ETH zürich

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

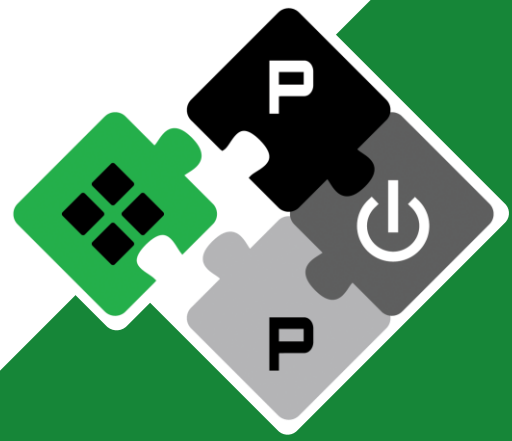
# CachePool: Many-core cluster of customizable, lightweight scalar-vector PEs for irregular L2 data-plane workloads


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**

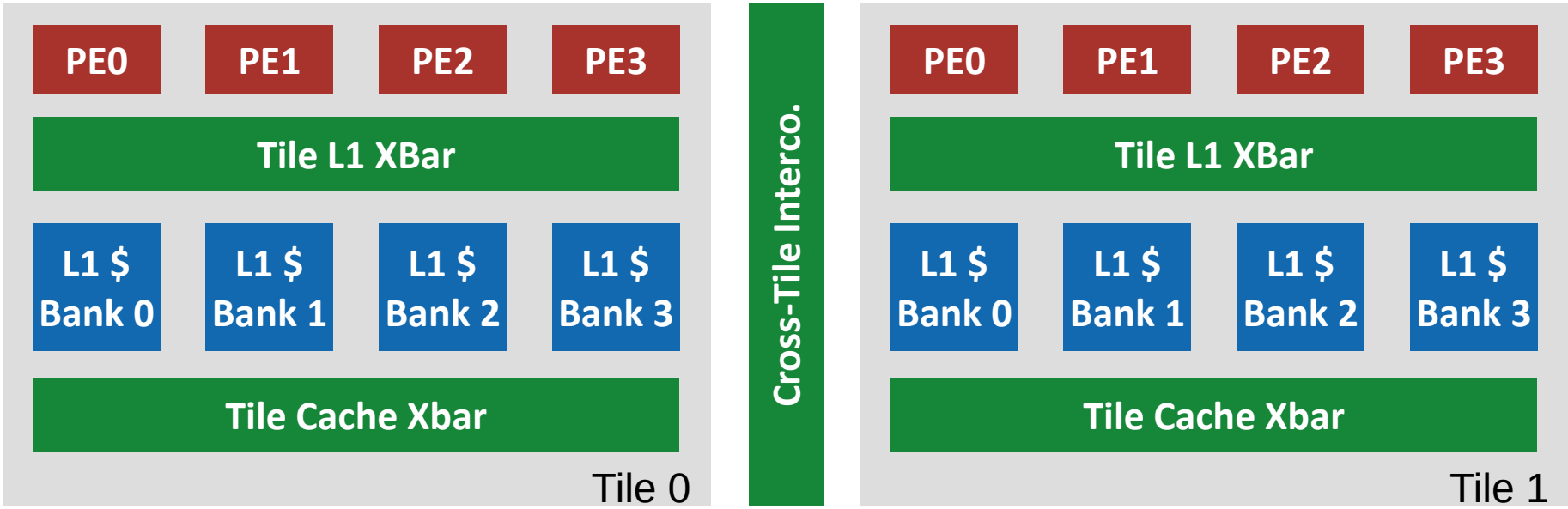
Open Source Hardware, the way it should be!

# Status Update: Hardware

- **Add Atomic support**
- **Push RTL to github**
  - Branch: dev/cachepool
  - Not everything is ready there (missing: cache controller)
- **TODOs:**
  - Cache refill xbar and dramsys for main memory => performance optimization
  - Separate Cluster level and Tile level => prepare for multi-tile configuration
  - Partitioning

