

Wind River Simics

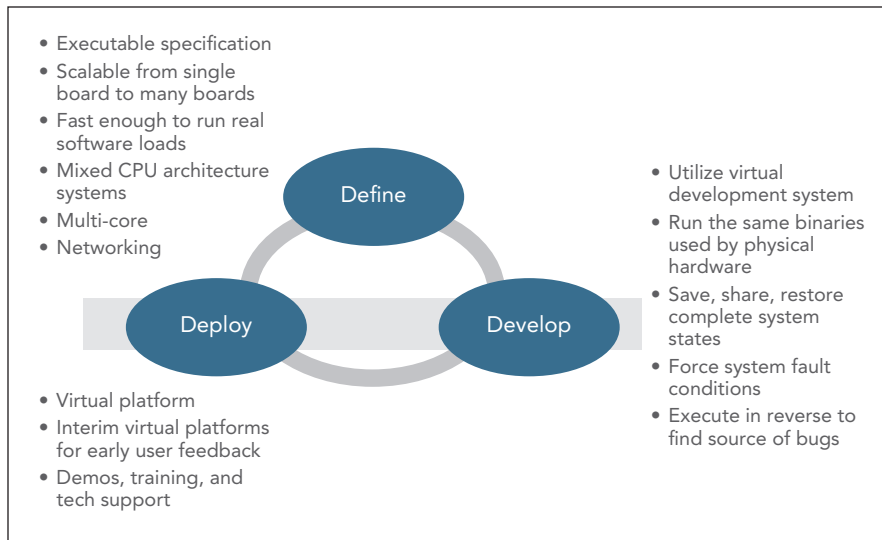
Table of Contents	Networks and Buses 4
Optimize Your Entire Product	Target RTOS Support 4
Life Cycle 2	Capabilities 5
Platform Engineering 2	Modeling 5
Application Development 2	Host Support..... 5
System Integration and Test 3	Target Support 5
Project Management..... 3	Target Devices..... 5
Quick and Efficient Modeling 4	Target CPU Architectures..... 5
Features and Capabilities 4	SoC Families..... 6
Scalability 4	Commercial Board Virtual
Connectivity/Interface 4	Platforms 6

Wind River Simics allows product teams to adopt a development methodology where physical system hardware is replaced by Simics virtual platforms running on a workstation or a PC. The virtual platform can run the same binary software as the physical hardware and is fast enough to be used as an alternative to physical

hardware for software development and testing. Simics virtual platforms are unique. They are fast and accurate enough to run a full software stack from hypervisor to application, and they guarantee repeatable software execution, full visibility/control of the virtual target hardware, and true reverse execution.

Simics allows companies to become more competitive by enabling improved product capability and quality while reducing technical risk, schedule risk, and development cost across the full product development life cycle.

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.



Simics improves a project's workflow by supporting true iterative development, hands-on system architecture investigations, and the ability to begin software development independent of hardware availability. By taking advantage of Simics as a consistent platform throughout the development process, teams are able to improve their entire product development life cycle to shorten time-to-market and customer adoption. Overall expenses are lowered and a higher-quality product is delivered to the market place, increasing the satisfaction of your customers.

Figure 1: Wind River Simics supports the full product life cycle

Optimize Your Entire Product Life Cycle

The entire product development team—including hardware architects, hardware designers, software designers, software testers, system integrators, and system testers—can utilize virtual platforms. Virtual platforms also reduce costs of technical support activities when you need to maintain unique variants of hardware for each of your customers.

In addition, virtual platforms can be used by sales and marketing teams to demonstrate new product capability and by customers for training purposes.

What Simics Can Do for You

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.

