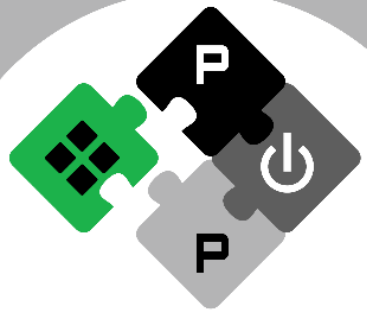
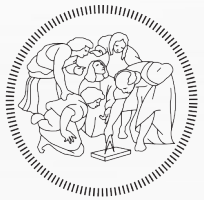


PULP: Parallel Ultra Low Power
a platform for next generation IoT Applications

ACA PROJECT 2016/2017
Prof. Cristina Silvano

Stefano Brandoli



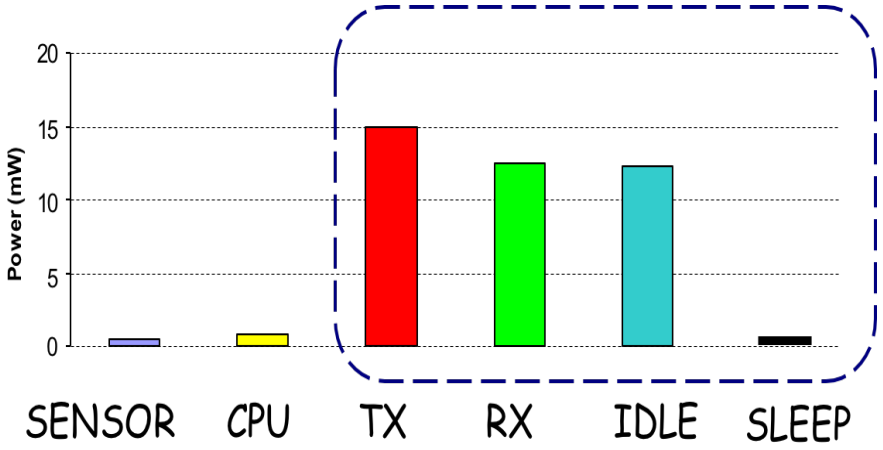
POLITECNICO
MILANO 1863

Introduction: IoT world

REALITY:
Sensor nodes have **limited power source (batteries)**

PROBLEM:
Recharging and/or battery replacement may be immaterial or too expensive

SOLUTIONS:
- **Networking Protocols**
- Low-power processing
- **Low-power processors**
- Low-power sensing
- **Energy Harvesting**



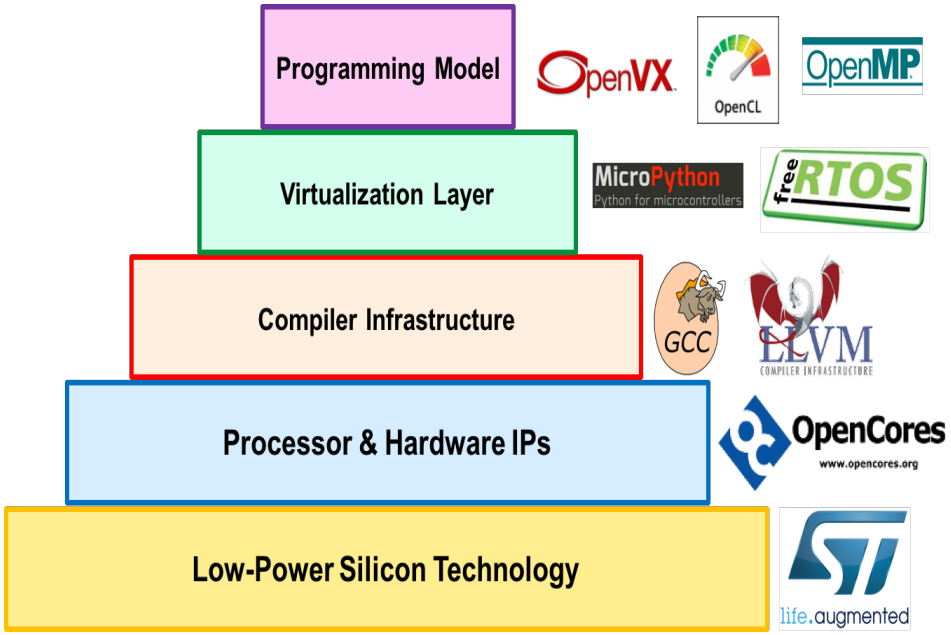
- **Tx/Rx consume much of the power**
  - ➢ disable them as soon as possible
- **Idle consumes similarly to Tx/Rx**
  - ➢ to reduce power consumption we have to reduce IDLE time
- **Low CPU Power consumption**
  - ➢ sensor node does simple tasks

