ΠΡΟΣΟΜΟΙΩΣΗ ΑΡΧΙΤΕΚΤΟΝΙΚΗΣ

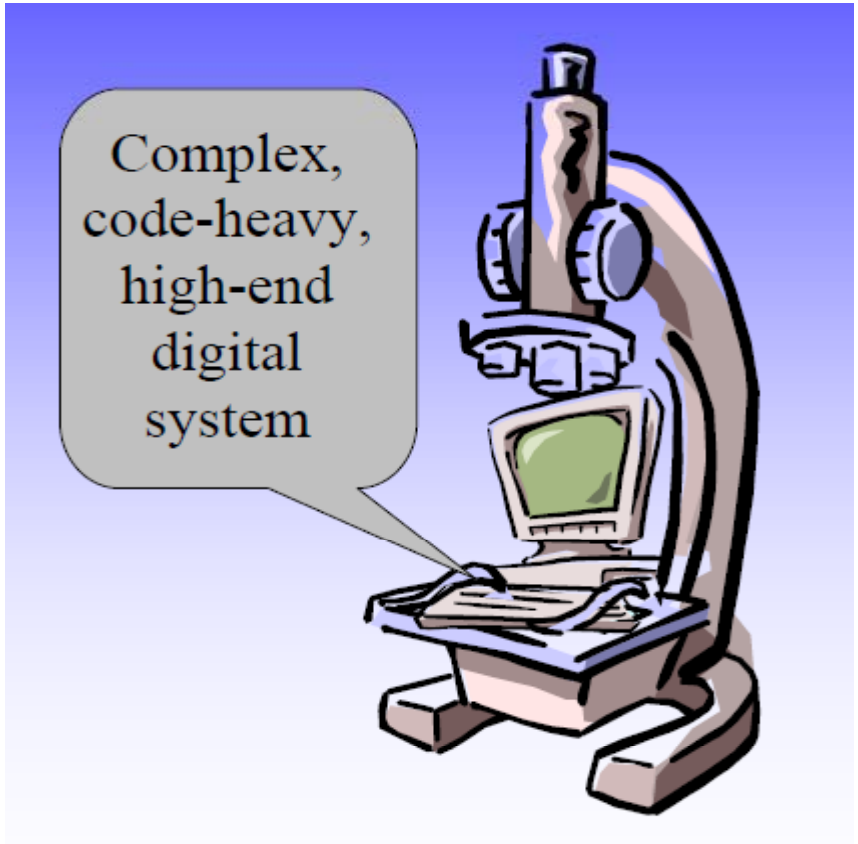
- **Διαφάνειες από το MicroArch 35 Tutorial του Simics**

http://www.cs.pitt.edu/~cho/cs2410/currentsemester/handouts/simics_tutorial.pdf

Εισαγωγή

- Αρχιτεκτονική Υπολογιστών
  - Σχεδίαση επεξεργαστή για την εκτέλεση 1 thread (pipeline, branch prediction)
  - Σχεδίαση για την εκτέλεση n threads (SMT resource allocation, threads scheduling)
  - Ετερογενείς αρχιτεκτονικές (Cell)
  - Παράλληλα συστήματα
  - Ιεραρχία μνήμης (cache sharing, coherence protocols, NUMA architectures)
  - Δίκτυα διασύνδεσης (on-chip interconnection networks)
  - Virtualization
  - Παράγωγή παράλληλου κώδικα (synchronization costs, locks, Transactional Memory, automatic parallelization)

Εισαγωγή



- Αρχιτεκτονική
- Σχεδίαση
- Υλοποίηση
- Δοκιμή/Έλεγχος
- Ανάλυση
- Ανάπτυξη (Deployment)
- Εκπαίδευση χρηστών

