MULTICORE DESIGN SIMPLIFIED

imperas

Simulation Based Analysis and Debug of Heterogeneous Platforms

Design Automation Conference, Session 60
4 June 2014

Simon Davidmann, Imperas

Agenda

- Programming on heterogeneous platforms
- Hardware-based software development, debug and test
- Software simulation – using virtual platforms
- Case study 1: Interprocessor communications with Freescale Vybrid-Kinetis platform
- Case study 2: OS porting, bring up and verification on Altera Cyclone V SoC FPGA
- Summary

Agenda

- Programming on heterogeneous platforms
- Hardware-based software development, debug and test
- Software simulation – using virtual platforms
- Case study 1: Interprocessor communications with Freescale Vybrid-Kinetis platform
- Case study 2: OS porting, bring up and verification on Altera Cyclone V SoC FPGA
- Summary

Programming Heterogeneous Platforms

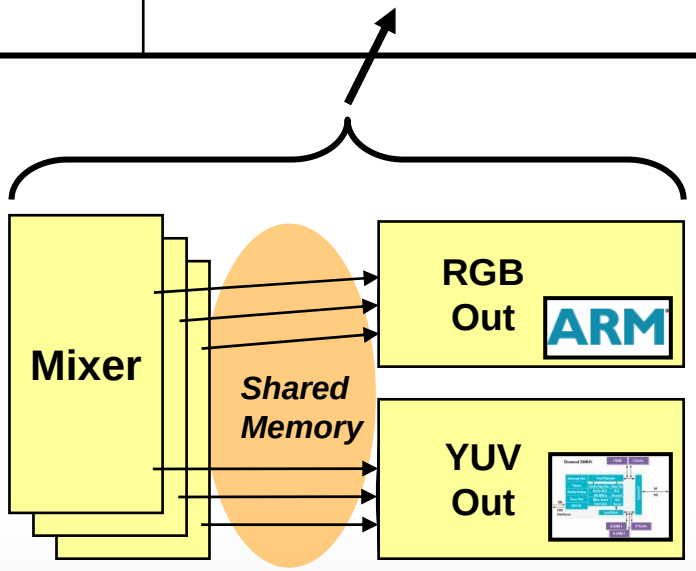
	Symmetric MultiProcessing (SMP) Architecture	Asymmetric MultiProcessing (AMP) Architecture

Programming Heterogeneous Platforms

	Symmetric MultiProcessing (SMP) Architecture	Asymmetric MultiProcessing (AMP) Architecture
Homogeneous Processor Resources		
Heterogeneous Processor Resources		

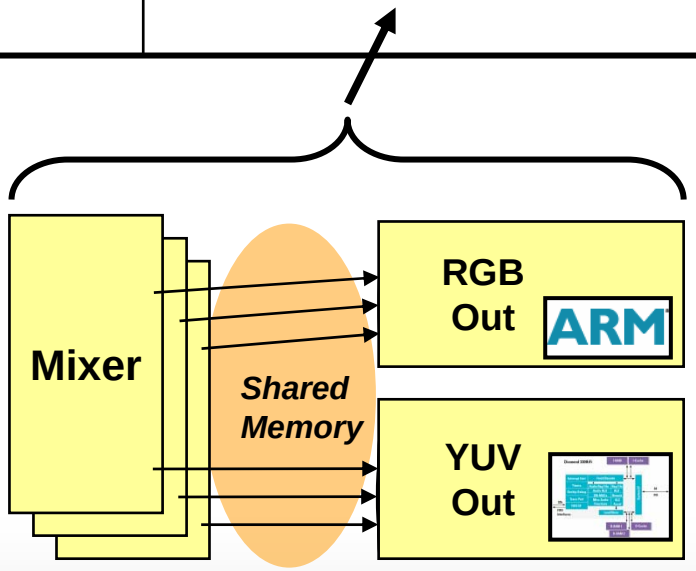
Programming Heterogeneous Platforms

	Symmetric MultiProcessing (SMP) Architecture	Asymmetric MultiProcessing (AMP) Architecture
Homogeneous Processor Resources		Heterogeneous system
Heterogeneous Processor Resources		Heterogeneous system



