MULTICORE DESIGN SIMPLIFIED

imperas

Software Verification for Low Power, Safety Critical Systems

29 Nov 2016, Simon Davidmann
info@imperas.com, **Imperas Software Ltd.**

TVS

MULTICORE DESIGN SIMPLIFIED

imperas

Software Verification for Low Power, Safety Critical Systems

Agenda

- Software Verification for Systems
- Low Power, Safety Critical

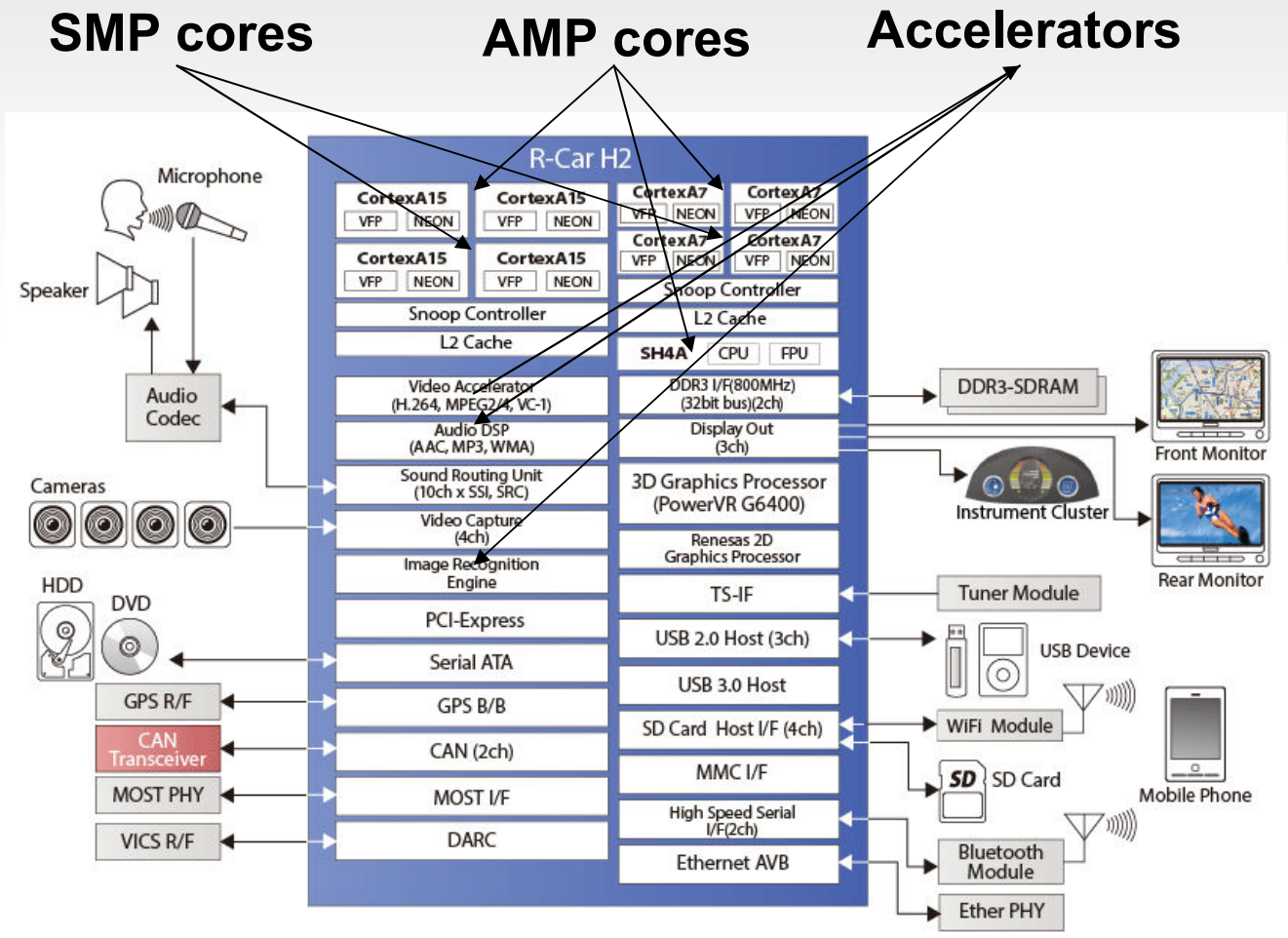
Agenda

MULTICORE DESIGN SIMPLIFIED
imperas

- Software Verification for Systems
  - Many processors with lots of software
  - Software Verification requirements
  - Embedded Software Development Issues
  - Adopting Simulation (Virtual Platforms)
  - Advanced tools available with simulators
- Low Power, Safety Critical

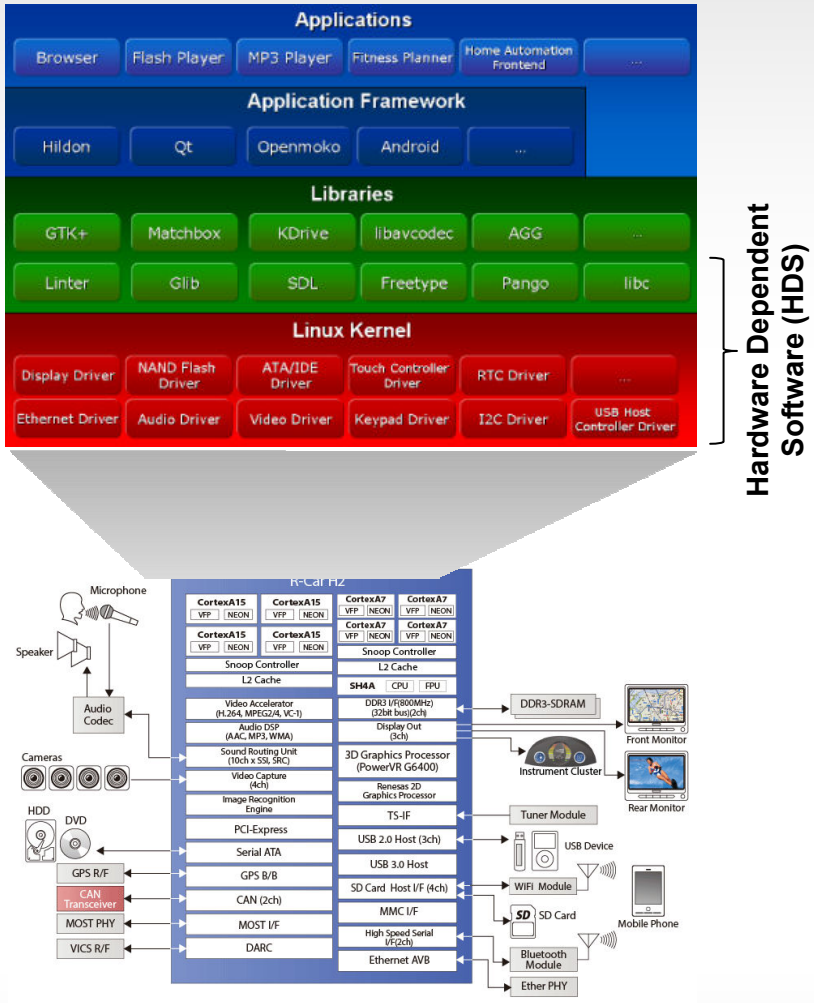
Modern SoCs Have Many Concurrent Processing Elements

MULTICORE DESIGN SIMPLIFIED
imperas



Renesas R-Car H2: Automotive infotainment and ADAS

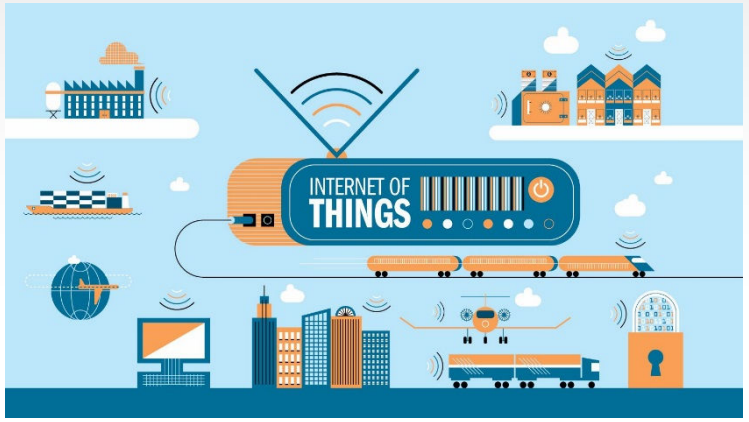
SW Verification Requirements



- Hardware Dependent Software (HDS)
  - Most complex foundation layer
    - Drivers, hypervisors, assembly libraries, operating system
  - Buried problems often appear elsewhere in a system, leading to misdirected analysis
  - Ripe for corner case type issues
  - Post development bugs hardest to fix
  - Testing needs to be platform centric not application centric
- Modern SoC SW verification is complex
  - SMP/AMP multicore interaction
  - Shared memory & devices
  - Extensive accelerators, peripherals
  - Externally authored, complex libraries
  - Complex SW/HW interaction (e.g. power)
    - How to estimate power consumption and test power management strategies over a wide range of conditions?

