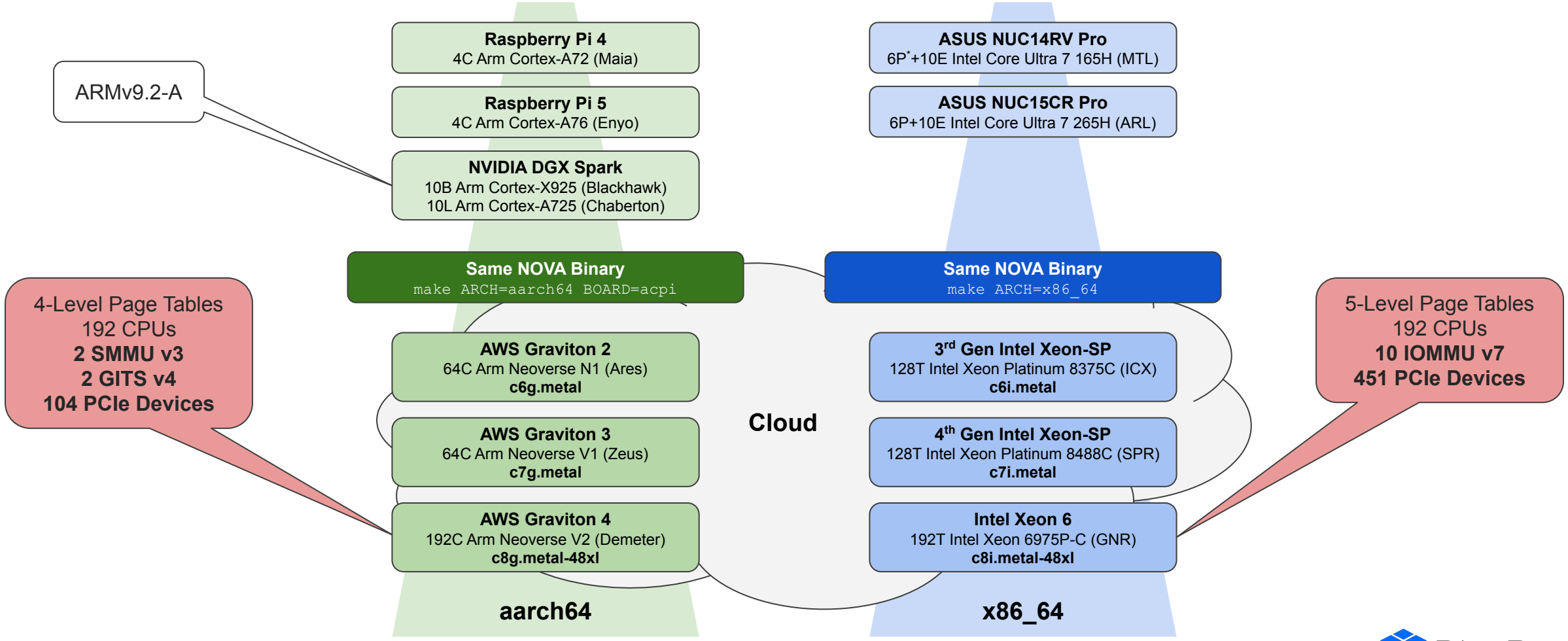
Formal Verification of Bare Metal Property™

Formal Verification Update

ROCQ

- Source Code
  - https://gitlab.com/bluerocksec/NOVA
- Project Website
  - https://bluerocksec.gitlab.io/formal-methods
- BRiCk C++ Logic
  - https://github.com/skylabsai/brick
- Proof Automation
  - https://skylabsai.github.io/auto-docs
- Open-Source Specs & Proofs (GPLv2)
  - Focus on architecture-independent C++
- What's next?
  - Verification of Memory Spaces
    - Relaxed Memory Reasoning
    - Interaction with Hardware: MMU / TLB