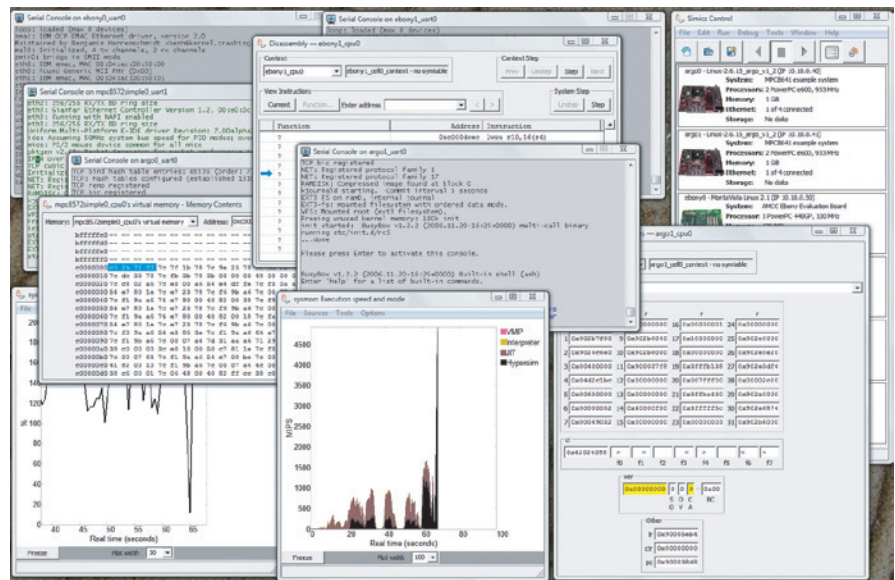


Figure 2: Debugging software with Wind River Simics

Platform Engineering

The platform engineering team is comprised of hardware architects, hardware designers, and low-level software developers. The platform engineering team defines and creates the product’s hardware architecture, system design, and foundation software.

In contrast to the paper trail that has been traditionally used during the architecture phase, virtual platforms enable a hands-on resolution of technical issues:

- Use multi-core, accelerators, or both?
- Execute a function in GPP or DSP?
- What memory size and speed?

The creation of a system’s foundation software—firmware, boot code, operating system, and device drivers—can quickly become challenging if

- Hardware is not yet available or prototypes are not reliable.
- The hardware spec keeps changing.
- It is not possible to access key device registers.

The Simics virtual platform evolves with the hardware design ensuring that developers remain productive without

access to physical hardware. Because a virtual platform is entirely software configurable, it evolves much faster than physical hardware. This allows architecture investigations and foundation software to be completed much earlier than would be possible with traditional development approaches.

Application Development

The application development team architects, writes, optimizes, and tests software. Application development can be challenging when

- Key portions of the application come from different development teams.
- Access to prototype hardware, debug tools, or instruments is limited.
- Debugging a single application requires access to the entire system.
- Some bugs are only visible when run on the complete system.
- Bugs previously observed are not easily repeatable.
- You are developing multi-core applications.

Simics has supported application development on simulated systems comprised of more than 50 boards and 700 processors,

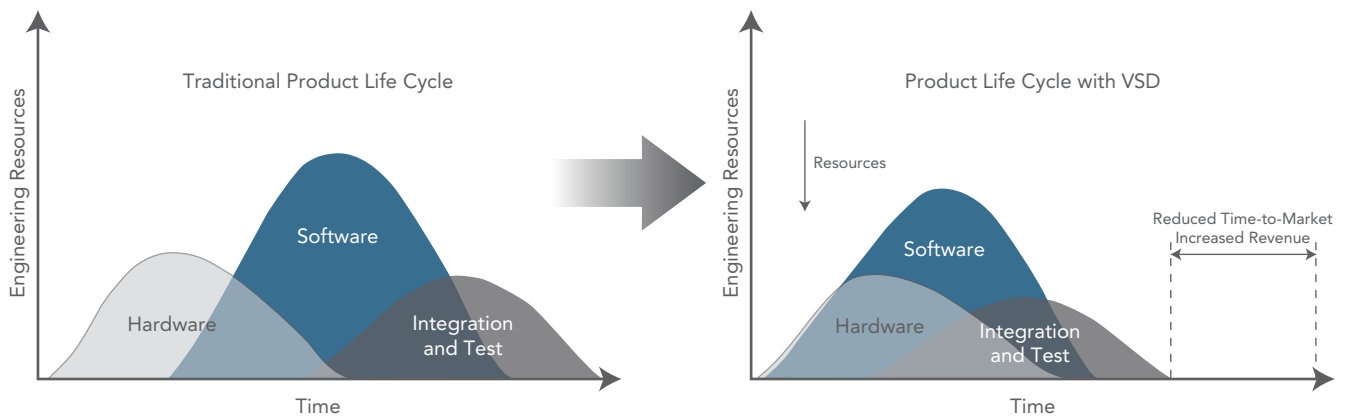


Figure 3: Wind River Simics reduces time-to-market, compared to traditional product development

