Simulating 100M Endpoint Sytems

Derek Chiou

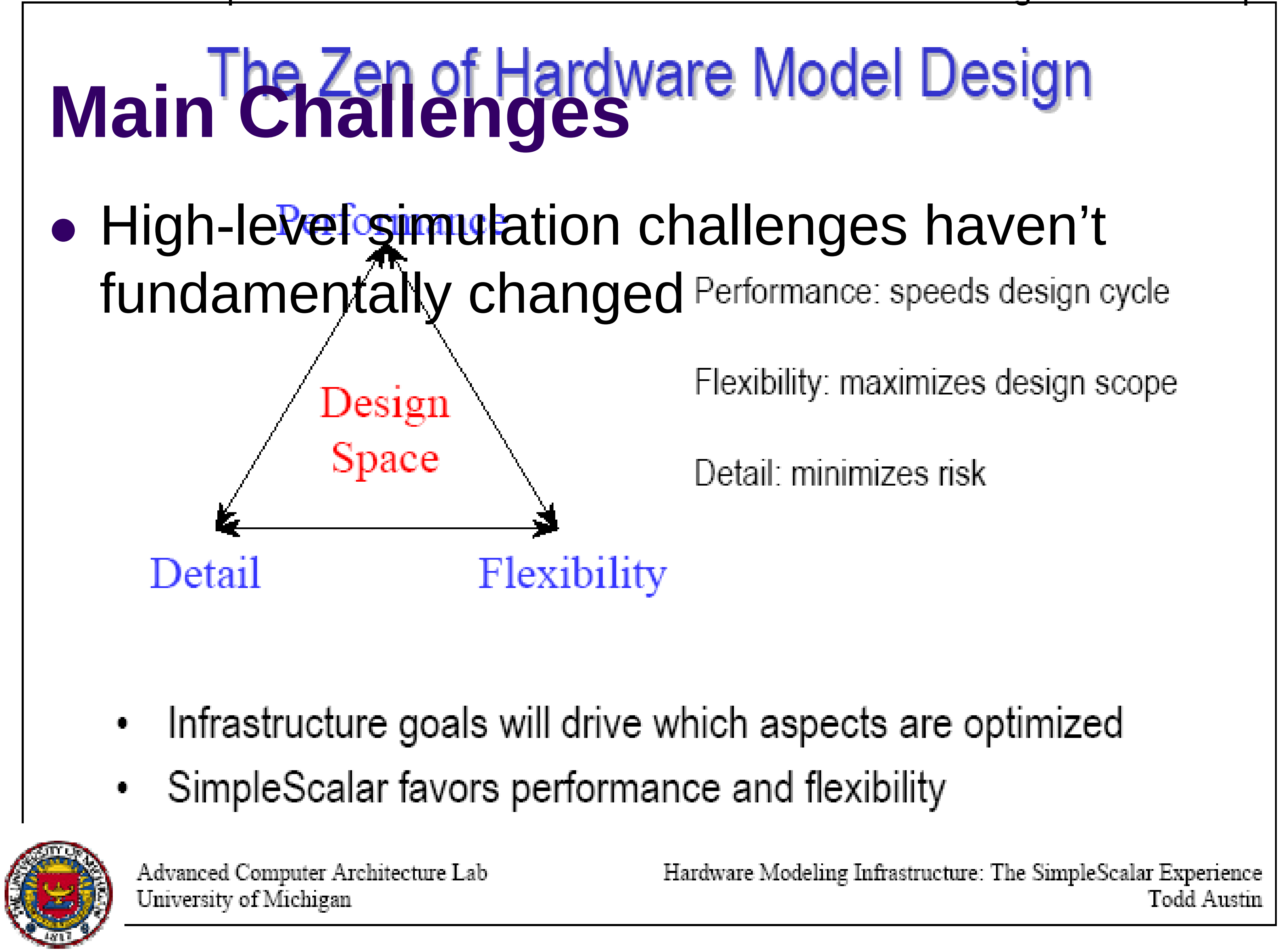
University of Texas at Austin

Electrical and Computer Engineering

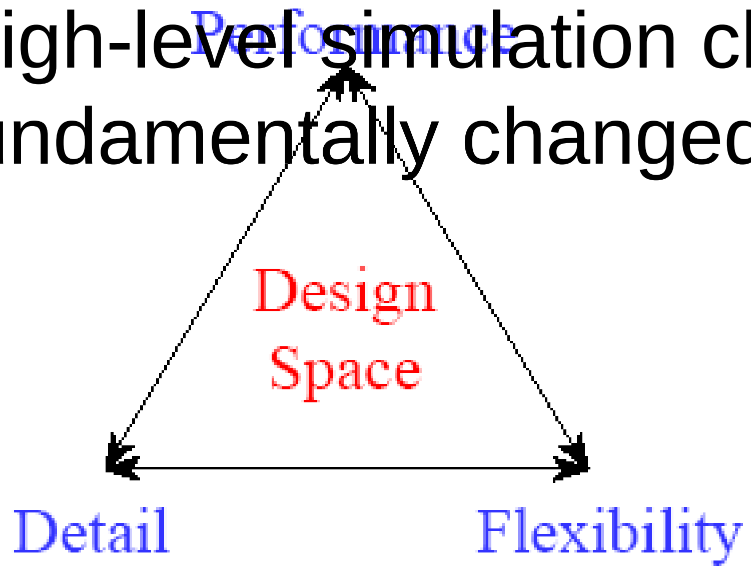
Stolen from http://research.microsoft.com/si/PPT/HardwareModelingInfrastructure.pdf

Main Challenges

- High-level simulation challenges haven't fundamentally changed



The Zen of Hardware Model Design



Performance: speeds design cycle

Flexibility: maximizes design scope

Detail: minimizes risk

- Infrastructure goals will drive which aspects are optimized
- SimpleScalar favors performance and flexibility

Advanced Computer Architecture Lab
University of Michigan

Hardware Modeling Infrastructure: The SimpleScalar Experience
Todd Austin

Some Challenges for Simulating 100M Endpoints (1)

- Simulation speed
  - Sheer size of target *system*
    - Hardware and software
  - Real software?
- Accuracy (potentially very accurate model)
  - Ability to calibrate less accurate models
    - The real system is often not available
    - Ability to tune accuracy as more information available, make the simulator faster
  - More than interconnect: Core, memory hierarchy
    - ***Core + memory needed to study NIC overhead***
  - Ensuring accuracy, calibration with real hardware/RTL

Some Challenges for Simulating 100M Endpoints (2)

- Complete
  - Run ***real*** workloads on real OS
- Flexibility
  - Want to be able to quickly explore

- Speed, accuracy, completeness needed to model interactions
  - Warmup times grow with target system size
    - Want to see instabilities/interactions that are the result of complex interactions after long warmups

A Solution: Hardware Acceleration

- Parallelize the simulator
  - Software (run on General Purpose MPP)
  - Hardware: FPGAs, special purpose ASICs, lots of minicores (MP GPUs, Tilera)
- Need a new ***simulator architecture***
  - Parallel host including hardware
  - Simulate rather than prototype
    - multiple host cycles for one target cycle
  - Hardware to collect and process stats
- Accuracy and flexibility?

Why Hardware (FPGAs)?

- FPGAs run roughly 10 times slower than processor (100MHz/1GHz)
- FPGAs much more parallel than processor
  - > 10 times as many things happening at the same time
- Flexible

