M4 – Parallelism

Snooping based Cache Coherence Protocol

Outline

- Parallelism
- Flynn's classification
- Vector Processing
  - Subword Parallelism
- Symmetric Multiprocessors, Distributed Memory Machines
  - Shared Memory Multiprocessing, Message Passing
- Synchronization Primitives
  - Locks, LL-SC
- Cache coherence

Cache Coherence

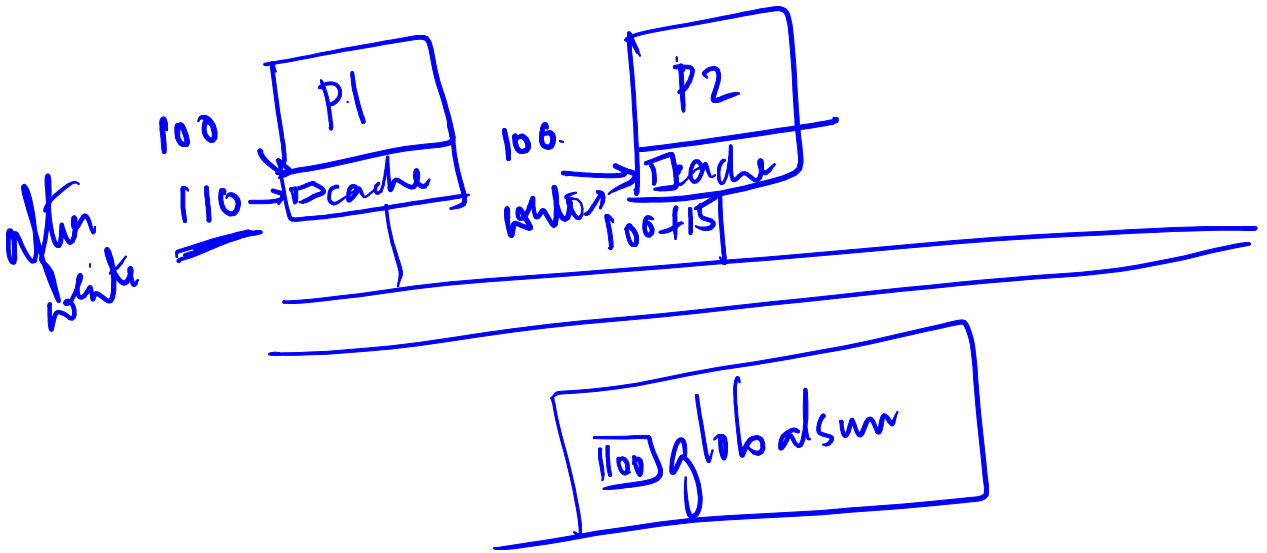
acquire lock()

globalsum += localsum;

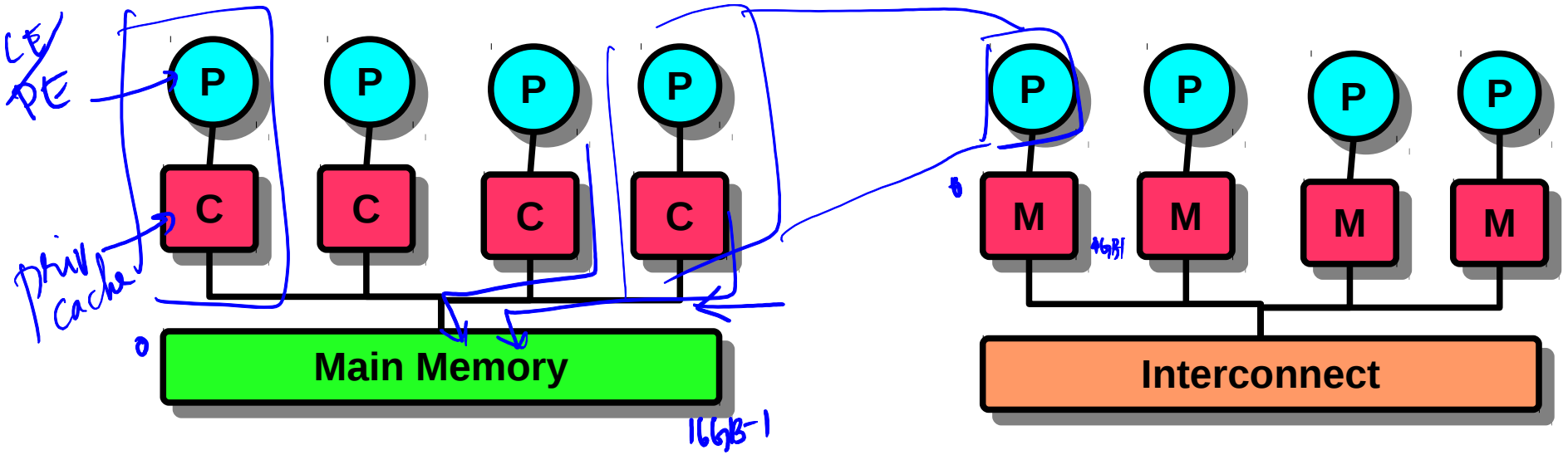
release lock

P_1 – localsum

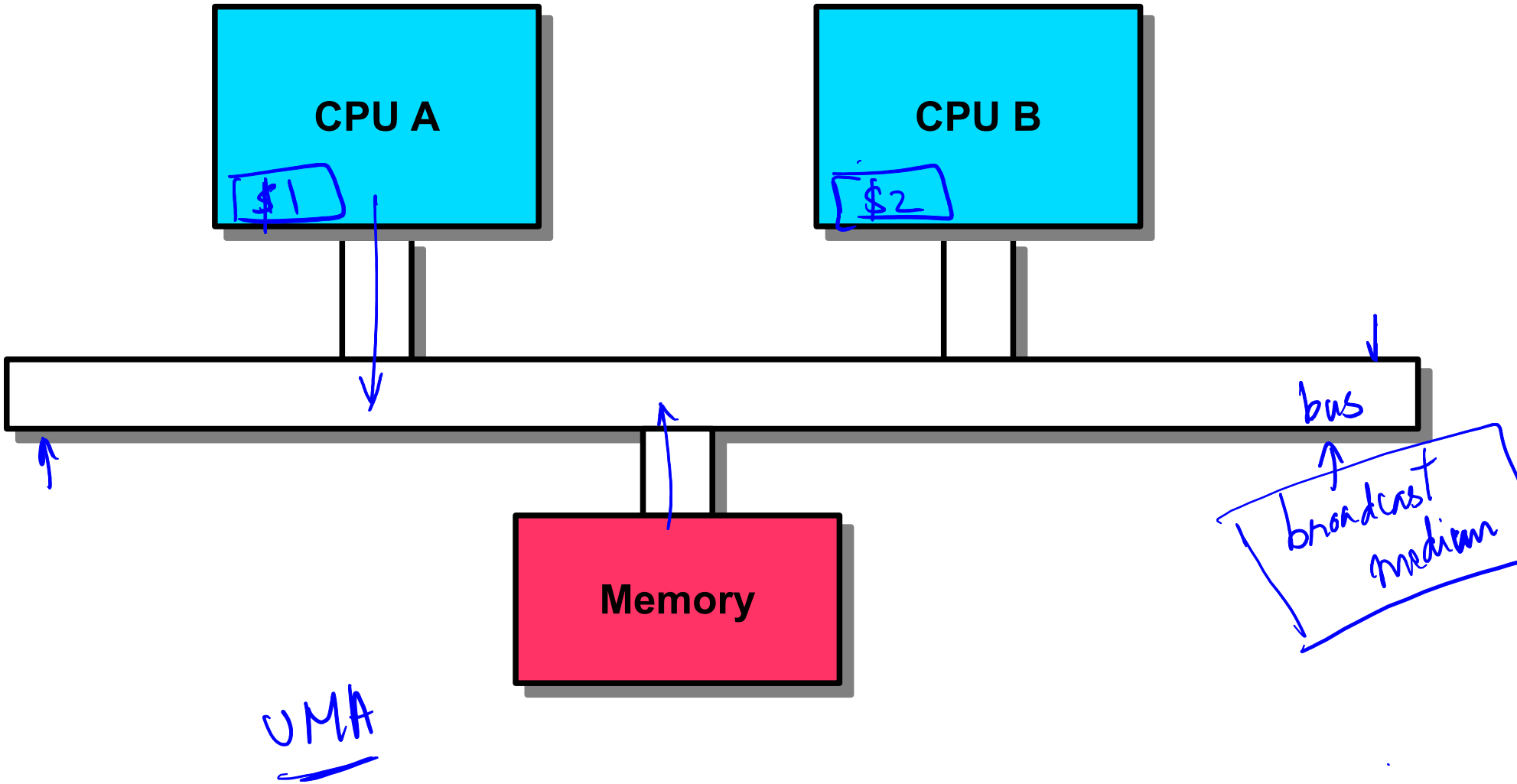
P_2 – – – –



Shared Memory vs. Distributed Memory



Multiprocessor Cache Coherence



Multiprocessor Cache Coherence

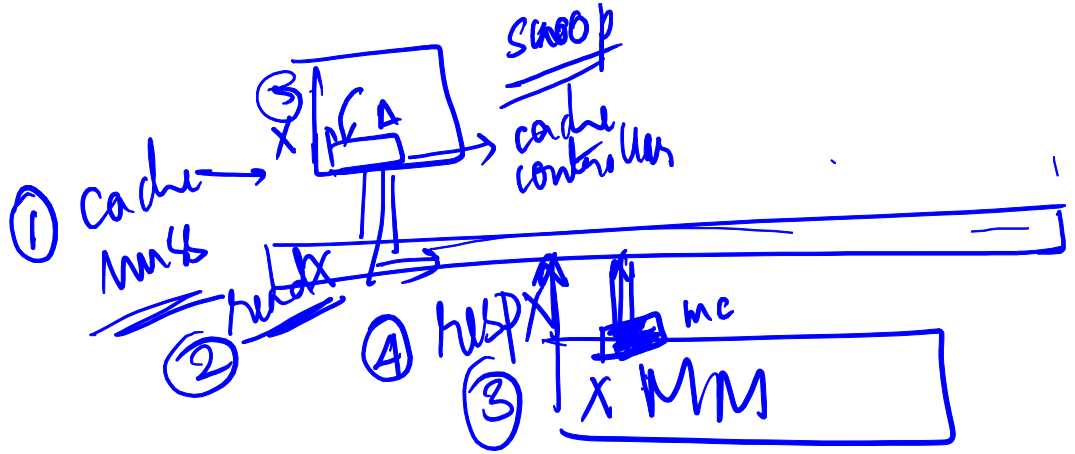
Time	Event	Cache contents for CPU A	Cache contents for CPU B	Memory contents for location X
0				1

Multiprocessor Cache Coherence

Time	Event	Cache contents for CPU A	Cache contents for CPU B	Memory contents for location X
0				1
1	CPU A reads X			1

Multiprocessor Cache Coherence

Time	Event	Cache contents for CPU A	Cache contents for CPU B	Memory contents for location X
0				1
1	CPU A reads X	1		1



Multiprocessor Cache Coherence

Time	Event	Cache contents for CPU A	Cache contents for CPU B	Memory contents for location X
0				1
1	CPU A reads X	1		1
2	CPU B reads X			

Multiprocessor Cache Coherence

Time	Event	Cache contents for CPU A	Cache contents for CPU B	Memory contents for location X
0				1
1	CPU A reads X	1		1
2	CPU B reads X	1		1

Multiprocessor Cache Coherence

Time	Event	Cache contents for CPU A	Cache contents for CPU B	Memory contents for location X
0				1
1	CPU A reads X	1		1
2	CPU B reads X	1	1	1

Multiprocessor Cache Coherence

Time	Event	Cache contents for CPU A	Cache contents for CPU B	Memory contents for location X
0				1
1	CPU A reads X	1		1
2	CPU B reads X	1	1	1
3	CPU A stores 0 into X			

Multiprocessor Cache Coherence

Time	Event	Cache contents for CPU A	Cache contents for CPU B	Memory contents for location X
0				1
1	CPU A reads X	1		1
2	CPU B reads X	1	1	1
3	CPU A stores 0 into X	0		0

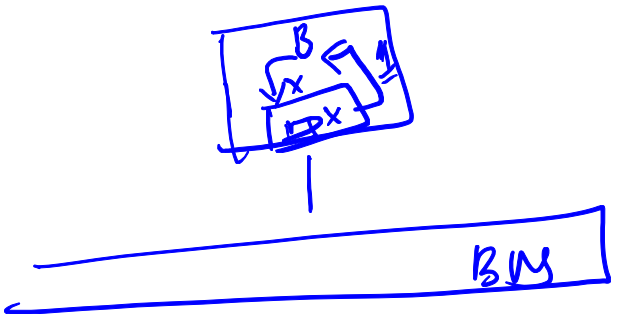
Multiprocessor Cache Coherence

Time	Event	Cache contents for CPU A	Cache contents for CPU B	Memory contents for location X
0				1
1	CPU A reads X	1		1
2	CPU B reads X	1	1	1
3	CPU A stores 0 into X	0	1	0

Multiprocessor Cache Coherence

Time	Event	Cache contents for CPU A	Cache contents for CPU B	Memory contents for location X
0				1
1	CPU A reads X	1		1
2	CPU B reads X	1	1	1
3	CPU A stores 0 into X	0	1	0

CPU B reads X



Multiprocessor Cache Coherence

Time	Event	Cache contents for CPU A	Cache contents for CPU B	Memory contents for location X
0				1
1	CPU A reads X	1		1
2	CPU B reads X	1	1	1
3	CPU A stores 0 into X	0	1→0	0

CPU B reads X

Return which value to CPU B?

↳0

Cache Coherence

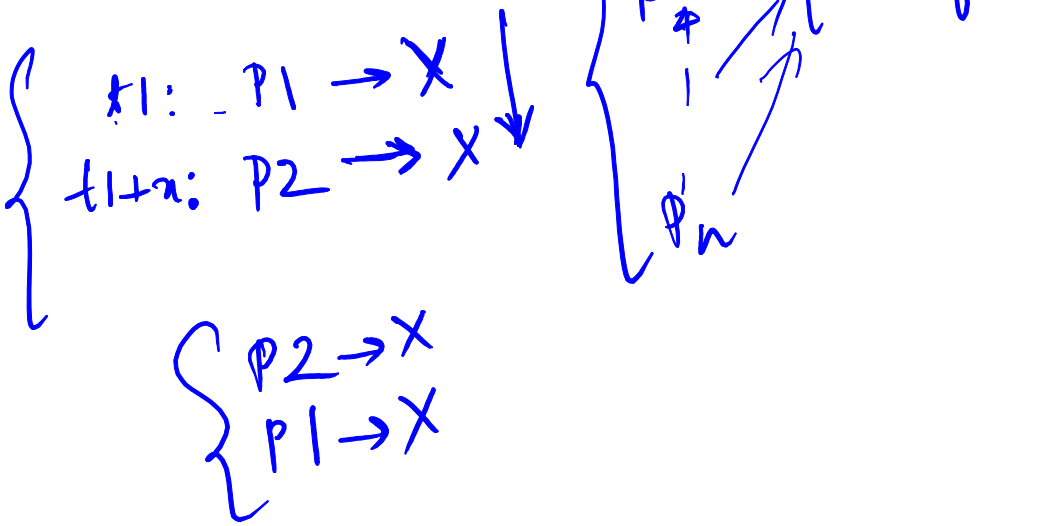
- Coherence
  - Which value to return on a read

Cache Coherence

- Coherence
  - Which value to return on a read
- A memory system is coherent if:
  - * any modification in P_1, visible to $P_2, P_3 \cdots P_n$

Cache Coherence

- Coherence
  - Which value to return on a read
- A memory system is coherent if:
  - Write Propagation
    -
  - Write Serialization
    -



Cache Coherence

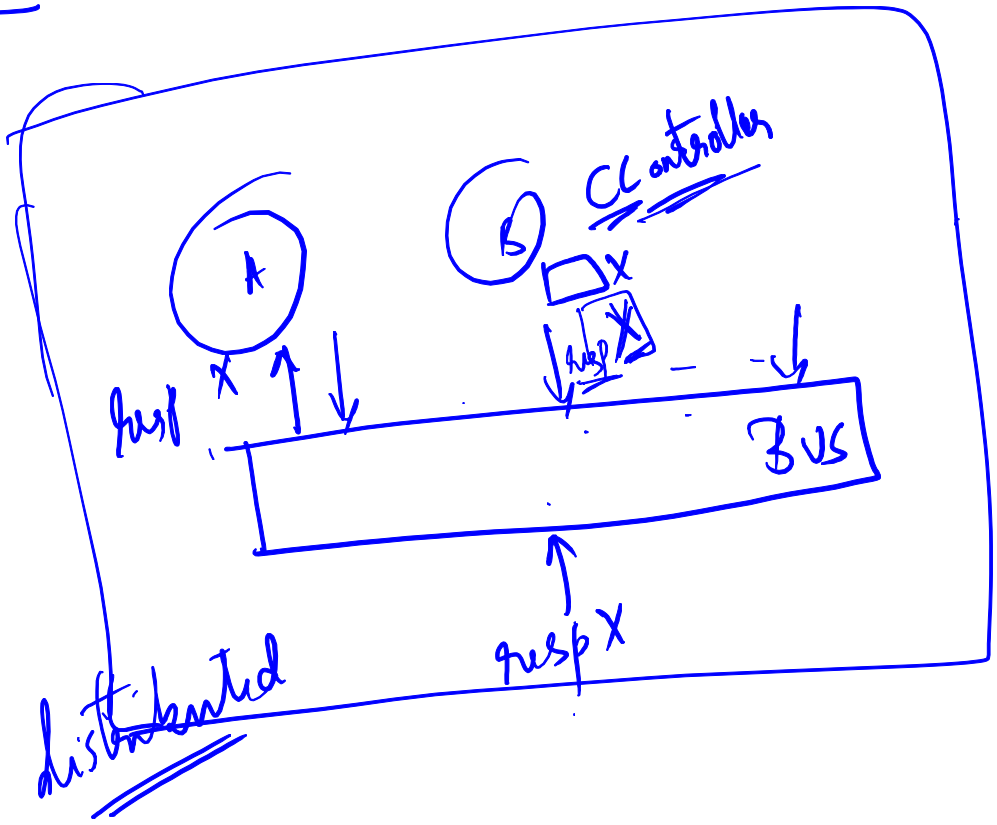
- Coherence
  - Which value to return on a read
- A memory system is coherent if:
  - Write Propagation
    - A write is visible after a sufficient time lapse
  - Write Serialization
    -

Cache Coherence

- Coherence
  - Which value to return on a read
- A memory system is coherent if:
  - Write Propagation
    - A write is visible after a sufficient time lapse
  - Write Serialization
    - All writes to a location are seen by every processor in the same order

Cache Coherence

- **Directory** based protocols
  -
- **Snooping** protocols
  -
  -



Cache Coherence

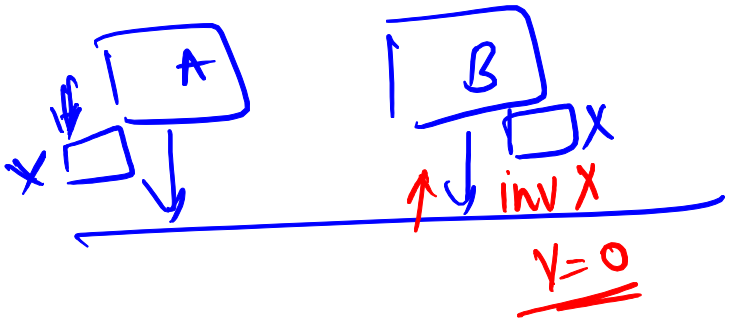
- **Directory** based protocols
  - Sharing status maintained in a directory
- **Snooping** protocols
  -
  -

