Next Generation RISC-V Interrupt Support

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

Outline

- AIA Specification Overview
- ACLINT Specification Overview
- AIA & ACLINT in Future Platforms
- Software Discussions

AIA Specification Overview

Motivation and high-level view of AIA specification

Western Digital.

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
    - S-mode can change priority and pending status of IRQs targeting M-mode
  - 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)**
  - https://github.com/riscv/riscv-aia/releases/download/0.2-draft.26/riscv-interrupts-026.pdf
  - Specification is stable (no anticipated changes) and will be ratified before RISC-V summit 2021
- **Defines three distinct components:**
  - Extended Local Interrupts (AIA CSRs)
  - Incoming Message Signaled Interrupt Controller (IMSIC)
  - Advanced Platform Level Interrupt Controller (APLIC)
- **RISC-V platform should implement only required RISC-V AIA components**
  - IMSIC requires AIA CSRs but APLIC is memory mapped so does not required AIA CSRs

RISC-V AIA specification (Contd.)

External interrupts using new RISC-V AIA specification

- **Extended Local Interrupts (AIA CSRs)**
  - Supports 64 local interrupts for both RV32 and RV64
  - Supports configurable priority for each local interrupt
  - Supports local interrupt filtering for M, S, and VS modes
  - Behavior of local interrupts 0 to 12 as defined by RISC-V privileged specification
  - Supports IMSIC CSRs for faster IMSIC configuration
- **Incoming Message Signaled Interrupt Controller (IMSIC)**
  - One IMSIC instance for each HART
  - Each IMSIC instance consist of multiple interrupt files
    - **One M-file** (M-level interrupt file), **one S-file** (S-level interrupt file), and **multiple VS-file** (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

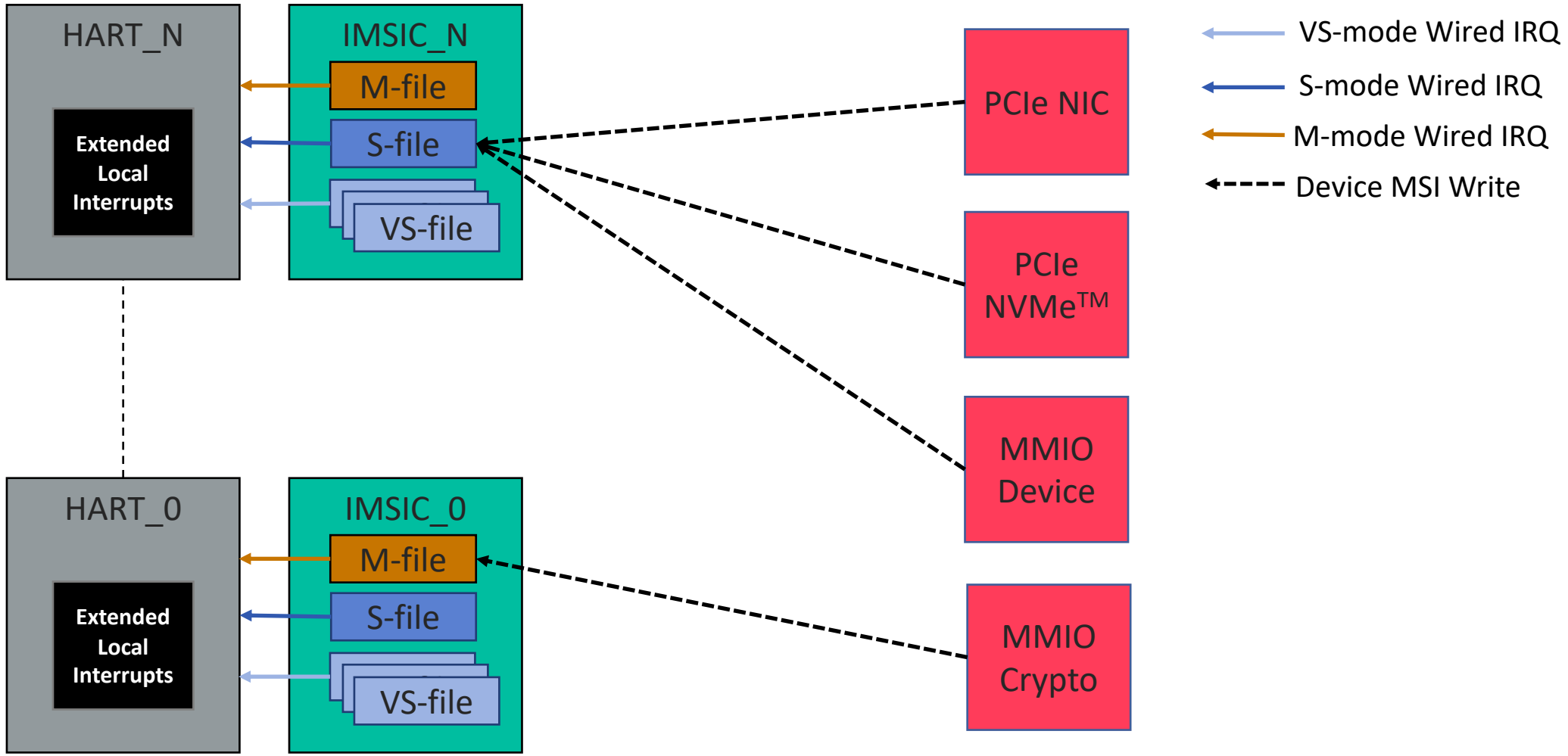
RISC-V AIA specification (Contd.)

External interrupts using new RISC-V AIA specification

- **Advanced Platform Level Interrupt Controller (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 and priority for each wired interrupt
  - Separate target register for each wired interrupt
  - 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:** Forwarding wired interrupt as MSI to associated HARTs
      - Each APLIC domain consumes fixed physical address space of 16KB

