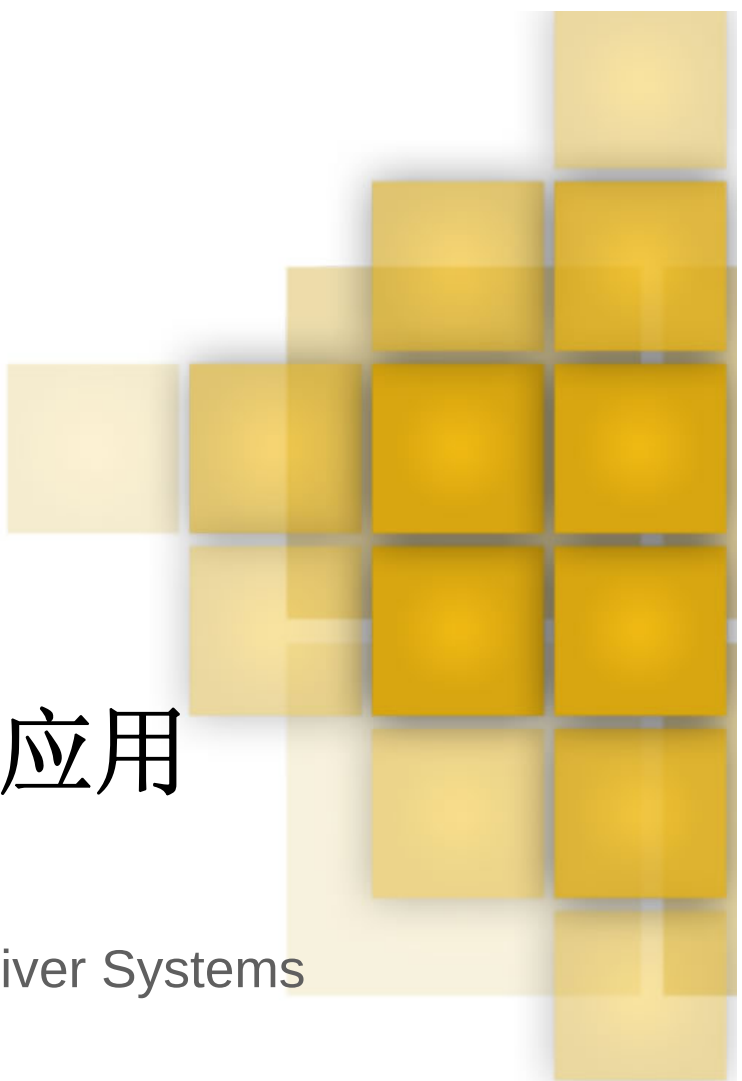




风河**Simics**产品在测试中的应用

Luke Yang, Simics Product Manager, Wind River Systems

背景和需求

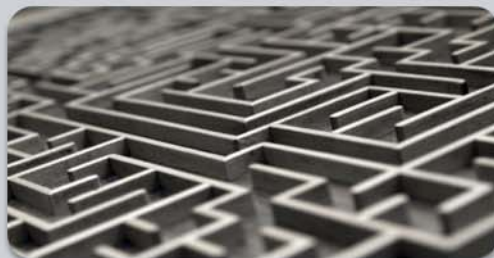


嵌入式软件测试面临的挑战



事半功倍

- 硬件成本和反复配置的高投入
- 测试团队人手不足
- 多样化的输入输出难以自动化



复杂系统的测试

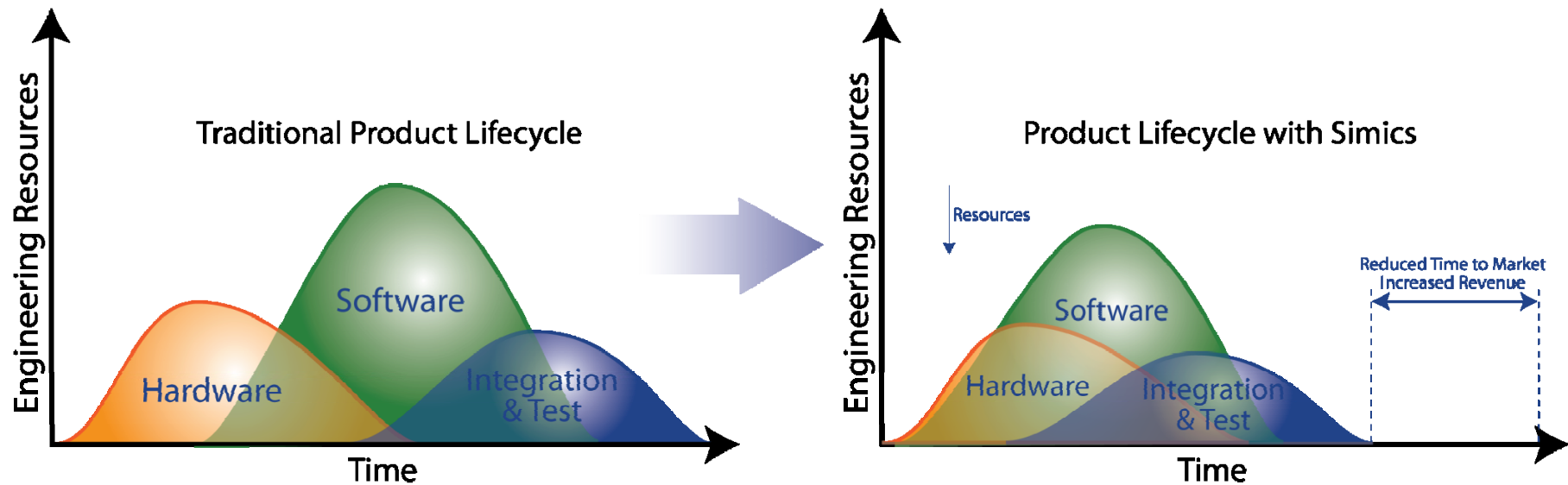
- 多样化、网络化的软硬件结构
- Bug难以稳定重现
- 如何做故障和可靠性测试
- 如何确认测试的有效性和充分性



全球性合作

- 测试和开发难以协作
- 硬件平台的版本难以统一
- 测试后期发现的问题导致项目进度受影响

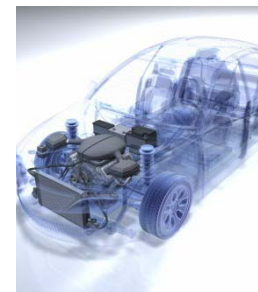
如果我们能更早地开始每个项目阶段?