including multi-core systems, heterogeneous designs, multi-OS, and network distributed systems. Simics provides complete control of hardware and time, support for seemingly impossible debug scenarios, and complete run-to-run repeatability. As a result, individual developers and application teams work more efficiently through the ability to share system snapshots and to work with the same compile/link/debug tools that they are accustomed to using on physical target hardware.

A Simics virtual platform makes it much easier to force, find, repeat, isolate, and kill bugs by doing the following:

- Running the same binaries that work on the physical target
- Freezing, saving, and restoring Simics snapshots
- Leveraging full visibility and control of every device, register, and communications interface (e.g., Ethernet, virtual machine environment, RapidIO, PCI Express, etc.) in the full system, even those that would be invisible on physical hardware
- Running software forward as with physical hardware or—unique to Simics—in reverse with complete register visibility and breakpoint capability
- Breaking on normal software parameters or on any hardware parameter (e.g., time, instruction, device, or memory access) for complete systems debugging
- Collecting trace, code coverage, profile, and other information without modification to source or binaries and without the need for hardware tools (e.g., JTAG)

System Integration and Test

System integration and test tasks can introduce significant risks to the successful delivery of the product, especially if

- The hardware is late but the delivery deadline has not moved.
- Tests cannot be completed due to lack of target hardware.
- A hardware glitch interrupts tests, costing hours to restart or redo.
- System faults cannot be forced without physically breaking the hardware.
- Bugs do not reproduce themselves from run-to-run.

When a virtual platform is used, integration can begin much earlier in the development process. Now “integration” moves from a single high-profile task on the Gantt chart to an evolving lower-risk activity that progresses forward as systems software becomes available. This progressive integration approach can identify system problems much earlier than traditional approaches, allowing them to be resolved easier, faster, and at significantly lower cost than otherwise possible.

Simics equips each engineer with virtual hardware that is dedicated to his or her own development, test, or integration tasks. Because the Simics virtual platform combines complete system visibility with the ability to stop and inspect the system at any time, engineers can perform comprehensive testing for any hardware or I/O fault condition such as response to corrupt packets, out of range readings, bad disk sectors, or any conceivable hardware failure. When bugs are detected during these tests, the test

engineer has only to freeze the system and send the checkpoint file to the software engineering team for guaranteed bug duplication and subsequent correction.

Project Management

Project managers focus on higher-level aspects of a program to ensure successful completion and delivery of their products:

- Design reviews
- Hardware/software team collaboration and communications
- Improved quality
- Looming delivery deadlines
- Risk reduction

Traditional development approaches, where the hardware team moves on to the next project immediately upon delivery of the platform, can result in schedule delays, loss of features, or rework because the engineering documentation is rarely 100% complete or accurate. Because engineers often “don’t really know what is needed until they get there,” critical design problems may volley back and forth.

In sharp contrast, Simics results in improved cross-communication and collaboration between hardware, software, and systems engineers. This happens because the virtual platform—representing hardware and allowing software to run—is used to define the product and uncover and resolve potential system issues long before they would otherwise be addressed. With a virtual platform, hardware and software developers can

