SIMICS ACCELERATOR

MARCH 2009

JAKOB ENGBLOM

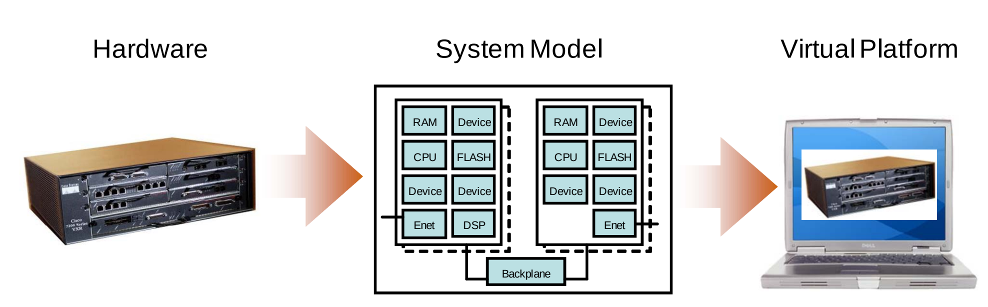
WWW.VIRTUTECH.COM

INTRODUCTION

Virtualized Systems Development™ is a development methodology where a system's physical hardware is replaced by a virtual model running on a workstation or PC. Simics allows the virtual hardware to run exactly the same binary software as the physical hardware, fast enough to be used as an alternative to physical hardware for the purpose of software development. The virtual hardware provides additional benefits such as improved debugging facilities, checkpointing and restart at any point, superior convenience and stability, access to the target long before prototype hardware to start software development early, and the ability to test faults and boundary cases with complete control and precision. The adoption of Virtualized Systems Development improves the handling of a product through its entire life cycle from bring-up through deployment, user training and support and even long-term maintenance.

Virtutech Simics is a flexible and scalable software product that models computer-based systems with high fidelity and high performance, enabling virtualized systems development as a methodology. As illustrated by Figure 1, Simics creates a virtual model of one or more hardware components, running that model on a regular workstation. The model is complete enough so that it can run the same software stack (operating system, drivers, middleware and application software) as the target machine, and fast enough to be useful in the development work of an individual software engineer.

Figure 1. Virtual Systems Development™



Simics creates *virtual hardware* that can be used just like the physical hardware, but with benefits of flexibility, full visibility and many other useful features. For example, the simulator is deterministic and controllable, providing a very powerful debug environment for parallel and distributed software running on multiple machines, multiple processors, or multicore processors. Today, Simics is extensively used by developers across a large range of systems, including servers, communications infrastructure equipment, aerospace and defense. The systems being modeled are often large and complex, containing many processors, peripheral devices, and large memories spread across many boards and connected over a variety of network types and buses.

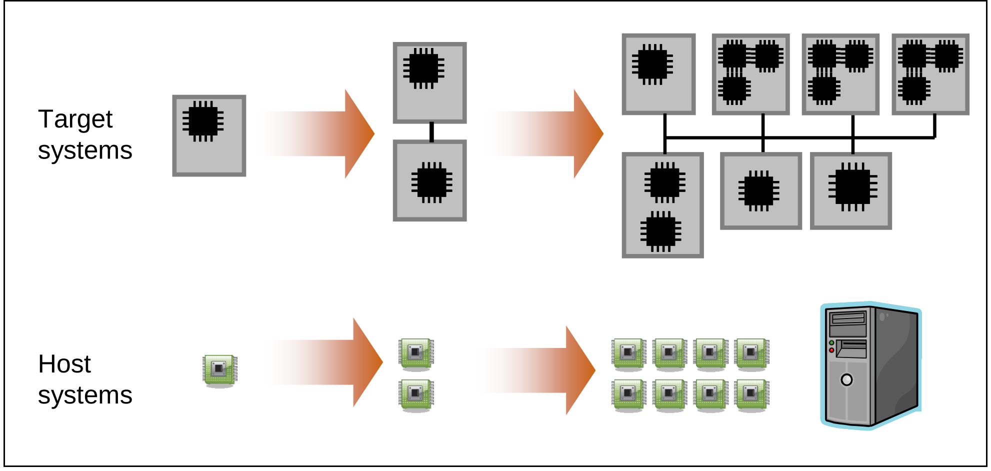
Simics Accelerator is a Virtutech product that complements Simics Hindsight and Simics model libraries to provide a unique set of scalability and execution speed capabilities for virtual platforms. With Accelerator, Simics takes advantage of multiprocessor and multicore hosts to dramatically improve the execution speed of large target system simulations. Accelerator's use of multiple host machines allows the size and complexity of these systems to scale beyond hundreds of nodes. In addition to enabling the use of more processor cores and host machines to run large virtual platform setups, Simics Accelerator also contains technology to exploit redundant data in the target system setup to further improve execution speed and reduce the memory requirements of Simics.

