ETH zürich

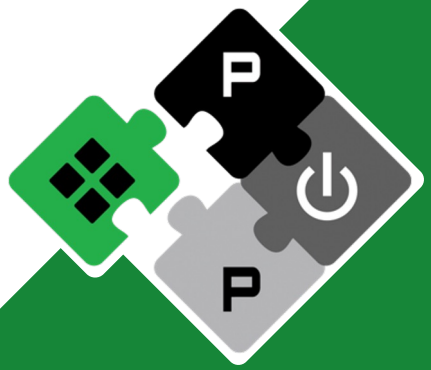
ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Domain-Specific Platforms
End-to-End Open-Source Design

From Dream to Reality

Luca Benini lbenini@iis.ee.ethz.ch
luca.benini@unibo.it

PULP Platform
Open Source Hardware, the way it should be!

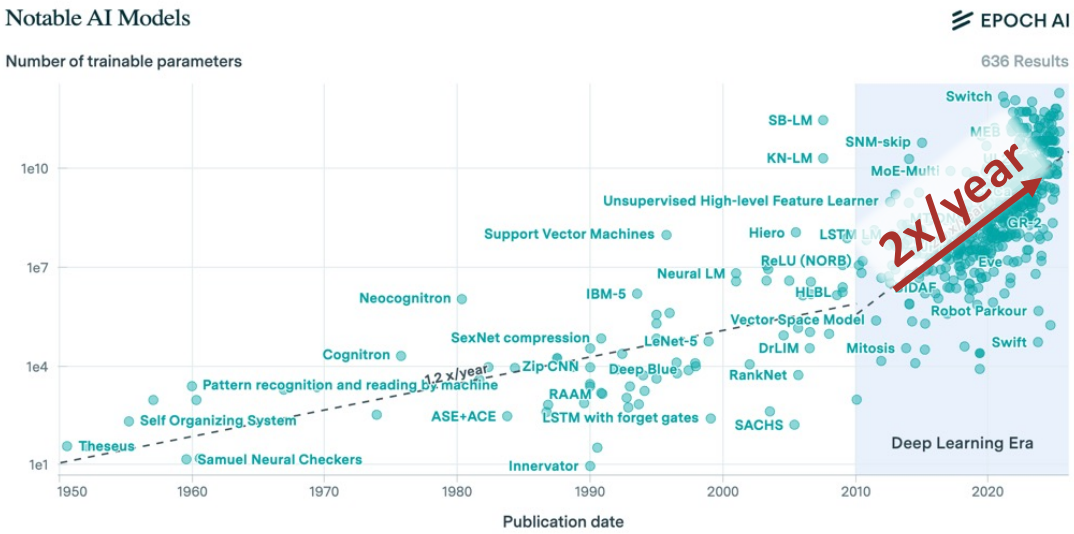


pulp-platform.org
@pulp_platform
company/pulp-platform
youtube.com/pulp_platform

AI Hardware Platforms: Model Scaling Laws

*Deep learning models continue to grow in **scale** and **complexity***

- Growing model sizes demand ever-increasing compute and memory

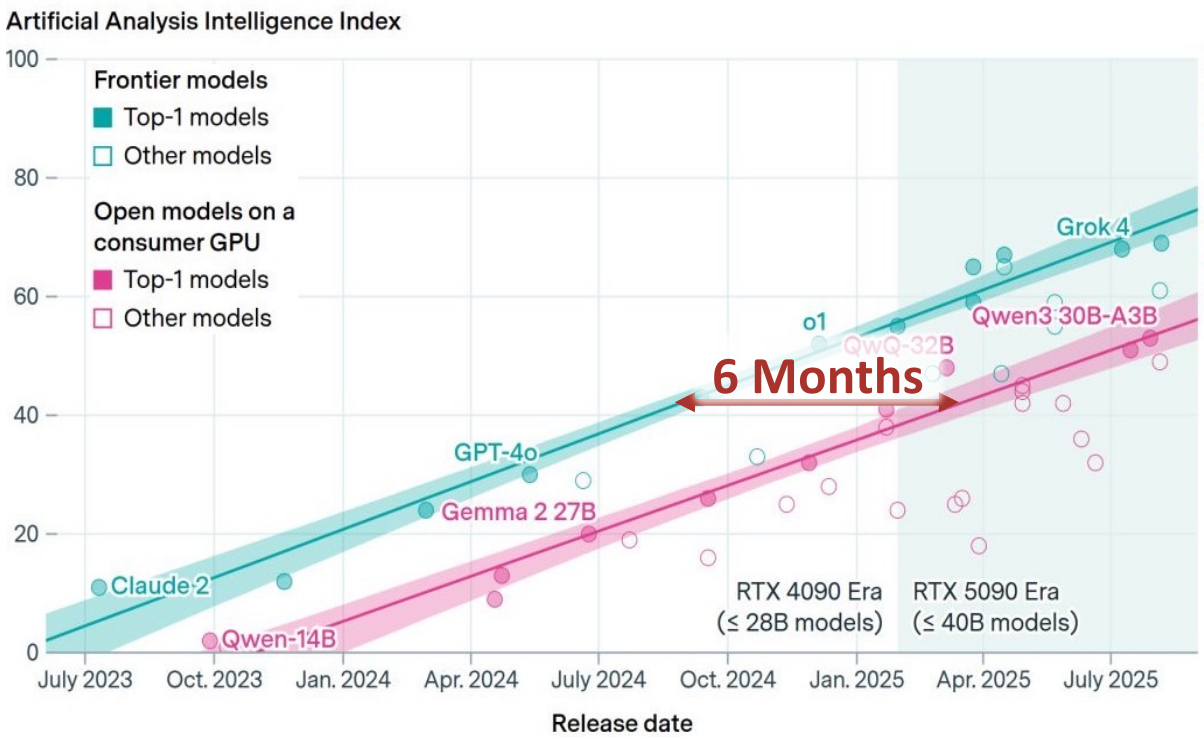


Source: https://epoch.ai/

AI Hardware Platforms: Scaling @Inference, @Edge

*Deep learning models continue to grow in **scale** and **complexity***

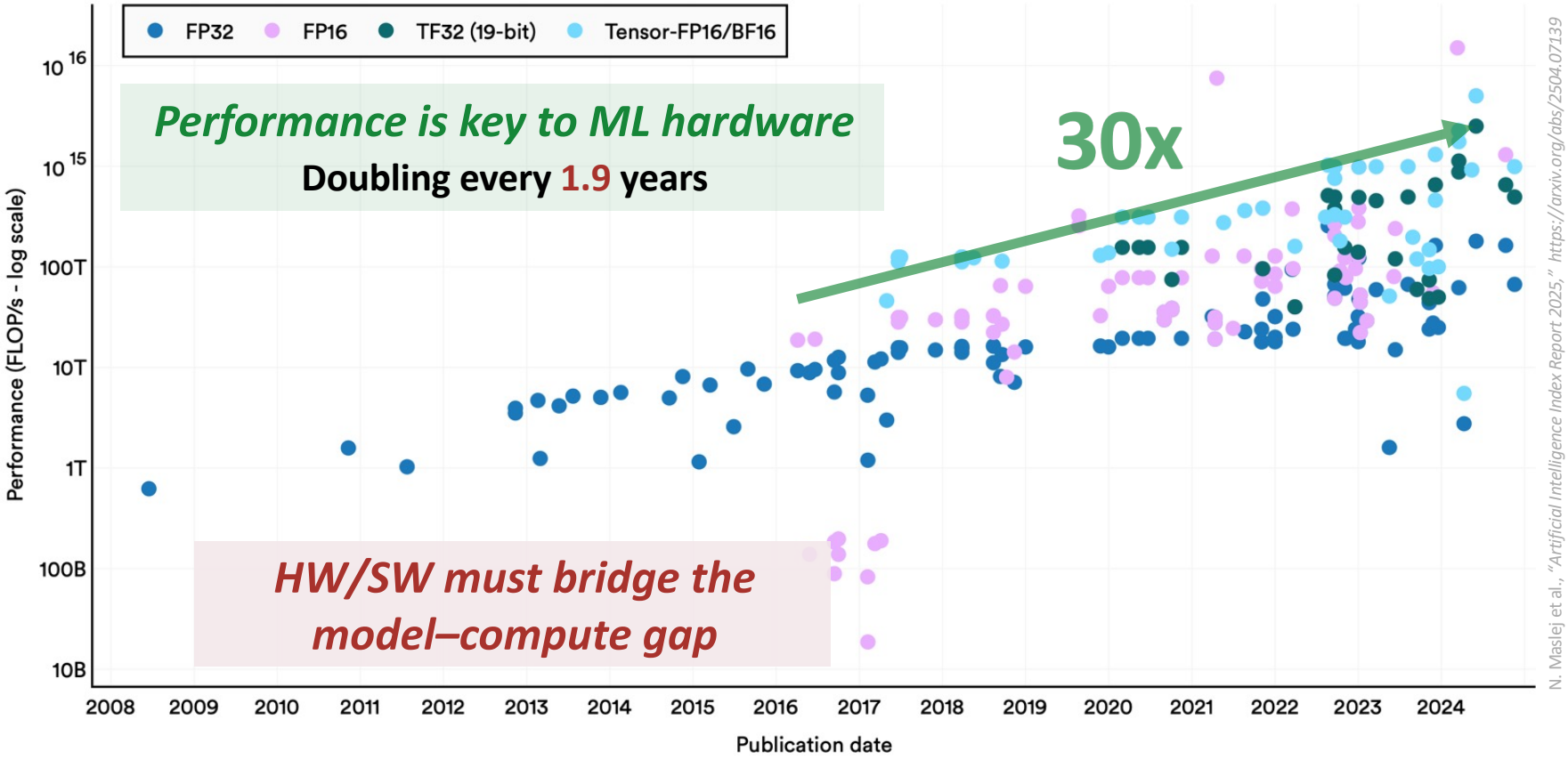
- Growing model sizes demand ever-increasing compute and memory
- ***Inference*** compute scales even faster than for training
- ***Edge*** models that fit on a single GPU trail the frontier by less than one year



Source: https://epoch.ai/

Hardware Scaling is key to AI Progress (cloud & edge)

Peak computational throughput of notable ML hardware (energy efficiency must track!)

