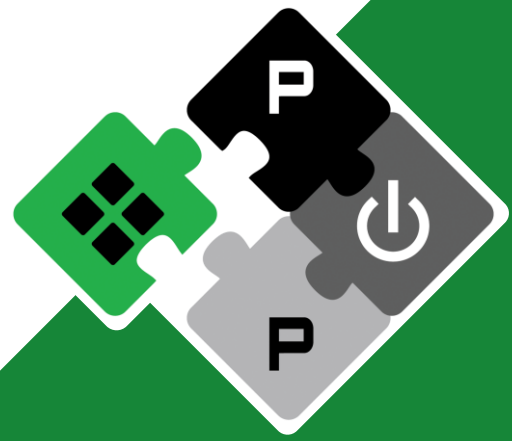
ETH zürich

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

MemPool Flavors: Between Versatility and Specialization in a RISC-V Manycore Cluster

Integrated Systems Laboratory (ETH Zürich)

Sergio Mazzola smazzola@iis.ee.ethz.ch
Yichao Zhang
Marco Bertuletti
Diyou Shen
Luca Benini

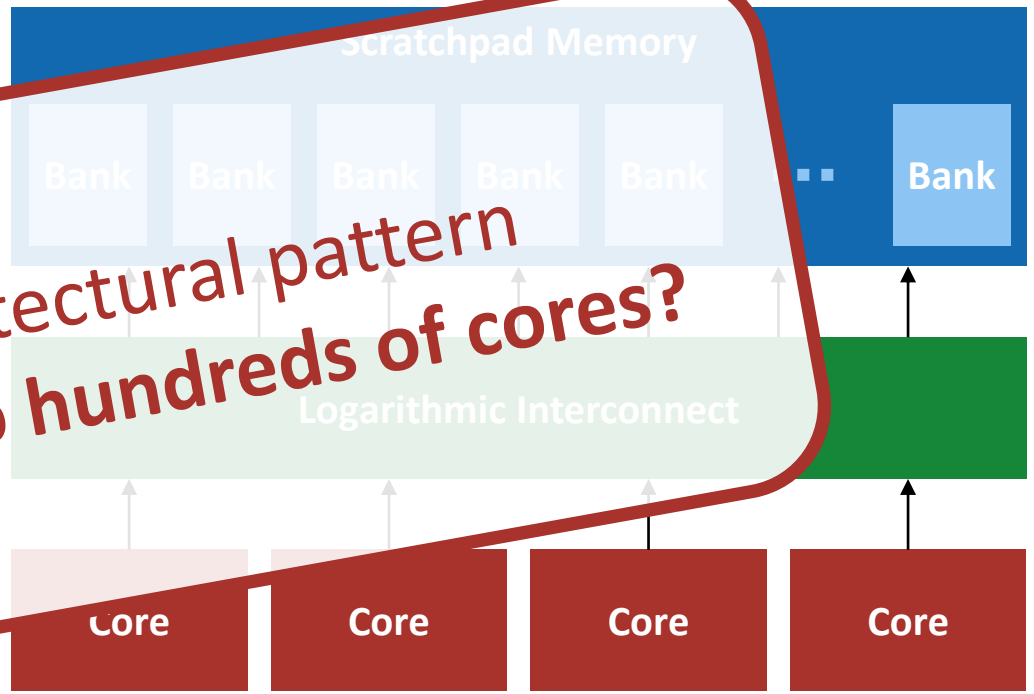


pulp-platform.org
@pulp_platform
company/pulp-platform
youtube.com/pulp_platform

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

The shared-memory cluster

- **Widely used building block**
  - Parallelism, flexibility
- **Low-latency access SPM**
  - Multi-banked
  - Efficient communication + synchronization
- **Fast logarithmic interconnect**
- **Efficient cores**
  - Individually programmable
  - ISA extensions
  - Hide SPM residual latency

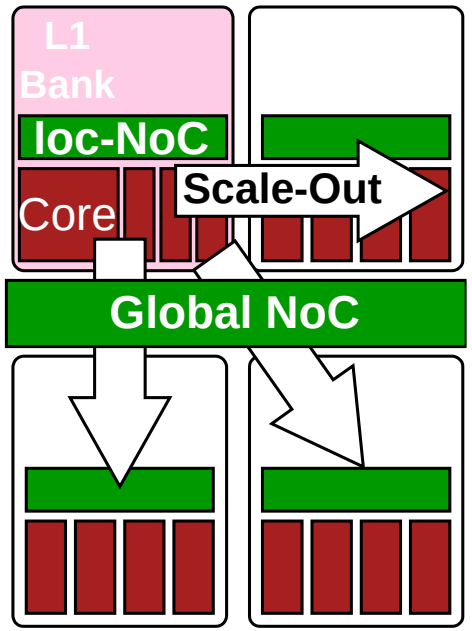


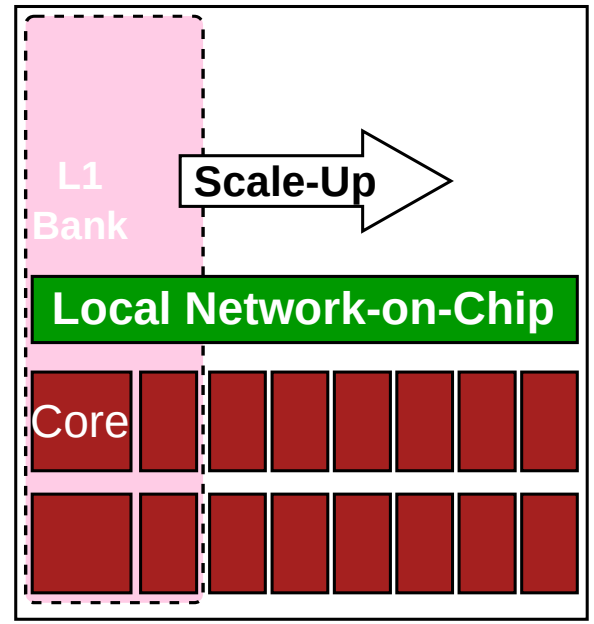
Common architectural pattern

But does it scale to hundreds of cores?

