

PULP PLATFORM

Open Source Hardware, the way it should be!



PULP Platform: Open source RISC-V based energy efficient computing solutions



Frank K. Gürkaynak <kgf@iis.ee.ethz.ch>

Digital Circuits and Systems Group
ETH Zürich



<http://pulp-platform.org>



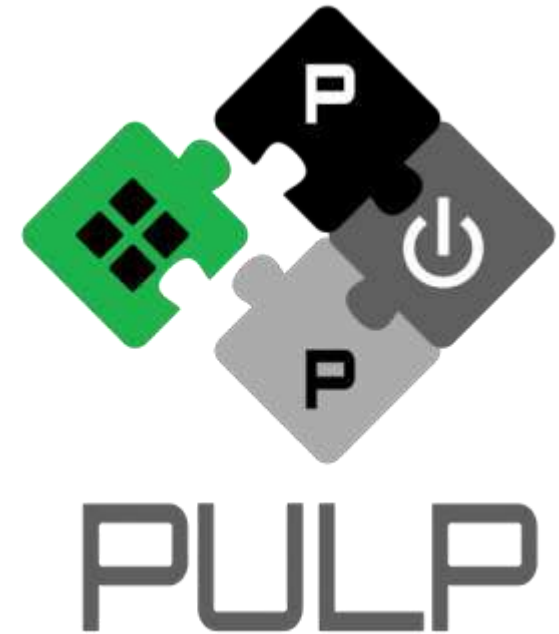
@pulp_platform

ETH zürich



How did we start in 2013?

- We wanted to design energy efficient computing systems
 - Equally efficient for IoT and HPC over a wide range
- Key points
 - Parallel processing
 - Near threshold computing
 - Efficient switching between operating modes
 - Making best use of technology
 - Heterogeneous acceleration
- Parallel Ultra Low-Power (**PULP**) platform was born





Who is behind PULP?



Frank



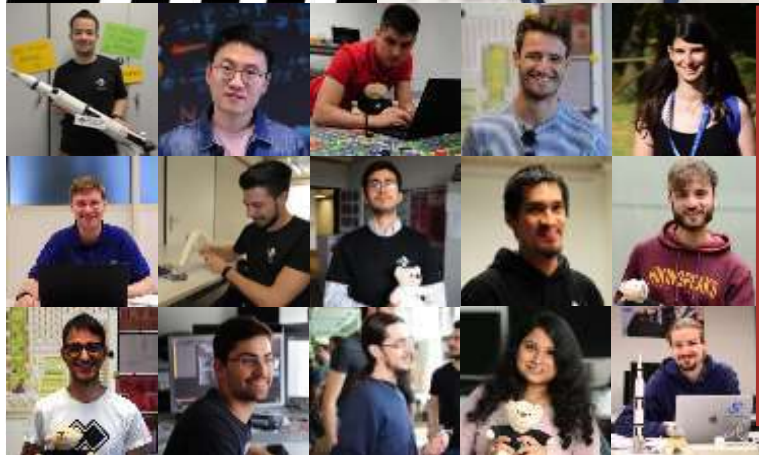
Prof. Luca Benini



Davide Rossi

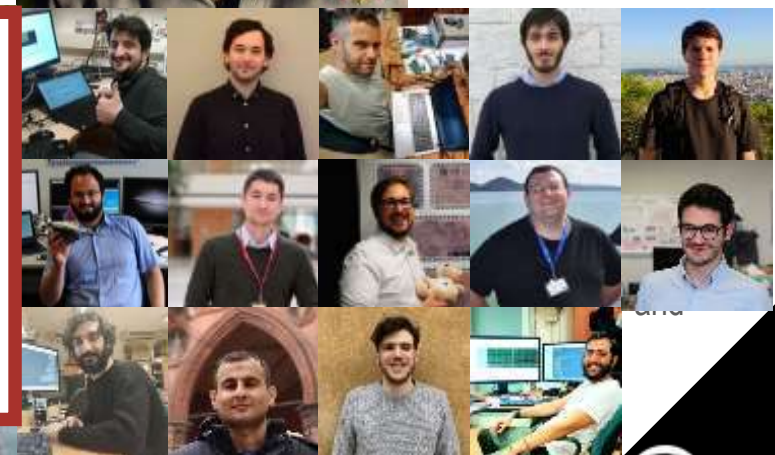


STUDIORUM DI BOLOGNA



In total about 60 people work on projects related to PULP in Zurich and Bologna

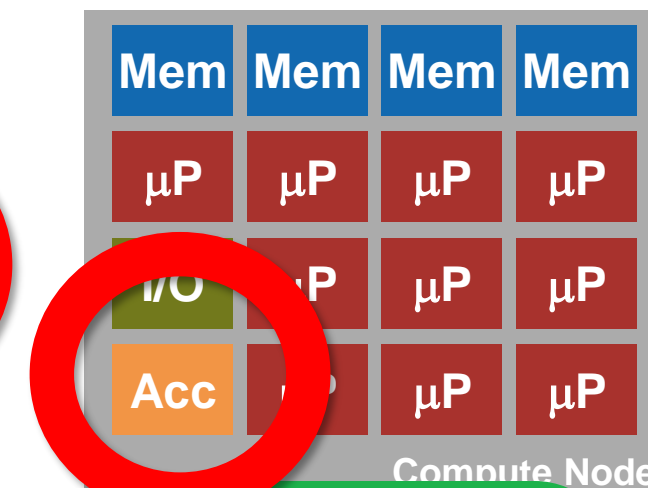
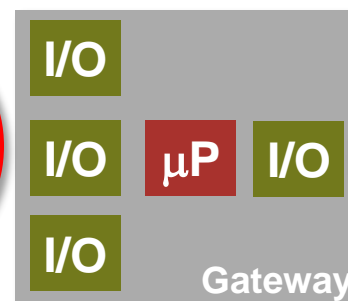
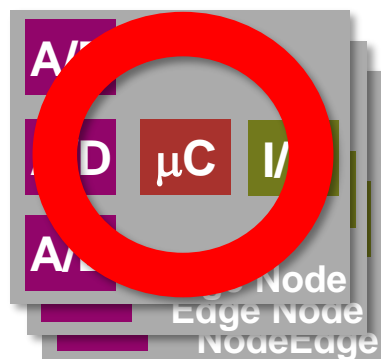
<https://pulp-platform.org/team.html>



Chief Architect in SEMI (2009-2012)

Too much to do and not enough resources

SIGNAL → **DATA** → **KNOWLEDGE**



Energy efficient systems that can process more on edge nodes



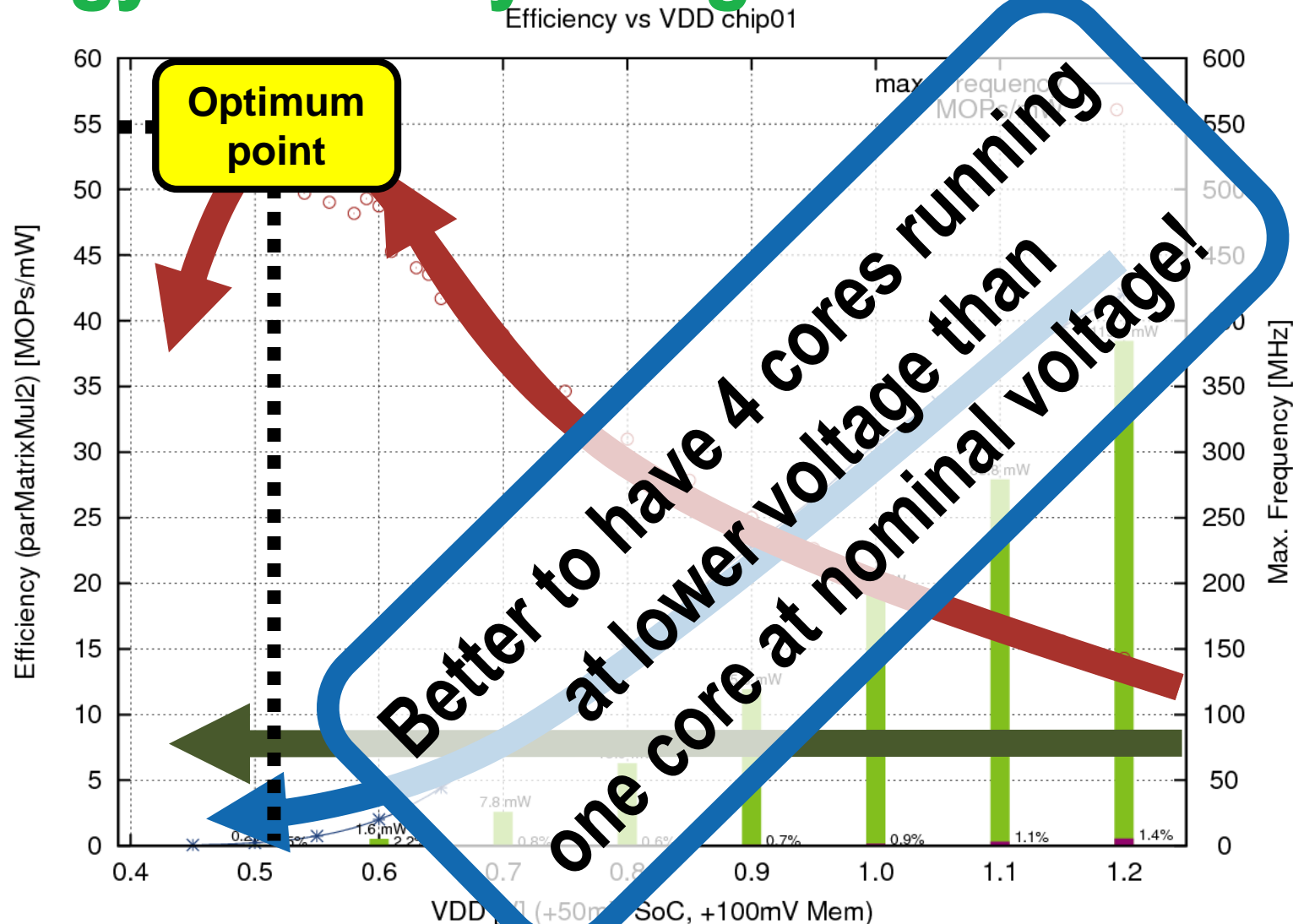
Less data to transmit, large savings on energy needed for data transmissions



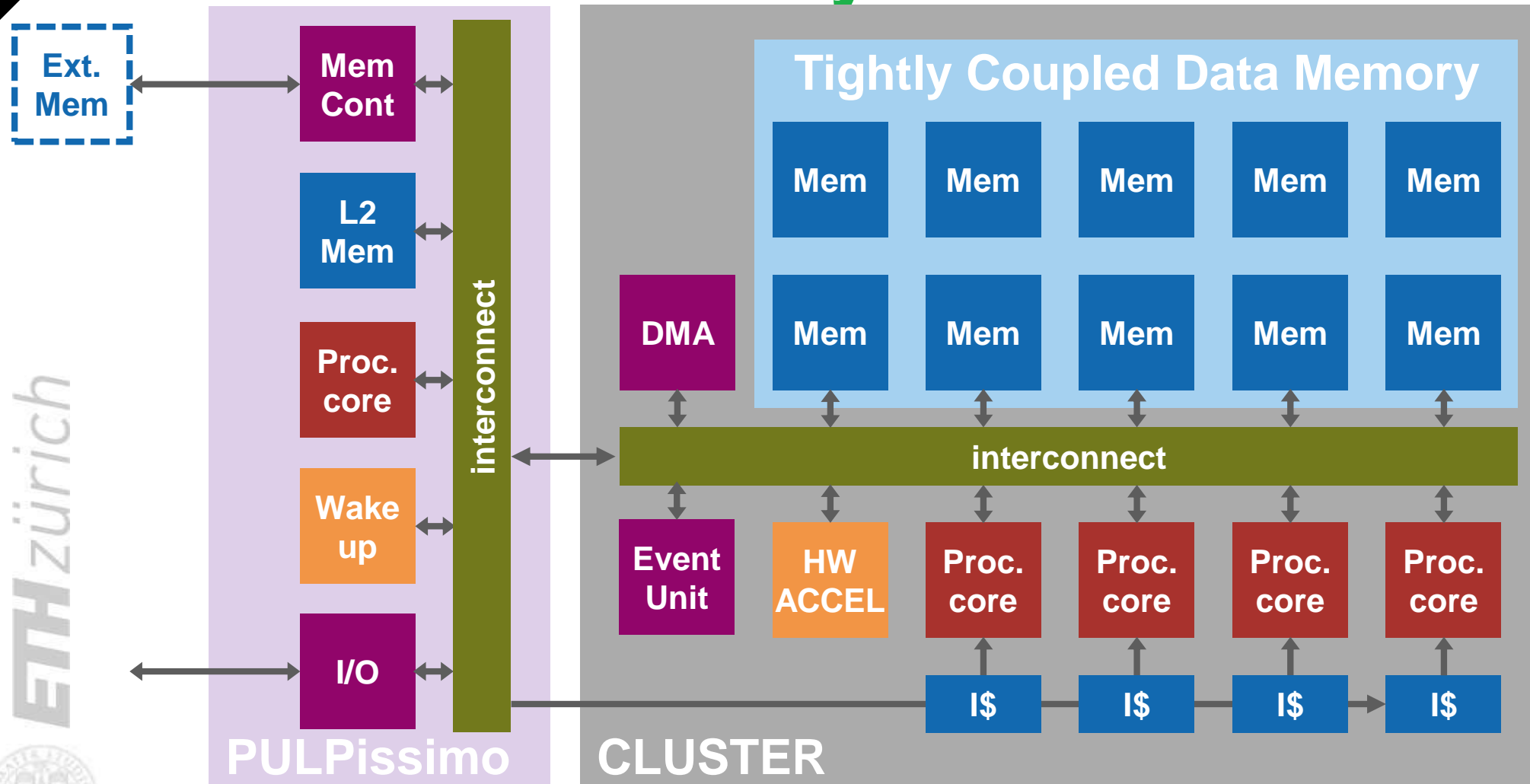
Accelerators for more efficient computing



Energy efficiency is higher at near threshold



Cluster based PULP systems





How PULP and RISC-V come together

- **Initially we did not want to design our own processors**
 - Wanted to use available processors (ARC, ARM..)
 - It proved difficult to design systems that we could share with our collaborators
- **Then we used OpenRISC cores (2013-2015)**
 - We had to completely redesign and optimize these cores
- **We moved to RISC-V starting in 2015**
 - Adapted the decoder of our optimized OpenRISC core
 - Make use of a growing SW development environment
 - ETH is one of the founding members of the RISC-V foundation





Our research is not implementing RISC-V cores

- **We develop efficient programmable architectures**
 - Processor cores of various capabilities are required for that
- **We need efficient implementations of cores for our research**
 - To produce relevant results, our cores have to perform **as good as other solutions**
 - We ended up spending quite an effort to make sure they perform really well
- **Processor core alone is not enough**
 - You need peripherals, interconnect solutions, programming support...
- **PULP Platform** provides us a playground for our research
 - And we share it as open source

RISC-V cores developed by PULP team

32 bit			64 bit
Low Cost Core	Core with DSP support	Core for Streaming	Linux capable Core
<ul style="list-style-type: none"> ■ IBEX <ul style="list-style-type: none"> ■ Zero-riscy ■ Micro-riscy ■ 2 options <ul style="list-style-type: none"> ■ RV32-ICM ■ RV32-CE <p><i>Maintained By LowRISC</i></p>	<ul style="list-style-type: none"> ■ CV32E40P <ul style="list-style-type: none"> ■ RV32-ICMXP ■ SIMD ■ Low Thops ■ Bit manipulation <p><i>Maintained By OpenHW</i></p>	<ul style="list-style-type: none"> ■ Snitch <ul style="list-style-type: none"> ■ RV32-IMAFD ■ Small int core ■ Signed 64bit FPU ■ Extensions for <p><i>Brand New Core</i></p>	<ul style="list-style-type: none"> ■ Ariane/CVA6 <ul style="list-style-type: none"> ■ RV64-ICMAFD <p><i>Maintained by OpenHW</i></p>

xPULP ISA extensions for performance

```
for (i = 0; i < 100; i++)
    d[i] = a[i] + b[i];
```

Baseline

```
mv    x5, 0
mv    x4, 100
Lstart:
    lb    x2, 0(x10)
    lb    x3, 0(x11)
    addi  x10, x10, 1
    addi  x11, x11, 1
    add   x2, x3, x2
    sb    x2, 0(x12)
    addi  x4, x4, -1
    addi  x12, x12, 1
    bne   x4, x5, Lstart
```

11 cycles/output

Auto-incr load/store HW Loop

```
mv    x5, 0
mv    x4, 100
Lstart:
    lb    x2, 0(x10!)
    lb    x3, 0(x11!)
    addi  x4, x4, -1
    add   x2, x3, x2
    sb    x2, 0(x12!)
    bne   x4, x5, Lstart
```