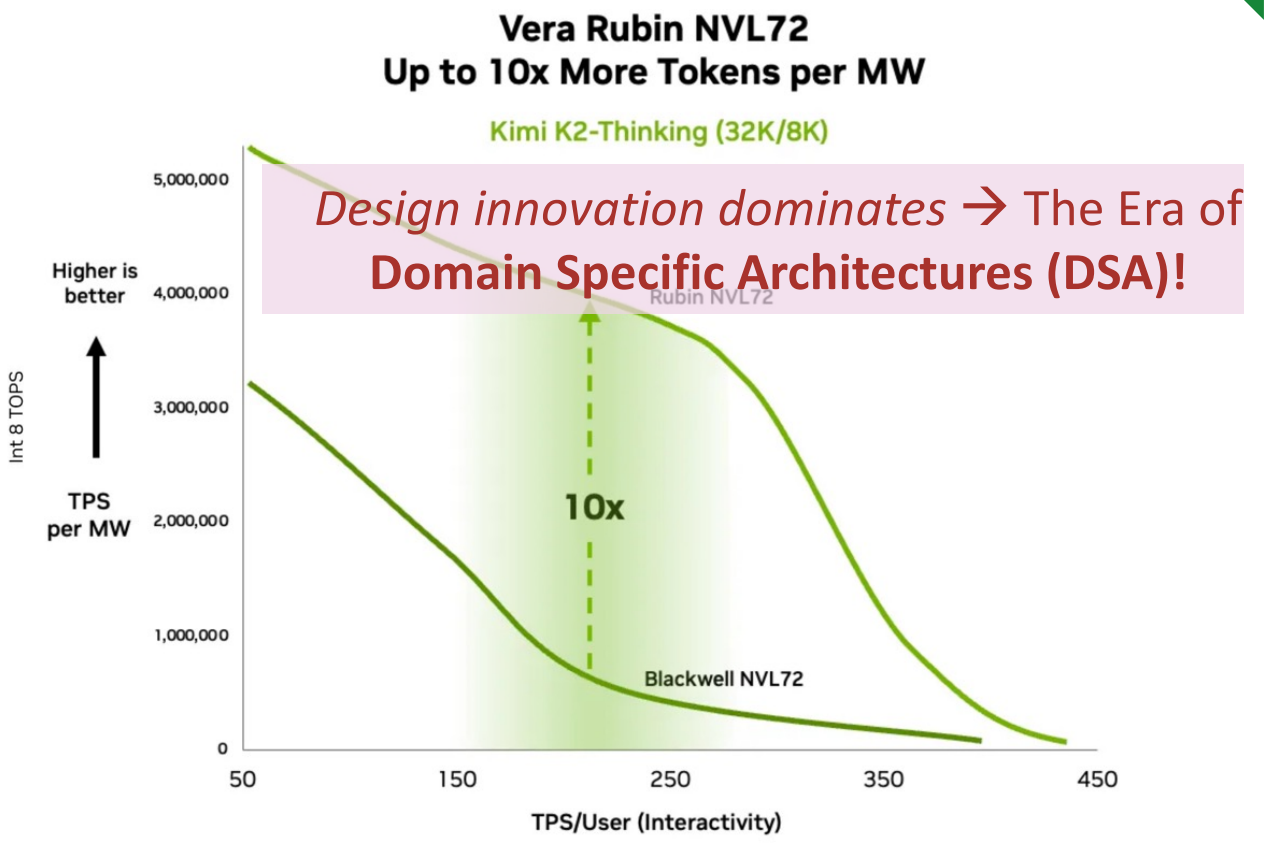


N. Maslej et al., *"Artificial Intelligence Index Report 2025,"* https://arxiv.org/abs/2504.07139

How is Industry doing it?

Gains from

- Number Representation
  - FP32, FP16, Int8
  - (TF32, BF16)
  - ~16x
- Complex Instructions
  - DP4, HMMA, IMMA
  - ~12.5x
- Process
  - 28nm, 16nm, 7nm, 5nm
  - ~2.5x
- Sparsity
  - ~2x
- Model efficiency has also improved – overall gain > 1000x



Design innovation dominates → The Era of **Domain Specific Architectures (DSA)!**

[Dally HotChips 2023]

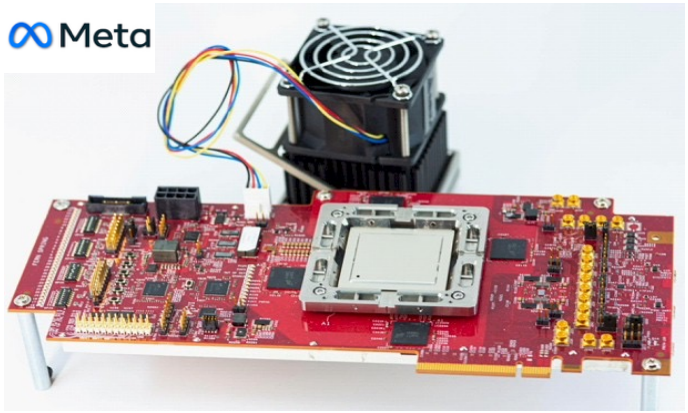
Innovation beyond “NVIDIA Gravity” is Challenging!

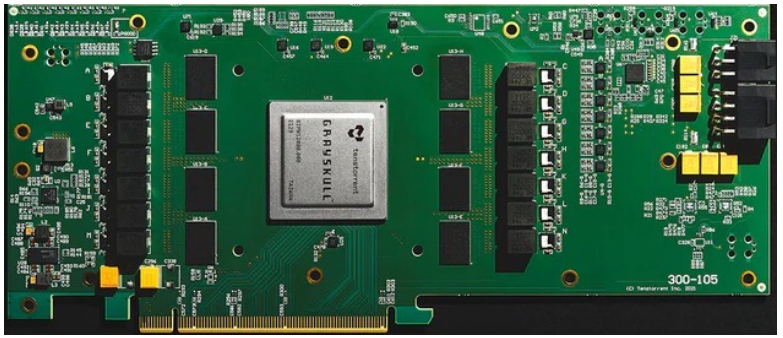
It’s the software → **flexibility** key for fast evolution!

Need an **open standard** to counter a monopoly

RISC-V®

RISC-V: The Free and Open RISC Instruction Set Architecture



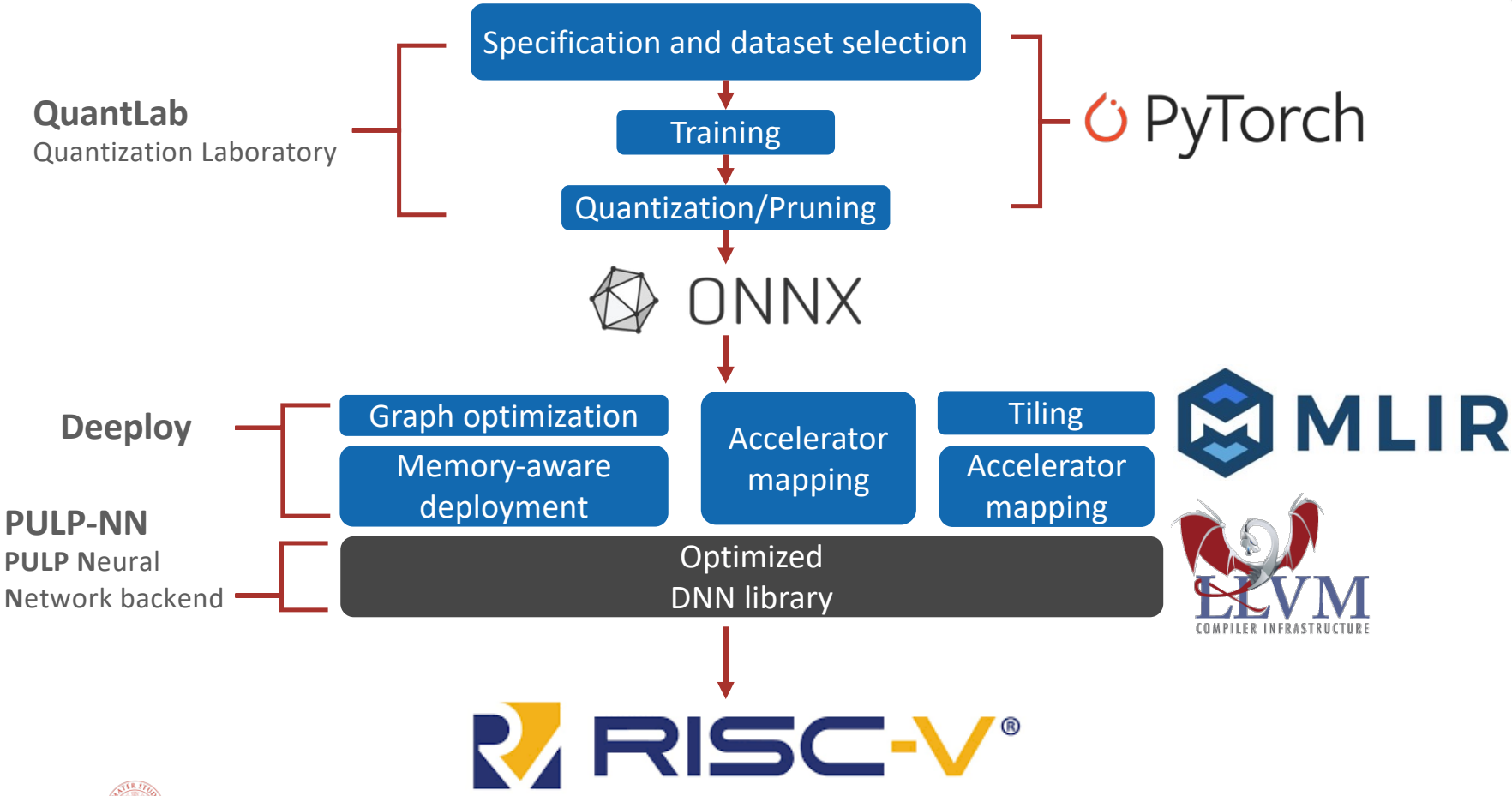


tenstorrent

ETH zürich

ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

Fully Open-Source AI SW Stack with RISC-V!



RISC-V is Open & Extensible: Key for DSA

- **RISC-V has Reserved opcodes for standard extensions**
- **Rest of opcodes free for custom implementations**
- **Custom extensions can be standardized**
  - Standard extensions will be frozen/not change in the future

inst[4:2] / inst[6:5]	000	001	010	011	100	101	110	111 ($> 32b$)
00	LOAD	LOAD-FP	*custom-0*	MISC-MEM	OP-IMM	AUIPC	OP-IMM-32	$48b$
01	STORE	STORE-FP	*custom-1*	AMO	OP	LUI	OP-32	$64b$
10	MADD	MSUB	NMSUB	NMADD	OP-FP	*reserved*	*custom-2/rv128*	$48b$
11	BRANCH	JALR	*reserved*	JAL	SYSTEM	*reserved*	*custom-3/rv128*	$\geq 80b$

Extensibility is fundamental in the RISC-V ISA!

Extensions at work: Achieving ~100% **dotp** Unit Utilization

8-bit Convolution

HW Loop

LD/ST with post increment

8-bit SIMD sdotp

8-bit sdotp + LD

RV32IMC

N

```
addi   a0,a0,1
addi   t1,t1,1
addi   t3,t3,1
addi   t4,t4,1
lbu    a7,-1(a0)
lbu    a6,-1(t4)
lbu    a5,-1(t3)
lbu    t5,-1(t1)
mul    s1,a7,a6
mul    a7,a7,a5
add    s0,s0,s1
mul    a6,a6,t5
add    t0,t0,a7
mul    a5,a5,t5
add    t2,t2,a6
add    t6,t6,a5
bne    s5,a0,1c000bc
```

RV32IMCXpulp

N/4

```
lp.setup
p.lw  w1, 4(a0!)
p.lw  w2, 4(a1!)
p.lw  x1, 4(a2!)
p.lw  x2, 4(a3!)
pv.sdotsp.b   s1, w1, x1
pv.sdotsp.b   s2, w1, x2
pv.sdotsp.b   s3, w2, x1
pv.sdotsp.b   s4, w2, x2
end
```

can we remove?

