Explorations in Instruction Level Parallelism

student **David Morano**
advisors **Professor David Kaeli**

Northeastern University
Computer Architecture Research

NUCAR talk 03/09/12

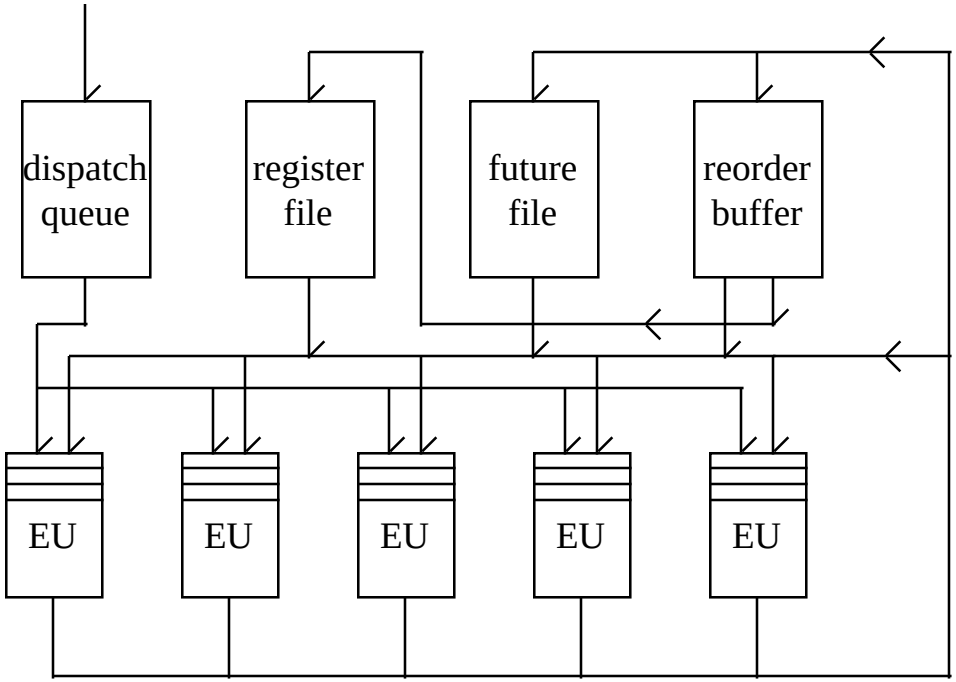
Outline

- **introduction & prior work**
- **existing microarchitectures**
- **active station**
  - active station state
  - operands
  - operand snooping
- **proposed microarchitecture**
  - comparison to Levo microarchitecture
- **research methodology**

introduction

- **existing microarchitectures have failed to extract high ILP !**
- **the Levo microarchitecture has demonstrated promise**
- **new work based on Levo**
  - we draw design and microarchitecture concepts from Levo
  - the Active Station concept appears to be central
- **goal is to explore modifying existing microarchitectures for higher ILP extraction**
- **we want to explore OoO execution with**
  - breaking control dependencies
  - breaking value depedencies (Levo last-value style)
  - and using time-tags as the dependency enforcement mechanism

