arm

Persistent Atomics for Implementing Durable Lock-Free Data Structures for NVM

William Wang, Stephan Diestelhorst
Arm Research

SPAA'19
24th June 2019

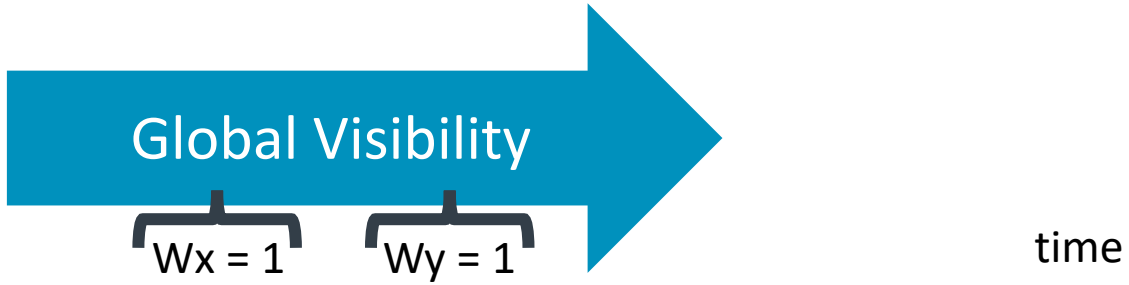
Global Visibility Order

P0
STR W0,[X1]
STR W2,[X3]

Thread 0
a: Wx = 1
po
b: Wy = 1

P0
STR W0,[X1]
DMB.ST
STR W2,[X3]

