Simics Brief Introduction

Presented By: Chunrong Lai

Software Engineer

Simics Technology Center (STC), Software and Services Group

26 Oct, 2012

What is Simics?

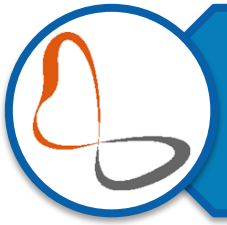
Simics is a ***high-performance, full system*** simulator used by software developers to simulate ***large and complex*** electronic systems.

- Simulate any size of target system
- Run unmodified binaries
- Other use cases as add-ons

Simics allows you to **break the rules** of embedded systems development

Simics History

Research project at SICS started in 1991

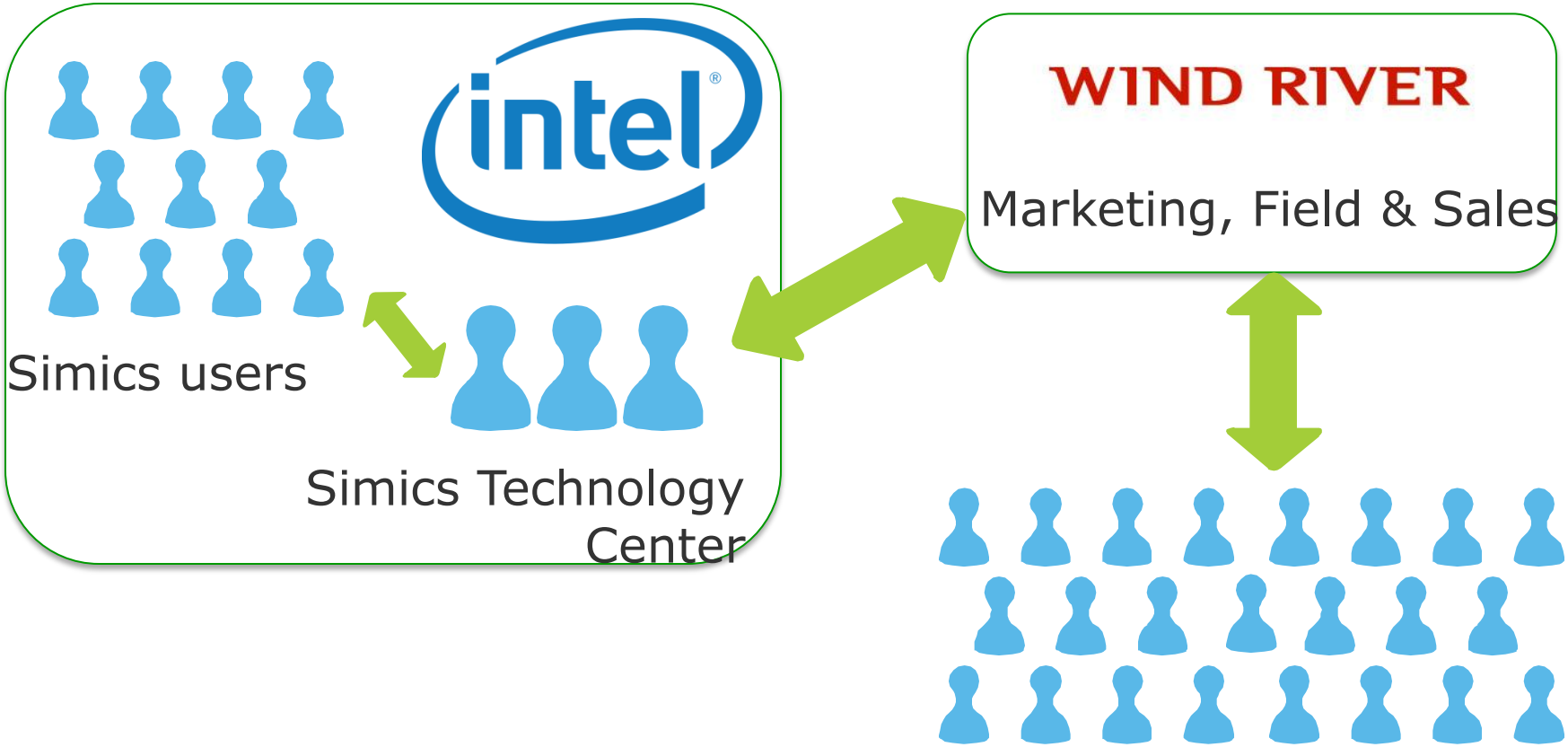


Virtutech was founded in Sweden in 1998 and incorporated in USA in 2003.

Acquired by Intel in 2010

Marketed and sold through Wind River (subsidiary)

Simics Customers



Widley used acroos different Intel groups/projects as an uniform software platform simulator

Simics External Customers, include network, communication, space, semiconductor and HPC field

What Wind River Simics Customers Say

Computing Server

- Shortens bring-up time from 22 weeks to 29 days

Networking

- Finds and solves problems in 30 minutes instead of three weeks and three people
- Configures complex labs in minutes instead of weeks

Industrial

- Prevents the loss of $10M by getting the ASIC right before manufacturing

Space

- Saves $6M in target hardware labs

Some Simics Features and Capabilities

- High performance
- Synchronized System Stop
- Save/restore of simulated state
- Repeatability
- Dynamic reconfiguration
- Large systems, hundreds of processors
- Runs all software unmodified
  - Windows, Linux, VxWorks, Hypervisor, etc
- Heterogenous systems
- Run simulation in reverse
- C/C++ debugging
- Code coverage
- Trace generation
- Non-intrusive inspection
- Real-world connections
- Hardware-in-the-loop
- Record/replay of user input
- Connections to hardware emulators
- Gear shift to cycle accurate models
- Host virtualisation for native IA performance
- Binary translation for cross target simulation
- Synchronize virtual time with external tools
- Modeling language for fast development
- Models in any language
  - DML, C/C++, SystemC, Python
- Large collection of model interfaces
- User developed simulator features
- Integrates with external tools
- Scripting for automated sessions
- Operating System awareness
- Process tracking
- Instruction and data profiling
- Supported, stable, well-documented API
- Cache modeling
- Advanced memory breakpoints
- Distributed simulation
- DHCP, DNS, FTP, TFTP services
- Connections to remote debuggers
  - E.g. GDB
- IP-XACT import/export
- Import of SystemC models
- Build kit for system panels
- Huge model library

Important Features for Full System Simulation

Features work together to form "super features"

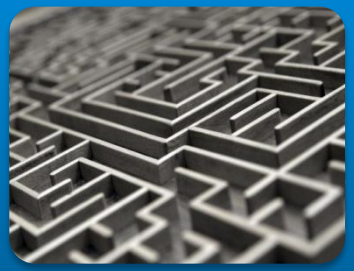
- checkpointing
- dynamic configuration
- repeatability and reversibility
- scripting
- visibility
- extensibility
- software abstraction

