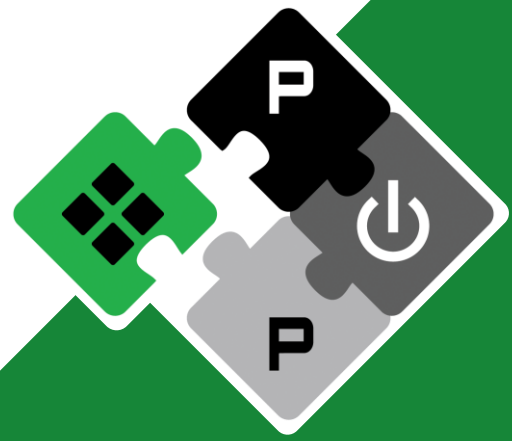
ETH zürich ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

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!

Outline

- **Work Package Progress Update**
- **Architecture Introduction: Snitch-Spatz Core Complex**
- **Hardware Development**
- **Software Development**
- **Open Discussion**

Outline

- **Work Package Progress Update**
- Architecture Introduction: Snitch-Spatz Core Complex
- Hardware Development
- Software Development
- Open Discussion

Work Package Progress

- **Phase 1 (01.11.2024 – 31.12.2024)**
  - Literature Study
- **Phase 2 (01.01.2025—31.10.2025)**
  - Manycore Architecture Design
  - Scalable Cache Design
  - Vector Processing Element Design
  - Preliminary Benchmarking
- **Phase 3 (01.11.2025—31.20.2026)**

Work Package Progress

- Phase 1 (01.11.2024 – 31.12.2024)
  - Literature Study
- **Phase 2 (01.01.2025—31.10.2025)**
  - Manycore Architecture Design
  - Scalable Cache Design
  - Vector Processing Element Design
  - Preliminary Benchmarking
- Phase 3 (01.11.2025—31.20.2026)

Work Package Progress

- **Phase 1 (01.11.2024 – 31.12.2024)**
  - Literature Study
- **Phase 2 (01.01.2025—31.10.2025)**
  - **Manycore Architecture Design**
  - **Scalable Cache Design**
  - Vector Processing Element Design
  - Preliminary Benchmarking
- **Phase 3 (01.11.2025—31.20.2026)**

Manycore & Cache Architecture Selection:

1. Shared-L1 cache with partitioning
2. Small WT private L1 with shared L2

Work Package Progress

- Phase 1 (01.11.2024 – 31.12.2024)
  - Literature Study
- **Phase 2 (01.01.2025—31.10.2025)**
  - Manycore Architecture Design
  - Scalable Cache Design
  - **Vector Processing Element Design**
  - **Preliminary Benchmarking**
- Phase 3 (01.11.2025—31.20.2026)

Vector PE integrated into the prototype cluster, benchmarking and analyzing undergoing

Work Package Progress

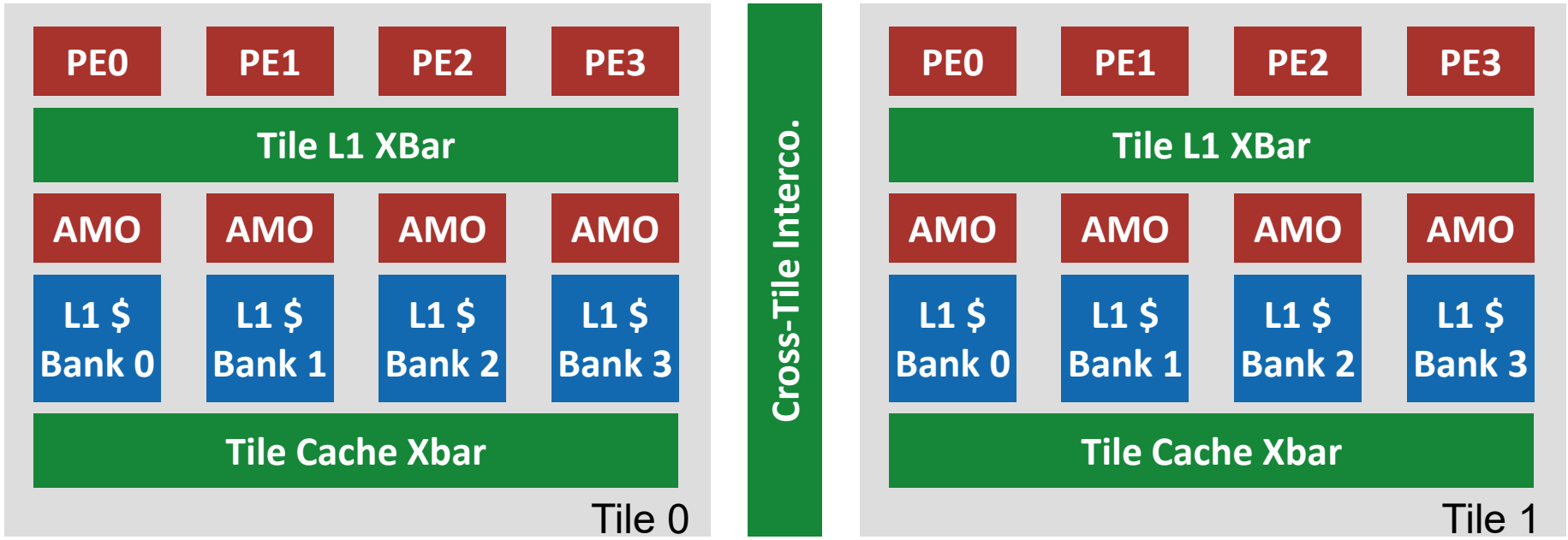
- Phase 1 (01.11.2024 – 31.12.2024)
  - Literature Study
- **Phase 2 (01.01.2025—31.10.2025)**
  - Manycore Architecture Design
  - Scalable Cache Design
  - Vector Processing Element Design
  - Preliminary Benchmarking
- Phase 3 (01.11.2025—31.20.2026)

- **Plans:**
  - Complete the prototype architecture design
    - chip-level cache interconnects (90% done)
    - change to 32b system for scalability
    - partition support
    - 4-8 weeks
  - Benchmark and evaluate the design using the agreed kernel extractions
    - Performance analysis
    - Initial backend run => physical feasibility check
    - 8 weeks
  - Based on the evaluation decide the next steps
    - Implementation of candidate 2 architecture
    - Hardware adjustments
  - Scale up the prototype to larger configurations
    - Transit to Phase 3