Scale-up vs. scale-out

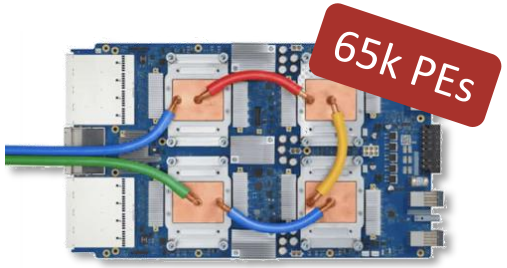
- **Large workloads → many cores + big memory**
- **More clusters?**
- **Bigger single cluster!**
  - Low-latency memory access
  - Reduce overhead of data chunks transfer
  - Keep high compute utilization
  - Physically-feasible interconnect
  - Easy to program





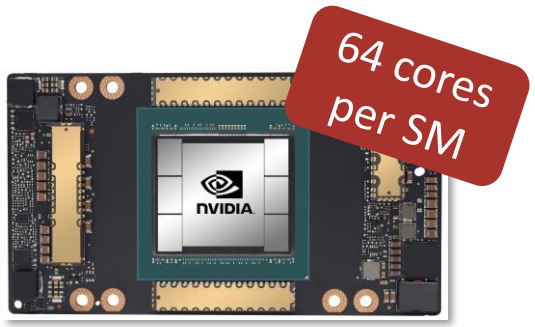
That's difficult! How do we **scale up**?

scalable



https://cloud.google.com/tpu

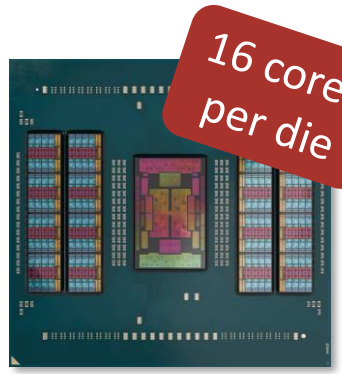
Google TPU
- Custom accelerator
- Specialized



https://developer.nvidia.com/blog/nvidia-ampere-architecture-in-depth/

NVIDIA Ampere GPU
- SIMT
- Complex mem hierarchy
- Flexible

MemPool



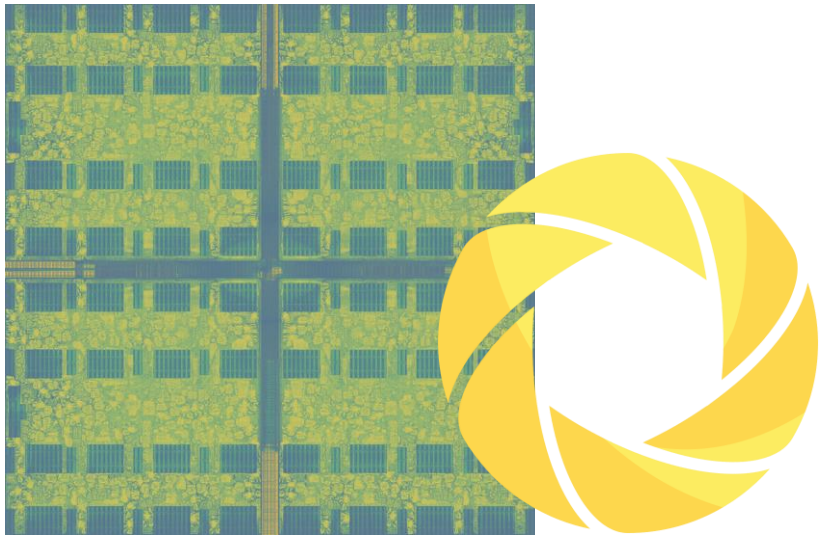
https://www.amd.com/en/products/processors/server/epyc/9005-series.html

AMD EPYC CPU
- Shared-mem
- Max flexibility
- Doesn't scale

versatile

MemPool: Open-source, RISC-V-based manycore cluster

- **Small, flexible cores**
  - **256+** RISC-V cores
- **1+ MiB of shared L1 data memory**
- **≤ 5 cycles latency**
  - without contention
  - Thanks to hierarchical architecture
- **Physical-aware design**
  - WC Frequency > 500 MHz
- **Open-source**

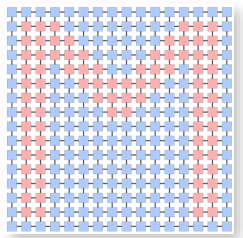


MemPool **Flavors**

MemPool

Vectorial MemPool

ITA MemPool

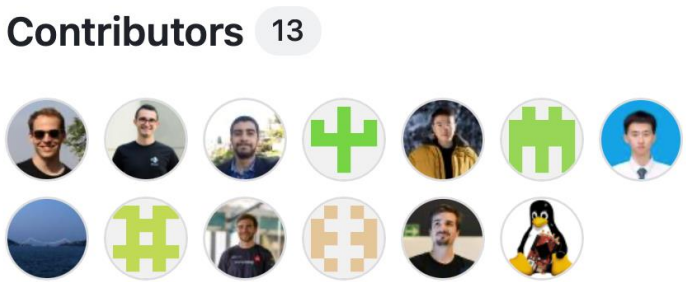


Systolic MemPool

TeraPool

NoC TeraPool

CachePool



289 stars

8 watching

49 forks

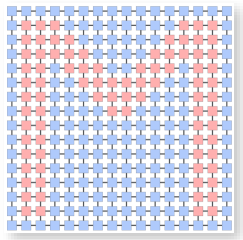
github.com/pulp-platform/mempool

MemPool **Flavors**

MemPool

Vectorial MemPool

ITA MemPool



Systolic MemPool

TeraPool

NoC TeraPool

CachePool

More than an architecture: **MemPool Ecosystem**

SOFTWARE

- Wide software **kernel library**
- Bare-metal **runtime**, OpenMP, Halide
- **GCC** and **LLVM** toolchain support
- Support for GVSOC and Banshee **platform emulators**

HARDWARE

BACKEND

- Mature backend flow in many **modern technologies**
- **2 tapeouts**

MinPool (2021)
16 cores, 200 MHz
TSMC65, 2.4mm x 2.4mm

Heartstream (2024)
64 cores, 720 MHz
GF12, 2.5mm x 2mm

More than an architecture: **MemPool Ecosystem**

SOFTWARE

- Wide software **kernel library**
- Bare-metal **runtime**, OpenMP, Halide
- **GCC** and **LLVM** toolchain support
- Support for GVSOC and Banshee **platform emulators**

