
POSTECH
포항공과대학교

# Simulation Walls For System Architects

**Jangwoo Kim**

January 7, 2011

*High Performance Computing Lab*
*Department of Computer Science & Engineering*
*Pohang University of Science and Technology (POSTECH)*

POSTECH

# SIMULATION: What is it?

- **Simulation is the key of all system R&Ds.**

Simulators are S/Ws to do
- explore design choices
- come up with a new design
- evaluate the design's
  - performance
  - power
  - reliability
  - …
- improve the design with feedbacks


No simulation → No product & No research!♪

# SIMULATION: When you fail on it..

- **WORST NIGHTMARE for system developers**
  - When simulation results are **wrong**
    - You spend MONEY & TIME only to find out your design is wrong.
  - When simulation takes **too long**
    - Your bright ideas never become a real product.
  - When simulation is **too expensive**
    - You don't want to buy a supercomputer to simulate your iPhone.

How serious are these threats?♪

# SIMULATION: Where are we now?

- **Simulation is 'THE' BOTTLENECK.**

Companies go out of business.

You lose jobs.

Your papers are rejected.♪

# SIMULATION: Where are we now?

- **Simulation is 'THE' BOTTLENECK.**

Companies go out

Simulation Walls

papers are rejected.♪

# SIMULATION: What do we want?

- **Correct Simulation**
  - Can we trust simulation results?
- **Fast Simulation**
  - Can we get useful results early enough?
- **Inexpensive Simulation**
  - Can we do it at low costs?

Simulation must be correct, fast, and efficient!♪

# SIMULATION: Why is it so difficult?

- **Must know everything about HW & SW!**
  - Hardware abstraction
    - How to simplify the details of model?
  - Simulator development
    - How to develop a good simulator?
  - Workload abstraction
    - What to run on the simulator?
  - Evaluation methodology
    - What to report as a single metric?

See the challenges are real?♪

# Outline

- Introduction
- **Challenges in Simulation**
  - **HW perspective**
  - SW perspective
- Next-generation simulation
- Incoming challenges