- 集成和测试阶段向左移动，而且Bug也更容易、更快、更低成本地被修补掉（更少的迭代次数和反复调试）
- 软件开发在硬件架构设计之后马上可以开始，并且在硬件开发过程中提供及时的反馈
- 硬件开发延迟对整个团队的影响变小
- 上面的几点使项目需要更少的资源
- 上市时间缩短了，从而增加了整体收入

风河 SIMICS

一个全系统软件模拟器



风河*Simics*是怎样一个工具？

风河Simics是一个全系统软件模拟器，供软件开发模拟硬件，她能模拟任何规模和复杂度的电子系统。

针对测试团队：模拟特定硬件设计，测试软件实现



- 模拟任何规模的目标系统
- 运行无需修改的目标代码
- 同样的OS、源代码和编译工具
- 全系统软件

Simics*帮你**突破**传统的 product 开发和测试流程.

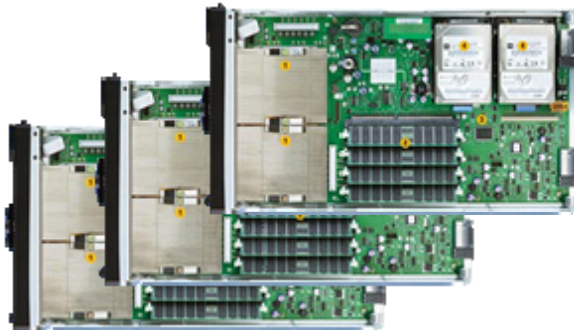
WIND RIVER

能模拟任何电子系统

- (软件)模拟使工程师人手一套系统，不管它多复杂...

...多板:

通过以太网或其他总线互联



仿真单板:

客户定制板或标准成品/参考设计板，包含CPU和板上所有外设

...整个机框:

通过VME或背板互联



...或是定制的完整数字系统:

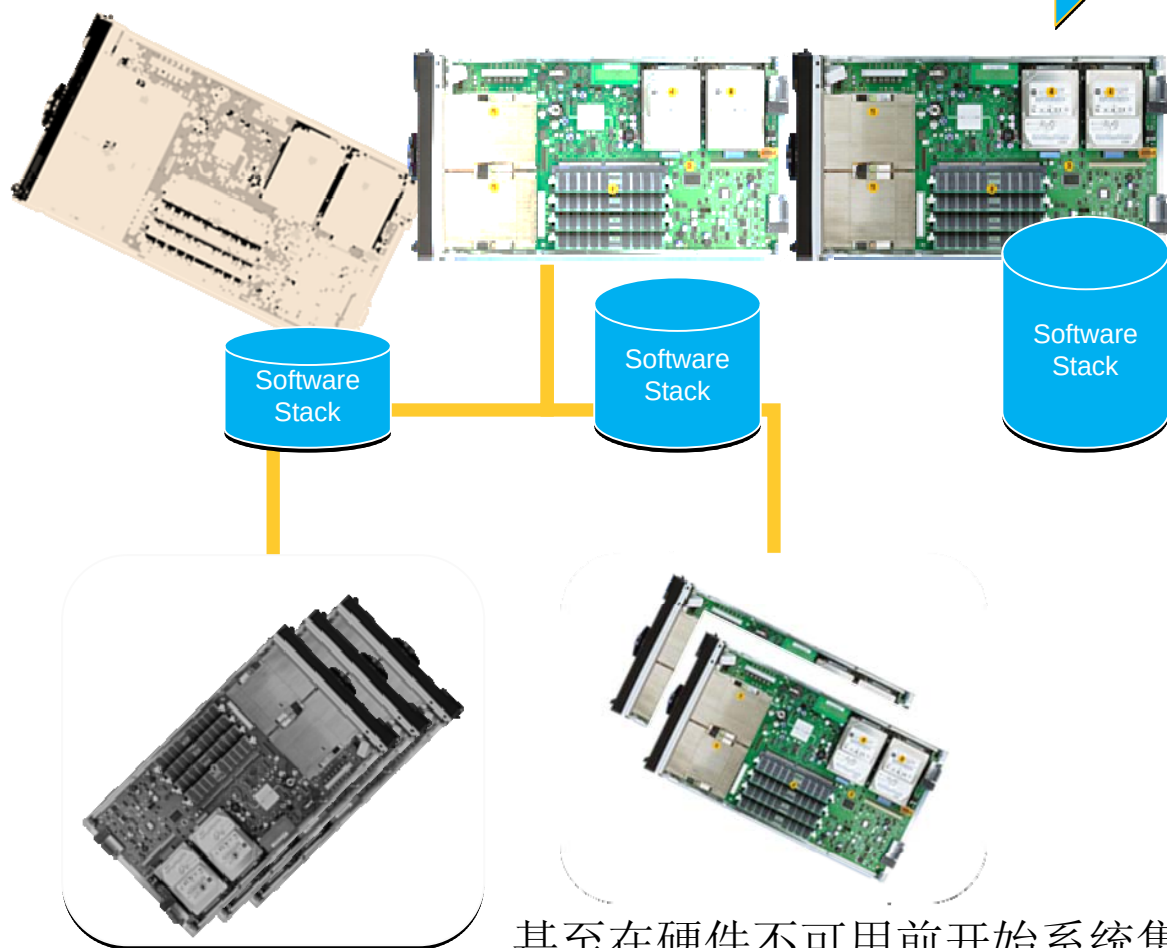
包括上百个CPU和外设



WIND RIVER

持续集成测试

不必推迟软件开发以等待芯片或新板全部完工...
早日开始



甚至在硬件不可用前开始系统集成

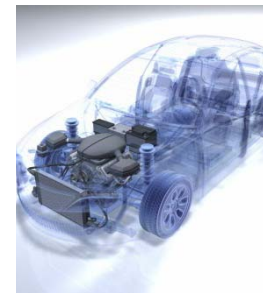
- 全项目周期的持续集成，不用等待硬件
- 更早、更频繁的集成测试
- 减少（小）：
 - 风险
 - 费用
 - 失误
 - 产品上市时间