PULP
Parallel Ultra Low Power

GOAL: ENABLE MORE COMPLEX IOT APPLICATIONS USING THE SAME POWER BUDGET (few mW)
- **Performance:** architecture parallelism
- **Energy Efficiency:** voltage scaling without losing performance (**near threshold operations**)

Introduction: The PULP platform

- Energy efficient **many core SoC**
- **Software frameworks** (*OpenMP, OpenCL, OpenVX*):
  - **exploit hardware parallelism**
  - enable agile portings, development, debugging, tuning
- **Custom toolset** (*Virtual Platform, profiling tools, FPGA emulation…*)
- Large number of IPs



pJ/op = target of uControllers and ASIC devices

- ➢ not able to deliver the required computational power

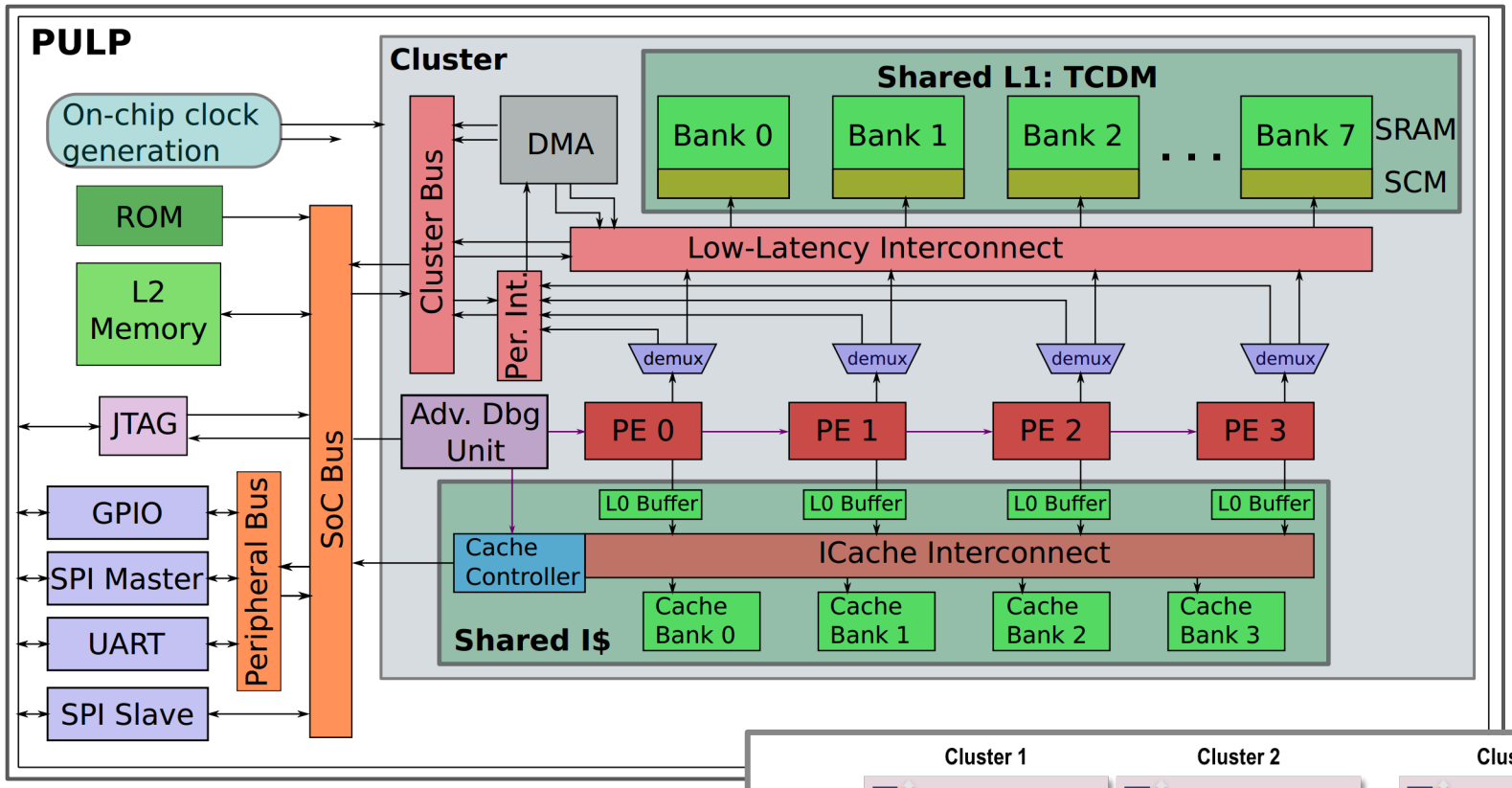
IDEA:

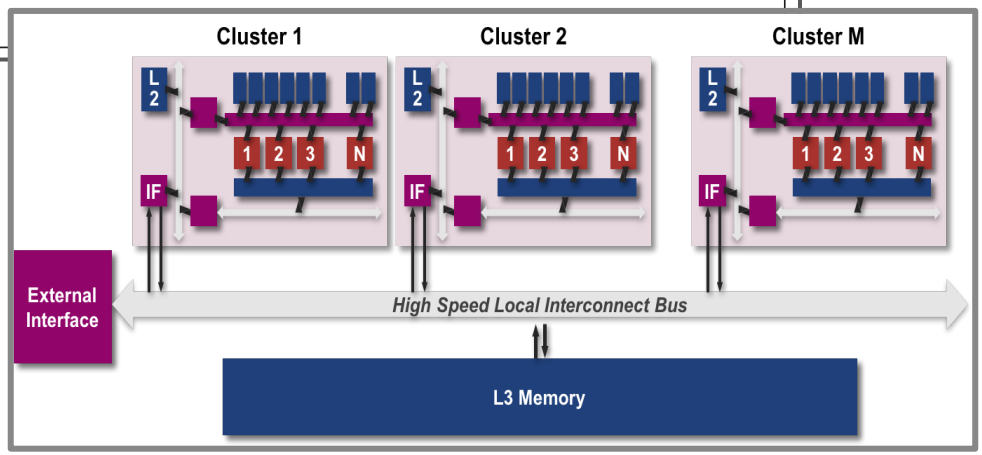
- **expose low power features of the technology**
- use low power silicon technology

FEATURES:

- **67 MOPS/mW**: 65nm bulk CMOS
- **193 MOPS/mW** (40MHz, 1mW): low power STM 28nm FDSOI