# Scope of computer systems

data center

e.g. Database

e.g., Google, Amazon

blade/rack system

e.g., MS office

desktop system

e.g., mobile apps

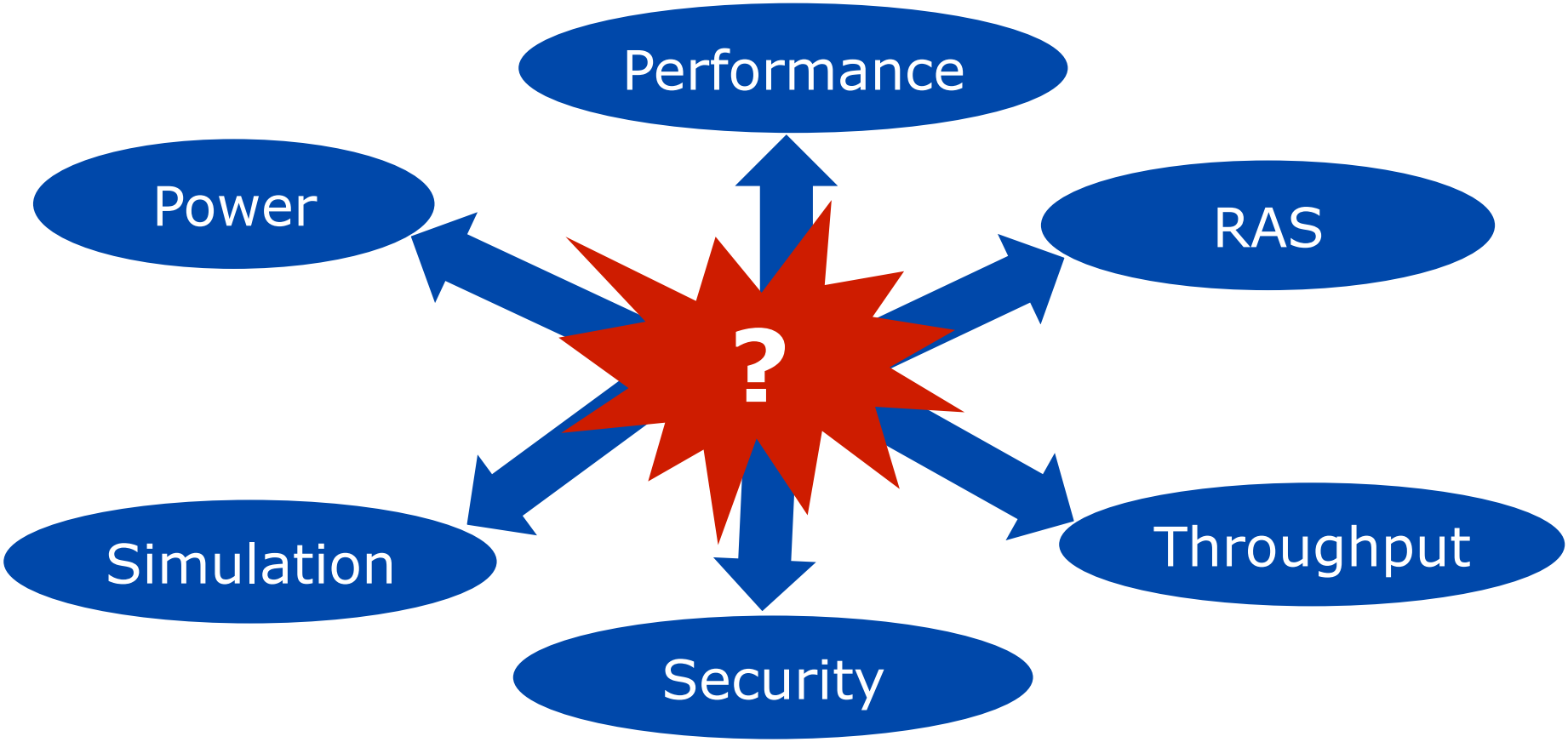
embedded system

e.g., u-Benchmark

components

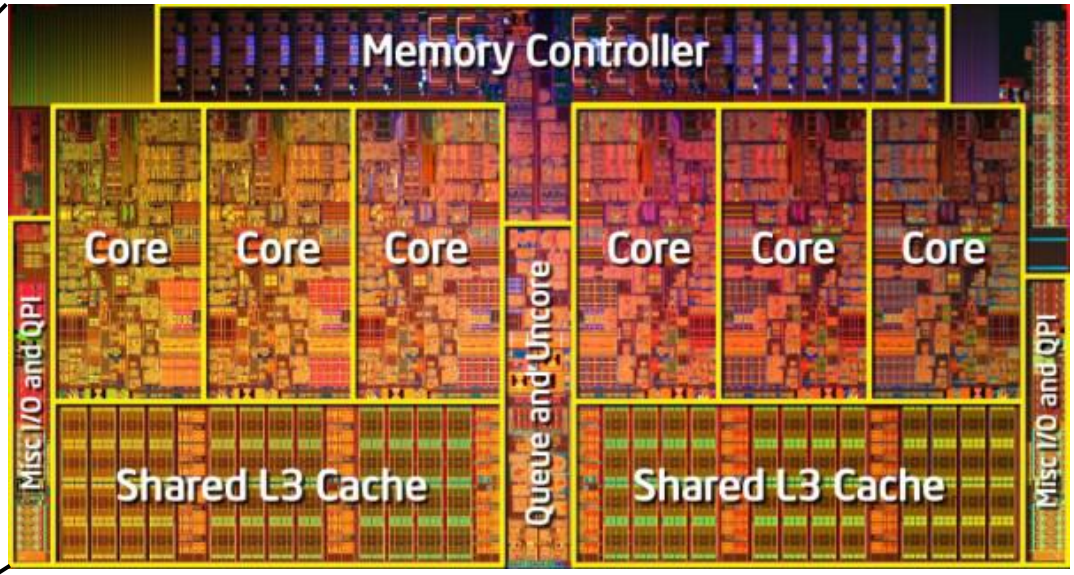
System =
Hardware + Software♪

# Design issues to be considered




Extremely difficult to find a balanced design

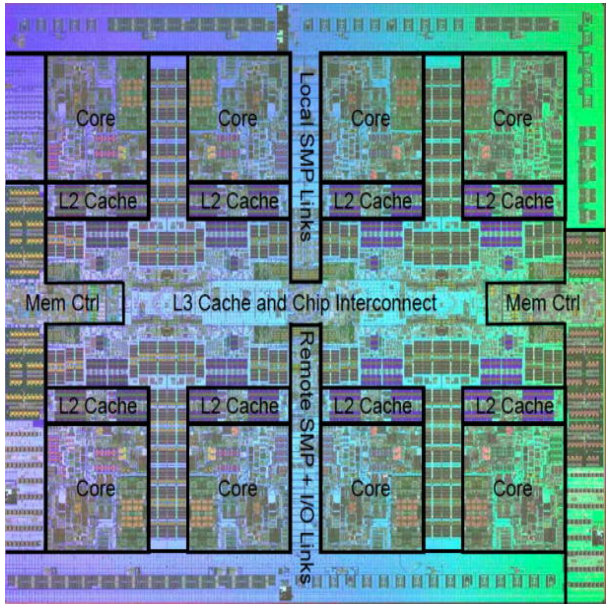
# Real world microprocessor (1/2)

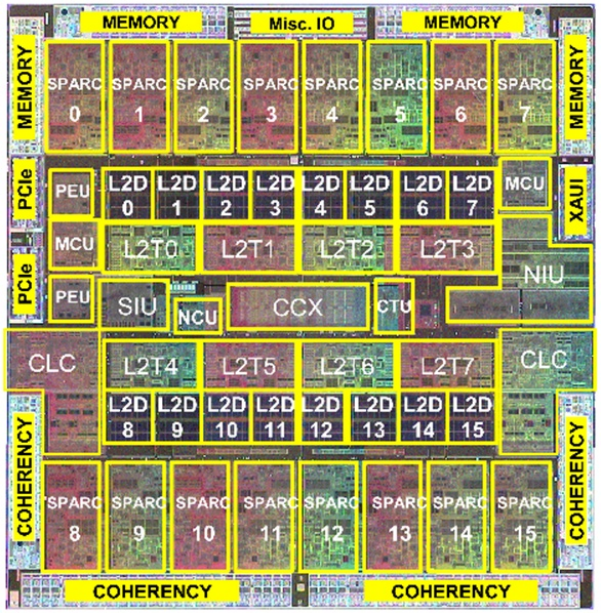

- Intel i7 Nehalem
- 12 threads on 6 cores / 12MB L3 cache
- > 100W TDP
- >1B transistors!

Multi-core/Multi-thread CPUs are REAL!♪

# Real world microprocessor (2/2)




- IBM Power7
- 32 threads on 8 cores
- 32MB L2 cache
- >1B transistors!




- Sun Rainbow Fall
- 128 threads on 16 cores
- 6MB L2 cache
- >1B transistors!

