SECTION 1: INTRODUCTION TO SIMICS
Familiarity survey

- C
- Python
- gdb
- Unix/Linux/Solaris
What are we doing in labs?

- Giving you an environment to:
  - Run code on a variety of platforms
    - Not all of which are actually available to us
  - Benchmark and experiment
  - Change things about the hardware
    - Access to multiple abstraction layers
  - See how architectural mechanisms work in practice on real software
What is Simics?

- Efficient, instrumented, system level instruction set simulator
  - Run as fast, or faster than, target machine
  - Gather detailed information at run time
  - Model target at level at which OS acts
  - ISA-aware, simulates each instruction
  - Runs unmodified OSes and workloads
Why are we using Simics?

- Scripting capabilities
- Academic licensing
- Can run real software, quickly
- Intro to functional/timing simulators
- Outside relevance
  - Program analysis, computer architecture research, and kernel debugging
- RAMP isn’t ready yet
Terminology

- **Host machine**
  - Machine/OS on which Simics is running

- **Target machine**
  - Machine/OS which Simics is simulating

- Neither the architecture nor the OS of either machine need be the same

- **Steps vs. cycles vs. instructions**
Environment

- Similar to gdb, command line interface
- Simics CLI has built in scripting
  - Can also write scripts in Python
- Checkpointing
- Different modes of execution
  - Fast, stalls, MAI
  - Speed vs. accuracy
Major Components

- **Functional**
  - **Modules**
    - Written in C, Python, DML
    - Devices, components, boards, machines...
  - Attached by Simics or Python scripts

- **Timing**
  - Memory, caches, Simics MAI
  - Declare or calculate delay of modules
Demo!
Gritty details

- Checkpoints will save time, but use wisely
  - They can be big
  - Later checkpoints will save as deltas to previous one
- Might compile code in separate environment
  - E.g. compile on Solaris/SPARC, run on Linux
- Need X11 at client machine
  - http://inst.eecs.berkeley.edu/connecting.html#xwindows
- Instructional servers
  - http://inst.eecs.berkeley.edu/cgi-bin/clients.cgi?choice=353cory
Scripting

- Not required, but can make life easier
- Telling simics what to do
  - Simics interface: .simics
  - Python interface: .py (or @ at start of .simics line)
- Disconnecting while it works
  - Read up on nohup or screen
- Don’t abuse the machines!
Newsgroup
- ucb.class.cs152
- news.csua.berkeley.edu
- Remember posts from Google groups aren’t visible outside Google
- Can use EECS web UI:
  - http://inst.eecs.berkeley.edu/webnews