Simics仿真平台可以支撑哪些测试？

支持多样化测试功能的测试平台，而非测试工具

- 单元测试
- 持续集成
- 功能性测试
 - 功能性白盒 (面向调试)
 - 功能性黑盒 (面向功能正确)
 - 覆盖率测试 (非侵入式)
 - 故障注入测试
- 软硬件集成测试

复杂系统下全系统模拟的优势



调试、测试复杂系统

可获得性

- 完整的系统难以对所有人可用
- 系统集成滞后
- 使用新芯片，新板时，硬件可能不存在或者不稳定

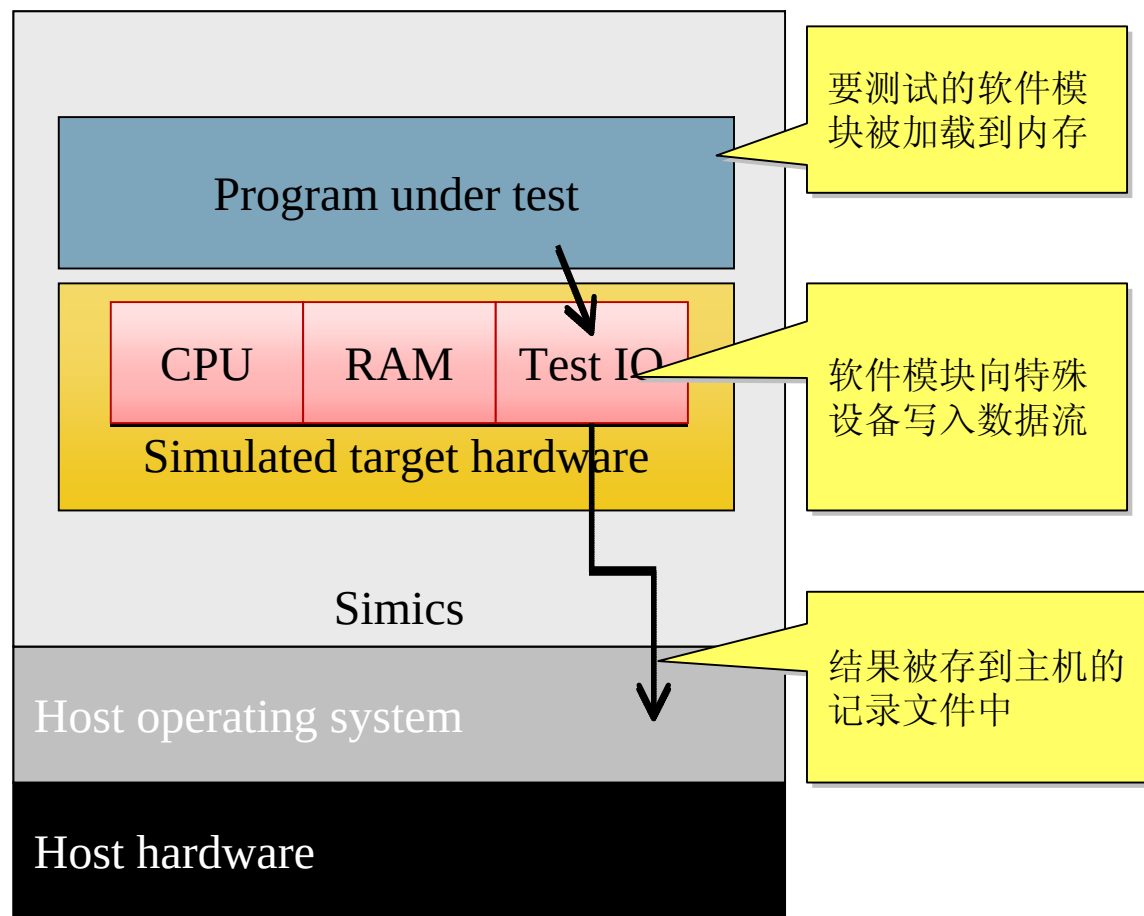
内部可见性

- 高集成度SoC屏蔽了可见性
- 多核
- 硬件加速器
- 模块间的数据通信

可控性

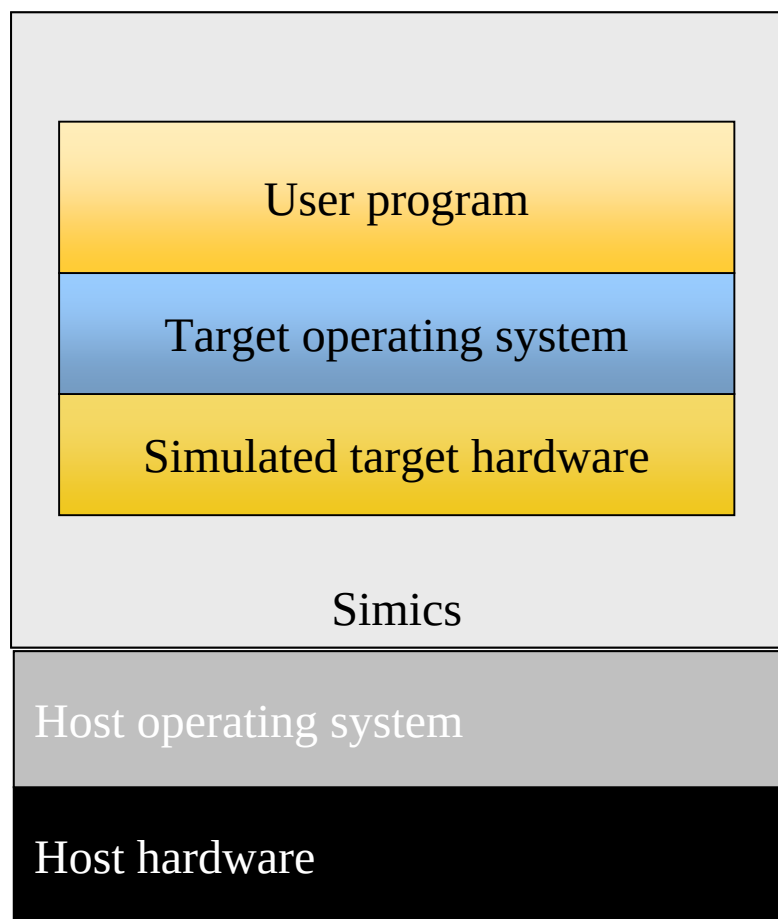
- 一块板上的断点被触发，其它板还在运行？能否有稳定的状态供后续分析调试？
- 一个核上的断点被触发，其它核还在运行？
- 怎样精确地复制测试环境，配置与现场？