We are getting 'Many-core/Many-thread' CPUs!♪

# Increasing HW complexity

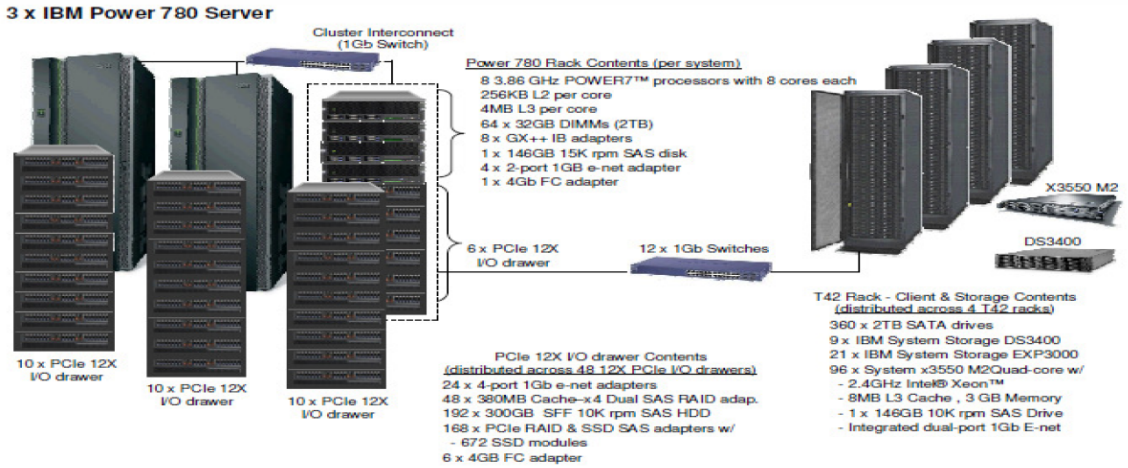
- **Existing systems already complex enough**
  - E.g., modern CPU designs
    - Multi-core / multi-thread architecture
    - Deep pipeline with speculative execution
    - Multi-level caches
- **New HW technologies make things worse**
    - Heterogeneous cores
    - Flash / PCM / 3D memory
    - FPGA, reconfigurable computing
    - Next-generation platform (e.g., smart phone, datacenter, etc.)

Simulation → too slow, buggy and expensive.♪

# Outline

- Introduction
- **Challenges in Simulation**
  - HW perspective
  - **SW perspective**
- Next-generation simulation
- Incoming challenges

# Real world workload: e.g., database, facebook




http://www.tpc.org/tpcc

- TPC-C run on DB2 / IBM Power 780 (8/17/2010) → **8M users, 10.3M tpmC**
- 768 threads /192 cores /24 CPUs
- ~2TB memory/~40TB SSD /~250TB disks → >$14M USD
- **2 hour measurement after 2.5 hour warm-up !!**

Real workloads are extremely large-scale, long-latency, full-system workloads.♪

# Increasing SW complexity

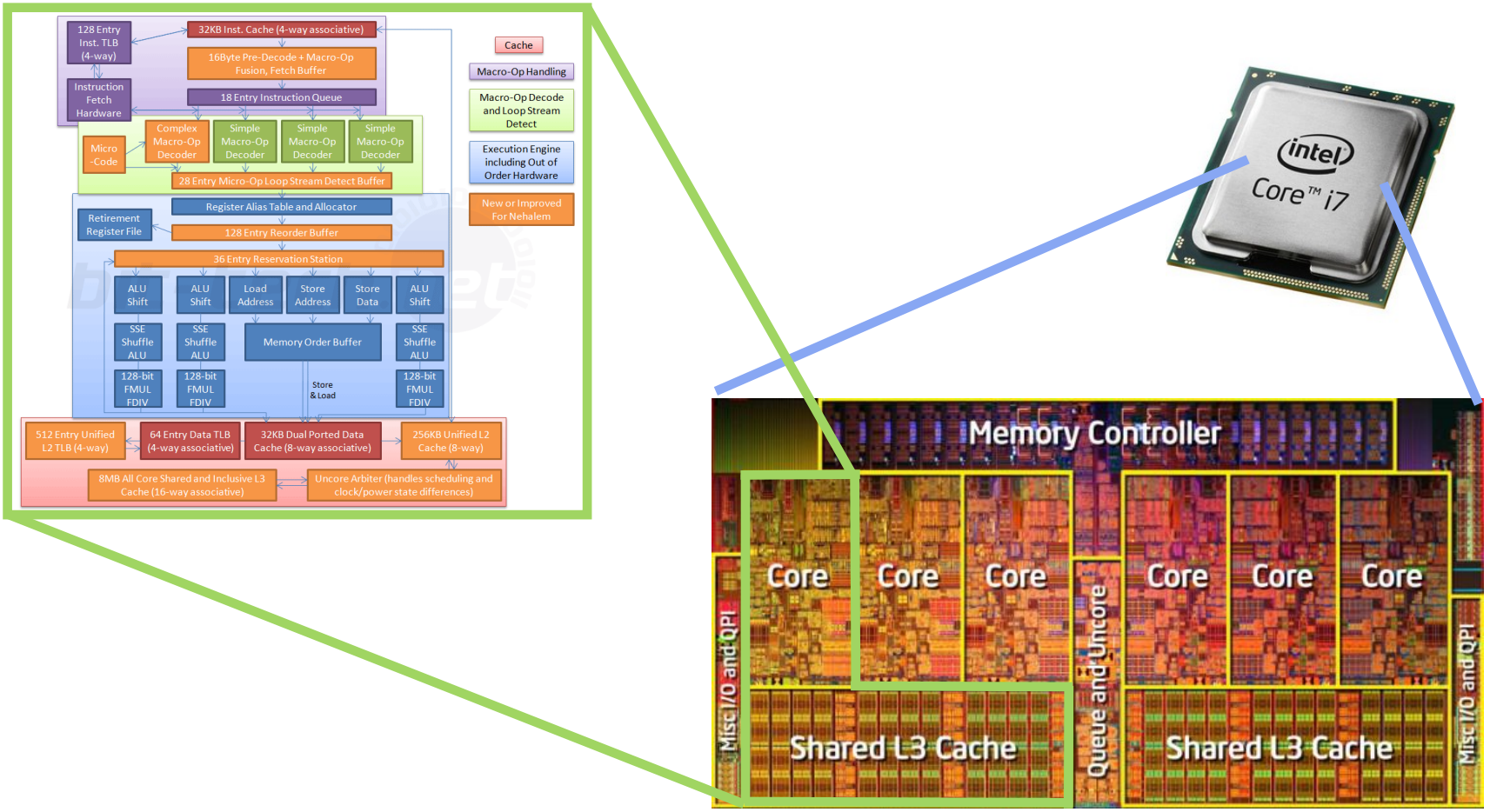
- **Existing SWs already complex enough**
  - E.g., Database engine, Web Server
    - Full-system requirement
    - Large-scale deployment
    - Long-latency benchmarking
- **Incoming SWs make things worse**
  - Cloud/ Web2.0+ // Handling many users?
    // Mixed workloads on Virtual Machine?
  - Mobile applications // App store?
  - ? // What is coming next?

