arm

Shared Virtual Addressing for high performance Arm Infrastructure platforms

LVC21F-217

Vivek Gautam
Open Source Software Group, Arm

© 2021 Arm

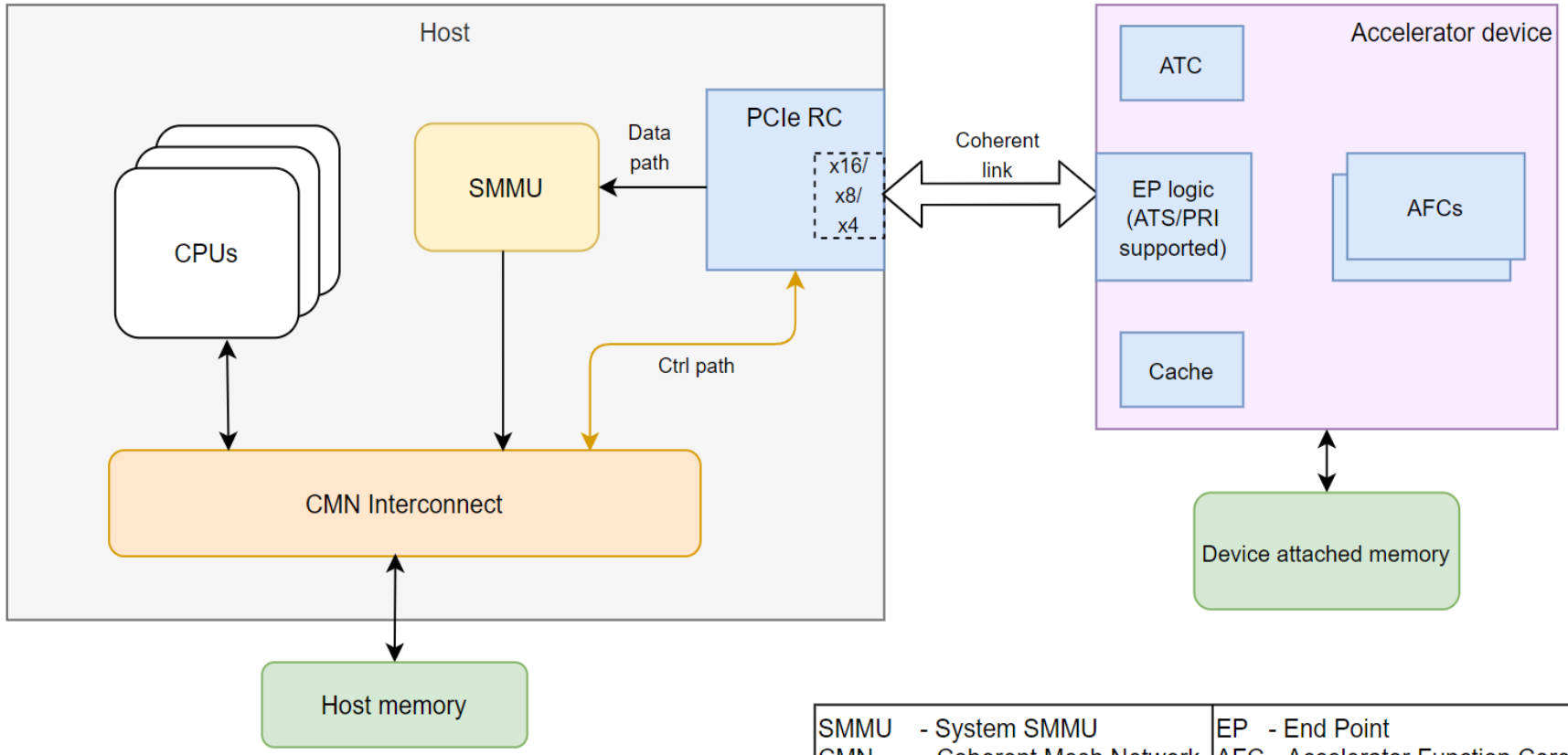
Agenda

- Introduction to Shared Virtual Addressing (SVA)
- Hardware and Software view
- Virtualization
- Current design for vSVA
- Upstream status