Simics与单元测试



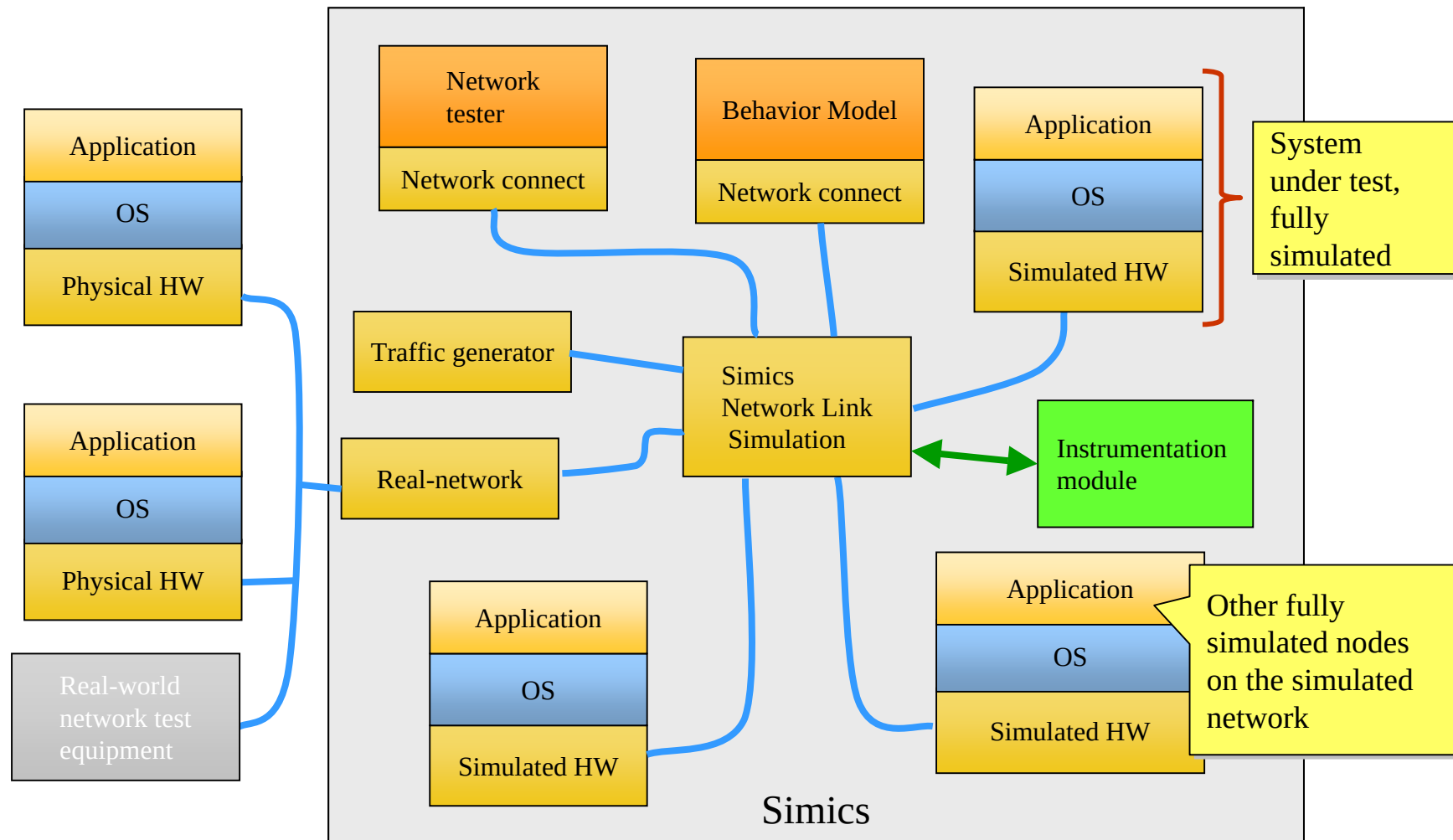
- 可控制/可配置
- 本地直接加载和运行目标代码
- 自动化 脚本化
- 运行结果确定，可重复
- 更真实的目标硬件
- 真实的开发编译工具

Simics与功能测试



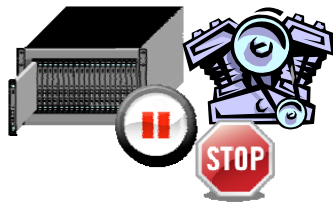
- 真实的应用软件运行在真实的OS/BSP/Driver上
- 完整的覆盖率测试
- 跟踪指令流提供Profiling作为性能优化的基础
- 与第三方测试软件的集成
 - Test Management, Testbed, etc
- 完整的自动化测试
 - 任何总线上的数据自动输入输出
- 更早的软硬件集成测试
 - 降低风险

全系统测试: 面向完整系统软件（多节点）



Simics的系统级调试和测试特性

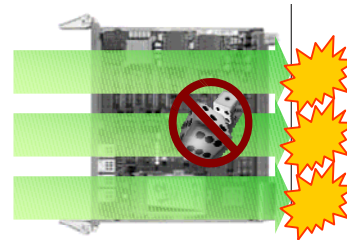
全系统的同步暂停



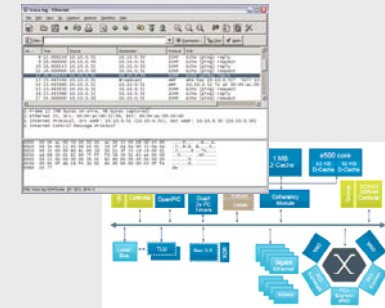
逆向运行;
无限制的断点



确定性运行;
可重复



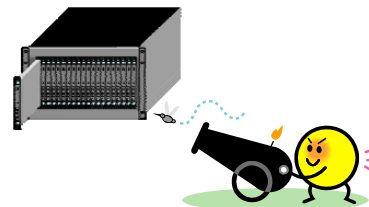
查看和跟踪任何系统组件和总线数据



全系统现场保存和恢复
(checkpoint)



