ETH zürich

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

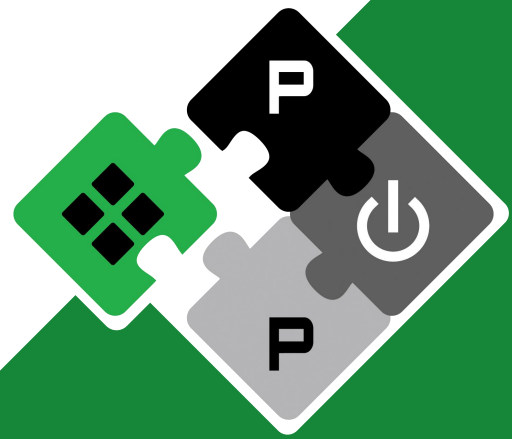
CachePool: Many-core cluster of customizable, lightweight scalar-vector PEs for irregular L2 data-plane workloads

Integrated Systems Laboratory (ETH Zürich)

Zexin Fu, Diyou Shen zexifu, dishen@iis.ee.ethz.ch

Alessandro Vanelli-Coralli avanelli@iis.ee.ethz.ch

Luca Benini lbenini@iis.ee.ethz.ch



@pulp_platform

pulp-platform.org

youtube.com/pulp_platform

PULP Platform

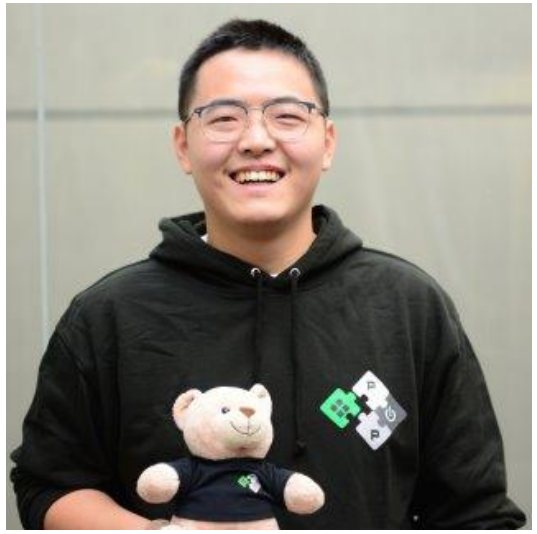
Open Source Hardware, the way it should be!

Team Introduction

- **Zexin Fu**
  - Education

Bachelor	Electrical Engineering	Huazhong University of Sci & Tech
Master	Data Sci & Info Tech	Tsinghua University
PhD	PULP	ETH Zürich

  - Research Experience
    - Manycore
    - Network-on-Chip
    - Cache Coherence
    - RISC-V Core
    - Fault Tolerant Cache Design



Team Introduction

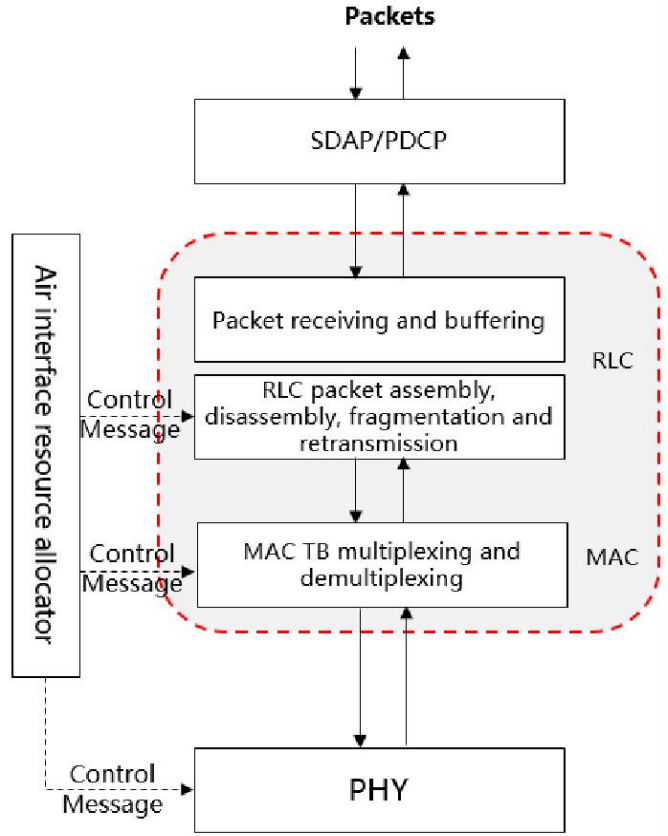
- **Diyou Shen**
  - Education

Bachelor	Electrical Engineering	University of Colorado Boulder
Master	EEIT	ETH Zürich
PhD	PULP	ETH Zürich