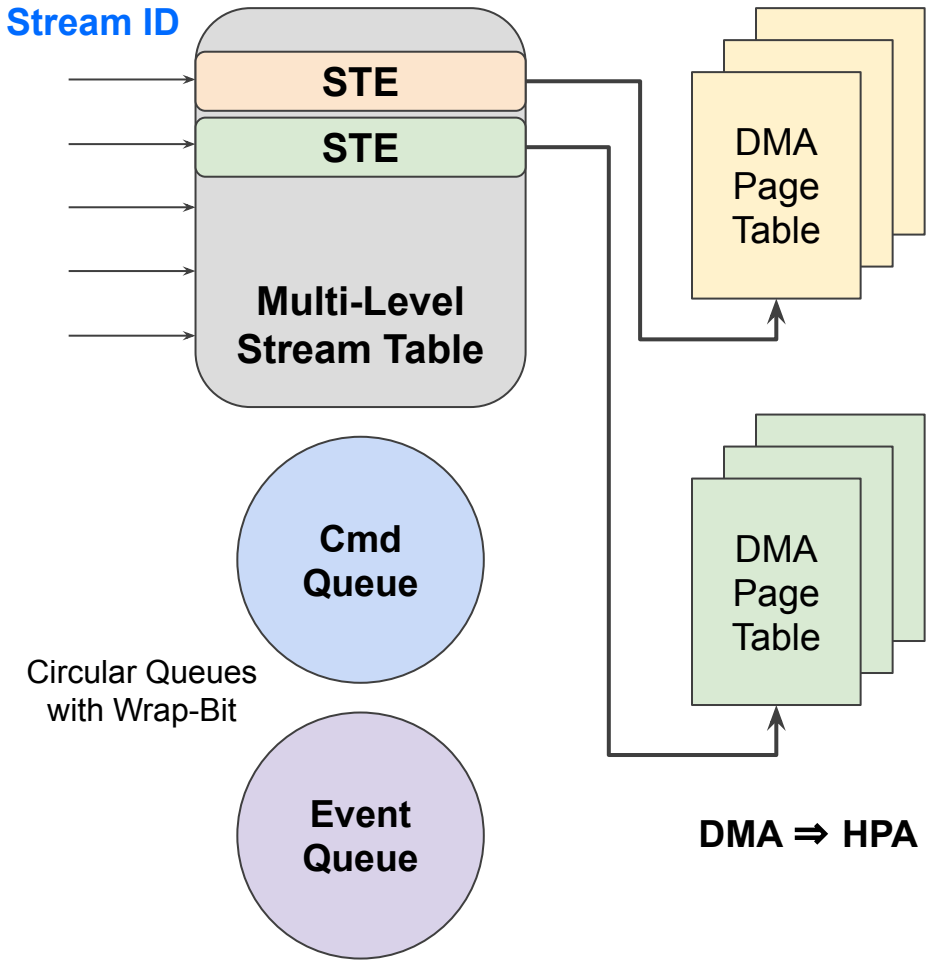
Multiple Architectures and Scalability



Kernel Functionality for User-Mode Device Drivers

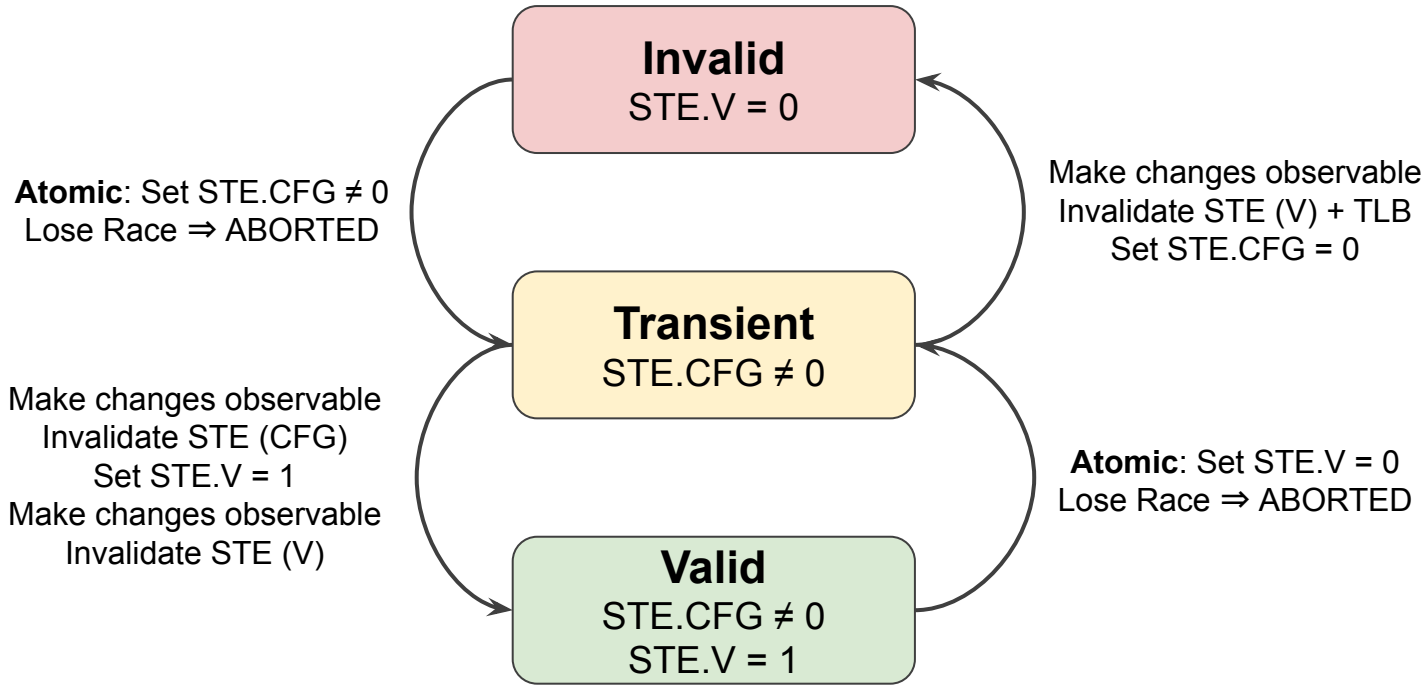
- Driving a device requires access to device registers (typically MMIO)
  - Can be mapped into a PD's Host Space (Host-Driven) or Guest Space (Guest-Assigned)
- Host driver needs to receive and acknowledge device interrupts
  - Interrupts signaled via UP on interrupt semaphore, DOWN acknowledges interrupt
- Access Control for DMA Transfers **DMA ⇒ HPA Remapping**
  - **`assign_dev`** assigns a Device to a DMA space
- Access Control for Global System Interrupts **GSI ⇒ CPU+SM Remapping**
  - **`assign_int`** assigns an Interrupt to a Device and configures/routes that interrupt

DMA Translation (Arm SMMUv3)

