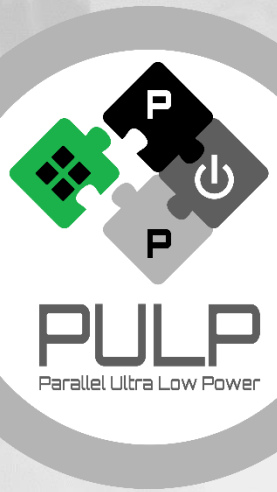
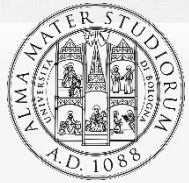


PULP: An Open-Source RISC-V ased Multi-Core Platform for In-Sensor Analytics

Workshop on Open Source Design Automation (OSDA) 2019 *29.03.2019*

Davide Rossi



[1]*Department of Electrical, Electronic and Information Engineering*

http://pulp-platform.org

ETH zürich

[2]*Integrated Systems Laboratory*

Parallel Ultra Low Power (PULP)

- Project started in **2013**
- A collaboration between University of Bologna and ETH Zürich
  - Large team. In total we are about 60 people, not all are working on PULP
- Key goal is

How to get the most BANG for the ENERGY consumed in a computing system

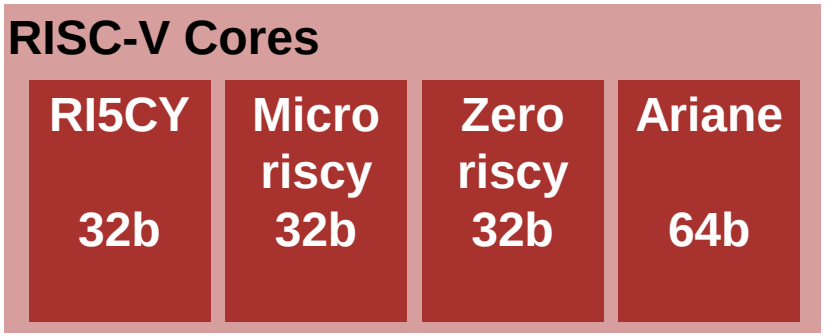
- We were able to start with a clean slate, no need to remain compatible to legacy systems.

ETH

How we started with open source processors

- Our research was not developing processors…
- … but we needed good processors for systems we build for research
- **Initially (2013) our options were**
  - Build our own (support for SW and tools)
  - Use a commercial processor (licensing, collaboration issues)
  - Use what is openly available (OpenRISC,..)
- **We started with OpenRISC**
  - First chips until mid-2016 were all using OpenRISC cores
  - We spent time improving the microarchitecture
- **Moved to RISC-V later**
  - Larger community, more momentum
  - Transition was relatively simple (new decoder)

We have developed several optimized RISC-V cores



Only processing cores are not enough, we need more

RISC-V Cores

- RI5CY 32b
- Micro riscy 32b
- Zero riscy 32b
- Ariane 64b

Peripherals

- JTAG
- SPI
- UART
- I2S
- DMA
- GPIO

Interconnect

- Logarithmic interconnect
- APB – Peripheral Bus
- AXI4 – Interconnect

Accelerators

- HWCE (convolution)
- Neurostream (ML)
- HWCrypt (crypto)
- PULPO (1st order opt)

All these components are combined into platforms

RISC-V Cores

- RI5CY 32b
- Micro riscy 32b
- Zero riscy 32b
- Ariane 64b

Peripherals

- JTAG
- SPI
- UART
- I2S
- DMA
- GPIO

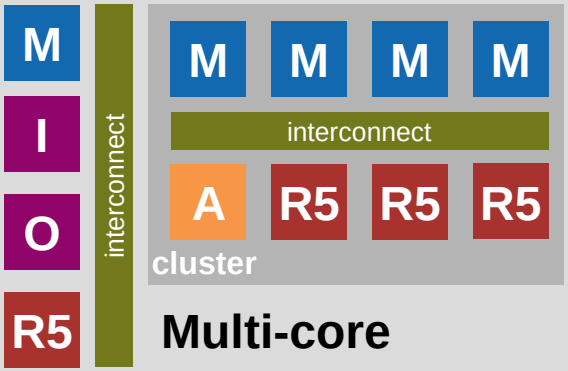
Interconnect

- Logarithmic interconnect
- APB – Peripheral Bus
- AXI4 – Interconnect

Platforms

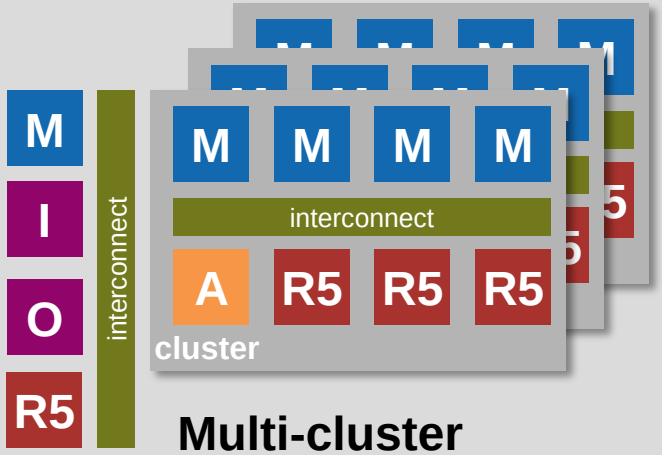
Single Core

- PULPino
- PULPissimo



Multi-core

- Fulmine
- Mr. Wolf



Multi-cluster

- Hero

IOT → HPC

Accelerators

- HWCE (convolution)
- Neurostream (ML)
- HWCrypt (crypto)
- PULPO (1st order opt)

RISC-V Instruction Set Architecture

- Started by UC-Berkeley in 2010
- RISC-V is an open standard governed by RISC-V foundation
  - Necessary for the continuity
  - Extensions are still being developed
- Defines 32, 64 and 128 bit ISA
  - No implementation, just the ISA
  - Different RISC-V implementations (both open and close source) are available
- The PULP project specializes in **efficient implementations of RISC-V cores and peripherals**

Spec separated into "extensions"

I	Integer instructions
E	Reduced number of registers
M	Multiplication and Division
A	Atomic instructions
F	Single-Precision Floating-Point
D	Double-Precision Floating-Point
C	Compressed Instructions
X	Non Standard Extensions

Our RISC-V family explained

32 bit

Low Cost Core

- **Zero-riscy**
  - RV32-ICM
- **Micro-riscy**
  - RV32-CE

ARM Cortex-M0+

Core with DSP enhancements

- **RI5CY**
  - RV32-ICMX
    - SIMD
    - HW loops
    - Bit manipulation
    - Fixed point

ARM Cortex-M4

Floating-point capable Core

- **RI5CY+FPU**
  - RV32-ICMFX

ARM Cortex-M4F

64 bit

Linux capable Core

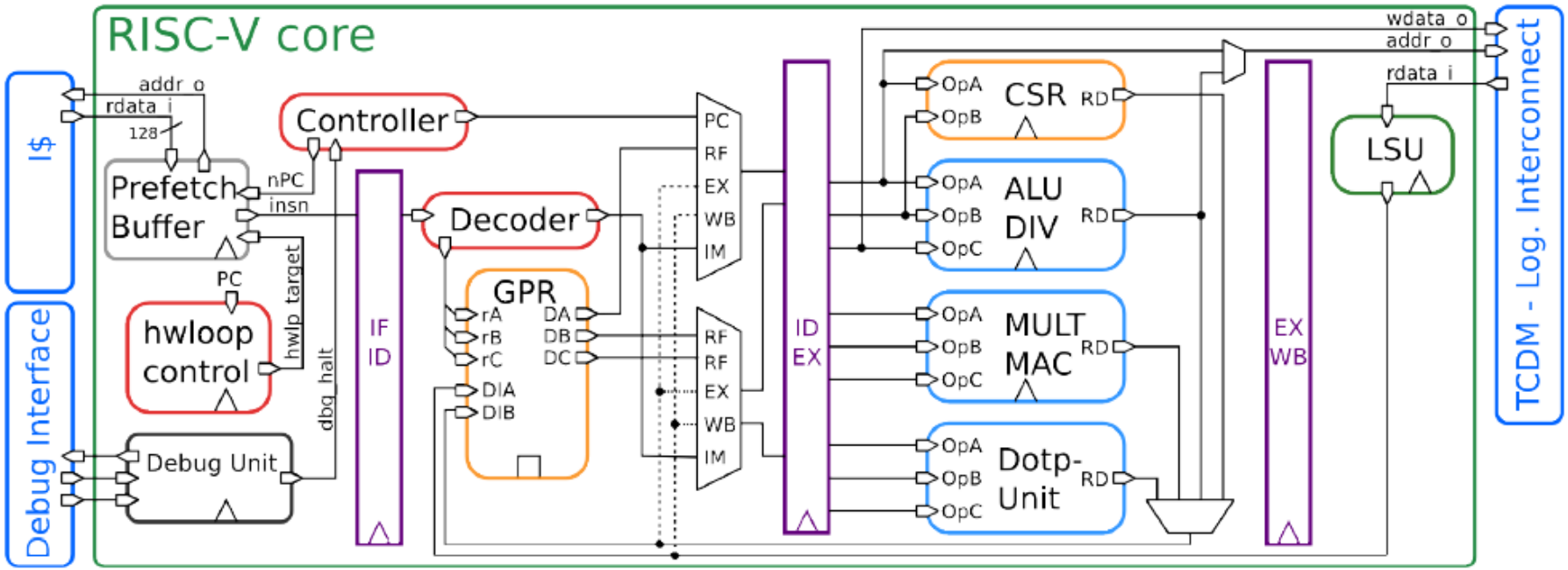
- **Ariane**
  - RV64-IMAFDCX
  - Full privilege specification

ARM Cortex-A55

PULP

ETH

RI5CY – Our workhorse 32-bit core



- 4-stage pipeline, optimized for energy efficiency
- 40 kGE, 30 logic levels, Coremark/MHZ 3.19
- Includes various extensions (Xpulp) to RISC-V for DSP applications

PULP

ETH

Our extensions to RI5CY (with additions to GCC)

- **Post–incrementing** load/store instructions
- **Hardware Loops** (`lp.start`, `lp.end`, `lp.count`)
- **ALU instructions**
  - Bit manipulation (count, set, clear, leading bit detection)
  - Fused operations: (add/sub-shift)
  - Immediate branch instructions
- **Multiply Accumulate** (32x32 bit and 16x16 bit)
- **SIMD instructions** (2x16 bit or 4x8 bit) with scalar replication option
  - add, min/max, dotproduct, shuffle, pack (copy), vector comparison

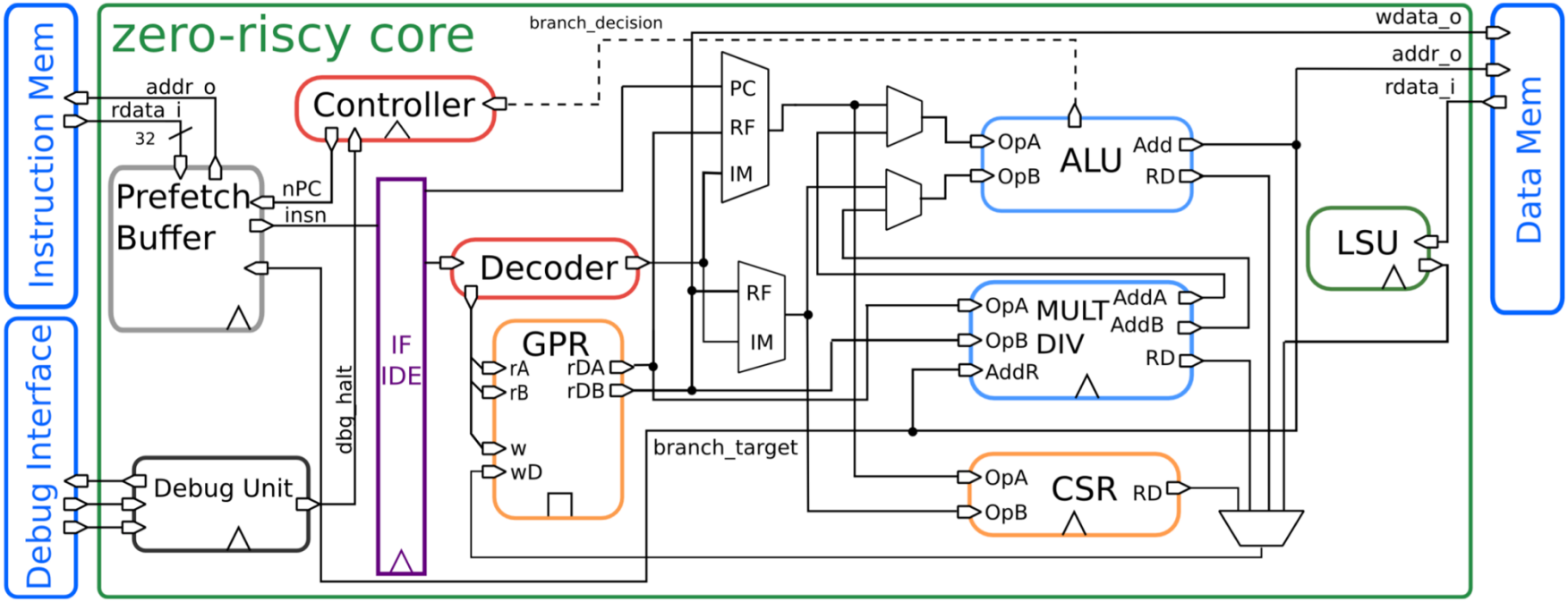
For 8-bit values the following can be executed in a single cycle (`pv.dotup.b`)

$$Z = D_1 \times K_1 + D_2 \times K_2 + D_3 \times K_3 + D_4 \times K_4$$

Why we designed other 32-bit cores after RI5CY?

- **RI5CY was built for energy efficiency for DSP applications**
  - Ideally all parts of the core are running all the time doing something useful
  - This does not always mean it is low-power
  - The core is rather large (> 40 kGE without FPU)
- **People asked us about a simple and small core**
  - Not all processor cores are used for DSP applications
  - The DSP extensions are mostly idle for control applications
  - **Zero-Riscy** was designed to as a **simple and efficient** core.
- **Some people wanted the smallest possible RISC-V core**
  - It is possible to further reduce area by using 16 registers instead of 32 (E)
  - Also the multiplier can be removed saving a bit more
  - **Micro-Riscy** is a parametrized variation of Zero-Riscy with **minimal area**

Zero/Micro-riscy, small area core for control applications

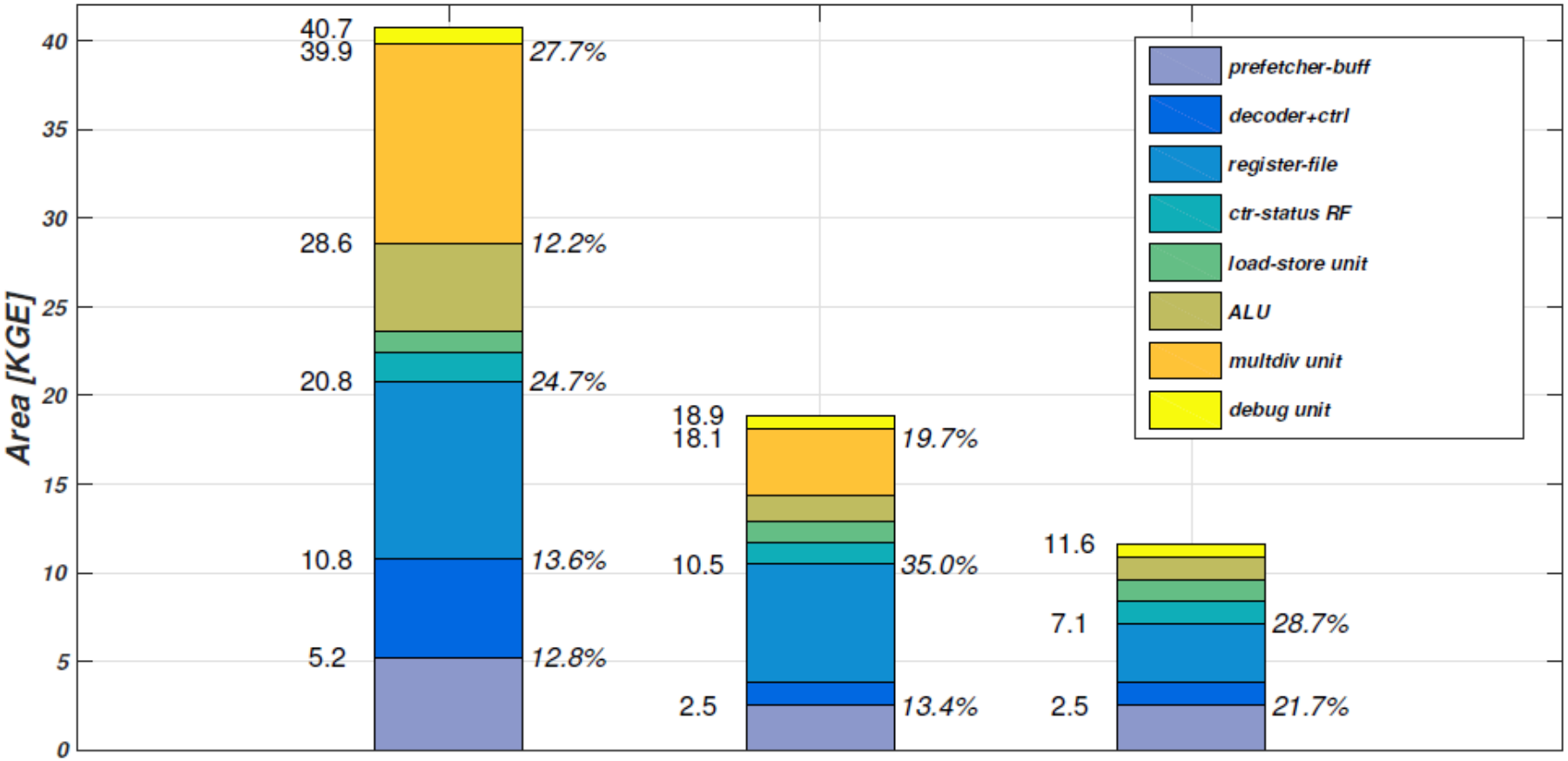


- Only 2-stage pipeline, simplified register file
- **Zero-Riscy** (RV32-ICM), 19kGE, 2.44 Coremark/MHz
- **Micro-Riscy** (RV32-EC), 12kGE, 0.91 Coremark/MHz
- Used as SoC level controller in newer PULP systems