Yes! dotp+ld

N/4

```
Init NN-RF (outside of the loop)
lp.setup
pv.nnsdotup.h  s0,ax1,9
pv.nnsdotsp.b  s1, aw2, 0
pv.nnsdotsp.b  s2, aw4, 2
pv.nnsdotsp.b  s3, aw3, 4
pv.nnsdotsp.b  s4, ax1, 14
end
```

9x less instructions than RV32IMC

14.5x less instructions at an affordable area cost (50%)

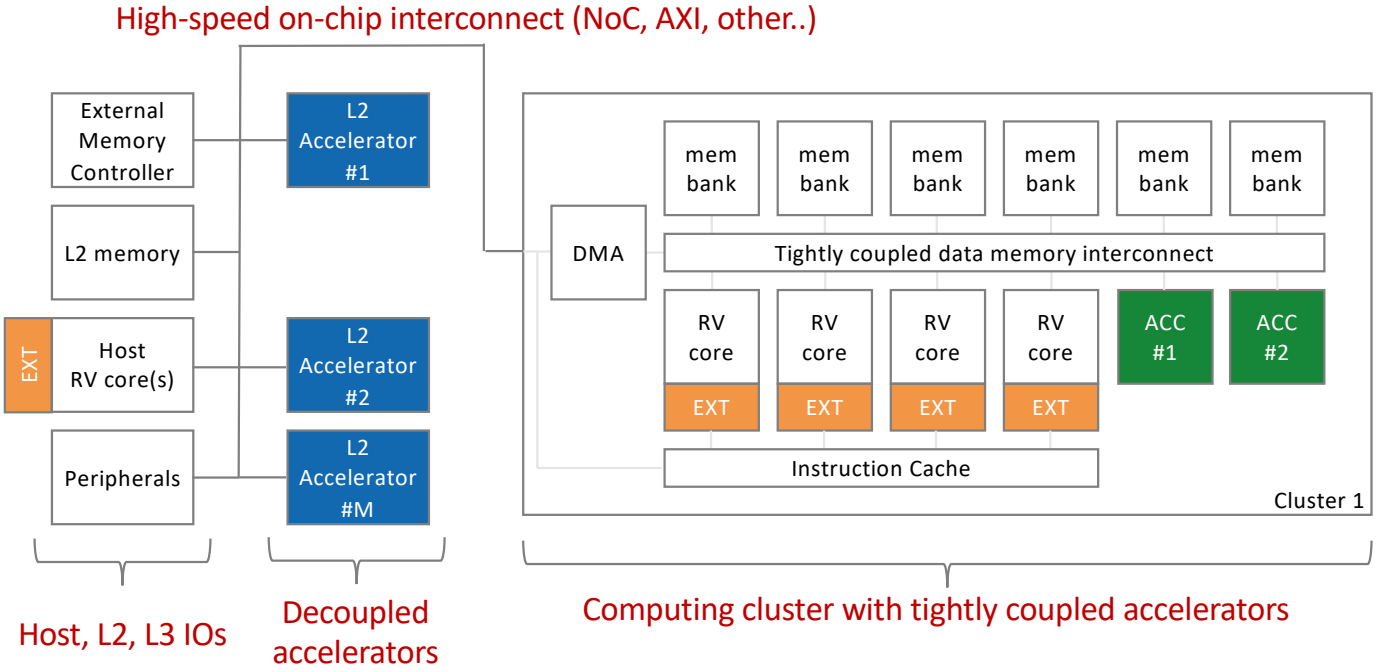
RISC-V Enables Domain Specific Architectures (DSAs)

Multiple Scales of acceleration

Extensions to processor cores
- ISA extension
- Share L0 memory

Shared-memory co-processor(s)
- ISA extension or offload
- Shared L1 memory

Decoupled Accelerator(s)
- Offload
- Shared L2(+) memory



RISC-V is a key enabler → max agility, enabling SW build-up, without vendor lock-in

Domain Specialization in perspective

Kraken: Using 22FDX tech, NT@0.6V, High utilization, minimal IO & overhead

Energy-Efficient RISC-V Core → **20pJ (8bit)**

ISA-based 10-20x →**1pJ (4bit)** → **ISA extension (e.g. RV XPULP)**

Configurable DP 10-20x → **100fJ (4bit)** → **Coprocessor (e.g. Tensor Unit)**

Highly specialized DP 100x →**1fJ (ternary)** → **Decoupled Accelerator (e.g. NVDLA)**

PULP: Open Hardware Toolbox for DSAs

RISC-V Cores and Vector Units

RI5CY *CV32E*	Zero R *Ibex*	Snitch	Spatz	Ariane *CVA6*	ARA
RV32	RV32	RV32	RVV	RV64	RVV

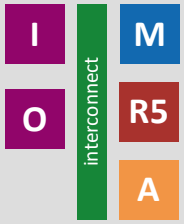
Peripherals

JTAG, SPI, UART, I2S, DMA, GPIO

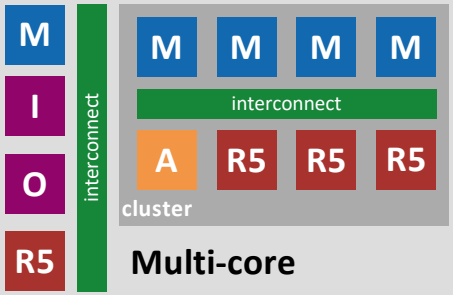
Interconnects

LIC, HCI, APB, FlooNoC, AXI4

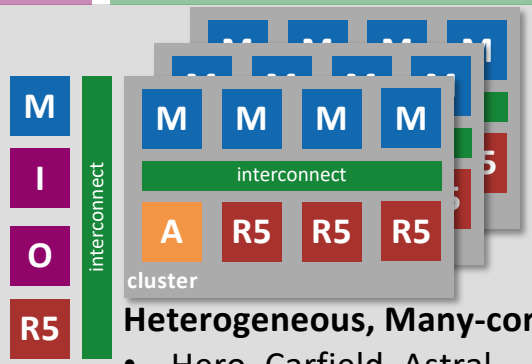
Platforms



Single core
- PULPino, PULPissimo
- Cheshire



Multi-core
- OpenPULP
- ControlPULP



Heterogeneous, Many-core
- Hero, Carfield, Astral
- Occamy, Mempool

IOT https://github.com/pulp-platform HPC

Accelerators and ISA extensions

XpulpNN, XpulpTNN	ITA (Transformers)	RBE, NEUREKA (QNNs)	FFT (DSP)	REDMULE (FP-Tensor)

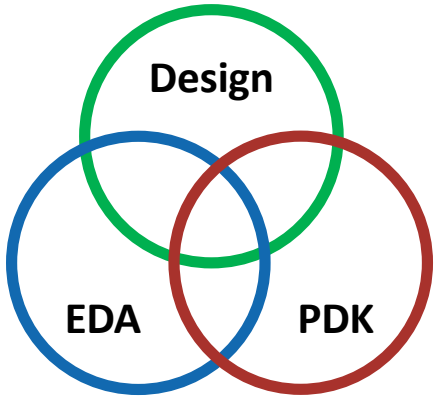
Beyond OpenHW: **End-to-end: all steps** of IC design

Design

- RTL / HDL descriptions (quite common)
- Schematics / Physical Design (may have dependencies to technology information)

Tools (EDA)

- Front-end tools (Synthesis)
- Back-end tools (Placement and Routing)
- Verification tools (Simulation)



Manufacturing (PDK)

- Design rules for manufacturing (separation, minimum width of metals)
- Layer stack information for parasitics (thickness, dielectric constants..)
- Device models (SPICE parameters) for simulation

End-to-end Open-Source IC Design is possible today!

Design: from PULP

`github.com/pulp-platform`

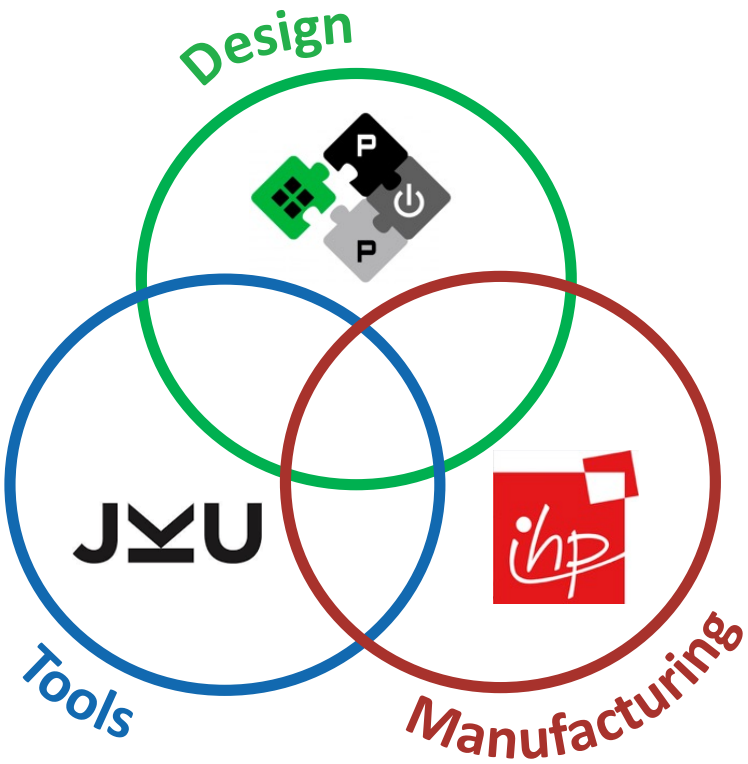
Tools: from Johannes Kepler University (JKU)

Reliable VM with large collection of open-source tools

`github.com/iic-jku/IIC-OSIC-TOOLS`

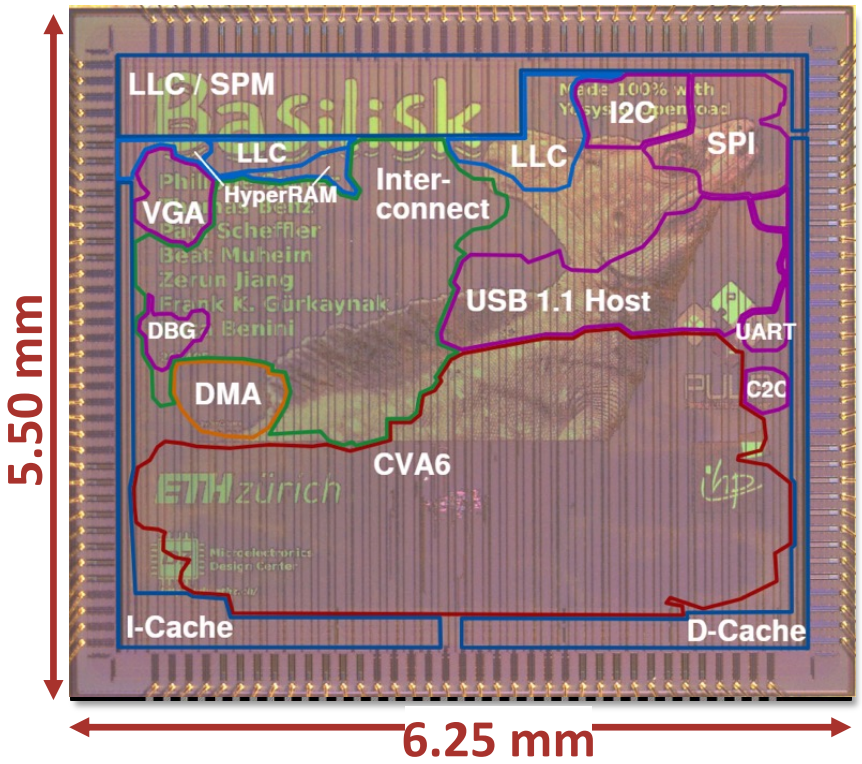
Manufacturing: IHP130nm

`github.com/IHP-GmbH/IHP-Open-PDK`

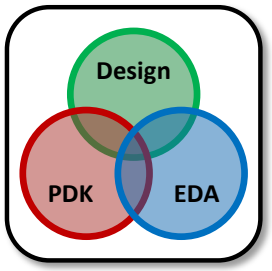


Is it practical? Can we really use this for large SoCs tapeouts?

