CVA6 Hypervisor Extension Support “CVA6-H”

Project Concept Proposal

Bruno Sá, José Martins, **Sandro Pinto**

Universidade do Minho / Zero-Day-Labs

OpenHW Group @ San Jose

December 15th, 2022

Project High-Level Overview

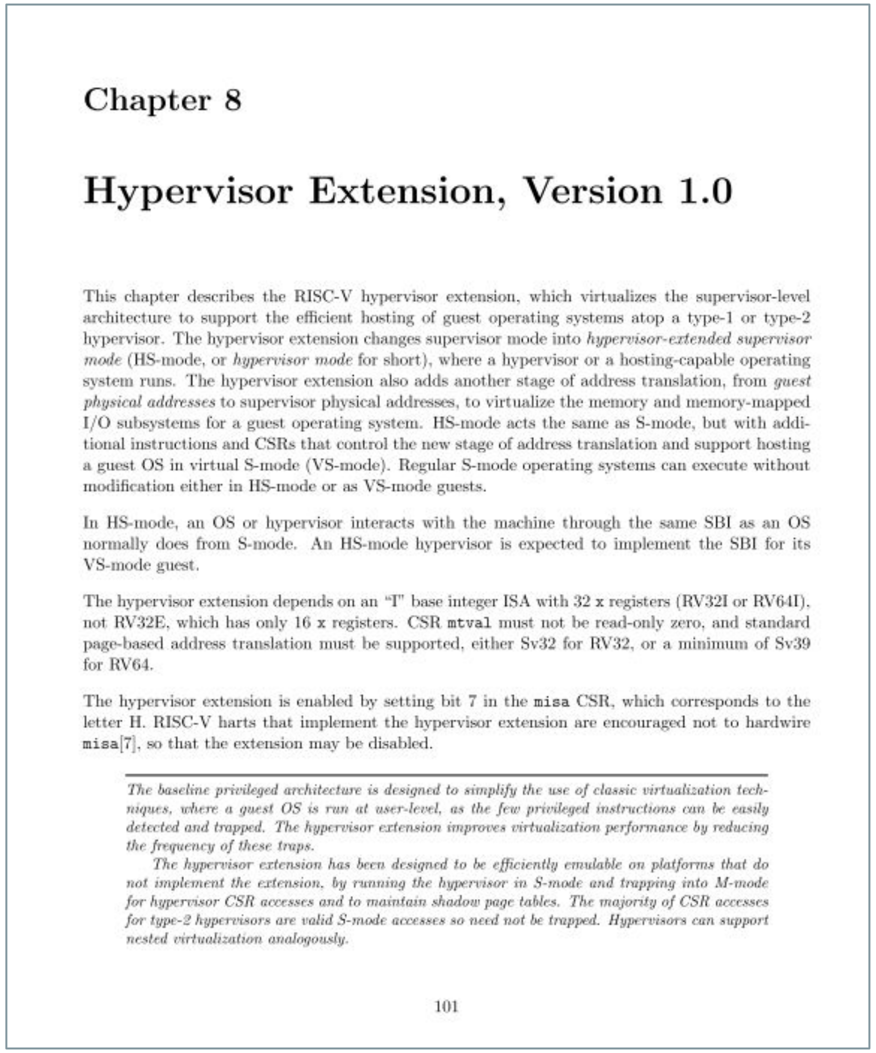
RISC-V Hypervisor Extension

- **The Hypervisor Extension was ratified and frozen in Q4 2021**

- **Designed for type-1 and type-2 hypervisors**
  - Nested virtualization is also supported

- **Additional orthogonal execution modes**
  - HS-mode (hypervisor-extended supervisor)
  - VS-mode & VU-mode

- **Additional CSR registers:**
  - HS-mode CSRs for hypervisor capabilities (e.g., hstatus)
  - HS-mode CSRs for accessing Guest/VM state (e.g., vsstatus)