PULP

ETH

Different 32-bit cores with different area requirements



RI5CY **Zero-riscy** **Micro-riscy**

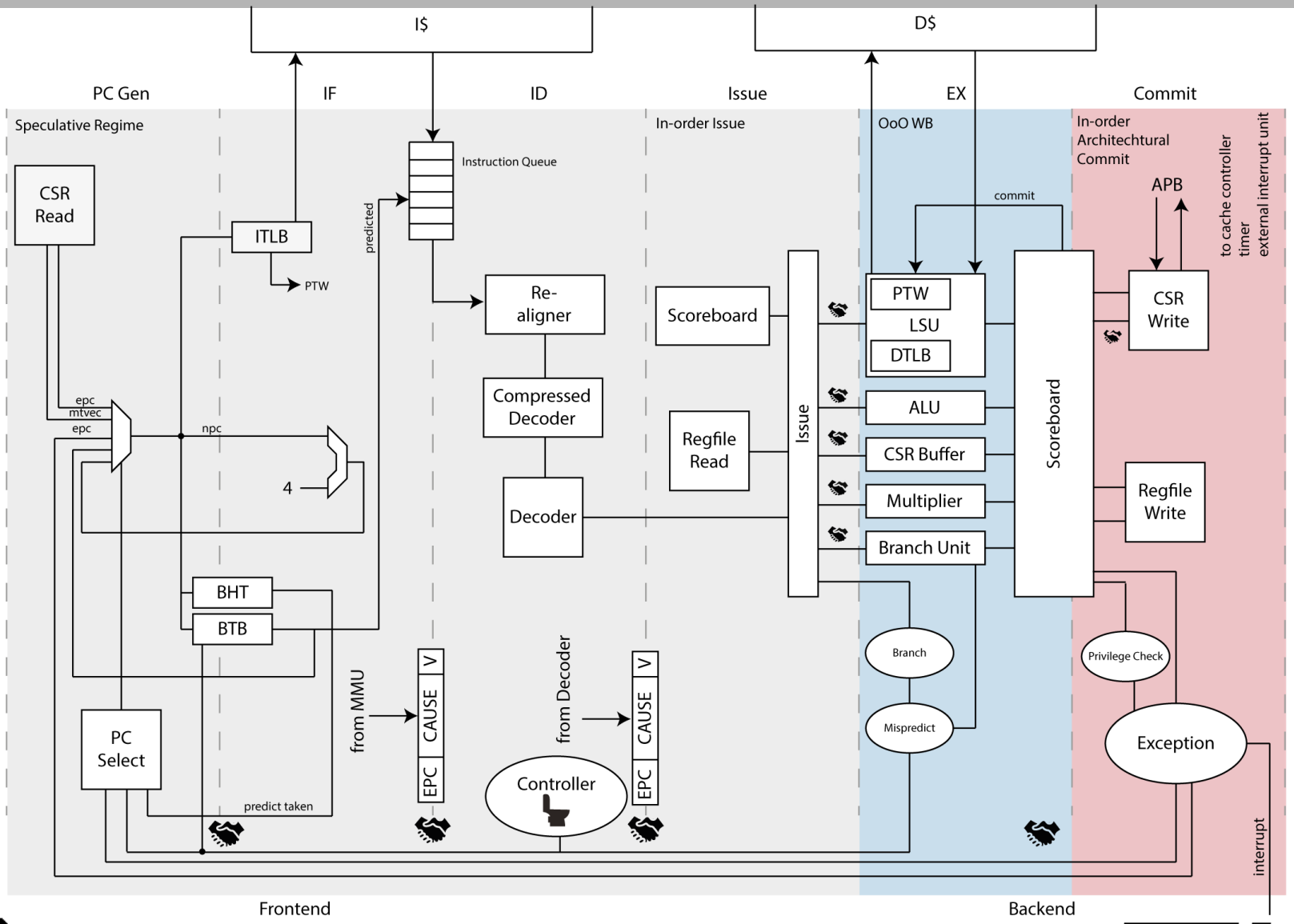
PULP

ETH

Finally the step into 64-bit cores

- **For the first 4 years of the PULP project we used only 32bit cores**
  - Most IoT applications work well with 32bit cores.
  - A typical 64bit core is much more than 2x the size of a 32bit core.
- **But times change:**
  - Using a 64bit Linux capable core allows you to share the same address space as main stream processors.
    - We are involved in several projects where we (are planning to) use this capability
  - There is a lot of interest in the security community for working on a contemporary open source 64bit core.
  - Open research questions on how to build systems with multiple cores.

ARIANE: Our Linux Capable 64-bit core

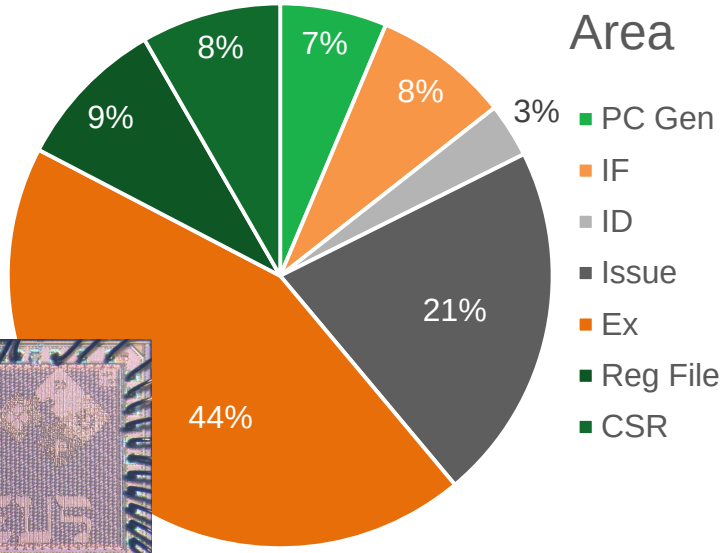


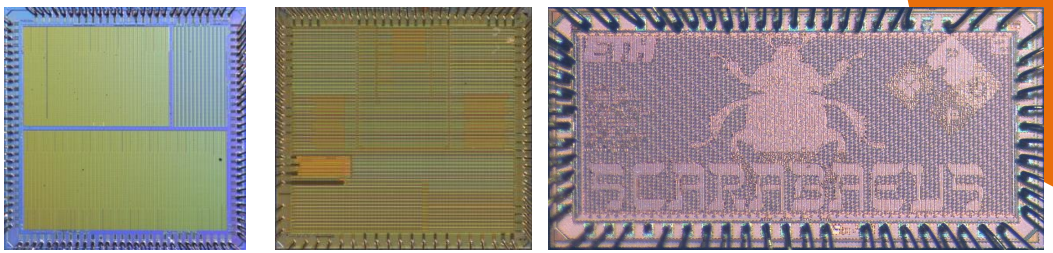
PULP

ETH

Main properties of Ariane

- Tuned for high frequency, 6 stage pipeline, integrated cache
  - In order issue, out-of-order write-back, in-order-commit
  - Supports privilege spec 1.11, M, S and U modes
  - Hardware Page Table Walker
- Implemented in GF22nm (Poseidon, Kosmodrom), and UMC65 (Scarabaeus)
  - In 22nm: ~1 GHz worst case conditions (SSG, 125/-40C, 0.72V)
  - 8-way 32kByte Data cache and 4-way 32kByte Instruction Cache
  - Core area: 175 kGE

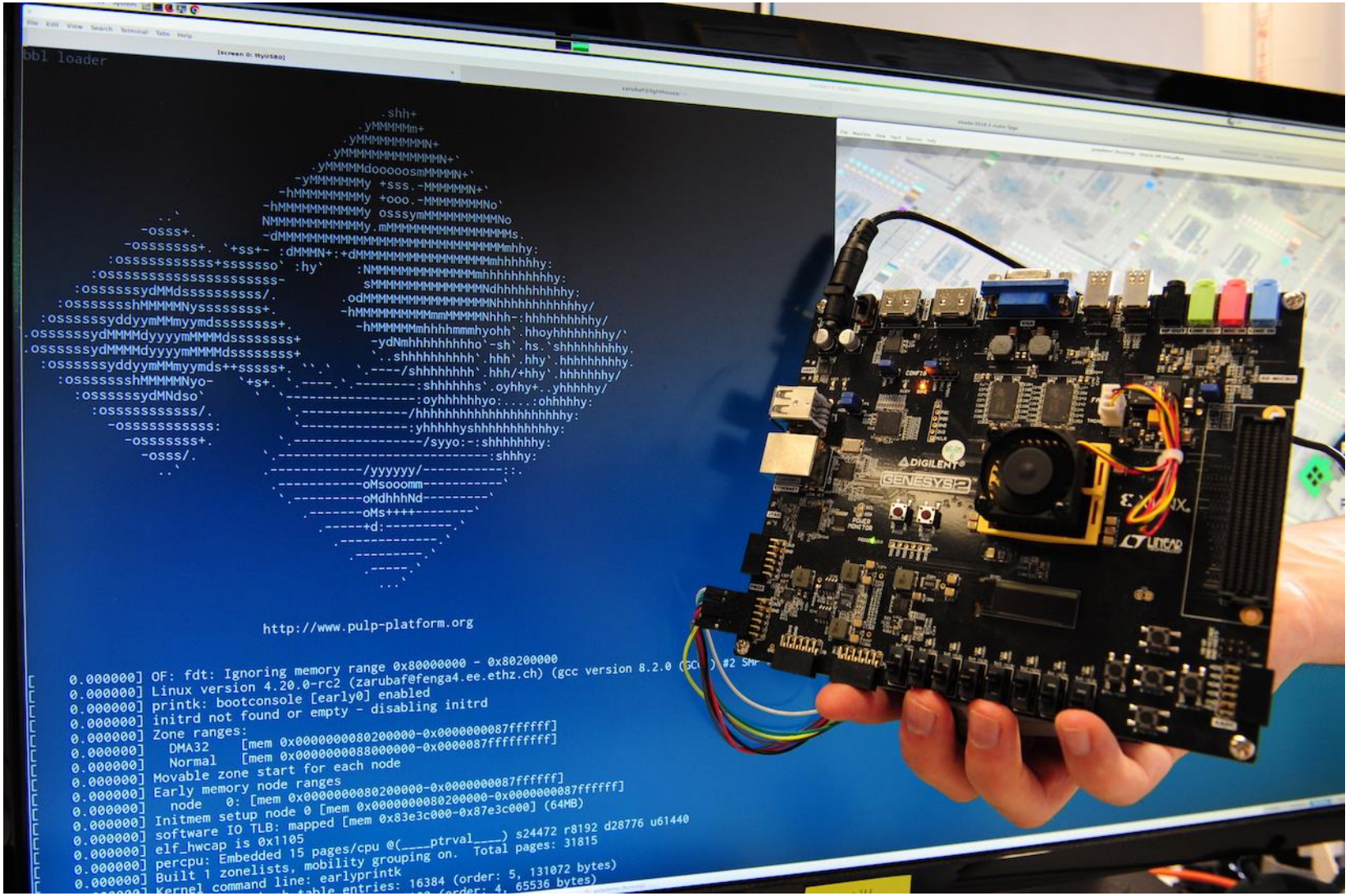




PULP

ETH

Ariane booting Linux on a Digilent Genesys 2 board

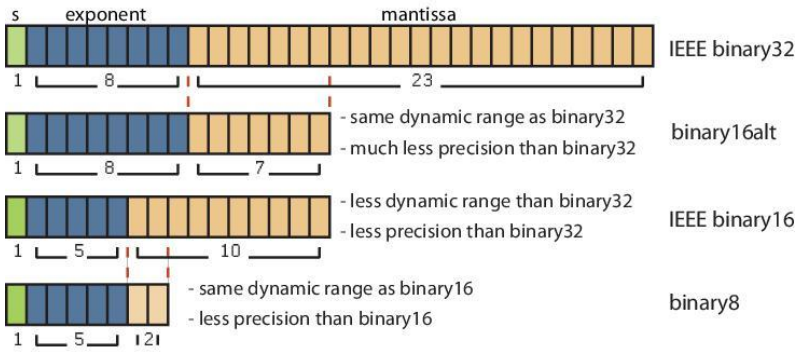


PULP

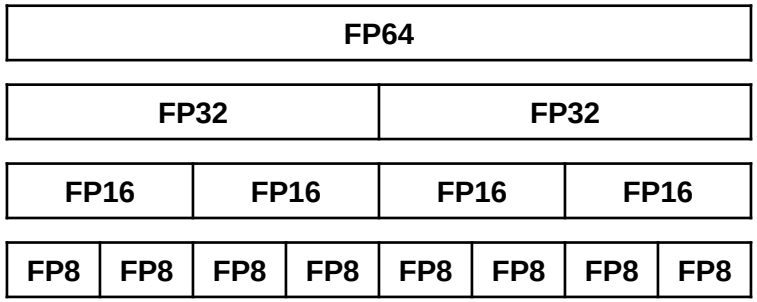
ETH

What About Floating Point Support?