Meet Basilisk: Open **RTL**, Open **EDA**, Open **PDK**

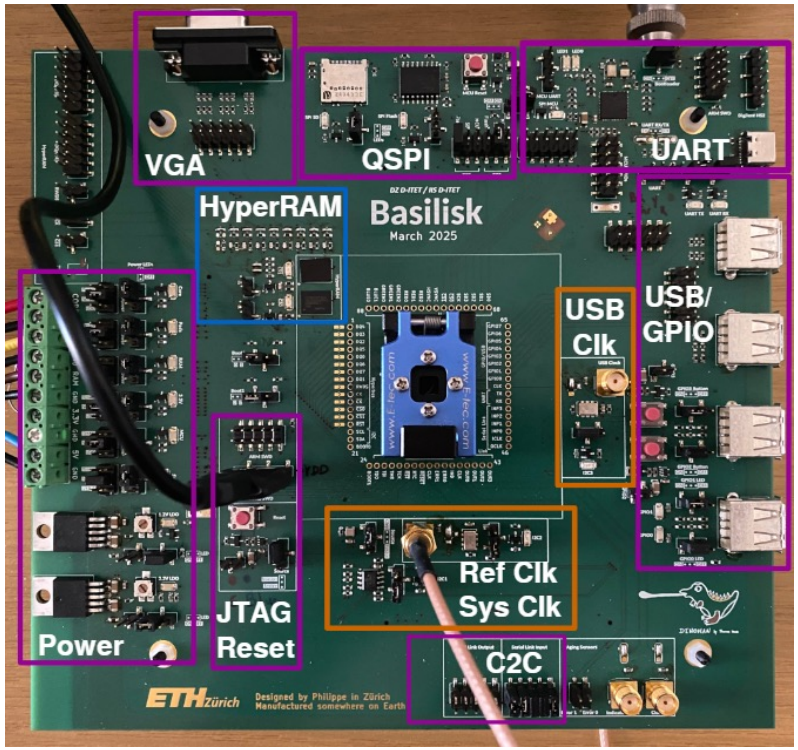


- **Designed in IHP 130nm OpenPDK**
  - **34mm²** (6.25mm x 5.50mm)
  - **~5× larger** than previous end-to-end OS designs
  - 2.7 MGE total, 1.14MGE logic
  - 24 SRAM macros (114 KiB)
  - **62MHz** at nominal voltage (1.2V)
- **RV64GC design runs Linux**
- **Active collaboration with**

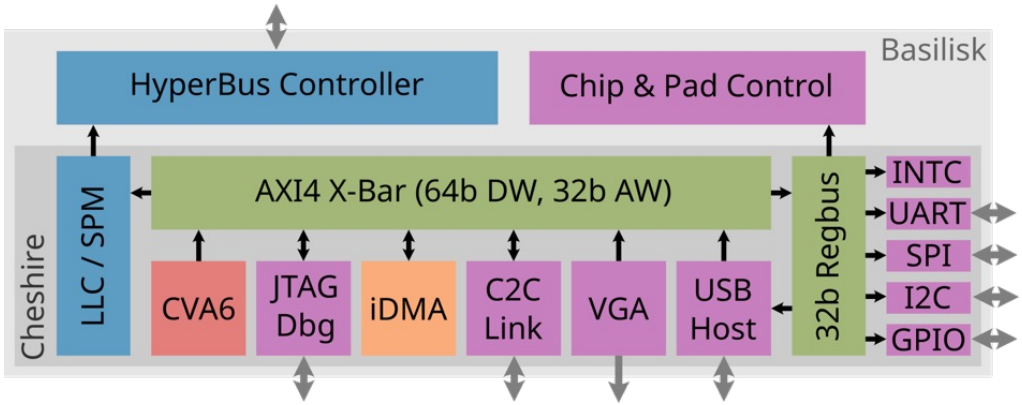


github.com/pulp-platform/cheshire-ihp130-o

Basilisk is a complete Linux-capable SoC

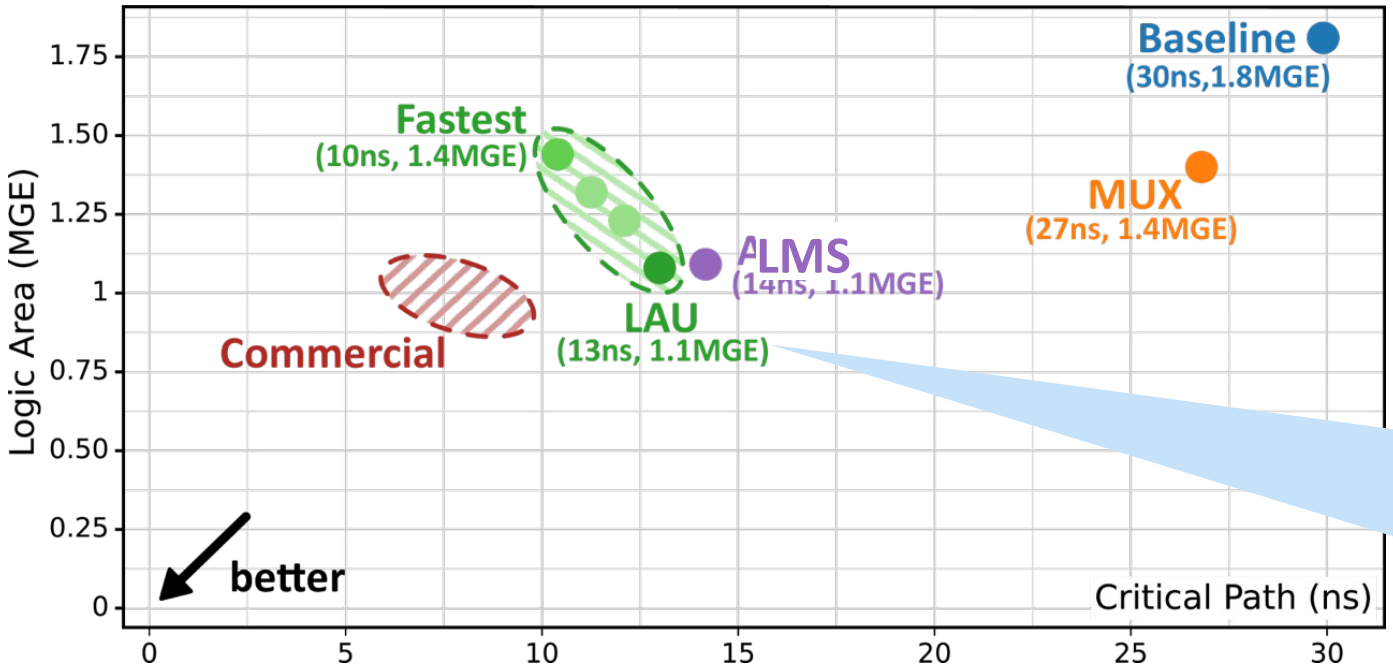


- **64-bit RISC-V core**
- **Rich peripherals:**
  - HyperRAM controller **@154MB/s**
  - C2C AXI-Link **@77MB/s**
- **Automatic boot via scratchpad**



arxiv.org/pdf/2505.10060

What about PPA gap wrt commercial EDA?



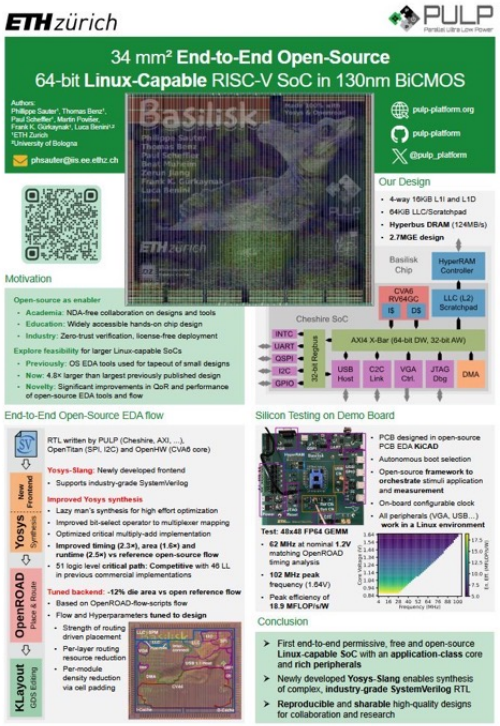
Yosys-slang full Sysverilog Frontend: @ <6sec runtime (from minutes)

Yosys synthesis: 1.1 MGE (1.6×) @ 77 MHz (2.3×), 2.5× less runtime, 2.9× less RAM

OpenROAD P&R: tuning -12% die area, +10% core utilization

Commercial EDA leads, but OS-EDA IS usable, now!

We presented Basilisk at HOT CHIPS



Industry Noticed!

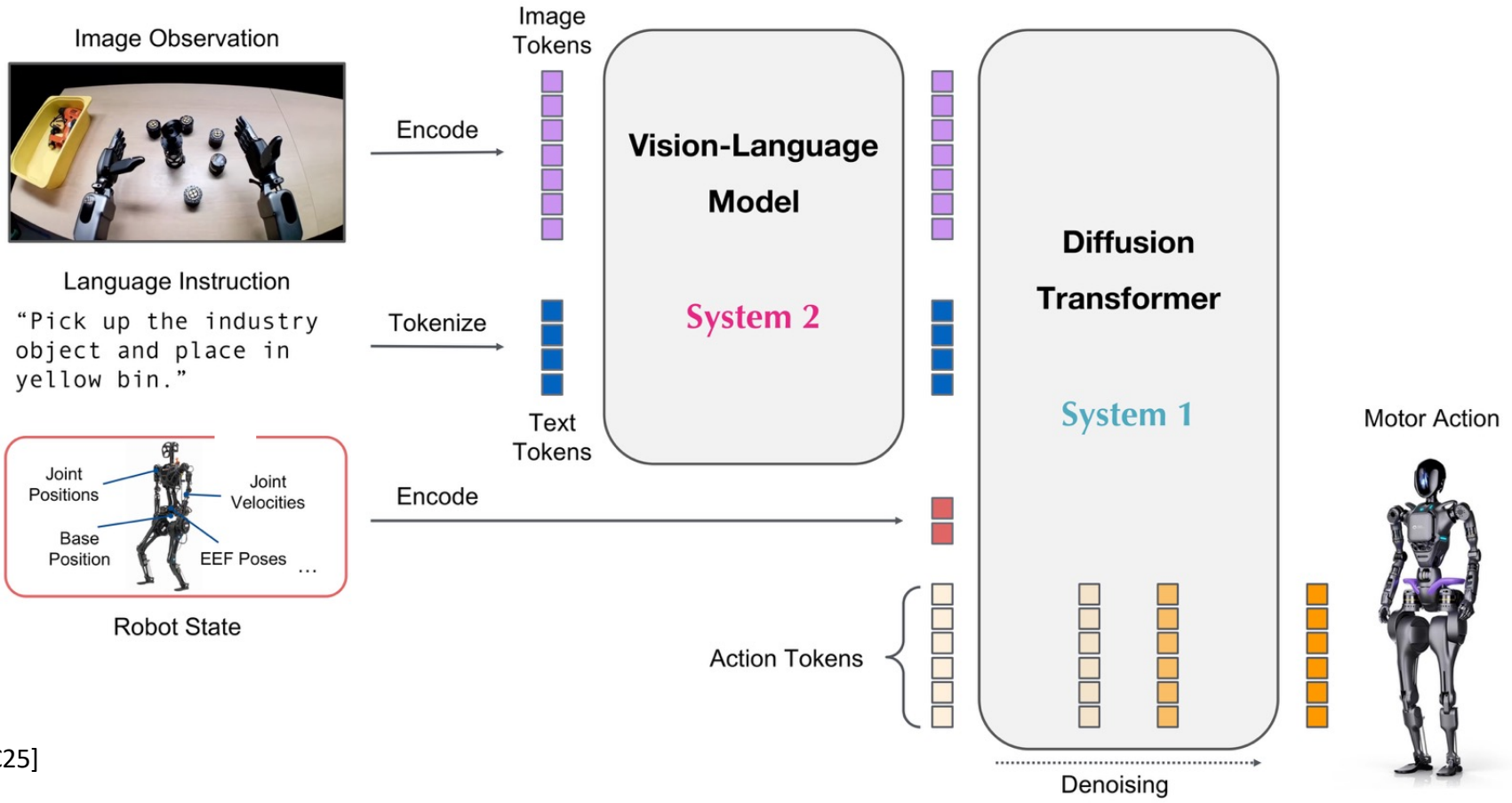
"Basilisk at Hot Chips 2025 Presented Ominous Challenge to IP/EDA Status Quo"*

