



# Open-source RISC-V Input/Output Physical Memory Protection (IOPMP) IP

Luís Cunha, Francisco Marques, Manuel Rodriguez, Tiago Gomes, Bruno Sá, Sandro Pinto



RISC-V Summit Europe 24 @Munich



# Agenda

01

## Introduction

Memory Protection Mechanisms & Specification Overview

02

## Implementation

Overview

03

## Evaluation

Functional Validation & Hardware Results

# Introduction

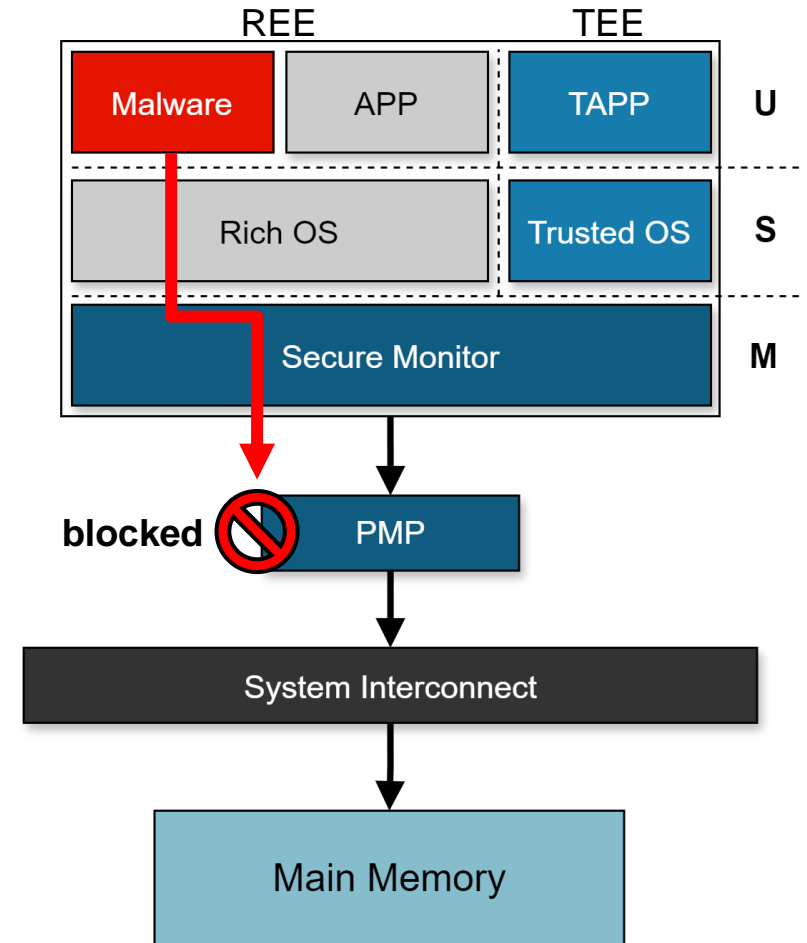
01

# Memory Protection

**Memory protection** is a recognized security capability

## MAIN PURPOSE:

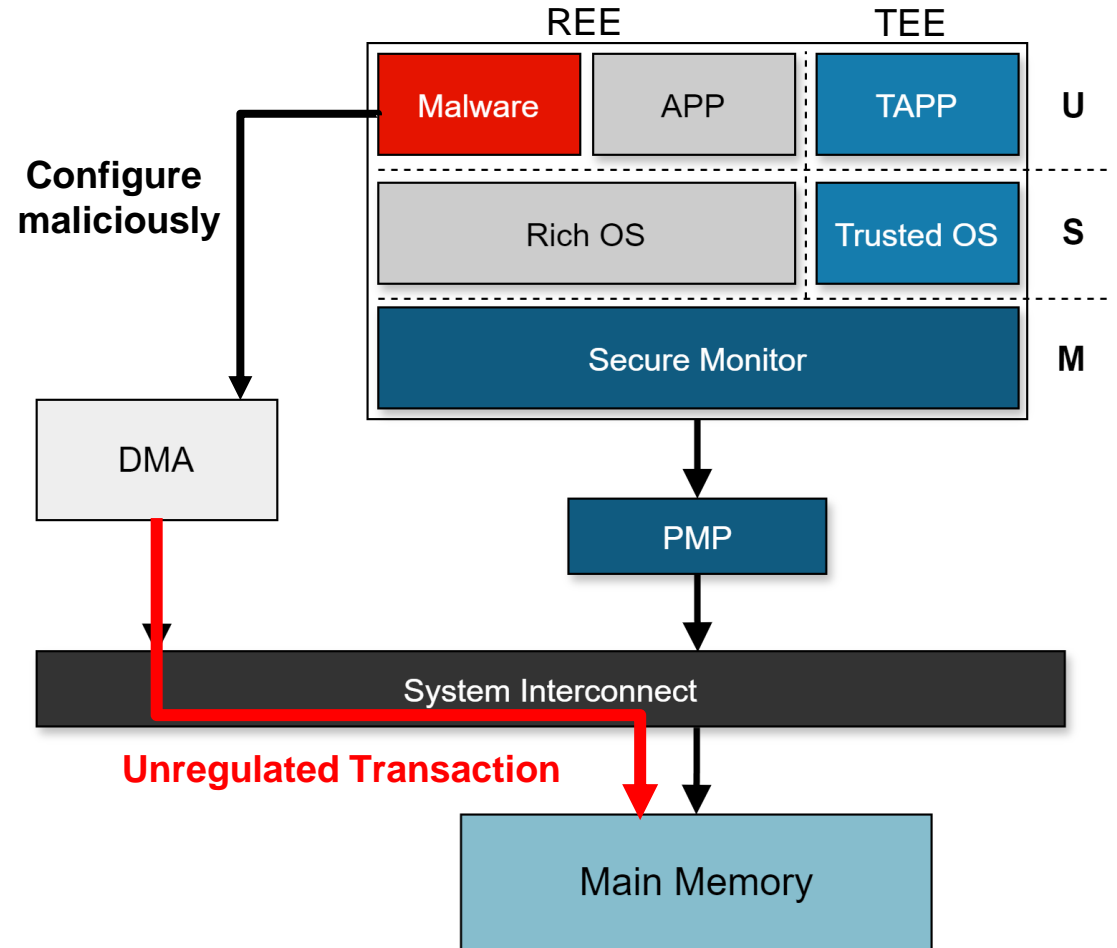
- Prevent a process from accessing memory that has not been allocated to it
- The **Physical Memory Protection (PMP)** (part of the privileged specification) enforces memory protection at the hart level



