UNIVERSITY OF DELAWARE

Design and exploration on novel Memory Models for Exascale Architectures

Pouya Fotouhi

Mentors

Prof. Guang R. Gao

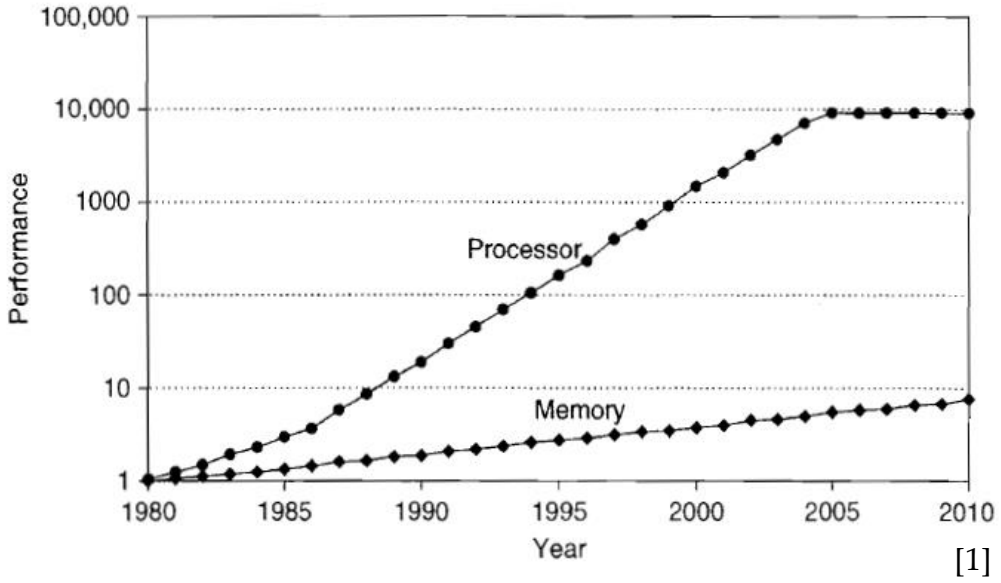
Prof. Stephane Zuckerman

Summer 2015

CAPSL

Introduction

- Technology trend in the past few decades:



[1]

- It got even worse in the past ten years using multi-core processors[1].

Introduction

- CPU clocks much faster than memory!
  - Memory wall.
  - Makes memory the bottleneck!

- How to overcome this issue?

Hint 1: Common features in memory access patterns.

  - Caching: Taking the advantage of both *temporal* and *spatial* locality.

Hint 2: The smaller the memory, the faster resolving the address. It means faster access!

  - Using memory hierarchy.

