Advanced Interrupt Architecture and Advanced CLINT

Anup Patel <anup.patel@wdc.com>
Western Digital System Software Research

John Hauser <jh.riscv@jhauser.us>
Independent Researcher

Outline

- AIA Specification Overview
- ACLINT Specification Overview
- Interrupts & Timer in Future RISC-V Platforms
- Software Status

AIA Specification Overview

Motivation and overview of AIA specification

RISC-V PLIC in existing platforms

External interrupts in existing RISC-V platforms

- RISC-V PLIC (originally SiFive PLIC) is widely used across existing RISC-V platforms to manage wired IRQs for M-mode and S-mode
- **Limitations:**
  - Consumes large amount of physical address space
    - A single PLIC instance targeting 4 HARTs requires > 2MB physical address space
    - Worst-case physical address space usage is 16MB
  - Global registers of PLIC are shared between M-mode and S-mode
    - Only one of M or S modes can be allowed to configure all interrupts
    - For security, either no M-mode interrupts, or S mode must make SBI calls to configure its interrupts
  - Configurable IRQ line sensing not supported by PLIC
    - Nature of each IRQ line (edge/level triggered) is fixed/hardwired by RISC-V platform
  - Message signaled interrupts (MSIs) not supported by PLIC
  - Interrupt virtualization not supported by PLIC

RISC-V AIA specification

External interrupts using new RISC-V AIA specification

- **RISC-V Advanced Interrupt Architecture (AIA)**
  - Address all limitations of RISC-V PLIC
  - https://github.com/riscv/riscv-aia/releases/download/0.2-draft.27/riscv-interrupts-027.pdf
  - **High-level design goals:**
    - Scalable for system with large number of HARTs
    - Define functionality as optional modular components
    - Support message signaled interrupts (MSIs)
    - Support interrupt virtualization
  - Specification is in stable state (no major changes anticipated)
- **Defines three modular (optional) components:**
  - Extended Local Interrupts (AIA CSRs)
  - Incoming Message Signaled Interrupt Controller (IMSIC)
    - IMSIC requires AIA CSRs
  - Advanced Platform Level Interrupt Controller (APLIC)

AIA: Extended Local Interrupts

Improved per-HART local interrupts using AIA CSRs

- Supports 64 local interrupts for both RV32 and RV64
- Supports configurable priority for each local interrupt
- Supports local interrupt filtering for M-to-S and HS-to-VS modes
- No impact on local interrupts 0 to 12 (defined by RISC-V privileged specification)

