The Persistence Bug: Dead or Alive?

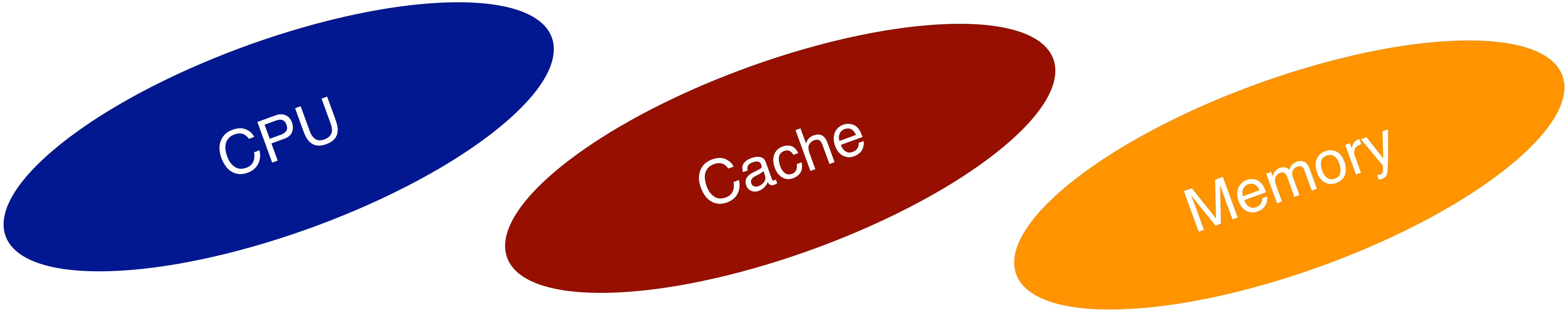
Erez Petrank

Andrew and Erna Viterbi Prof. of Computer Science

TECHNION
Israel Institute of Technology

Persistence

- “Continuing to exist despite interference” Merriam-Webster
- Persistent memory allows continuing to compute after a crash
- If all “data” persists then we can simply continue computation