Locality

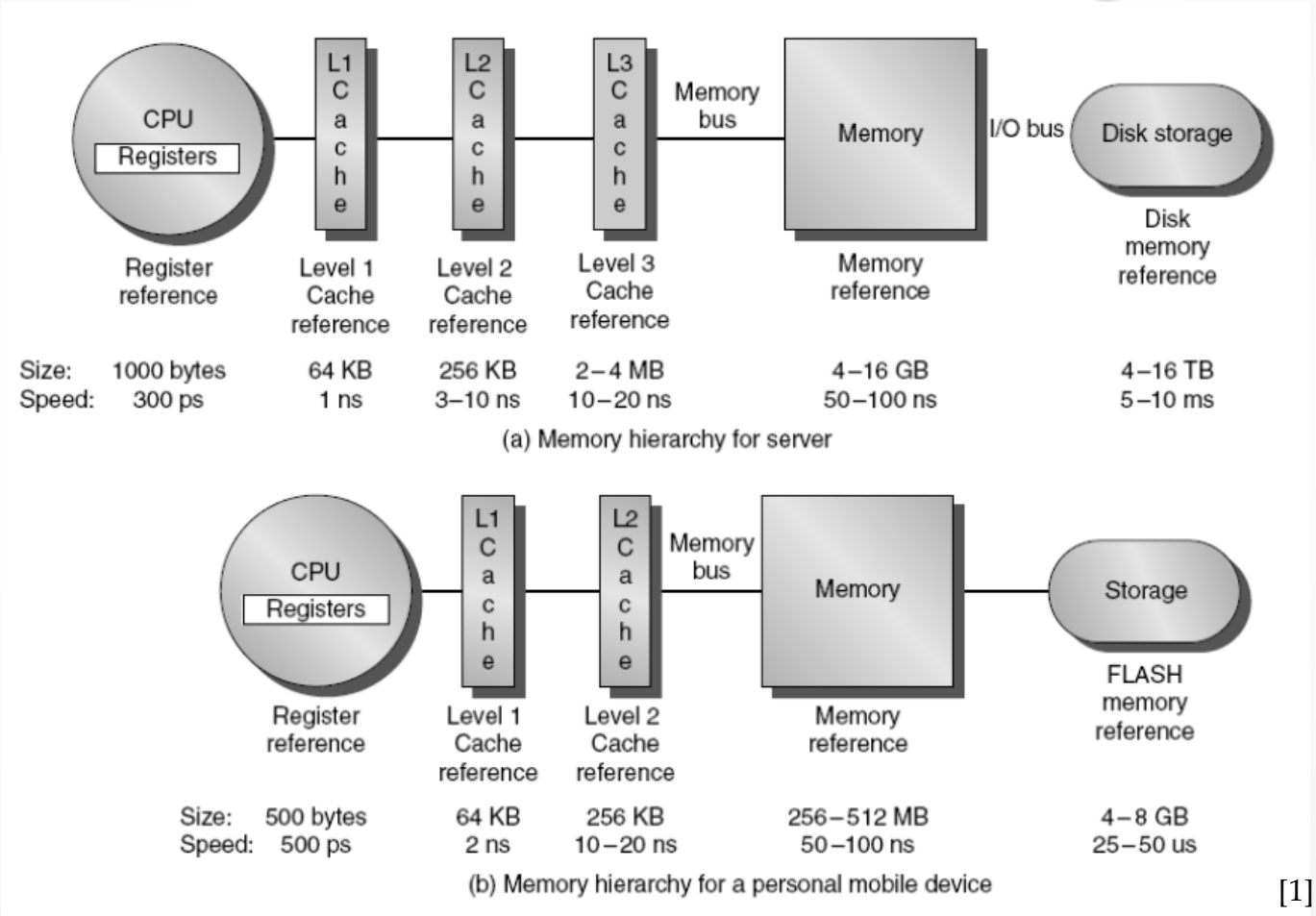
- **Temporal Locality:** Items accessed recently are likely to be accessed again soon.

  e.g. loop instructions.

- **Spatial Locality:** Items around those accessed recently are likely to be accessed soon.

  e.g. accessing an array. [2]

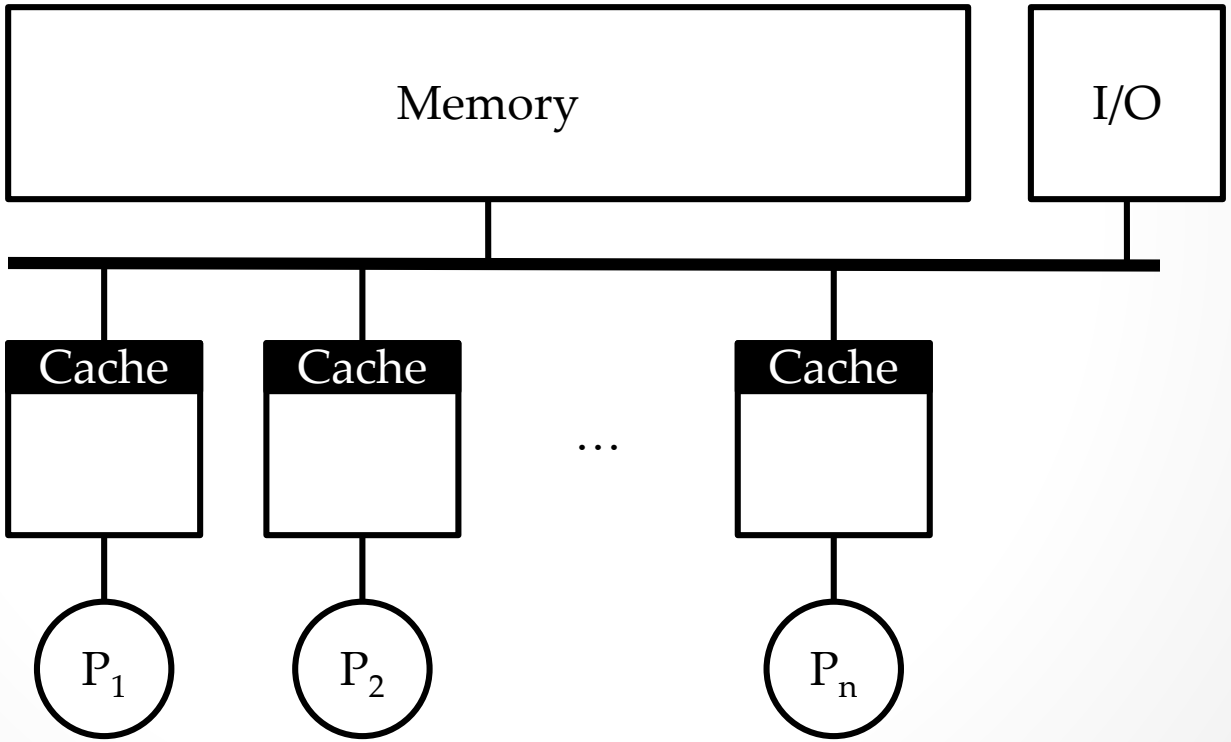
Cache Hierarchy



[1]