Chapter 8

Hypervisor Extension, Version 1.0

This chapter describes the RISC-V hypervisor extension, which virtualizes the supervisor-level architecture to support the efficient hosting of guest operating systems atop a type-1 or type-2 hypervisor. The hypervisor extension changes supervisor mode into *hypervisor-extended supervisor mode* (HS-mode, or *hypervisor mode* for short), where a hypervisor or a hosting-capable operating system runs. The hypervisor extension also adds another stage of address translation, from *guest physical addresses* to supervisor physical addresses, to virtualize the memory and memory-mapped I/O subsystems for a guest operating system. HS-mode acts the same as S-mode, but with additional instructions and CSRs that control the new stage of address translation and support hosting a guest OS in virtual S-mode (VS-mode). Regular S-mode operating systems can execute without modification either in HS-mode or as VS-mode guests.

In HS-mode, an OS or hypervisor interacts with the machine through the same SBI as an OS normally does from S-mode. An HS-mode hypervisor is expected to implement the SBI for its VS-mode guest.

The hypervisor extension depends on an "I" base integer ISA with 32 **x** registers (RV32I or RV64I), not RV32E, which has only 16 **x** registers. CSR `mtval` must not be read-only zero, and standard page-based address translation must be supported, either Sv32 for RV32, or a minimum of Sv39 for RV64.

The hypervisor extension is enabled by setting bit 7 in the `misa` CSR, which corresponds to the letter H. RISC-V harts that implement the hypervisor extension are encouraged not to hardwire `misa`[7], so that the extension may be disabled.

The baseline privileged architecture is designed to simplify the use of classic virtualization techniques, where a guest OS is run at user-level, as the few privileged instructions can be easily detected and trapped. The hypervisor extension improves virtualization performance by reducing the frequency of these traps.

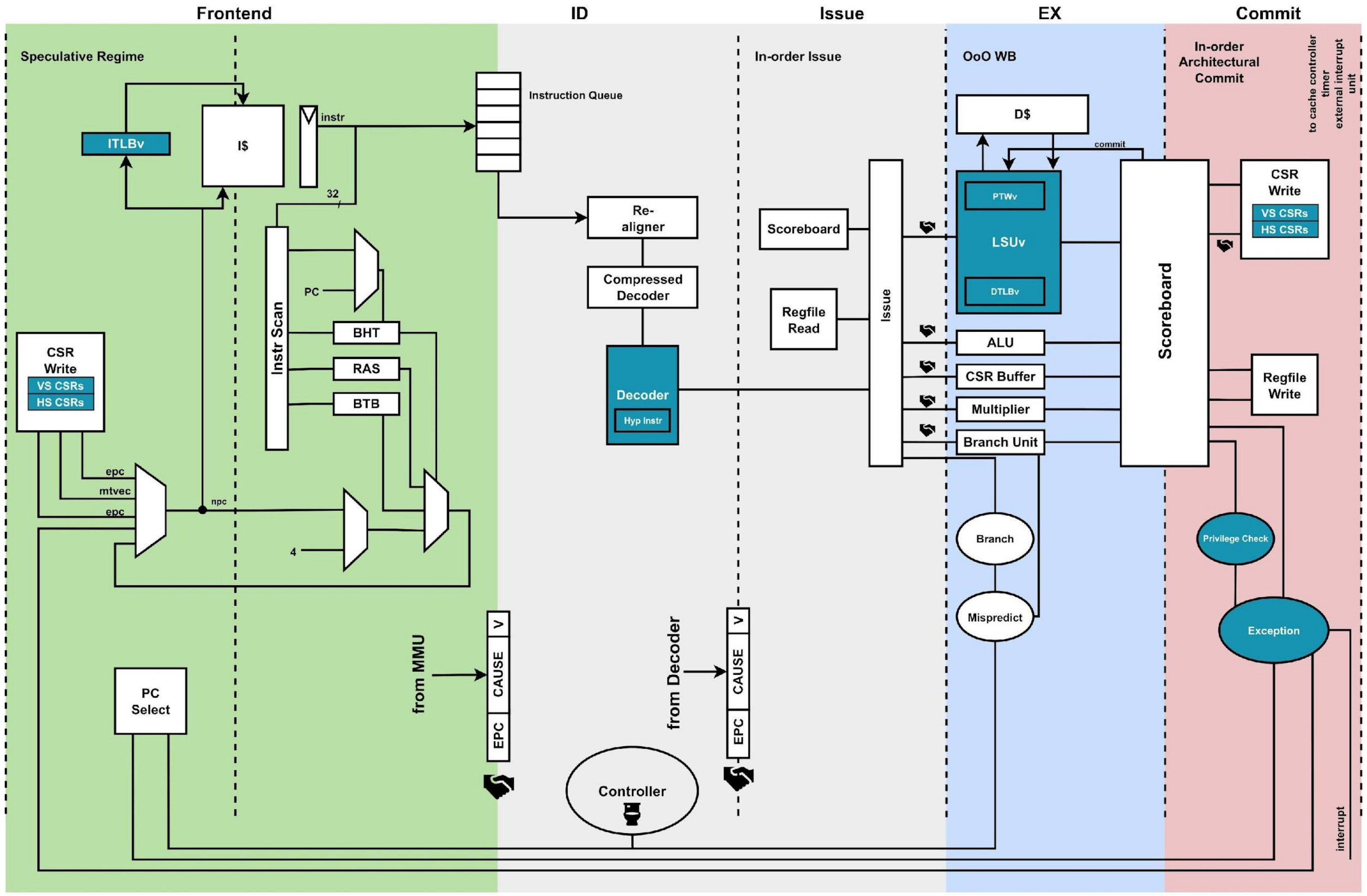
The hypervisor extension has been designed to be efficiently emulable on platforms that do not implement the extension, by running the hypervisor in S-mode and trapping into M-mode for hypervisor CSR accesses and to maintain shadow page tables. The majority of CSR accesses for type-2 hypervisors are valid S-mode accesses so need not be trapped. Hypervisors can support nested virtualization analogously.