Εισαγωγή

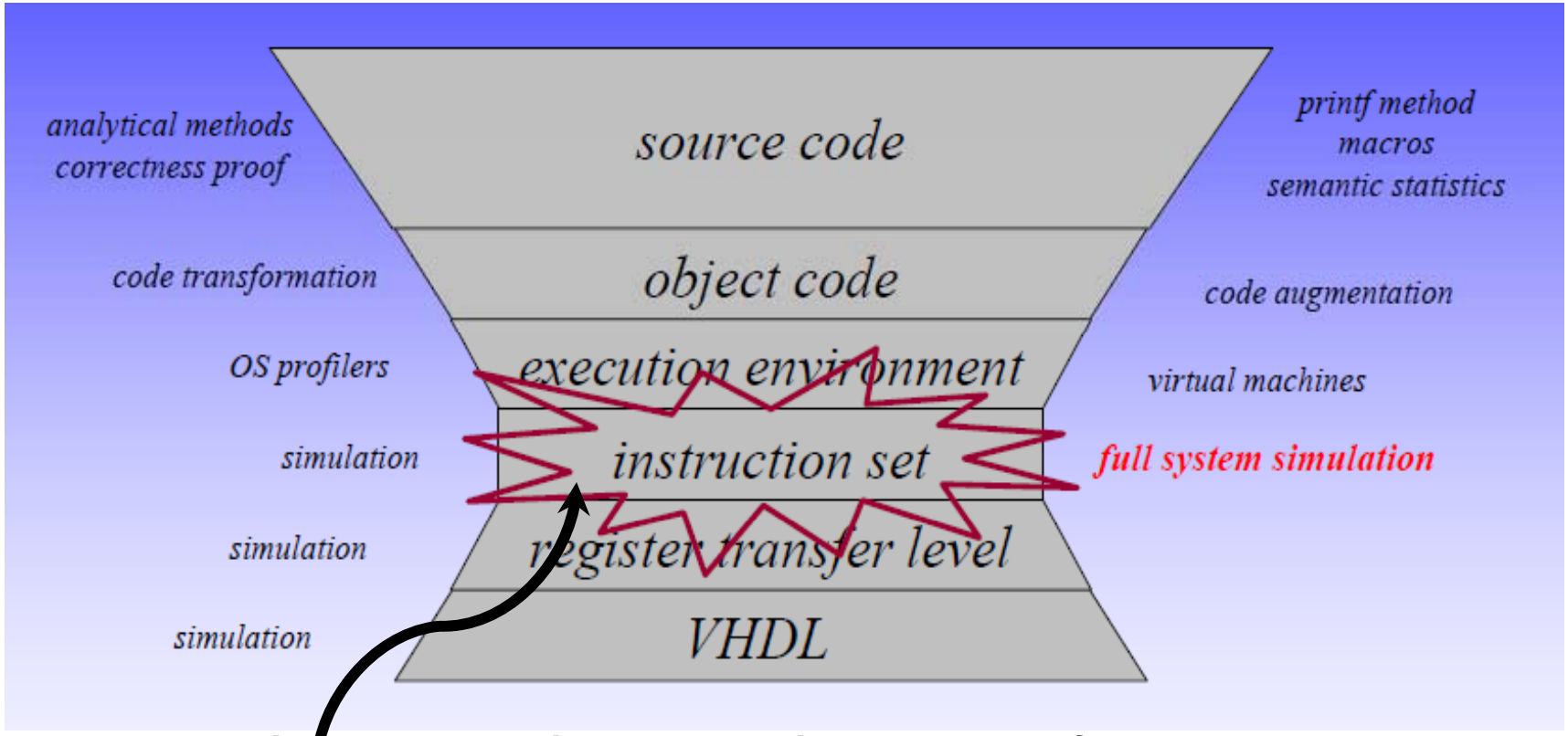
- Χρήση υπαρχόντων μηχανημάτων
  - Μεγάλο κόστος (πχ. Sun SPARC Enterprise T5120 server (64 threads, 128GB mem) ~ 12.800 $)
  - Αδυναμία παρέμβασης στο υλικό τους (συγκεκριμένο pipeline, caches, interconnection network)
  - Περιορισμένη δυνατότητα παρακολούθησης και μετρήσεων (π.χ. performance counters : λίγοι, μικρό documentation)
  - Περιορισμός στο σήμερα. Πώς μελετάς μελλοντικές αρχιτεκτονικές (π.χ. chip με 100 ή 1000 threads;)
- Λύση : Simulation (προσομοίωση)

Προσομοίωση Αρχιτεκτονικής (1)

- Απαιτήσεις
  - Γενικότητα (Generality)
    - » Μπορεί το εργαλείο να αναλύσει τα workloads?
    - » Parallel Systems, Multithreading, Multiple address spaces, OS code, Network Systems, etc.
  - Πρακτικότητα (Practicality)
    - » Μπορεί το εργαλείο να χρησιμοποιηθεί αποδοτικά;
    - » Host assumptions, compiler assumptions, OS modifications, workload language assumptions
  - Εφαρμοσιμότητα (Applicability)
    - » Μπορεί το εργαλείο να απαντήσει στα ερωτήματα μας;
    - » Restricted state that can be monitored, restrictions on parameter visibility, restricted length of observations.