Thread 0
a: Wx = 1
dmb
b: Wy = 1

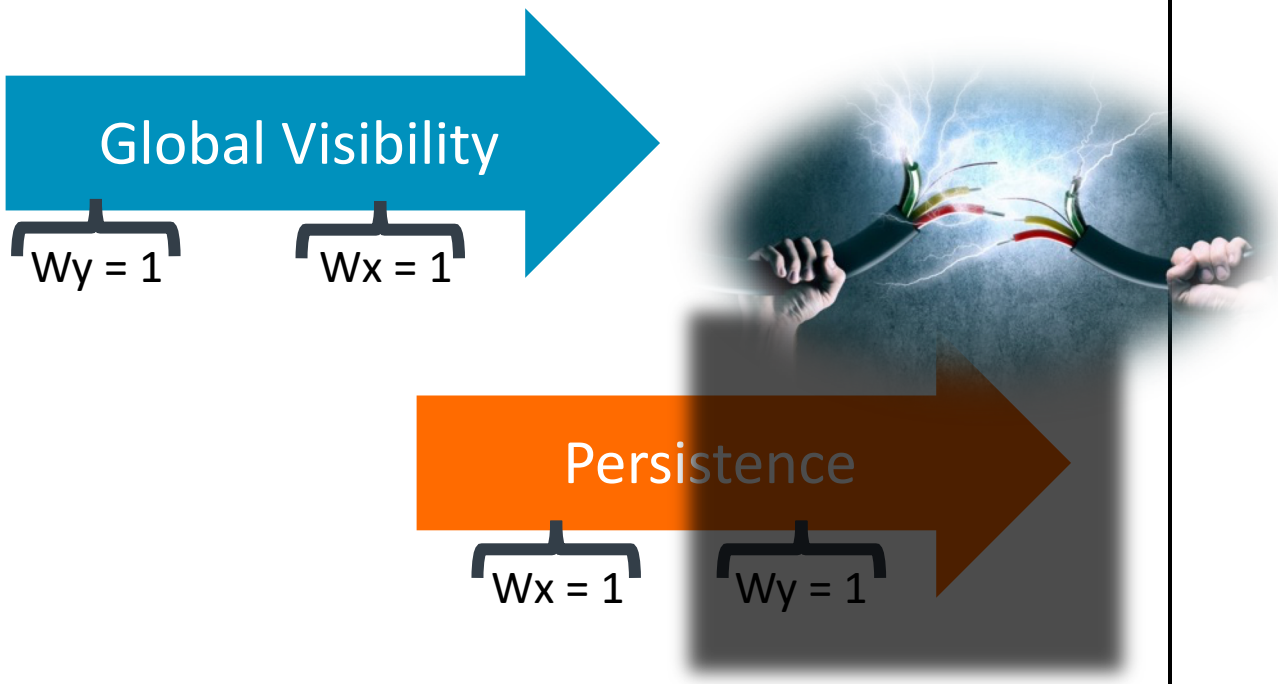


arm

View of the NVM: Persist Order

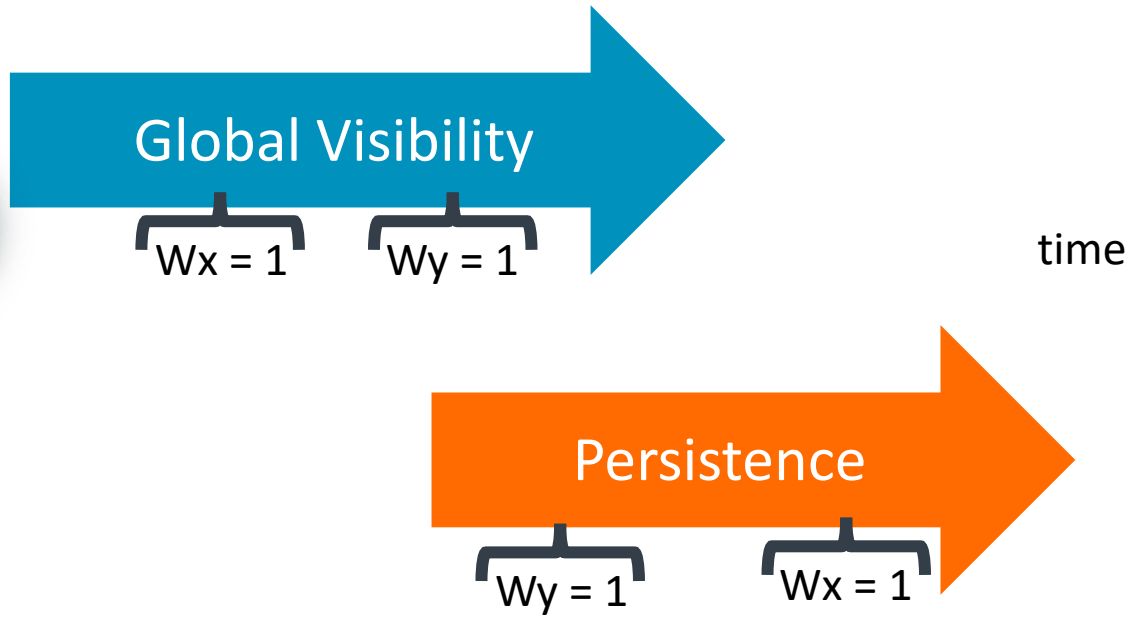
P0
STR W0,[X1]
STR W2,[X3]