101

https://github.com/riscv/riscv-isa-manual/releases/latest

CVA6 Hypervisor Extension

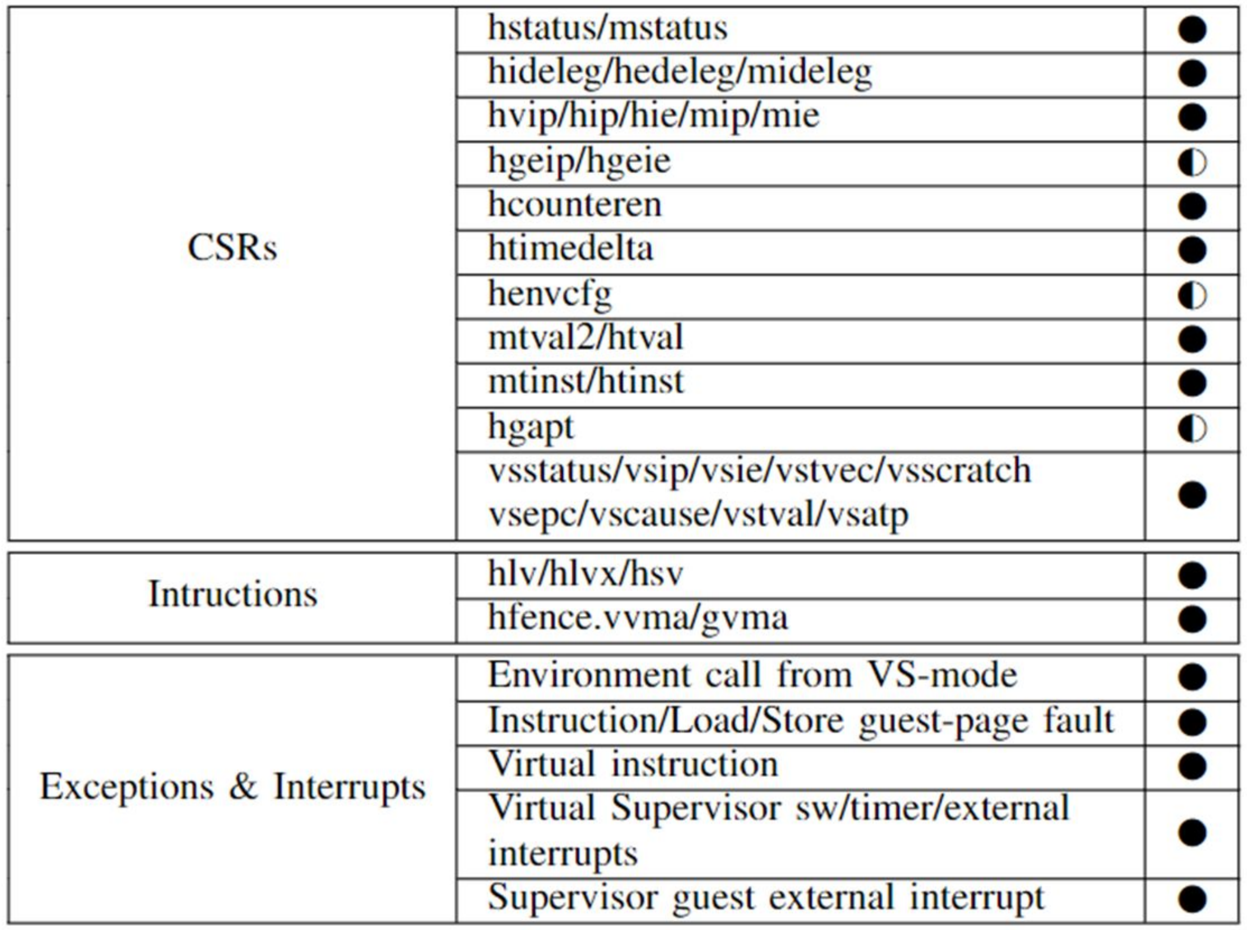
- CSR
- Decoder
- Exceptions
- PTW
- (I/D)TLB



Specification Checklist

- H Extension, Version 1.0.0
- RV64 and sv39x4

Feature
Privilege Modes
Hypervisor and Virtual Supervisor CSRs
Hypervisor Instructions
Machine-level CSRs
Two-stage Address Translation
Traps



CSRs	hstatus/mstatus	●
	hideleg/hedeleg/mideleg	●
	hvip/hip/hie/mip/mie	●
	hgeip/hgeie	◐
	hcounteren	●
	htimedelta	●
	henvcfg	◐
	mtval2/htval	●
	mtinst/htinst	●
	hgapt	◐
	vsstatus/vsip/vsie/vstvec/vsscratch vsepc/vscause/vstval/vsatp	●
Intructions	hlv/hlvx/hsv	●
	hfence.vvma/gvma	●
Exceptions & Interrupts	Environment call from VS-mode	●
	Instruction/Load/Store guest-page fault	●
	Virtual instruction	●
	Virtual Supervisor sw/timer/external interrupts	●
	Supervisor guest external interrupt	●

Summary of market or input requirements

Project Requirements

- **Hypervisor Extension specification version 1.0**
- **Hypervisor Extension optional via parameterization**
- **Functional validated with multiple Hypervisors**
- **Funcional validated with multiple Guest VM and Configurations:**
  - Bare-metal Applications
  - Linux (CVA6-SDK)
  - FreeRTOS
- **Tested and deployed in multiple FPGAs:**
  - Genesys2
  - VCU118
- **Extend CVA6-SDK (hypervisor)**
  - Bao Hypervisor