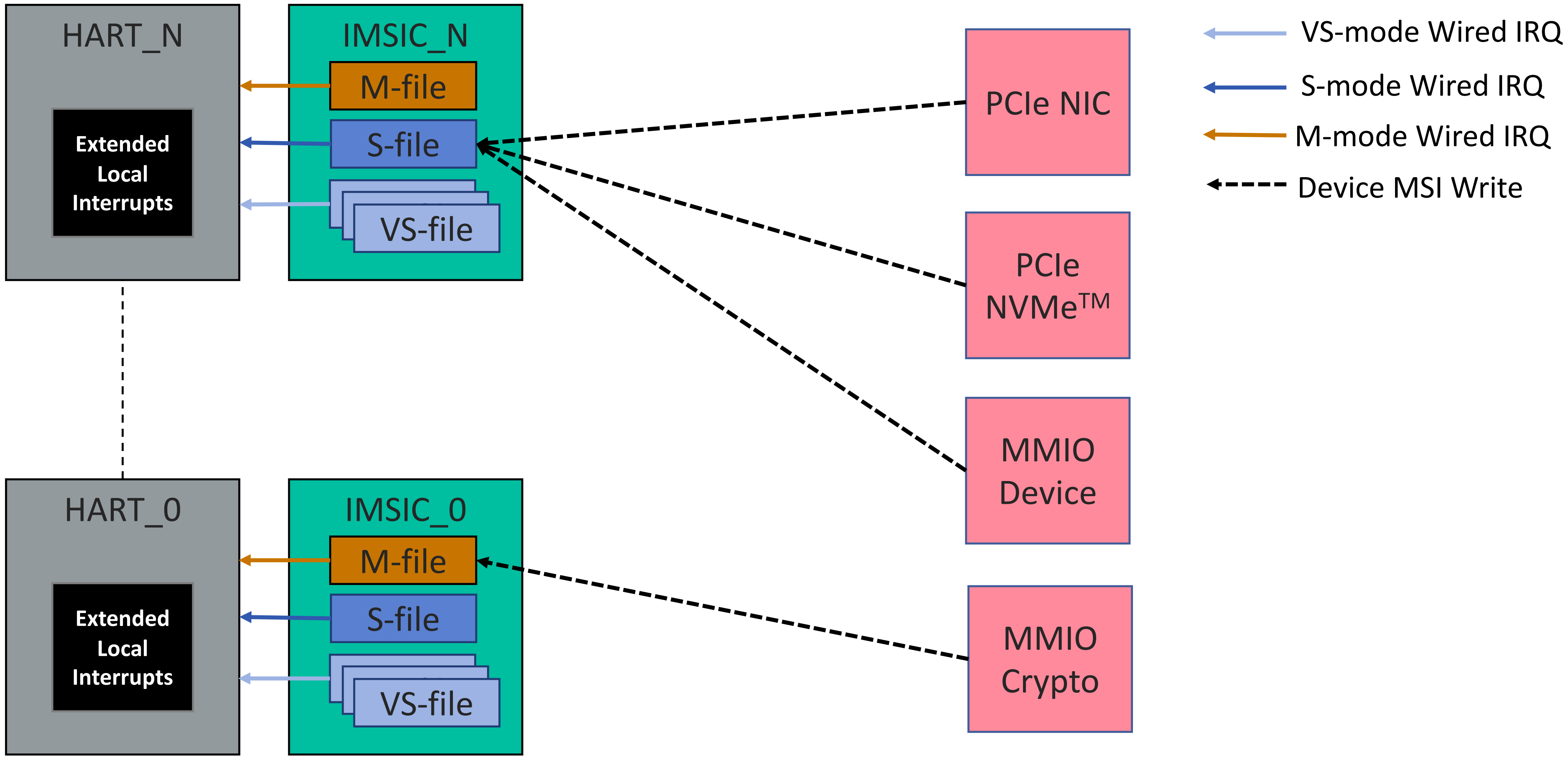
AIA: MSIs using IMSIC

External message signaled interrupts using IMSIC

- One IMSIC instance next to each HART
  - No limit on maximum number of HARTs
- Each IMSIC instance consist of multiple interrupt files
  - **One M-file** (M-level interrupt file), **one S-file** (S-level interrupt file), and **multiple guest-files/VS-files** (VS-level interrupt files)
  - Each interrupt file consumes 4KB of physical address space
- Interrupt file configuration done via AIA CSRs
- Each interrupt file supports up to 2047 interrupt identities
- MSI virtualization supported using VS-files for HARTs with H-extension

AIA: MSIs using IMSIC (Contd.)

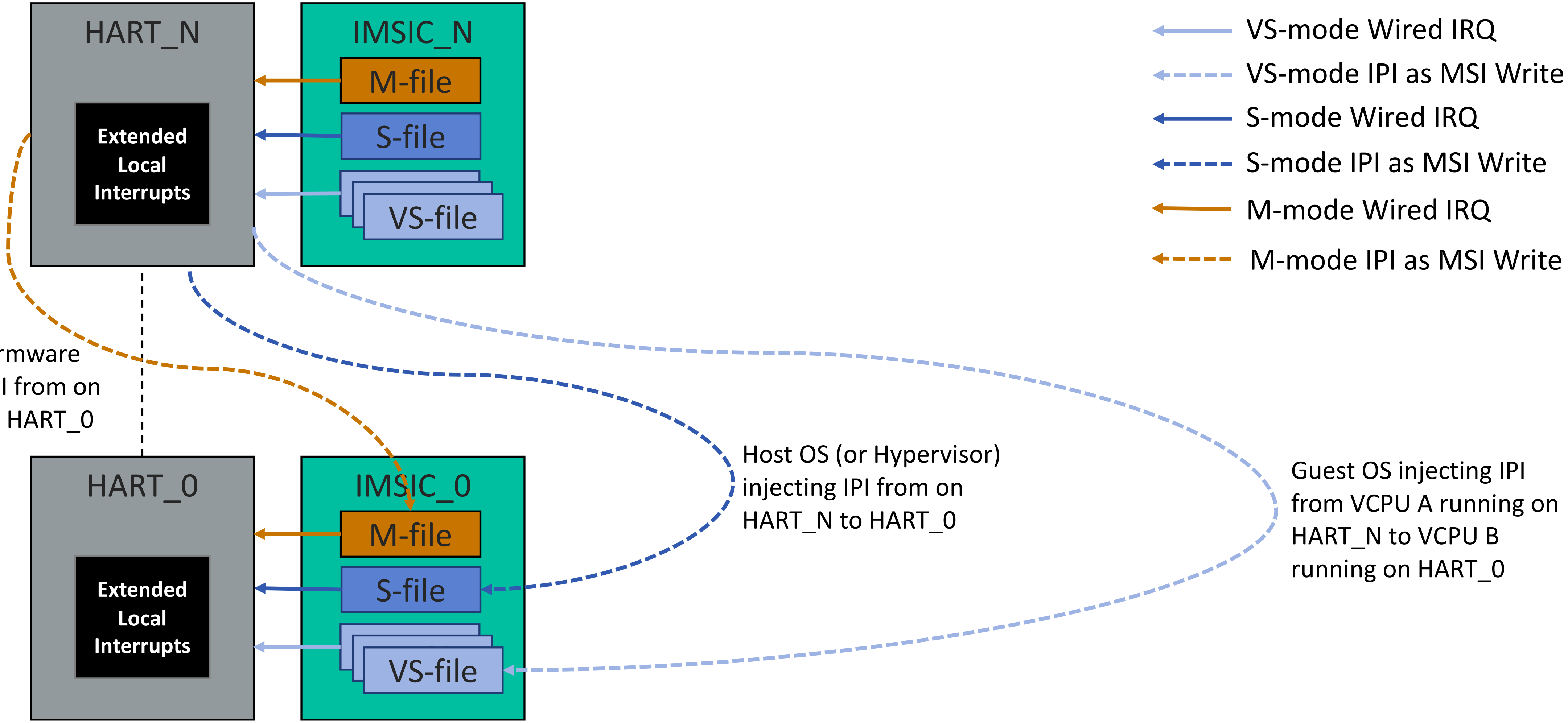
Components of AIA involved in handling MSIs



Western Digital.

AIA: IPIs as software injected MSIs

Components of AIA involved in providing IPIs as MSI writes



M-mode firmware injecting IPI from on HART_N to HART_0

Host OS (or Hypervisor) injecting IPI from on HART_N to HART_0

Guest OS injecting IPI from VCPU A running on HART_N to VCPU B running on HART_0

Western Digital.