Poster: lnkd.in/daB6HskB

*semiwiki.com/ip/risc-v/361204-basilisk-at-hot-chips-2025-presented-ominous-challenge-to-ip-eda-status-quo/

ETH zürich | ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

Back to AI: Embodied Gen.AI → Scale & Efficiency



[GTC25]

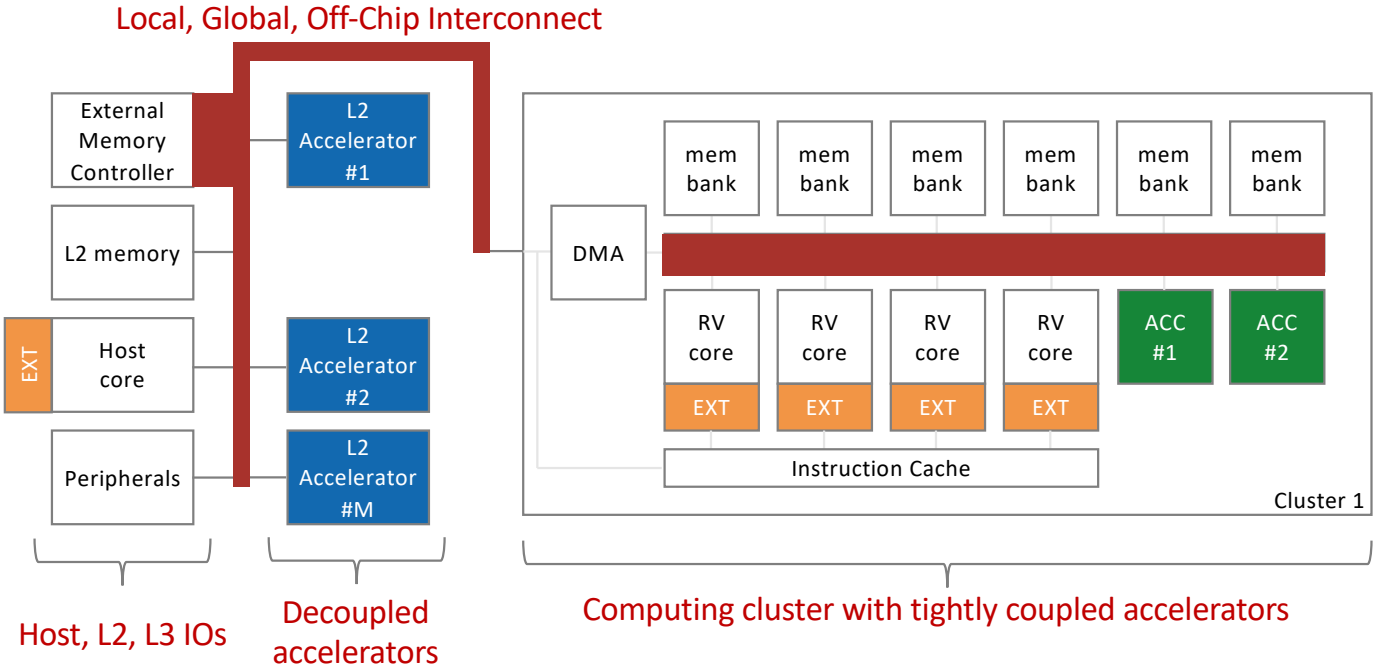
Back to AI: Scaling up DSA

Multiple Scales of acceleration

Extensions to processor cores
- ISA extension
- Share L0 memory

Shared-memory co-processor(s)
- ISA extension or offload
- Shared L1 memory

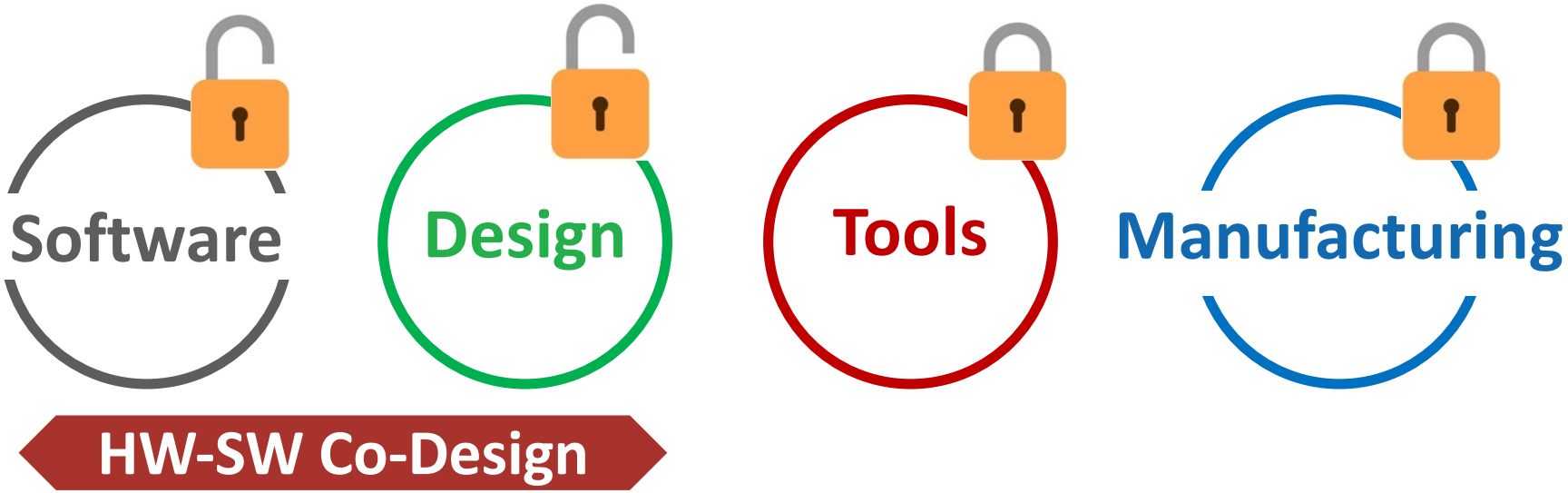
Decoupled Accelerator(s)
- Offload
- Shared L2(+) memory



Local, global, package, system → Specialization at scale

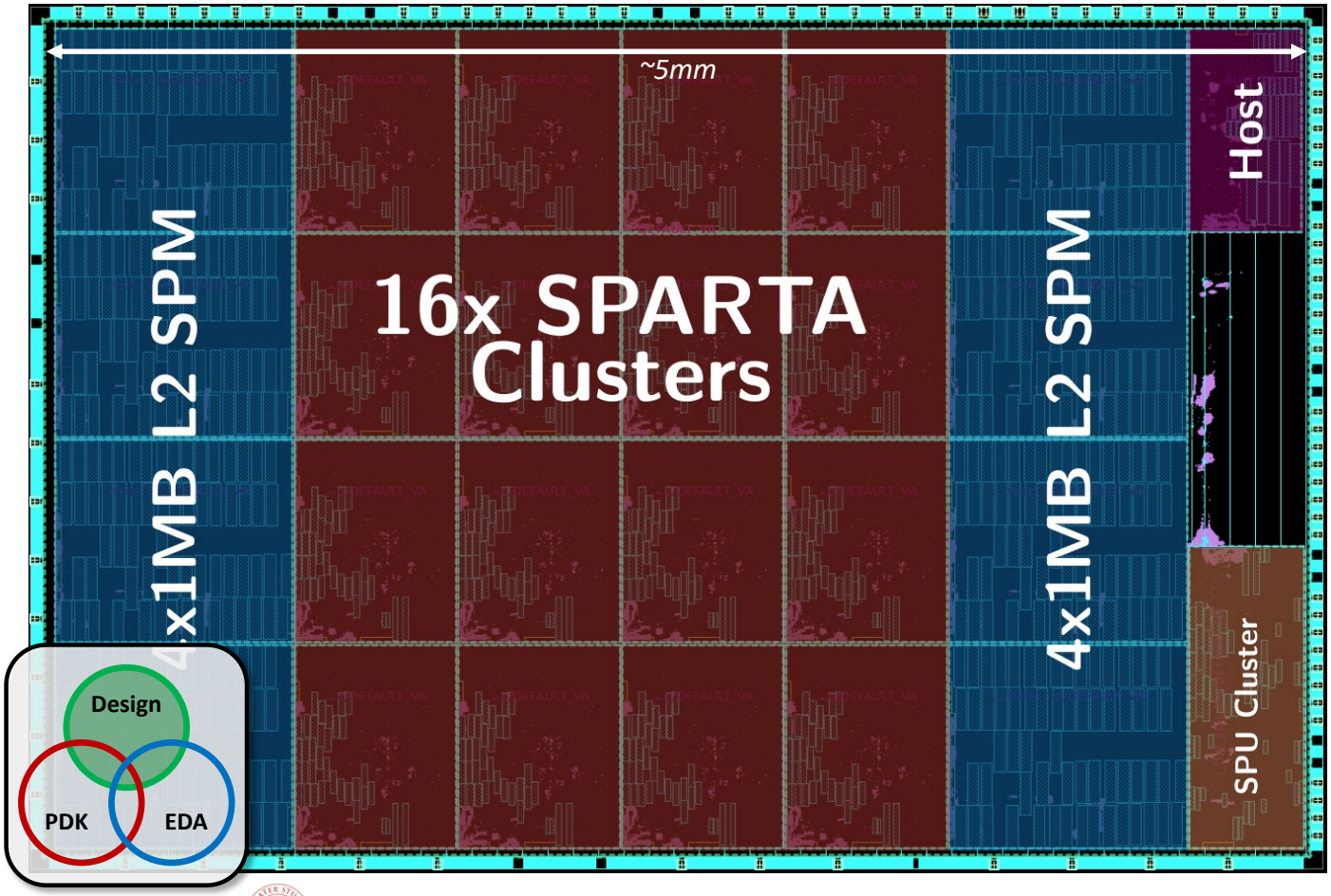
Open, Scaled DSA in Advanced CMOS?

Extreme Performance + Energy Efficiency is required!



Get the best from advanced nodes and closed-source EDA!

Here is Picobello: our latest design in TSMC 7nm



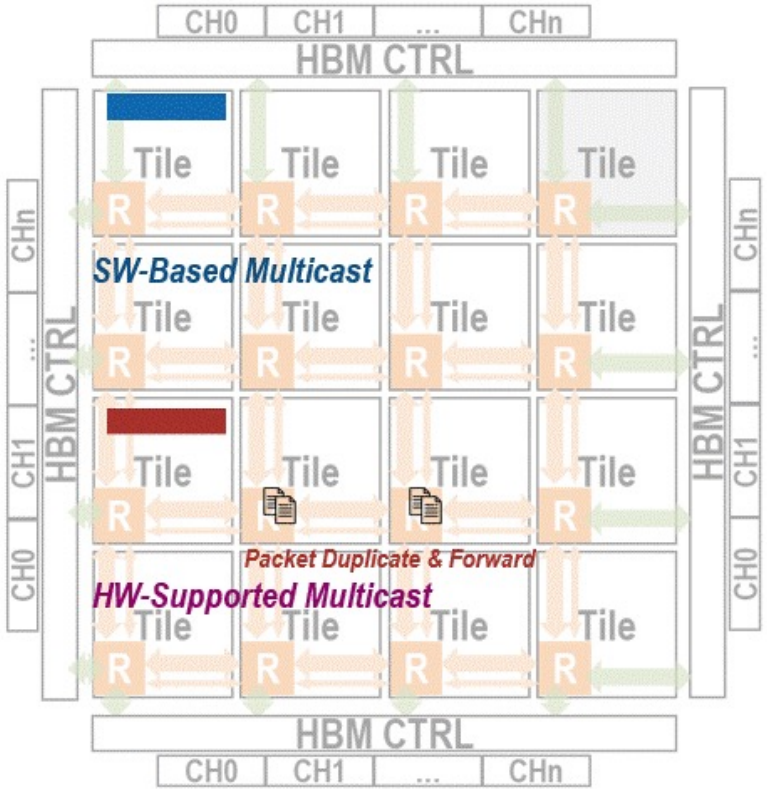
- 16 SPARTA clusters totaling 144x RISC-V cores with FP8-FP64-bit support
- 8x 1MB of on-chip L2
- Linux capable CVA6 Host
- Peripherals (JTAG, SPI, I2C)
- Running at 1+ GHz (WC),
  > 256 GFLOP/s (FP64)
  > 2 TFLOP/s (FP8)
- Tape-out August 2025
- Part of the EU Pilot project