Προσομοίωση Αρχιτεκτονικής (2)

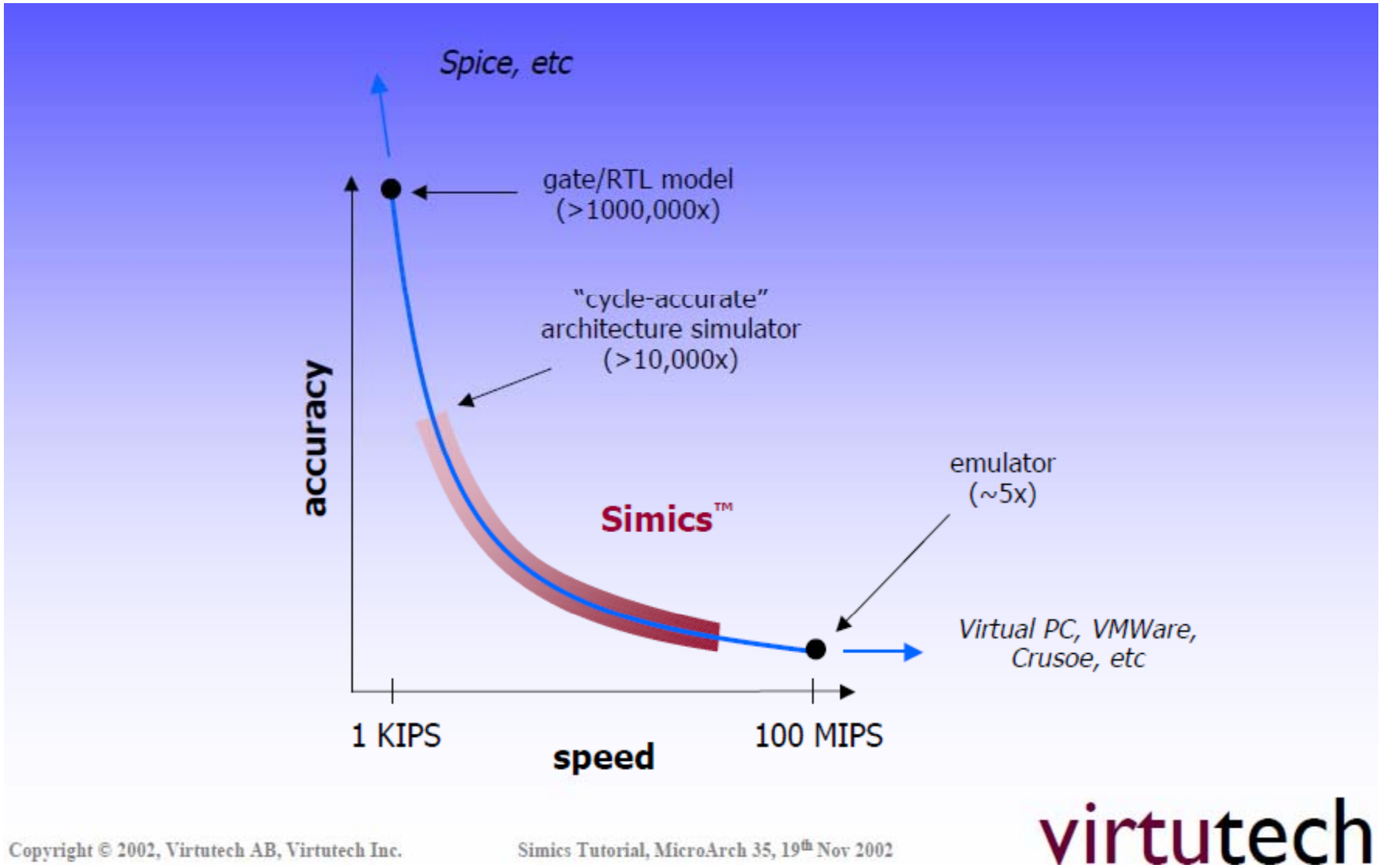
- Προσομοίωση = SW studying SW



- Το σημείο στο οποίο συναντώνται οι software και hardware engineers. Είναι το πιο χαμηλό επίπεδο στο οποίο έχει πρόσβαση το software και ταυτόχρονα είναι το πιο σταθερό (stable), καλύτερα ορισμένο (best defined) και λιγότερο πολύπλοκο (least complicated) επίπεδο.

Προσομοίωση Αρχιτεκτονικής (3)

- Ταχύτητα (Speed) vs. Ακρίβεια (Accuracy)



Προσομοίωση Αρχιτεκτονικής (4)

- Πλεονεκτήματα
  - Early availability
  - Ευκολία χρήσης
    - » Πλήρης διαφάνεια και ευκολία παρακολούθησης και μετρήσεων
    - » Διαφορετικά επίπεδα λεπτομέρειας και ακρίβειας
      - Pipelines, caches, branch predictors, …
      - Hardware devices (timer, drives, cards, …)
    - » Έλεγχος καινοτόμων προτάσεων/ιδεών
  - Κόστος
    - » Open source (Free)
    - » Academic licenses (Free ή μικρό κόστος για support)

Προσομοίωση Αρχιτεκτονικής (5)

- Προκλήσεις
  - Χρόνος ανάπτυξης των μοντέλων (modeling time)
  - Έλεγχος ορθότητας μοντέλων (validation)
  - Ταχύτητα προσομοίωσης
- Active research field
- Πληθώρα επιλογών
  - Virtutech Simics (x86, SPARC, MIPS, Leon, …)
  - AMD SimNow (x86)
  - Simplescalar (Alpha)
  - SMTSIM (Alpha SMT)

