Virtutech Simics

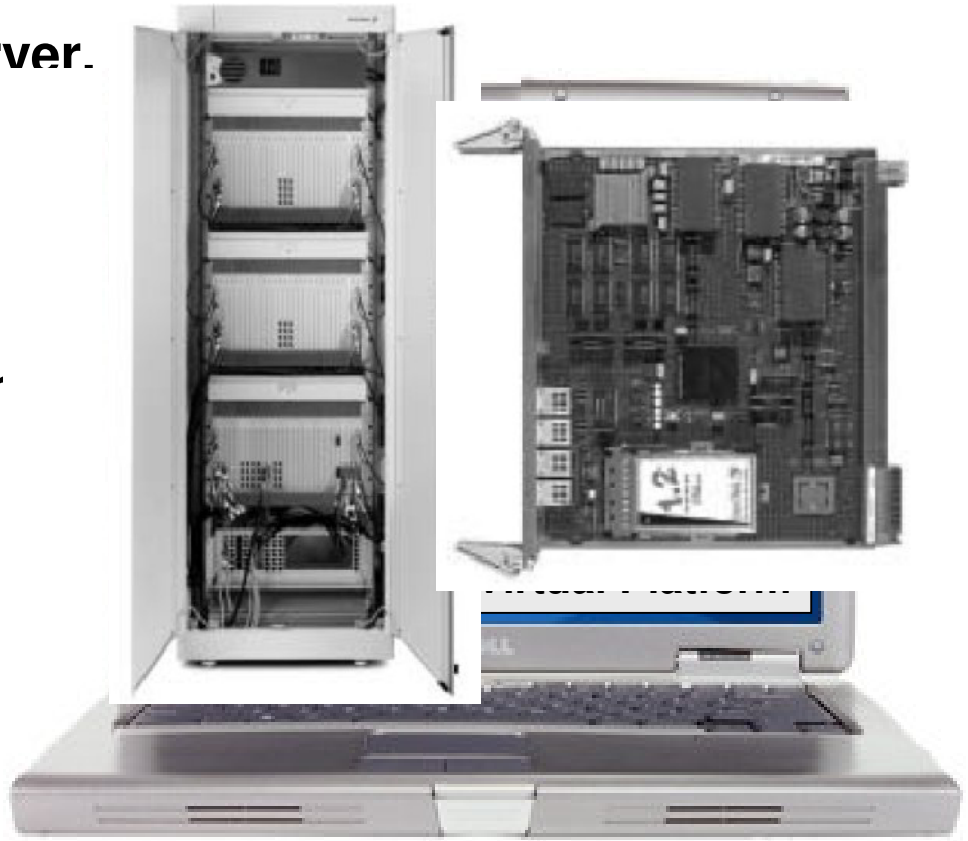
System Architecture Specification And Exploration Using a Fast Functional Simulator,

Chad Margolin, Virtutech

Introduction

What is a Virtual Platform?

- A piece of software
- Running on a regular PC, server or workstation
- Functionally identical to the target hardware
- Runs the same software as th physical hardware system