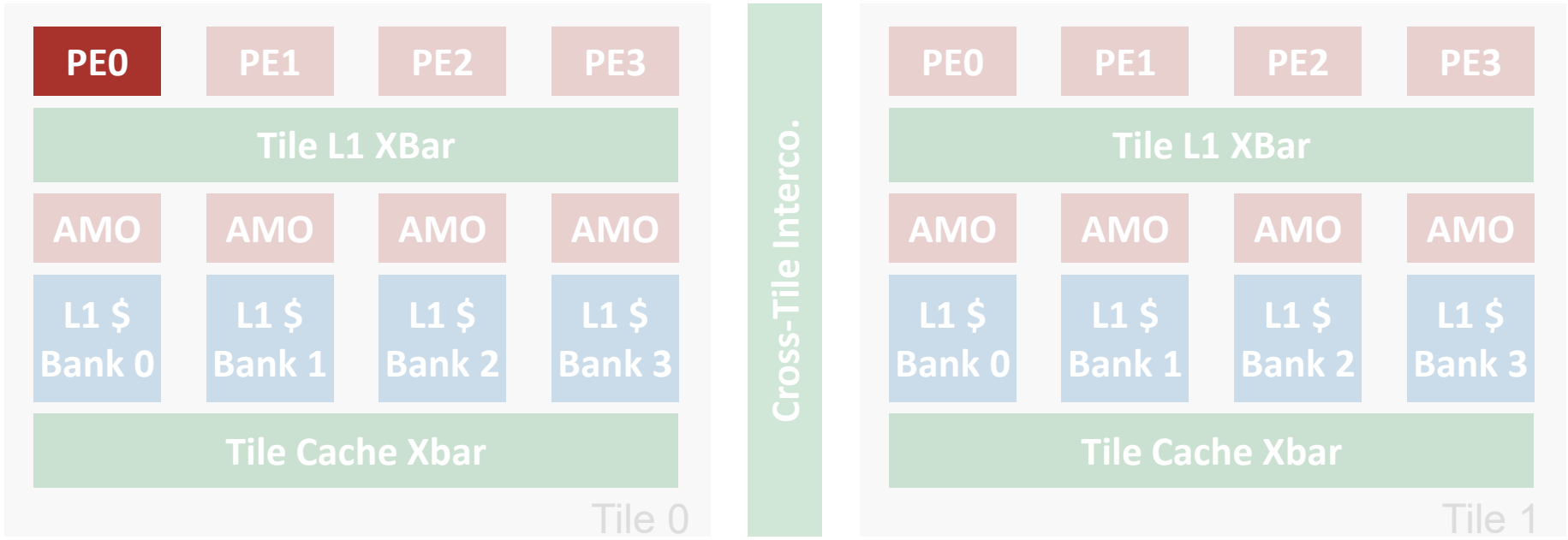
Outline

- Work Package Progress Update
- **Architecture Introduction: Snitch-Spatz Core Complex**
- Hardware Development
- Software Development
- Open Discussion

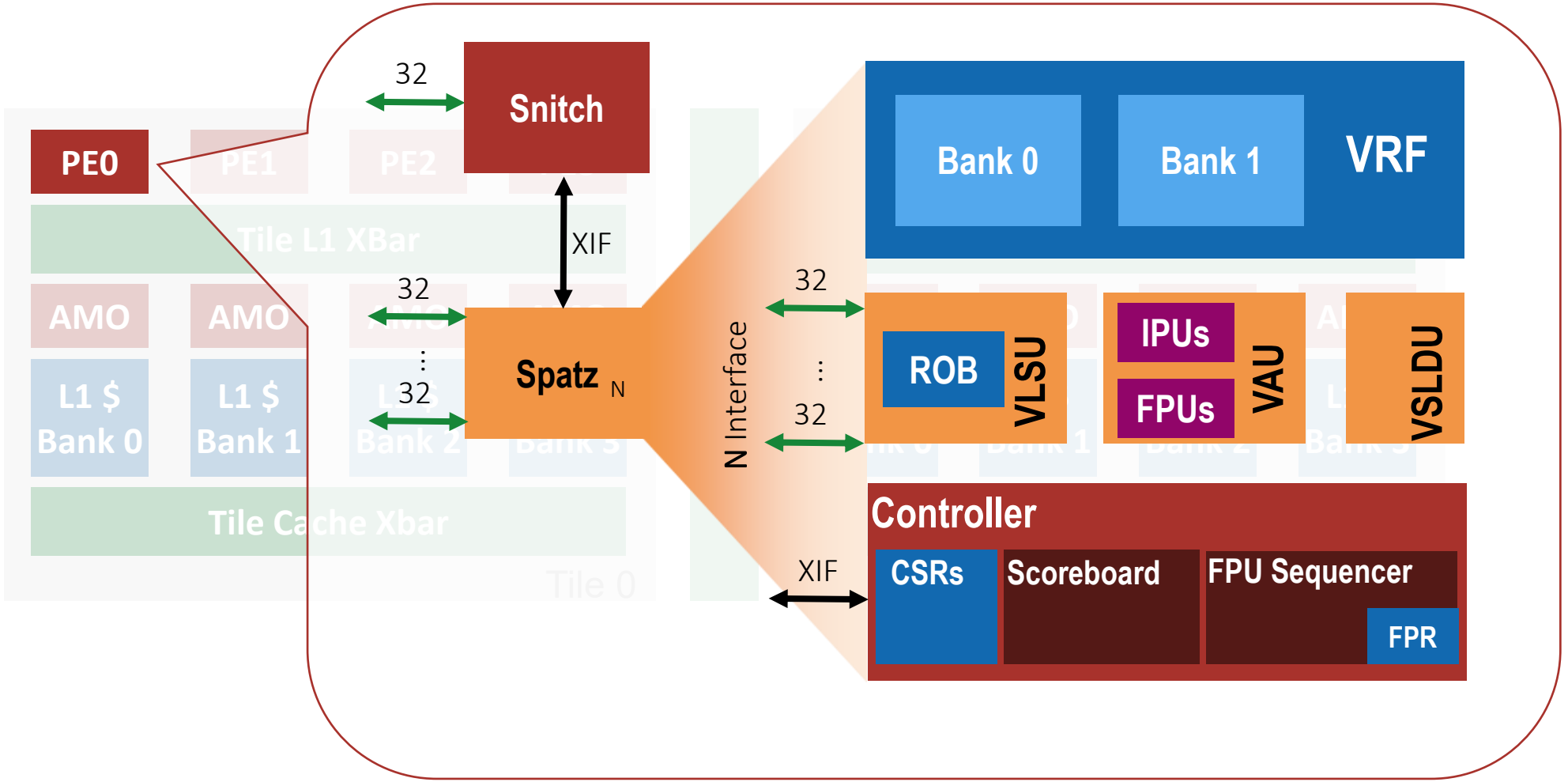
Architecture Introduction: Snitch-Spatz Core Complex