# Memory Protection

**Memory protection** is a recognized security capability

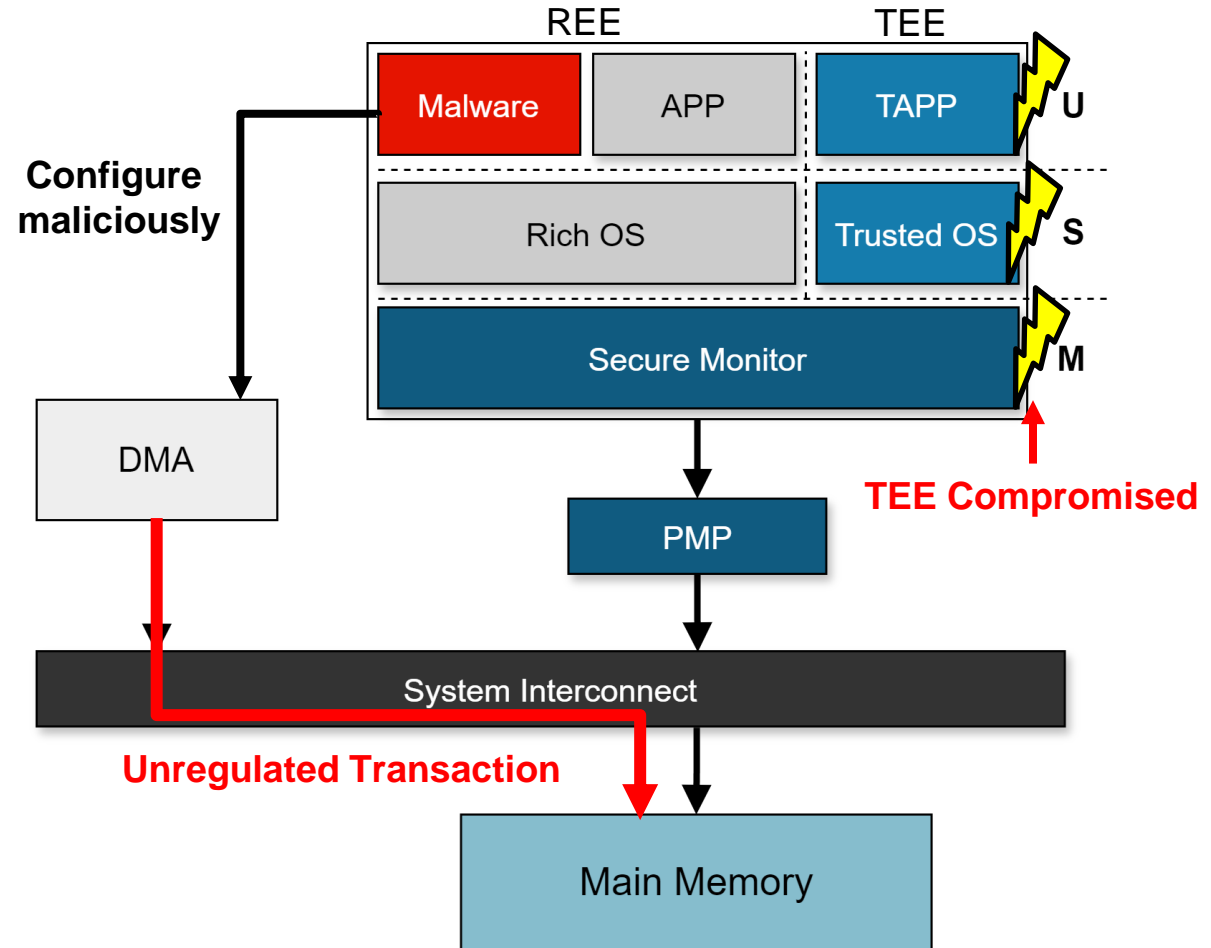
- The PMP **ONLY** enforces memory protection at the hart level
- Other devices are **unsupervised**
- Unsupervised devices can potentially compromise overall System-on-Chips



# Memory Protection

**Memory protection** is a recognized security capability

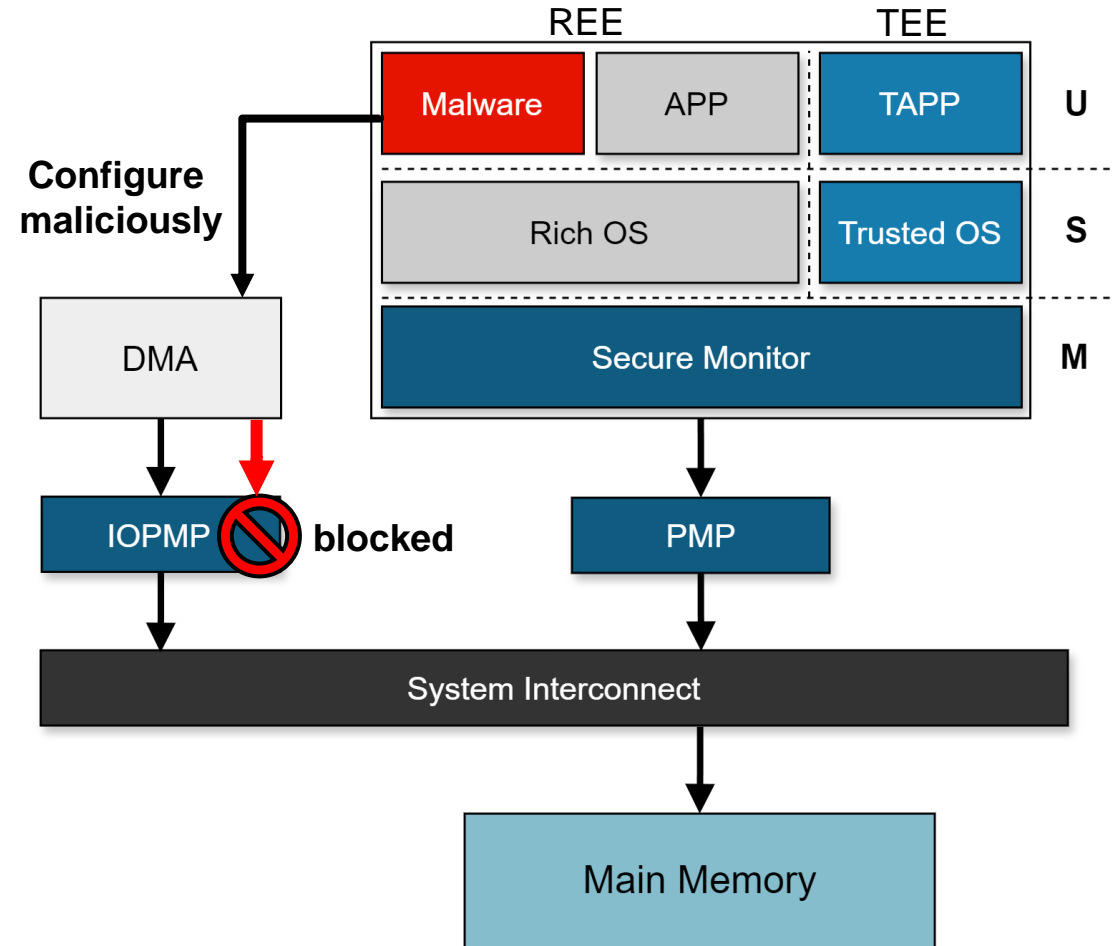
- The PMP **ONLY** enforces memory protection at the hart level
- Other devices are **unsupervised**
- Unsupervised devices can potentially compromise overall System-on-Chips