Cache Coherence

- **Directory** based protocols
  - Sharing status maintained in a directory
- **Snooping** protocols
  - Sharing status is stored in the cache controller
  - Cache controller snoops broadcast medium

Cache Coherence

- **Write Invalidate** protocols
  -
- **Write Update** protocols
  -



Cache Coherence

- **Write Invalidate** protocols
  - Invalidates other processors' copies on a write
- **Write Update** protocols
  -

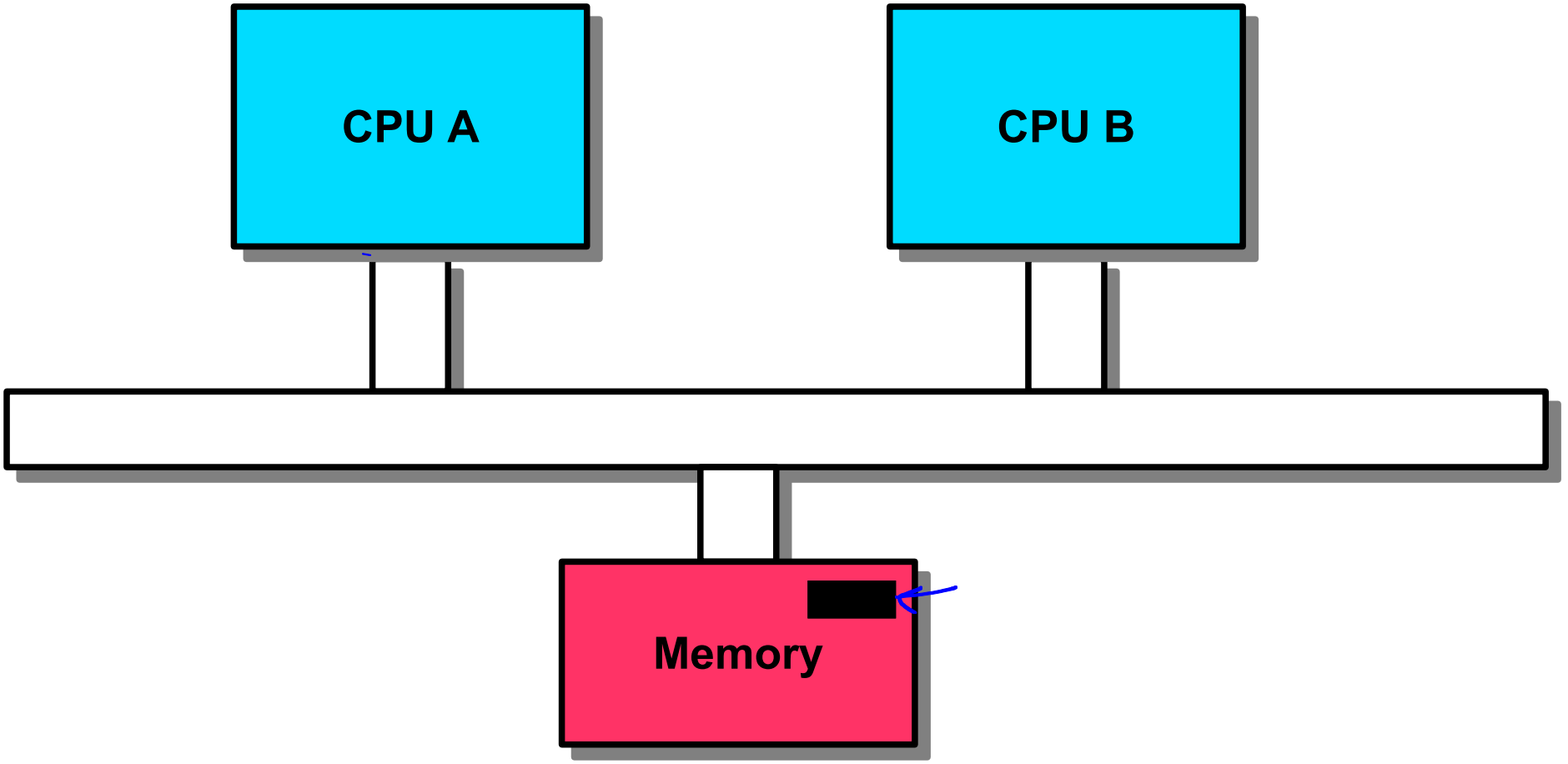
Cache Coherence

- **Write Invalidate** protocols
  - Invalidates other processors' copies on a write
- **Write Update** protocols
  - Updates all data copies on a write

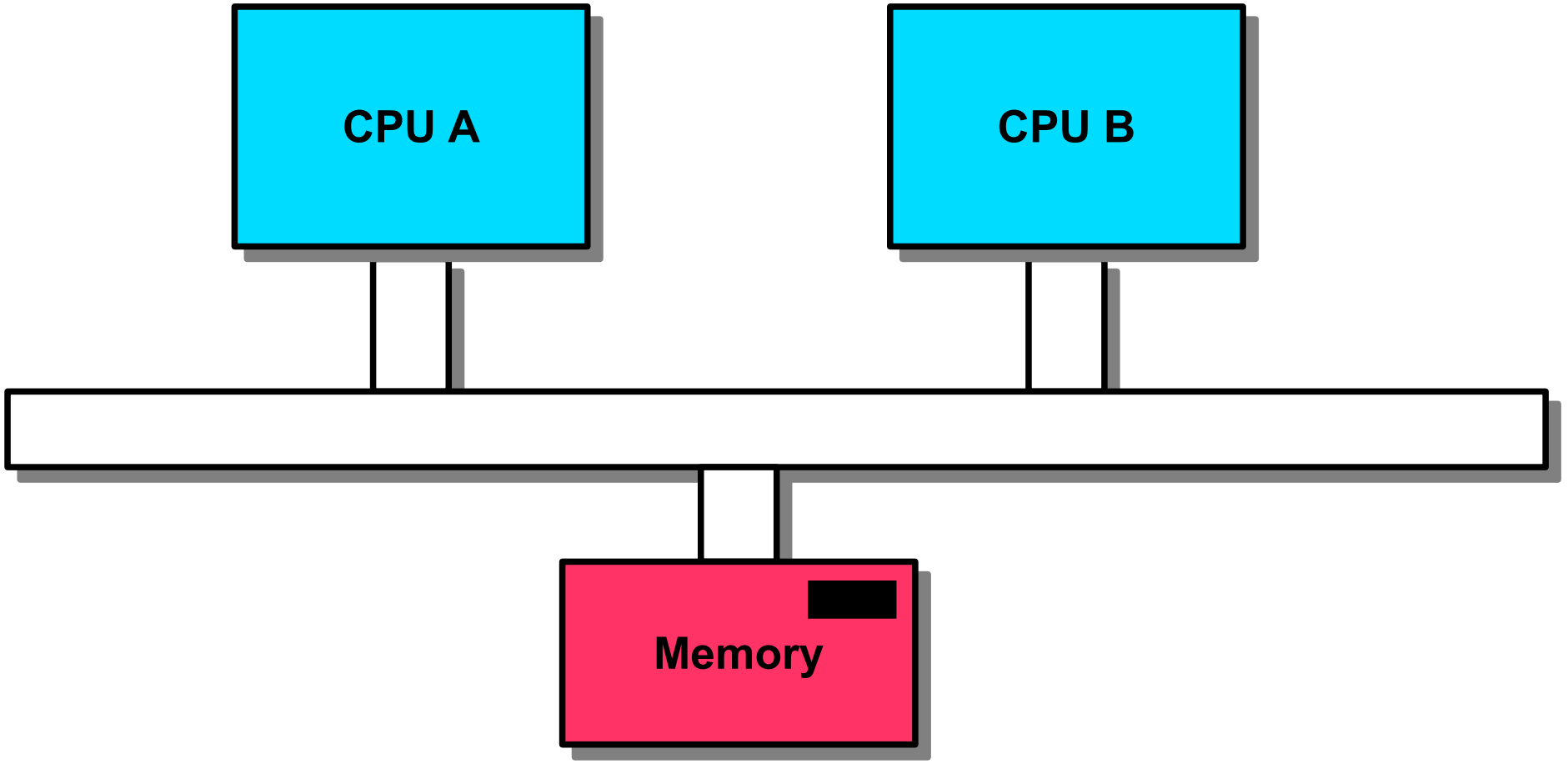
Cache Coherence

- Sharing Status
  - **Invalid (I)**
  - **Shared (S)** (or Clean)
  - **Modified (M)** (or Dirty)