Potential Enhancements

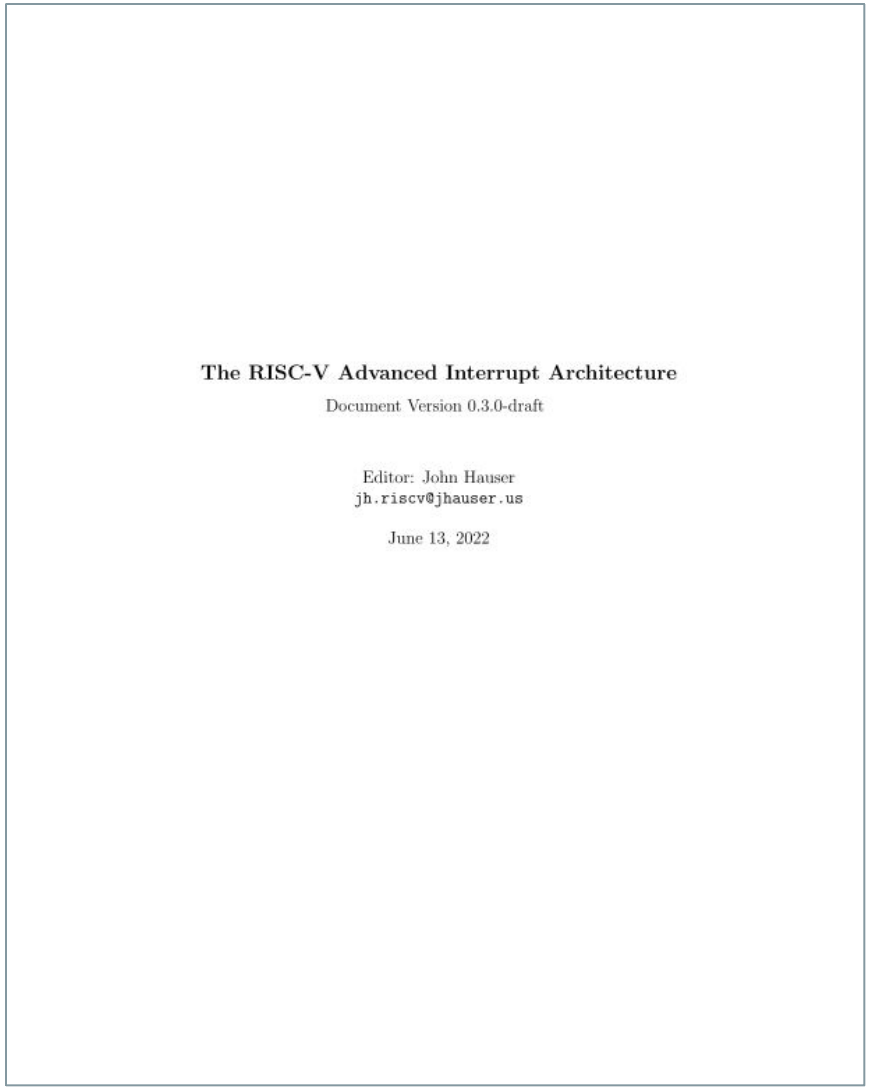
- **CVA6 Hypervisor Extension RV32 Variant:**
  - RV32 Virtual Supervisor and Hypervisor CSRs
  - MMU: SV32x4 translation scheme
- **MMU uArch Enhancements:**
  - PTE Cache
  - PTW G-Stage TLB
  - L2 TLB
  - Svnapot
- **Relevant Extensions:**
  - Sstc (S- and VS-mode timer)
  - Svnapot
  - Svinval
- **Virtualization at the System Level:**
  - IOMMU
  - AIA

RISC-V AIA

- **Advanced Interrupt Architecture**
  - Advanced Platform-Level Interrupt Controller (APLIC)
  - Incoming MSI Controller (IMSIC)
  - Hardware assistance for VM interrupts
  - Interprocessor Interrupts (IPIs)
  - I/O MMU support for MSI to VMs