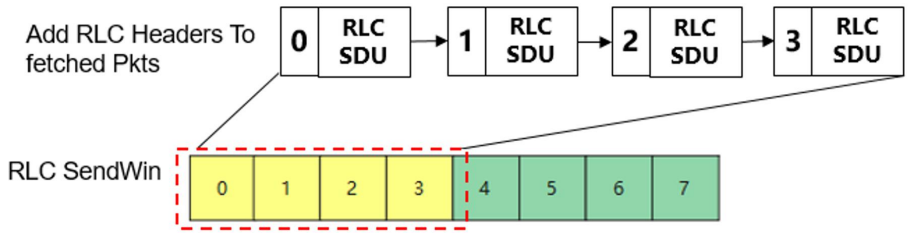
  - Research Experience
    - Manycore Cluster
    - Vector Processor
    - Fault Tolerant Interconnection
    - Cache Design

Target Workload

- **Data Plane Protocol**
  - **Data structure**
    - **Tasks dependency**: independent between each user
    - **Packet count per user**: high variance.
    - **Code footprint**: small
    - **Data footprint**: large, sparse, non-contiguous (e.g., linked lists)
  - **Performance target**
    - Total throughput: 10M packets/sec
    - Each RLC throughput: >1k packets/sec
    - Active RLC: > 1k
    - TTI: 62.5 us - 1ms, (typical 500 us)
  - **Operations**
    - No FP requirement
    - Mixed scalar & vector operations



Data-Plane Protocol Subsystem



RLC Data Structure

Hardware Design Guideline

- Data structure
  - Small code footprint -> **Small Instruction Cache**
  - **Sparse**, non-contiguous data footprint -> **Data cache**
  - Large address range for each user -> **Coherence support**
- Performance target
  - High throughput -> **Manycore cluster**
  - Tight TTI -> **Low latency interconnect**
- Operations
  - **Scalar** & **Vector** operations