SMP - Write Invalidate

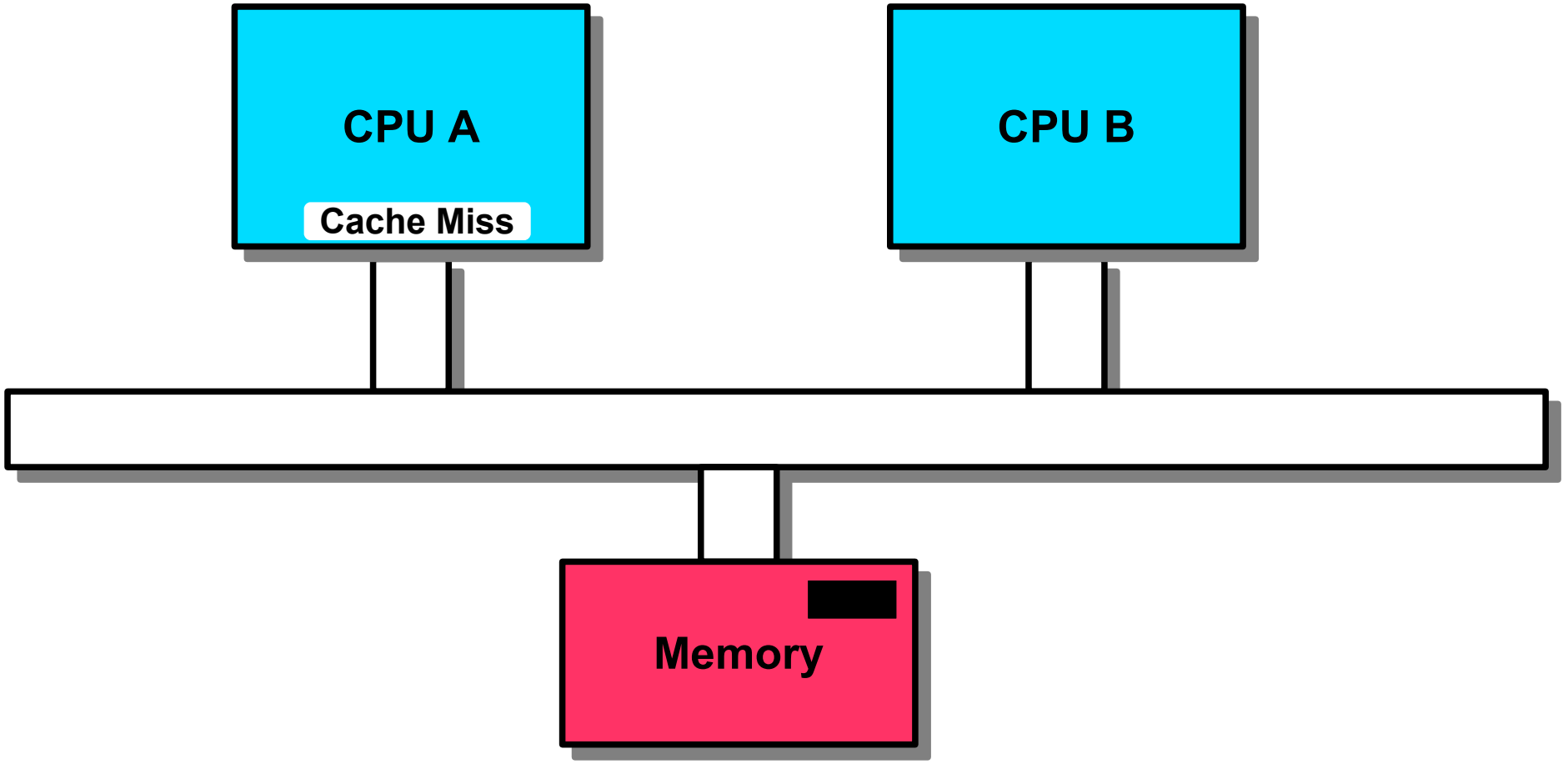


SMP - Write Invalidate



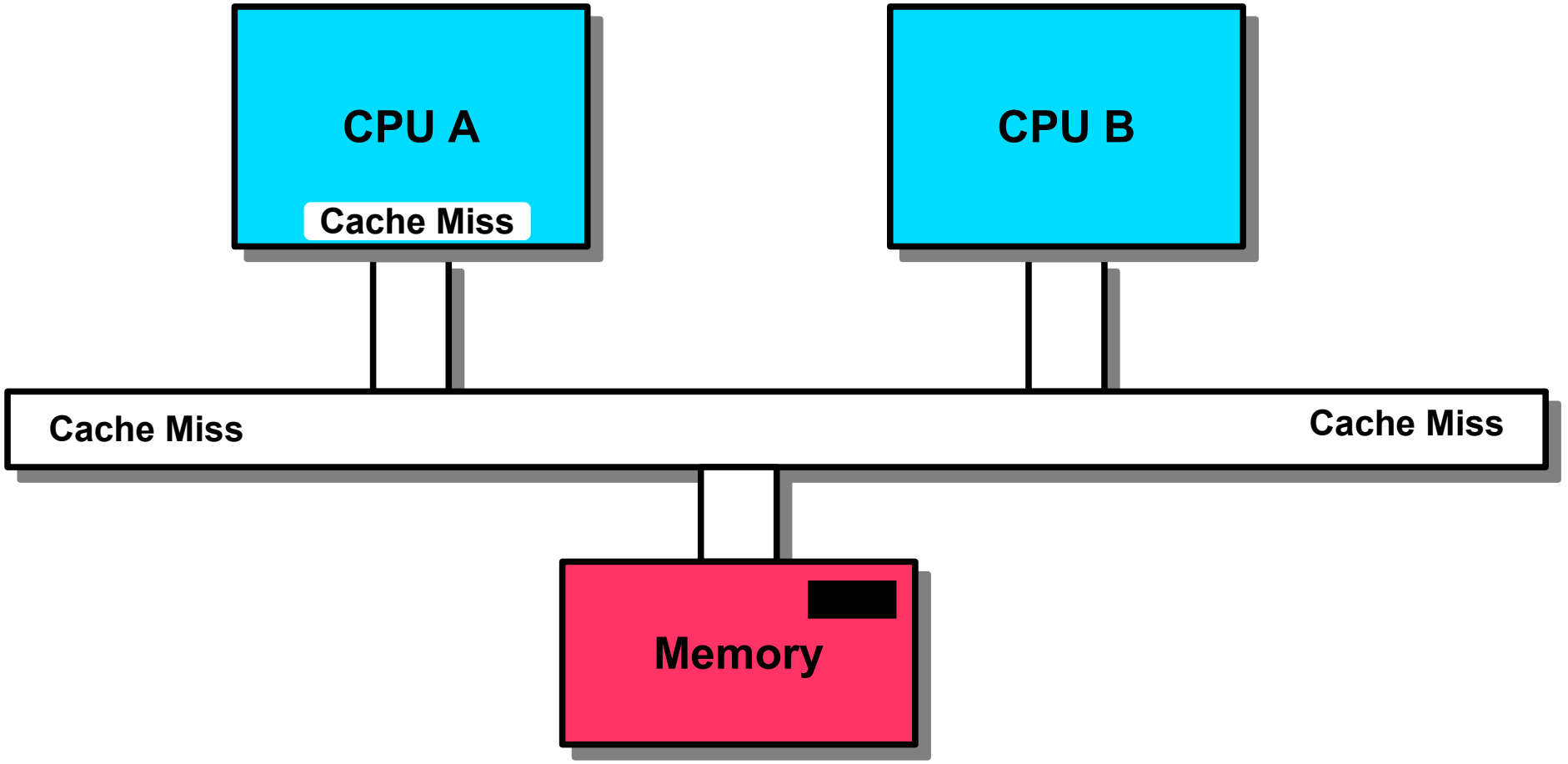
CPU A reads X

SMP - Write Invalidate



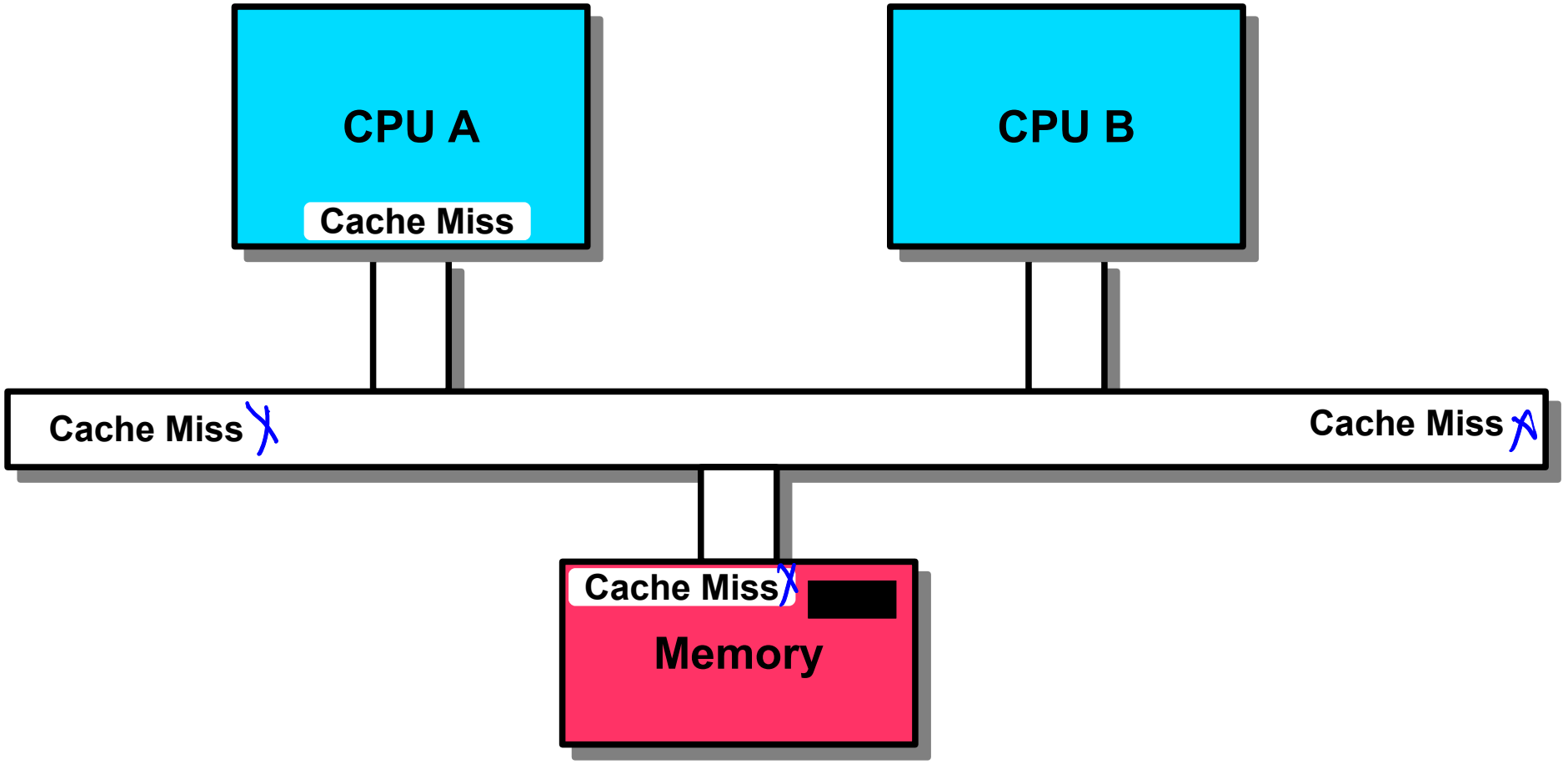
CPU A reads X

SMP - Write Invalidate



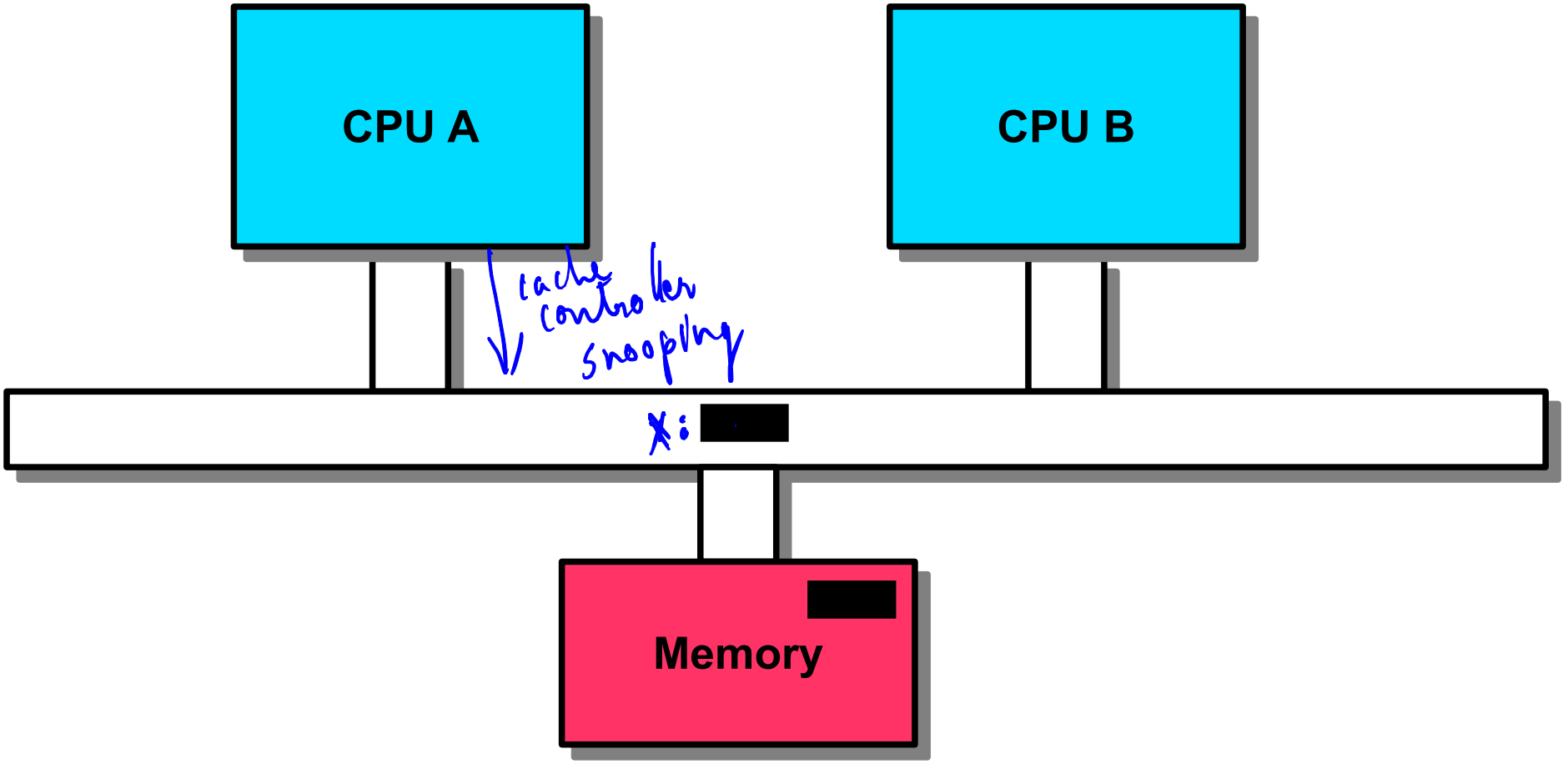
CPU A reads X

SMP - Write Invalidate



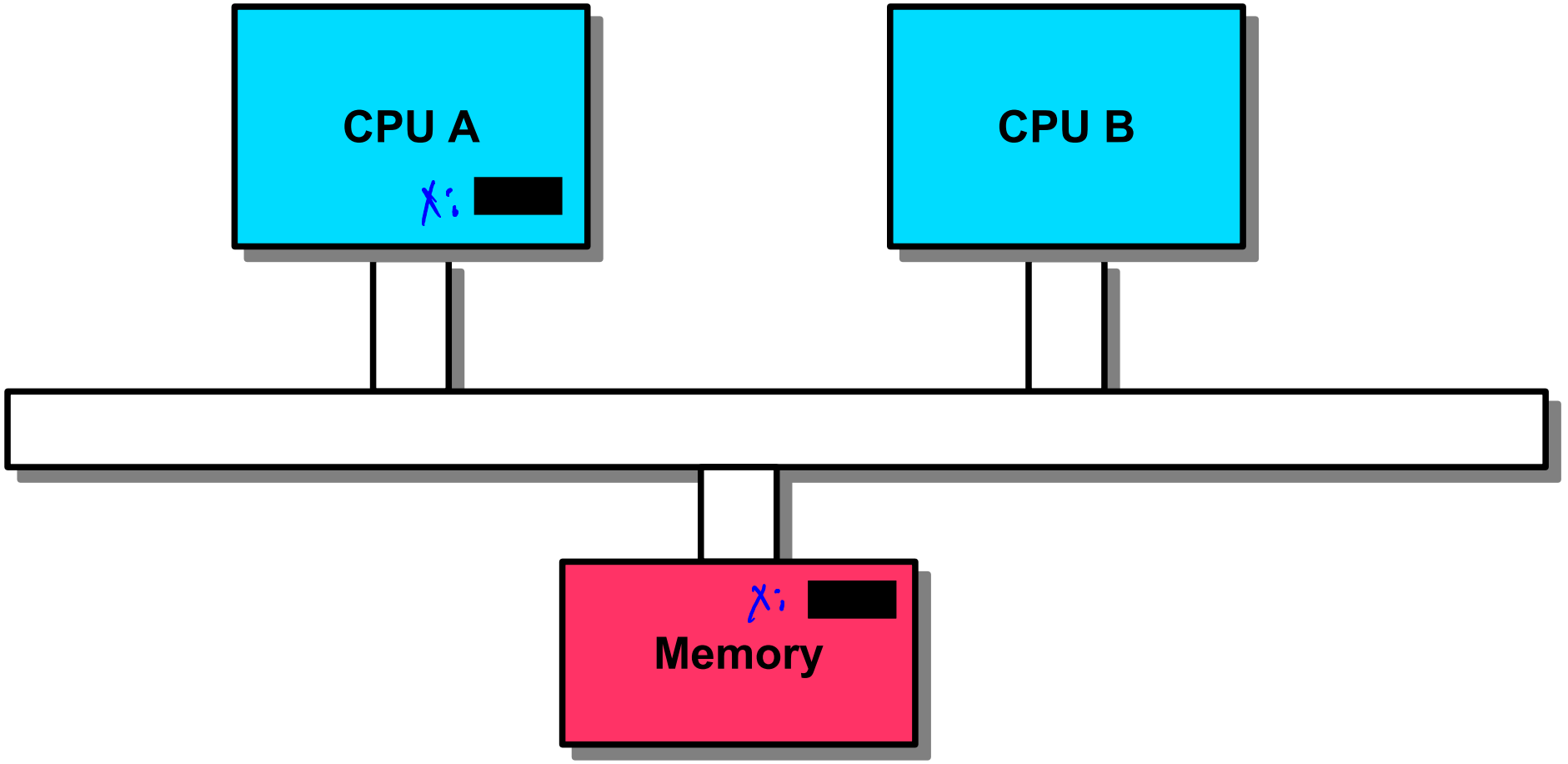
CPU A reads X

SMP - Write Invalidate



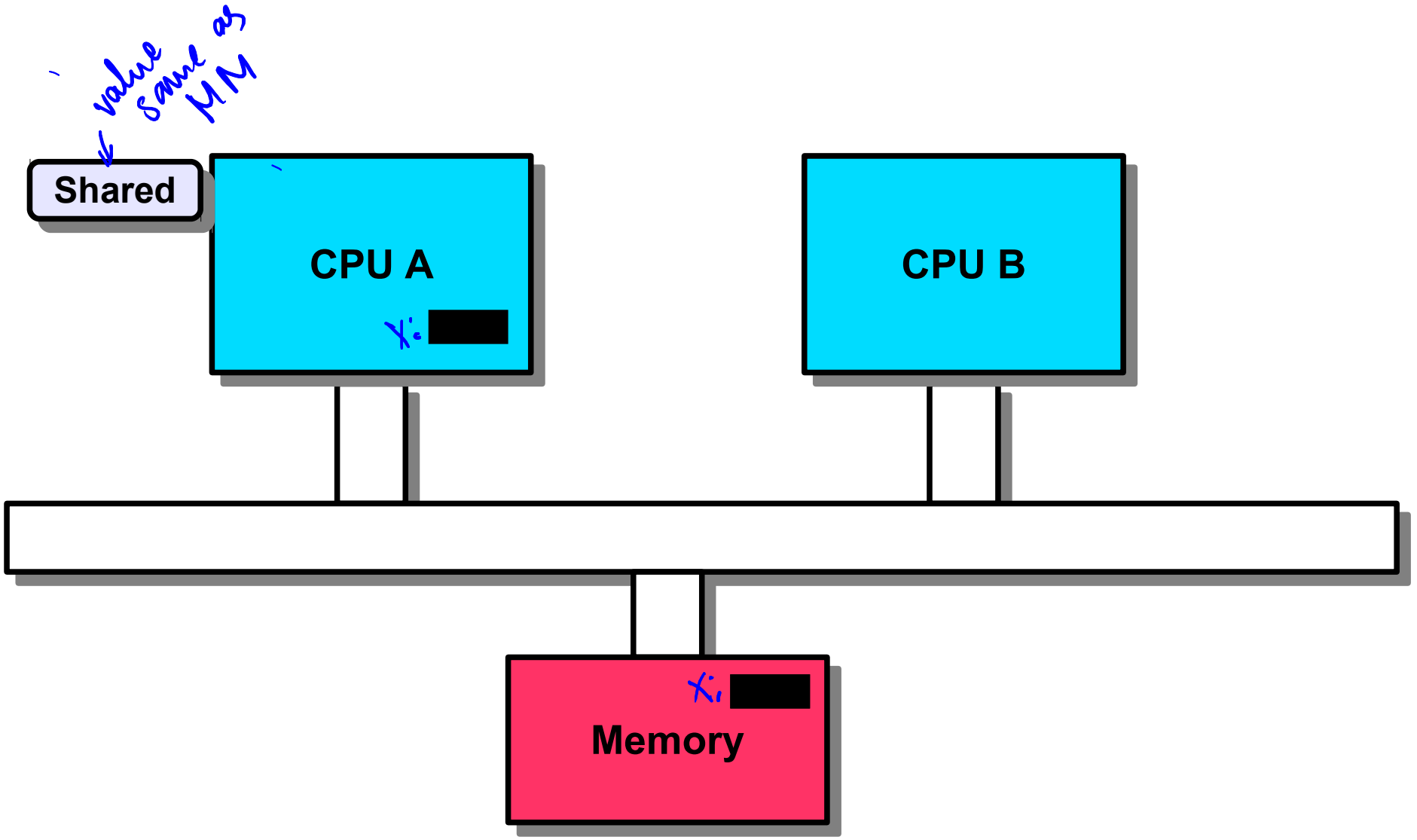
CPU A reads X

SMP - Write Invalidate



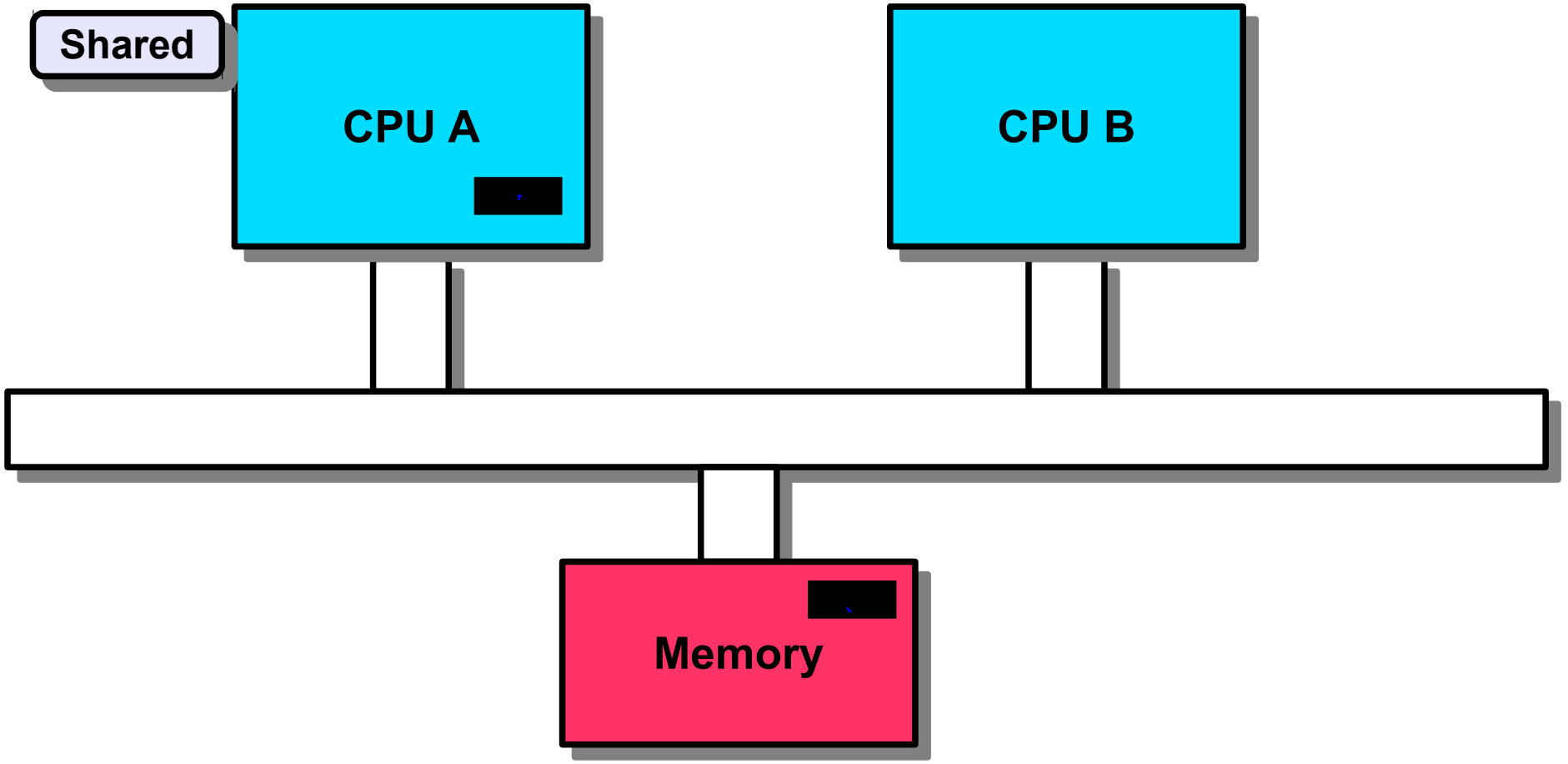
CPU A reads X

SMP - Write Invalidate

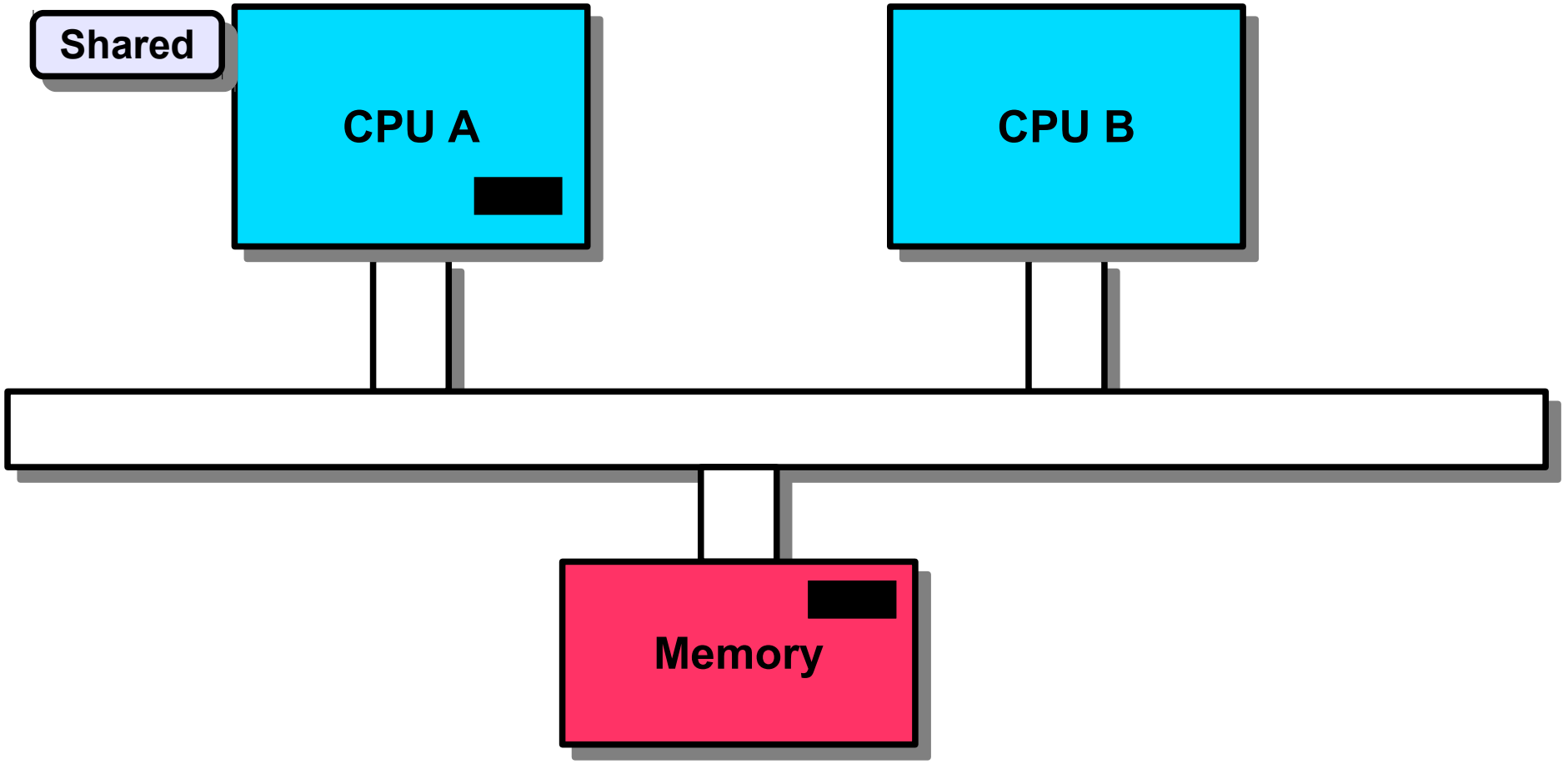


CPU A reads X

SMP - Write Invalidate

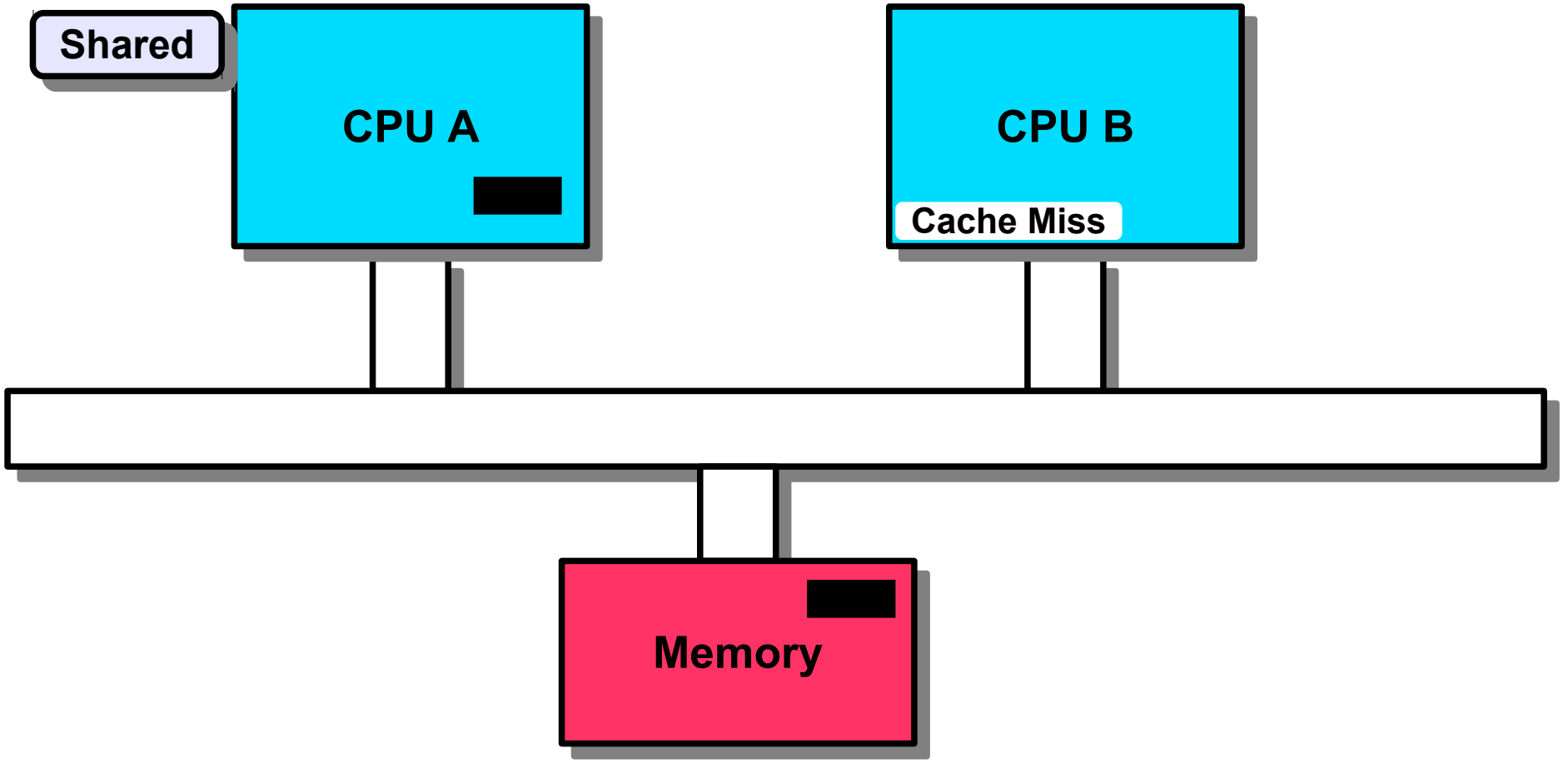


SMP - Write Invalidate



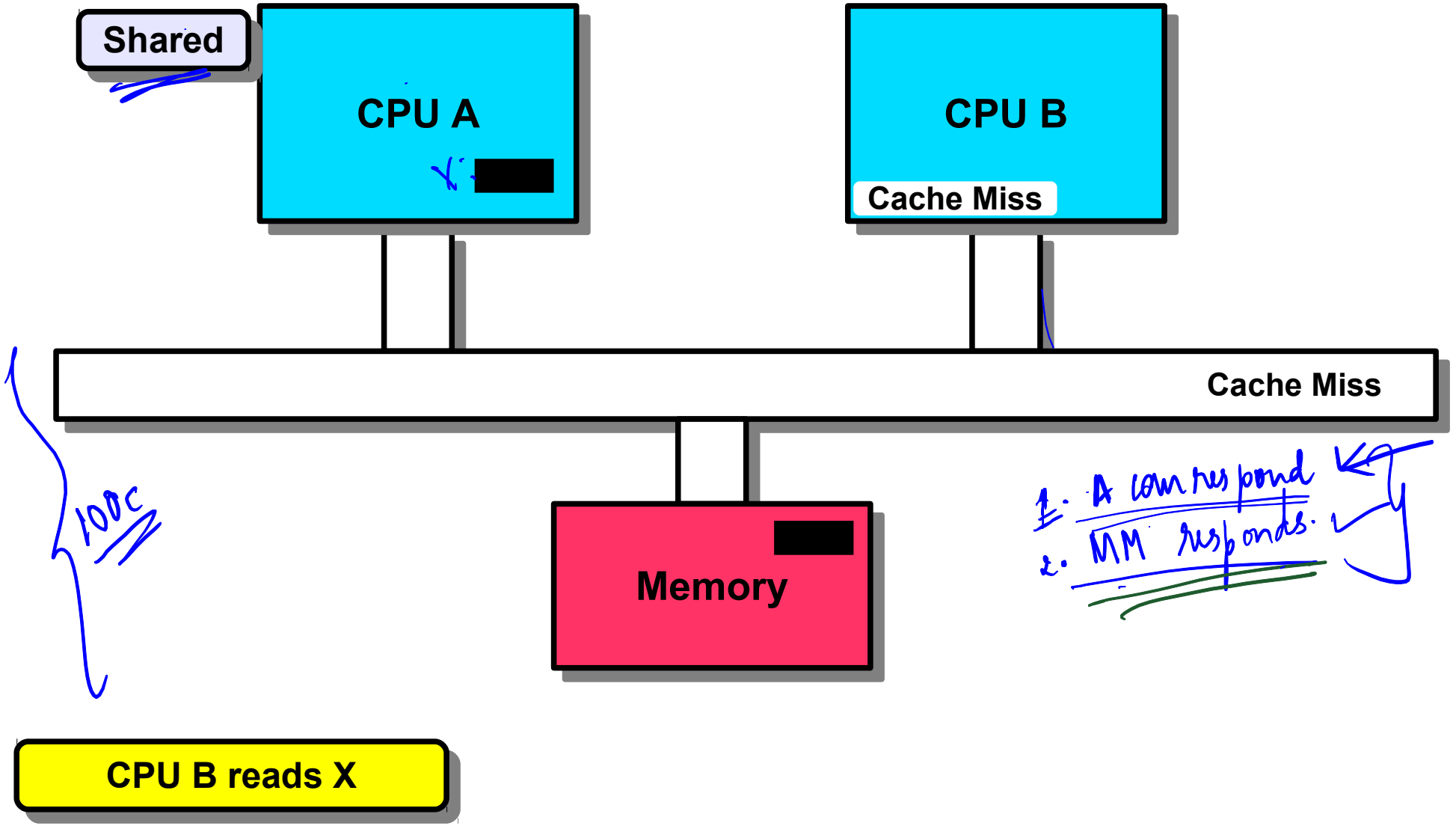
CPU B reads X

