Practical Implications of SPMP-based Virtualization

Manuel Rodríguez
José Martins
Luís Cunha
Kajetan Nuernberger
Thomas Roecker
Sandro Pinto

RISC-V Summit Europe 2026
@Bologna

June 9th, 2026

RIGOLETTO
HIGH PERFORMANCE
RISC-V SOLUTIONS

Agenda

01 Introduction
Background and Motivation

02 Multi-layered SPMP Architecture
SPMP for Hypervisor and PoC Implementation

03 Evaluation
Validation, HW Costs, and Timing Impact

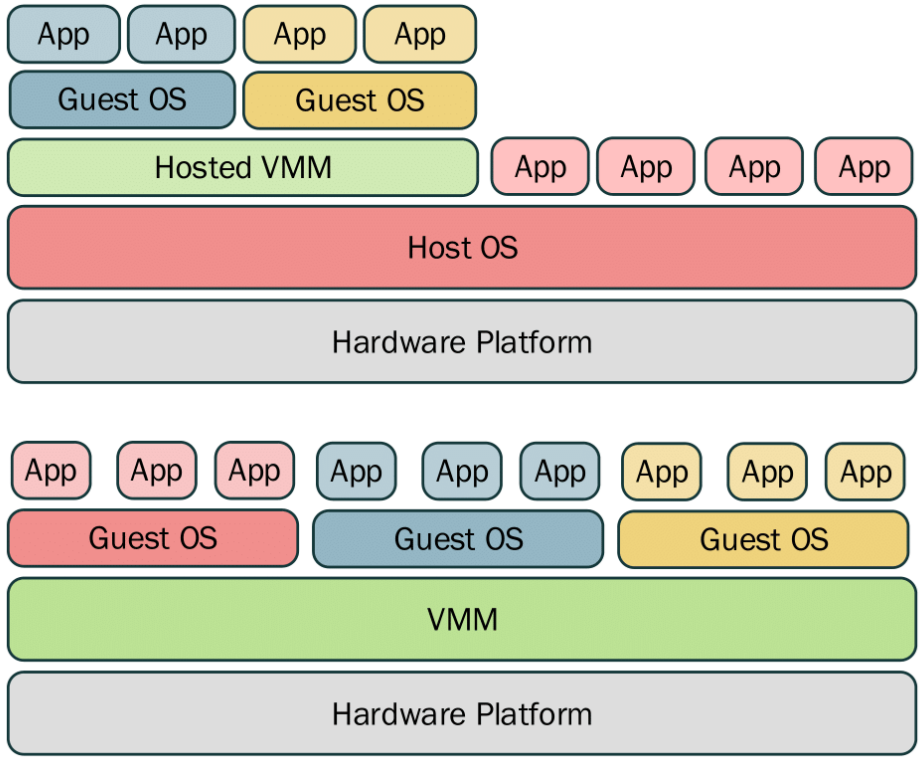
04 Conclusion
Takeaways

Virtualization

Consolidation of multiple workloads onto the same hardware platform.

*A **Hypervisor** is to an OS, as an OS is to a process.*

- **Main functions**
  - Resource/Workload Management
  - Abstraction
  - Cost Reduction
  - Protection / Isolation
- **Well-established**
  - Servers (*load balancing*, *power management*)
  - Desktops (*cross-platform*)
  - **Embedded / Mixed-Criticality Systems** (*isolation*, *consolidation*, *safety*, *security*)

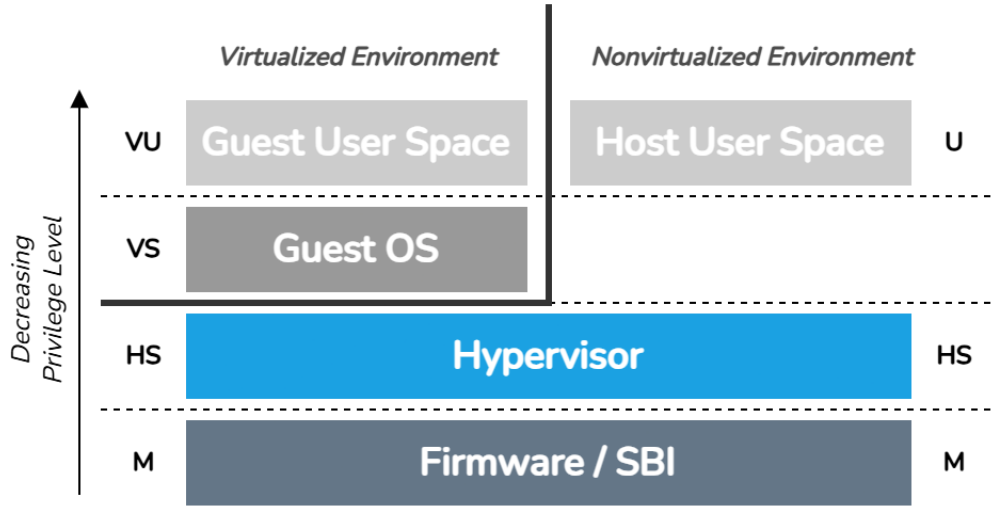


APU

MCU

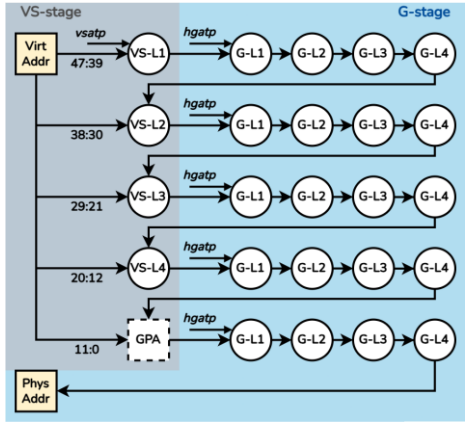
RISC-V Virtualization Today

- **RISC-V Hypervisor Extension**
  - a) New privilege modes
    - HS → Hypervisor
    - VS → Guest OS/Baremetal
    - VU → Guest Apps