# Memory Protection

**Memory protection** is a recognized security capability

- The RISC-V community has been working towards the specification of the **Input/Output Physical Memory Protection (IOPMP)**
- **Mediate and manage** device accesses to memory by performing permission checks



# IOPMP Specification

- Mediate bus access from devices
- Multiple device Support
- 5 Operation Models:
  - Full
  - Rapid-K
  - Dynamic-K
  - Isolation
  - Compact-K
- Priority and non-priority rules



## RISC-V IOPMP Architecture Specification

RISC-V IOPMP Task Group

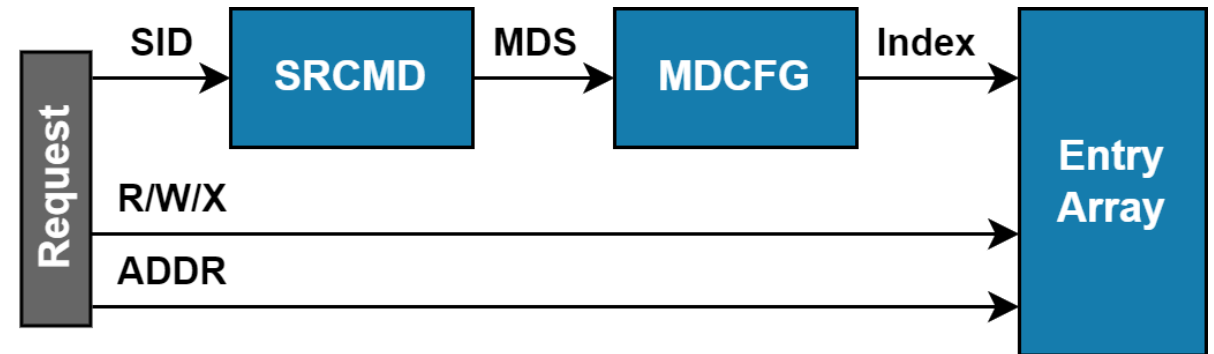
Version 1.0.0-draft5, December, 2023: This document is in development. Assume everything can change. See <http://riscv.org/spec-state> for details.



# IOPMP Specification

Three structures to assess the validity of a transaction:

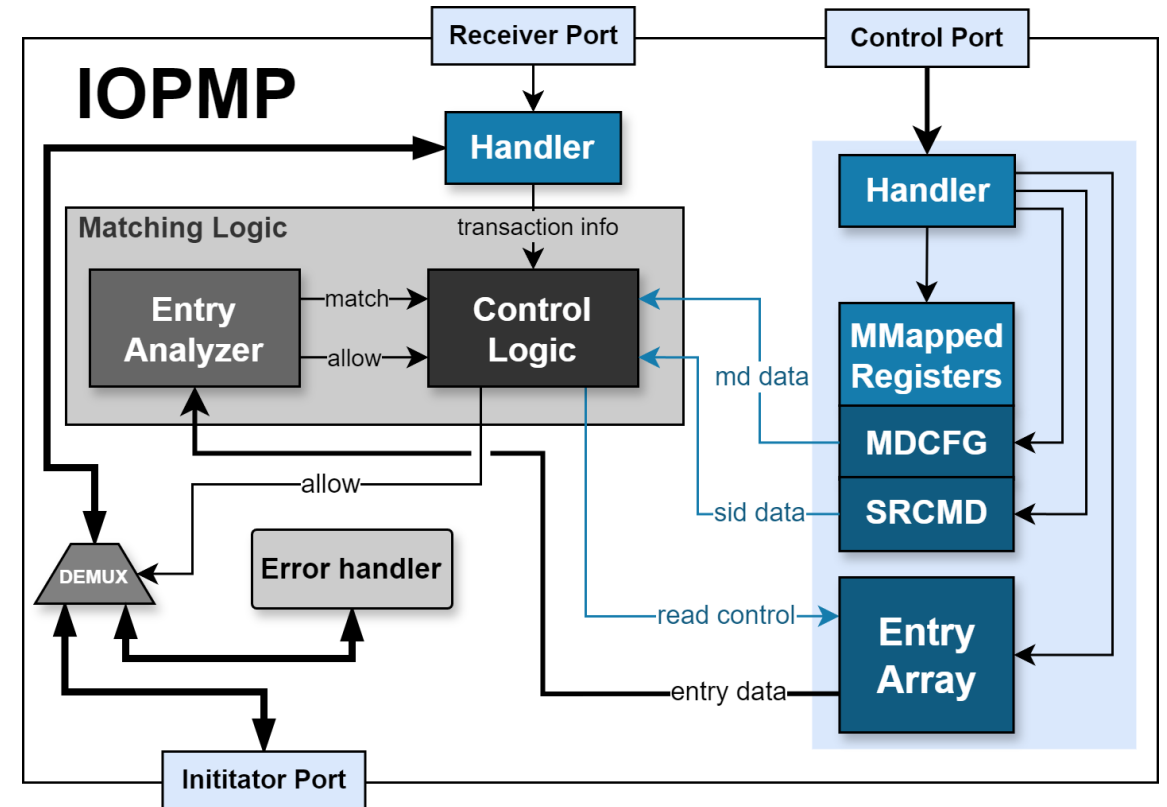
- **Entry** – defines a physical address range and a set of rules
- **Memory Domain** – Used to define groups of entries
- **Source Identifier** – Identifies the device making the transaction request (recently changed to RRID)