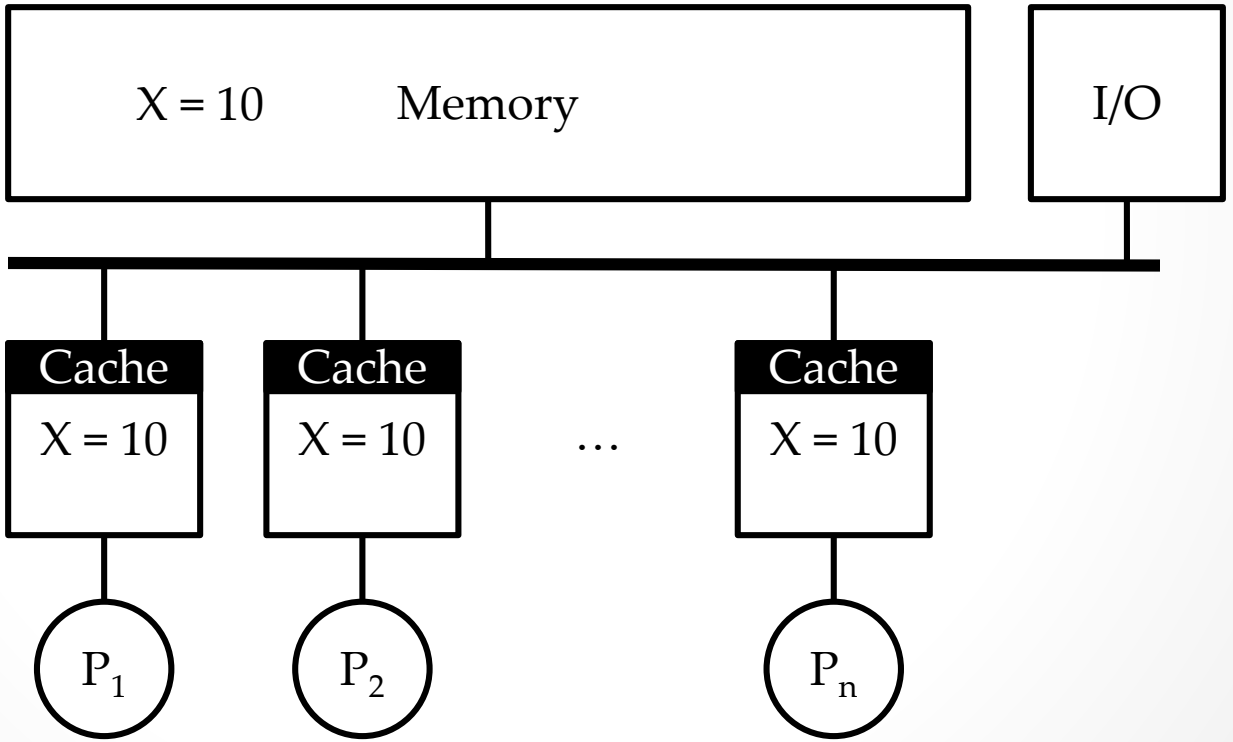
Cache Hierarchy

- But we live in the era of multi-processors. So we have:



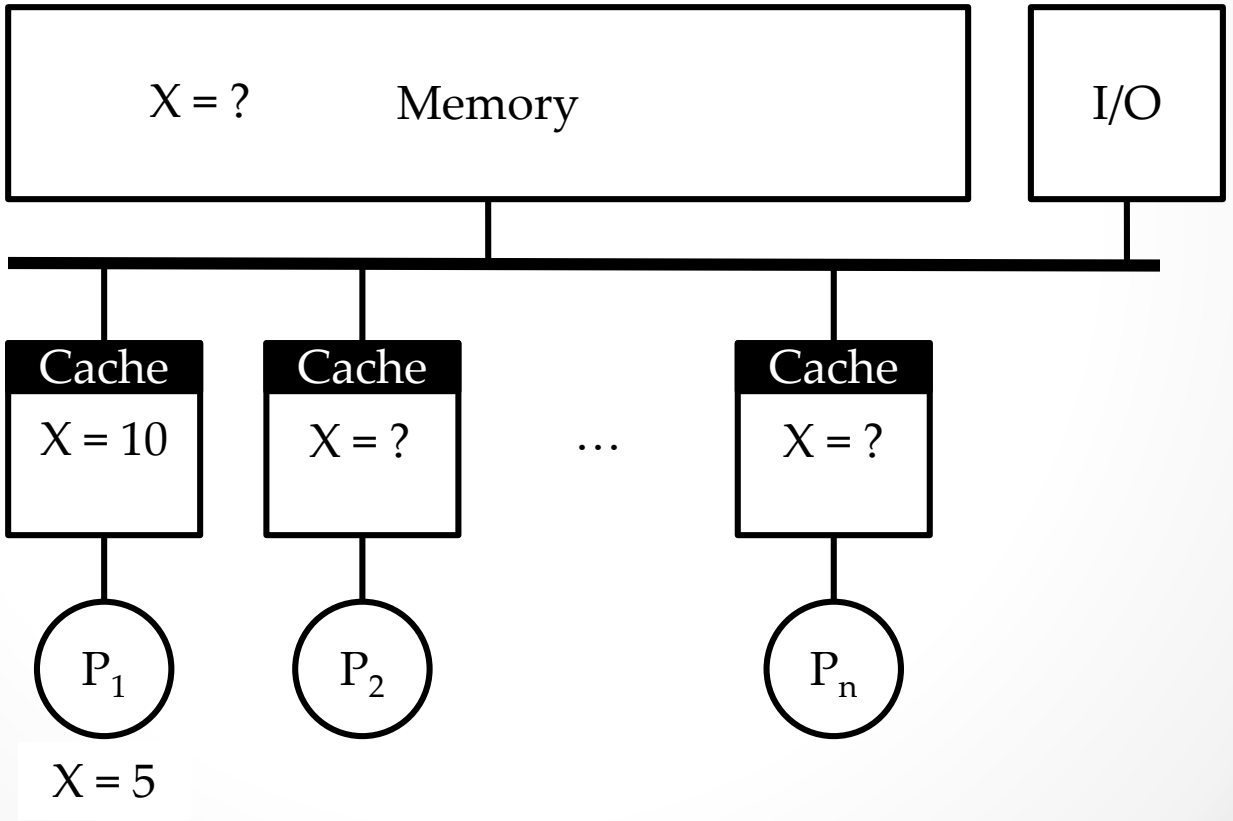
Cache Coherency

- Let's assume we have the variable X with initial value of 10 at all caches:



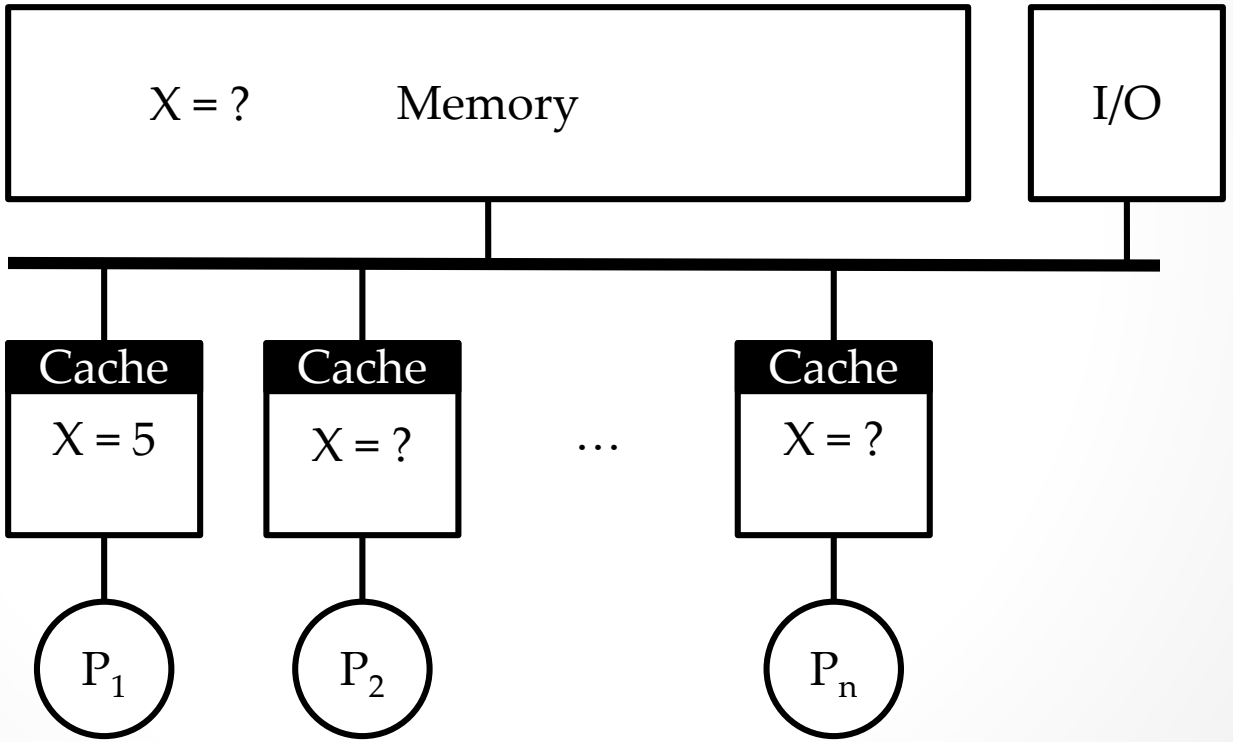
Cache Coherency

- Now P1 writes X with 5.
  - Should we update memory? What about other copies in caches?



Cache Coherency

- And now P_2 tries to read X.
  - What value it supposed to get? Where that value should come from?

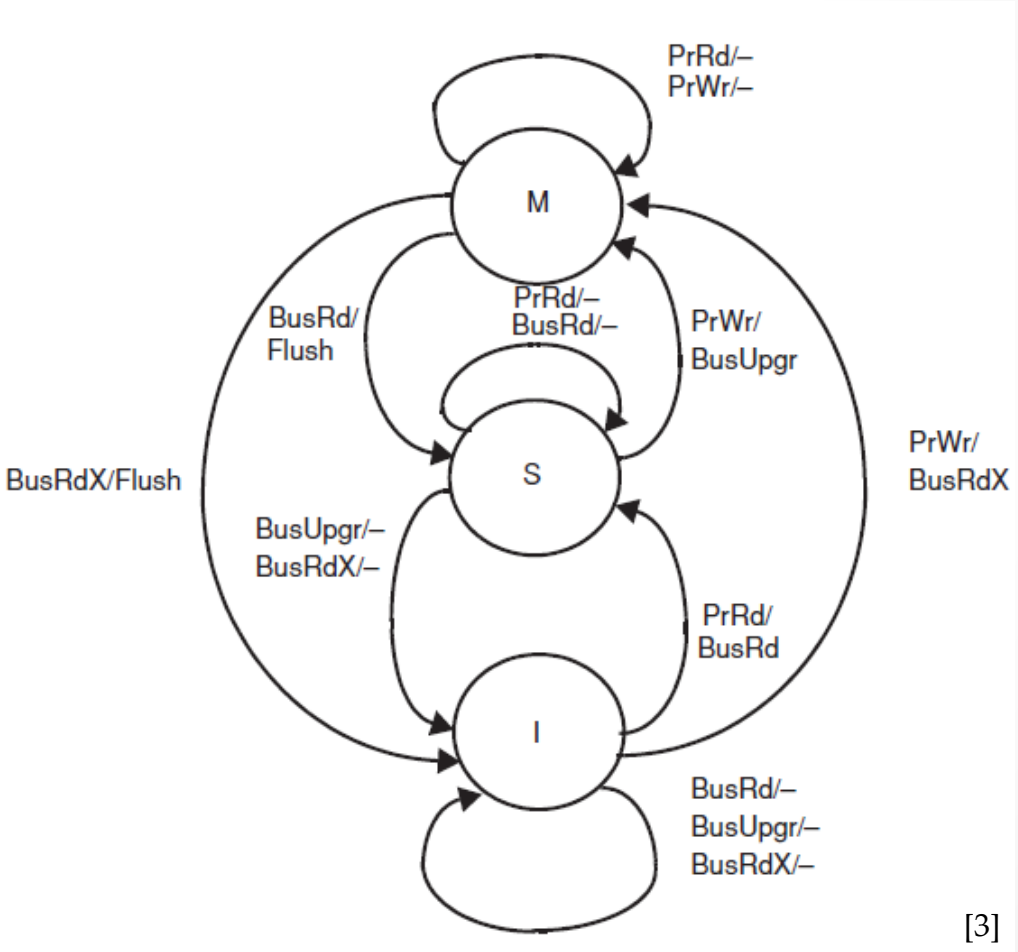


Cache Coherency Protocols

- A protocol to address:
  - **Upon a write:** How to inform other processors?
  - **Upon a read miss:** Where is the data we are looking for? [2]
- Examples:
  - MSI
  - MESI
  - MOESI