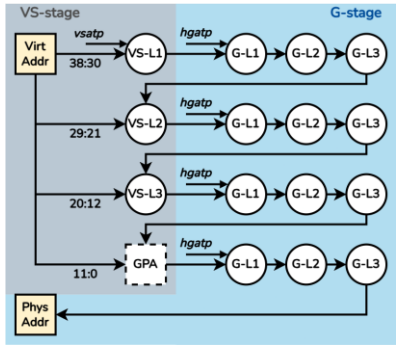


  - b) Two-stage (*nested*) translation

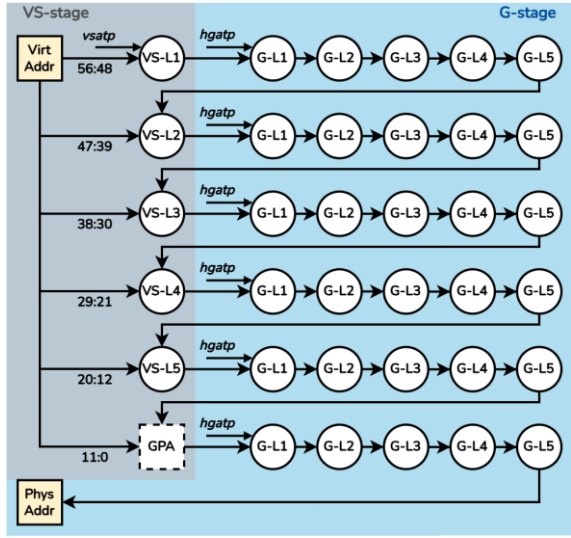
***48-bit** Virt. Addr.*



***39-bit** Virt. Addr.*



***57-bit** Virt. Addr.*



Why MMU-less Virtualization?

Different targets…

- Real-time control
- Safety-critical software
- Mixed-criticality consolidation

…different requirements

- Determinism
- Bounded latency

Servers → *average performance & flexibility*

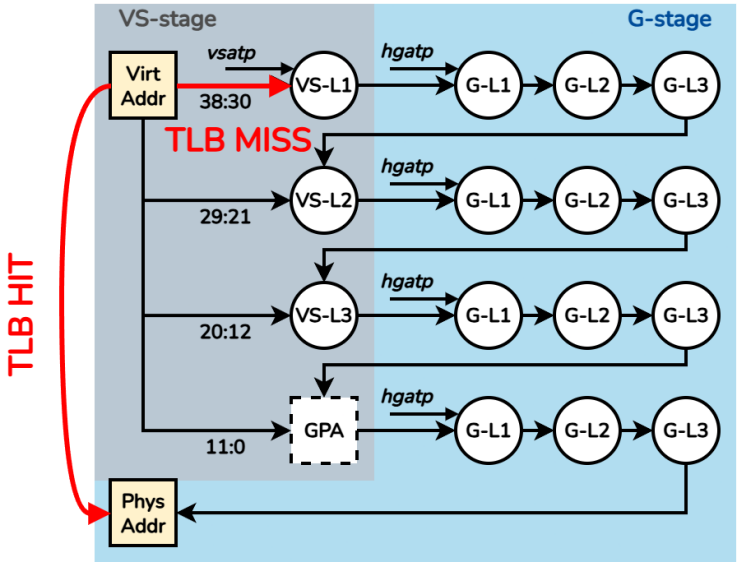
MCUs/real-time → ***worst-case latency & predictability***

Nested translation introduces

- TLB misses & page table walks
- Variable memory access latency
- Reduced predictability

39-bit Virt. Addr.