HARDWARE

BACKEND

- Mature backend flow in many **modern technologies**
- **2 tapeouts**

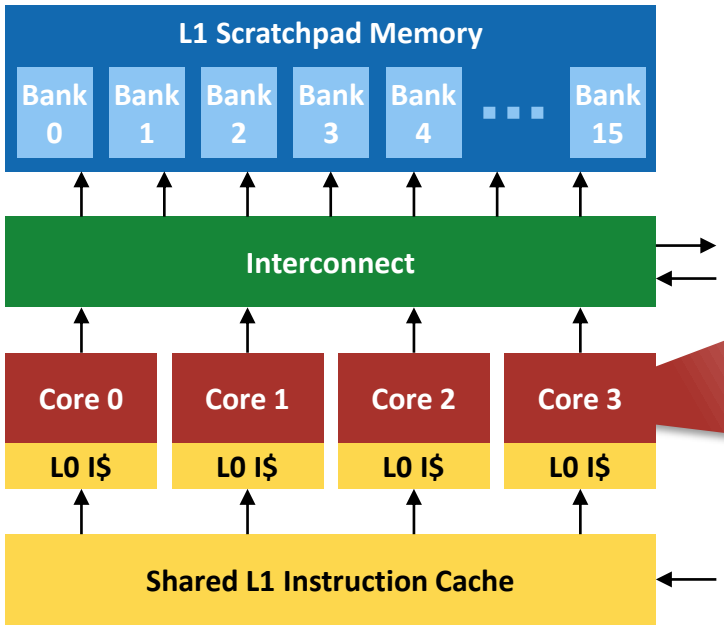
MinPool (2021)
16 cores, 200 MHz
TSMC65, 2.4mm x 2.4mm

Heartstream (2024)
64 cores, 720 MHz
GF12, 2.5mm x 2mm

Baseline MemPool

- **4** 32-bit RISC-V cores
- **16** SPM banks
- Single-cycle memory access

MemPool **Tile**



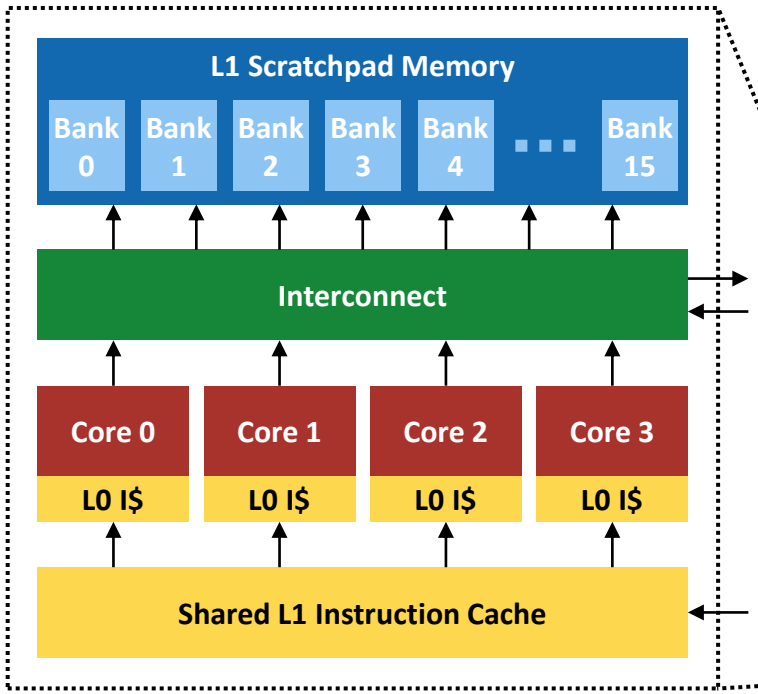
- Lean RISC-V **32b Snitch** individually programmable
- Extensible, open ISA
  - Integer **DSP**
  - **Floating-point**
- Lightweight scoreboard
  - **Latency-tolerant** LSU

Baseline MemPool

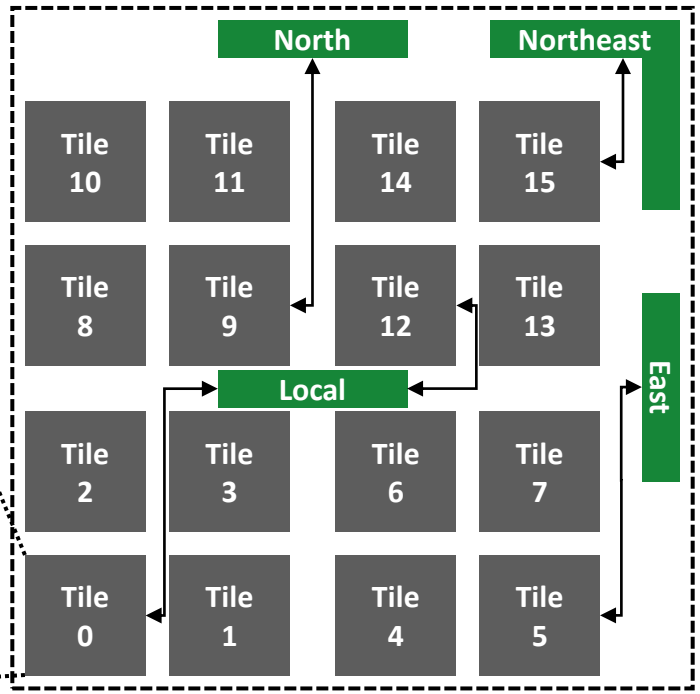
- **4** 32-bit RISC-V cores
- **16** SPM banks
- Single-cycle memory access

