LEVO

NUCAR

an URI / NEU collaboration

preserving dependencies
in a
large distributed microarchitecture

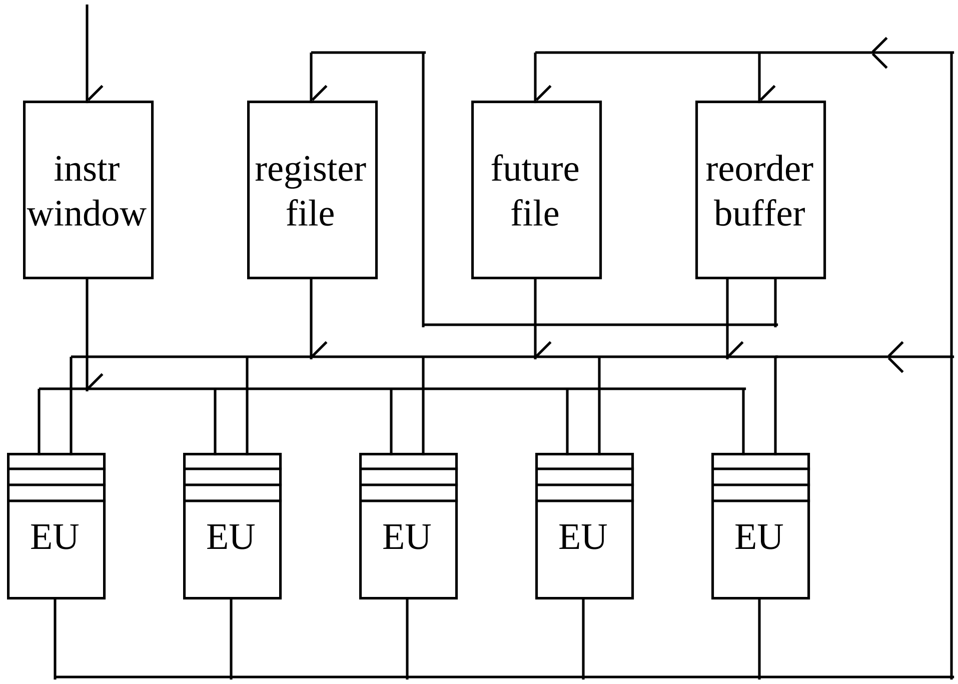
student **David Morano**
advisors **Professor David Kaeli**
Professor Augustus Uht

NUCAR talk 02/01/10

Outline

- **background**
  - reservation station
  - register files
  - reorder buffer
- **RE-FLOW distributed microarchitecture**
  - execution window
  - active stations and renaming
  - snoop/snarf
- **results**