Embedded Software Development Issues
(in no specific order)

MULTICORE DESIGN SIMPLIFIED
imperas



- Schedule
- Quality
- Functionality
- Timing, power constraints
- Security / safety
- Predictability of the software engineering task: management accuracy on software resource and schedule requirements is +/- 50%
- Unknown / unmeasurable delivery risk

Hardware-Based Software Development

MULTICORE DESIGN SIMPLIFIED
imperas

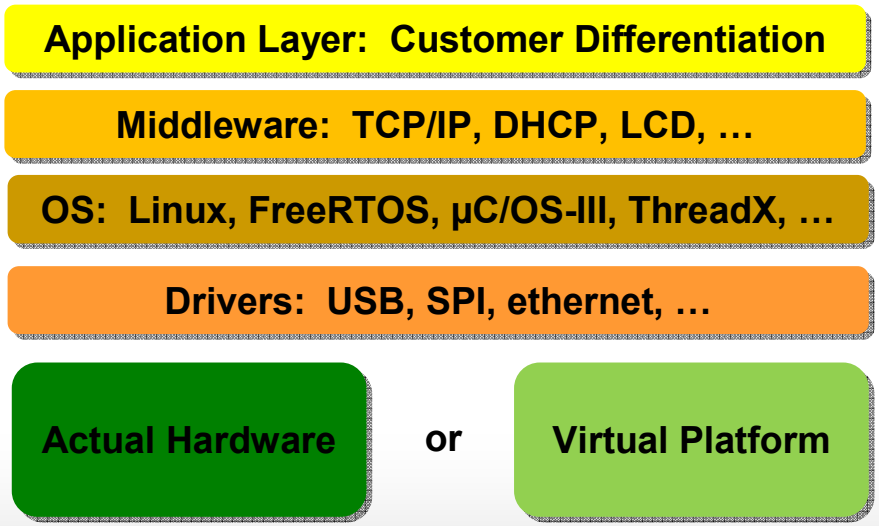
- Has timing/cycle accuracy
- JTAG-based debug, trace
- Traditional development board or hardware emulator based testing
  - Late to arrive
  - Limited physical system availability
  - Emulators are too slow to run enough system scenarios
  - Limited external test access (controllability)
  - Limited internal visibility
    - How to observe power consumption?
- To get around these limitations, software is modified
  - printf
  - Debug versions of OS kernels
  - Instrumentation for specific analytical tools, e.g. code coverage, profiling
  - Even using different tool chains and libraries

- Modified software may not have the same behavior as clean source code

Virtual Platforms Complement Hardware-Based Software Development

MULTICORE DESIGN SIMPLIFIED
imperas

- Current methodology employs testing on hardware
  - Proven methodology
  - Has limitations
  - We are at the breaking point
- Virtual platform based methodology delivers controllability, visibility, repeatability, automation

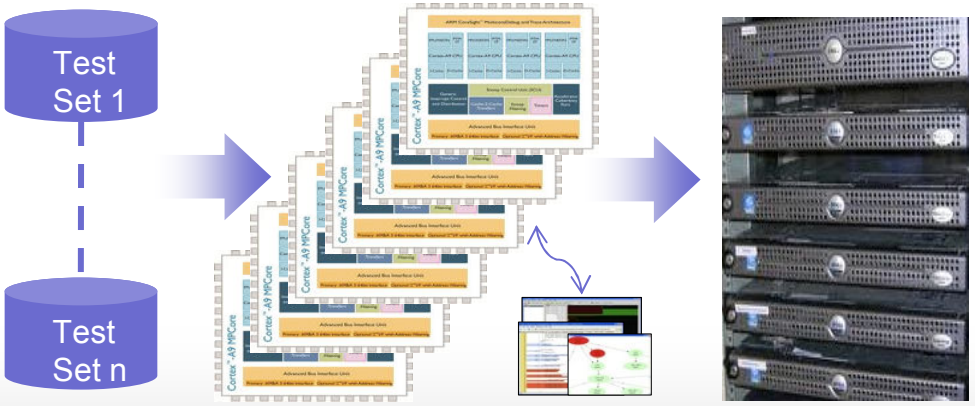


Virtual platforms – software simulation – provide a complementary technology to the current hardware-based methodology

Advantages of Virtual Platform Based Software Development
(Instruction Accurate Simulation)

MULTICORE DESIGN SIMPLIFIED
imperas

- Earlier system availability
- Easy access for entire team
- Runs actual binaries, e.g. runs ARM executables on x86 host
- Fast, enables quick turnaround and comprehensive testing
- Full controllability of platform both from external ports and internal nodes
  - Corner cases can be tested
  - Errors can be made to happen
- Full visibility into platform: if an error occurs, it will be observed by the test environment
- Easy to replicate platform and test environment to support automated continuous integration (CI) and test automation / regression testing on compute farms