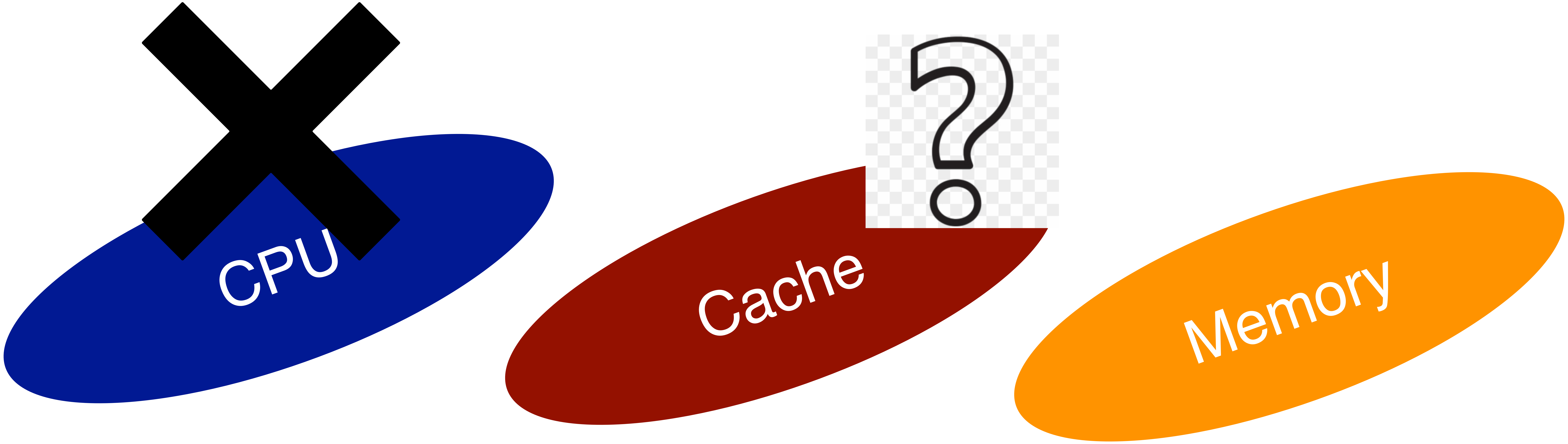
CPU

Cache

Memory

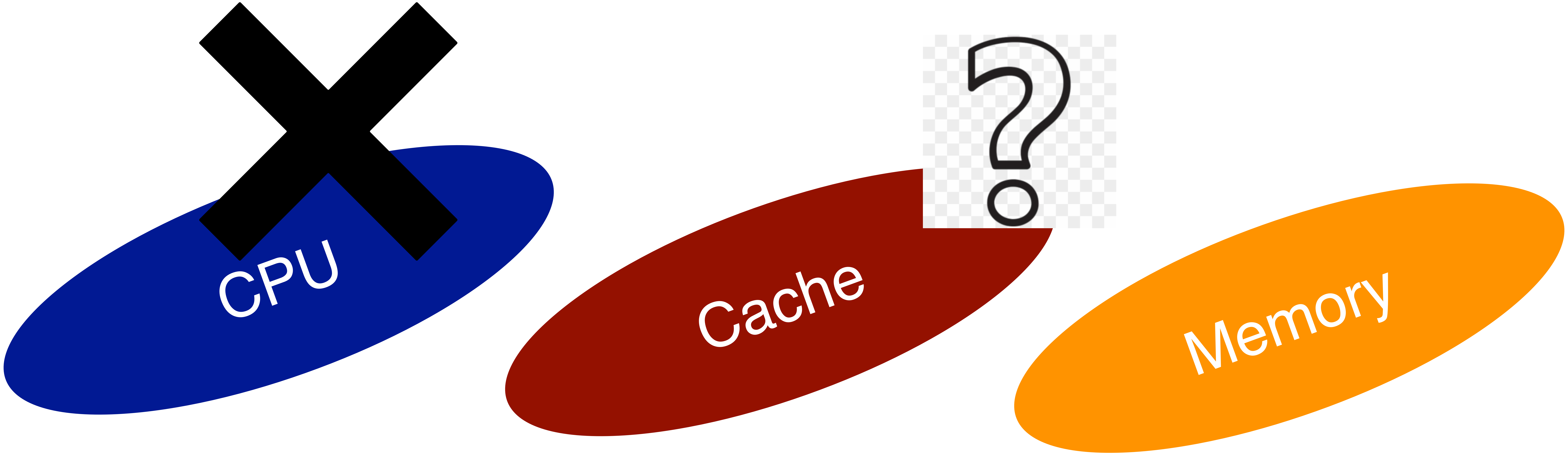
Problem

- Some systems (ADR Optane) only guarantee memory persistence
- Other systems (extended ADR Optane) cover the cache as well (not more)



Programming for eADR?

- “Realize, however, that during the transition from ADR to eADR, there will be servers with only ADR and servers that will have both. **It is then the application’s responsibility** to detect the platform’s capabilities where it is running and **implement the logic** that avoids flushing only when eADR is present. If you use any of the libraries from the Persistent Memory Development Kit (PMDK), all this is already done for you.”



https://www.intel.com/content/www/us/en/developer/articles/technical/eadr-new-opportunities-for-persistent-memory-applications.html

