ETH zürich

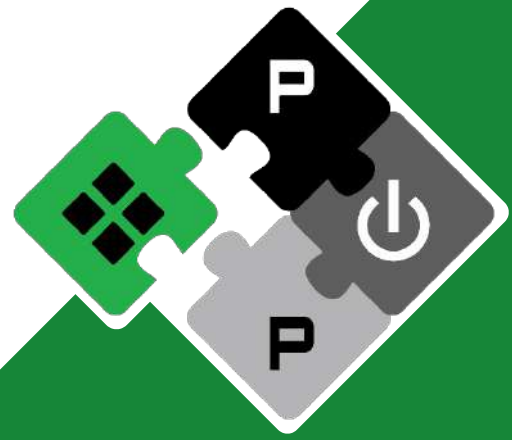
ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Leveraging an open ISA from Handcrafted to Agentic Design

Luca Benini

lbenini@iis.ee.ethz.ch

luca.benini@unibo.it



@pulp_platform

pulp-platform.org

youtube.com/ pulp_platform

PULP Platform

Open Source Hardware, the way it should be!

Physical AI: Artificial Intelligence Everywhere

Humanoid



[Unitree26]

Interceptor Drone



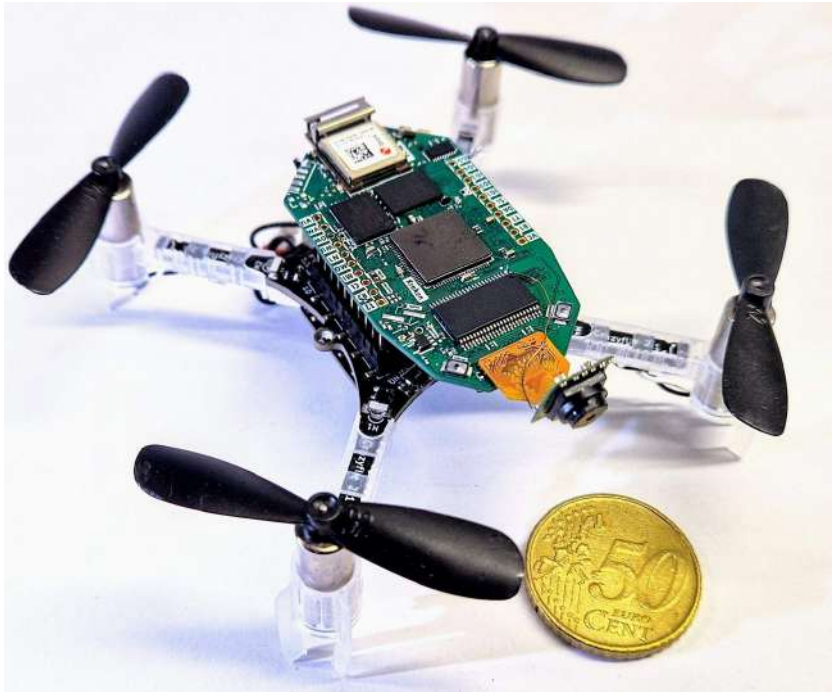
[GeneralCherry26]

Humanoid, Interceptor Drone
P_{avg} < 200 W*

*[research.mobiusriskgroup.com/p/the-energy-diet-of-humanoid-robots]

Physical AI: Artificial Intelligence Everywhere

Nano-Drone



Cubesat



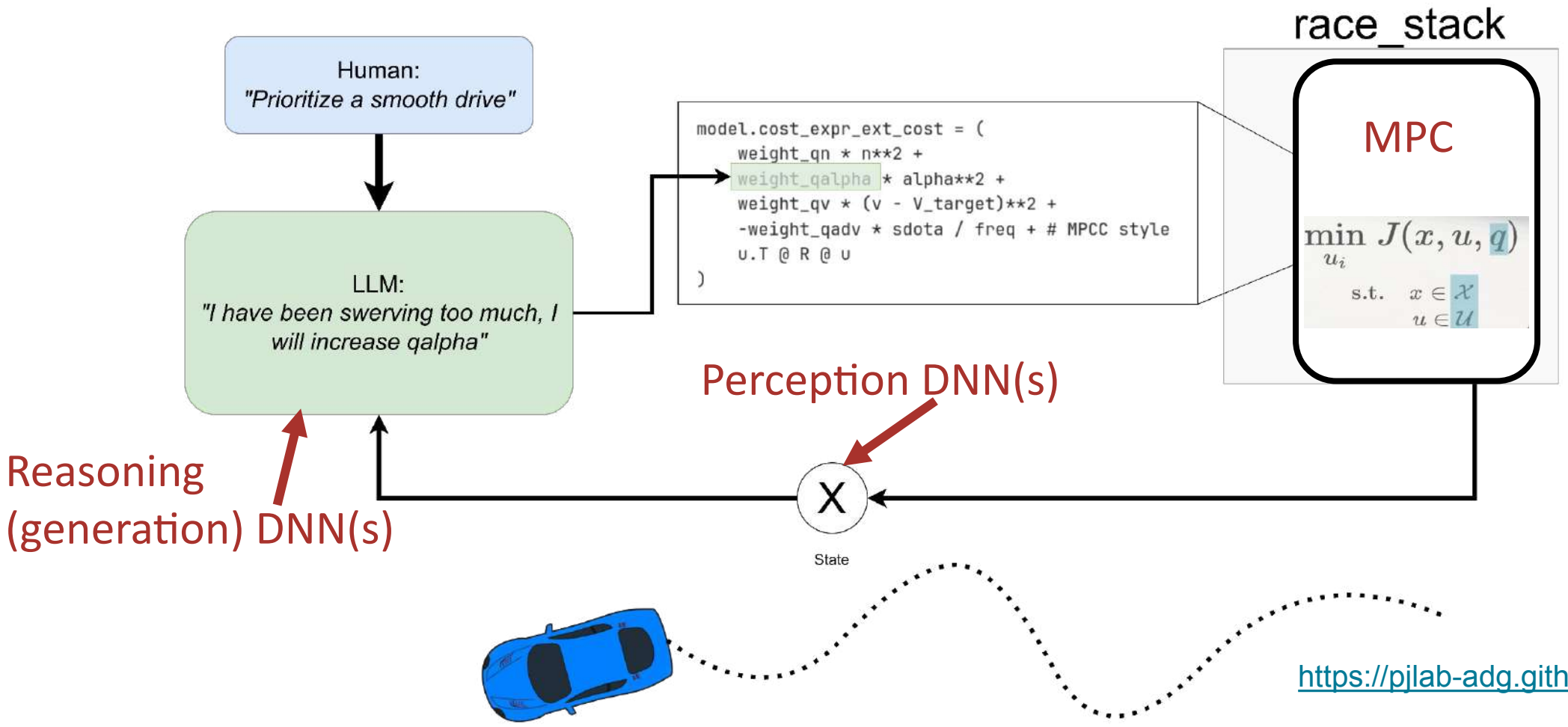
Cubesat
P_{avg} < 1.5 W

Nanodrone
P_{avg} < 150 mW

Efficient and Reliable operation key for Phy.AI!

Perception and Agentic Reasoning in Physical AI

LLM Reasoning on Human Commands & Robot Observations

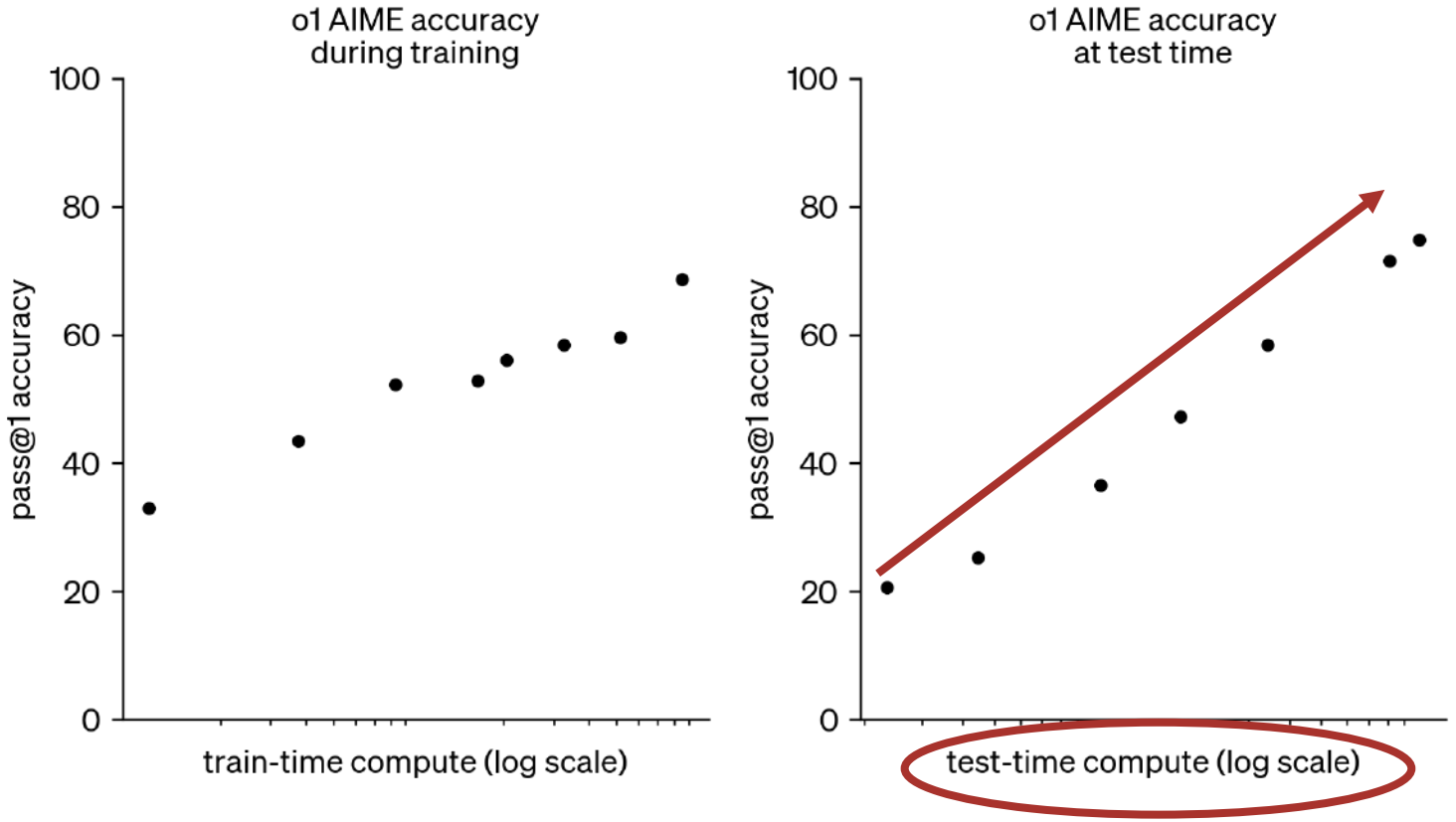


https://pjlab-adg.github.io/DiLu/

Bigger Models, Bigger Bottlenecks **@ Inference**

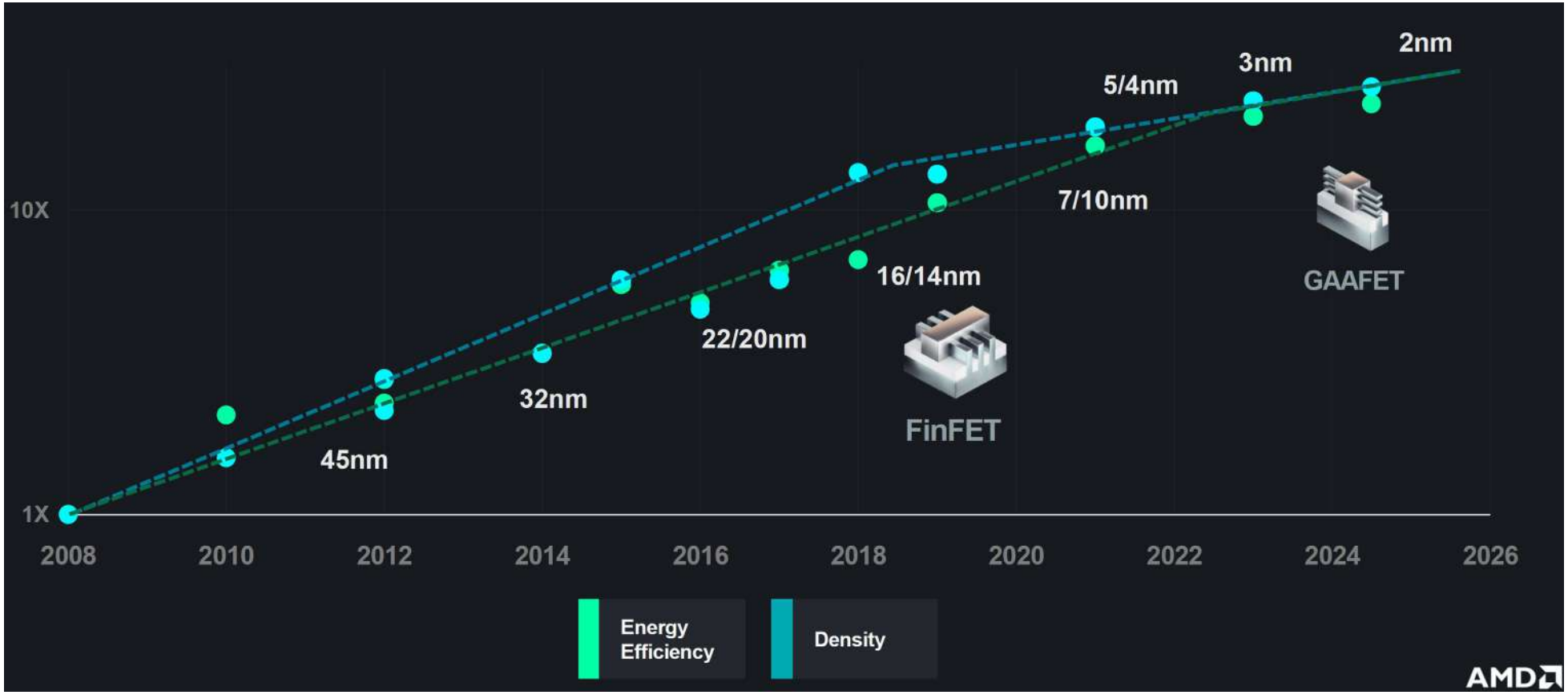
*Deep learning models continue to grow in **scale** and **complexity***

- Growing model sizes demand ever-increasing compute and memory
- Agentic inference compute scales faster than training



[openai.com/index/learning-to-reason-with-llms/]

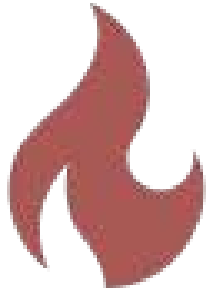
Efficiency Challenge: Moore is not Enough!