Thread 0
a: Wx = 1
po
b: Wy = 1



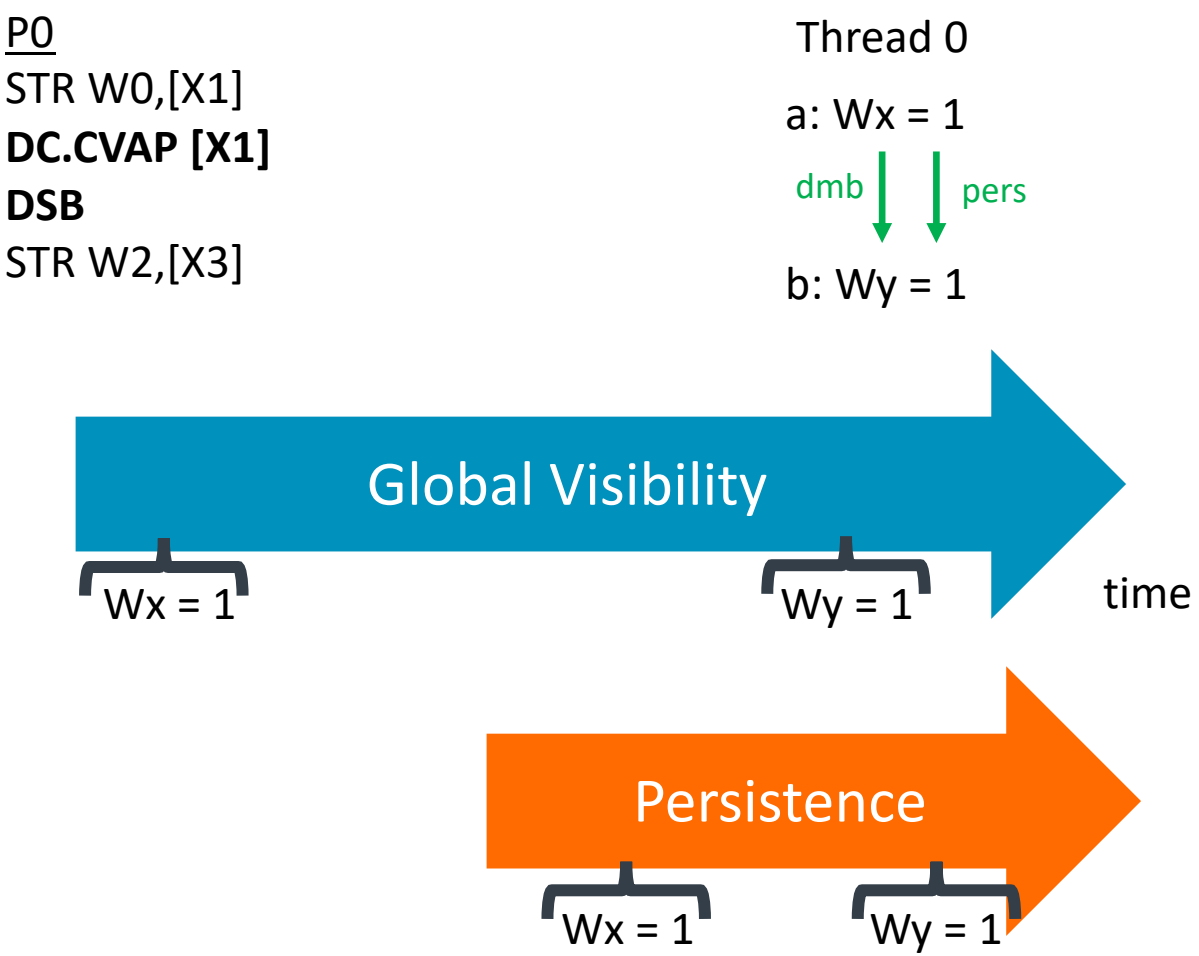
P0
STR W0,[X1]
DMB.ST
STR W2,[X3]