MSI

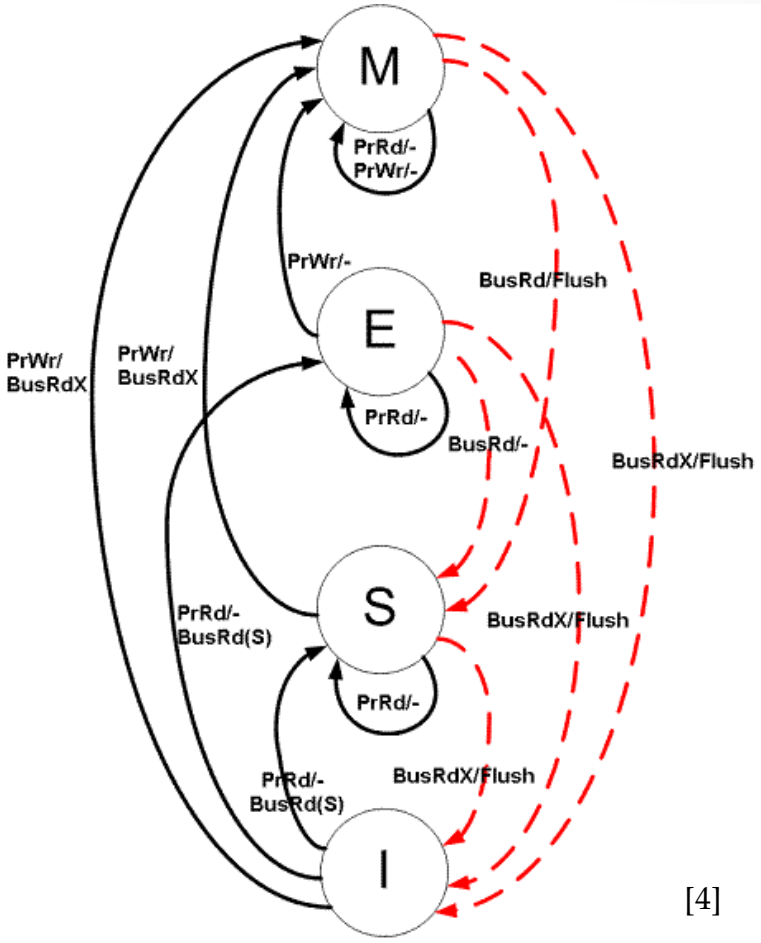
- **M**odified:
  - One valid copy.
  - Memory is stale.
- **S**hared:
  - One copy or more.
  - Memory is clean.
- **I**nvalid



[3]

MESI

- **M**odified:
  - One valid copy.
  - Memory is stale.
- **E**xclusive:
  - One valid copy.
  - Memory is clean.
- **S**hared:
  - One copy or more.
  - Memory is clean.
- **I**nvalid



Memory Consistency Model

- A Consistency Model is a set of rules between hardware and software on memory operations to illustrate and ensure the correct execution of programs.
- Memory Consistency Models impose ordering restrictions on memory accesses.
- It can greatly affect the performance of a system specially when memory latency increases. [5]
- Still need an intuitive definition?

Memory Consistency Model

- What happens when at least two concurrent memory operations arrive at the same memory location x?
  - What happens when a data-race (*i.e.* at least one of the two memory operations is a write) occurs at some memory location x?
- Memory Consistency Models try to answer that question. [6]
- Examples:
  - SC
  - PC
  - RC
  - LC

Sequential Consistency

- A very strong model.
- Considered very intuitive.
- Provides strong guarantees. (everyone on the same page)

- "the result of any execution is the same as if the operations of all the processors were executed in some sequential order, and the operations of each individual processor appear in this sequence in the order specified by its program" [7]

- But extremely expensive to implement.
- Precludes many optimizations.

Location Consistency

- More relaxed compared to SC.
- Does NOT (necessarily) maintain coherency.
- Ordering constrains are captured at location level using a partial order for each location called Partially Ordered Multiset (*pomset*).
- Reads return a value from the *value set* based on the pomset of the location.
- Authors proved four features for LC:
  - Weakness (the model is weaker than RC)
  - Equivalence (it is equivalent to RC for programs with no data race)
  - Monotonicity (same value set is valid for a *more parallel* version of the program)
  - Nonintrusive (reads would not a affect the value set)

Project Plan

- Implement a simulation framework to model Virtual Processors (VPs) with Caches.