Πλατφόρμες Προσομοίωσης

- Διάφοροι τύποι προσομοιωτών
  - Trace-driven vs. Execution-driven
  - Cycle-level accurate vs. Functional models
  - Full system simulators (π.χ. Simics)

- Διάφοροι τύποι προσομοίωσης
  - Πλήρης εκτέλεση εφαρμογής
  - Χρήση στατιστικών μεθόδων

Functional vs. Timing Simulation

- Functional Simulation
  - Προσομοίωση της λειτουργικότητας των εντολών (instructions semantics and functionality)
  - Μεταβολή του state (registers, memory, …)
  - Σωστό program output
- Timing Simulation
  - Functional simulation
  - Λεπτομερής υλοποίηση των διαφορετικών δομών που χρησιμοποιούνται
  - Χρονισμός γεγονότων, προκειμένου να υπολογισθεί ο χρόνος εκτέλεσης του προγράμματος
- Functional simulation πολύ πιο γρήγορο

Full System Simulator

- Πολλοί διαδεδομένοι προσομοιωτές (π.χ. Simplescalar) προσομοιώνουν μόνο τον κώδικα της εφαρμογής που επιλέγει ο χρήστης
  - Έλλειψη OS
  - Hack για την προσομοίωση system calls
- Full system simulators
  - Ρεαλιστικοί
  - Εκτέλεση πραγματικών εφαρμογών
  - Προσομοίωση OS (π.χ. Simics boots Linux)
- Ακρίβεια (accuracy) ;
- Ταχύτητα ;

Παράδειγμα χρόνων προσομοίωσης

- spec2k with gcc and small inputs

Case	Time (sec)	Ratio to "native"	Ratio to "functional"
Native	1.054	1	-
sim-fast	167	158	1
sim-outorder	4,247	4,029	25
simics (bare)	461	437	1
simics w/ ruby	41,245	39,131	89
simics w/ ruby+opal	155,621	147,648	338

Measured by Lei Jin on *antimony* (3.8GHz Xeon w/ 8GB memory)