- **64** cores
- **256** SPM banks
- 3-cycle latency

MemPool **Tile**



MemPool **Group**



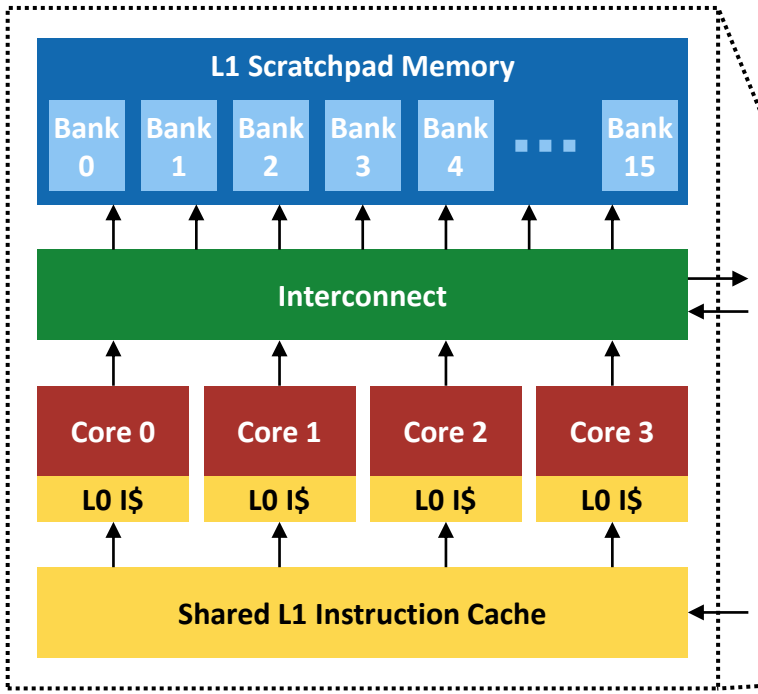
Baseline MemPool

- **4** 32-bit RISC-V cores
- **16** SPM banks
- Single-cycle memory access

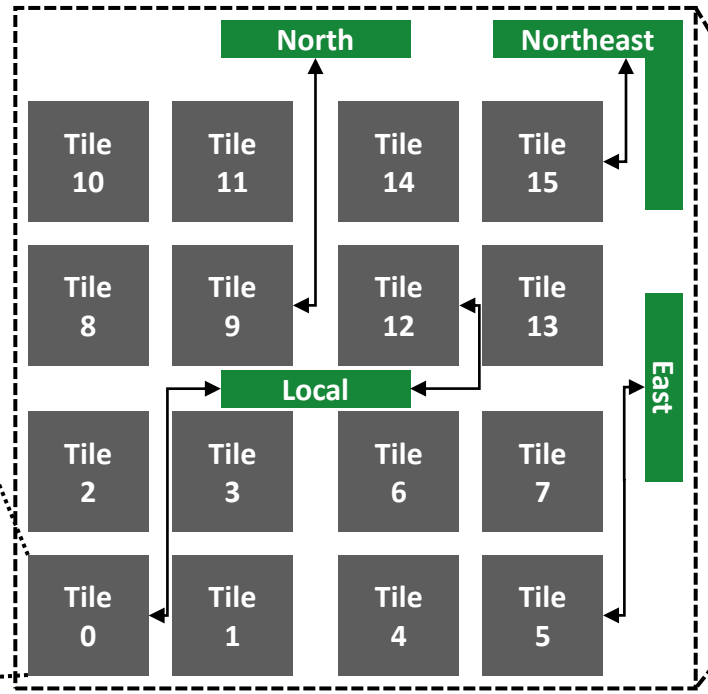
- **64** cores
- **256** SPM banks
- 3-cycle latency