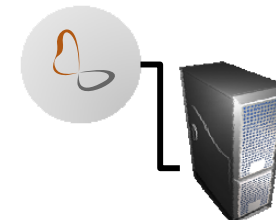
对任何组件进行可重复
的故障注入



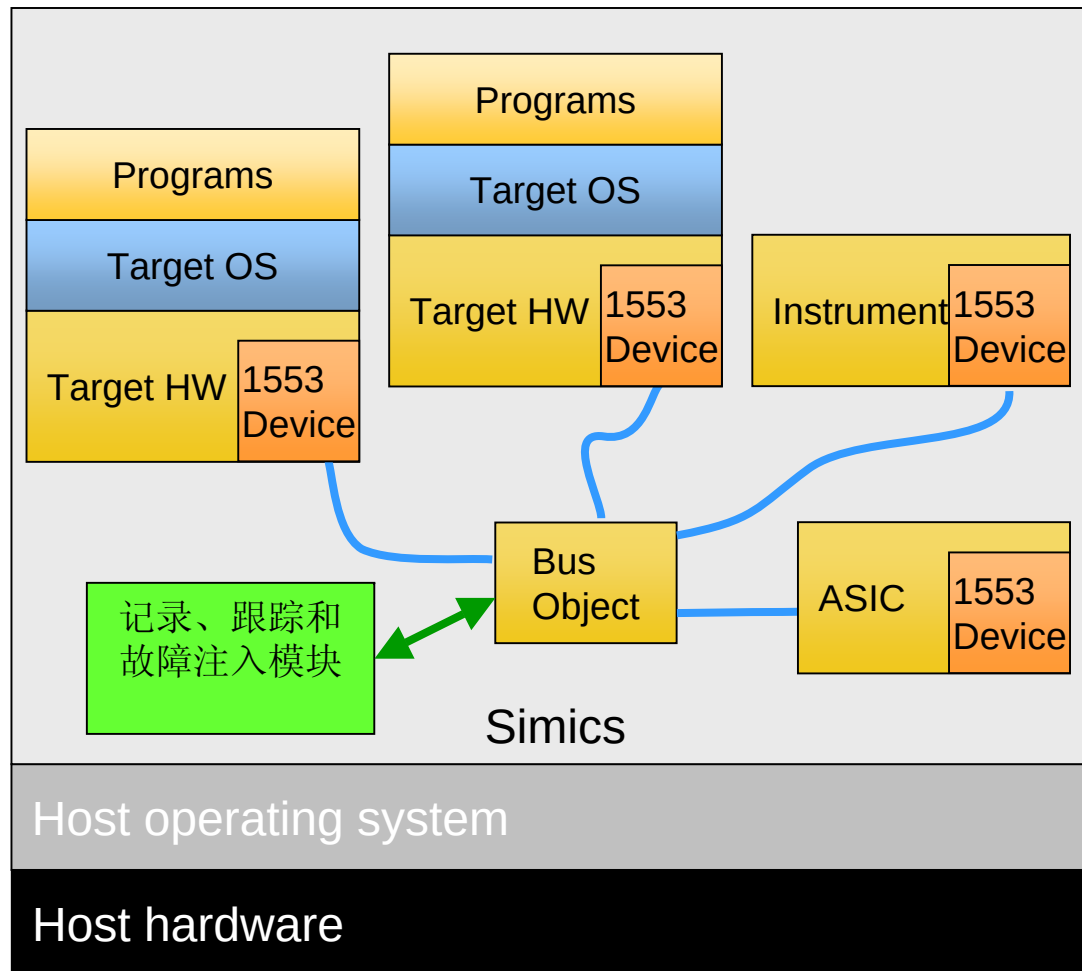
脚本化;
自动化

```
con0.wait-for-string "$"  
con0.record-start  
con0.input "/ptest.elf 5\n"  
if ($r == "fail.") {  
  echo "test failed"  
}
```

跟真实硬件互联



系统级仿真的例子 (航空子系统)



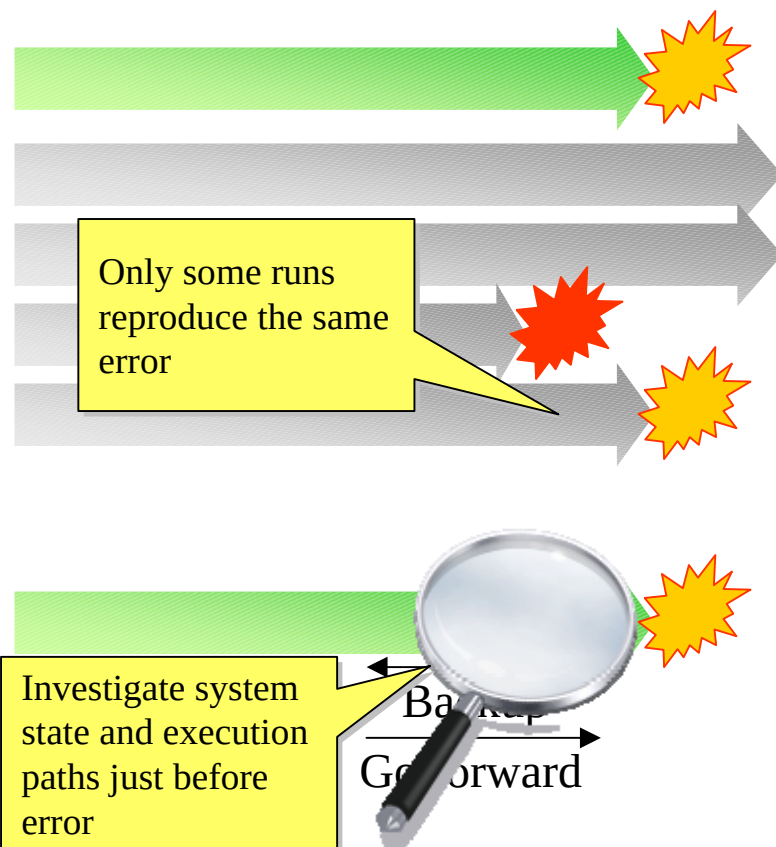
- Full computers
- ASICs
- Instruments
- Extensible
 - API for adding devices
 - API for instrumentation modules
- Multiple buses
- Multiple bus connections