# Implementation

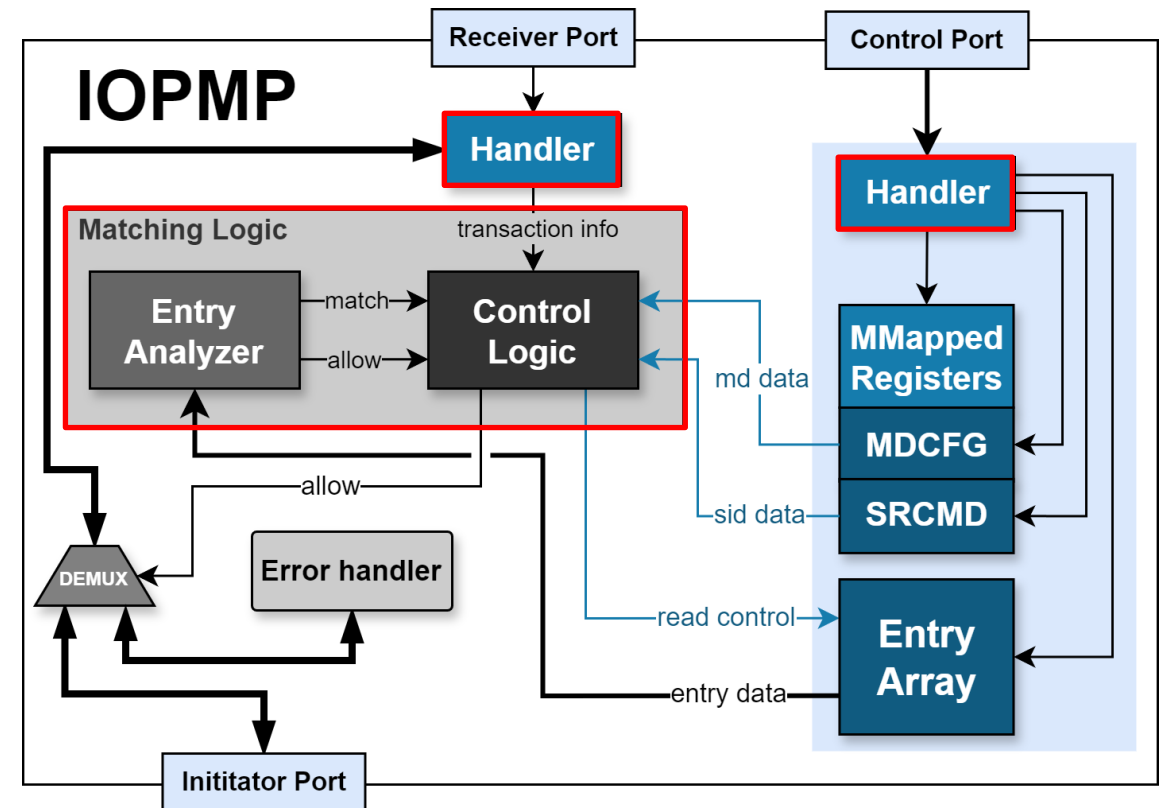
# IP Overview

Feature	Notes
Model	Full Model
Configuration Protection	Only mandatory implemented
Programming Protection	Registers not implemented but the IP is stalled while configuring entries
Error Reporting	Only mandatory implemented
TOR Support	Implemented
Programmable Priority Entries	Implemented
Source Enforcement	Implemented



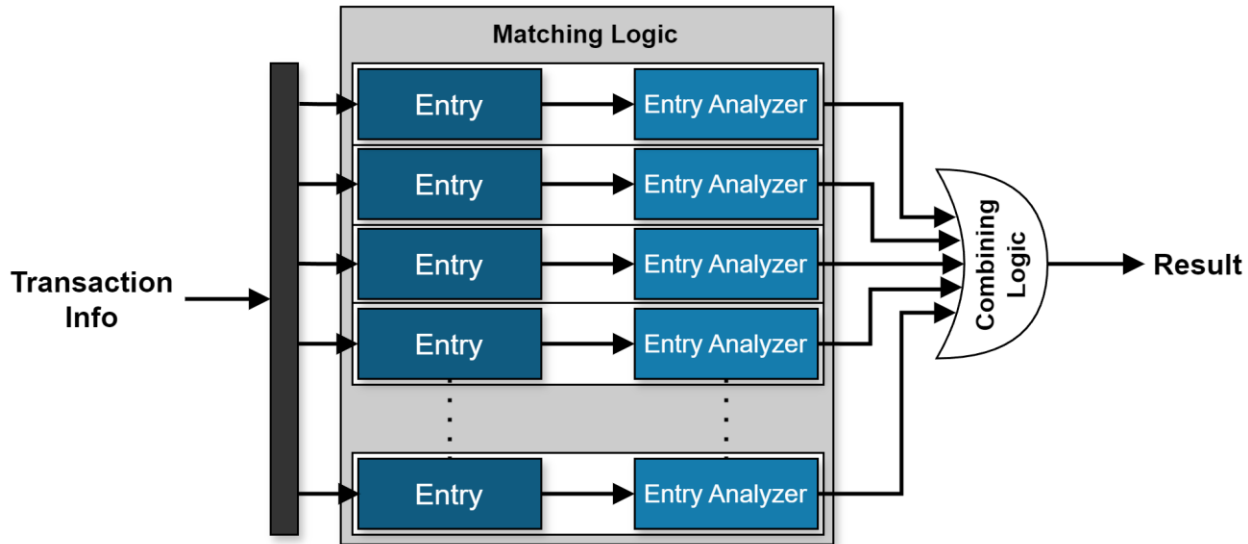
# IP Microarchitecture

- The **Communication Handler** extracts transaction's information
  - SID
  - Base Address
  - Transaction's Length
  - Transaction Type (R,W,X)
- The **Matching Logic** calculates the corresponding Entry indexes and validates the transaction