Up to 15 memory accesses per request



MMU

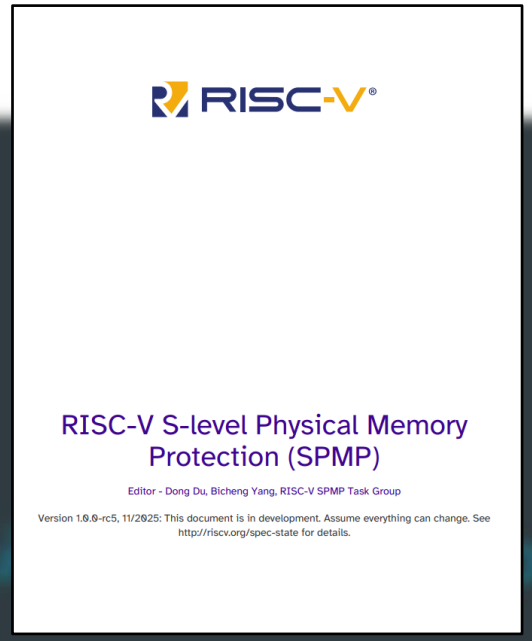
Memory Management Unit

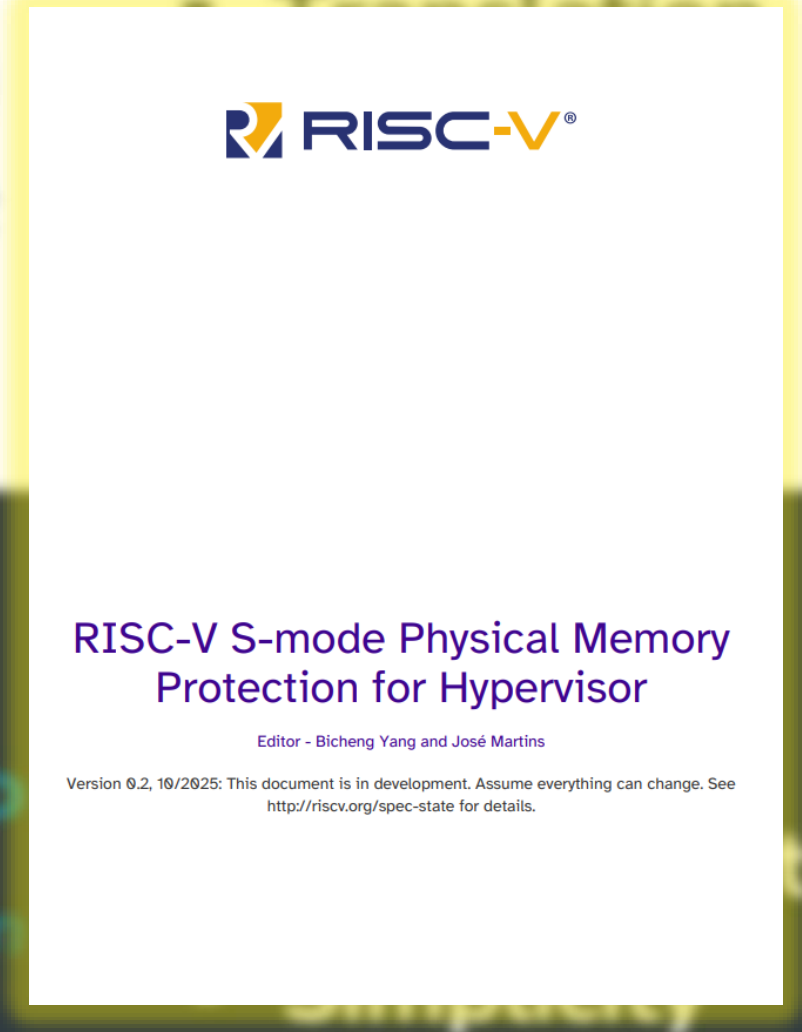
- Translation
- Protection
- Flexibility

PMP

Physical Memory Protection

- Protection
- Predictability
- Simplicity

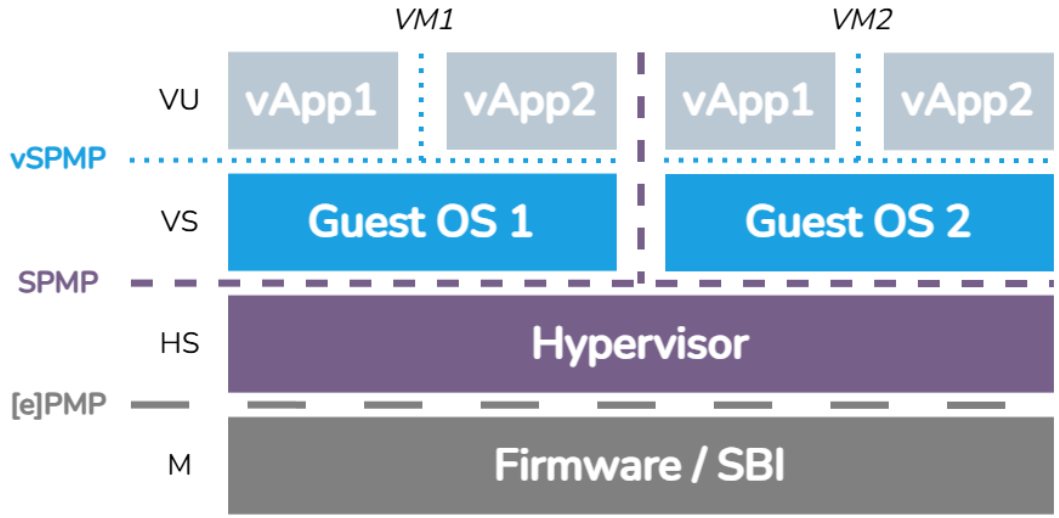

RISC-V®
RISC-V S-level Physical Memory Protection (SPMP)
Editor - Dong Du, Bicheng Yang, RISC-V SPMP Task Group
Version 1.0.0-rc5, 11/2025: This document is in development. Assume everything can change. See http://riscv.org/spec-state for details.