conventional microarchitecture

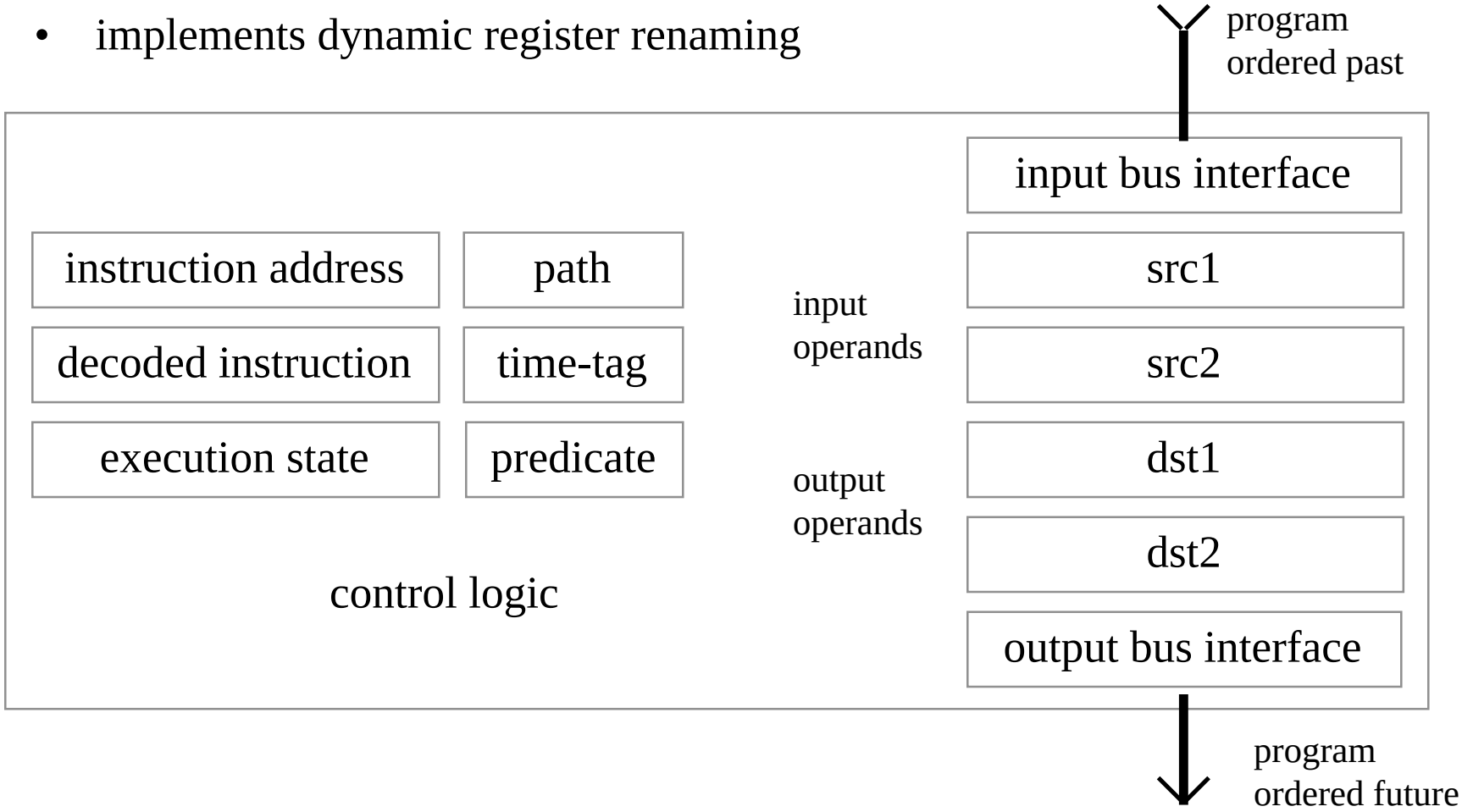


variations are numerous !

- dispatch stack
- reservation stations grouped and called an "instrustion window"
- physical register renaming with direct update to the register file

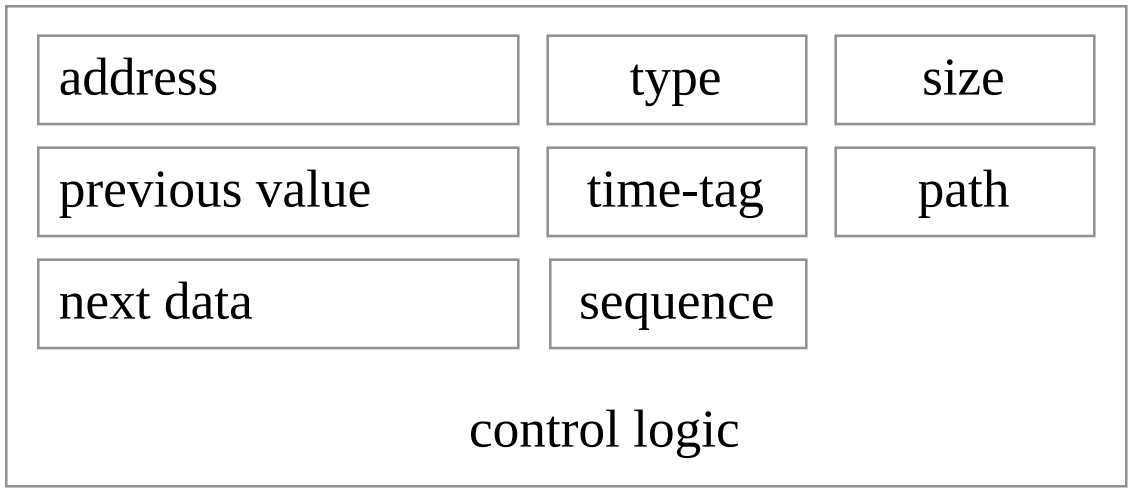
active station

- similar to Tomasulo's reservation station
- implements dynamic register renaming