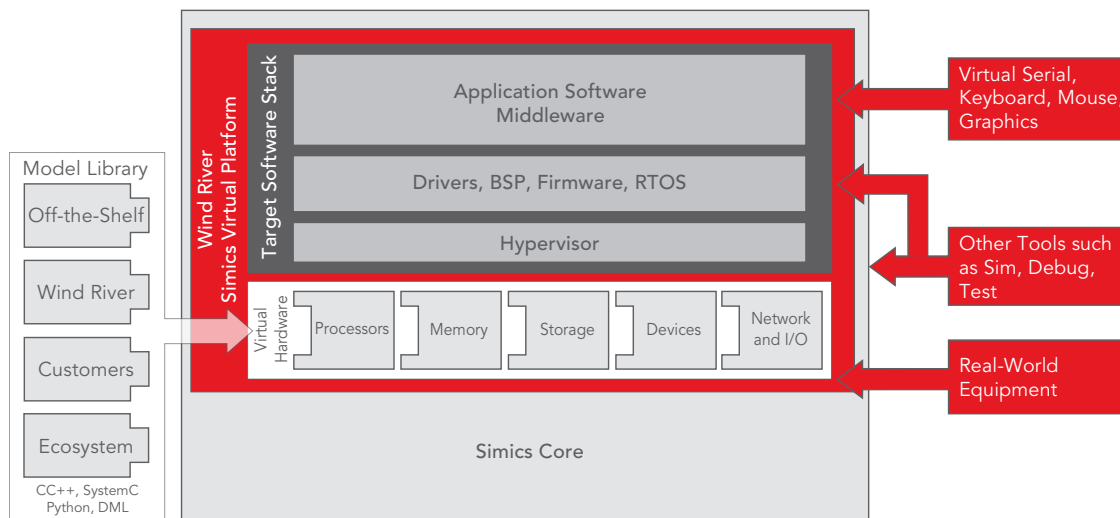


Figure 4: Wind River Simics block diagram

work in parallel to identify and resolve issues holistically, long before “it’s too late to do it right.” Design reviews take on a whole new tone when a virtual model of the product can be demonstrated, discussed, and modified long before the actual product is available.

A Simics virtual platform is able to replace your target hardware to run your entire target software stacks without modification. In addition, the same software tools used to build and debug the target continue to be used with the virtual platform.

Simics builds models hierarchically. At the bottom of every virtual platform, you will find individual models for physical target hardware such as microprocessors, memory, network interfaces, and bus controllers. These low-level device models combine into boards, boards combine into chassis, chassis into racks, and racks into network-connected subsystems. This modular approach ensures that the creation of new systems is straightforward.

Models are available off the shelf from Wind River or can be developed by Wind River, the customer, or Wind River’s ecosystem of partners.

Quick and Efficient Modeling

Simics supports mixed language models, where one component might be written in one language and another component in another language. This includes support for standard languages such as SystemC and TLM 2.0 and other languages including

C, C++, Python, and DML, the modeling tool available with Wind River Simics. DML is tailored to allow efficient creation of device models through a specialized C-based syntax that automatically supports features such as device state inspection, check-pointing, and reverse execution. Simics also makes it easy to define a device’s programming registers directly in DML, within any of the other supported modeling languages as appropriate to those languages, or by directly importing IP-XACT register definition files.

In Simics, a complete virtual platform is constructed by gluing individual models together with Python scripts. This approach isolates the more difficult model building tasks from the system configuration tasks and it allows variations and combinations of virtual systems to be quickly and easily created and managed.

Features and Capabilities

Simics provides a combination of unique features and capabilities to meet the needs of software and system developers. Only Simics has the ability to run full production software stacks on complex and large system models consisting of mixed CPU architectures and operating systems, multi-core processors, or network-connected platforms, all with enough accuracy and speed to satisfy developers.

Scalability

- SoC (system-on-chip)
- Single board

- Multi-board, multi-core, multiprocessor
- Mixed architecture systems
- Distributed and rack-based systems
- Networked systems
- Shared-memory and local-memory systems

Connectivity/Interface

- Wind River
- Eclipse plug-in
- Command line, batch, remote via Telnet
- GDB
- Cadence
- CriticalBlue
- Enea
- Freescale
- IBM Rational
- PolyCore Software
- Most major commercial debuggers

Networks and Buses

- Ethernet, AFDX, ATM, serial
- MIL-STD-1553, ARINC 429, SpaceWire, IEEE 1394 (FireWire)
- RapidIO, PCI and PCI Express, USB, I2C
- Shared-memory rack backplanes

Target RTOS Support

- VxWorks
- Wind River Linux
- Wind River Hypervisor
- Enea
- Express Logic
- Green Hills
- IBM AIX
- Linux
- Microsoft
- MontaVista
- NetBSD and FreeBSD
- QNX Software Systems
- Sun Solaris
- In-house and most commercial real-time operating systems (RTOS)