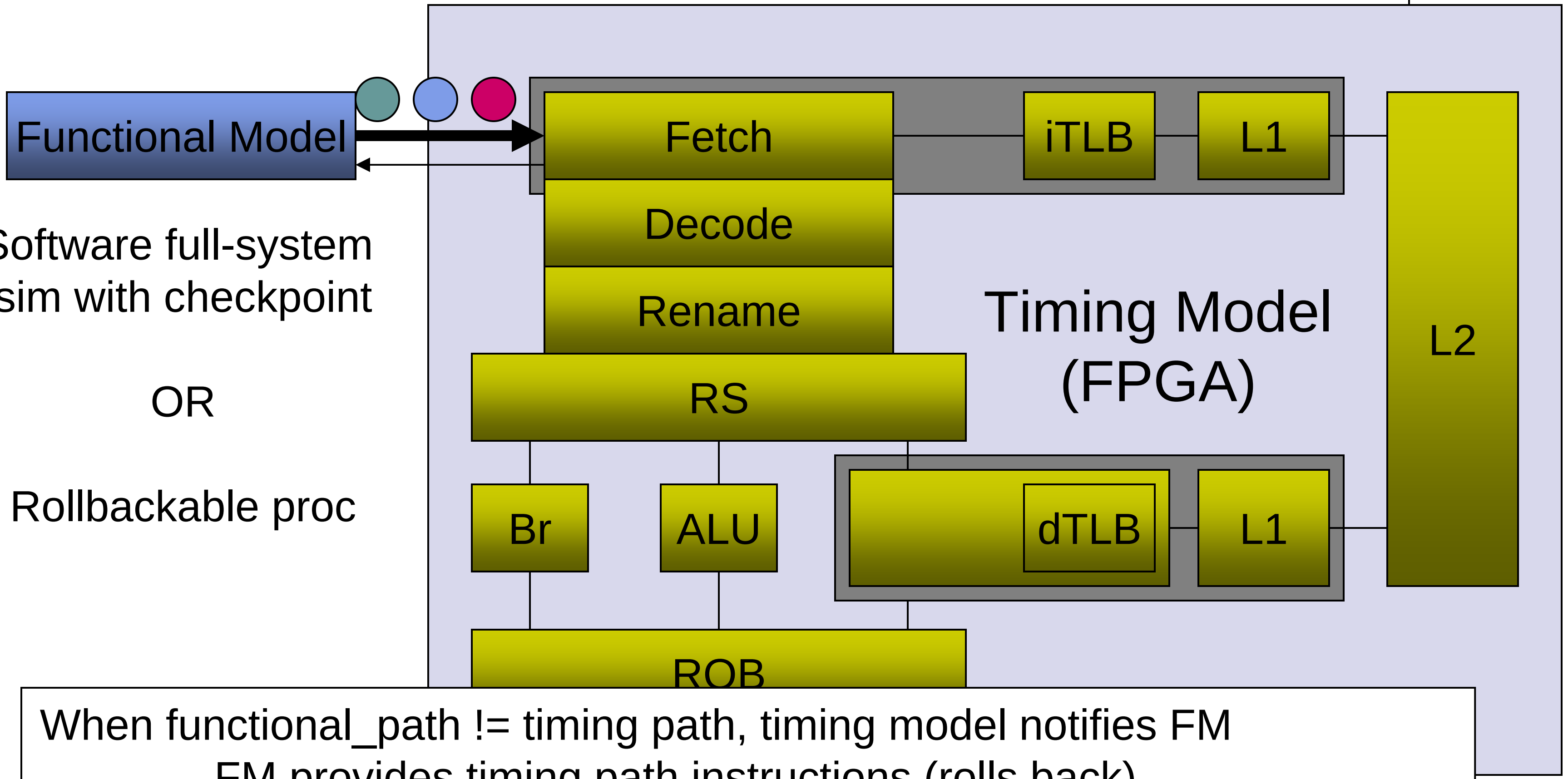
- Examples of FPGA-accelerated simulators that run 1000 times faster than pure software
  - Implies a factor of 10K better efficiency
- May be better-than-FPGA architectures for simulation
  - will require huge concurrency

Making Hardware Simulation Tractable: Partitioning

- Reuse partitions to reduce/eliminate hardware overhead and thus increase flexiblity and accuracy
  - RAM->CAM->Cache->Branch Predictor
  - Functional/timing (avoids a lot of issues)
- Permits best host technology for each partition
  - Hybrid software/hardware solutions enabled
- Existing Examples
  - FAST (Sunwoo, Patil, Reinhart, Kim, Johnson, Chiou @ Texas)
    - Functional/timing, timing in FPGA
  - HASim (Emer et al. @ Intel & MIT)
  - RAMP (Berkeley, CMU, Intel, MIT, Stanford, Texas, Washington)
    - Module level (processor/memory/network) partitioning