Simulation becoming too slow, too buggy and too expensive.♪

# Outline

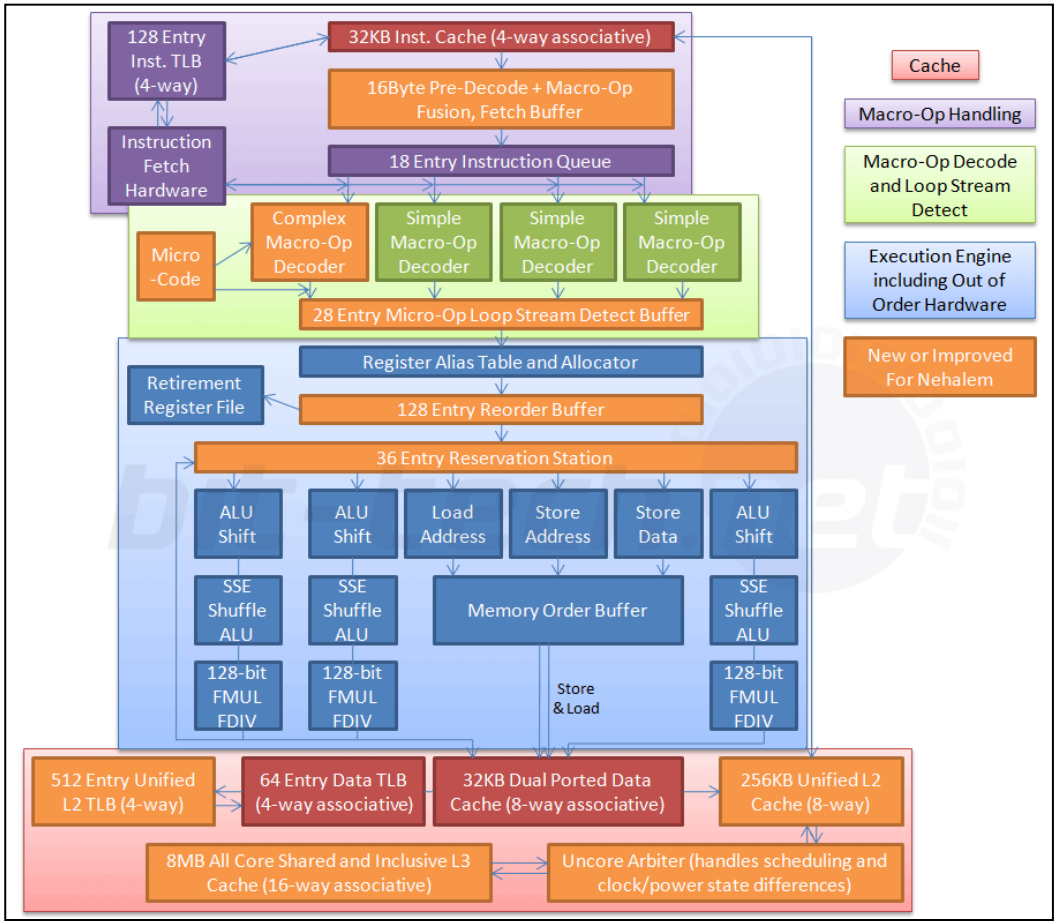
- Introduction
- Simulation challenges
- Next-generation simulation
  - **Scalable simulation**
  - Full-system simulation
  - Simulation acceleration
  - Simulation validation
- Incoming challenges

# Abstraction: What to simulate?




Abstraction can be applied at every level.♪

# Architect's view of CPU




- Core pipeline
  - CISC->RISC translation
  - Hyper-threading
  - Branch prediction
  - Out-of-order execution
- Multi-core/multi-thread
  - 2 thread per core
  - 4-6 cores per chip
- Cache
  - 32KB L1 i-cache
  - 32KB L1 d-cache
  - 256KB L2 cache
  - 8-12MB L3 cache