- **F** (single precision) and
  D (double precision) extension in RISC-V
- Uses separate floating point register file
  - specialized float loads (also compressed)
  - float moves from/to integer register file
- Fully IEEE compliant
- **RI5CY** support for **F**
- **Ariane** for **F** and **D**
- **Alternative FP Format** support (<32 bit)



Packed-SIMD support for all formats

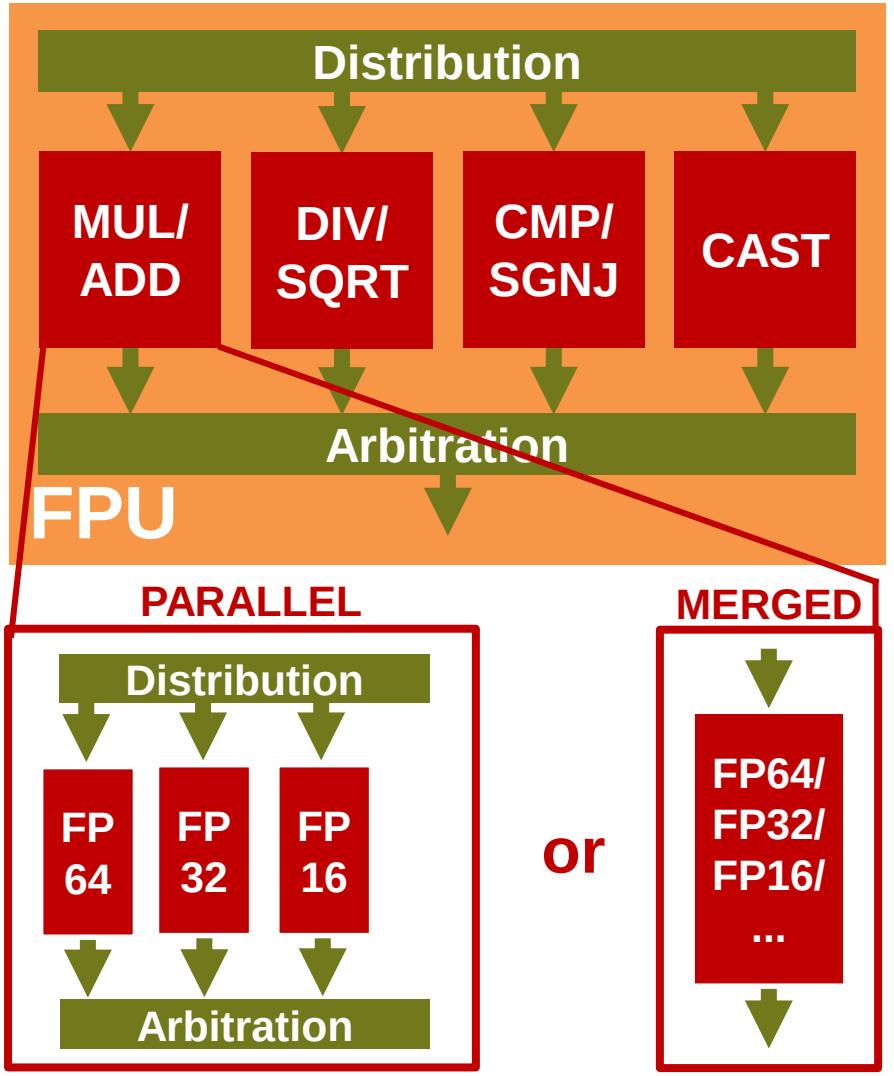


Unified FP/Integer register file

- Not standard
- up to **15 %** better performance
  - Re-use integer load/stores (post incrementing ld/st)
  - Less area overhead
  - Useful if pressure on register file is not very high (true for a lot of applications)

Parametric Floating-Point Unit for Transprecision

- **Main FP operation groups**
  - **MUL/ADD**: Add/Subtract, Multiply, FMA
  - **CMP/SGNJ**: Comparisons, Min/Max etc.
  - **CAST**: FP-FP casts, Int-FP / FP-Int casts
- **Parametrizable**
  - Number & Encoding of **Formats**
  - Packed-SIMD **Vectors**
  - **# Pipeline Stages** (per Op and Format)
  - **Implementation** (per Op and Format)
    - **PARALLEL** for best Speed
    - **MERGED** (or Iterative) for best Area
- **Special Functions** for Transprecision
  - Cast-and-Pack 2 FP Values to Vector
  - Casts amongst FP Vectors + Repacking
  - Expanding FMA (e.g. FP32 += FP16*FP16)



PULP

ETH

Memory Protection (PMU and MMU)

Physical Memory Protection (PMP)

- Protect the physical memory when the core runs in U or S privilege level
- Up to 16 entries for address filtering
- Configuration held in 4 CSRs
  pmpcfg[0-3]
- Whether Store (W), Load (R) and Fetch (X) is allowed
- Address matching modes:
  - Naturally aligned power-of-2 regions (NAPOT) or aligned 4 Byte (NA4)
  - Boundaries >, < (TOR)
- Implemented in **RI5CY**

Supervisor Memory Translation and Protection (for Linux-like systems)

- Effectively needs TLBs
- Register to configure base page number (**satp**)
- Translation Mode
  (32, 39, 48 virtual addressing)
- Address Space Identifier (**ASID**)
- Implemented in **Ariane**

PULP

ETH

The pulp-platforms put everything together

RISC-V Cores

- RI5CY 32b
- Micro riscy 32b
- Zero riscy 32b
- Ariane 64b

Peripherals

- JTAG
- SPI
- UART
- I2S
- DMA
- GPIO

Interconnect

- Logarithmic interconnect
- APB – Peripheral Bus
- AXI4 – Interconnect

Platforms

Single Core

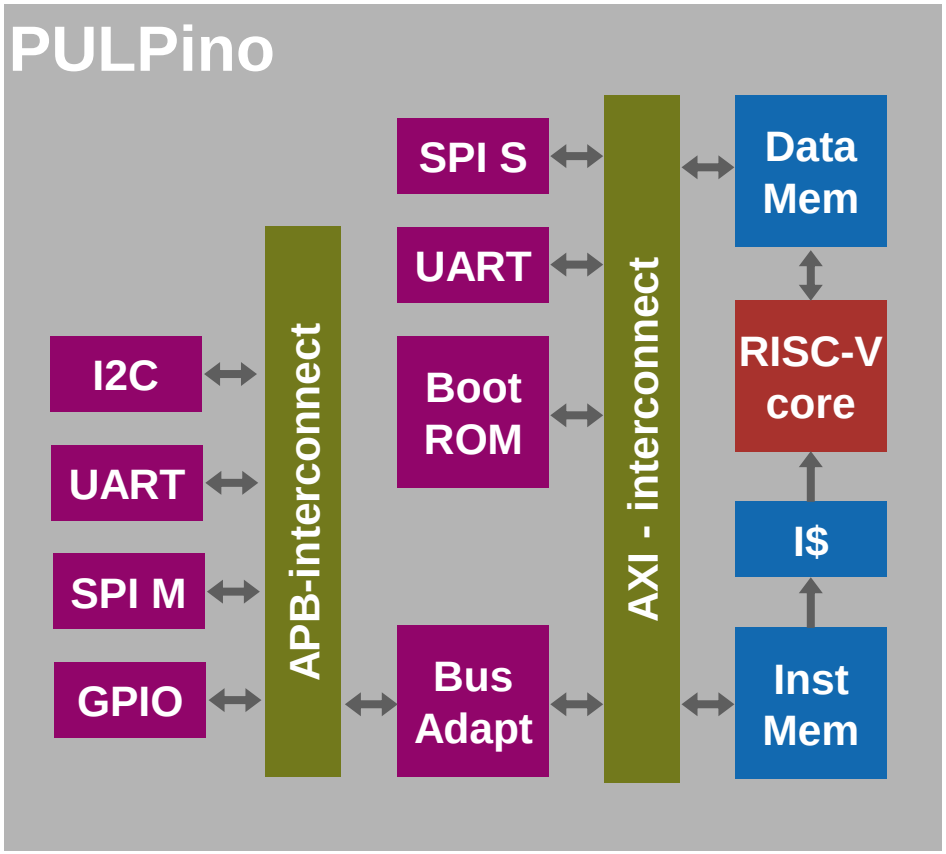
- PULPino
- PULPissimo

Accelerators

- HWCE (convolution)
- Neurostream (ML)
- HWCrypt (crypto)
- PULPO (1st order opt)

PULPino our first single core platform

- **Simple design**
  - Meant as a quick release
- **Separate Data and Instruction memory**
  - Makes it easy in HW
  - Not meant as a Harvard arch.
- **Can be configured to work with all our 32bit cores**
  - RI5CY, Zero/Micro-Riscy
- **Peripherals copied from its larger brothers**
  - Any AXI and APB peripherals could be used

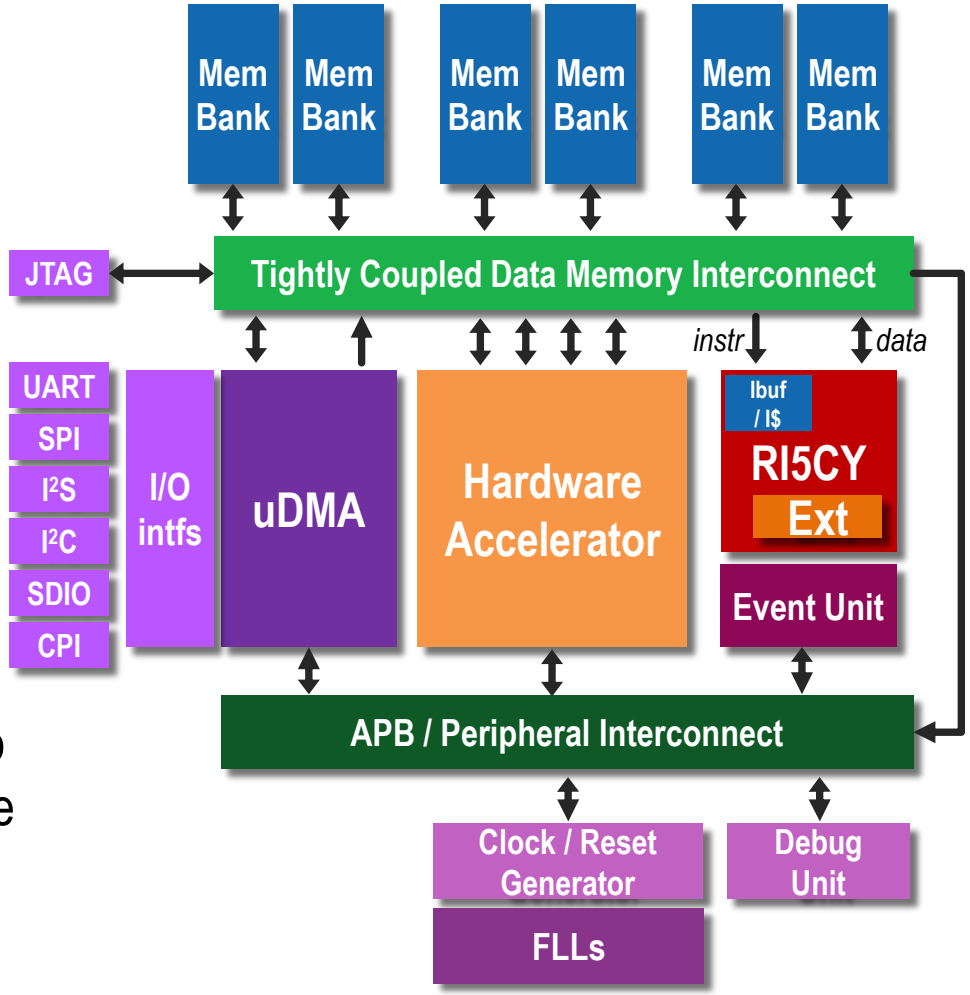


PULP

ETH

PULPissimo the improved single core platform

- **Shared memory**
  - Unified Data/Instruction Memory
  - Uses the multi-core infrastructure
- **Support for Accelerators**
  - Direct shared memory access
  - Programmed through APB bus
  - Number of TCDM access ports determines max. throughput
- **uDMA for I/O subsystem**
  - Can copy data directly from I/O to memory without involving the core
- **Used as a fabric controller in larger PULP systems**

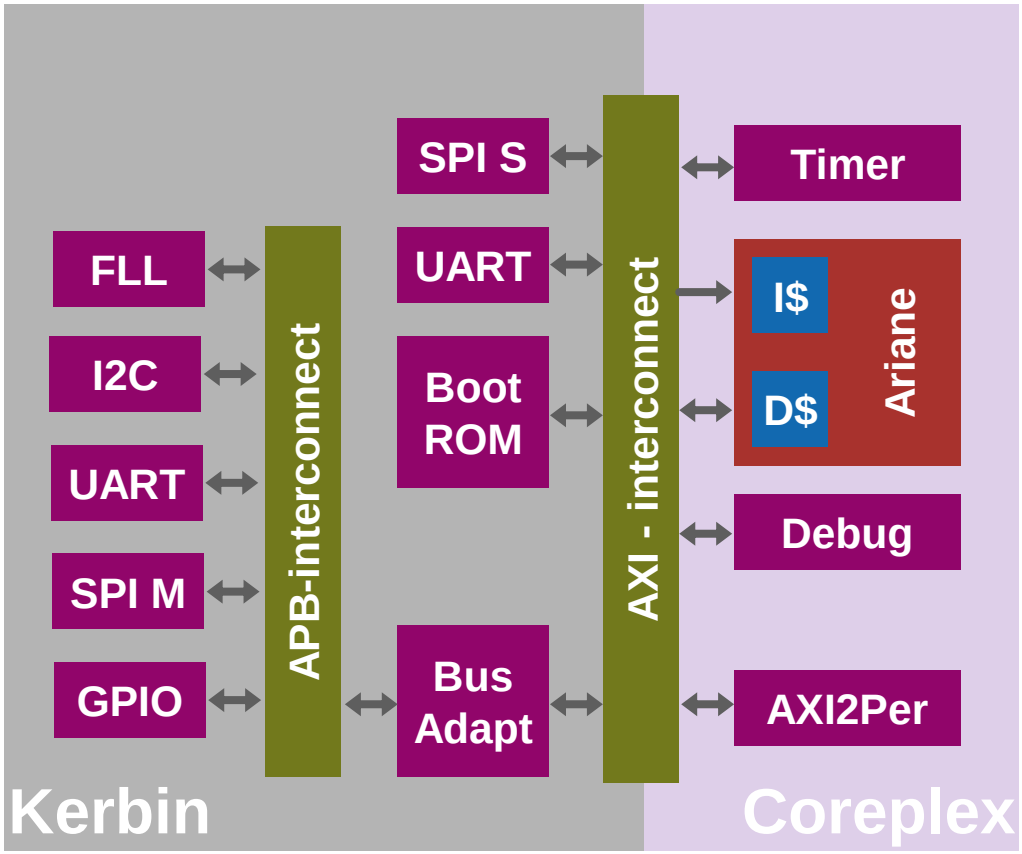


PULP

ETH

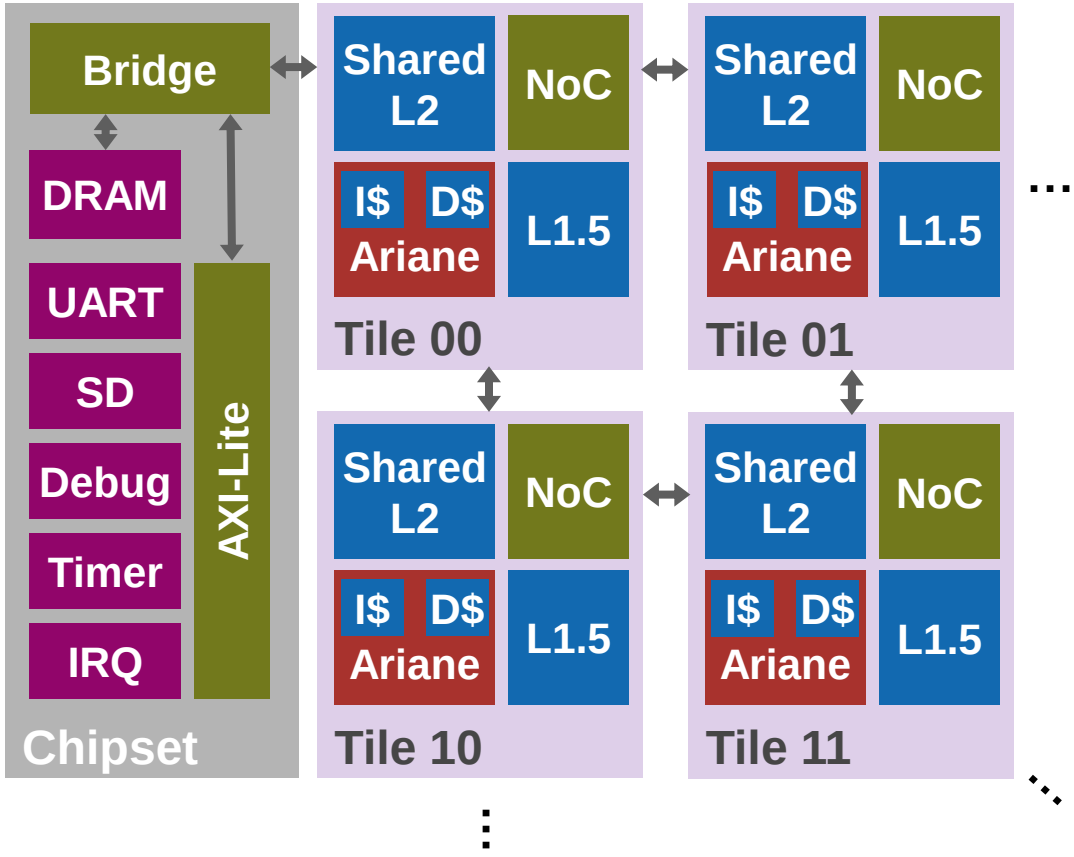
Kerbin the single core support structure for Ariane

- **Still work in progress**
  - Current version is very simple
  - Useful for in-house testing
  - A more advanced version will likely be developed soon



OpenPiton and Ariane together, the many-core system

- **OpenPiton**
  - Developed by Princeton
  - Originally OpenSPARC T1
  - Scalable NoC with coherent LLC
  - Tiled Architecture
- **Still work in progress**
  - Bare-metal released in Dec ’18
  - Update with support for SMP Linux will be released soon



PULP OpenPiton ETH

The main PULP systems we develop are cluster based

RISC-V Cores

- RI5CY 32b
- Micro riscy 32b
- Zero riscy 32b
- Ariane 64b

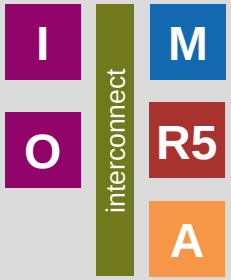
Peripherals

- JTAG
- SPI
- UART
- I2S
- DMA
- GPIO

Interconnect

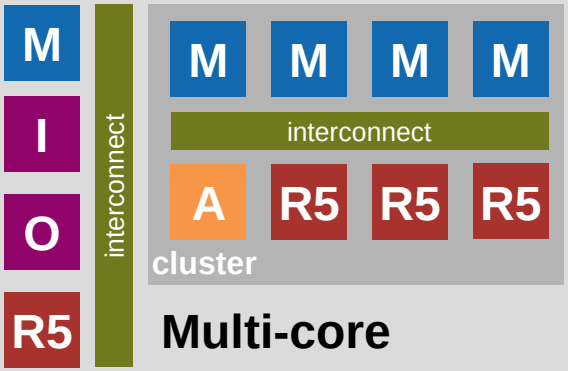
- Logarithmic interconnect
- APB – Peripheral Bus
- AXI4 – Interconnect