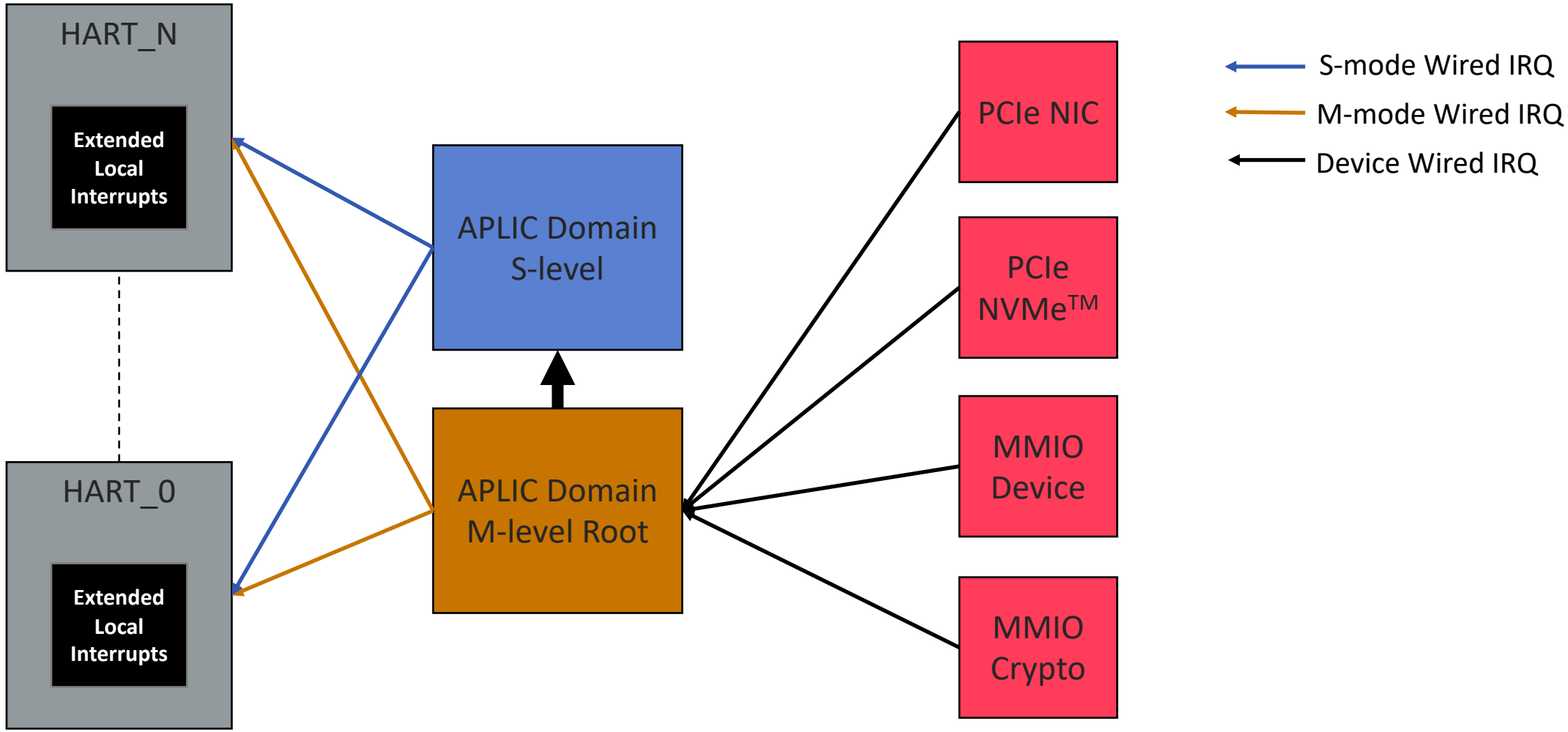
AIA: Only MSI external interrupts

Components of AIA involved in handling MSIs



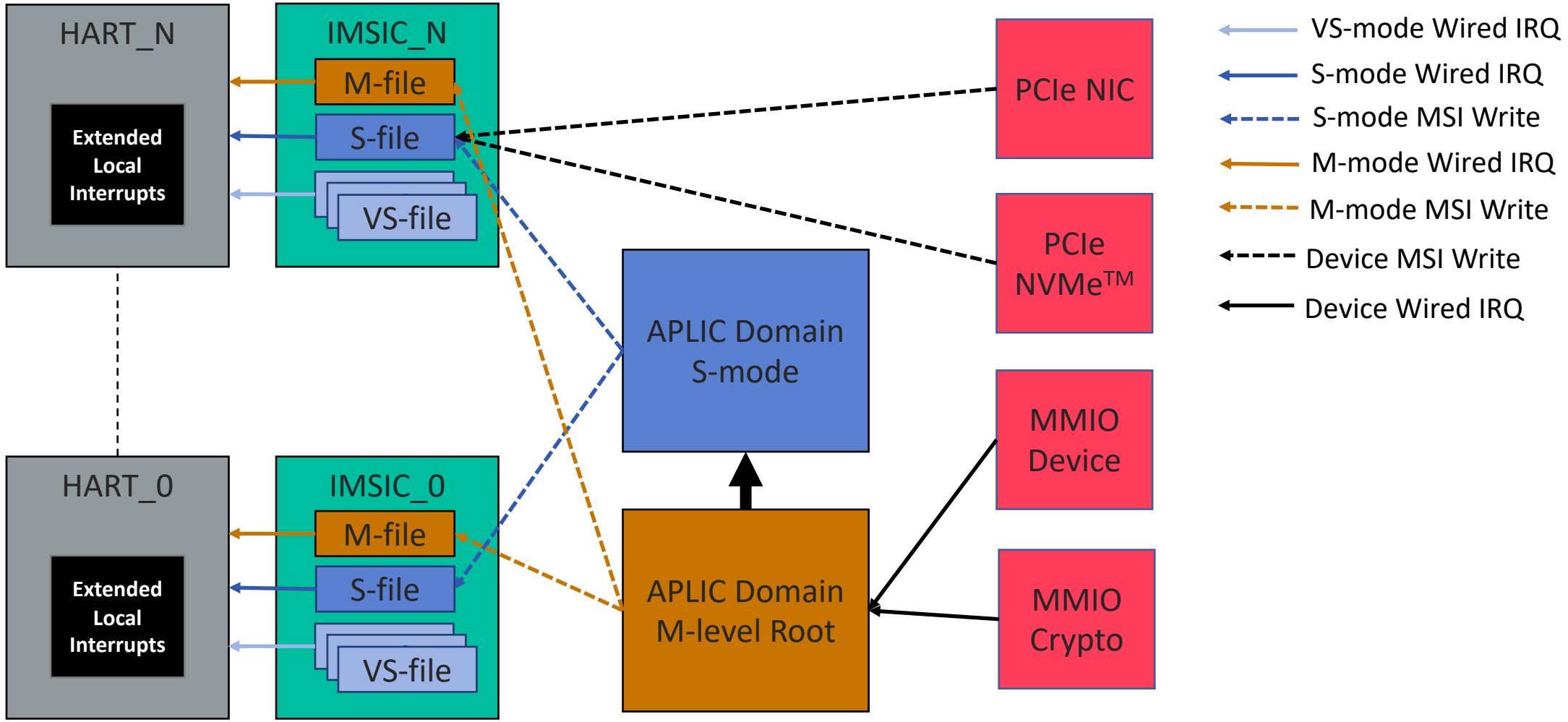
AIA: Only wired external interrupts

Components of AIA involved in handling wired IRQs



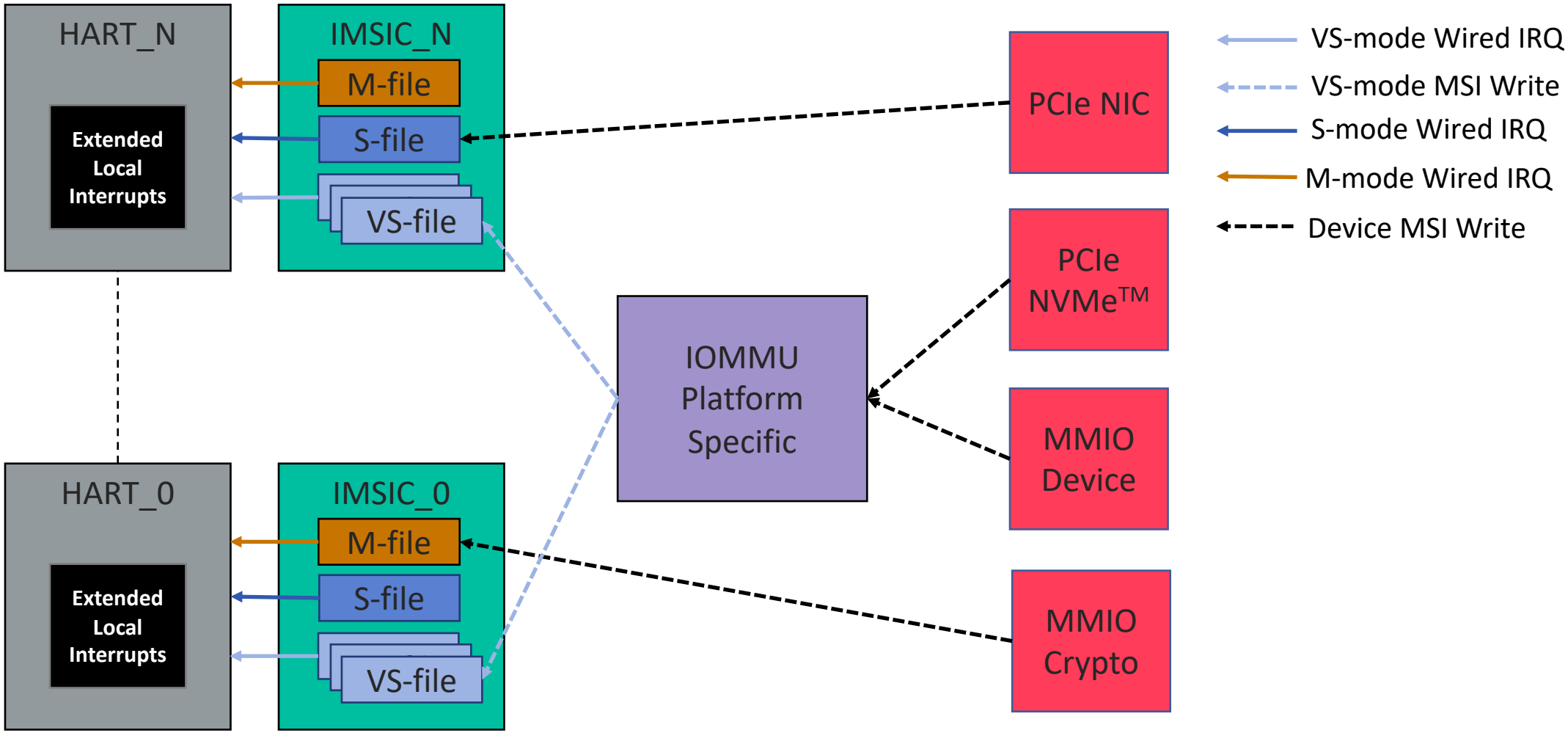
AIA: Both wired and MSI external interrupts