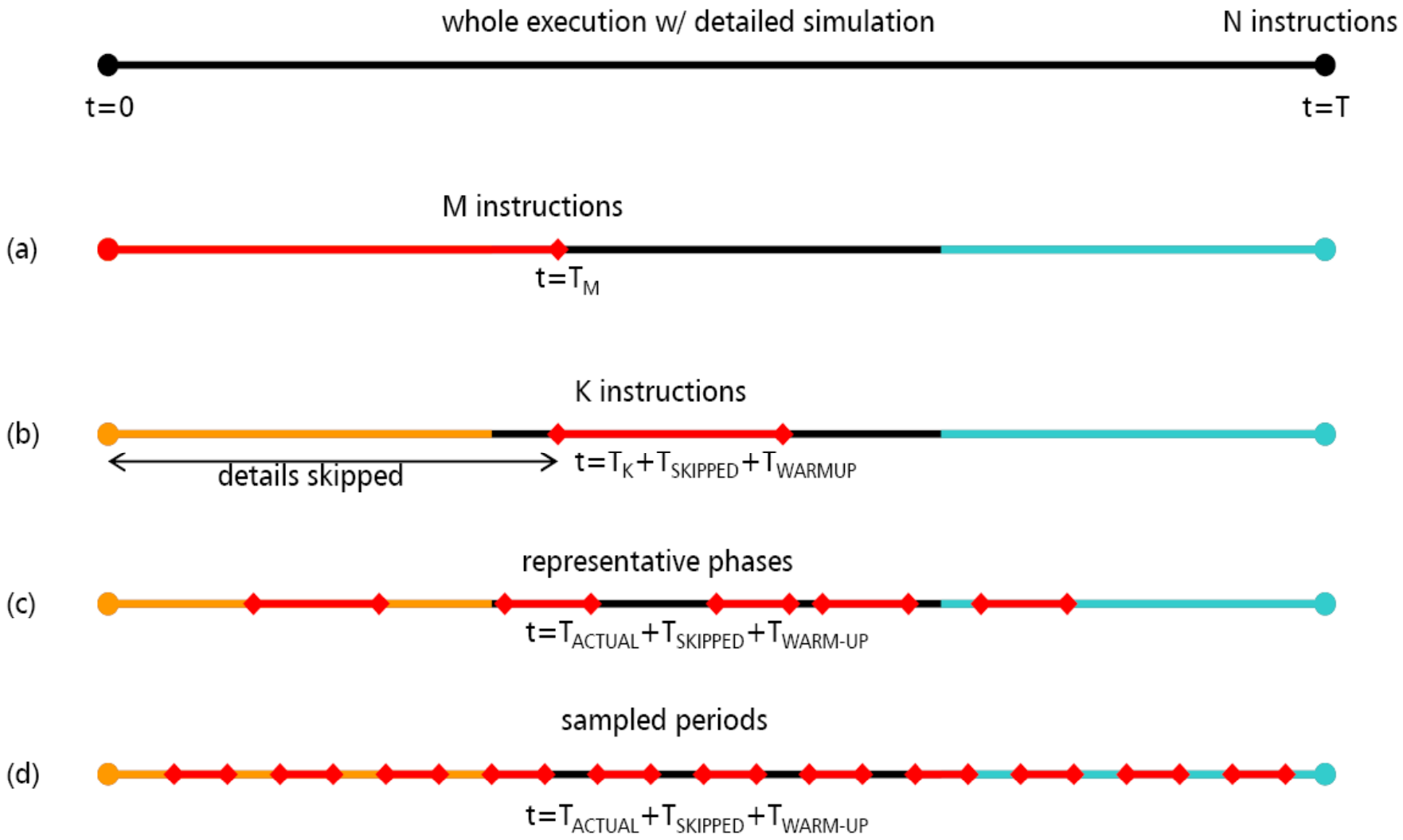
Επιλογή περιβάλλοντος προσομοίωσης

- Κριτήρια Επιλογής
  - Modularity simulator
  - Extensibility simulator
  - Επίπεδο ακρίβειας simulator
  - Ταχύτητα simulator
  - Μέγεθος του design space που θέλουμε να μελετήσουμε
  - Επιλογή κατάλληλων benchmarks

Στατιστικά Προσομοίωσης

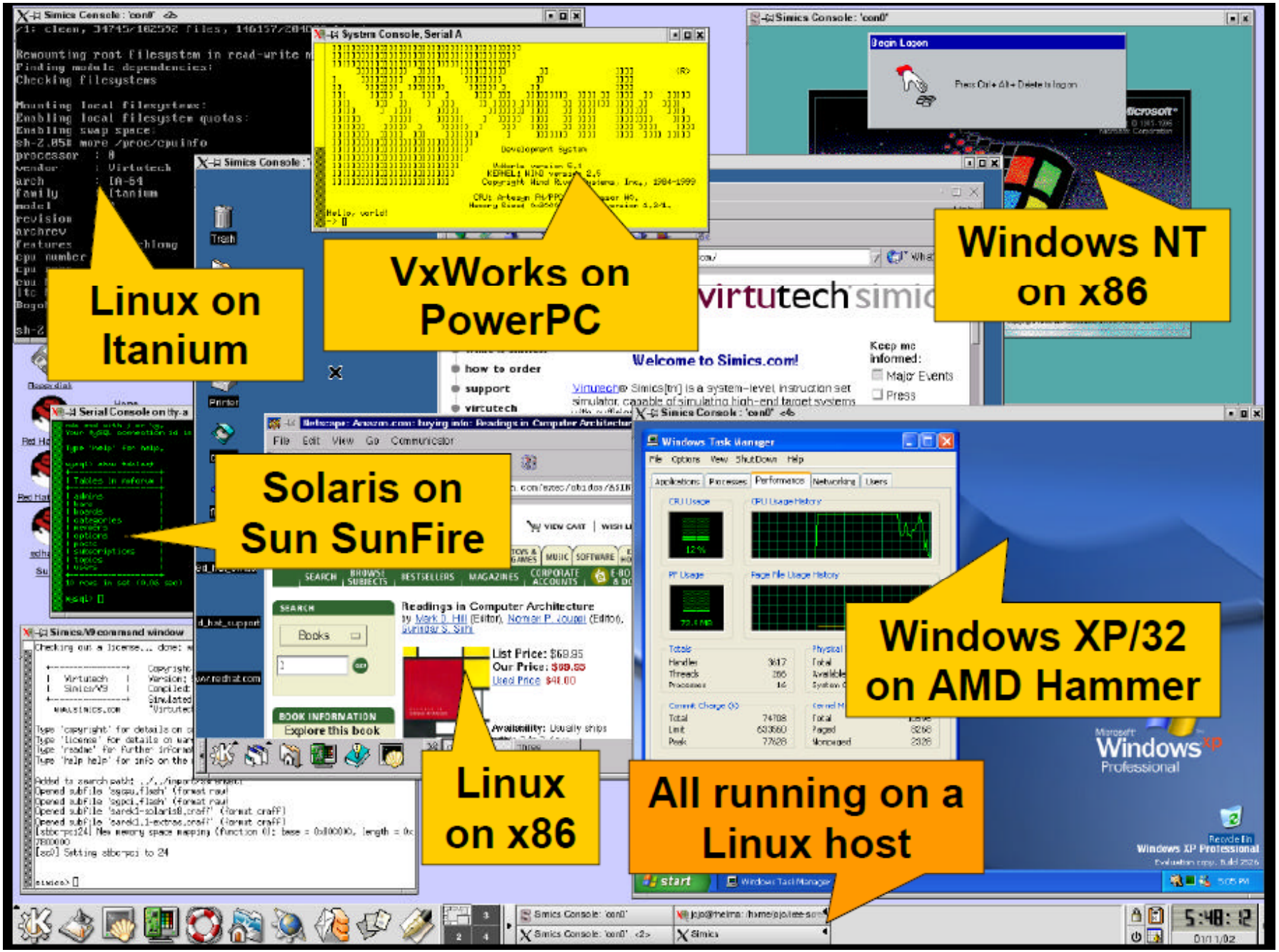
- Ο σκοπός ενός timing simulation είναι η συγκέντρωση πληροφοριών και μέτρηση διαφόρων μεγεθών
  - IPC
  - Memory access cycles
  - On-chip network contention
- Τα προγράμματα παρουσιάζουν διαφορετικές φάσεις
  - Initialization phase
  - Main phase
  - Wrap-up phase
- Πότε παίρνουμε τα στατιστικά που μας ενδιαφέρουν;