© 2021 Arm

Why Shared virtual addressing

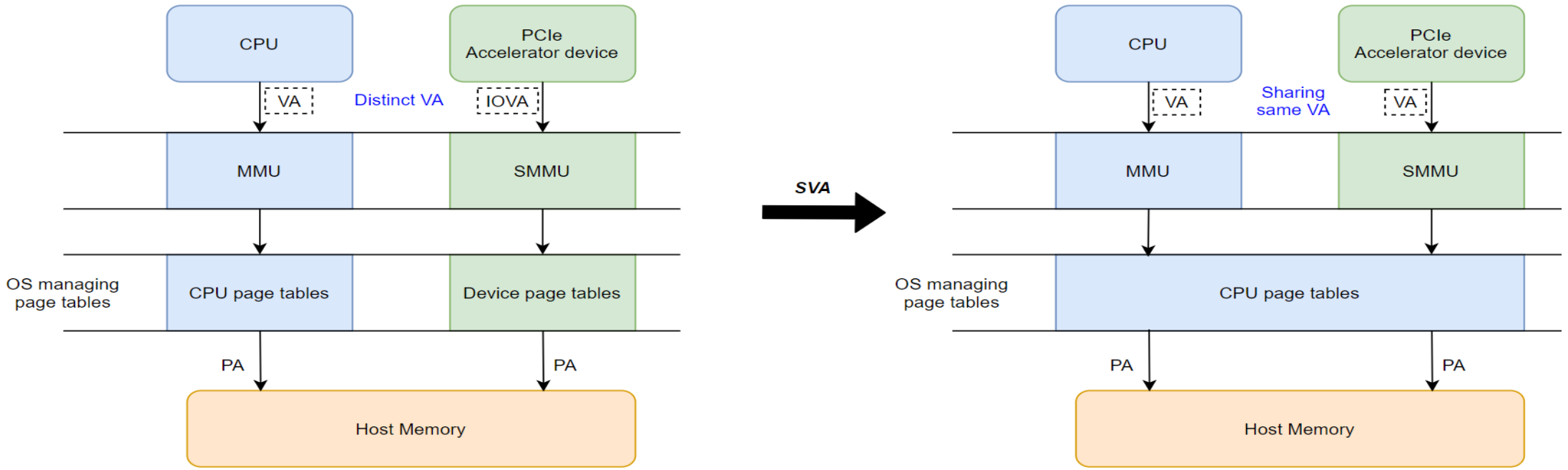
- High performance accelerators – such as GPGPUs, NPUs, SmartNICs etc. are used in infrastructure platforms.
- PCIe – the backbone bus of infrastructure.
- Hardware coherency requirement for accelerators.



SMMU - System SMMU	EP - End Point
CMN - Coherent Mesh Network	AFC - Accelerator Function Core
PCIe RC - PCIe root complex	ATC - Address Translation Cache

© 2021 Arm

What is Shared virtual addressing (SVA)



Adjacent methodologies

- OpenCL 2.0
  Shared virtual memory
- HSA
  Shared virtual memory
- CUDA
  Unified virtual memory

- SVA allows sharing same virtual address space between IO devices (accelerators) and application processors.
- The ability to perform DMA on a process address space rather than using a separate DMA address space.

© 2021 Arm

Requirements of SVA

- PCIe Address Translation Service (ATS)
- PCIe Page Request Interface (PRI)
- PCIe Process Address Space ID (PASID)

PCIe Device features

- SMMU-v3 support for ATS
- SMMU-v3 support for PRI - PRIQ
- SMMU-v3 support for PASID - SSID
- SMMU-v3 support for Nested translation

