Protego: A Low-Overhead Open-Source I/O Physical Memory Protection Unit for RISC-V

Funding acknowledgement:
877056 - A Cognitive Fractal and Secure EDGE based on an unique Open-Safe-Reliable-Low Power Hardware Platform Node (EC)

Protego: A Low-Overhead Open-Source I/O Physical Memory Protection Unit for RISC-V

Nils Wistoff
Integrated Systems Laboratory
ETH Zürich
Zürich, Switzerland
nwistoff@iis.ee.ethz.ch

Andreas Kuster
WANDS
Nanyang Technological University
Singapore, Singapore
mail@andreaskuster.ch

Michael Rogenmoser
Integrated Systems Laboratory
ETH Zürich
Zürich, Switzerland
michaero@iis.ee.ethz.ch

Robert Balas
Integrated Systems Laboratory
ETH Zürich
Zürich, Switzerland
balasr@iis.ee.ethz.ch

Moritz Schneider
Institute of Information Security
ETH Zürich
Zürich, Switzerland
moritz.schneider@inf.ethz.ch

Luca Benini
Integrated Systems Laboratory
ETH Zürich and University of Bologna
Zürich, Switzerland
lbenini@iis.ee.ethz.ch

***Abstract*—Physical memory protection is a hardware mechanism designed to prevent unauthorized access to specific memory regions, enabling the deployment of Trusted Execution Environments (TEEs). The RISC-V instruction set architecture specifies PMP for RISC-V cores but leaves other system bus masters as found in heterogeneous computing systems out of scope. This work presents Protego, an open-source I/O physical memory protection (IOPMP) unit based on the RISC-V PMP specification that extends PMP to other system bus masters. We demonstrate that Protego is effective in protecting sensitive data in memory and preventing unauthorized access at small hardware costs of below 40 kGE for a 64-bit system and negligible performance impact, making it a valuable tool for creating TEEs in heterogeneous computing systems.**

***Index Terms*—security, heterogeneous computing, physical memory protection, trusted execution environment, RISC-V**

I. Introduction

As computing systems become more complex and multi-functional, the potential for bugs and security vulnerabilities increases with an ever-growing software stack. Additionally, these systems are frequently entrusted with confidential information while simultaneously executing tasks of varying levels of trustworthiness.

A widely-employed strategy to address this threat is minimizing the trusted code base as much as possible. This objective can be accomplished through utilizing trusted execution environments (TEE) that rely on hardware-supported memory protection in order to isolate an execution environment from a potentially untrustworthy operating system (OS). The RISC-V privileged specification [1] describes support for a memory protection mechanism called physical memory protection (PMP), and several RISC-V implementations have incorporated it into their designs [2].

This work has been supported in part by 'Fractal' project under grant agreement No 877056 that receives funding from ECSEL-JU as part of the EU Horizon 2020 research and innovation programme, and in part by the ETH4D Humanitarian Action Challenges Application on "Secure Infrastructure for Humanitarian Organizations".

In a heterogeneous computing system, other system bus masters besides RISC-V cores may co-exist. For instance, these could be direct memory access (DMA) engines, other accelerators, non-RISC-V-compliant processors, or off-chip master ports. The existing RISC-V PMP specification does not apply to such devices. Therefore, the RISC-V community has proposed and discussed an I/O physical memory protection unit (IOPMP), which imposes memory protection rules for such peripheral devices [3]. Figure 1 illustrates such a system configuration.

In this work, we present Protego, an open-source IOPMP unit for the RISC-V ecosystem implemented in SystemVerilog. We demonstrate its effectiveness in protecting system memory from unauthorized access by all masters in the system and show that Protego has a low hardware overhead while not impacting the overall system performance.

The remainder of this paper is structured as follows: Section II elaborates the background and related work. Section III presents the proposed architecture of Protego. We evaluate the hardware overhead and performance implications of this implementation in Section IV. Finally, Section V concludes this work.