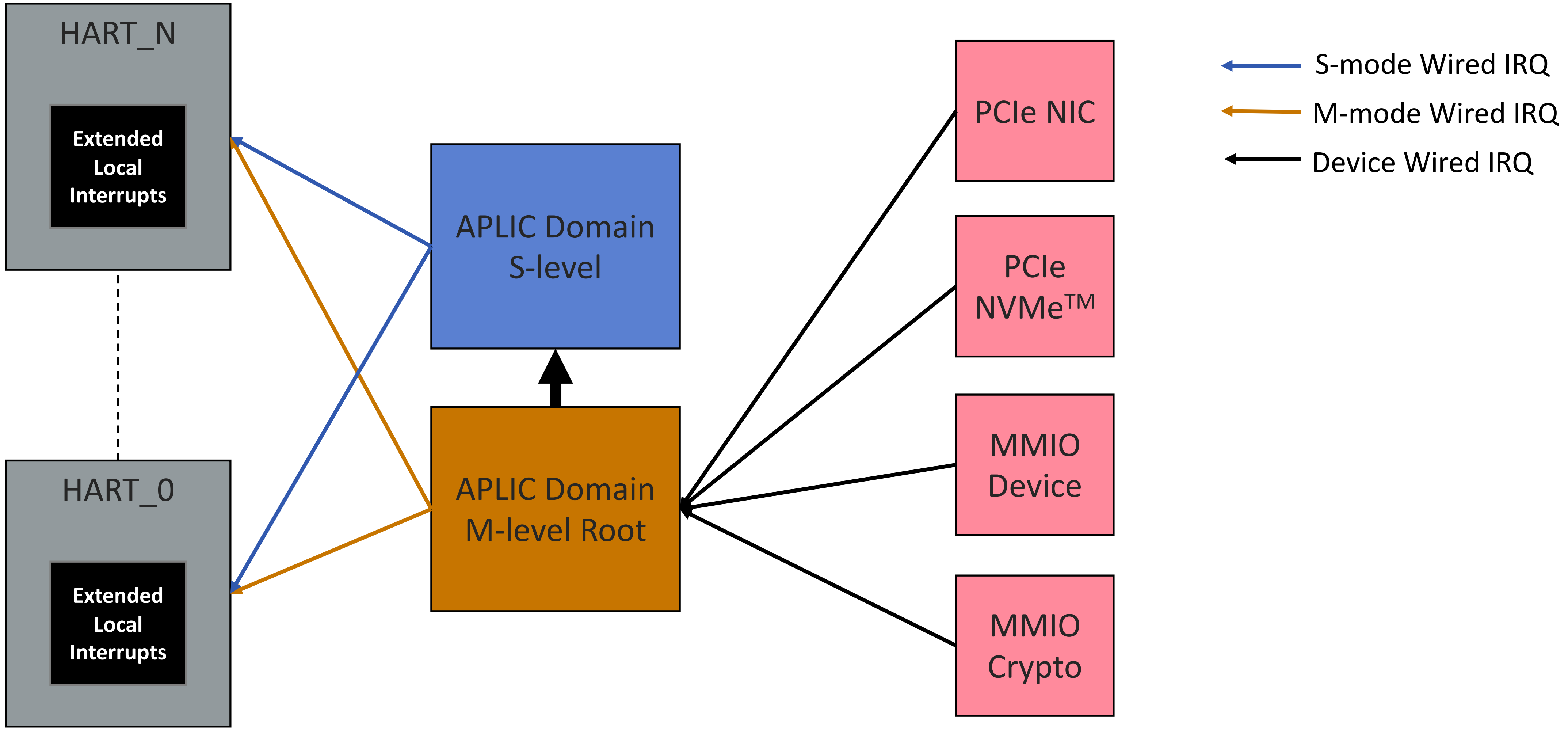
AIA: Wired Interrupts using APLIC

External wired interrupts using APLIC

- Hierarchical APLIC domains
  - Wired interrupts from devices only connect to **root APLIC domain**
  - Each APLIC domain targets a particular privilege level of associated HARTs
  - **An APLIC domain can delegate interrupts** to any of the child APLIC domains
- Configuration done via memory mapped registers (AIA CSRs are not required)
- Configurable line-sensing, priority and target HART for each interrupt source
- Support up to 1023 interrupt sources and up to 16384 HARTs
- Supports two modes:
  - **Direct mode:** Directly injecting external interrupt to associated HARTs
    - Each APLIC domain consumes physical address space between 16KB to 528KB
  - **MSI mode:** Forward wired interrupt as MSI to associated HARTs
    - Each APLIC domain consumes fixed physical address space of 16KB