8 cycles/output

lp.setupi 100, Lend

```
    lb    x2, 0(x10!)
    lb    x3, 0(x11!)
    add   x2, x3, x2
Lend:    sb x2, 0(x12!)
```

5 cycles/output

Packed-SIMD

```
lp.setupi 25, Lend
    lw    x2, 0(x10!)
    lw    x3, 0(x11!)
    pv.add.b x2, x3, x2
Lend: sw x2, 0(x12!)
```

1,25 cycles/output



The extensions translate to real speed-ups

- **8-bit convolution**

- Open source DNN library

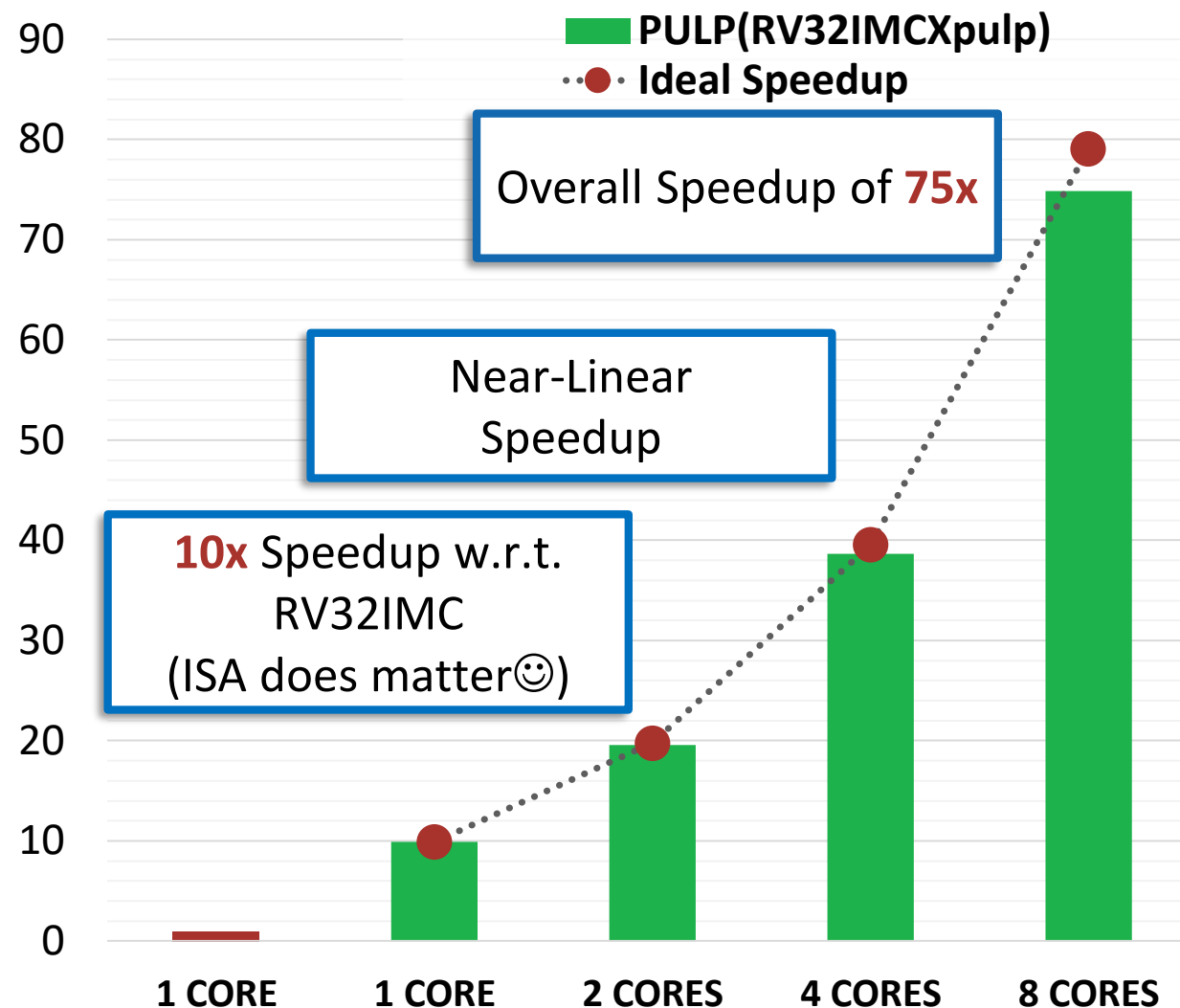
- **10x** through xPULP

- Extensions bring real speedup

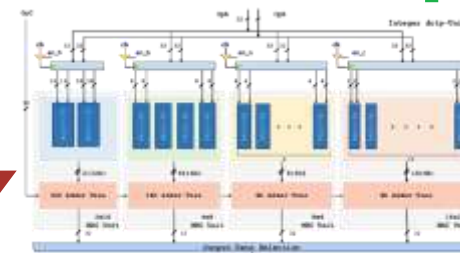
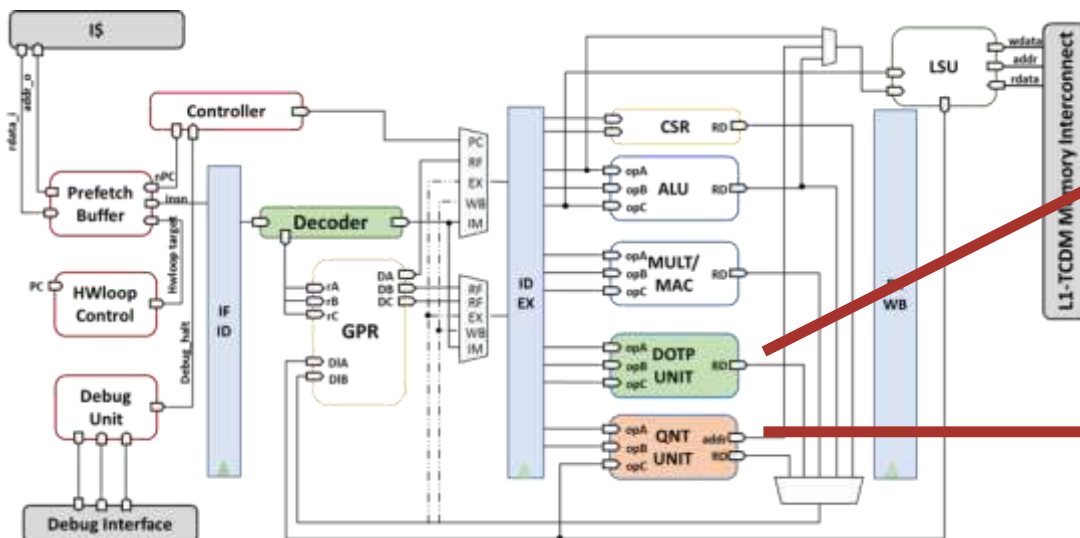
- **Near-linear speedup**

- Scales well for regular workloads.

- **75x** overall gain



RISC-V ISA Extensions for extreme quantization



2-bit & 4-bit SIMD DOTP + OP Isolation



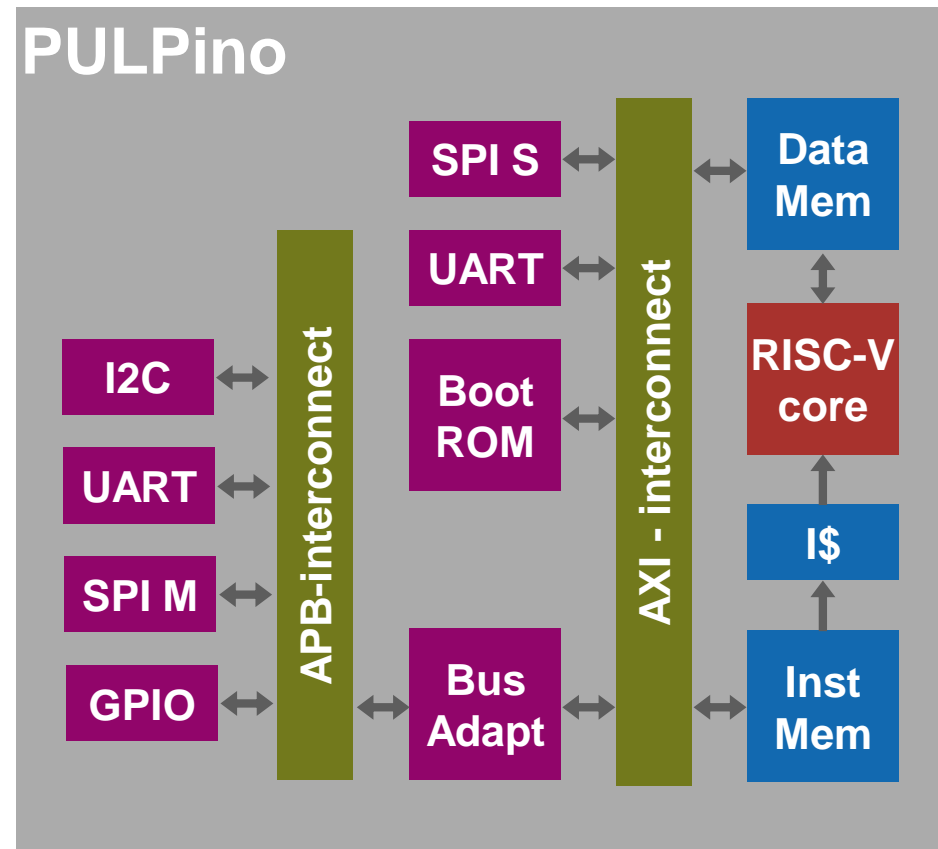
QNT UNIT: 2 Quantizations in 9 Cycles

■ Overheads (28nm FDX PULPissimo impl.):

- Area: ~11% (vs. Ri5CY)
- Timing Overhead: negligible (integrated in PULPissimo)
- 8-bit MatMul power overhead: 1.8% (integrated in PULPissimo)
- GP-app power overhead: 3.5% (integrated in PULPissimo)

PULPino: Our first single core platform

- **Simple design**
 - Meant as a quick release
- **Separate data and inst. mem**
 - Makes it easy in HW
 - Not meant as a Harvard arch.
- **Can use all our 32bit cores**
 - RI5CY, Zero/Micro-Riscy (Ibex)
- **Peripherals from other projects**
 - Any AXI and APB peripherals could be used



PULP has released a large number of IPs

RISC-V Cores

RI5CY	Ibex	Snitch	Ariane + Ara
32b	32b	32b	64b

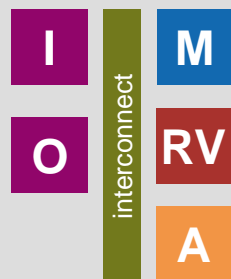
Peripherals

JTAG	SPI
UART	I2S
DMA	GPIO

Interconnect

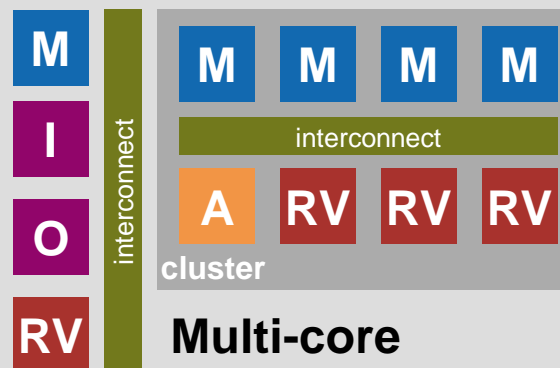
Logarithmic interconnect
APB – Peripheral Bus
AXI4 – Interconnect

Platforms



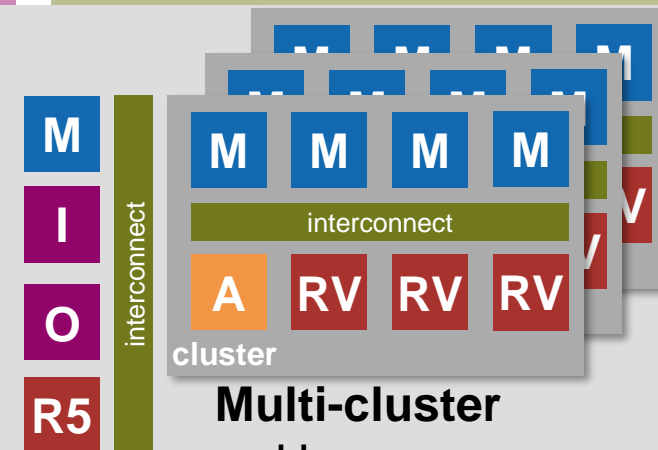
Single Core

- PULPino
- PULPissimo



Multi-core

- OpenPulp
- Mr. Wolf



Multi-cluster

- Hero
- Snitch

IOT

HPC

Accelerators

HWCE
(convolution)

Neurostream
(ML)

HWCrypt
(crypto)

PULPO
(1st ord. opt)





Why Open Source Hardware

- It is a **necessity**

- We can not afford to make everything ourselves, we need to collaborate
- Makes it possible to work together quickly
- Your results are more trustworthy, anybody can verify it!

- It **works**

- We have actually more projects, and more funding due to open source activities
- We were able to start many interesting and fruitful collaborations

- It **helps others as well**

- Many companies, universities, individuals are using pieces of PULP, even commercially

PULP uses a permissive open source license

- All our development is on GitHub

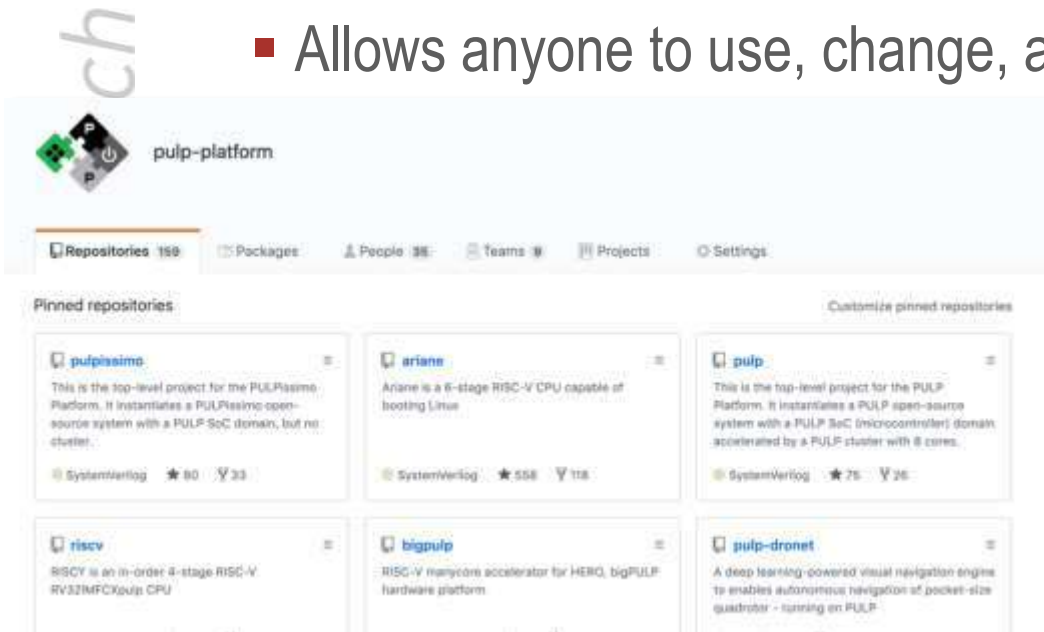
- HDL source code, testbenches, software development kit, virtual platform