Arm IP support

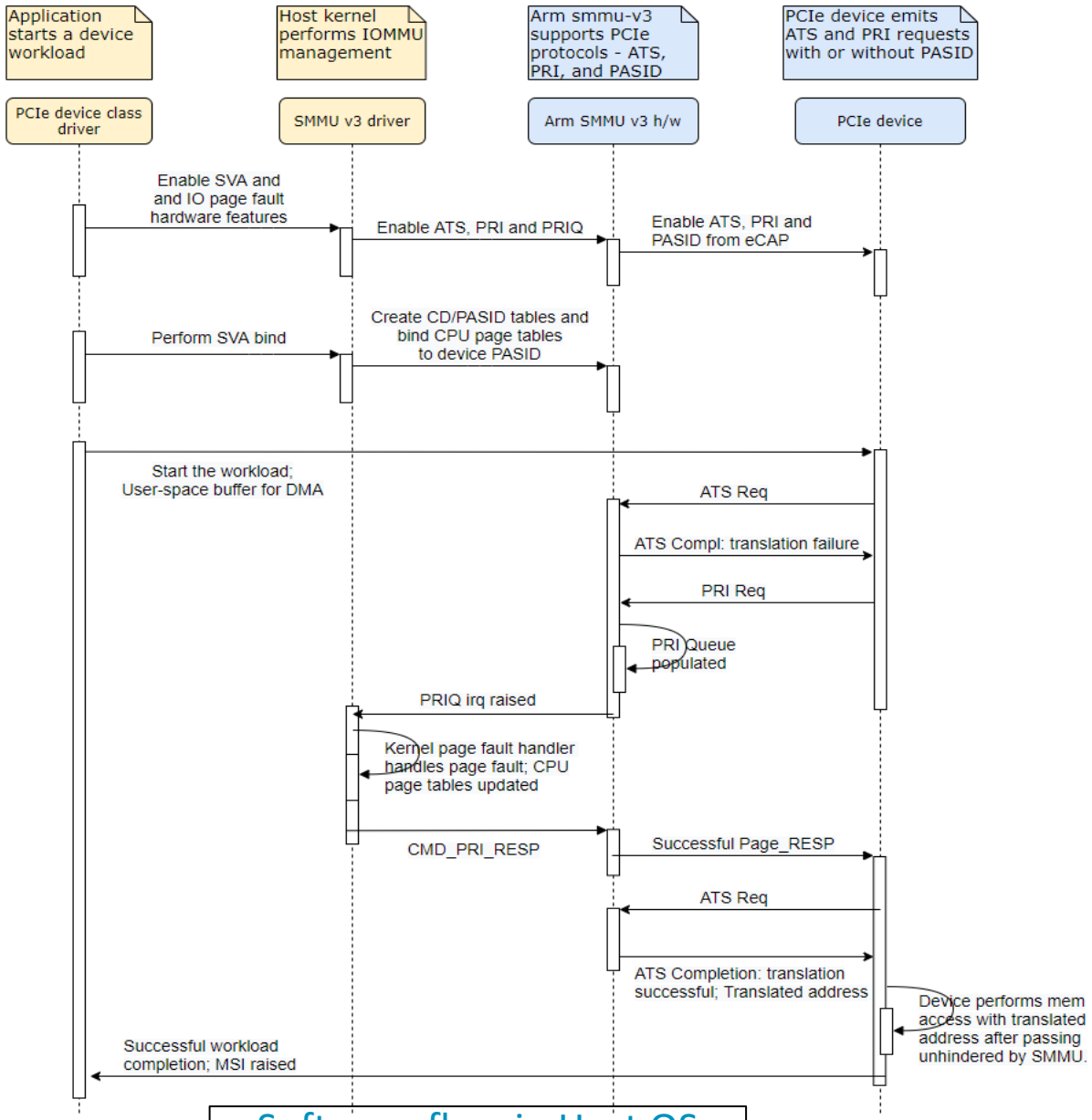
- Paravirtualized IOMMU – virtio-iommu
- Virtual Function IO framework (VFIO) – Device assignment
- VMM - kvmtool

Software support for Virtual machine

© 2021 Arm

Host SVA system

- SVA bind:
  - Binds process address space with device PASID; programs CPU page tables in SMMU-v3 Context descriptors.
- PRIQ interrupt handler:
  - Calls kernel page fault handler to update CPU page tables.
  - Sends CMD_PRI_RESP once page fault is handled.