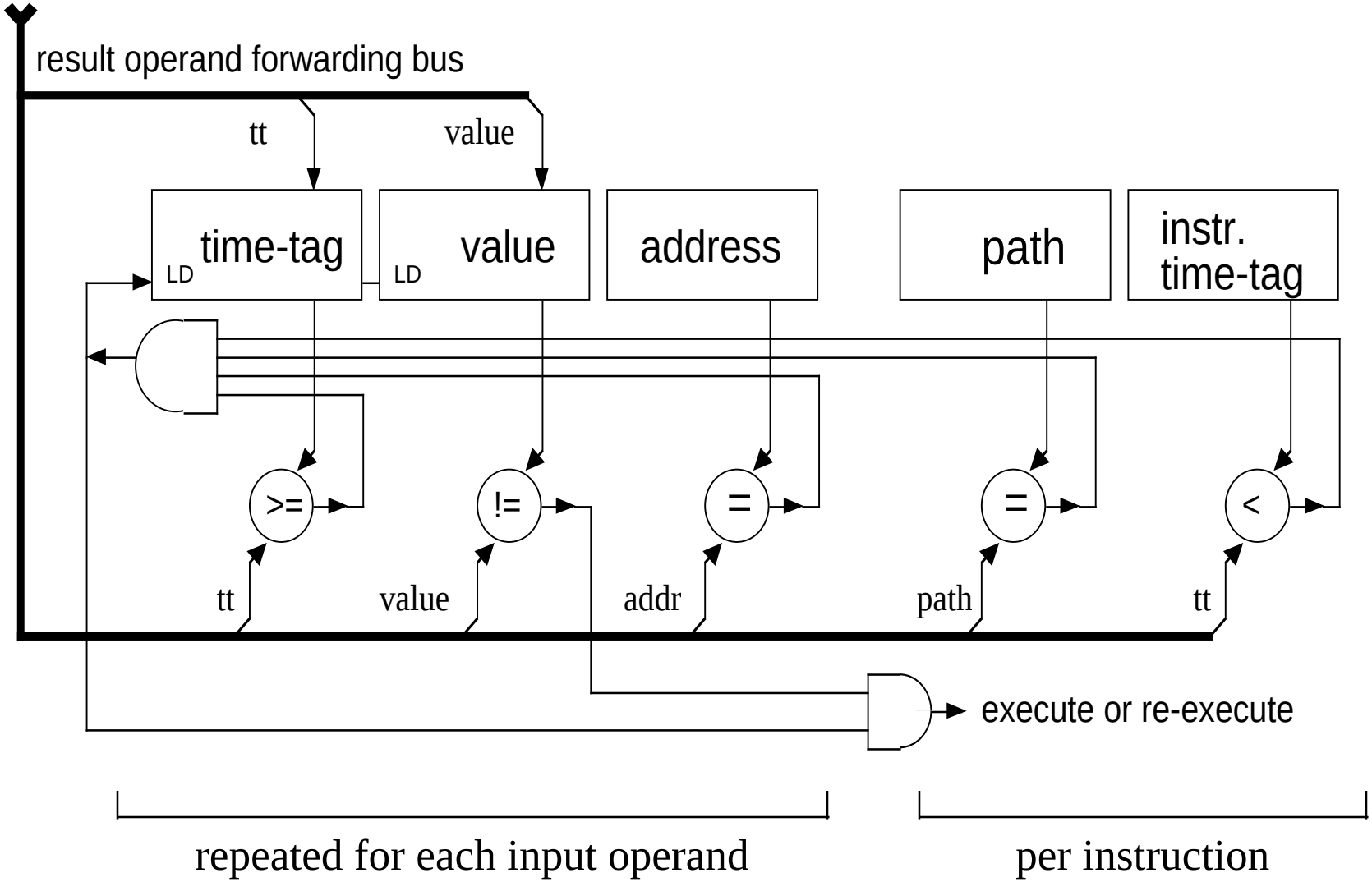
operand block

- holds all information about one operand
- includes necessary logic to snoop for updates