Platforms



Single Core

- PULPino
- PULPissimo



Multi-core

- Fulmine
- Mr. Wolf

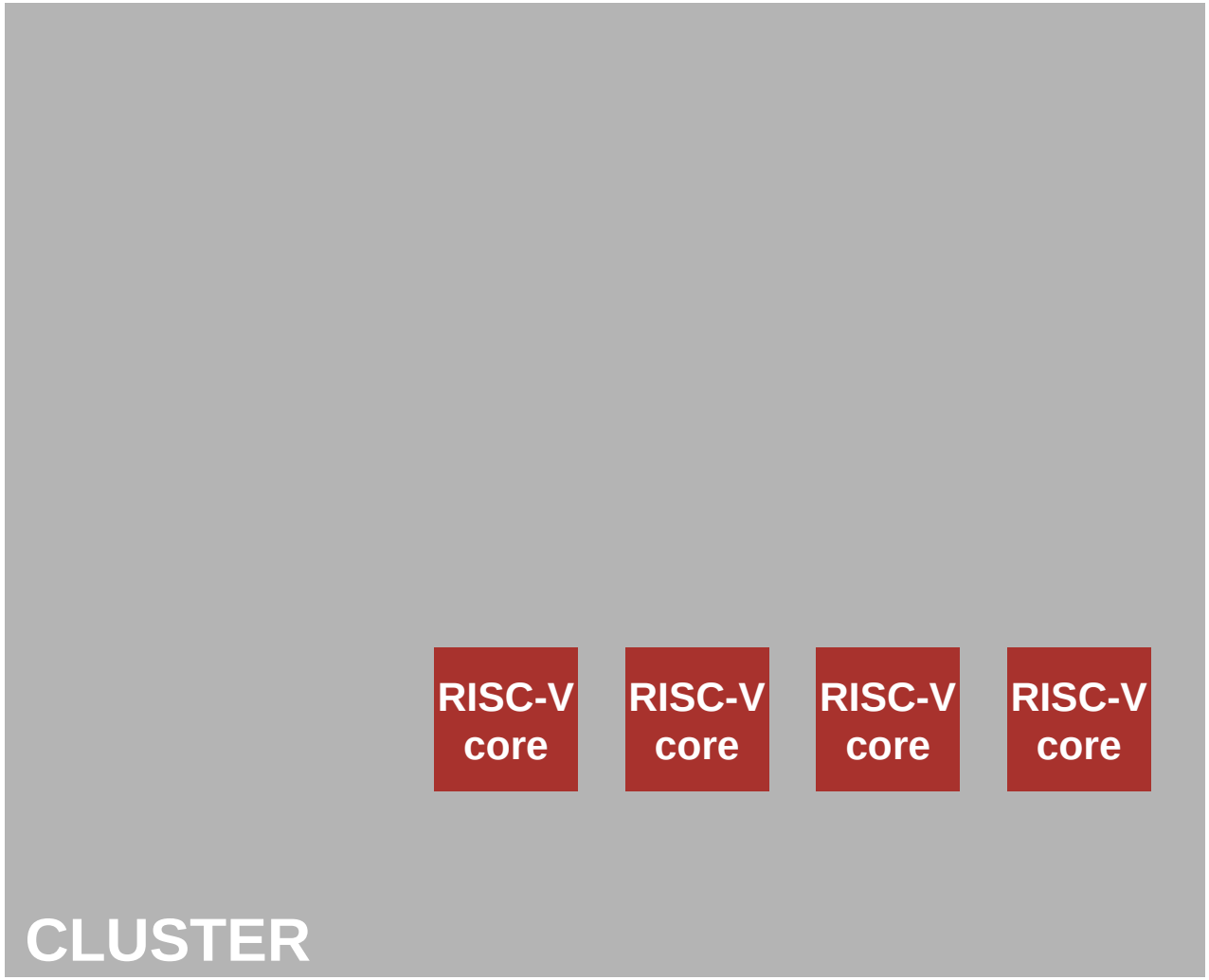
Accelerators

- HWCE (convolution)
- Neurostream (ML)
- HWCrypt (crypto)
- PULPO (1st order opt)

The main components of a PULP cluster

- **Multiple RISC-V cores**
  - Individual cores can be started/stopped with little overhead
  - DSP extensions in cores
- **Multi-banked scratchpad memory (TCDM)**
  - **Not a cache**, there is no L1 data cache in our systems
- **Logarithmic Interconnect allowing all cores to access all banks**
  - Cores will be stalled during contention, includes arbitration
- **DMA engine to copy data to and from TCDM**
  - Data in TCDM managed by software
  - Multiple channels, allows pipelined operation
- **Hardware accelerators with direct access to TCDM**
  - No data copies necessary between cores and accelerators.

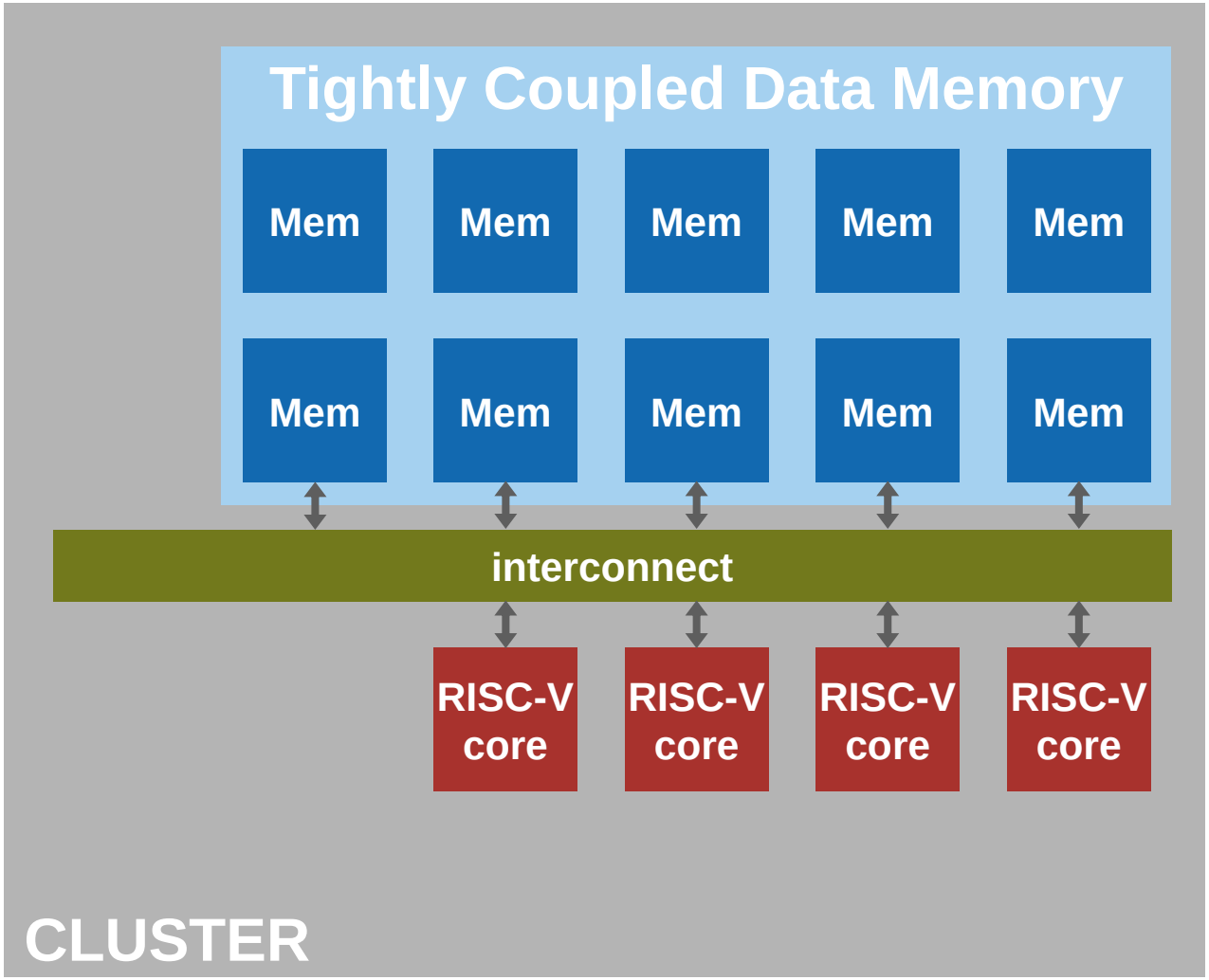
PULP cluster contains multiple RISC-V cores



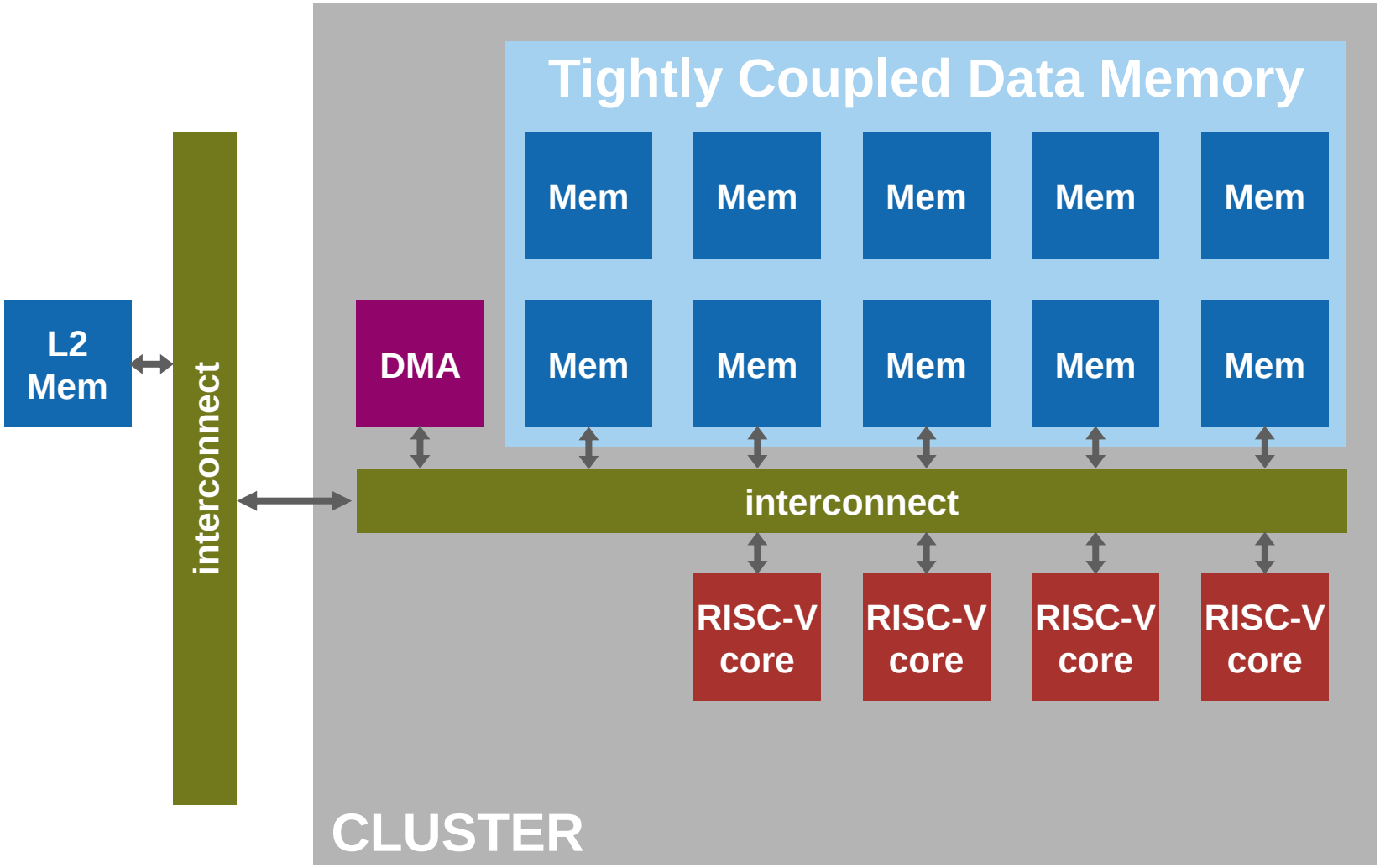
PULP

ETH

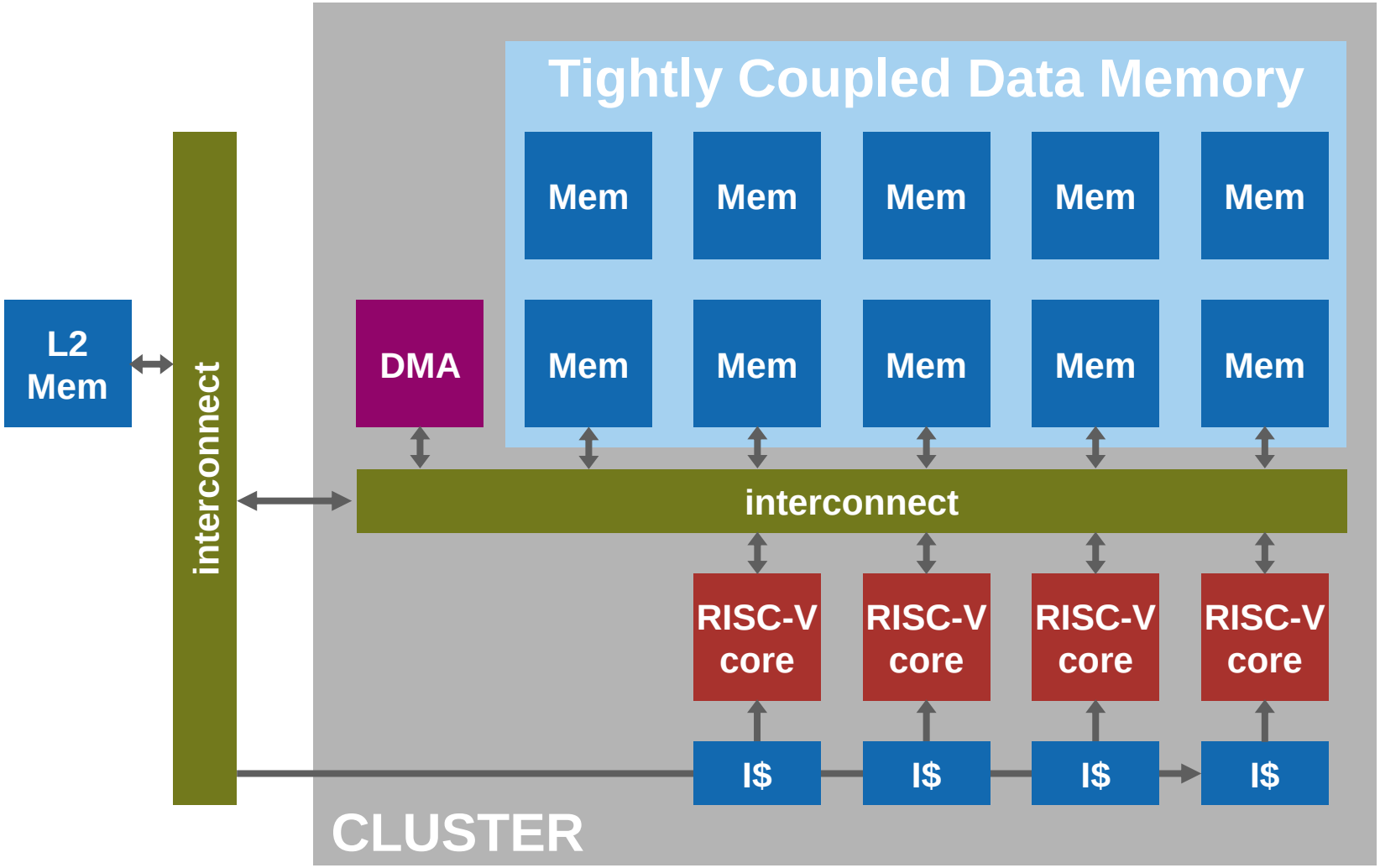
All cores can access all memory banks in the cluster