Project Plan

- Implement a simulation framework to model Virtual Processors (VPs) with Caches.

- To evaluate a variety of currently existing cache protocols
  - *i.e.,* cache coherence: MSI, MESI, MOSI, MOESI; cache-based Location Consistency.

Scalability **Performance drops!**

Project Plan

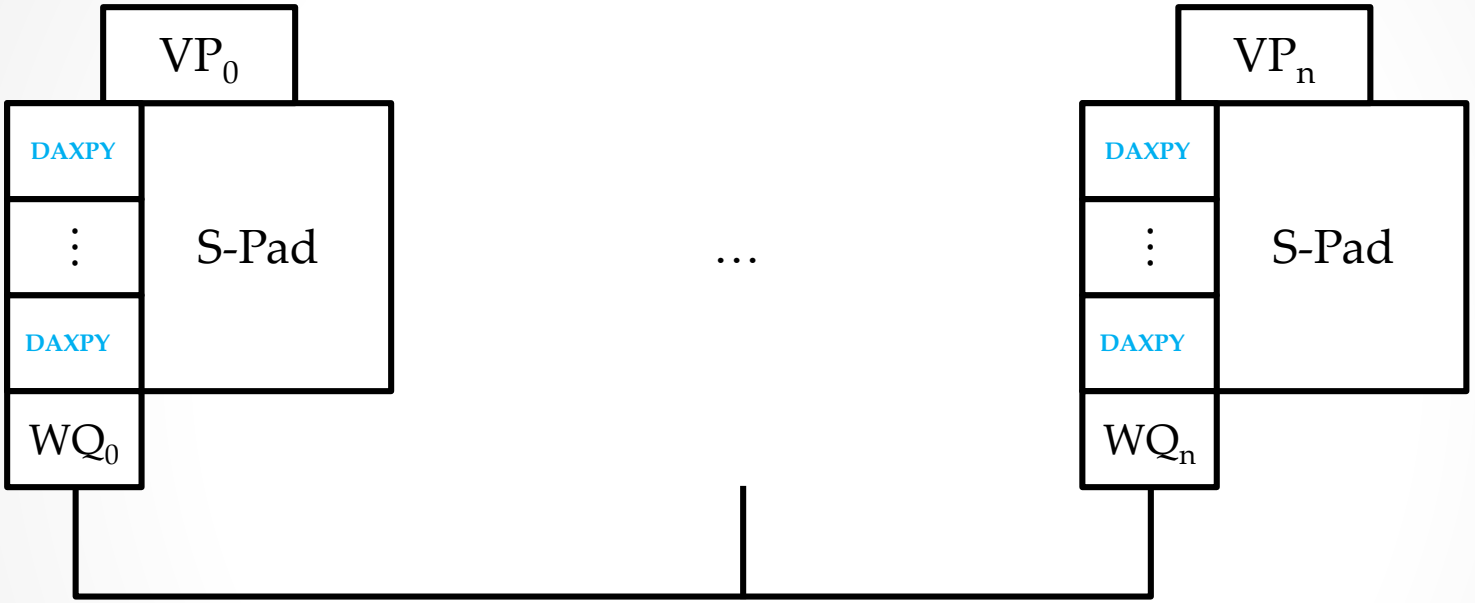
- Implement a simulation framework to model Virtual Processors (VPs) with Caches.

- To evaluate a variety of currently existing cache protocols
  - *i.e.,* cache coherence: MSI, MESI, MOSI, MOESI; cache-based Location Consistency.

