Wind River's SIMICS

Wind River Simics is a fast, functionally-accurate, full system simulator. Simics creates a high-performance virtual environment in which any electronic system - from a single board to complex, heterogeneous, multi-board, multi-processor, multicore systems - can be defined, developed and deployed. Using Simics, software developers can execute and debug their target software stack, from hypervisor to application.

Simics enables companies to adopt new approaches to the product development life cycle resulting in dramatic reduction in project risks, time to market, and development costs while also improving product quality and engineering efficiency. Simics allows engineering, integration and test teams to use approaches and techniques that are simply not possible on physical hardware.

Simics can simulate any digital system including a basic CPU + memory, a custom FPGA or ASIC, an individual board, or a rack full of boards. Large, complex, mixed architecture systems can be modeled with off-the-shelf support for several processor families (e.g. ARM, Intel, MIPS, PowerPC, Tensilica, and TI DSP), hundreds of IO devices, and standard communications, backplane and network protocols.

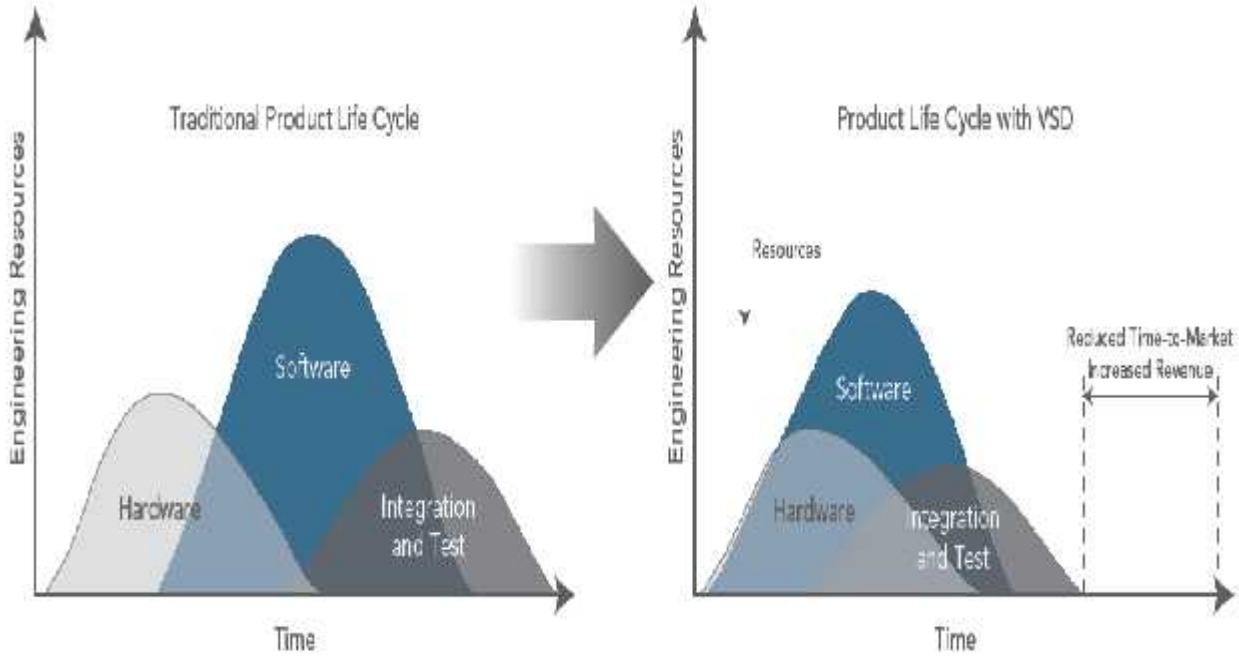
Simics allows us to teach device driver programming with a less painstaking process than with real hardware. Moreover, it allows us to take measurements, like for example interrupt latency, to show important concepts that would be more difficult to understand with real hardware.

Simics virtual platforms can be as complex or as simple as your actual physical hardware. They can contain multi-core processors, multiple processor boards, or multiple-board systems. This provides the ability to debug your system as a whole, instead of just debugging individual pieces. Simics virtual platforms can contain a mix of different target hardware architectures, including mixed endian architectures. Simics virtual platforms eliminate problems associated with using physical hardware for product development such as limited target availability, flaky prototype hardware, and hardware delays.

Issues with Real Hardware Implementation

With real hardware huge investments are needed for supporting a similar scenario. For example, in a real good Embedded Systems Lab each student requires: two evaluation boards (one with PowerPC and one with LEON), USB to serial cables, two hardware emulators for debugging (one for PowerPC and one for LEON), and adequate power supply. The total bill would be quite high!By using Simics we can reduce Time to Market and development cost and can do lots of debugging and development of new boards using different architectures for different purposes.