Manycore

+

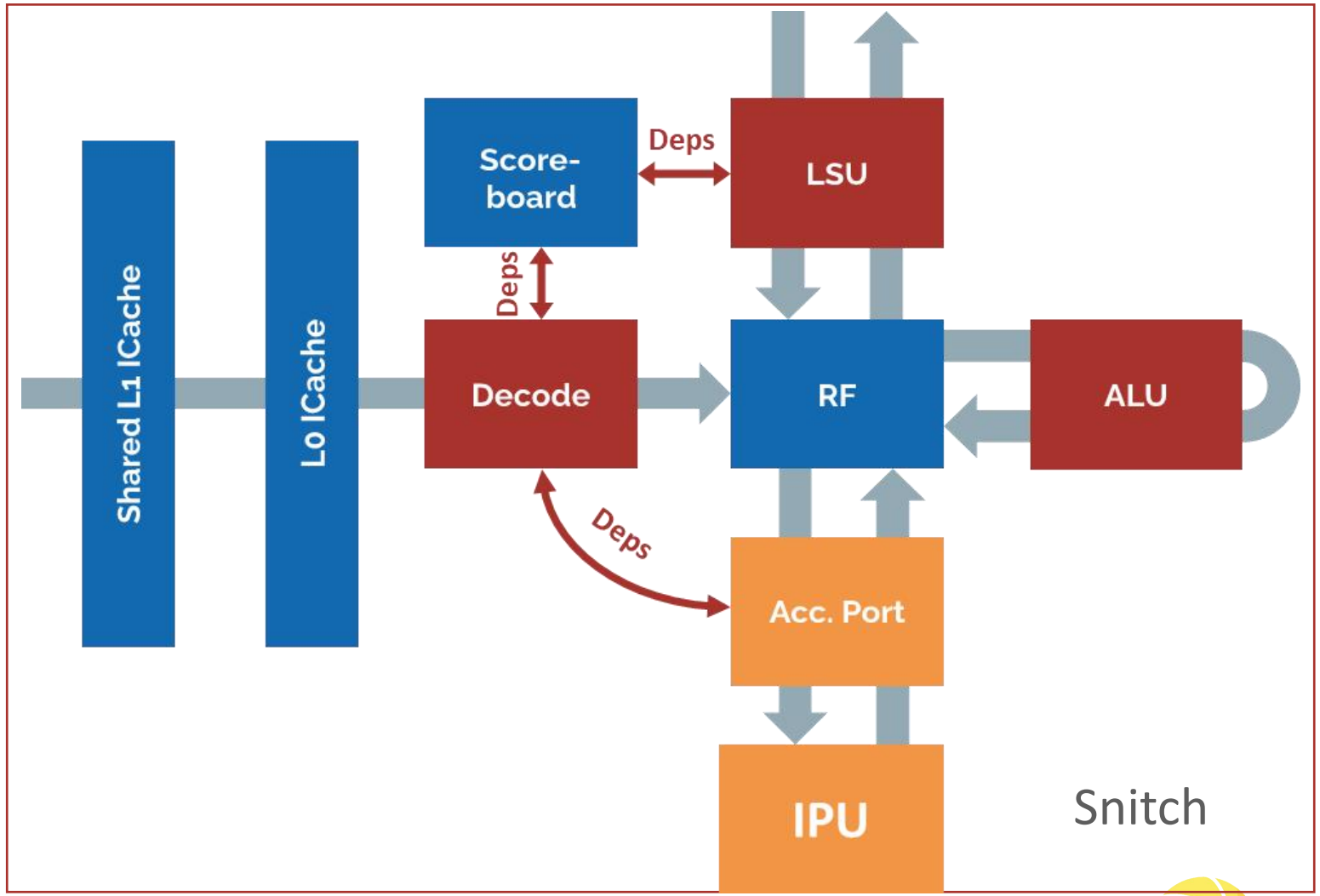
Cache

+

Vector

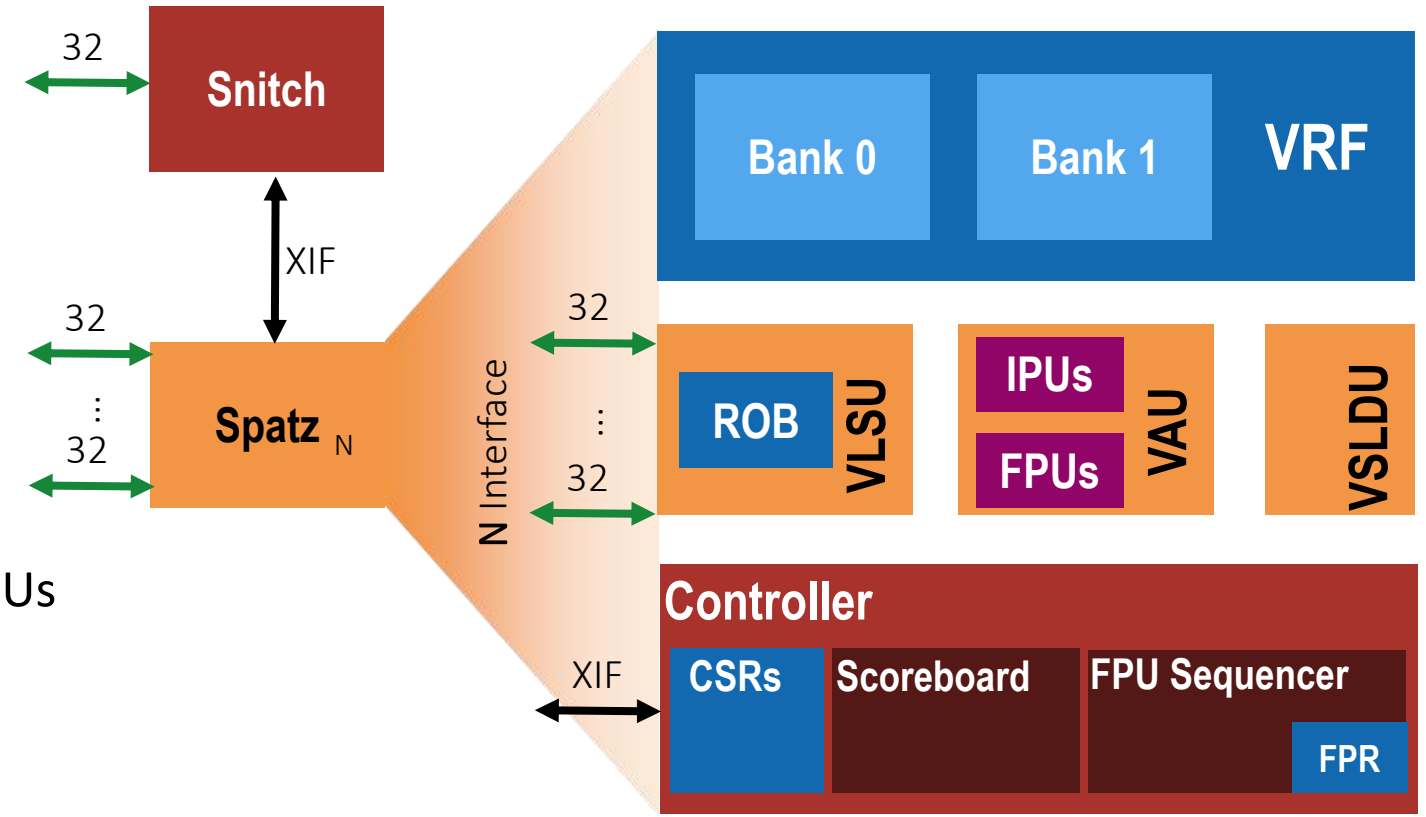
Processing Elements

- **Snitch Scalar Core**
  - RV32IMA
  - Lightweight - Scalable
  - Latency-tolerant
    - Out-of-order memory access
    - ROB