Components of AIA involved in handling both wired IRQs & MSIs



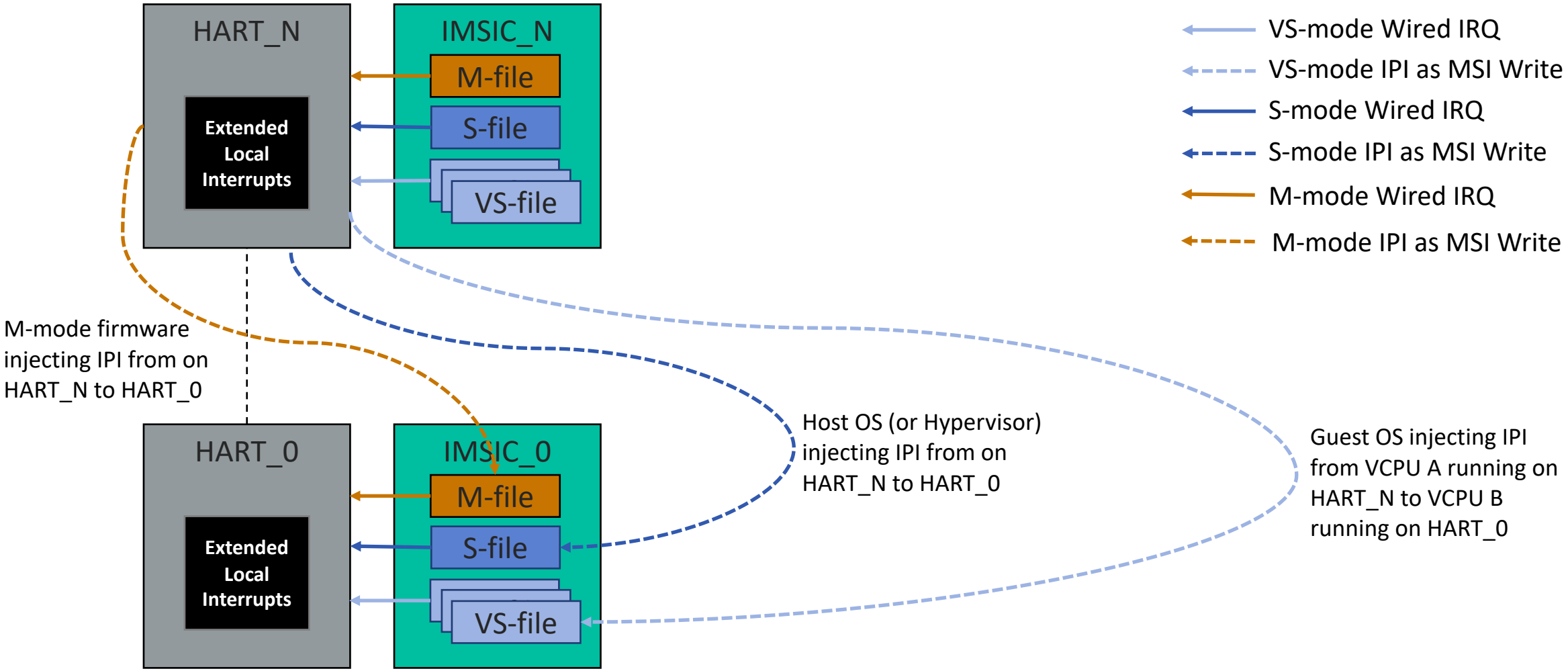
AIA: Device MSIs forwarded to Guest/VM

Components of AIA involved in forwarding MSIs to Guest/VM



AIA: IPIs as software injected MSIs

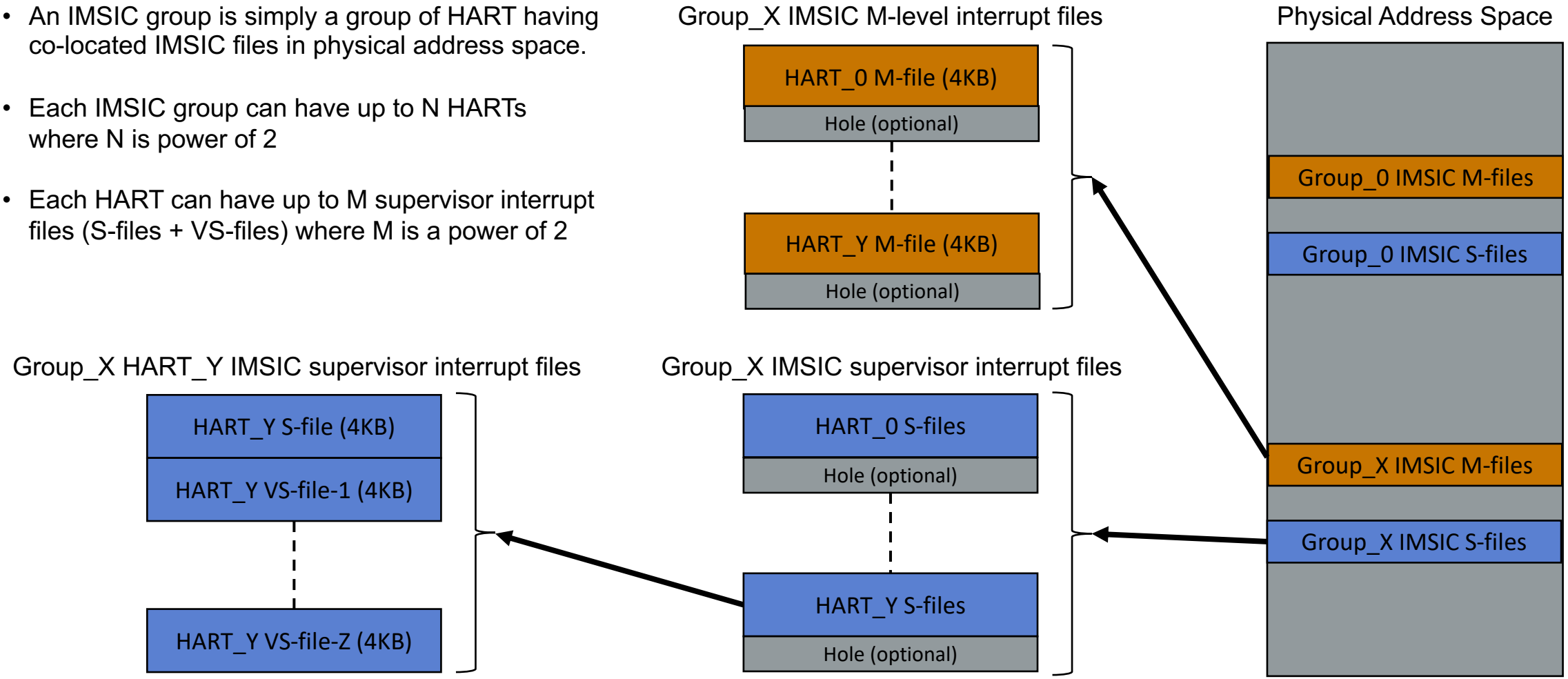
Components of AIA involved in providing IPIs as MSI writes



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



ACLINT Specification Overview

Motivation and high-level view of ACLINT specification

Western Digital.

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-rc1/riscv-aclint-1.0-rc1.pdf
  - Specification is stable (no changes anticipated) and will be ratified before RISC-V summit 2021
  - More modular and defines each functionality as a separate device
  - Defines a separate device for S-mode software interrupts
- **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**

AIA & ACLINT in Future Platforms

How will future RISC-V platforms use AIA and ACLINT ?

Western Digital.

AIA & ACLINT for OS-A platforms

Possible uses of AIA and ACLINT in OS-A platforms

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) Phase1	**SBI IPI**	**SBI IPI**	**MTIMER** (CLINT) Phase1	**SBI Timer**	**SBI Timer**
Only Wired IRQs	NA	NA	NA	**APLIC M-level** Phase1	**APLIC S-level** Phase1	**APLIC S-level** (Emulate) Phase2	**MSWI** Phase1	**SSWI** Phase1	**SBI IPI**	**MTIMER** Phase1	**Priv Sstc** In-progress	**Priv Sstc** In-progress
MSIs and Wired IRQs	**IMSIC M-file** Phase1	**IMSIC S-file** Phase1	**IMSIC S-file** (Emulate) Phase2	**APLIC M-level** Phase1	**APLIC S-level** Phase1	**APLIC S-level** (Emulate) Phase2	**IMSIC M-file** Phase1	**IMSIC S-file** Phase1	**SBI IPI**	**MTIMER** Phase1	**Priv Sstc** In-progress	**Priv Sstc** In-progress
MSIs, Virtual MSIs and Wired IRQs	**IMSIC M-file** Phase1	**IMSIC S-file** Phase1	**IMSIC VS-file** Phase2	**APLIC M-level** Phase1	**APLIC S-level** Phase1	**APLIC S-level** (Emulate) Phase2	**IMSIC M-file** Phase1	**IMSIC S-file** Phase1	**IMSIC VS-file** Phase2	**MTIMER** Phase1	**Priv Sstc** In-progress	**Priv Sstc** In-progress

Software Discussions

Discussions for ACLINT and AIA software support

Western Digital.

QEMU AIA & ACLINT support

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