One big issue is physical memory versus virtual memory. In the Simics model developer use the physical address to map the custom device on the processor bus, while the virtual address should be used in the device driver. Moreover, depending on the configuration of the processor the same physical address may correspond to different virtual addresses, and this often makes the life of student complex. Simics is really useful to cope with this issue, as student can look at the MMU configuration, easily translate physical addresses to virtual ones, and look at each memory space of the system in an easily-readable way.



Reduce Time to Market or Reduce Time for Project Development

Besides all these reasons, working with virtual platforms is much easier: in case of hardware hang-ups it is easy to stop the simulation and to analyze the state of the system to figure out what was wrong, there aren't dead times as using checkpoints it is easy to save time when bugs requires frequent booting, and infinite number of booting is possible without the risk of stressing to much the hardware.

How Simics Transform the Product Life-cycle?

Use virtual target for architectural analysis:

- Pre-silicon architecture analysis using actual target software
- Legacy system upgrade analysis using actual target software

Eliminate hardware availability and flaw issues:

- Hardware and software co-development
- Develop target software before hardware is available

Utilize virtual target instead of host-based development:

- Advanced target hardware for everyone
- Easy collaboration among entire team

Eliminate system availability issues:

- Iterative and incremental integration and test
- Debugging at the system level

Utilize virtual platform even after development is complete:

- Maintenance of legacy products for five, 10, 20+ years
- Support of many different customer configurations

Reusable Assets Enable Agile and Iterative Development

What Simics can do for you?

Simics is an ideal platform for software developers — both at the board bring-up level and the software application level. At the board bring up level, Simics provides early access to virtual hardware to allow developers to have drivers, BSPs, and RTOS's ready to go when physical hardware arrives. In addition, debugging this type of code can often be challenging because developers don't have access to internal states and registers of hardware devices. Simics provides visibility and control of all internal states and registers and provides tracing and logging of device accesses. All of these features help simplify debugging of low-level software.
Application developers also benefit from Simics because they can develop, debug, and test their code on an entire system — not just on a reference board or in a host environment. The benefits of doing this are two-fold.
First, integration doesn't occur at the end of the project where bugs are costly to fix. Instead, integration occurs throughout application development because everyone has access to the entire virtual system.
Secondly, the hardest to find bugs frequently occur in the interaction between major components of the system — like boards or network nodes. By having all of these under the control of Simics, these bugs can be provoked and then reverse execution can help track down what caused them to occur.

Experiment - DREAM IT. SIMULATE IT. VALIDATE IT.

Simics makes it easy to experiment with different hardware setups, software applications, and platform configurations to validate assumptions before committing to system design. Test the way your software behaves and scales by varying memory size, core count, processor speed, and number of boards in a network, in any variation of circumstances. And run software setups from various generations on different hardware to ensure that the old works with the new as it should.

Test & Run - FLY WHAT YOU TEST AND TEST WHAT YOU FLY

Run your software on the virtual system just as it would run on the physical system, without the limitations of physical systems. With Simics, the software binaries you run on the virtual system are the same you'll run on the real target—and all developers can run them from the very start. You can also integrate the real software running on a virtual target with hardware-in-the-loop and simulation-in-the-loop setups. Simics can interface to the real world as well as to virtual worlds, allowing software to be tested and run in virtually real setups.

Simics allows software development, QA, and integration to start earlier and progress concurrently. Develop and test individual pieces of the system on the fly, avoiding a risky "big-bang" integration phase at the end. Automate your complete test environment, with unlimited numbers of target systems to run tests in parallel. And run automated nightly builds that test the latest check-ins, with complete reports in the morning.

Simics also allows you to easily test error handling and fault recovery code, using fault injection to affect arbitrary parts of the hardware (and sometimes software) state with perfect precision and with no risk of breaking expensive hardware. With Simics, you can easily test system parts that are otherwise almost impossible to test at all.

Debug - GROUNDBREAKING DEBUGGING TECHNIQUES

Simics makes problem isolation and analysis much more efficient. Debug all your software (including BIOS, drivers, low-level software, and OS-level code) without access to hardware and with perfect control over the virtual target. Start, stop, single-step, and reverse the system to find out exactly where and why it breaks—and email the resulting setup and analysis to colleagues around the globe. With Simics' determinism you have perfect repeatability—you know that issues found on the virtual target can be repeated any time, anywhere.

Use the scripting and programmability of Simics to automate bug detection. The Simics debugger debugs an entire target system as a unit, including all levels of the software stack and even multiple OS instances and boards. And with Simics' OS awareness you can debug individual user space applications alongside the OS, hypervisor, and BIOS.

Develop - DEVELOP ON THE REAL SYSTEM—FROM THE START

Many application developers use substitutes for their software development, debugging, and testing tasks because of bad experiences using target hardware. Since they then use a different toolchain and run their code on a different architecture, many problems do not appear until the code is compiled for the real target and integrated onto the target system. Because this activity occurs late in the development process, problems found at this stage can have huge repercussions on the overall schedule and product reliability. With Simics, application developers can use the real target from the start, with the same toolchain, libraries, operating system API, and operating system behavior. With virtual targets there is no lack of hardware access, no managing of complex configuration and different setups, no flaky hardware not working, and no expensive shipping of hardware.