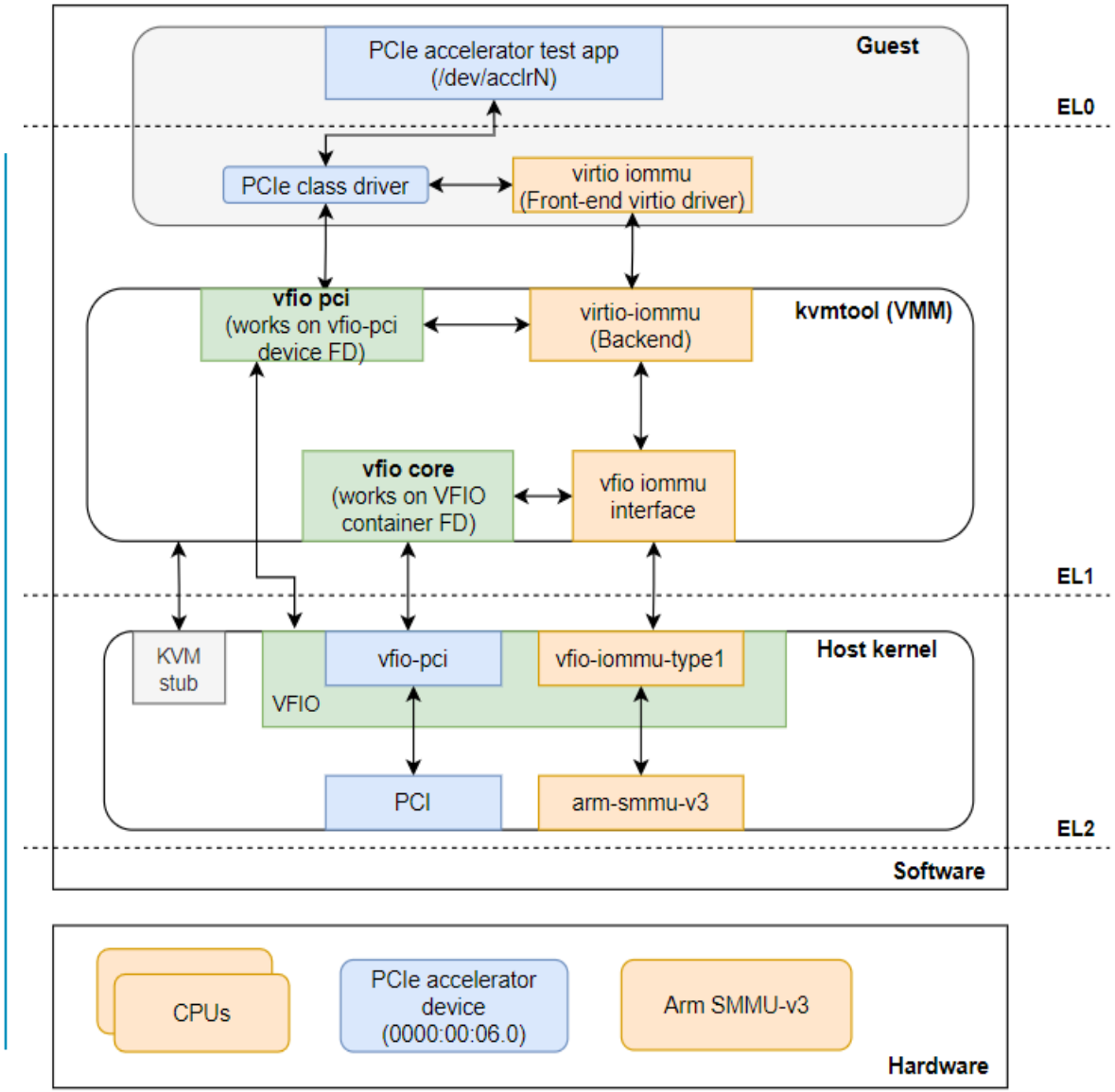


Software flow in Host OS

© 2021 Arm

Virtualization and vSVA

- Key components:
  - PCIe passthrough – via vfio-pci.
  - DMA remapping in guest - Nested translation support – via emulated/para-virtualized iommu device.
  - DMA fault handling – via VFIO-IOMMU interface.
  - PCI Page request interface for I/O page fault – via PCIe and arm-smmu-v3 PRI support.