Programming Heterogeneous Platforms

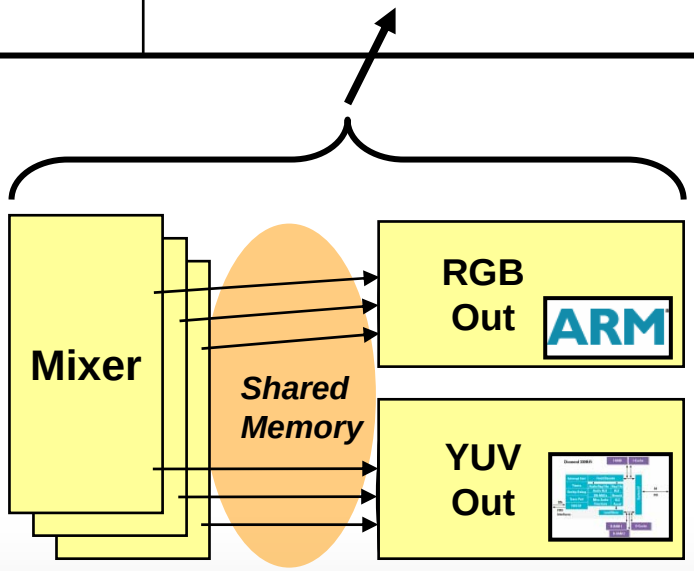
	Symmetric MultiProcessing (SMP) Architecture	Asymmetric MultiProcessing (AMP) Architecture
Homogeneous Processor Resources	Not heterogeneous, but often a subsystem of a heterogeneous system	Heterogeneous system
Heterogeneous Processor Resources	N/A (big.LITTLE?)	Heterogeneous system



Programming Heterogeneous Platforms

	Symmetric MultiProcessing (SMP) Architecture	Asymmetric MultiProcessing (AMP) Architecture
Homogeneous Processor Resources	Not heterogeneous, but often a subsystem of a heterogeneous system	Heterogeneous system
Heterogeneous Processor Resources	N/A (big.LITTLE?)	Heterogeneous system

- Programming languages, tools and paradigms are broadly discussed
  - OpenCL
  - HSA/HSAIL
  - C/C++ …
  - Brute force!