This is 'micro-architecture' of Intel i7 CPU. ♪

# Even this is already too complex

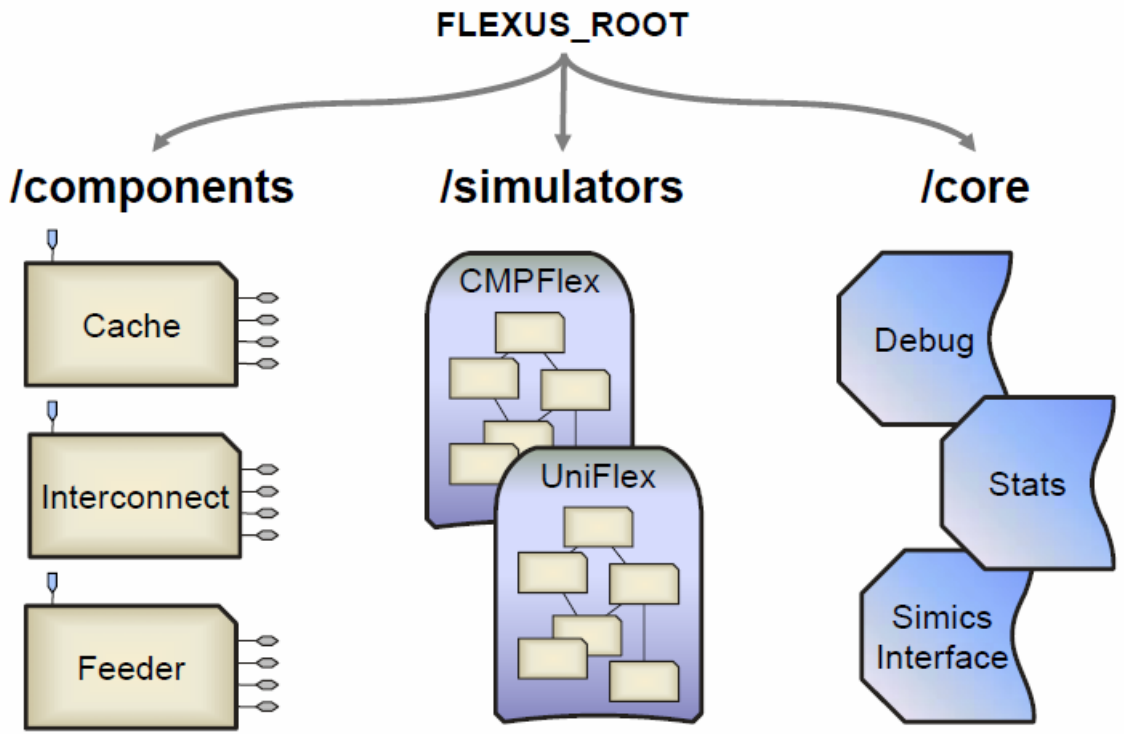
- **CPU simulator is a very very very large SW.**
  - >100K lines of high perf. codes (usually in C/C++)
  - Everything should be written for design explorations.
    - More cores? more threads?
    - Higher clock? deeper pipeline?
    - In-order execution VS out-of-execution?
    - Larger caches? high associative caches?
    - Different cache protocols?
    - Memory consistency?
    - …

How to make a flexible and scalable simulator?♪

# Flexus: open-source CPU simulator (1/2)

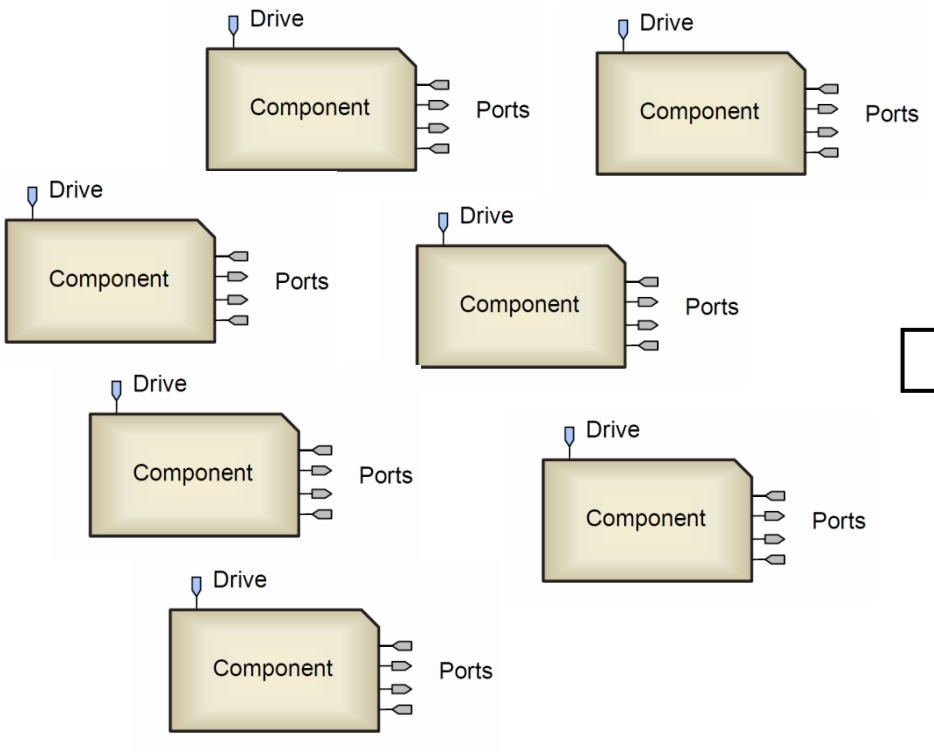
- Cycle-accurate CPU and system simulator built by 'modules.'
- Multi cores, multi threads, multi nodes and distributed memory.
- > 100K lines written in C++ by >5 PhDs at CMU for 3+ years
- Widely used for research

Modular simulation is the key.♪

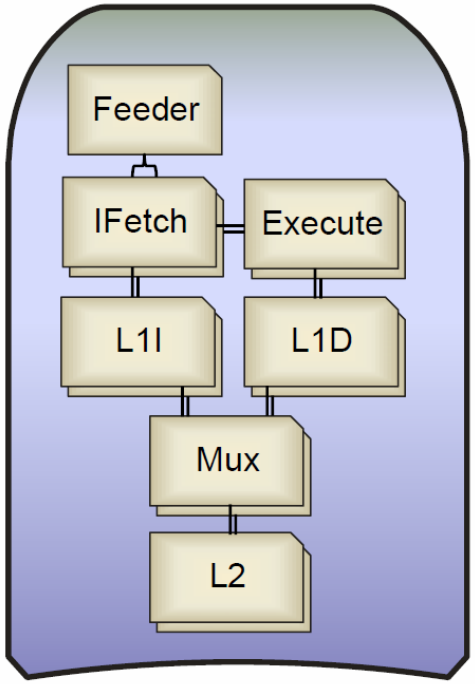


# Flexus: open-source CPU simulator (2/2)

**Various modules**




**CPU simulator**




Simulator is made by connecting modules
→ scalable & flexible!♪

# Warning..

**- Modular simulation is for scalability & flexibility.**

**- Typical software engineering issues still remain.**

(e.g., slow and expensive simulation,

many-people involved project,

impractical code review,

bugs bugs bugs bugs bugs ..)