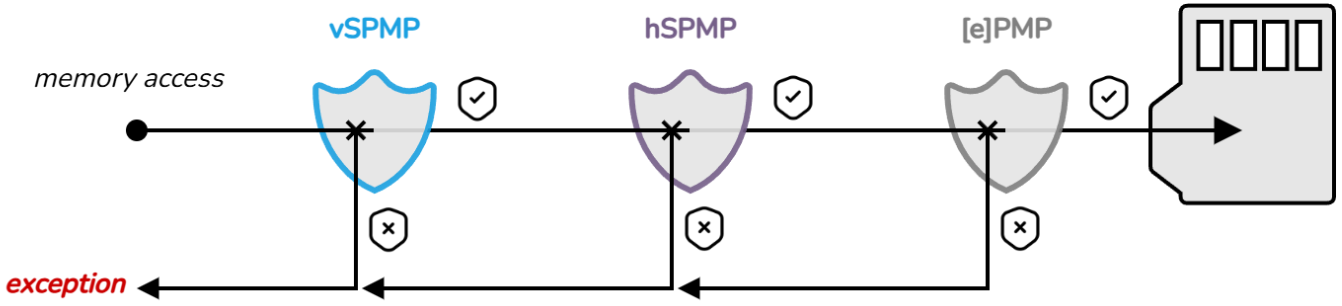

RISC-V®
RISC-V S-mode Physical Memory Protection for Hypervisor
Editor - Bicheng Yang and José Martins
Version 0.2, 10/2025: This document is in development. Assume everything can change. See http://riscv.org/spec-state for details.

RISC-V SPMP for Hypervisor

Virtualization… *without virtual memory!?*

- **Two PMPs for Hypervisor and VMs**
  - ***vSPMP*** → controlled by VMs
  - ***SPMP*** → controlled by Hypervisor
  - Coexists with M-mode **PMP/ePMP**
  - Up to 64 entries per layer (*i.e., 192 max*)
  - Enable bitmask register per layer





The Big Concern: Practical Implications

Isn’t it too expensive?

Critical path degradation?

Feasible to implement?

Area explosion?

Frequency impact?

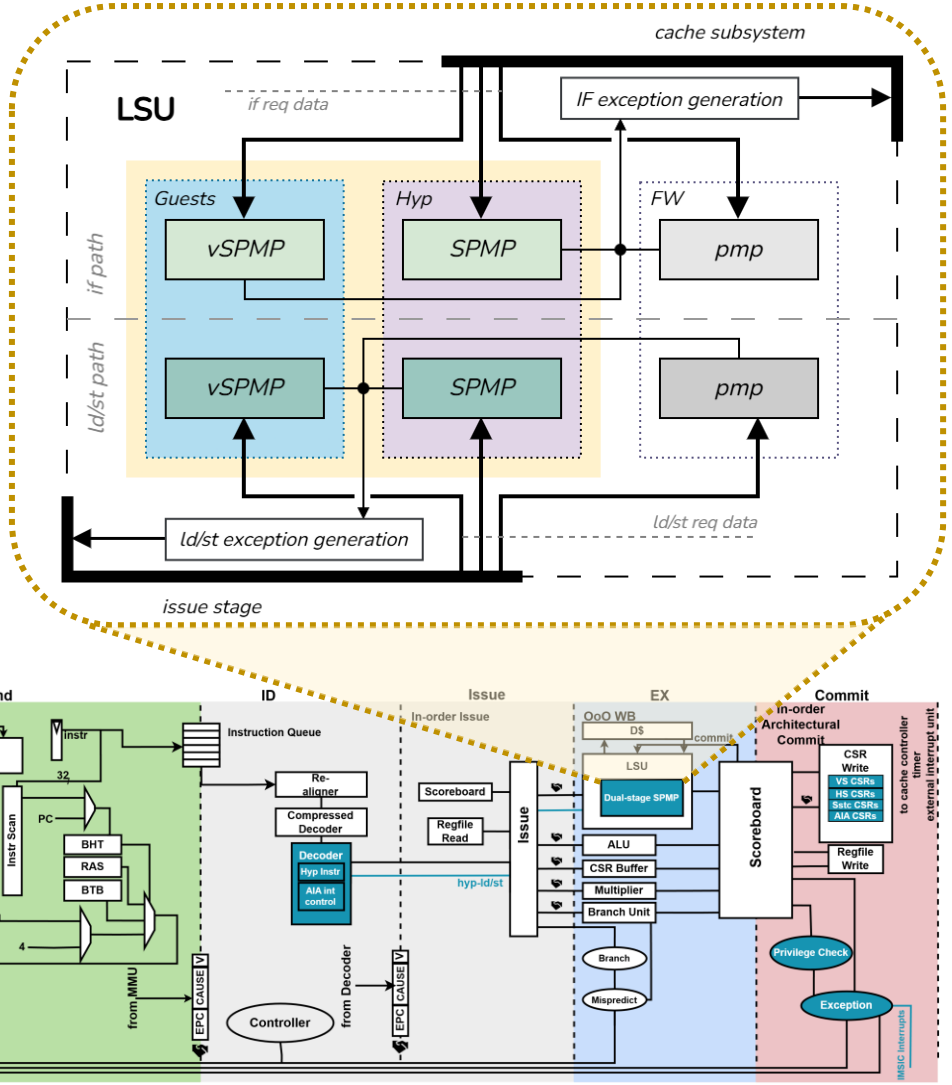
How many entries?

What is an acceptable trade-off?