ETH zürich

ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

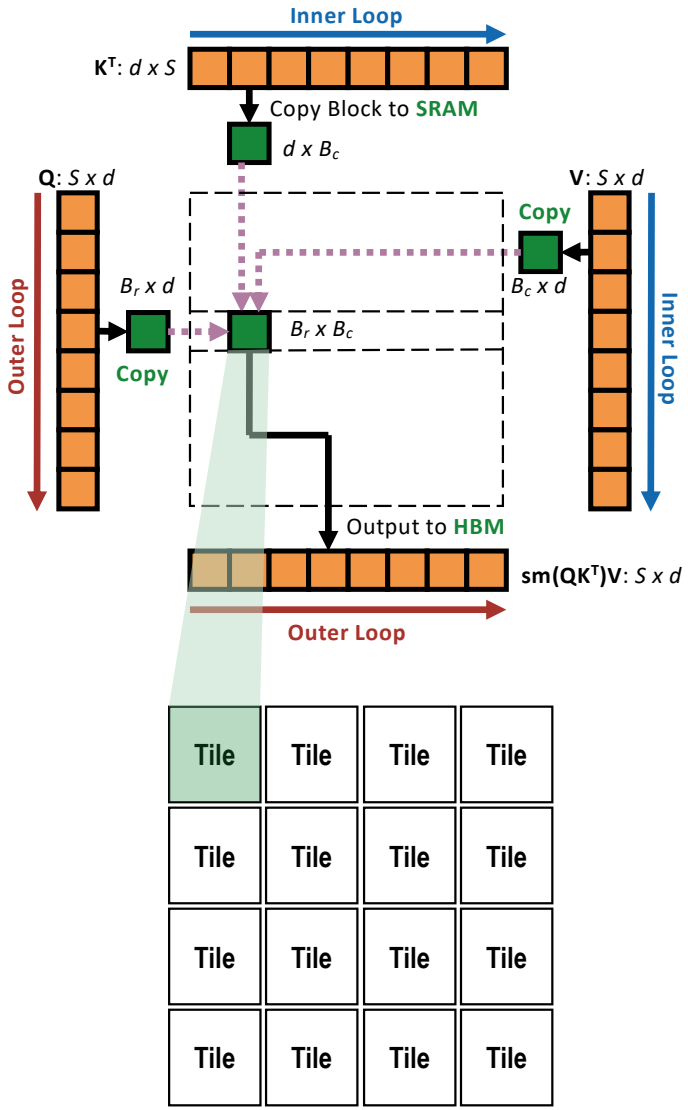
Dataflow and NoC Collective Primitives Co-Design

- **We are targeting attention-like kernels on tile-based many-PE architecture. The goal is:**
  - **Achieve high utilization** of the tiles' matrix engines
  - **Minimize** energy-hungry **off-chip accesses**
- **Architecture and dataflow co-explored**
  - Dataflow leverages **collective primitives on NoC fabric**
    - Accelerate inter-tile collective communications



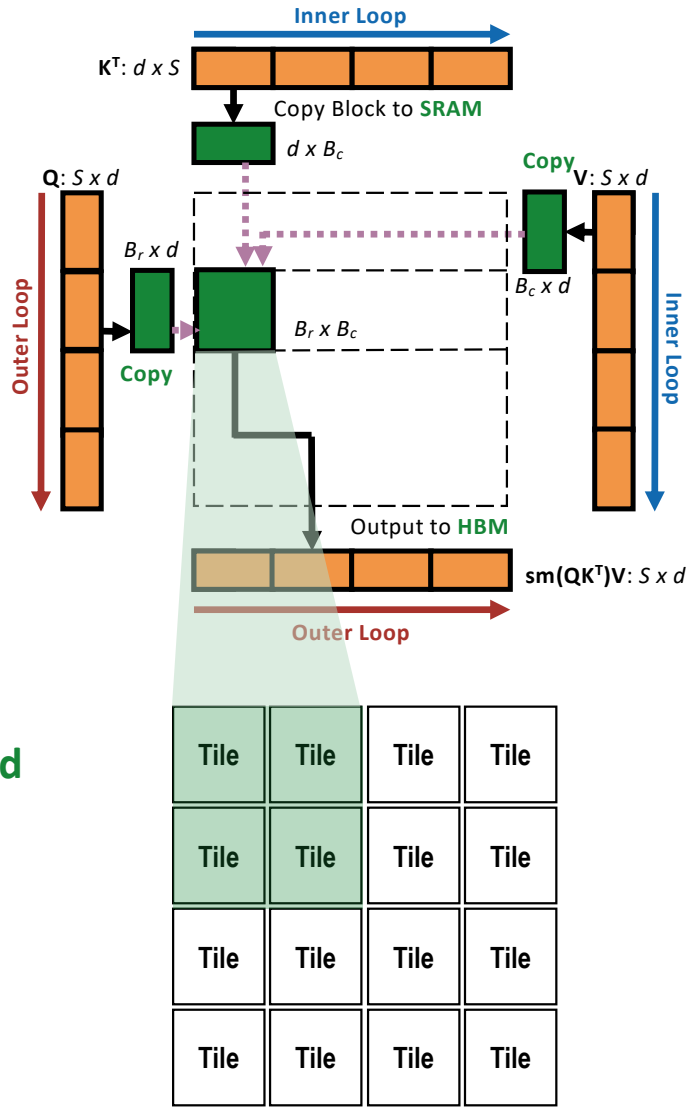
FlatAttention

- **Start from FlashAttention**
  - Fuse microkernels of each head attention
  - MHA workload is partitioned to tiles over:
    - Batch size, number of heads, output sequence dimension
  - **Every tile processes independently**
    - Every tile need to access in HBM
    - No communication between tiles is required
  - Results in an HBM I/O complexity of
    - $\mathrm{IO} = 2 \cdot H \cdot B \cdot D \cdot S \cdot \left(1 + \frac{S}{M}\right)$
    - Sequence length **S**, head dimension **D**, number of heads **H**, batch size **B** and block size **M:=Br=Bc**



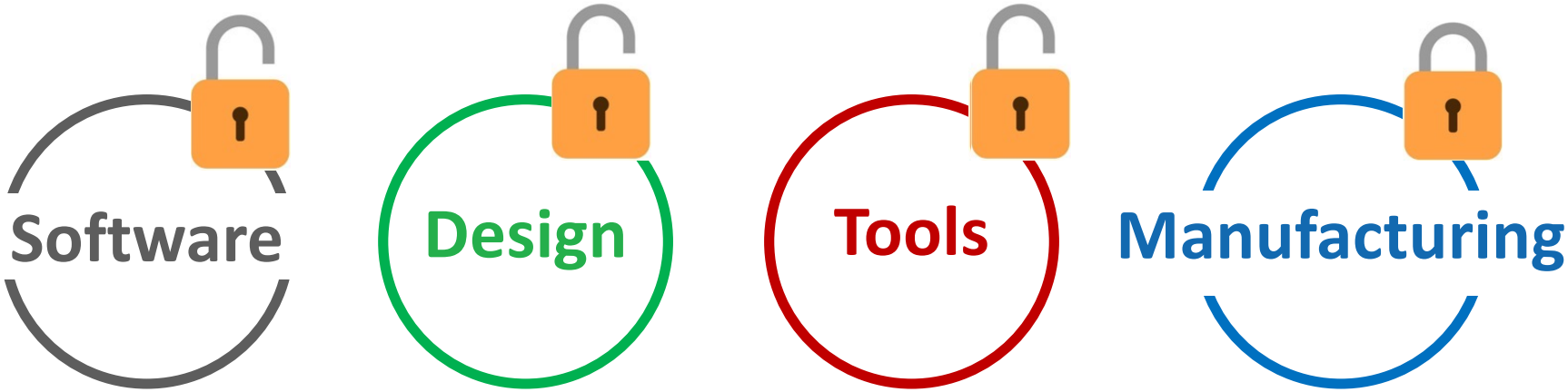
FlatAttention Motivation

- **FlatAttention – Redefine how MHA is parallelized**
  - **Leverages multiple tiles as a unified entity**
  - Process an **MHA block of a significantly larger size**
    - The aggregate L1 memory of a group of tiles
    - Collectively store the block
  - When **N tiles are grouped** together, HBM I/O complexity:
    - $\mathrm{IO} = 2 \cdot H \cdot B \cdot D \cdot S \cdot \left(1 + \frac{S}{\sqrt{N} \cdot M}\right)$
  - **More tiles are grouped, less HBM accesses needed**



Open EDAs for open Scaled DSA in Advanced CMOS?

Extreme Performance + Energy Efficiency is required!

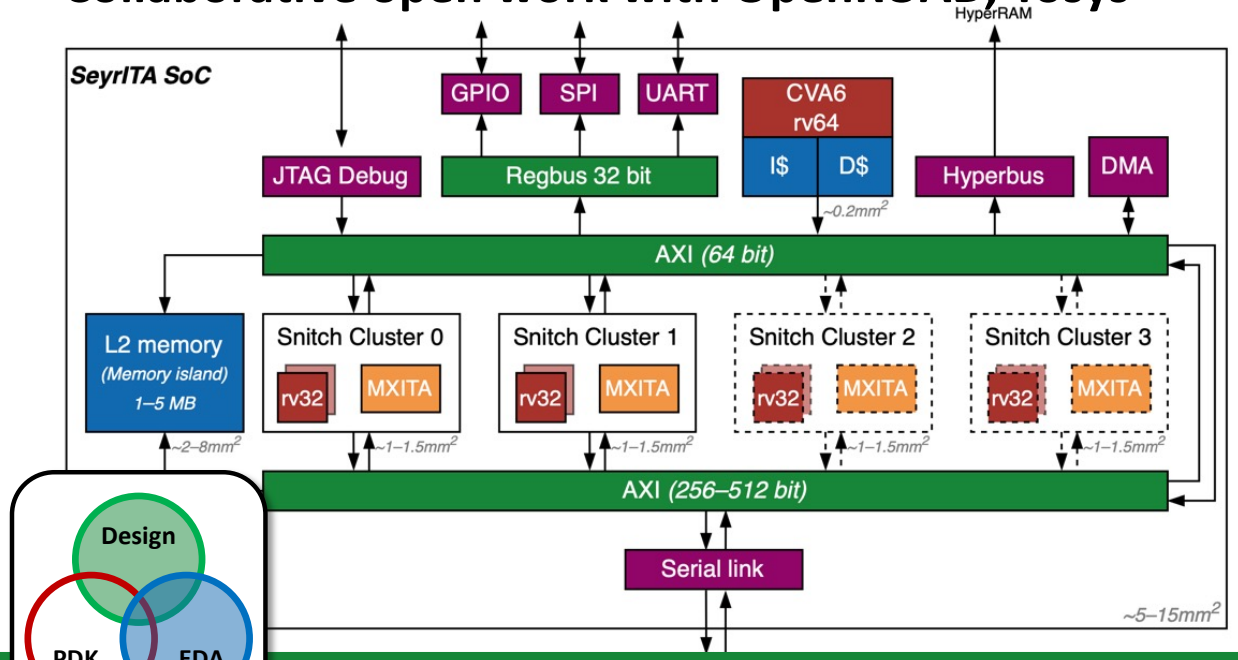


HW-SW Co-Design

HW-SW & Tools Co-Optimization

SeyrITA, Scaled DSA for Gen.AI **with open EDA**

- **Designing a research relevant SoC using open source EDA in GF22**
  - State of the art accelerator for embodied AI, using Integer Transformer Accelerator (ITA)
- **Collaborative open work with OpenROAD, Yosys**