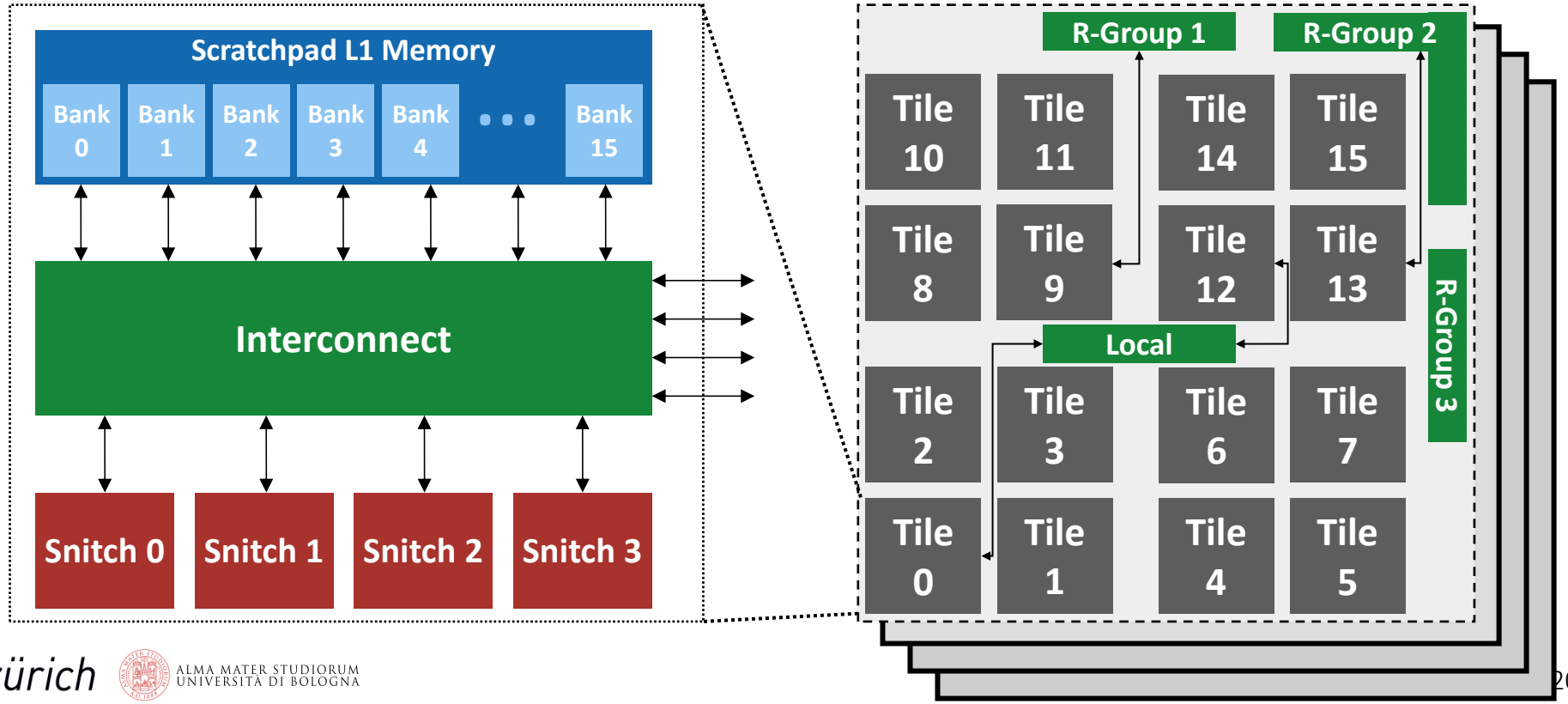
Processing Elements

- **Snitch Scalar Core**
  - RV32IMA
  - Lightweight - Scalable
  - Latency-tolerant
    - Out-of-order memory access
    - ROB
- **Spatz Vector Core**
  - Based on RVV
  - Configurable number of FPU/IPUs
  - Latency-tolerant
    - OoO memory access
    - ROB