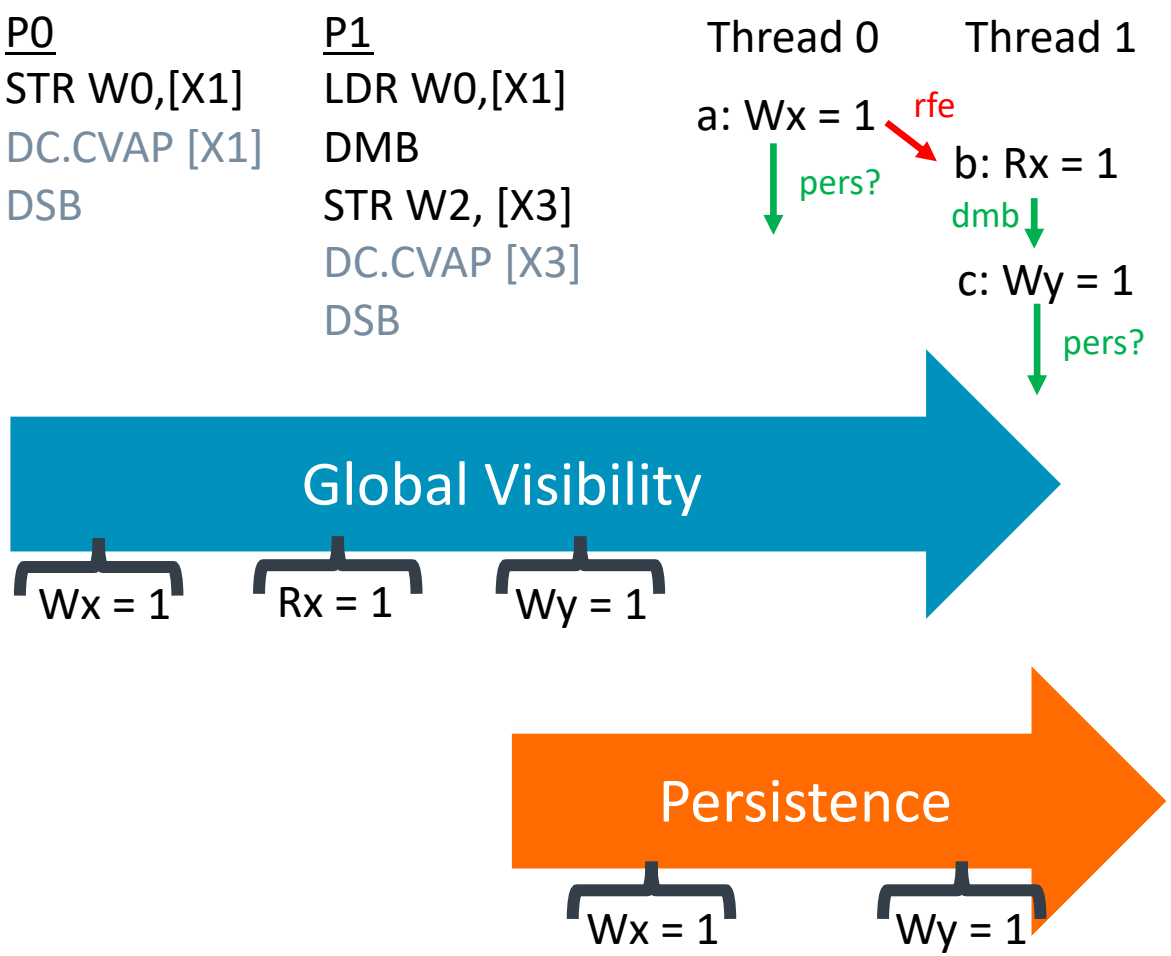
Thread 0
a: Wx = 1
dmb
b: Wy = 1



arm

Enforcing Persist Order

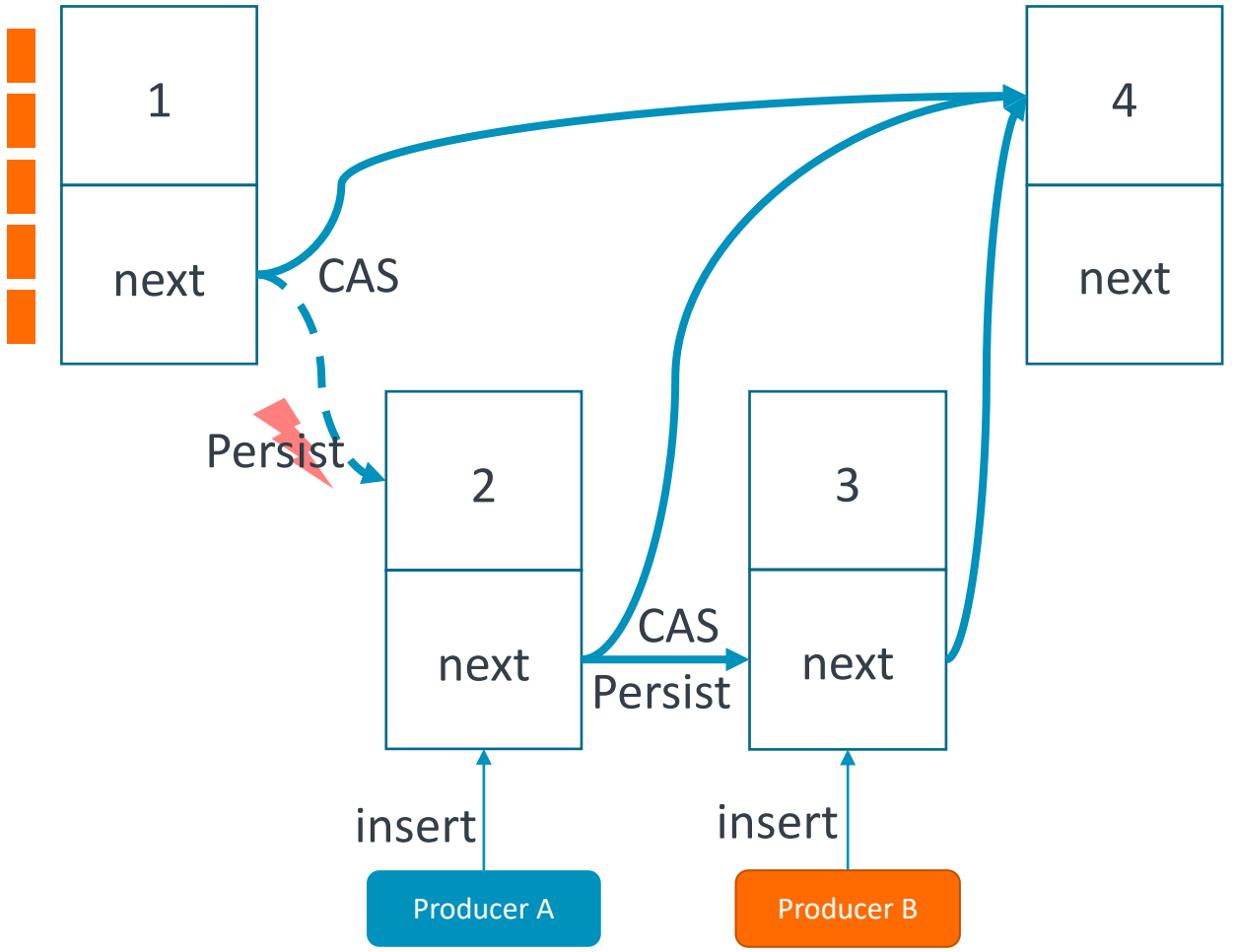




arm

Challenge: Data Loss In Concurrent Linked List

```
1. if(CAS(&last->next, next, node)) {
2.   Persist(&last->next);
3.   DSB
4. }
```