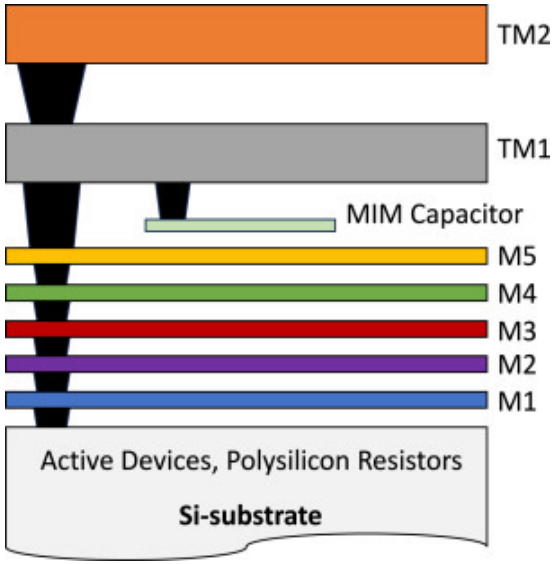
https://github.com/pulp-platform/ita

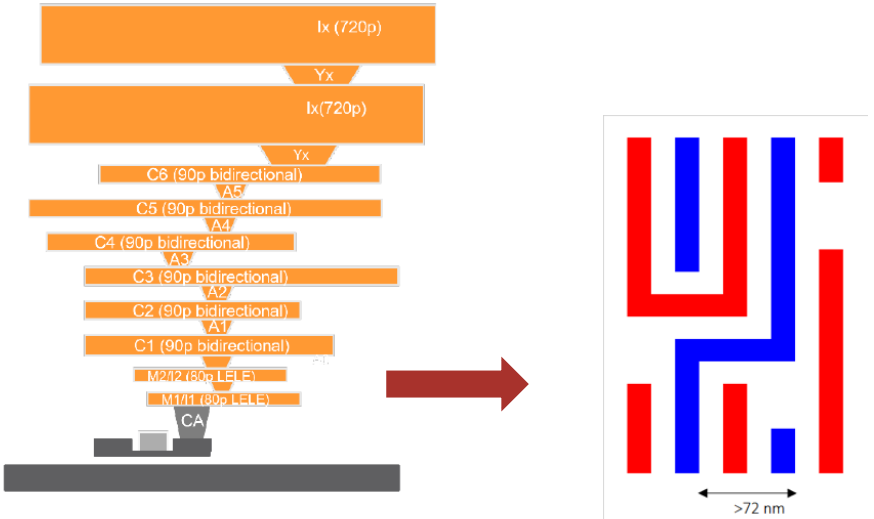
GlobalFoundries™

- **Challenging work**
  - Large design, modern technology
  - We encounter problems daily
  - We try to solve them one problem at a time
  - Confident we will get it done
- **Target tape-out**

  2026

Old Versus Modern Nodes: Metal Stacks

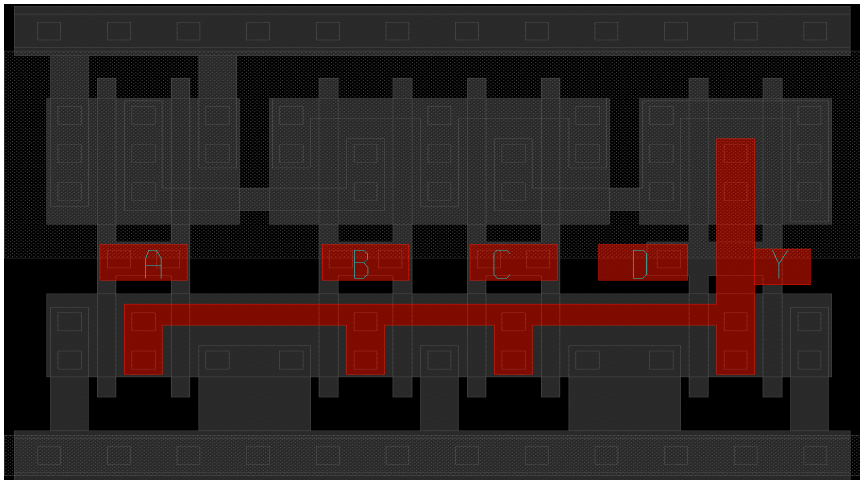




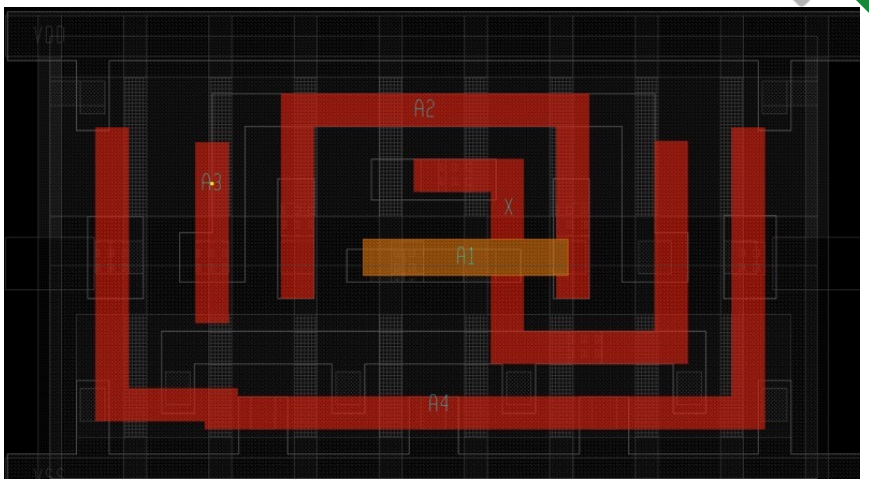
- **IHP130 metal stack**
  - 7 metal layers with two width groups
  - 420nm M1 pitch

- **GF22FDX metal stack**
  - 10 metal layers with 3 width groups
  - 80nm M1 pitch
  - M1, M2 need double-patterning →colored routing

Old Versus Modern Nodes: Standard Cells



IHP130 NOR4 Cell 5.76µm x 4.22µm



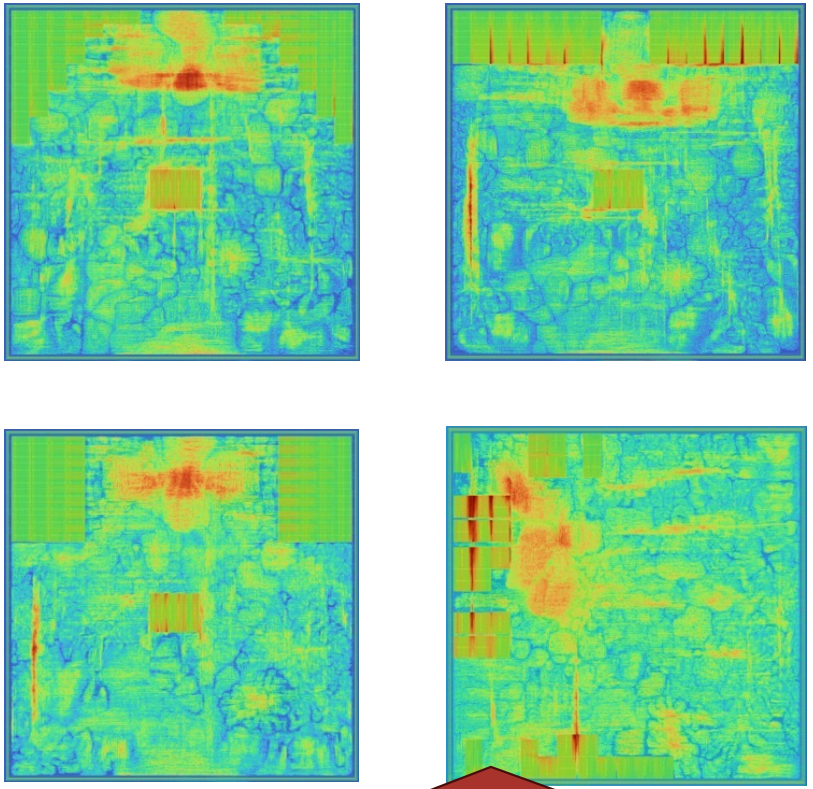
GF22 NOR4 Cell YYµm x ZZµm

- **IHP130 Cells**
  - Larger, with lower density
  - Simple pin shapes
- **GF22FDX Cells**
  - Smaller and denser
  - Pins on multiple layers
  - Irregular pin shapes

Much more complex Synthesis and P&R tooling!

Working on SeyrITA Tapeout

- **First** large **22nm tapeout** with open-source tools
- **Improve tools** and close the **performance** gap
- Identify and **implement missing features** along the way
- **Active Collaboration with**



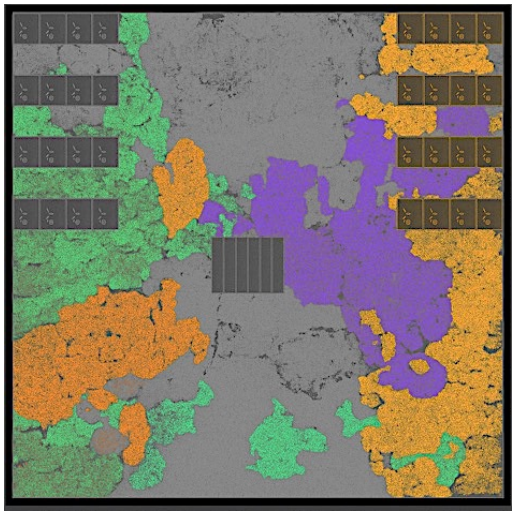
Cluster floorplan exploration

ETH zürich ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

Significant Improvements in QoR

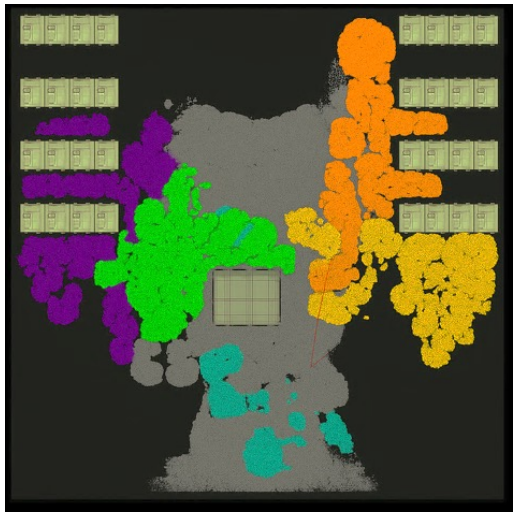
- In addition to tool fixes and improvements, **aggressive hyperparameter tuning**
- All leading to **56% higher frequency** and **42% area reduction!**