- QEMU ACLINT patches are reviewed and waiting to be picked by QEMU maintainer
- QEMU AIA patches already under review
  - Branch riscv_aia_v2 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”

ACLINT Software Status

How far did we get with ACLINT software support ?

- Complete proof-of-concept done (QEMU, OpenSBI, and Linux)
- Device tree bindings
  - Addressed most comments from DT-bindings maintainer (Rob Herring)
  - Two compatible strings will be required
    - **Implementation specific:** “<vendor>,<chip | family>-aclint-[mtimer | mswi | sswi]”
    - **Device specific:** “riscv,aclint-[mtimer | mswi | sswi]”
  - Implementation specific compatible string to be used for detecting work-arounds
    - Example: Some MTIMER implementations might not support 64-bit MMIO accesses
- OpenSBI patches already up-streamed
- Linux patches
  - Addressed all comments from Linux IRQ maintainer (Marc Z)

Device Tree: ACLINT MTIMER device

Representing ACLINT MTIMER in device tree

Single MTIMER Device

```
…
mtimer0: mtimer@30000000 {
                compatible = "riscv,aclint-mtimer";
                reg = <0x30000000 0x8>,      /* MTIME */
                      <0x30008000 0x20>;    /* MTIMECMPs */
                interrupts-extended = <&cpu1_intc 7>,
                                      <&cpu2_intc 7>,
                                      <&cpu3_intc 7>,
                                      <&cpu4_intc 7>;
};
…
```

Two MTIMER Devices Sharing Same MTIME

```
…
mtimer0: mtimer@30000000 {
                compatible = "riscv,aclint-mtimer";
                reg = <0x3a000000 0x8>,      /* MTIME */
                      <0x30000000 0x20>;    /* MTIMECMPs */
                interrupts-extended = <&cpu1_intc 7>,
                                      <&cpu2_intc 7>,
                                      <&cpu3_intc 7>,
                                      <&cpu4_intc 7>;
};
mtimer1: mtimer@31000000 {
                compatible = "riscv,aclint-mtimer";
                reg = <0x3a000000 0x8>,      /* MTIME */
                      <0x31000000 0x20>;    /* MTIMECMPs */
                interrupts-extended = <&cpu5_intc 7>,
                                      <&cpu6_intc 7>,
                                      <&cpu7_intc 7>,
                                      <&cpu8_intc 7>;
};
…
```