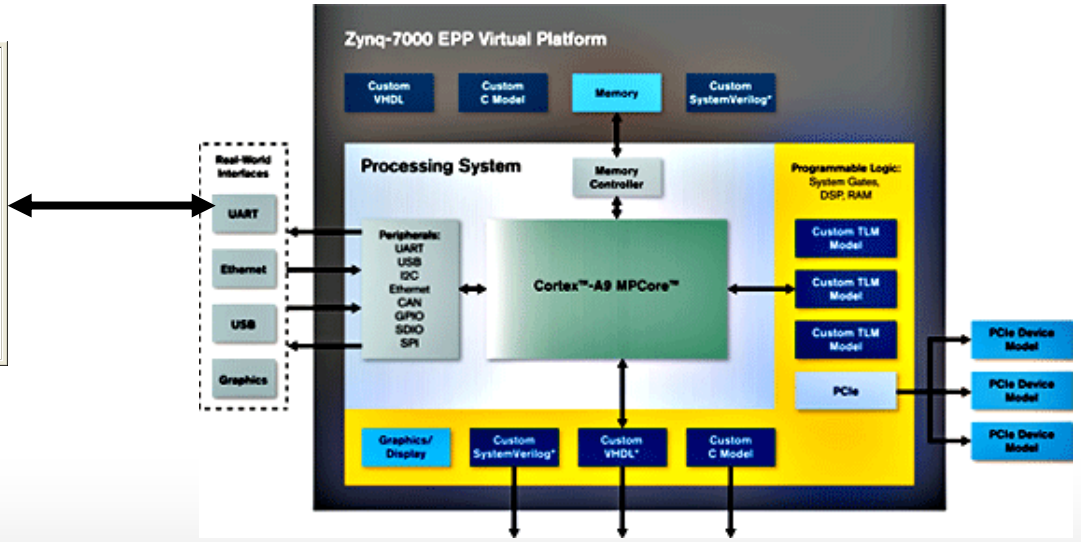


Building the Virtual Platform

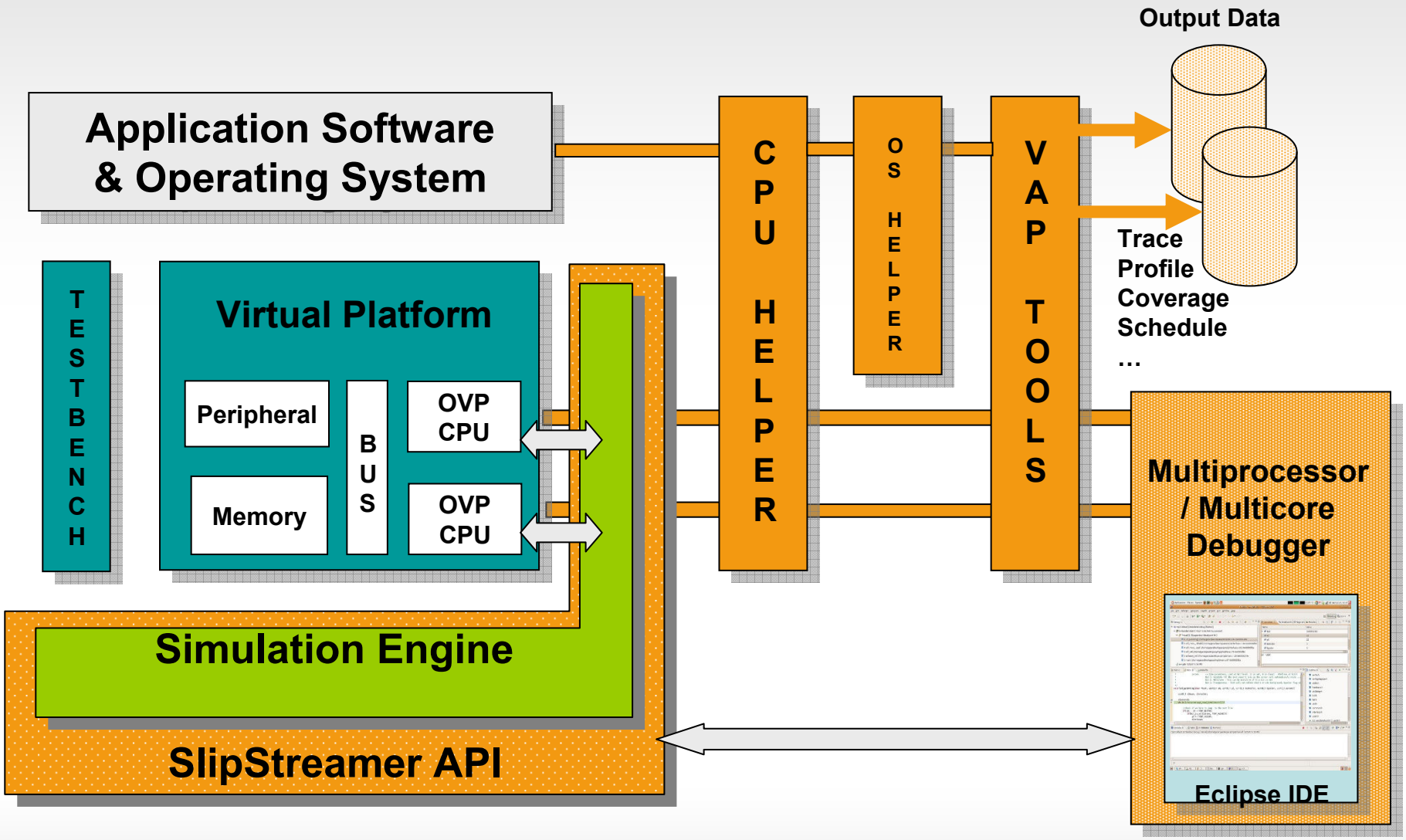
MULTICORE DESIGN SIMPLIFIED
Imperas

- The virtual platform is a set of models that reflects the hardware on which the software will execute
  - Could be 1 SoC, multiple SoCs, board, system; no physical limitations
  - Functionally accurate, such that the software does not know that it is not running on the hardware
- Models are typically written in C or SystemC
- Models for individual components – interrupt controller, UART, ethernet, … – are connected just like in the hardware
- Peripheral components can be connected to the real world by using the host workstation resources: keyboard, mouse, screen, ethernet, USB, …

```
rtc-pl031 mb:rtc: rtc core: registered pl031 as rtc0
mmci-pl18x mb:mmci: mmc0: PL181 manf 41 rev0 at 0x10005000 irq 41,42 (pio)
usbcore: registered new interface driver usbhid
usbhid: USB HID core driver
ALSA device list:
  No soundcards found.
oprofile: using arm/armv7-ca9
TCP cubic registered
NET: Registered protocol family 17
VFP support v0.3: implementor 41 architecture 3 part 30 variant 9 rev 2
rtc-pl031 mb:rtc: setting system clock to 1970-01-01 00:00:00 UTC (0)
Freeing init memory: 172K
input: AT Raw Set 2 keyboard as /devices/mb:kmi0/serio0/input/input0
input: ImExPS/2 Generic Explorer Mouse as /devices/mb:kmi1/serio1/input/input1

This root FS contains most basic linux utilities (implemented with busybox)
and the Lynx web browser.

Kernel config is available through /proc/config.gz

Welcome to OVP simulation from Imperas

Log in as root with no password.
Imperas login:
```

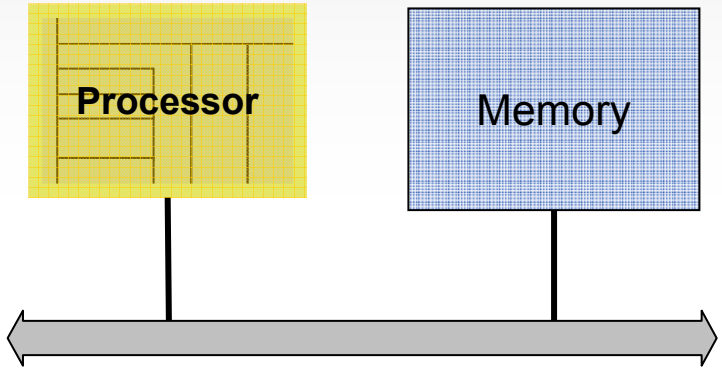


Imperas Tool Architecture



Running Software on a Virtual Platform BareMetal Platform

MULTICORE DESIGN SIMPLIFIED
imperas

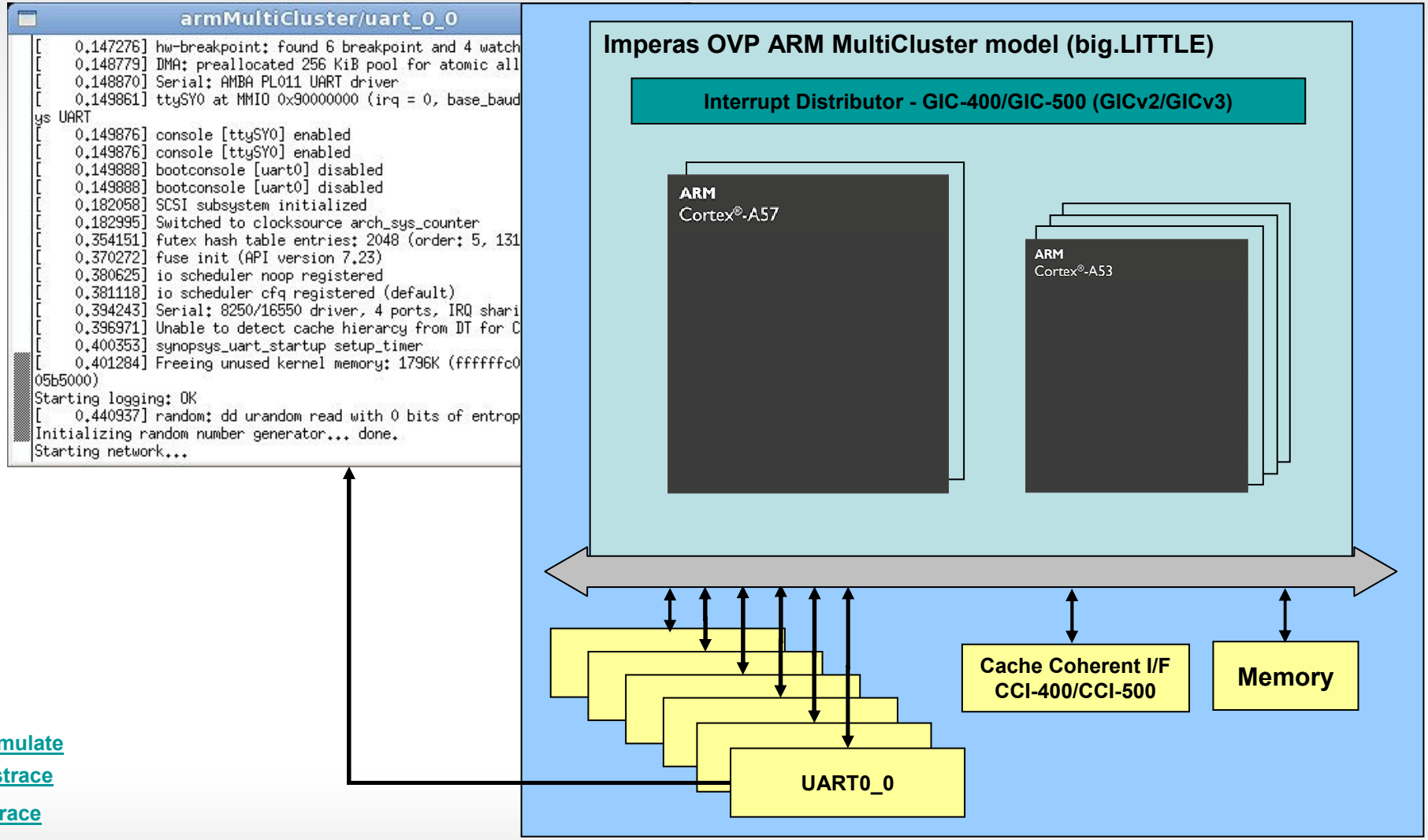


- Quickly execute any program on a ‘default’ BareMetal platform
- Simply run the executable with the name of the application object

simulate
fast

Virtual Platform Booting Linux Kernel on ARM Cortex-A57 / Cortex-A53 big.LITTLE

MULTICORE DESIGN SIMPLIFIED
imperas



simulate
ostrace
qtrace

And many many tools can be used with the simulation

MULTICORE DESIGN SIMPLIFIED
imperas

- Trace
- Coverage
- Profile
- Memory analysis
- Dynamic (Temporal) Assertions
- osAware: context, schedule, task, …