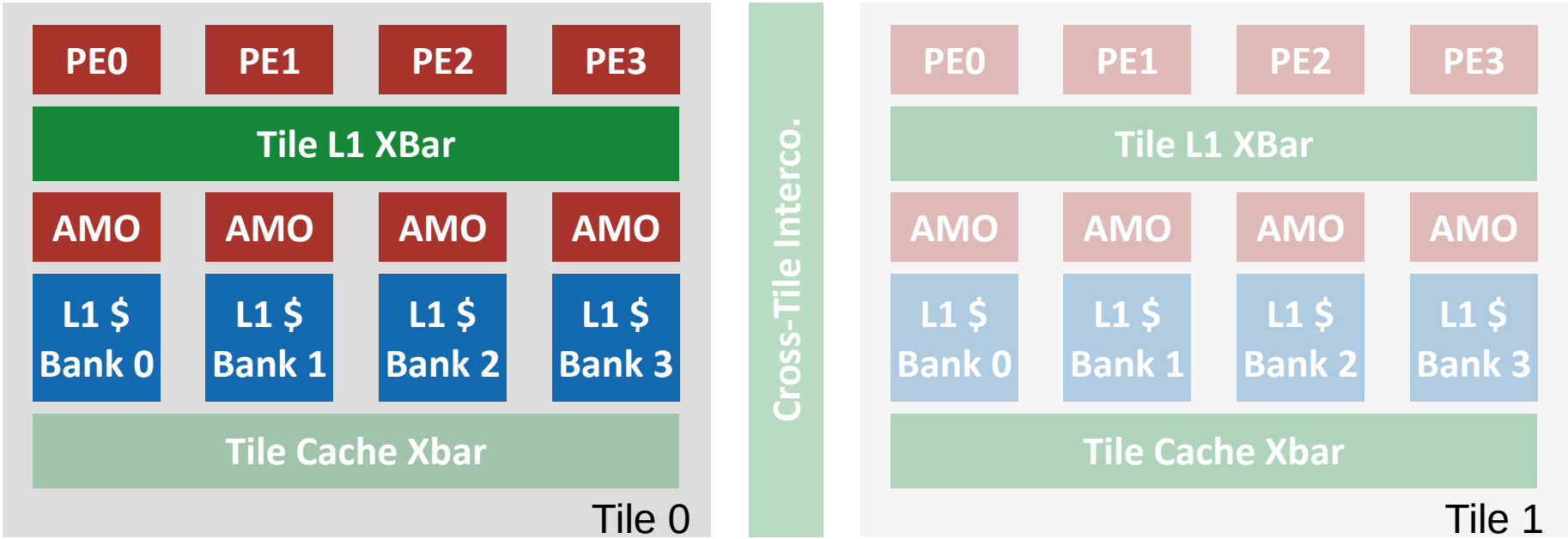
# Status Update: Hardware

- **Finish the implementation of a single Tile**
  - Add atomic support for Cache
    - Test with simple spin-lock