SMP - Write Invalidate

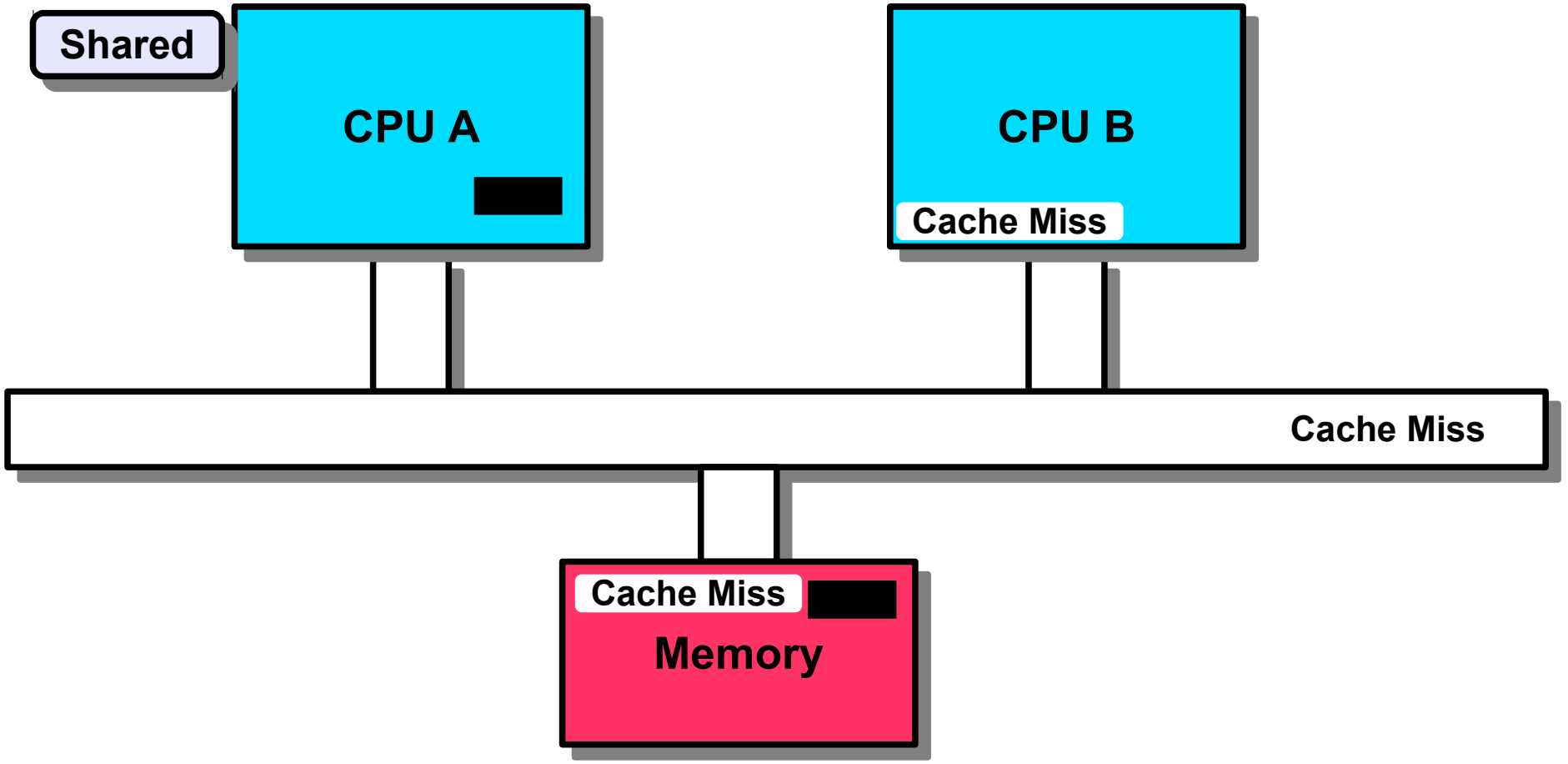


CPU B reads X

SMP - Write Invalidate

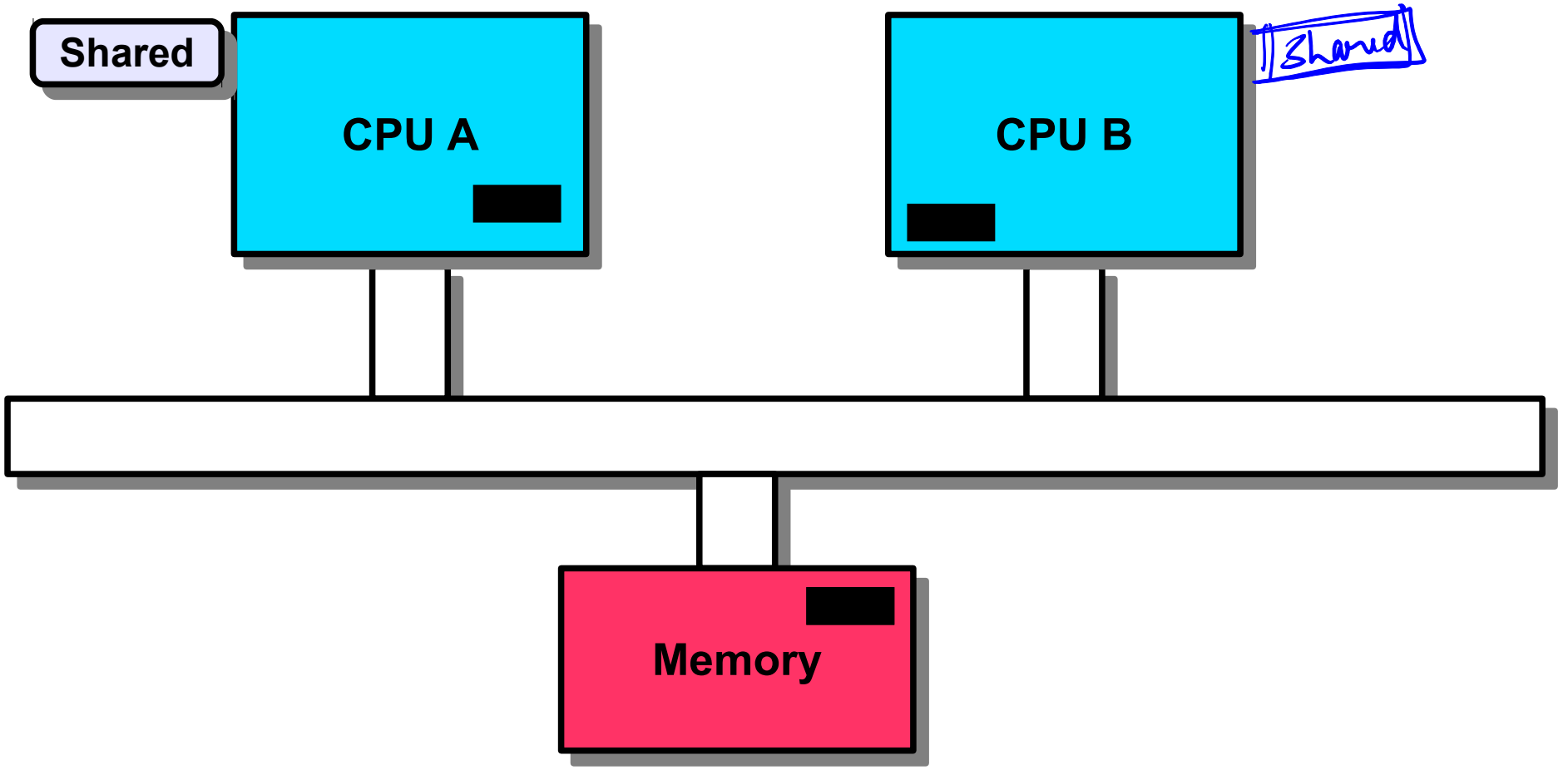


SMP - Write Invalidate



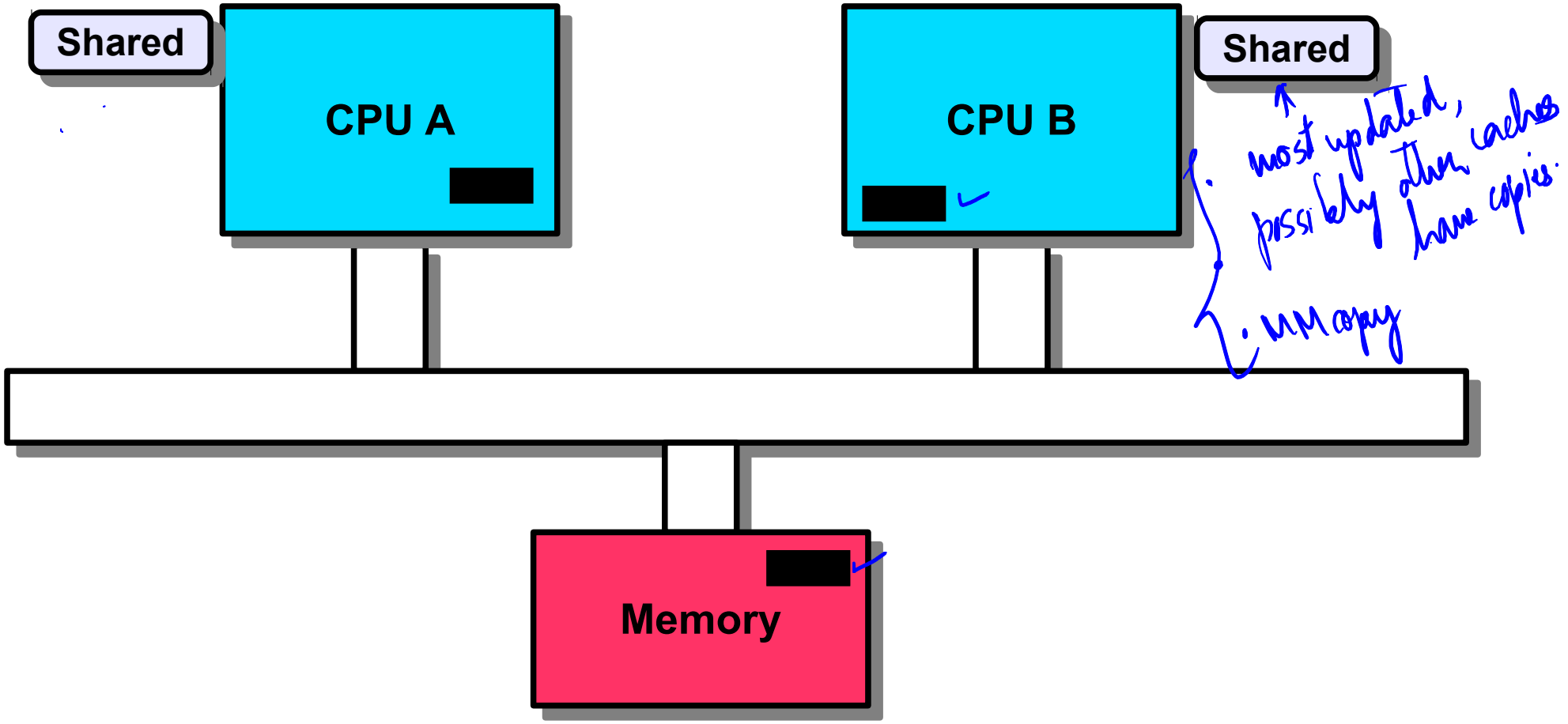
CPU B reads X

SMP - Write Invalidate



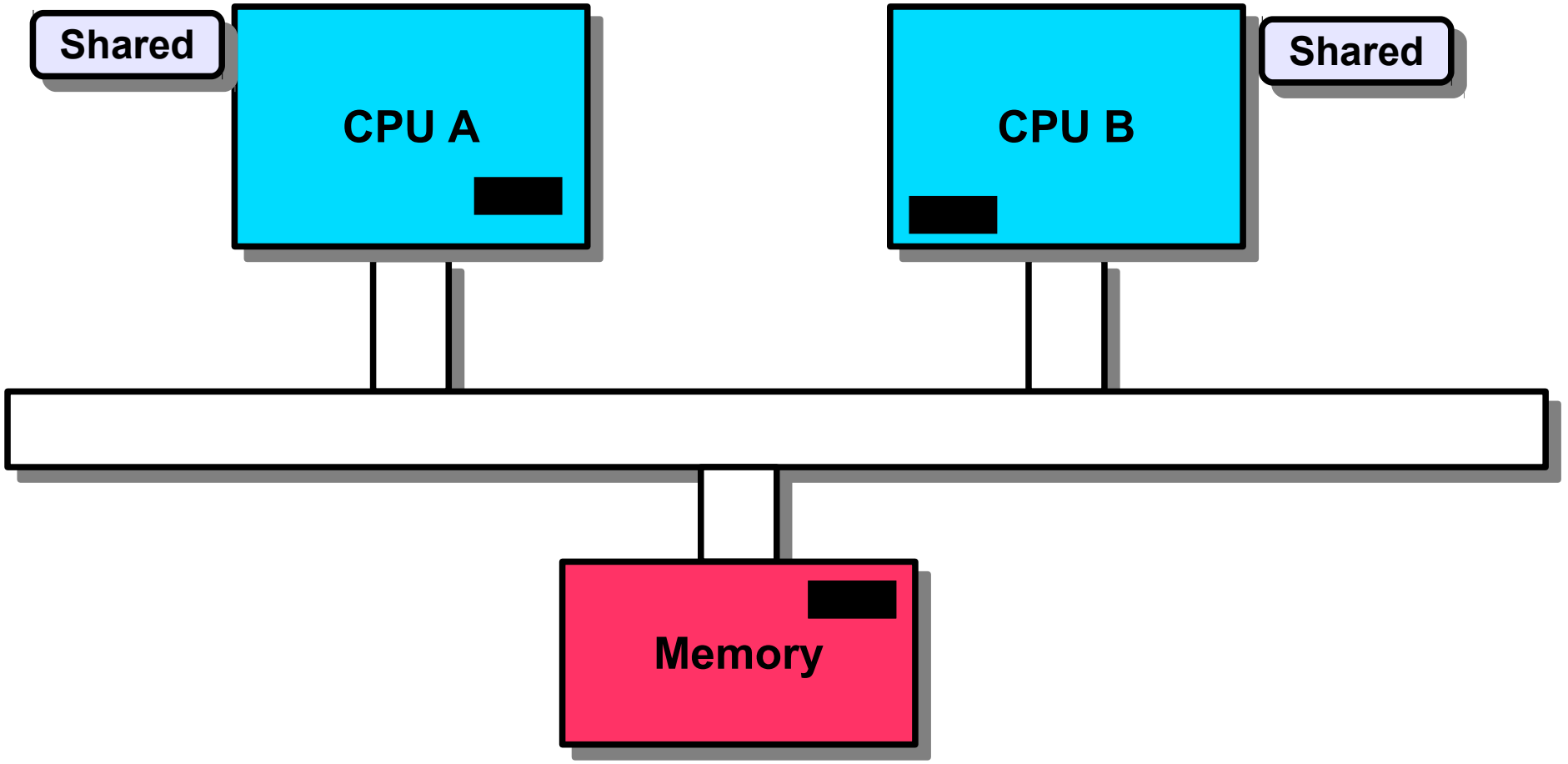
CPU B reads X

SMP - Write Invalidate

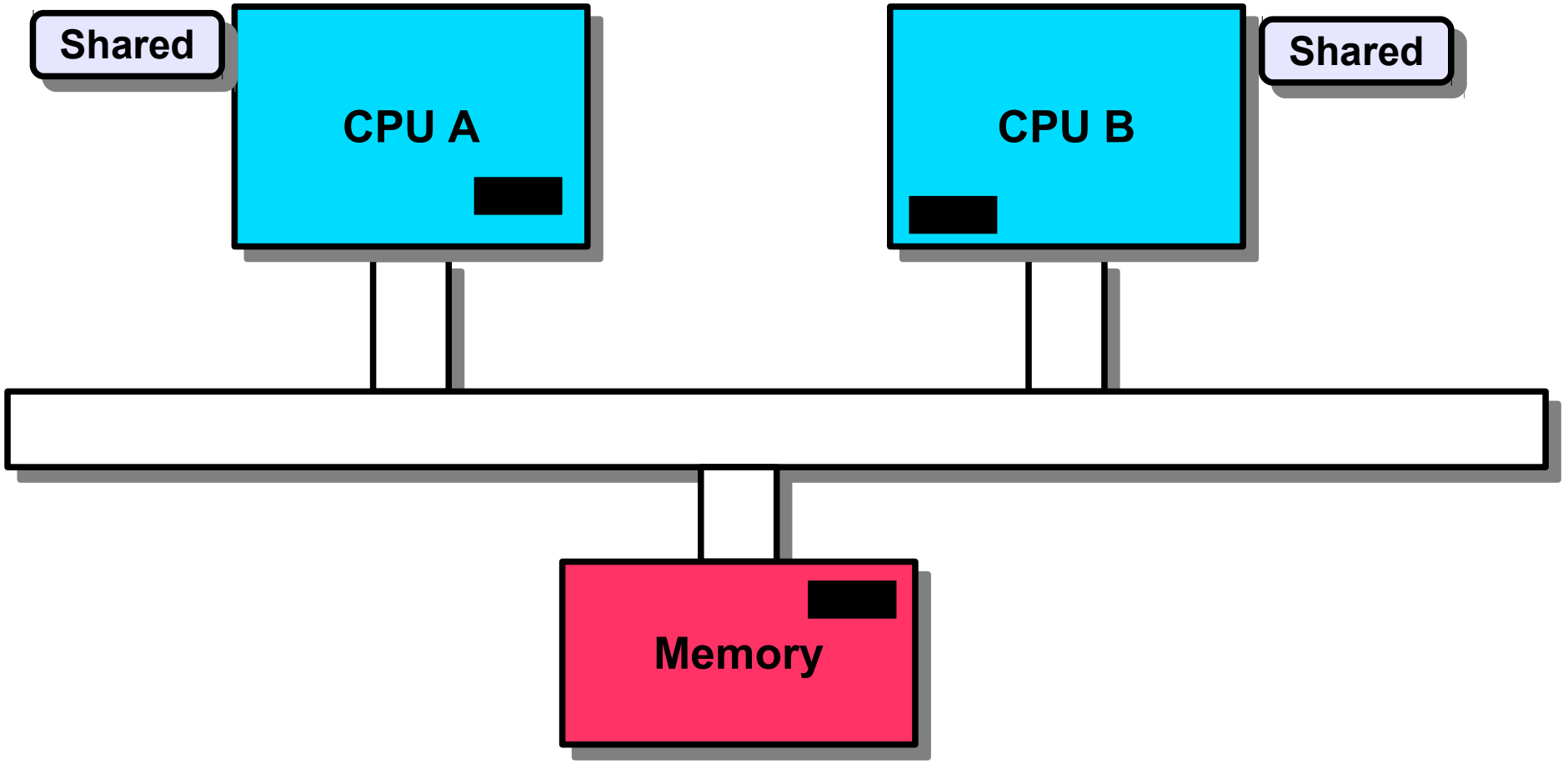


CPU B reads X

SMP - Write Invalidate

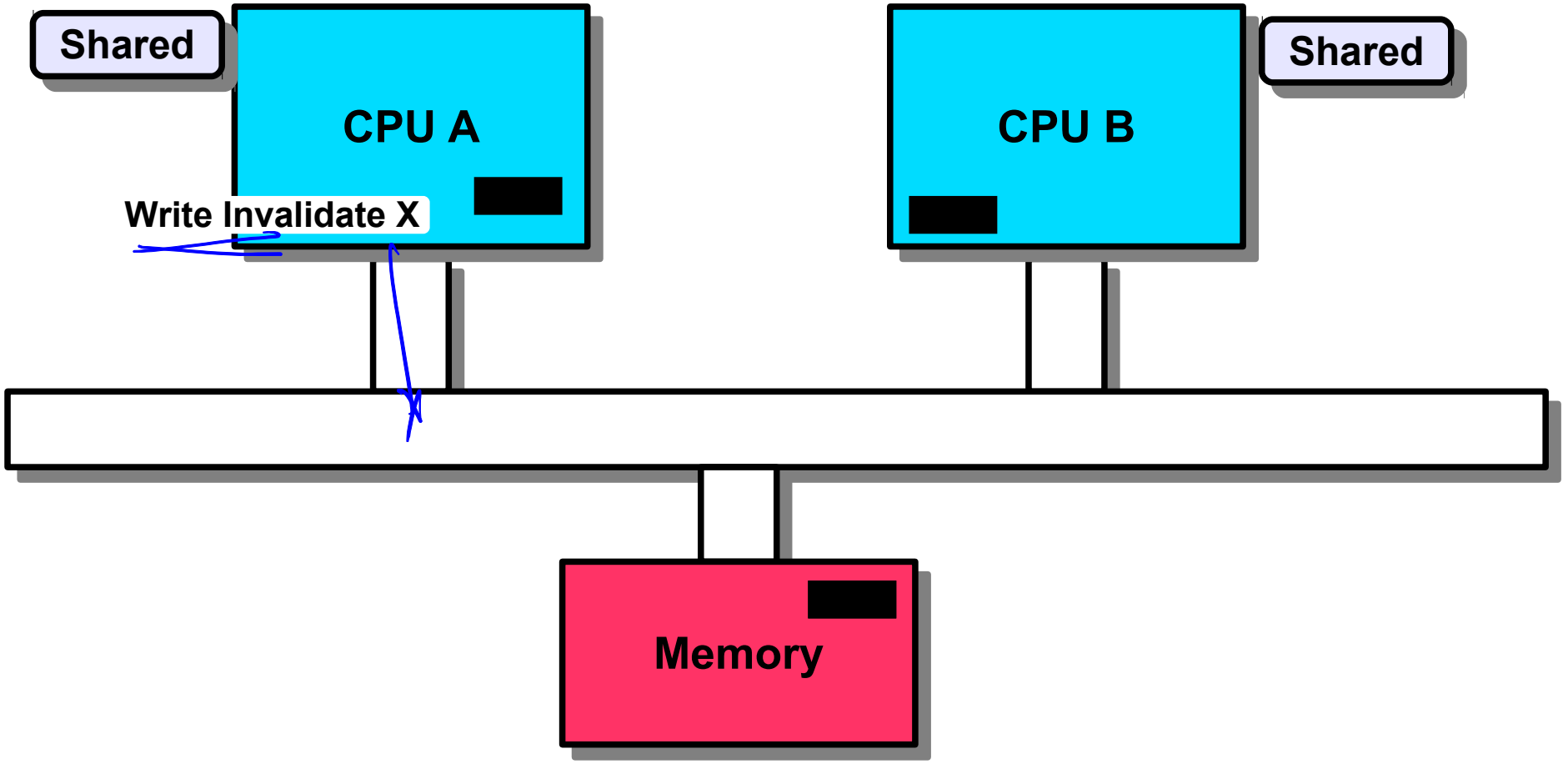


SMP - Write Invalidate



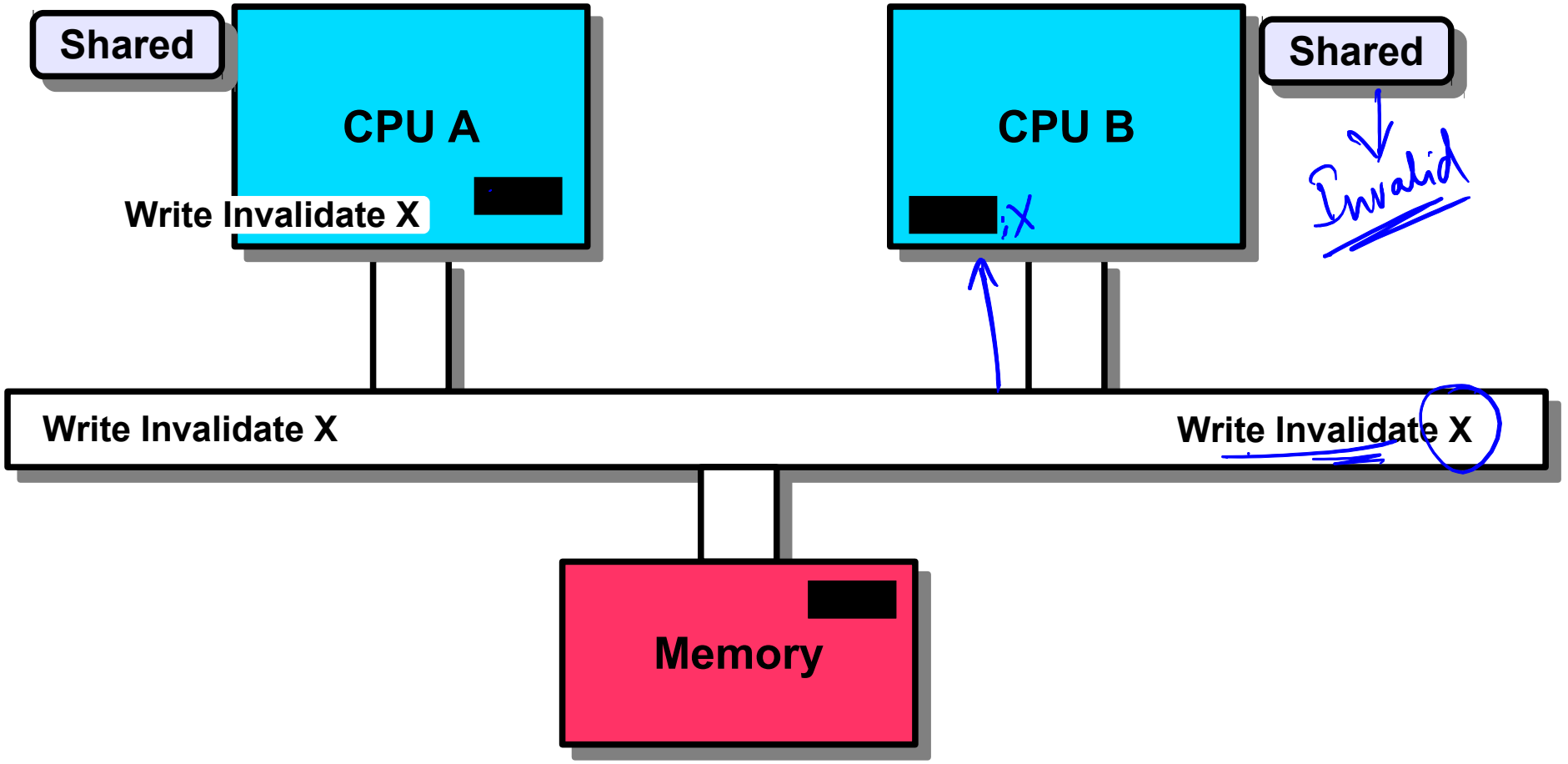
CPU A writes X

SMP - Write Invalidate



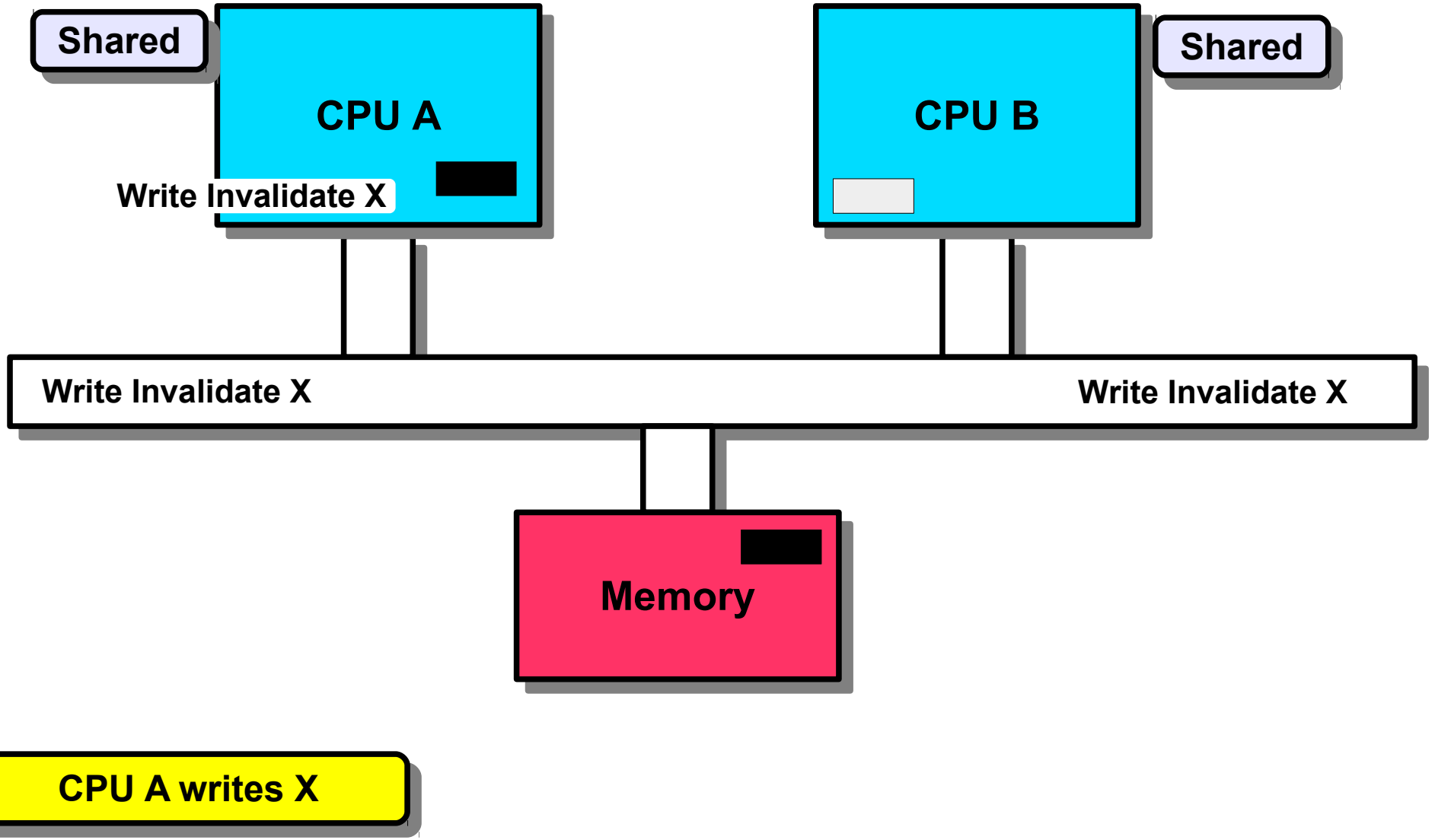
CPU A writes X

SMP - Write Invalidate

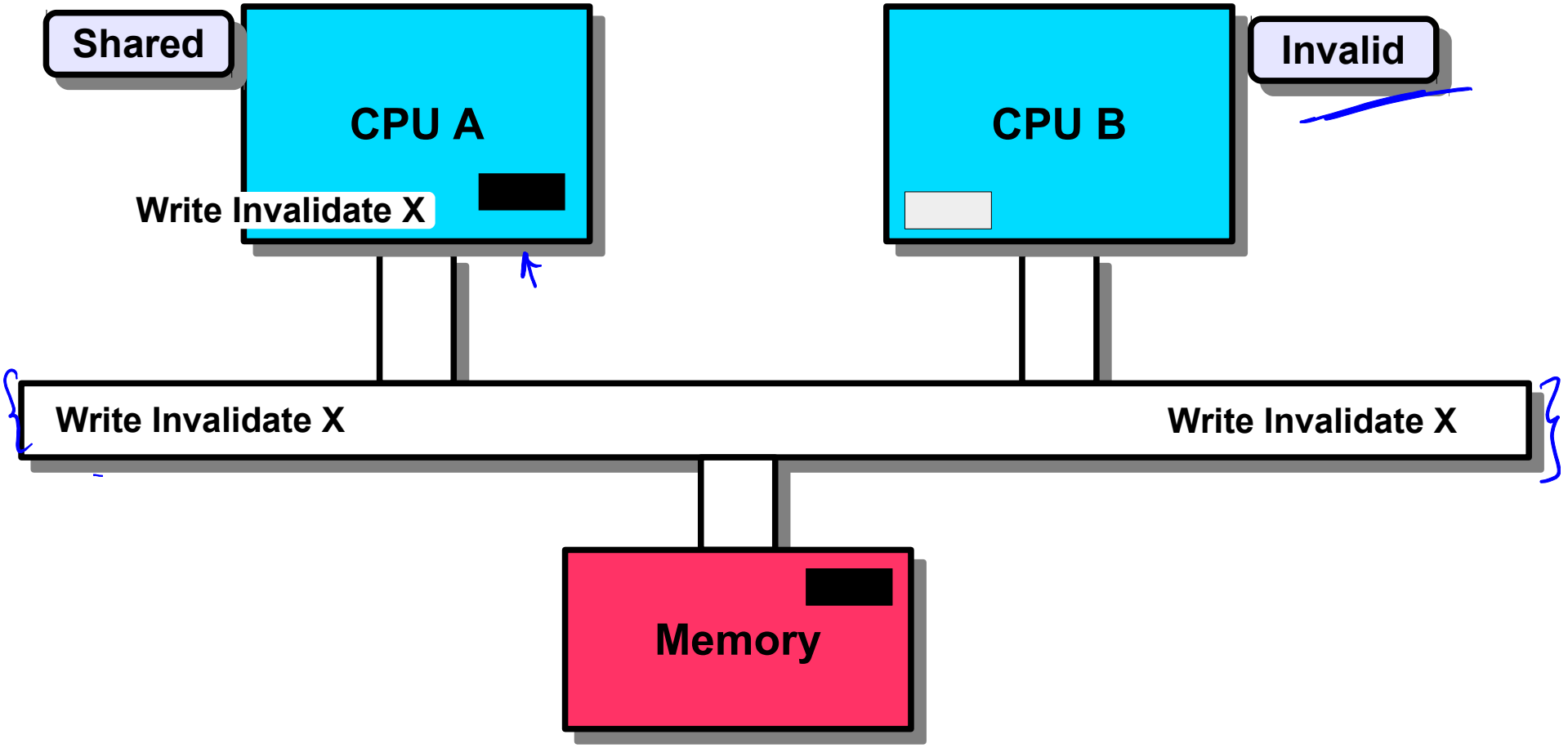


CPU A writes X

SMP - Write Invalidate

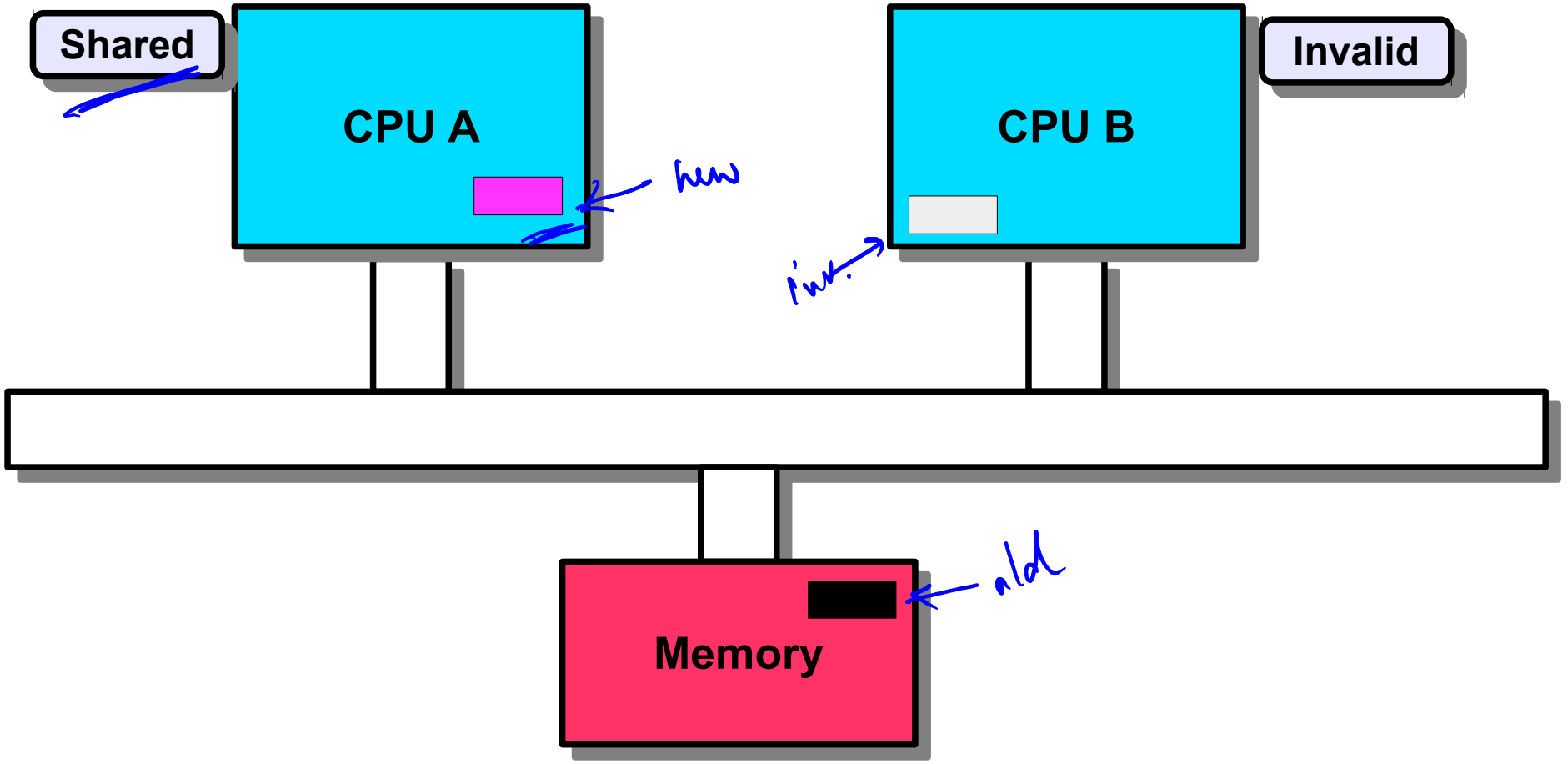


SMP - Write Invalidate