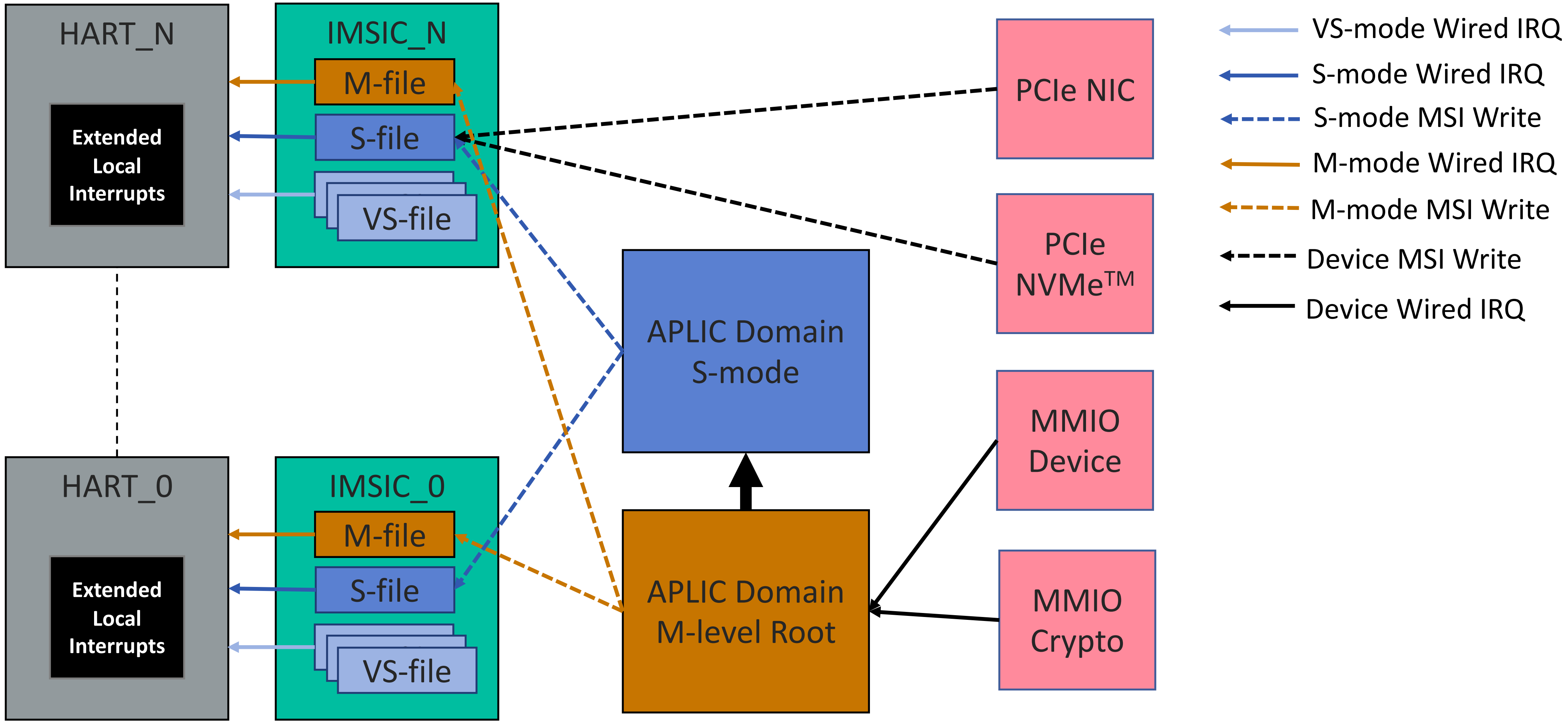
AIA: Wired interrupts using APLIC direct mode

Components of AIA involved in directly injecting wired IRQs



AIA: Wired interrupts using APLIC MSI mode

Components of AIA involved in forwarding wired IRQs as MSIs



Western Digital.

AIA: Virtualization Support

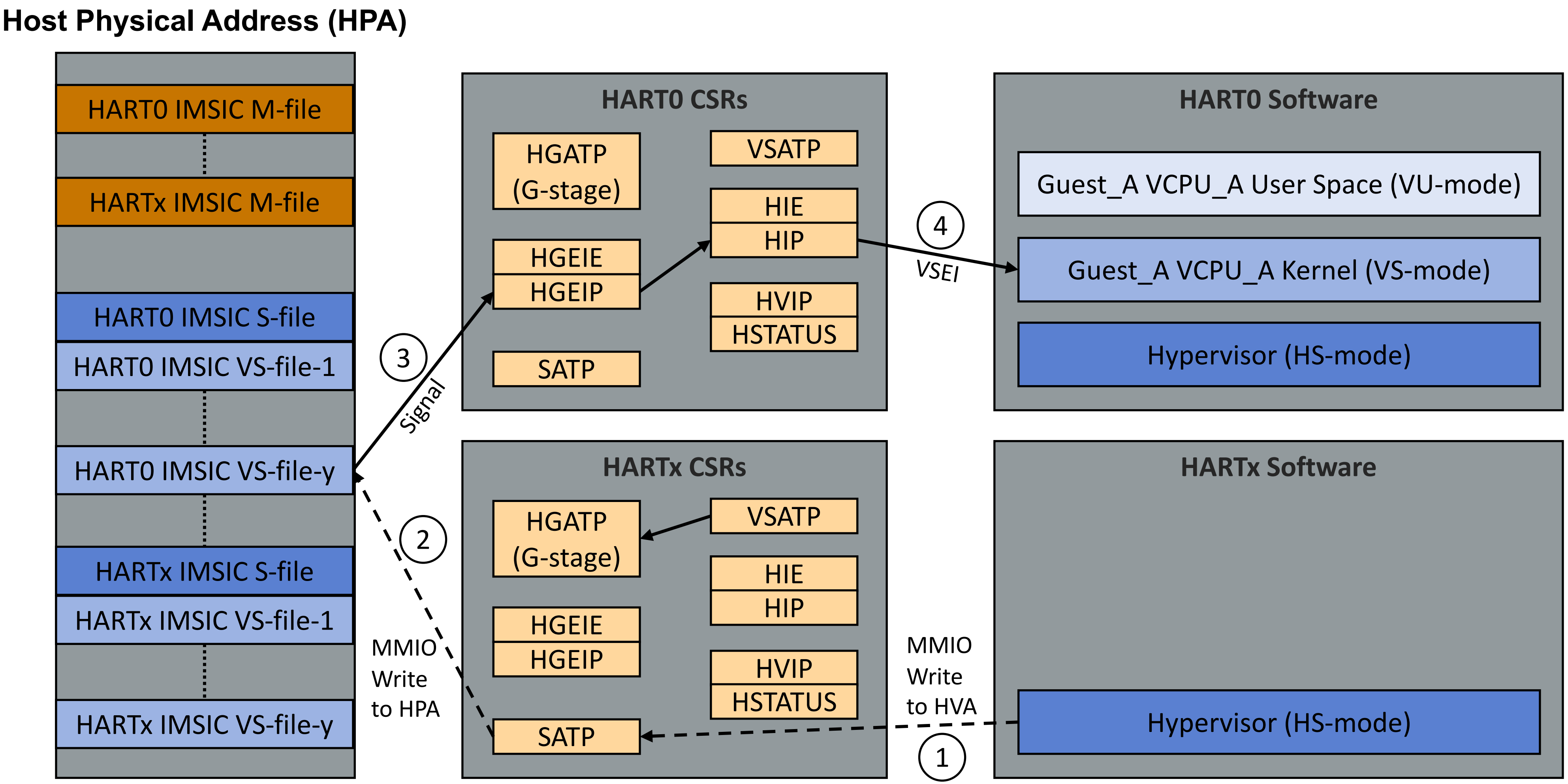
How is virtualization supported for different AIA components ?