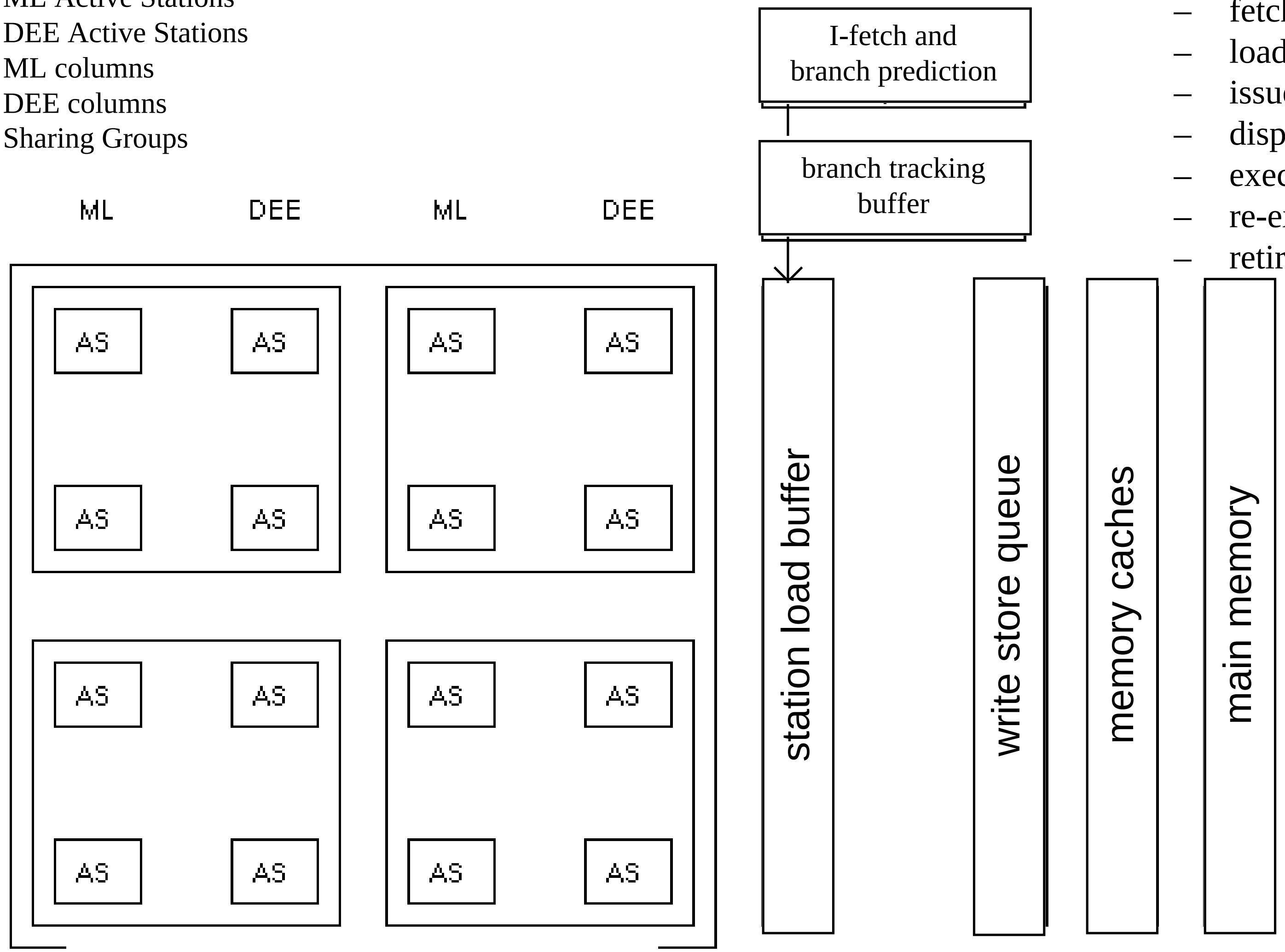
conventional microarchitecture



high-level block diagram (2)