Data is copied from a higher level through DMA



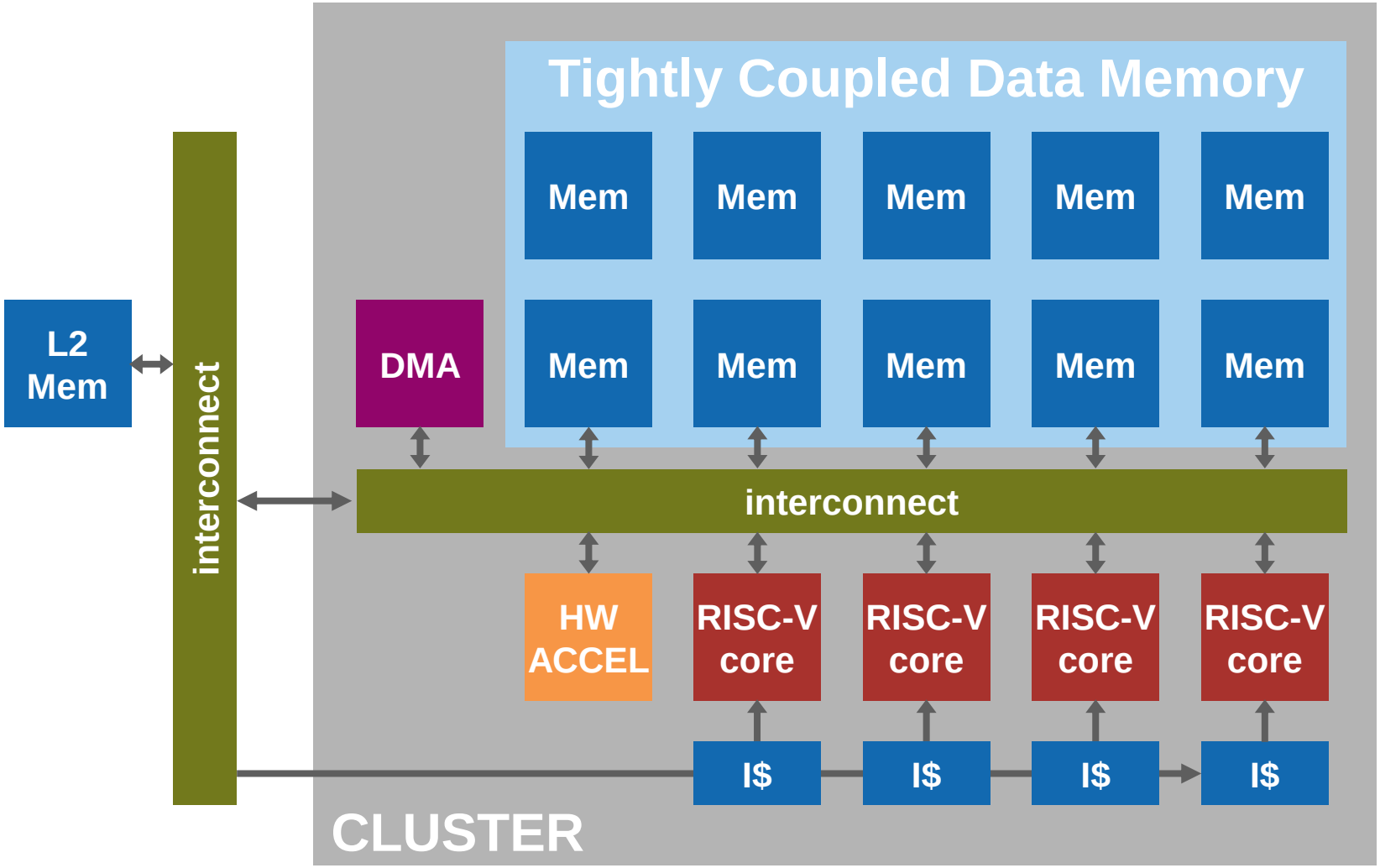
There is a (shared) instruction cache that fetches from L2



PULP

ETH

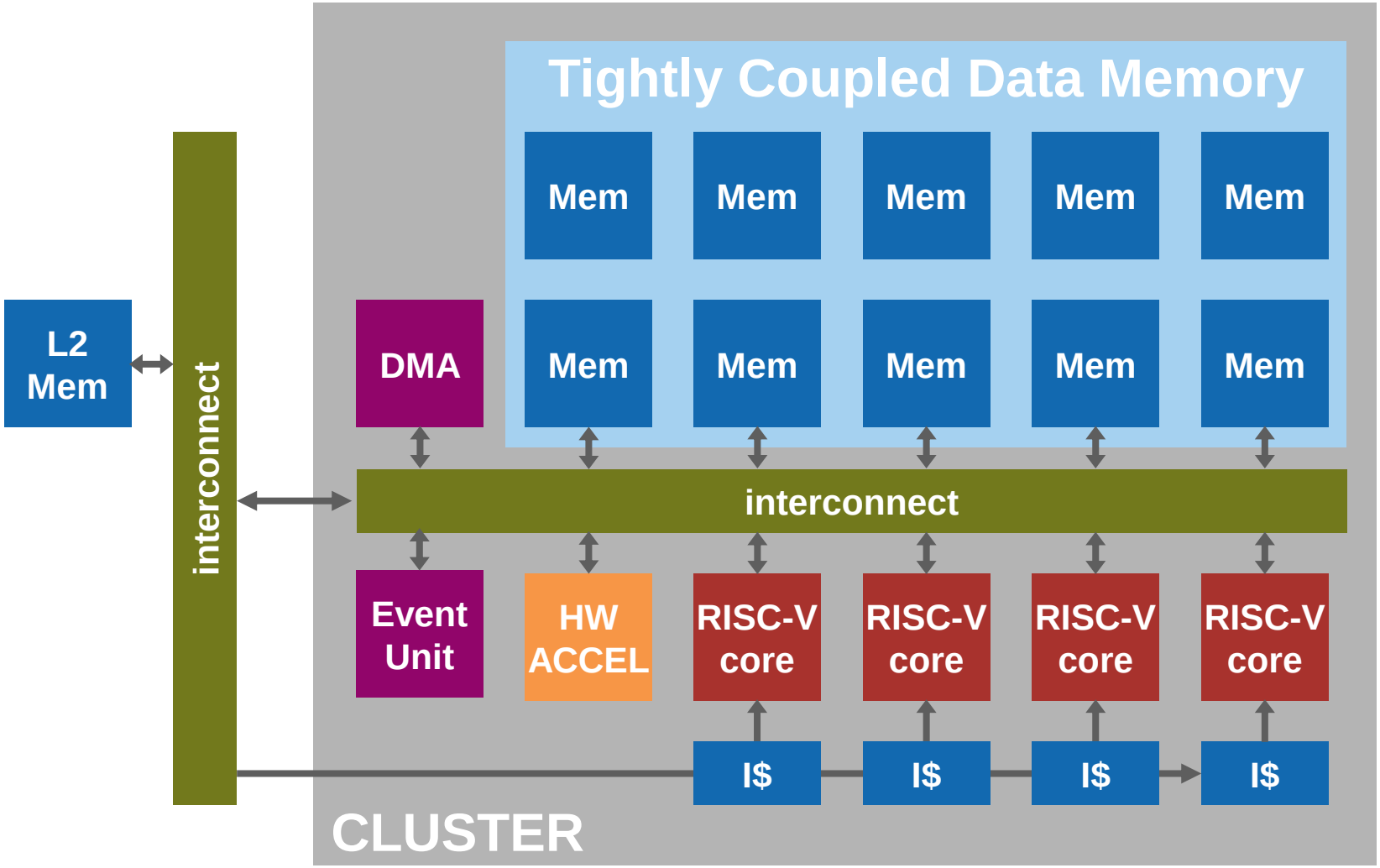
Hardware Accelerators can be added to the cluster



PULP

ETH

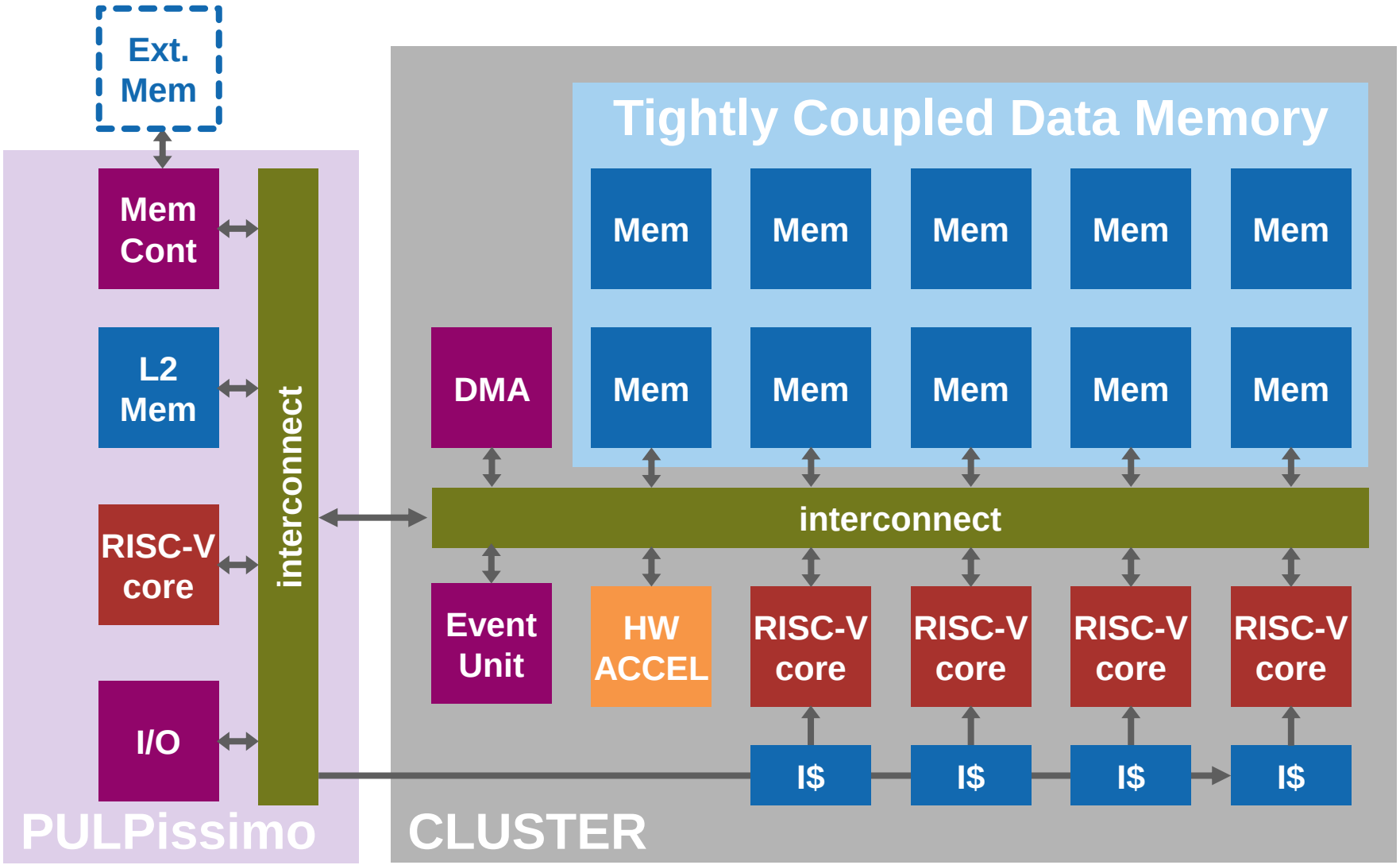
Event unit to manage resources (fast sleep/wakeup)



PULP

ETH

An additional microcontroller system (PULPissimo) for I/O



PULP

ETH

Finally multi-cluster PULP systems for HPC applications

RISC-V Cores

- RI5CY 32b
- Micro riscy 32b
- Zero riscy 32b
- Ariane 64b

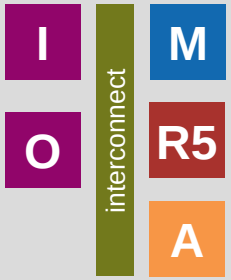
Peripherals

- JTAG
- SPI
- UART
- I2S
- DMA
- GPIO

Interconnect

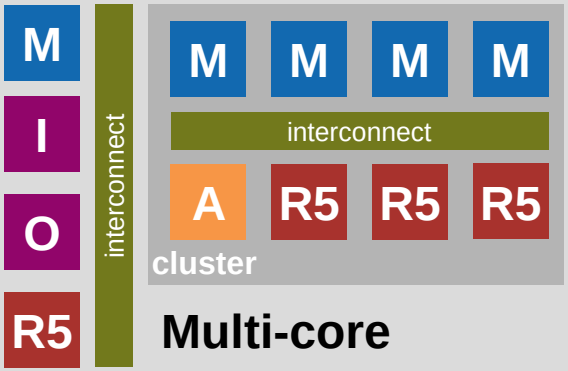
- Logarithmic interconnect
- APB – Peripheral Bus
- AXI4 – Interconnect

Platforms



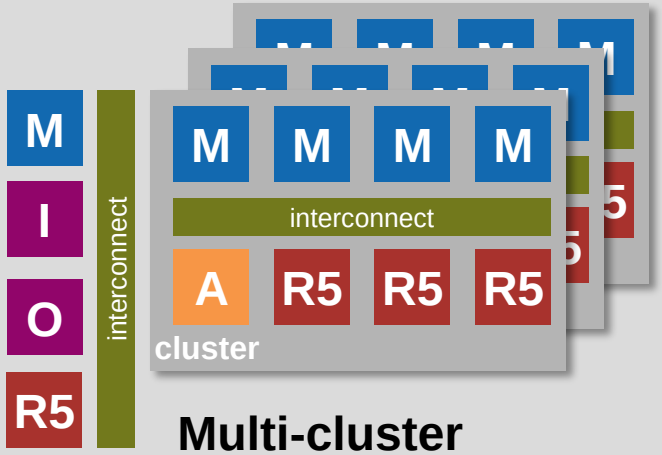
Single Core

- PULPino
- PULPissimo



Multi-core

- Fulmine
- Mr. Wolf



Multi-cluster

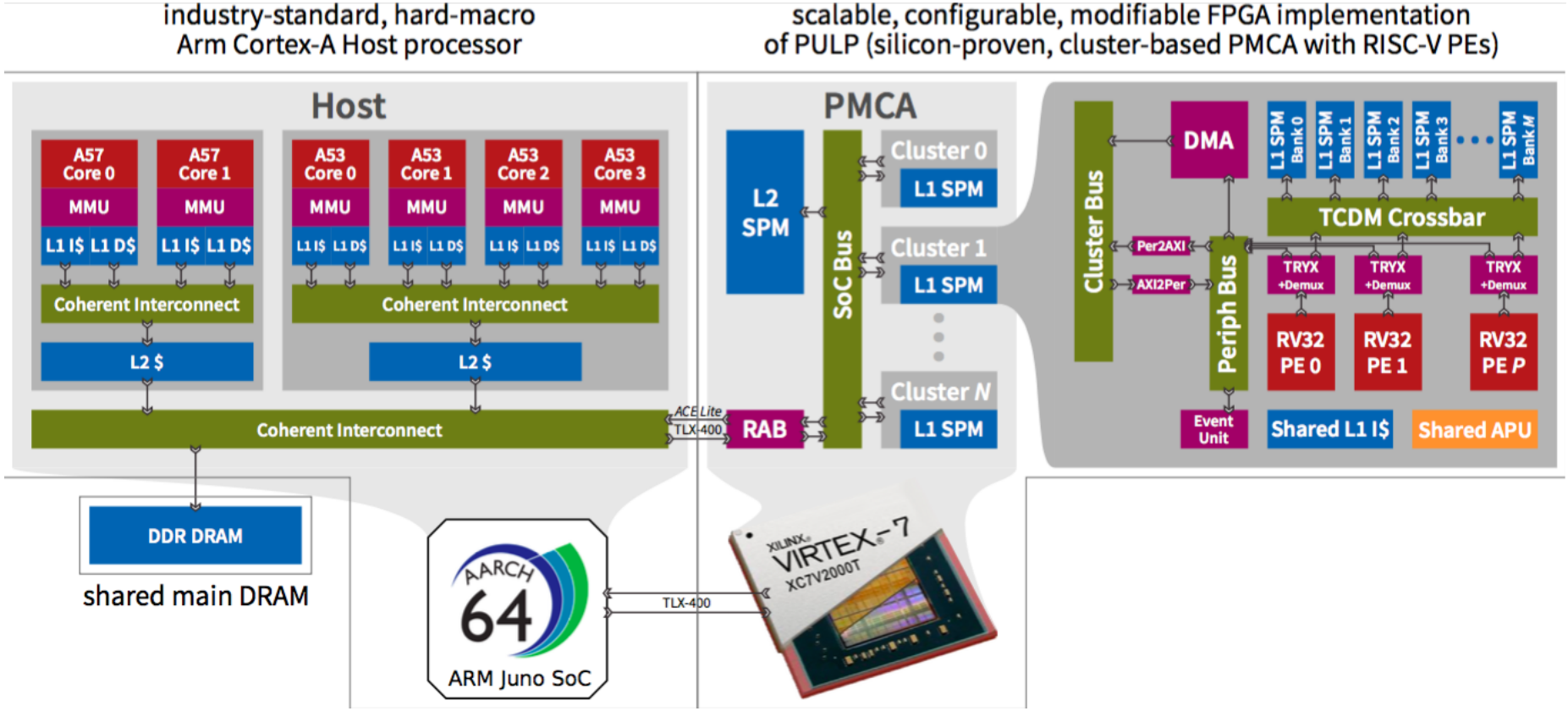
- Hero

IOT → HPC

Accelerators

- HWCE (convolution)
- Neurostream (ML)
- HWCrypt (crypto)
- PULPO (1st order opt)