- All at the flick of a switch…
- All non-intrusively…
- All without requiring any changes to software…
  - Runs with production binaries

=> Use of simulation (virtual platforms) is the modern way to develop and verify embedded software

And…

- When it comes to Continuous Integration
- When it comes to test automation
- When it comes to regression testing

=> Virtual platforms become essential for embedded software development

Agenda

MULTICORE DESIGN SIMPLIFIED
imperas

- Software Verification for Systems
- Low Power, Safety Critical

Collaborations

MULTICORE DESIGN SIMPLIFIED
imperas

Pontifical Catholic University of Rio Grande do Sul, (PUCRS) Porto Alegre Brazil.

University of Leicester, Leicester, UK.

Institute for Information Technology, Oldenburg, Germany

This work was partially funded by the European Union's Horizon2020 research and innovation programme under the grant agreement No 687902.

More information at
http://www.safepower-project.eu/

Working on Verification of Software

MULTICORE DESIGN SIMPLIFIED
imperas

- Timing Analysis
- Power Analysis
- Fault Simulation

Working on Verification of Software

MULTICORE DESIGN SIMPLIFIED
imperas

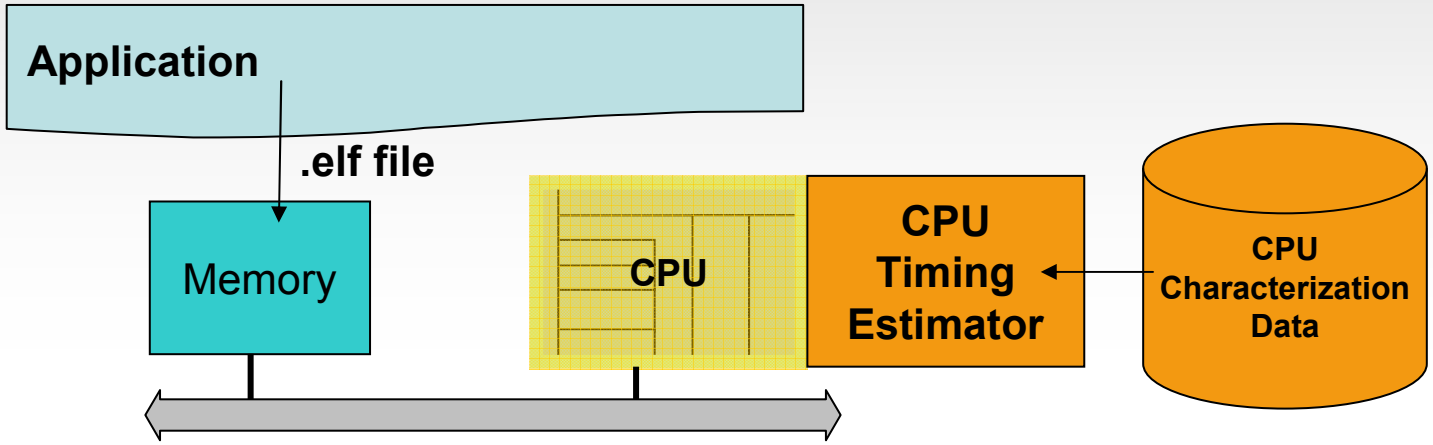
- Timing Analysis
  - using Instruction Accurate simulation / virtual platforms

  => software performance
- Power Analysis
  - using Instruction Accurate simulation / virtual platforms

  => software affected power
- Fault Simulation
  - using Instruction Accurate simulation / virtual platforms for software verification and compliance to standards such as ISO 26262

  => software fault tolerance

CPU Timing Estimator

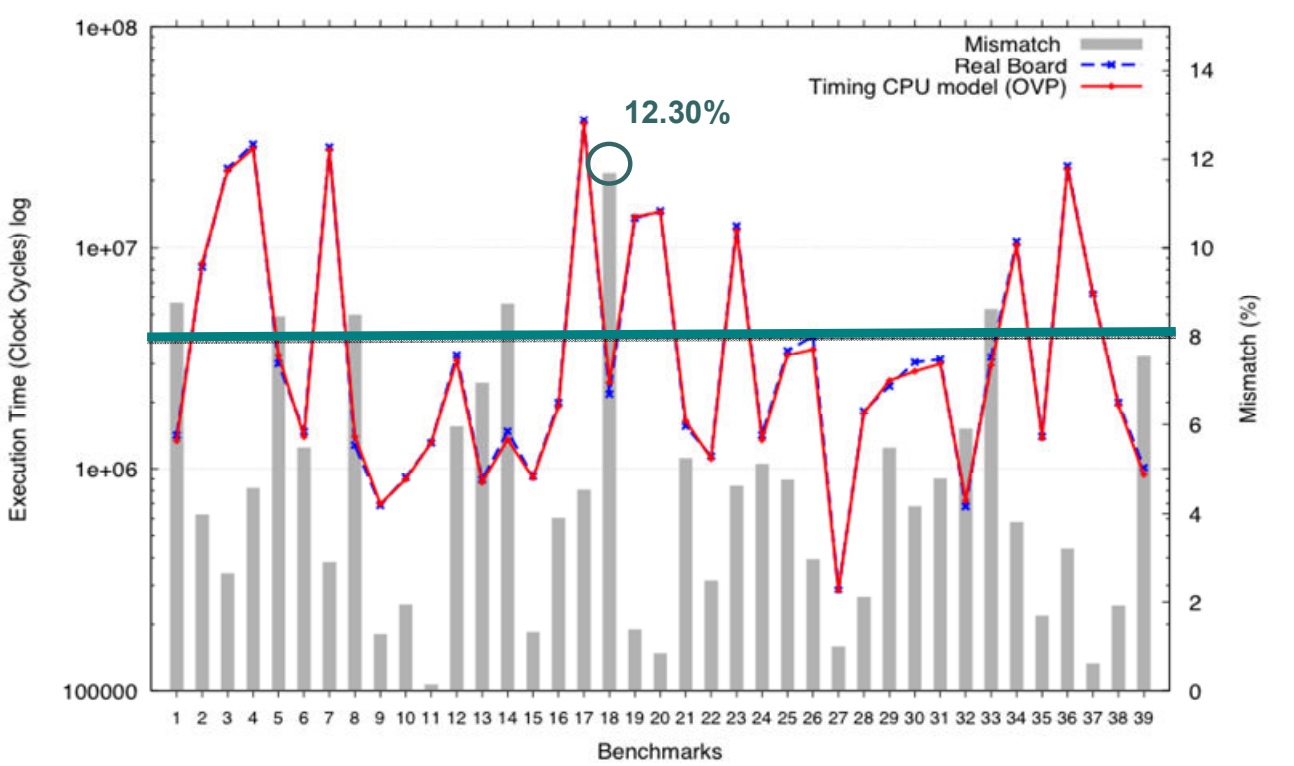


- CPU Characterization Data for each CPU variant
  - Limitation: In-order processors
- Timing Estimator loaded onto CPU instance as binary intercept library
  - No edits/changes needed in CPU model binary, or platform, or other models
- Controlled by simulation command line arguments
- Takes account of instructions, branches, sequences, memory

Accuracy Evaluation vs Board

MULTICORE DESIGN SIMPLIFIED
imperas

- Example:
  - Cortex-M4F running FreeRTOS
  - WCET and other benchmarks



- Worst case error <13%
- Average error across all benchmarks run <5%
- Most errors are <8%
- Simulation speeds with timing up to 500Mips
- Caveat: performance and accuracy are application dependent

Working on Verification of Software

MULTICORE DESIGN SIMPLIFIED
imperas

- Timing Analysis
- Power Analysis
- Fault Simulation

European Commission | Horizon 2020 European Union funding for Research & Innovation