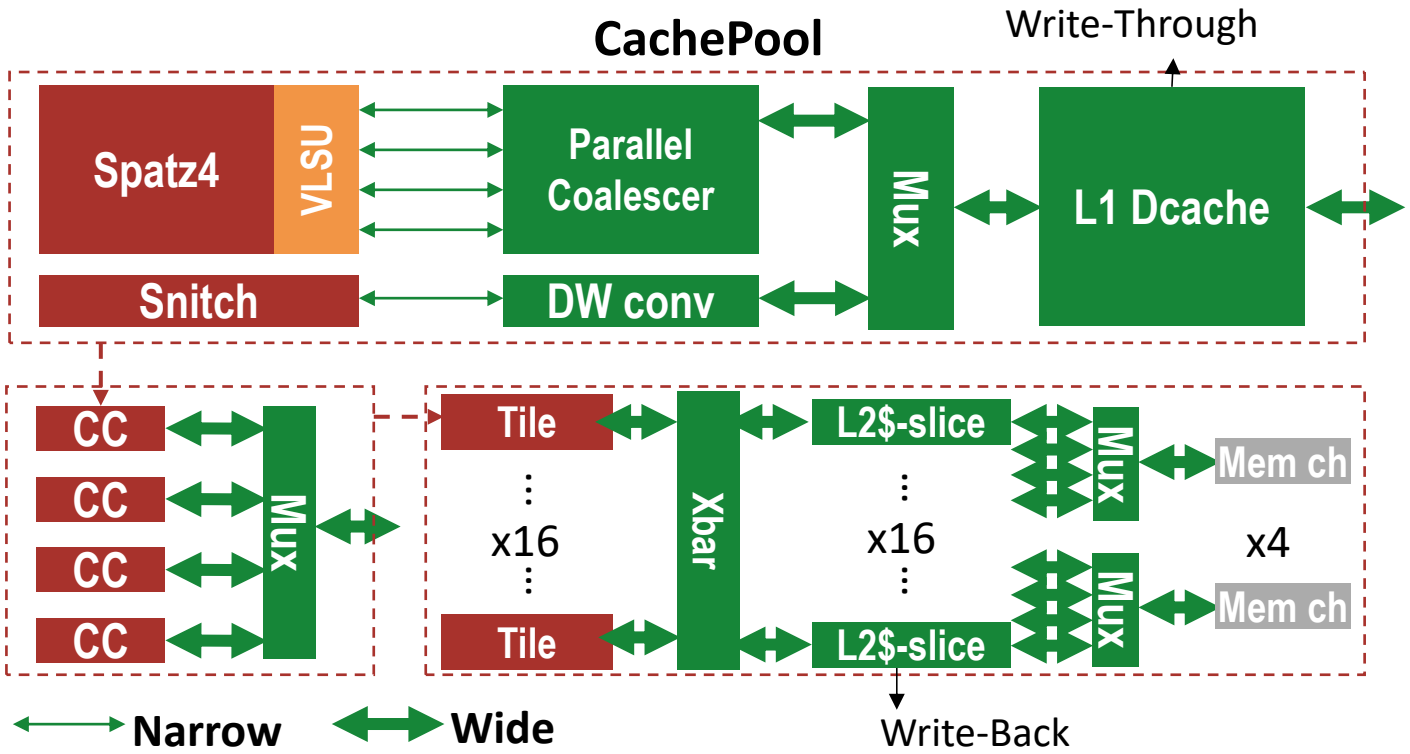
Our Baseline: MemPool

- **MemPool Family: scalable many-core shared L1-TCDM cluster**
  - Physically Feasible, scale-up to 1024 **extendable tiny RISC-V** cores (TeraPool-SDR)
  - We can replace the Snitch cores to Spatz core complexes to support **Vector** insn.
  - SPM needs to be replaced by **Cache** for sparse data pattern.



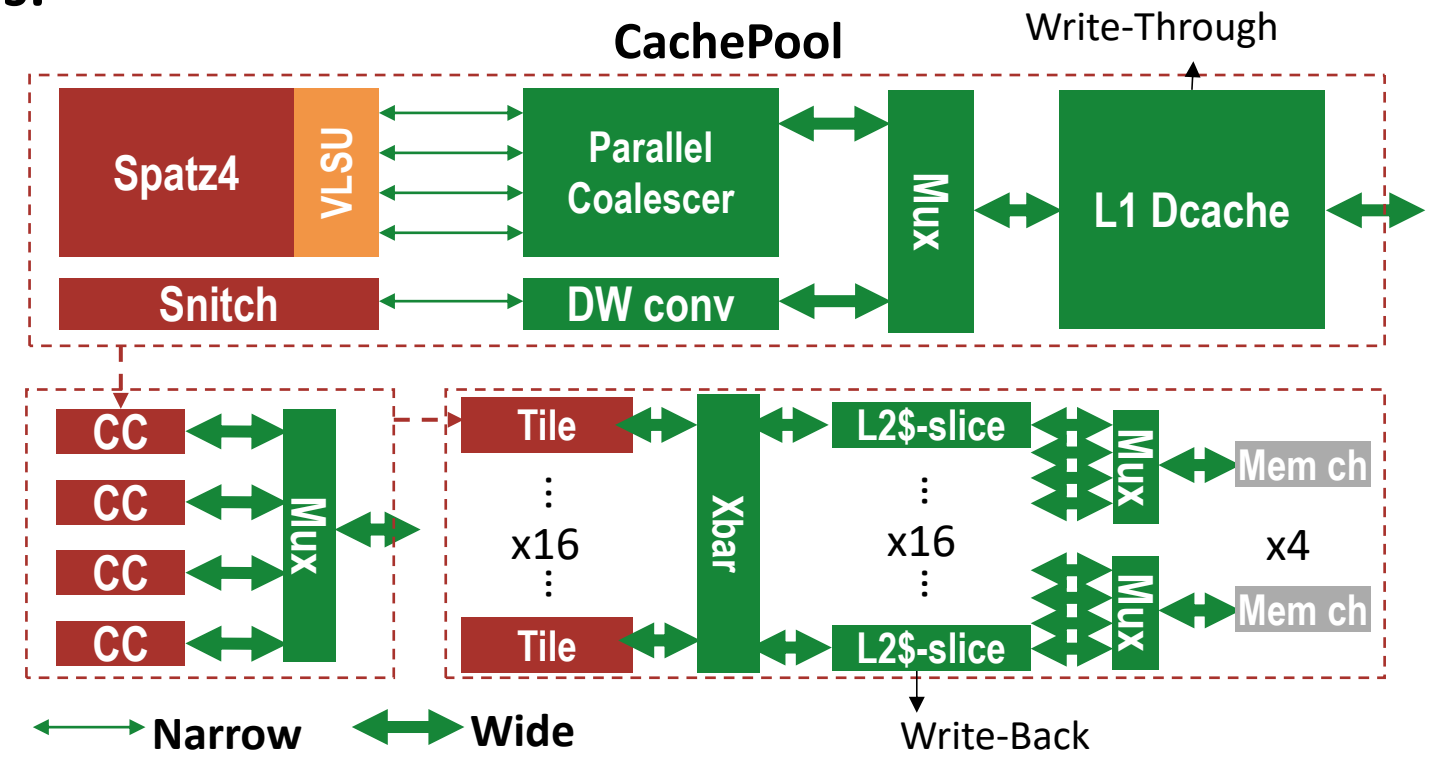
CachePool: Cache-based, Manycore, Vector-Scalar PEs