Use a Simics Quick Start Platform to equip application developers with a target of the right architecture and operating system long before the actual target hardware has been designed, developed, and manufactured.

Enable - EMPOWER ALL COLLABORATORS

Enable your product users with virtual systems in all phases of the product lifecycle. For example, a semiconductor vendor can seed its customers with early access to a new chip, far in advance of prototype silicon. An internal development group can provide platform integrators and application development teams with virtual boards while the real board is being designed. Partner companies working on different parts of a program can share a common system for development. Operations teams in sales, customer satisfaction, partner enablement, and training can get huge boosts by deploying simulated hardware to their teams. Sales can demonstrate new products before they are available to shrink the gap between product release and revenue. And with simulated hardware, training is no longer limited by the expense constraints of physical hardware, making it possible to train more people, faster.

Configure - ENDLESS CONFIGURATIONS IN NO TIME AT ALL

With Simics, a target system setup is just software, so you can control almost any parameter of the target system using scripts and Simics modules—from simple things, such as the MAC address of a board, to the more complex, such as particular software loaded on a certain board when it is inserted into a certain position of a backplane. Simics uses a programmatic model for configuring systems, making it easy to vary configurations under script control. And configurations work the same today as they will tomorrow—anywhere in the world.

Easily store multiple configurations of the system with instant recall of any setup. Set up with an infinite supply of boards, and with no need to physically visit the lab to change configurations. And use the System Editor to inspect and interactively change setups during a run.

Keep a virtual version of in-field systems up to date for testing hardware and software updates, making modifications, and upgrading products. Customer service can have every customer configuration at its fingertips to effortlessly reproduce errors and solve customer issues.

Communicate - ONE TEAM. ONE SYSTEM. EFFORTLESS COMMUNICATION

Simics makes it possible to consider the system's hardware, software, current state, and execution history as a single unit that can be copied, communicated, and replicated with ease by anyone on the team, anywhere in the world. With Simics, communication between development, test, integration, support, and sustain teams is effortless.

Simics assets such as device models, target configurations, system setups, automation scripts, checkpoints, traffic recordings, and debug configurations are all designed to be shared between users and teams. Metadata can be added in the form of user-editable comments, documentation strings, or time-stamped notes during the system execution to help the recipient of a setup understand how the current state was created.

Platform development teams can build, configure, and boot complete platform setups and distribute them to all developers so they always have the latest setup. And when a QA team observes test failures, it can use Simics checkpoints and determinism to reliably repeat the failing test cases on the developer's machine—with cycle-perfect precision and guaranteed success

Simics Key Capabilities

- A complete functional virtual platform with fidelity and performance
- Runs unchanged binaries—drivers, BSP, software stack and applications
- Use the same build settings and compilers as the physical target hardware
- Make custom virtual platforms broadly available
- Provides true reverse execution and debugging
- Deterministic execution means that bugs are trivially reproducible and easier to find
- Create scripts for hardware fault injection during testing
- Supports single, multicore, multiple processor, and multiple machine configurations (racks, clusters, and distributed systems)
- Simplifies debugging of multicore and distributed systems
- Supports networks of arbitrary topology and networks of networks

Key Points of Using Simics

Define System Architecture

- Leave paper behind and create an "executable specification."
- Make design decisions: How many CPUs? DSP or GPP? Cache size? Which software optimizations?

Develop Software

- Eliminate scheduling problems due to limited target hardware availability.
- Perform "impossible" debugging.
- Send a bit-exact system snapshot file to another developer for collaboration.
- Say goodbye to "nonrepeatable bugs."
- Eliminate big-bang integration

Start early and integrate progressively.

- Improve quality by testing hardware corner cases.
- Develop software applications across heterogeneous OS environments and hardware architectures.

Deploy Virtual Platforms

- Deploy virtual platforms to support field teams.
- Demo your system on a laptop.
- Support new customer configurations easily and inexpensively.

WindRiver's Simics User List – Academics

1. Indian Institute of Technology, Kanpur
2. Indian Institute of Technology, Guwahati
3. National Institute of Technology, Shillong
4. Birla Institute of Technology, Pillani
5. NIT, Agartala

WindRiver's Simics User List – Industrial

Aerospace and Defense

1. Airbus
2. BAE Systems
3. Boeing
4. GE Aviation
5. General Dynamics
6. Honeywell
7. Iridium
8. JAXA
9. L3 Communications
10. Lockheed Martin
11. NASA
12. Northrop Grumman
13. Raytheon

Industrial & Medical

1. AMCC
2. Emerson
3. Freescale
4. IBM
5. Intel
6. Hitachi
7. Rockwell Automation
8. Tektronix
9. Xerox

Network Equipment

1. Alcatel Lucent
2. Cisco
3. Ericsson
4. IP Wireless
5. Huawei
6. Motorola
7. Nortel