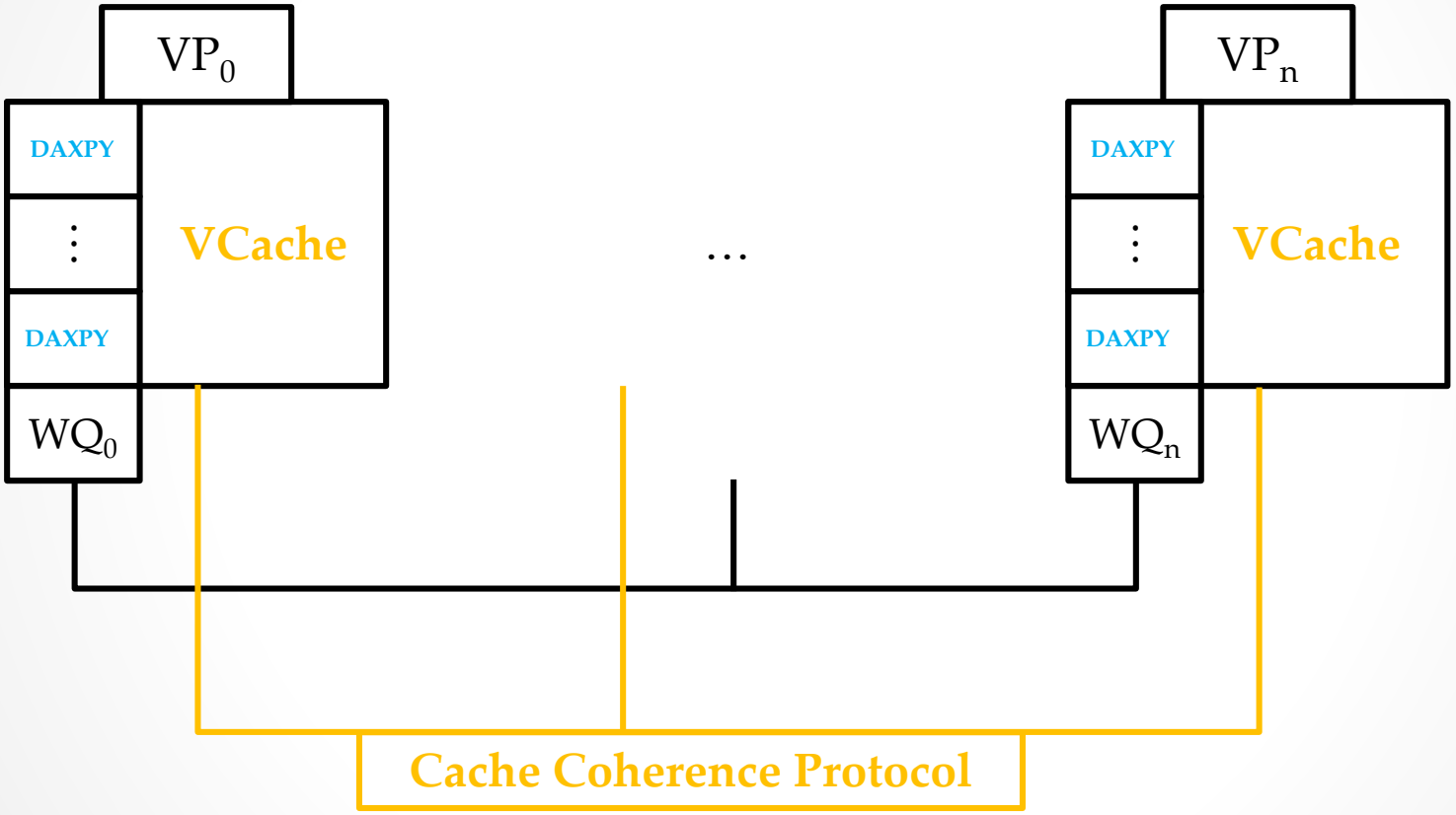
Scalability

- By measuring the number of:
  - Control messages.
  - And cache line copies transferred between virtual processors.

Current Progress



Future Work


VP0
DAXPY
DAXPY
WQ0
VCache
…
VPn
DAXPY
DAXPY
WQn
VCache
Cache Coherence Protocol

Refrences

1. John L. Hennessy and David A. Patterson. 2011. Computer Architecture, Fifth Edition: A Quantitative Approach (5th ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
2. Professor Chengmo Yang's Slides for CPEG652. University of Delaware. Fall 2013.
3. Michel Dubois, Murali Annavaram, and Per Stenstrm. 2012. Parallel Computer Organization and Design. Cambridge University Press, New York, NY, USA.
4. en.wikipedia.org/wiki/MESI_protocol
5. David Mosberger. 1993. Memory consistency models. SIGOPS Oper. Syst. Rev. 27, 1 (January 1993), 18-26. DOI=10.1145/160551.160553 http://doi.acm.org/10.1145/160551.160553
6. Professor Stephane Zuckerman Slides on Memory Consistency Models. University of Delaware. CAPSL. Fall 2011.
7. L. Lamport. 1979. How to Make a Multiprocessor Computer That Correctly Executes Multiprocess Programs. IEEE Trans. Comput. 28, 9 (September 1979), 690-691. DOI=10.1109/TC.1979.1675439 http://dx.doi.org/10.1109/TC.1979.1675439
8. Guang R. Gao and Vivek Sarkar. 2000. Location Consistency-A New Memory Model and Cache Consistency Protocol. IEEE Trans. Comput. 49, 8 (August 2000), 798-813. DOI=10.1109/12.868026 http://dx.doi.org/10.1109/12.868026

Questions?

Thank You!