- **AIA CSR virtualization**
  - Separate VS-mode CSRs for Guest/VM
  - Local interrupt priorities for VS-mode virtualized using hvictl, hviprio1 and hviprio2 CSRs
- **IMSIC virtualization**
  - Multiple guest-files (or VS-files) for each HART for virtualizing interrupt file for Guest/VM
  - VS-file assigned to a Guest VCPU is mapped in G-stage and selected using hstatus.VGEIN
  - Hypervisor can:
    - Inject emulated IRQs by writing to MMIO register of VS-file assigned to Guest VCPU
    - Route/forward device MSIs to MMIO register of VS-files using IOMMU
    - Take VS-file interrupt by setting appropriate bit in hgeie CSR with hie.SGEI == 1
- **APLIC only supports virtualization partly**
  - Hypervisor will trap-n-emulate all MMIO registers
  - But with MSI mode, there will be no MMIO traps at time of handling interrupts

AIA: Emulated IRQs using IMSIC VS-files

No extra traps involved in handling emulated IRQs using VS-files

Example: MSI from software emulated (or paravirt) device on HARTx targeting Guest_A VCPU_A using IMSIC VS-file-y on HART0

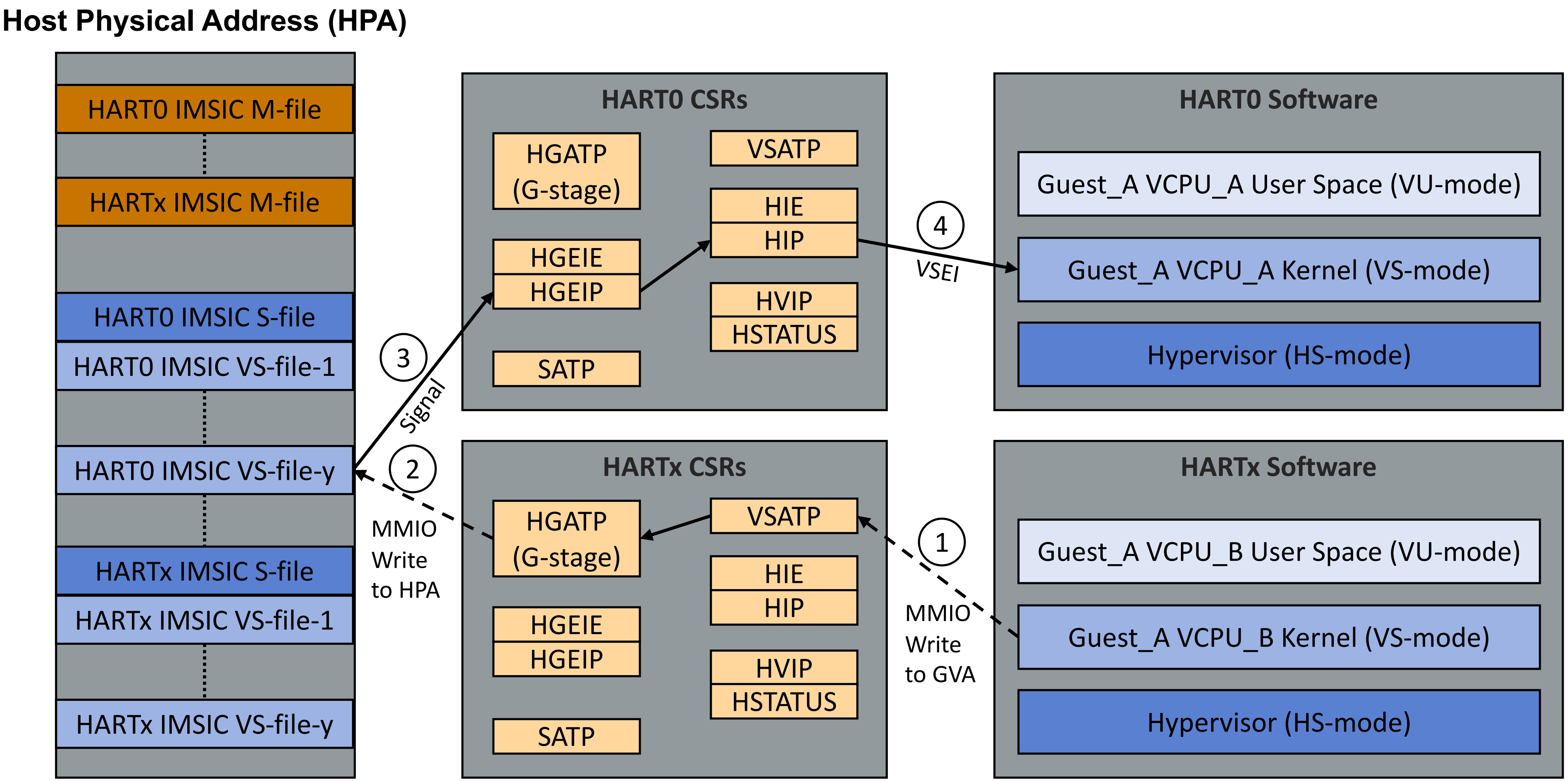


Western Digital.