[AMD HotChips24]

Model complexity
10× every **~2.5** years

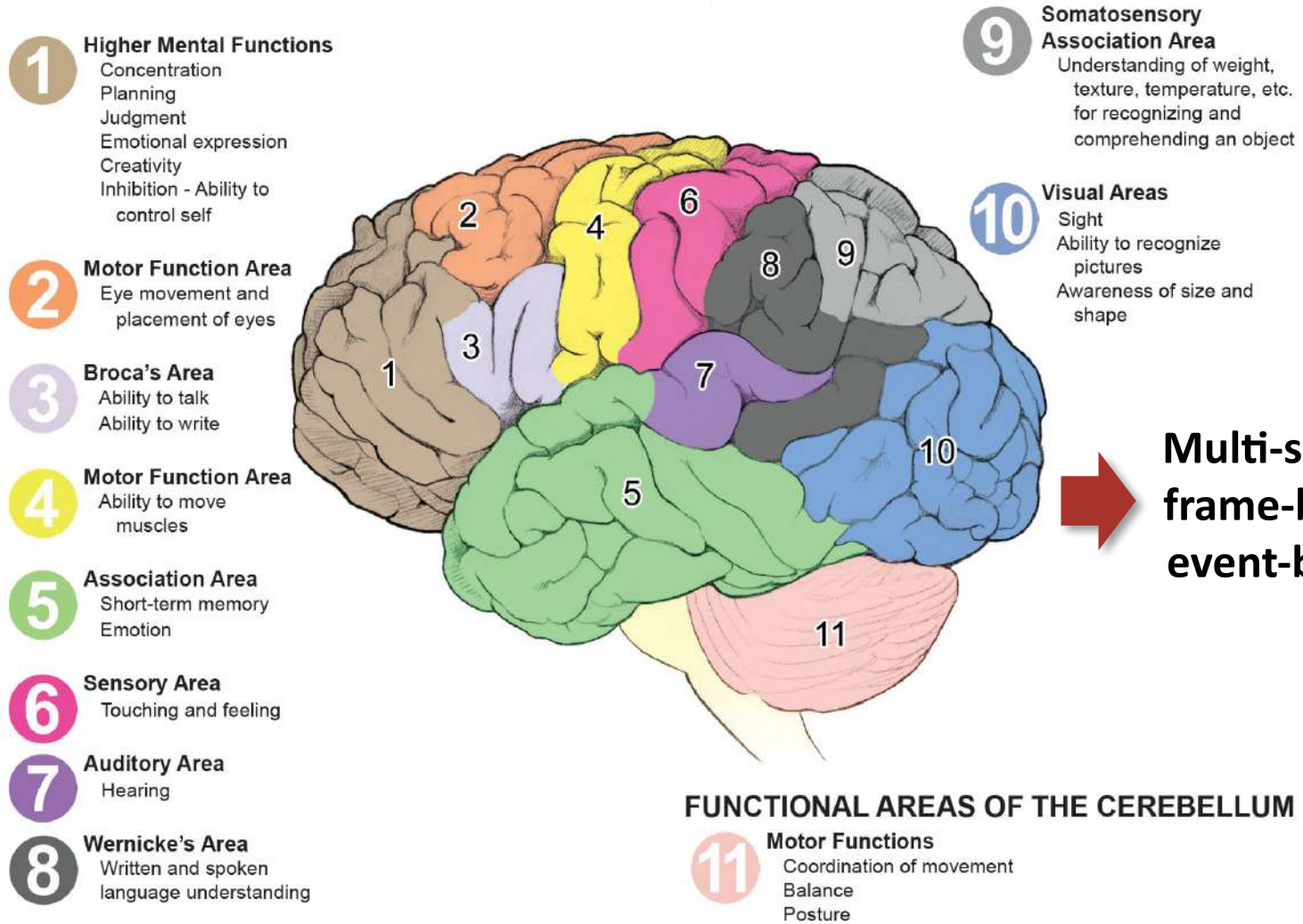
Moore's Law
10x every **12 years!**



Algorithm, Architecture, Design

Efficiency via Domain Specialization

Brain-inspired: Multiple areas, different structure different function!

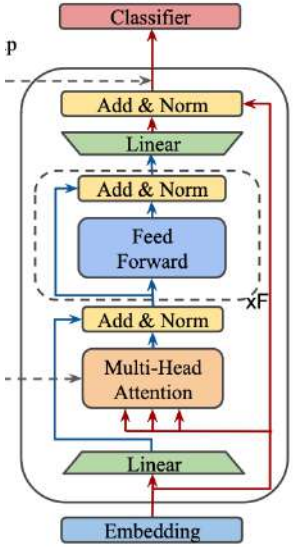


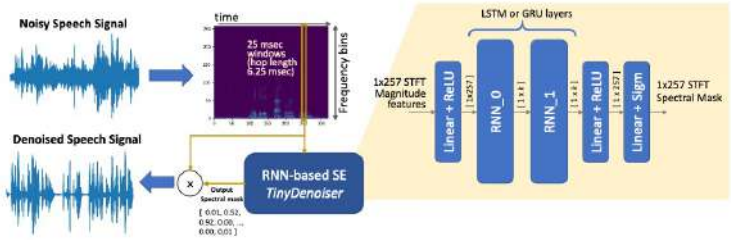
Multi-sensor frame-based event-based → **Perception** → **Multi-modal Agentic reasoning**

A Fast-Evolving Model Zoo

[Z. Sun et al.] MobileBERT

Encoder Transformer





[M. Rusci et al.] TinyDenoiser

Recurrent models

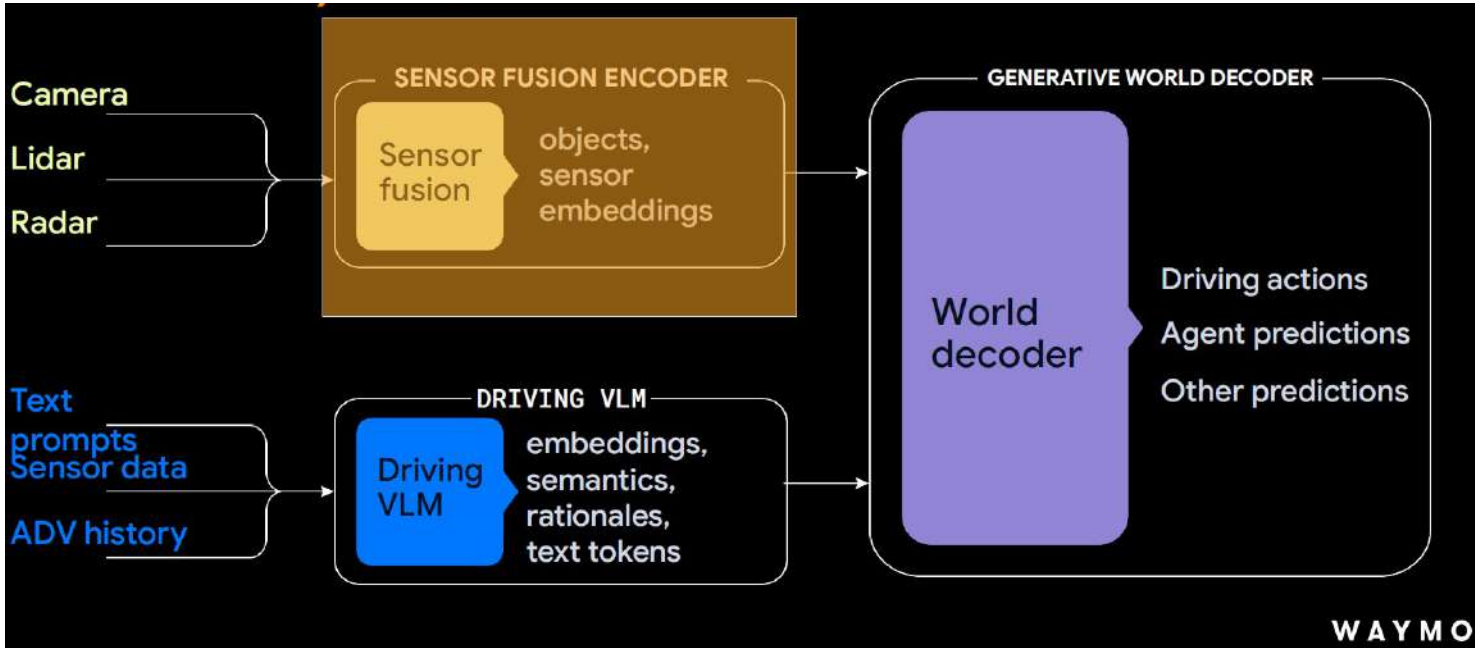
State based models



[M. Oquab et al.]

Multimodal Transformer

[Waymo HC26] **Multi-stage Multi-modal**

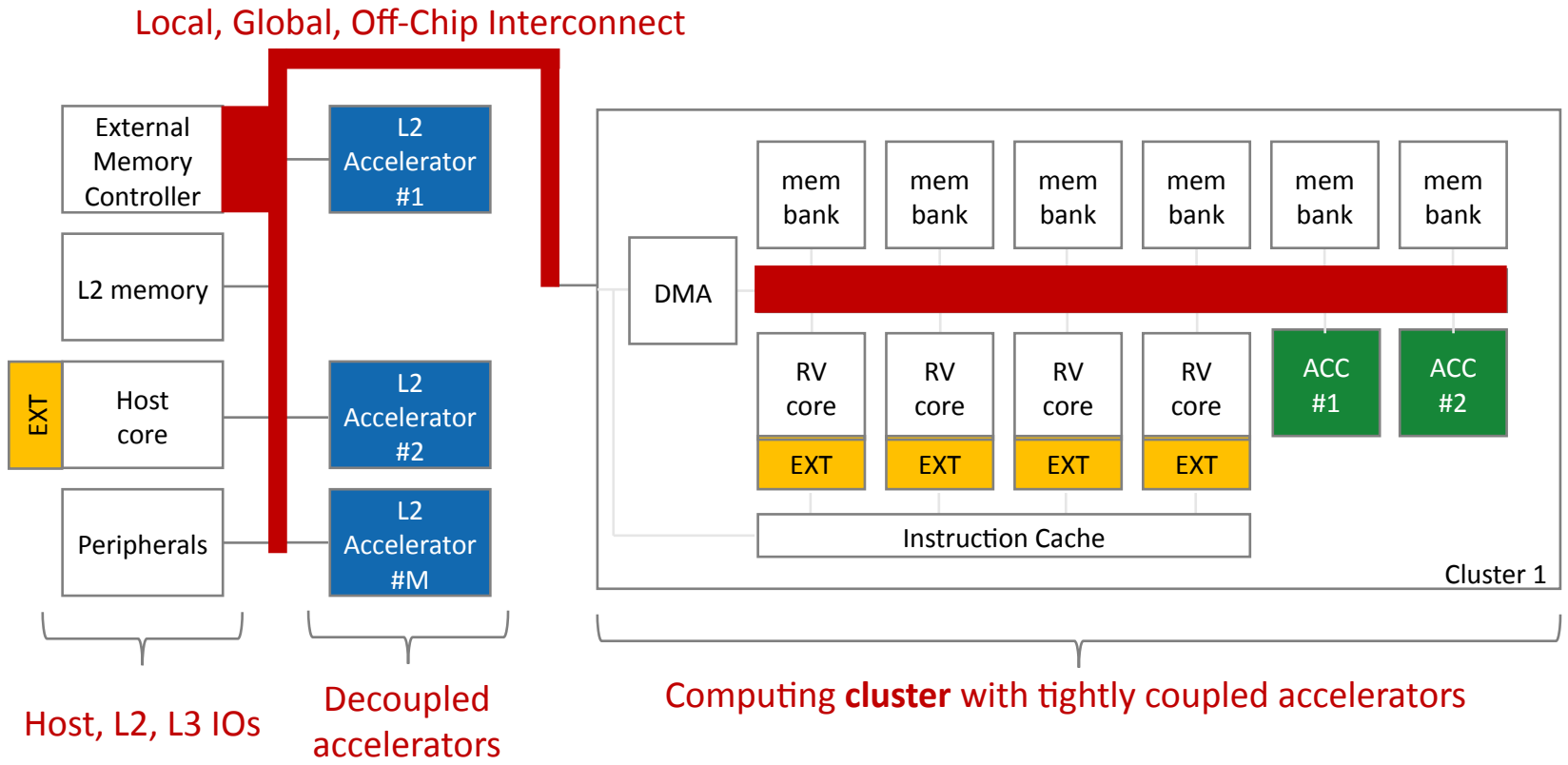


Must run on the specializazion ridge: specialize fast, but not too much!