MULTICORE DESIGN SIMPLIFIED imperas

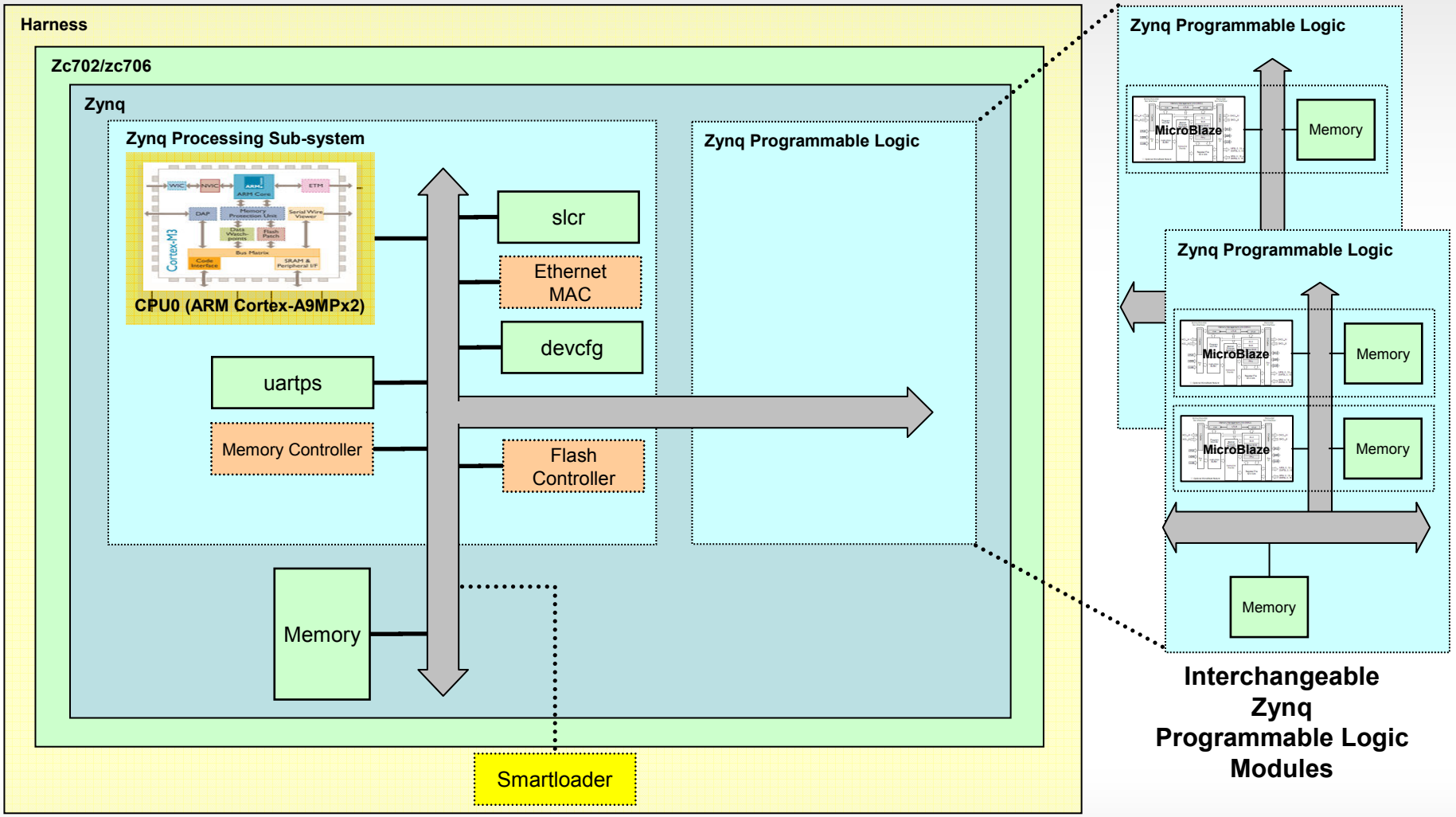
SAFEPOWER

Eu SafePower project

- Explorative project to enable the development of low power mixed-criticality systems through the provision of a reference architecture, platforms and tools to facilitate the development, testing, and validation

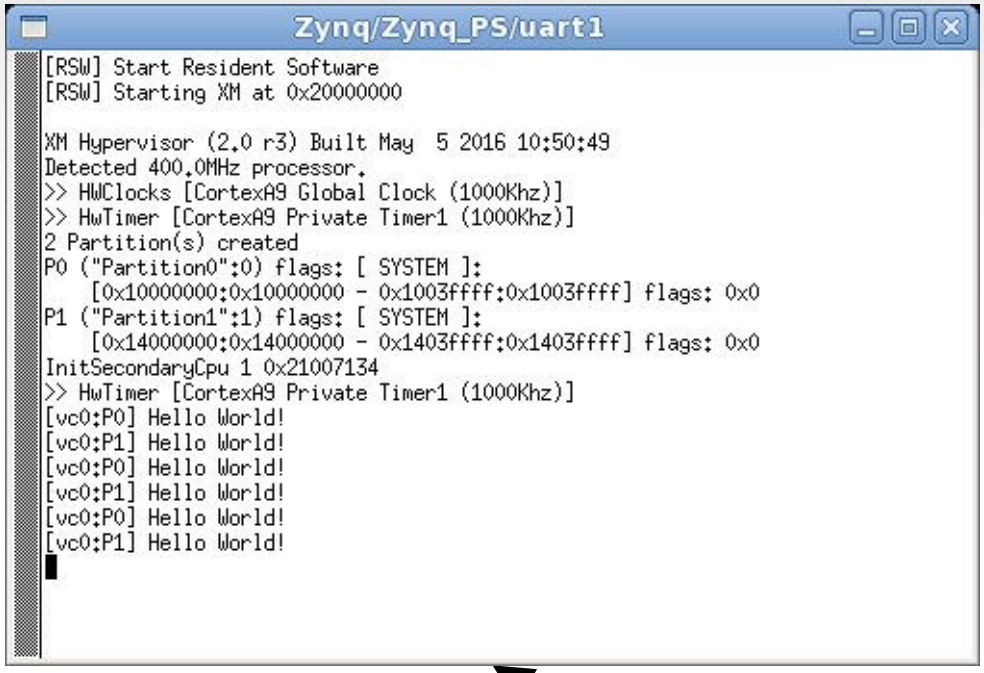
SafePower
Xilinx Zynq Virtual Platform

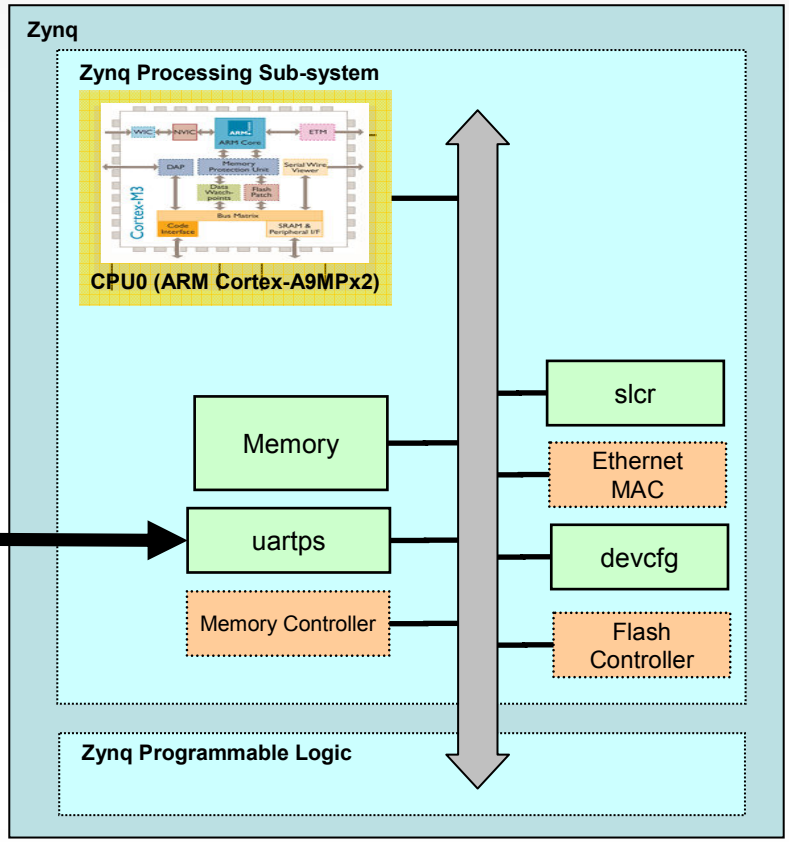
MULTICORE DESIGN SIMPLIFIED
imperas



SafePower:
Xilinx Zynq zc706 Virtual Platform
fentISS XtratuM Hypervisor on ARM Cortex-A9MPx2