Checkpoint - 提高生产率

- 真实系统:
 - 硬件搭建费时
 - 软件引导费时
 - 复杂的设置与手工操作才精确地回到想要的位置
 - 例如, 某个错误即将发生前的状态
- 全系统模拟:
 - 任意时刻保存系统快照/检查点(checkpoint)
 - 花几秒钟重新载入
 - 并能在世界任何地方重新载入!

Reverse Executing-重现故障，加速调试

- 系统暂停，时间回溯
 - 不需要从头开始重新运行
 - 从故障发生的时间点逆向运行开始调查
- 优点
 - 不需要盲目重启，希望bug能够重现
 - 在一次运行中精确调查发生了什么
 - 逆向运行中可设置断点和观察点
 - 从系统崩溃后逆向运行到正常状态
- 非常强大的并程序和网络系统的调试工具



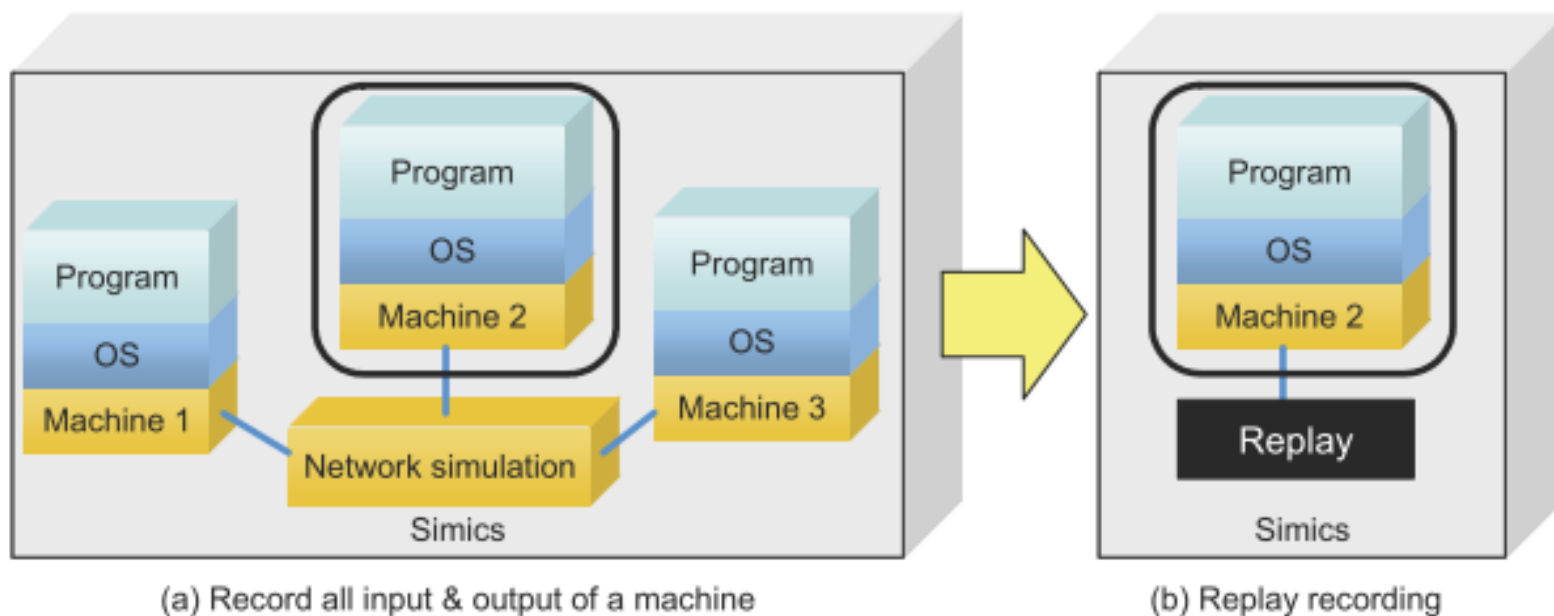
Fault Injection – 测试系统鲁棒性

- Simics能模拟任何硬件设备的故障现象或者边界条件
- 非破坏性、可重复、可自动化
- 支持日常的故障注入回归测试，而不是传统上系统集成之后做单次的破坏性测试

CPU	Memory	Networking	Redundancy	IO device
Different Frequencies	Bit error	Packet data error	Subsystem broken	Unexpected boundary data
Rare core numbers	Corrupted Register	Packet dropping at any rate	Hot plug/unplug	HW implementation error