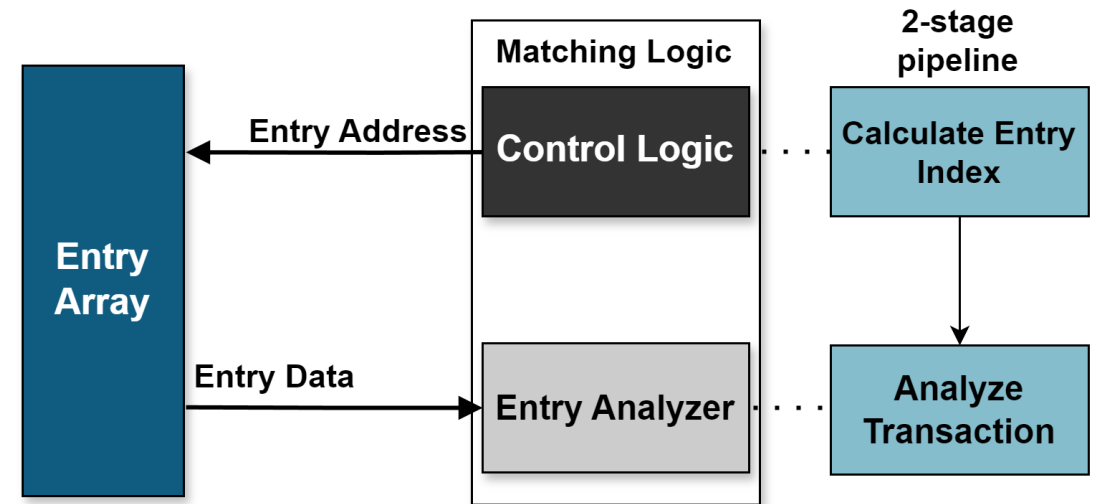
# Matching Logic

## Parallel Approach



- + Performance
- - Scalability

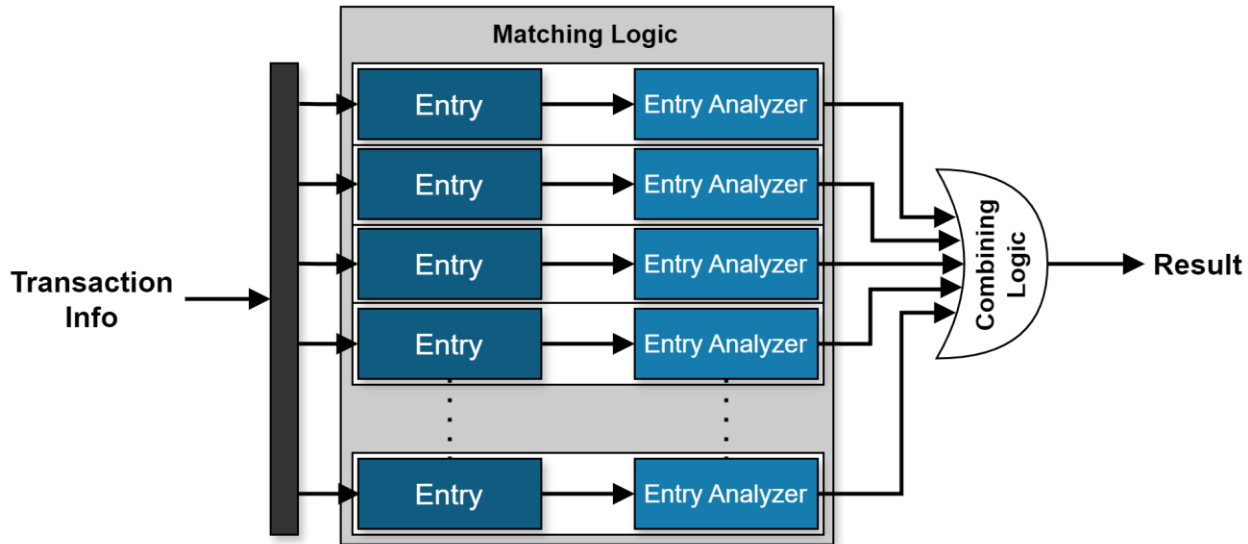
## Sequential Approach



- - Performance
- + Scalability
- + Modularity

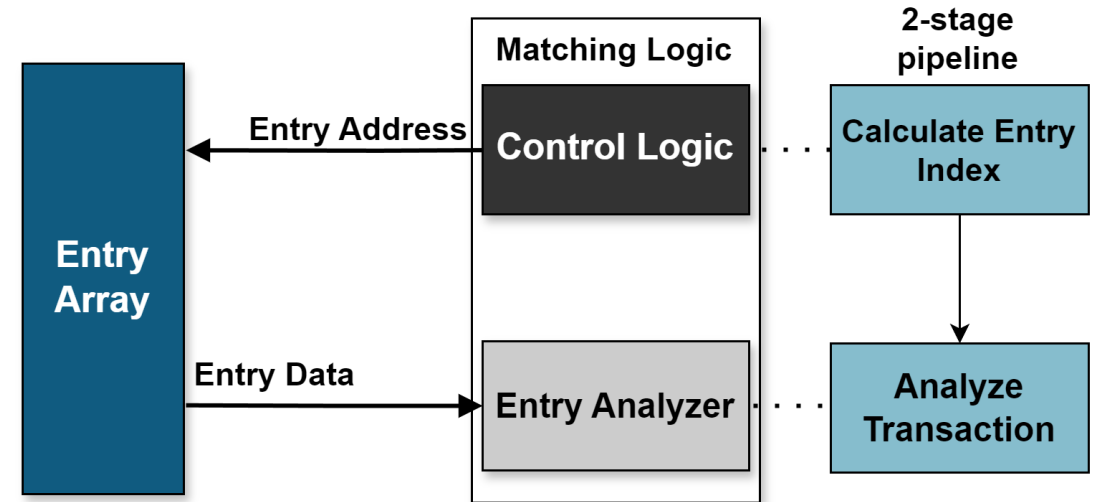
# Matching Logic

## Parallel Approach



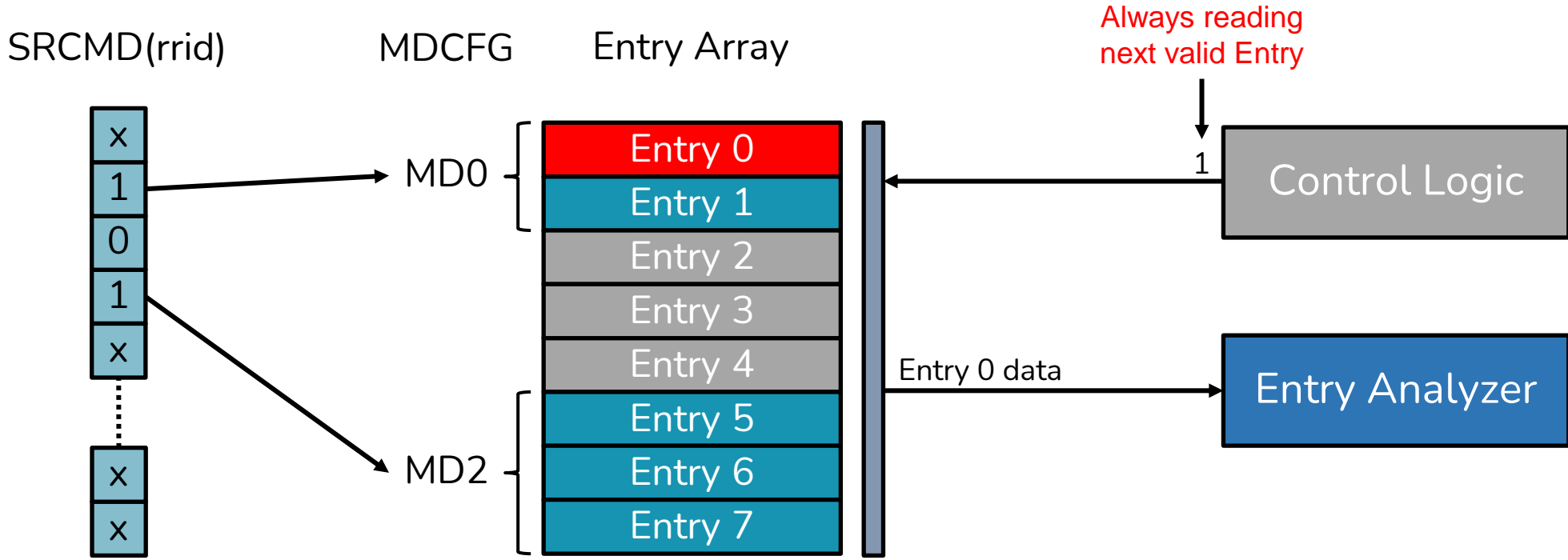
- + Performance
- - Scalability

## Sequential Approach

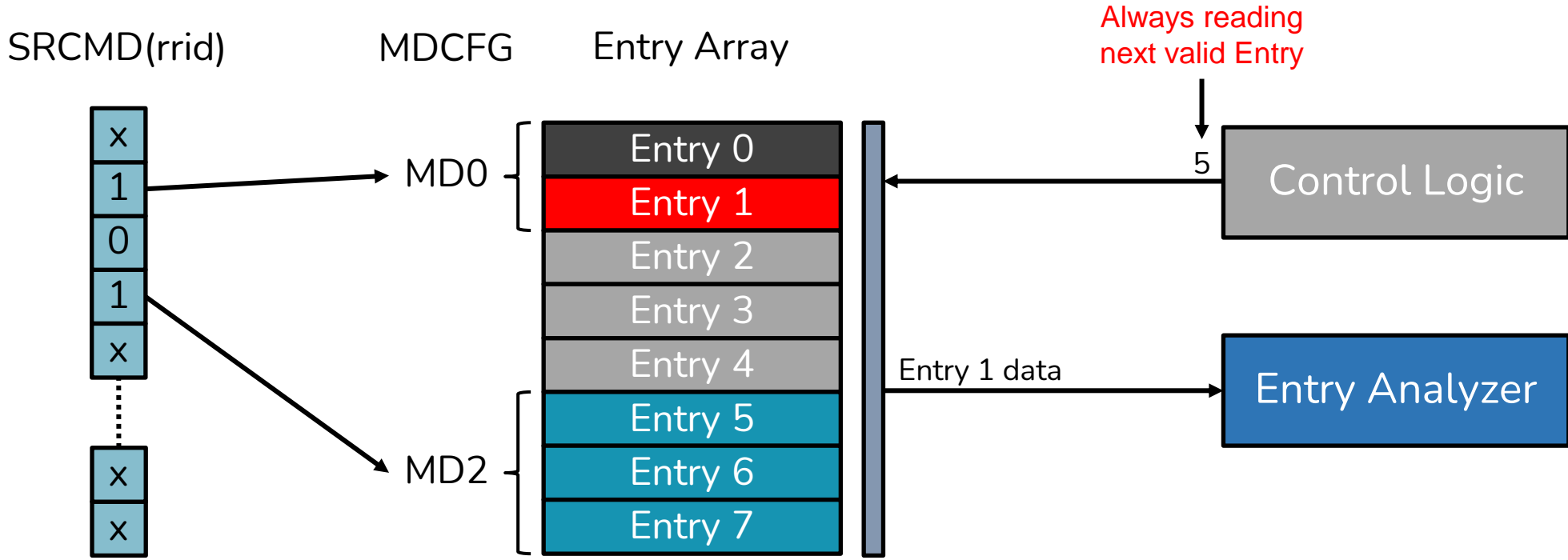


- - Performance
- + Scalability
- + Modularity

# Entry Navigation



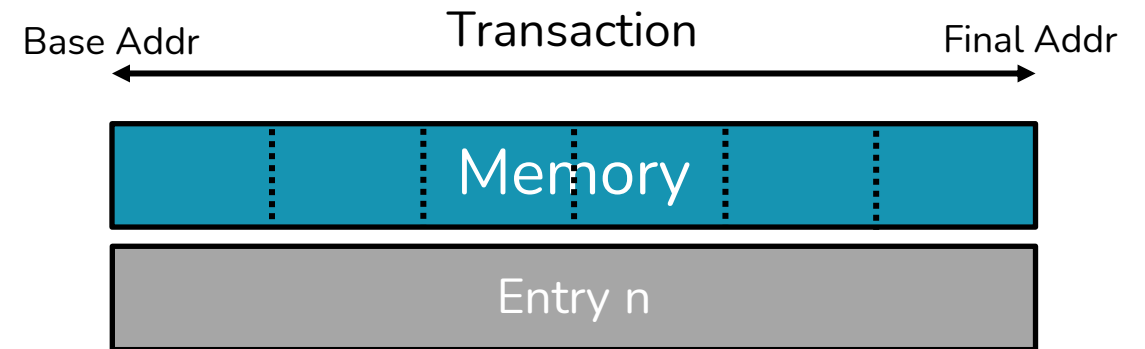
# Entry Navigation





# Transaction Validation

1. Entry Analyzer detects a match with Entry n with the base address of the transaction
2. Entry Analyzer checks if the entire length of the transaction is validated by Entry n
3. Entry Analyzer checks the permissions
4. As Entry n refers to the entire memory region of the transaction, the Entry analyzer signals a **match and allow**

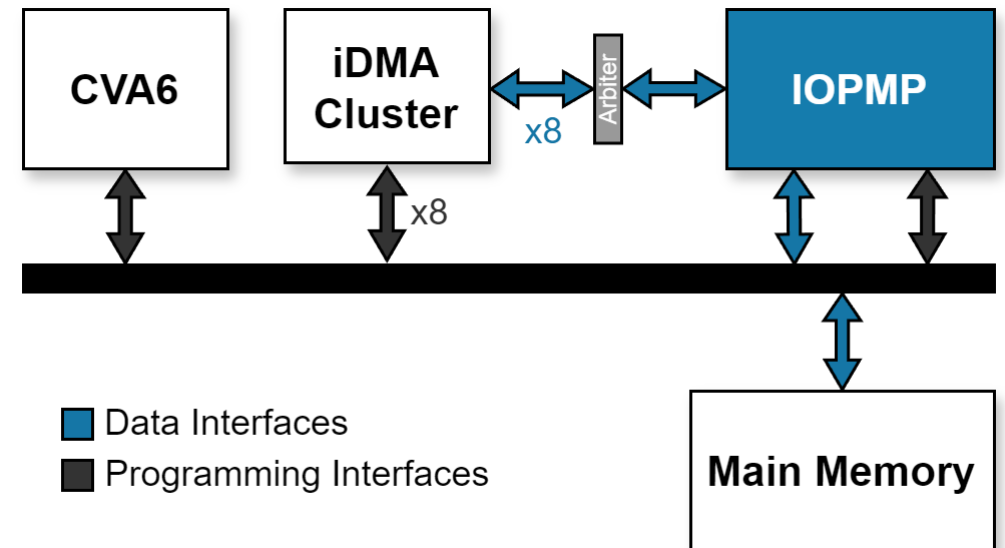


The entire **transaction** is validated within a clock cycle, after finding corresponding entry

# Evaluation

# Functional Evaluation

- Genesys2 FPGA @100MHz
- Two configurations of a single core CVA6-based SoC:
  - IOPMP in Source Enforcement managing 1 PULP iDMA device
  - IOPMP managing 8 PULP iDMA devices
- Validated SW stacks:
  - Custom Baremetal Framework
  - OpenSBI + Baremetal
  - OpenSBI + Linux



# Hardware Resources

Configurations feature:

- Same number of MDs and Supported Sources
- Contains 8 entries for each MD/Source

	Entries	Sources / MDs	LUT	FF	BRAM
<b>Baseline</b>	8	1	2179	1285	4
<b>x8</b>	64	8	2878 (32%)	1862 (45%)	4 (0%)
<b>x16</b>	128	16	3533 (62%)	2414 (88%)	4 (0%)
<b>x32</b>	256	32	5000 (129%)	3597 (180%)	4 (0%)
<b>x64</b>	512	64 / 63*	7149 (228%)	6346 (393%)	4 (0%)

\*Specification Maximum

# Hardware Resources

Configurations feature:

- Same number of MDs and Supported Sources
- Contains 8 entries for each MD/Source