Flexible Domain-Specific Architecture (DSA)

Multiple dimensions for Specialization

- Extensions to processor cores
  - ISA extension
  - Same L0 (RF)
- Tensor Engines
  - ISA extension or fast offload
  - Decoupled L0
- Decoupled Accelerators
  - Explicit Coarse Grained offoad
  - Decoupled L1
- Specialized Mem and Interco
  - Multicast/reductions/compression
  - Compute near/in mem & NoC



***Specialization + Scalability,* the two keys!**

Specialization in Perspective: Amdahl's Law

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

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

↓

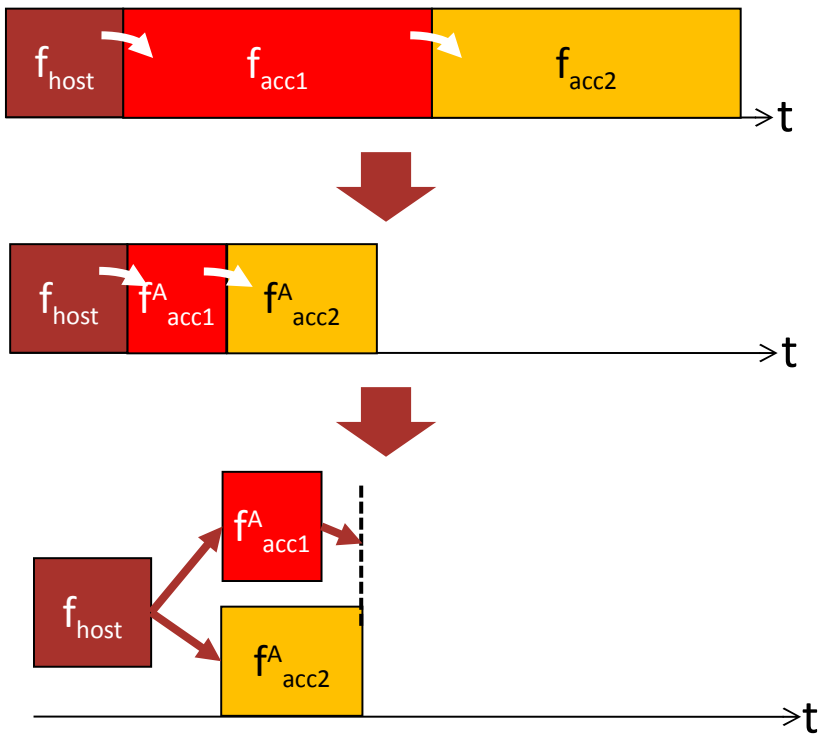
ISA-based 10-20x → **1pJ (4bit)**

↓

Tightly coupled 10-20x → **100fJ (4bit)**

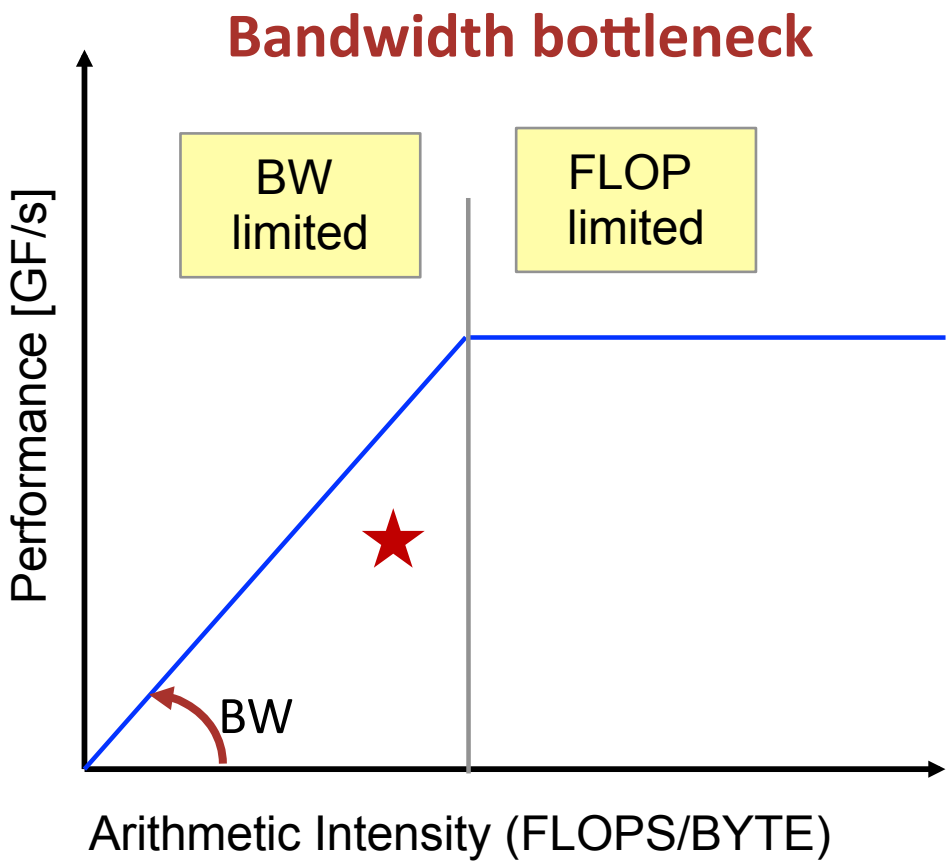
↓

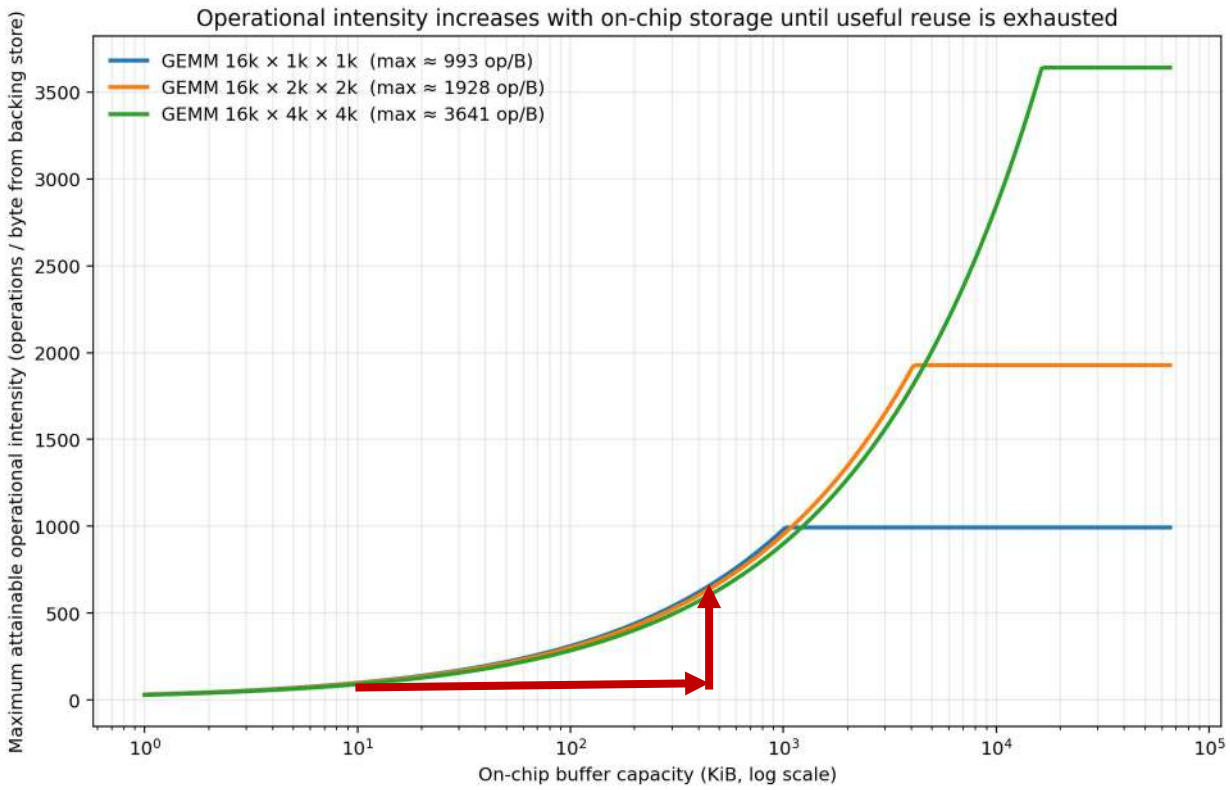
Decoupled 100x → **1fJ (ternary)**



Specialization game: efficiency vs. utilization Tradeoff

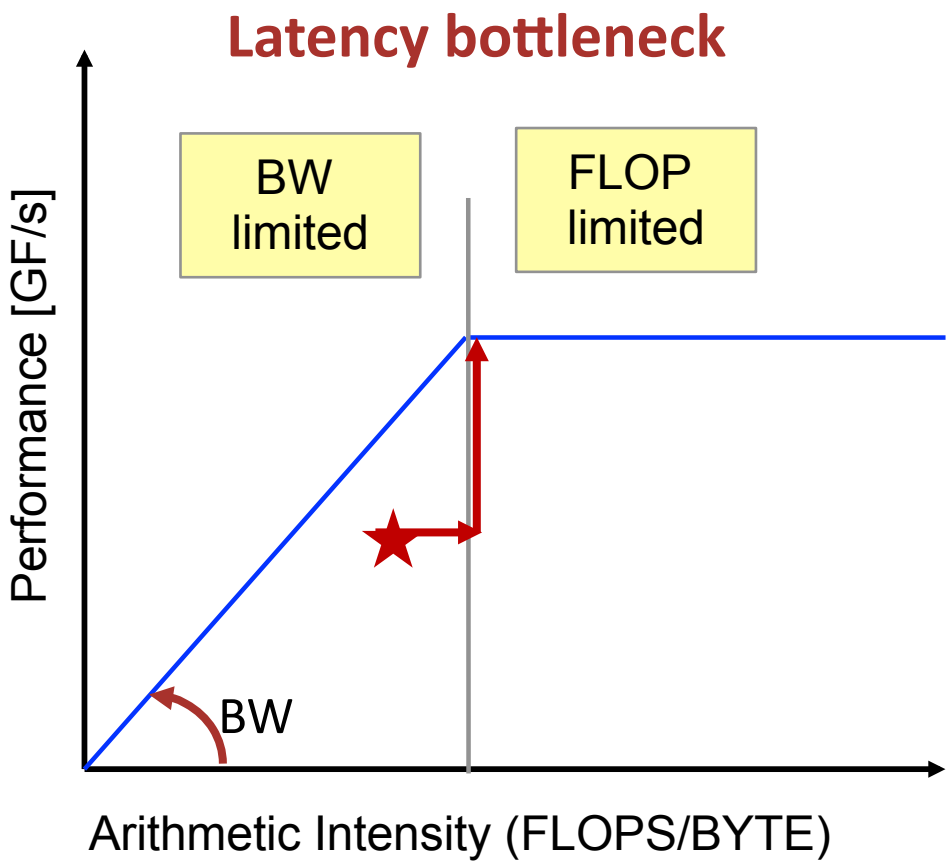
Scalability in Perspective: Patterson (roofline), Little Laws





On-chip buffering increases arithmetic intensity

Scalability in Perspective: Patterson (roofline), Little Laws



Assuming compute is fast enough, buffer has to be big enough to hide tile transport time

↓

Eg. 1KB/ns x 100 (ns) → 100KB

↓

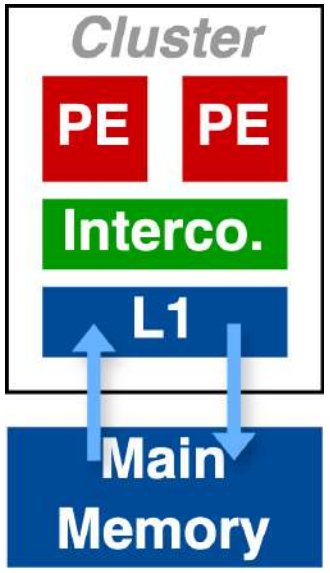
2x for double buffering

↓

BufferSize (B) ≥ 2 x Bandwidth (B/s) x Latency (s)

Need IO-bandwith, on-chip Buffering, and latency tolerant Architecture

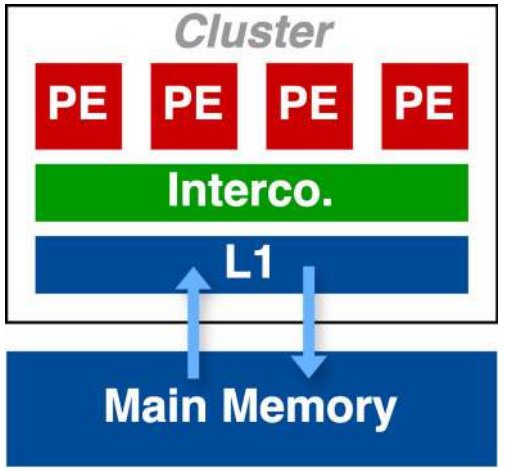
Cluster Scaling: *Scale-Up* vs. *Scale-Out*



Cluster: A group of **PEs** tightly coupled with a shared **L1 memory** through low-latency **interconnect**