Wind River Simics Product Family

- **Wind River Simics Hindsight:** User interface, simulation framework, debugging
- **Wind River Simics Model Builder:** New device models and machine configurations
- **Wind River Simics Extension Builder:** Open interface to the Simics simulator that enables customers to extend Simics with custom functionality (plug-ins) and connect to additional workflows
- **Wind River Simics Analyzer:** Analyzing, code coverage, and debugging capabilities of software applications for large distributed and heterogeneous systems as well as less complex systems
- **Wind River Simics Ethernet Networking:** Virtual-world and real-world network connectivity
- **Wind River Simics Accelerator:** Utilize multiprocessor and multi-machine hosts for large simulations
- **Wind River Simics Virtual Platform:** Model of the physical target system, from an off-the-shelf hardware board to a complete custom system

Capabilities

- Complete execution control and debug
- Set breakpoints on memory, temporal, and any I/O
- Inspect and modify every hardware state
- Control components and configuration
- Profile, trace, and log (data, instructions, and registers)
- Utilize nonintrusive code coverage
- Inspect and edit memory and registers
- Control speed
- Run system forward or in reverse
- Analyze cache
- Automate tests with scripting
- Control performance and synchronization
- Simulate networks
- Utilize target OS awareness
- Fully control Simics and models
- Utilize graphical full system execution time line of processes and threads
- Extend with user-defined modules

Modeling

- OSCI SystemC TLM-2
- The SPIRIT Consortium IP-XACT import and export
- C, C++, and Python API
- DML (Device Modeling Language)
- Parameterized and hierarchical system models

Host Support

- Linux (x86, x86-64)
- Windows (x86)

Target Support

Wind River has an extensive library of boards, devices, and architectures. Check http://www.virtutech.com/products/simics_model_library.html for the most up-to-date list.

Target Devices

- Memory and system controllers
- Interrupt and 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
- Various communication devices such as those for FireWire, SpaceWire, etc.

Target CPU Architectures

ARM Architecture

- StrongARM
- XScale
- ARM7 (v4)
- ARM9 (v5)
- ARM11 (v6)

Intel and AMD

- Intel386
- Intel486
- Intel Pentium
- Intel Pentium MMX
- Intel Pentium Pro
- Intel Pentium II
- Intel Pentium III
- Intel Pentium 4
- Intel Pentium 4E
- Intel Pentium M

- Intel Core
- Intel Core 2
- Intel Core i7
- Intel Xeon variants
- AMD Athlon
- AMD Athlon 64
- AMD Opteron

MIPS Architecture

- MIPS 4K
- MIPS 5K
- PMC RM7000
- PMC E9000
- Cavium cnMIPS 64
- RMI XLR MIPS64

Power Architecture

- Freescale e300
- Freescale e500 v1
- Freescale e500 v2
- Freescale e500 mc
- Freescale e600
- Freescale MPC603e
- Freescale MPC750, MPC755 ("G3")
- Freescale MPC74xx ("G4")
- IBM PowerPC 405
- IBM PowerPC 403
- IBM PowerPC 440
- IBM PowerPC 464FP
- IBM PowerPC 750 (FX/GX)
- IBM PowerPC 970, 970MP
- IBM POWER6
- IBM Cell SPE

Renesas

- H8S
- H8/300
- SuperH SH-4

SPARC Architecture

- SPARC-V8
- SPARC-V9
- Gaisler LEON2

Tensilica

- Xtensa

Texas Instruments

- TMS320C64x DSP generation
- TMS320C64x+ DSP generation

SoC Families

Freescal

- QorIQ P1/P2, P4 families including P40x0 and P20x0
- PowerQUICC II (MPC82xx)
- PowerQUICC II Pro (MPC83xx)
- PowerQUICC III (MPC85xx)
- MPC8641/D

IBM

- Cell BE

Renesas

- SH7756
- H8S/2164, H8/3867

Texas Instruments

- TI C6414
- TI C6455

Cavium

- OCTEON 38xx/58xx

Commercial Board Virtual Platforms

- BAE Systems RAD750 3U/6U
- Curtiss-Wright SVME-183
- Curtiss-Wright CHAMP-AV6
- GE Fanuc VG5
- Sun SPARC Servers
- x86 PCs
- Various SBC development boards