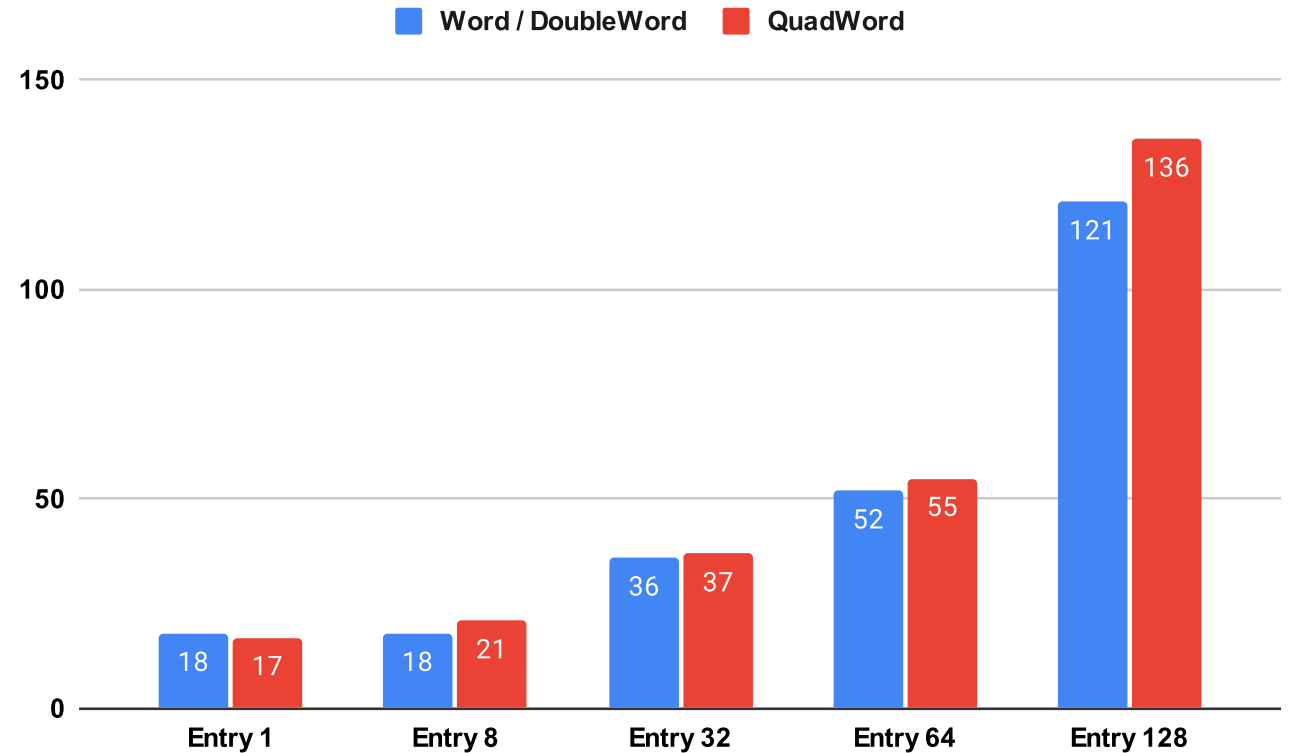
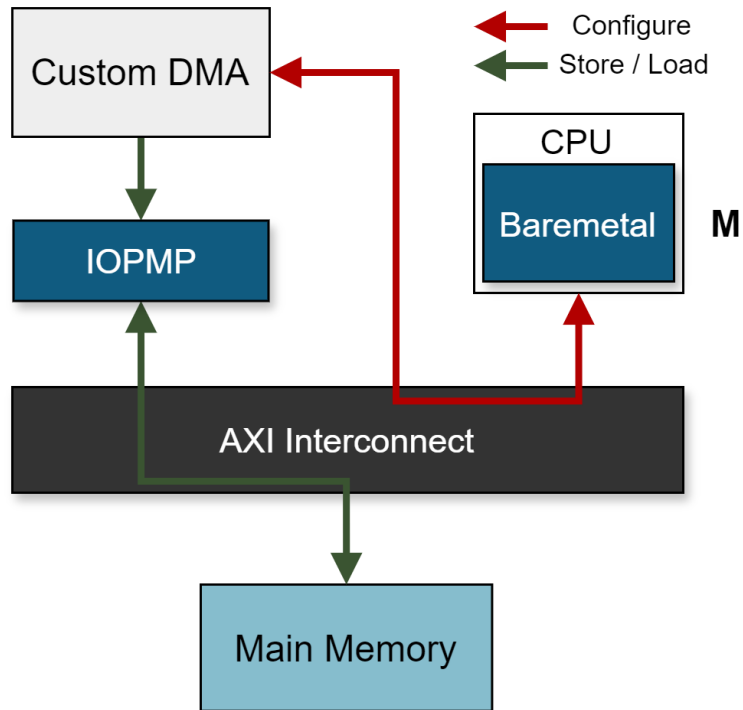
	Entries	Sources / MDs	LUT	FF	BRAM
<b>Baseline</b>	8	1	2179	1285	4
<b>x8</b>	64	8	2878 (32%)	1862 (45%)	4 (0%)
<b>x16</b>	128	16	3533 (62%)	2414 (88%)	4 (0%)
<b>x32</b>	256	32	5000 (129%)	3597 (180%)	4 (0%)
<b>x64</b>	512	64 / 63*	7149 (228%)	6346 (393%)	4 (0%)

↓  
3.5% LUT, 1,6% FF, and 0,9% BRAM increase on a CVA6-based SoC

\*Specification Maximum

# Latency Results

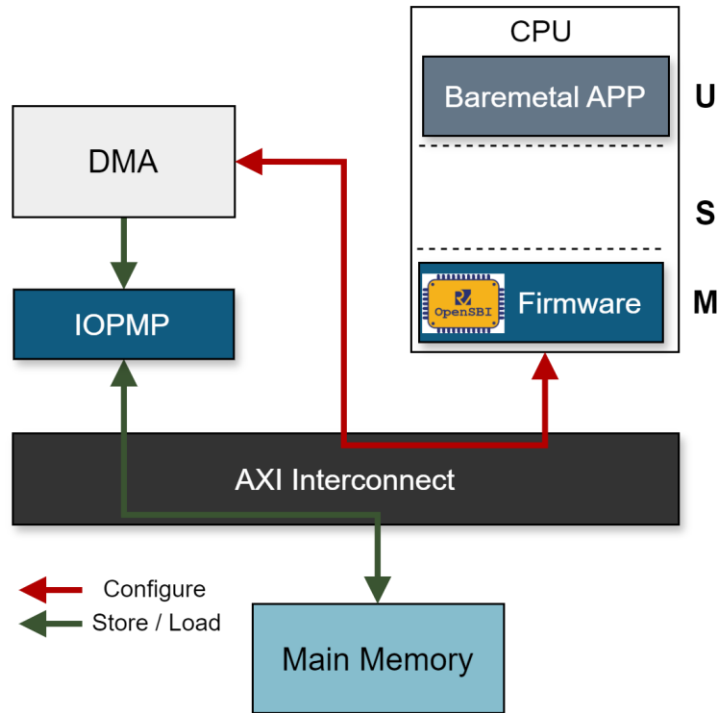
- Device performing independent store/load operations
- Results refer to the entry that matches the transaction