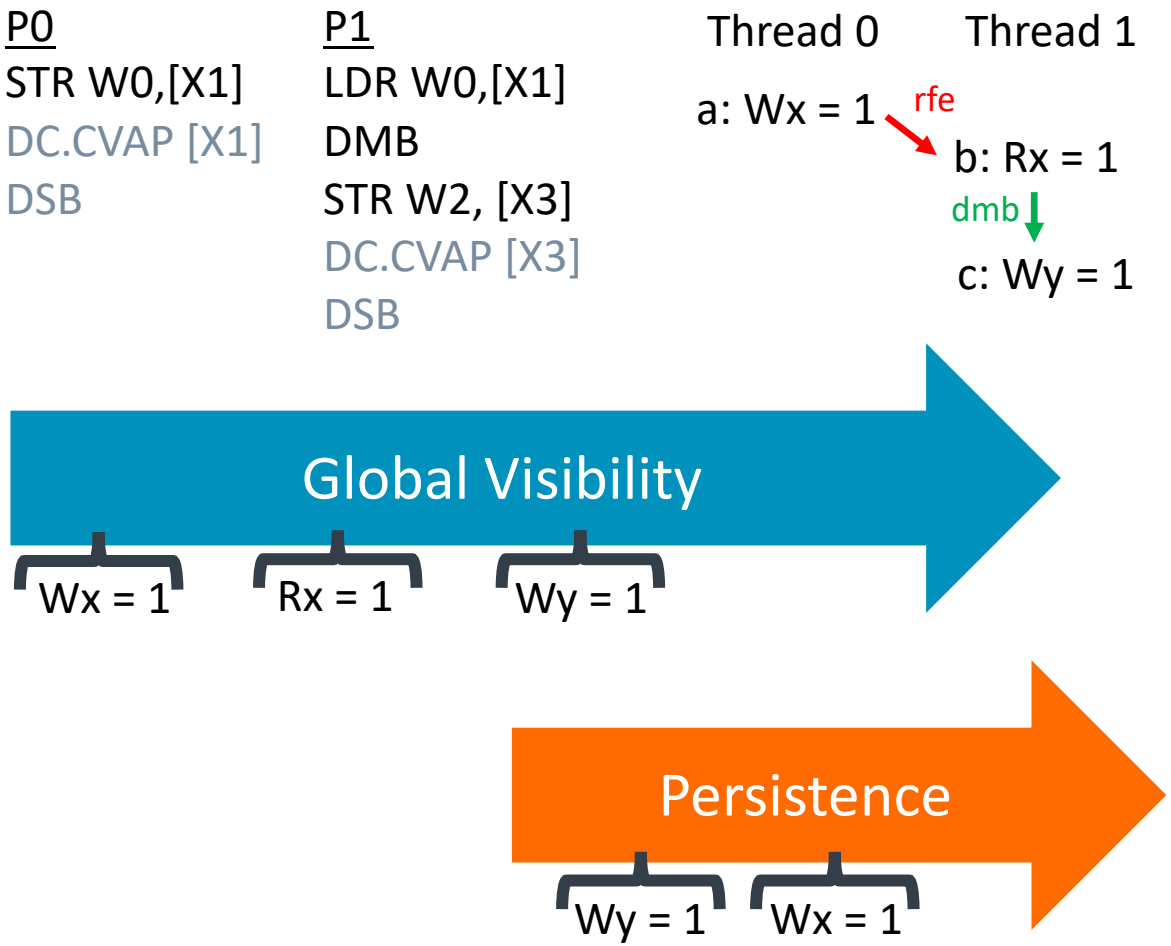
- Producer B observed A's updates, but cannot / does not enforce the persists
- Loss of *transitivity*

Ref: Valois 1995, Lock-Free Linked Lists Using Compare-and-Swap.
For two other problems similar to this one.