Why Is It Hard

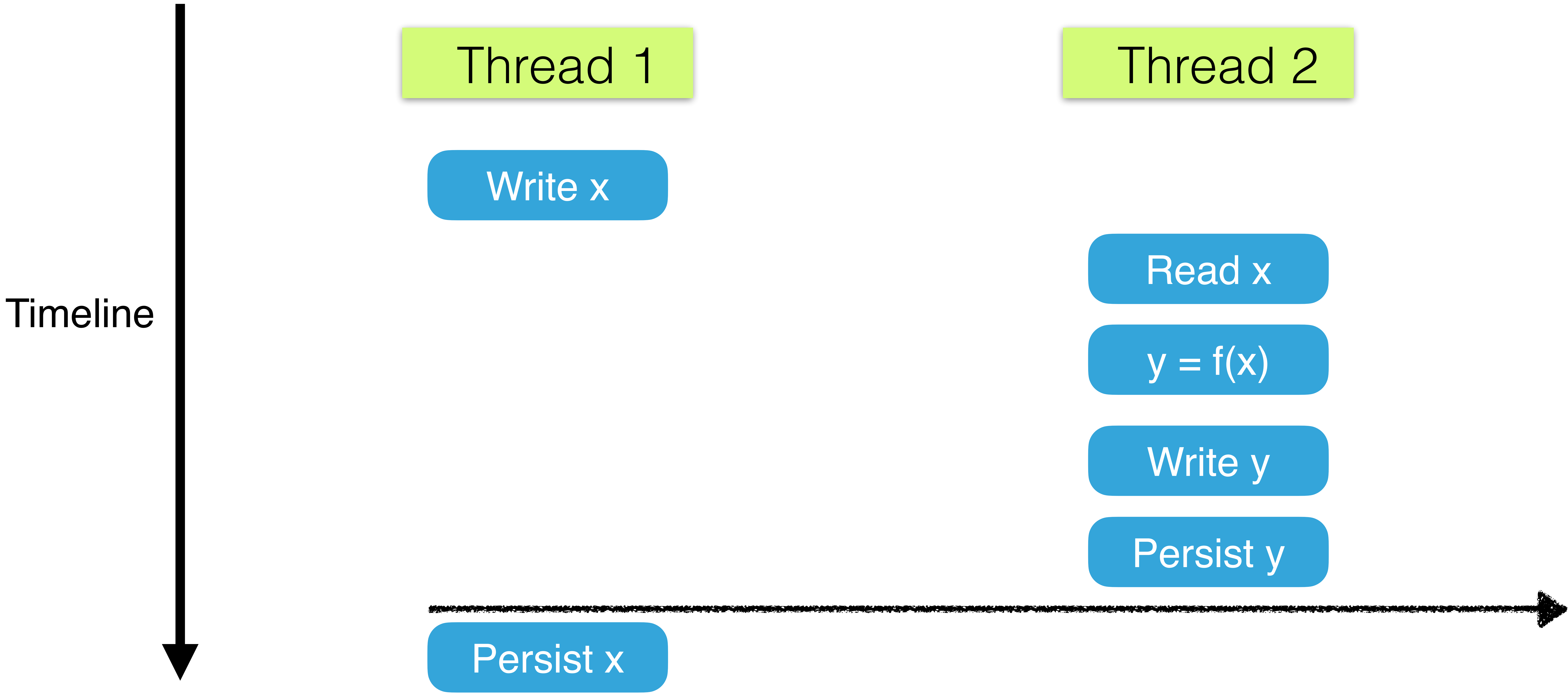
- Without cache persistence, need to persist “important” data.
- Persist: CLWB (flush), SFENCE (wait for it)
- Main issue: persisting is costly so use sparingly
- Is it enough to flush after each write?

Not so bad!

Not so correct!

Persist Every Write

Main issue: the persistence bug



Timeline

Thread 1

Write x

Persist x

Thread 2

Read x

y = f(x)

