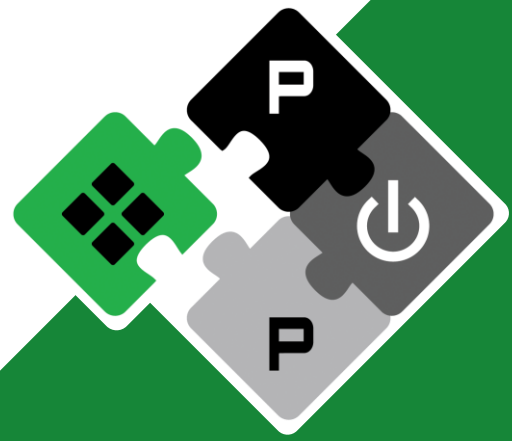
ETH zürich ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

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



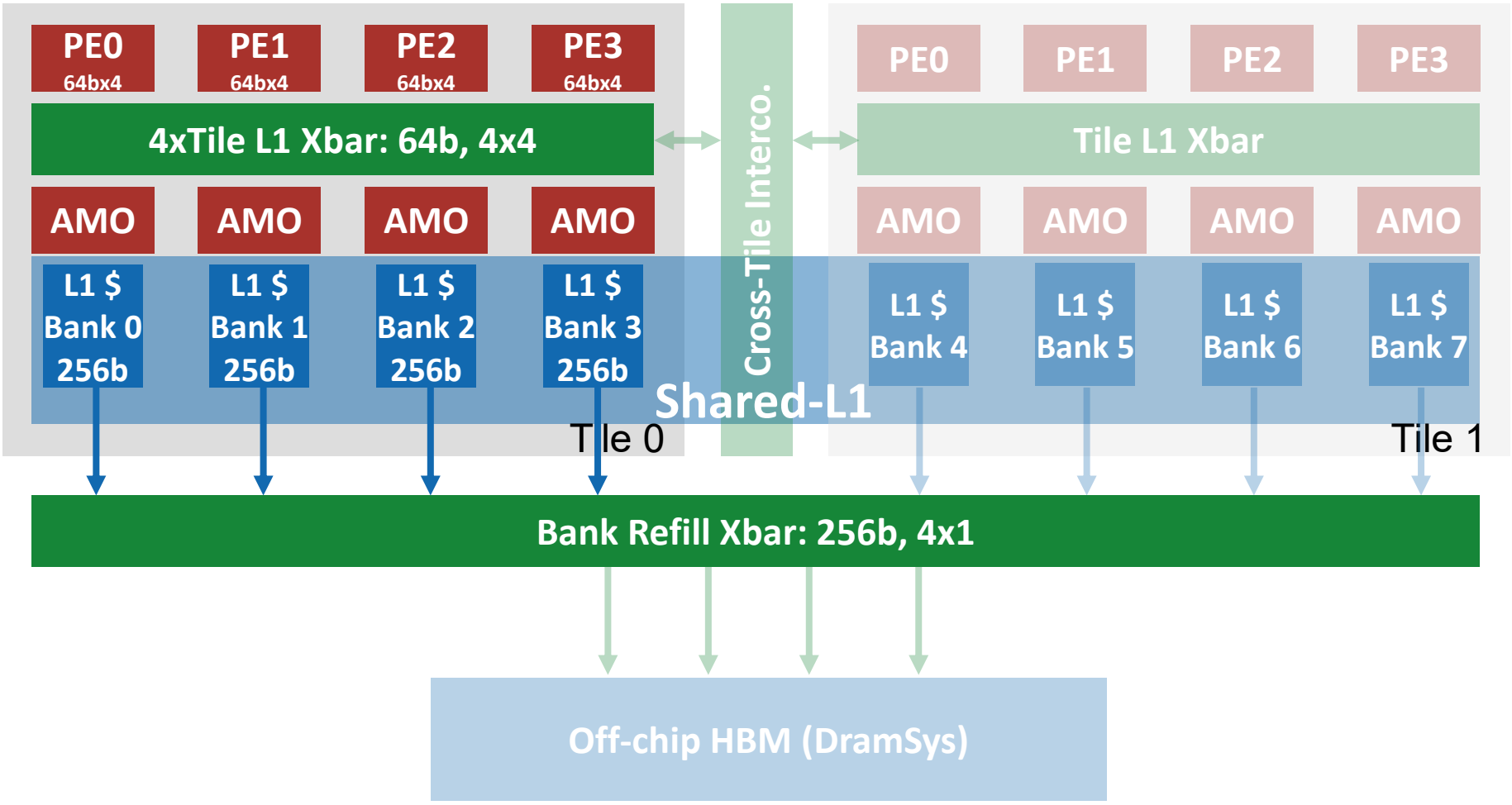
@pulp_platform

pulp-platform.org

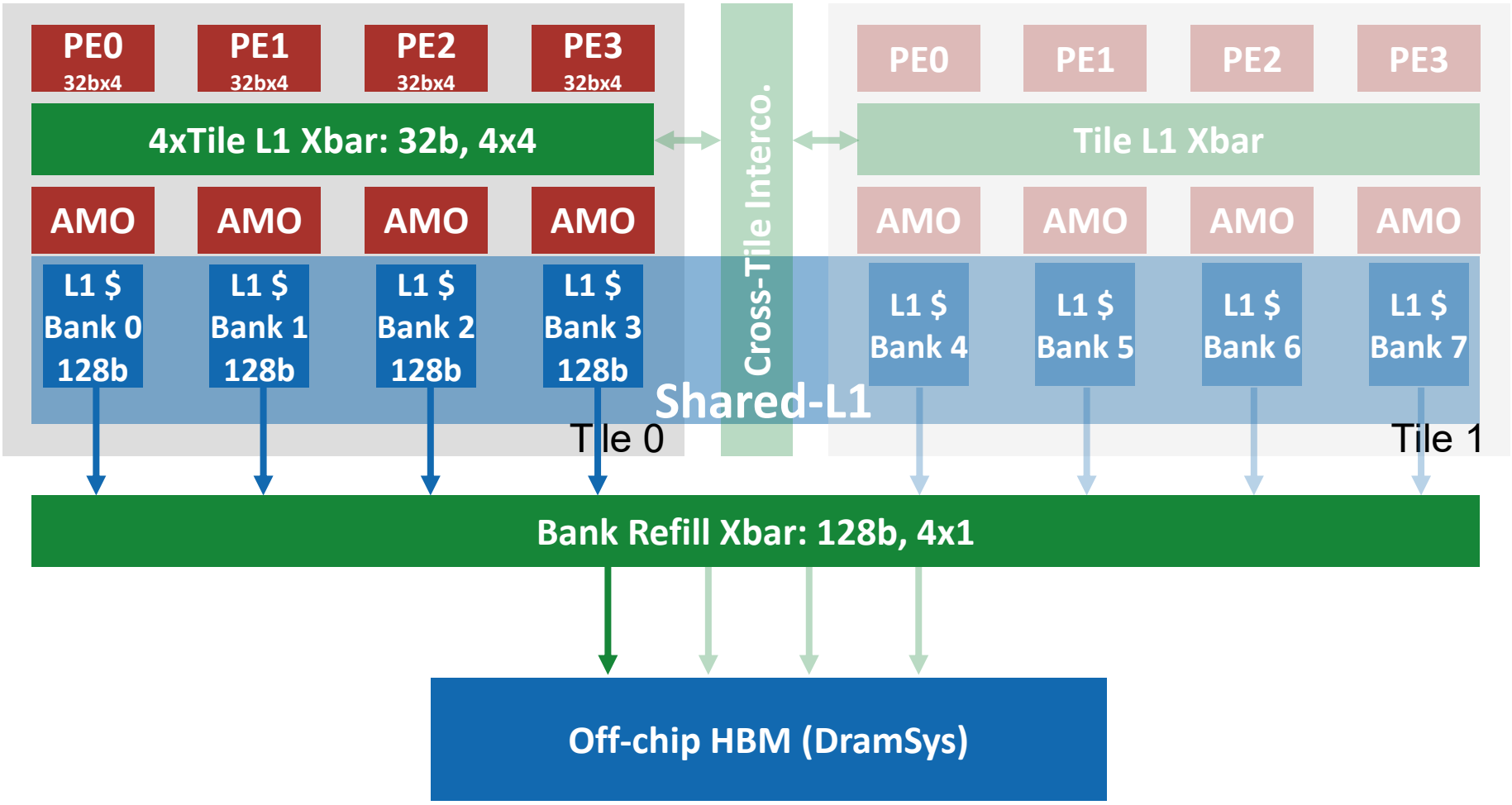
youtube.com/pulp_platform

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

Hardware Development



Hardware Development



Hardware Development

- **Complete**
  - Integration of DramSys Simulator (80%)
  - Add a 32b configuration for cores
    - This change reduces the cacheline from 256b to 128b
    - Bank refillXbar scalability consideration
- **In Progress**
  - Configurations of DramSys HBM channels adjustment on Cache-Refill Xbar
  - Adjust UART modules (for printing function) for DramSys
- **TODO**
  - PPA analysis: plan to start very soon (in 1-2 weeks)