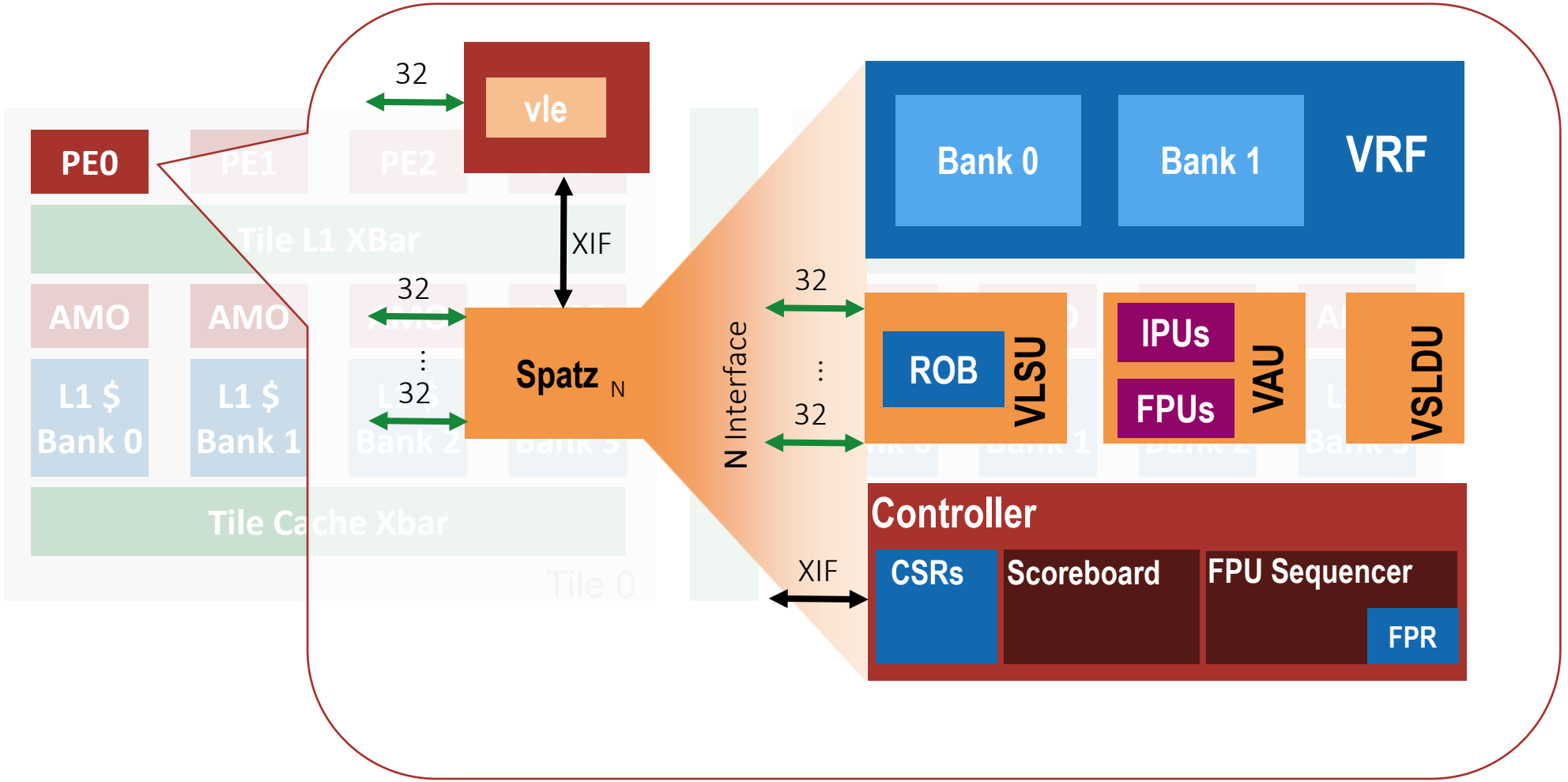
Architecture Introduction: Snitch-Spatz Core Complex



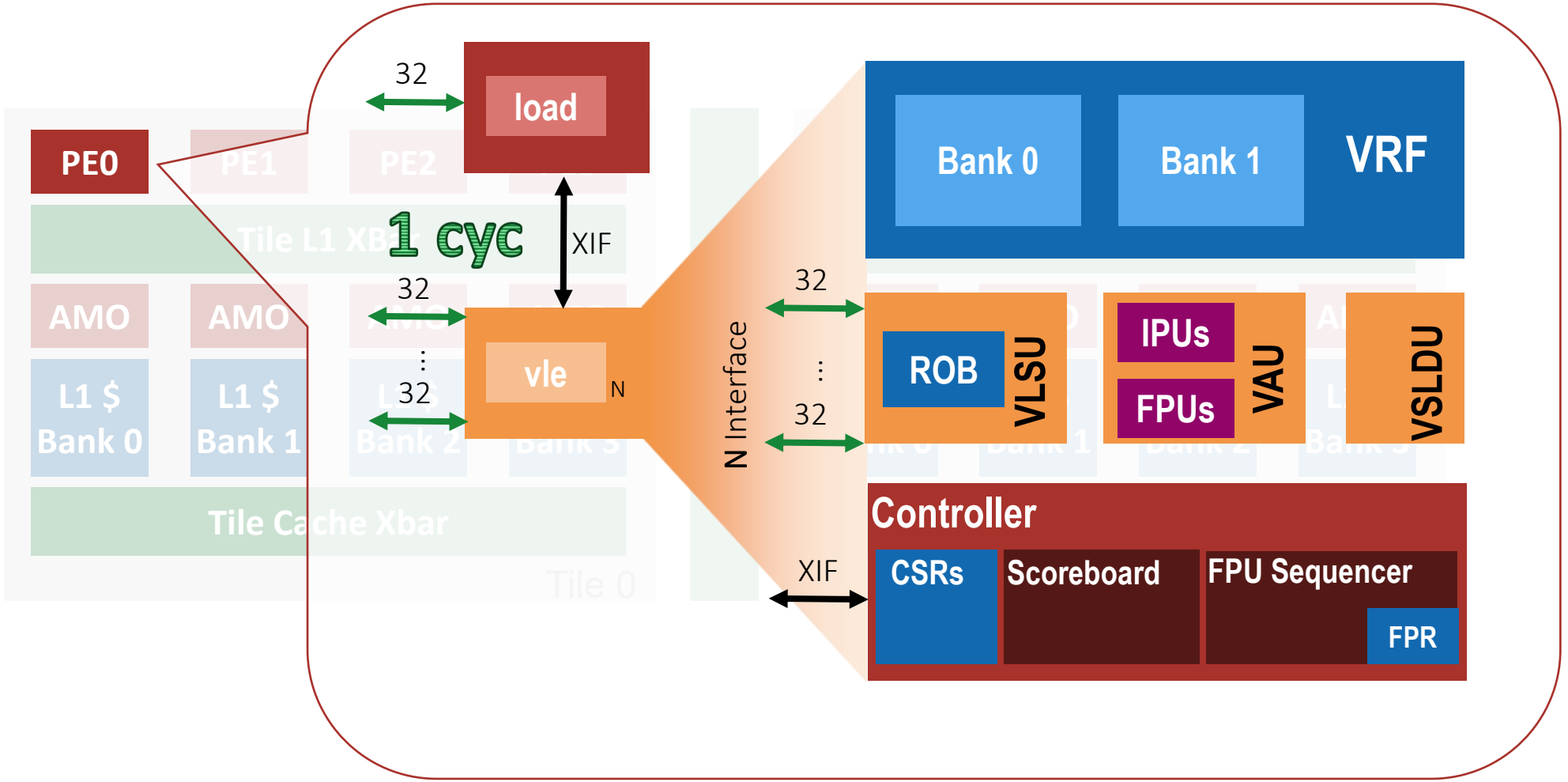
Architecture Introduction: Snitch-Spatz Core Complex