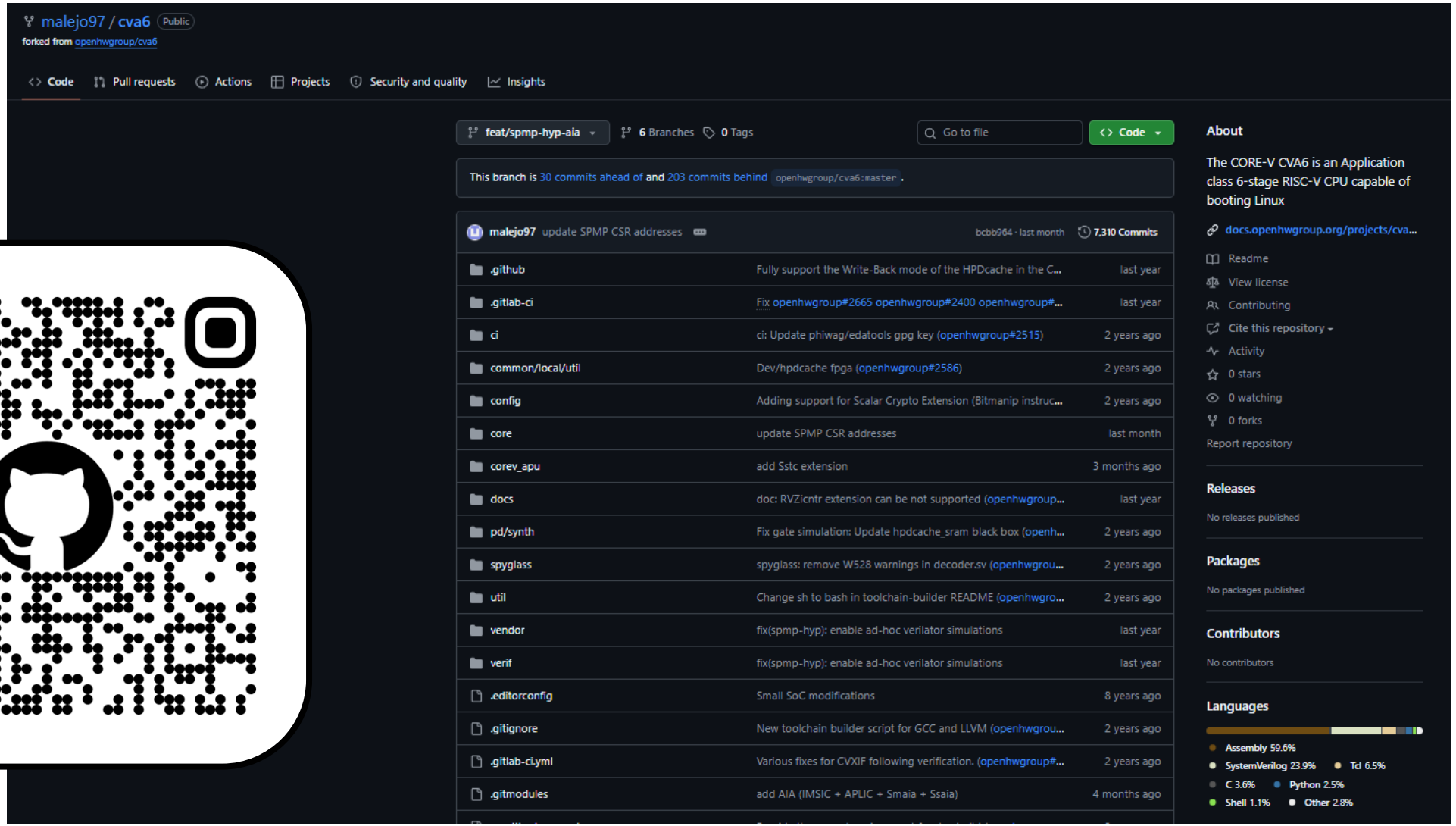
Can a multi-layered PMP architecture remain practical for MCU virtualization?

PoC Implementation: SPMP-based CVA6

- **Integrated into MMU-less CVA6**
  - RV64 and RV32
  - x2 separate SPMPs
    - *Instruction Fetch (IF) path*
    - *Load/Store (LD/ST)* path
- **Parameterizable**
  - SPMP vs MMU
  - Number of entries
  - Context-switch optimization
- **Static delegation M->HS->VS**
- **Fully combinatorial checks**



