Redundancy Schemes for PULP Systems

European Test Symposium (ETS) 2023

Special Session on dependable RISC-V

Yvan Tortorella yvan.tortorella@unibo.it

Ph.D. Student at University of Bologna

Supervisors: Francesco Conti, Luca Benini

PULP Platform

Open Source Hardware, the way it should be!

@pulp_platform

pulp-platform.org

youtube.com/pulp_platform

Motivation

Increasing demand for strong processing capabilities in space

- Image processing
- On-board orbit determination and reconfiguration
- Fault Detection, Isolation and Recovery (FDIR)

Cosmic rays badly affect on-board computers

- Single Event Upset, Single Event Transient, Transient Faults
- Cannot be deleted, only attenuated!

If we cannot get rid of transient faults, we must tolerate them!

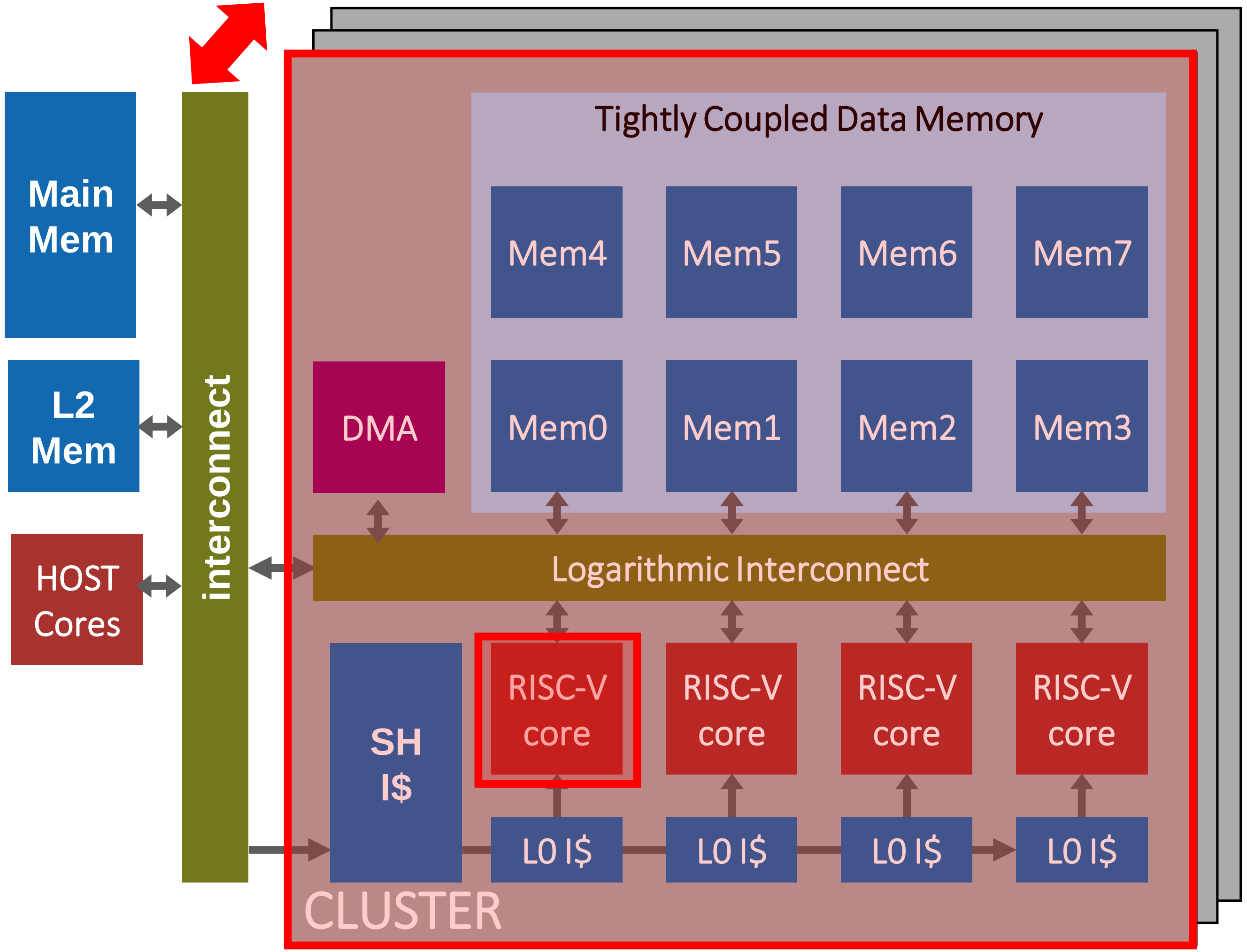
Yes, but at what cost?

- Software approaches: easy, but time consuming (bad performance)
- Hardware approaches: good performance, require open platforms

PULP: Open-source, open HW/SW development!