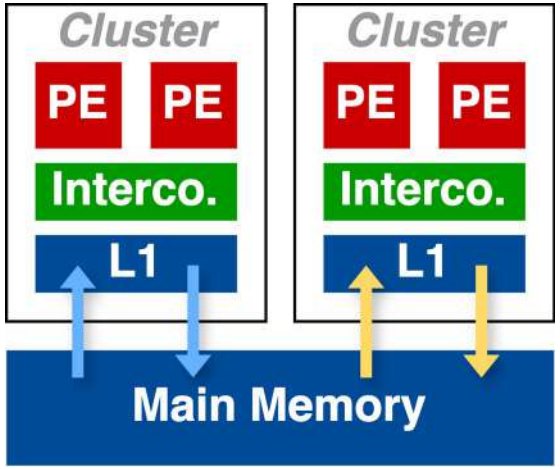
Scale-up

Increase the number of PEs in a cluster



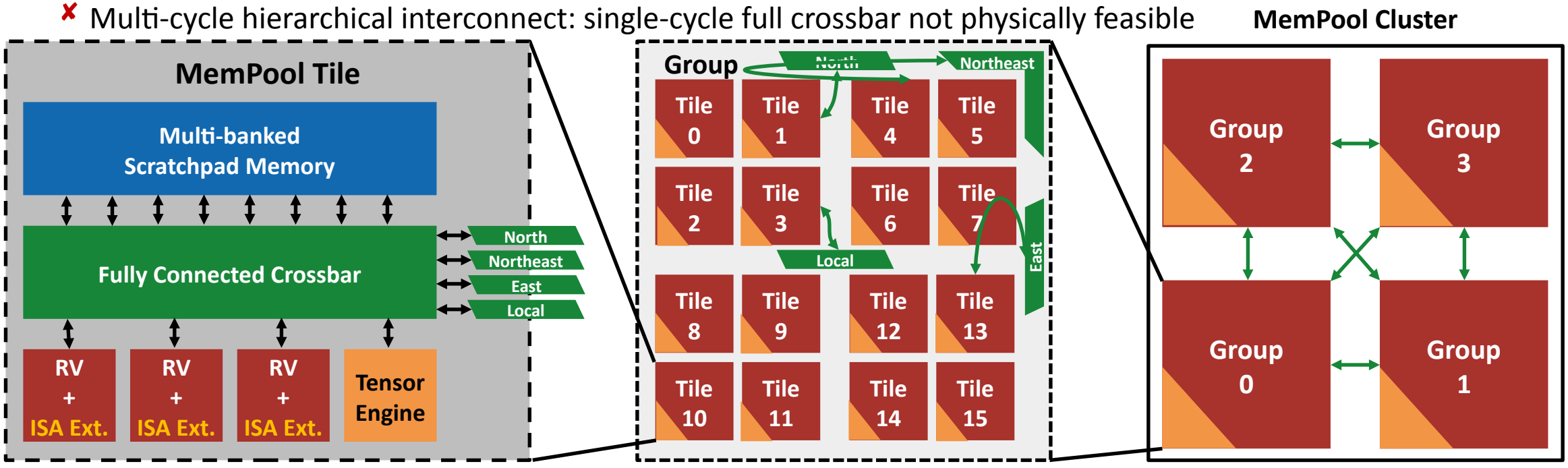
Scale-out

Increase the number of parallel Clusters



Scale-up to Large Clusters

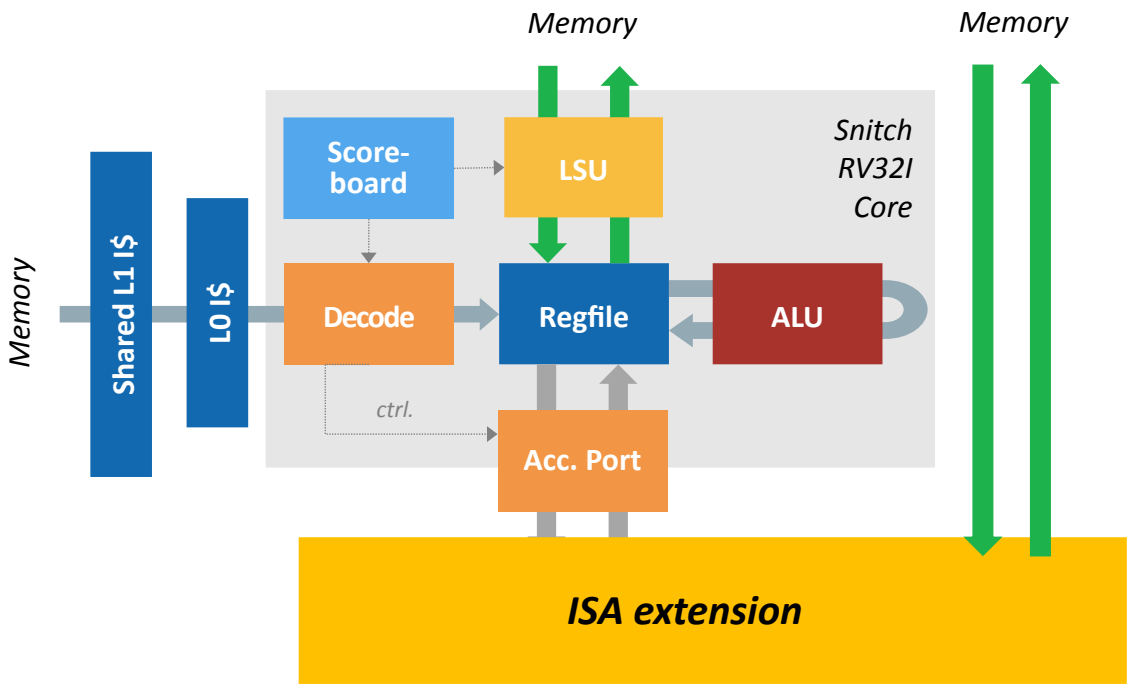
- **A large "MemPool": thousands of PEs and and of MB of shared L1 data memory. Why?**
  - ✓ Better global latency tolerance if $L1_{size} > 2\times L2_{latency} \times L2_{bandwidth}$ (Little's law + double buffer)
  - ✓ Smaller data partitioning overhead, larger Arithmetic Intensity: N^3/N^2 for MMUL grows linearly with **N**!
  - ✗ Multi-cycle hierarchical interconnect: single-cycle full crossbar not physically feasible



Neet PE latency tolerance for multi-cycle cluster interconnect!

Snitch Core: Latency Tolerant, Extensible RV PE

- ***Snitch*: tiny (20KGE), latency tolerant RV core**
  - Through a scoreboard ld/st queue
    → can issue multiple non-blocking memOPs
  - **Tolerates up to ~10 cycles latency to memory**
    → bound by L0 (register file) size
- **Paired with *ISA extension subsystem***
  - SIMD, Vector (RVV), Tensor (IME, VME, AME)
  - Quantized, MX data types
  - Non-linearities
  - …

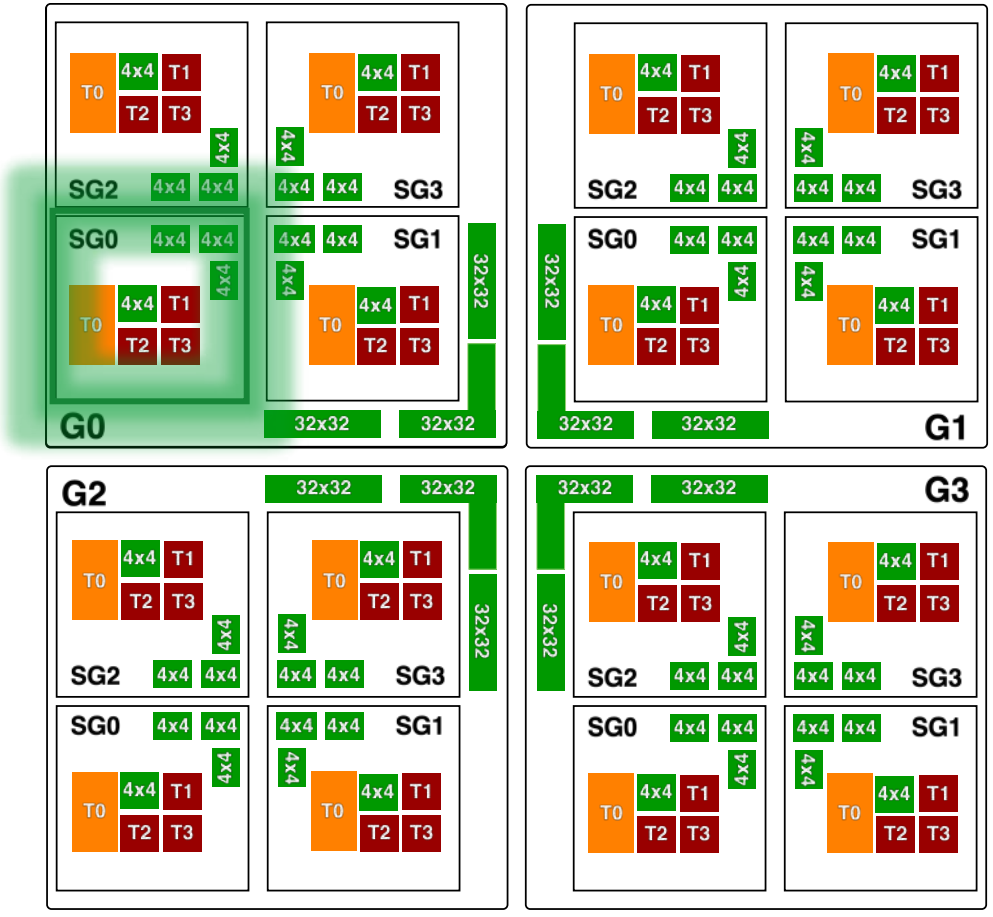


TensorPool (TOP view)

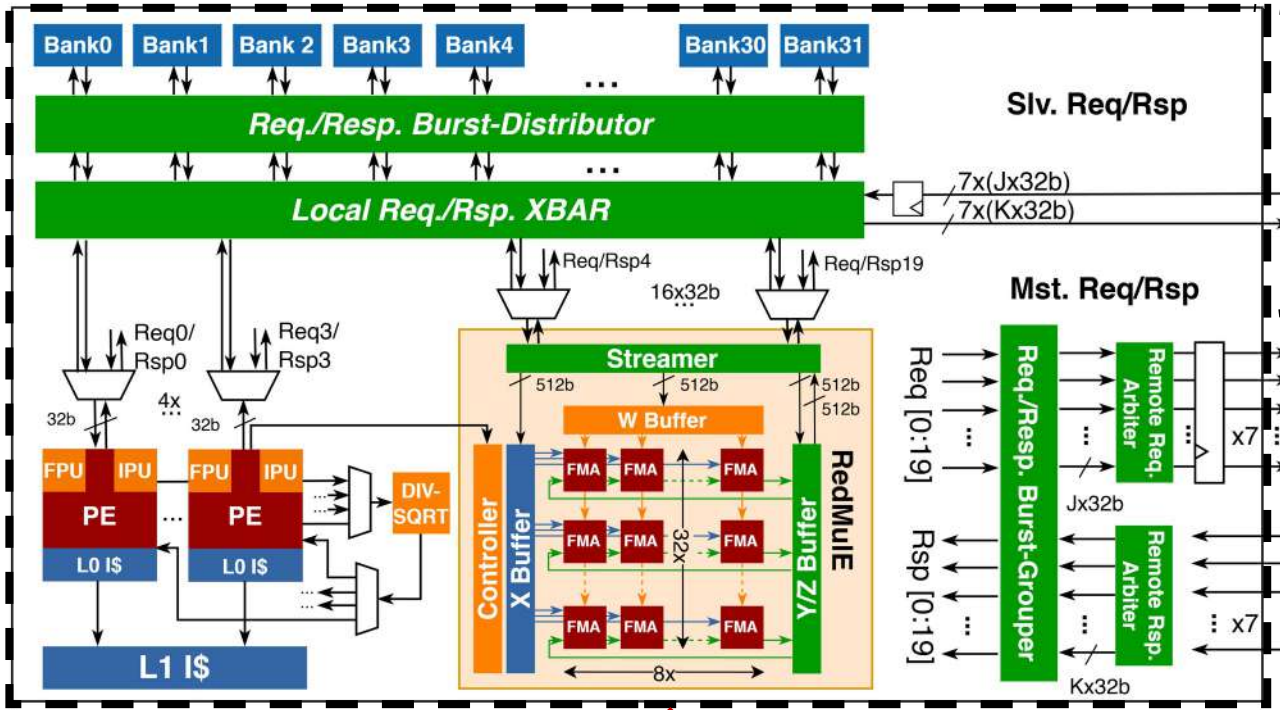
- Hierarchical Implementation
  - 4 PEs (+ 1 Tensor Engine) / Tile
  - 4 Tiles / SubGroup
  - 4 SubGroups/Group
  - 4 Groups / Cluster

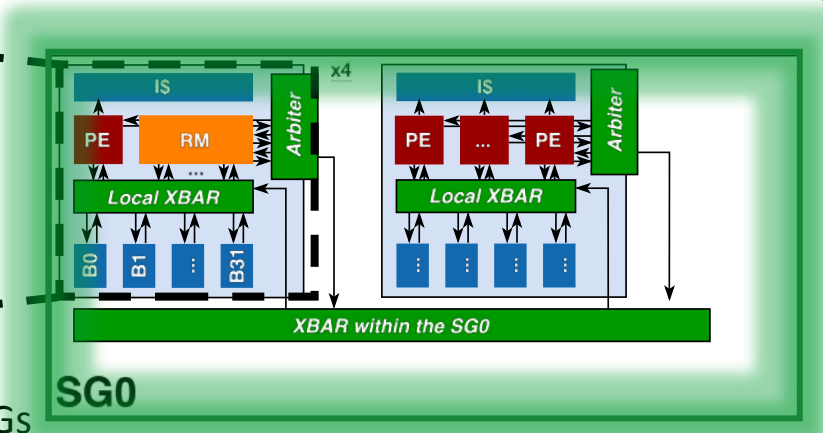
- Peak Performance:
  - (1 TE + 4x4 PEs) / SG
  - $(1x256_{TE} + 2_{f16}x16_{PE})$ x 16 SGs

  $= \mathbf{4608}\ MAC_{f16b}$ /Cycle/Cluster



TensorPool (Tile View)





To Local SG
To Remote SGs
To Remote Grups
From Local SG
From Remote SGs
From Remote Grups

4 Cores, 1 DivSqrt,
32 TCDM-Banks, L1 I$

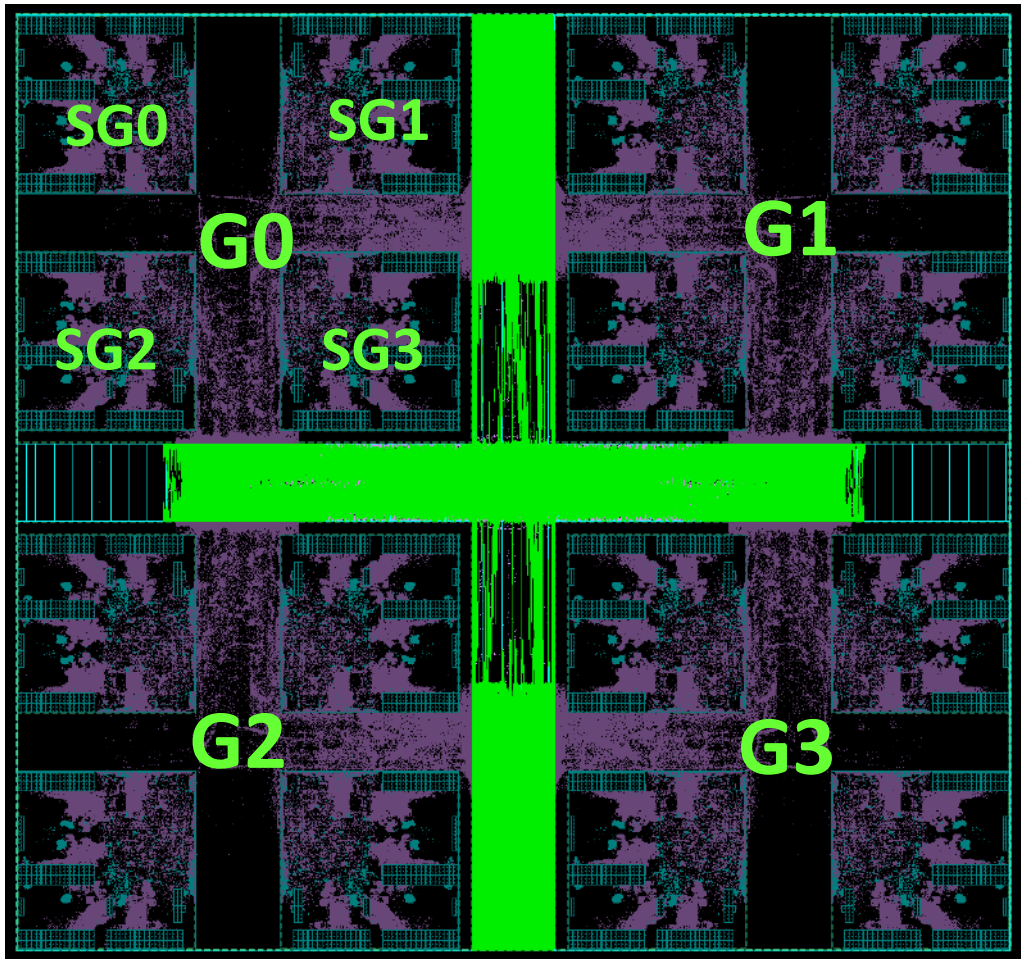
Tensor Engine connects to local-Tile interconnect (RV AME)