Heterogenous Research Platform



- First released in 2018
- Allows a PULP cluster to be connected to a host system

PULP

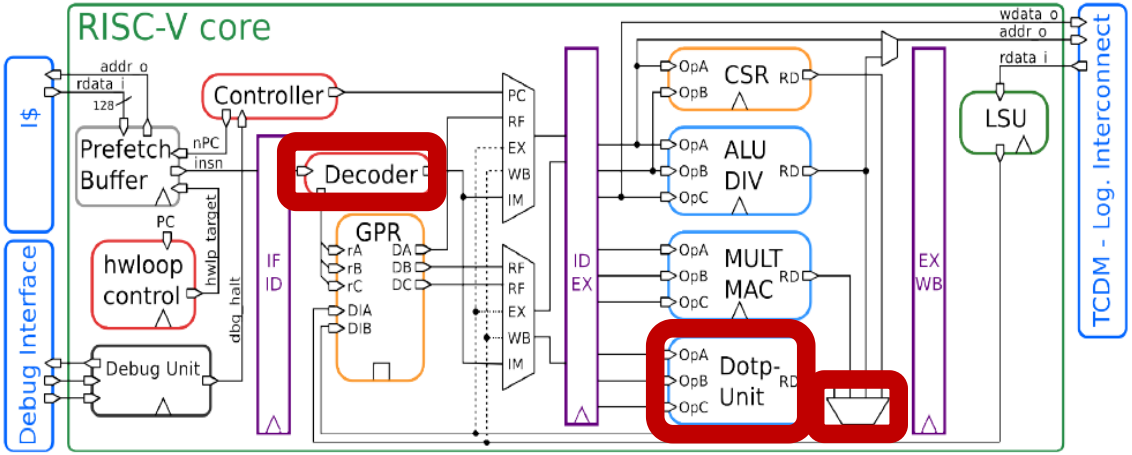
ETH

How to accelerate processing in PULP systems

- **Standard peripheral talking over AXI/APB**
  - Common in practice, nothing new or special
- **Instruction set extensions**
  - Possible in RISC-V, RI5CY has many new DSP instructions
  - Goal is to get '**good instructions**' into standard extensions at some point
- **Shared functional units**
  - Amortizes expensive extensions (FPU/DIV) between multiple units
- **Additional pipeline stages**
  - Work in Patronus for Control flow Integrity
- **Shared memory accelerators**
  - Our bread and butter, PULPopen, NTX
- **Cluster as an accelerator**
  - HERO, BigPULP, etc

What Kind of Acceleration: ISA Extensions

- High Flexibility, relatively small performance boost
- Integrated in the Pipeline of Processors (ID Stage, EX Stage, WB Stage)
- Suffer from Register File Bandwidth Bottleneck (only 2 operands…)
- Require Adapting Compiler and Binutils
- Auxiliary Processing Units (APU), interface available in the RI5CY
- Examples: Dot product (already implemented), bit-reverse, butterfly…



Programming model

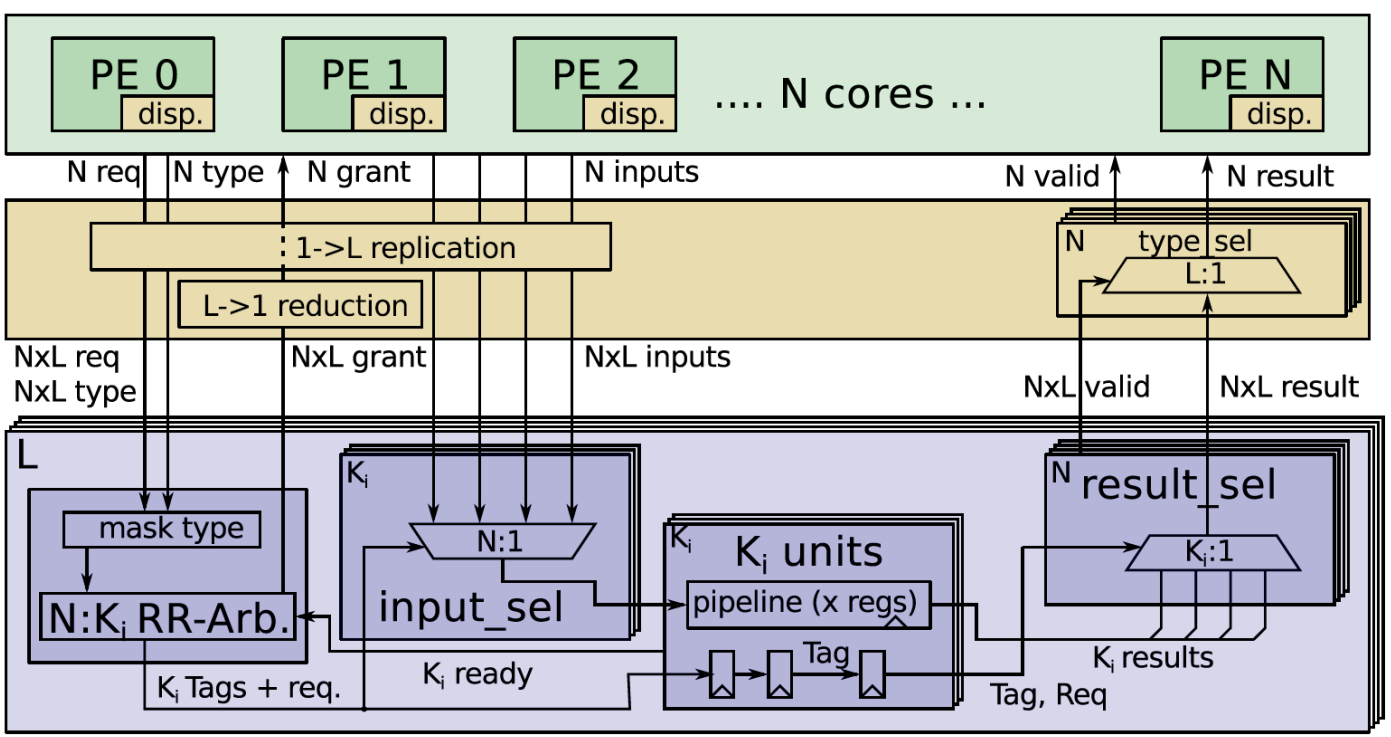
```
#define SumDotp(a, b, c) \
__builtin_pulp_sdotsp2(a, b, c)

for (int k = 0; k < (N>>1); k++) {
    VA = VectInA[k];
    VB = VectInB[k];
    S = SumDotp(VA, VB, S);
}
```

PULP

ETH

What Kind of Acceleration: Shared functional units

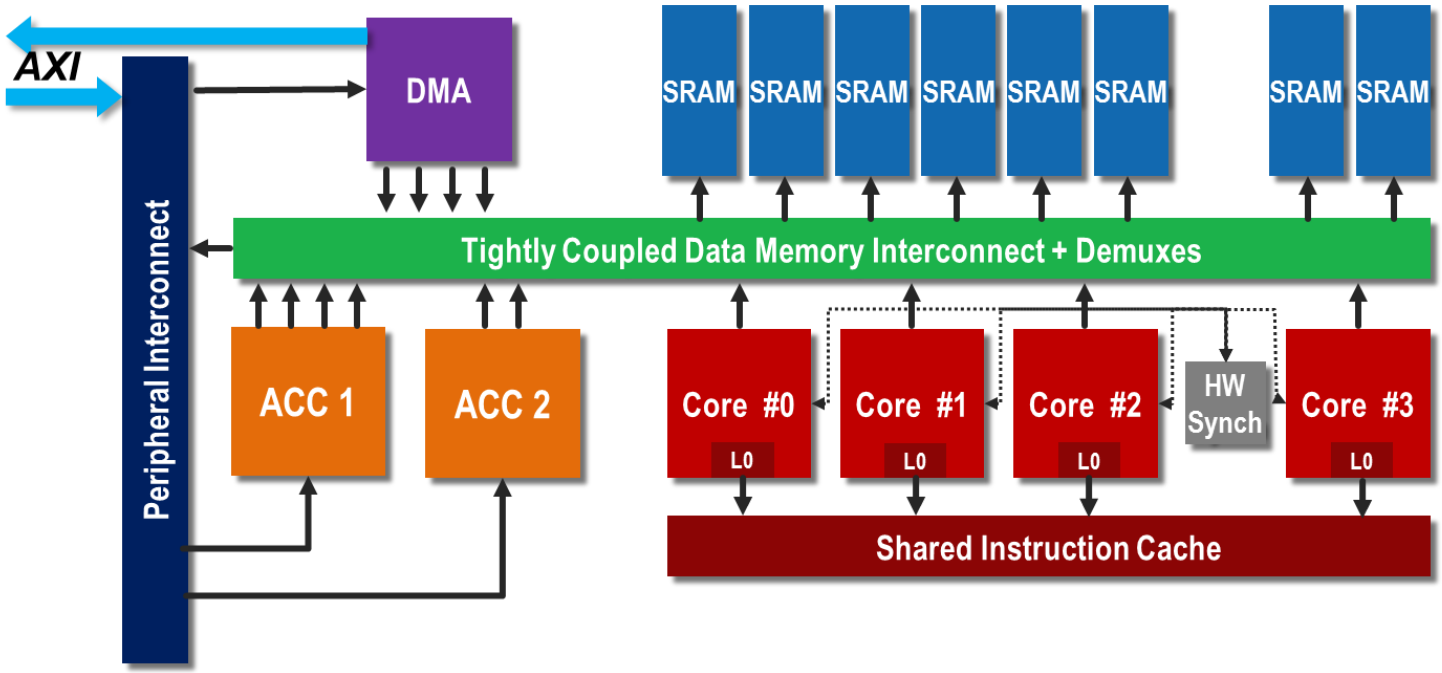
- The same as previous one, but one unit can be shared among multiple cores
- Useful to save area for low-utilization instructions (i.e. < 1/#cores%)
- Examples: Shared FPU (SQRT, DIV…)



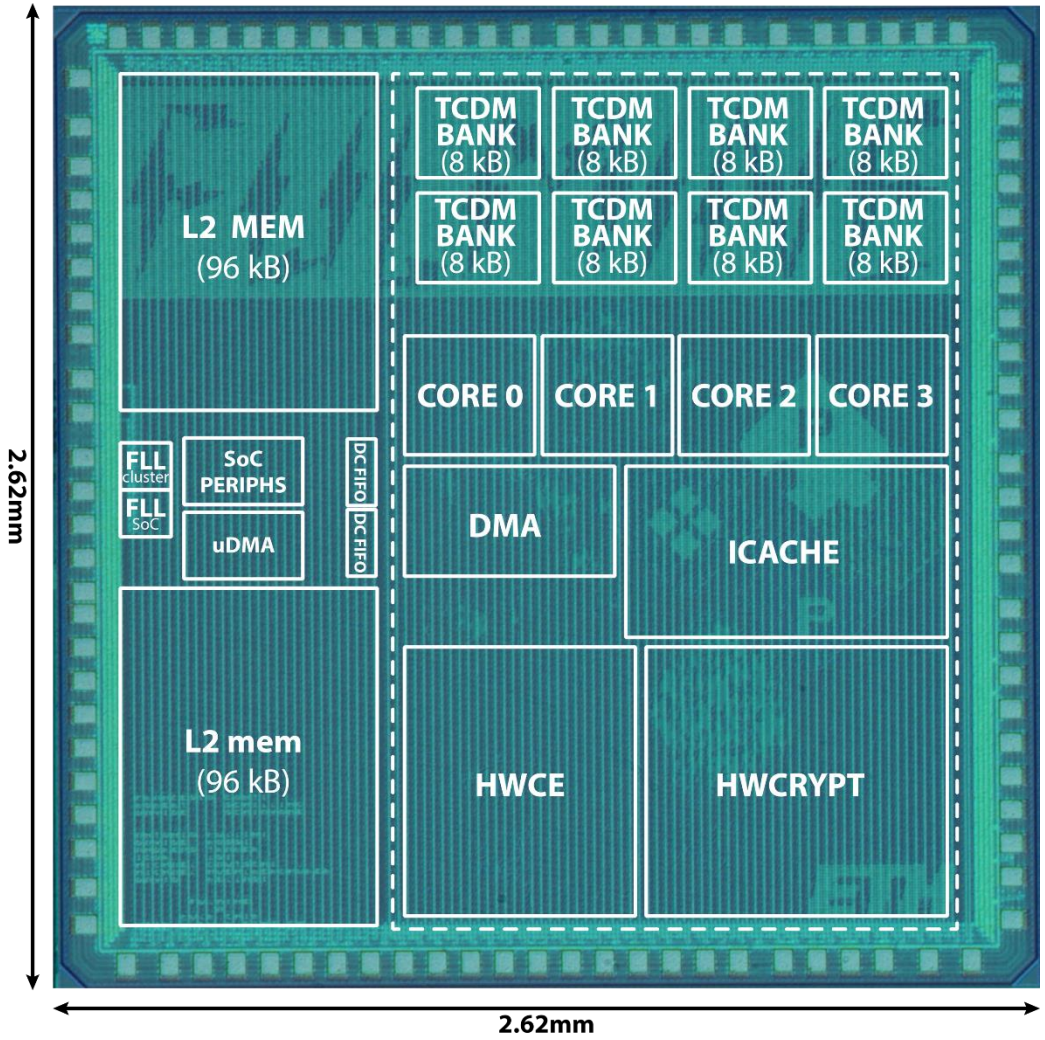
What Kind of Acceleration: Shared memory accelerators

Coarse-Grained Shared-Memory Accelerators

- DFGs mapped In Hardware (ILP + DLP) →Highest Efficiency, Low Flexibility
- Sharing data memory with processor for fast communication → low overhead
- Controlled through a memory-mapped interface
- Typically one/two accelerators shared by multiple cores



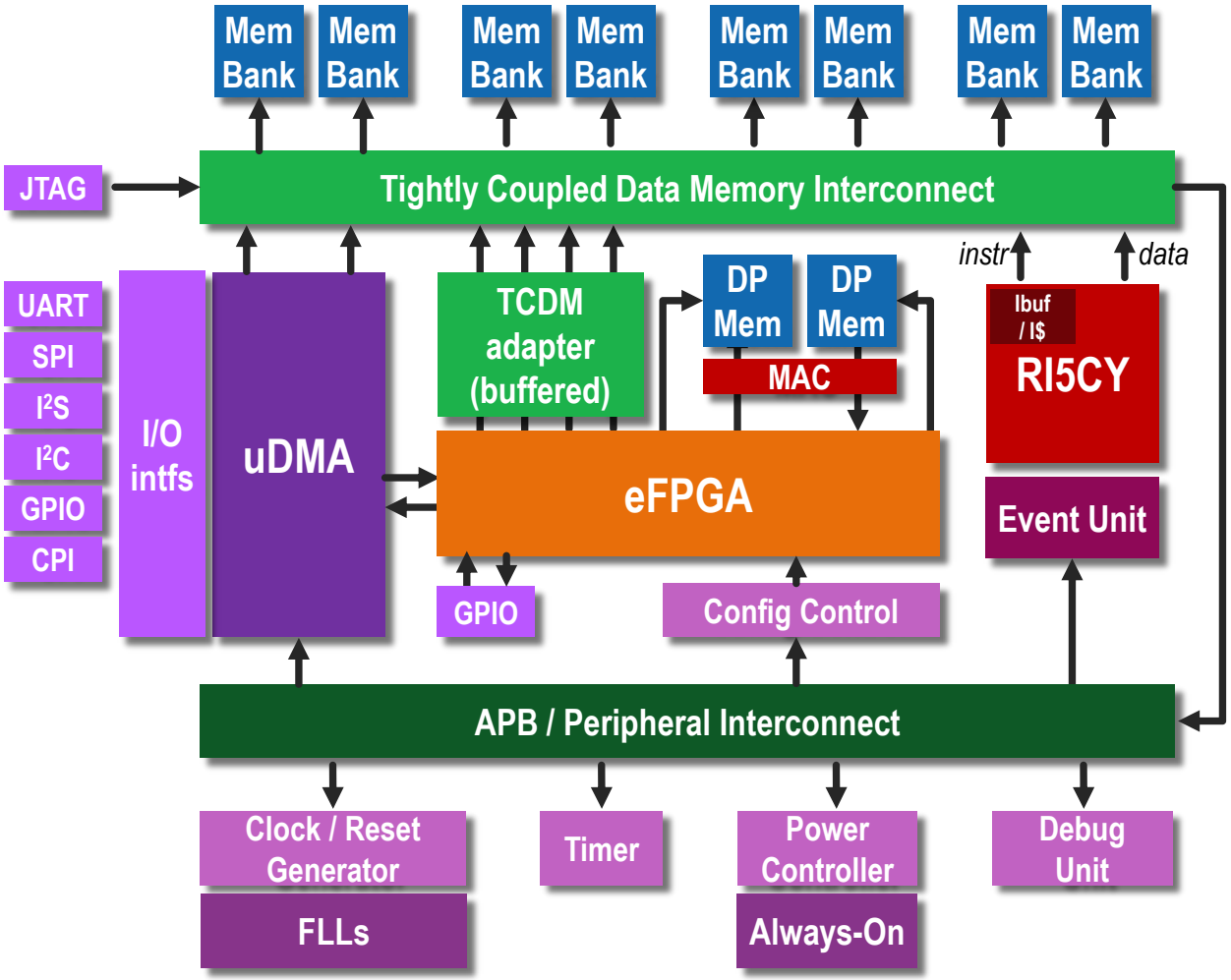
Fulmine: a HW-Accelerated Secure IoT System-on-Chip