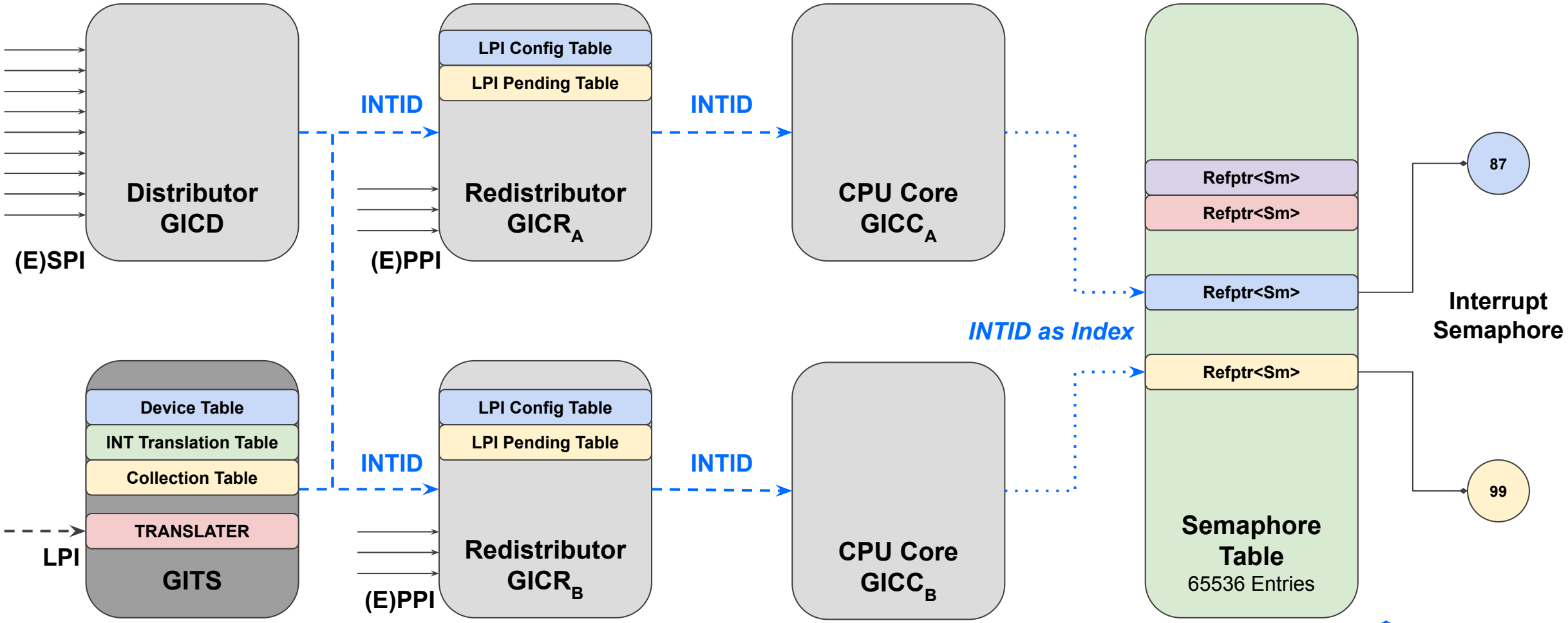


Stream Table Entry (State Transitions)

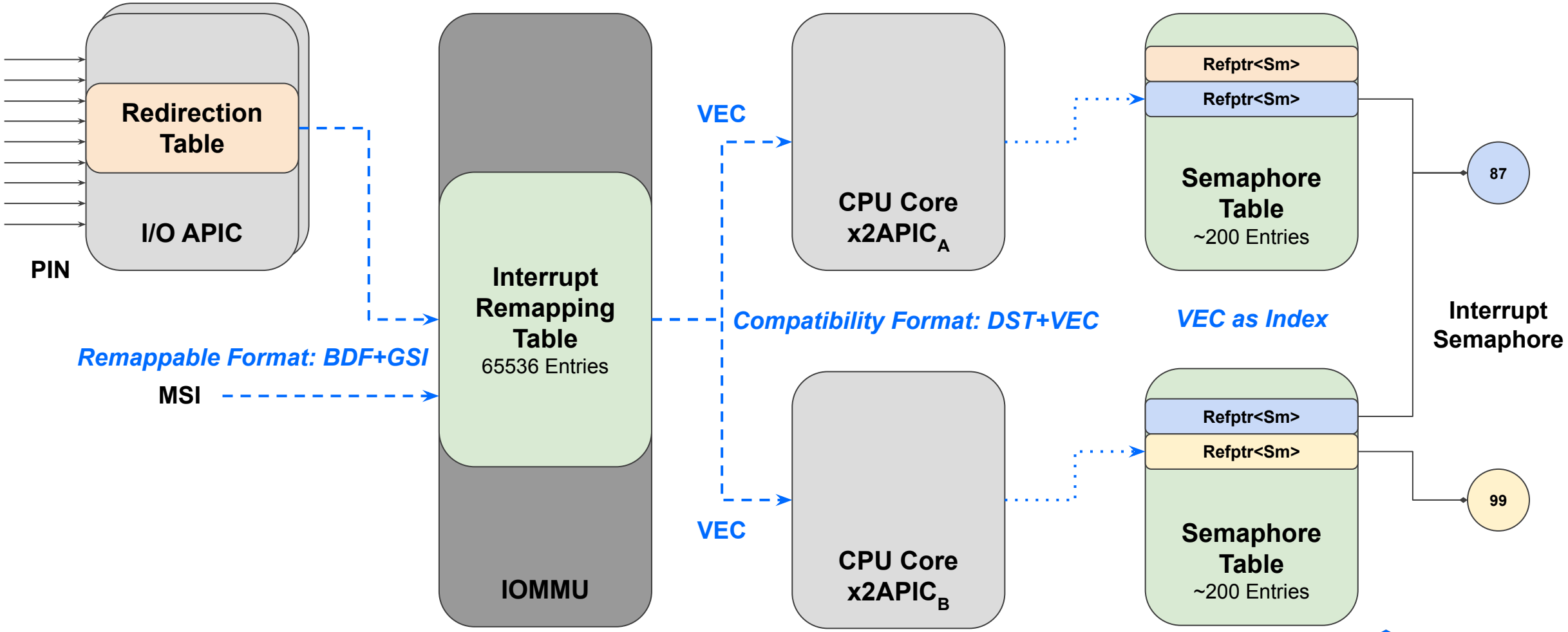
(512-bit STE cannot be configured atomically)



Interrupt Subsystem (Arm GICv3)



Interrupt Subsystem (x86)