Huge Scale-up design space: ISA, PE microarchitecture, Interconect

The limit of Scale-up

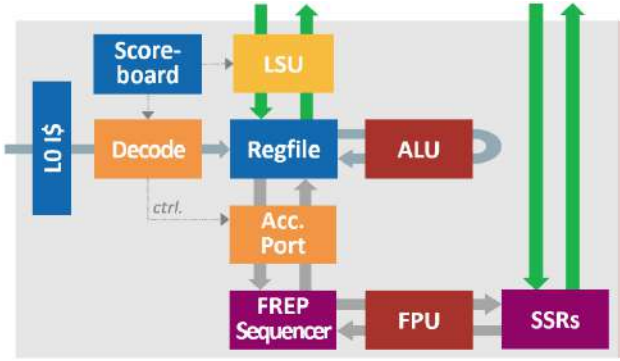
TSMC's 7nm FinFET (N7)
Synopsys' FusionCompiler 2025.06



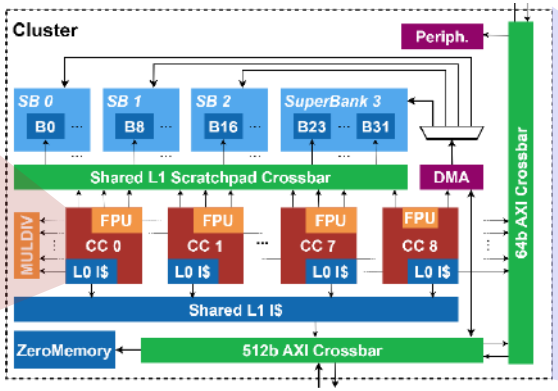


Scaling Out: Beyond the Cluster

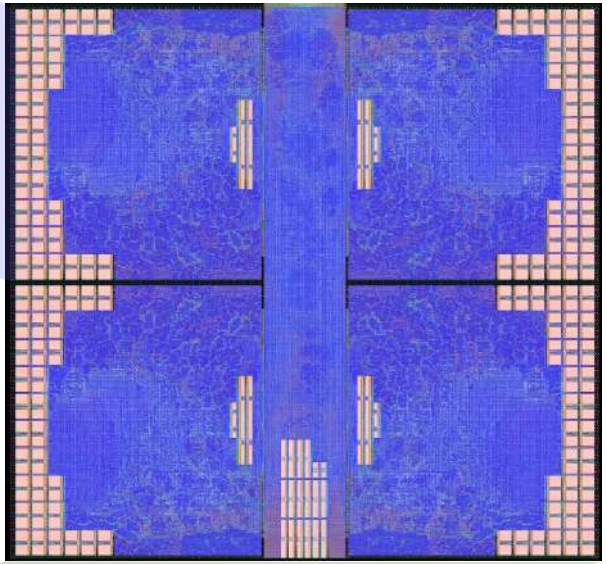
Snitch Core



Snitch Cluster



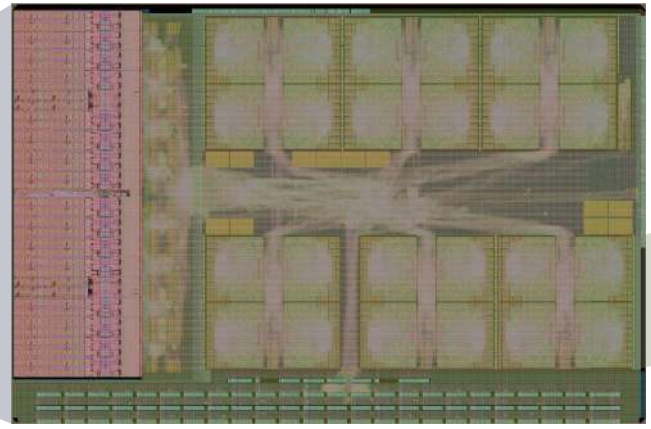
Occamy Group



Occamy System

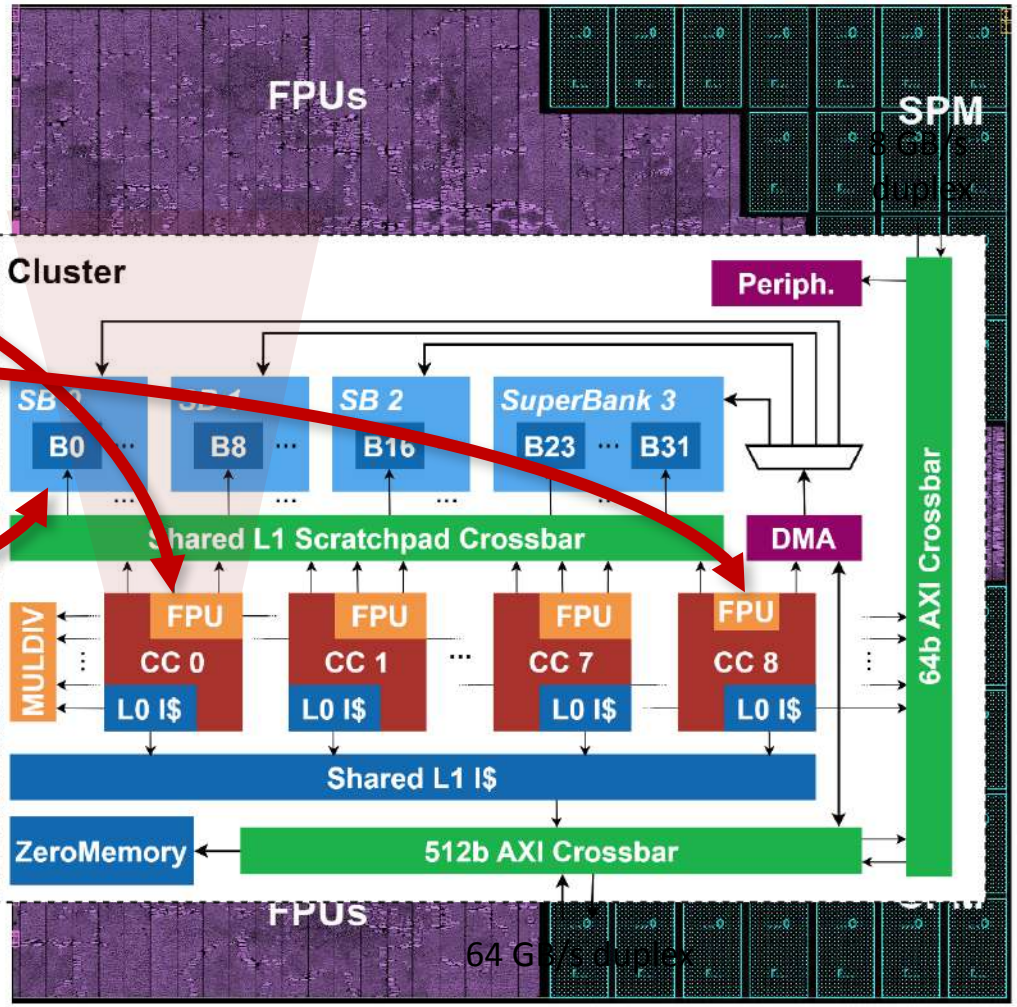


Occamy Chiplet

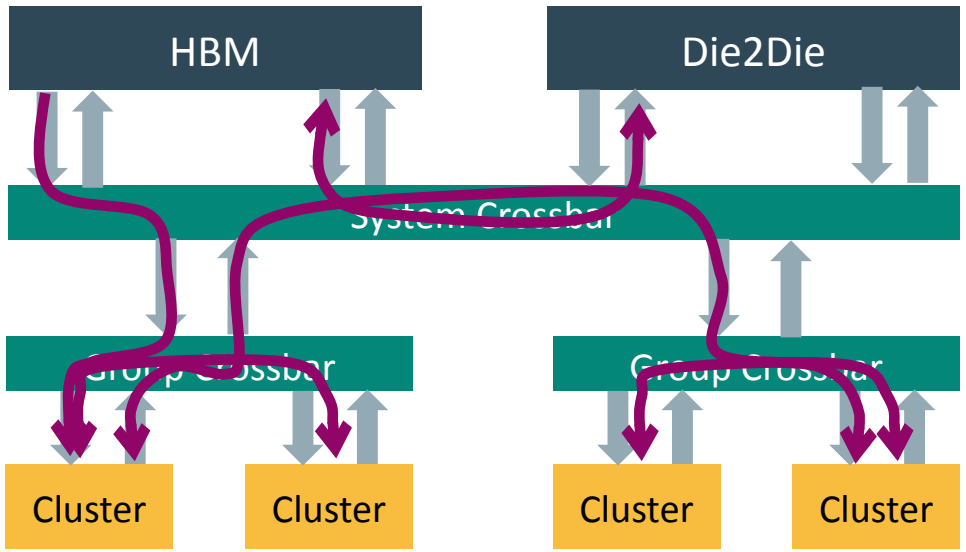


Global Interconnect Latency Tolerance

- **Compute engines**
  - Scalar, vector, tensor
- **DMA engine**
  - Widei interface to interconnect
  - HW support for autonomous tensor transfers
- **L1 TCDM**
  - low-latency shared scratchpad
  - Double-buffer large chunks (KBs) with DMA



- **DMA (10% overhead) boosts latency tolerance of L2+→L1: 100s of cycles**



Problem: HBM Accesses are critical in terms of

- Access energy
- Congestion
- High latency

Instead reuse data on lower levels of the memory hierarchy

- Between **clusters**
- Across **groups**

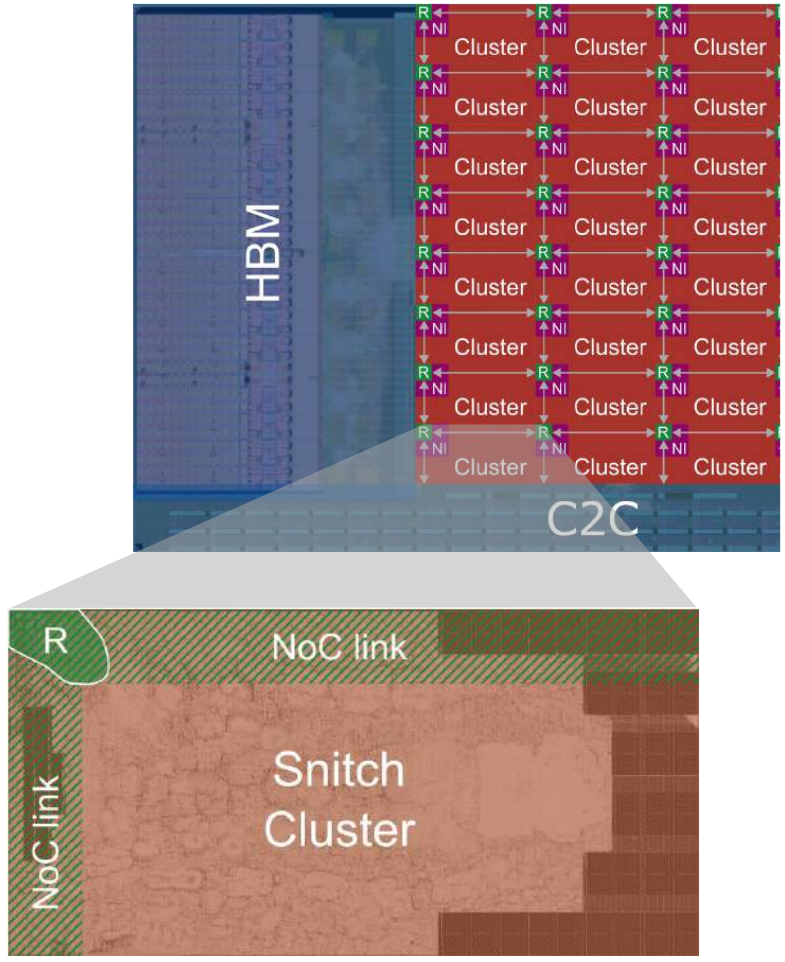
Smartly distribute workload

- **Clusters**: Tiling, Depth-First
- **Chiplets**: E.g. Layer pipelining

Big trend!