- **UMC 65nm technology**
  - 6.86 mm^2
- **4 cores, 2 accelerators**
  - HWCE for 3D conv layers
  - DSP-optimized cores
  - HWCRYPT for AES
- **64 kB of L1, 192 kB of L2**
- **First version of uDMA for I/O with no SW intervention**
  - QSPI master/slave
  - I2C
  - I2S
  - UART

F. Conti et al., An IoT Endpoint System-on-Chip for Secure and Energy-Efficient Near-Sensor Analytics, IEEE TCAS-I 2017

The Quicklogic eFPGA integration in PULPissimo



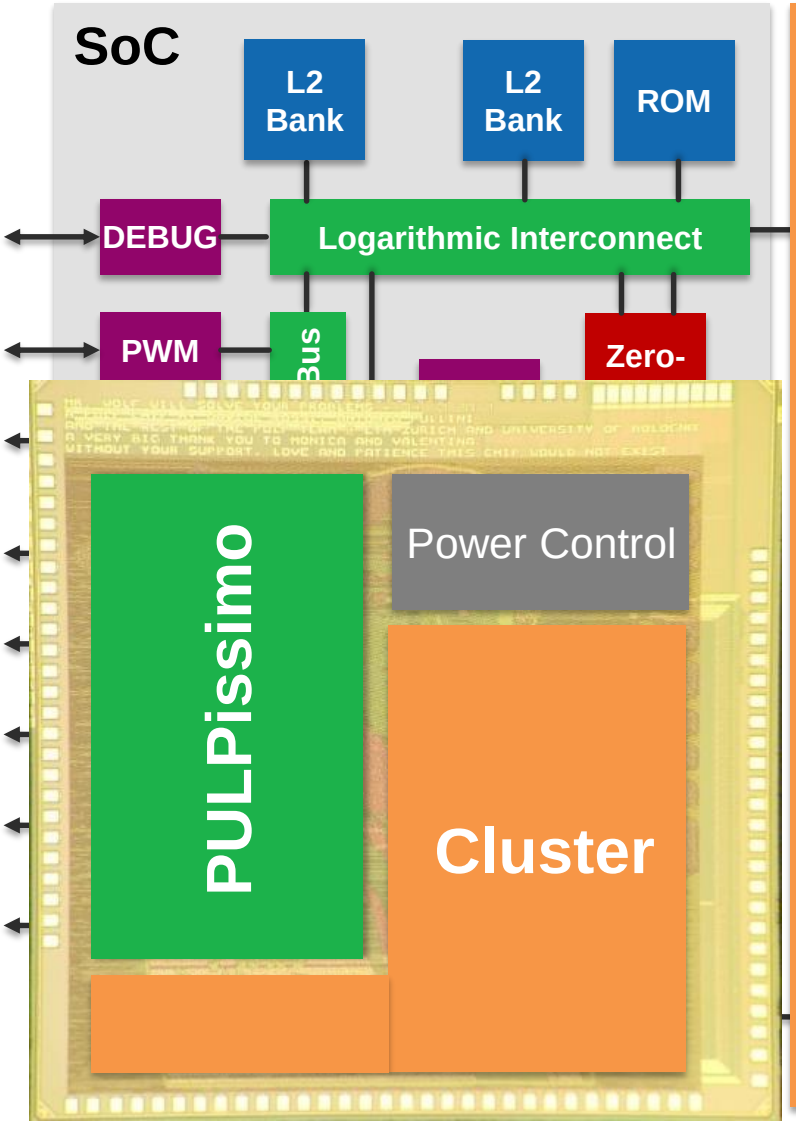
Blocks not drawn to scale

- APB port for configuration, programming and control
- Direct 4x TCDM access for eFPGA
  - 128 bits/cycle
  - 4 independent R/W
- Possibility to use uDMA

PULP

ETH

What Kind of Acceleration: Cluster as an accelerator

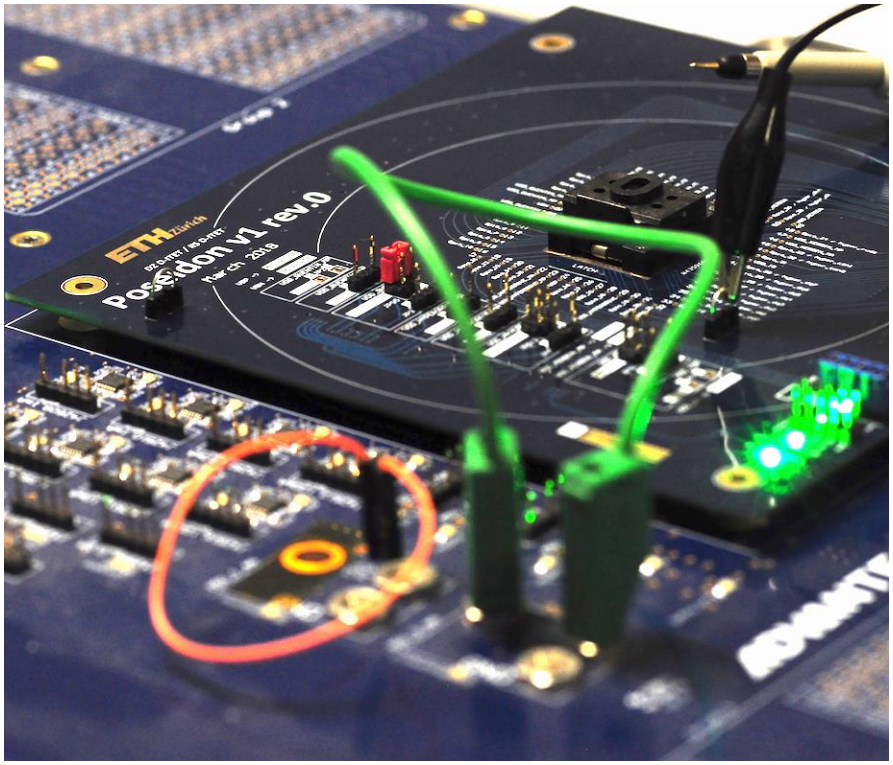


Cluster of RISC-V cores as an accelerator

PULP

ETH

We have designed more than 25 ASICs based on PULP

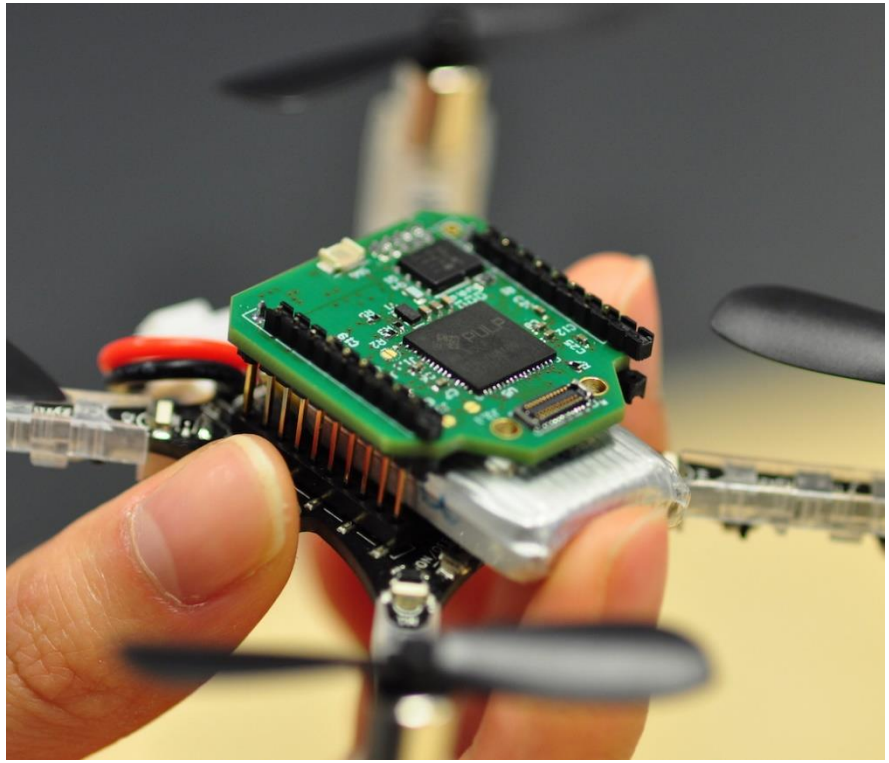


ASICs meant to go on IC Tester

- Mainly characterization
- Not so many peripherals

ASICs meant for applications

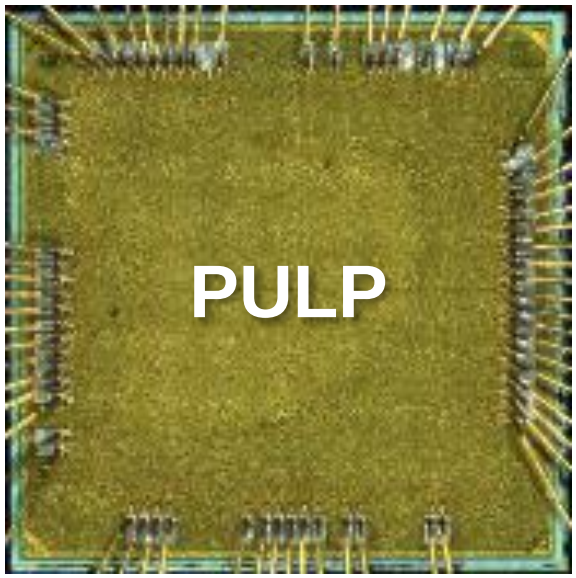
- More peripherals (SPI, Camera)
- More on-chip memory

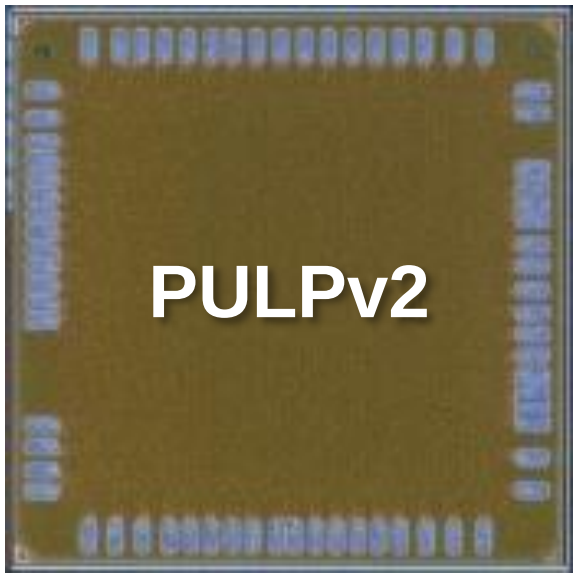


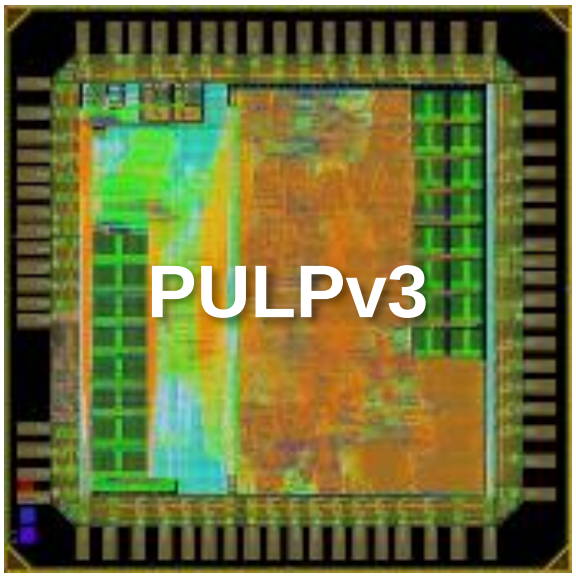
PULP

ETH

Brief illustrated history of selected ASICs

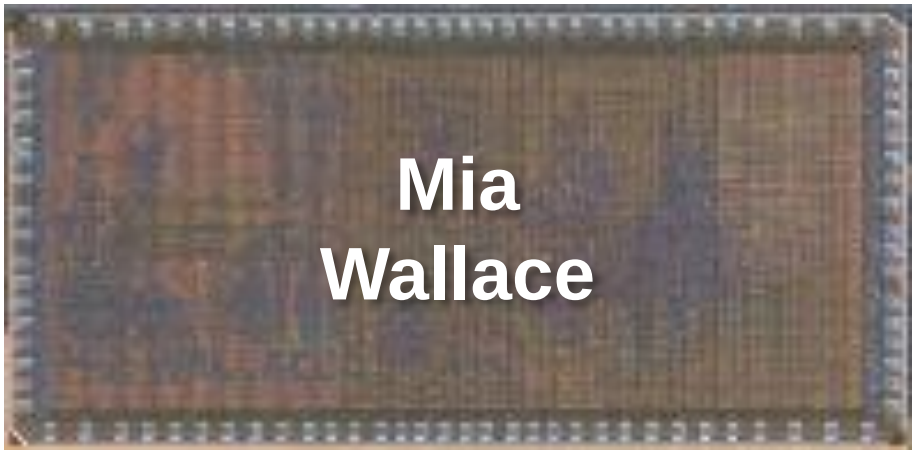






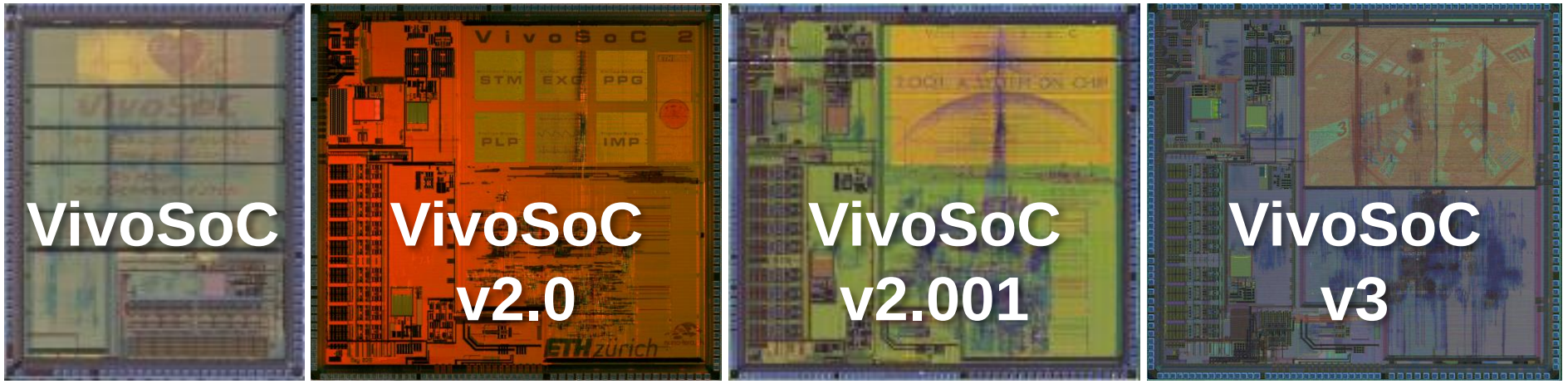
- All are 28 FDSOI technology, RVT, LVT and RVT flavor
- Uses OpenRISC cores
- Chips designed in collaboration with STM, EPFL, CEA/LETI
- PULPv3 has ABB control

The first system chips, meant for applications



- First multi-core systems that were designed to work on development boards. Each have several peripherals (SPI, I2C, GPIO)
- **Mia Wallace** and **Fulmine** (UMC65) use OpenRISC cores
- **Honey Bunny** (GF28 SLP) uses RISC-V cores
- All chips also have our own FLL designs.

