Bus-Based Multiprocessors

Shared Memory Model:

\[ P_1 \rightarrow P_2 \rightarrow P_3 \rightarrow P_4 \rightarrow \ldots \]

Shared Memory Implementation:

\[ \text{CPU core} \rightarrow \text{Physical Memory (DRAMs)} \]
\[ \text{CPU core} \rightarrow \text{Memory Controller} \]
\[ \text{Bridge} \rightarrow \text{to I/O subsystem} \]
Cache Coherence Problem

(Assume writeback caches)

1. \( P_1 \) performs \text{ld} A
2. \( P_1 \) performs \text{st} A
   \( \rightarrow \) A is dirty in \( P_1 \)'s cache
3. \( P_2 \) performs \text{ld} A

\( \rightarrow P_2 \) loads stale copy of A from memory.

(Assume write through caches)

1. \( P_1 \) performs \text{ld} A
2. \( P_2 \) performs \text{ld} A
3. \( P_1 \) performs \text{st} A
4. \( P_2 \) performs \text{ld} A

\( \rightarrow P_2 \) loads stale copy of A from cache.
Memory Consistency Model

⇒ When sequential consistency is broken.

Sequentially consistent order:

- \( P_1 \) writes \( A = 1 \)
- \( P_1 \) writes \( X = 1 \)
- \( P_2 \) reads \( X = 1 \)
- \( P_2 \) reads \( A = 1 \)
- \( P_2 \) writes \( B = A = 1 \)

⇒ Caches reorder writes.
- \( P_1 \) order: \( A = 1, X = 1 \)
- \( P_2 \) order: \( X = 1, A = 1 \)

Assume write back caches:

1. \( P_1 \) writes \( A = 1 \)
2. \( P_1 \) writes \( X = 1 \)
3. \( X \) is written back to memory
4. \( P_2 \) reads \( X = 1 \)
5. \( P_2 \) reads \( A = 0 \)
6. \( P_3 \) writes \( B = A = 0 \)
Snoopy Cache Coherence Buses

Cache Bus

Main Memory

Snoopy Cache Coherence Hardware

state  tag  data

... ...

state  tag  data

Snoopy Cache Coherence Hardware
Bus-Based Write Invalidate

Diagram:
- Invalid
  - Write miss
  - Remote write miss
- Exclusive (dirty, only copy)
  - Read hit
  - Write hit
  - Remote read miss
- Shared (clean, multiple copies)
  - Remote read miss
  - Remote write hit/miss

Transitions:
- Processor-induced transitions
- Bus-induced transitions
Read Miss:
--place request on bus
--reply from memory --> exclusive
--reply from cache --> shared state

Read hit:
--no change

Bus read miss:
--Exclusive --> shared, provide copy
--Shared --> shared, provide copy
--Modified --> shared, provide copy, writeback to memory

Write Miss:
--place request on bus
--reply from memory --> Modified
--reply from cache --> Modified

Write hit:
--Exclusive --> modified
--Shared --> modified, place request on bus
--Modified --> modified

Bus write hit:
--Shared --> invalid
--Exclusive (impossible)
--Modified (impossible)