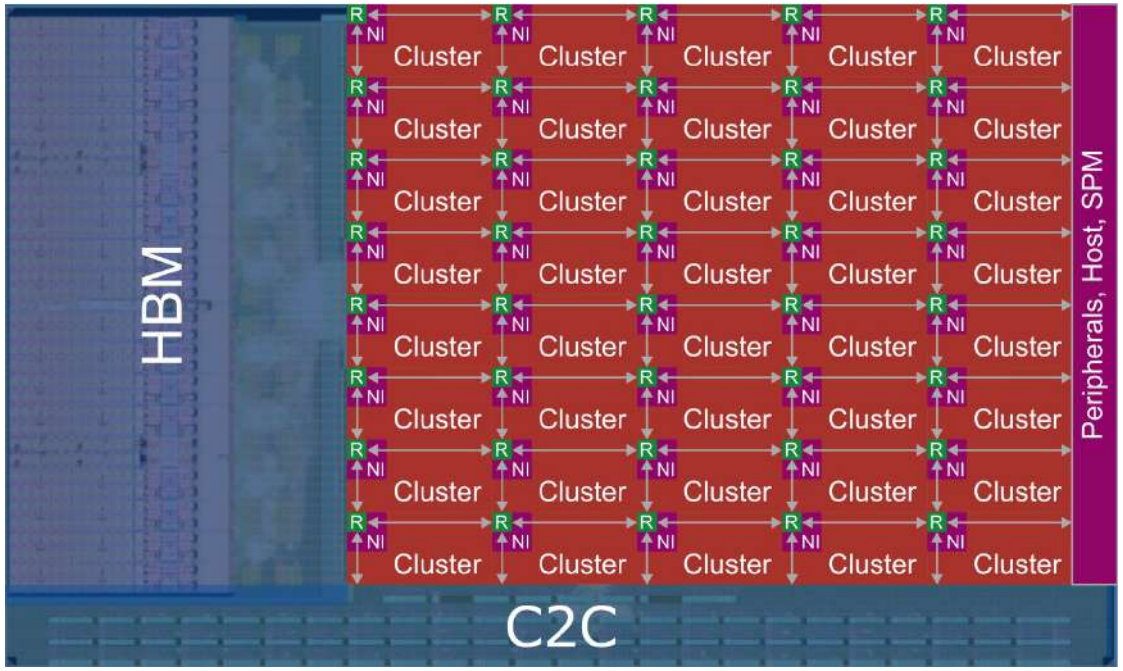
Physical design aware, computational NoC → FlooNoC

- **Key idea 1 – wires on top of clusters**
- **Key idea 2 – support computation in NoC**
  - Smart multicast
  - Reductions
- **Potential for big area/performance gains**
  - Only **~10%** interconnect area
  - **66%** more clusters than with classical on-chip networks
  - *High Bandwidth*: **629Gbps/link**
  - *High Energy-Efficiency*: **0.19pj/B/hop**

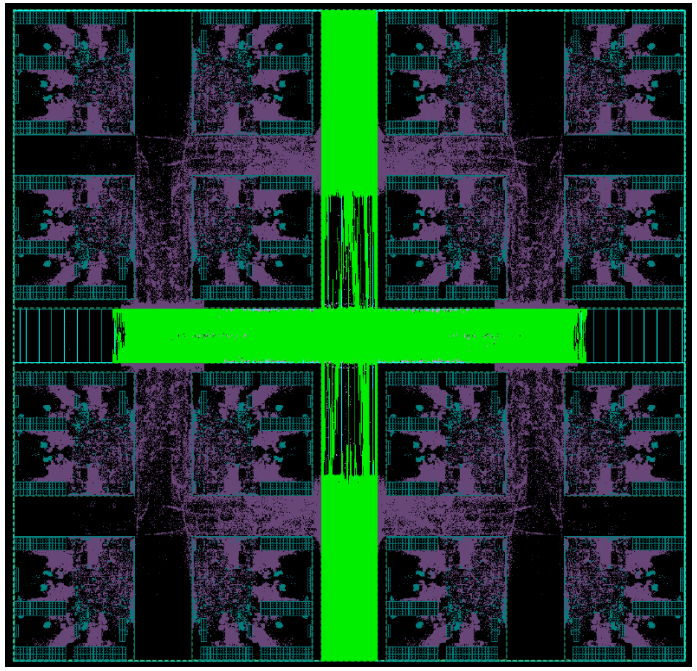


Huge Scale-up Design Space too: NoC, Compute in/near NoC

What are the Scaling Bottlenecks?



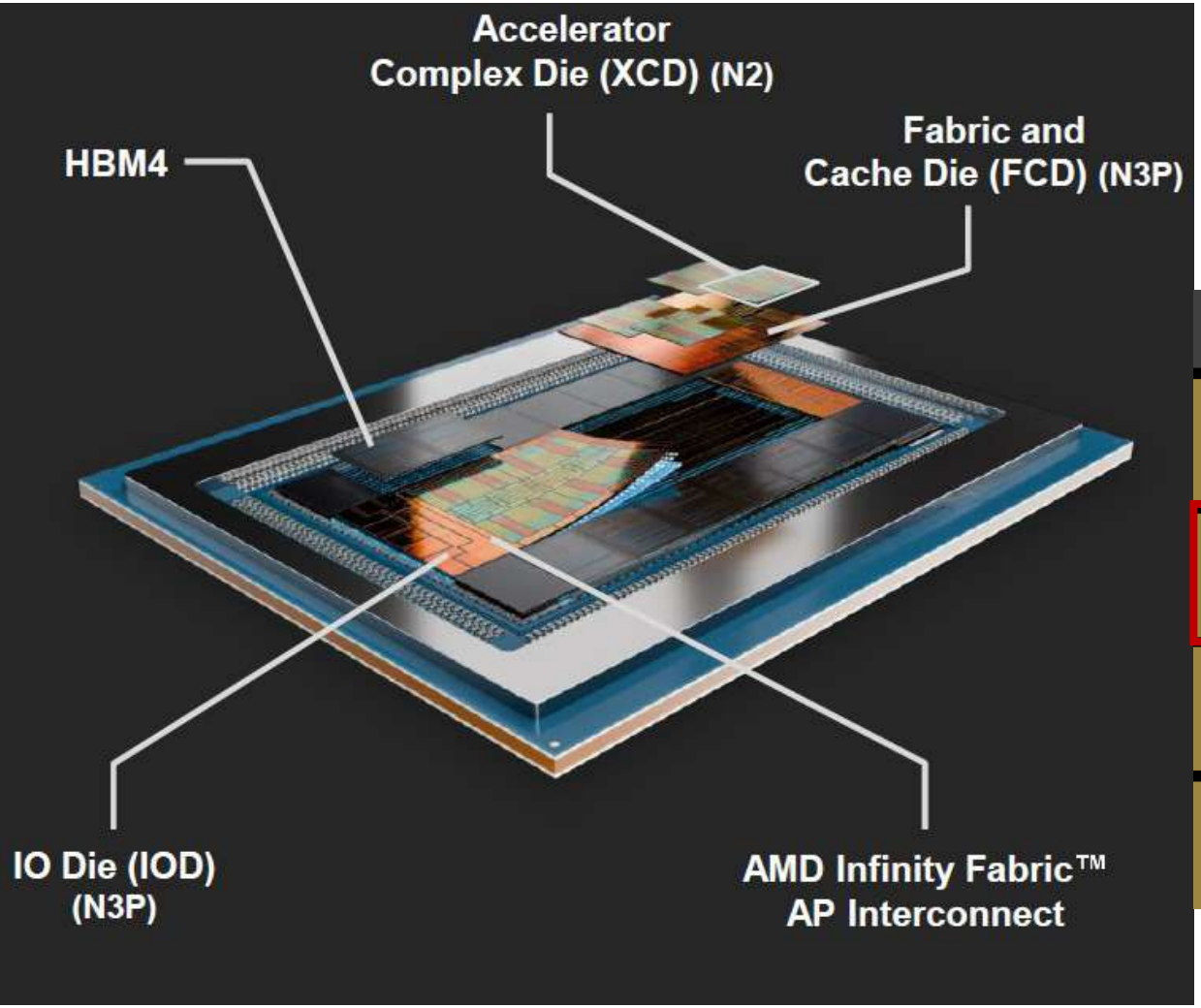
Scale-out → IO area even at reticle size



Scale-up → Cluster Interconnect cost

Nowhere to go, but UP → 3D to the rescue!

Heterogeneous 3D for Scale-out: Already in Products!

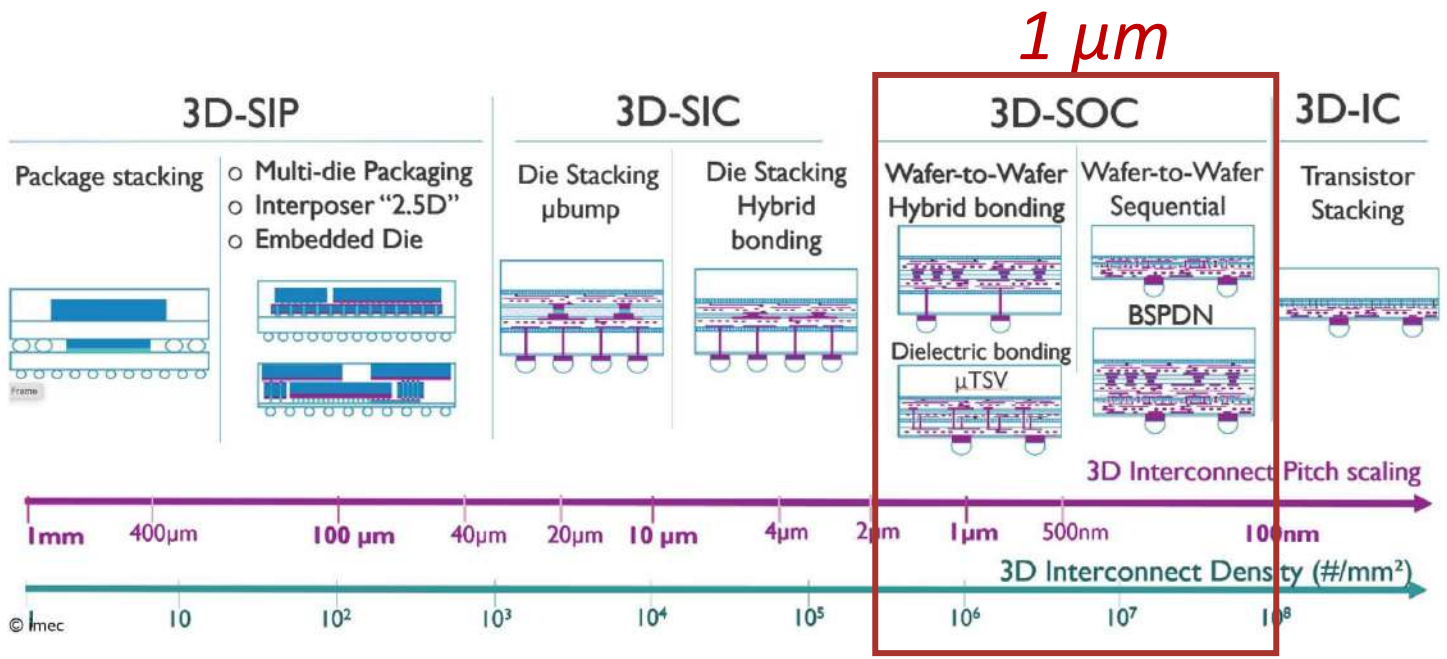


Innovation	What's New	AI Workload Impact
Chiplet Architecture	Compute, memory, cache, and I/O functions are partitioned across specialized dies	Optimizes each function independently for performance, power, and manufacturability
3D Hybrid Bonded XCDs	Compute dies stacked and connected through high-density die-to-die interconnects	Higher compute density and improved performance-per-watt
HBM4 Integration	12 HBM4 stacks 432 GB capacity 23.3 TB/s bandwidth	Larger models, KV caches, and activations remain resident in local memory
CoWoS-L Advanced Packaging	High-bandwidth on-package communication between compute, memory, and I/O dies	Low latency and high bandwidth enable both GPU-wide single operators and sharding techniques like KV cache parallelism, TP, and EP

AMD – Instinct MI455x Hot Chips 2026

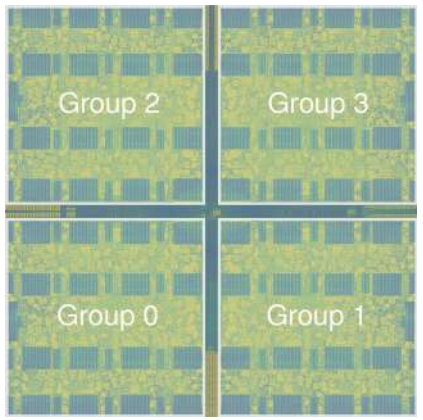
Pushing the 3D Design Space

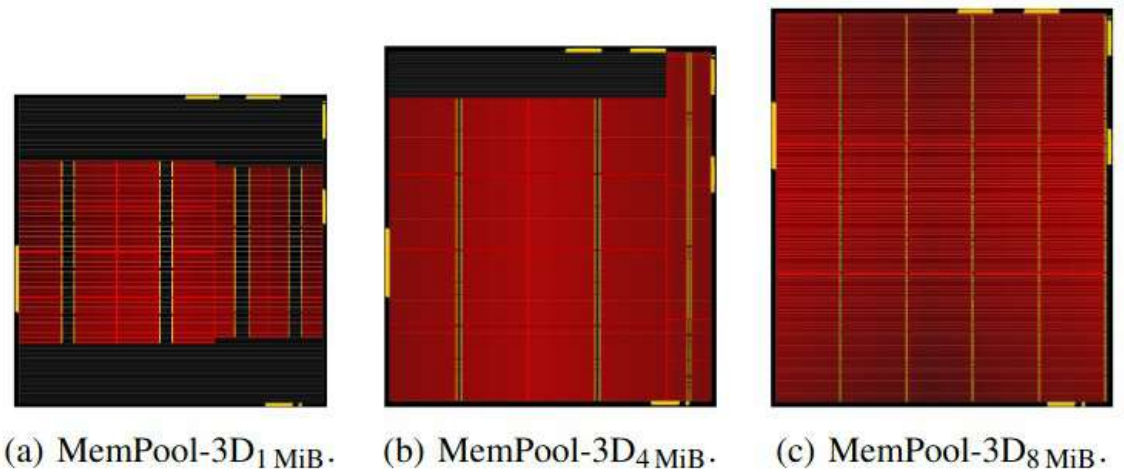
- Scaling vertical connectivity from 10s of *µm to tens of nm*



https://www.imec-int.com/en/articles/wafer-wafer-hybrid-bonding-pushing-boundaries-400nm-interconnect-pitch

- **Memory-on-Logic (MoL)** implementation of MemPool
- MemPool with 1, 2, 4, 8 MiB of L1 on a second stacked die
- More memory on the same x-y area





(a) MemPool-3D$_{1\,\text{MiB}}$. (b) MemPool-3D$_{4\,\text{MiB}}$. (c) MemPool-3D$_{8\,\text{MiB}}$.