# Outline

- Introduction
- Challenges in simulation
- Next-generation simulation
  - Scalable simulation
  - **Full-system simulation**
  - Simulation acceleration
  - Simulation validation
- Incoming issues

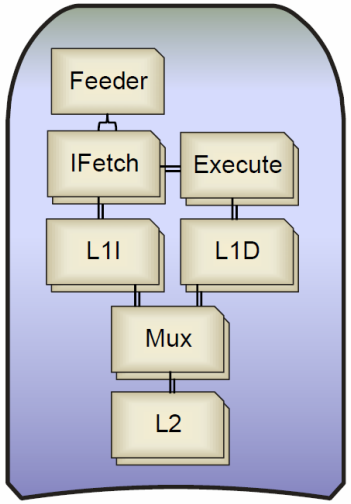
# Full-system simulation is a MUST!

- **What is a full system?**
  - Made of entire stacks of HW and SW
    - HW: CPU + memory + I/O + network + …
    - SW: OS + device drivers + commercial applications + …
  - Runs system-level applications (e.g., DB, Web server).
  - → **Modern mobile devices are full systems.**
- **Full-system simulation is difficult.**
  - Difficult to model every detail of HW and SW.

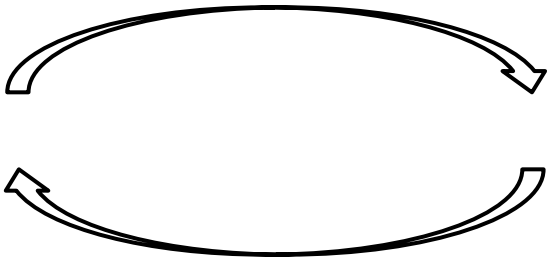
How to do efficient full-system simulation?♪

# Full-system simulation using VM (1/2)

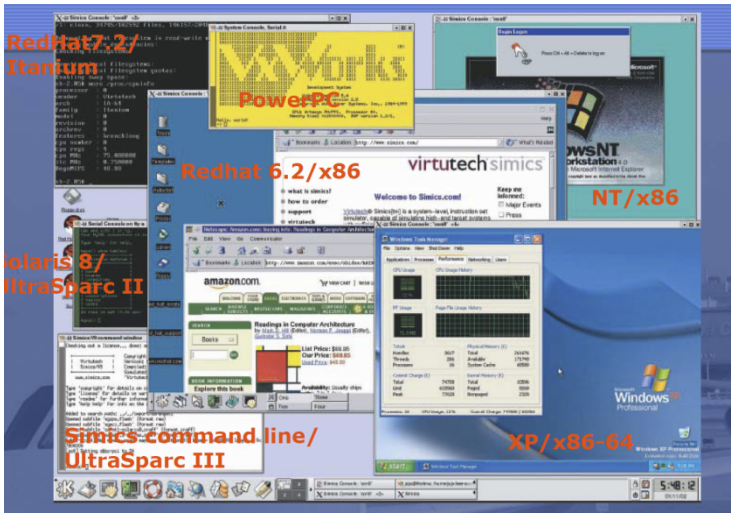
**Flexus : detailed timing simulator**




- cycle accurate timing simulation



**Simics: Virtual Machine functional simulator**




- full-system functional simulation

For system-level functionality, we can use existing Virtual Machine S/Ws!♪

# Full-system simulation using VM (2/2)

- **Simics (or similar VMs)**
  - supports various full-system platforms
    - ISA: x86, PowerPC, SPARC, ARM, etc.
    - OS: Window, Linux, Solaris, AIX, etc.
    - Devices: memory, disk, bridges, etc.

    **→ Can run system-level workloads on commercial OS!**
- **Flexus (or other timing sims)**
  - communicate with VM to offload full-system functionality (e.g., IO system call handling)

Flexus & Simics → detailed timing simulation of full-system commercial workloads!♪

# Outline

- Introduction
- Challenges in simulation
- Next-generation simulation
  - Scalable simulation
  - Full-system simulation
  - **Simulation acceleration**
  - Simulation validation
- Incoming challenges

# Detailed simulation is very SLOW

- **Typical simulation speed**
  - Speed granularity as Instruction Per Second (IPS)
    - Real machine (e.g., Intel CPU) : 1,000,000,000 IPS
    - Same ISA/arch functional VM simulator : 500,000,000 IPS
    - Different ISA/arch functional VM simulator : 1,000,000 IPS
    - Timing simulator (e.g., Flexus) : 1,000 IPS
  - 1 min on real machine → >1 year on timing simulator
  - However, real-world workloads require long-period benchmarking (e.g., hours for TPCC on database engine)