arm

Solution

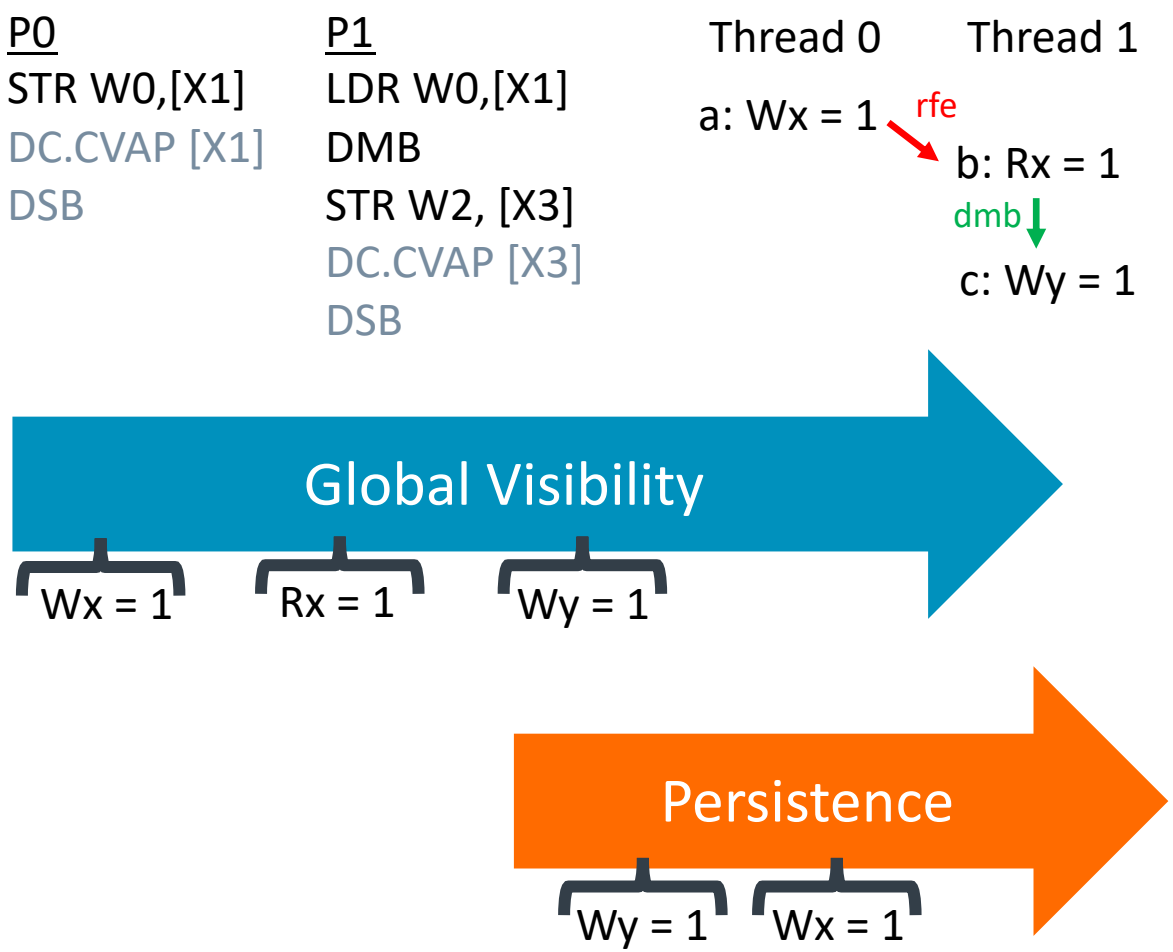
- Basic idea: delay consumer's persist operation until producer's persist operation is done
- Various arch options
  - Delay producers visibility until persistence is done



arm

Solution

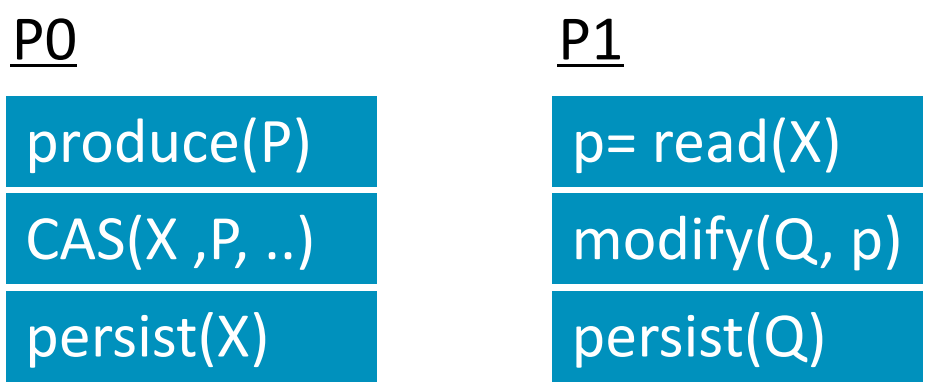
- Basic idea: delay consumer's persist operation until producer's persist operation is done
- Various arch options
  - Delay producers visibility until persistence is done
  - Delay all consumer's persists
  - Delay dependent consumer persists
- New instructions for combining persist and store for critical last publishing store