PULP: Energy-Efficient Computing System

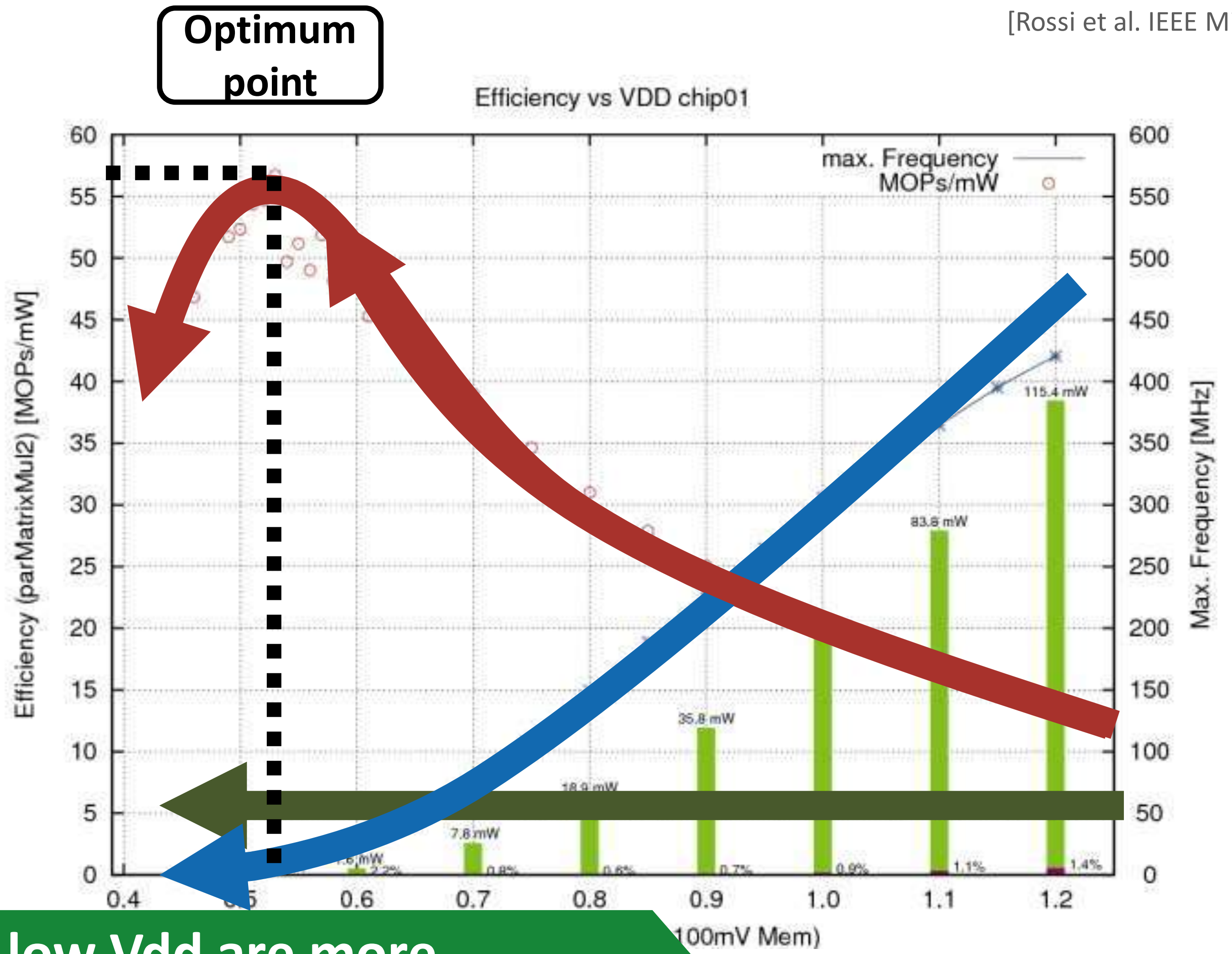
- Parallel Ultra-Low-Power
- Core
  - Improving core efficiency with ISA and uAch extensions
- Cluster
  - Efficient shared-mem cluster
  - From a few to thousand processing element
- Full platform
  - Heterogeneity: host, processor, accelerator
  - One or multiple accelerators
  - From chips to chiplets (2D to 3D)



Why parallel? Performance + energy eficiency

[Rossi et al. IEEE Micro 2017]

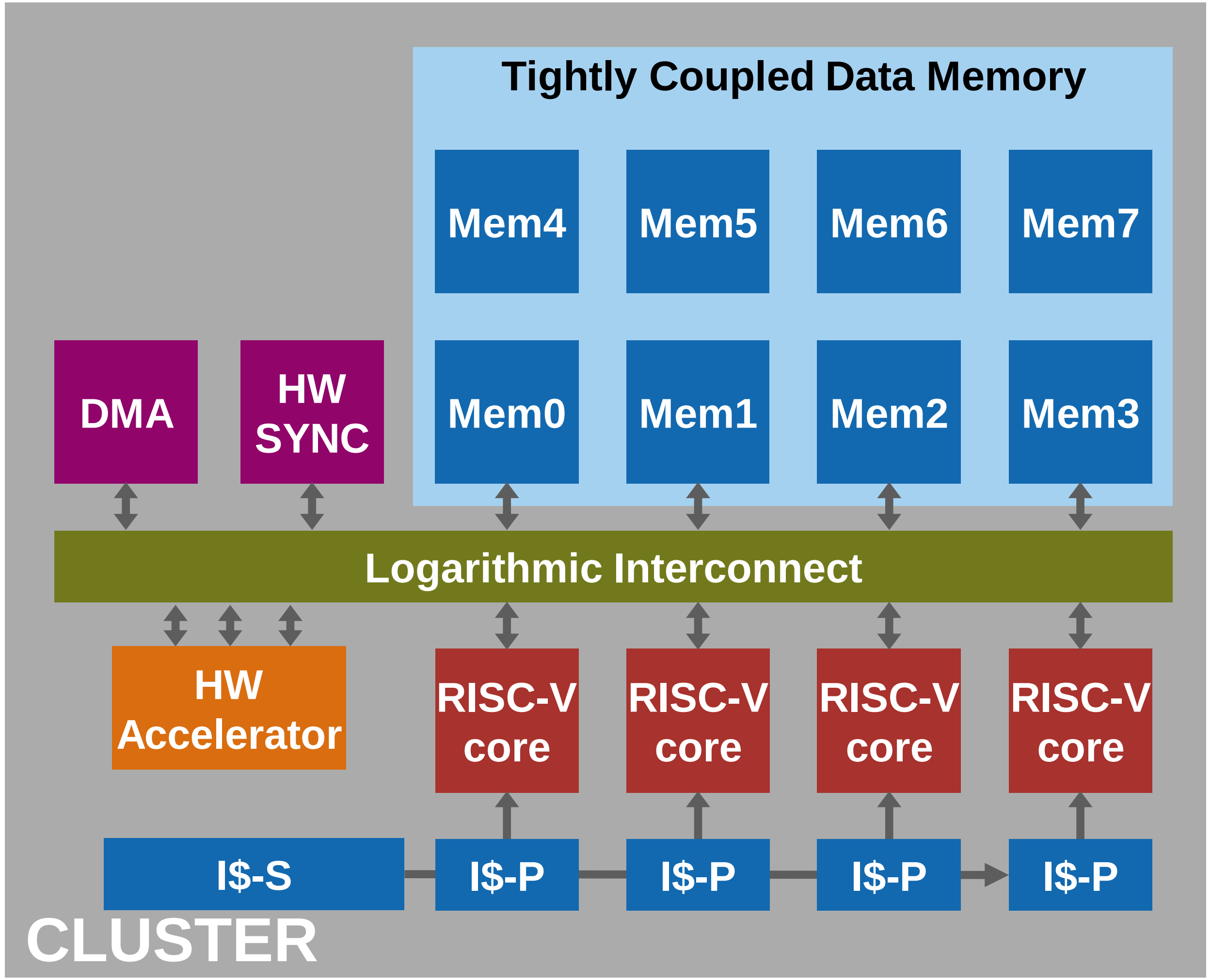
- As VDD decreases, operating speed decreases as well.
- However efficiency increases→ more work done per Joule
  - Until leakage effects start to dominate
- Put more units in parallel to get performance up and keep them busy with a parallel workload