Must accelerate simulation as much as possible!♪

# Reduce the size of workload

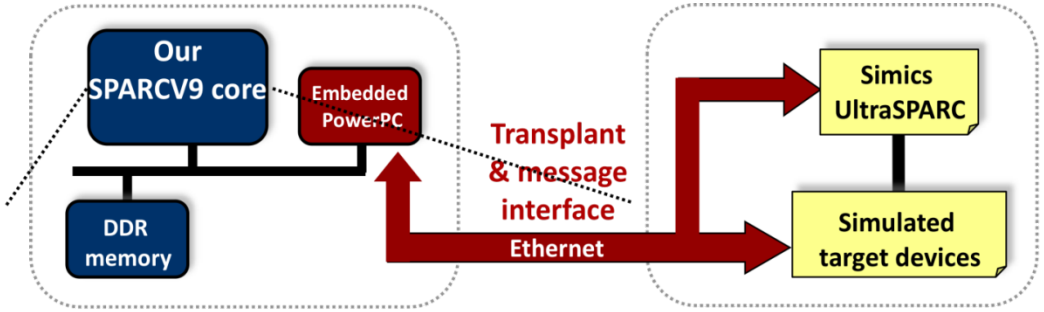
- **Parallel runs of only fractions of workload**
  - Run only few frequently-visited long routines
    - Require fast profiling on a real machine
    - Difficult to apply for parallel workloads
  - Alternatively, run many small samples in parallel
    - Simulating small samples (e.g., 1M instructions) in parallel
    - A small sample must restore some uArch info (e.g., cache state)
    - Too slow to re-create samples even with a fast simulator

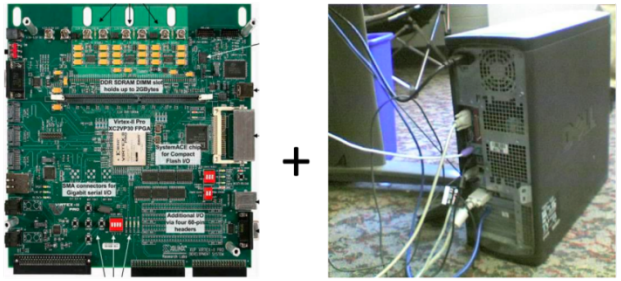
Real machine : 1 hour
→ Sequential simulation to create 1K samples : > 1 month
→ Parallel timing simulation with 1K samples : < 1 day♪

# HW/SW co-simulation♪

**Fast, functional CPUs on FPGA**

**Simics: Virtual Machine functional simulator**



HW/SW co-simulation accelerated Flexus sample creation by > 100x!♪

# Outline

- Introduction
- Challenges in simulation
- Next-generation simulation
  - Scalable simulation
  - Full-system simulation
  - Simulation acceleration
  - **Simulation validation**
- Incoming challenges

# Can we trust simulation results?

- **Suppose you modeled & simulated a CPU**
  - 4-way execution engine architecture
    - Can commit up to 4 instruction per cycle (Ideal IPC=4)
    - However, you got IPC of 2 for a target workload (< Ideal IPC)
  - Now, we are in serious dilemma!
    - Is this the real performance?
    - Is there a bug in model?
    - Is there a bug in simulator?
    - Is there a bug in workload?

**Unfortunately, all of these could be true!**

We must validate our simulation results!♪

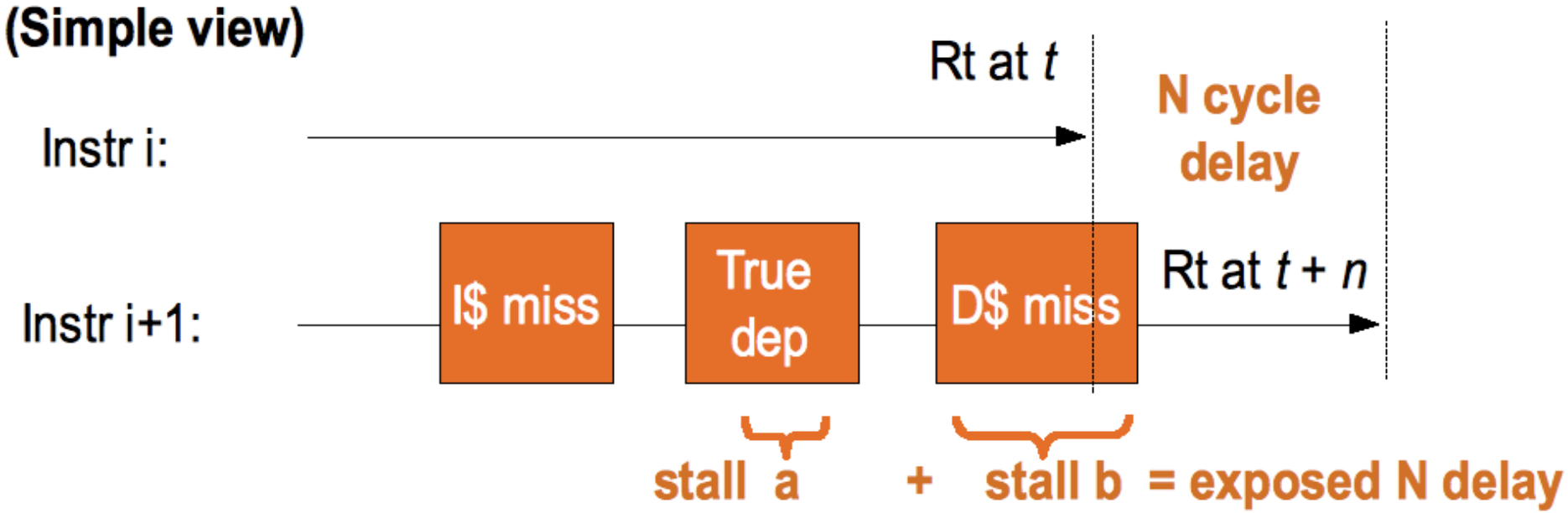
# Simulation validation is difficult

- **Old methods do not work well any more**
  - Instruction Per Clock (IPC) / Clock Per Instruction (CPI)
    - Can't explain internal pipeline behaviors
  - Performance counters
    - Can't explain timing behaviors
  - Per-cycle pipeline-view outputs
    - Human eyes can track of only 100s instructions at a time
  - Per-cycle RTL signal-view outputs
    - Human eyes can track of only few instructions at a time