arm

In Software

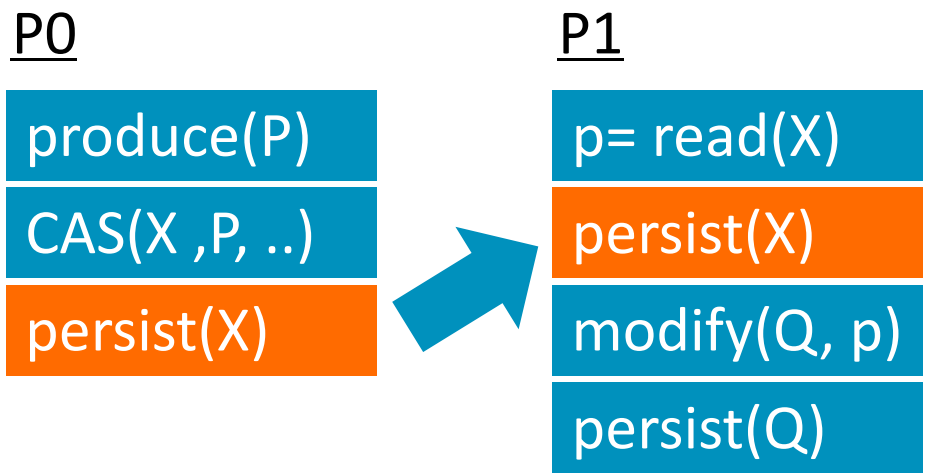
- Readers persist all locations read -> bloat, slow
- Tell reader to persist / to wait
- Single out-of-band location -> scalability??
- Multiple out-of-band locations -> hello mini-STM ?!?
- Borrow payload -> steals payload bits
  - Wang, T., Levandoski, J. and Larson, P.A., 2018, April. Easy lock-free indexing in non-volatile memory. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*



arm

In Software

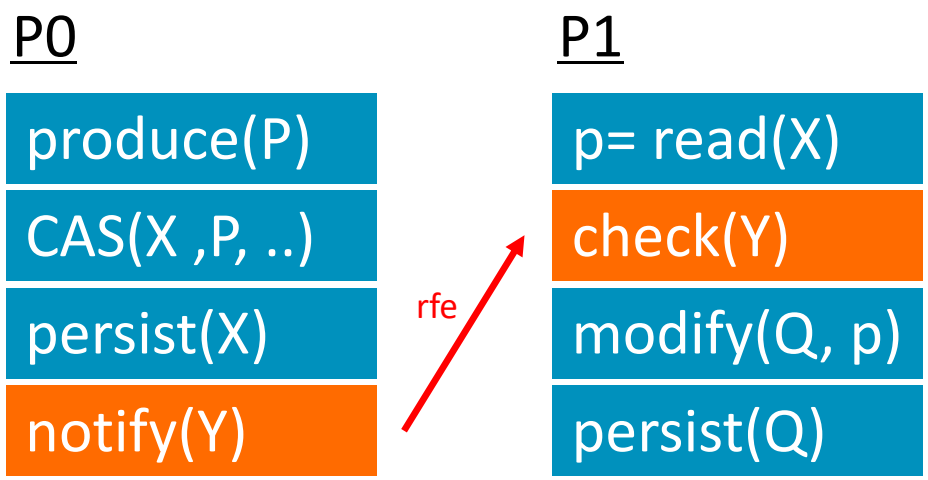
- Readers persist all locations read -> bloat, slow
- Tell reader to persist / to wait
- Single out-of-band location -> scalability??
- Multiple out-of-band locations -> hello mini-STM ?!?
- Borrow payload -> steals payload bits
  - Wang, T., Levandoski, J. and Larson, P.A., 2018, April. Easy lock-free indexing in non-volatile memory. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*



arm

In Software

- Readers persist all locations read -> bloat, slow
- Tell reader to persist / to wait
- Single out-of-band location -> scalability??
- Multiple out-of-band locations -> hello mini-STM ?!?
- Borrow payload -> steals payload bits
  - Wang, T., Levandoski, J. and Larson, P.A., 2018, April. Easy lock-free indexing in non-volatile memory. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*



arm

Summary

- Persistent memory introduces a new level of reasoning
- Arm ISA extensions for flushing to *point of persistence:* DC CVAP

- Simple persist operations do not allow transitive ordering of persists
- Tricky case closing store of lock-free section
- Extending the ISA (and uarch) to synchronize visibility and persist orders

- Next: persistency memory model extensions
- Next: performance graphs etc