- **AIA spec**
  - Version 0.3.0 (AIA TG)
  - Targeting ratification @ Q4 2022

- **Open question**
  - Interrupt virtualization support for APLIC



The RISC-V Advanced Interrupt Architecture

Document Version 0.3.0-draft

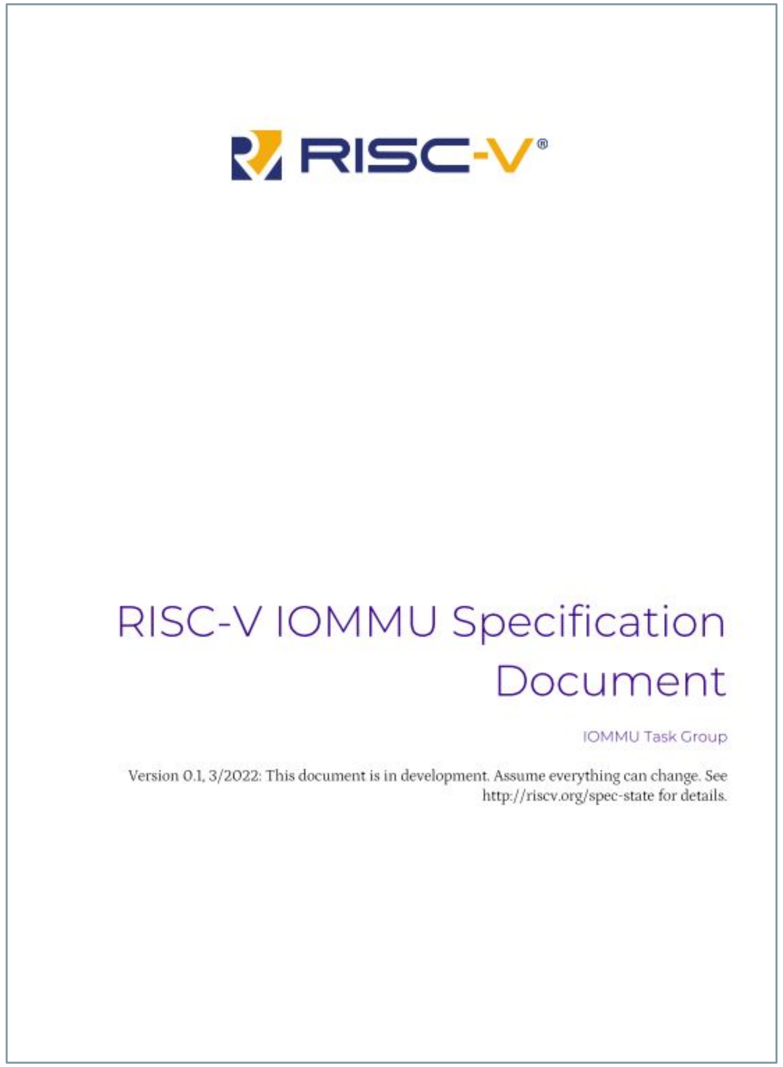
Editor: John Hauser
jh.riscv@jhauser.us

June 13, 2022

https://github.com/riscv/riscv-aia/releases/

RISC-V IOMMU

- **Input-Output Memory Management Unit**
  - Support single or two-stage address translation
  - Supports VM and process isolation
  - Supports translation of AIA MSIs
  - Support for PCIe ATS and PRI
- **RISC-V IOMMU**
  - Version 0.1 -> not official spec (IOMMU TG)
  - Targeting ratification @ Q4 2022



RISC-V IOMMU Specification Document

IOMMU Task Group

Version 0.1, 3/2022: This document is in development. Assume everything can change. See http://riscv.org/spec-state for details.

https://github.com/riscv-non-isa/riscv-iommu/blob/main/riscv-iommu.pdf

Project Impact and Industry Landscape

Who would make use of OpenHW output ?