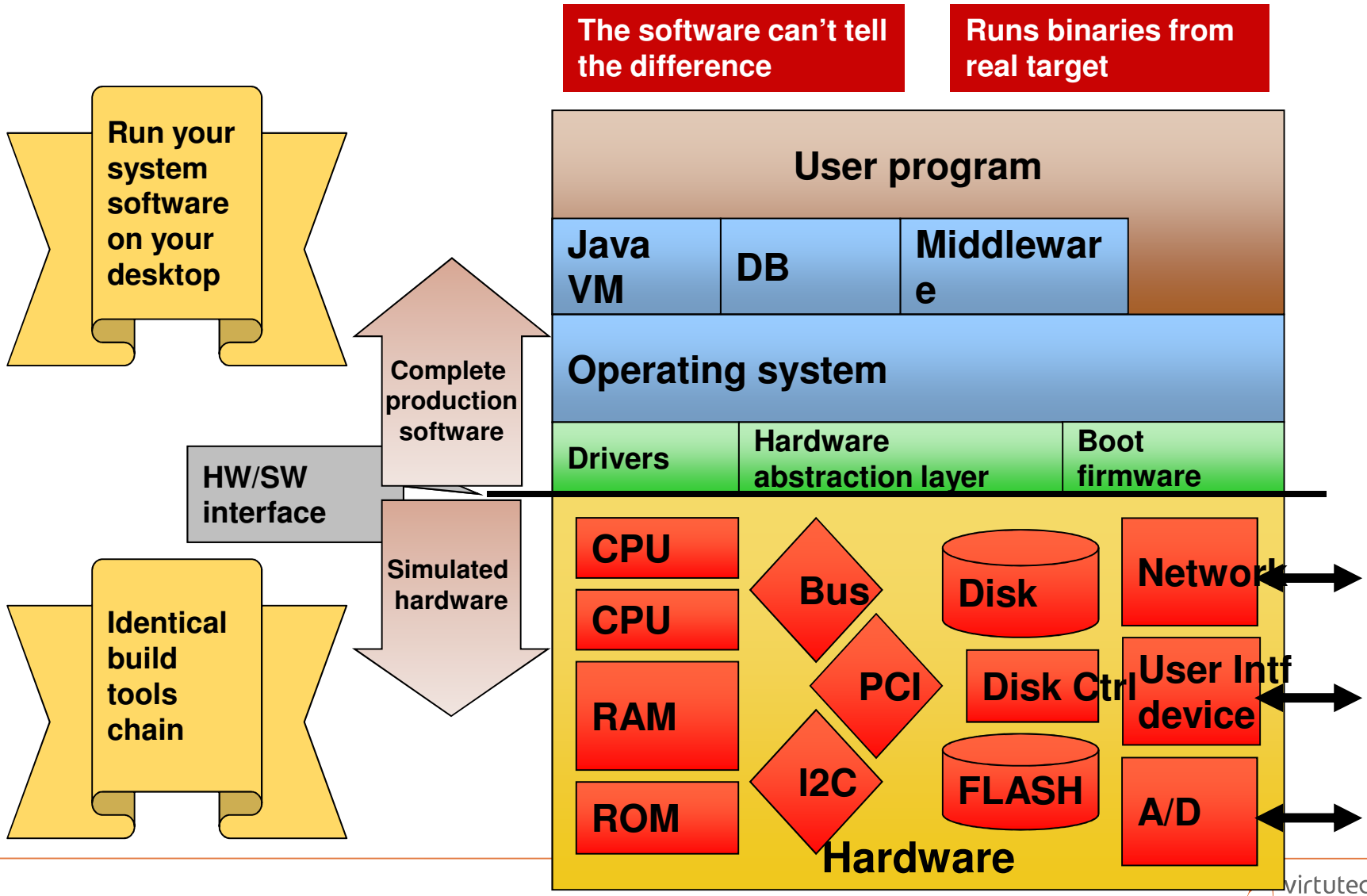


What Virtutech Does

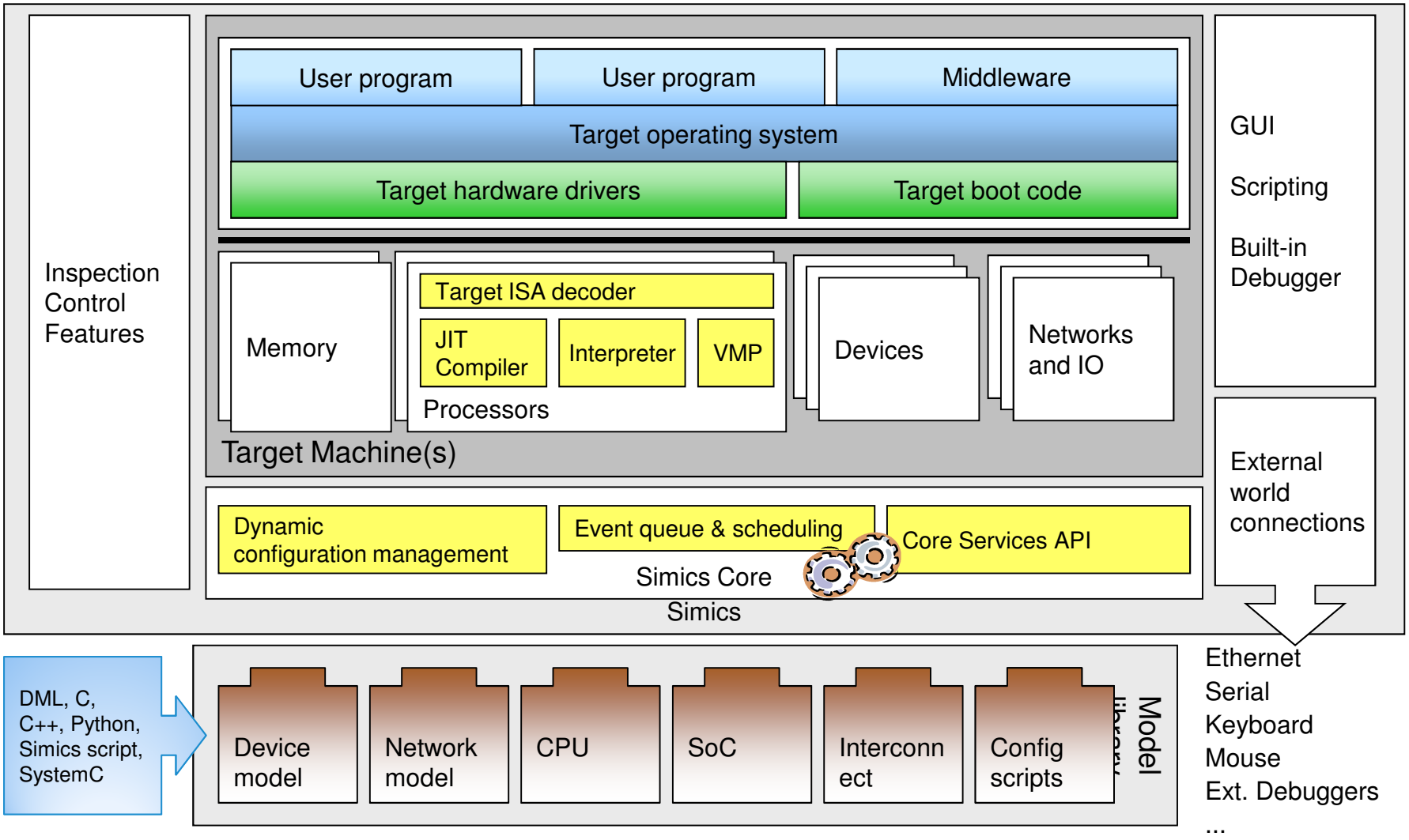
- **Provider of Simics: a high-performance, high fidelity, full system simulator**
  - **High Performance** – fast enough to run ***real*** software loads (typically 100's of MIPS, up to multiple GIPS)
  - **High Fidelity** – run full production software, including firmware, device drivers, hypervisor, RTOS/OS, application software
  - **Full System** – simulate entire systems with complete machines, networks, backplanes, etc, not just processor cores, or SoCs, or boards
  - **Development Environment** – provide the most powerful and effective develop, debug and test platform via tools and system control and visibility
- **The true value of Simics is through enablement of process change: Virtualized Software Development**

- Introduction
- **Hardware Model Architecture**

Simulating the Electronic System



Simics® Architecture



virtutech

DML

- **Declarative**
- **Domain-specific for Simics**
- **Efficient coding**
  - 5 times smaller than C
  - Quick start modeling
  - Iterative lazy development
- **Fast compiled models**