N cores running at moderate f, low Vdd are more energy efficient than a single core at N×f, high Vdd

Designing a PULP multi-core cluster

- Multiple (2-16) parallel computing cores
- Tightly Coupled Data Memory (TCDM) with low-latency interconnect
- Hierarchical Instruction Cache
  - Speeds up instruction fetching
- DMA
  - Facilitates memory transfers
- Event Unit
  - Facilitates core synchronization
- Dedicated HW accelerators

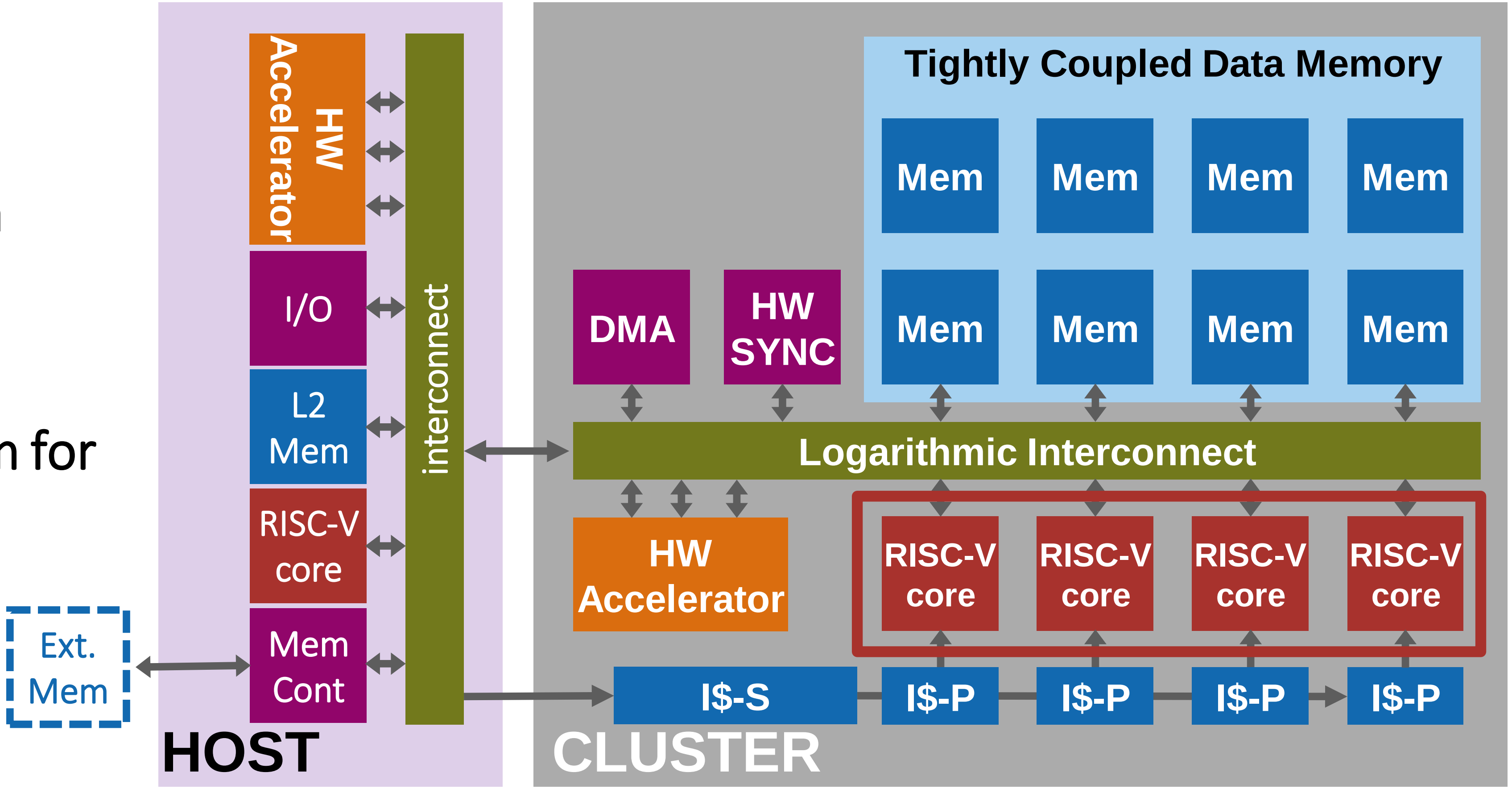


From a cluster to a full platform!