<https://github.com/pulp-platform>



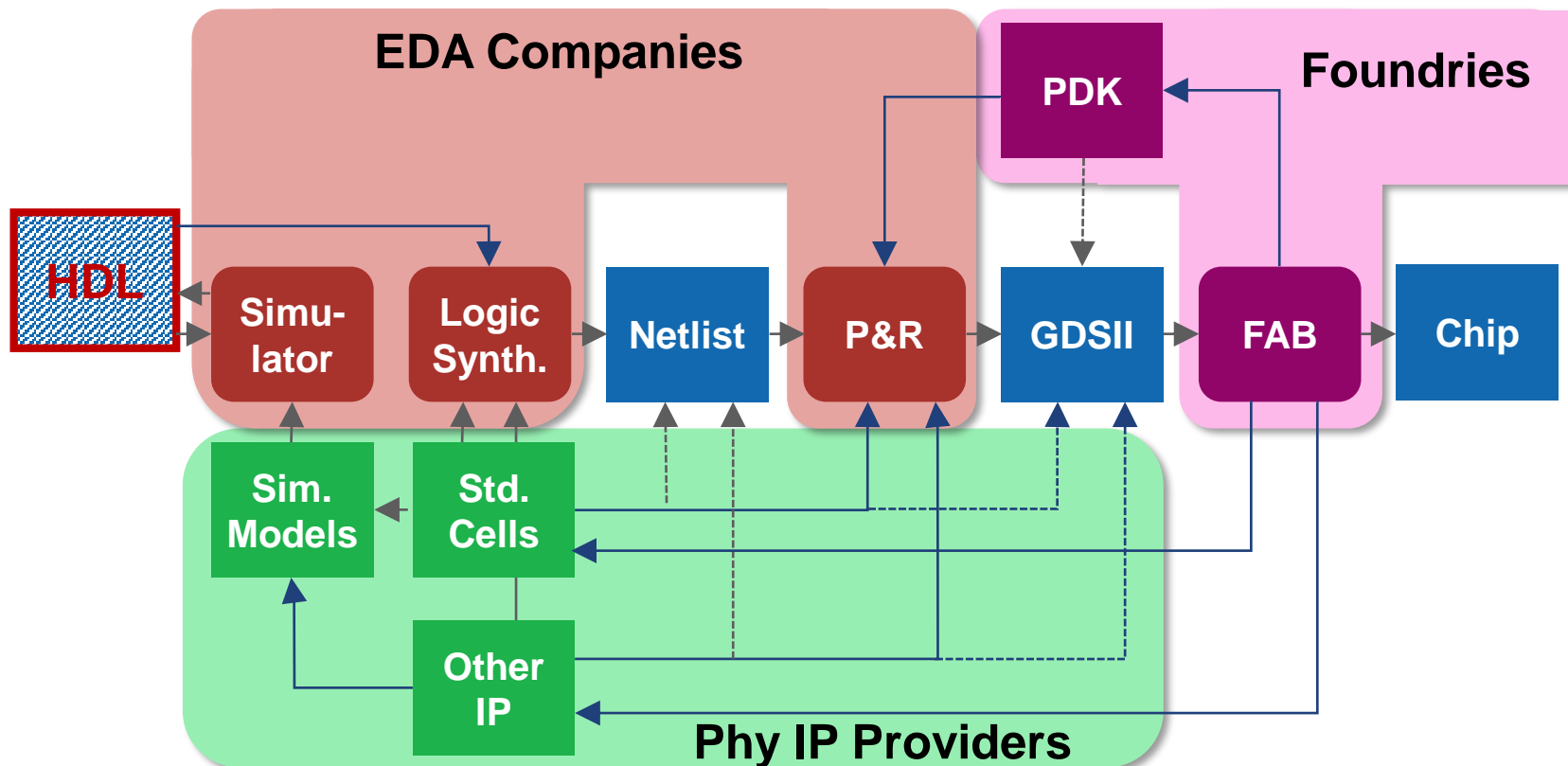
- PULP is released under the permissive Solderpad license

- Allows anyone to use, change, and make products without restrictions.



Nice, but what exactly is “open” in OSCHW?

- Only the first stage of the silicon production pipeline
→ **RTL source code** (*permissive**, e.g. Apache is key for industrial adoption)
- Later stages contain closed IP of various actors → not open source by default












“See: <https://cern-ohl.web.cern.ch/> (CERN-OHL-S, -W, -P)







Open source collaboration scheme explained





Direct research collaborators on PULP





Politecnico di Torino	
University of Cambridge	
USI Lugano	
TU Kaiserslautern	
University of Cagliari	





IBM Research Zurich	
EPF Lausanne	
CSEM Neuchatel	
Princeton University	





Technische Universität Graz	
CEA-Leti Grenoble	
Fraunhofer-Gesellschaft	
Sapienza Università di Roma	





Academic users we are aware of





Università di Genova	
Politecnico di Milano	
Fondazione Bruno Kessler	
Lund University	



Stanford University	
UC Los Angeles	
UC San Diego	
Columbia University	


Universitat Bar-Ilan	
İstanbul Teknik Üniversitesi	
NCTU Hsinchu	
University of Zagreb, FER	


TUT Tampere	
RWTH Aachen	
IST University of Lisboa	
UFRN Rio Grande do Norte	

TU Darmstadt	
Universität Bremen	
Hongik University Seoul	
IIT Kharagpur	

LIRMM Montpellier	
University of Stuttgart	
University of Tübingen	
TU München	

FORTH Hellas	
Kyoto University	

Chalmers Göteborg	
FAU Erlangen-Nürnberg	

NTNU Trondheim	
----------------	---------------------------------------------------------------------------------------



commits



GitHub

ETH zürich



Open source collaboration scheme explained

Industrial users of PULP

Custom
IP



What is PULP doing to maintain our cores?

- **We (ETHZ and University of Bologna) are research groups**
 - Motivated to develop new architectures and systems
 - We needed efficient RISC-V cores (and peripherals) for our work
 - Not so good (or interested) in providing industrial level support for these cores
- **We need help to**



Academic open source → Industrial open source



OPENHW GROUP™
— PROVEN PROCESSOR IP —

Rick O'Connor (OpenHW CEO, former RISC-V foundation director)

- **OpenHW Group** is a not-for-profit, global organization (EU,NA,Asia) driven by its members and individual contributors where HW and SW designers collaborate in the development of open-source cores, related IP, tools and SW such as the **Core-V** family of cores.
- **OpenHW Group** provides an infrastructure for hosting high quality open-source HW developments in line with industry best practices.



RI5CY, ARIANE



OpenHW Members include

- Amazon
- Thales
- Huawei
- NXP
- Globalfoundries
- CSEM
- EM-Marin

A vertical, application-focused open approach



- OpenTitan is **the first open source** silicon project building a transparent, high-quality reference design for silicon root of trust (RoT) chips.
- Founding Partners

ETH zürich

GD Giesecke+Devrient
Creating Confidence

Google

lowRISC

nuvoTon

Western Digital.



Feel the momentum!

Ibex RISC-V core, flash interface, communications ports, cryptography accelerators, and more.

Vibrant repository

35+

Contributors

1300

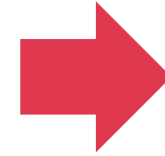
Contributions

470

GitHub Issues



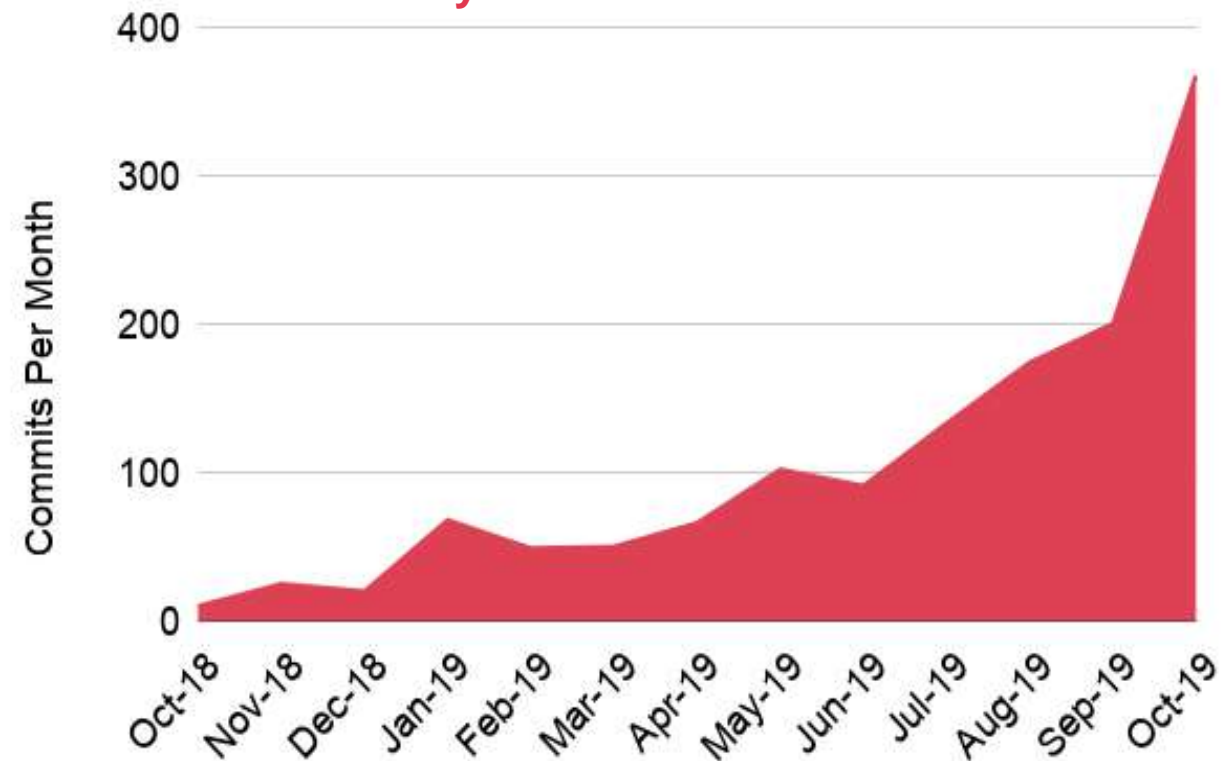
PULP
Parallel Ultra Low Power



lowRISC

Zero-Riscy

Ibex



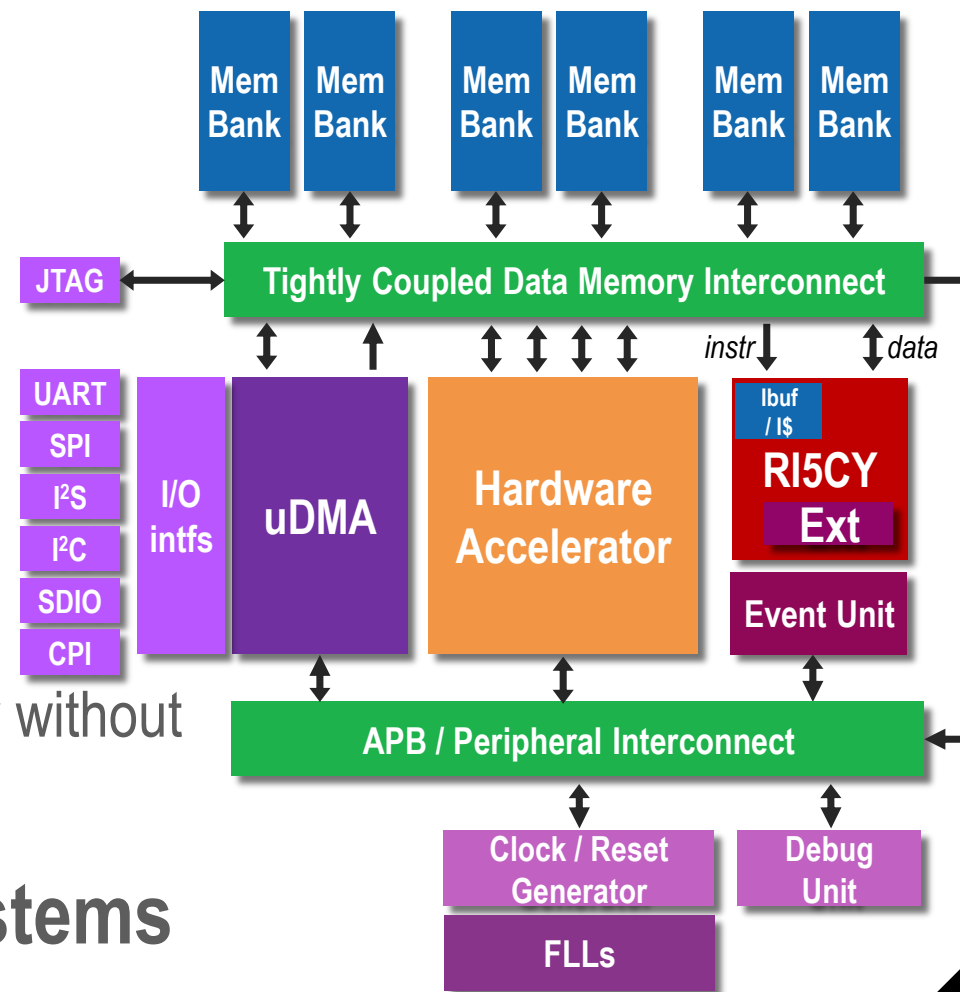


What kind of acceleration?

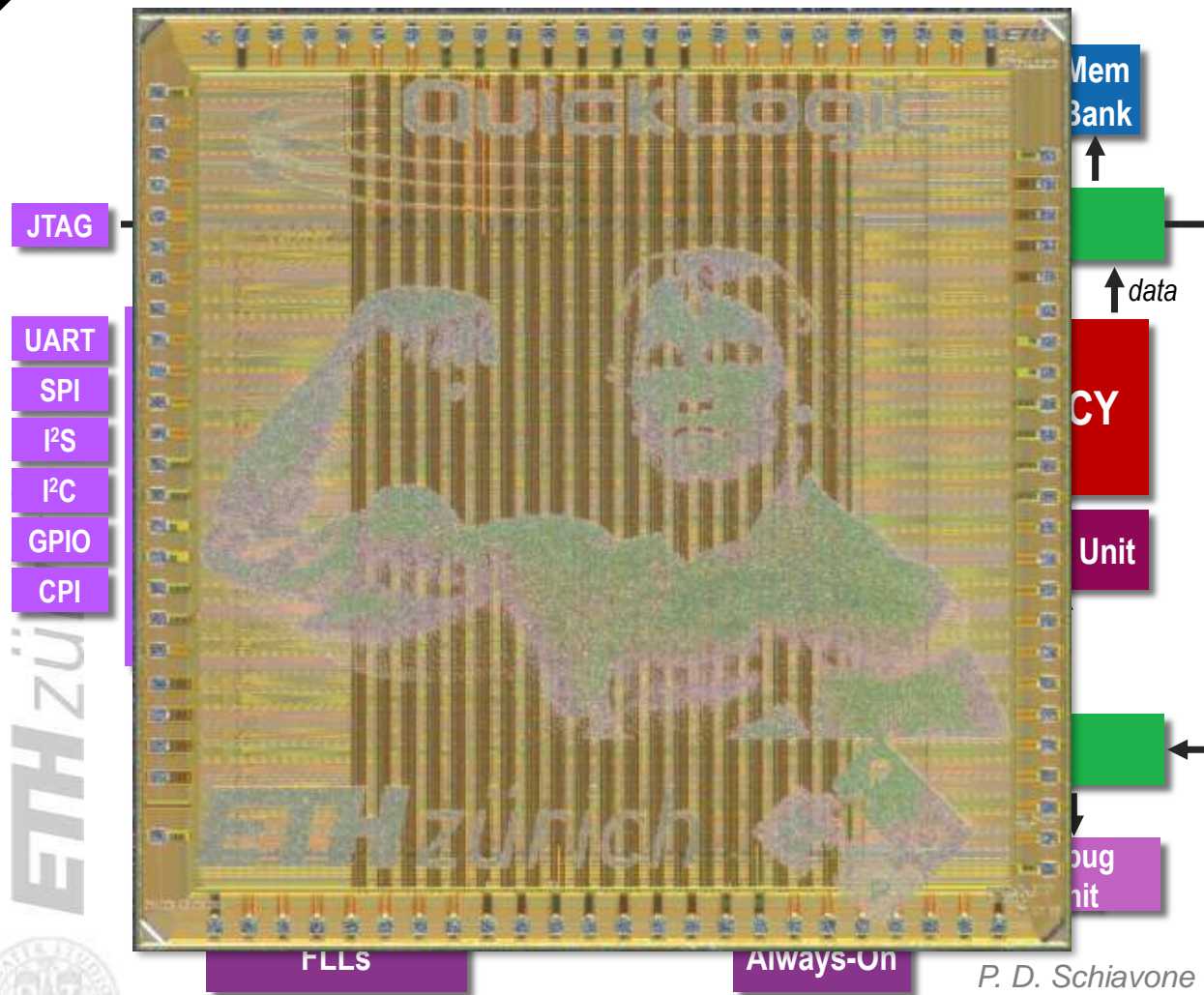
- *Standard peripheral talking over AXI/APB (standard)*
- *Instruction set extensions (already discussed)*
- **Shared functional units**
 - Amortizes expensive extensions (FPU/DIV) between multiple units
- **Shared memory accelerators**
 - Our bread and butter, PULPopen, NTX
- **Cluster as an accelerator**
 - HERO, BigPULP, Snitch, Mempool etc

PULPissimo: The better single core platform

- **Shared memory**
 - Unified Data/Instruction Memory
- **Support for Accelerators**
 - Direct shared memory access
 - Programmed through APB bus
- **uDMA for I/O subsystem**
 - Can copy data directly from I/O to memory without involving the core
- **Used as controller in larger systems**