Baseline



231 MHz – 7.7 MGE

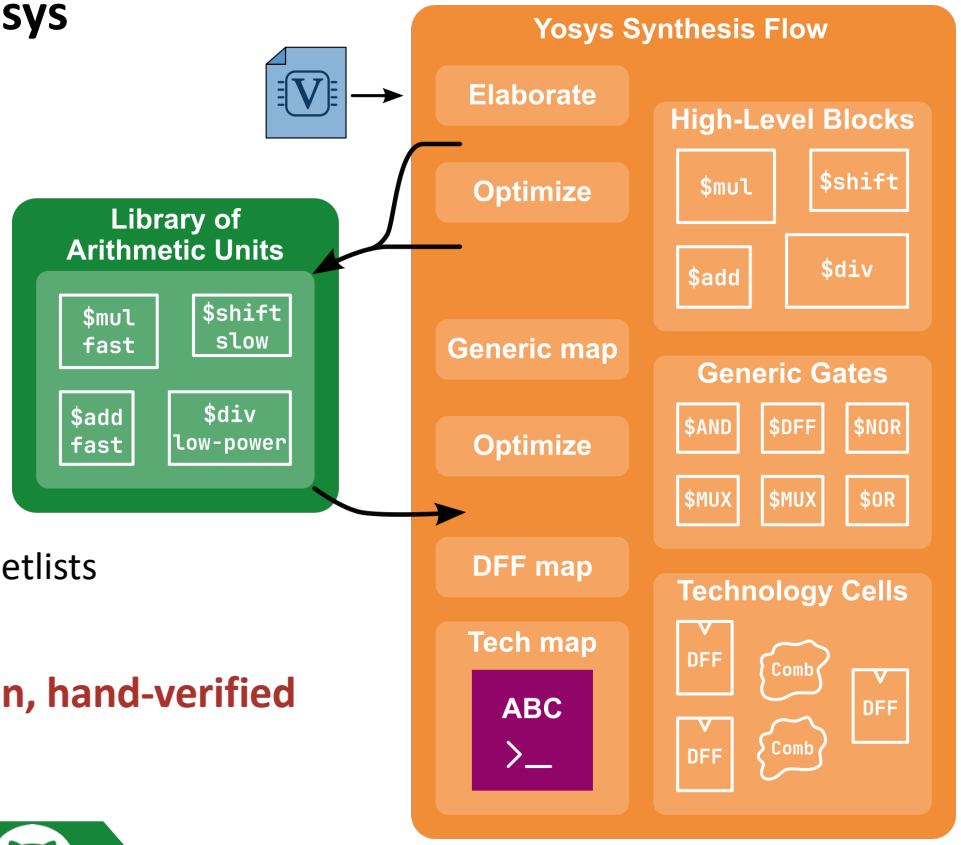
Optimized



360 MHz – 4.5 MGE

Not only Back-end! Library of Arithmetic Unit (LAU)

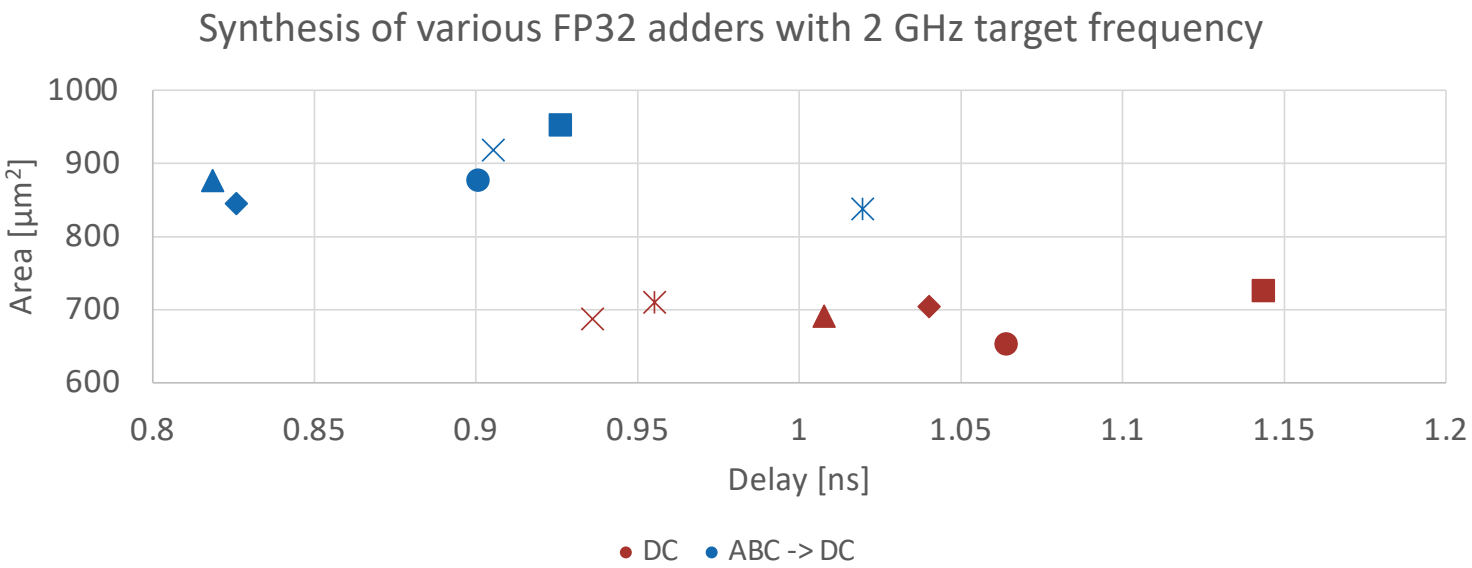
- **Block replacement is implemented in Yosys**
  - Detect and replace arithmetic operators
  - Currently: manual selection
  - **Next: algorithmic, AI based!**
- **No open-source LAU**
  - Rich, optimizd library is key to good results
  - We built it
    - A wide range of arithmetic operations
    - 3 different performance variants of generic gate netlists
    - Thoroughly QoR evaluated and optimized
  - **SystemVerilog port from VHDL: LLM translation, hand-verified**



github.com/pulp-platform/elau

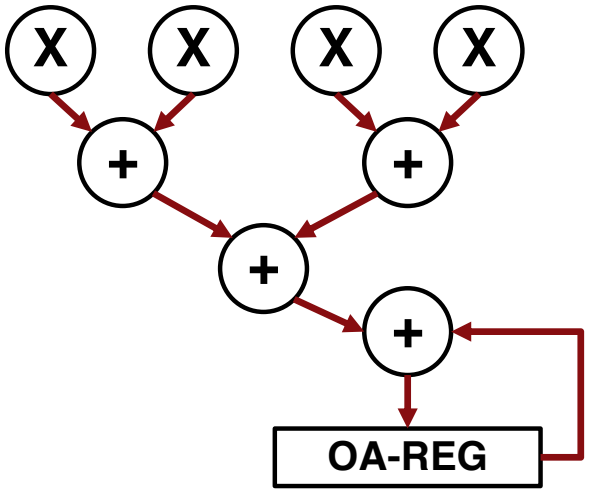
There is still room for improvement in Synthesis!

- Explored various FP32 adders:
  - Applied ABC logic optimizations before Synopsys Design Compiler synthesis, leading to higher frequency Pareto points for several designs.

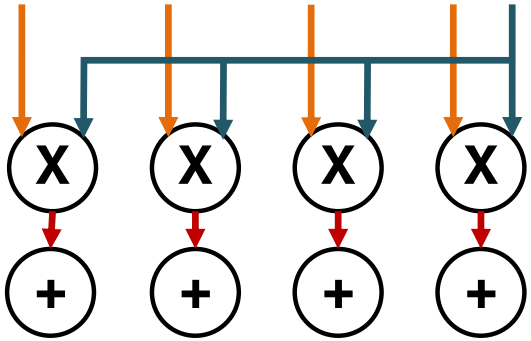


Each marker represents a different design

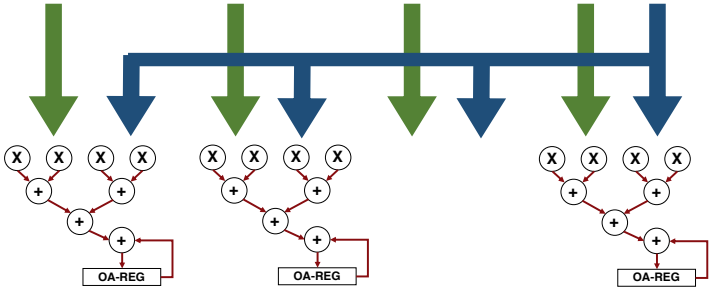
Specialization + EDA multiplicative effect



Inner Product



Outer Product



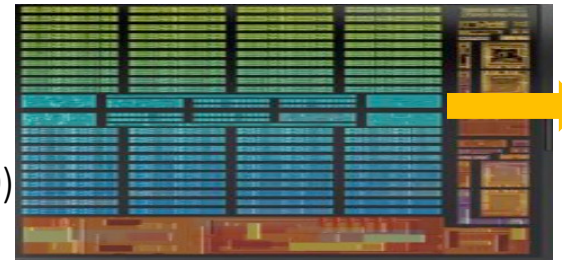
Mixed

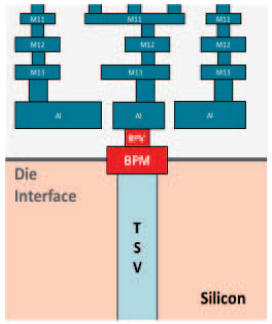
Precision tuning – OP/Mem tuning - deep arithmetic optimization – operand network tuning...

Co-Specialize SW, HW, EDA & Technology is the frontier

What Happens Next?

- 3.5D v1
  - 3D stacking on logic + 2.5D HBM (AMD MI300)
  - Face (top) to Back (bottom)
  - Die (top) to Wafer (bottom)

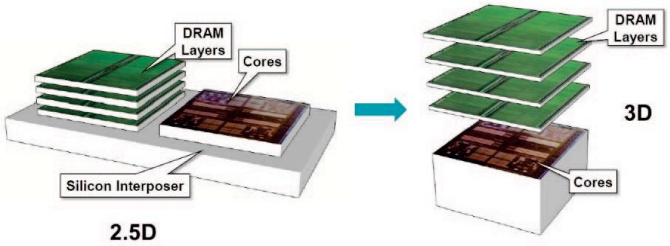


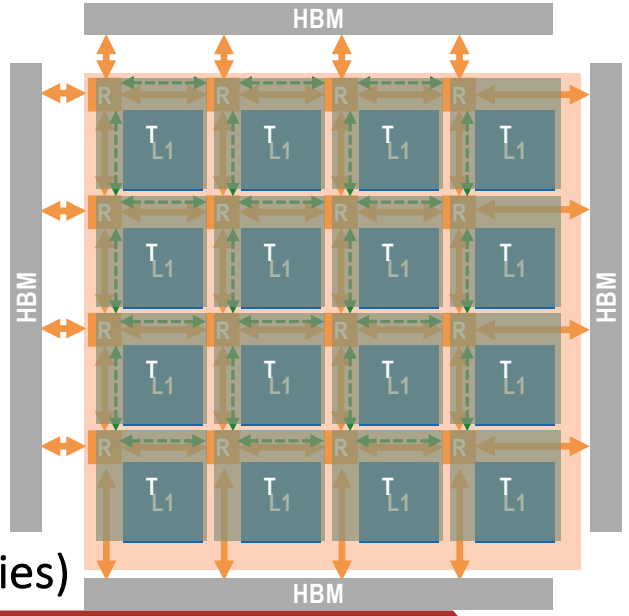


MI300 Instinct™

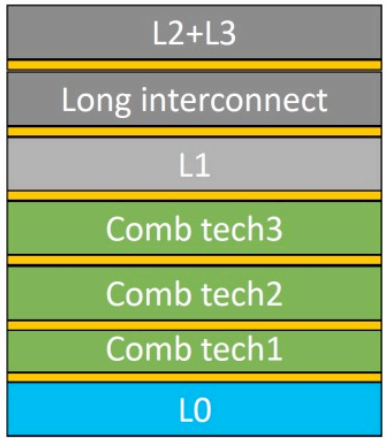
Logic die
Memory + IO die

- 3.5D v2?





V1
SRAM+NOC+IO at the bottom



- Monolithic 3D (CMOS2.0+3D memories)

Technology is going "full 3D", OS-EDA is right there*

*github.com/lamda-bbo/Open3DBench


http://pulp-platform.org
@pulp_platform
The future is bright for open DSA!