AIA: Virtualized IPIs using IMSIC VS-files

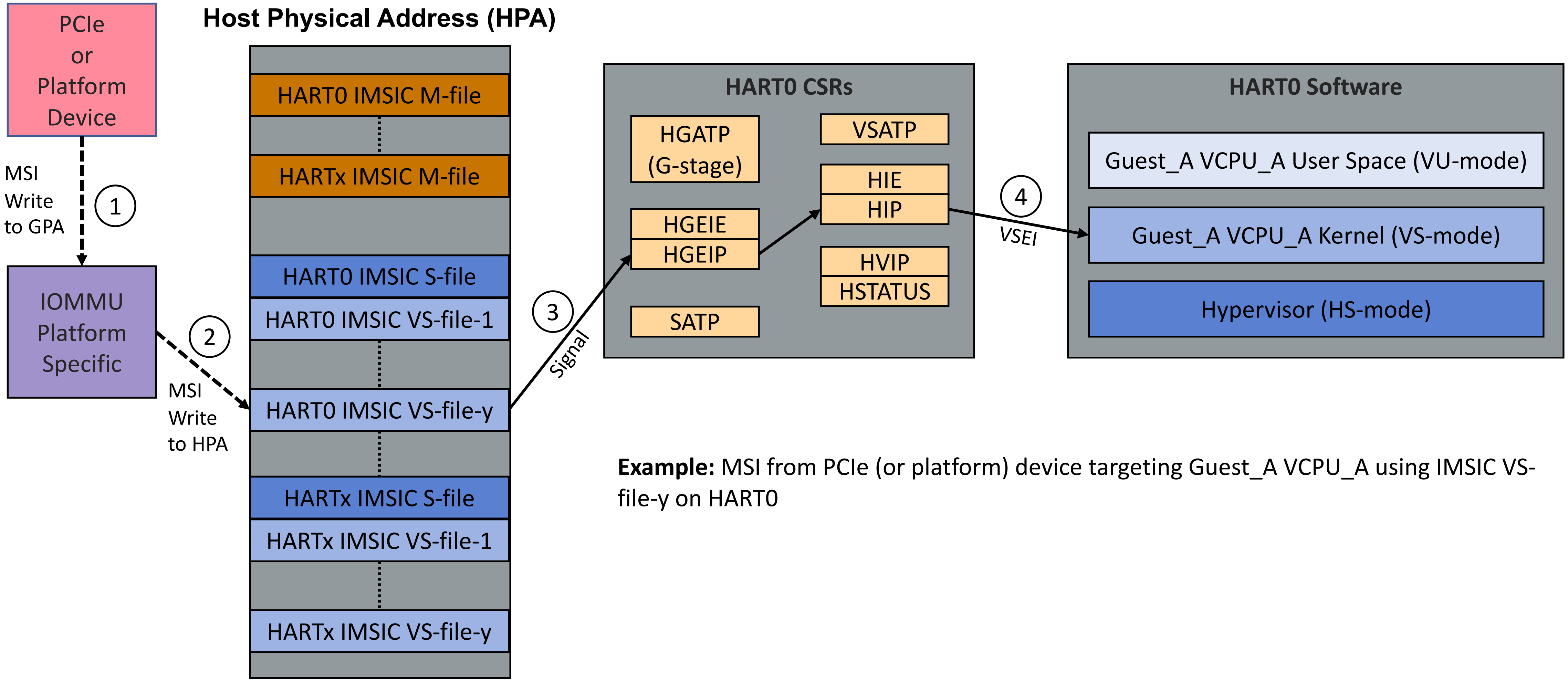
No extra traps involved in handing virtual IPIs using VS-files

Example: Virtual IPI from Guest_A VCPU_B on HARTx targeting Guest_A VCPU_A using IMSIC VS-file-y on HART0



AIA: Device MSIs using IMSIC VS-files

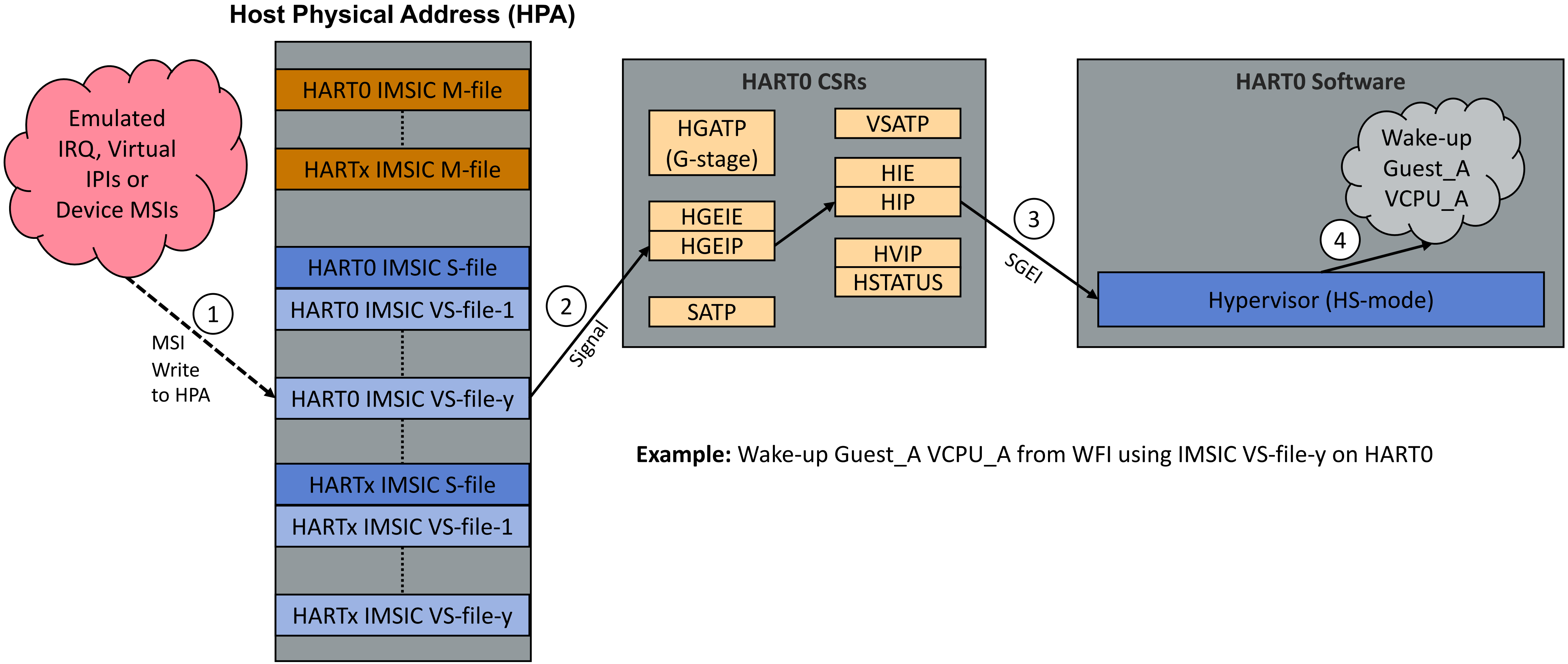
Direct injection of device MSIs using IMSIC VS-files



