S. Di Girolamo [digirols@inf.ethz.ch]

Cache Coherence

Design of Parallel and High-Performance Computing – Recitation Session

Slides credits: Pavan Balaji, Torsten Hoefler
https://htor.inf.ethz.ch/teaching/mpi_tutorials/ppopp13/2013-02-24-ppopp-mpi-basic.pdf

Requirements of a cache coherent system

Cache coherence requirements

A memory system is coherent if it guarantees the following:

Write propagation: updates are eventually visible to all readers

Write serialization: writes to the same location must be observed in order

Everything else: memory model issues (later)

Snooping

Shared bus or (broadcast) network

Directory-based

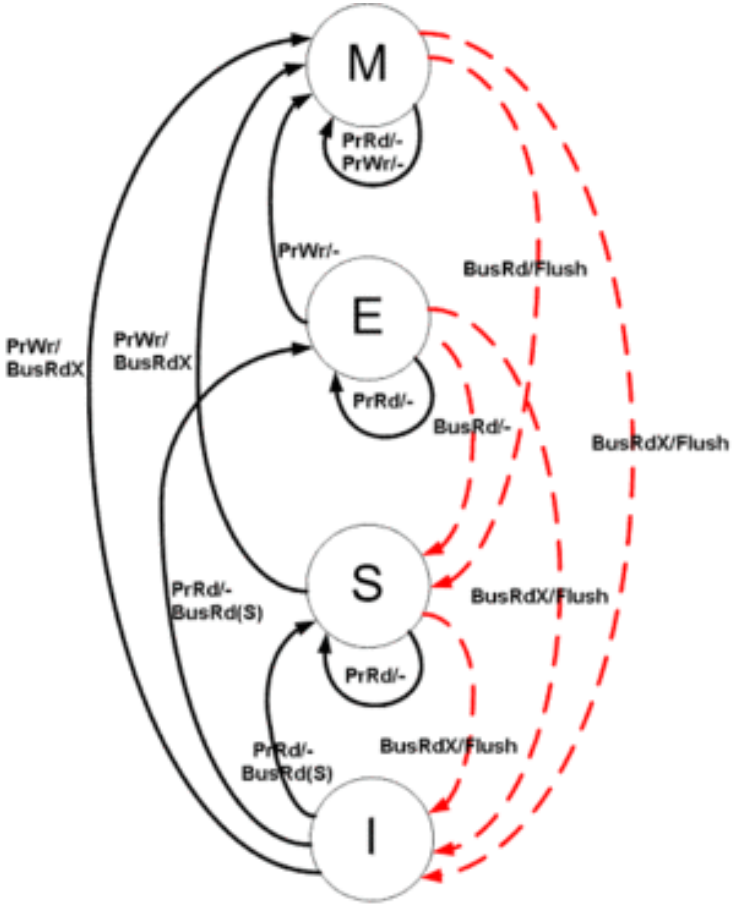
Record information necessary to maintain coherence: e.g., owner and state of a line etc.

MESI

- **Most common hardware implementation of discussed requirements**
  aka. “Illinois protocol”

Each line has one of the following states (in a cache):

- **Modified (M)**
  - Local copy has been modified, no copies in other caches
  - Memory is stale
- **Exclusive (E)**
  - No copies in other caches
  - Memory is up to date
- **Shared (S)**
  - Unmodified copies *may* exist in other caches
  - Memory is up to date
- **Invalid (I)**
  - Line is not in cache



Transitions in response to local reads

- **State is M**
  - No bus transaction
- **State is E**
  - No bus transaction
- **State is S**
  - No bus transaction
- **State is I**
  - Generate bus read request (BusRd)

    May force other cache operations (see later)
  - Other cache(s) signal “sharing” if they hold a copy
  - If shared was signaled, go to state S
  - Otherwise, go to state E
- **After update: return read value**

Transitions in response to local writes

- **State is M**
  - No bus transaction
- **State is E**
  - No bus transaction
  - Go to state M
- **State is S**
  - Line already local & clean
  - There may be other copies
  - Generate bus read request for upgrade to exclusive (BusRdX*)
  - Go to state M
- **State is I**
  - Generate bus read request for exclusive ownership (BusRdX)
  - Go to state M

Transitions in response to snooped BusRd

- **State is M**
  - Write cache line back to main memory
  - Signal “shared”
  - Go to state S (or E)
- **State is E**
  - Signal “shared”
  - Go to state S and signal “shared”
- **State is S**
  - Signal “shared”
- **State is I**
  - Ignore

Transitions in response to snooped BusRdX

- **State is M**
  - Write cache line back to memory
  - Discard line and go to I
- **State is E**
  - Discard line and go to I
- **State is S**
  - Discard line and go to I
- **State is I**
  - Ignore

- **BusRdX* is handled like BusRdX!**

Exercise

e) Assume a machine with 32bit addresses and 4 processors with directly-mapped L1 caches. Each cache is 4MiB in size, with 128B wide cache lines. MESI is used as cache-coherency mechanism. The memory is byte-addressable.

1. How many bits of the address are used as tag-, set-, and offset-bits? Assume the offset bits are the least significant ones, while the tag bits are the most significant. (4pt)

2. For the following sequence of instructions, list the cacheline index (CL) that they target, the new state of the cacheline, and the bus signal. Use the provided table. Assume that initially all the cachelines are invalid (I). Only fill fields which change their value. (8pt)

Inst.	Processor 1		Processor 2		Processor 3		Processor 4		Bus Signal
	CL	State	CL	State	CL	State	CL	State	
`P1: R(0x00100041)`									
`P3: R(0x00100159)`									
`P1: R(0x00100178)`									
`P4: W(0x0010014E)`									
`P2: R(0x00100186)`									
`P1: W(0x00100186)`									