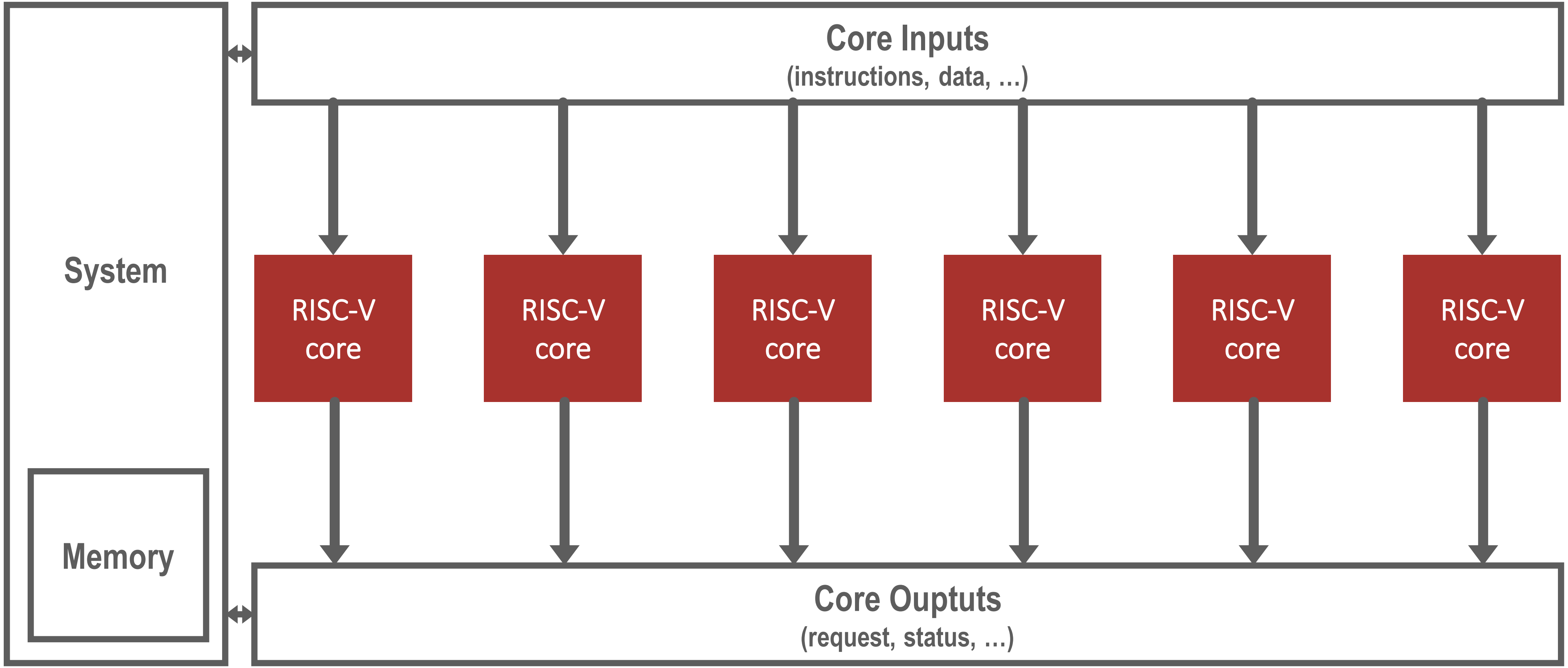
But how do we protect it?

How do we move from a cluster to a system?

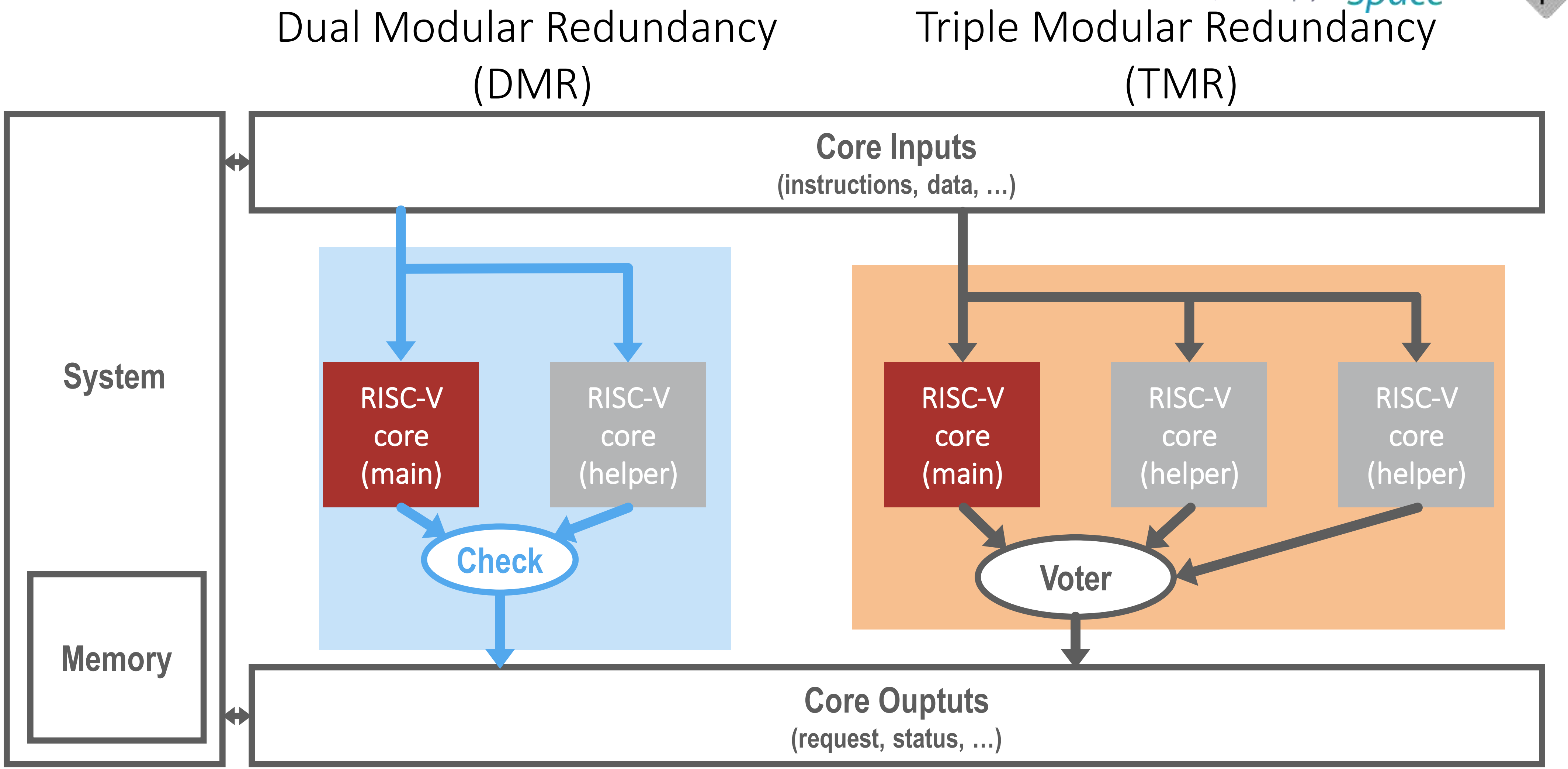
Connecting a host system for decision-making



Protecting the Cores



Redundant Cores Grouping: how to choose?