- **256** cores
- **1024** SPM banks (**1 MiB**)
- 5-cycle latency

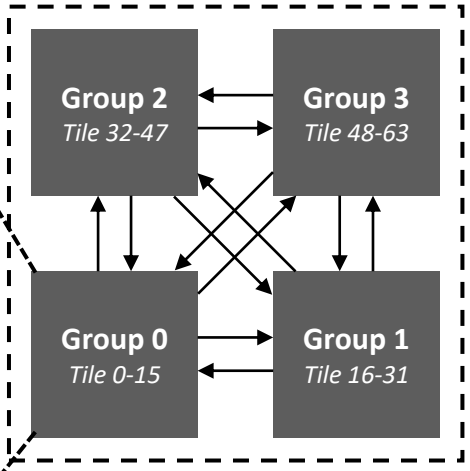
MemPool **Tile**



MemPool **Group**

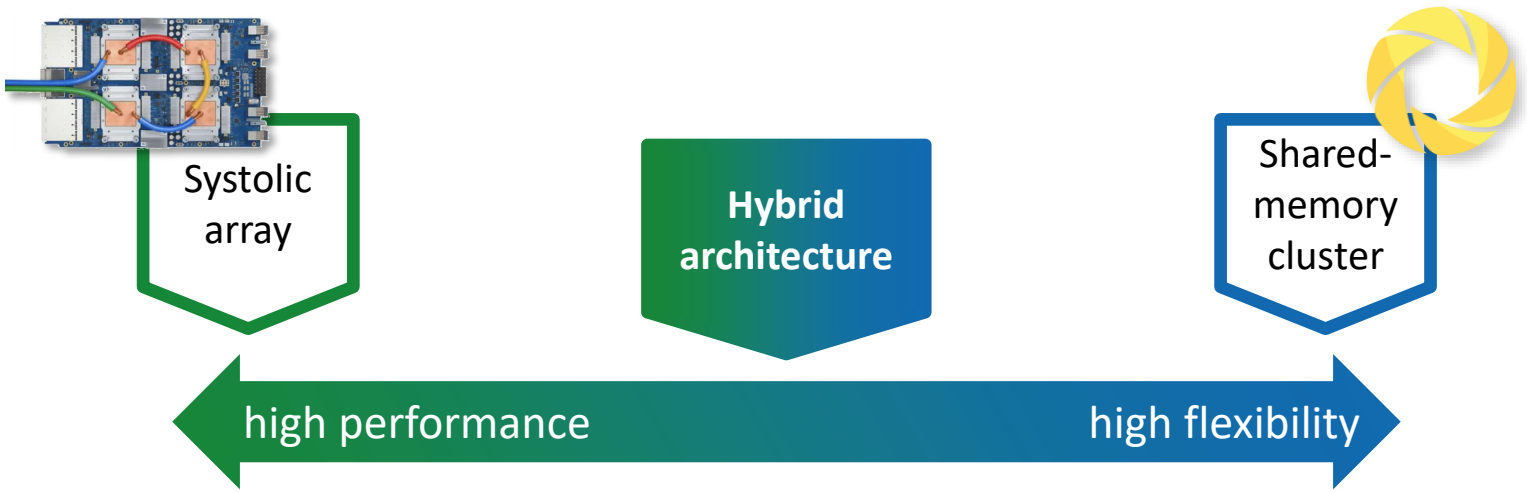


MemPool **Cluster**



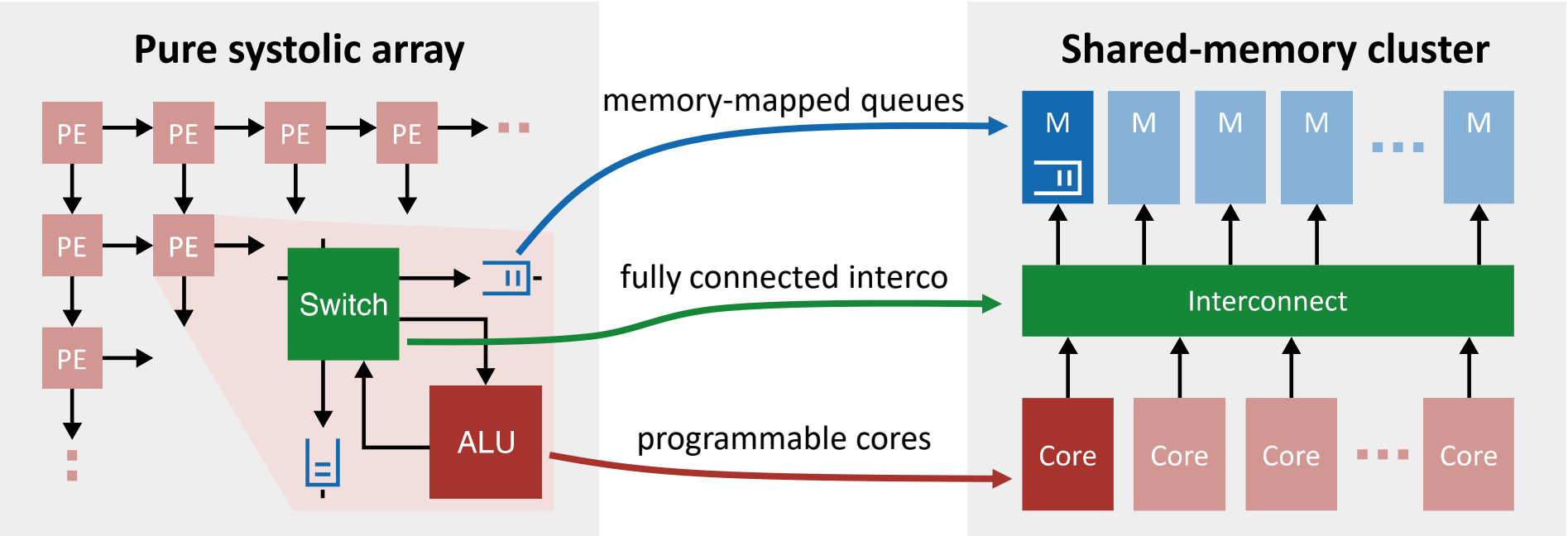
Systolic MemPool

- **Efficient systolic operation mode in shared-memory**
  - Leverage **regular dataflow** of systolic workloads
  - Keep the **flexibility** of a shared-memory system



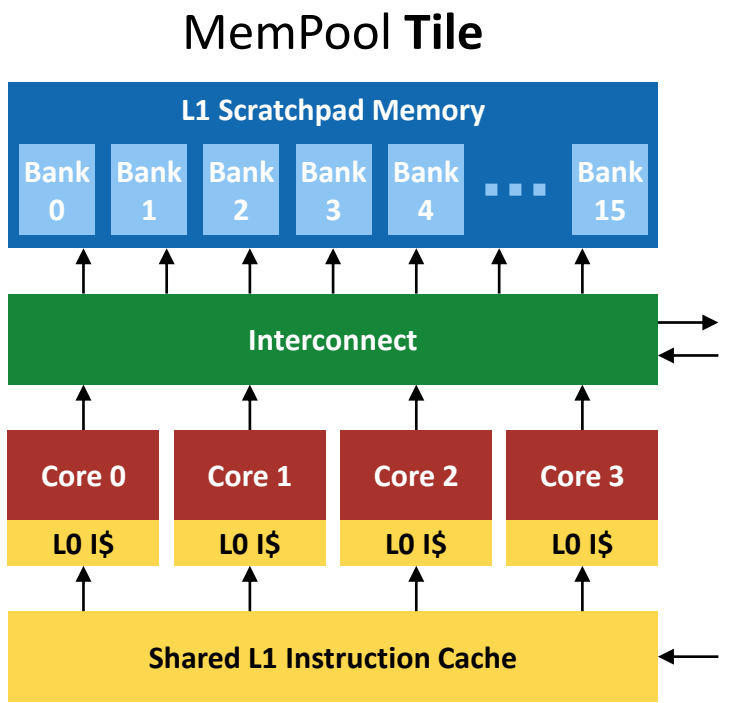
Systolic MemPool

- **Efficient systolic operation mode in shared-memory**
  - Leverage **regular dataflow** of systolic workloads
  - Keep the **flexibility** of a shared-memory system