SIMICS USE CASES

The Challenges of Systems Development

Do More with Less

- Target hardware Issues
- Reduced staffing
- Shorter Time-To-Market



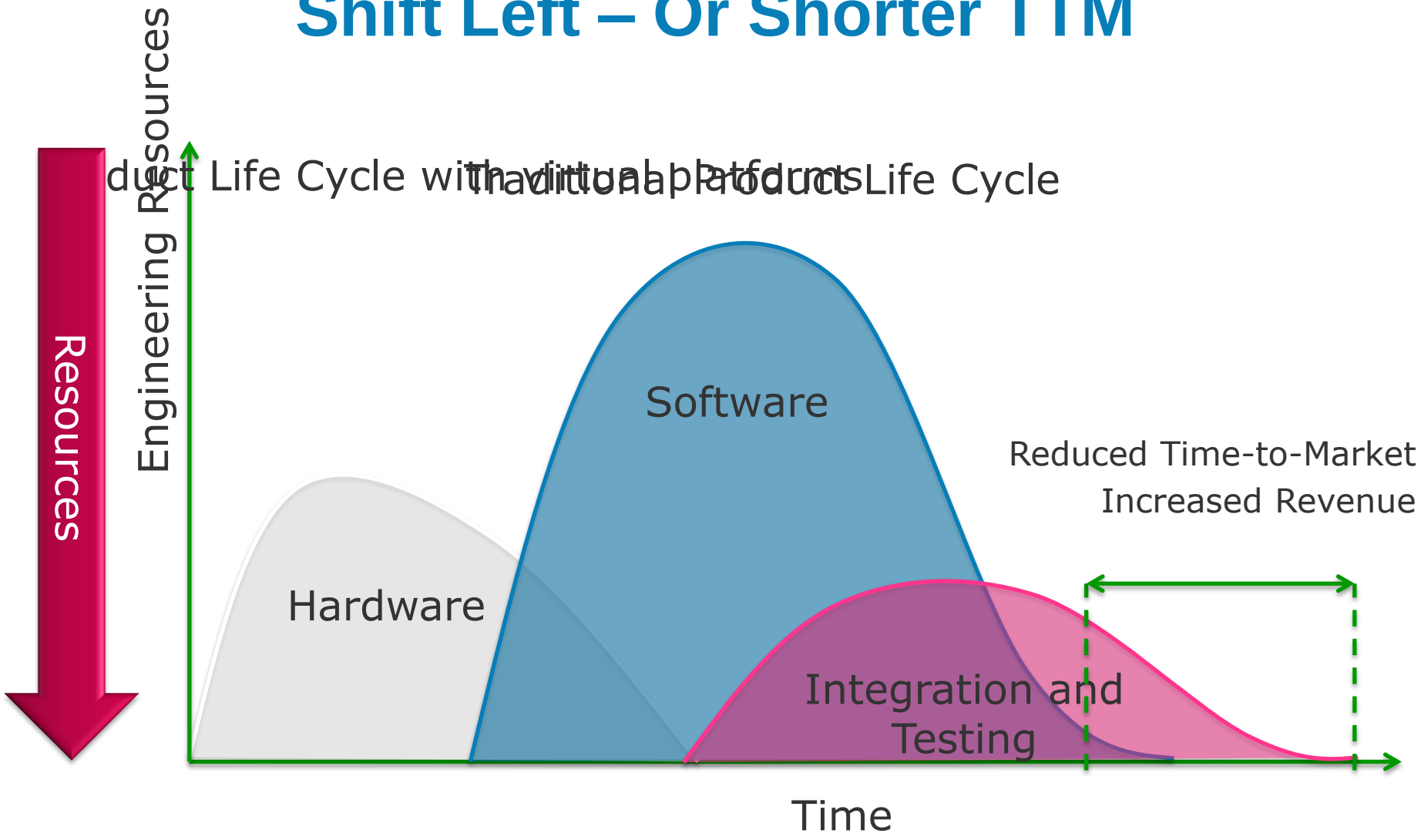
Growing Complexity

- Multi-core SoCs, accelerators
- Heterogeneous systems
- Networked, large systems

Global Teams

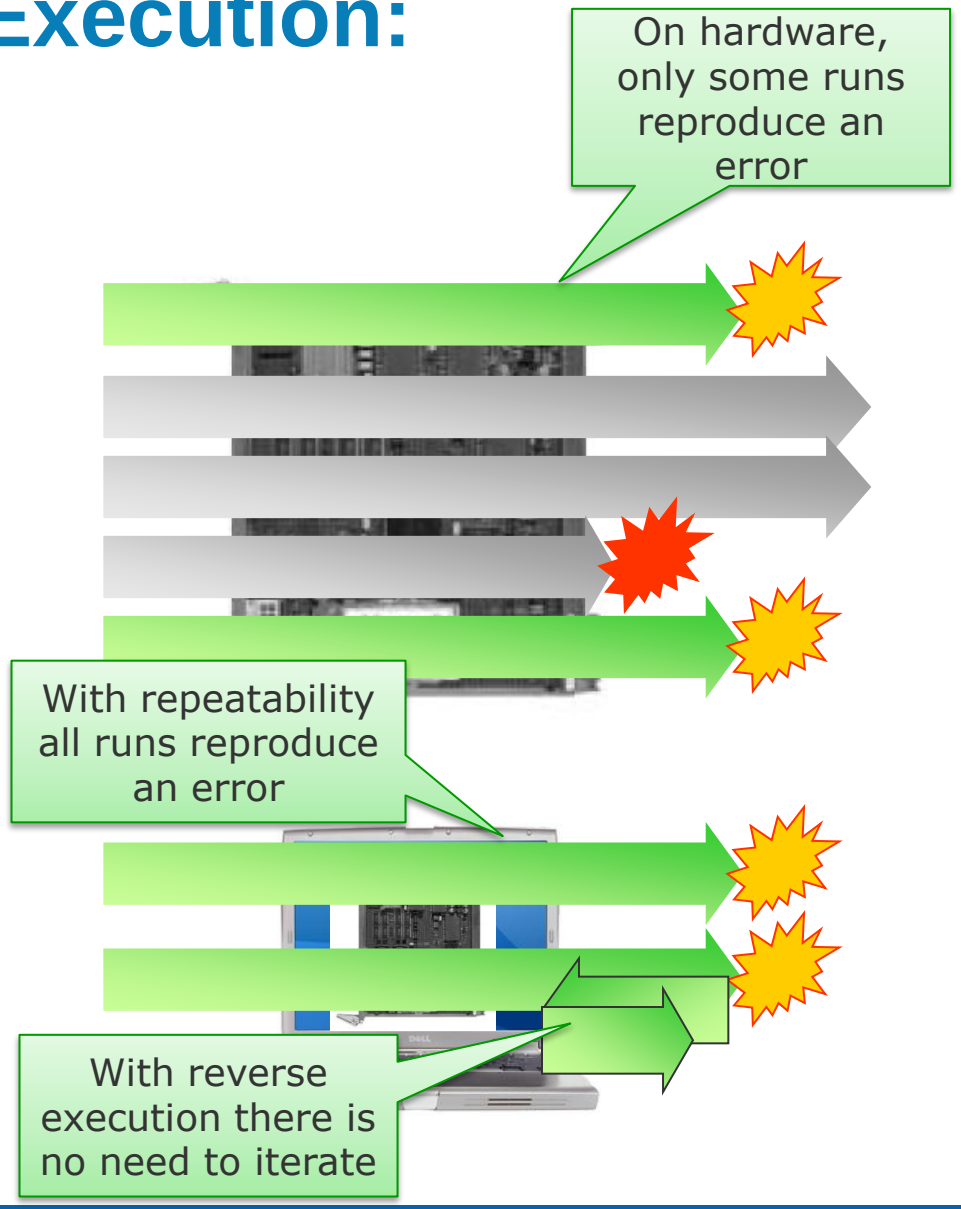
- Engineering teams are no longer co-located
- Multi-company collaboration
- Collaboration and communication challenges

Shift Left – Or Shorter TTM



Repeated/Reverse Execution: Do More with Less

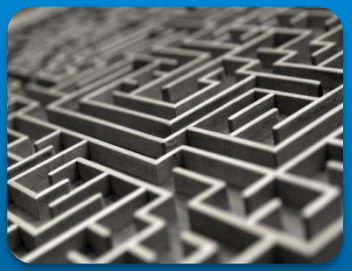
- Repeat any run trivially
  - No need to rerun and hope for bug to reoccur
- Stop and go back in time
  - Instead of rerunning program from start
  - Breakpoints and watchpoints backwards in time
  - Investigate exactly what happened this time
- This control and reliable repeatability is very powerful for parallel code!