Device Tree: ACLINT MSWI and SSWI devices

Representing ACLINT MSWI and SSWI in device tree

Single MSWI Device

```
…
mswi0: mswi@60000000 {
                compatible = "riscv,aclint-mswi";
                reg = <0x60000000 0x10>;
                interrupts-extended = <&cpu1_intc 3>,
                                      <&cpu2_intc 3>,
                                      <&cpu3_intc 3>,
                                      <&cpu4_intc 3>;
                interrupt-controller;
};
…
```

Single SSWI Device

```
…
sswi0: sswi@70000000 {
                compatible = "riscv,aclint-sswi";
                reg = <0x70000000 0x10>;
                interrupts-extended = <&cpu1_intc 1>,
                                      <&cpu2_intc 1>,
                                      <&cpu3_intc 1>,
                                      <&cpu4_intc 1>;
                interrupt-controller;
};
…
```

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 from DT-bindings maintainer (Rob Herring)
- Linux, KVM, and KVMTOOL patches yet to be sent for review
  - Branch riscv_aia_v1 at https://github.com/avpatel/linux
  - Branch riscv_kvm_aia_v1 at https://github.com/avpatel/linux (Blocked on KVM RISC-V patches)
  - Branch riscv_aia_v1 at https://github.com/avpatel/kvmtool (Blocked on KVM RISC-V patches)

Device Tree: AIA Local Interrupts support

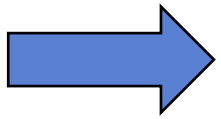
Representing Hart level AIA support in device tree

- A new compatible string for using AIA CSRs in RISC-V INTC driver
  - This only includes non-IMSIC CSRs because IMSC is optional

HART without AIA

```
…
cpu1: cpu@1 {
        compatible = "riscv";
        device_type = "cpu";
        reg = <1>;
        riscv,isa = "rv64imafdc";
        mmu-type = "riscv,sv39";

        cpu1_intc: interrupt-controller {
                #interrupt-cells = <1>;
                compatible = "riscv,cpu-intc";
                interrupt-controller;
        };
};
…
```



HART with AIA

```
…
cpu1: cpu@1 {
        compatible = "riscv";
        device_type = "cpu";
        reg = <1>;
        riscv,isa = "rv64imafdc";
        mmu-type = "riscv,sv39";

        cpu1_intc: interrupt-controller {
                #interrupt-cells = <1>;
                compatible = "riscv,cpu-intc-aia“,
                             "riscv,cpu-intc";
                interrupt-controller;
        };
};
…
```

Device Tree: AIA IMSIC support

Representing IMSIC support in device tree

- IMSIC files are follow IMSIC addressing scheme
  - One DT node for IMSIC files at M-level
  - One DT node for IMSIC files at supervisor level (S & VS-level)
- IMSIC DT node properties:
  - compatible
  - reg (One regset for each HART group)
  - interrupts-extended (Mapping interrupt files to HART)
  - imsic,ipi-range (optional)
  - imsic,guest-index-bits (0 <= x < 8) (optional)
  - imsic,hart-index-bits (0 <= x < 16) (optional)
  - imsic,group-index-shift (24 <= x < 55) (optional)
  - imsic,group-index-bits (0 <= x < 8) (optional)
  - imsic,num-ids (One less than multiple of 64) (63 <= x <= 2047)
  - interrupt-controller
  - msi-controller

```
imsics_mlevel: imsics@10000000 {
                compatible = "riscv,imsics";
                reg = <0x10000000 0x5000>;
                interrupts-extended = <&cpu0_intc 11>,
                                      <&cpu1_intc 11>, <&cpu2_intc 11>,
                                      <&cpu3_intc 11>, <&cpu4_intc 11>;
                imsic,ipi-range = <1 7>;
                imsic,num-ids = <127>;
                interrupt-controller;
                msi-controller;
};
…
imsics_slevel: imsics@11000000 {
                compatible = "riscv,imsics";
                reg = <0x11000000 0x4000>;
                interrupts-extended = <&cpu1_intc 9>,
                                      <&cpu2_intc 9>, <&cpu3_intc 9>,
                                      <&cpu4_intc 9>;
                imsic,ipi-range = <1 7>;
                imsic,num-ids = <127>;
                interrupt-controller;
                msi-controller;
};
…
dev@20000000 {
                compatible = “vendor,some-device”;
                reg = <0x20000000 0x1000>;
                msi-parent = <& imsics_slevel>;
};
```

Device Tree: APLIC without IMSIC support

Representing APLIC without IMSIC support in device tree

- Multiple APLIC instances on NUMA systems
  - Each APLIC instance has multiple domains
  - Mandatory one DT node for each APLIC domain
- Mandatory DT properties of APLIC DT node:
  - compatible
  - reg
  - interrupts-extended
    - Number of entries same as number of IDC structures
    - Mapping of each IDC structure to HART privilege-mode
  - aplic,num-sources
    - Minimum 1 and maximum 1023
  - interrupt-controller
  - #interrupt-cells
    - should be 2

```
aplic0: aplic@40000000 {
                compatible = "riscv,aplic";
                reg = <0x40000000 0x8000>;
                interrupts-extended = <&cpu1_intc 11>, <&cpu2_intc 11>,
                                      <&cpu3_intc 11>, <&cpu4_intc 11>;
                aplic,num-sources = <32>;
                interrupt-controller;
                interrupt-cells = <2>;
                aplic-children = <&aplic1>, <&aplic2>; /* OpenSBI (Optional) */
                aplic-delegate = <0x0 0x1f &aplic1>; /* OpenSBI (Optional) */
};
aplic1: aplic@41000000 {
                compatible = "riscv,aplic";
                reg = <0x41000000 0x8000>;
                interrupts-extended = <&cpu1_intc 9>, <&cpu2_intc 9>,
                                      <&cpu3_intc 9>, <&cpu4_intc 9>;
                aplic,num-sources = <32>;
                interrupt-controller;
                interrupt-cells = <2>;
};
…
dev@20000000 {
                compatible = “vendor,some-device”;
                reg = <0x20000000 0x1000>;
                interrupts-extended = <&aplic1 23 0x4>, <&aplic1 33 0x4>;
};
```

Device Tree: APLIC with IMSIC support

Representing APLIC with IMSIC support in device tree

- Multiple APLIC instances on NUMA systems
  - Each APLIC instance has multiple domains
  - Mandatory one DT node for each APLIC domain
- Mandatory DT properties of APLIC DT node:
  - compatible
  - reg
  - msi-parent
    - IMSIC instance DT node handling MSIs from APLIC
  - aplic,num-sources
    - Minimum 1 and maximum 1023
  - interrupt-controller
  - #interrupt-cells
    - should be 2

```
imsics_slevel: imsics@11000000 {
                compatible = "riscv,imsics";
                reg = <0x11000000 0x4000>;
                interrupts-extended = <&cpu1_intc 9>, <&cpu2_intc 9>,
                                      <&cpu3_intc 9>, <&cpu4_intc 9>;
                imsic,ipi-range = <1 7>;
                imsic,num-ids = <127>;
                interrupt-controller;
                msi-controller;
};
…
aplic1: aplic@41000000 {
                compatible = "riscv,aplic";
                reg = <0x41000000 0x8000>;
                msi-parent = <&imsics_slevel>;
                aplic,num-sources = <32>;
                interrupt-controller;
                interrupt-cells = <2>;
};
…
dev@20000000 {
                compatible = “vendor,some-device”;
                reg = <0x20000000 0x1000>;
                interrupts-extended = <&aplic1 23 0x4>, <&aplic1 33 0x4>;
};
```

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

Additional slides

Western Digital.