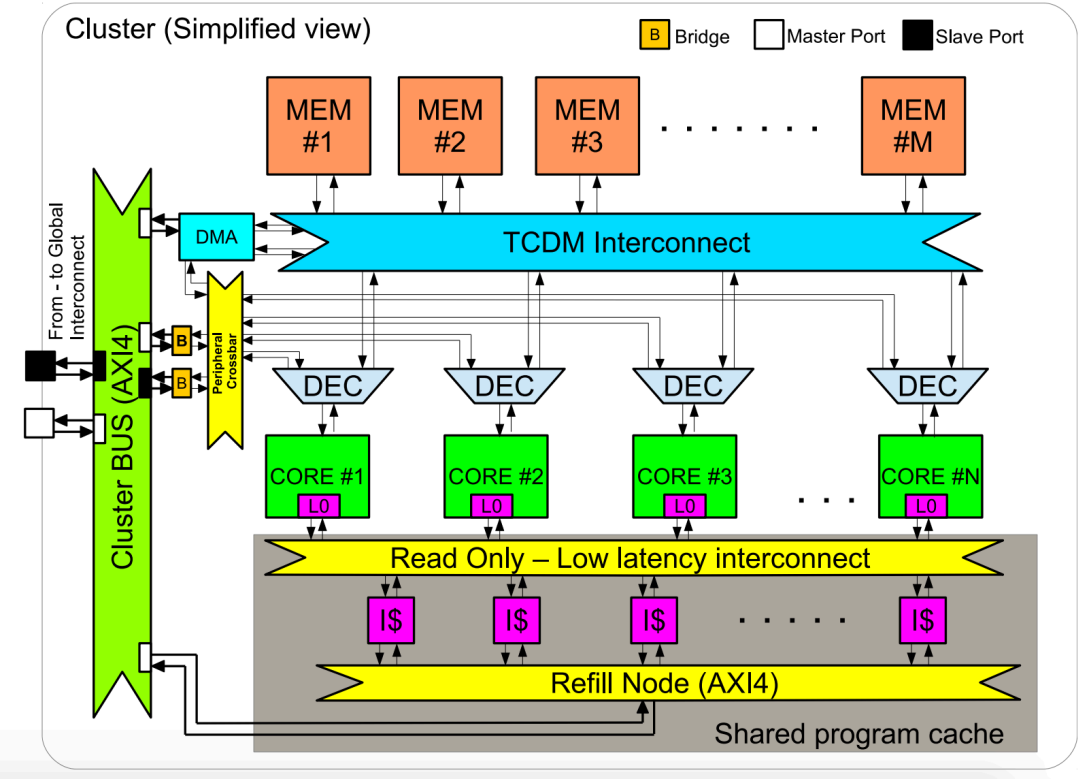
Architecture: Overview





- 2-16 cores per cluster: tipically 4
- Multiple clusters per chip
- Cores (PEs) using extended RISC-V ISA
- Shared L2 memory (32 kB - 128 kB)
- Shared L3 outside clusters

Architecture: Shared I-Cache



- The **Refill Node (AXI4) of the shared I-Cache** connects to the system bus
- System bus connects to:
  - L2 memory
  - Several peripherals
- Shared I-Cache implemented with SCMs

Architecture: I-Cache

- **Shared 4-way set associative I-Cache** with **4 cache banks** of 1 KB each

- **Shared I-Cache vs Private I-Cache**

PROs

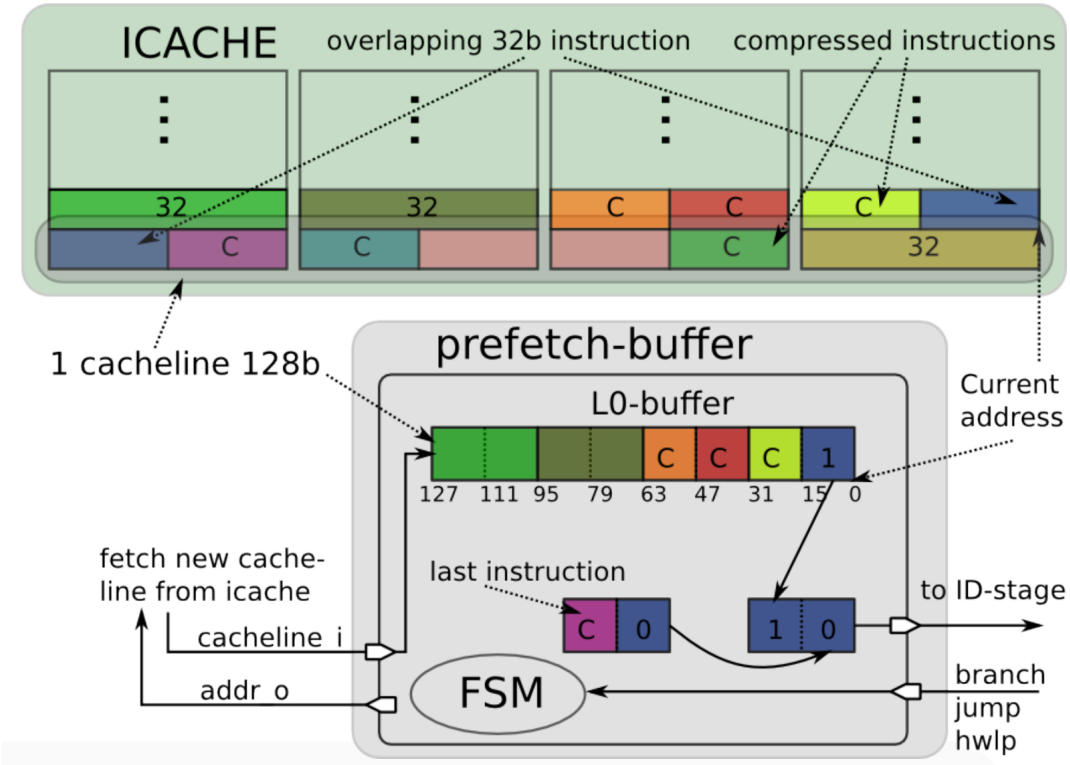
- **Avoid Data Replication**:
  several cores fetch the same code segment, that is stored in each **private I-cache** (copies). Multiple refill requests are asserted at the same address, creating intense traffic to a slow and far L2 memory subsystem (**~10-40 cycles**)
- **Better Miss Ratio**:
  bigger memory space seen by each core

