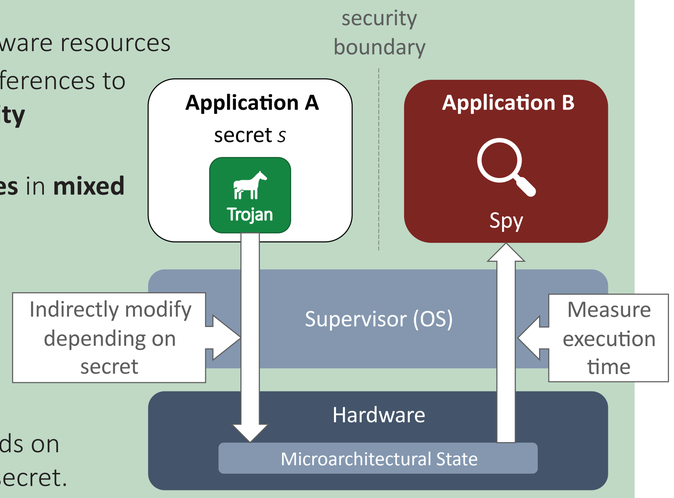
ISA Support for Hardware Resource Partitioning in RISC-V

Nils Wistoff[1], Robert Balas[1], Alessandro Ottaviano[1], Gernot Heiser[2], Luca Benini[1,3]
[1]ETH Zürich, [2]UNSW Sydney, [3]University of Bologna

1. Problem: Timing Interference

General Concept:
- Applications **compete** for shared hardware resources
- **Timing channels** levarage timing interferences to transfer information, **bypassing security boundaries**.
- Interference can cause **deadline misses** in **mixed criticality systems**.

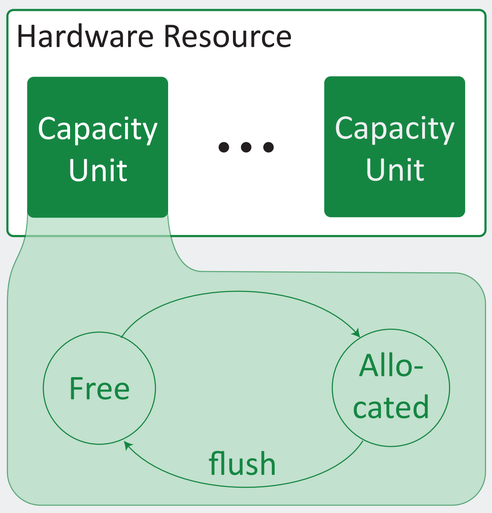
Timing Channel Example:
- Trojan: utilise cache depending on secret.
- Spy: measure execution time.
- Spy's measured execution time depends on Trojan's cache utilisation depends on secret.



2. Solution: Temporal & Spatial Partitioning

Split every hardware resource into one or more **capacity units** that can be allocated to a domain. (*spatial partitions*)

Capacity units are **flushed** when being **reallocated**. (*temporal partitioning*)



3. Extending CBQRI

CBQRI: RISC-V Capacity and Bandwidth Controller Quality-of-Service Register Interface [1]
- Proposes to split HW resources into capacity units.
- Memory requests are tagged with recource control IDs (RCIDs)

\+

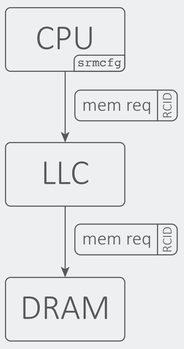
Flushable capacity units: Mechanism(s) to write back dirty state and clear capacity unit.

Lifecycle: Flush capacity units between allocations.

No inter-block accesses: Do not return hits from a capacity unit allocated to another domain.

System-level ubiquity: *All* HW components comprise one or more flushable capacity units.

Time padding: Mechanism(s) to enforce constant context-switch latency.



Enable *temporal partitioning*

Guarantee strong *spatial partitioning*

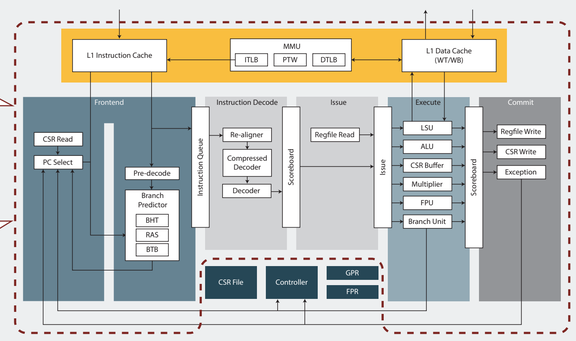
Prevent interference through *any* component