TECHNOLOGY TRENDS

Today we observe two parallel trends that affect the manner in which a simulation must scale. Target systems contain more boards with more chips on each board, and host systems are quickly evolving to increase the number of execution cores on every machine (Figure 2).

Multiprocessor computers used to be rare, found mainly in data center servers and compute farms. However the introduction of dual-core server and desktop processors by IBM, Intel, and AMD in 2005 began a shift towards multiprocessing as the standard. In 2008, dual-core processors became the baseline for engineering workstations, and in 2009 we are beginning to see quad-core and eight-core machines become the standard for new computer purchases.

Figure 2. Target and host system trends



In the same time frame, Virtutech has seen its partners and customers building simulations of larger and more complex systems. For example, telecommunications and data communications infrastructure systems often contain several dozen processors spread across dozens of boards or individual network nodes. The number of boards and the complexity of the boards have both increased in response to the need to handle increasing traffic flows and application performance demands. The inherent complexity of such systems, both from a hardware perspective and software perspective, provides a strong incentive to adopt a virtual systems development methodology as a means to improve the system development process. But at the same time, the sheer size of the systems makes achieving a sufficient execution speed for in the virtual system a challenge.

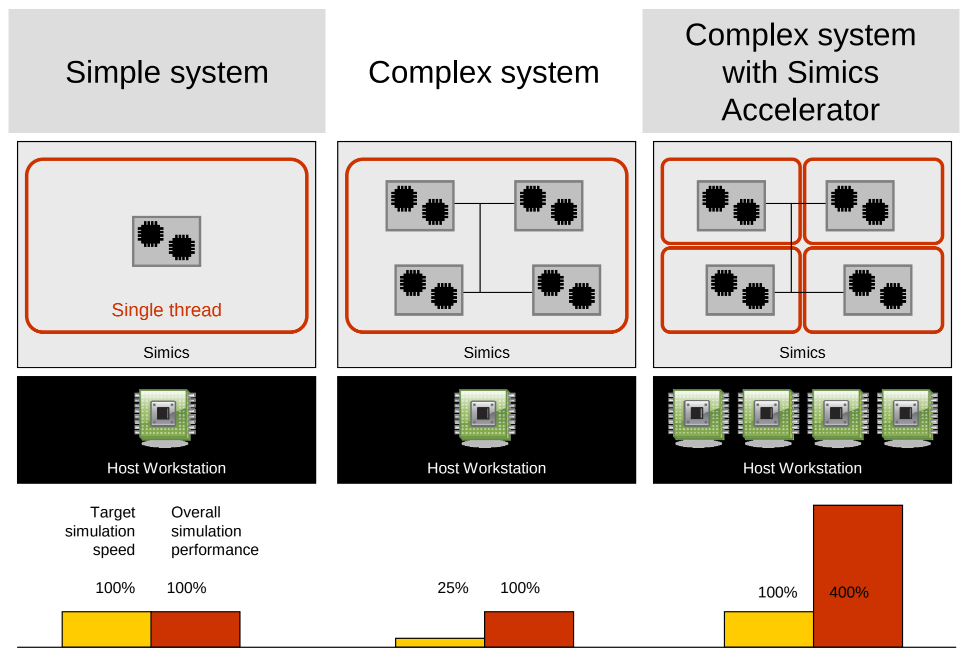
With Simics Accelerator, Virtutech is taking advantage of the trend towards multicore host computers and clustered computing to conquer the complexity of the large target system simulation.