The Challenges of Systems Development

Do More with Less

- Target hardware Issues
- Reduced staffing
- Shorter Time-To-Market



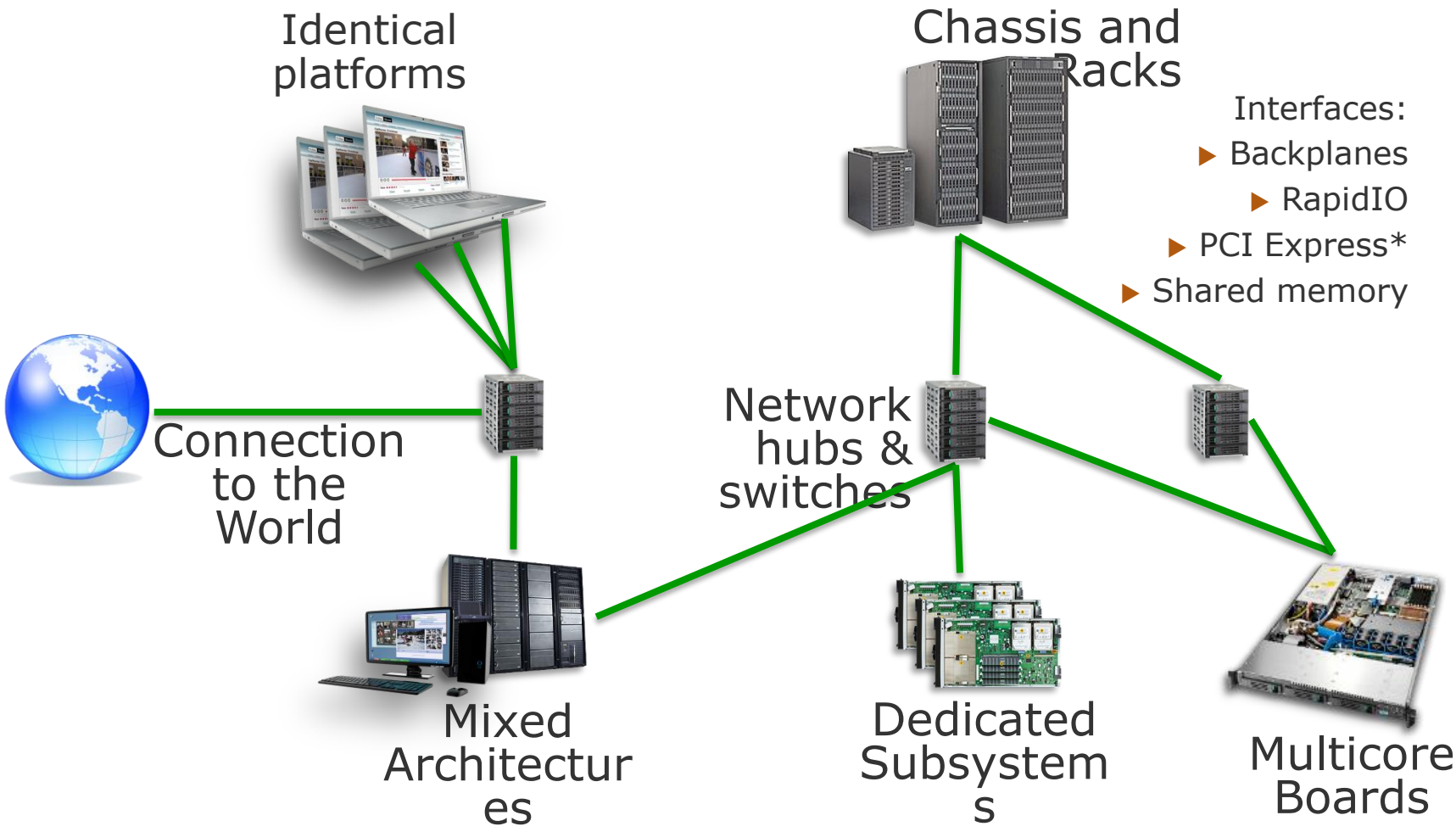
Growing Complexity

- Multi-core SoCs, accelerators
- Heterogeneous systems
- Networked, large systems

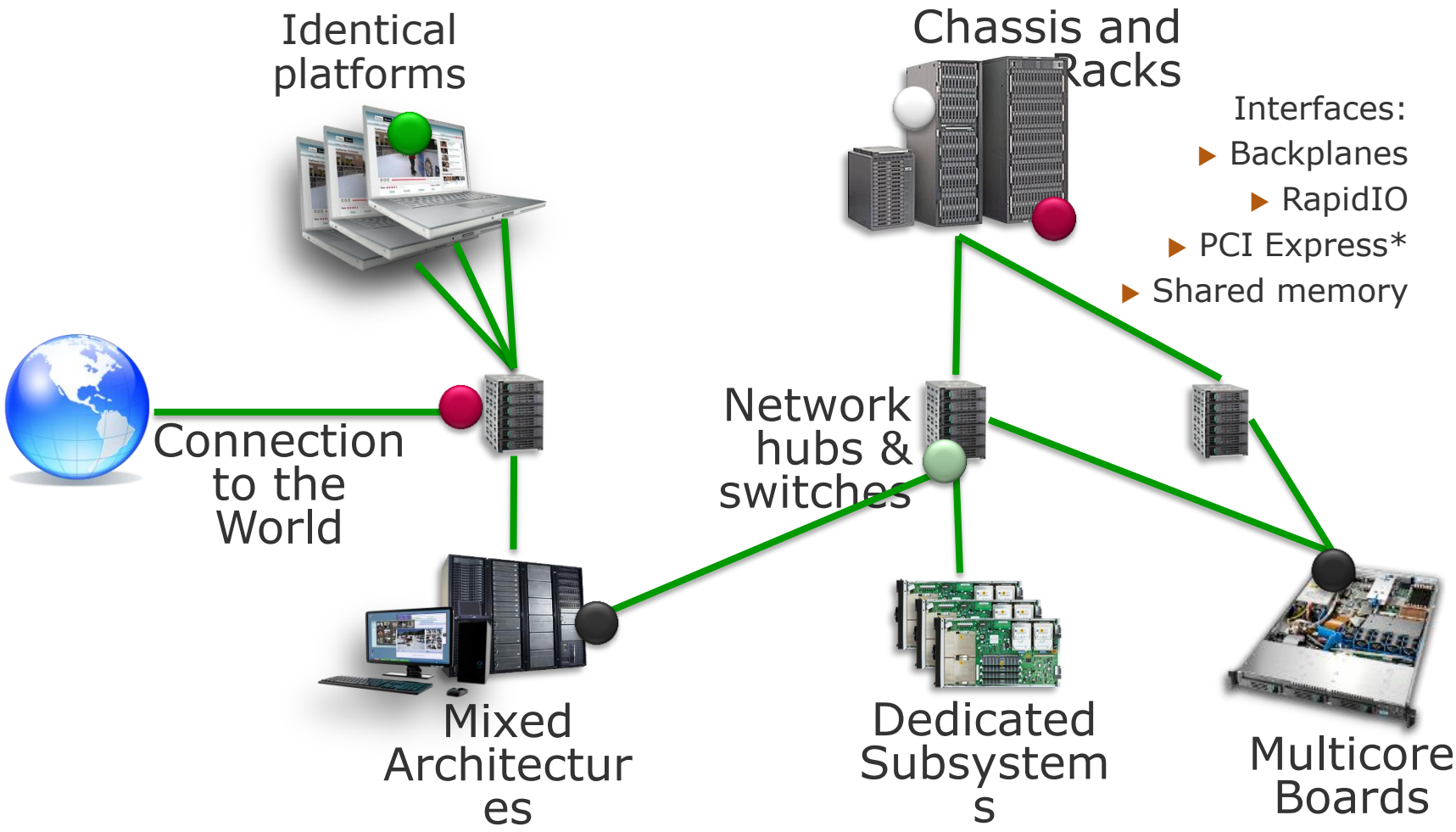
Global Teams

- Engineering teams are no longer co-located
- Multi-company collaboration
- Collaboration and communication challenges