Memory-on-Logic



	MemPool-2D	MemPool-3D
128 KiB	Footprint: *1.000*	Footprint: *0.665*
	7% larger	*Same footprint*
256 KiB	Footprint: *1.074*	Footprint: *0.665*
512 KiB	Footprint: *1.299*	Footprint: *0.737*
1 MiB	Footprint: *1.572*	Footprint: *0.857*

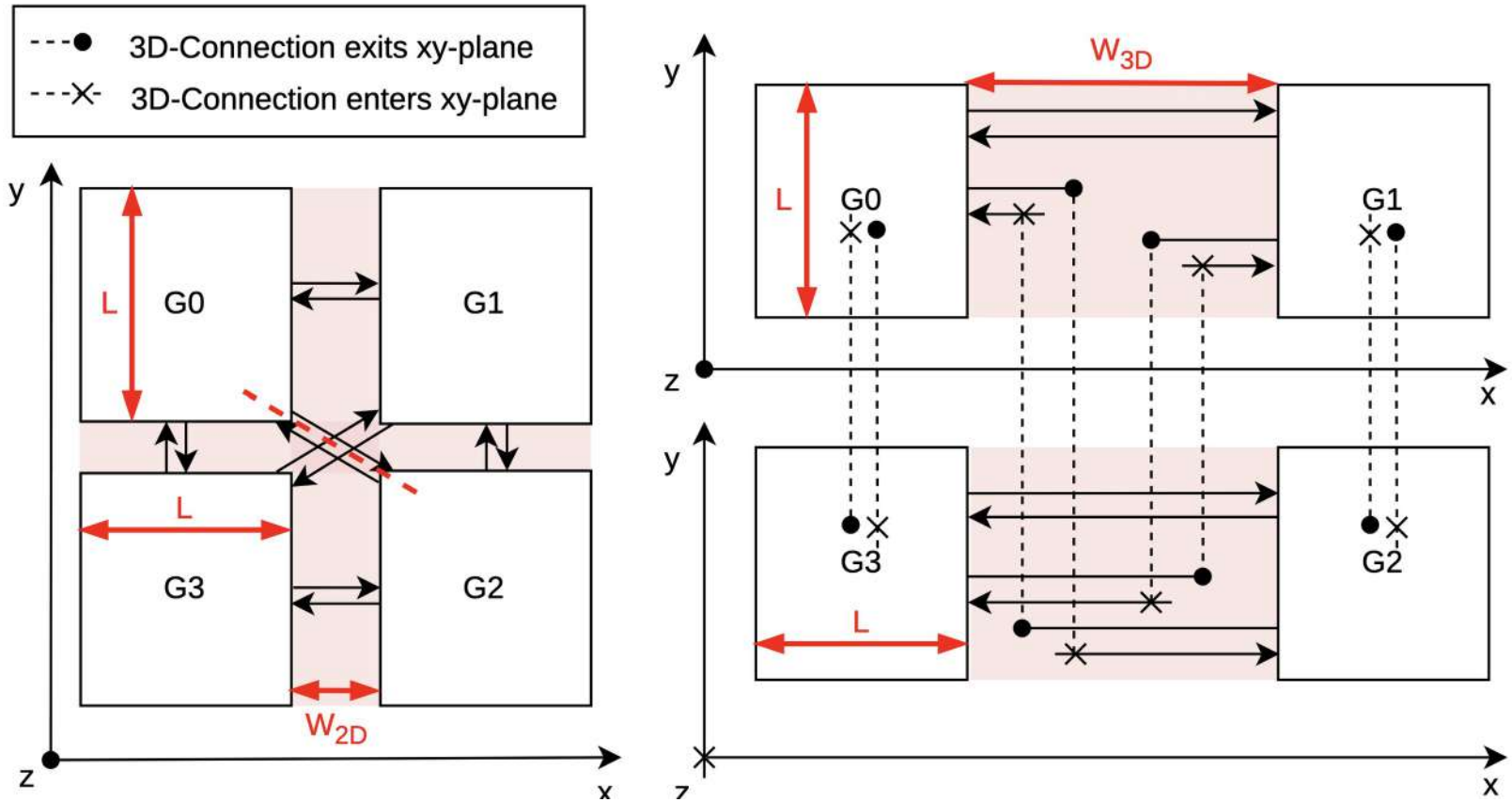
14% smaller, despite 8x the L1 capacity!

Similar trends for **Wire length** and **#Buffers**

MoL is a significant scale-up booster!

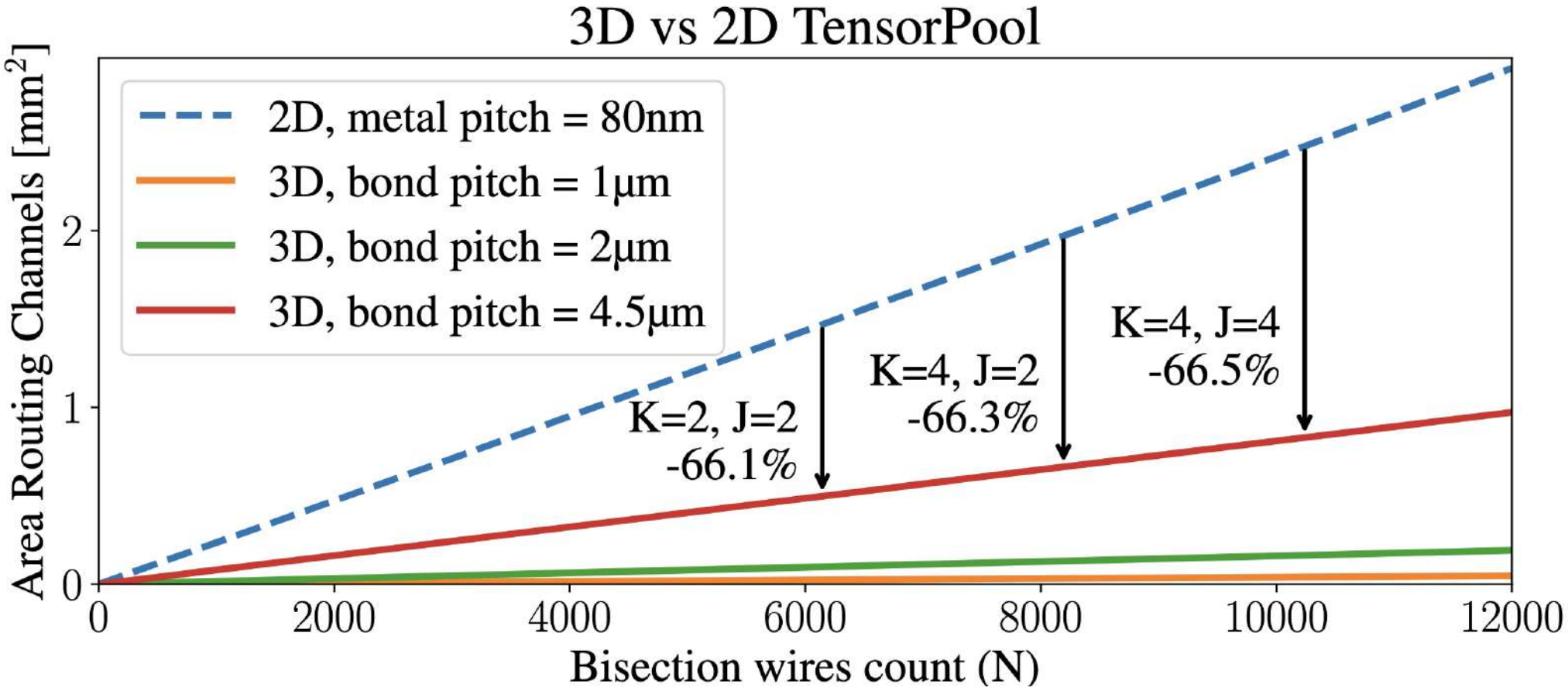


Logic-on-Logic



- From a 2D floorplan with routing corridors and cross-path connections
- To a 3D floorplan with stacked Groups and a central routing corridor

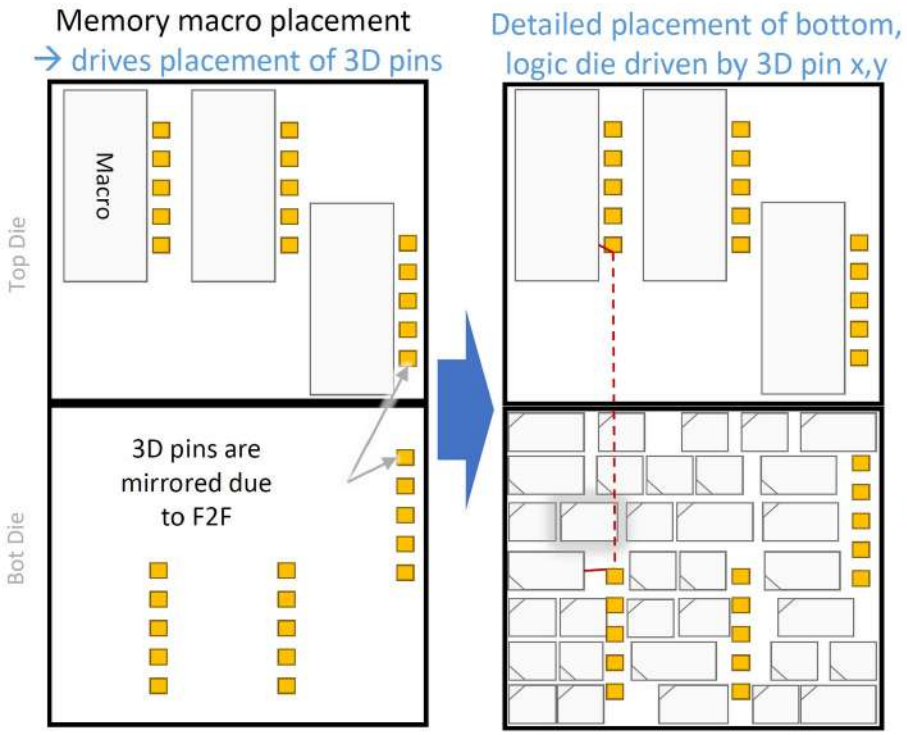
Quantifying Area Efficiency Gains

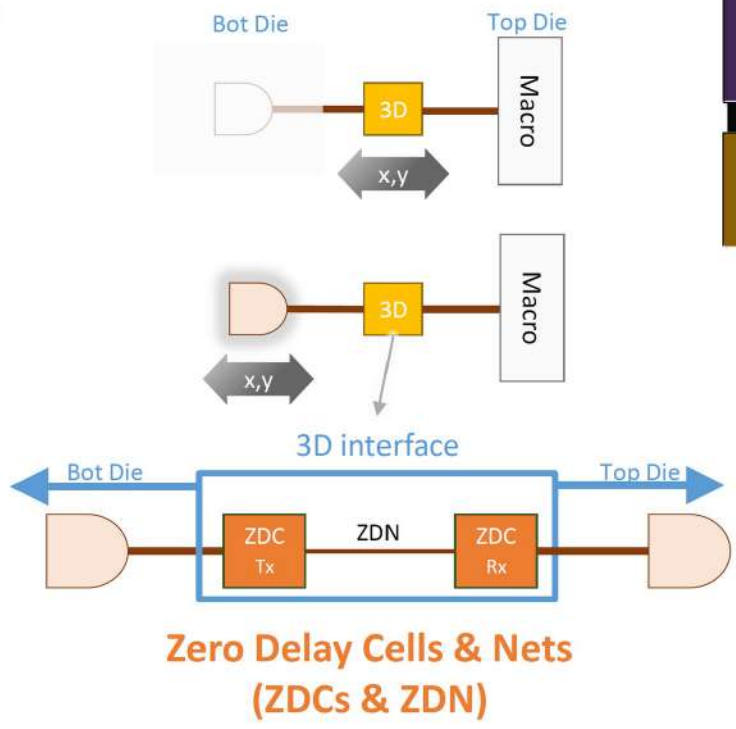


LoL is more flexible than MoL!

Design Tools Support is essential

[IMEC, Synopsys DATE25]







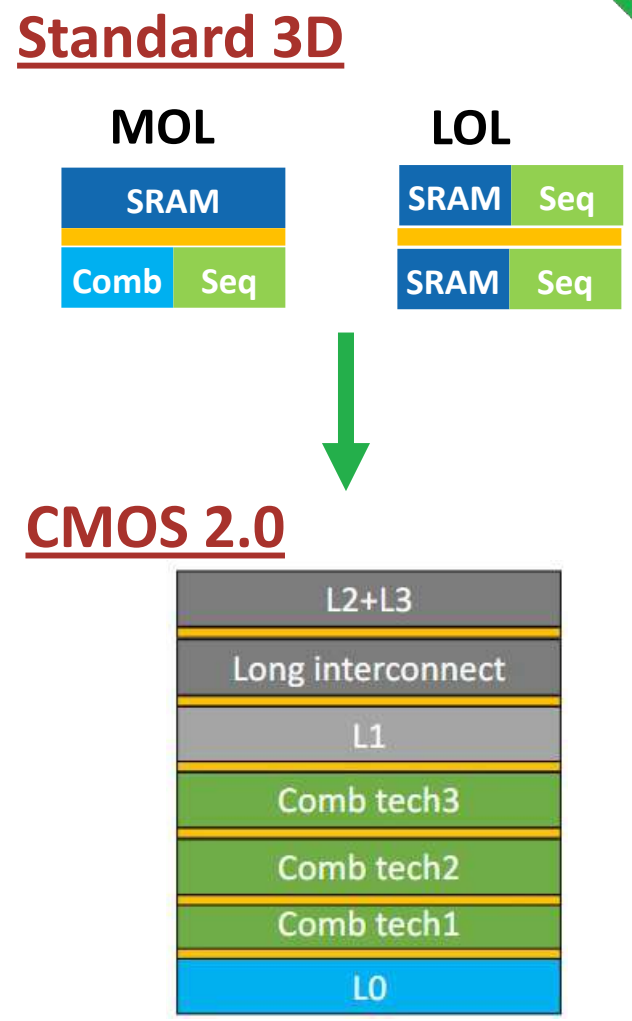
Accurate timing, power analysis



Full physical design (MoL F2F)