MULTITHREADING SIMICS

Using multicore and multiprocessor host machines to virtualize multiple-board target machines is an obvious idea. If you have a host machine with multiple cores, then it should be able to run larger workloads (or existing workloads faster). With Simics Accelerator, this is exactly what is achieved.

Figure 3 illustrates the basic benefit provided by host multithreading support in Simics Accelerator. As the baseline, a single board is run in the Simics simulator. The host platform provides a certain amount of raw processing power, equating to simulation speed in terms of throughput in the simulation system ("overall simulation performance") as well as a certain perceived speed when compared to the real world time.

Figure 3. Multithreading a multiple-board simulation



When the target system complexity increases by moving from a single board to four boards, the same raw simulation speed has to be divided across the boards. Consider a traditional host example where Simics uses only a single host processor to run this system. Because in this example the simulated target system is now four times more complex than our base system yet simulated using the same single-processor host, the user perceives a 4x reduction in the simulation's interactive speed. In order to regain the perceived simulation response and performance, Simics needs a way to spread out the simulation work load across four times as many processors.

With Simics Accelerator and a quad-core host, the same four-board setup can achieve four times (in theory) the raw performance of the single-core host. Simics would use one thread to execute the simulation of each board. In essence, each board runs on its own host processor. This means that the user-perceived speed of execution would match the single-board case, despite the target system being four times larger.

In this way, Simics leverages more processors and more processor cores in the host system to run even larger target system simulations by leveraging the multicore host trend to conquer the target system complexity trend.

Simics Accelerator obtains an increase in simulation speed without weakening the fundamental properties that make Simics a powerful system development tool: determinism, repeatability, perfect control, full insight, and reversibility. All existing Simics features remain available regardless of whether Simics runs on a single processor or a multicore host.

Simics Accelerator can either use a static or dynamic work allocation of target machines to host execution threads. The default dynamic work allocation typically gives the best results and is fully automatic and invisible to the user.

Hierarchical Synchronization

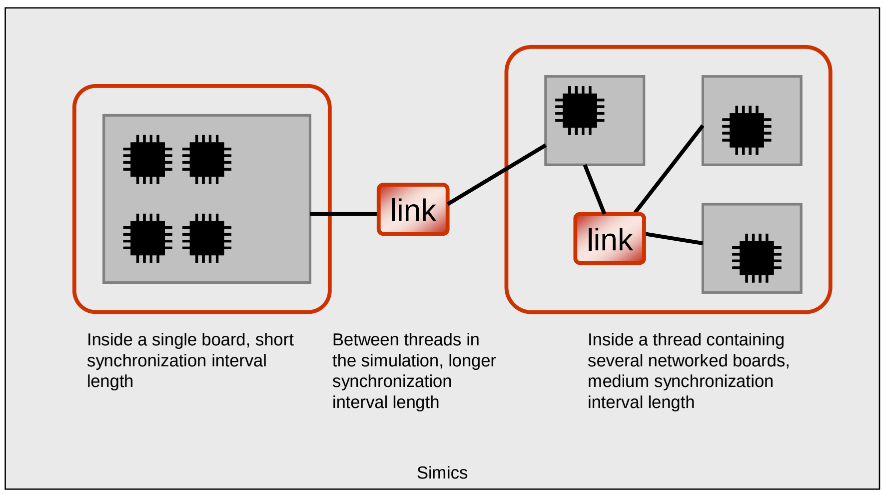
To maintain control, determinism and repeatability of execution in a multithreaded simulation, Simics Accelerator applies the principles of distributed synchronized simulation proven since 2000 in Simics distributed simulation technology[1]. To combine controllability with performance and to avoid too much synchronization overhead, Simics Accelerator ensures that separate Simics components communicate with each other only at certain specified synchronization intervals. These intervals are defined by the simulation configuration and do not depend on the underlying transport between hosts. Messages are delayed in virtual time until a time when they can be safely and deterministically delivered to the recipient, regardless of how the parallel execution of the system works out in detail. This approach ensures that the simulation will remain deterministic and will always compute

[1] This technology was informally known as "Simics Central". It has been since been replaced by the new distributed simulation capability in Accelerator 2.0 discussed later in this white paper.

the same result. The scheduling of the threads in the simulation has no effect on the outcome of the simulation.