AIA: APLIC target MSI address generation

How does APLIC generate target MSI address ?

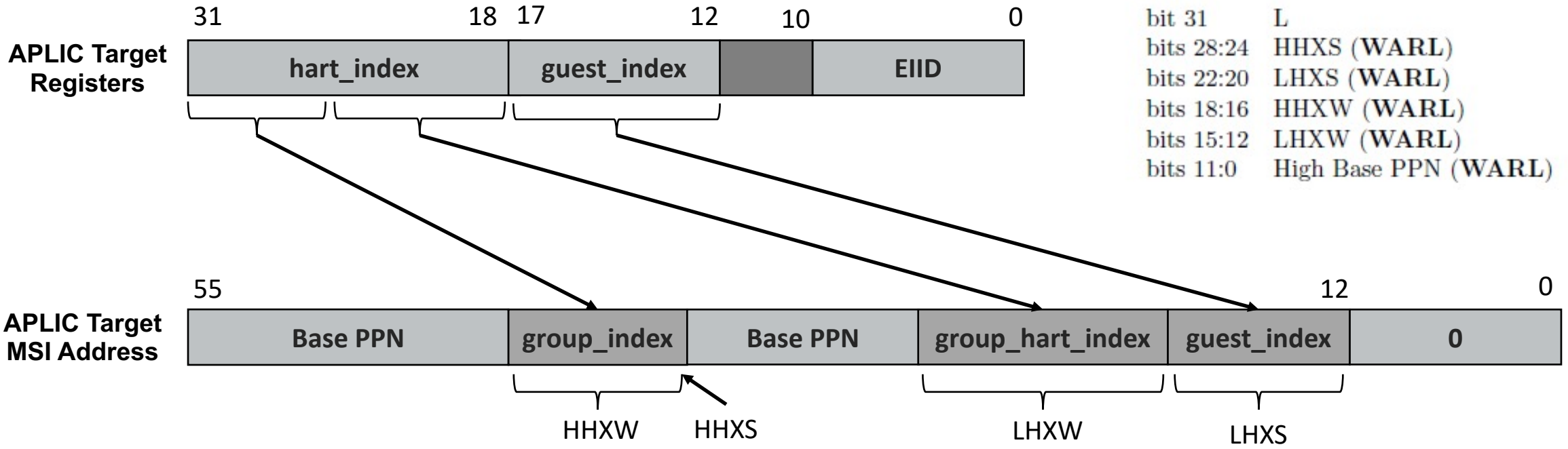
$g = (\text{machine-level hart index} >> \text{LHXW})\ \&\ (2^{\text{HHXW}} - 1)$
$h = \text{machine-level hart index}\ \&\ (2^{\text{LHXW}} - 1)$
$\text{MSI address} = \big(\text{Base PPN} \mid (g << (\text{HHXS} + 12)) \mid (h << \text{LHXS}) \mid \text{Guest Index}\big) << 12$

When implemented, `smsiaddrcfg` has this format:

bits 31:0	Low Base PPN (**WARL**)

and `smsiaddrcfgh` has this format:

bit 31	L
bits 28:24	HHXS (**WARL**)
bits 22:20	LHXS (**WARL**)
bits 18:16	HHXW (**WARL**)
bits 15:12	LHXW (**WARL**)
bits 11:0	High Base PPN (**WARL**)



KVM in-kernel irqchip support

How will KVM support AIA for Guest/VM ?

- In-kernel AIA irqchip is **an optional feature** of KVM RISC-V
  - Desired IRQCHIP can be emulated entirely in KVM user space
- In-kernel AIA irqchip **consist of:**
  - An optional APLIC with only MSI delivery mode
  - One IMSIC file for each Guest VCPU
- APLIC is **always trap-n-emulated** in software
  - No traps from Guest/VM while handling interrupts due to MSI delivery mode
- IMSIC can be **trap-n-emulated or virtualized by hardware**
  - Only stopei CSR traps for Guest/VM while handling interrupts
  - Zero traps (or vmexits) when Guest/VM uses IMSIC VS-file

KVM in-kernel irqchip support (Contd.)

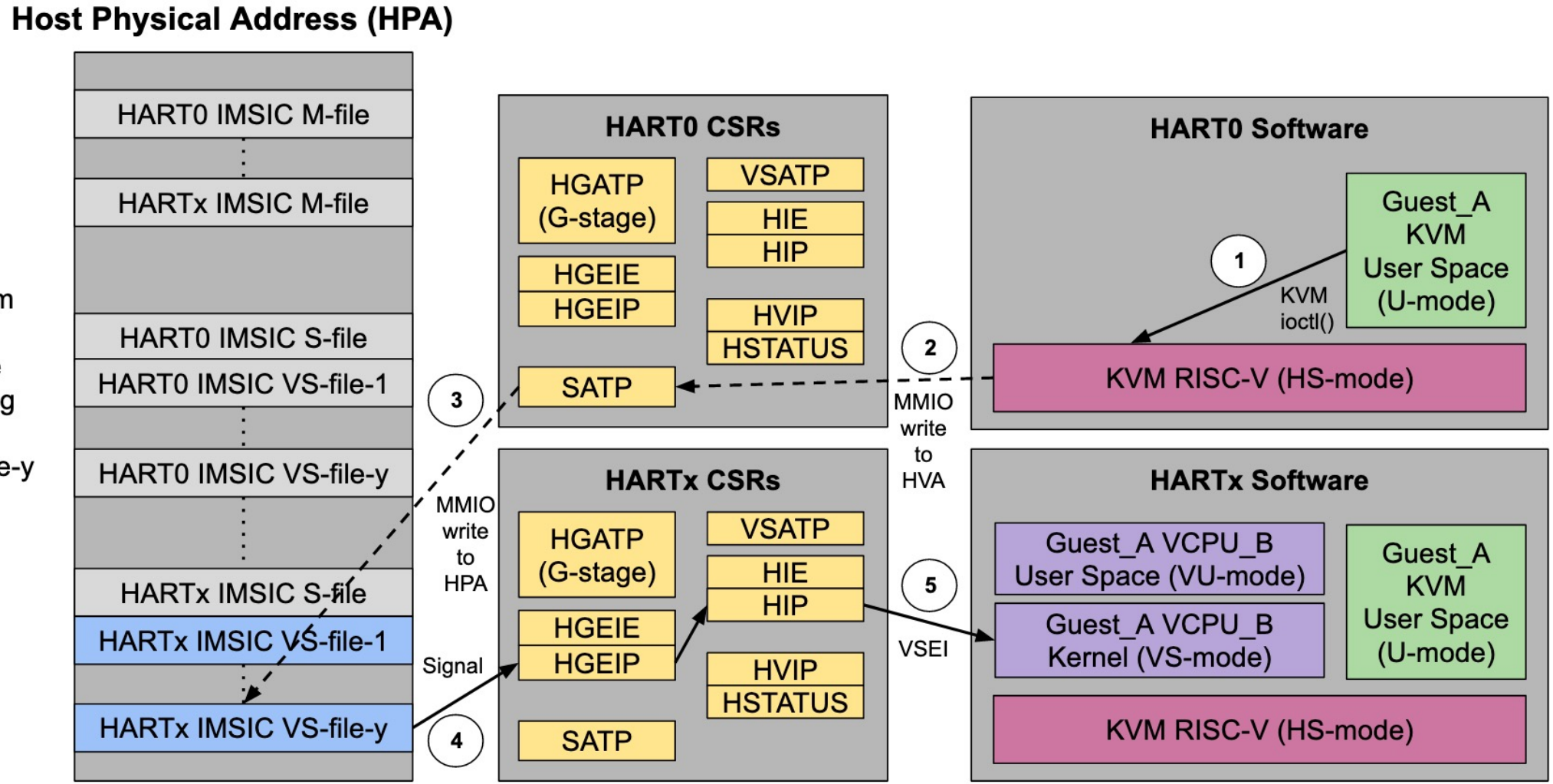
How will KVM support AIA for Guest/VM ?