CONs

- **Less speed**
  more complex design (vs Direct Mapped)
- **Increased HW complexity**

- **L0 Pre-fetch Buffer (128 bit per core):**
  - Contains a **complete cache line** (128 bit)
  - **Reduce pressure (and contention) on shared I-Cache**:
    only in case of miss the request is forwarded to the shared I-Cache (in the same cycle)

Architecture: L0 Pre-Fetch Buffer



Support for Compressed Instruction (16 bit):

Inevitably some 32 bit instructions will become unaligned.

Requires an **additional fetch cycle** for which the **processor has to be stalled**

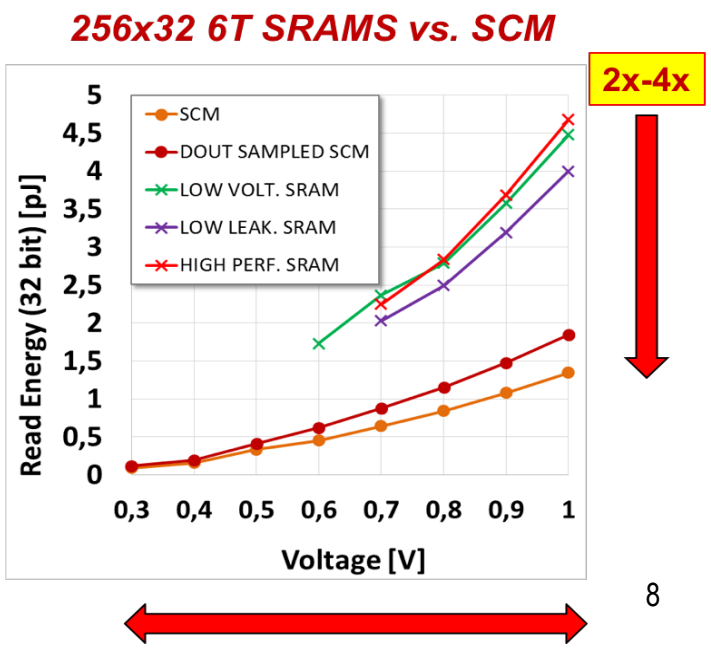
- **SOLUTION: use an additional register to keep the last instruction:**
  this register will contain the lower 16 bit of the instruction which can be combined with the higher part and forwarded to the ID-stage