This means that the semantics of a simulation are the same on a single processor, dual processors, or sixteen processors. A problem found in the target system can be investigated on any available host machine – regardless of the number of processing cores or nodes. There is no need to use the same host machine or cluster of machines to mirror the same target system execution (which is unfortunately often the case for parallel software) of the original Simics host.

Figure 4. Hierarchical synchronization of boards



Short synchronization intervals are suitable for tightly-coupled target systems like shared-memory computers or boards with devices connected over fast shared-memory interconnects like VME or PCI Express. Long intervals are appropriate for high-latency networks like Ethernet between Internet nodes, or between target machines that are geographically distributed in the physical system modeled.

Simics Accelerator, does not require you do to make a single choice for the length of the synchronization intervals. Rather, a hierarchical system of synchronization intervals is employed. As shown in Figure 4, logical subsystems in the simulation can have their own local synchronization interval length, appropriate to the internal requirements of that subsystem. Between subsystems, intervals tend to be longer since too short synchronization intervals will reduce the simulation speed overall.

SHARING A LARGE SERVER

Simics Accelerator also makes it easy to partition the use of a large powerful host server to run multiple Simics sessions in parallel, each using a subset of the available host cores. It is possible to specify for each Simics session exactly how many host cores it is allowed to use. For example, a 16-core host can be shared between four 4-way parallel Simics sessions, without the Simics sessions blocking each other (apart from obvious competition for available host memory).

Figure 5. Sharing a server

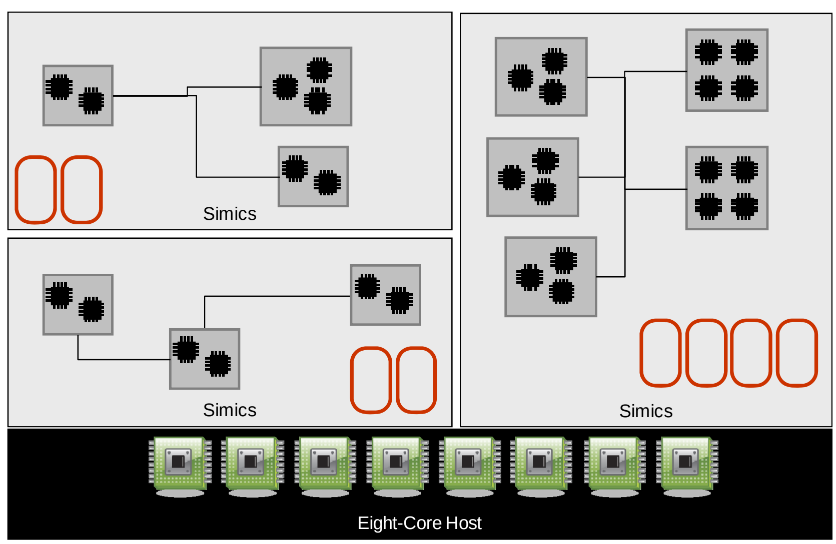
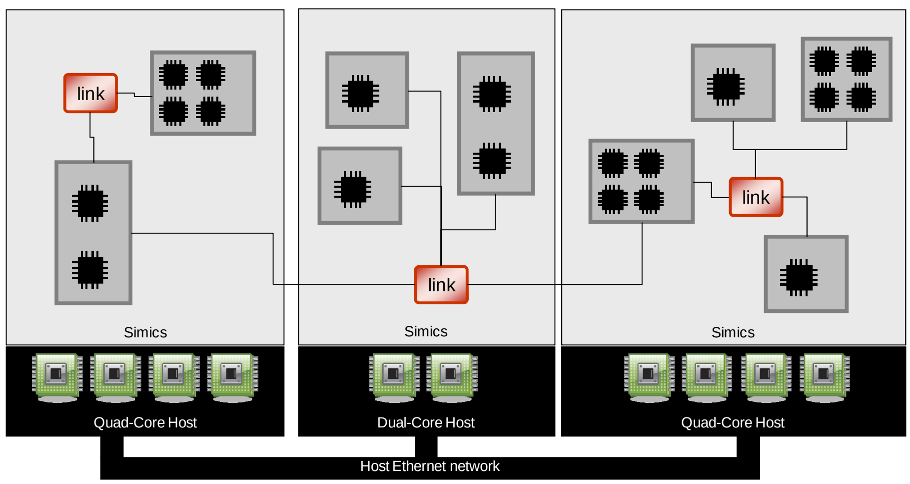