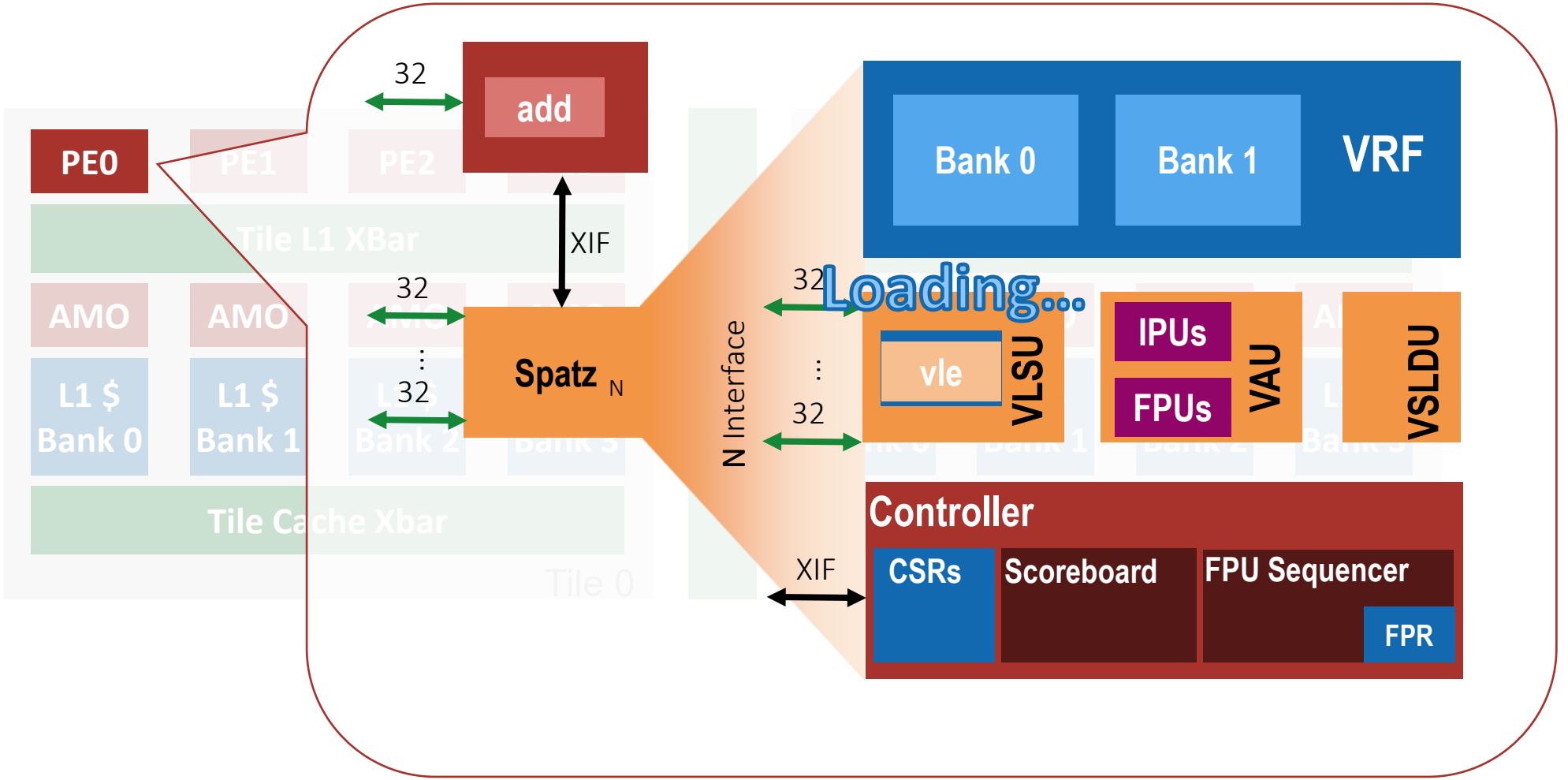
Architecture Introduction: Snitch-Spatz Core Complex



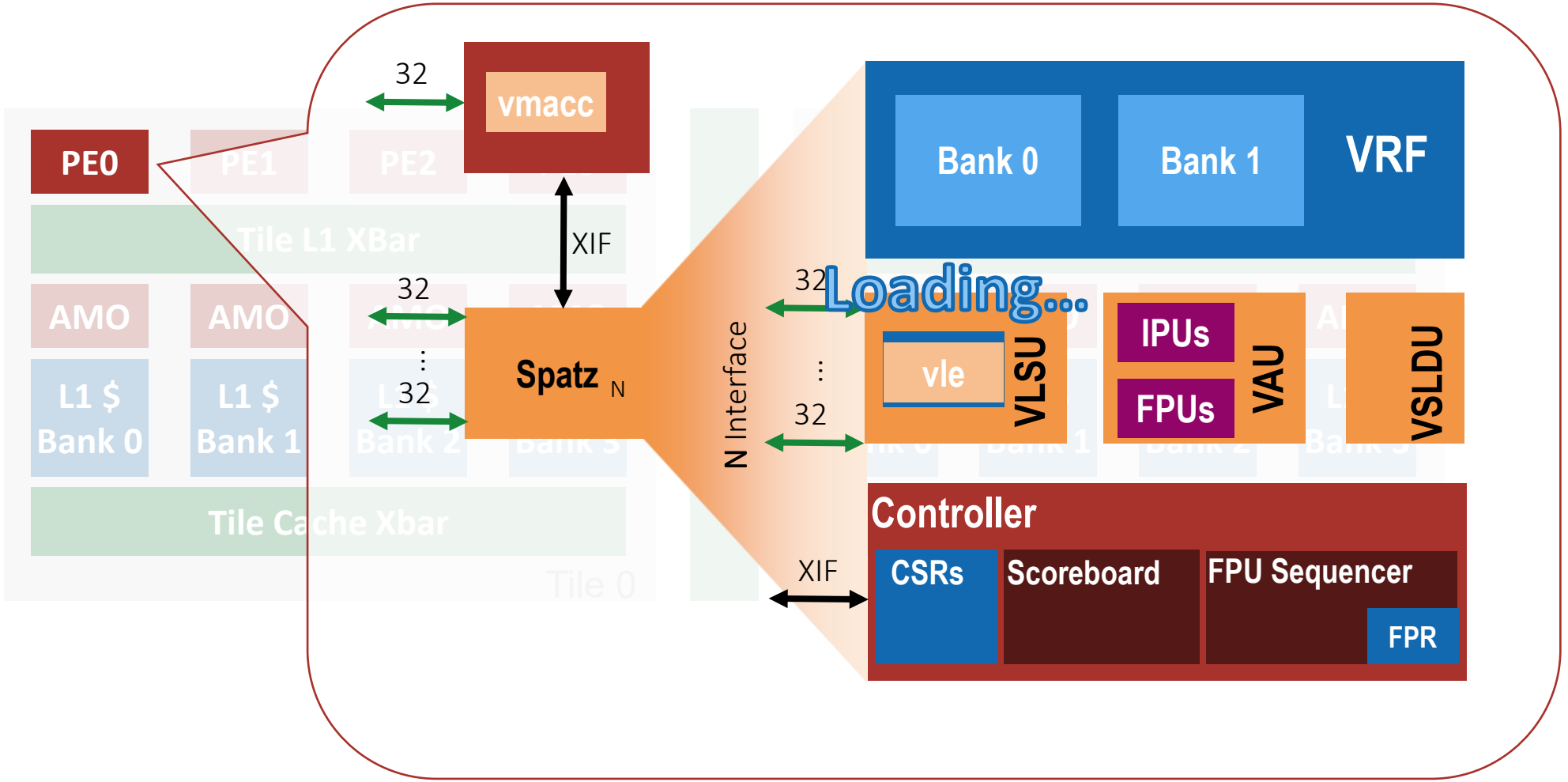
Architecture Introduction: Snitch-Spatz Core Complex