  - Performance evaluation
  - Partition support

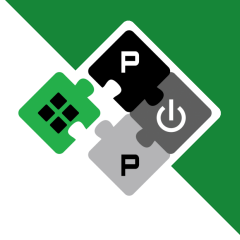
Software Development

- **GEMV kernel**
- **Spatz**
  - VLSU: vle64, vse64
  - VFPU: vfmul, vfmacc, vfadd
- **Snitch**
  - Pointer increment
  - Control flow
- **Fine-grained scalar-vector interleaving**

```
for (int col=0; col < N; col+=2) {
  // Load chunk a
  asm volatile("vle64.v v0, (%0)" ::"r"(a_));
  a_ += M;

  // Multiply and accumulate
  if (col == 0) {
    asm volatile("vfmul.vf v4, v0, %0" ::"f"(*b_));
  } else {
    asm volatile("vfmacc.vf v4, %0, v0" ::"f"(*b_));
  }
  b_++;
  …
}
asm volatile("vfadd.vv v4, v4, v12");
asm volatile("vse64.v v4, (%0)" ::"r"(c_));
avl -= vl;
c_ += vl;
b_ = b;
```

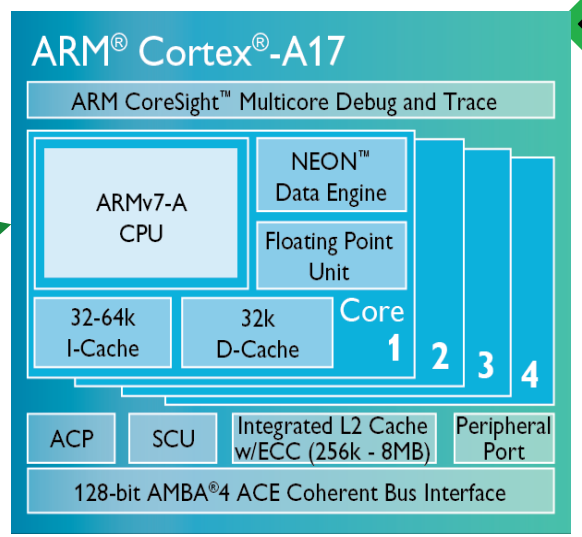
Software Development

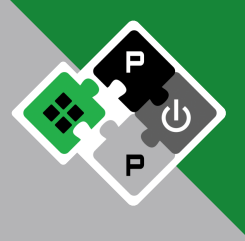


- **Complete**
  - GEMV kernel functional test
- **In Progress**
  - GEMM kernel
  - GEMV kernel performance calibration: HW is not stable for now
  - Continue the RLC data management kernel
- **TODO**
  - Performance evaluation after dramsys integration is fully completed
  - Other kernels for the control algorithm

Evaluation Methodologies

- **RLC**
  - Main rule of thumb: TTI and latency requirements
  - SoA: compare with the existing HW platform (Is this still the latest?)
  - Cross-comparison: compare between architecture candidates
- **Control (Vector-Scalar Mixed Kernels)**
  - Scalar-Vector comparison: compare with the implementation using purely scalar cores
    - Need to match the same maximum throughput (same peak performance)
  - Vector FPU utilization
- **Architecture**
  - PPA: maximum frequency, power consumption, area, physical feasibility





Thank you!