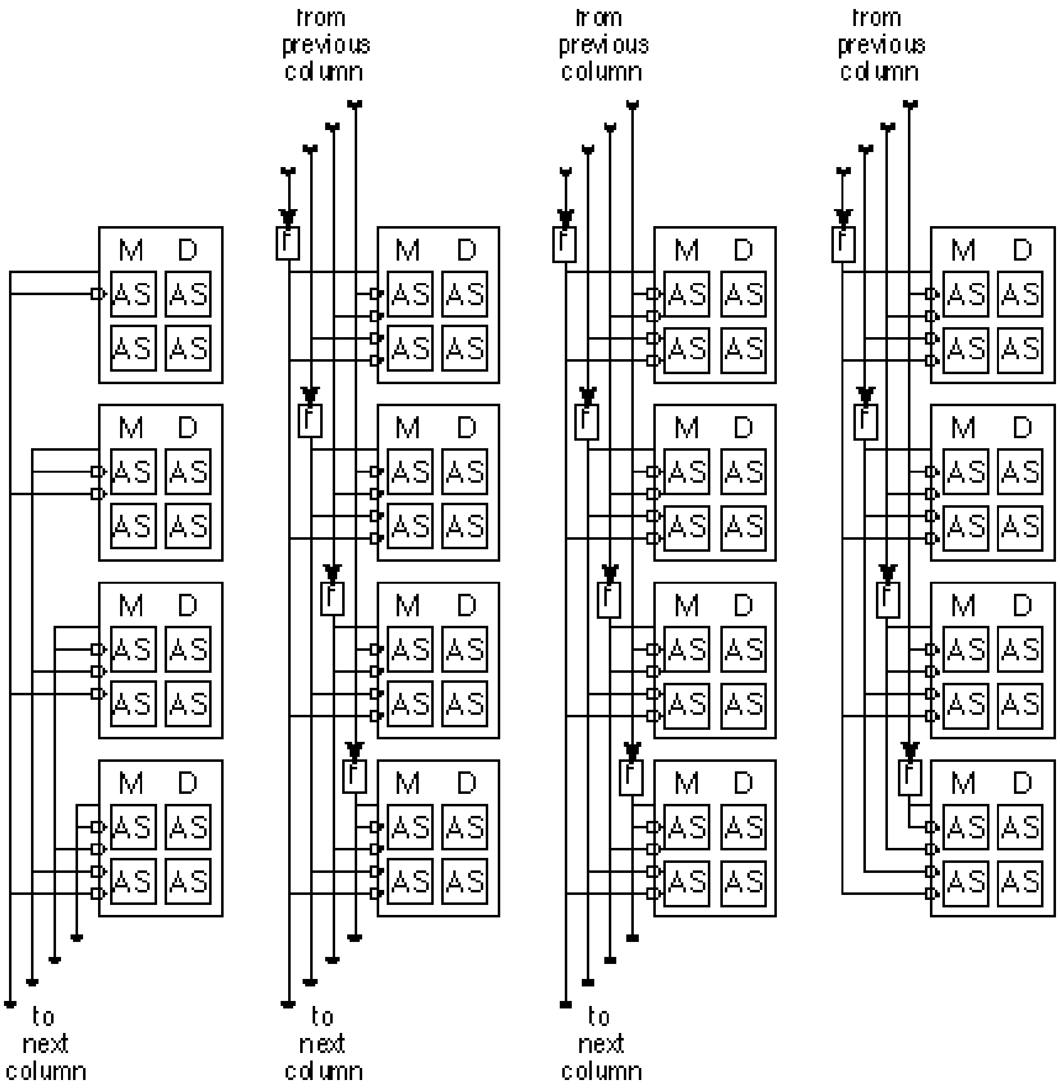
8 ML Active Stations
8 DEE Active Stations
2 ML columns
2 DEE columns
4 Sharing Groups

- **execution flow**
  - fetch
  - load
  - issue
  - dispatch
  - execute
  - re-execute !
  - retire