- **Models redistributable as binaries**
- **Modeling time:**
  - Days to weeks
  - Depends on model complexity

```
bank b {
  register DMA_control size 4 @ 0x20 {
    field EN  [31]  "Enable DMA";
    field SWT [30]  "Software Trigger";
    field TS [15:0] "Transfer size";
    method after_write(memop) {
      inline $do_dma_transfer();
    }
  }
  register DMA_source  size 4 @ 0x24;
  register DMA_dest    size 4 @ 0x28;

  method do_dma_transfer() {
    if ($DMA_control.EN==1) {
      local uint16 count = $DMA_control.TS;
      local uint8  local_buf[4];
      local exception_type_t result;

      while(count>0) {
        // copy memory details elided...
        $DMA_source += 4;
        $DMA_dest += 4;
        count -= 1;
      }
      // clear SWT bit, update TS
      $DMA_control.SWT = 0;
      $DMA_control.TS = count;
    } else {
      if($DMA_control.SWT==1) {
        // enable bit not set, so we cannot transfer
        log "info", 2 : "EN bit not set, SWT=1 has
     no effect";
      }
    }
...
```


virtutech

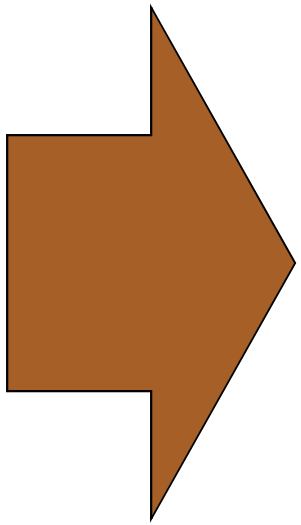
Example: moving to a Hardware Accelerator from a Software Task