Systolic MemPool

- **Low-overhead ISA extensions for**

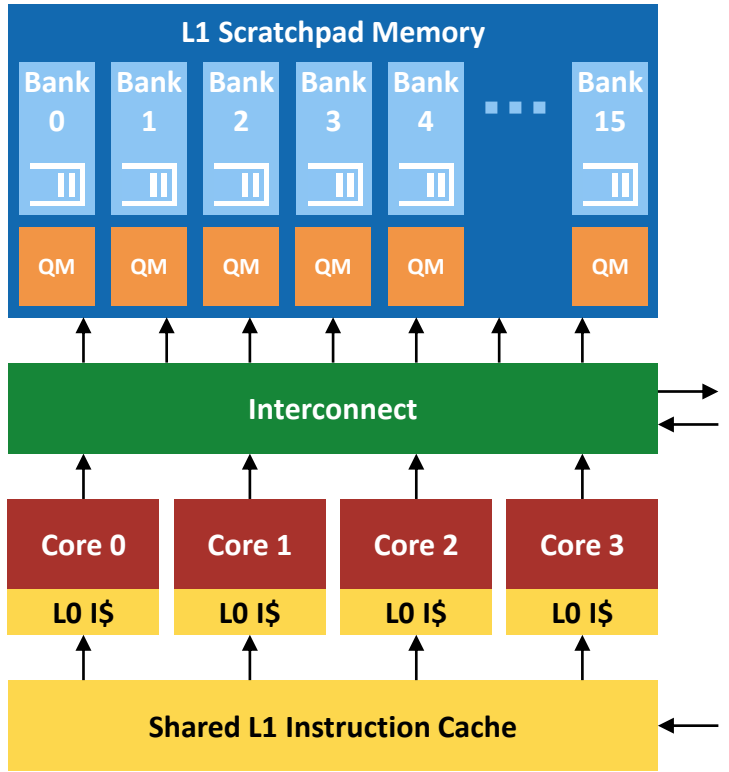


Systolic MemPool

- **Low-overhead ISA extensions for**
  - 1-cycle access to memory-mapped queues

Removes **queue management** overhead

MemPool **Tile**



Systolic MemPool

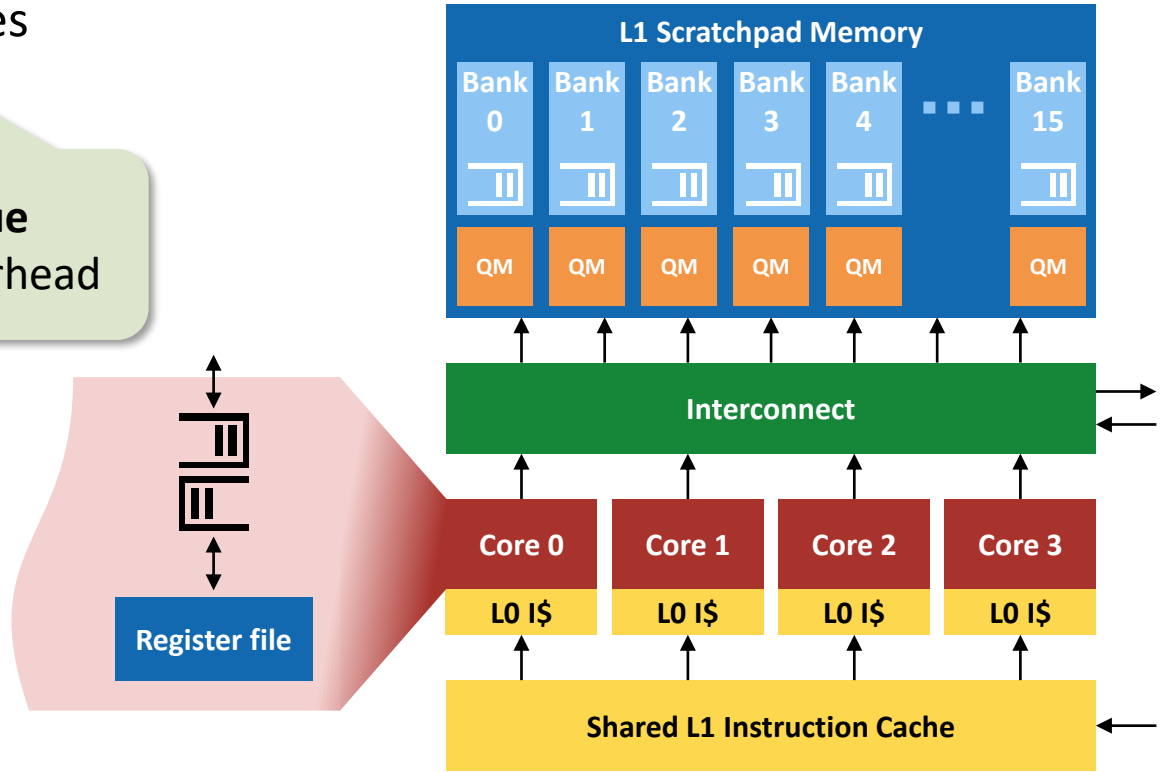
- **Low-overhead ISA extensions for**
  - 1-cycle access to memory-mapped queues
  - Autonomous access to queues