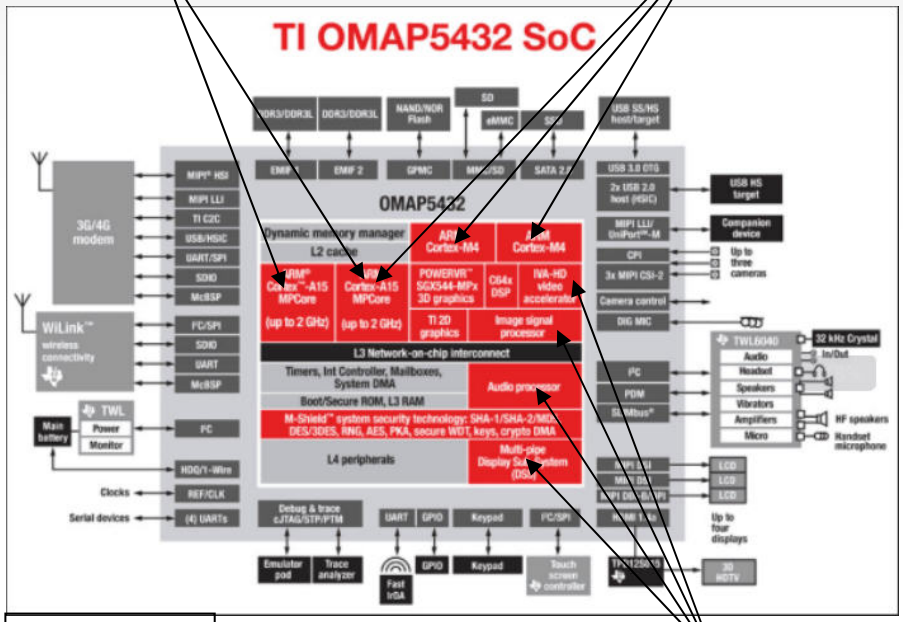
Example Heterogeneous Systems

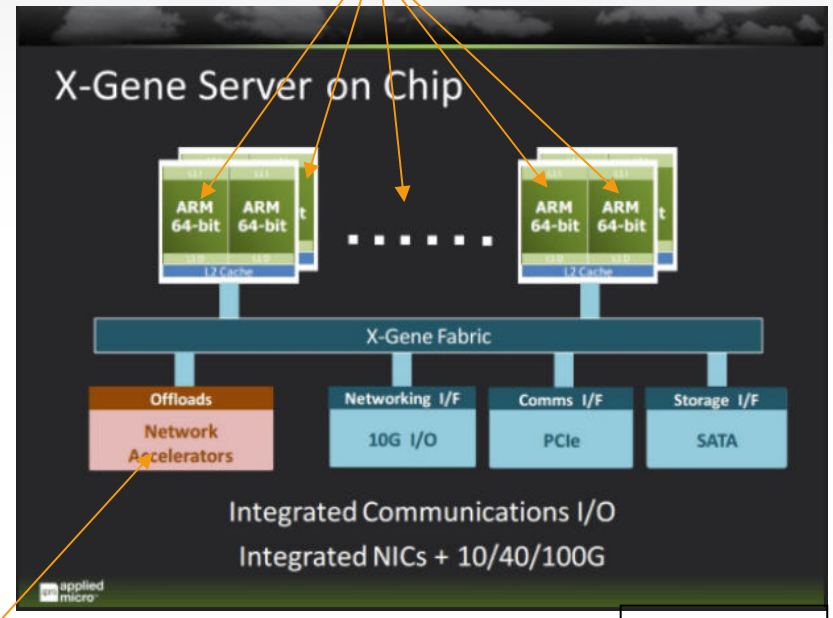
MULTICORE DESIGN SIMPLIFIED imperas

SMP cores

AMP cores

SMP cores





Mobile

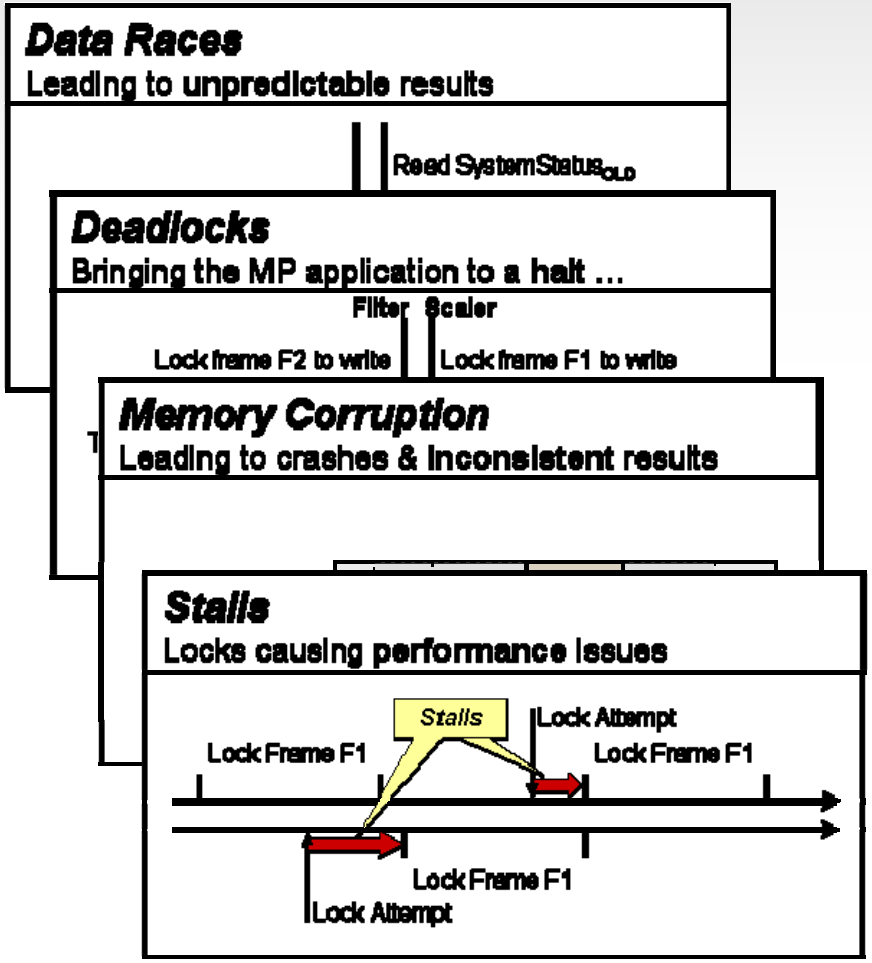
Accelerators

Server

How to Analyze and Debug Heterogeneous Platforms?

MULTICORE DESIGN SIMPLIFIED
imperas

- Heterogeneous systems have unique issues
  - Multiple processors
  - Different multiple processors
  - Multiple ISAs
  - Multiple operating systems
  - Hypervisors
  - Interprocessor communications (coherent and non-coherent)
  - Memory issues
  - Race/deadlock/stall conditions
  - …



…

Agenda

- Programming on heterogeneous platforms
- Limitations of hardware-based software development, debug and test
- Software simulation (virtual platform) advantages for operating system porting, bring up and verification
- Case study 1: Interprocessor communications with Freescale Vybrid-Kinetis platform
- Case study 2: OS porting, bring up and verification on Altera Cyclone V SoC FPGA
- Summary

Limitations of Hardware-Based Software Development

MULTICORE DESIGN SIMPLIFIED
imperas

- Traditional breadboard / emulation based testing
  - Limited physical system availability
  - Limited external test access (controllability)
  - Limited internal observability
  - Typically 6 months or more until available to team
- To get around these limitations, software is modified
  - printf added to code
  - Debug versions of OS kernels
  - Code is instrumented for specific analytical tools, e.g. code coverage, profiling
- Modified software may not have the same behavior as clean source code