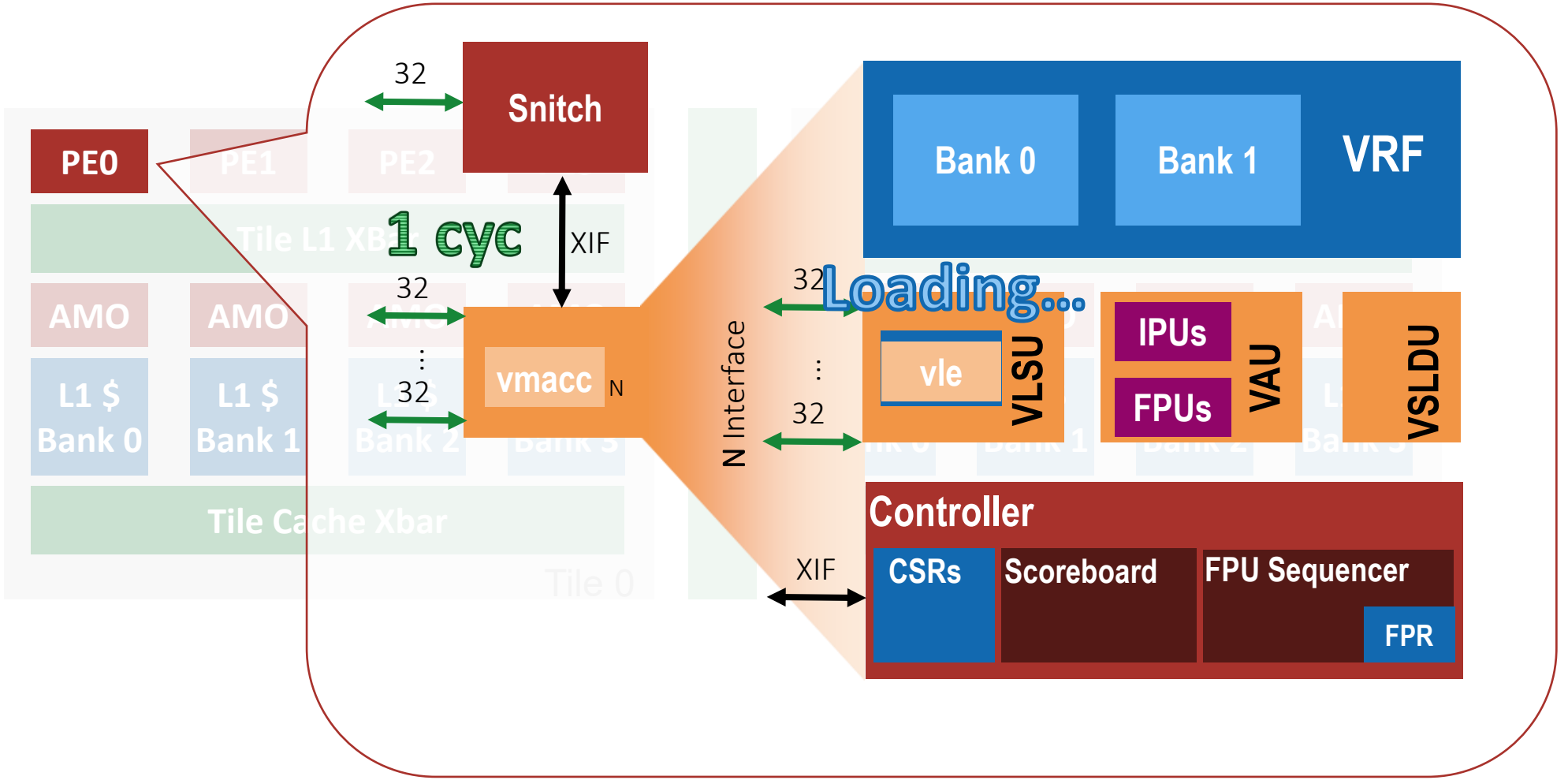
Architecture Introduction: Snitch-Spatz Core Complex



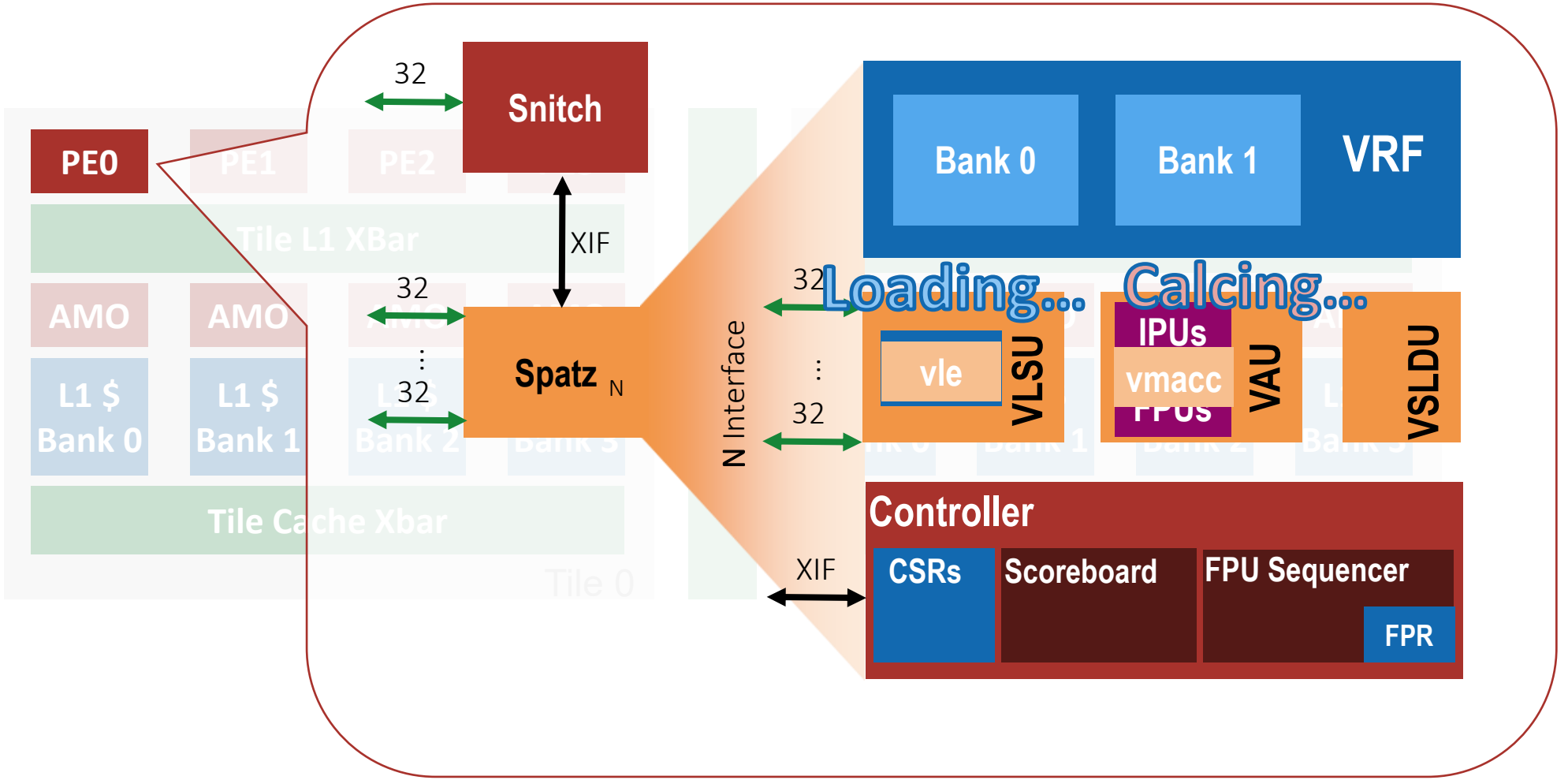
Architecture Introduction: Snitch-Spatz Core Complex