MICRO 2007

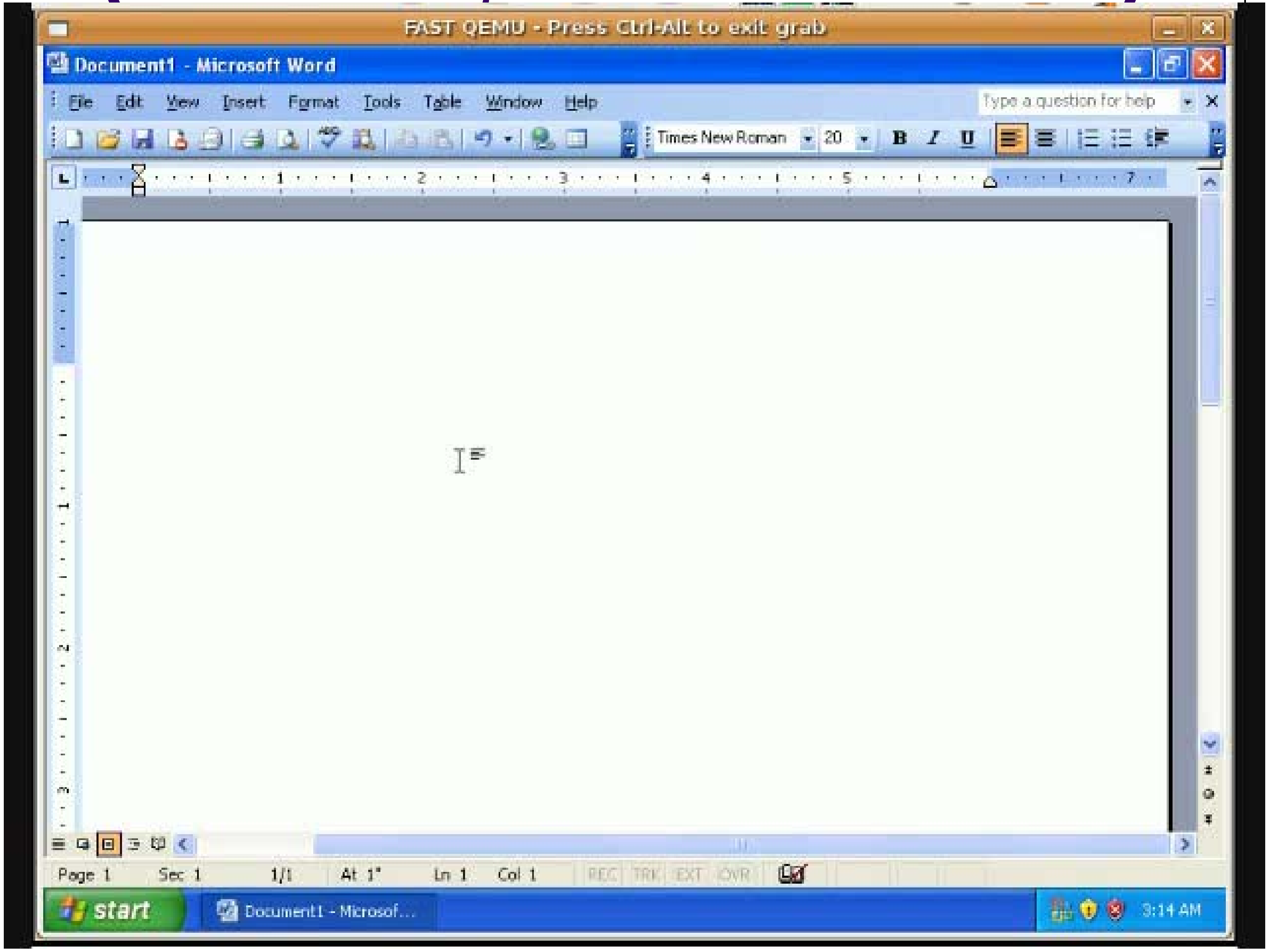
FPGA-Accelerated Simulation Technologies (FAST): Parallelized Simulators



When functional_path != timing path, timing model notifies FM
FM provides timing path instructions (rolls back)

The better the micro-architecture, the more frequently paths are the same

FAST Prototype in Real Time (~1.2MIPS, 1000x Intel/AMD)


FAST QEMU - Press Ctrl-Alt to exit grab
Document1 - Microsoft Word
File Edit View Insert Format Tools Table Window Help
Type a question for help
Times New Roman 20
Page 1 Sec 1 1/1 At 1" Ln 1 Col 1 REC TRK EXT OVR
start
Document1 - Microsof...
3:14 AM

Making Hardware Simulation Tractable: Multithreading Hardware

- Every host cycle a different target entity is simulated
  - Results of one cycle not needed on next cycle
    - Enables pipelining of complex target functionality (much easier)
  - Shares resources
    - One router model can be used to model performance of *n* routers
    - Need state per simulated target entity (DRAM/Disk based)
      - Prefetching easy since it is known what will be done next
  - ***Supports huge targets***
- Existing Examples (CPU)
  - Protoflex (Chung, Hoe @ CMU)
    - Currently simulates 16 cores, full-system at 50+MIPS aggregate
    - RAMP-Gold (Tan, Asanovic @ Berkeley)

Will complexity doom us to build what we can model?

- It often does (build without simulation?)
  - So, let's improve our modeling capabilities
- But, complexities have grown with resources

- Need ability to simulate whatever is practical to build
  - *Requires parallelized simulators*
    - Simulating large targets is a pleasingly parallel application
    - Can then ***buy*** additional simulation scale and performance
    - Hardware is a good way to parallelize
- Can simulation effort be reused for implementation?
  - Generate implementation from simulator
  - Eliminates calibration issues