We need a new way to validate sim. results♪

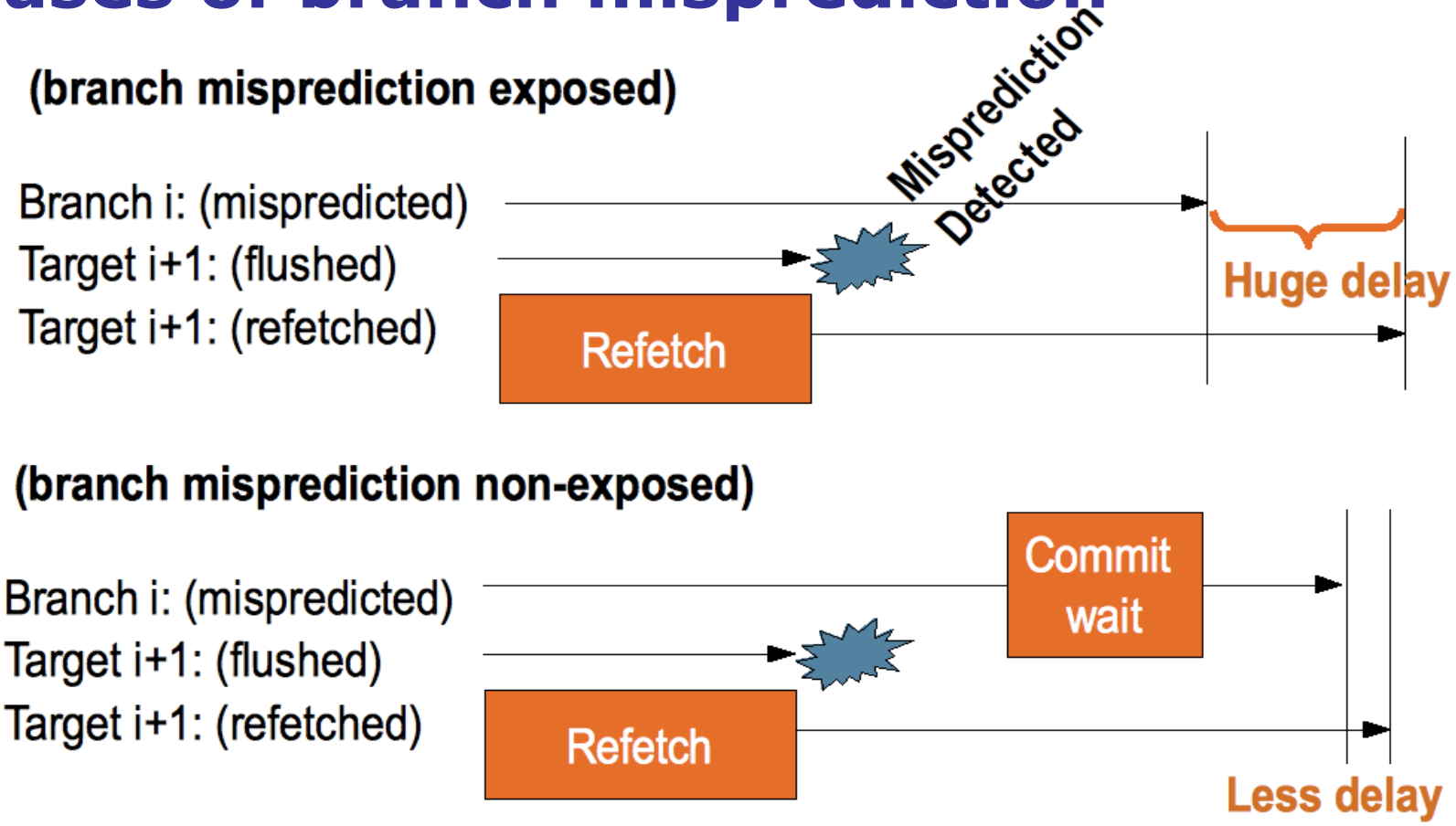
# Sim. validation with Perf. Analyzer (1/3)

- **Let's find out where we lose timing**
  - If an instruction in a given pipe stage does not move to the next stage, there is a perf. losing event. What is it?



# Sim. validation with Perf. Analyzer (2/3)

- **Cases of branch misprediction**




Now we know the exact impact of misprediction!♪

# Sim. validation with Perf. Analyzer (3/3)

- **We can tell where and why we lost timing**
  - E.g. " 35% loss due to L2 d-cache miss
    30% loss due to branch misprediction
    20% loss due to floating-point division
    15% loss due to data dependency "

    **can compare these values!**
  - Key advantages
    - Algorithm needs to analyze only in-flight instructions
    - Statistical data-correlation methods can be applied
    - Post-analysis tools can be used to avoid re-simulations
    - Sim VS Sim comparison is easy(e.g., RTL vs Timing)

Currently being implemented in Flexus @ POSTECH!♪

# Outline

- Introduction
- Challenges in simulation
- Next-generation simulation
- **Incoming challenges**

# Future (or now) challenges (1/2)

- **Simulation issue**
  - How to simulate 100s cores, 100s nodes, data centers?
  - How to measure power, reliability, variability, etc?
  - How to parallelize a S/W simulator?
  - How to apply HW/SW co-simulation more aggressively?
- **Workload issue**
  - What is a representative workload for a future system?
  - How to profile future workloads?
  - How to re-size future workloads?

# Future (or now) challenges (2/2)

- **Full system simulation issue**
  - How to parallelize VM?
  - How to move on open-source VM?
- **Validation issue**
  - How to validate trace/execution-driven simulations?
  - How to validate RTL simulations?
  - How to validate multi-phase simulations?

A long, challenging journey waiting for us.♪

# Final messages

- Simulation is THE KILLER.

- We needs fast, accurate, inexpensive simulators.

- No real solution yet.

- Let’s work on it together!

# Question?

## Thank You!

Jangwoo Kim
e-mail: jangwoo@postech.ac.kr
http://www.postech.ac.kr/~jangwoo