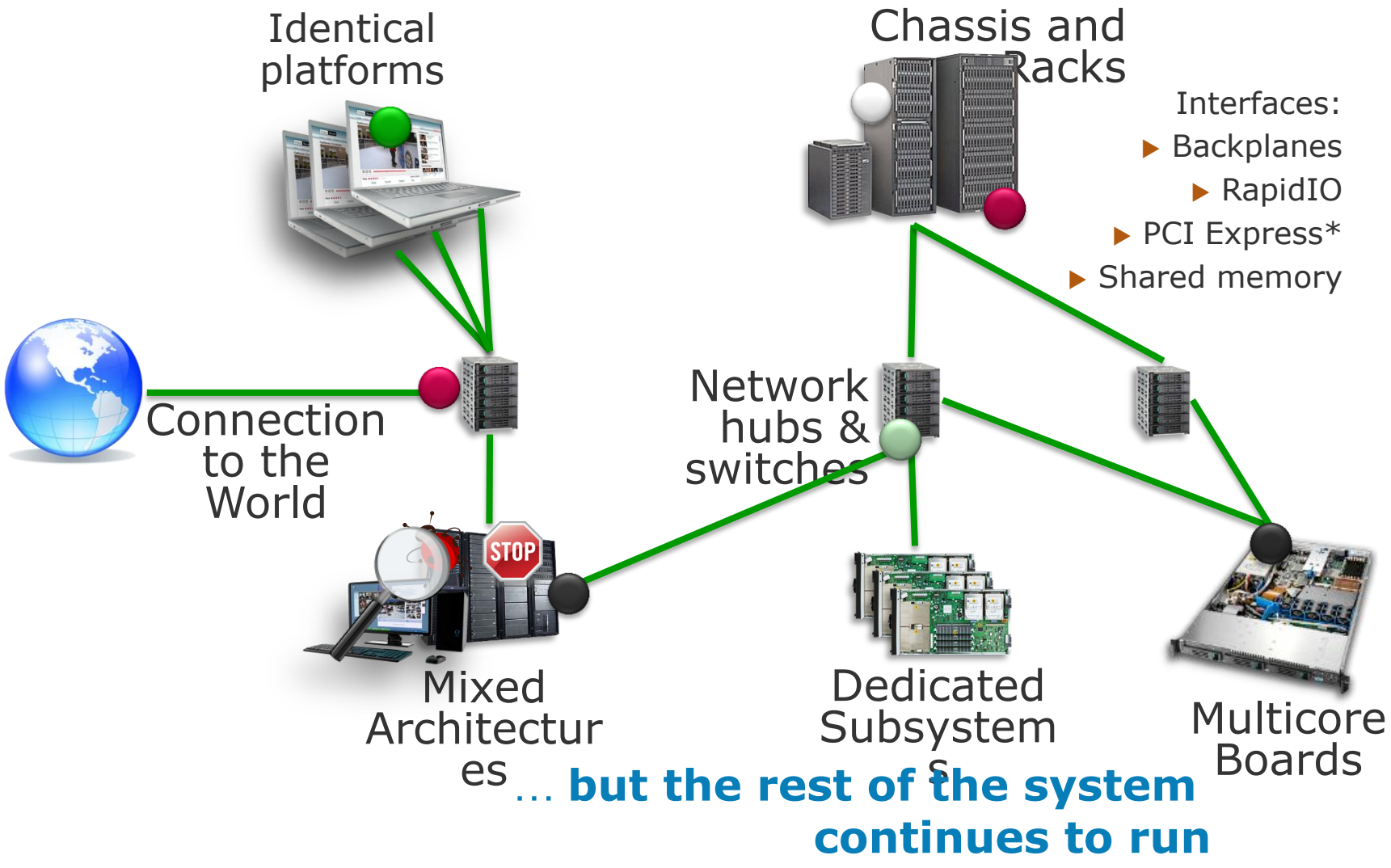
The System is a Network!