Στατιστικά Προσομοίωσης



Simics

- Full system simulator
  - Υποστήριξη διαφορετικών συστημάτων (x86, Sun, ARM, …)
  - Εκτέλεση OS και workloads χωρίς προσθήκες/αλλαγές
- Εύχρηστο interface για microarchitecture modeling
  - Ο Simics παρέχει το functional simulation και ο χρήστης αποφασίζει για το timing των διαφόρων γεγονότων
  - Υλοποιήσεις cache memories
  - Ενορχήστρωση για συγκέντρωση πληροφοριών κατά το run time
- Ευρέως αποδεκτός στο architecture research community
- Θα τον χρησιμοποιήσουμε στις ασκήσεις
  - Academic license for NTUA


Linux on Itanium
VxWorks on PowerPC
Windows NT on x86
Solaris on Sun SunFire
Linux on x86
All running on a Linux host
Windows XP/32 on AMD Hammer

Simics

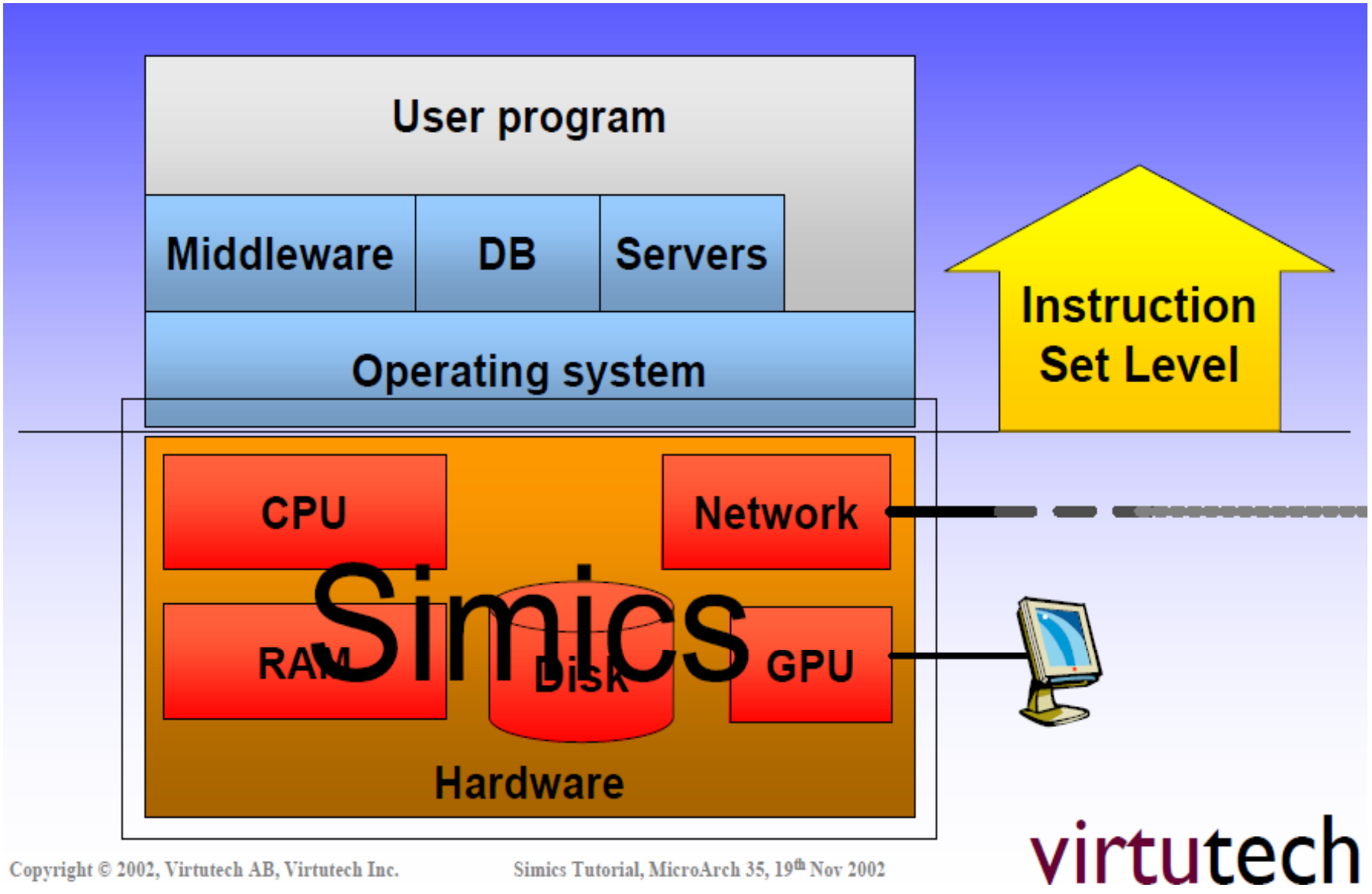
- Instruction Set Level Simulation
  - Οι πραγματικές εφαρμογές παίρνουν ώρες σε πραγματικά μηαχνήματα
  - Χρειαζόμαστε αρκετά μεγάλες ταχύτητες για να τρέξουμε ένα σημαντικό κομμάτι αυτών των εφαρμογών.