PoC Implementation: SPMP-based CVA6



https://github.com/malejo97/cva6/tree/feat/spmp-hyp-aia

Validation – Bao @SPMP-based CVA6

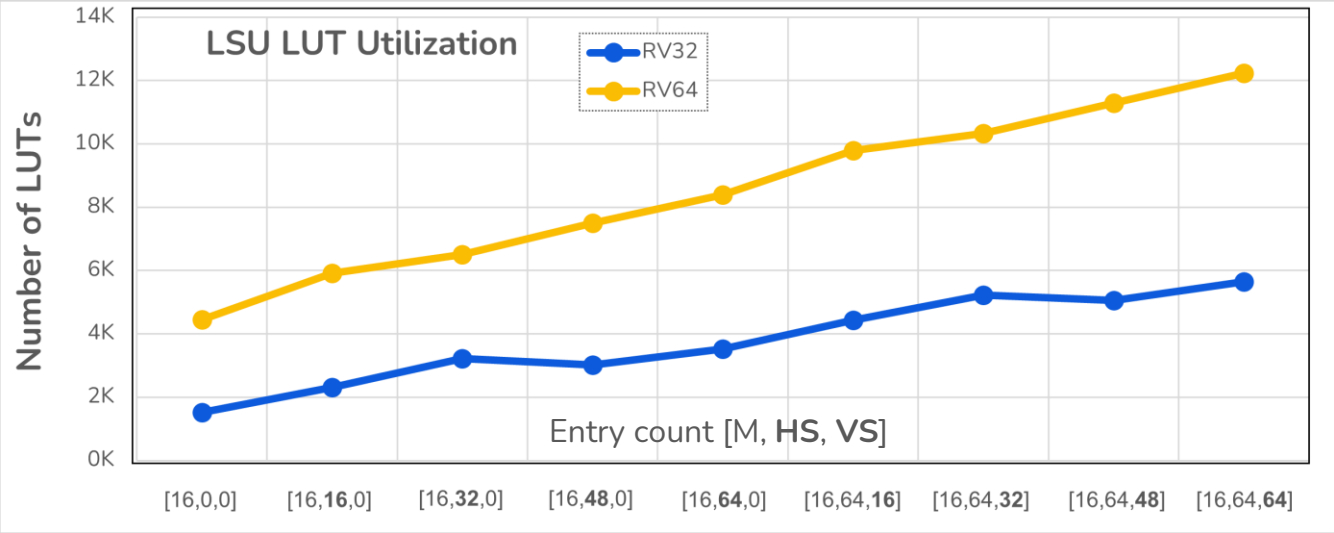
```
                                                                    08:00:44 [65/73]
OpenSBI v1.3
   ____                    _____ ____ _____
  / __ \                  / ____|  _ \_   _|
 | |  | |_ __   ___ _ __ | (___ | |_) || |
 | |  | | '_ \ / _ \ '_ \ \___ \|  _ < | |
 | |__| | |_) |  __/ | | |____) | |_) || |_
  \____/| .__/ \___|_| |_|_____/|____/_____|
        | |
        |_|

Platform Name             : ARIANE RISC-V
Platform Features         : medeleg
Platform HART Count       : 1
Platform IPI Device       : aia-imsic
Platform Timer Device     : aclint-mtimer @ 1000000Hz
Platform Console Device   : uart8250
Platform HSM Device       : ---
Platform PMU Device       : ---
Platform Reboot Device    : ---
Platform Shutdown Device  : ---
Platform Suspend Device   : ---
Platform CPPC Device      : ---
Firmware Base             : 0x80000000
Firmware Size             : 178 KB
Firmware RW Offset        : 0x20000
Firmware RW Size          : 50 KB
Firmware Heap Offset      : 0x24000
Firmware Heap Size        : 34 KB (total), 2 KB (reserved), 9 KB (used),)
Firmware Scratch Size     : 4096 B (total), 180 B (used), 3916 B (free)
Runtime SBI Version       : 1.0

Domain0 Name              : root
Domain0 Boot HART         : 0
Domain0 HARTs             : 0*
Domain0 Region00          : 0x10000000-0x10000fff M: (I,R,W) S/U: (R,W)
```

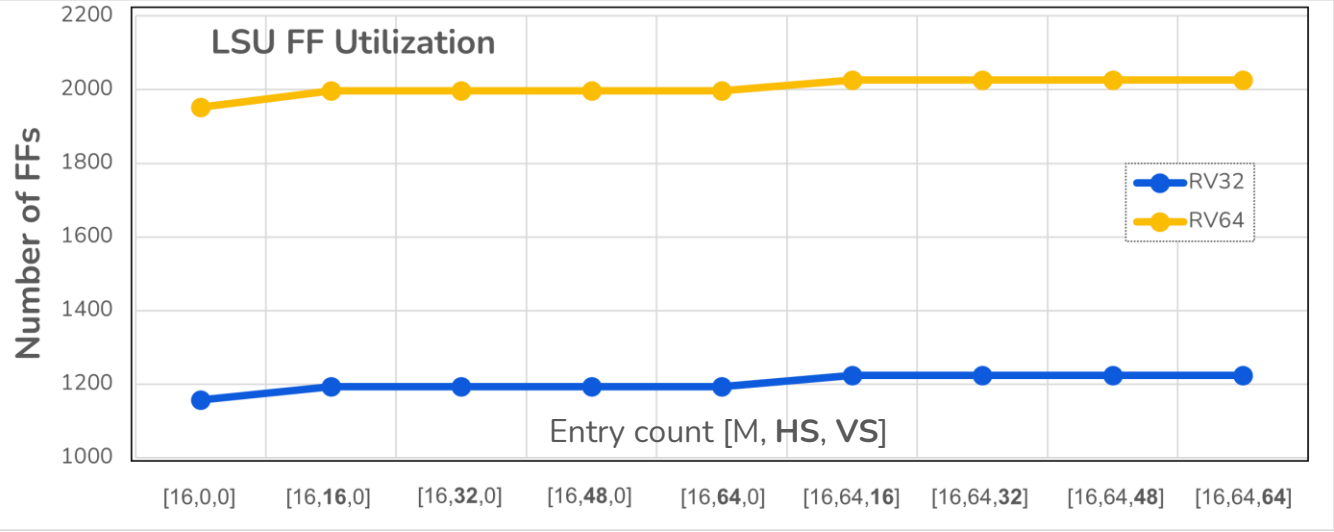
```
Welcome to minicom 2.9

OPTIONS: I18n
Port /dev/ttyUSB0, 08:00:39

Press CTRL-A Z for help on special keys

 Bao Hypervisor  (May 28 2026 - 07:45:22)
*** Booting Zephyr OS build 4.4.99 ***

uart:~$ kernel version
Zephyr version 4.4.99
uart:~$ device list
devices:
- bao-ipc@8a000000 (READY)
- aplic@d000000 (READY)
  DT node labels: aplic
- imsic@28000000 (READY)
  DT node labels: imsic
- serial@10000000 (READY)
  DT node labels: uart0
uart:~$

CTRL-A Z for help | 115200 8N1 | NOR | Minicc
```