CPU A writes X

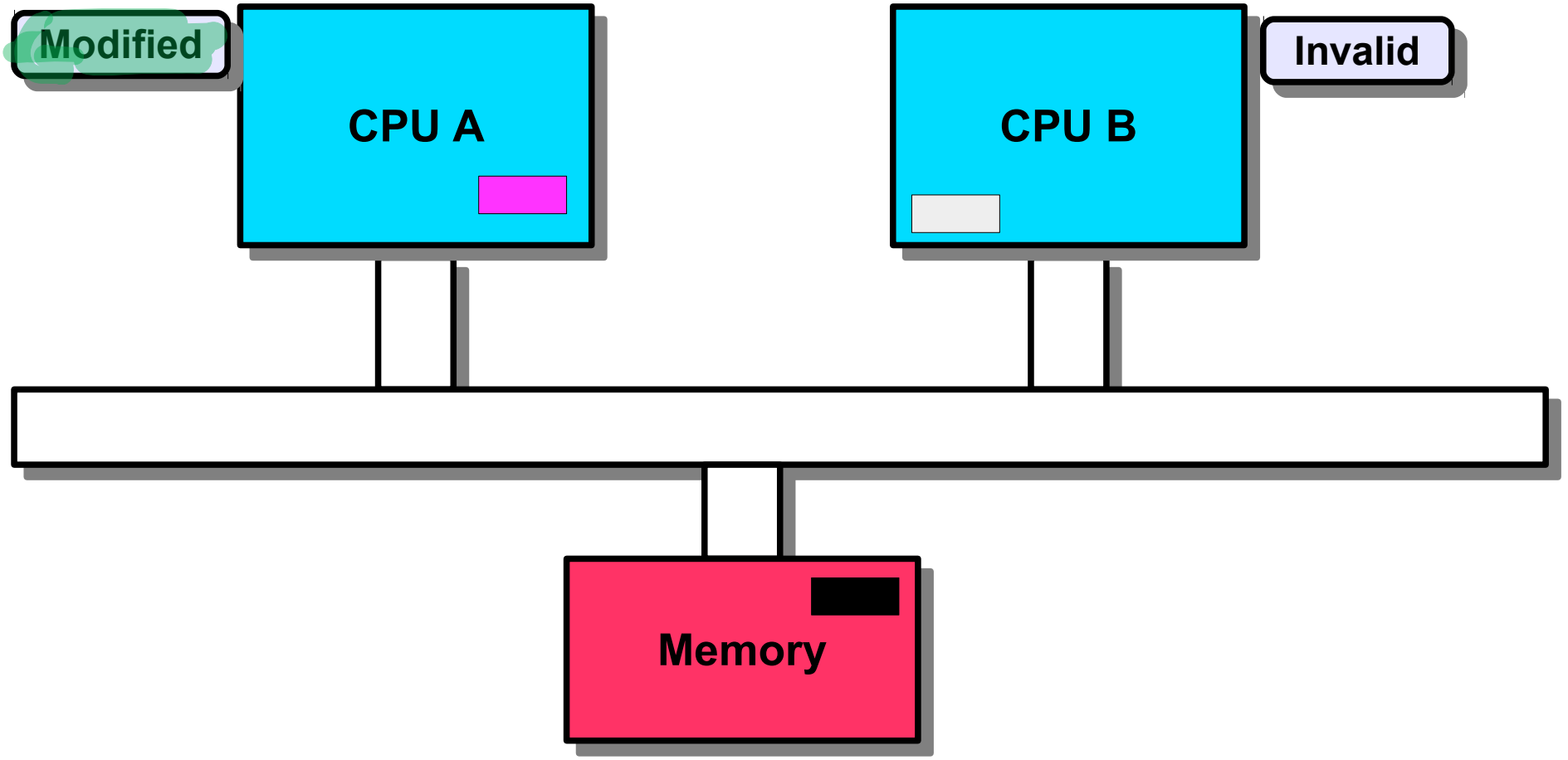
SMP - Write Invalidate



CPU A writes X

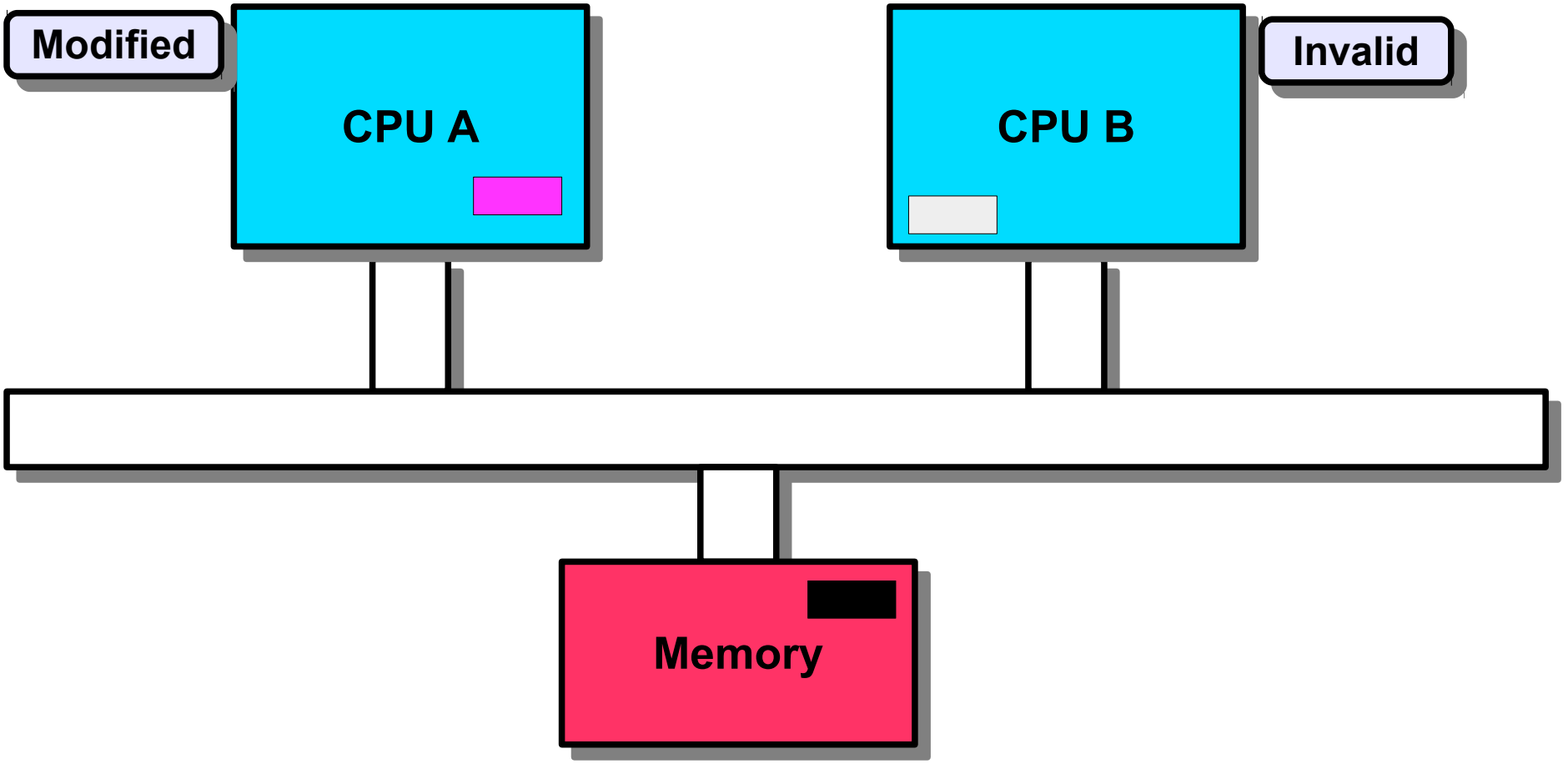
Write-back,

SMP - Write Invalidate

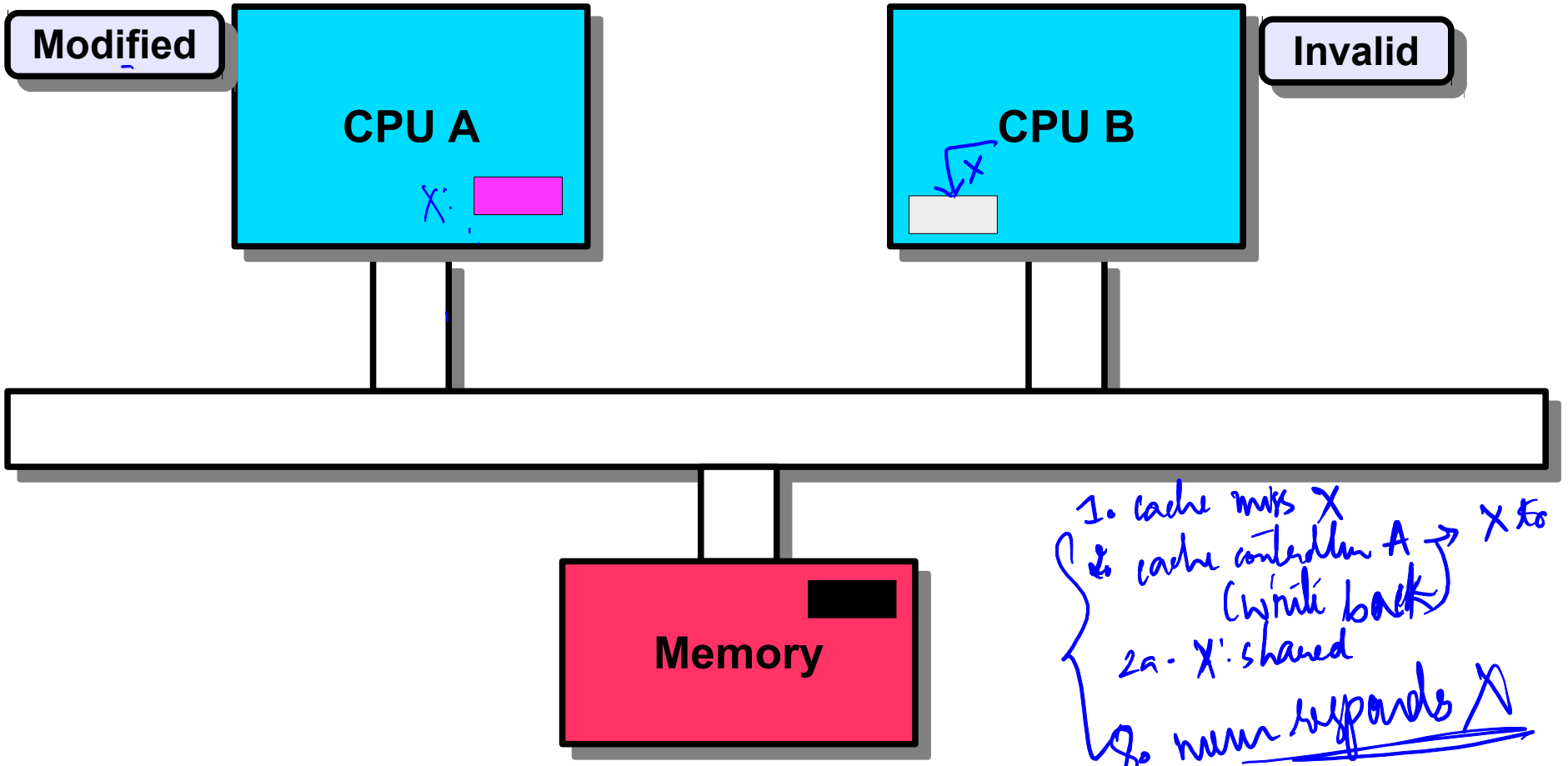


CPU A writes X

SMP - Write Invalidate

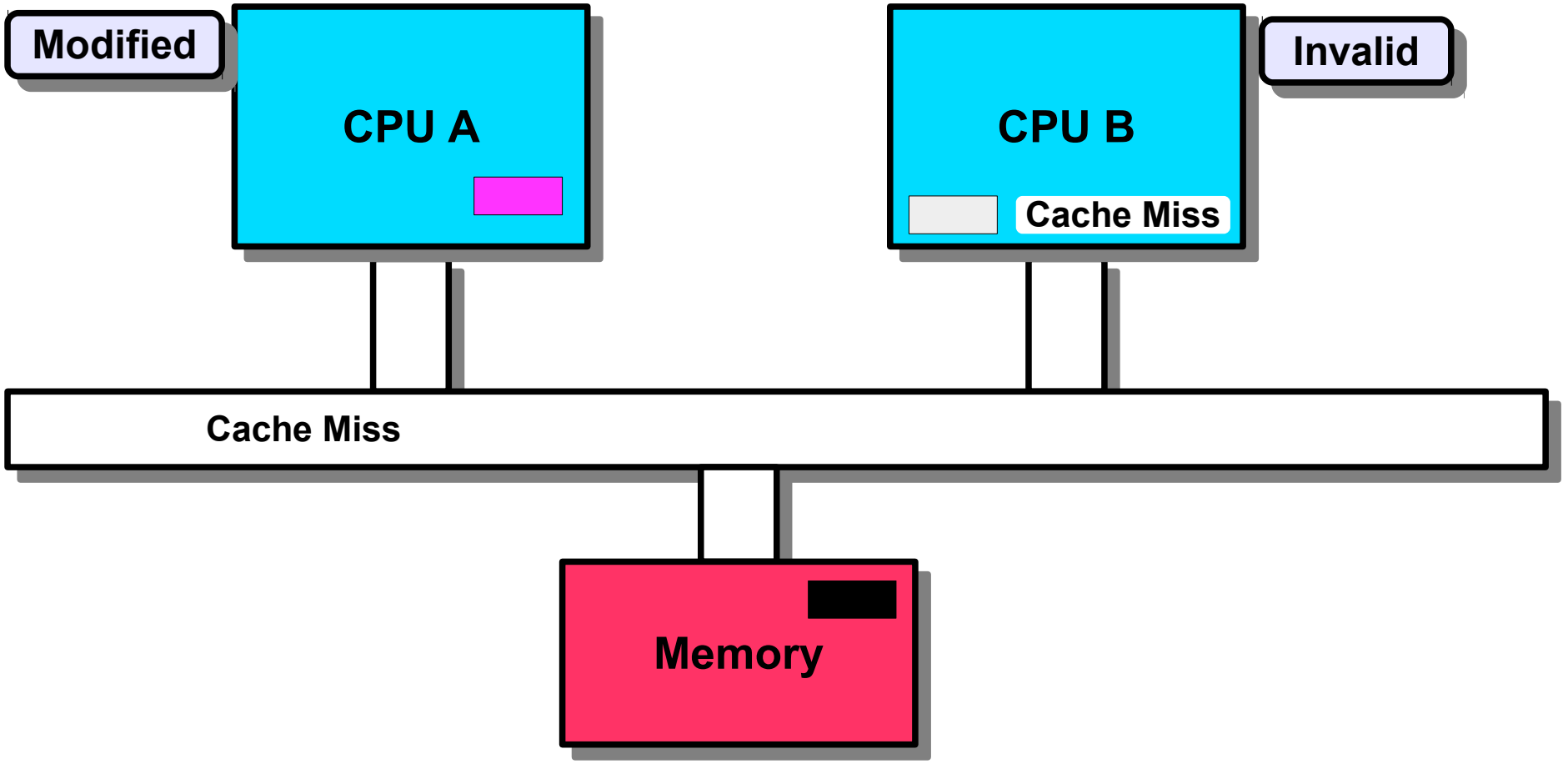


SMP - Write Invalidate



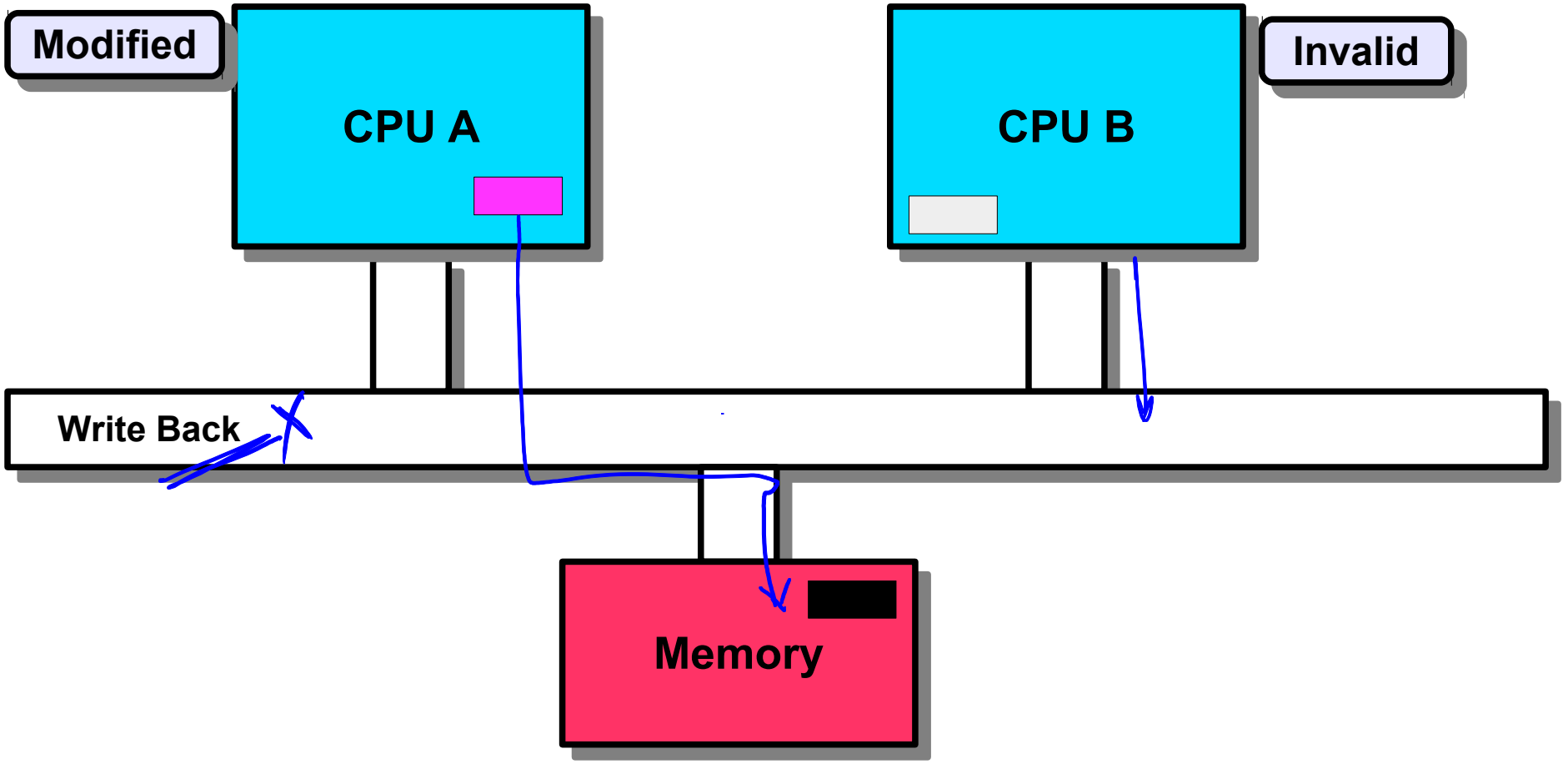
CPU B reads X

SMP - Write Invalidate



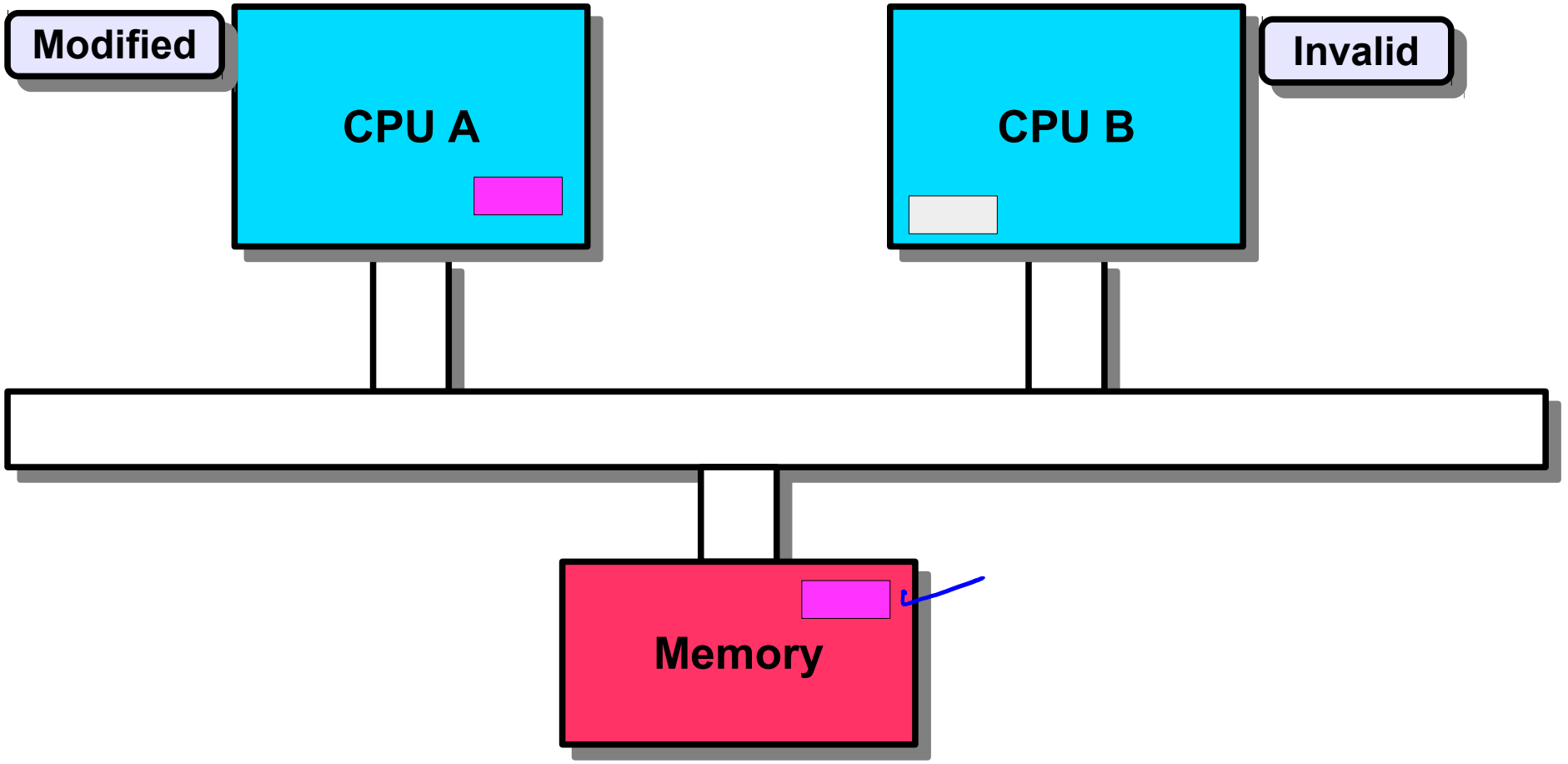
CPU B reads X

SMP - Write Invalidate



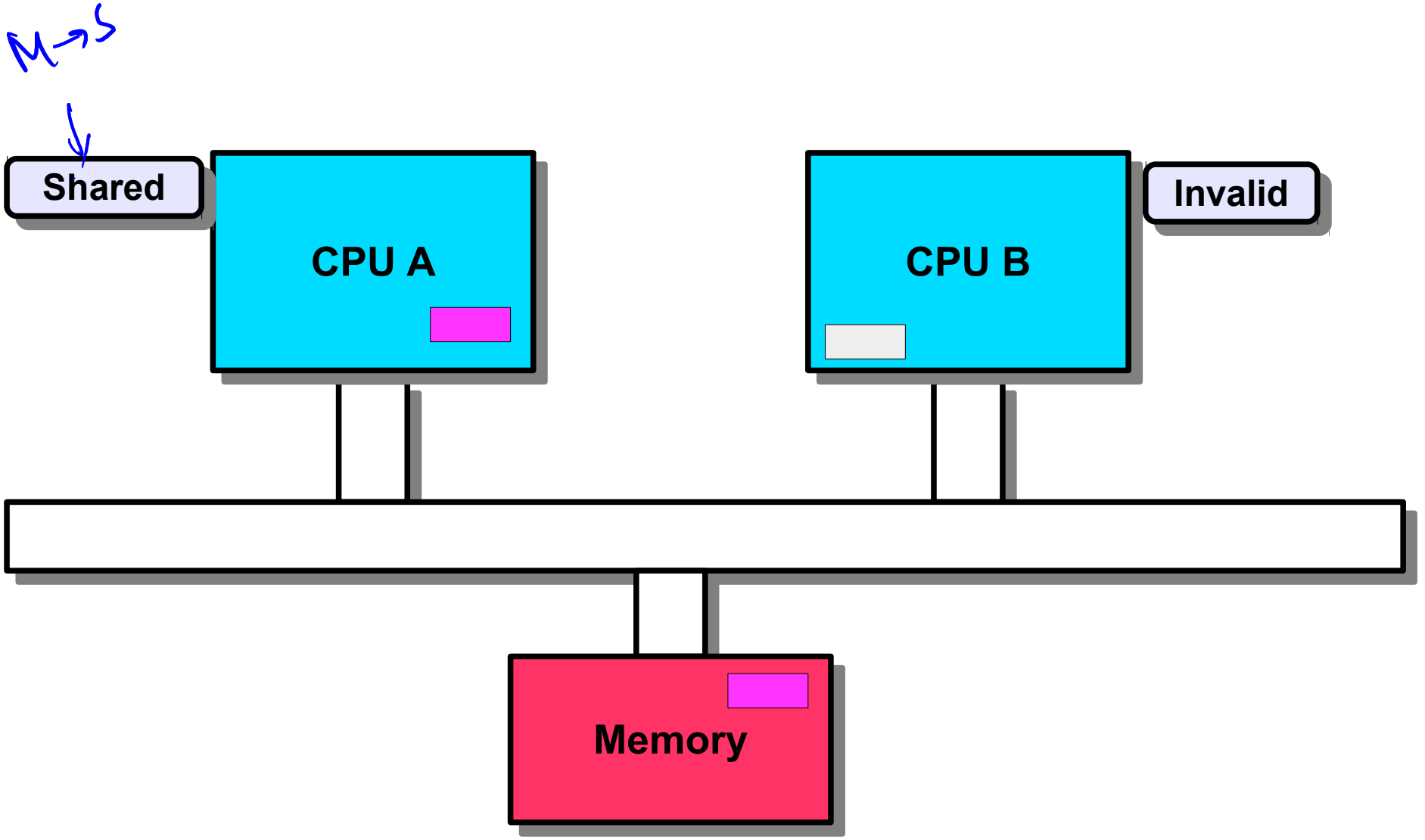
CPU B reads X

SMP - Write Invalidate



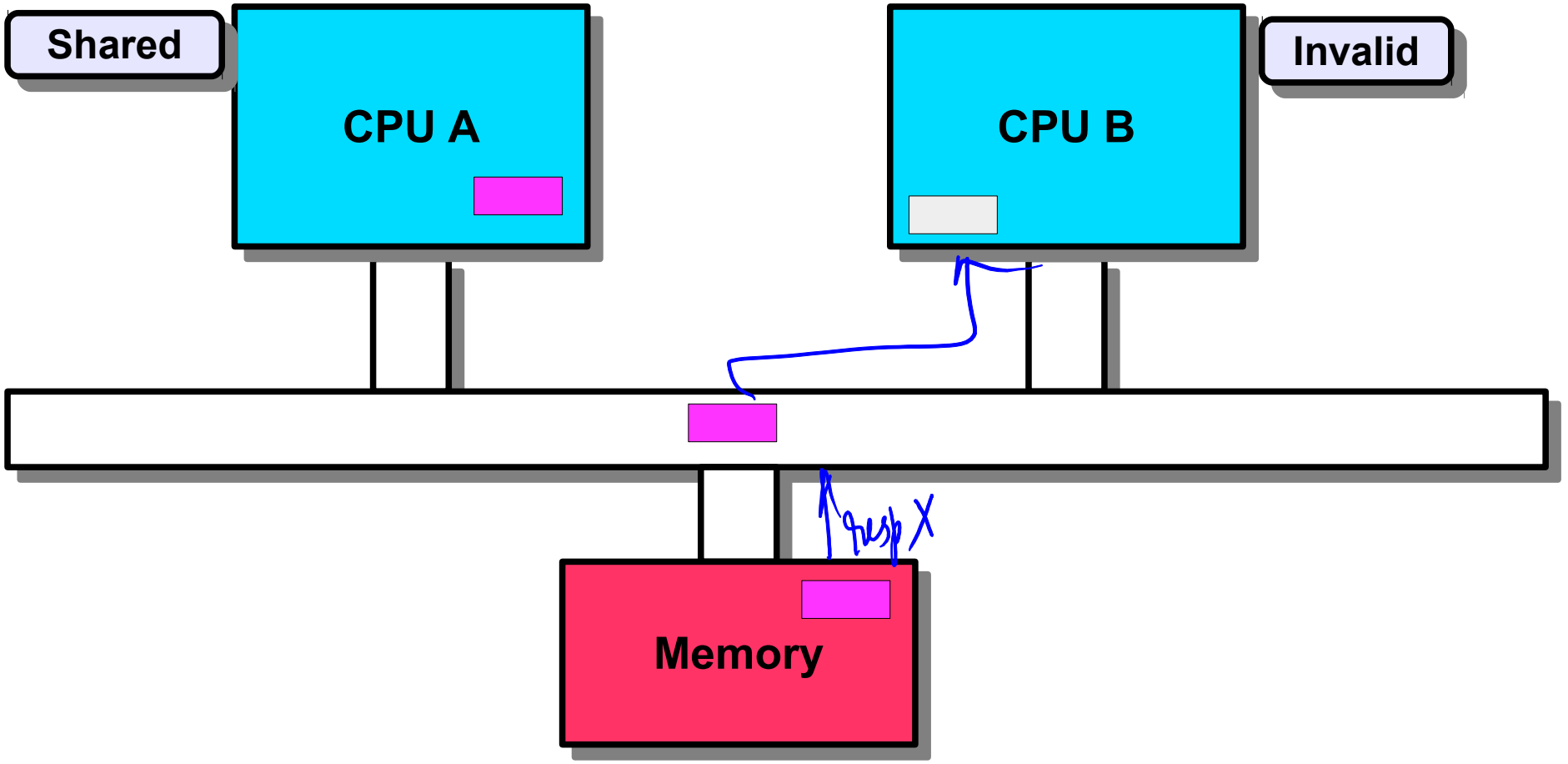
CPU B reads X

SMP - Write Invalidate



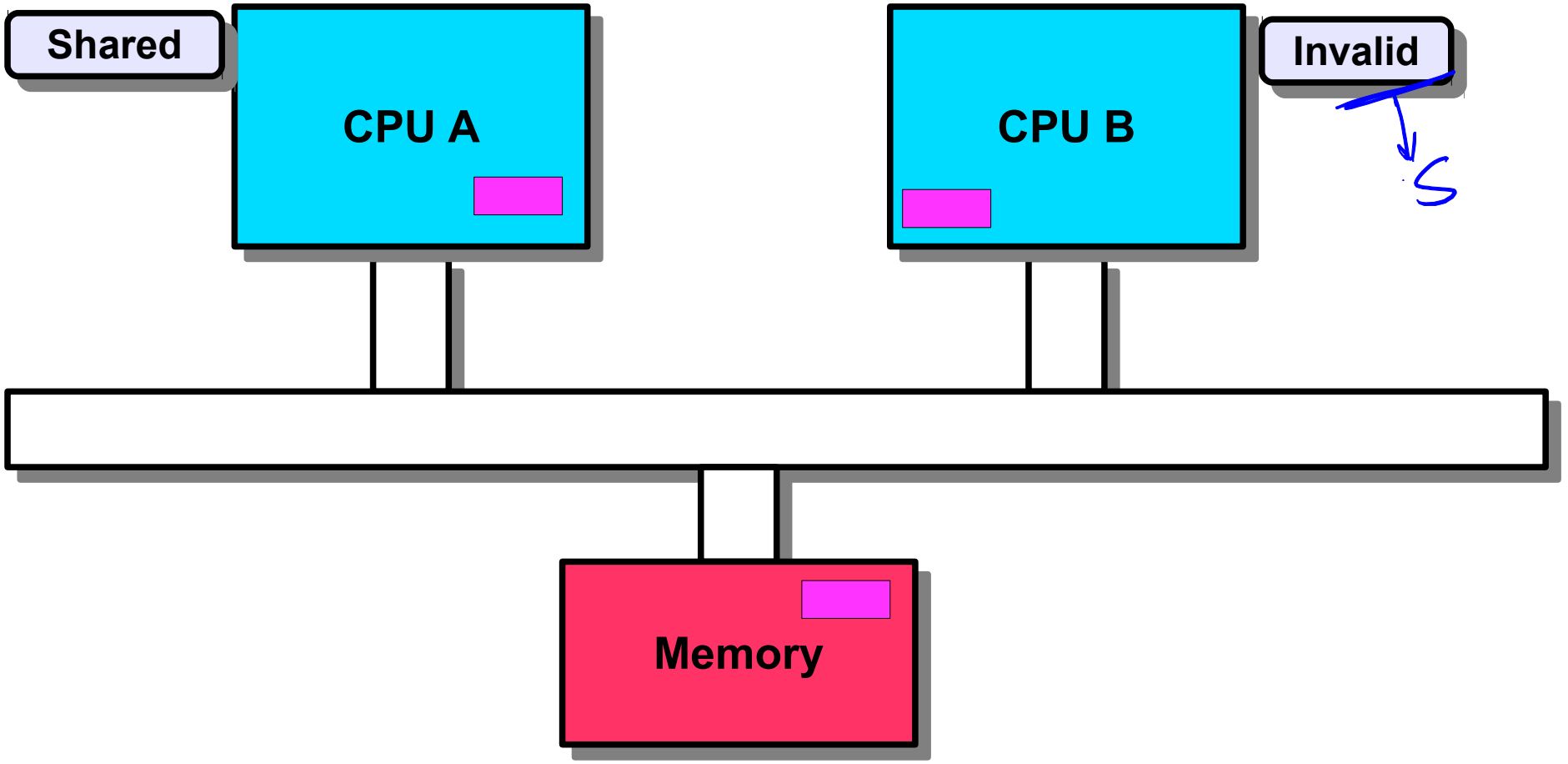
CPU B reads X

SMP - Write Invalidate



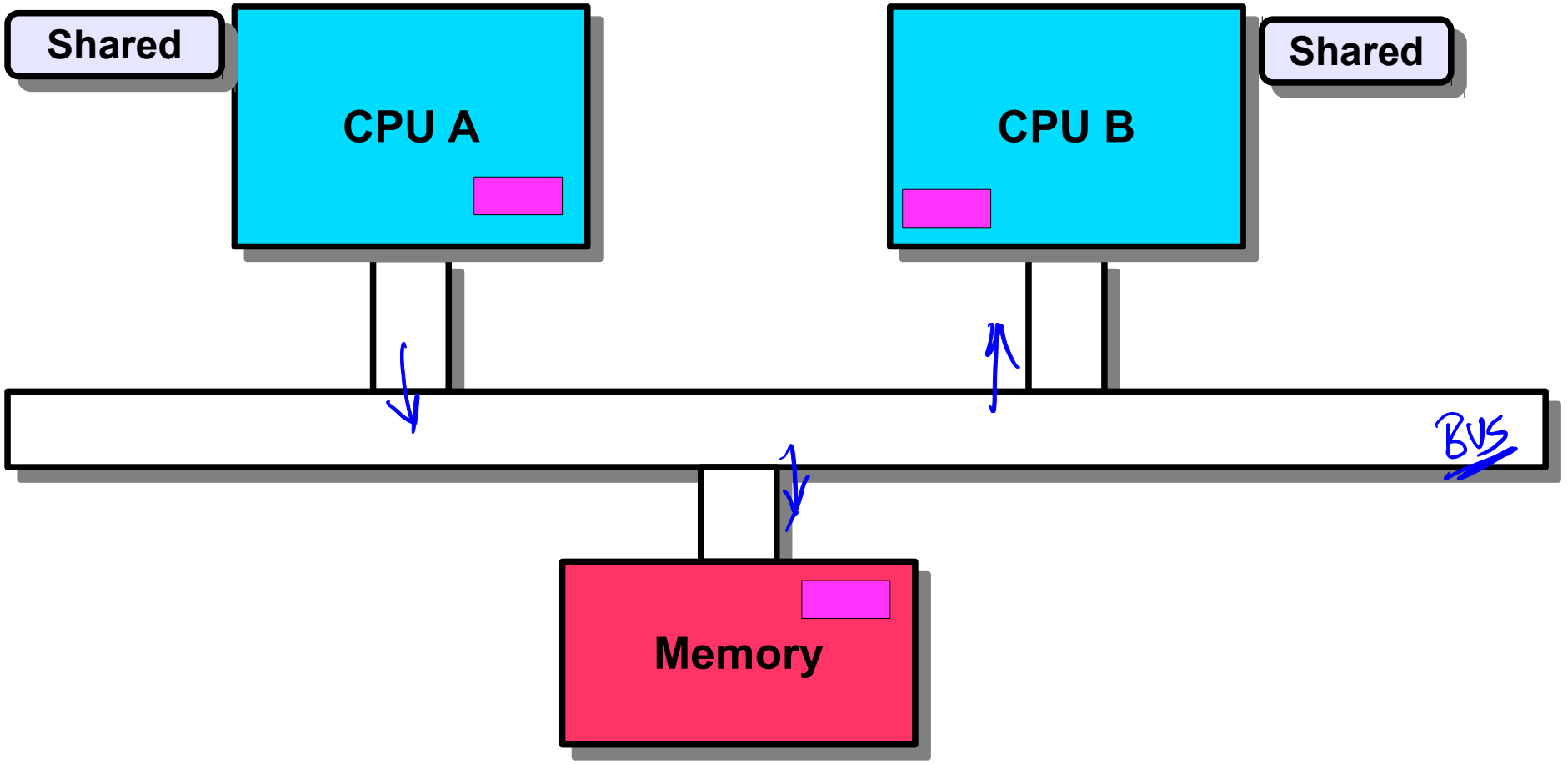
CPU B reads X