Write y

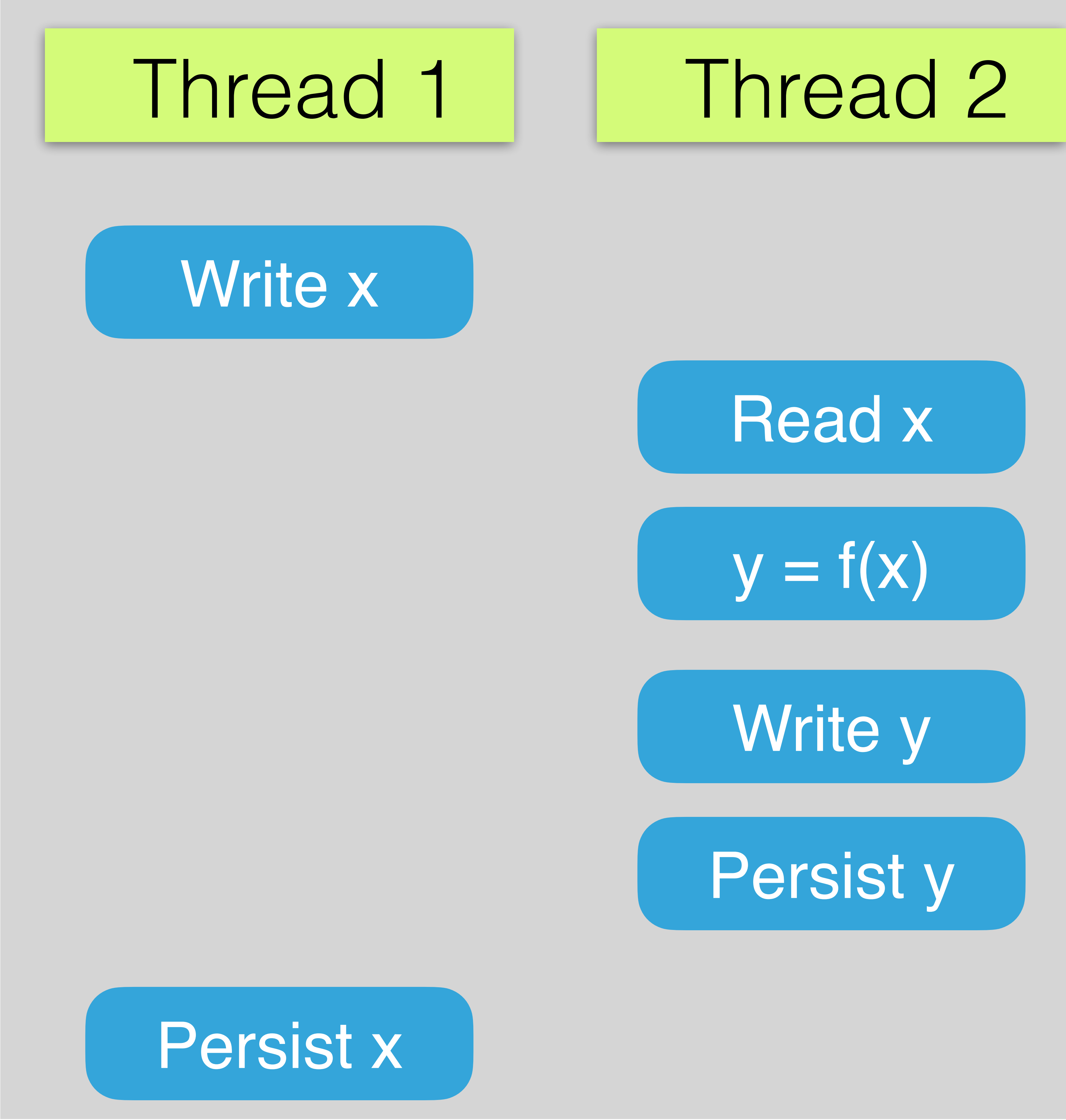
Persist y

The Persistence Bug

- Something that depends on x may persist before x does

- T2's Read of x does not happen before T1's write of x
  and y depends on x
  and x may not persist before T2 writes y