Architecture: Shared Data Memory

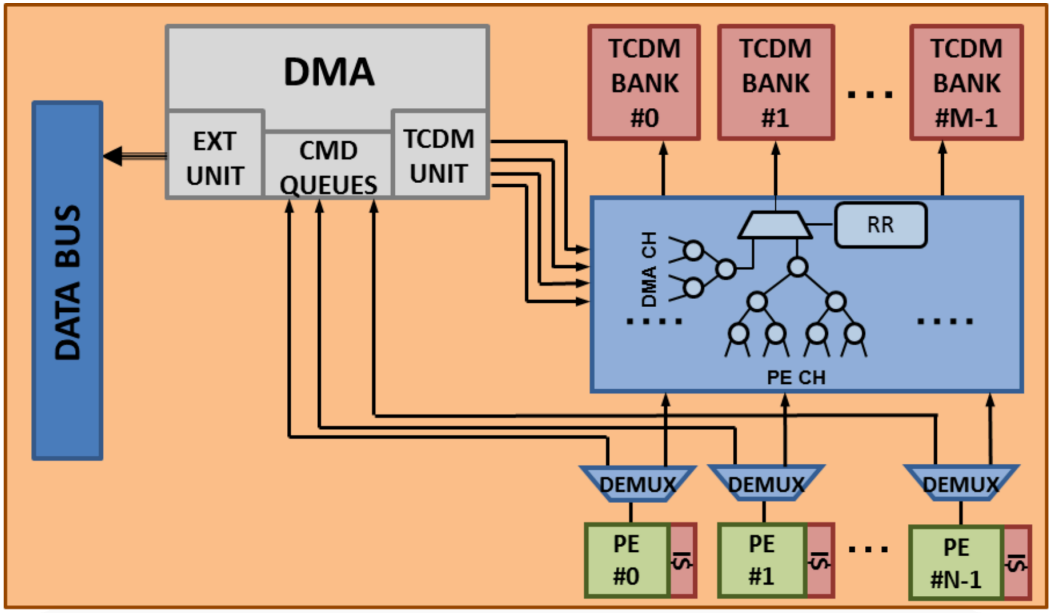
- **Cores don't have private Data Caches:**
  - avoid **memory coherence** overhead
  - **improving power and area efficiency** for data memory

- **Shared L1 multi-banked TCDM (*Tightly Coupled Data Memory*):**
  - parametric number of word-level interleaved banks
  - acts as a software-managed shared data scratchpad
  - logical banks connected to the processors through a non-blocking interconnect to minimize banking conflicts
  - **1 cycle communication between PEs and memory banks (read/write)**
  - **1 port for every memory bank**
  - **read unaligned data in only 2 cycles**

- TCDM logical banks are **heterogeneous memory**:
  - **SRAMs** (16x4 KB banks):
    - higher density than SCMs (3x-4x)
    - **performance degrades at low voltage**
    - low technology portability
  - **Standard Cell Memory (SCM) banks** (16x0.5 KB):
    - **work over the same voltage range as logic**
    - **energy/access lower than that of SRAMs**
    - high technology portability



General Architecture: DMA



GOAL: overlap as much as possible computation and memory transfers

DMA:

- Enables communication with other clusters, L2 memory and external peripherals
- Hides latencies of slow peripherals and of L2 memory data transfers
- **DMA + TCDM -> Double Buffering**

General Architecture: DMA

- **Lock-free per-core FIFO command queues:**
  - **Each processor has the ability to configure DMA transfers** by sending commands to a dedicated memory mapped control interface
  - If the command queue is full, the incoming request is stalled and it is committed when one location is released
  - Enqueuing and arbitration of transfer requests from PEs is performed internally by the DMA. This process is transparent to the software
- **No large internal data buffers:**
  - exploiting the cluster's shared memory as local repository