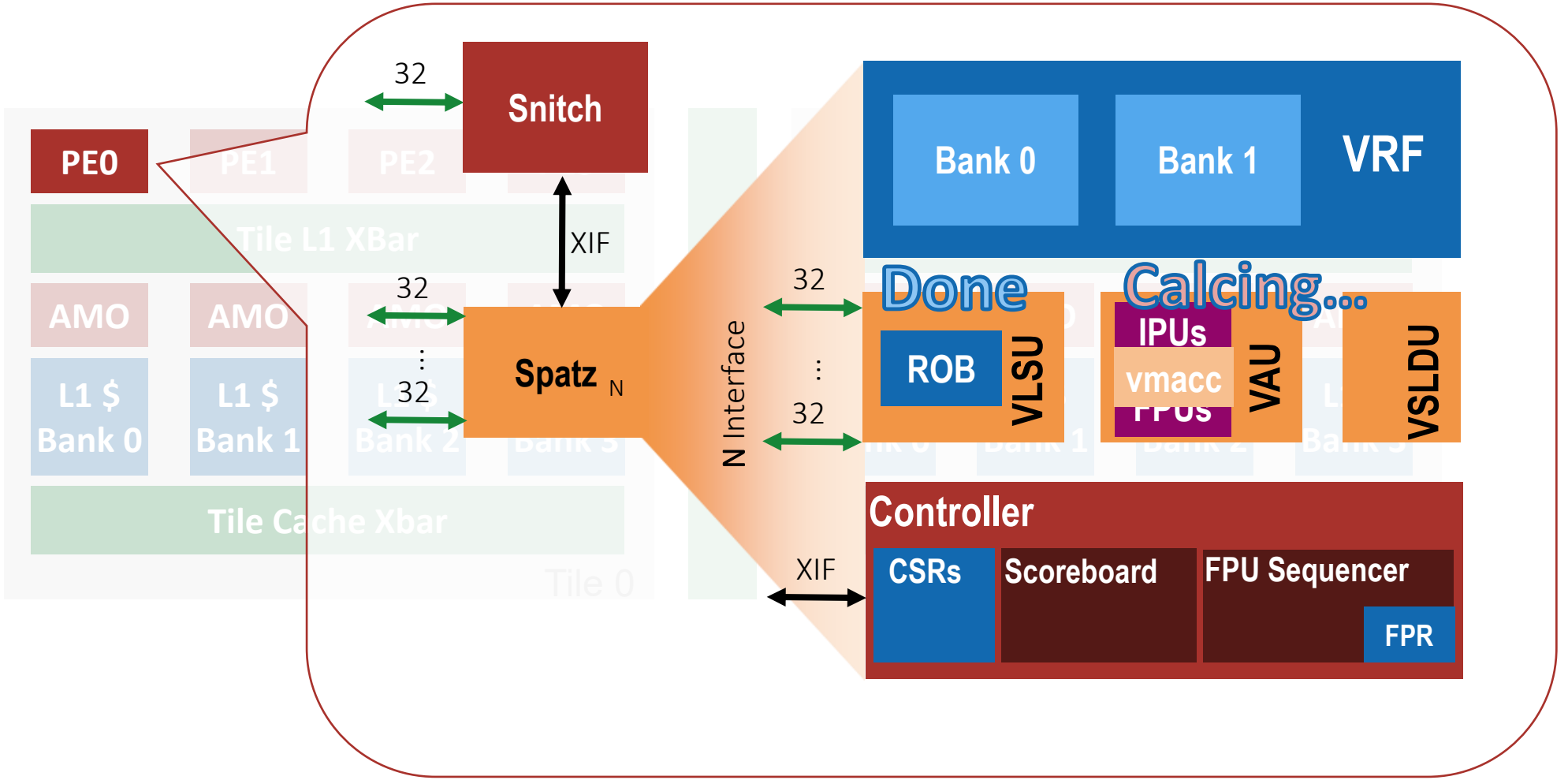
Architecture Introduction: Snitch-Spatz Core Complex



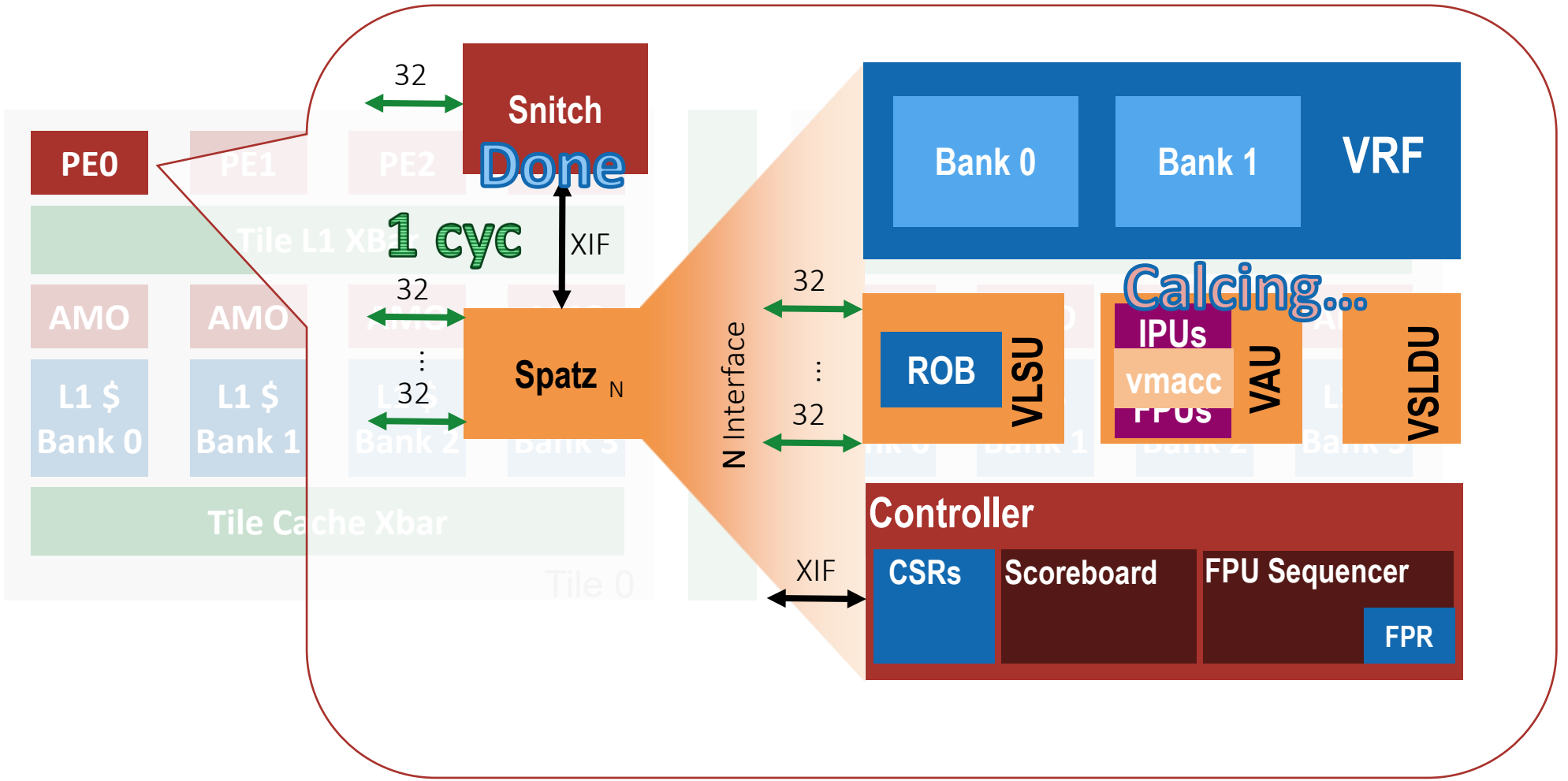
Architecture Introduction: Snitch-Spatz Core Complex