- Defined in [Che et al., ASPLOS 22]
  - PMRace tester

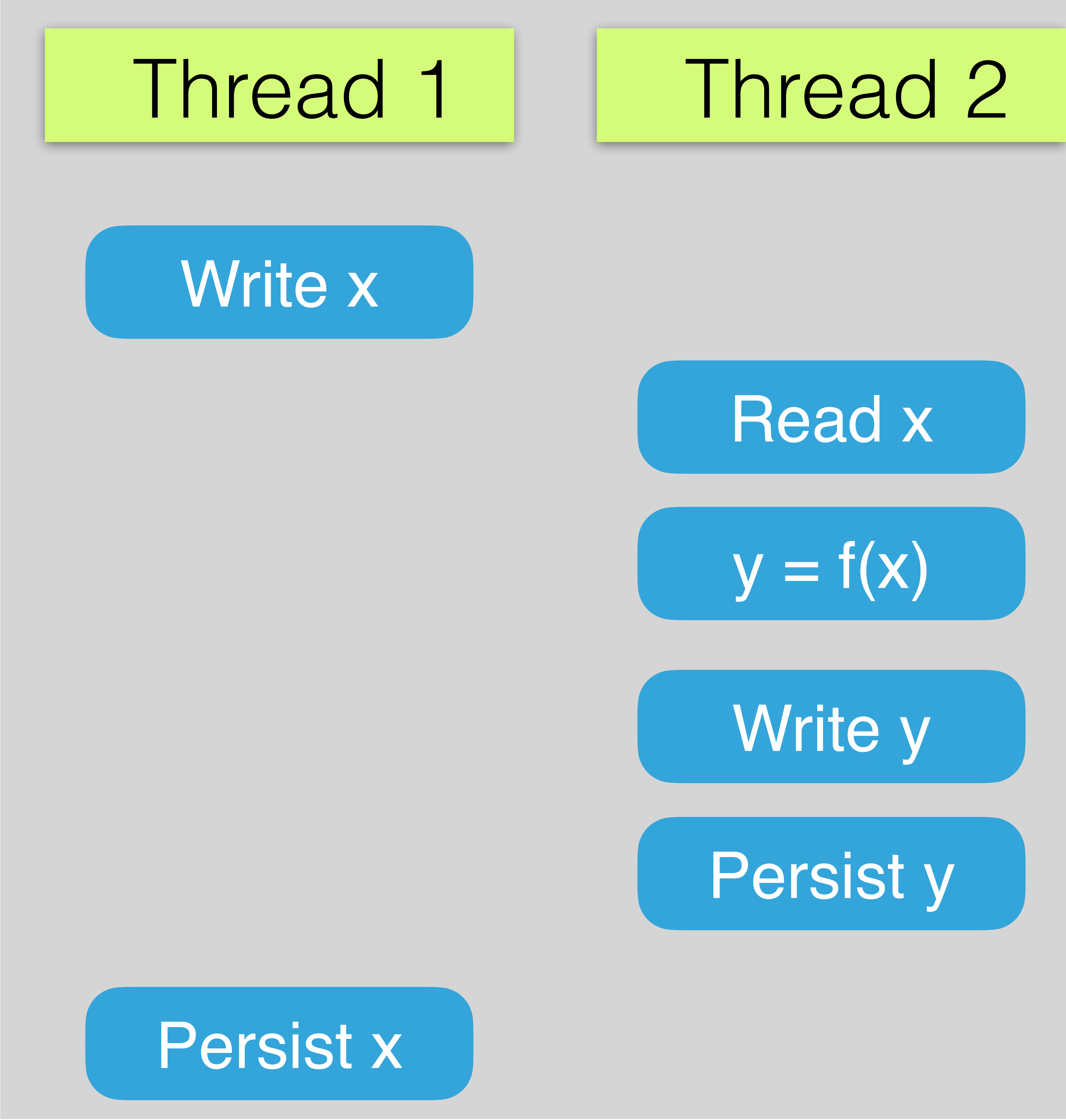


The Persistence Bug

- Something that depends on x may persist before x does

- Solution: persist after reading
  - Correct but expensive

- How do general transformations avoid this bug?
  - NVTraverse [Friedman, Ben David, Wei, Blelloch, Petrank]: carefully identify dangerous reads and persist.
  - Mirror: …