Interrupt Differences

Arm

- Global Pin-Based (GICD)
  - (E)SPI configuration registers
- Per-CPU Pin-Based (GICR)
  - (E)PPI configuration registers
- Message-Signaled (GITS)
  - Several LPI-related tables

Interrupt asserts a **global INTID**

x86

- Global Pin-Based (IOAPIC)
  - IOAPIC Redirection Table
- Per-CPU Pin-Based (LAPIC)
  - Local APIC Vector Table
- Message-Signaled (IOMMU)
  - One interrupt remapping table

Interrupt asserts a **per-CPU vector**

Interrupt Semaphore Tables

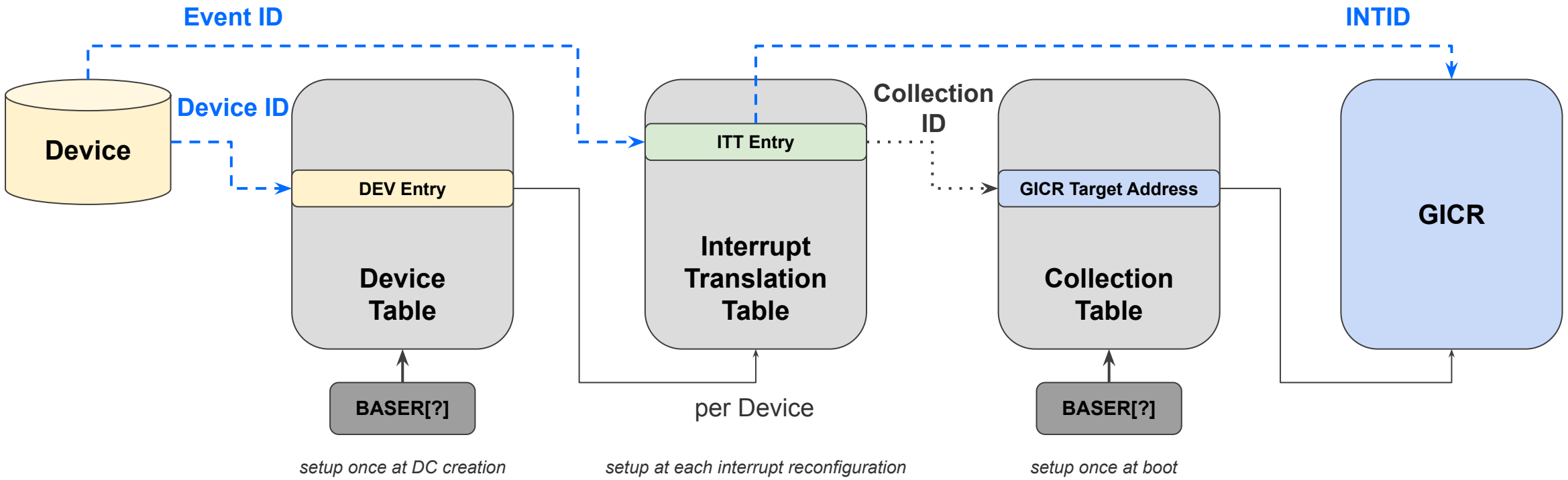
Global (Arm)

- Indexed by GSI number (INTID)
  - Checks GSI ≤ GSI_{MAX}
  - Table index must match GSI
- Interrupt Scalability Limit
  - 2^{16} GSIs in the entire system
  - Practical Limit: Number of LPIs

CPU-Local (x86)

- Indexed by local VEC number
  - Checks VEC ≤ VEC_{MAX}
  - Table index can be any VEC
- Interrupt Scalability Limit
  - 2^{16} GSIs per PCI segment group
  - Practical Limit: ~200 per CPU

GIC Interrupt Translation Service



- Which BASER[0…7] maps what table type is only defined for GIC ≥ v4p1

  ⇒ System software must discover+remember the table type of each BASER

Programming GIC ITS Base Registers

- Indirect bit (63) defines if a table is one-level (flat) or two-level (hierarchical)
  - *This field is RAZ/WI for GIC implementations that only support flat tables*
- Shareability bits (11:10) define outer/inner shareable or not shareable
  - *It is implementation-defined whether this field has a fixed value or can be programmed*
- PageSize bits (9:8) define the page size used by the table as 4K, 16K or 64K
  - *If the GIC implementation supports only a single, fixed page size, this field might be RO*
- Size bits (7:0) allow up to 256 pages to be concatenated at the top-level