SMP - Write Invalidate



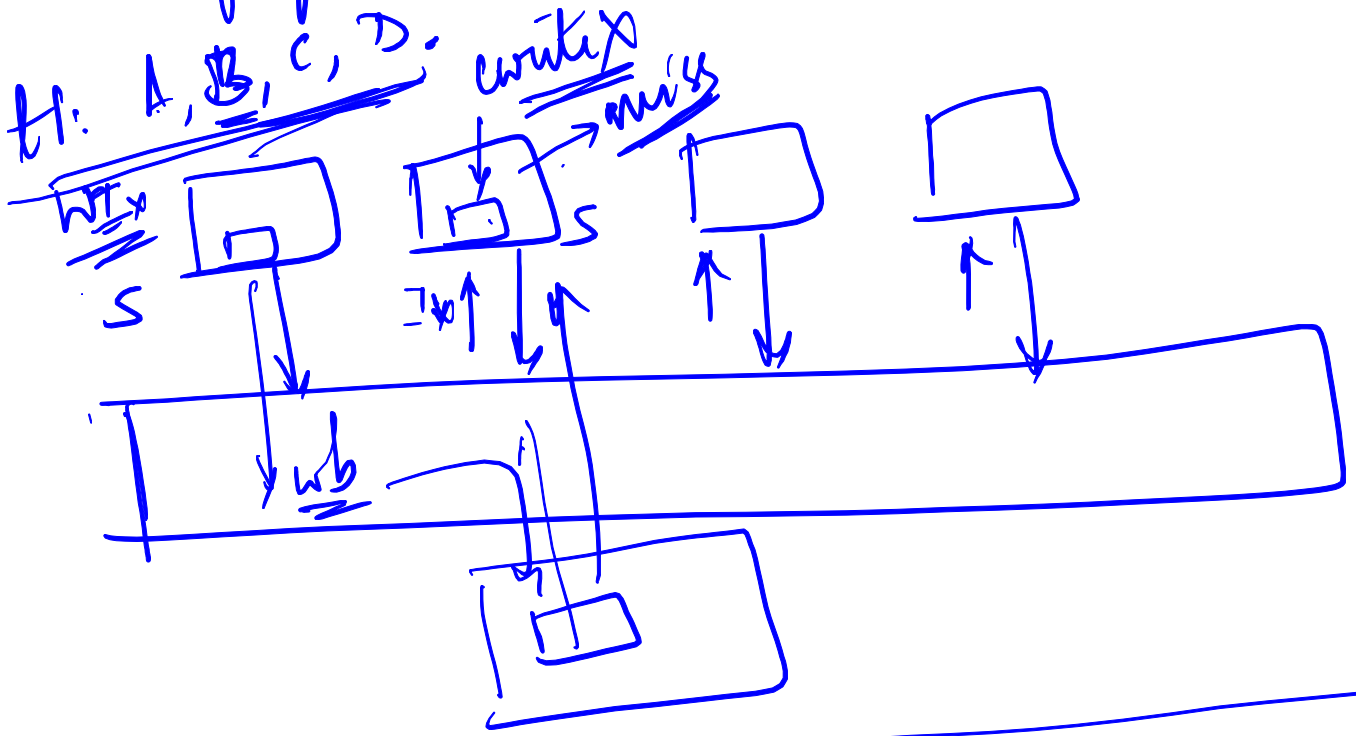
CPU B reads X

SMP - Write Invalidate

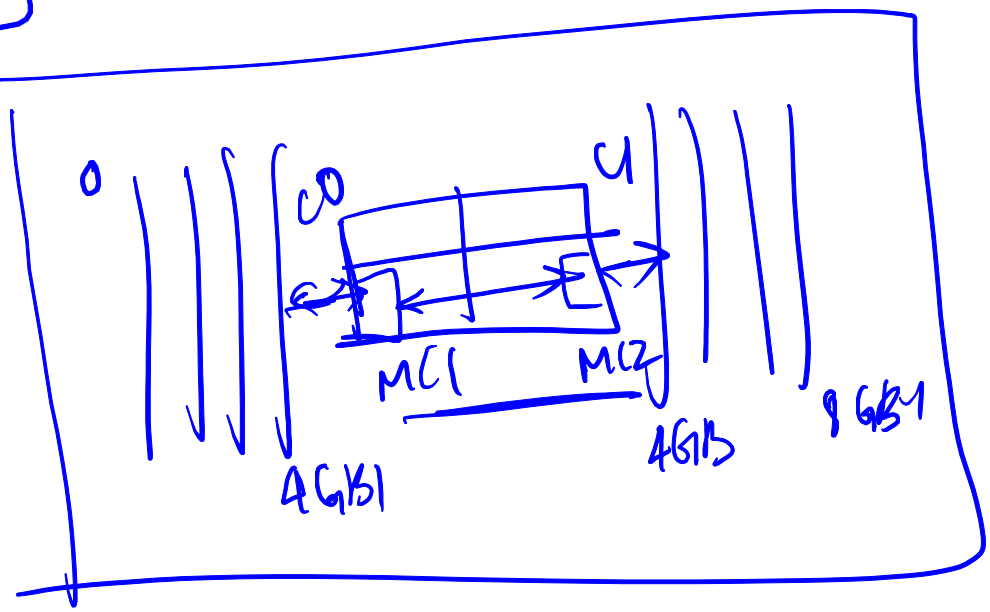


CPU B reads X

1. 4 modified flags?



* lottery
* priority
* round robin



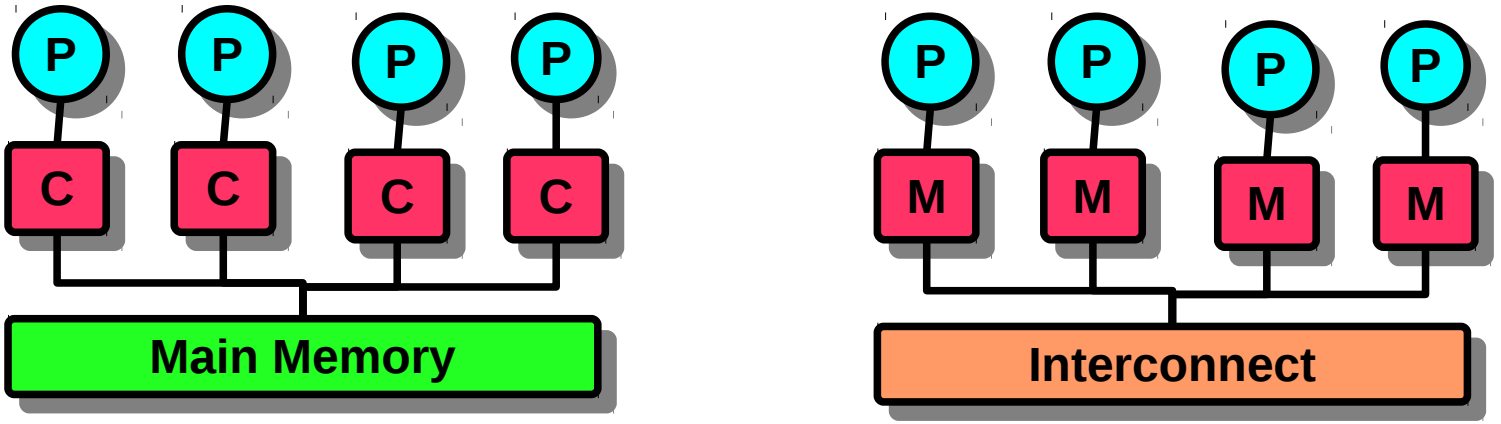
Slides Contents

- Rajeev Balasubramonian, CS6810, University of Utah.
-

Extra

Shared Memory vs. Message Passing

- **Shared Memory Machine**: processors share the same physical address space
  - Implicit Communication, Hardware controlled cache coherence
- **Message Passing Machine**
  - Explicit communication – programmed
  - No cache coherence (simpler hardware)
  - Message passing libraries: MPI



Cache Coherence

- Consistency
  - When should a written value be available to read
  - **Memory Consistency Models**