MULTICORE DESIGN SIMPLIFIED
imperas

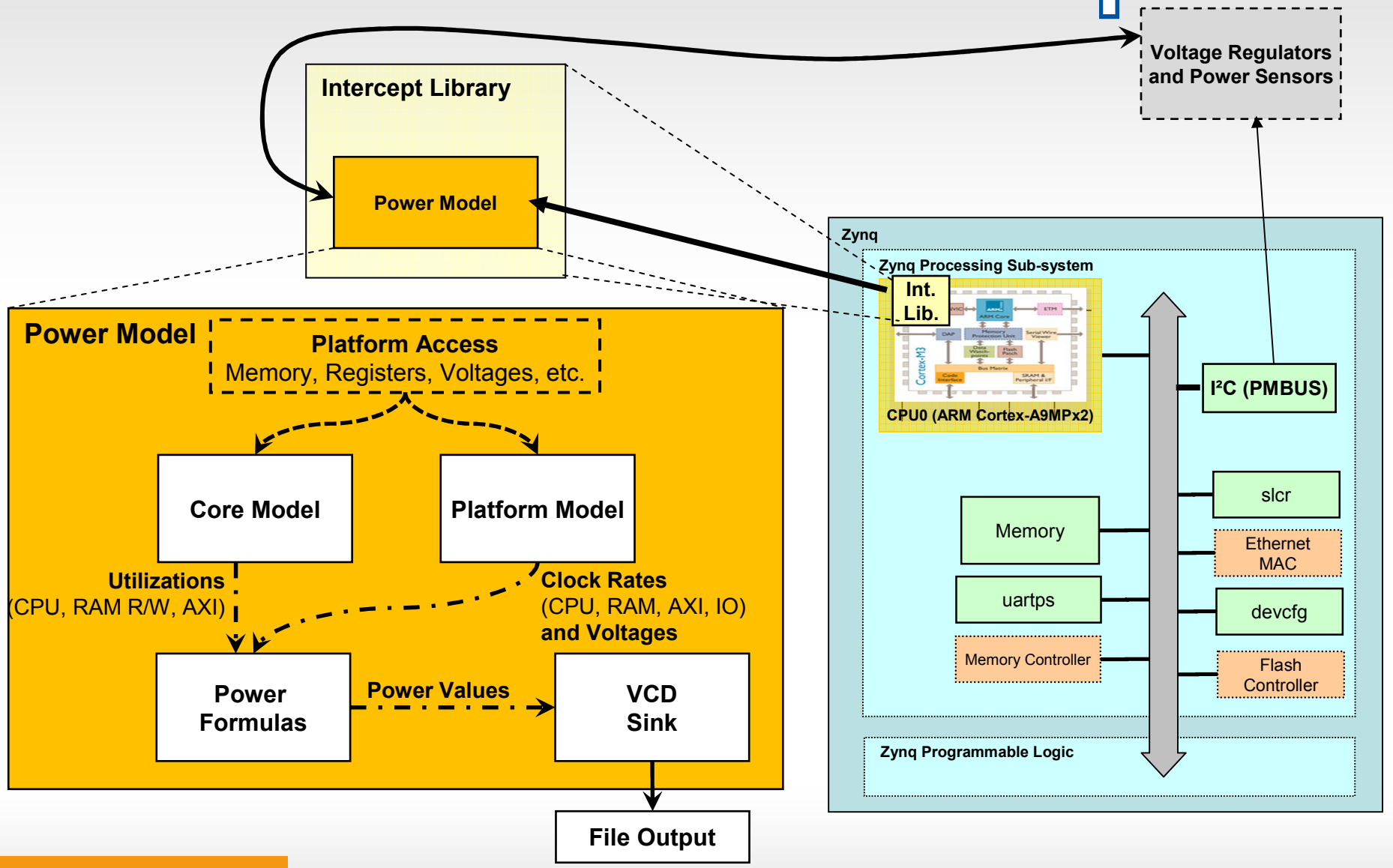




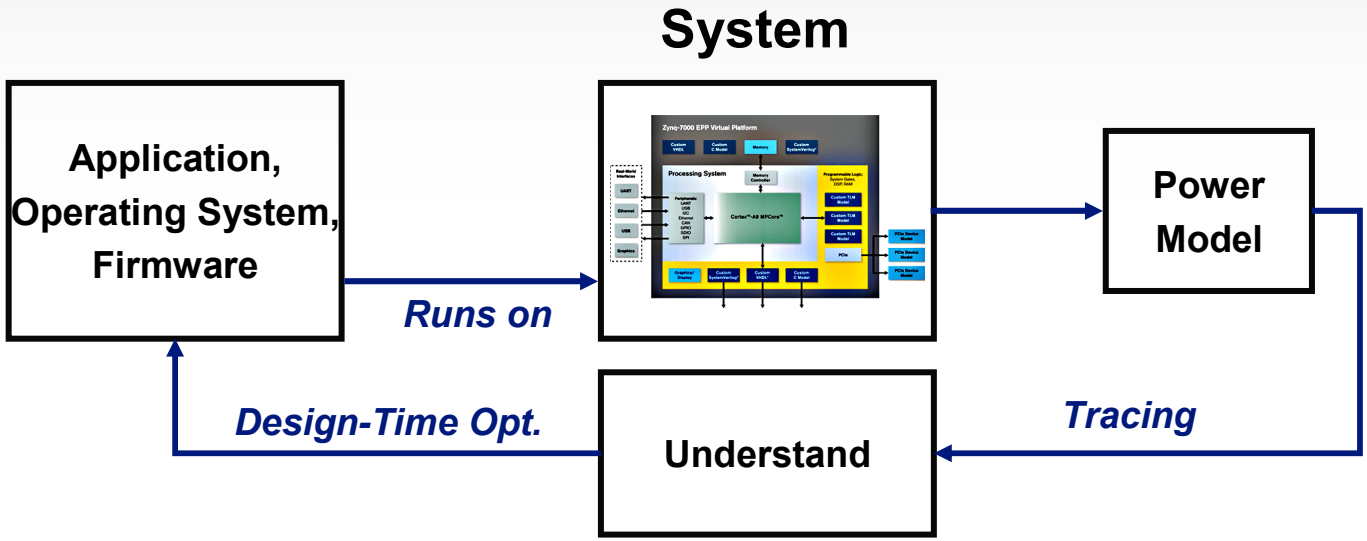
Shown running fentISS Xtratum 'Hello World' example application