Figure 5 shows an example of a server sharing setup, where an eight-core server is running three Simics processes. Two of them use two host cores each, and one that is larger than the other has been assigned four cores for its use. With this setup, the server is not oversubscribed and each Simics should be able to proceed at full speed.

DISTRIBUTED SIMULATION

Simics Accelerator 2.0, available with Simics 4.2, allows the use of multiple hosts to further increase the scalability of Simics. Using a set of host machines connected over Ethernet networks, more processor cores and most host memory can be used to run simulations of large complex systems[2]. Distributed simulation provides the next level of scalability beyond the cores available in host machines, and makes it possible to use compute clusters to accelerate Simics simulations.

Figure 6. Distributed simulation example



As shown in Figure 6, with distributed simulation, individual target machines or boards are assigned to each host machine, and each host machine then runs its assigned set of target machines across multiple host cores.

The semantics of a distributed Simics simulation is deterministic and defined by the latencies assigned to the synchronization hierarchy for the target

[2] Note that this is a new technology that replaces the distributed simulation technology called "Simics Central", and which was used until Simics 4.0.

system. The execution of the virtual system is fully repeatable, just like for a multithreaded simulation within a single host machine.

PAGE SHARING

In addition to multithreading, Simics Accelerator also employs a technology known as *page sharing*. Page sharing means that Simics detects that multiple memory pages in target RAM, ROM, FLASH or disk have the same contents. Page sharing ensures that these duplicate pages are only stored as a single copy in host machine memory. Thus, the memory consumption of Simics is reduced and execution speed increased as a result of better cache locality and reduced pressure on the virtual memory system.