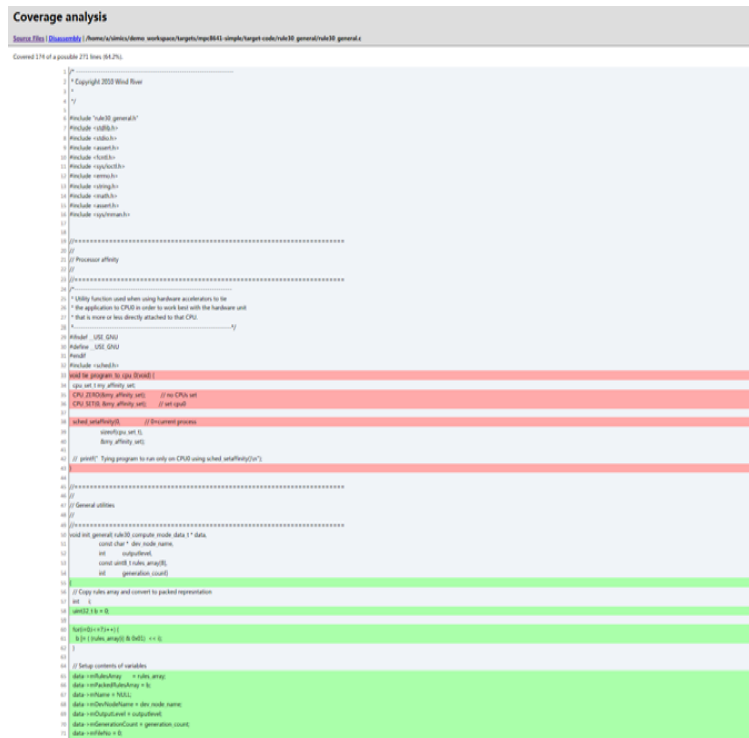
基于数据流的测试

- 能够访问不开放给软件的设备内部数据比如cach、硬件加速器等
- 跟踪任意总线或设备上的数据流
- 记录和重放数据流



基于指令流的测试

非侵入式的覆盖率测试



性能分析

```
simics> mpc8641d_simple.soc.cpu_0__branch_recorder.address-profile-data 2
start = 0x0 stop = 0x1000000
```

```
View 0 of mpc864d_simple.soc.cpu_0_branch_recorder: execution count
64-bit physical addresses, profiler granularity 4 bytes
Each cell covers 17 address bits (128 kilobytes).
```

```
column offsets:
```

[illegible]

1775888101 (1775M) counts shown. 158303996 (158M) not shown.

Simics Timing – Simics虚拟时间域

- Simics仿真的整个系统运行在统一的虚拟时间域中，所有设备保持时间同步
- 在仿真硬件和目标软件中可以引入基于虚拟时间的延时
- Simics虚拟时间可以通过API输出给外部工具用于同步和集成
- 虚拟时间域可以放慢来适应外部较慢的工具或设备

SIMICS的市场应用及案例



部分Simics客户列表

通讯
(Data & Telco)



国防和
航空航天



BAE SYSTEMS

GENERAL DYNAMICS



Honeywell

NORTHROP GRUMMAN



半导体



高端应用系统



WIND RIVER



HITACHI
Inspire the Next

Wind River Simics Models Device List

Processors/Cores

68000

Motorola 68020, 68040

8051

Intel I8051

ARM

ARM v4T
ARM v5
ARM v6
ARM v7A-AR

H8

H8/300
H8S/2000

Intel Architecture

Intel 80386
Intel 80486
Intel Pentium (MMX, Pro, II, III, 4, 4E, M)
Intel Core
Intel Core 2
Intel Nehalem
Intel Sandy Bridge
Intel Atom Bonell

MIPS

MIPS 3K
MIPS 4K
MIPS 5K/f
PMC RM7000
PMC E9000
Cavium cnMIPS64
Netlogic ec4400

Power Architecture

Freescale e300
Freescale e500
Freescale e500mc
Freescale e5500
Freescale e600
Freescale e6500

Freescale MPC603e
Freescale MPC75x ("G3")
Freescale MPC74xx ("G4")
IBM PowerPC 403
IBM PowerPC 405
IBM PowerPC 440
IBM PowerPC 464FP
IBM PowerPC 476FP
IBM PowerPC 750(fx,gx)
IBM PowerPC 970
IBM POWER6
IBM Cell Broadband Engine

SuperH

SuperH SH-4

SPARC

SPARC-V8 (Leon II)
SPARC-V9 (UltraSPARC)

Tensilica

Xtensa

Texas Instruments DSP

C64 DSP
C64+ DSP

x86 AMD

AMD Athlon
AMD Athlon64
AMD Opteron

SoCs and Chipsets

AMCC

PPC405gp
PPC440gp

Broadcom

Netlogic XLP832

Cavium

Octeon 38xx/58xx

HIREC

HR5000

Freescale

MPC8641/D
PowerQUICC II (MPC82xx)
PowerQUICC II Pro (MPC83xx)
PowerQUICC III (MPC85xx)
QorIQ P1020
QorIQ P1022
QorIQ P2020
QorIQ P2041
QorIQ P3041
QorIQ P40x0
QorIQ P5020
QorIQ T4240

Intel

Intel 440BX
Intel 5100 MCH
Intel 6300ESB
Intel 8254x, 82559, 82571, 82599 Ethernet
Intel 82801CA (ICH3S)
Intel 82870P2 (P64H2)
Intel E7501 MCH
Intel E7520 MCH
Intel ICH10
Intel ICH9R
Intel PIIX4
Intel QM67 (Huron River)
Intel US15WP (Menlow)
Intel X58 IOH (Tylersburg)
Romley platform
Crystal Forest platform

LSI

LSI ACP34xx

Marvell

Discovery III

Texas Instruments

TI C6414
TI C6455

Preconfigured Virtual Systems

ARM Versatile Express
BAE RAD750
Curtis Wright SVM183
Freescale HPC-NET
Freescale QorIQ P1/P2/P3/P4/P5 DS
Freescale PowerQUICC II, II Pro, III boards
Intel 440BX
Intel 5100 with Core 2
Intel Crystal Forest CRBs
Intel E7501
Intel E7520
Intel PC/AT
Intel QM67 with Sandy Bridge Mobile
Intel Romley CRBs
Intel US15WP with Atom Silverthorne
Intel X58 Express
Sun SPARC Servers
Technologic TS7200
Wind River SBC750

Device Types

ARINC 429
Crypto accelerators
Display Adaptors
EEPROMs
Ethernet controllers
IDE and SATA disks
I²C controllers and devices
Interrupt controllers
Memory controllers
MIL-STD-1553
NOR and NAND Flash
PCI bridges
PCI and PCIe controllers
PCIe switches

Pattern match accelerators
RapidIO controllers
RapidIO switches
Real-Time Clocks
SCSI controllers and devices
Serial devices
System controllers
Temperature sensors
Timers
USB devices and disks
+ many more

Target Operating Systems

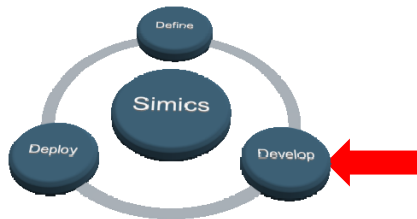
Simics is not limited by target OS; these are examples of OS known to run on Simics

Wind River VxWorks
Wind River Linux
Wind River Hypervisor
BIOS, UEFI BIOS
DOS
Enea OSE
FreeBSD
Freescale Hypervisor
GreenHills Integrity
IBM AIX
In-house RTOSes
Kernel.org Linux
Microsoft Windows
Monta Vista Linux
Ubuntu Linux
QNX Neutrino

Please visit www.windriver.com/products/simics for the most recent list.