Example: MSI from PCIe (or platform) device targeting Guest_A VCPU_A using IMSIC VS-file-y on HART0

AIA: WFI wake-up using IMSIC VS-files

Waking-up Guest VCPU using IMSIC VS-files



Example: Wake-up Guest_A VCPU_A from WFI using IMSIC VS-file-y on HART0

Western Digital.

ACLINT Specification Overview

Motivation and overview of ACLINT specification

SiFive CLINT in existing platforms

Timer and IPIs using SiFive CLINT in existing platforms

- **SiFive Core-Local Interruptor (CLINT)**
  - https://static.dev.sifive.com/FU540-C000-v1.0.pdf
  - Specification owned by SiFive
  - Present on almost all existing RISC-V platforms
  - Provides M-mode timer (i.e. MTIME and MTIMECMP registers for each HART)
  - Provides M-mode software interrupts
- **Limitations:**
  - Does not allow RISC-V platforms to implement only M-mode timer
    - It is one composite device providing both M-mode timer and software interrupts
  - Does not allow multiple M-mode timers to share same MTIME register
    - Multi-cluster platforms will tend to have MTIMECMP registers local to each HART cluster
  - Does not provide S-mode software interrupts
    - S-mode software must use SBI IPI calls which adds overhead in IPI injection

RISC-V ACLINT Specification

Timer and IPIs using new RISC-V ACLINT specification

- **RISC-V Advanced Core-Local Interruptor (ACLINT)**
  - https://github.com/riscv/riscv-aclint/releases/download/v1.0-rc2/riscv-aclint-1.0-rc2.pdf
  - More modular and defines each functionality as a separate device
  - Specification is in stable state (no major changes anticipated)
- **Defines three separate devices**
  - MTIMER (M-mode timer)
  - MSWI (M-mode software interrupts)
  - SSWI (S-mode software interrupts)
- **Backward compatible with SiFive CLINT**
  - Offset range 0x0000 to 0x3FFF is ACLINT MSWI device
  - Offset range 0x4000 to 0xBFFF is ACLINT MTIMER device
- **Existing RISC-V platforms are already compatible with RISC-V ACLINT**

Interrupts & Timer in Future Platforms

How AIA & ACLINT will be used in future RISC-V platforms ?

AIA & ACLINT usage in Future platforms

Recommended ways of using AIA & ACLINT

RISC-V AIA Specification

RISC-V ACLINT Specification

RISC-V SBI Specification

RISC-V Privilege Specification

OS-A Platforms	MSIs			Wired Interrupts			IPIs			Timer		
	M-level	S-level	VS-level	M-level	S-level	VS-level	M-level	S-level	VS-level	M-level	S-level	VS-level
Legacy Wired IRQs	NA	NA	NA	PLIC	PLIC	PLIC (Emulate)	MSWI (CLINT)	SBI IPI	SBI IPI	MTIMER (CLINT)	SBI Timer	SBI Timer
Only Wired IRQs	NA	NA	NA	APLIC M-level	APLIC S-level	APLIC S-level (Emulate)	MSWI	SSWI	SBI IPI	MTIMER	Priv Sstc	Priv Sstc
MSIs and Wired IRQs	IMSIC M-file	IMSIC S-file	IMSIC S-file (Emulate)	APLIC M-level	APLIC S-level	APLIC S-level (Emulate)	IMSIC M-file	IMSIC S-file	SBI IPI	MTIMER	Priv Sstc	Priv Sstc
MSIs, Virtual MSIs and Wired IRQs	IMSIC M-file	IMSIC S-file	IMSIC VS-file	APLIC M-level	APLIC S-level	APLIC S-level (Emulate)	IMSIC M-file	IMSIC S-file	IMSIC VS-file	MTIMER	Priv Sstc	Priv Sstc

Software Status

Discussions for ACLINT and AIA software support

QEMU AIA & ACLINT support

How far did we get with AIA & ACLINT emulation in QEMU ?

- **QEMU ACLINT support already up-streamed**
- QEMU AIA patches already under review
  - Branch riscv_aia_v5 at https://github.com/avpatel/qemu
- QEMU virt machine will support all OS-A interrupt and timer configurations
  - **Legacy Wired IRQs (Default)**
    - Command-line "-M virt"
  - **Only Wired IRQs**
    - Command-line "-M virt,aclint=on,aia=aplic"
  - **MSIs and Wired IRQs**
    - Command-line "-M virt,aclint=on,aia=aplic-imsic"
  - **MSIs, Virtual MSIs, and Wired IRQs**
    - Command-line "-M virt,aclint=on,aia=aplic-imsic,aia-guests=3"

Western Digital.