- Silicon Manufacturers
- Chip/SoC Designers
- FPGA Vendors
- Verification Companies
- Software Companies
- Universities / Research Centers

RISC-V Hypervisors

Organization	Hypervisor	Status	License	Reference
Linux	KVM	Complete	GPLv2	GitHub: https://github.com/kvm-riscv/linux Tutorial: https://github.com/kvm-riscv/howto/wiki
Xvisor Project (Anup)	Xvisor	Complete	GPLv2	GitHub: https://github.com/avpatel/xvisor-next
Bao Project / UMinho	Bao	Complete	GPLv2	GitHub: https://github.com/bao-project/bao-hypervisor
Xen Project	Xen	WiP	GPLv2	GitHub: https://github.com/xen-project/xen
Siemens	Jailhouse	WiP	GPLv2	GitHub: https://github.com/siemens/jailhouse
seL4	seL4 VMM	PoC	BSD 2-Clause	Github: https://github.com/SEL4PROJ/sel4_riscv_vmm
RIVOS	Salus	WiP	-	Github: https://github.com/rivosinc/salus
fentISS	Xtratum NG	WiP	Commercial	Website: https://fentiss.com/products/hypervisor/

RISC-V Cores: Hypervisor Extension

Organization	Core	Status	License	Reference
CHIPS Alliance	Rocket	Complete	Apache-2.0	GitHub: https://github.com/chipsalliance/rocket-chip
Cobham Gaisler	NOEL-V	Complete	Commercial / GPL	Website: https://gaisler.com/index.php/products/processors/noel-v Website (GRLIB): https://gaisler.com/index.php/downloads/leongrlib
SiFive	P200, P400, P500, P600	Complete	Commercial	Website: https://www.sifive.com/cores/performance
StarFive	Dubhe	Complete	Commercial	Website: https://starfivetech.com/en/site/riscv-core-ip
InCore Semi	Chromite	WiP	BSD 3-Clause	GitLab: https://gitlab.com/incoresemi/core-generators/chromite
IIT Madras	Shakti	WiP	BSD 3-Clause	GitLab: https://gitlab.com/shaktiproject/cores

Why OpenHW Should Do This Project ?

Project Members, Participants, and Timeline

OpenHW Members Collaboration

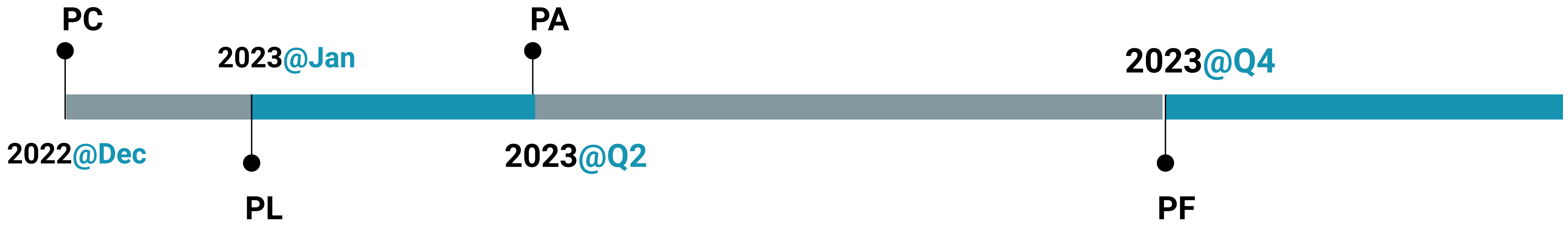
- **Already committed:**
  - University of Minho
  - Zero-Day Labs
- **Possible collaboration:**
  - Imperas (Verification?)
  - UC Santa Barbara (Openpiton)
  - Intel Pathfinder
  - (Others are welcomed :))

Project Team

- **Project Leader(s):**
  - Sandro Pinto
  - Bruno Sá
  - José Martins

- **Technical Project Leader(s):**
  - Sandro Pinto
  - Bruno Sá
  - José Martins

Project Timeline



Q&A