II. Background and Related Work

A. Trusted Execution Environments

Trusted Execution Environments (TEEs) provide secure isolation of sensitive computing tasks [4]–[6]. Their key idea is to reduce the trusted computing base (TCB) to an absolute minimum, e.g., by excluding the underlying operating system (OS) or hypervisor from the TCB. Not only software but also external attackers and devices such as peripherals are supposed to be restricted from accessing code and data inside the TEE using dedicated hardware mechanisms.

On RISC-V, the Keystone framework [7] provides TEEs using physical memory protection (PMP) [1]. Schneider et al. have ported PMP and Keystone to CVA6, an open-source,

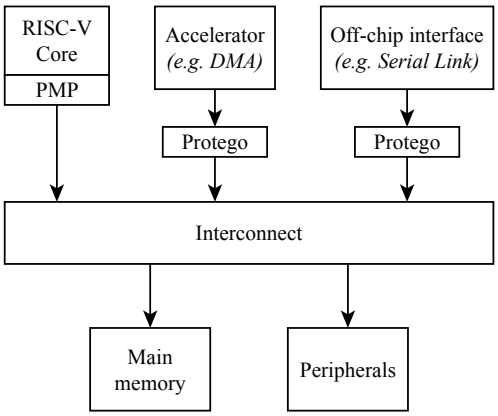

Fig. 1. Example illustration of a heterogeneous system equipped with Protego.

application-class, 64-bit RISC-V core [2], [8]. These additions focus on systems where PMP-capable RISC-V cores are the only bus masters. To protect systems with heterogeneous interconnect masters in a similar fashion, PMP needs to be extended to those non-compliant masters.

B. RISC-V IOPMP Proposal

Concurrently to this work, the RISC-V community has drafted a first version for a RISC-V IOPMP specification [3], [9]. The proposal designs the IOPMP unit to be placed on the slave side of the memory interconnect, directly before the protected slave. As a consequence, each master's memory requests are tagged with a source ID (SID).

In contrast, Protego is intended to be placed on the master side of the interconnect, see Figure 1. This prevents illegal memory accesses from entering the interconnect in the first place, mitigating potential denial-of-service attacks by a malicious AXI master.

An in-depth comparison of both architectures, including their respective hardware implications, is an item for future work.

III. Architecture

Protego features a slave and a master AXI port, as well as a configuration register interface. The slave AXI port connects to an otherwise unprotected AXI master. The master AXI port forwards allowed transactions to the downstream memory system, e.g., by connecting to an AXI interconnect. The configuration register interface is used for setting up the PMP rules enforced by Protego.

A. Configuration

Protego's configuration registers are adapted from the conventional RISC-V PMP control and status registers (CSRs) defined in the RISC-V privileged specification [1], which defines a parametric number of entries. For each such IOPMP entry x, there are two configuration registers as shown in Figure 2:

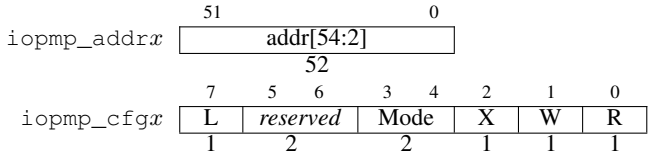

Fig. 2. Protego's configuration registers for entry x. Adapted from the RISC-V PMP specification [1].

`iopmp_addr`x holds the base address (and size) of the memory range that the corresponding IOPMP entry applies to, right-shifted by two bits. The encoding corresponds to that of `pmpaddr` in the RISC-V privileged specification [1].

`iopmp_cfg`x has five fields: The 'R', 'W', 'X' fields define the read, write, and execute permissions, respectively. 'Mode' holds the IOPMP entry's mode and can be set to disabled (OFF, `0`), top of range (TOR, `1`), naturally aligned four-byte region (NA4, `2`, only selectable if the IOPMP's granularity is 4 bytes) and naturally aligned power-of-two region (NAPOT, `3`). These modes are detailed in the RISC-V privileged specification [1]. Finally, 'L' can be set to lock the IOPMP entry until the system is reset.