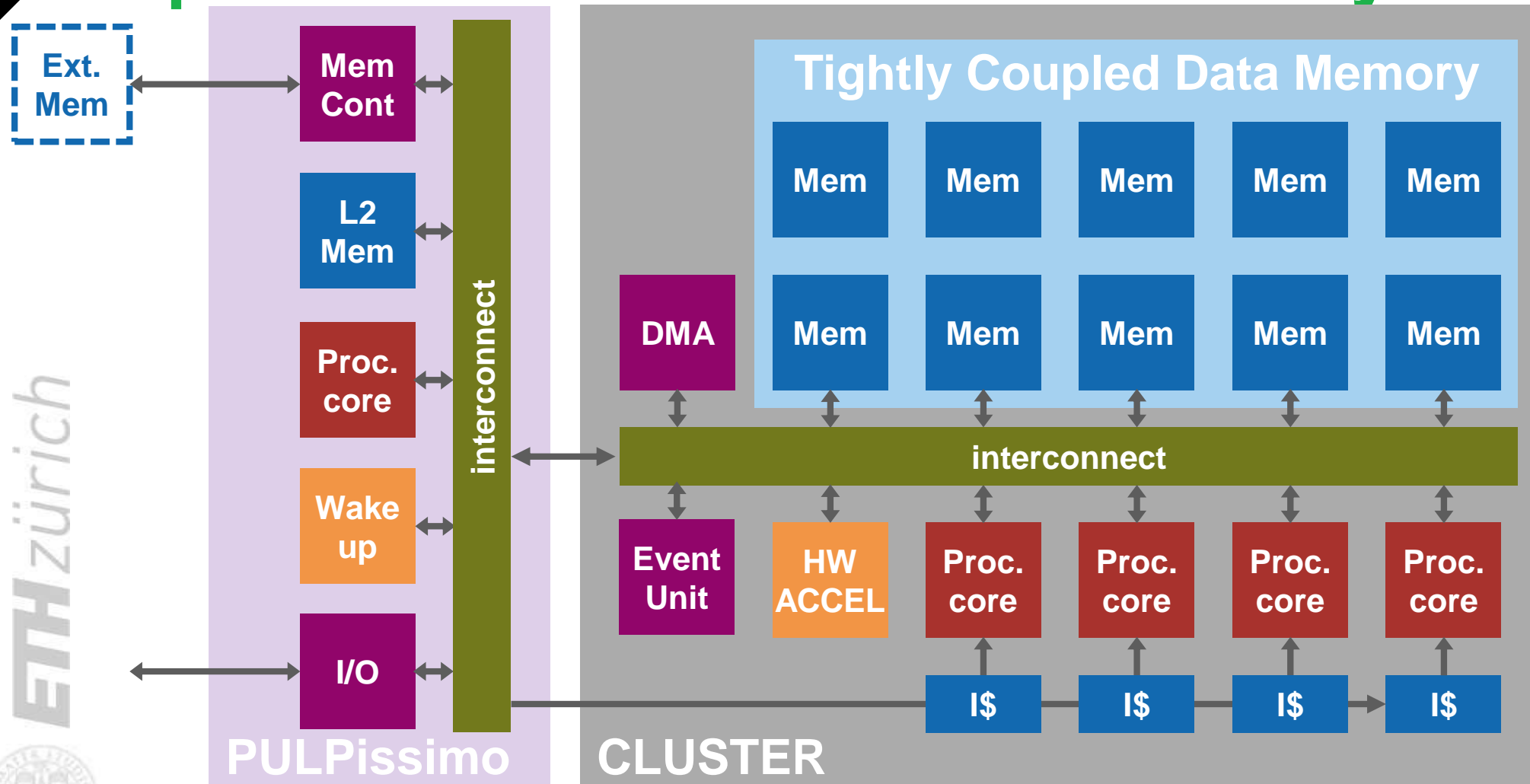
Arnold: a Collaboration with Quicklogic



- **Chip in 22nm FDX**
 - Combines e-FPGA (Quicklogic)
 - with PULPissimo (single core uC)
- **Multiple operation modes**
 - Configurable peripheral
 - Accelerator for core
 - Accelerator for independent I/O

P. D. Schiavone et al., "Arnold: An eFPGA-Augmented RISC-V SoC for Flexible and Low-Power IoT End Nodes," 10.1109/TVLSI.2021.3058162.

Open PULP: our main cluster based system



Deploying DNNs on PULP

[Burrello et al. TCOMP21]

QuantLab

Quantization Laboratory

NEMO

NEural Minimization for pyTorch

DORY

Deployment Oriented to memoRY

PULP-NN

PULP Neural Network backend



ONNX



PyTorch



specification & dataset selection

training

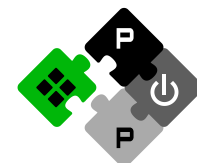
quantization & pruning

graph optimization

memory-aware deployment

optimized DNN library

github.com/pulp-platform/nemo, [/dory](https://github.com/pulp-platform/dory), [/pulp-nn](https://github.com/pulp-platform/pulp-nn)

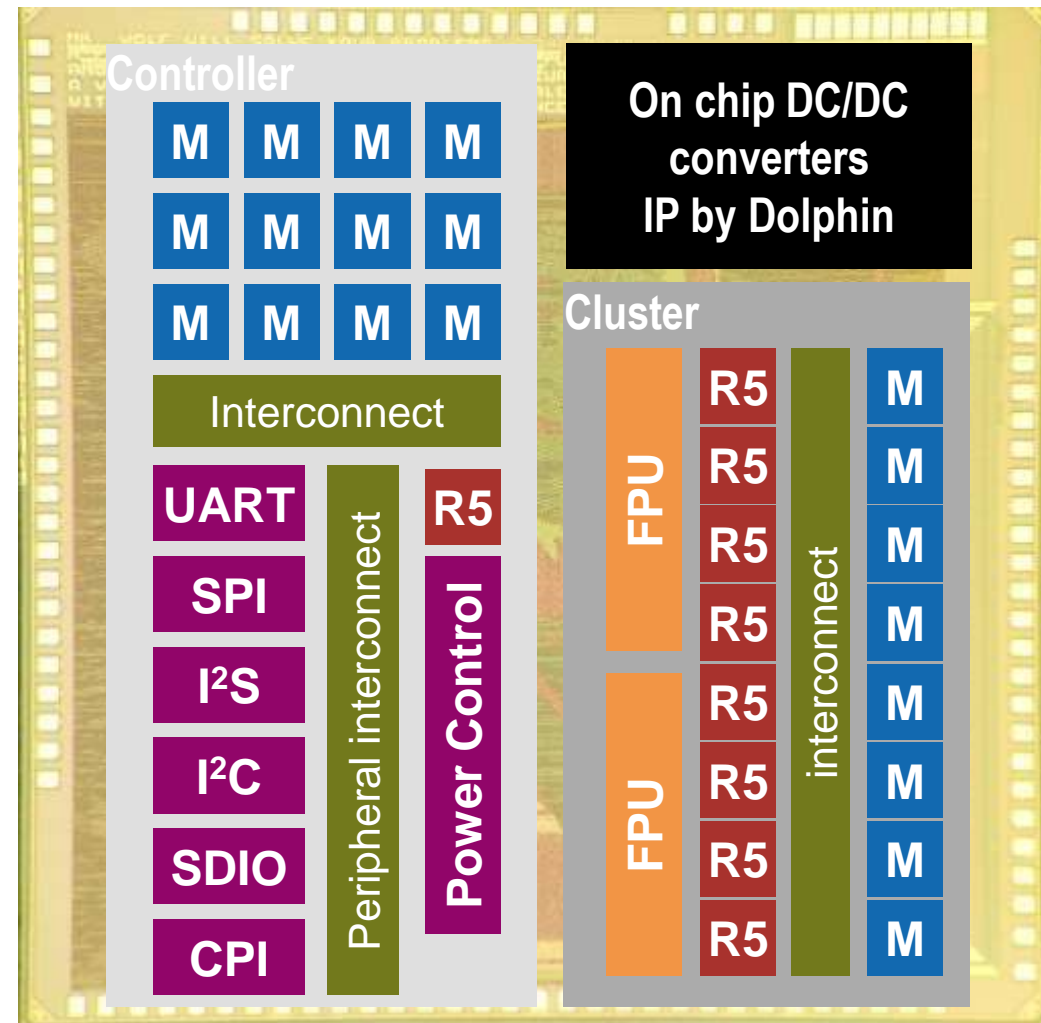


HW



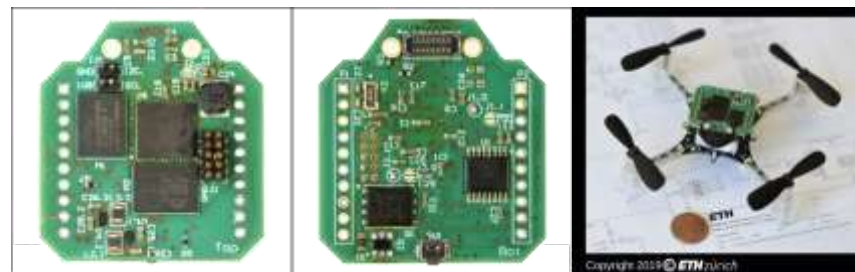
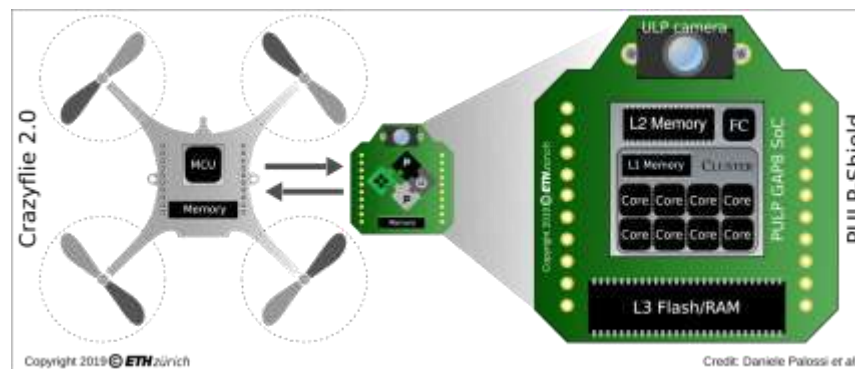
Mr. Wolf (TSMC 40): 8+1 core IoT Processor

- **One cluster with**
 - 8 RISC-V cores
 - 2x shared FPU units
 - 64 kByte of TCDM
- **One controller with**
 - 512 kByte L2 RAM
 - Peripherals
- **On chip voltage regulators**
 - By Dolphin Integration





AI-deck



- 
- GREENWAVES
TECHNOLOGIES



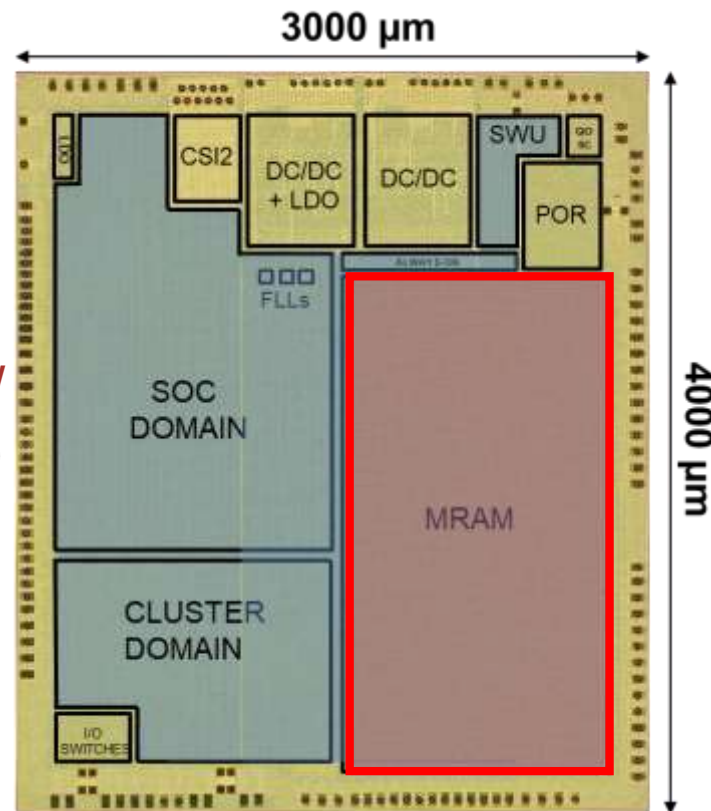
- ~ 8 g – 40x28 mm
- PULP GAP8 SoC
- Off-chip DRAM/Flash
- QVGA ULP Camera
- WiFi module

[1] F. Conti, D. Palossi, A. Marongiu, D. Rossi, and L. Benini. "Enabling the heterogeneous accelerator model on ultra-low power microcontroller platforms." *IEEE DATE*, 2016.

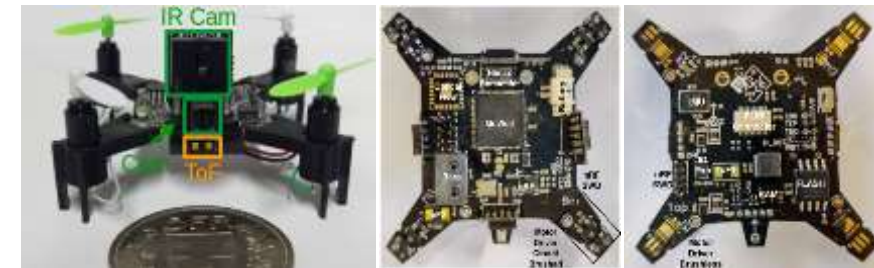
[2] D. Palossi, F. Conti, and L. Benini "An open source and open hardware deep learning-powered visual navigation engine for autonomous nano-UAVs." IEEE DCSS, 2019.

New Generation: Vega & Fünfliber

- 22nm FDX, 1.7MB SRAM, f_{\max} 450MHz, power: 1.7uW-49.4mW
- RISC-V cluster (8cores +1) 614GOPS/W @ 7.6GOPS (8bit DNNs), 79GFLOPS/W @ 1GFLOP (32bit FP appl)
- Multi-precision HWCE(4b/8b/16b) 3x3x3 MACs with normalization / activation: 32.2GOPS and 1.3TOPS/W (8bit)
- Fully-on chip DNN inference with 4MB MRAM**



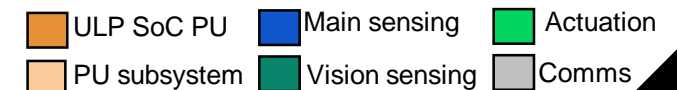
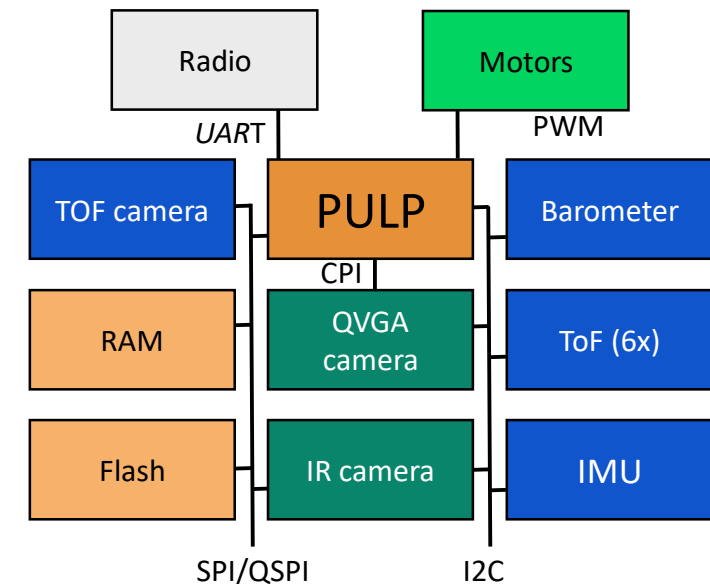
In cooperation with **GREENWAVES TECHNOLOGIES**