客户案例

Honeywell
Boeing 787 Program



■ Use-case:

- 解决 DO-178B 认证测试问题

■ 问题:

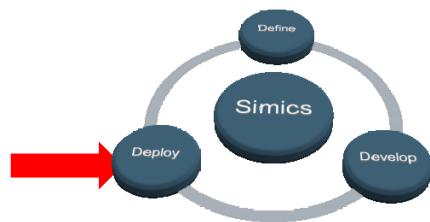
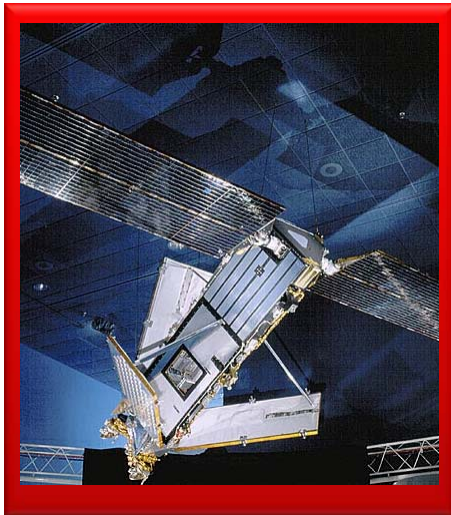
- 运行一轮认证测试需要几周的时间， 因为：
 - 有限的目标机数量
 - 硬件速度很慢
 - 脆弱的主机/目标机通讯连接
- 无法经常运行软件认证测试，导致很多回归性的Bug

■ Simics 解决方案:

- 仿真目标机，并行运行多个实例，来并行运行整个测试用例集
- 软件认证测试能够在一个晚上完成
- 每天都可以进行回归测试
- 通过仿真平台（非侵入式）收集运行数据，用于DO-178B认证要求的 A级代码覆盖率分析

客户案例

Iridium
Satellite



■ Use-case:

- 硬件可用性问题（铱星系统）
- 不断退化的硬件需要软件不断修改

■ 问题:

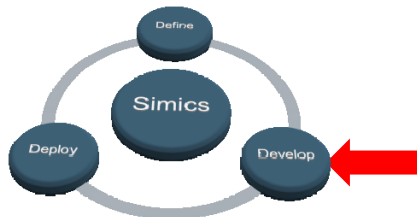
- 太空中老化的卫星系统; 需要用软件补丁来弥补硬件缺陷
- 在地面上很难创建修补用的软件补丁, 因为缺少实际硬件

■ Simics 解决方案:

- 仿真系统硬件, 并模拟硬件的退化来测试软件
- 在上传补丁之前, 能够在地面上用仿真平台完整地测试（单元测试和系统测试）

客户案例

Large European Telecom
Infrastructure Provider



■ Use-case:

- Extremely large, complex telecom system; limited developer access to complete system; system testing

■ Problem:

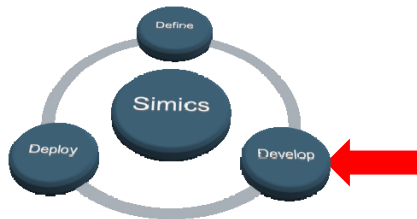
- Few people had access to the complete system; couldn't debug & test system as a whole
- Many different target architectures & RTOS's made it difficult to have a common toolset
- It took weeks to reconfigure lab system for a specific customer configuration

• Simics Solution:

- Created large, complex virtual system containing hundreds of processors, racks, devices with ATM, Ethernet, and serial connections
- Provided common development/debug platform
- Ready access to entire system for test, QA, customer support

客户案例

Chinese Mobile
Infrastructure Customer



■ Use-case:

- Complex & large system development risk mitigation & cost control with highly heterogeneous target hardware

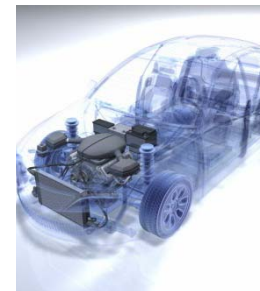
■ Problem:

- System hardware comprised of many different kinds of target architectures making debug & development difficult
- Multi-Chassis hardware cost limits access for development and test impacting quality
- Difficult to test system scaling

■ Simics Solution:

- Provide Simics model of very complex & heterogeneous systems
- Simics unique speed and scalability distribution to meet demanding use cases

总结



Simics作为系统测试平台，有如下价值：

- 提前集成，提前测试，形成统一的持续集成持续测试平台
- 搭建完整系统仿真平台，给每个工程师配备完整系统随时测试
- 实现真实硬件无法完成的测试特性：
 - 非侵入式数据流和控制流的跟踪、记录、重放、分析，包括覆盖率测试
 - 系统运行可重复
 - 运行可控、可编程，实现基于脚本的自动化输入、自动化调试和自动化测试
 - 全系统暂停和回退运行
 - 平台易修改，易实现各种设备的故障注入测试
- 与任意第三方工具的集成形成统一的测试方案

Wind River* Simics* Q&A

- 能模拟任何规模和复杂度的目标系统
- 运行无需修改的目标二进制映像
- 帮您**突破**传统的产品开发流程

加速

左移产品时间表

- * 促进系统结构提升
- * 更早开发软件
- * 更早集成与测试

-66% 产品上市时间

优化

管理复杂性

- * 管理大系统
- * 调试大系统
- * 增进工程协作

-35% 调试时间

替换

使用虚拟硬件

- * 每个工程师独立可用的目标系统
- * 升级与支持传统产品
- * 销售与市场支持

-45% 项目总开销

WIND RIVER