During startup, a trusted, privileged entity, such as a secure monitor, can set up system-wide physical memory protection by configuring all PMPs (RISC-V cores) and IOPMPs (other system bus masters). This includes the address range of the memory-mapped IOPMP configuration registers themselves to ensure that the IOPMPs cannot be reconfigured by malicious masters. The IOPMP configuration and locked PMP entries become effective immediately, while non-locked PMP rules only take effect after dropping into lower-privileged, untrusted software.

B. Implementation

Internally, Protego features two PMP units that check the legality of any read or write request against the PMP configuration. Burst requests are checked for not exceeding a single 4 KiB-aligned memory range, as demanded by the AXI specification. If the checks pass, an AXI Demux [10] forwards the request to the AXI master port of Protego. Otherwise, it sends the request to an Error Slave, which returns an error response over the AXI slave port to the AXI master.

IV. Evaluation

A. Functional Evaluation

To evaluate the efficacy of our proposed unit, we instantiate a system featuring two memory bus masters: a PMP-equipped 64-bit RISC-V core (CVA6 [8] and a DMA engine (iDMA [11]) protected by Protego. We execute a DMA attack *without* and *with* IOPMP in place. The output of this program is shown in Listing 1, indicating the following sequence of events:

a) Lines 1-7: The test is initialized. A 4 KiB-aligned source value is initialized to `0x2A`, while a second 4 KiB-aligned destination array is zeroed. The PMP and IOPMP are probed.

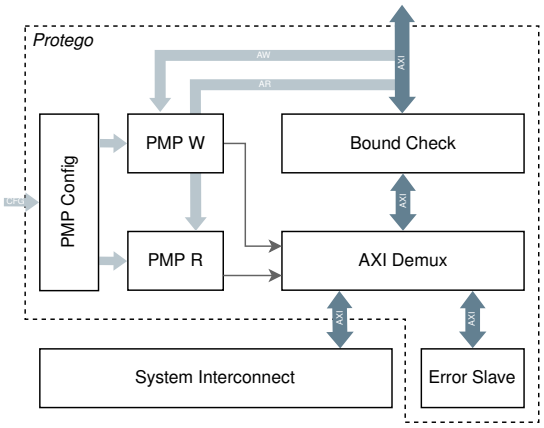

Fig. 3. Protego architecture.

Listing 1
DMA ATTACK OUTPUT.

```
Hello CVA6 from iDMA!
Source array @ 0x0000000080FFE000
Destination array @ 0x0000000080FFD000
Detect PMP: PMP0 detected
PMP granularity: 00000008
IO-PMP0: detected
IO-PMP granularity: 00001000
iopmp_addr0: 003FFFFFFFFFFFF
iopmp_cfg0: 000000000000001F
Test register read/write
Initiate dma request
Start transfer
transfer_id: 00000002
done_id: 00000002
dst[0]: 0000002A
Transfer finished
Try reading dst: 0x000000000000002A
Try reading dst: 0x000000000000002A
Transfer successfully validated.
Reset destination array.
IO-PMP: Lock src array.
iopmp_addr0: 00000000203FF9FF
iopmp_cfg0: 0000000000000018
Initiate dma request
Start transfer
transfer_id: 00000003
done_id: 00000003
dst[0]: 00000000
Transfer finished
Try reading dst: 0x0000000000000000
assertion failed: dst
Spin-loop.
```

b) Lines 8-9: Protego is configured to allow all accesses (feed-through) by setting all bits of `iopmp_addr0` (*match all*), setting the mode of entry 0 to NAPOT and permitting reads, writes, and execution.

c) Lines 10-19: iDMA is programmed to load the source array into the destination array. The transfer succeeds, as the destination now reads `0x2A`.

d) Lines 20-23: The destination array is reset (zeroed), and Protego is configured to disallow memory access to the source array by setting `iopmp_addr0` to the source array's address, right-shifted by two (the trailing ones indicate a 4 KiB range), and by setting `iopmp_cfg0.Mode` to NAPOT and clearing the read, write, and execute flags.

e) Lines 24-31: Once again, iDMA is programmed to load from the source to the destination array. As expected, the access fails this time, as the destination array still reads `0` after the transfer.