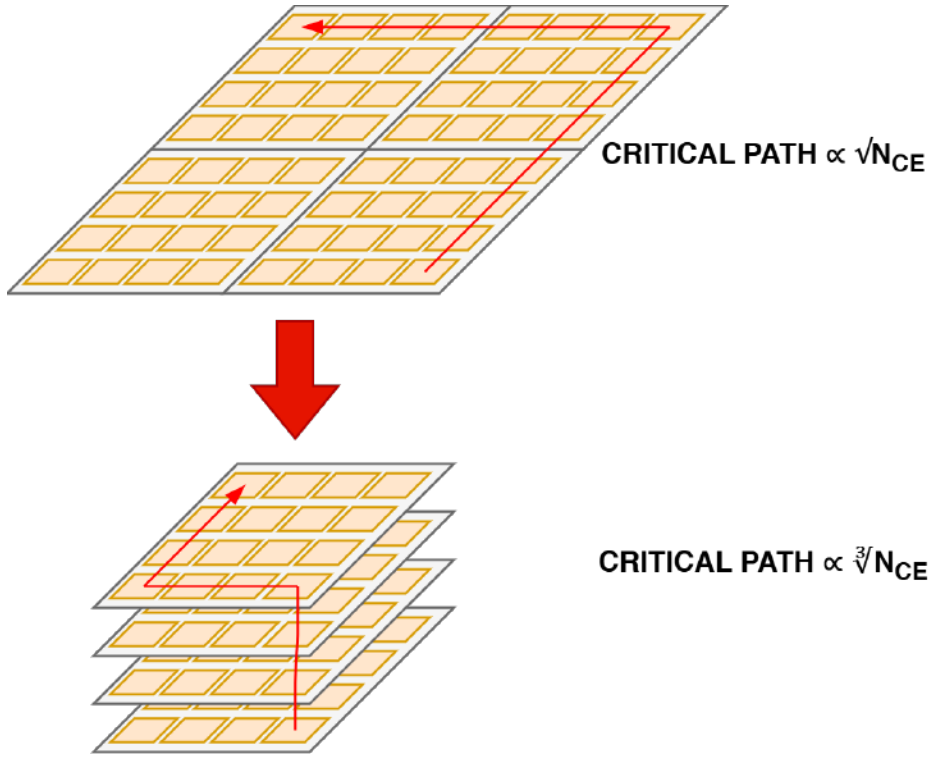
CMOS 2.0 for Sub-µm Inter-Die Pitch

- **From Standard 3D technology**
  - F2F wafer bonding with 1um pitch
  - 3D approaches
    - Memory-on-Logic (MOL) & Logic-on-Logic (LOL)
- **To CMOS 2.0**
  - More fine grained (100nm and below) vertical pitch
  - Multiple B2B + F2B wafer bonded layers (heterogeneous)
- **Opportunities**
  - Better timing, Less wiring, Area footprint
  - Huge potential for layer customization
  - Lots of STCO (system technology co-optimization) needed



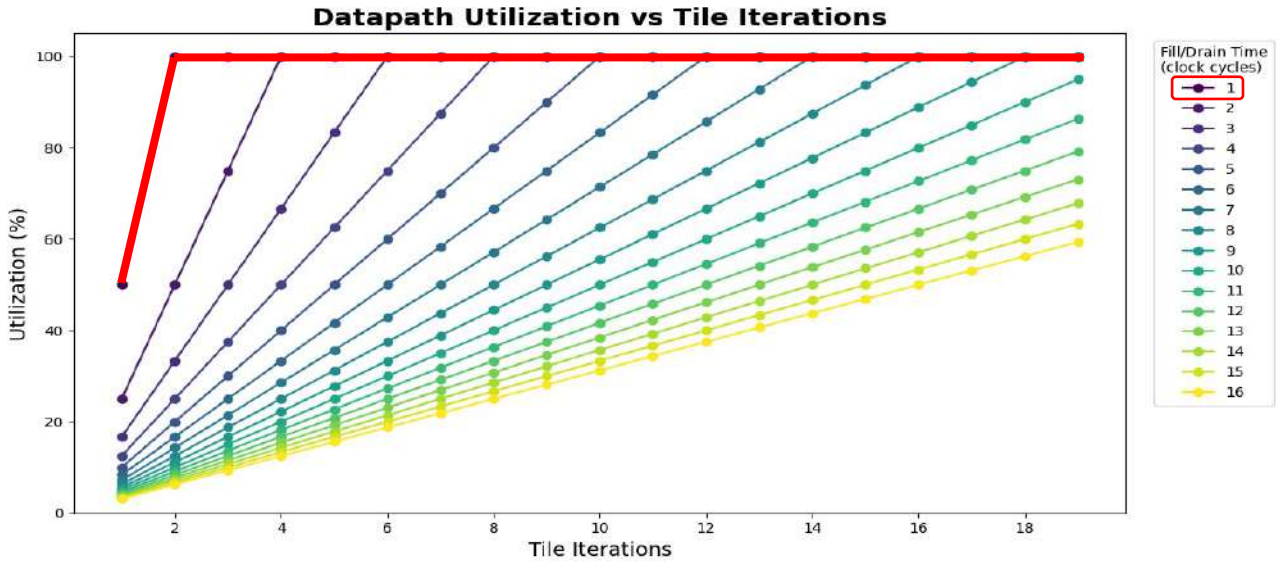
Bungee: A Native CMOS2.0 Tensor Unit Design

- At a fixed computing elements budget, in a planar design the longest path is **proportional to** $\sqrt{N_{CE}}$
- With the possibility of folding logic on logic with a very fine vertical pitch, this number **drops to** $\sqrt[3]{N_{CE}}$
  - And vertical paths are much are much shorter too!
- Routing on the third dimension increases the number of units we can reach in a single clock
- Build extremely low-latency TU at large scale



Why a Low-Latency Design?

- **In a tensor unit, pipelining latency is a problem because of pipeline priming effect**
  - Lowers utilization unless we work on small tiles relative to the size of the matrices being multiplied
- **Want to be fast with lage MAC arrays while keeping latency (i.e. pipelining) low,**
  - Also, eliminating pipeline registers can yield further efficiency gains

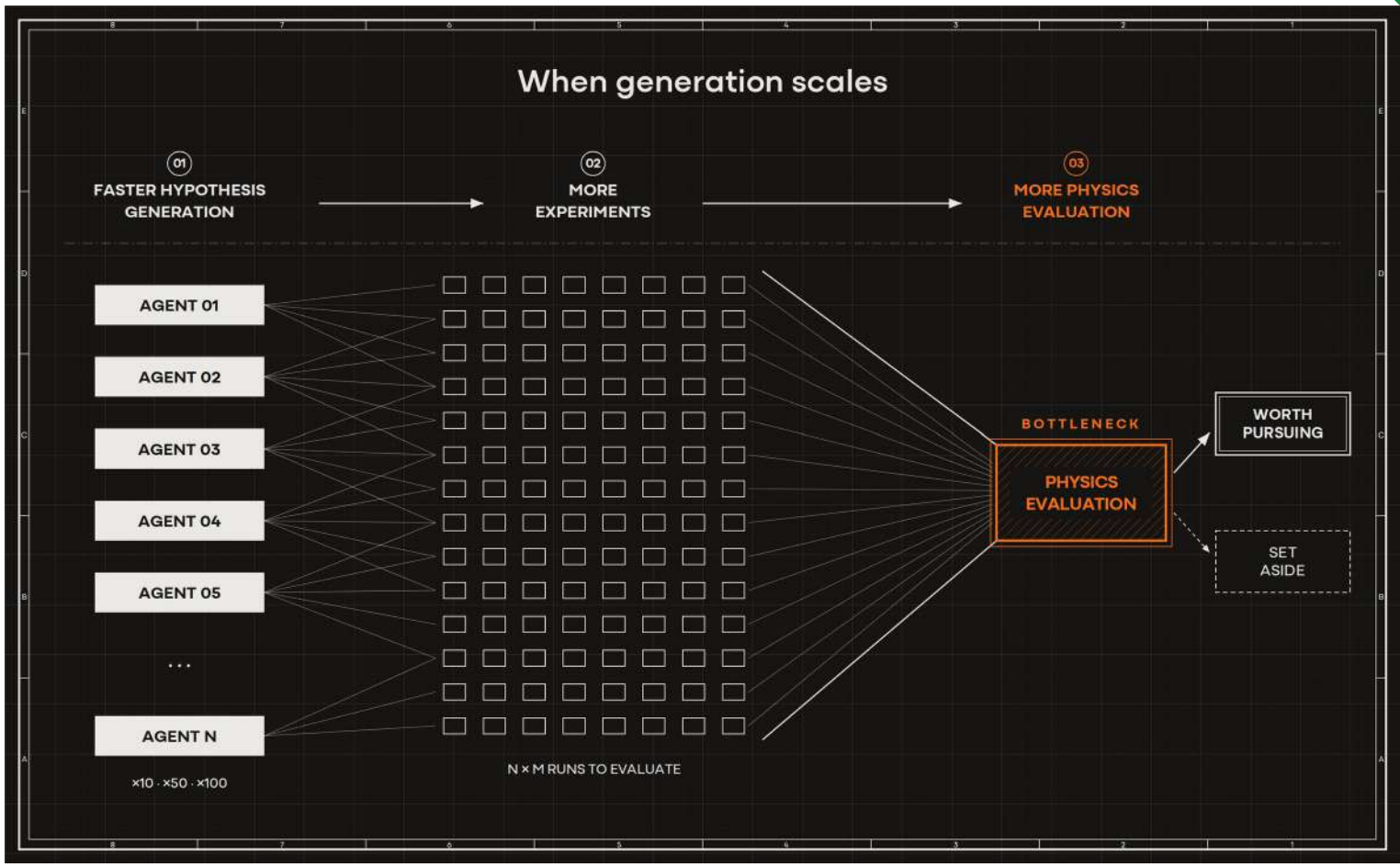


Design Space "Explosion" with 3D!

From Handcrafting to Agentic Optimization

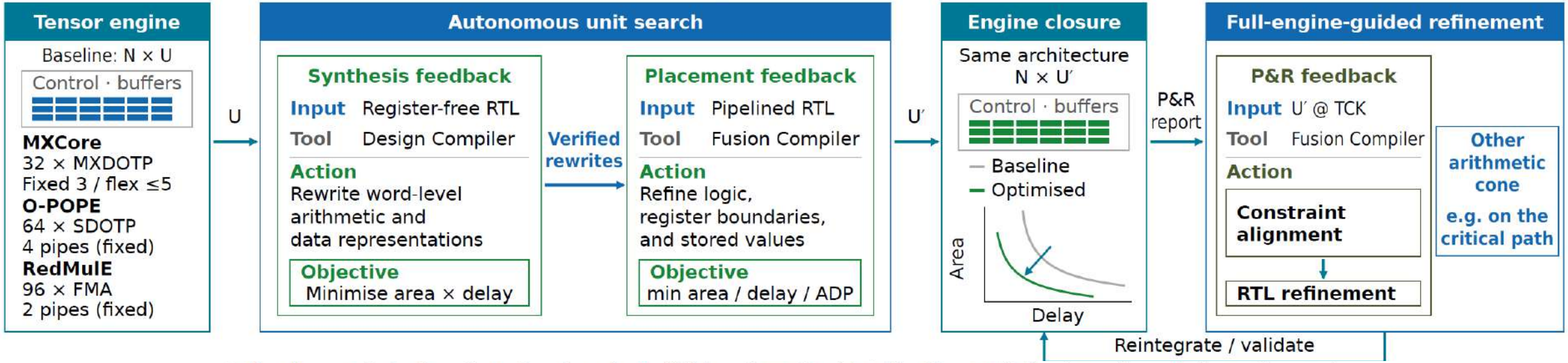
- **Design Space is exploding**
  - Domain adaptation is required at the month time scale
  - Architecture and implementation: scale-up, scale-out, 3D
- **How to explore it?**
  - Hand-crafting is not viable at the speed required
  - Need speed and Intelligence → AI agents

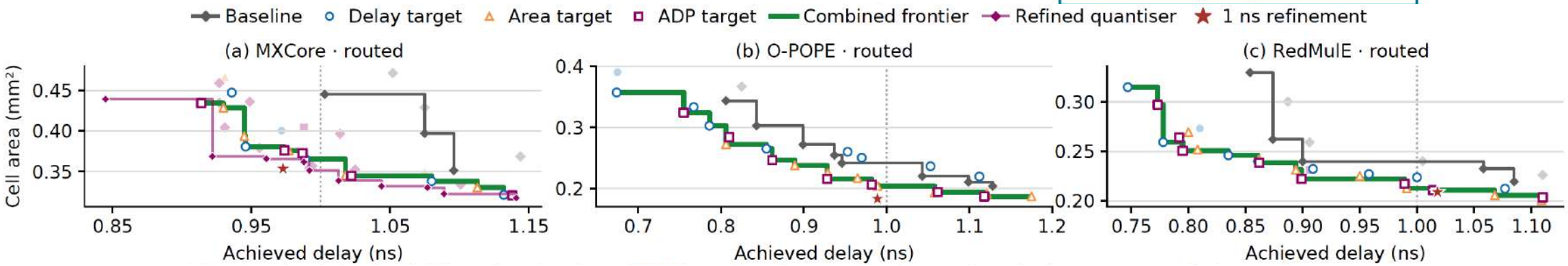
https://chipagents.ai/blogs/ai-native-flow-for-chip-design



ACTERO: Agentic NPU Super-optimizer

Arithmetic-Centric Tensor Engine RTL Optimisation with Multi-Fidelity Feedback and Agentic Search





The Future is AI-Boosted Extreme Co-design

Flexibility

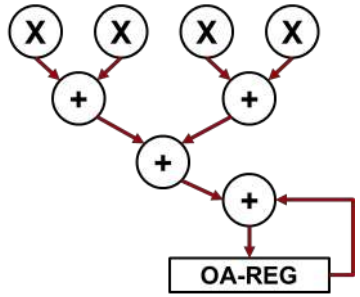
Energy-Efficient RV Core → **20pJ (8bit)**

ISA-based 10-20x →**1pJ (4bit)**

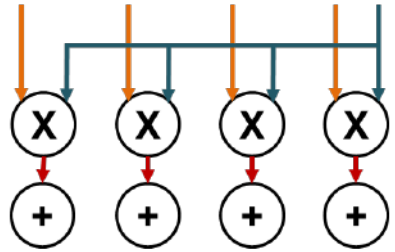
Configurable DP 10-20x → **100fJ (4bit)**

Highly specialized DP 100x →**1fJ (ternary)**

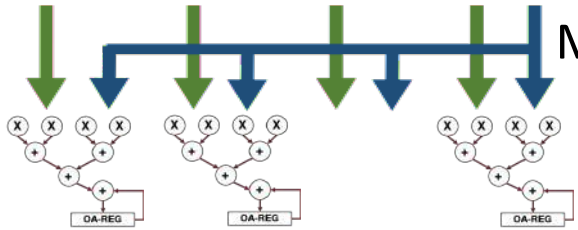
AI- boosted EDA



Inner Product



Outer Product



Mixed

Precision tuning – OP/Mem tuning - deep arithmetic optimization – 3D placement...

Co-Specialize SW, HW, EDA & Technology is the frontier & Open Platforms are Key!



Thank You!