Control: A Lot Happens in a System

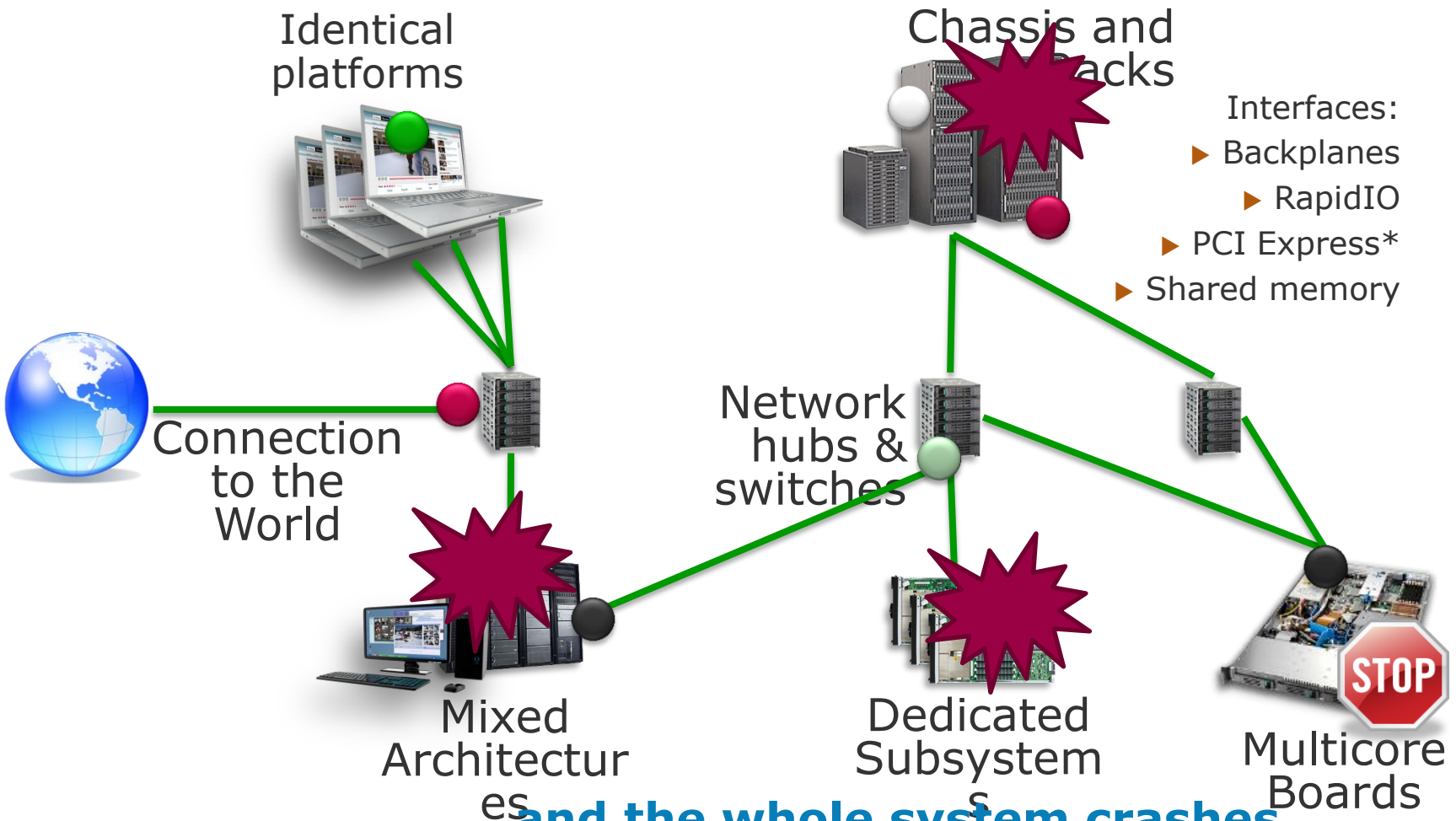


Control: A Single Component may stop …



… but the rest of the system continues to run

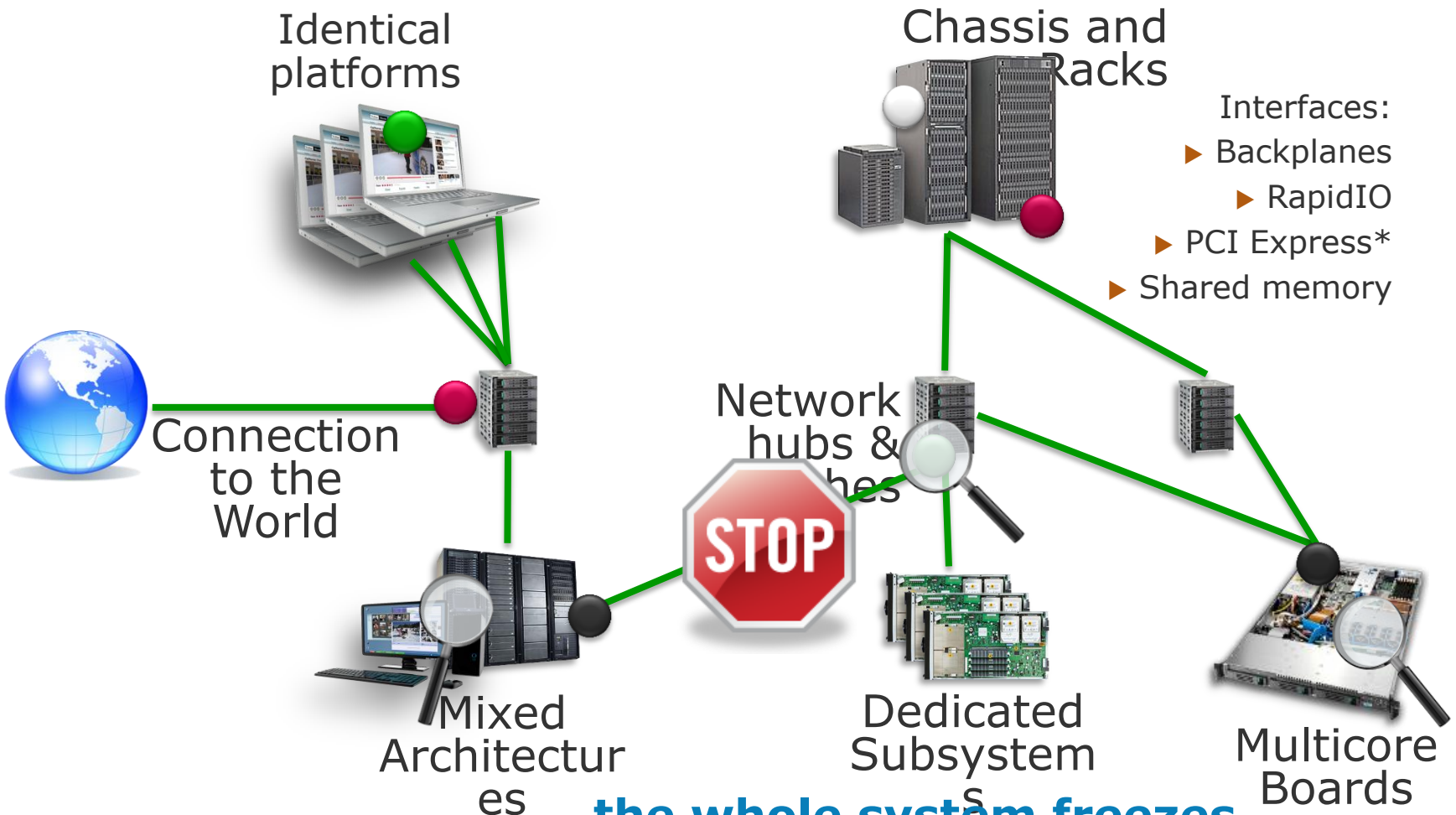
… or a Single Component Stops …



… and the whole system crashes.

But where is the bug?