SafePower ARM Cortex-A9 Power Model Overview

MULTICORE DESIGN SIMPLIFIED
Imperas

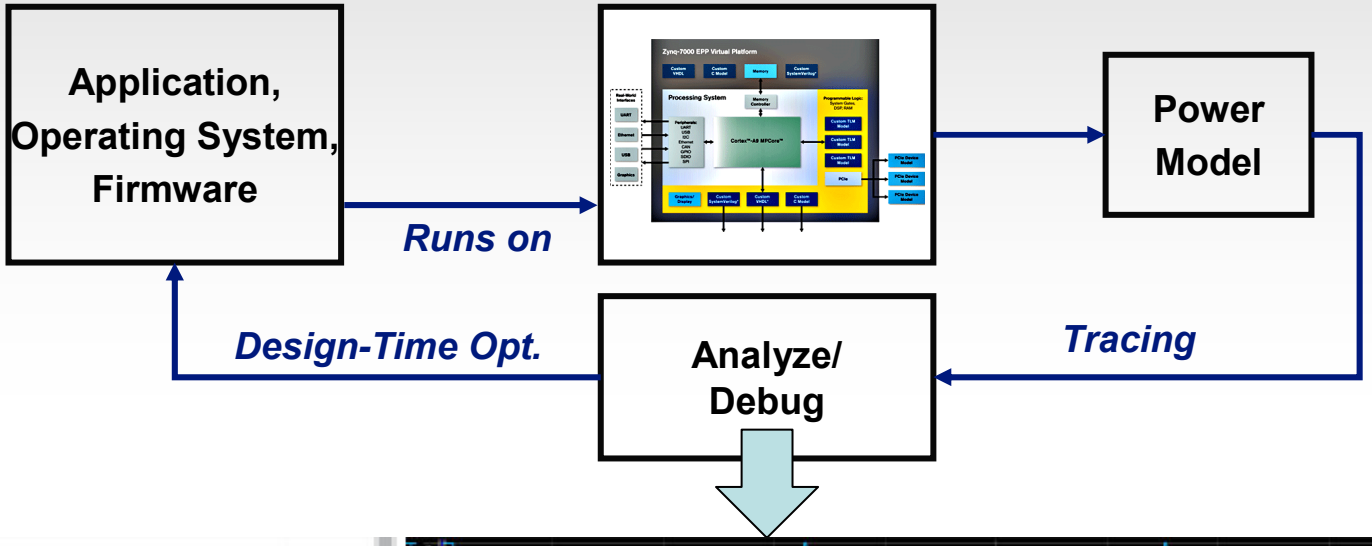


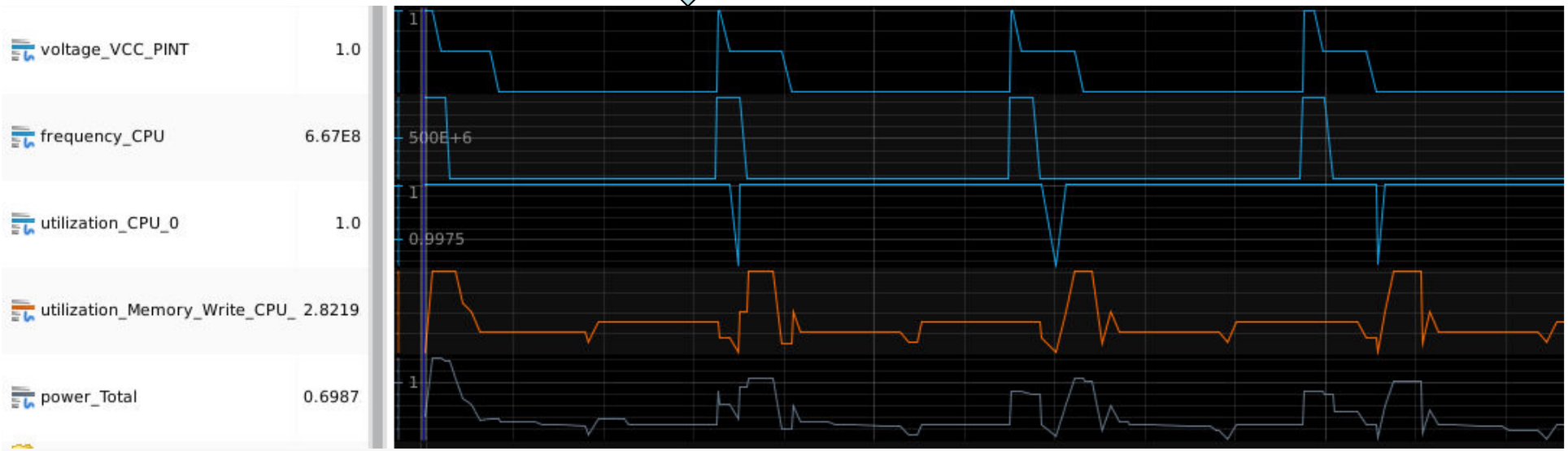
SafePower Dynamic Frequency Voltage Scaling (DVFS) Analysis