Architecture Introduction: Snitch-Spatz Core Complex



Architecture Introduction: Snitch-Spatz Core Complex



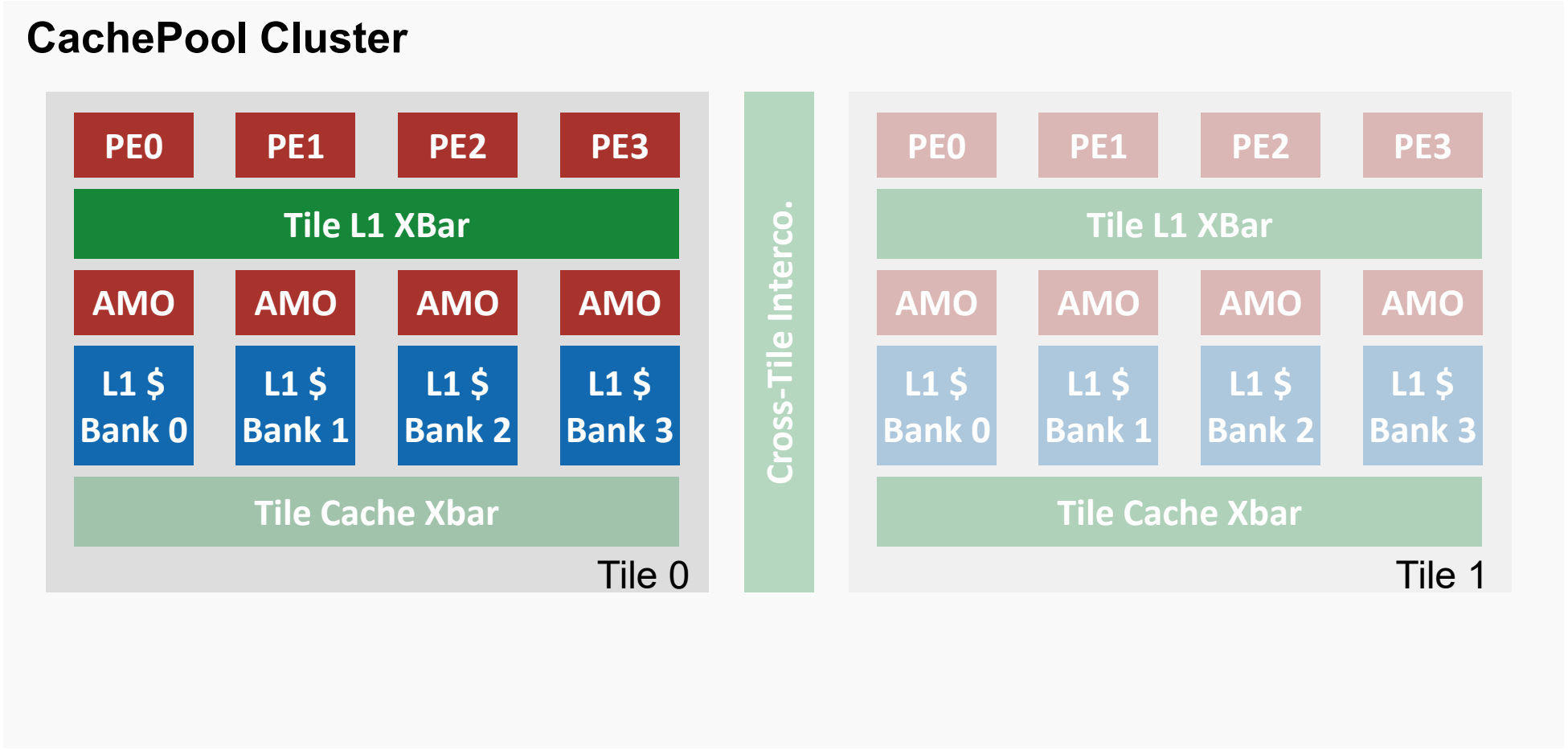
Architecture Introduction: Snitch-Spatz Core Complex

- **Conclusion**
  - Snitch and Spatz are tightly-coupled
  - Snitch can offload vector instructions to Spatz in **one** single cycle
  - Snitch can execute independent instructions in parallel while Spatz is working
  - Spatz can execute up to four instructions in parallel
    - In different units (VAU, VLSU, VSLDU, VController, FPU Sequencer)
    - Dependency handled by scoreboard and vector chaining
  - Minimal or even **no overhead** in Scalar-Vector mixing kernels!