⇒ System software must keep trying various settings until a configuration sticks

NOVA API: Naming Devices and Interrupts

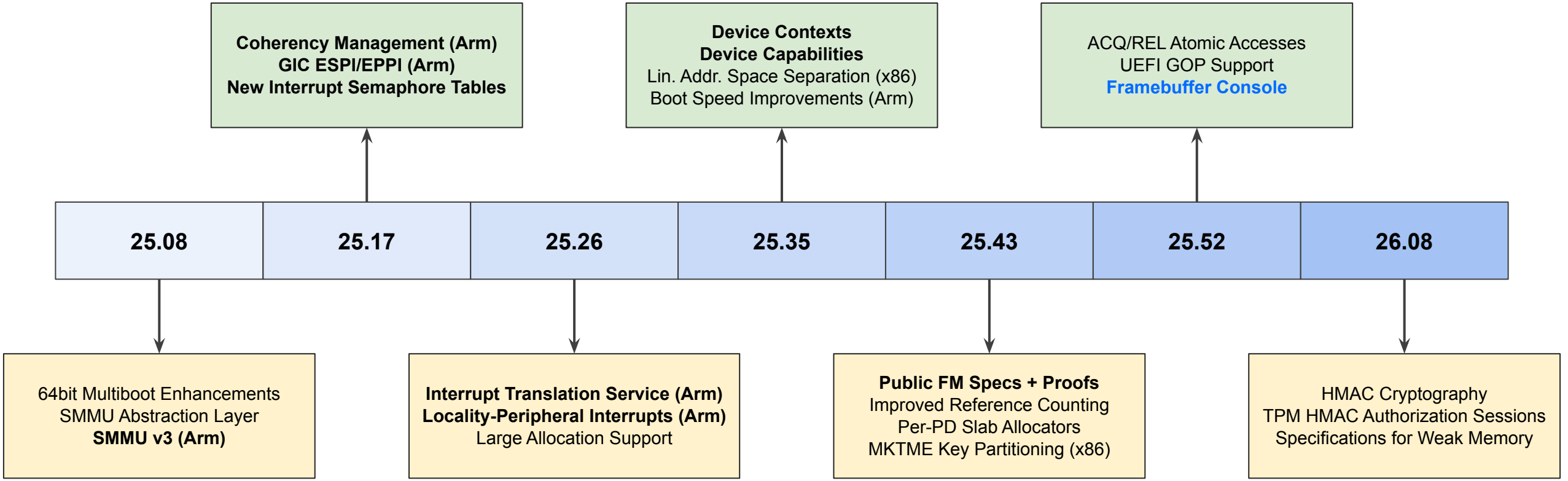
- Access Control requires naming and delegation/revocation of resources
  - NOVA Authorization Model: Capabilities
- Multiple hypercalls need to name a device or an interrupt
  - The naming scheme is dictated by the hardware architecture instead of being generic
- Idea: Store the architecture-specific details once upon object creation
  - Interrupt details ⇒ in **Interrupt Semaphore** via Interrupt Semaphore Descriptor (ISD)
  - Device details ⇒ in **Device Context** via Device Topology Descriptor (DTD)