© 2021 Arm

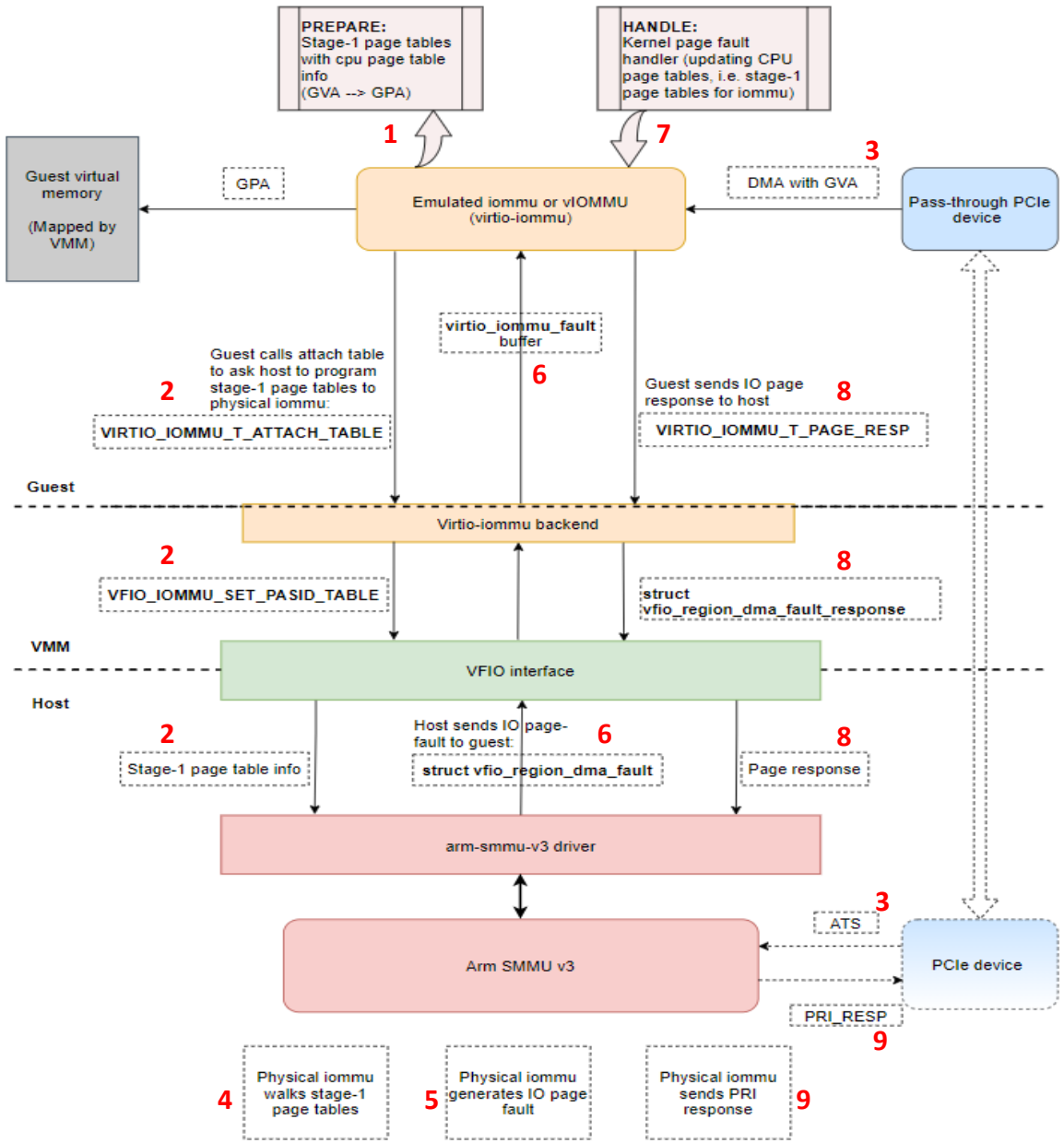
Current design of vSVA

- Built on top of VFIO and IOMMU API changes by Eric Auger –
  - [PATCH v15 00/12] SMMUv3 Nested Stage Setup (IOMMU part)
  - [PATCH v13 00/13] SMMUv3 Nested Stage Setup (VFIO part)
- On going IOMMU user-API proposal –
  - [RFC v2] /dev/iommu uAPI proposal
- Design for virtio-iommu should be mostly independent of IOMMU/VFIO uAPI changes.
- Few additions proposed to virtio-iommu specification and driver:
  - virtio-iommu requests:
    - VIRTIO_IOMMU_T_ATTACH_TABLE
    - VIRTIO_IOMMU_T_INVALIDATE
    - VIRTIO_IOMMU_T_PAGE_RESP
  - virtio-iommu feature bits:
    - VIRTIO_IOMMU_F_ATTACH_TABLE
    - VIRTIO_IOMMU_F_SVA

© 2021 Arm

Current design (contd.)

- vSVA flow is based on current design.
- Future virtio-iommu changes will incorporate dependencies arising from IOMMU uAPI proposal.



© 2021 Arm

Upstreaming status

Linux kernel

- Major work has been done towards enabling SVA support in guest using virtio-iommu.
- The changes include support for nested page table, support to handle dma fault from host kernel, and to send page fault response from guest kernel.
- Patches:
  - Nested page table support with virtio-iommu:
    [PATCH RFC v1 00/15] iommu/virtio: Nested stage support with Arm
  - Patches to add SVA support to virtio-iommu:
    [PATCH RFC v1 00/11] iommu/virtio: vSVA support with Arm
  - Publishing next version soon.

kvmtool

- SVA changes on kvmtool based on virtio-iommu driver that was added by Jean Philippe (not yet upstreamed):
  - https://jpbrucker.net/git/kvmtool/log/?h=virtio-iommu/devel

© 2021 Arm

Thank you!

Questions?

© 2021 Arm

arm

Thank You
Danke
Gracias
谢谢
ありがとう
Asante
Merci
감사합니다
धन्यवाद
Kiitos
شكرًا
ধন্যবাদ
תודה

© 2021 Arm

The Arm trademarks featured in this presentation are registered trademarks or trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. All rights reserved. All other marks featured may be trademarks of their respective owners.

www.arm.com/company/policies/trademarks

© 2021 Arm