Combining PULP with analog front-end for Biomedical apps

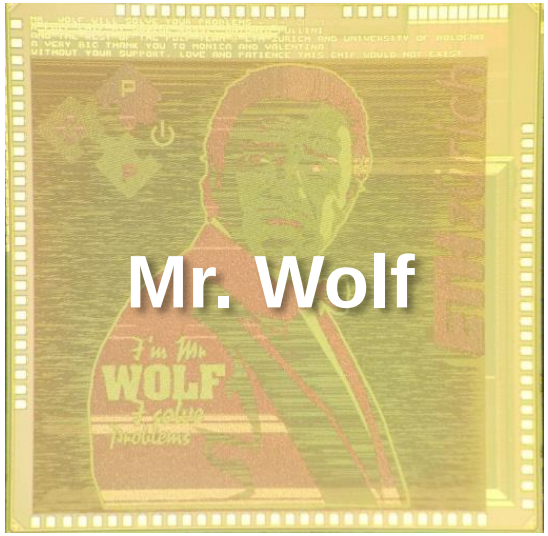


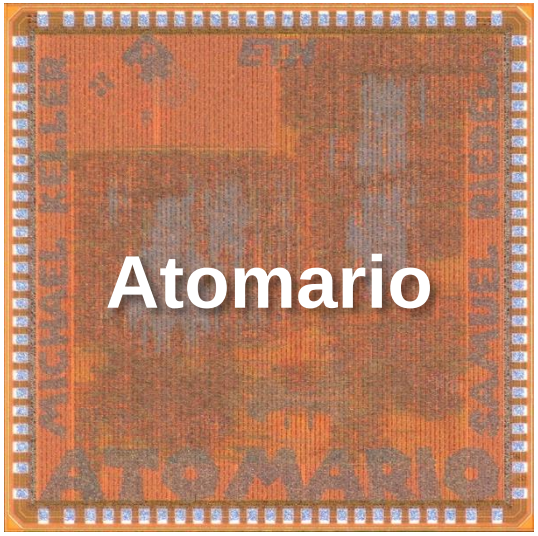
- Designed in collaboration with the Analog group of Prof. Huang
- All chips with SMIC130 (because of analog IPs)
- First three with OpenRISC, VivoSoC3 with RISC-V

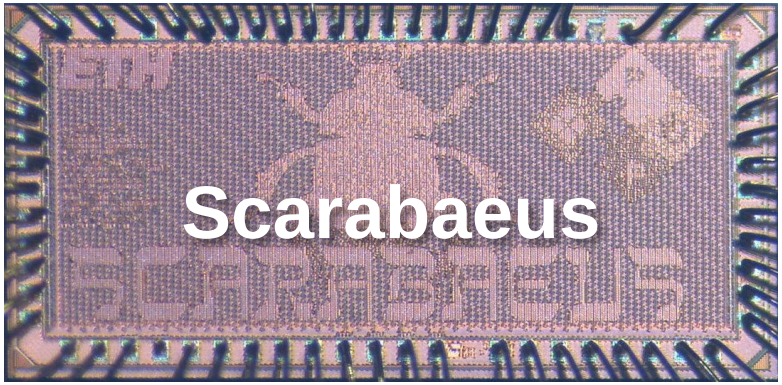
PULP

ETH

The new generation chips from 2018

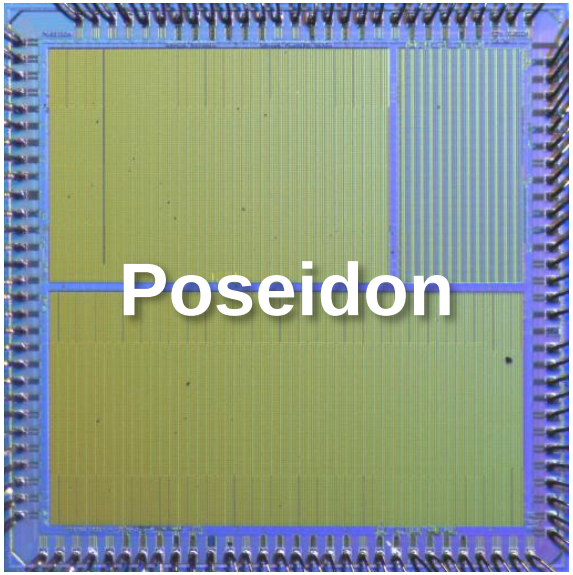


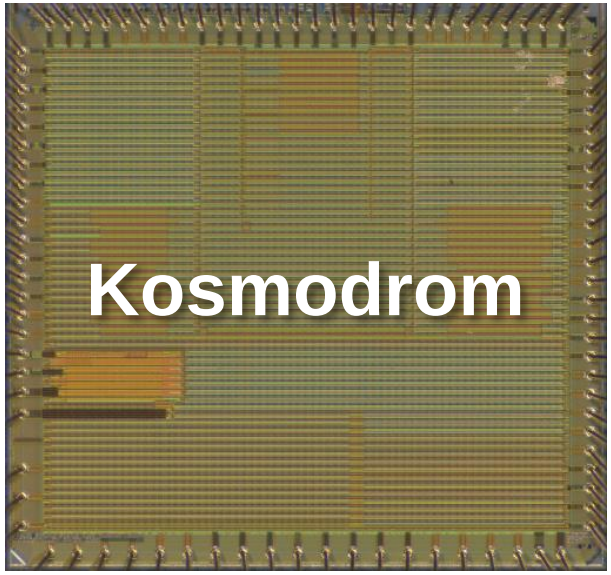


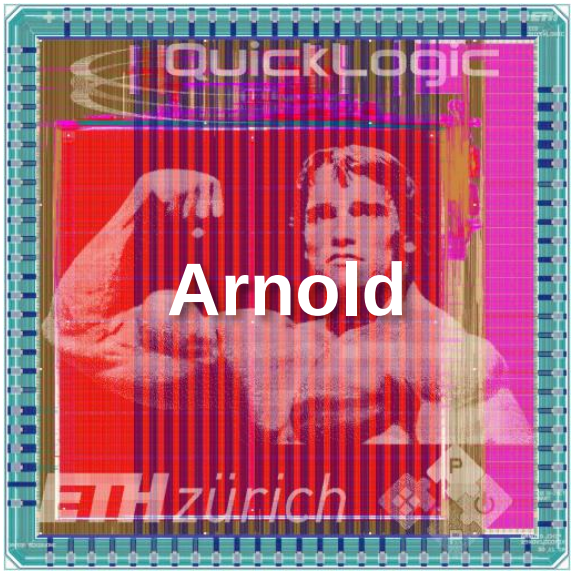


- System chips in TSMC40 (Mr. Wolf) and UMC65
- **Mr. Wolf**: IoT Processor with 9 RISC-V cores (Zero-riscy + 8x RI5CY)
- **Atomario**: Multi cluster PULP (2x clusters with 4x RI5CY cores each)
- **Scarabaeus**: Ariane based microcontroller

The large system chips from 2018





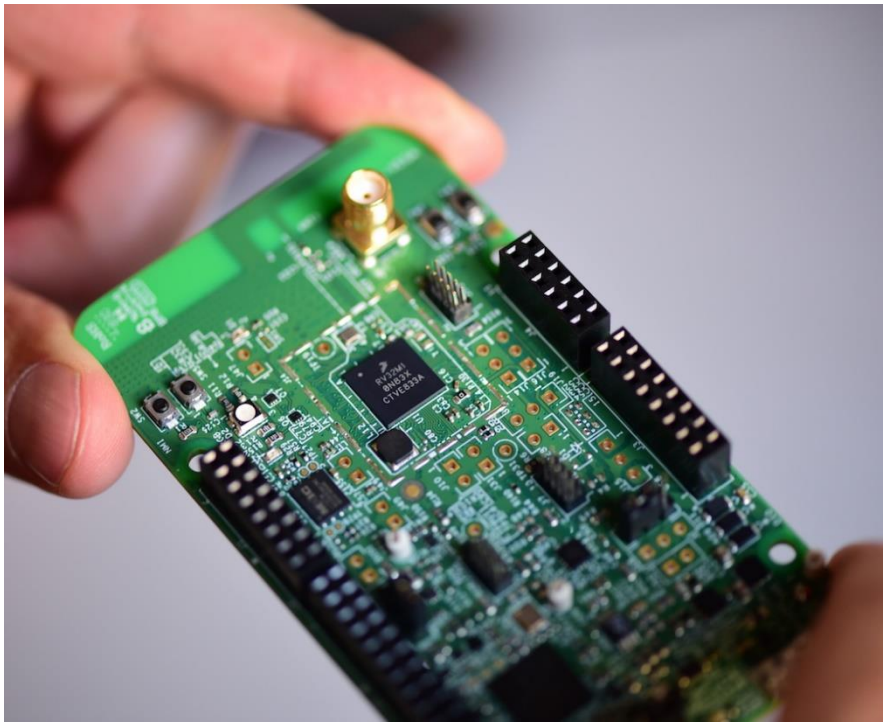


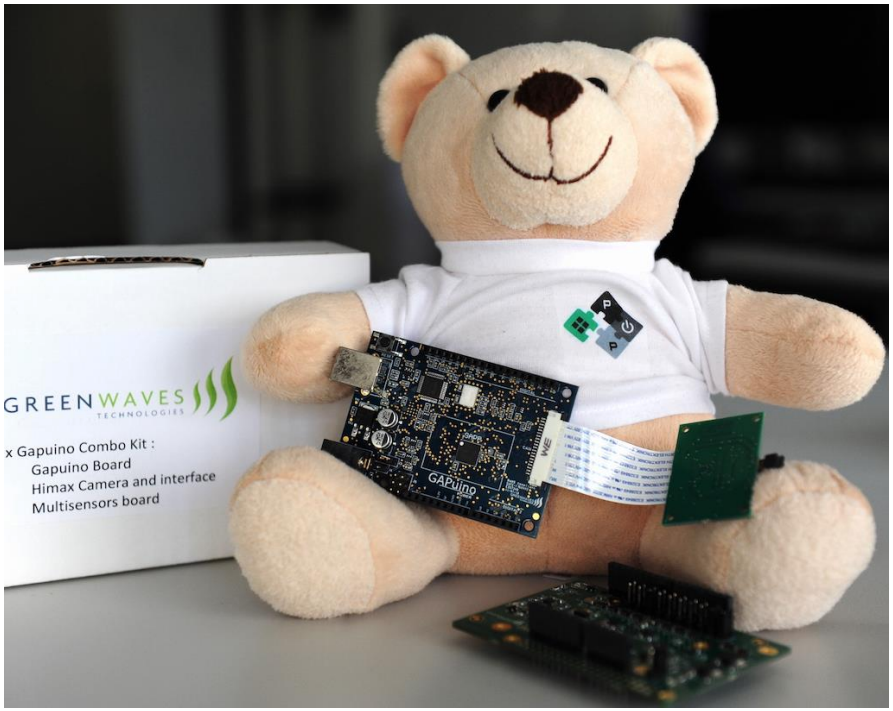
- All are 22nm Globalfoundries FDX, around 10 sqmm, 50-100 Mtrans
- **Poseidon**: PULPissimo (RI5CY) + Ariane
- **Kosmodrom**: 2x Ariane + NTX (due this week)
- **Arnold**: PULPissimo (RI5CY) + Quicklogic eFPGA

You can buy development boards with PULP technology

VEGA board from open-isa.org

- Micro-controller board with RI5CY and zero-riscy





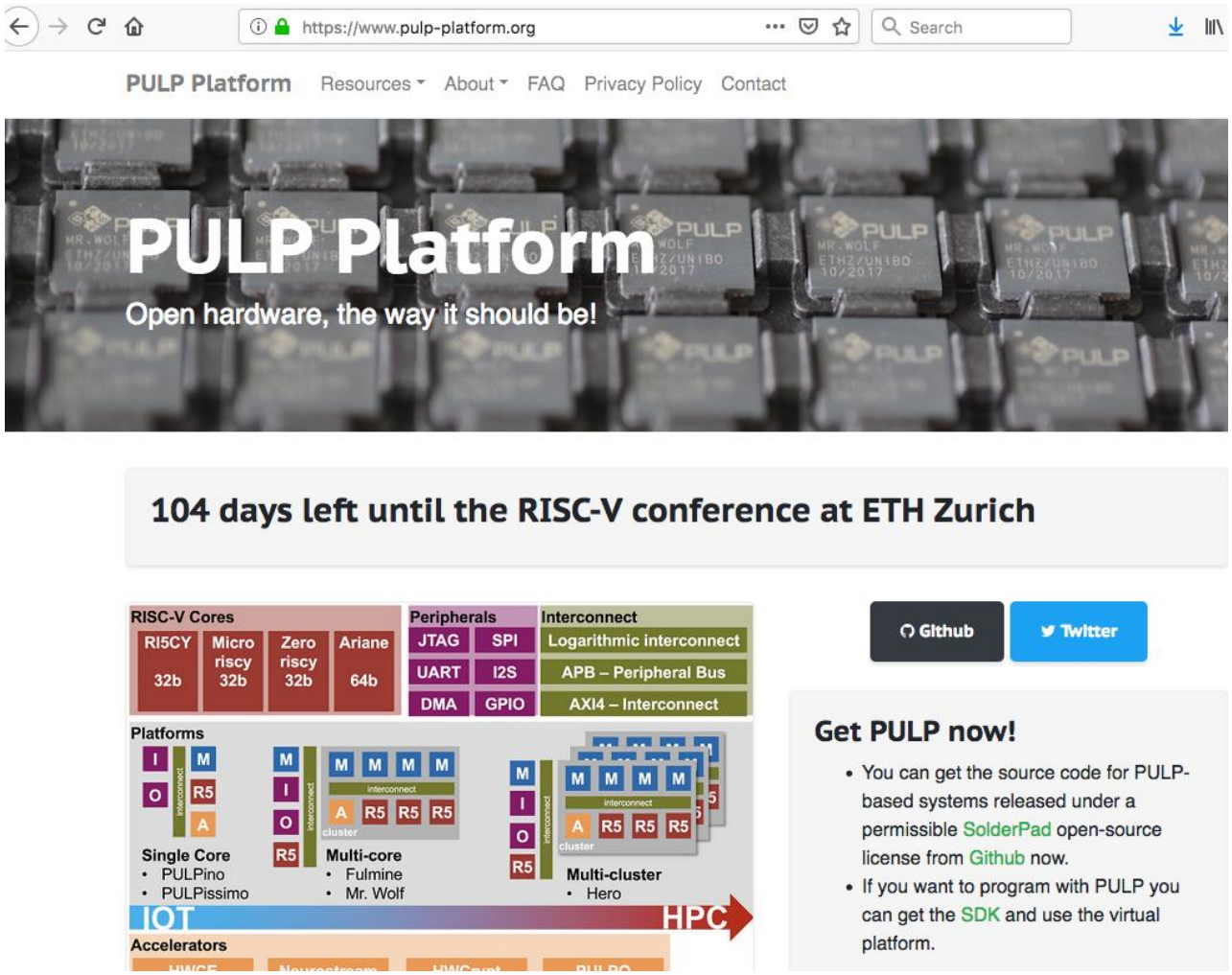
GAPUINO from Greenwaves

- PULP cluster system with Nine RI5CY cores

PULP

ETH

We firmly believe in Open Source movement



First launched in February 2016 (Github)

All our development is on open repositories

Contributions from many groups

PULP

ETH

We provide PULP with SOLDER Pad License

- Similar to Apache/BSD, adapted specifically for Hardware
- Allows you to:
  - Use
  - Modify
  - Make products and sell them

  without restrictions.

SOLDERPad

http://www.solderpad.org/licenses/

- Note the difference to **GPL**
  - Systems that include PULP do not have to be open source (Copyright not Copyleft)
  - They can be released commercially
  - LGPL may not work as you think for HW

Silicon and Open Hardware fuel PULP success

- **Many companies (we know of) are actively using PULP**
  - They value that it is **silicon proven**
  - They like that it uses a **permissive open source license**

PULP

ETH

http://pulp-platform/wosh

WOSH

the Week of Open Source Hardware

June 11-14 Zürich, SWITZERLAND

- Official RISC-V Workshop (June 11-12)
- RISC-V foundation member meetings (June 13)
- Eurolab4HPC, Open Source Innovation Camp (June 13)
- Licensing and IP rights for Open source HW (June 13)
- FOSSI: Path to high quality IP, Open source EDA tools (June 14)
- Tutorials, demos, hackathons

PULP

ETH

Week of Open Source Hardware, June 11-14, Zurich

	RISC-V FOUNDATION	WOSH SESSIONS	WOSH TUTORIALS
Tuesday 11th of June	RISC-V Day1		
Wednesday 12th of June	RISC-V Day2		
Thursday 13th of June	Member meetings	Licensing Innovation with OSH	PULP OpenPiton
Friday 14th of June		High Quality IP in OSH Open Source EDA tools	MyriadRF OpenISA.org Greenwaves Hero

Paid Registration

OPEN Now

FREE Registration (expect early March)

Working on Program Now

If you have ideas Contact me

PULP

ETH

Future of PULP, what we expect

- **Working on a non-profit organization to:**
  - Manage the distribution of PULP
  - Governance
  - Technical Support
  - Continuity
- **ETH Zürich and University of Bologna will be contributors**
  - We can concentrate on our research, energy efficient processor systems
  - Continue using PULP in our work
  - Become a regular contributor
- **Hope to announce during the WOSH in Zurich**
  - There is some work towards this goal, nothing official yet

QUESTIONS?