Zephyr®

BAO

Evaluation – HW Utilization





> LSU LUT utilization grows by ~x3 at last config (*144 entries total*)

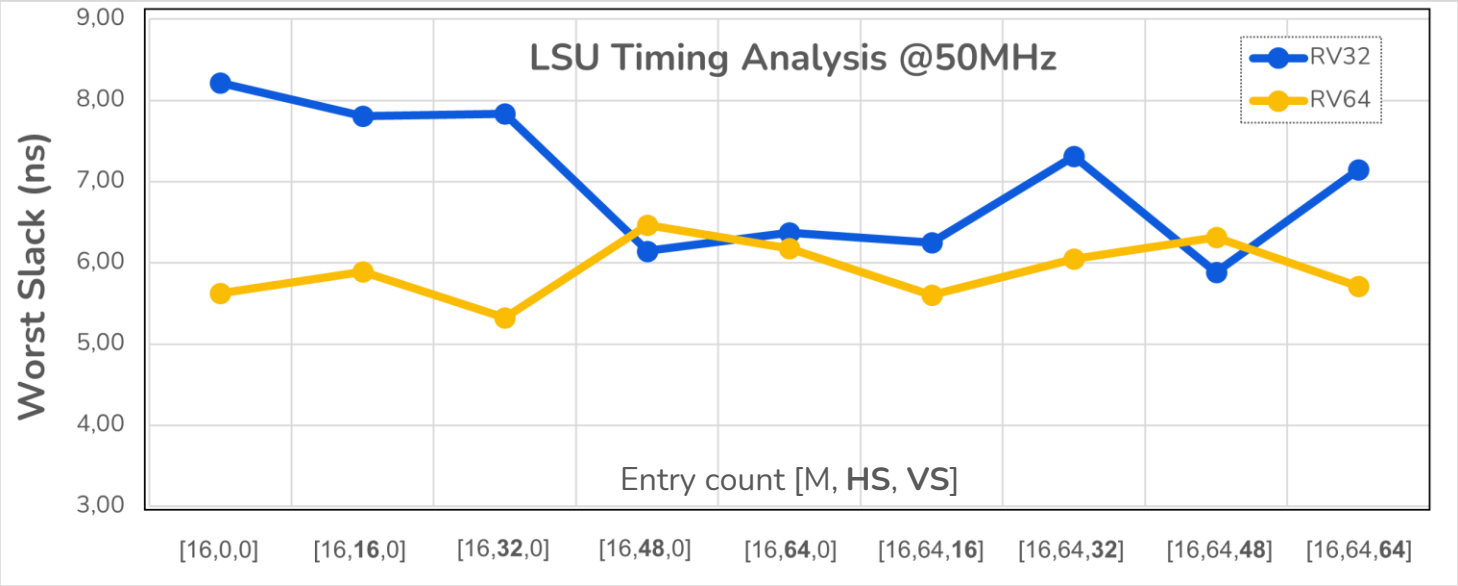
> Multi-layer PMP does not produce exponential scaling

> FF utilization nearly constant

Area increases (*as expected*), but…

growth is **additive, not explosive**

Evaluation – Timing



> **No catastrophic degradation**

LSU WS remains positive

> **Multilayer ≠ deeper critical path**

Parallel checks, not serial

Expected: Timing collapse

Observed: Stability

> **Results show scaling behavior**

Absolutes are implementation-specific

FPGA/ASIC target, clock freq., µArch, etc.