Are we really sacrificing performance and area/energy efficiency?

Maybe not all the computing tasks in space have the same level of criticality

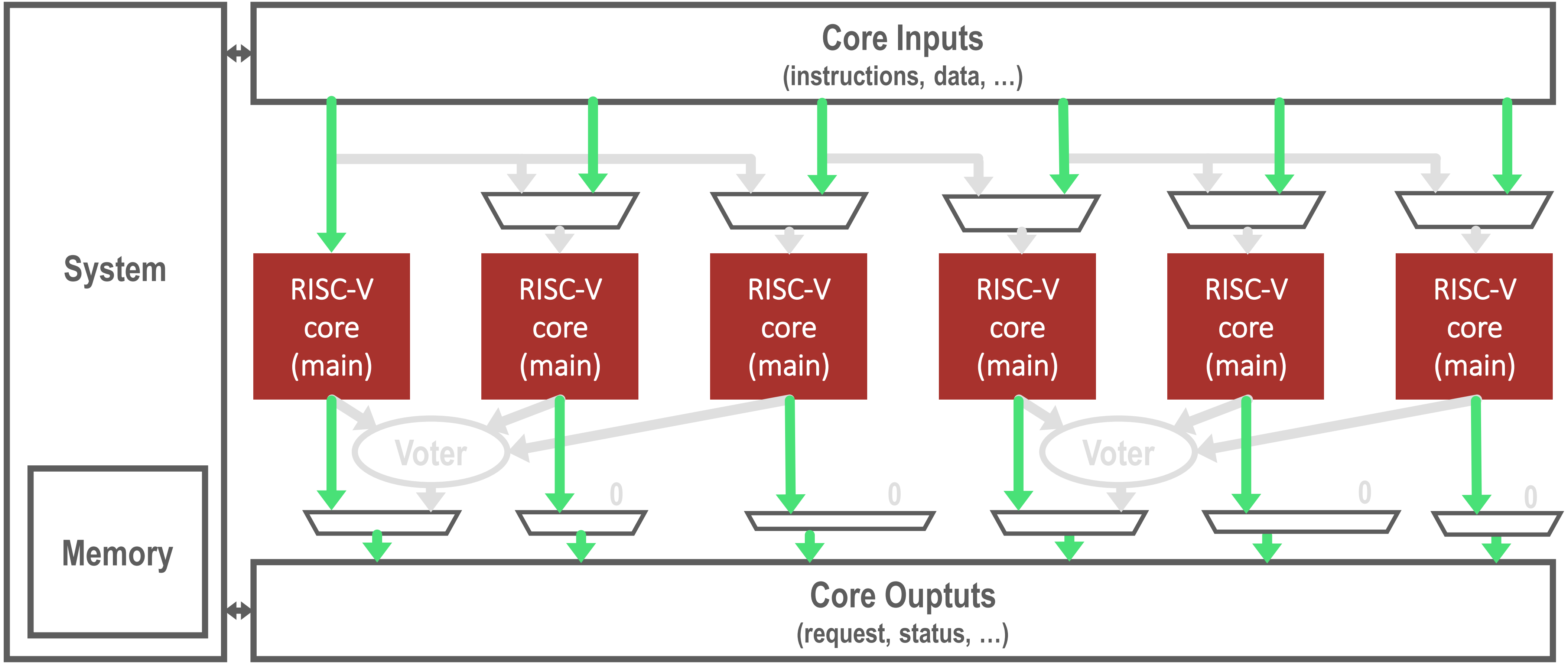
Maybe we are not even in space when we start computing...

An ETS23 example: maybe not all the layers of a Neural Networks need a high level of reliabi

We strive for flexibility! → Reconfiguration: **Hybrid Modular Redundancy**

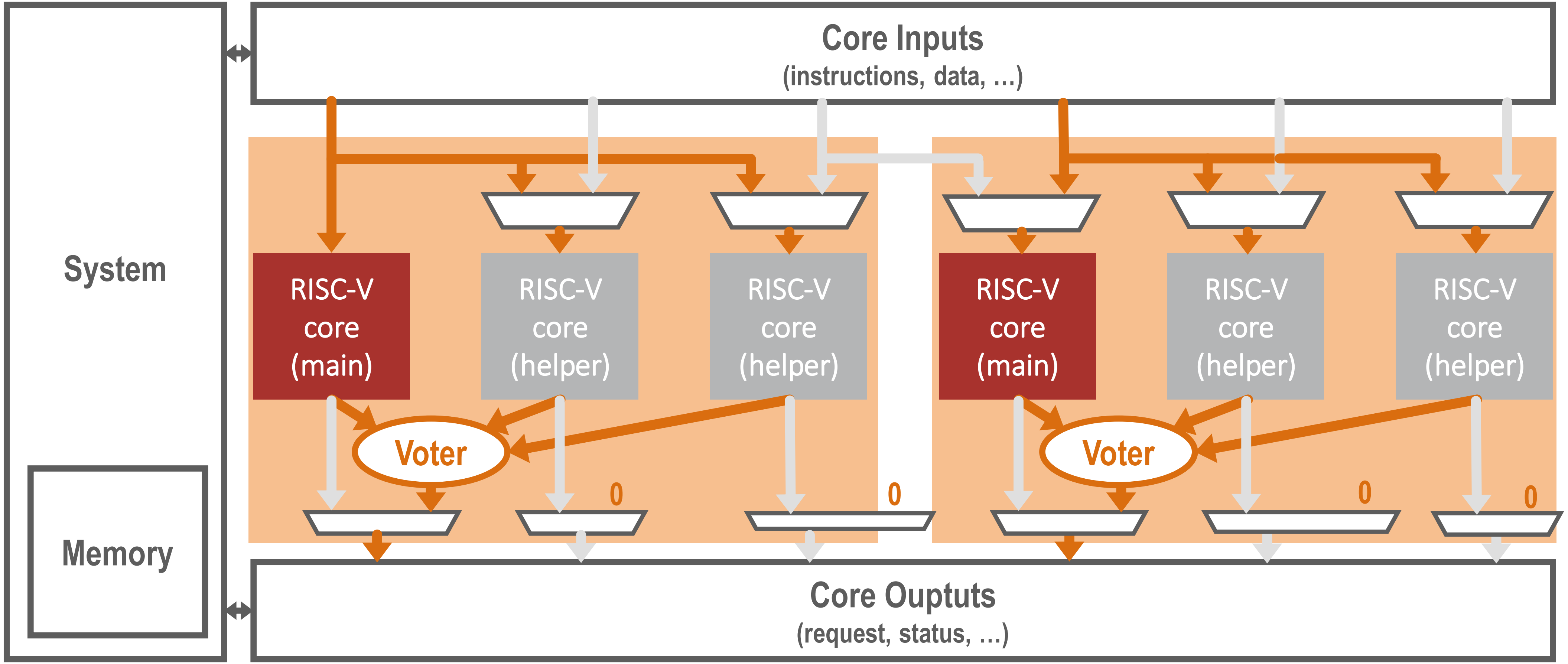
Hybrid Modular Redundancy: Reconfigurable