Control: Simics Synchronized System Stop

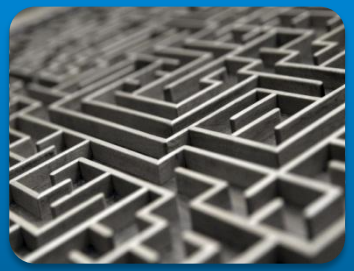


… **the whole system freezes in an operational state**

The Challenges of Systems Development

Do More with Less

- Target hardware Issues
- Reduced staffing
- Shorter Time-To-Market



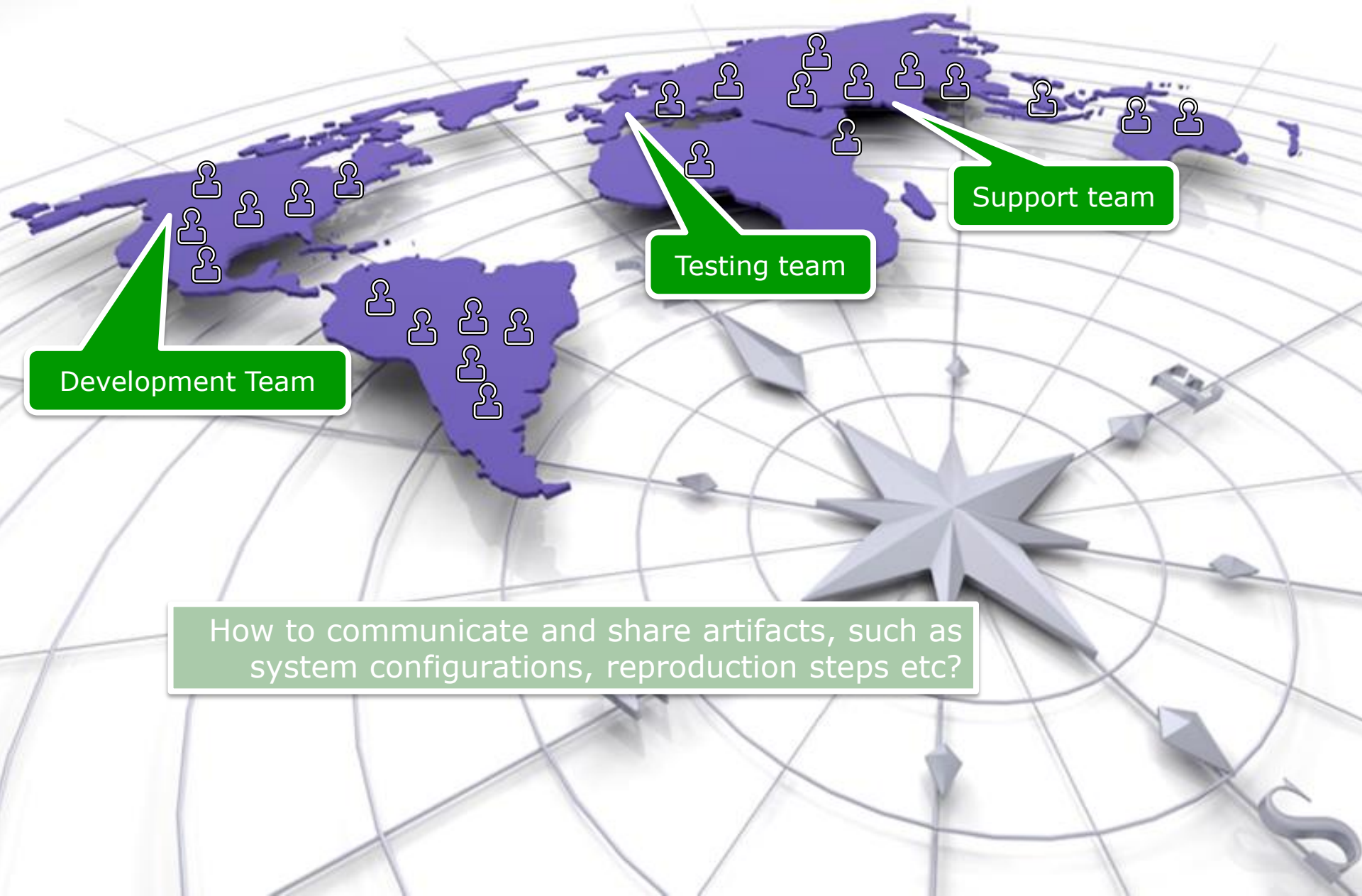
Growing Complexity

- Multi-core SoCs, accelerators
- Heterogeneous systems
- Networked, large systems

Global Teams

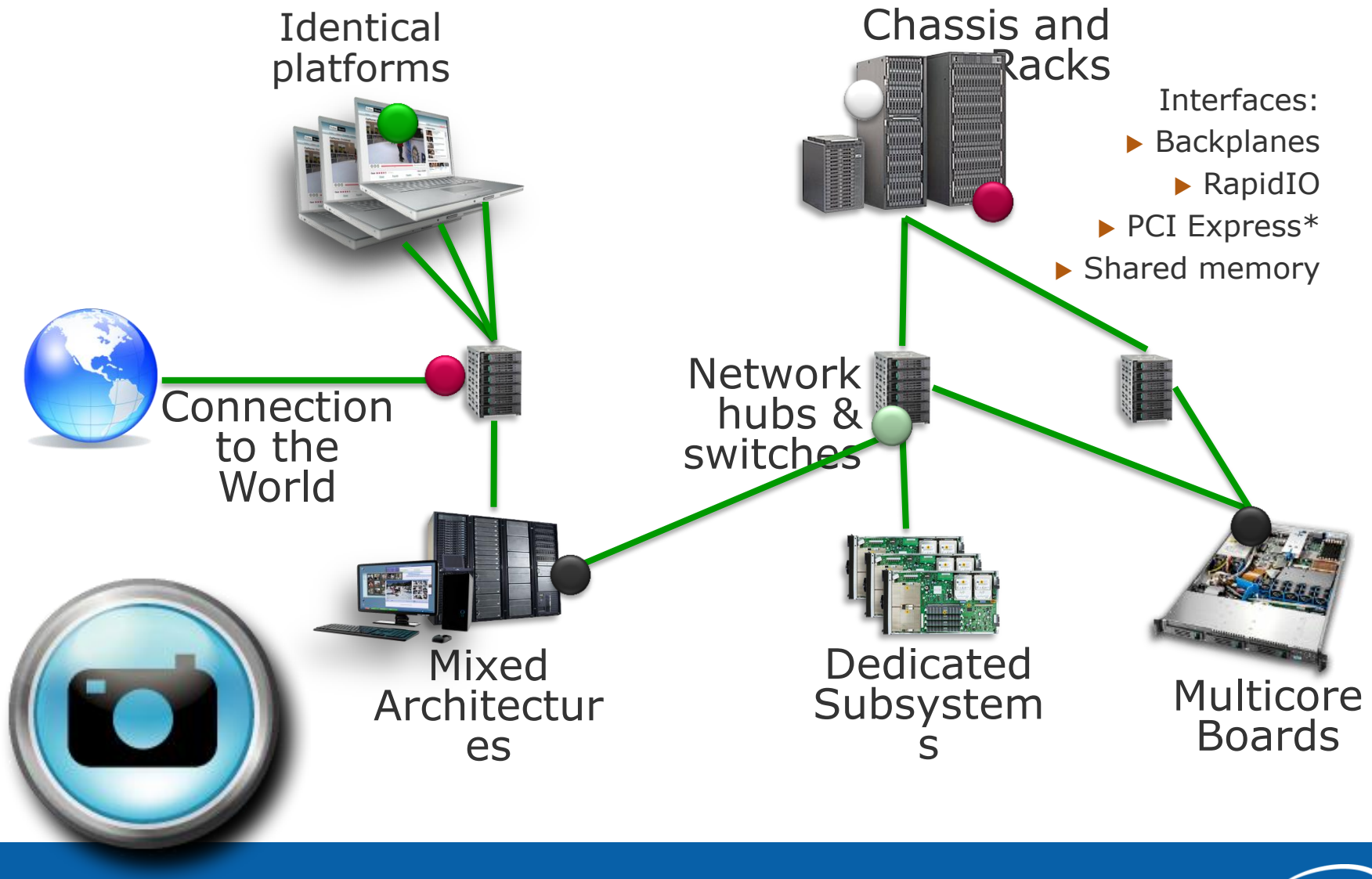
- Engineering teams are no longer co-located
- Multi-company collaboration
- Collaboration and communication challenges