Hence, we conclude that Protego is effective in protecting the memory region of the source array. Steps to reproduce this experiment, including its source code, are available online at https://github.com/pulp-platform/axi-io-pmp.

TABLE I
DEFAULT PARAMETERS OF PROTEGO USED FOR EVALUATION.

Parameter	Value
AXI4 address width	64 bits
AXI4 data width	64 bits
AXI4 ID width	4 bits
AXI4 user width	1 bit
Inflight transactions	4
Number of IOPMP entries	16
IOPMP granularity	4 KiB

B. Hardware Costs

To evaluate the hardware overhead of Protego, we synthesize the design in GLOBALFOUNDRIES 22FDX technology in typical corners (0.8 V / 25 °C). We assume the default parameters for Protego as listed in Table I. We vary the clock frequency between 500 MHz and 2 GHz. We constrain the I/O delays to 30 % of the clock period.

Figure 4 shows the Area of the design constrained to different clock period constraints and converted into kilo gate equivalent (kGE). It ranges from approximately 36 kGE for a 500 MHz synthesis run up to 42 kGE at 2 GHz.

To study the contributions of the different sub-components of Protego to the overall design area, we run synthesis with cycle time constraint of 700 ps (1.4 GHz frequency) while preserving the hierarchical module boundaries. The corresponding area breakdown is shown in Figure 5. The main contributors to Protego's design area are the configuration registers and the two PMP units, which occupy approximately 30 % of the total design area, each. The remaining area is used by the AXI Demux, the AXI error slave, and glue logic in the top level.

C. Performance Implications

To analyze Protego's performance implications in terms of cycle latency and throughput, we focus on the path between the AXI master and slave ports. Most components on this path, particularly the PMP units and the bound check, are purely combinatorial, meaning that they process and propagate beats within the same clock cycle. The AXI Demux [10] contains sequential state to track in-flight transactions and may stall when the selector changes while there are in-flight transactions downstream to preserve ordering and comply with the AXI4 protocol. In the case of Protego, such a stall may occur after accessing an illegal memory address, which causes the aforementioned switch of the AXI Demux select. We

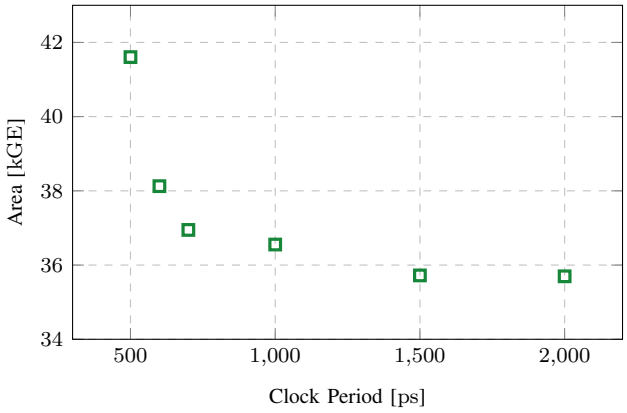

Fig. 4. Area versus timing of Protego.