Independent Mode: high performance, no reliability



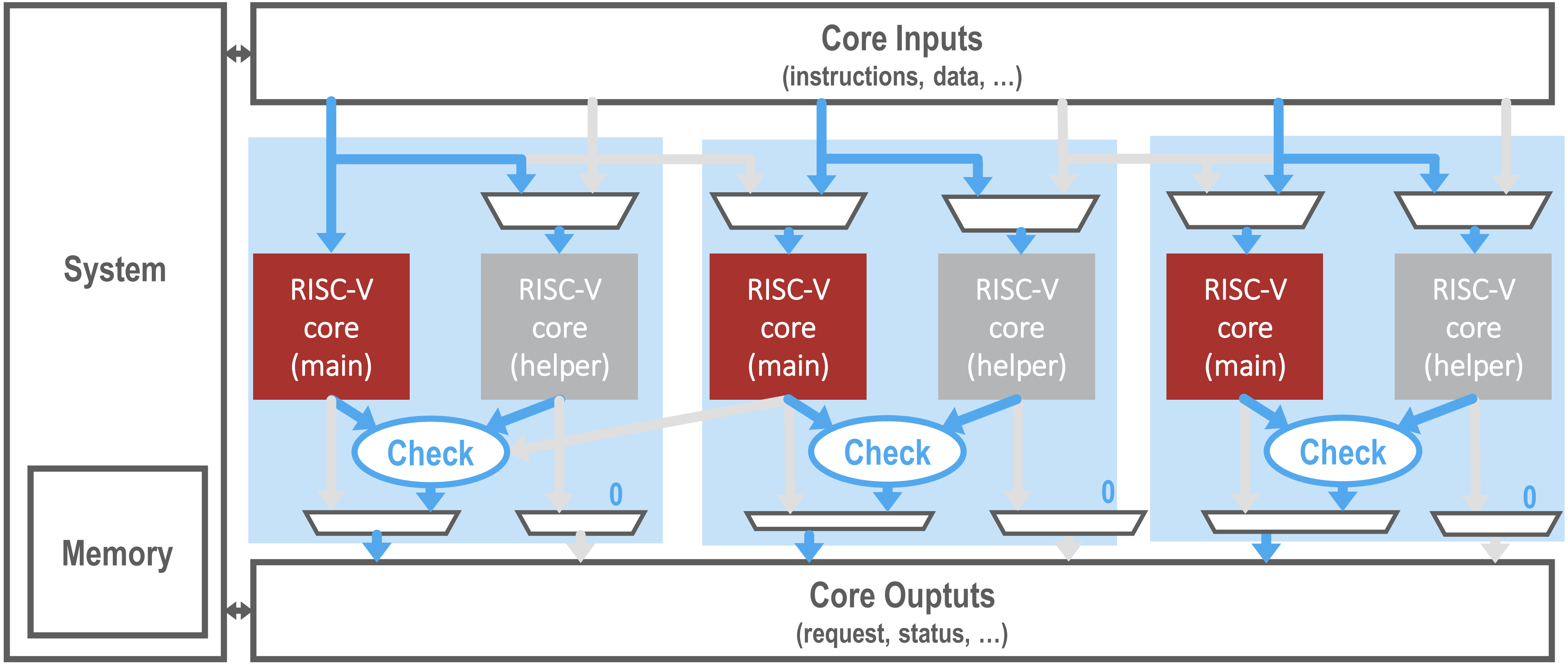
Hybrid Modular Redundancy: Reconfigurable

TMR Mode: low performance, high reliability, quick recovery



Hybrid Modular Redundancy: Reconfigurable

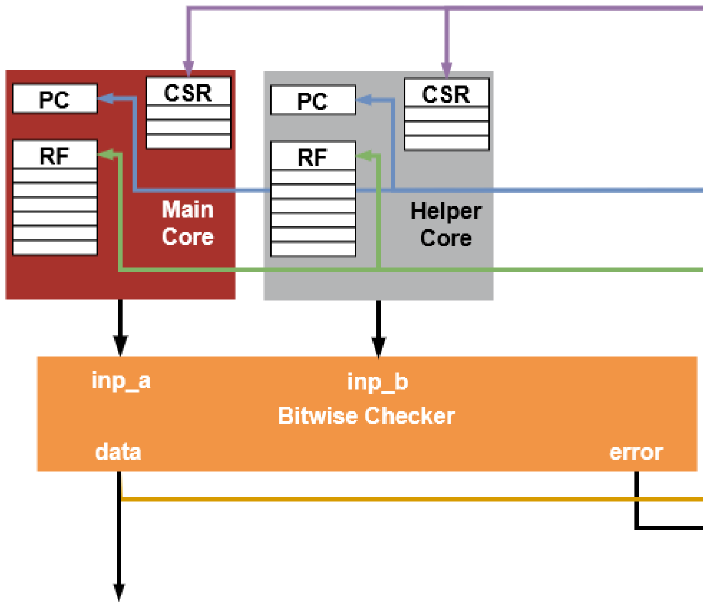
DMR Mode: good performance, good reliability, slow recovery