⇒ Keeps the rest of the hypercall API clean and access-control checks generic

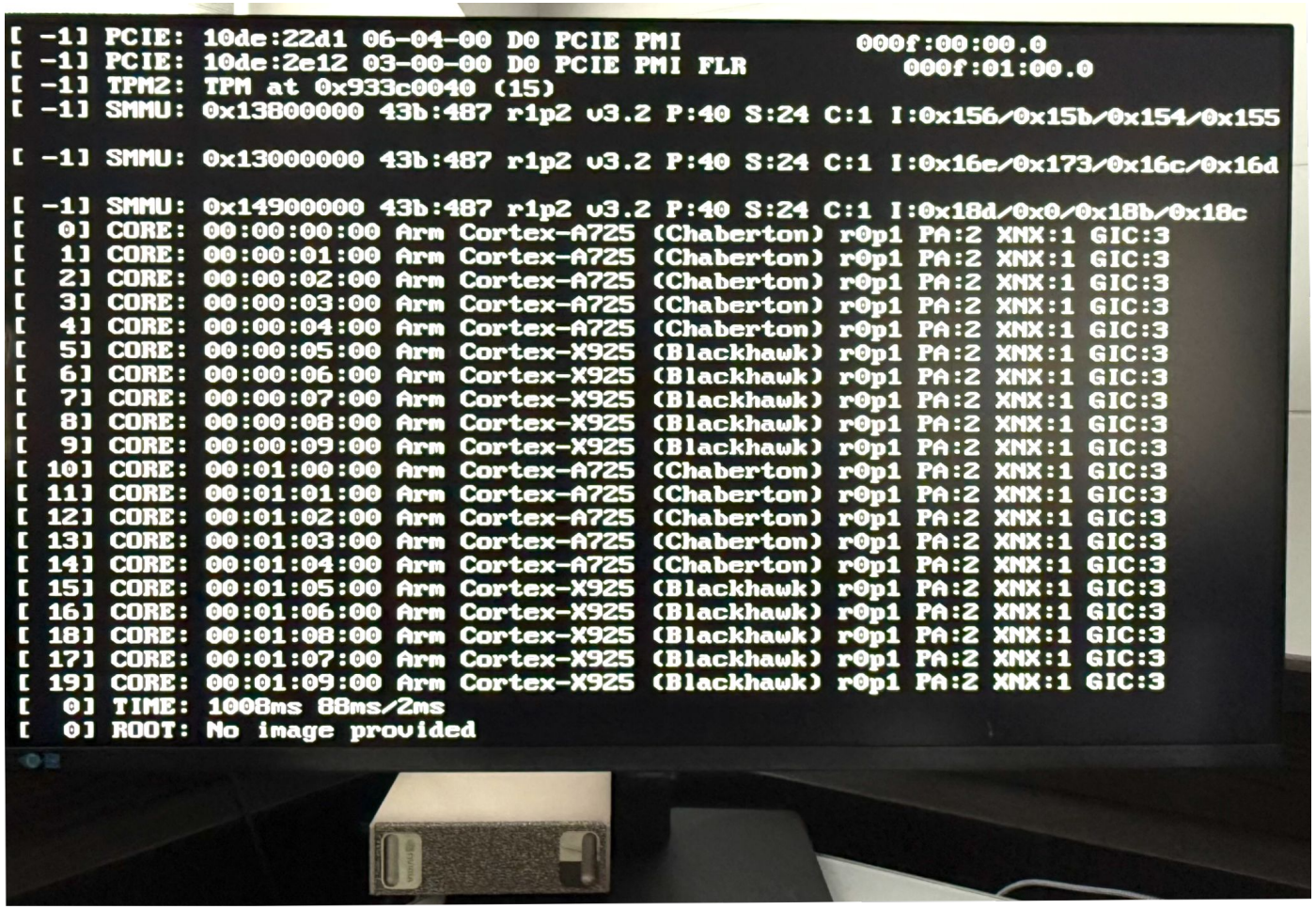
NOVA API: Device Contexts

- Encapsulate architecture-dependent information for a platform device
  - Device Topology (Unique Identification)
    - Arm: Device ID, Stream ID
    - x86: PCI Segment / Bus / Device / Function
  - Information about associated DMA / Interrupt Translation Units
    - Arm: SMMU / GITS address
    - x86: IOMMU address
- Provide access control via capability permissions for assigning the device as
  - Source of DMA transfers targeting a DMA space (**`assign_dev`**)
  - Source of MSI/LPI targeting an interrupt semaphore (**`assign_int`**)

NOVA Releases and Features



Framebuffer Console



- **Framebuffer Performance**
  - NVIDIA DGX Spark (Arm)
    - 2048x1536 @ 858 fps
  - Raspberry Pi4 (Arm)
    - 1920x1200 @ 245 fps
  - ASUS NUC15 Pro (x86)
    - 1920x1080 @ 4537 fps
- Using the UART console will slow the framebuffer down

What's Next?

- We plan to make significant portions of the BlueRock user-mode stack publicly available under an open-source license in the near future
- Reducing the barrier of entry
  - There will be a global Kconfig for the entire software stack
  - The build process will put all components into one big monolithic image
  - The same mono-image can be booted in several different ways:
    - As a Multiboot2-compatible kernel image
    - As a bzImage from a Linux-compliant bootloader (e.g., GRUB, u-Boot, etc.)
    - As a UEFI executable from firmware (e.g., UEFI shell, UEFI boot menu)

Questions and Discussion

The NOVA microhypervisor is licensed under GPLv2

Releases: https://github.com/udosteinberg/NOVA/tags

More Information: hypervisor.org