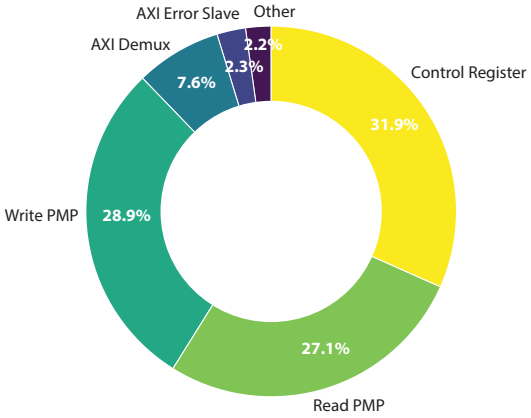

Fig. 5. Hierarchical area breakdown of Protego at 1.4 GHz.

assume that in the common case, AXI masters generally access permitted memory regions, meaning that the AXI Demux maintains a direct feed-through connection between Protego's AXI master and slave ports. This implies a zero-cycle latency overhead and full throughput as long as all accesses succeed. If Protego lies on a critical path on a system level, it may be desirable to add an AXI Cut [10] before or after the module. This would prevent Protego from introducing timing violations at additional hardware costs and an extra cycle of latency.

V. Conclusion

In this paper, we present Protego, a fully open-source, zero-cycle latency IOPMP unit that enables physical memory protection in heterogeneous computing systems. We demonstrate that Protego is capable of preventing DMA attacks that bypass conventional PMPs while allowing for full throughput and no added latency. Protego's hardware overhead is small at approximately 40 kGE, and it can be synthesized at up to 2 GHz in GlobalFoundries 22FDX technology.

To the best of our knowledge, Protego is the first open-source hardware implementation of an AXI-compliant IOPMP. There is significant potential for further exploration, for instance, by comparing Protego to the recently drafted RISC-V IOPMP specification and the IOPMP architectures based on it.

The source code of Protego is available online at https://github.com/pulp-platform/axi-io-pmp.

References

[1] A. Waterman, Y. Lee, D. Patterson, and K. Asanovic, "The RISC-V instruction set manual," *Volume II: Privileged Architecture*, vol. 2, 2016.

[2] M. Schneider, A. Dhar, I. Puddu, K. Kostiainen, and S. Capkun, "Composite enclaves: Towards disaggregated trusted execution," *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, vol. 2022, pp. 630–656, 2020.

[3] P. Ku, C. Tang, and RISC-V IOPMP Task Group, "RISC-V IOPMP specification document," 2023. [Online]. Available: https://github.com/riscv-admin/iopmp/blob/main/specification/riscv_iopmp_specification.pdf

[4] V. Costan and S. Devadas, "Intel SGX explained," *IACR Cryptology ePrint Archive*, vol. 2016, p. 86, 2016.

[5] V. Costan, I. A. Lebedev, and S. Devadas, "Sanctum: Minimal hardware extensions for strong software isolation," in *USENIX Security Symposium*, 2016.

[6] J. Winter, "Trusted computing building blocks for embedded Linux-based ARM Trustzone platforms," in *Scalable Trusted Computing*, 2008.

[7] D. Lee, D. Kohlbrenner, S. Shinde, K. Asanović, and D. X. Song, "Keystone: an open framework for architecting trusted execution environments," *Proceedings of the Fifteenth European Conference on Computer Systems*, 2020.

[8] F. Zaruba and L. Benini, "The cost of application-class processing: Energy and performance analysis of a Linux-ready 1.7-GHz 64-bit RISC-V core in 22-nm FDSOI technology," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 11, pp. 2629–2640, 2019.

[9] Y. Chen, H. Chen, S. Chen, C. Han, W. Ye, Y. Liu, and H. Zhou, "DITES: A lightweight and flexible dual-core isolated trusted execution soc based on RISC-V," *Sensors*, vol. 22, no. 16, 2022.

[10] A. Kurth, W. Rönninger, T. Benz, M. Cavalcante, F. Schuiki, F. Zaruba, and L. Benini, "An open-source platform for high-performance non-coherent on-chip communication," *IEEE Transactions on Computers*, vol. 71, no. 8, pp. 1794–1809, 2022.

[11] T. Benz, M. Rogenmoser, P. Scheffler, S. Riedel, A. Ottaviano, A. Kurth, T. Hoefler, and L. Benini, "A high-performance, energy-efficient modular DMA engine architecture," *Unpublished*, 2023.