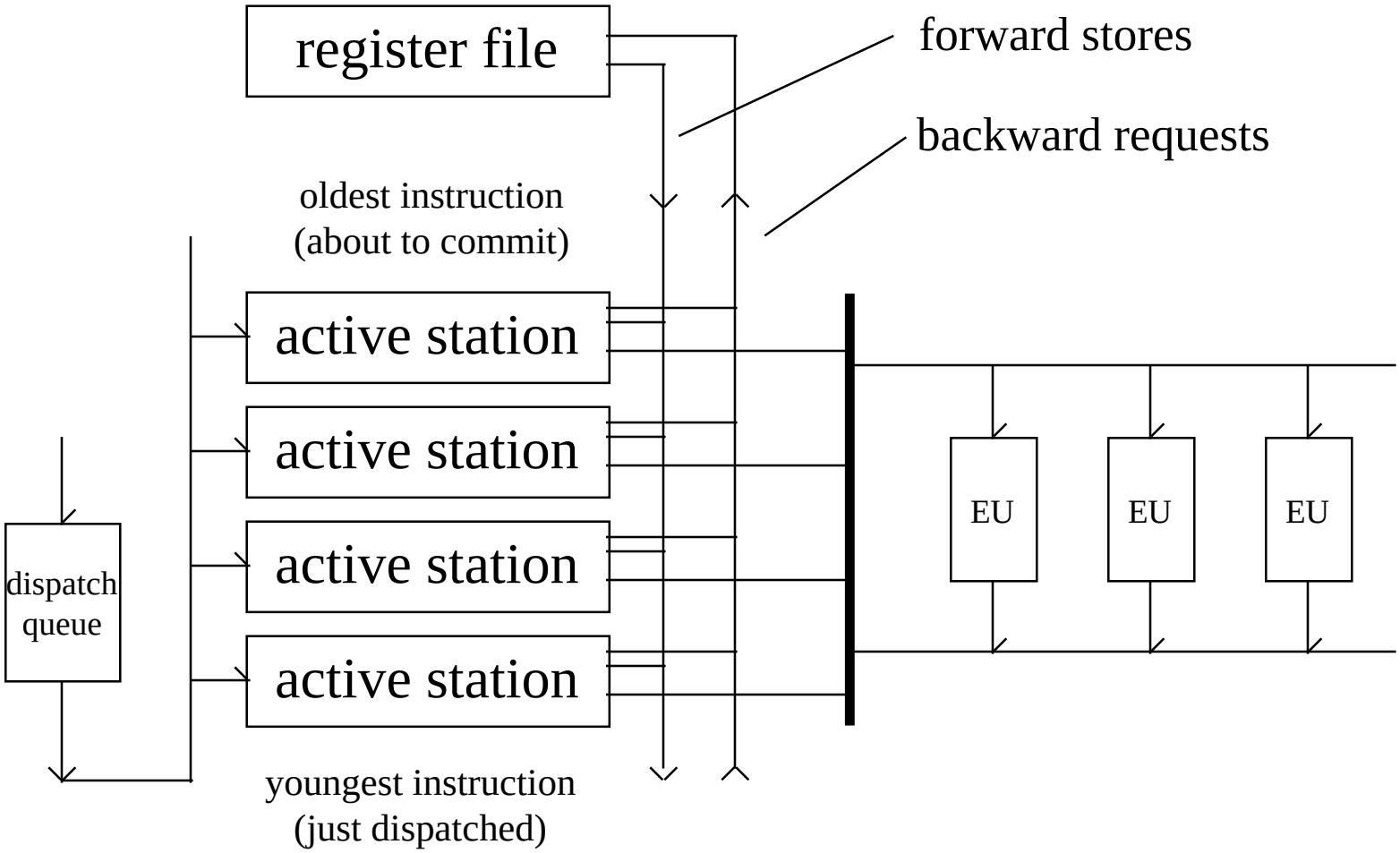


- operand names take the form -- type : path : time-tag : seq : addr
- example for a register -- "register : 1 : 27 : 3 : r6"
- predicates are operands also but have additional state