- Coherence
  - Which value to return on a read
- A memory system is coherent if:
  - Write Propagation
    - A write is visible after a sufficient time lapse
  - Write Serialization
    - All writes to a location are seen by every processor in the same order

Multiprocessor Cache Coherence

- A **read** by a processor P to a location X that **follows a write** by P to X, with no writes of X by another processor occurring between the write and the read by P, always **returns the value written** by P.
- A **read** by a processor to location X that **follows a write** by another processor to X returns the written value if the **read and write** are **sufficiently separated in time** and no other writes to X occur between the two accesses.
- **Writes** to the same location are **serialized**; that is, two writes to the same location by any two processors are seen in the same order by all processors.

Write Invalidate Coherence Protocol

Processor activity	Bus activity	Contents of CPU A's cache	Contents of CPU B's cache	Contents of memory location X
				0

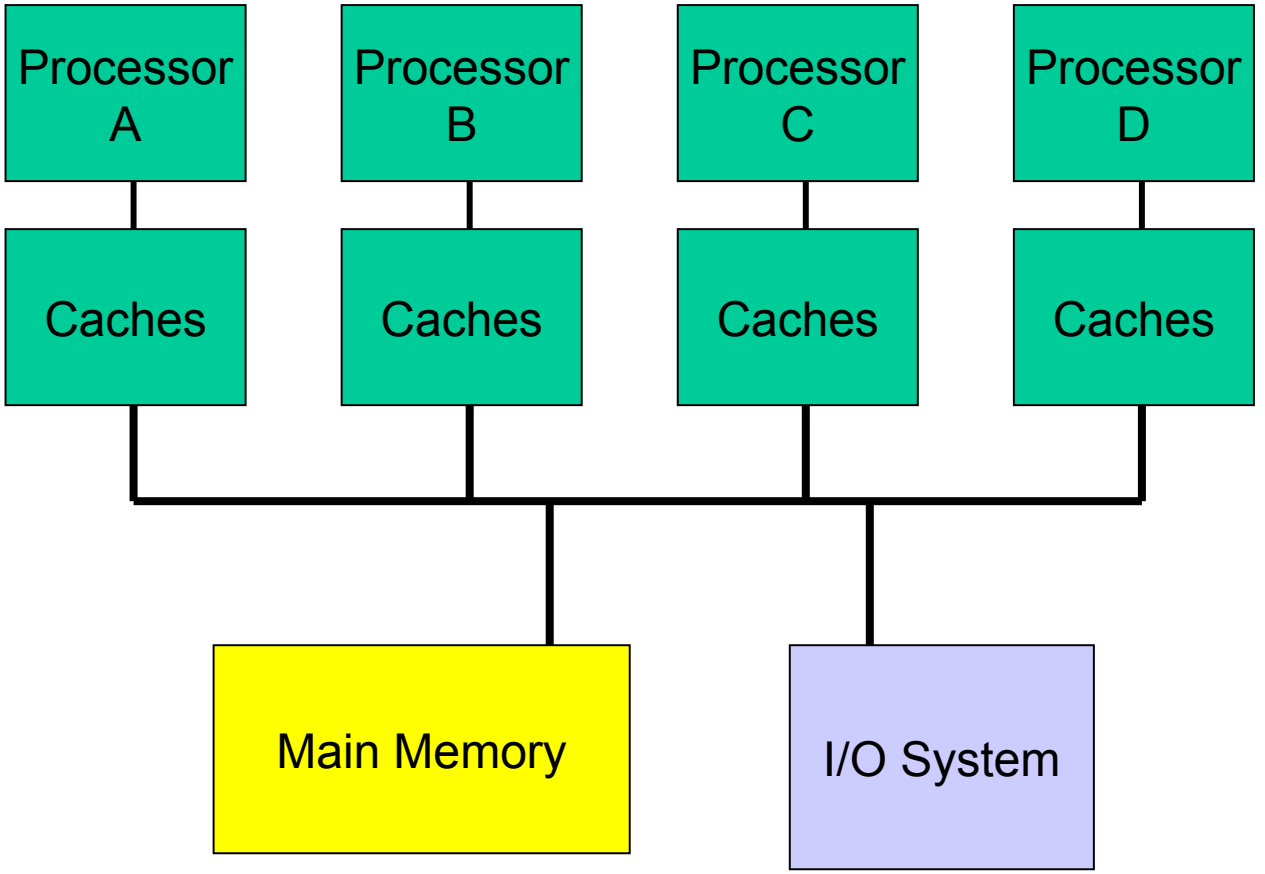
Writeback / Writethrough
Enforcing write serialization
- Bus Arbitration

Tag Contention, Duplication

SMP Cache Coherence

- MSI Protocol
- MESI Protocol
  - Exclusive state: No invalidate messages on writes.
  - Intel i7 uses MESIF
- MOESI Protocol
  - Owned state: Only valid copy in the system. Main memory copy is stale.
  - Owner supplies data on a miss.

SMP Example



A: Rd X
B: Rd X
C: Rd X
A: Wr X
A: Wr X
C: Wr X
B: Rd X
A: Rd X
A: Rd Y
B: Wr X
B: Rd Y
B: Wr X
B: Wr Y