- **Architecture Features:**
  - Scalable manycore heterogeneous cluster
    - Configurable number of PEs
    - Configurable scalar-vector ratio
    - Configurable FP support
  - Cache-based
    - Fitting sparse data pattern
    - Using coalescer to utilize burst BW
    - Write-through L1 to ease coherence design

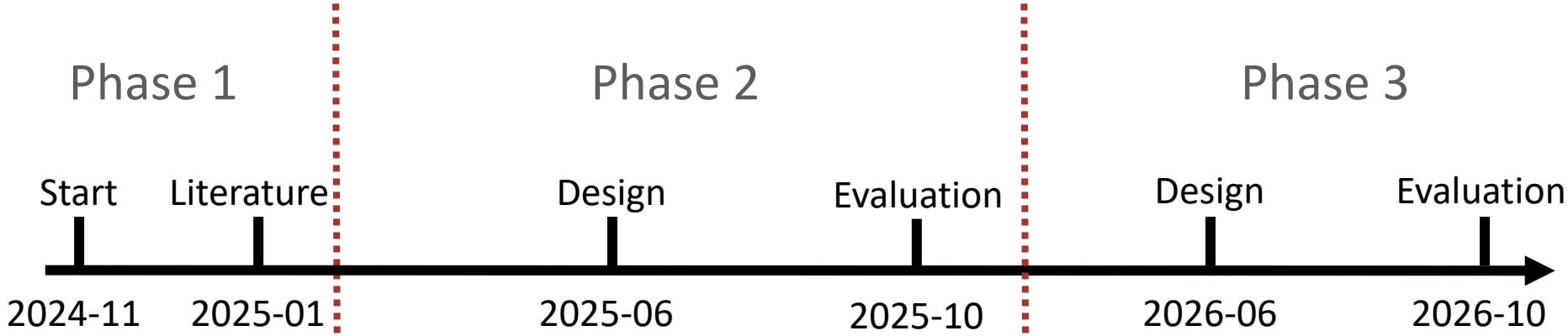


CachePool: Cache-based, Manycore, Vector-Scalar PEs

- **Architecture Research Directions:**
  - Cluster configuration exploration
  - L1 D-cache, or L1-less, coalescer exploration and design
  - Shared L1 per tile/cc? Cache coherence intra-tile?
  - Interconnection design for large-L2 cache slices.
  - Scalar / Vector processing ratio



Preliminary Timeline



Literature study on State-of-Art

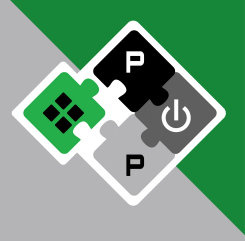
- Complete the initial design of manycore cluster, cache and vector processing element.
- Evaluate current architecture.

- Complete the optimization of the hardware based on previous results.
- Evaluate the final architecture.

Open Discussion

- Discussion Points:
  - Details: data structure, packet size, operations, data dependency, …
  - Open-source kernel examples on Data Plane
  - Performance guideline
  - Literature suggestion

- Meeting arrangement
  - Period meeting schedule
  - Technical contacts / Communication channels



Thank you!