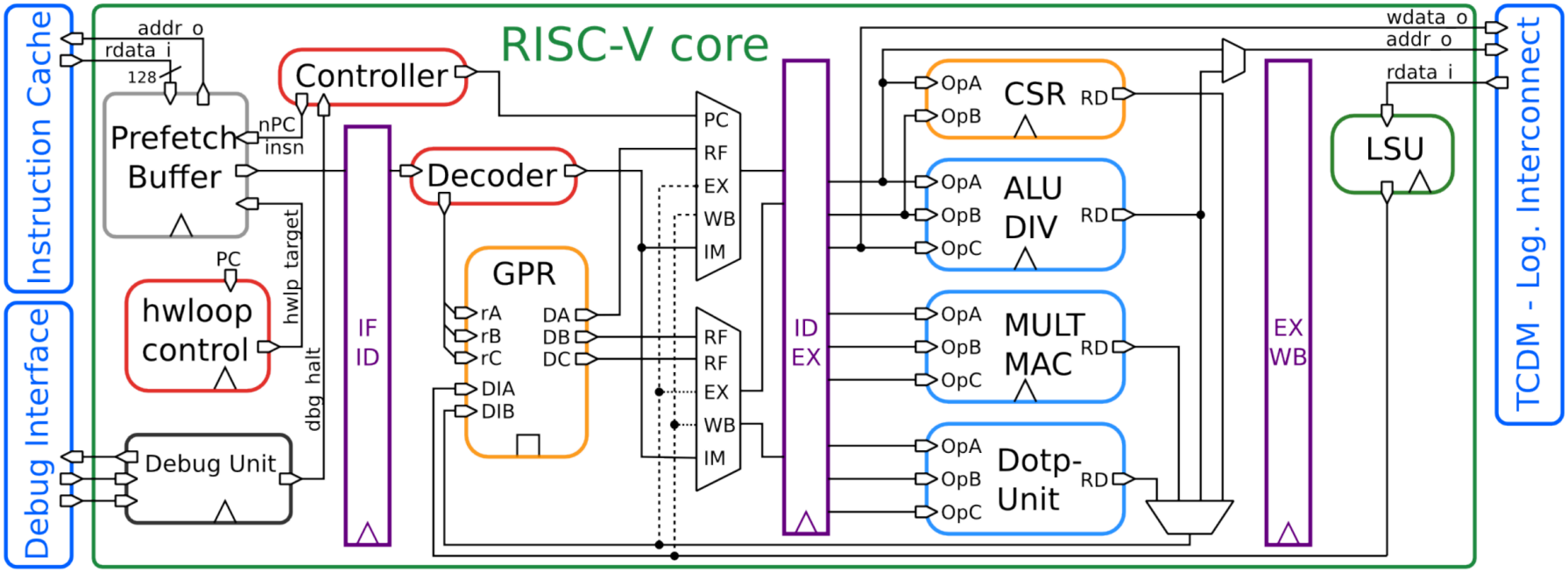
- **External Unit (EXT UNIT):**
  - decoupled read and write channels to optimize the performance when concurrent TX and RX transactions are required
  - parametric number of outstanding transactions (up to 16)
  - handles on-the-fly reordering of incoming packets

- **TCDM Unit:**
  - 4 (2 read, 2 write) ports (32-bit) for the connection with the TCDM interconnect
  - **TCDM interconnect is implemented as two separate trees**, one handling requests from processors and one used for the DMA: **Round Robin arbitration**

Power Management

- Each **PULP cluster and the rest of the SoC** are in **different power and clock domains** (**DVFS – Dynamic voltage and frequency scaling**)
- **Each processor within a cluster** can be separately **disabled, clock-gated and reverse body biased when idle**
- **The cluster is able to keep active an arbitrary number of processing elements**, while the others consume zero dynamic power and up to 10x less leakage power
- An **event unit (HW SYNCRONIZER)** automatically **manages transitions of the cores between the active and idle states**
- Processors can be put in idle state with a write operation on a memory mapped control register. After going in sleep mode, a core remains idle until a configurable event is triggered
- **Events can be issued by all IO peripherals, the DMA, and timers**

RI5CY: Optimized RISC-V core



- **Shallow 4-stage in-order 32b RISC-V processor core**
- **Comparable to an ARM Cortex M4** in terms of area and performance
- **Focus on energy efficiency: allow the core to sleep and wait for an event**

RI5CY: RISC-V ISA + Custom Extensions

- Full support for **RV32I Base Integer Instruction Set**
- Full support for **RV32C Standard Extension for Compressed Instructions**
  - **Smaller code size, less pressure on instruction cache**
  - Requires support for 16-bit aligned memory access
- Full support for **RV32M Integer Multiplication and Division Instruction Set**

➢ **PULP specific extensions:**

- **Hardware Loops**
- **Post-Incrementing load and stores**
- **ALU extensions (min, max, abs, avg)**
  common in IoT scenarios
- **Multiply-Accumulate (d = a × b + c)**
  MAC with three input operands
- **PACKET SIMD:**
  - **example: Dot Product**: instructions that can multiply four pairs of 8 bit numbers and accumulate them in a single instruction

PULP

RI5CY Extension: Hardware Loops

- Execute a piece of code multiple times, without the overhead of branches or updating a counter
- 0 stall cycles for jumping to the first instruction of a loop
- Start address, end address, counter -> registers mapped in CSR address space
- Max 2 nested loops
- Minimum loop size is two instructions
- Loop counter > 0: the loop body is always entered at least once