Front

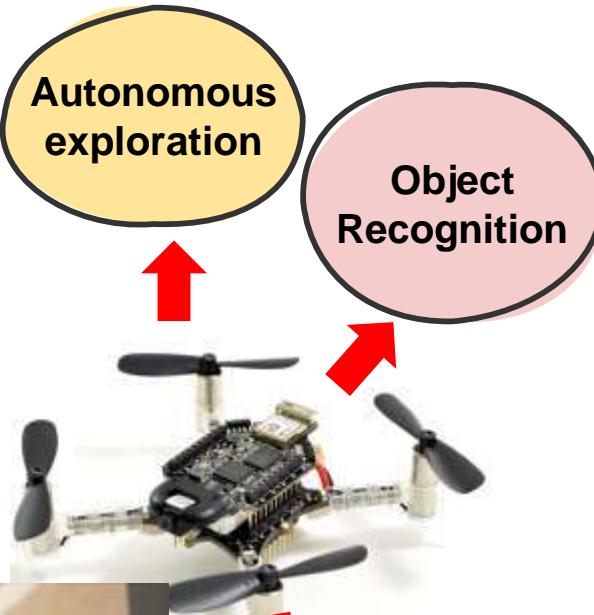
Top

Bottom

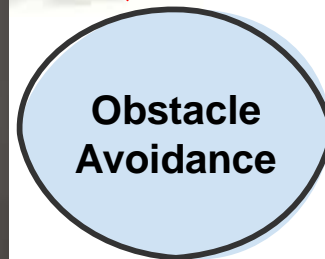




Multi-tasking autonomous nano-drone



ETH zürich



Rescue reconnaissance



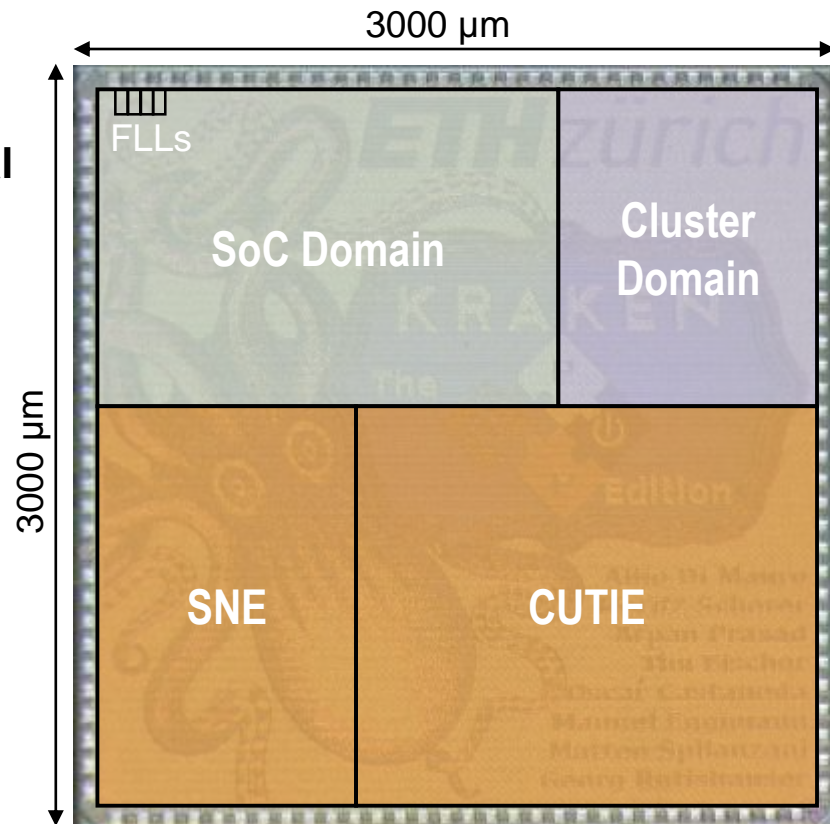
Safe indoor human-drone interaction



What's next? Heterogeneous Accelerators

The *Kraken*: TCNs and SNNs at The Extreme Edge

- RISC-V Cluster (8 Cores + 1)
- **CUTIE** – dense ternary neural network accelerator
- **SNE** – energy-proportional spiking neural network accelerator
- DVS Interface for hardware support of event-based vision
- PULPO – Floating point linear algebra accelerator



Technology	22 nm FDSOI
Chip Area	9 mm ²
SRAM SoC	1 MB
SRAM Cluster	128 KB
VDD range	0.55 V - 0.8 V
Cluster Freq	~370MHz
SNE Freq	~250MHz
CUTIE Freq	~140MHz

HW acceleration in perspective

- 22FDX, NT@0.6V, High utilization, Minimal IO & overhead

Energy-Efficient RV Core → 20pJ (8bit)



ISA-based 10-20x → 1-5pJ (8bit)



XPULPV2 & **V3**



Configurable DP 10-20x → 20-100fJ (4bit)



HWCE, RBE, **NE**



Fully specialized DP 10-20x → 1-5fJ (ternary)



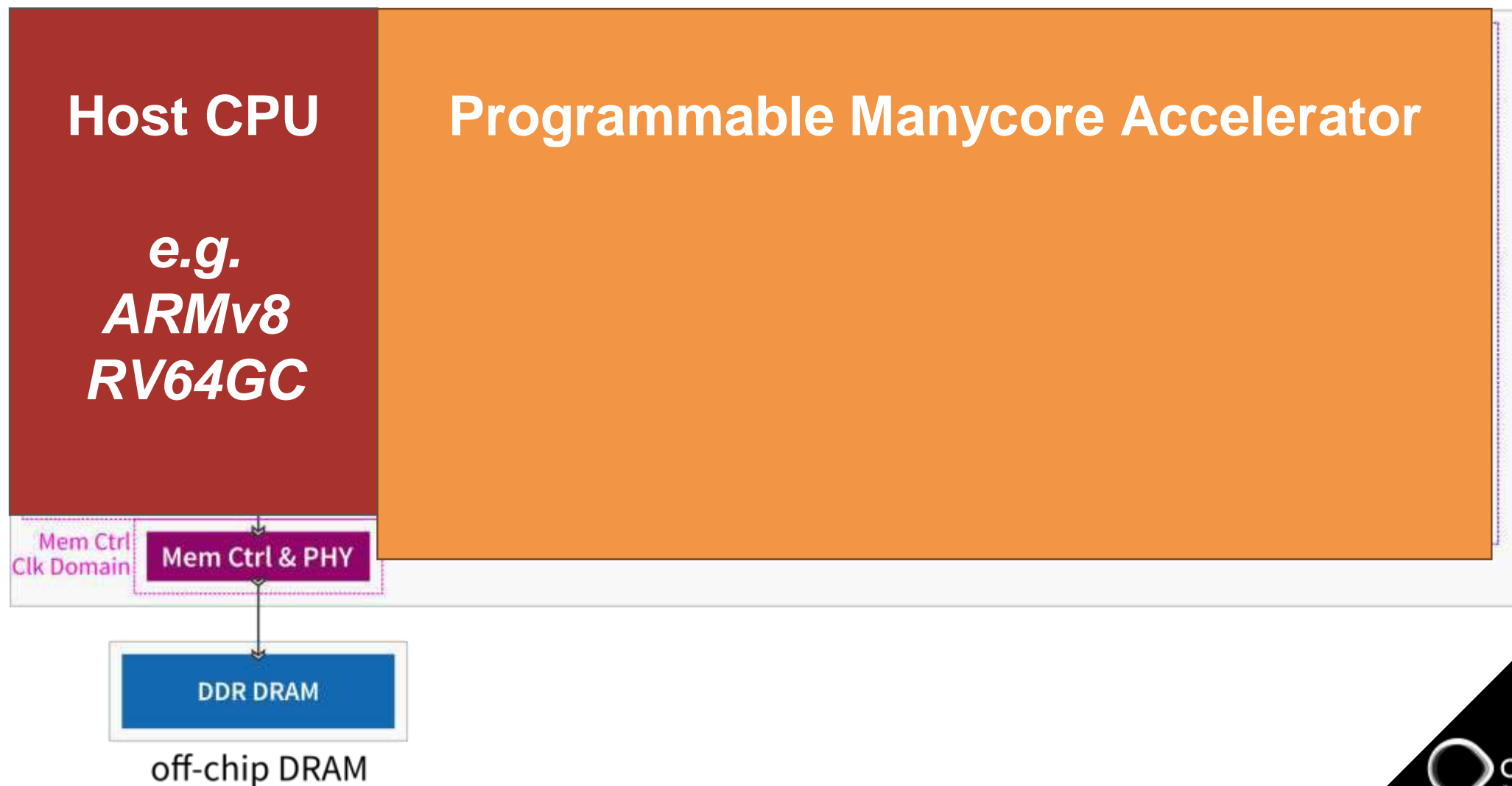
XNE, **CUTIE***

*sub 1fJ in 7nm



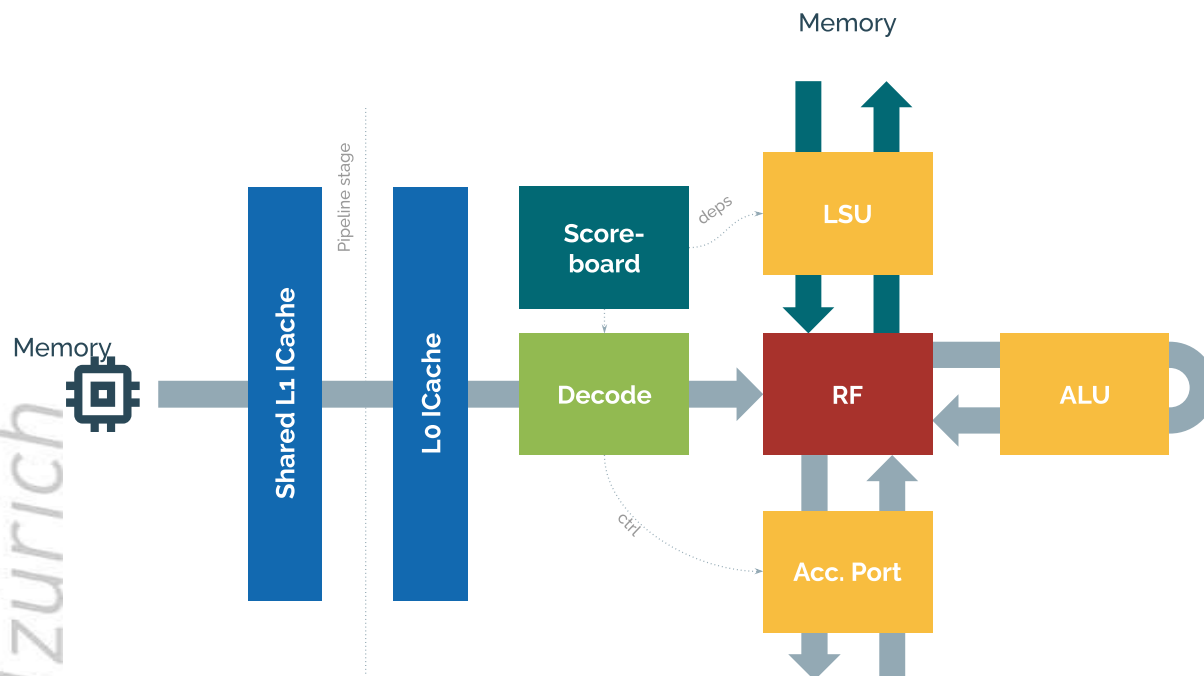
Cluster as heterogenous accelerator (HERO)

HEROv3



Snitch: Tiny Control Core

A versatile building block.



- Simplest core
- Focus on key features:
 - Lightweight microarchitecture
 - **Extensibility:** Performance through ISA extensions
 - Latency tolerant
 - Competitive frequency
- Around 15-25 kGE



Stream Semantic Registers

LD/ST elision

- Intuition: High FPU utilization \approx high energy-efficiency
- Idea: Turn register R/W into memory loads/stores.
- Extension around the core's register file
- Address generation hardware

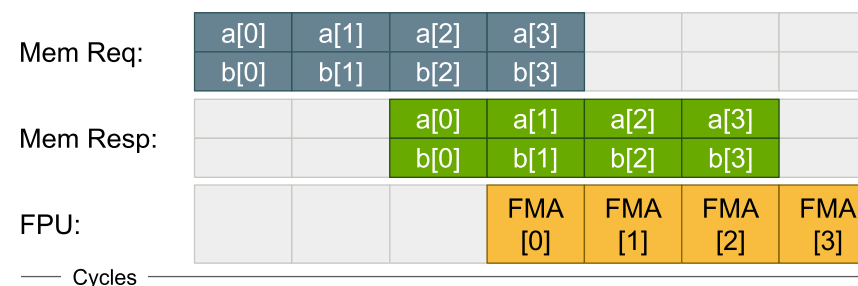
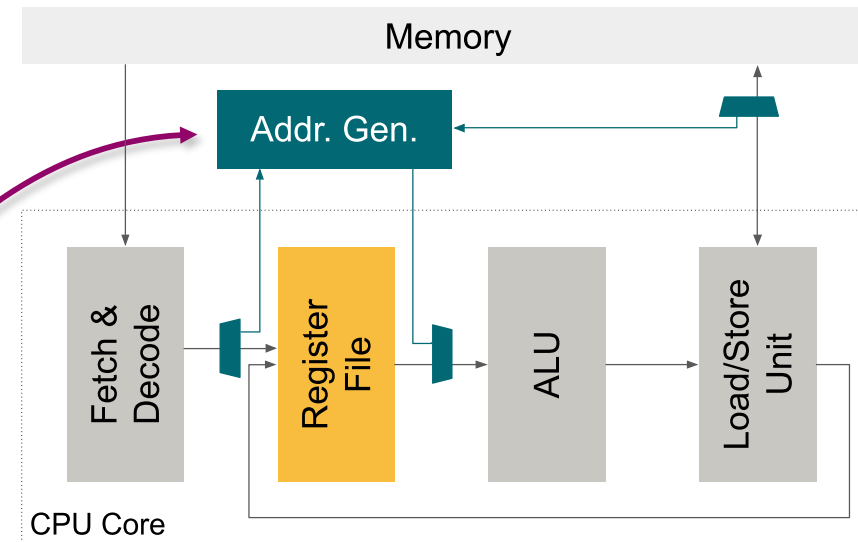
```

loop:
fld r0, %[a]      →   scfg 0, %[a], ldA
fld r1, %[b]      →   scfg 1, %[b], ldB
fmadd r2, r0, r1   →   loop:
                    fmadd r2, ssr0, ssr1
  
```

- Increase FPU/ALU utilization by $\sim 3x$ up to 100%

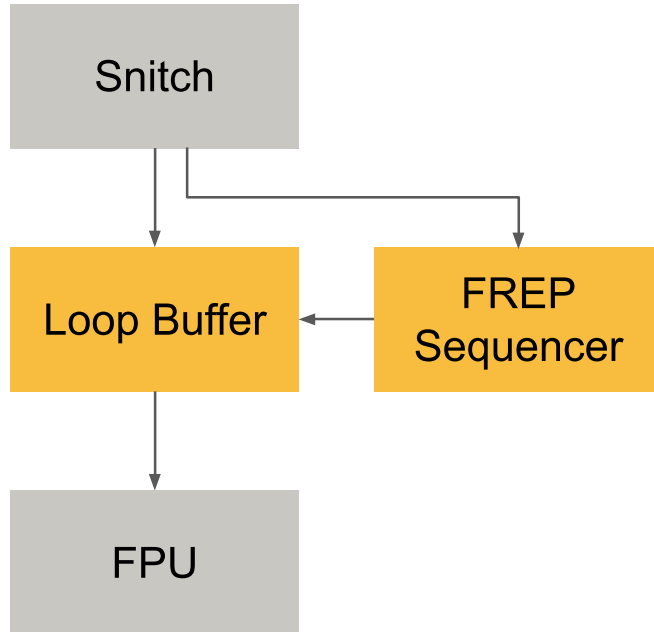
- SSRs \neq memory operands

- Perfect prefetching, latency-tolerant



Floating-point Repetition Buffer

Remove control flow overhead in compute stream



```

mv    r0, zero
loop:
  addi r0, 1
  fmadd r2, ssr0, ssr1
  bne  r0, r1, loop
  
```



```

frep  r1, 1
loop:
  fmadd r2, ssr0, ssr1
  
```

- Programmable micro-loop buffer
- Sequencer steps through the buffer, independently of the FPU
- Integer core free to operate in parallel: *Pseudo-dual issue*
- High area- and energy-efficiency