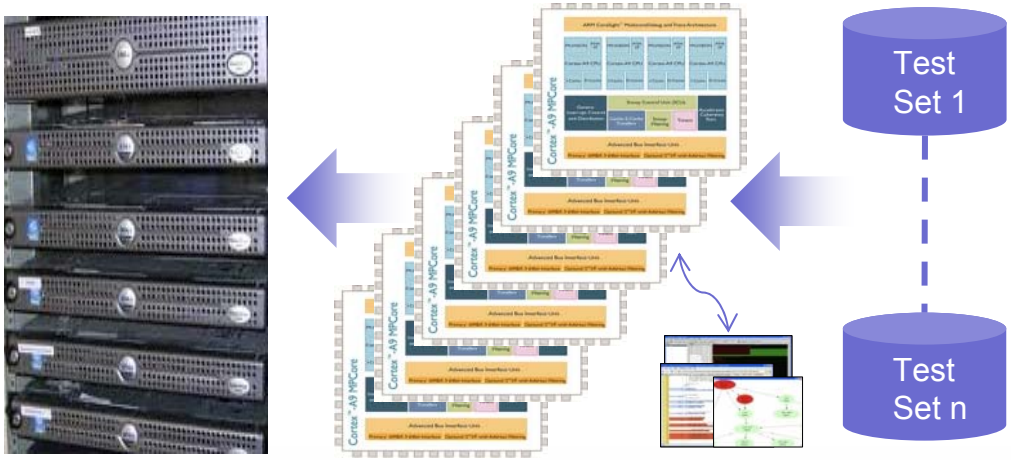
Agenda

- Programming on heterogeneous platforms
- Limitations of hardware-based software development, debug and test
- Software simulation (virtual platform) advantages for operating system porting, bring up and verification
- Case study 1: Interprocessor communications with Freescale Vybrid-Kinetis platform
- Case study 2: OS porting, bring up and verification on Altera Cyclone V SoC FPGA
- Summary

Advantages of Virtual Platform Based Software Development

MULTICORE DESIGN SIMPLIFIED
imperas

- Earlier system availability
- Full controllability of platform both from external ports and internal nodes
- Full visibility into platform
- Performance can be faster than real time
- Easy to replicate platform and test environment to support regression testing on compute farms

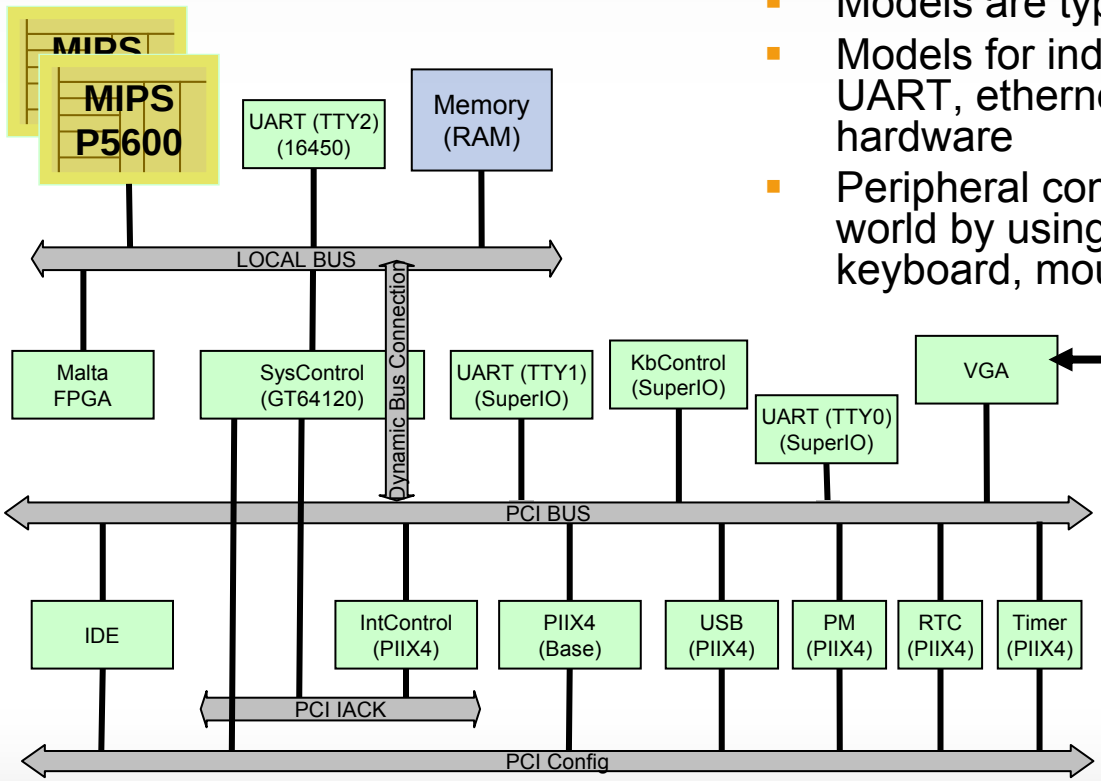


Virtual Platforms (Software Simulation)