Prevent interference through *context-switch latency*

4. Example: CPU

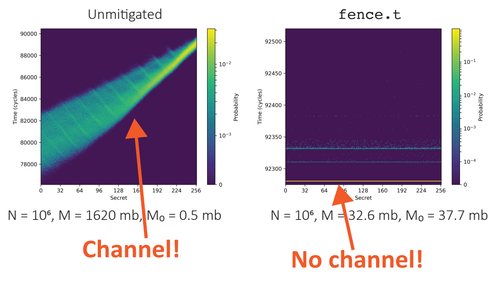
Consider CPU microarchitecture as a resource with a **single capacity unit**

`fence.t` instruction: **flush on context switch** (*temporal partitioning*) [3]



Closes all observed timing channels [3]

L1D channel bench results [2]

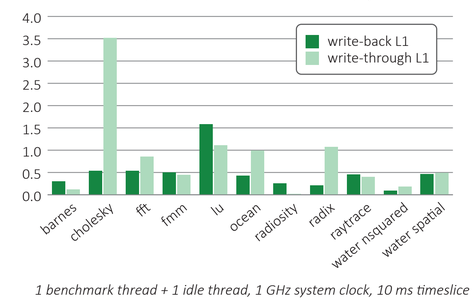


N = 10^6, M = 1620 mb, M_0 = 0.5 mb — **Channel!**

N = 10^6, M = 32.6 mb, M_0 = 37.7 mb — **No channel!**

Minimal performance impact [3]

Splash-2 benchmark overhead (%)



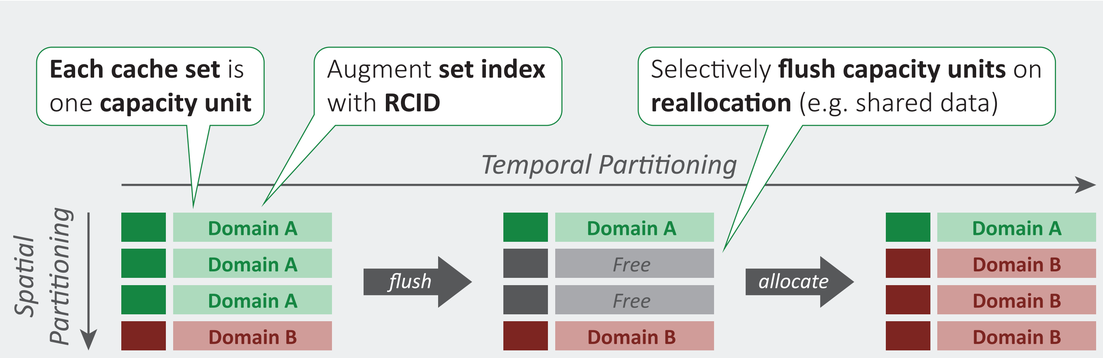
1 benchmark thread + 1 idle thread, 1 GHz system clock, 10 ms timeslice

5. Example: Last-Level Cache

Each cache set is one **capacity unit**

Augment **set index** with **RCID**

Selectively **flush capacity units** on **reallocation** (e.g. shared data)



6. Conclusions & Future Work

Conclusion:
- We propose a **minimal ISA extension** for **interference-free HW resource partitioning**.
- Preliminary results suggest **low hardware costs** and a **minimal performance overhead**.

Future Work:
- Extend evaluation to **further system components**. (e.g. AXI [4], LLC, DRAM controller)
- Create consistent **semantics** and **specification**.

References

[1] RISC-V CBQRI Task Group. "RISC-V Capacity and Bandwidth QoS Register Interface Version 1.0-rc3". 2024. url: https://github.com/riscv-non-isa/riscv-cbqri/releases/download/v1.0-rc3/riscv-cbqri.pdf
[2] Qian Ge, Yuval Yarom, and Gernot Heiser. "No Security Without Time Protection: We Need a New Hardware-Software Contract". In: APSys'18. ACM, 2018, 1:1–1:9. doi: 10.1145/3265723.3265724.
[3] Nils Wistoff, Moritz Schneider, Frank K. Gürkaynak, Gernot Heiser, and Luca Benini. "Systematic Prevention of On-Core Timing Channels by Full Temporal Partitioning". In: IEEE Trans. Comput. 72.5 (2023), pp. 1420–1430. doi: 10.1109/TC.2022.3212636.
[4] Thomas Benz, Alessandro Ottaviano, Robert Balas, Angelo Garofalo, Francesco Restuccia, Alessandro Biondi, Luca Benini. "AXI-REALM: A Lightweight and Modular Interconnect Extension for Traffic Regulation and Monitoring of Heterogeneous Real-Time SoCs". In: DATE'24. IEEE, 2024. doi: 10.3929/ethz-b-000642320.

e-mail: nwistoff@iis.ee.ethz.ch