Rapid Recovery: shared hardware extension

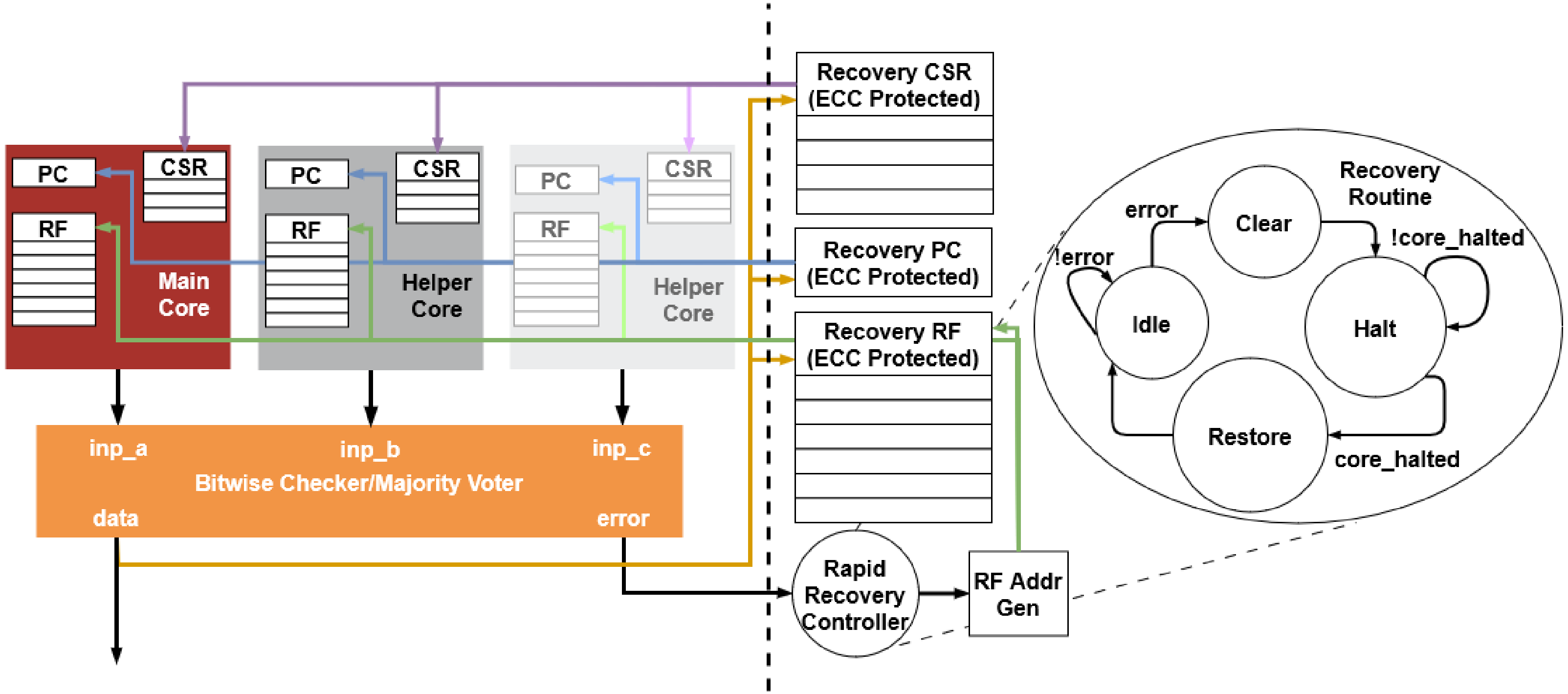
Rapid recovery mechanisms require **dedicated hardware** extensions

- The **state** of the core is defined by its internal registers (**CSRs, RF, PC**)
- Additional **backup copies** of the status registers, protected by **ECC**
- **Cycle-by-cycle backup** -> allow for recovery to the **most recent safe state**
- Quick recovery routine (**24 cycles**!)

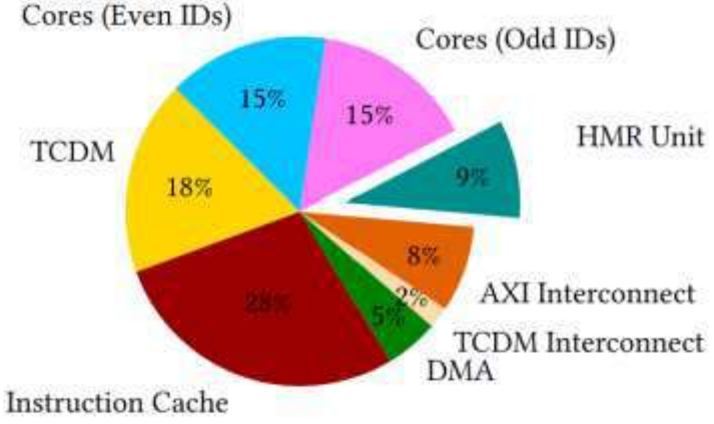


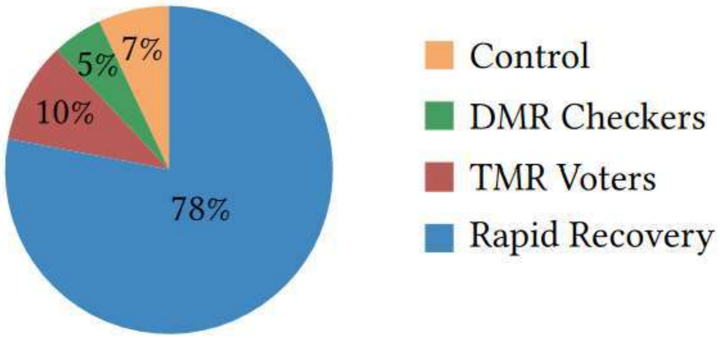
Rapid Recovery: shared hardware extension

- **Cycle-by-cycle backup** of the cores state in **ECC**-protected Status Registers
- Quick recovery procedure (**24 cycles**!)
- **Shared logic** between TMR and DMR modes



HMR, yes... but at what cost?