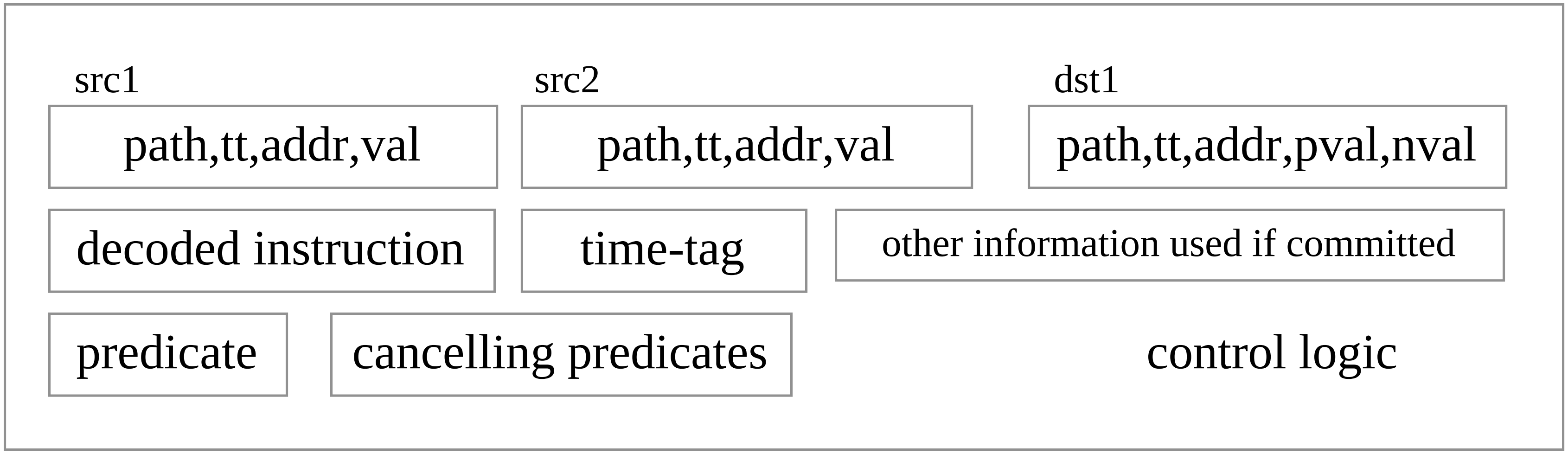
instruction execution window

sharing group forwarding buses



active station

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



src1

path,tt,addr,val

src2

path,tt,addr,val

dst1

path,tt,addr,pval,nval

decoded instruction

time-tag

other information used if committed

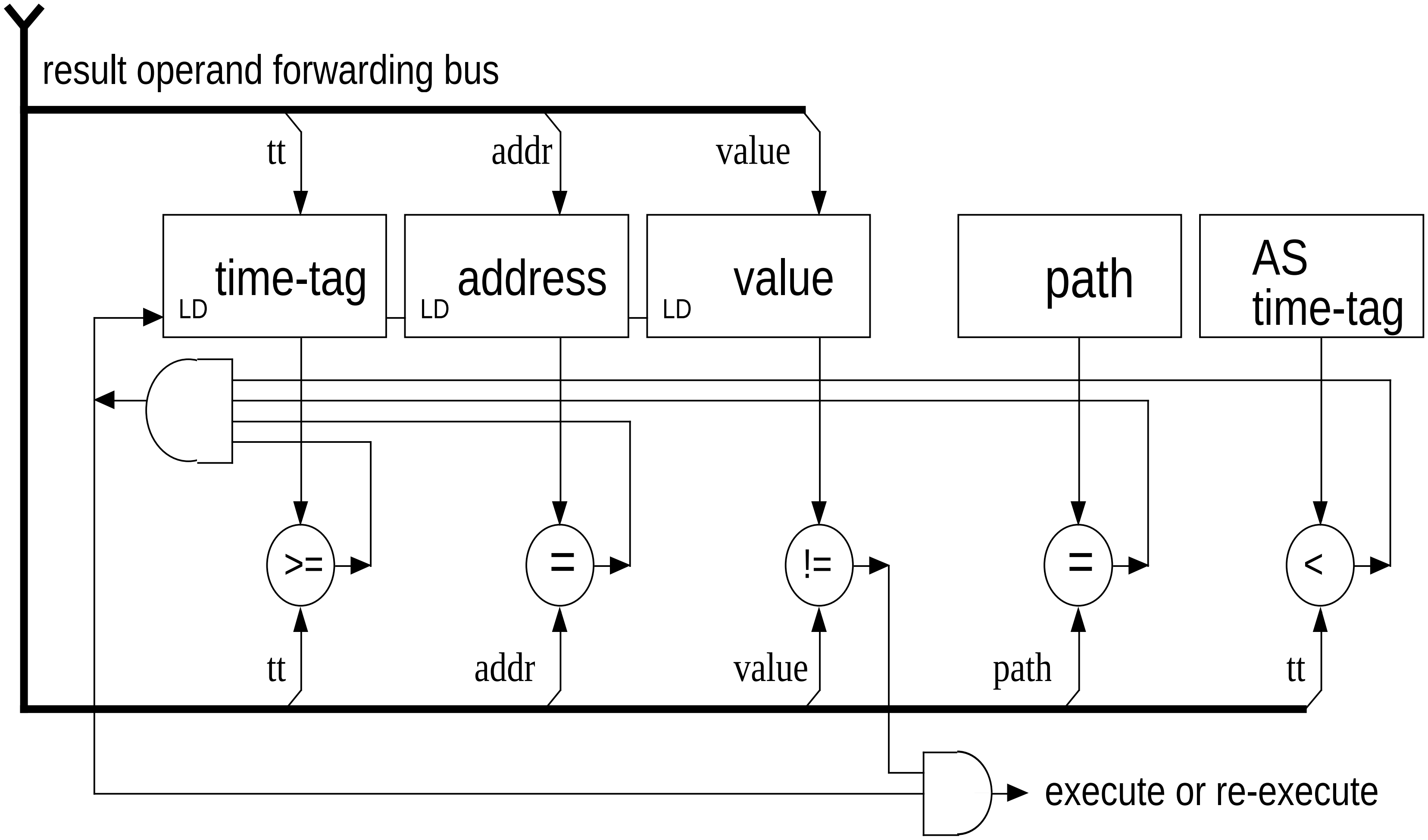
predicate

cancelling predicates

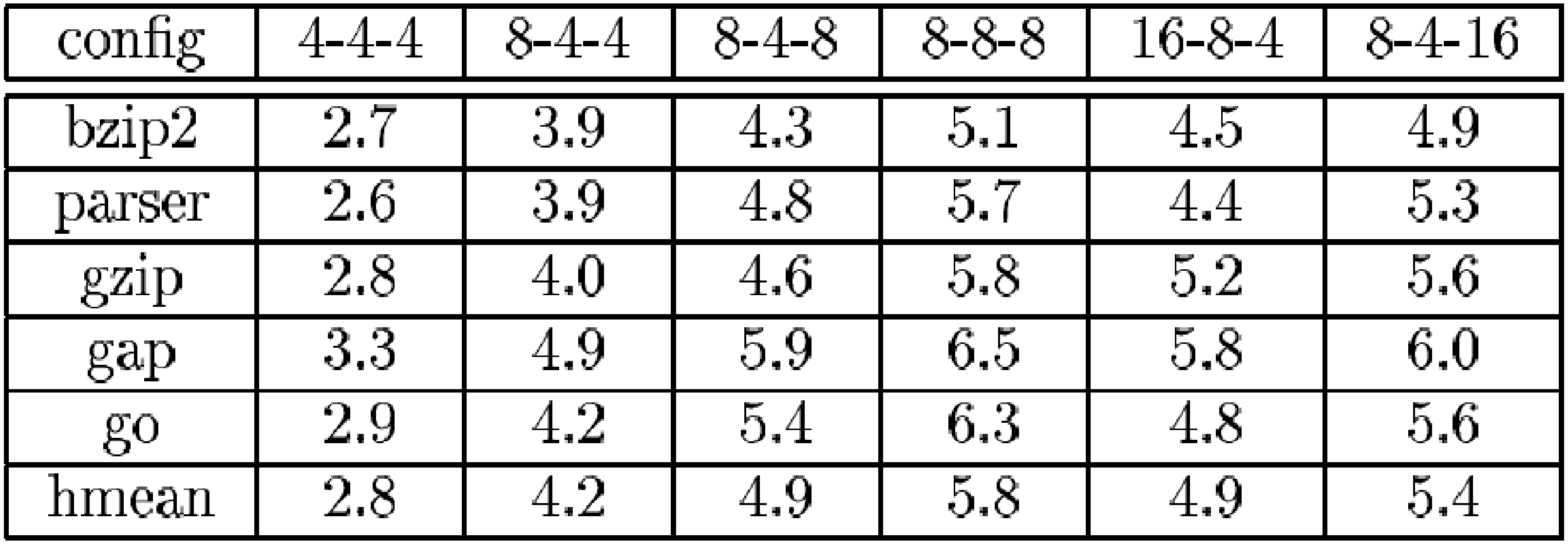
control logic

- register names take the form :
    - path : time-tag : addr
    - for example "1 : 27 : r6"

snoop/snarf operation



results



config	4-4-4	8-4-4	8-4-8	8-8-8	16-8-4	8-4-16
bzip2	2.7	3.9	4.3	5.1	4.5	4.9
parser	2.6	3.9	4.8	5.7	4.4	5.3
gzip	2.8	4.0	4.6	5.8	5.2	5.6
gap	3.3	4.9	5.9	6.5	5.8	6.0
go	2.9	4.2	5.4	6.3	4.8	5.6
hmean	2.8	4.2	4.9	5.8	4.9	5.4

- **"Levo" speculation in effect**
- **DEE paths in effect, dynamic/static i-fetch**
- **L1 instruction 100% hit**
- **L1 data 1 clock hit penalty, 10 clock mis penalty, 32kBytes 2-way set associative, 4-way interleaved**
- **L2 100% hit**
- **bus delay 1 clock, forwarding unit delay 1 clock**

summary

- **avoidance of difficult access to central microarchitectural resources is now possible**
- **time tags are used to enforce all program dependencies (control, data, memory)**
- **a scalable distributed microarchitecture is possible (large)**
- **ILP speedups are possible now due to the large microarchitectural concurrency**