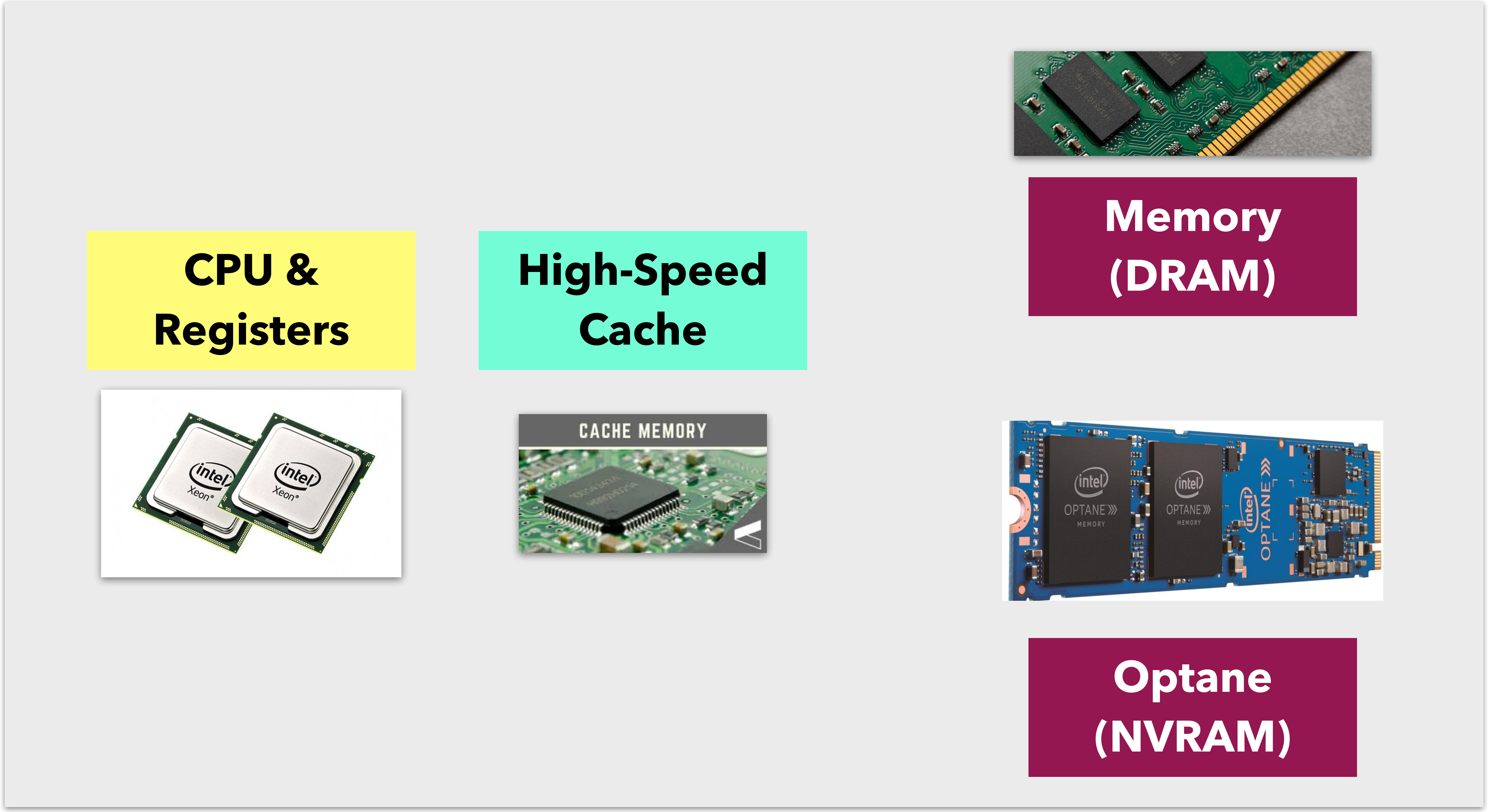
Mirror

Friedman, Ramalhete, Petrank PLDI 21

- Ideas:
    - Always read persisted values
    - Exploit DRAM

Mirror: A transformation for Lock-Freedon

The Optane Architectures



CPU & Registers

High-Speed Cache

Memory (DRAM)

Optane (NVRAM)

Mirror Idea

Note 1: reads are a lot more frequent than writes

Note 2: NVRAM reads are 3x DRAM reads

Main idea: avoid reading from NVRAM

Implementation: keep a replica of data structure in DRAM (mirror)