- At any point in time, a Guest VCPU uses to one of the following:
  - **IMSIC SW-file** (trap-n-emulated by software)
  - **IMSIC VS-file** (virtualized by hardware)
- **Guest VCPU changes IMSIC file when migrated to new HART**
- Three modes of operation for in-kernel AIA irqchip:
  - Emulation (**EMUL**)
    - Always use IMSIC SW-file (i.e. trap-n-emulate) for each Guest VCPU
  - HW Acceleration (**HWACCEL**)
    - Always use of IMSIC VS-file (i.e. hardware virtualization) for each Guest VCPU
    - Only available when underlying host has VS-files in IMSIC of each HART
  - Automatic (**AUTO**)
    - Use IMSIC VS-file for Guest VCPU when available otherwise use IMSIC SW-file
    - Only available when underlying host has VS-files in IMSIC of each HART

KVM: Emulated IRQs using IMSIC VS-files

Events involved in injecting emulated IRQs using IMSIC VS-files

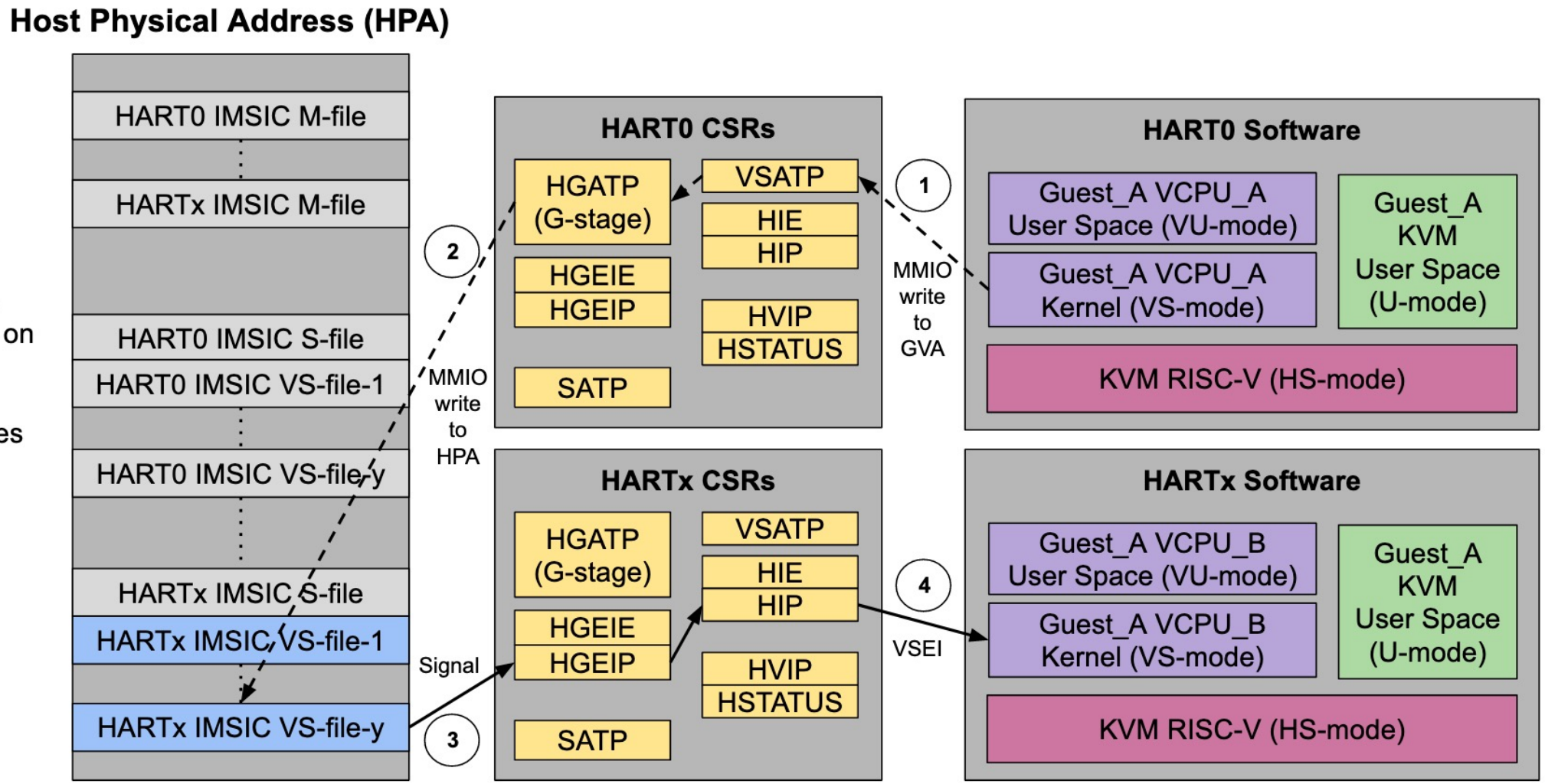
Example: MSI from software emulated (or paravirt) device on HART0 targeting Guest_A VCPU_B using IMSIC VS-file-y on HARTx