Removes **queue management** overhead

Elides loads & stores: completely **autonomous dataflow**

- **Any systolic topology**
- **Reconfigurable at runtime**

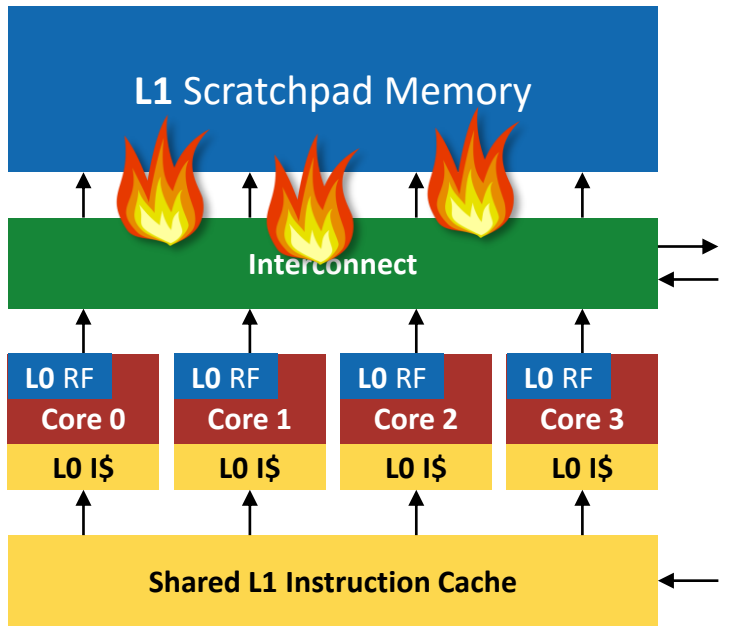
MemPool **Tile**



Vectorial MemPool

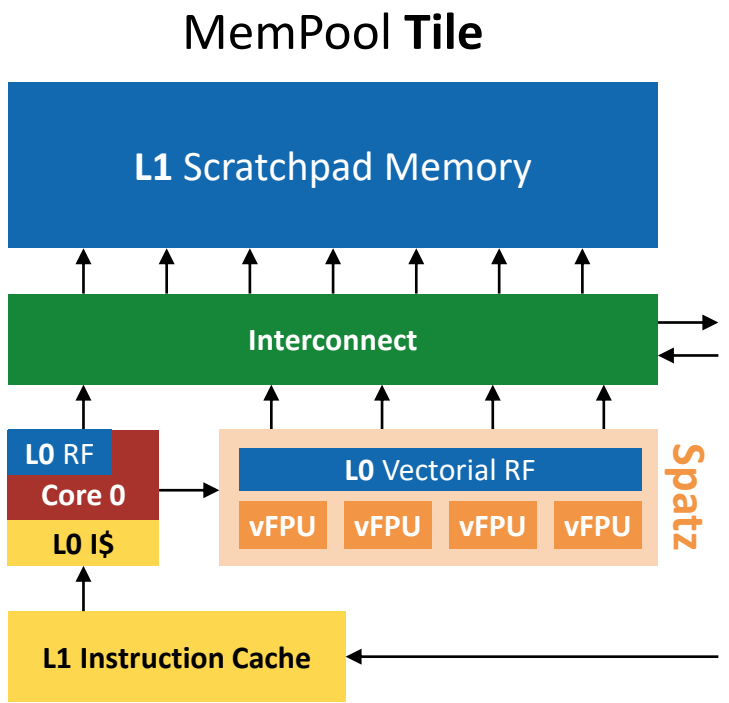
- **Intuition**
  - Von Neumann bottleneck
  - Trade off **larger L0** size for **lower L1 bandwidth**
  - Higher data reuse **closer** to functional units
- **Data L0 = core's register file**
  - Scalar architecture: *L0 size not a knob*
  - **Vectorial** architecture: *VRF flexible by design*

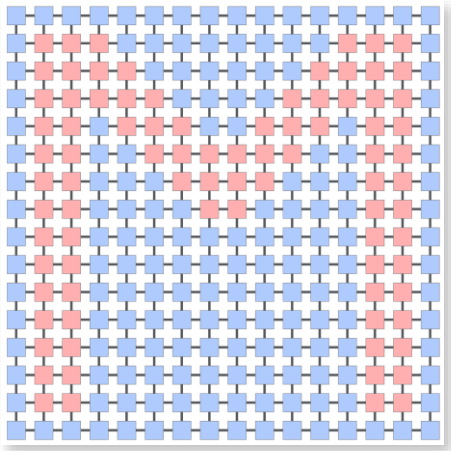
MemPool **Tile**



Vectorial MemPool

- **Spatz**
  - Fighting the von Neumann Bottleneck with a flock of **short-vector** machines for **DLP (SIMD)**
  - Based on **RVV ISA**
  - Multiple, lane-dedicated memory ports
  - Optimized sparse accesses (gather/scatter)
- **64 cores, 256 vect FPUs**



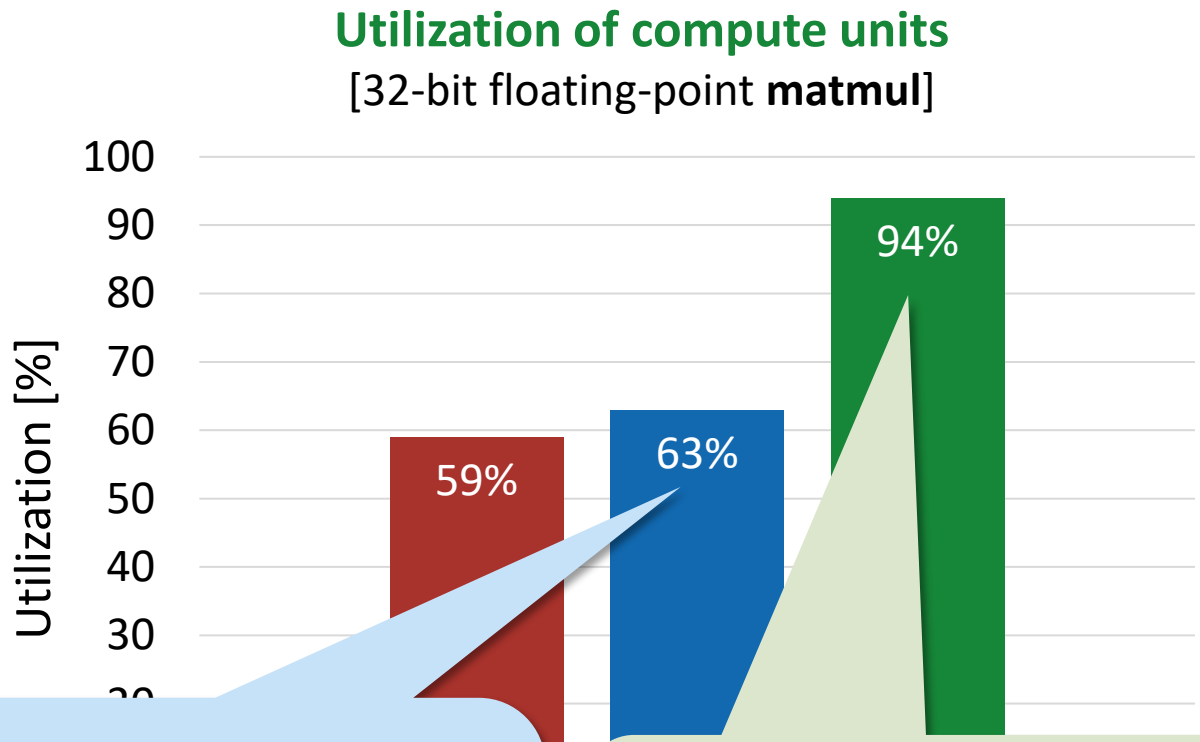




Flavor Tasting

aka, the result section...

Matmul again… what flavor to pick?

- **Baseline MemPool**
- **Systolic MemPool**
  - 256 cores
  - 256 FPUs & Int DSP units
- **Vectorial MemPool**
  - 64 cores
  - 64 Vector Units
    - = 256 FP lanes



7% better utilization!