Read only from DRAM, write to both DRAM and NVRAM

Yields highly efficient durable data structures

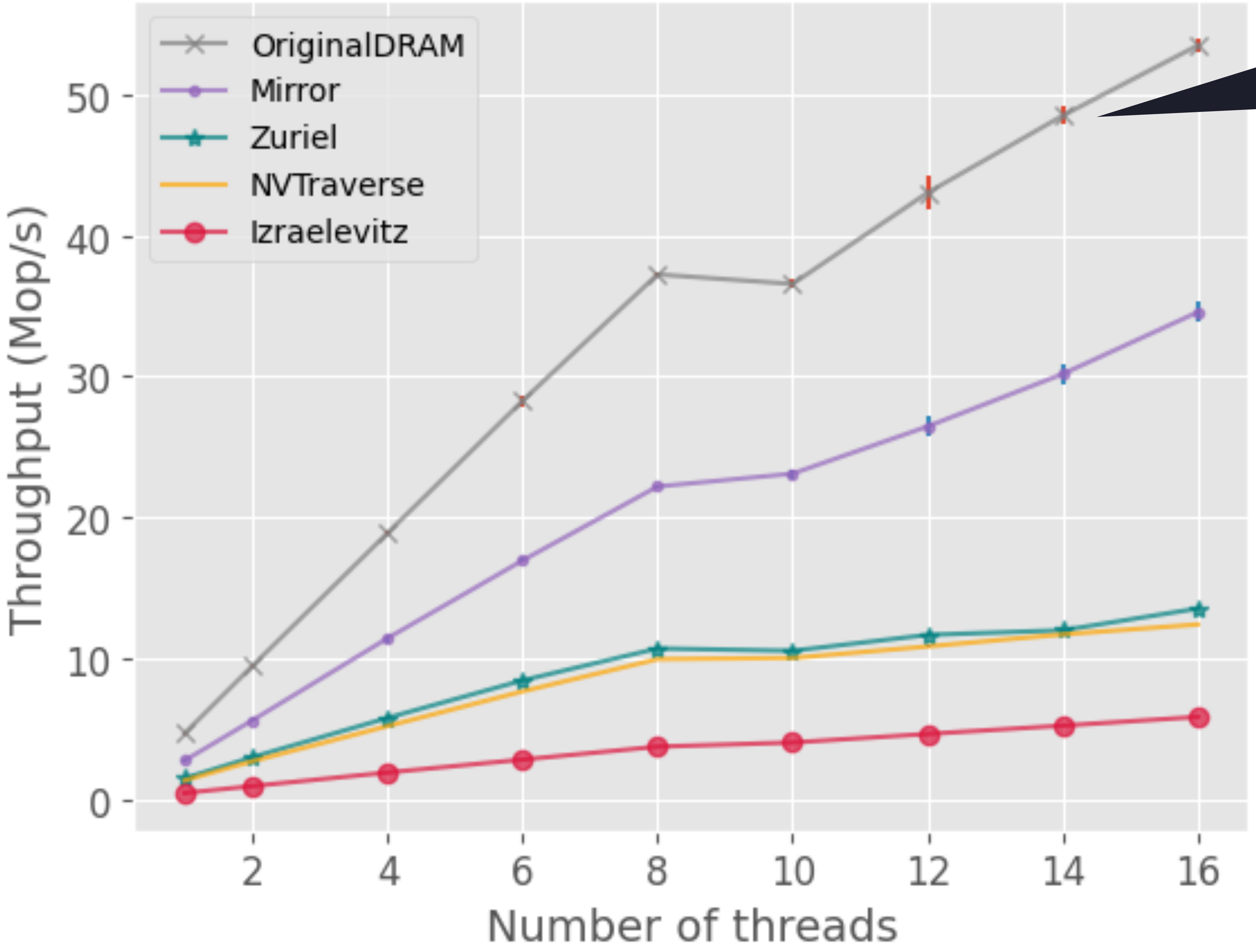
Volatile replica

Persistent replica

Evaluation

- Data Structures: linked-list, hash-table, BST, skip-list
- Competitors:
  - Original version (non-persistent)
  - Izraelevitz construction (general) [IzraelevitzMendesScott '16]
  - NVTraverse (general) [FriedmanBenDavidWeiBlellochPetrank '20]
  - Link-Free & SOFT (hand tuned) [ZurielFriedmanSheffiCohenPetrank '19]
  - pmemkv key-value store (general) [Intel '19]
- Platform - Intel Optane, Intel 6234 3.3GHz, 2 processors with 8 cores, Ubuntu 18.04.1

Hash Throughput

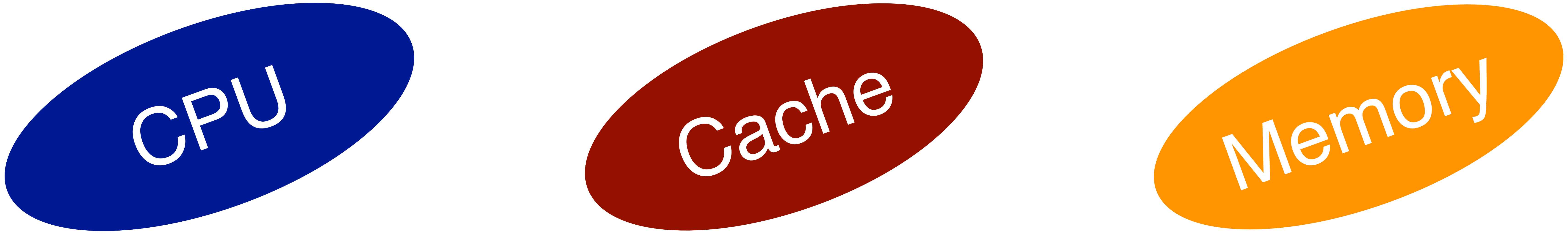


Not Persistent!