KVM: IPIs using IMSIC VS-files

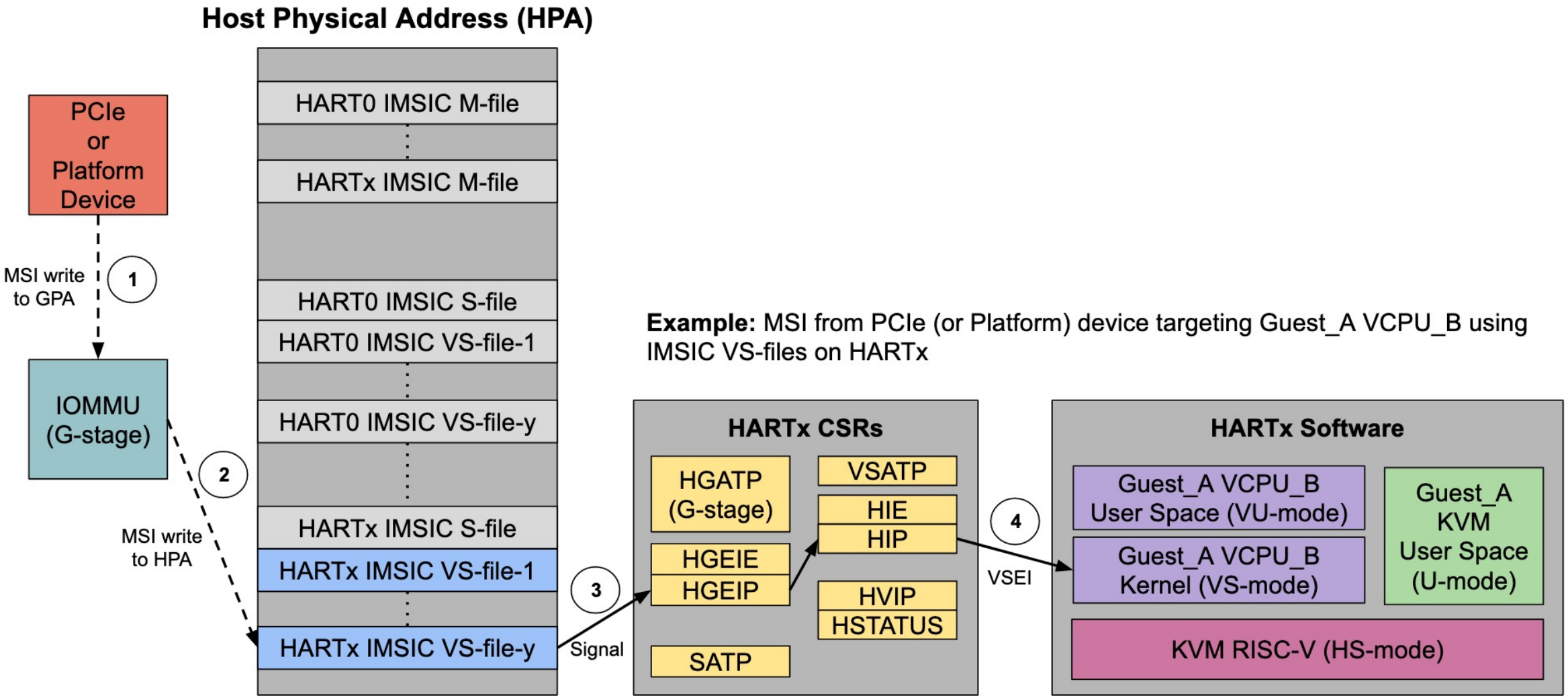
Events involved in injecting IPIs using IMSIC VS-files

Example: IPI from Guest_A VCPU_A on HART0 targeting Guest_A VCPU_B using IMSIC VS-files on HARTx



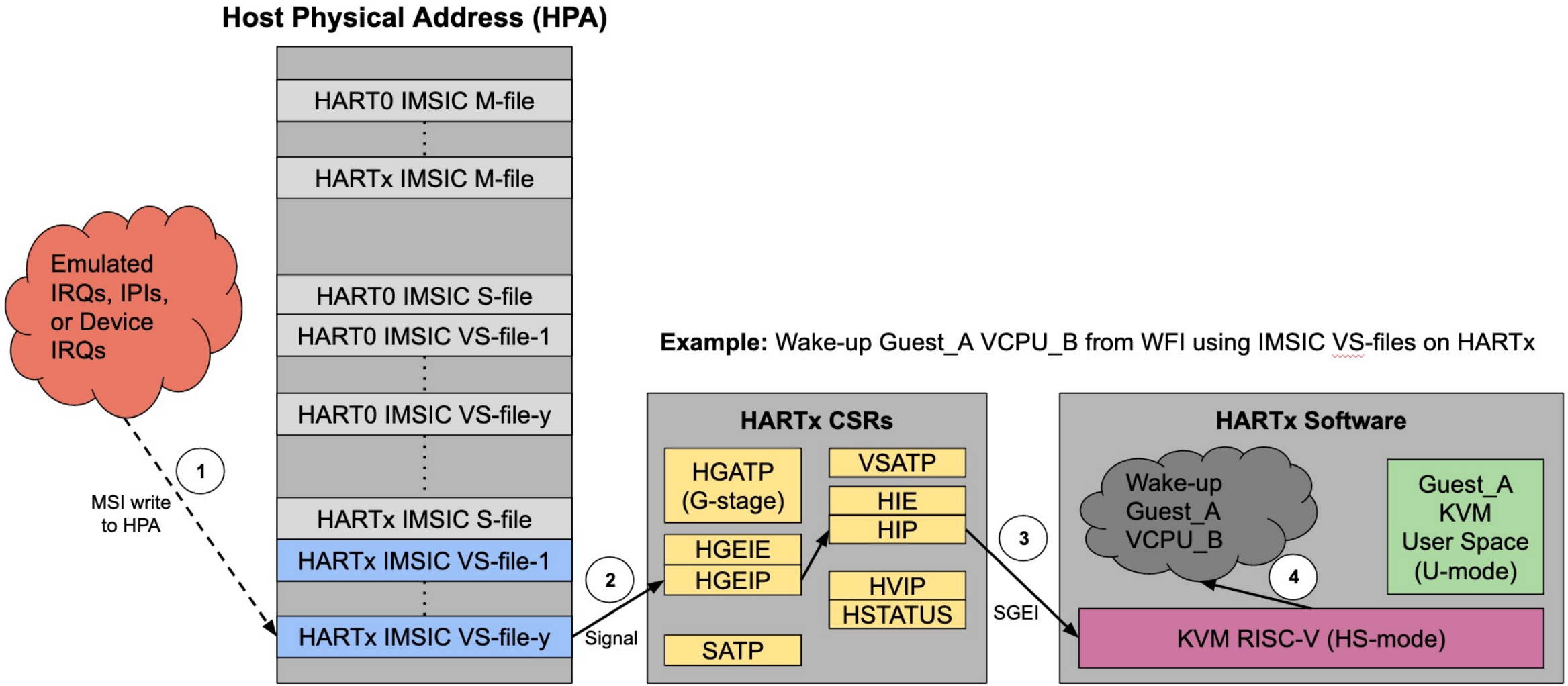
KVM: Device IRQs using IMSIC VS-files

Events involved in forwarding device IRQs using IMSIC VS-files



KVM: WFI wake-up using IMSIC VS-files

Events involved in waking-up VCPU from WFI using IMSIC VS-files



KVM: Change IMSIC VS-file of a Guest VCPU

Flow of how KVM RISC-V changes IMSIC VS-file of Guest VCPU