	Number of Ops (B)
Windows XP Boot	5
Linux RH 6.0 Boot	4
Windows XP Install	361
SPECint2000 (train)	279

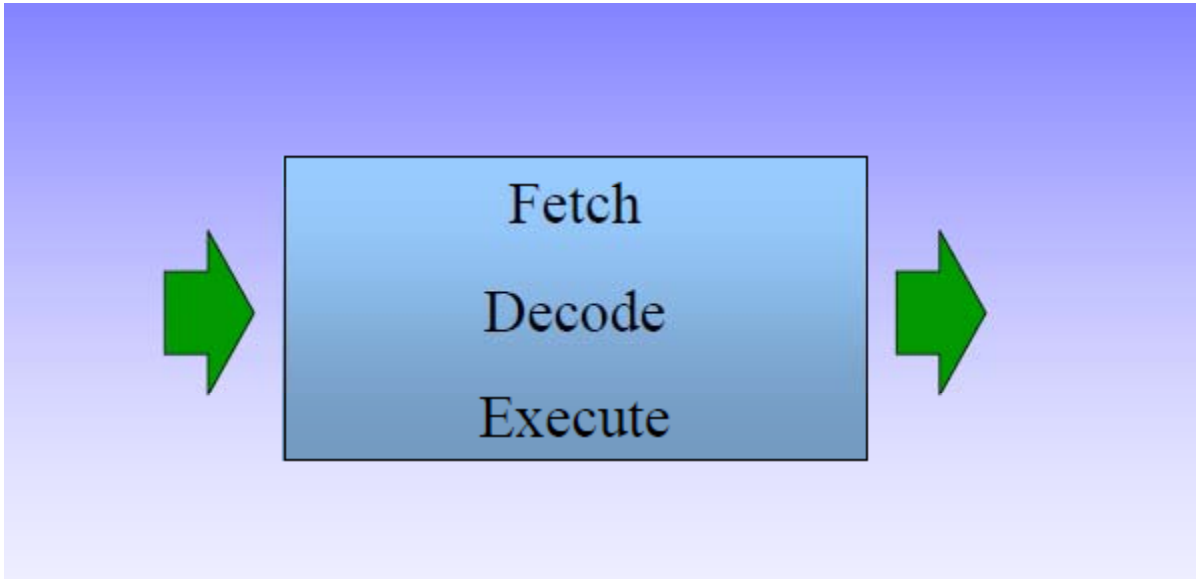
Simics Tutorial, MicroArch 35, 19th Nov 2002