Today's Teams are Geographically Dispersed

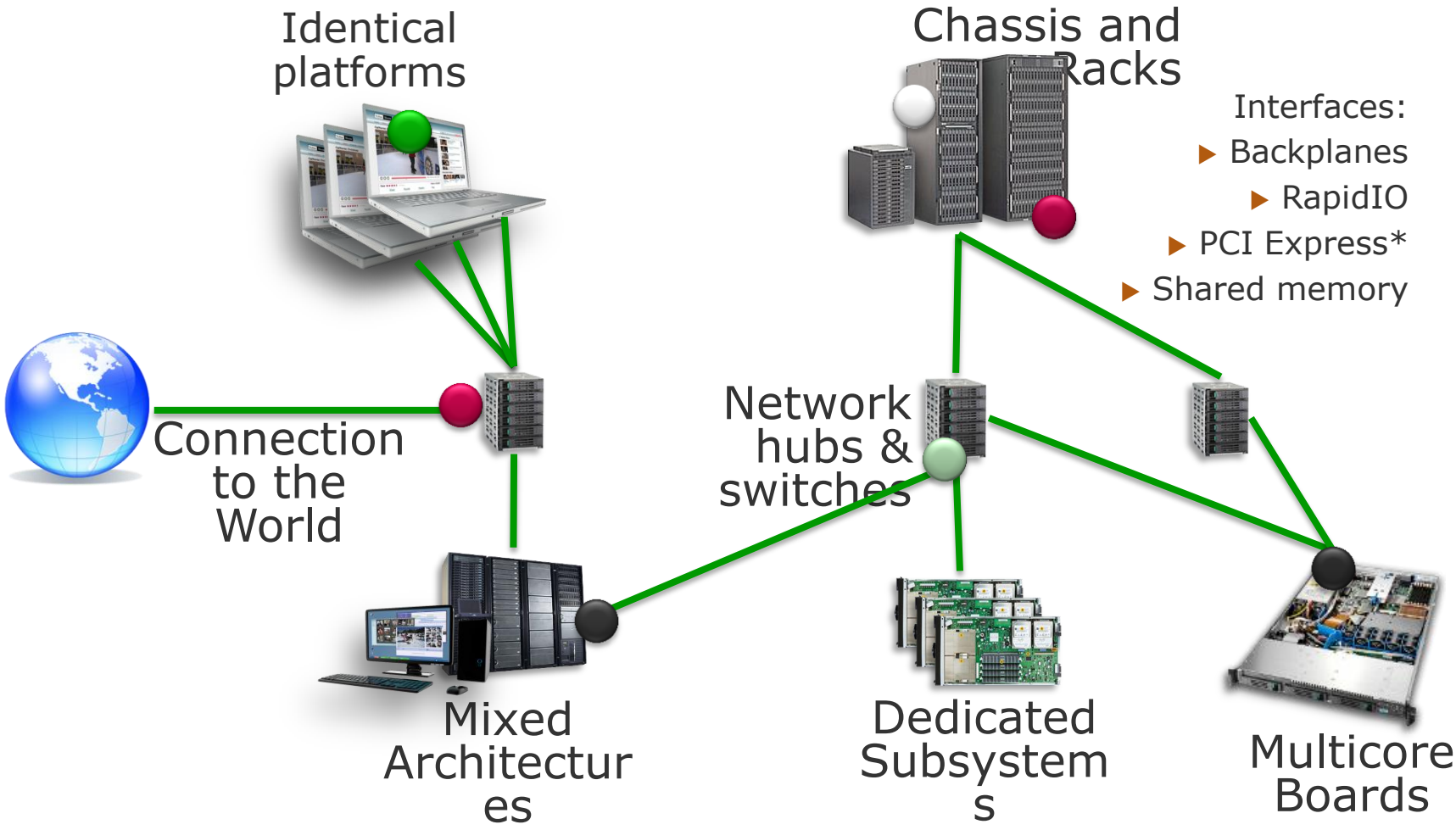


How to communicate and share artifacts, such as system configurations, reproduction steps etc?

Control: Taking a Check Point



Control: Resume



Checkpoint Collaboration

Simics Virtual Platform:
Exact configuration of the target system that a bug was found on.

Simics Script:
Automate the actions that led to a bug.

Simics Checkpoint:
A snapshot of the full system that can be restarted on any machine anywhere.

Testing Team
finds a bug

Development Team
Loads checkpoint and resumes execution to find the source of the bug.

Impact on Productivity

- Consider:
  - The time it takes to physically set up a system
  - The time it takes to boot a system
  - The time it takes to get a system to the exact state you want

- With Simics Full System Simulation:
  - Take a snapshot of it
  - Re-load it seconds later
  - Re-load it from any other place in the world!

- Start Simics:
  - http://www.windriver.com/products/simics/
  - Supported through http://www.simics.net
  - Contact: Luke.yang@windriver.com

SIMICS IN ACADEMIA

Computer Architecture

- Computer architecture research
  - Cache and memory hierarchies in MP systems
  - ACM Athena lecture "Shared Caches in MultiCores" in ISCA2010
- Add-ons
  - UW-Madison Multifacet GEMS
    - OOO processor models, memory hierarchies
  - CMU SimFlex
    - Timing-accurate processor, memory and interconnect
  - UIUC FeS2
    - Timing-first multiprocessor x86 simulator
- Teaching case:
  - UC Berkeley: Graduate Computer Architecture Spring 2012
  - UC Berkeley: **Computer Architecture and Engineering Spring 2012 (**undergraduate**)**

Operating Systems

- Teaching
  - Carnegie-Mellon University: **Operating System Design and Implementation**
- Research
  - useful simulate additional hardware support or novel hardware features, or novel hardware configurations and computer organizations.

Fault Injection, Reliable software

- Some publications using Simics for fault injection include
  - Myhrman and Svärd: *Studying Fault Injection in WCDMA Base Station Processors Using Simics Simulator*, MSc Thesis, Chalmers University of Technology, May 2005
  - Bastien, A Technique for Performing Fault Injection in System Level Simulations for Dependability Assessment, Master's Thesis, University of Virginia, January 2004.

Workload characterization

- Software profile
  - profiling the amount of time spent in the OS kernel vs. the time spent in user mode
- Cache behaviors
- Instruction traces

Academic Use

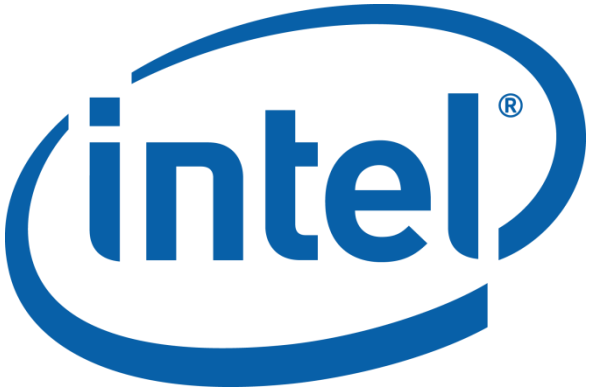
- http://blogs.windriver.com/engblom/
  - Blog posts about Simics
  - Information on Wind River academic program
  - Interviews with Simics users
- Projects by interviewed researchers
  - Improved HW support for virtualization
  - Multicore partitioning for avionic systems
  - Fault injection in reliable embedded systems

WindRiver Academic Program

- Always been available at low or no cost
- Available platforms
  - Simics 4.2, soon Simics 4.4
  - ARM, PPC, SPARC V8, SPARC V9, X86
  - Intel 440BX with several different processors, 486sx, Pentium II, Pentium 4, one x86-64 CPU
- Apply the usage
  - http://www.windriver.com/universities/
  - simics.university@windriver.com
  - Supported through sub forums at http://www.simics.net

Intel Simics Academic Program

- Academic program by invitation only
  - To get serious, well-known participants
  - From an Intel individual, "sponsor"
- Simics 4.6
  - Most recent Simics version
  - Intel Core i7 with X58 and ICH10
  - Intel Tunnel Creek (Atom based)
- Launch expected in early Q4, 2012
- Independent of Wind River Academic Program
  - Wind River provides non-Intel platforms
- Same support process at http://www.simics.net
- Licenses granted one year at a time