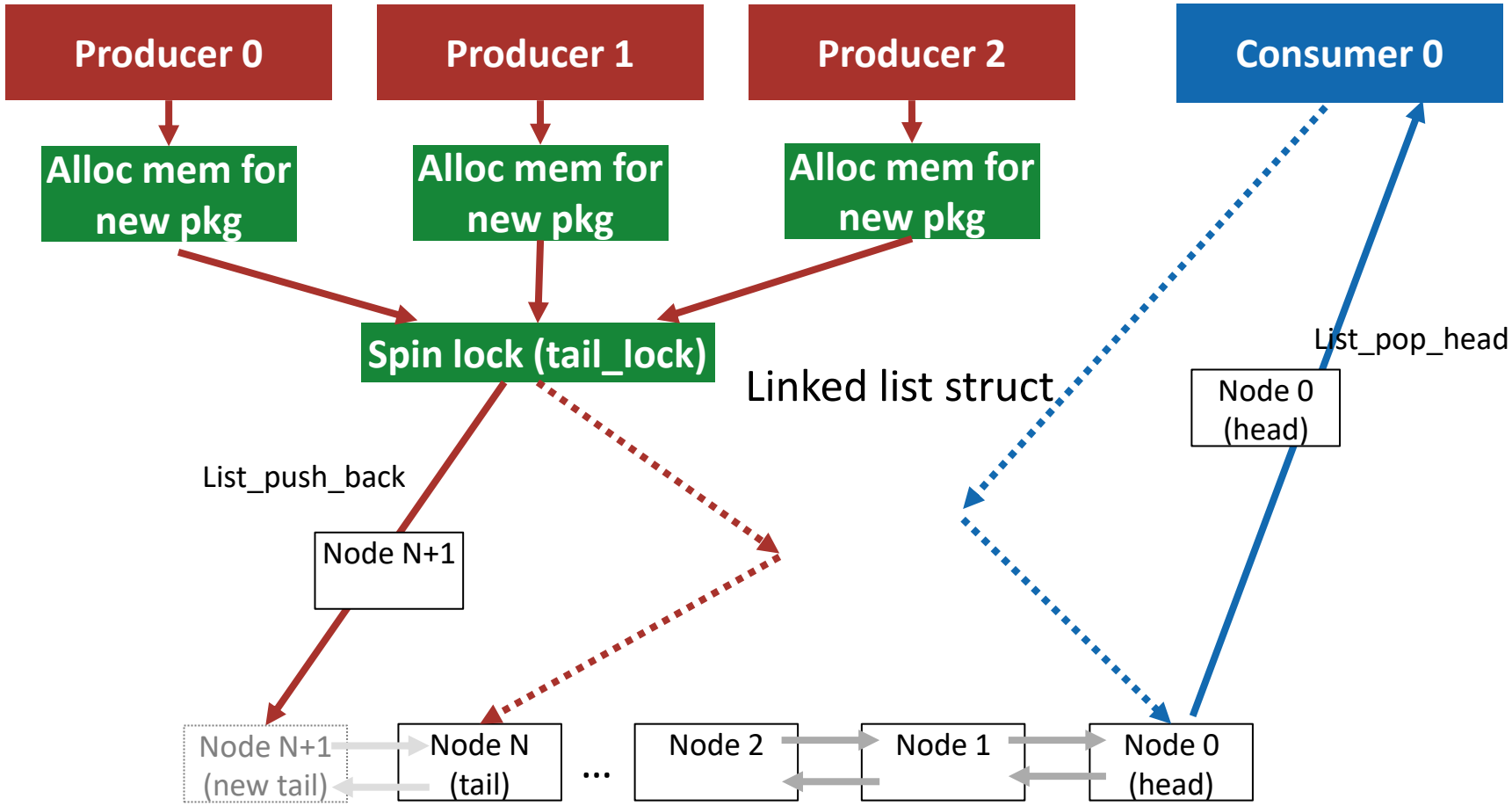
# Status Update: Hardware

- **Finish the implementation of a single Tile**
  - Add atomic support for Cache
    - Test with simple spin-lock



# Software Status 1

## 1. Multiple producers, single consumer double linked-list kernel

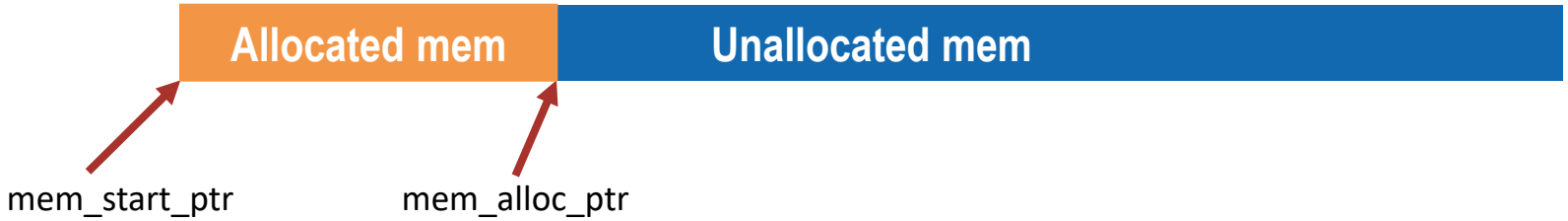


# Software Status 2

## 2. Memory management runtime

- Assuming a fixed memory block size (a page) for each allocation and deallocation

1) Initial allocation: allocate new page from the sequential memory space



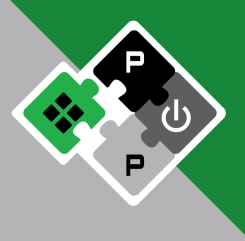

2) Free stage: push ptr of the deallocated page into a free mem linked list

3) Common allocation: pop the head node from the mem linked list, get the ptr to the new page

# Software Status

- **Progress of work**

  1. Multiple producers, single consumer double linked-list kernel
     - Multiple producers, single consumer read/write shared linked list
     - TODO:
       - Debug the existing functions
       - The simulation of the data movement of the consumer
       - Try to use multiple cores for data movement
  2. Memory management runtime
     - Dynamic memory allocation, deallocation, and reclamation at fine granularity (per package)



# Thank you!