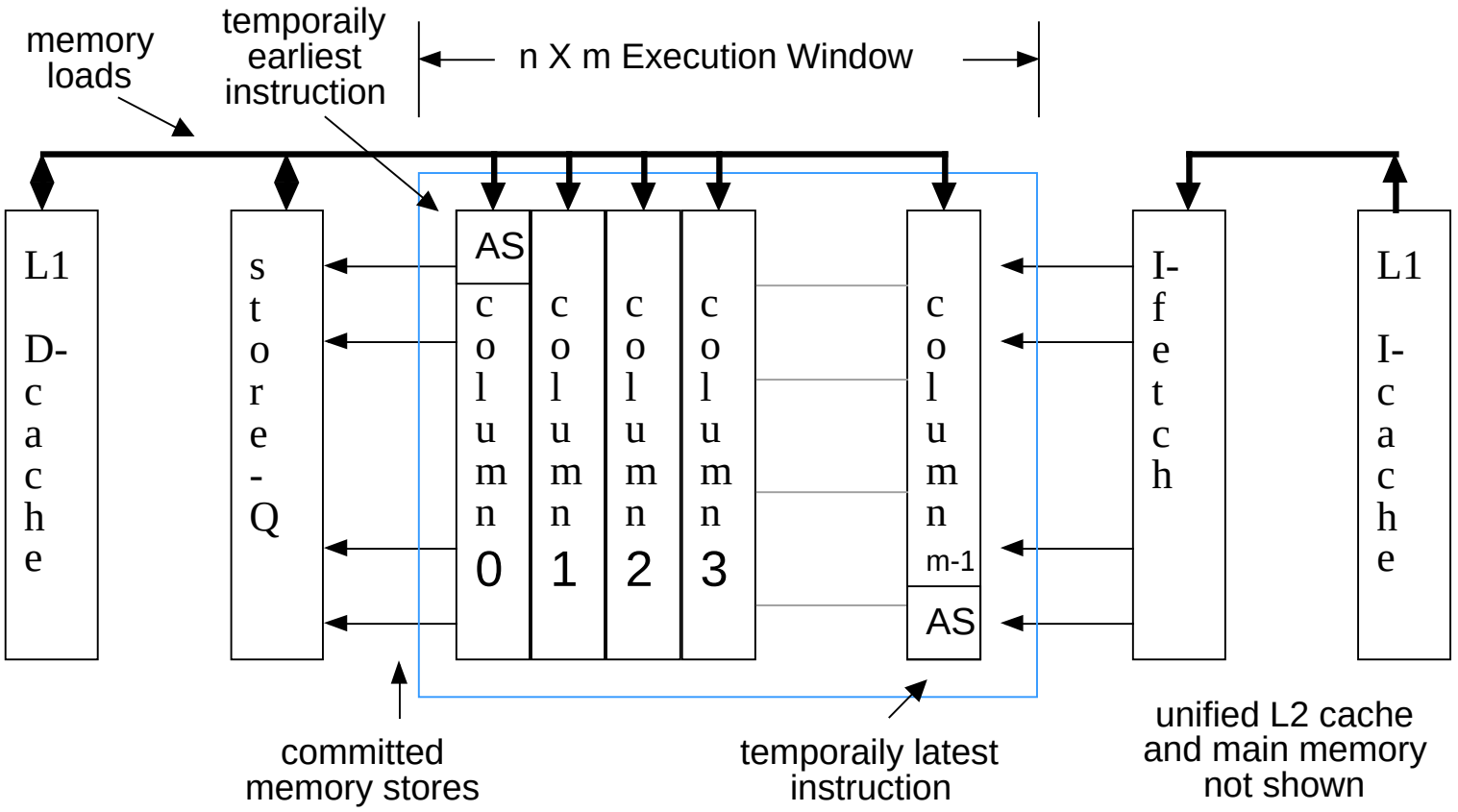
snoop/snarf operation



research microarchitecture

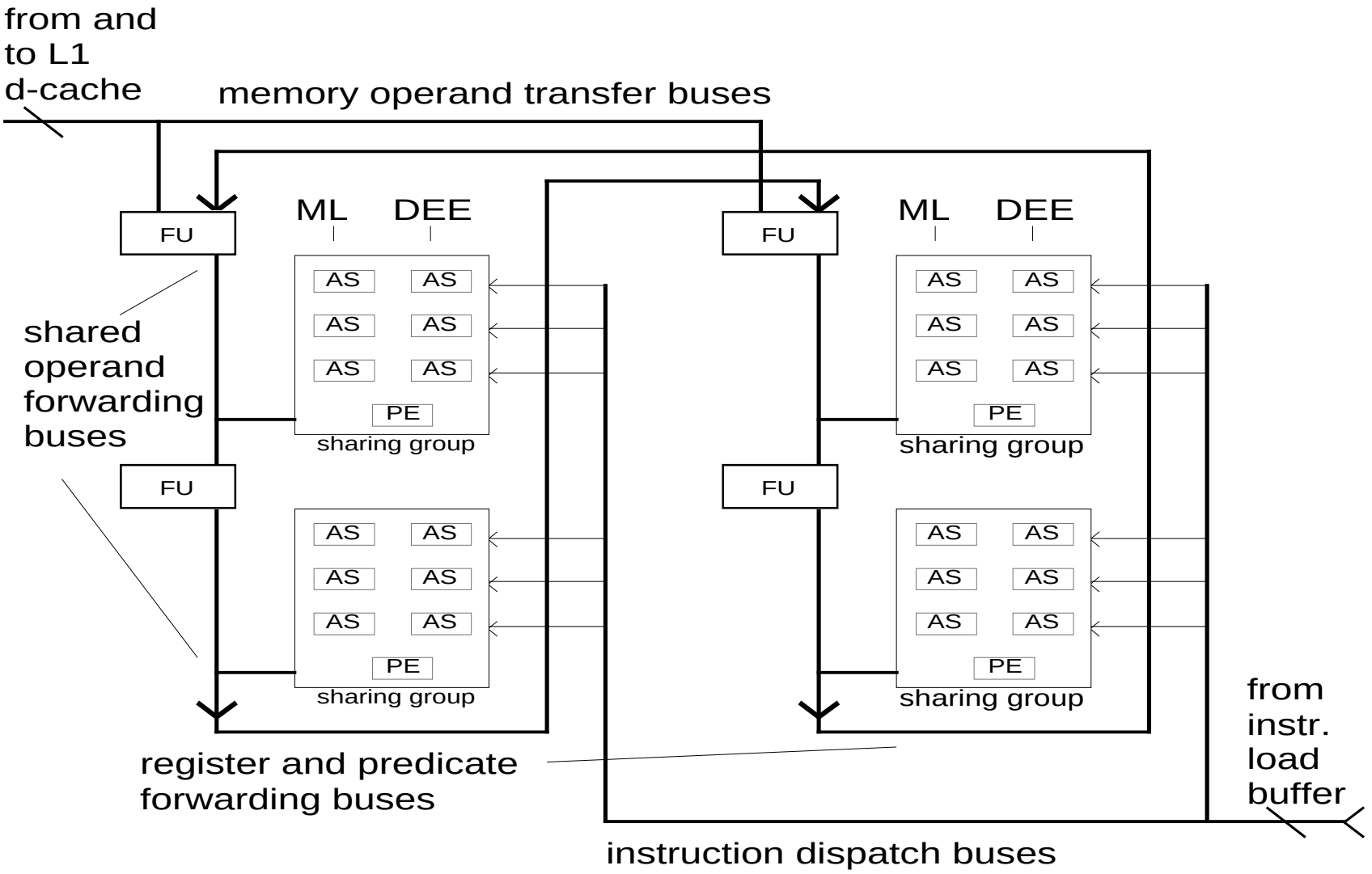


Levo microarchitecture (1)



- **ASes arranged in columns forming the Execution Window**
- **processing elements (PEs) are distributed throughout e-window**
- **columns logically rotate as whole columns are loaded and committed**

Levo microarchitecture (2)



research methodology

- **use SimpleScalar "back-end" to build a new simulator**
  - program loading
  - instruction execution
  - system call execution
- **the new simulator functionally implements our microarchitecture**
  - active stations
  - operand passing
- **I also built a "checker" that is used to independently execute the target program to guarantee correct execution**
  - Alireza Kahlafi (June 2001)
  - different from MASE checker (need different and additional information)

Explorations in Instruction Level Parallelism

student **David Morano**
advisors **Professor David Kaeli**

Northeastern University
Computer Architecture Research

NUCAR talk 03/09/12