- Move a software function into a hardware accelerator
- Define and refine the hardware-software interface
- Analyze the performance requirements of the accelerator
- Determine when a hardware accelerator is usefully faster than keeping a pure software implementation
- Provide an executable specification for the detailed hardware design

Evaluation Proceedures

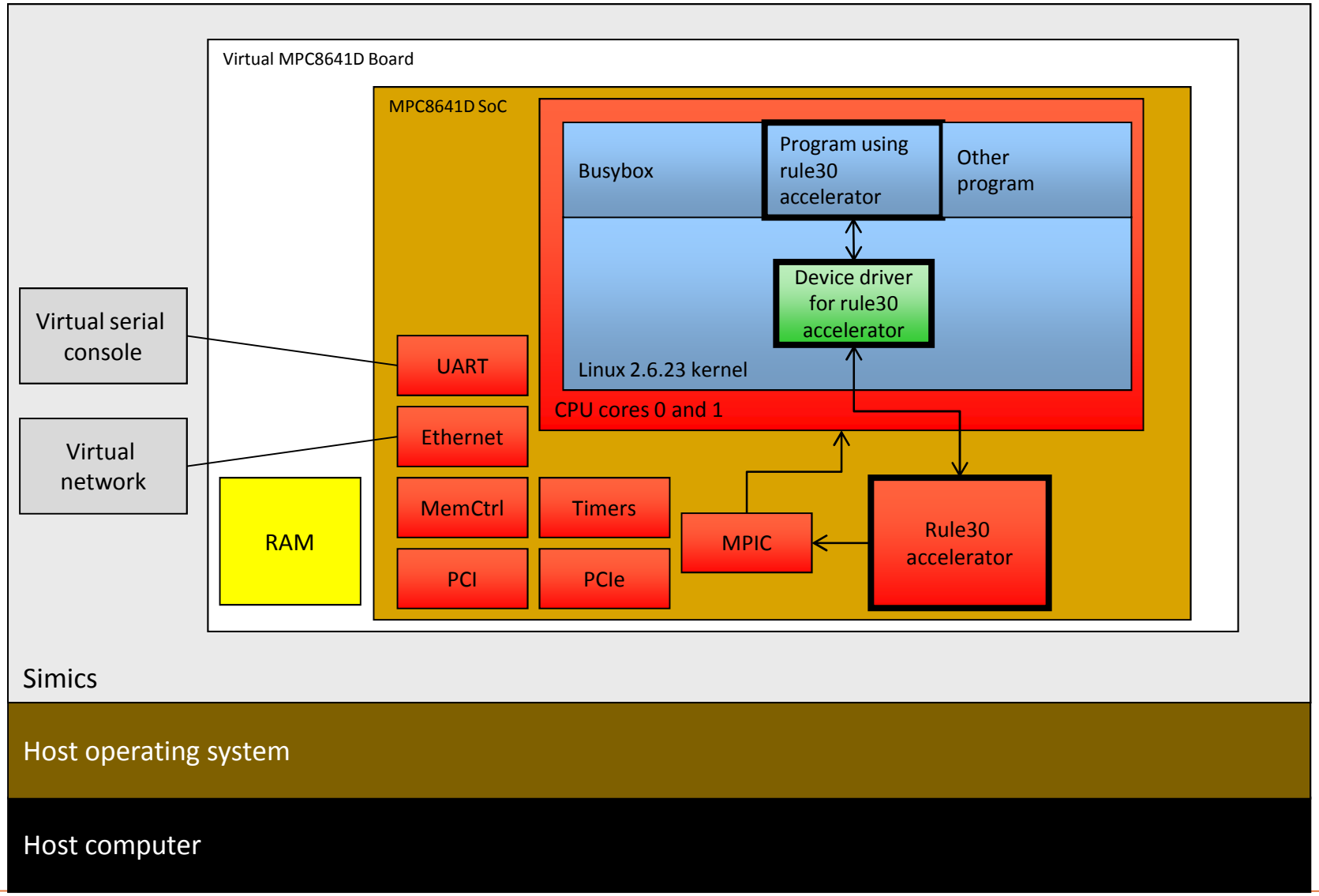
- Baseline system with rule30 implemented