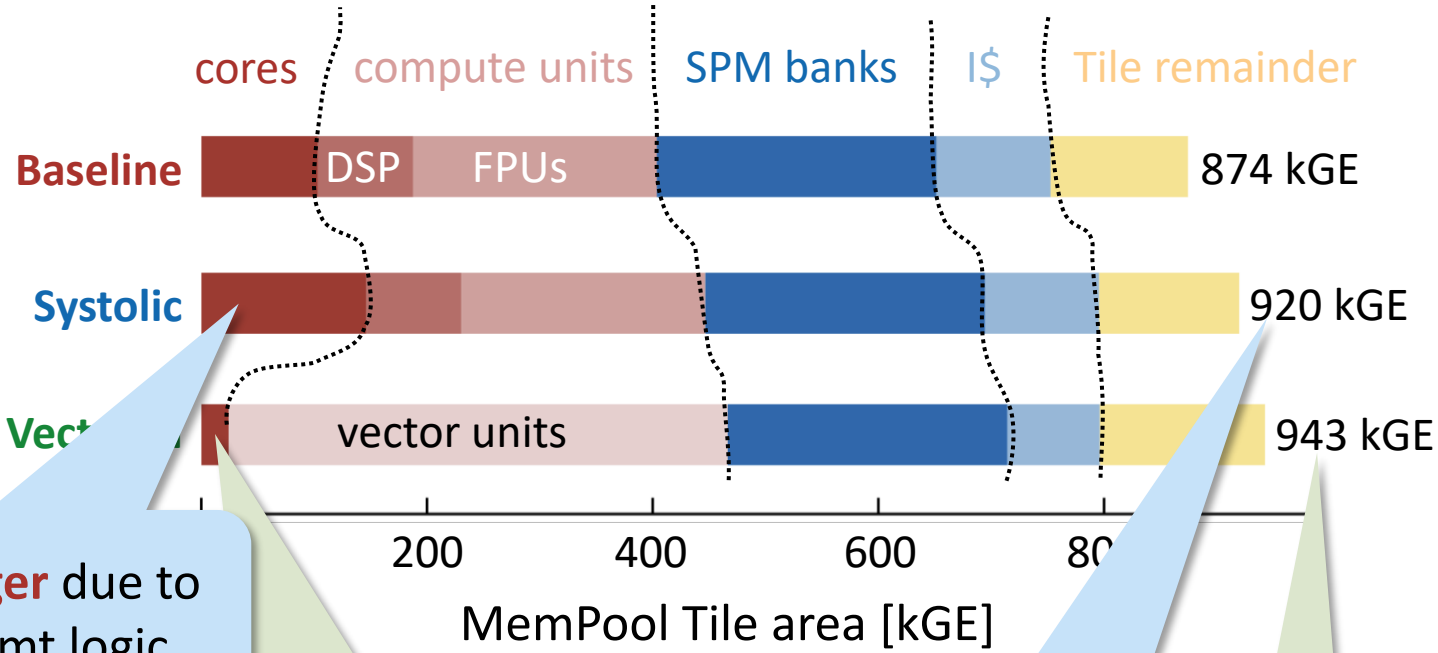
- No **control** instructions (autonomous load&store)
- Implicit **synchronization** (queues)
- Reduced memory **conflicts**

60% better utilization!

- Reduce instructions to fetch (**SIMD**)
- Reduced L1 **memory access**

...It depends on what you want: cost trade-off (**area**)

- **Baseline MemPool**
- **Systolic MemPool**
  - 256 cores
  - 256 FPUs & Int DSP units
- **Vectorial MemPool**
  - 64 cores
  - 64 Vector Units
    - = 256 FP lanes



Cores **40% bigger** due to FIFOs and mgmt logic

Only 1 core per Tile: **reduced flexibility** for non-vectorial workloads

Tile **only 5%** bigger

8% bigger

Technology & target
GF 12nm FinFET
800 MHz, worst-case corner

MemPool: An open-source, RISC-V research platform

- **Scaled-up, flexible shared-L1 manycore cluster covering a large trade-off space**
  - From classic scaled-up shared-memory cluster…
  - …to exotic systolic array architectures
  - but always with an eye on **versatility** and **programmability**
- **(Very) active development since 2020**
  - Large ecosystem including kernel **libraries**, **toolchains**, **emulators**
  - Tapeouts in **TSMC65** and **GF12**
  - A plethora of publications

`github.com/pulp-platform/mempool`

2021

- *MemPool: A Shared-L1 Memory Many-Core Cluster with a Low-Latency Interconnect*
- *3D SoC integration, beyond 2.5D chiplets*

2022

- *MemPool-3D: Boosting Performance and Efficiency of Shared-L1 Memory Many-Core Clusters with 3D Integration*
- *Hier-3D: A Hierarchical Physical Design Methodology for Face-to-Face-Bonded 3D ICs*
- *Spatz: A Compact Vector Processing Unit for High-Performance and Energy-Efficient Shared-L1 Clusters*
- *Thermal Performance Analysis of Mempool RISC-V Multicore SoC*

2023

- *Towards Chip-Package-System Co-optimization of Thermally-limited System-On-Chips (SOCs)*
- *Efficient Parallelization of 5G-PUSCH on a Scalable RISC-V Many-Core Processor*
- *MemPool Meets Systolic: Flexible Systolic Computation in a Large Shared-Memory Processor Cluster*
- *Fast Shared-Memory Barrier Synchronization for a 1024-Cores RISC-V Many-Core Cluster*
- *MemPool: A Scalable Manycore Architecture with a Low-Latency Shared L1 Memory*
- *Impact of 3-D Integration on Thermal Performance of RISC-V MemPool Multicore SOC*
- *MinPool: A 16-core NUMA-L1 Memory RISC-V Processor Cluster for Always-on Image Processing in 65nm CMOS*

2024

- *LRSCwait: Enabling Scalable and Efficient Synchronization in Manycore Systems through Polling-Free and Retry-Free Operation*
- *Enabling Efficient Hybrid Systolic Computation in Shared L1-Memory Manycore Clusters*
- *MX: Enhancing RISC-V's Vector ISA for Ultra-Low Overhead, Energy-Efficient Matrix Multiplication*
- *TeraPool-SDR: An 1.89TOPS 1024 RV-Cores 4MiB Shared-L1 Cluster for Next-Generation Open-Source Software-Defined Radios*