Estimating SPMP Requirements

> **Estimation inputs**

Hyp/guest memory layout

Number of VMs/vCPUs

Shared memory

> **Device isolation**

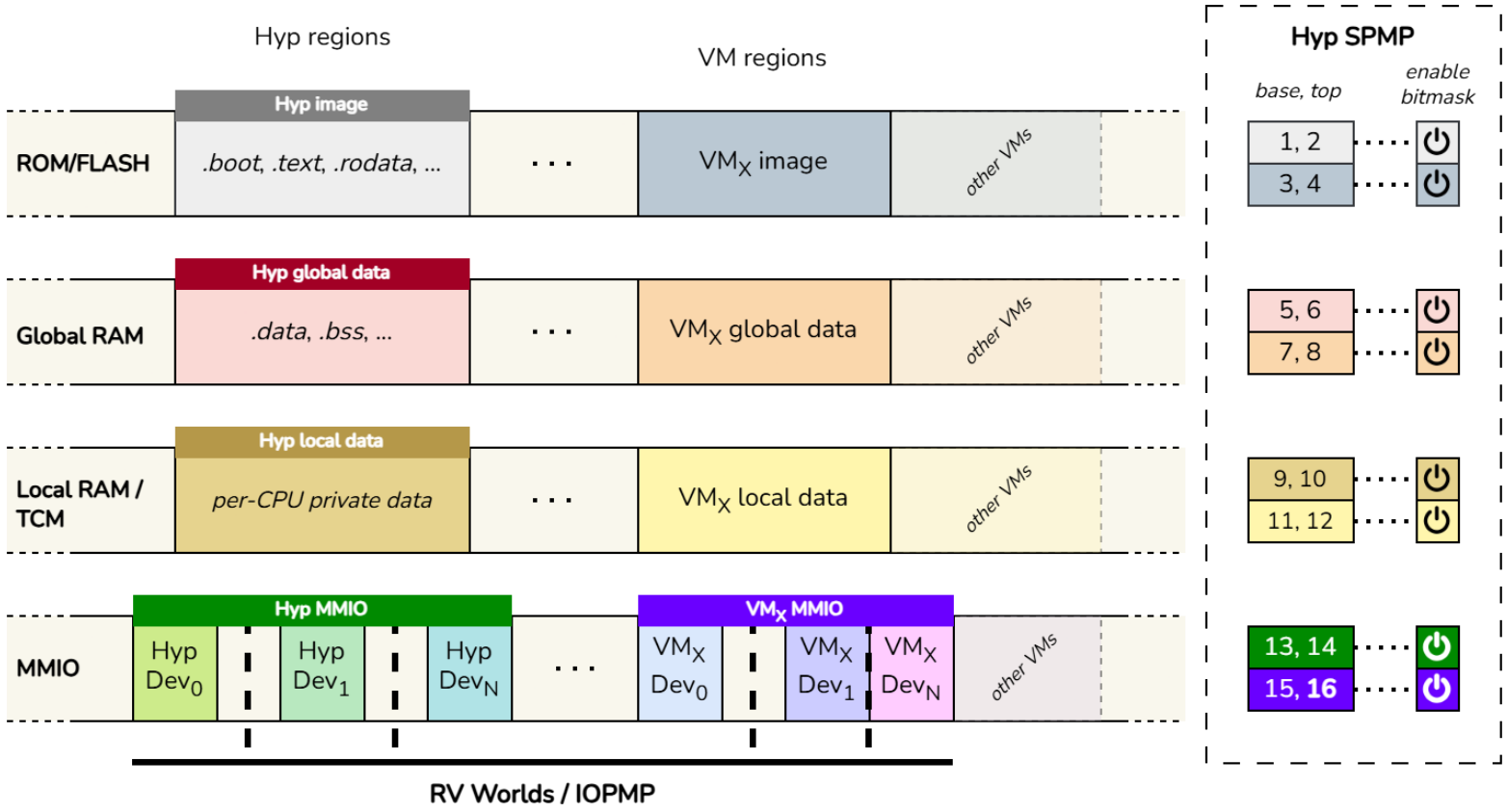
RV Worlds / IOPMP

Reduced SPMP pressure

> **Design trade-offs**

Static allocation → resident VM entries

Dynamic allocation → reprogram VM regions



More entries ⇄ *Less reprogramming*

Fewer entries ⇄ *More reprogramming*

Takeaways

Area grows predictably
with SPMP entry count

Multi-layer ≠ timing disaster

More entries ⇄ **less reprogramming**
fewer entries ⇄ lower provisioning

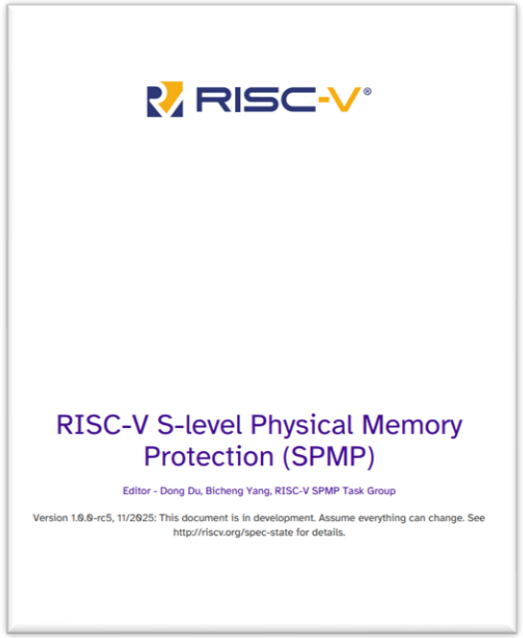
Device isolation → RV Worlds / IOPMP

Evaluation shows scaling behavior
not universal evidence of feasibility

Call to Action!

Baseline SPMP

(Native RTOS / Task isolation)

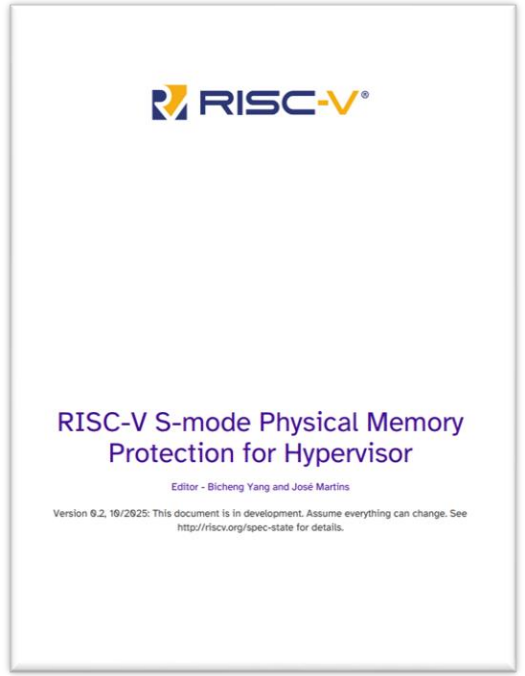


- *Public Review*

Willing to contribute?

SPMP for Hypervisor

(MCU Virtualization)



- *Development Phase*

THANK YOU!

<id11674@alunos.uminho.pt>