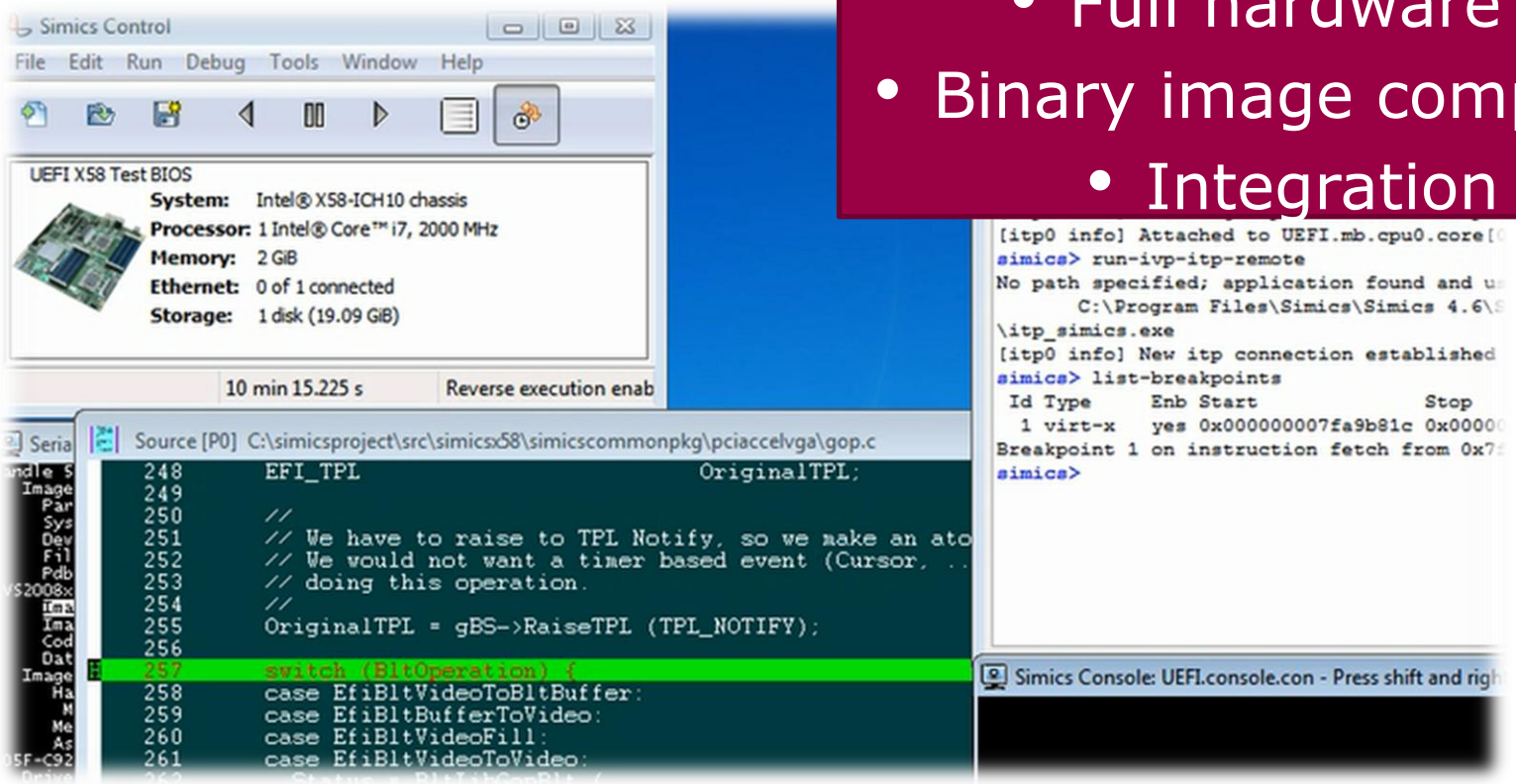


Free Discussion

BACKUP

Case Study:
ITP Integration With Simics

- Full hardware visibility
- Binary image compatibility
- Integration with ITP



Hybrid Simulation

- Integrate a detailed model with Simics
  - Timing
  - Power & thermal
- Run real workload on fast Simics model
- Switch to detailed for interesting part
  - Only replace units of interest
- Requires model specific work

Controlled Environment

- Examine viruses, Trojans, cyber attack software
- Run unmodified software
- Full inspection capabilities
- Sandboxed multi-machine environment

SIMICS FORUM SUPPORT FOR ACADEMIA USERS

SIMICS FORUM SUPPORT FOR ACADEMIA USERS

Academia	Posts	Last Post
Academic Site License - License Support (Public)	411 (24 new)	19 Sep 2012, 14:46
Academic Users (Note: Public)	13269 (61 new)	7 Oct 2012, 12:23
Academic Users - Simics 4.0 (Public)	1878 (120 new)	4 Oct 2012, 02:41
Chalmers	267	28 May 2010, 06:37
CMU	176	23 Jan 2009, 12:29
GE Edison	139	18 Jan 2012, 16:32
KTH	101	8 Apr 2008, 02:17
Texas A&M University	21	22 Sep 2009, 00:53
UART (Uppsala)	163	11 May 2006, 04:25
University of Pennsylvania	291	26 Oct 2009, 03:55
University of Texas at Austin	0	-
Wisconsin-Madison	1157	19 Oct 2009, 09:00
External Bug Reports & Feature Requests	Posts	Last Post

Virtual Platform Library

Target CPU Architectures

PowerPC Architecture
- AMCC PowerPC 440
- Freescale e300
- Freescale e500
- Freescale e500-mc
- Freescale e600
- Freescale MPC603e
- Freescale MPC750, MPC755 (“G3”)
- Freescale MPC74xx (“G4”)
- IBM PowerPC 405
- IBM PowerPC 403 core
- IBM PowerPC 464FP core
- IBM PowerPC 750(fx,gx)
- IBM PowerPC 970, 970MP
- IBM POWER6
- IBM Cell Broadband Engine

Texas Instruments
- C64 DSP
- C64+ DSP

ARM Architecture
- ARM 5TE
- ARM 9EJ
- ARM 9E
- ARM 11

x86 Architecture
- Intel 80386
- Intel 80486
- Intel Pentium
- Intel P4
- Intel P4 with EM64T
- Intel Core
- Intel Core 2
- Intel Core 2 Duo
- AMD Athlon
- AMD Athlon64
- AMD Opteron

MIPS Architecture
- MIPS 4K
- MIPS 5K
- MIPS 5Kf
- PMC RM7000
- PMC E9000
- Cavium cnMIPS64
- RMI MIPS64

SPARC Architecture
- SPARC-V8
- SPARC-V9

Renesas
- H8S Microcontroller
- SuperH SH-4

SoC's Modeled

AMCC
- PowerPC405 & PowerPC440

Freescale
- QorIQ™ P4080
- PowerQUICC II (MPC82xx)
- PowerQUICC II Pro (MPC83xx)
- PowerQUICC III (MPC85xx)
- MPC8641/D

Texas Instruments
- TI C6414
- TI C6455

HIREC
- HR5000

Cavium
- Octeon 38xx/58xx

Virtual Platforms

- Freescale MPC8572E
- Freescale QorIQ P4080
- Curtis Wright SVME183 (PPC7447A)
- Freescale HPC-NET with MPC8641D
- BAE RAD750 with Ethernet & MIL-STD-1553
- Wind River SBC750 (PPC750)
- Sun SPARC Server Family
- Virtual PC (AMD x86-32 & x86-64)
- Virtual PC (Intel x86-32 & x86-64)

Target Devices

- Memory and system controllers
- Interrupt & DMA controllers
- Ethernet controllers
- PCI and PCI Express
- Serial ports
- USB devices and disks
- SCSI controllers and devices
- I2C controllers and devices
- RapidIO controller and devices
- Communication devices such as those for Firewire, Spacewire, etc

Target Operating Systems

- Enea OSE
- GreenHills Integrity
- IBM AIX
- In-house RTOSes
- Linux
- Microsoft Windows (including Vista)
- Monta Vista Linux
- NetBSD, FreeBSD
- QNX Neutrino
- RTEMS
- Sun Solaris
- Wind River VxWorks

Not limited to any RTOS, OS, or exec

Our list of CPU, SoC, device models and virtual platforms changes frequently. Please visit our online list at: www.virtutech.com/products/simicsmodels

Section 1: Introduction to Simics

Henry Cook

CS152 - Spring 2008

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

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

- 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=13