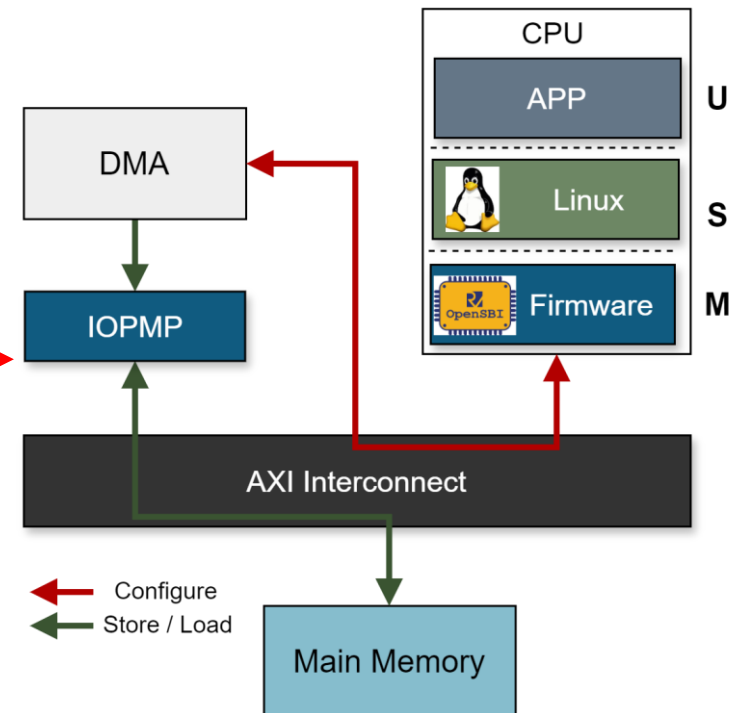
Clock Cycle penalty  
(average over 1000 transactions)

# Latency Results

## Baremetal + DMA



## Linux + DMA



Performance penalties only up to 3% with 64 entries

*With more complex SW stacks and more complex workloads, the performance penalty is less noticeable!*

TAKEAWAY



Design and implementation of the first  
**OPEN-SOURCE** IOPMP IP compliant  
with the version 1.0.0-draft5

# CALL TO ACTION!

The screenshot shows the GitHub interface for the repository `zero-day-labs / riscv-iopmp`. The repository is public and has 5 stars and 1 fork. The main content area displays a file tree for the `master` branch, which has 2 branches and 0 tags. The file tree includes folders like `doc`, `include`, `packages`, `rtl`, and `vendor`, and files like `.gitignore`, `LICENSE.Apache`, `LICENSE.Solderpad`, `Makefile`, `README.md`, and `lint_checks.sv`. A commit by `luisccc` is highlighted, with the message "fix: Minor fixes and compatibility improvements with the output inter...". The right sidebar shows repository details, including the README, license information (Apache-2.0), activity, and contributors. The contributors list includes `luisccc` (Luís Cunha) and `D3boker1` (Francisco Marques).

Product Solutions Resources Open Source Enterprise Pricing

Search or jump to... Sign in Sign up

zero-day-labs / riscv-iopmp Public

Notifications Fork 1 Star 5

Code Issues Pull requests Actions Projects Security Insights

master 2 Branches 0 Tags

Go to file Code

luisccc fix: Minor fixes and compatibility improvements with the output inter... f354739 · last month 3 Commits

doc	Initial commit	last month
include	Initial commit	last month
packages	Initial commit	last month
rtl	fix: Minor fixes and compatibility improvements with the out...	last month
vendor	Initial commit	last month
.gitignore	Initial commit	last month
LICENSE.Apache	Initial commit	last month
LICENSE.Solderpad	fix: Rename LICENSE.Solerpad to LICENSE.Solderpad	last month
Makefile	Initial commit	last month
README.md	Initial commit	last month
lint_checks.sv	Initial commit	last month

README Apache-2.0 license License

## RISC-V IOPMP

About

IOPMP IP

- Readme
- Apache-2.0, Unknown licenses found
- Activity
- Custom properties
- 5 stars
- 0 watching
- 1 fork

Report repository

Releases

No releases published

Packages

No packages published

Contributors 2

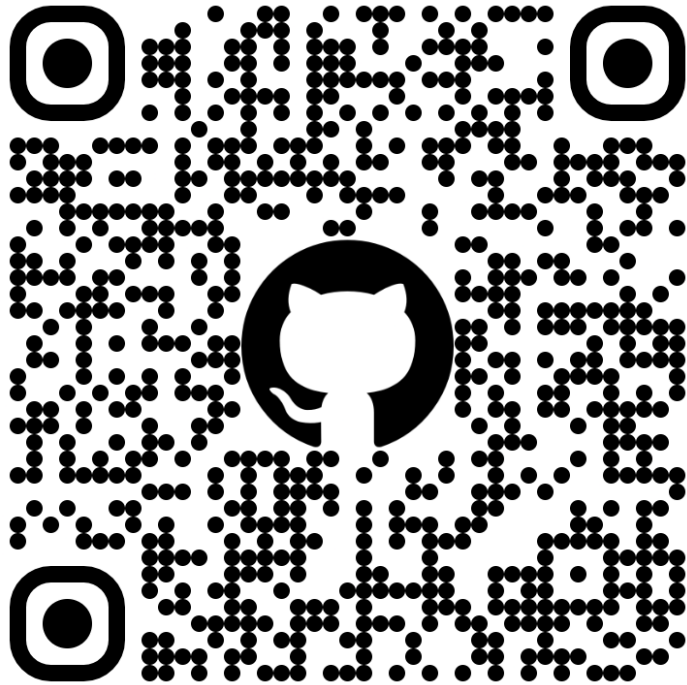
- luisccc Luís Cunha
- D3boker1 Francisco Marques

Languages

# CALL TO ACTION!

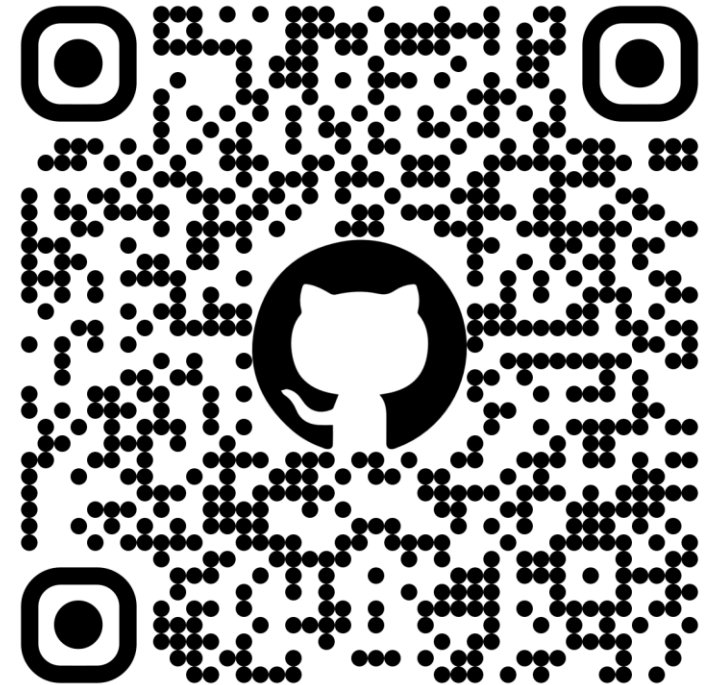
## Basic IP

<https://github.com/zero-day-labs/riscv-iopmp>



## CVA6 with IOPMP

<https://github.com/zero-day-labs/cva6/tree/feat/iopmp>



# THANK YOU!

Luís Cunha (UMinho)

*id11207@alunos.uminho.pt*