Efficient Data Mover

hide L2, main memory latency



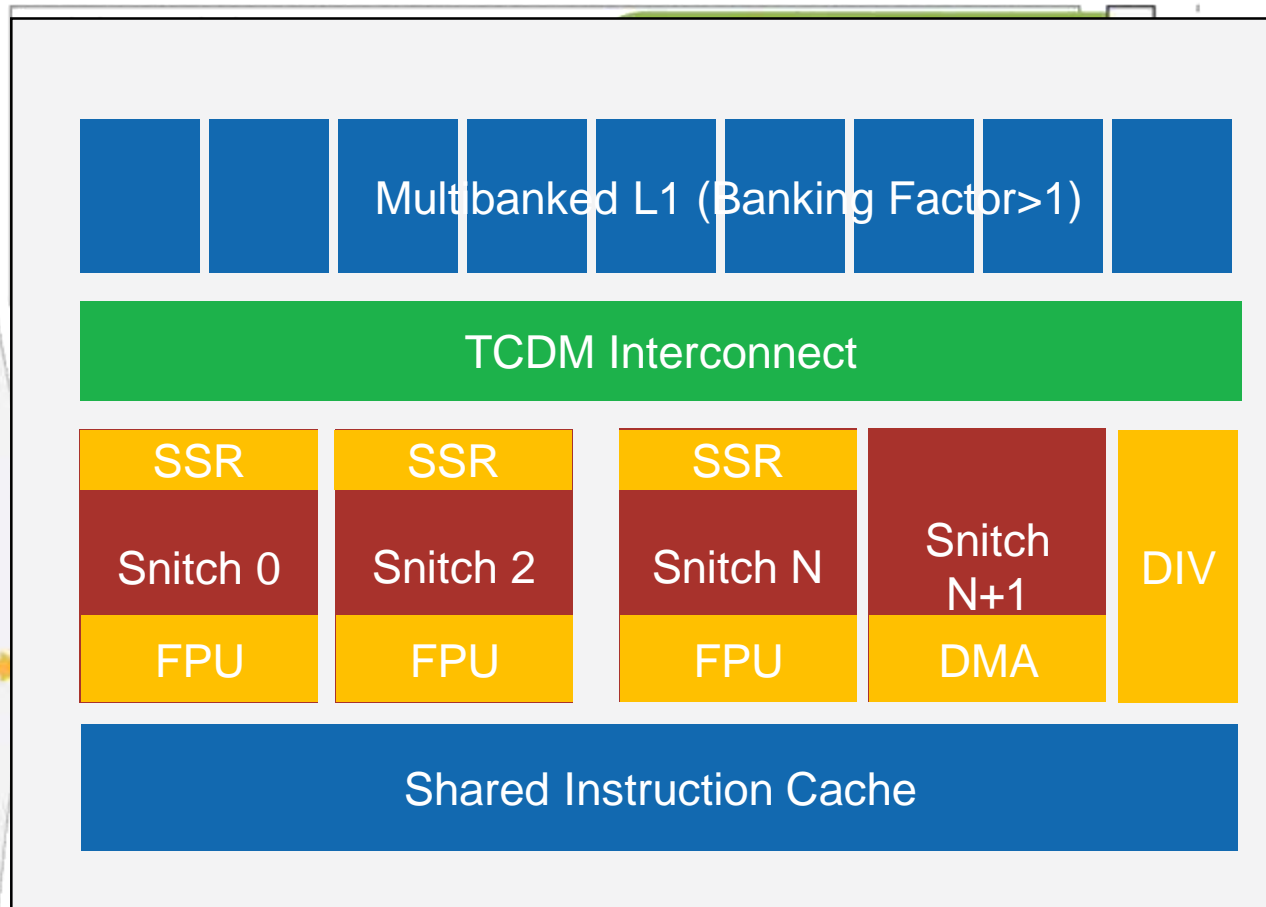
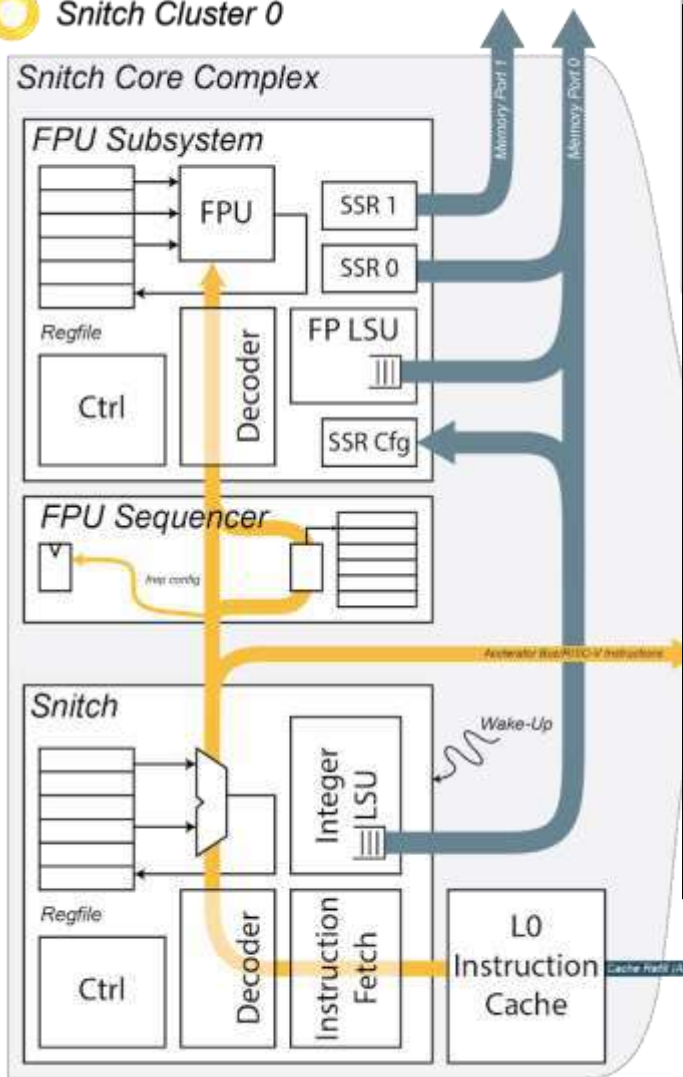
- 64-bit AXI DMA
- Tightly coupled with Snitch
- Operates on wide 512-bit data
- Hardware support to autonomously manage data
- Higher-dimensionality can be supported
- Intrinsics/library for easy programming

```

// setup and start a 1D transfer, return transfer ID
uint32_t __builtin_sdma_start_oned(
    uint64_t src, uint64_t dst, uint32_t size, uint32_t cfg);
// setup and start a 2D transfer, return transfer ID
uint32_t __builtin_sdma_start_twod(
    uint64_t src, uint64_t dst, uint32_t size,
    uint32_t sstrd, uint32_t dstrd, uint32_t nreps, uint32_t cfg);
// return status of transfer ID tid
uint32_t __builtin_sdma_stat(uint32_t tid);
// wait for DMA to be idle (no transfers ongoing)
void __builtin_sdma_wait_for_idle(void);
  
```

Snitch Cluster Architecture

Snitch Cluster 0



Legend:





And where does the Energy go?

In an 8-core cluster

Inevitable to have local memory
(e.g., GPU/GPU L1 cache, vector register file)

Integer core uses
2% of power

L1 Memory
47.19mW
27%

FPU uses **50%** of power

Integer Core
4.24mW
2%

SSR/FREP
9.52mW
5%

FPU
87.44mW
49%

ICACHE
4.82mW
3%

SSR/FREP hardware
uses **5%** of power

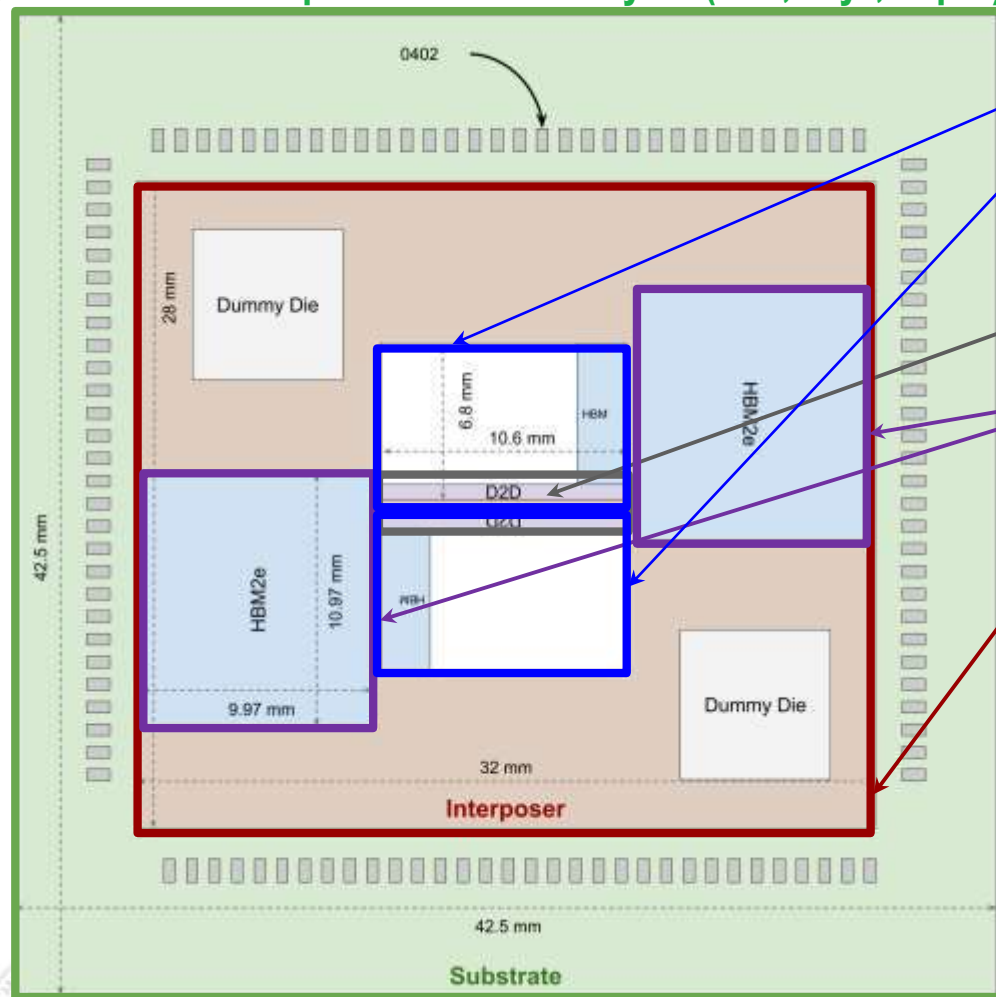
Miscellaneous
25.26mW
14%

Spending energy where it contributes to the result → **High Efficiency**



Occamy: a 12LPP+ Stream Computing Chiplet

Silicon implementation of all key IPs (core, Phys, chiplet)



Dual-chiplet configuration in 12LP+

- Area: $\sim 70\text{mm}^2$

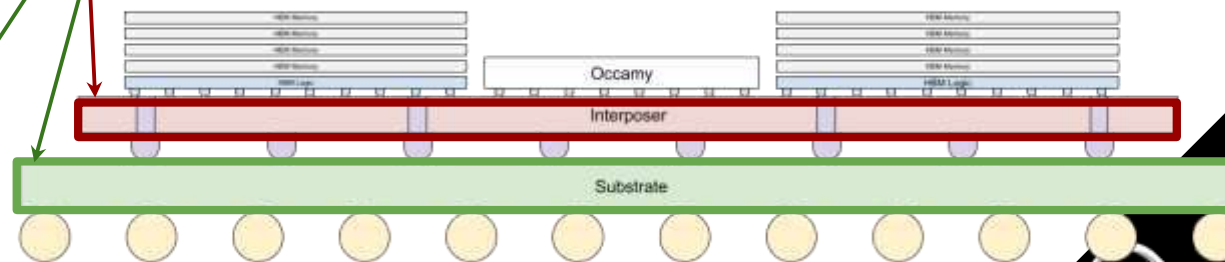
Chip2chip link

HBM2e memories

Interposer

- 65nm tech, passive

Substrate



Snitch Cluster

8 Snitch compute cores

- Single-stage, small Integer control core

9th Core: DMA

- 512 bit data interface
- Efficient data movement

128 kB TCDM

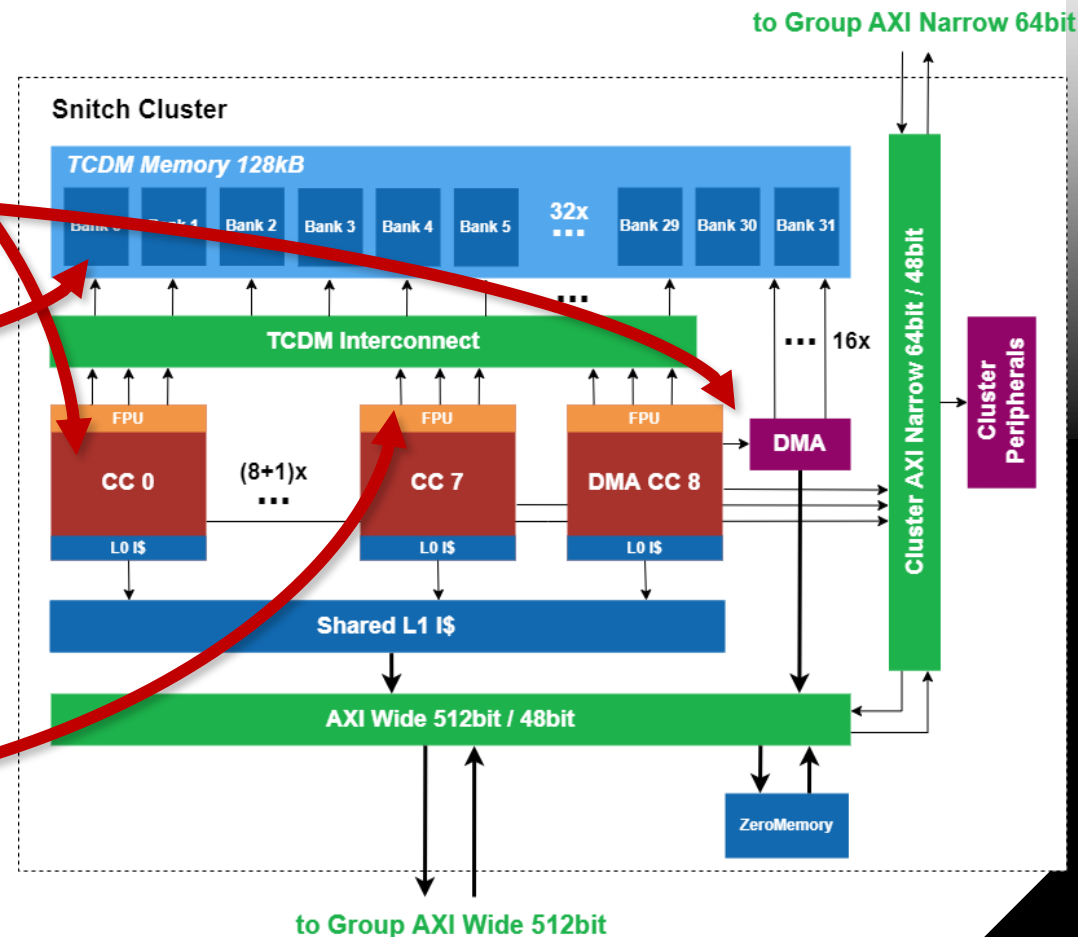
- Scratchpad for predictable memory accesses
- 32 Banks

Custom ISA extensions

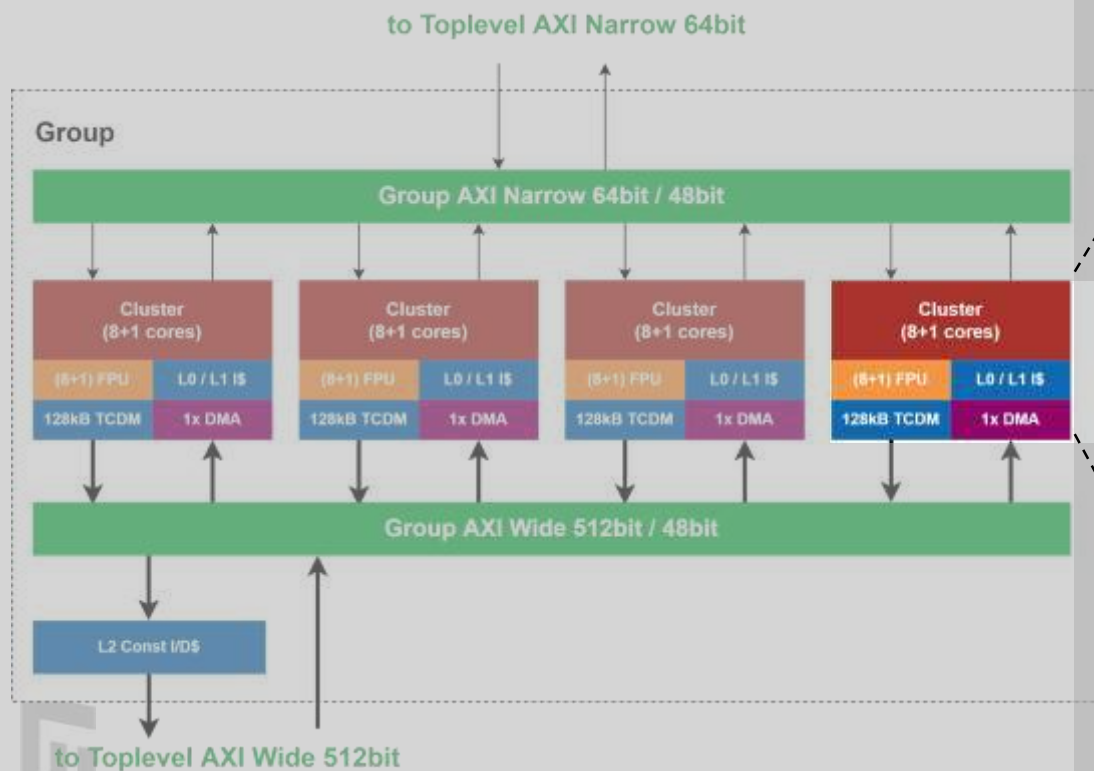
- Xfrep, Xssr
- **New: Xissr sparsity support**