AIA Software Status

How far did we get with AIA software support ?

- Complete proof-of-concept done (QEMU, OpenSBI, Linux, KVM, and KVMTOOL)
- Device tree bindings
  - Already reviewed on RISC-V AIA and platform mailing lists
  - Need to send RFC PATCHES for review on Linux mailing lists
- OpenSBI, Linux, KVM, and KVMTOOL patches yet to be sent for review
  - Branch riscv_aia_v1 at https://github.com/avpatel/opensbi (Firmware support)
  - Branch riscv_aia_v1 at https://github.com/avpatel/linux (Linux Driver support)
  - Branch riscv_kvm_aia_v1 at https://github.com/avpatel/linux (KVM support)
  - Branch riscv_aia_v1 at https://github.com/avpatel/kvmtool (KVM user-space support)

ACLINT Software Status

How far did we get with ACLINT software support ?

- Complete proof-of-concept done (QEMU, OpenSBI, and Linux)
- Device tree bindings (reviewed and finalized on Linux mailing list)
  - Two compatible strings will be required
    - **Implementation specific:** “<vendor>,<chip | family>-aclint-[mtimer | mswi | sswi]”
    - **Device specific:** “riscv,aclint-[mtimer | mswi | sswi]”
- OpenSBI ACLINT support already up-streamed
- Linux patches already in-review since past 6+ months
  - Branch riscv_aclint_v5 at https://github.com/avpatel/linux

Western Digital®

Western Digital and the Western Digital logo are registered trademarks or trademarks of Western Digital Corporation or its affiliates in the US and/or other countries.

The NVMe word mark is a trademark of NVM Express, Inc.

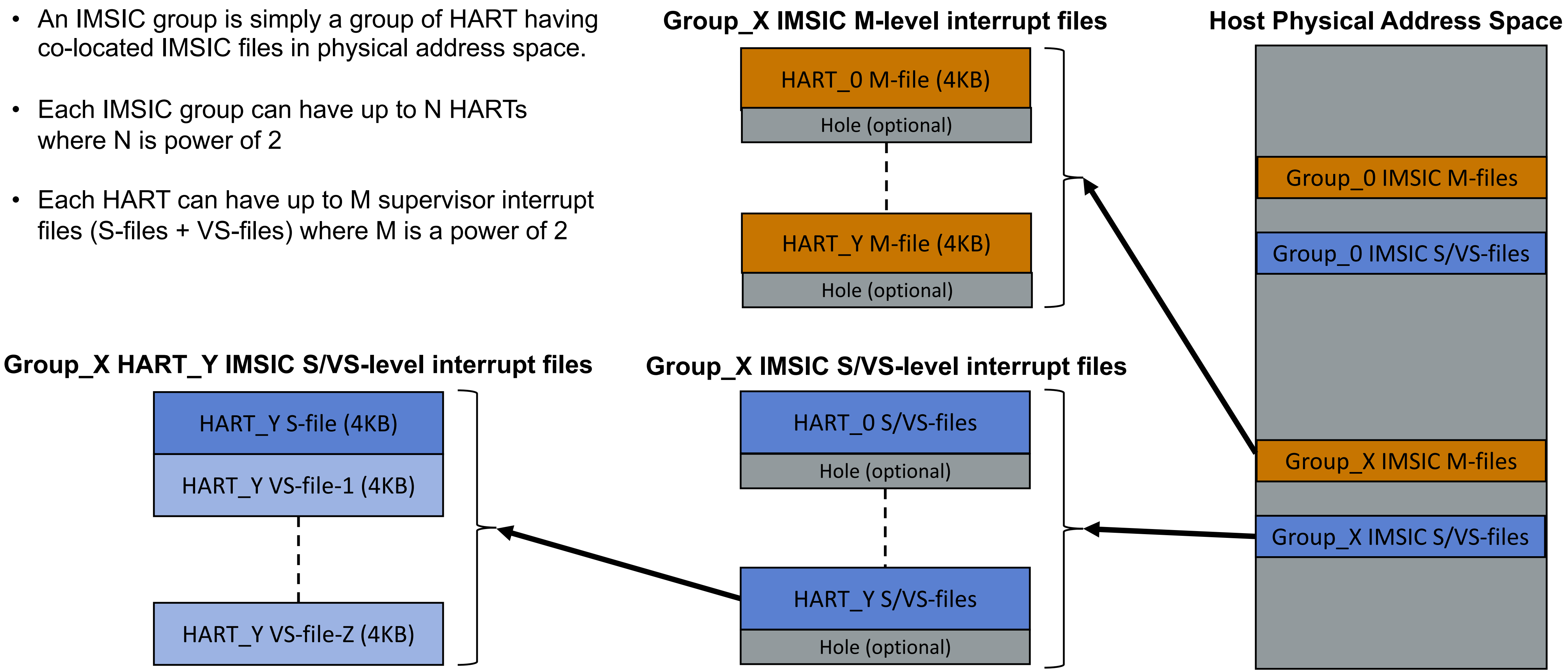
All other marks are the property of their respective owners.

Backup/Reference

AIA: IMSIC interrupt files arrangement

How are IMSIC files organized in physical address space ?

- An IMSIC group is simply a group of HART having co-located IMSIC files in physical address space.
- Each IMSIC group can have up to N HARTs where N is power of 2
- Each HART can have up to M supervisor interrupt files (S-files + VS-files) where M is a power of 2



Western Digital.