Figure 7. Redundant memory contents in target systems

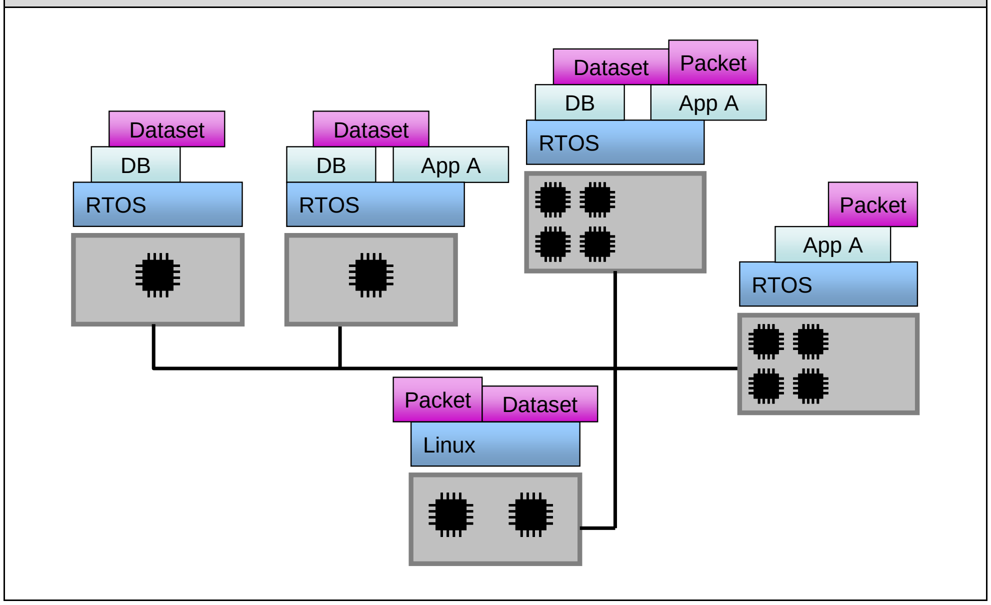


Figure 7 shows how Simics Accelerator page sharing supports redundant data and how code can appear in large systems. Typically, many boards run the same operating system binaries. Application binaries repeat across the boards as several boards cooperate to handle large loads. Redundant in-memory databases replicate data between boards as a way to ensure robustness. Incoming network packets (or rather packet streams) might be copied onto several boards in the system as they are processed from ingress to egress. Note that software and data repeat even when the boards contain different hardware, as the same software stack is used for multiple variants of boards and multiple generations of boards. The resulting system-level redundancy is exploited by Simics to store each relevant set of code or data only once.

Data can also be repeated within a single machine. For example, a bootloader might copy itself from flash memory to RAM for execution, which results in duplicate memory contents. This is also exploited in Simics, and can result is quite significant savings on its own. An extreme example of duplicate data is when an operating system zeros out memory pages as they are allocated. Obviously, all zeroed pages are identical so the simulation can clearly benefit from page sharing.

Page sharing is implemented in the Simics core memory handling, is invisible to the target systems and requires no special work in the system models to enable. Shared data is detected the first time a processor accesses a particular page, by interactive user command, or when Simics hits the configured maximal memory usage. To retain correctness, when a shared piece of data is written with unique data, a local page, not shared with other processors, is created.

Page sharing makes it possible to virtualize much larger systems than previously, as memory size was often a limiting factor to large-scale simulation.

RESULTS

In addition to the major technologies of multithreading, distributed simulation, and page sharing, there are also internal scalability improvements in Simics 4.0 and Simics 4.2 that remove bottlenecks and improve performance when simulating large target systems. The net results are an impressive increase in scalability and performance for large simulations.

Claiming any particular speed-up from general technology like Simics Accelerator is really impossible. In practice, results will vary depending on the nature of the target hardware and the behavior of the target software.

In our experience, multithreaded simulation can reach perfect scalability, running four times as fast on a quad-core host and eight times as fast on an eight-core host. The main limitation is in the balance of the target system. If there is one target machine that is much heavier to simulate than others, it will tend to dictate the overall speed of the model.

The benefits of page sharing are easier to understand, especially for many of today's multicore/multiplatform system architectures. For example, in a system where three identical MPC8572E-based dual-core machines run an SMP Linux with standard network configurations, we find that up to 96% of data is common across the machines when they arrive at the boot prompt. In this example, page sharing reduces the Simics memory needs for the three machines down to what is needed for a single machine. Another interesting data point is booting a single PowerPC440-based machine with Linux. In this single-machine setup, after bootup, page sharing detected that 20% of the memory was redundant, probably due to code being copied from flash to ram before being executed and some zero-valued memory pages. Furthermore, repetition between machines made it possible to store only about 10% of the total data in an eight-machine network.

SUMMARY

Simics Accelerator 2.0 represents a leap forward for virtual platform technology, improving scalability in terms of the size and speed of target systems that can be used in Virtualized Systems Development. Simics Accelerator takes advantage of the prevalence of multicore and multiprocessor workstations to run large simulations in a parallel manner, potentially increasing the speed of simulation many times compared to single-processor single-thread execution.

CONTACT INFORMATION

North and South America	Europe, Middle-East, Africa
sales_americas@virtutech.com	sales_emea@virtutech.com
Asia-Pacific	**Japan**
sales_apac@virtutech.com	sales_japan@virtutech.com
http://www.virtutech.com	

Virtutech, Inc., 2001 Gateway Place, Suite 201E, San Jose, CA 95110, USA

CONTACT INFORMATION

North and South America sales_americas@virtutech.com	**Europe, Middle-East, Africa** sales_emea@virtutech.com
Asia-Pacific sales_apac@virtutech.com	**Japan** sales_japan@virtutech.com
http://www.virtutech.com	

Virtutech, Inc., 2001 Gateway Place, Suite 201E, San Jose, CA 95110, USA