1 FPU per Snitch core

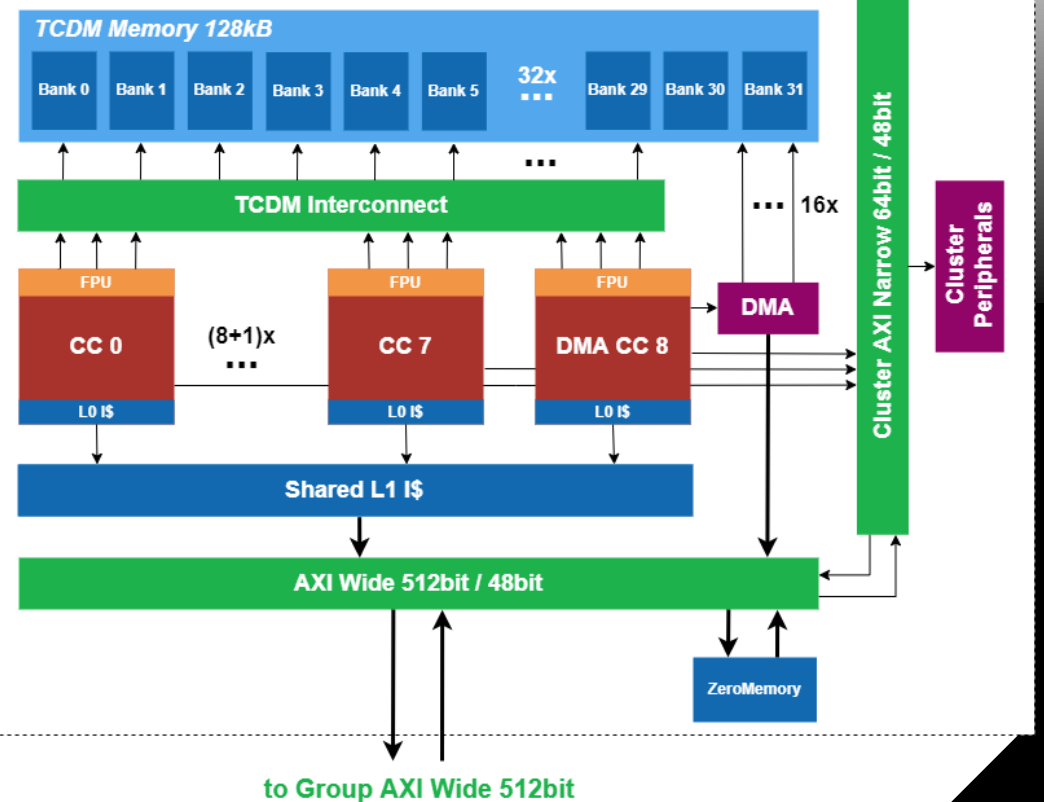
- Decoupled and heavily pipelined
- Multi-format FPU (+SIMD)
- **New: Minifloat support + SDOTP**



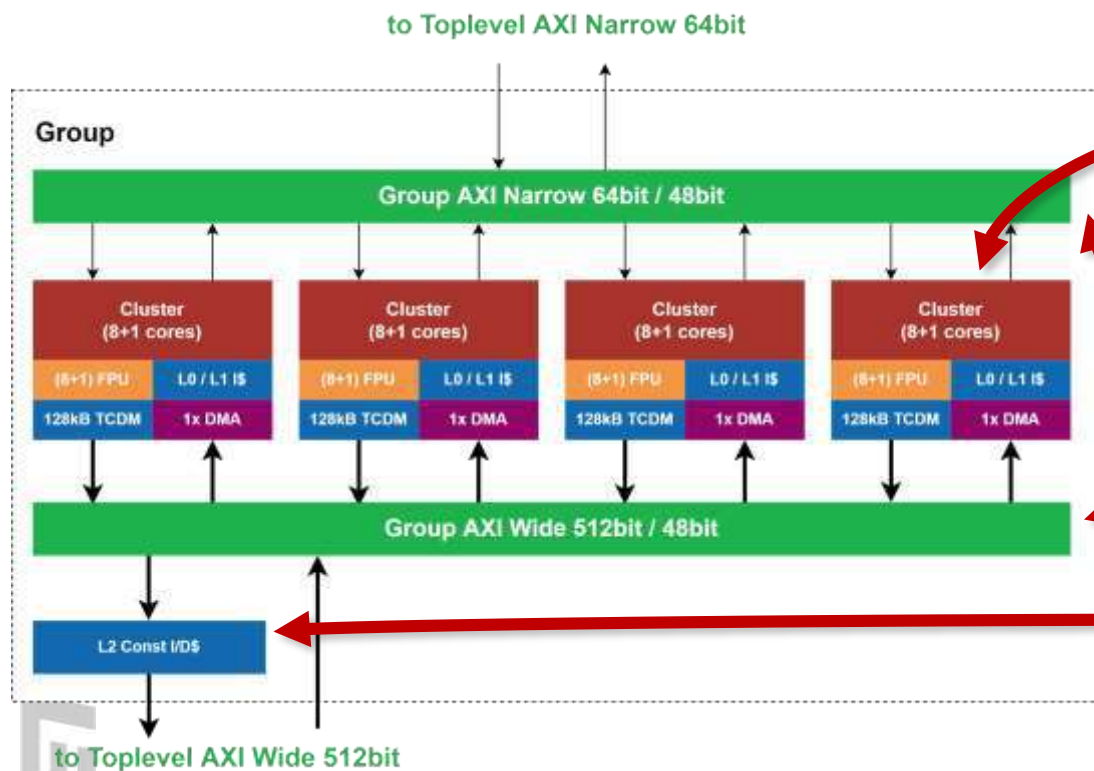
Snitch Group



Snitch Cluster



Snitch Group



4 Clusters per Group

- Single-stage, small Integer control core

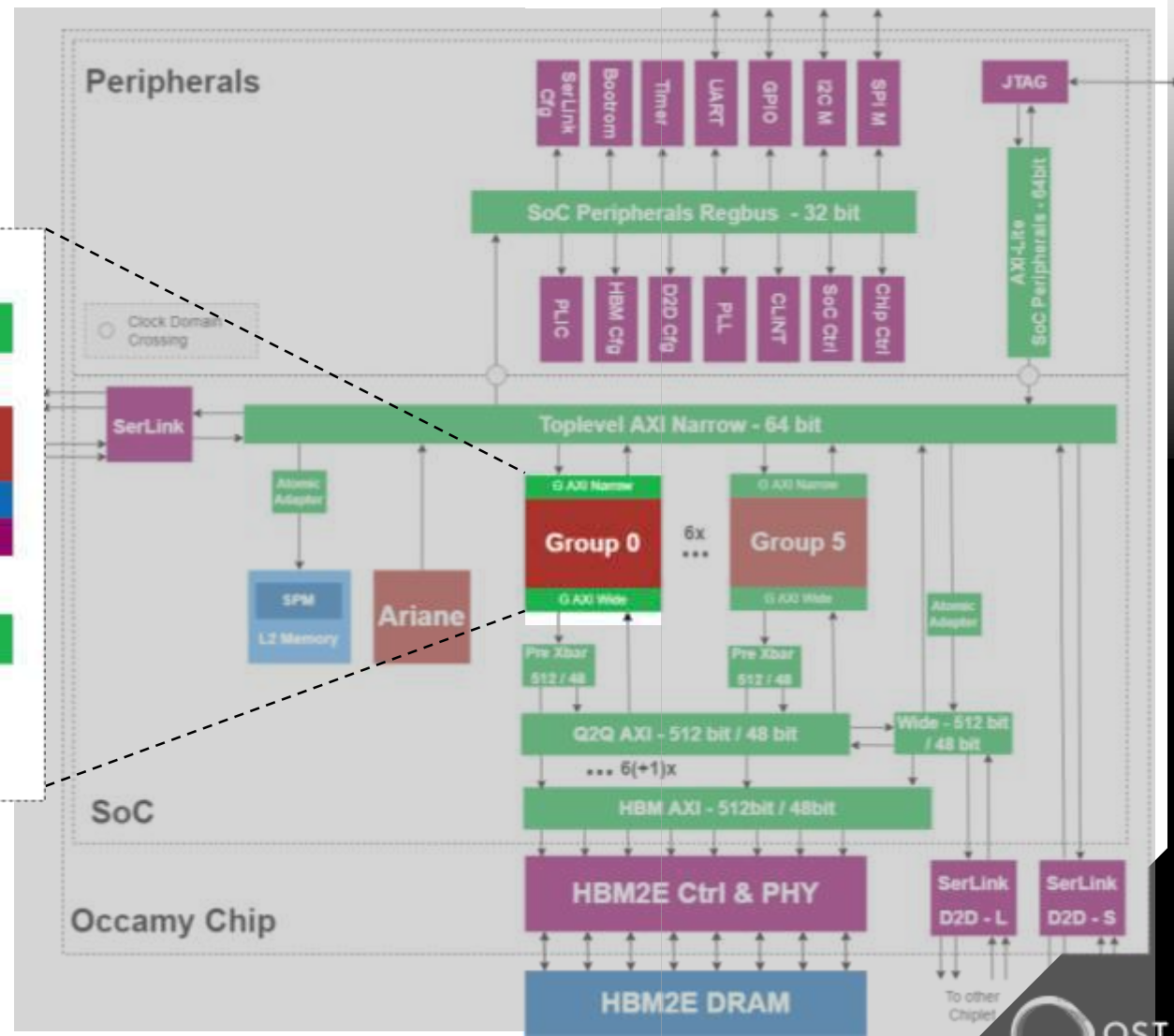
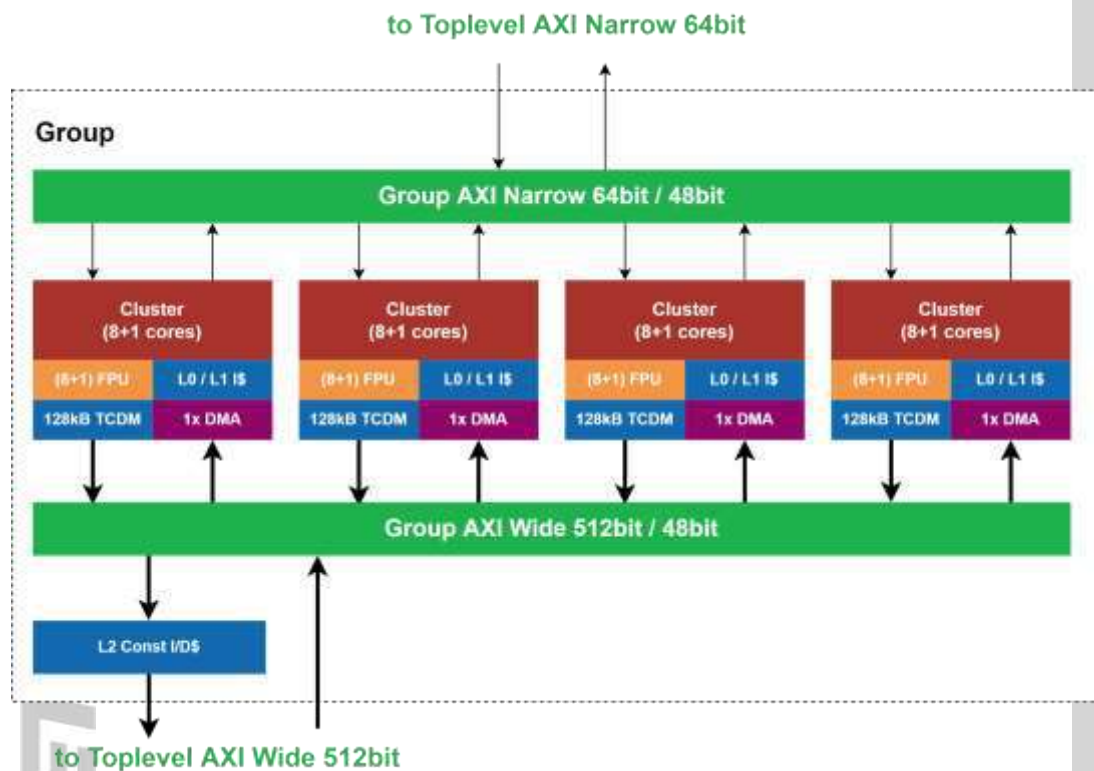
2 AXI Busses