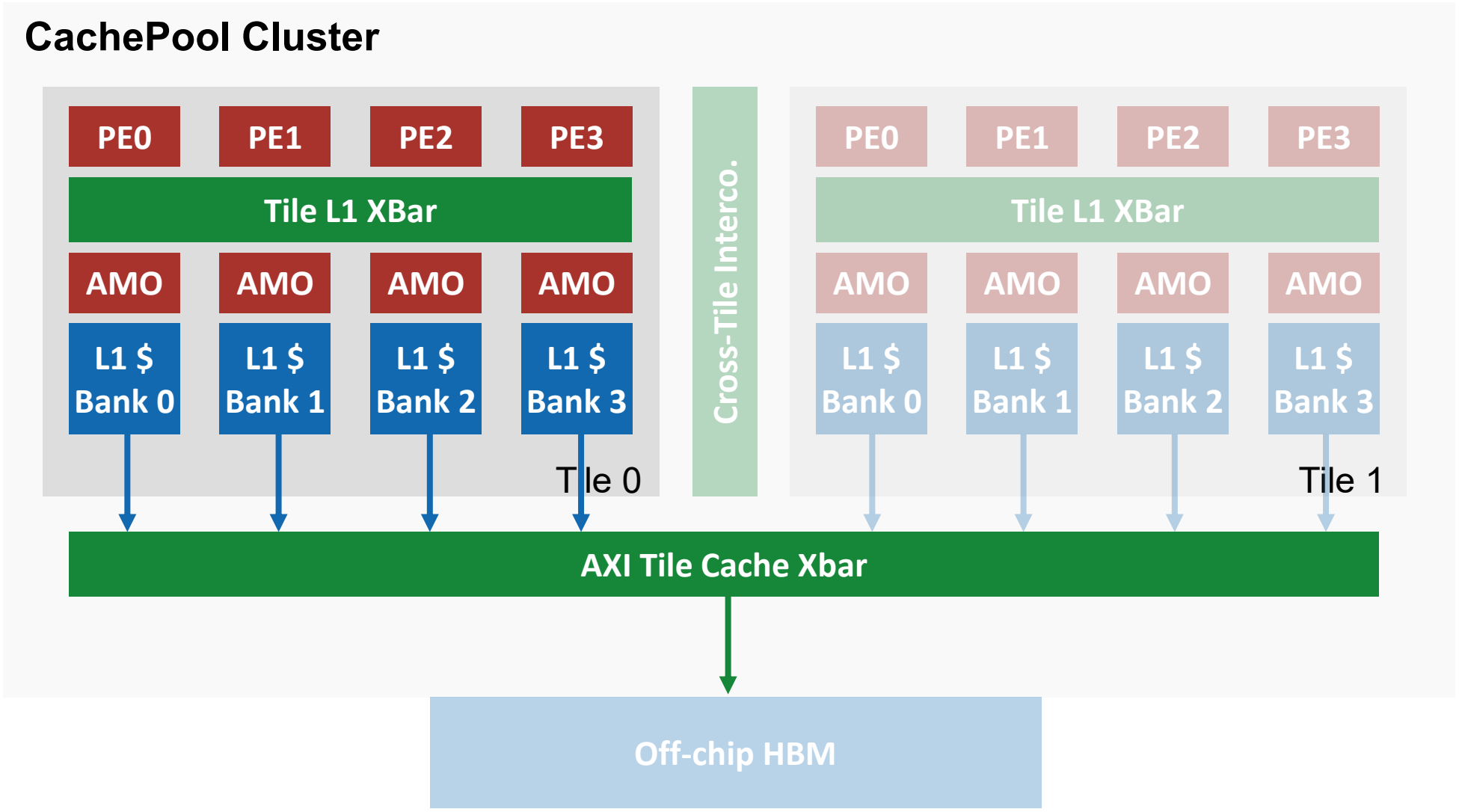
Outline

- Work Package Progress Update
- Architecture Introduction: Snitch-Spatz Core Complex
- **Hardware Development**
- Software Development
- Open Discussion

Hardware Development



Hardware Development

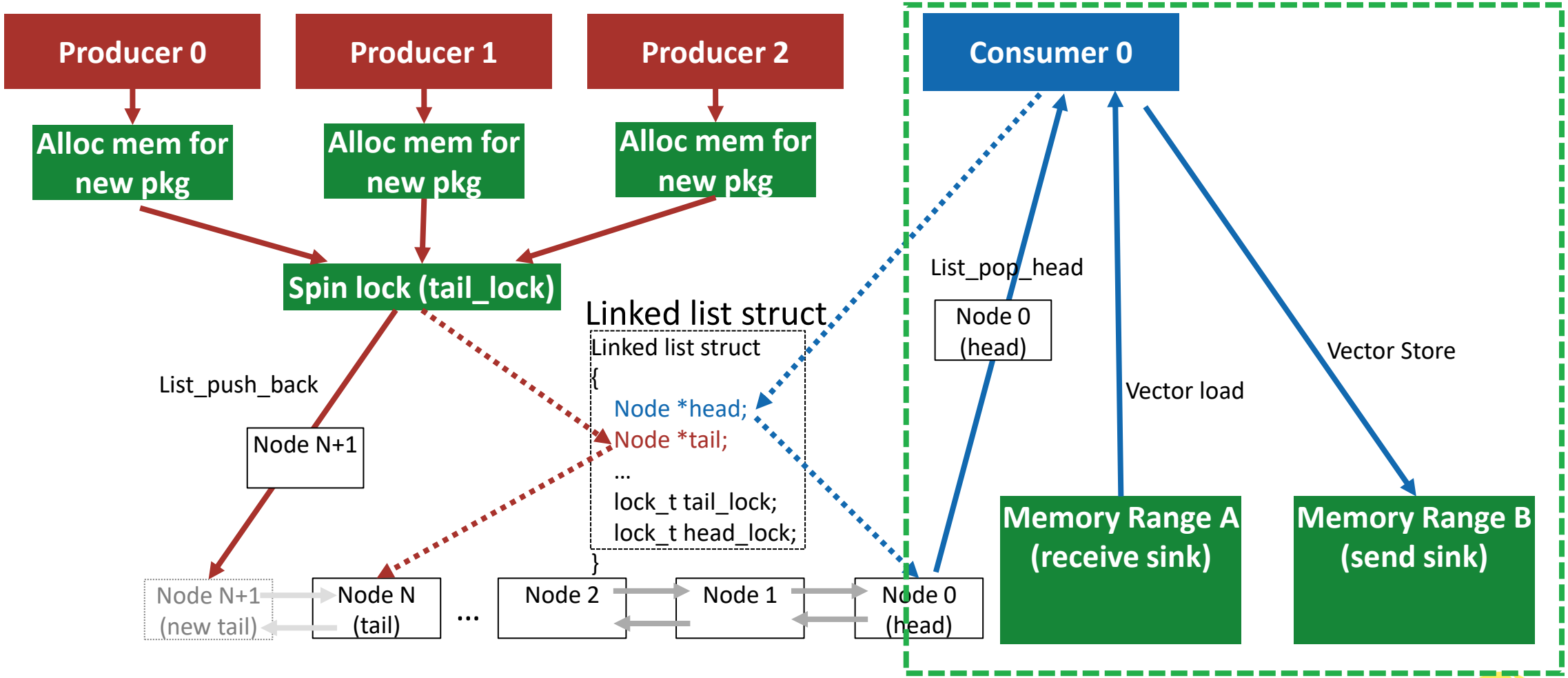


Outline

- Work Package Progress Update
- Architecture Introduction: Snitch-Spatz Core Complex
- Hardware Development
- **Software Development**
- Open Discussion

Software Updates

Use Spatz VLSU to move data, simulate the RLC pkg sending process

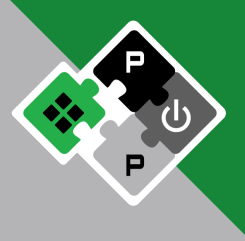


Software Updates

- **These vector load/store ops can put heavy pressure on memory BW**
- **Next step:**
  - Continue the RLC data management kernel
  - Analyze more realistic memory BW performance after memory simulator (Dramsys) get integrated
  - Start working on the RLC control kernels (incl. scalar+vector workloads)
    - Sparse Matrix-Vector Multiplication (SpMV)
    - Maximum Value Sorting
    - Logarithm Calculation Using Taylor Expansion
    - Sum Reduction

Outline

- Work Package Progress Update
- Architecture Introduction: Snitch-Spatz Core Complex
- Hardware Development
- Software Development
- **Open Discussion**



Thank you!