virtutech

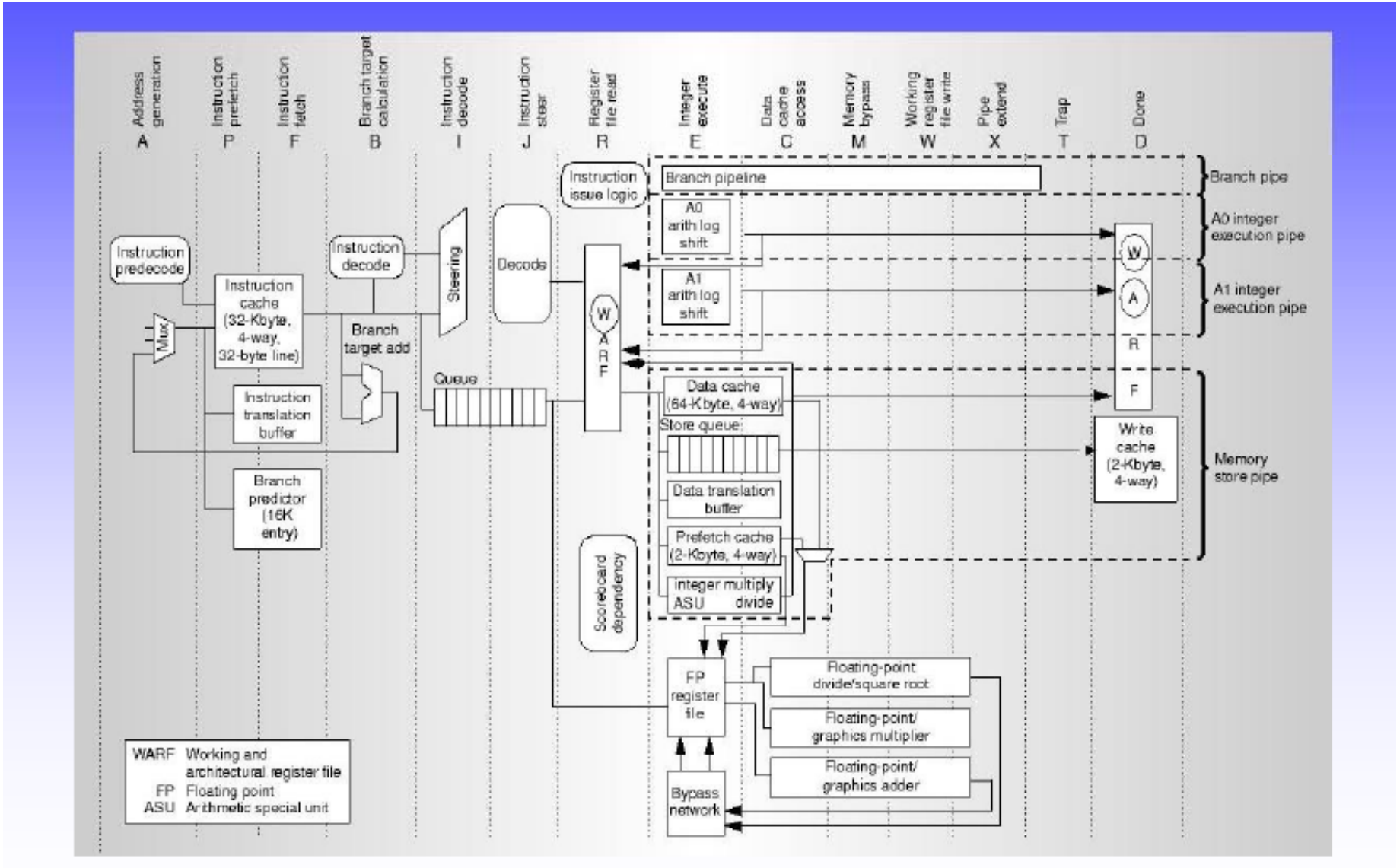
Simics


User program
Middleware
DB
Servers
Operating system
Instruction Set Level
CPU
Network
Simics
RAM
Disk
GPU
Hardware
virtutech
Copyright © 2002, Virtutech AB, Virtutech Inc.
Simics Tutorial, MicroArch 35, 19th Nov 2002

Simics Pipeline



Ultra SPARC III Pipeine



Simics

- Simics → Functional Simulation
  - Instruction Set semantics
  - Devices
- User → Timing
  - Ο χρήστης μπορεί να καθορίζει το timing.
  - Υπάρχουν modules τα οποία παρέχουν αυτή τη δυνατότητα.
    - » GEMS Ruby → Memory Hierarchy
    - » GEMS Opal → Processor pipeline
    - » SimFlex → Memory Hierarchy

Simics Terminology

- Host machine
  - Το μηχάνημα/OS στο οποίο τρέχει ο Simics
- Target machine
  - Το μηχάνημα/OS το οποίο προσομοιώνει ο Simics
- Δεν απαιτείται η αρχιτεκτονική και το OS των 2 μηχανημάτων να είναι τα ίδια
  - Compile σε διαφορετικά μηχανήματα!
- Steps vs. cycles vs. instructions

Simics Environment

- Command line interface, παρόμοιο με το gdb
- Scripting
  - Δυνατότητα scripting και σε Python
- Ανάπτυξη μοντέλων (C, Python)
- Checkpointing
- Διαφορετικοί τρόποι προσομοίωσης
  - Fast, stalls, MAI
  - Speed vs. accuracy

Simics Major Components

- Functional
  - Modules
    - » C, Python, DML
    - » Devices, components, boards, machines…
  - Ενεργοποιούνται με Simics ή Python scripts
- Timing
  - Memory, caches, Simics MAI
  - Ο χρήστης ορίζει τις καθυστερήσεις του κάθε module

Simics Demo