- Load checkpoint
- Add the rule30 accelerator device
- Load the device driver onto the target, and initialize it
- Load the test software onto the target
- Initialize a Linux process tracker

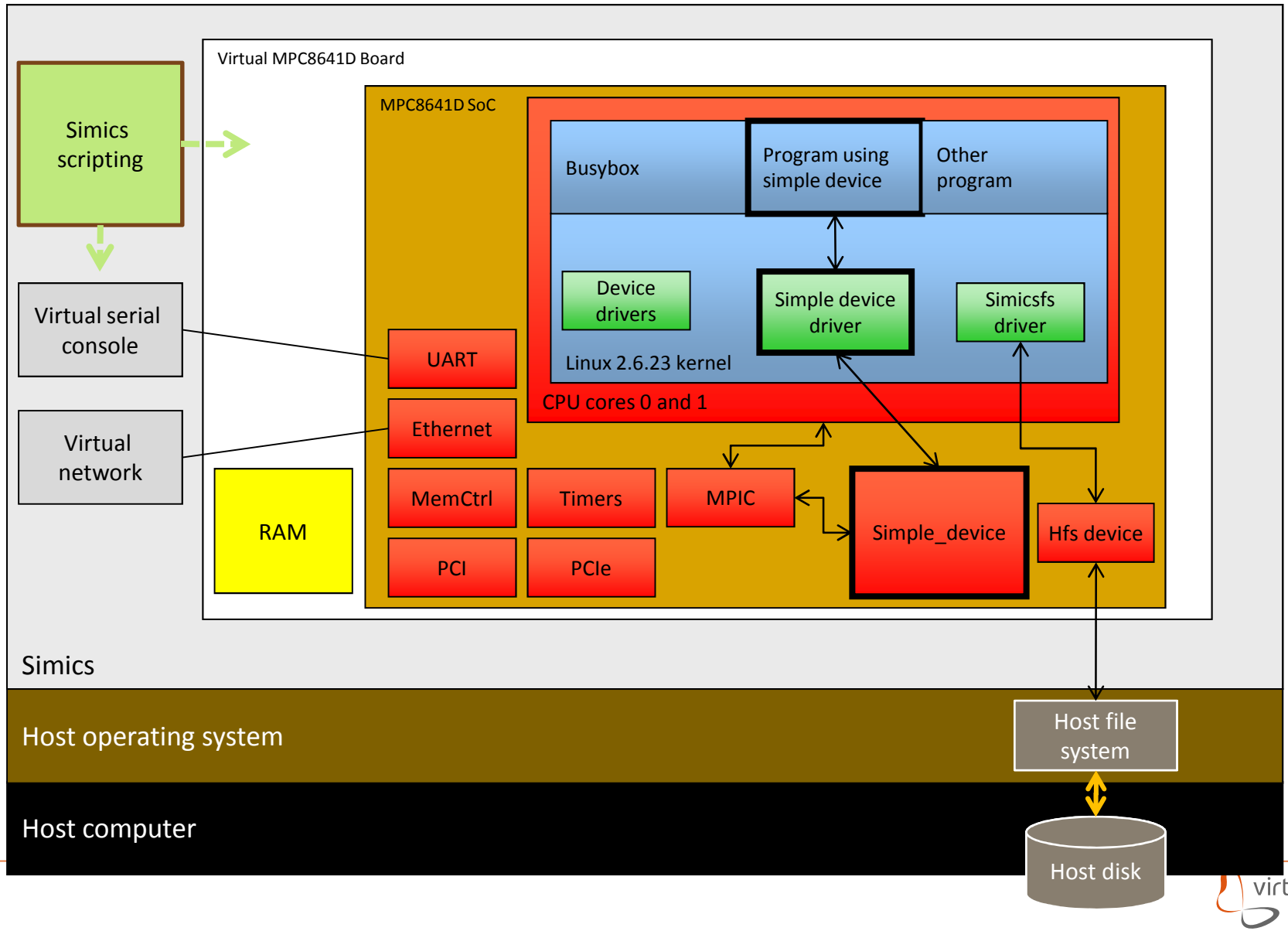


- Profiling
- Tracing
- State Inspectoin
- Hardware Control
- Software Control
- Time Control
- Scripting