SafePower
Power Traces Analysis

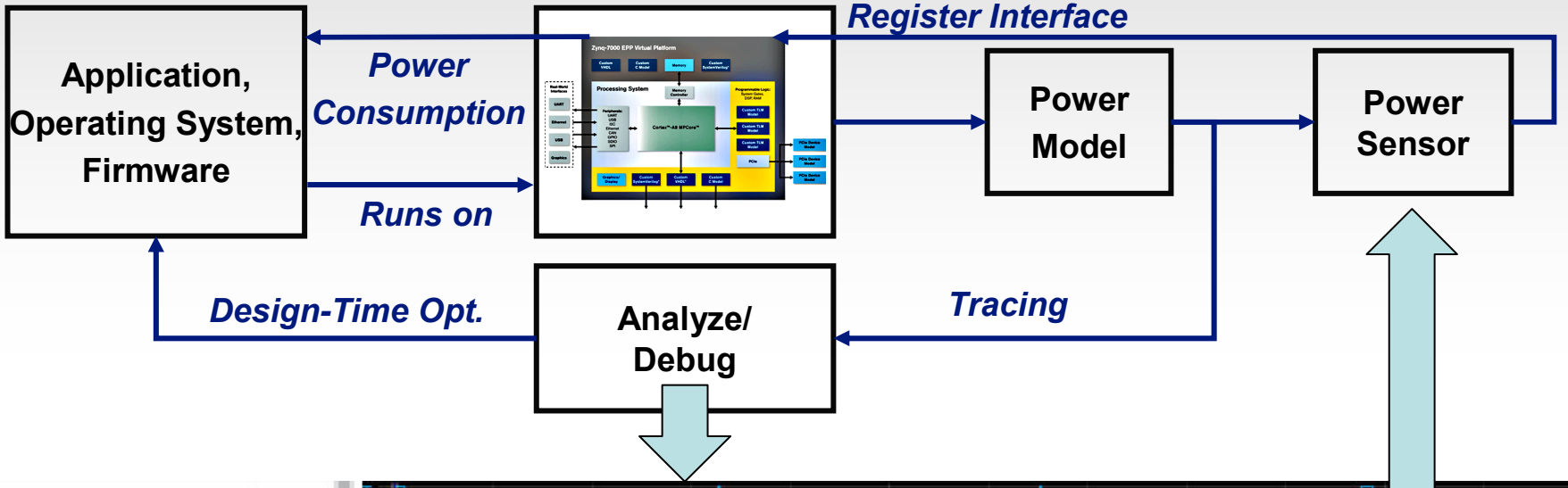
MULTICORE DESIGN SIMPLIFIED
imperas

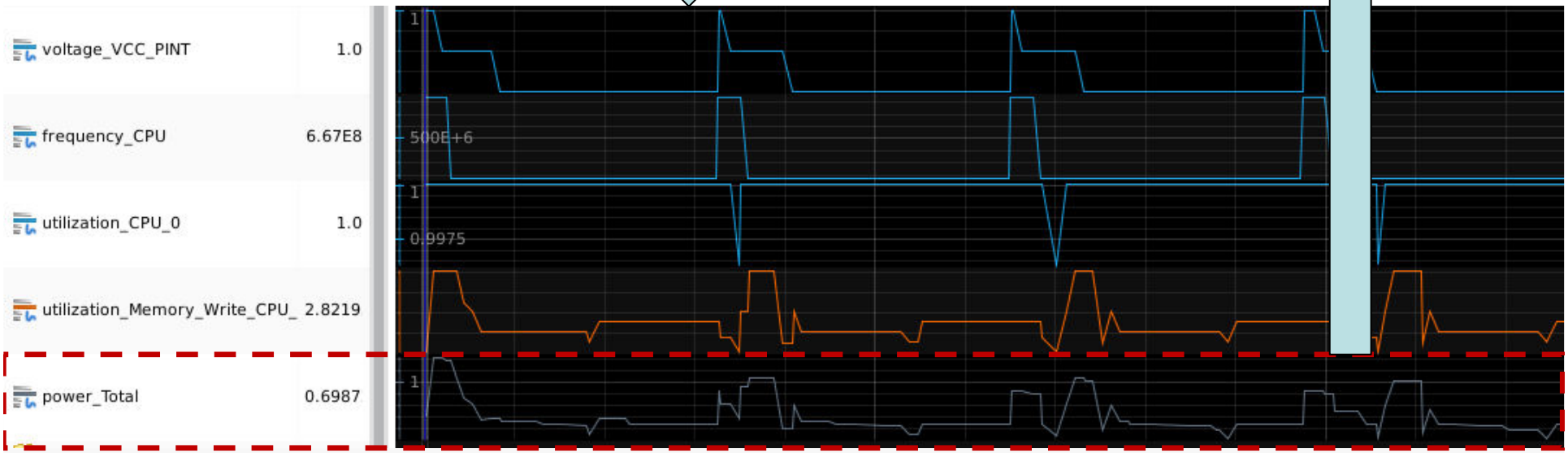




SafePower : Power Monitoring in SW

MULTICORE DESIGN SIMPLIFIED
Imperas





Working on Verification of Software

MULTICORE DESIGN SIMPLIFIED
imperas

- Timing Analysis
- Power Analysis
- Fault Simulation

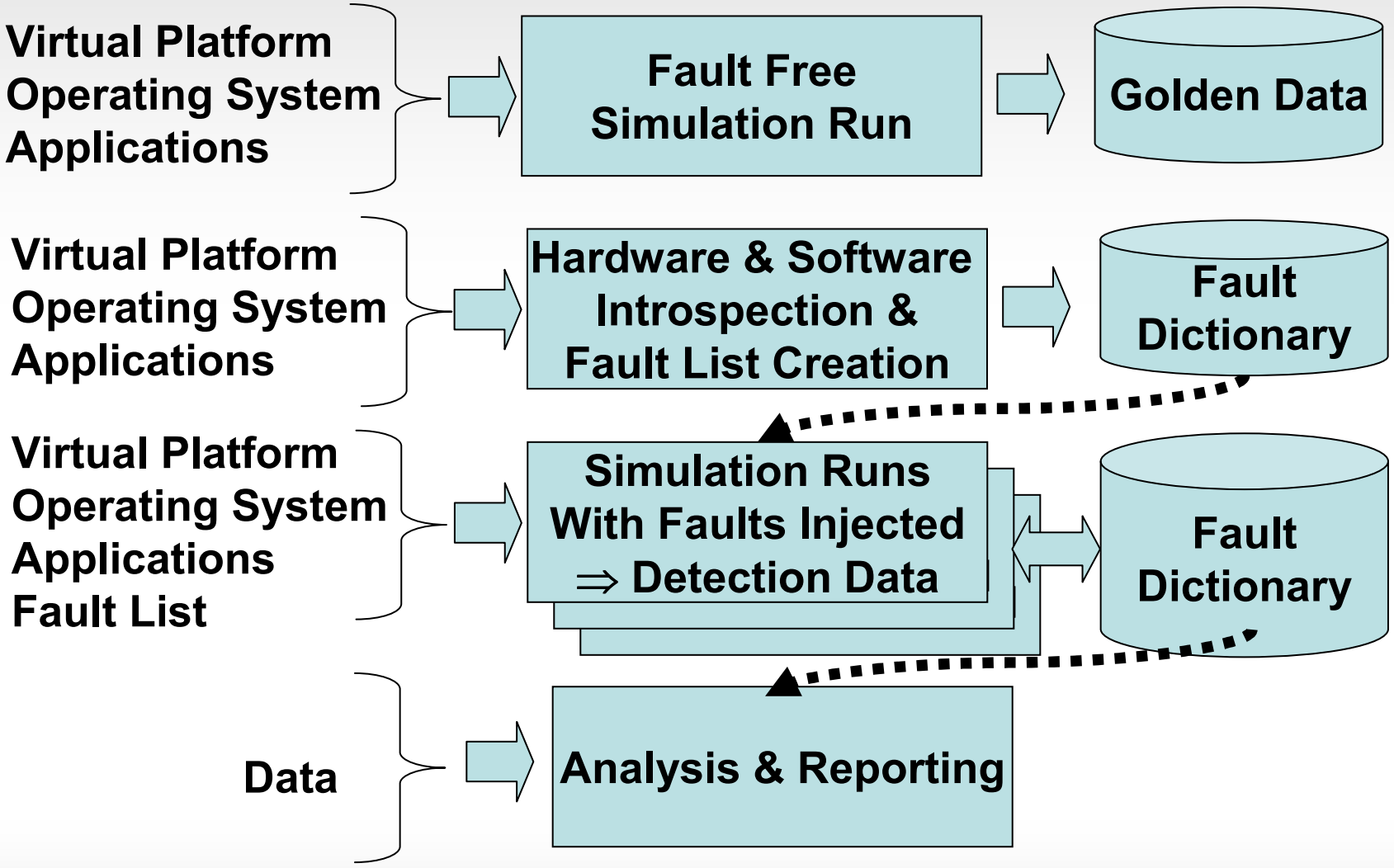
Fault Simulation is needed

MULTICORE DESIGN SIMPLIFIED
Imperas

- Compliance with standards requires products to demonstrate reliability and tolerance to fault injections
  - Eg: automotive ISO26262 requires this
- Fault Simulation proved in the HW world it was a good way to show hardware had been manufactured fault free
- Fault Simulation can be used ***with virtual platforms*** to show that software is reliable under the presence of faults

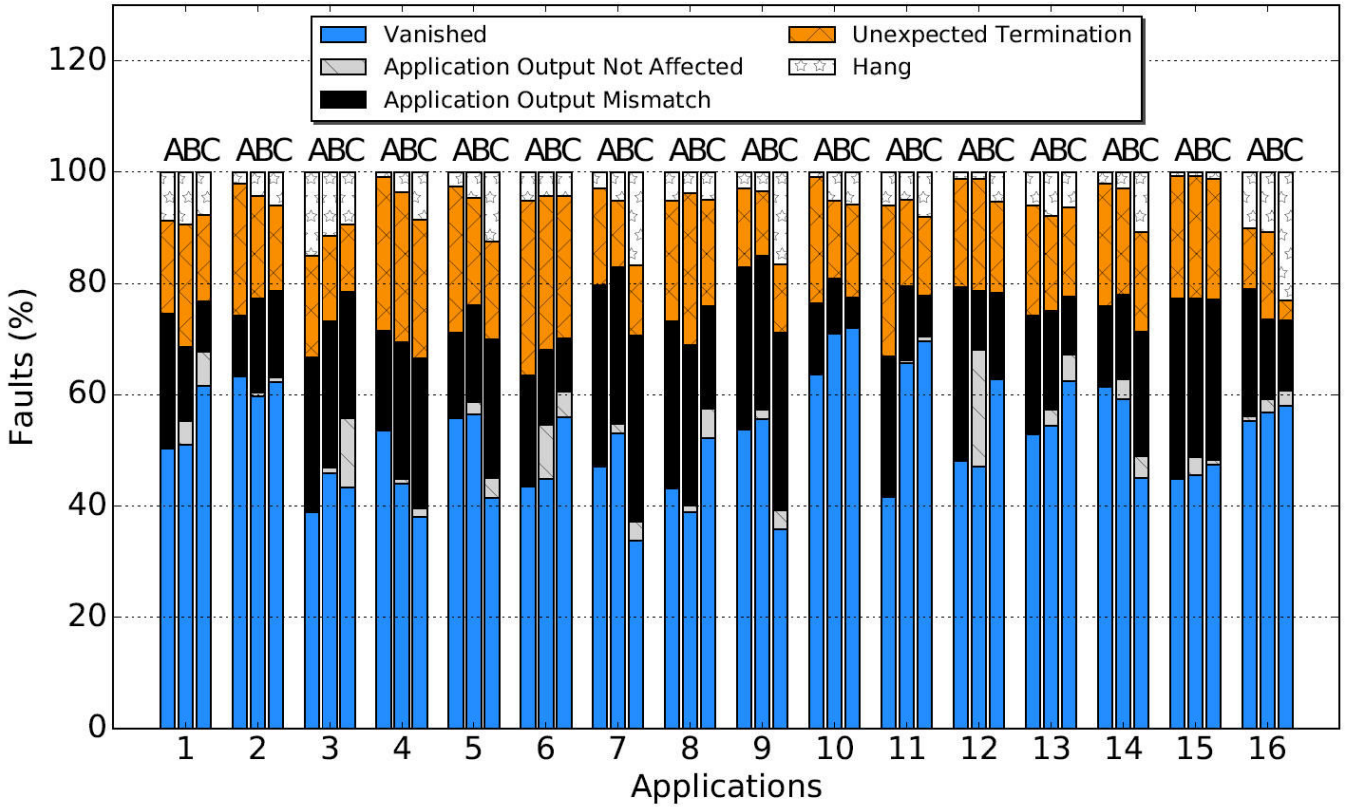
Fault Simulation Overview
4 stages

MULTICORE DESIGN SIMPLIFIED
imperas



Fault Simulation Analysis & Reporting

MULTICORE DESIGN SIMPLIFIED
imperas



- [A, B, C] = [Cortex-A9MPx1, X2, X4]
- 8,000 faults generated for each processor configuration
- 16 applications (Rhodinia HPC benchmarks) running under Linux 3.13

Summary

MULTICORE DESIGN SIMPLIFIED
imperas

- Virtual platforms – software simulation – provide a complementary technology to hardware-based testing of software
- Virtual platforms can run full production binaries including hypervisors and hardware virtualization to assist in verification of safety critical systems
- Besides functional correctness, timing properties and power consumption are gaining importance
- Virtual platforms become essential with adoption of Continuous Integration methods for embedded systems software

- Virtual platforms for software verification help you
  - Achieve higher quality software
  - Reduce development schedules
  - Increase software project predictability
  - Reduce delivery risk

MULTICORE DESIGN SIMPLIFIED

imperas

Thank you

For more information: info@imperas.com
www.imperas.com
www.OVPworld.org