PULP Cluster Area [mm^2]		Overhead
Baseline	0.604	-
DMR	0.605	0.3%
TMR	0.608	0.7%
HMR	0.612	1.3%
With Rapid Recovery		
DMR	0.654	8.4%
TMR	0.657	8.8%
HMR	0.660	9.4%

	DMR	TMR	DMR Rapid Recovery	TMR Rapid Recovery
Recovery Latency [cycles]	Application dependant	363	24	24
Mode switching [cycles]	703	598	603	515

PULP

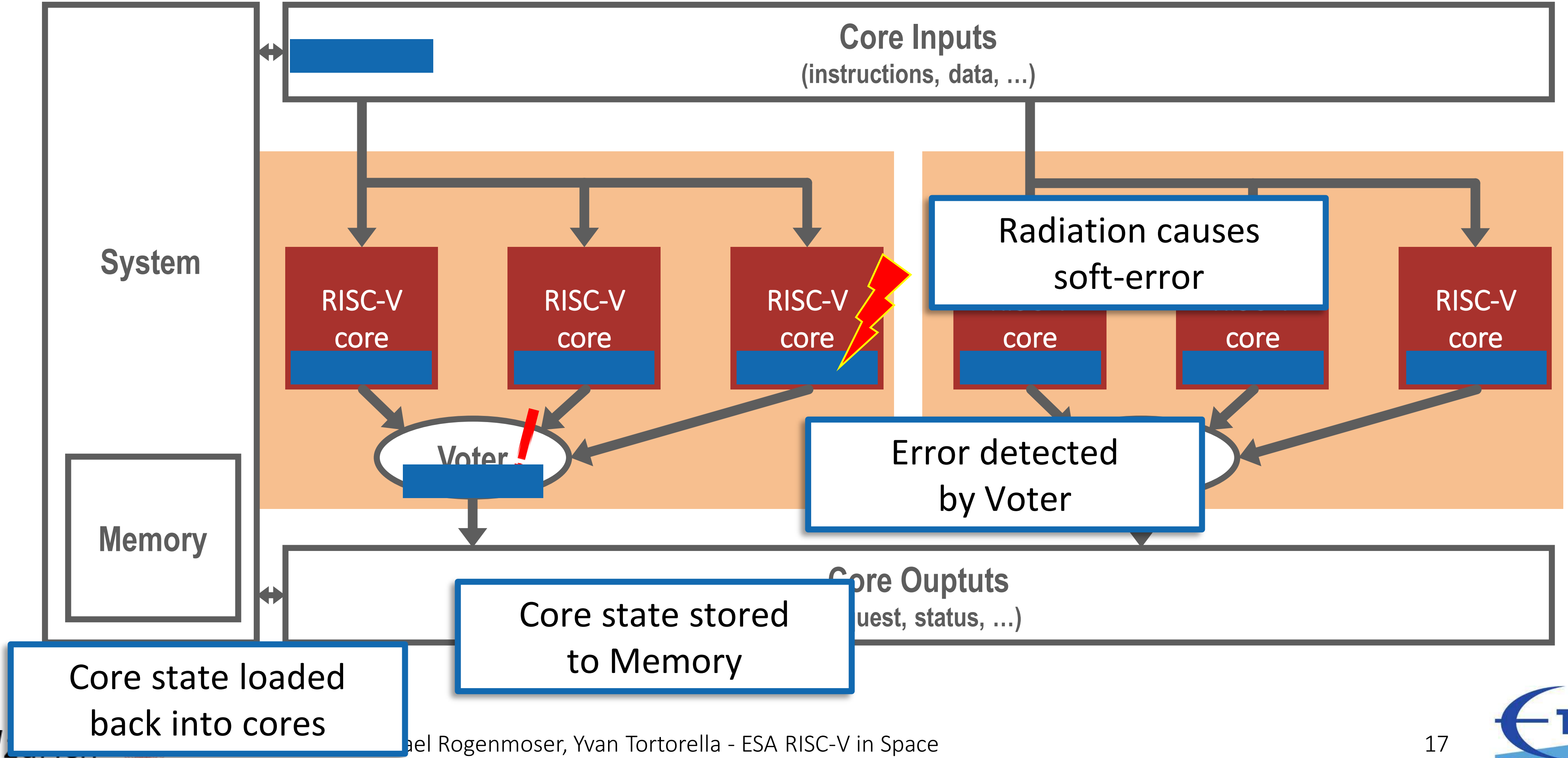
Parallel Ultra Low Power

Luca Benini, Ahmad Mirsalari, Alessandro Capotondi, Alessandro Nadalini, Alessandro Ottaviano, Alessio Burrello, Alfio Di Mauro, Andrea Borghesi, Andrea Cossettini, Angelo Garofalo, Arpan Prasad, Chi Zhang, Corrado Bonfanti, Cristian Cioflan, Cyril Koenig, Daniele Palossi, Davide Rossi, Fabio Montagna, Florian Glaser, Francesco Conti, Georg Rutishauser, Germain Haugou, Gianna Paulin, Giuseppe Tagliavini, Hanna Müller, Jannis Schoenleber, Lorenzo Lamberti, Luca Bertaccini, Luca Colagrande, Luca Valente, Maicol Caini, Manuel Eggimann, Manuele Rusci, Marco Bertuletti, Marco Guermandi, Matheus Cavalcante, Matteo Perotti, Mattia Sinigaglia, Michael Rogenmoser, Moritz Scherer, Moritz Schneider, Nazareno Bruschi, Nils Wistoff,, Paul Scheffler, Philipp Mayer, Robert Balas, Samuel Riedel, Segio Mazzola, Sergei Vostrikov, Simone Benatti, Thomas Benz, Thorir Ingolfsson, Tim Fischer, Victor Javier Kartsch Morinigo, Victor Jung, Viviane Potocnik, Vlad Niculescu, Xiaying Wang, Yichao Zhang, Yvan Tortorella, Frank K. Gürkaynak, all our past collaborators

and many more that we forgot to mention

http://pulp-platform.org @pulp_platform

Re-synchronization



On-Demand Redundancy Grouping