- 64-bit narrow interface: config
- 512-bit wide interface: DMA

Constant Cache

- D/I-Cache hierarchy

Snitch Group



Occamy – Chiplet Architecture Overview

2 AXI Busses

- 64-bit narrow interface: config
- 512-bit wide interface: DMA

Peripherals

- Complex address space management

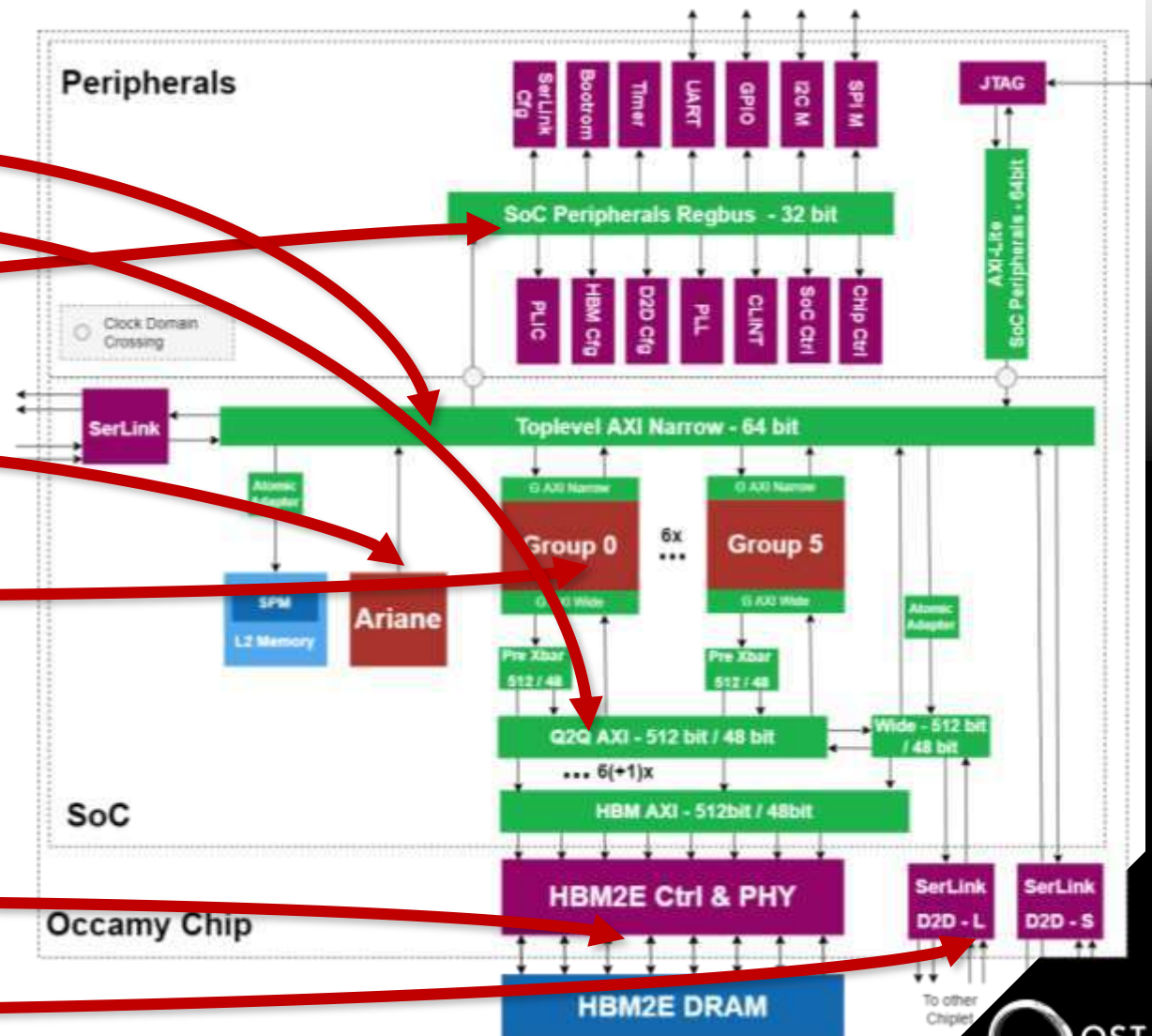
Linux-capable manager core CVA6

6 Groups: 216 cores/die

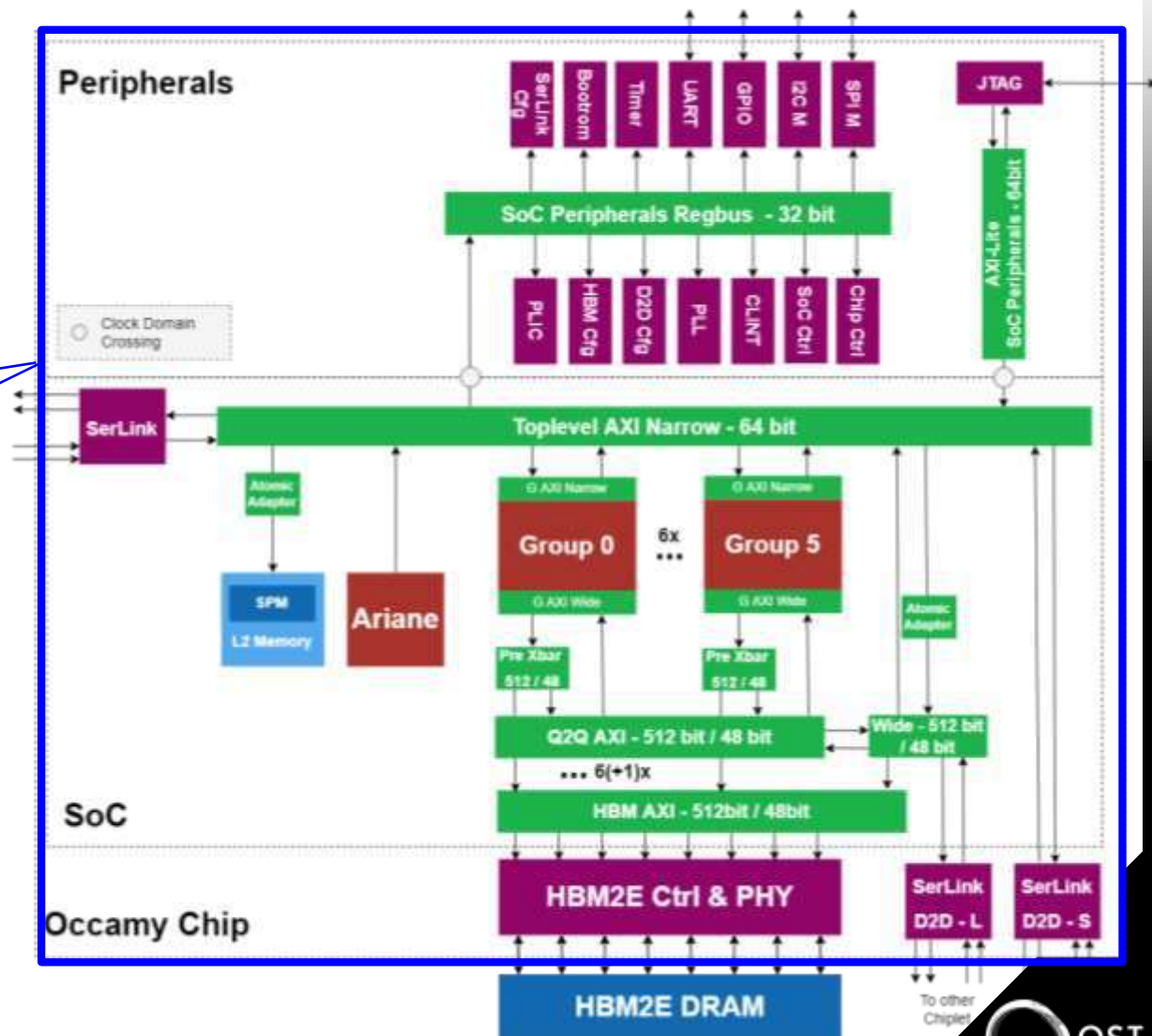
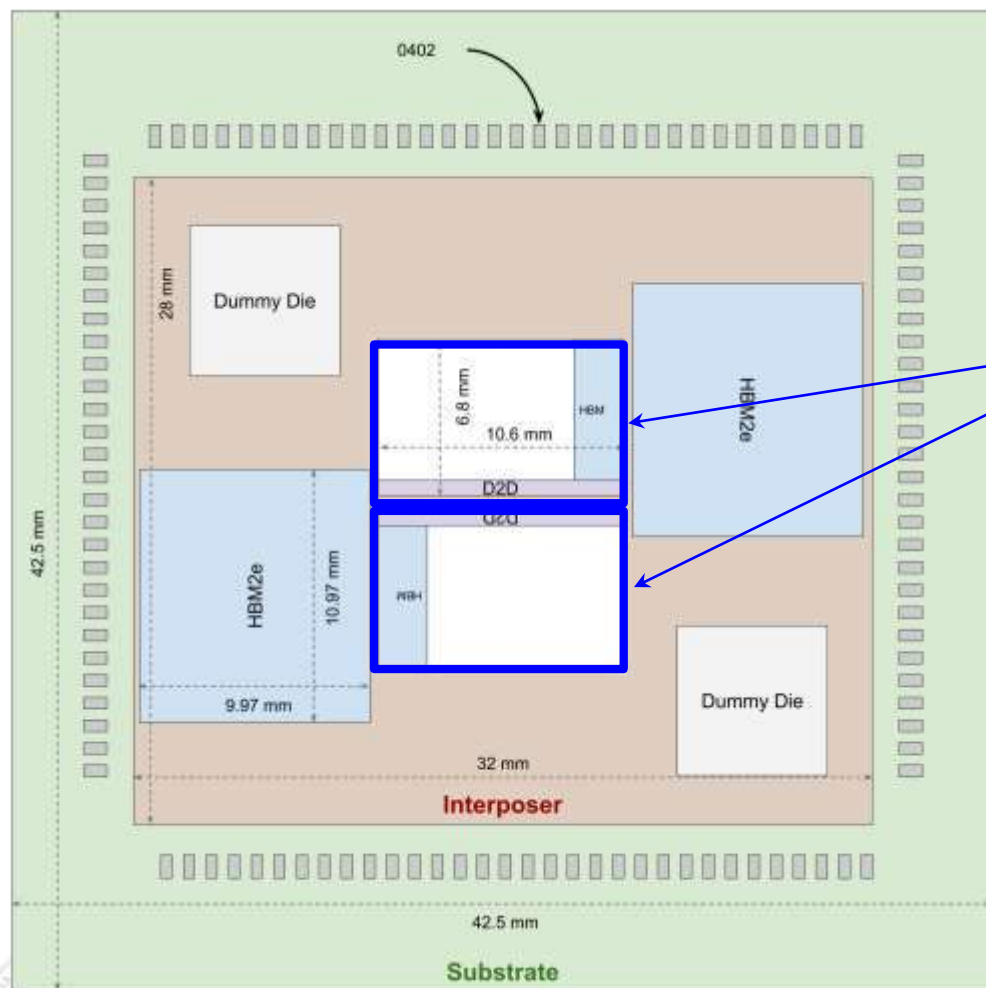
- 4 cluster / group:
 - 8 compute cores / cluster
 - 1 DMA core / cluster
- 512bit Constant Cache

8-channel HBM2e (8GB)

D2D serial link



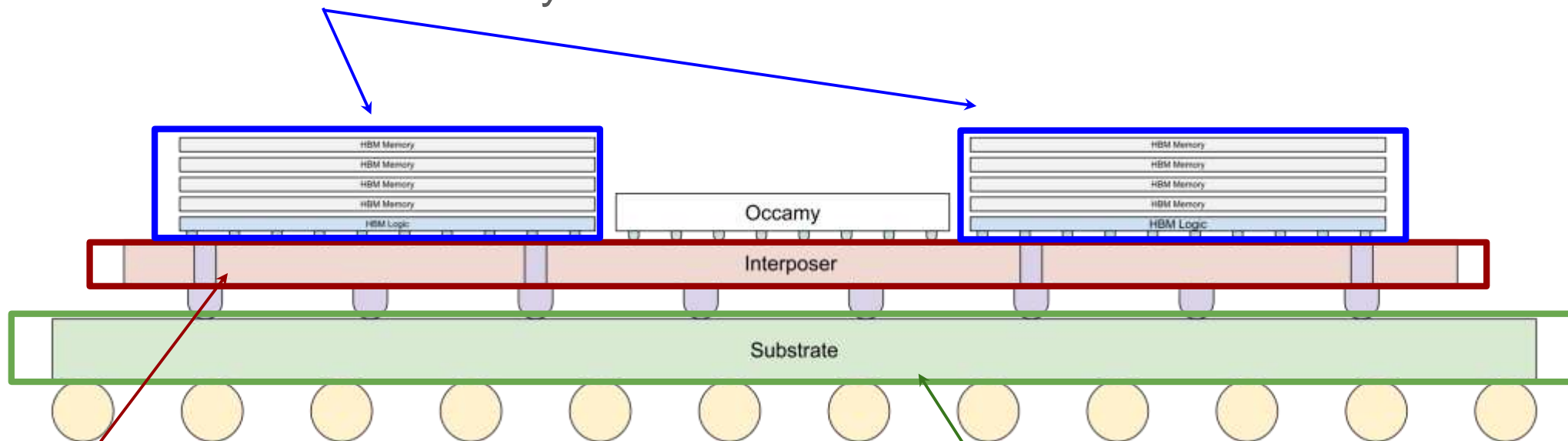
System View



2.5D Integration of Occamy

HBM2e:

- 2x 8GB HBM2e Memory Stacks



Interposer:

- Technology: 65nm, passive (only BEOL)
- 4 Metal layers
- Area: full reticle, 32mm x 28mm
- Thickness: 775μm

Substrate:

- HDBU
- JEDEC: 42.5mm x 42.5mm

Programming Model

```

void main() {
    unsigned repetition = 2, bound = 4, stride = 8;
    static int data[8] = {1,2,3,4,5,6,7,8};

    __builtin_ssr_setup_1d(0, repetition, bound, stride, data);
    static volatile double d = 42.0;

    __builtin_ssr_enable();

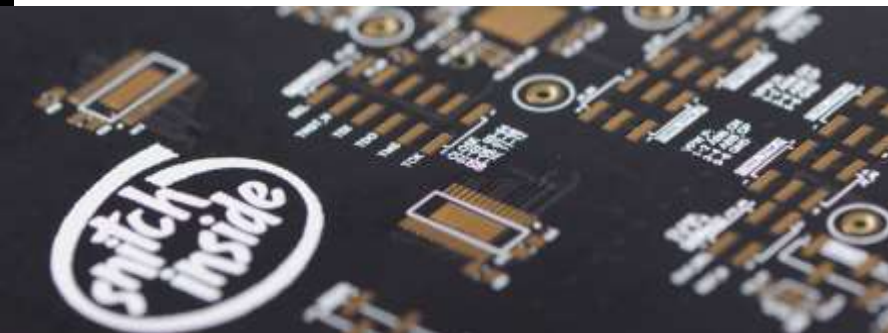
    __builtin_ssr_push(0, d);
    volatile double e;
    e = __builtin_ssr_pop(0);
    __builtin_ssr_disable();
}

```

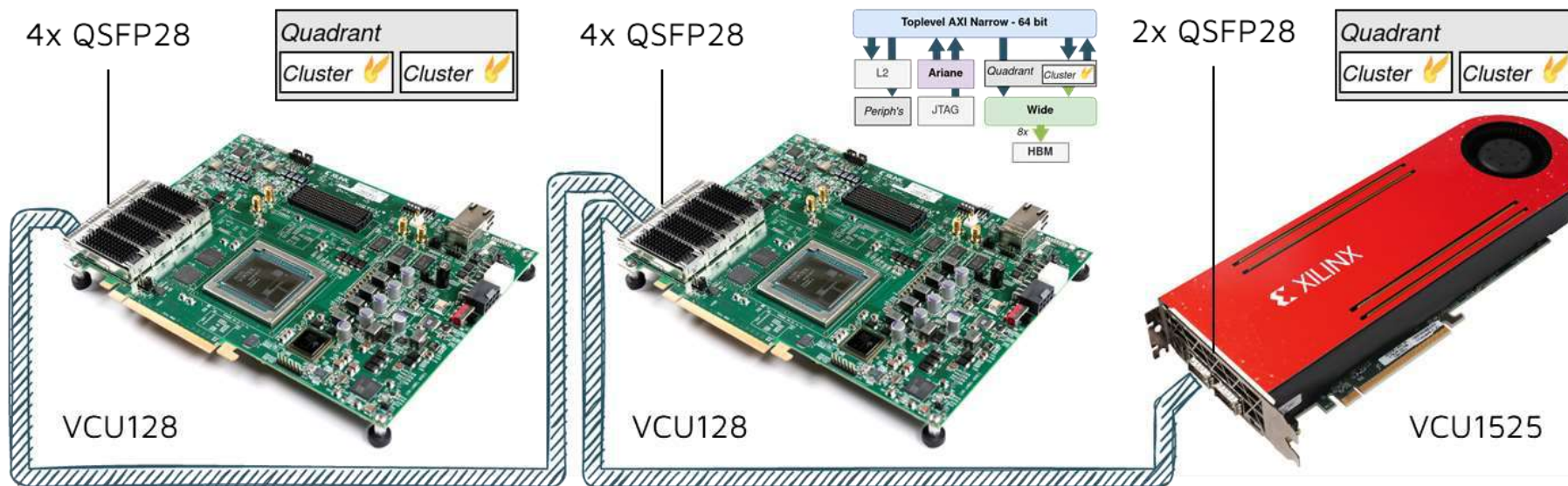
- Multiple layers of abstraction:
 - Hand-tuned assembly
 - LLVM intrinsics
 - FREP inference
 - High-level frameworks:
 - DaCE:** spcl.inf.ethz.ch/Research/DAPP/
 - Pytorch+Dory:** tiling of neural networks
- Bare-metal runtime
- Basic OpenMP runtime



Prototyping and Emulation



- “Quad-chiplet” prototype board with FPGA interface
- Occamy mapped onto 2x VCU128 (with HBM) + 1x VCU1525
 - 1x CVA6
 - 2-4x 9-core Snitch cluster

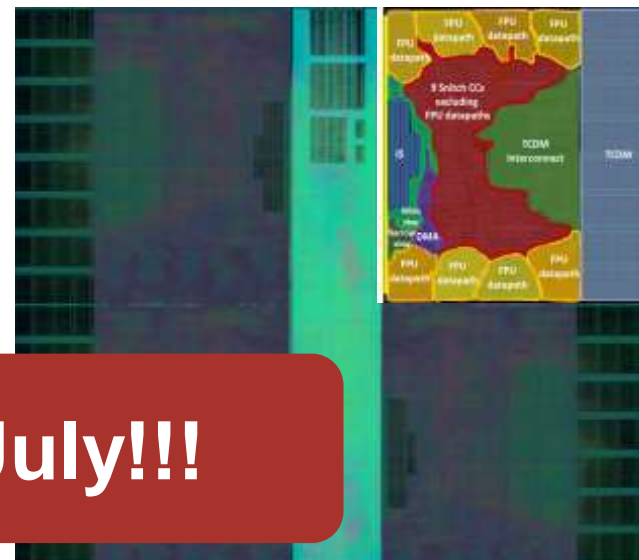


Conclusion

Occamy: silicon demonstration Stream Processor on a 12LP+ chiplet

- We keep improving the architecture:
 - FPU with SIMD Mini-float support
 - Sparsity support
 - Atomics and interrupts
 - I-Cache hierarchy
- **Peak system performance @1GHz:**
 - FP64: 768 GFLOp/s
 - FP32: 1.536 TFLOp/s
 - FP16: 3.072 TFLOp/s
 - FP8: 6.144 TFLOp/s
- **“Quad-chiplet” board with FPGA interface**
- **FPGA emulation working**
 - CVA6 boots Linux
 - Snitch drivers

Tape-out Occamy Mid July!!!





PULP

Parallel Ultra Low Power

Luca Benini, Alessandro Capotondi, Alessandro Ottaviano, Alessio Burrello, Alfio Di Mauro, Andrea Borghesi, Andrea Cossettini, Andreas Kurth, Angelo Garofalo, Antonio Pullini, Arpan Prasad, Bjoern Forsberg, Corrado Bonfanti, Cristian Cioflan, Daniele Palossi, Davide Rossi, Fabio Montagna, Florian Glaser, Florian Zaruba, Francesco Conti, Georg Rutishauser, Germain Haugou, Gianna Paulin, Giuseppe Tagliavini, Hanna Müller, Luca Bertaccini, Luca Valente, Manuel Eggimann, Manuele Rusci, Marco Guermandi, Matheus Cavalcante, Matteo Perotti, Matteo Spallanzani, Michael Rogenmoser, Moritz Scherer, Moritz Schneider, Nazareno Bruschi, Nils Wistoff, Pasquale Davide Schiavone, Paul Scheffler, Philipp Mayer, Robert Balas, Samuel Riedel, Segio Mazzola, Sergei Vostrikov, Simone Benatti, Stefan Mach, Thomas Benz, Thorir Ingolfsson, Tim Fischer, Victor Javier Kartsch Morinigo, Vlad Niculescu, Xiaying Wang, Yichao Zhang, Frank K. Gürkaynak, all our past collaborators **and many more that we forgot to mention**



<http://pulp-platform.org>



@pulp_platform