- Varying threads
- Initial size: 8M keys
- 20% updates

Mirror with ADR and eADR

- eADR eliminates Mirror’s need to flush (visibility == persistence)
  - But Mirror’s use of DRAM is useful for performance
- ADR requires Mirror’s flushes when writing to NVRAM



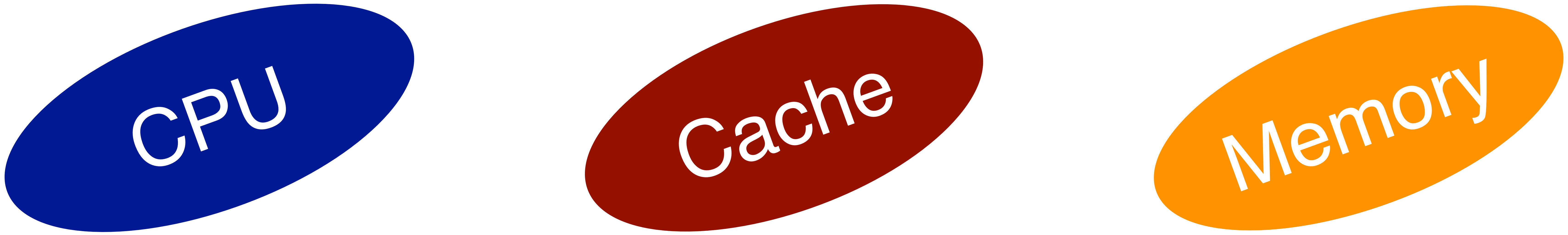
CPU

Cache

Memory

Back to the Persistence Bug

- How will the next commercial NVRAM look like?
- How will CXL deliver eADR?
    - Both memory and processor to use “batteries”
- Why not “eeADR” to persist everything (including CPU)?
    - Hardware support for persisting CPU info
- Other platforms?



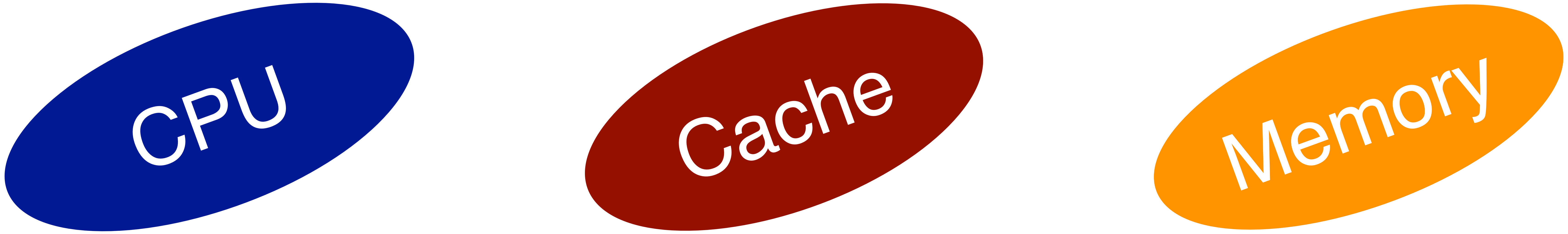
CPU

Cache

Memory

Terminology & Challenges

- ADR, eADR, eeADR?
- Proposal: ‘memory persistence’, ‘cache persistence’, & ‘CPU persistence’
- Research: we have studied memory persistence extensively.
    - Is CPU persistence as easy as it sounds for SW? How difficult is it for HW?
    - What are the good open questions for cache persistence? (Detectability partially studied.)



CPU

Cache

Memory

Conclusion

- The persistence bug
- How two general transformations avoid it
- Research beyond Optane?
- Three futures: memory-, cache-, & CPU-persistence
- Mirror’s use of DRAM is beneficial to all
- Questions: what’s more likely? What should we pursue?

CPU

Cache

Memory