Simple vector addition

```
for (i = 0; i < 100; i++) {
  d[i] = a[i] + 1;
}
```

Baseline

```
        mv   x4, 0
        mv   x5, 100
Lstart: lw   x2, 0(x10)
        addi x10, x10, 4
        addi x2, x2, 1
        sw   x2, 0(x11)
        addi x11, x11, 4
        addi x4, x4, 1
        bne  x4, x5, Lstart
```

7 instr. + branch cost

with hardware loops

```
        lp.setupi 100, Lend
        lw   x2, 0(x10)
        addi x10, x10, 4
        addi x2, x2, 1
        sw   x2, 0(x11)
Lend:   addi x11, x11, 4
```

5 instr.

RI5CY Extension: Post-incrementing LD/ST

- Perform a load/store operation from/to the data memory while at the same time increasing the base address by the specified offset
- Reduce the number of required instructions to execute code with regular data access patterns (like loops)
- ALU and LSU have a separate RF WRITE PORT to support this ISA extension

Simple vector addition

```
for (i = 0; i < 100; i++) {
  d[i] = a[i] + 1;
}
```

with hardware loops

```
      lp.setupi 100, Lend
      lw   x2, 0(x10)
      addi x10, x10, 4
      addi x2, x2, 1
      sw   x2, 0(x11)
Lend: addi x11, x11, 4
```

5 instr.

with hardware loops & post-incr.

```
      lp.setupi 100, Lend
      lw   x2, 4(x10!)
      addi x2, x3, 1
Lend: sw   x2, 4(x11!)
```

3 instr.

References: Papers

- Davide Rossi, Francesco Conti, Andrea Marongiu, Antonio Pullini, Igor Loi, Michael Gautschi, Giuseppe Tagliavini, Philippe Flatresse, Luca Benini
  PULP: A Parallel Ultra-Low-Power Platform for Next Generation IoT Applications
  https://drive.google.com/file/d/0B08HOJaoGEHSQXd2UmpkU1lSVTQ

- Michael Gautschi, Pasquale Davide Schiavone, Andreas Traber, Igor Loi, Antonio Pullini, Davide Rossi, Eric Flamand, Frank K. Gürkaynak, Luca Benini
  A near-threshold RISC-V core with DSP extensions for scalable IoT Endpoint Devices
  https://arxiv.org/pdf/1608.08376.pdf

- Igor Loi, Davide Rossi, Germain Haugou, Michael Gautschi, Luca Benini
  Exploring Multi-banked Shared-L1 Program Cache on Ultra-Low Power Tightly Coupled Processor Clusters
  http://dl.acm.org/citation.cfm?id=2747288

- Davide Rossi, Igor Loi, Germain Haugou, Luca Benini
  Ultra-low-latency lightweight DMA for tightly coupled multi-core clusters
  https://drive.google.com/open?id=0B08HOJaoGEHSWUtINWxYYXkzZEk

- Michael Gautschi, Andreas Traber, Antonio Pullini, Luca Benini, Michele Scandale, Alessandro Di Federico, Michele Beretta, Giovanni Agosta
  Tailoring instruction-set extensions for an ultra-low power tightly-coupled cluster of OpenRISC cores
  https://drive.google.com/open?id=0B08HOJaoGEHSYV9SY0FDSnNjVnc

References: General

- http://iis-projects.ee.ethz.ch/index.php/PULP
- http://www.pulp-platform.org/
- http://home.deib.polimi.it/cesana/teaching/IoT/2017/classes/2.Hardware.pdf
- https://riscv.org/wp-content/uploads/2016/01/Wed1315-PULP-riscv3_noanim.pdf
- http://www.pulp-platform.org/wp-content/uploads/2017/02/ri5cy_user_manual.pdf
- https://www.dropbox.com/s/qs3jbqqiz0948tj/PULP_ORCONF15.pptx?dl=0
- http://www.hotchips.org/wp-content/uploads/hc_archives/hc27/HC27.24-Monday-Epub/HC27.24.10-IoT-Epub/HC27.24.111-PULP-Rossi-DEL-ETH-2.pdf