Software Setup



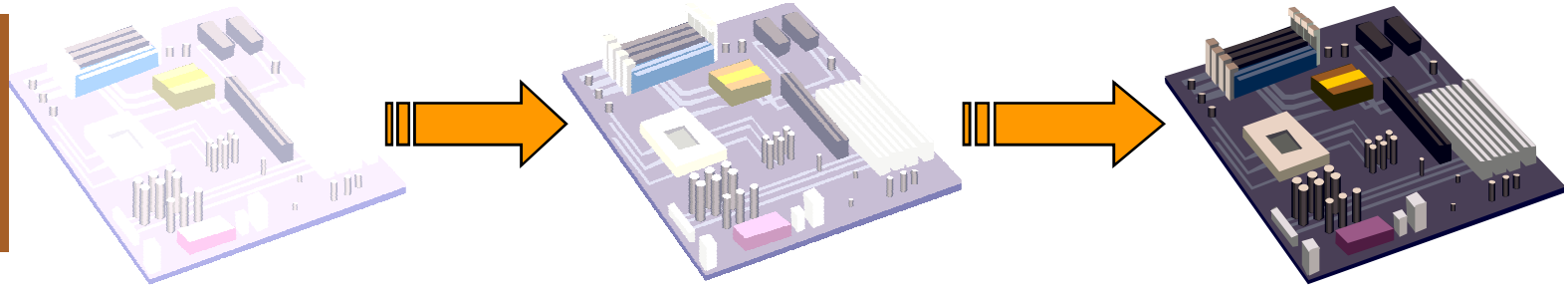
All bits


Virtual MPC8641D Board
Simics scripting
Virtual serial console
Virtual network
RAM
MPC8641D SoC
Busybox
Program using simple device
Other program
Device drivers
Simple device driver
Simicsfs driver
Linux 2.6.23 kernel
CPU cores 0 and 1
UART
Ethernet
MemCtrl
Timers
MPIC
PCI
PCIe
Simple_device
Hfs device
Simics
Host operating system
Host file system
Host computer
Host disk

- Introduction
- Hardware Model Architecture
- **Resulting Process Change**

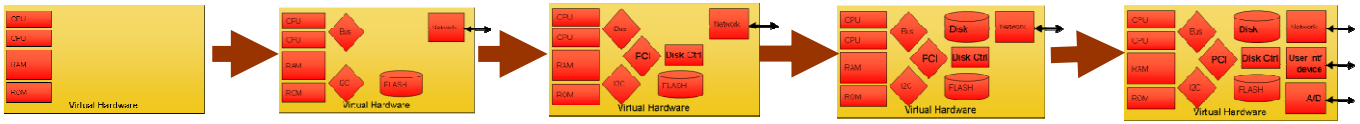
Virtual Platforms Evolve as your Hardware Evolves

Physical Hardware



- Start with virtual reference board & evolve virtual hardware as your hardware architecture evolves
- Can create new iterations faster & in less time
- Enables hardware/software co-design
- Fast feedback at each iteration
- Reduces time to market

Virtual Hardware



virtutech

- Introduction
- Hardware Model Architecture
- Resulting Process Change
- **Summary**

Summary: Advantages of a Virtutech Virtual Platform

The Virtual Platform is your asset to be modified, distributed and used by Architects, Engineers, Test/integration, Sales, Marketing and Support teams

- **Full System Simulation**
  - Simulate the entire real-world system
  - Connect the virtual platform to the physical world
- **Availability**
  - Easy to copy & distribute
  - Put on every engineer's & tester's desk
  - Requires no special lab
  - Good for global reach
  - Available before hardware is completed
- **Flexibility**
  - Evolve virtual platform as hardware design evolves
  - Fewer fixed lab setups
- **Configurability**
  - Easy to modify system and create new configurations
  - Easy to store and share configurations
- **Inspectability**
  - Any variable can be observed, even if hidden in the real world
- **Controllability**
  - Any variable or property can be changed
  - Controlled experiments, not real-world random
- **Parameterized**
  - Easy to change parameters in the design, such as memory size, processor count and speed
- **Scalability**
  - A single board up to 100's of boards with a network
  - Can be stand-alone or connected to the physical world
- **Reliability**
  - Virtual platforms are software
  - Uploads don't fail, cables don't break, nothing overheats

virtutech

- Introduction
- Hardware Model Architecture
- Resulting Process Change
- Summary
- **Questions?**

Thank you

Virtutech Simics