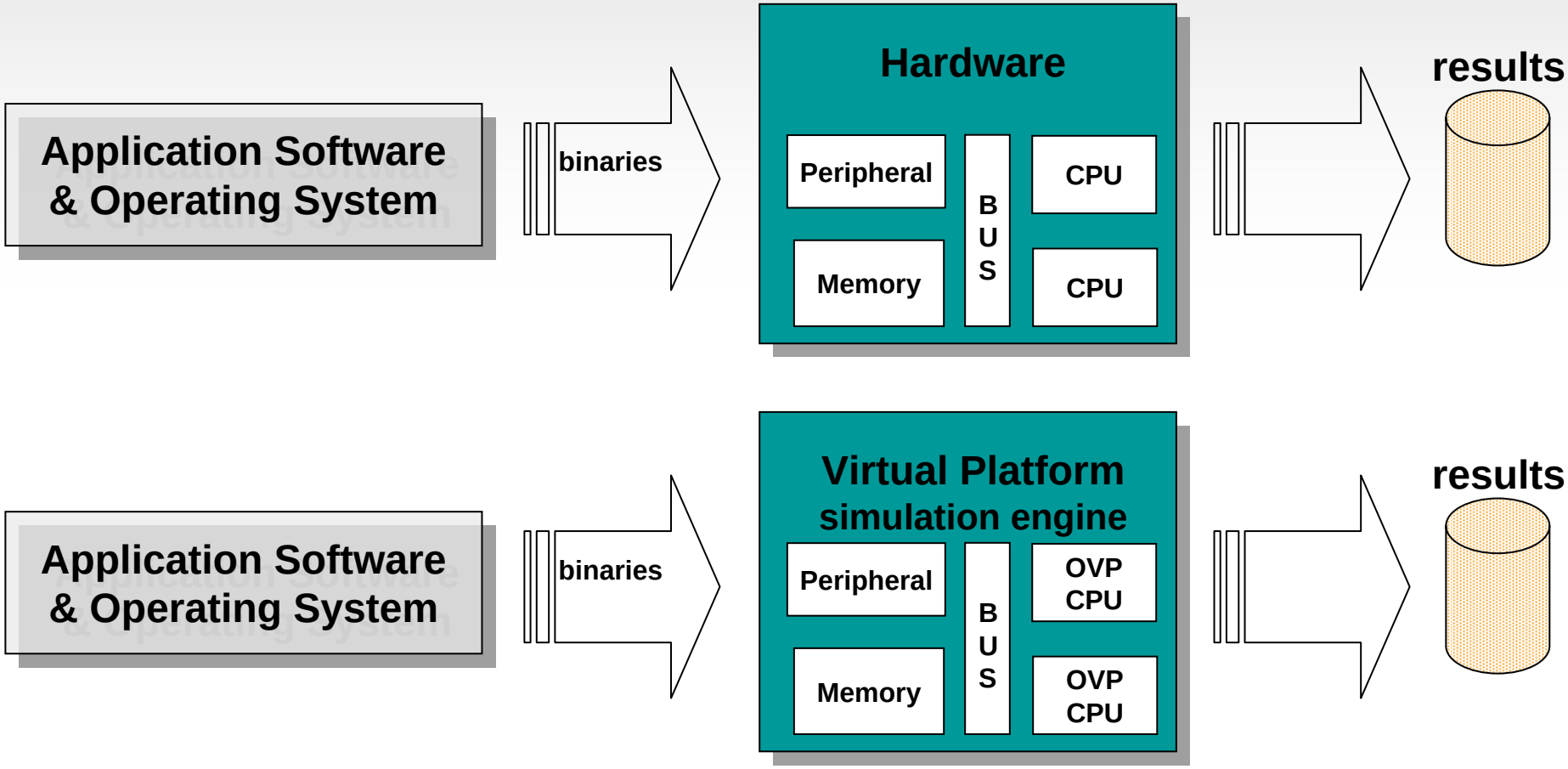
MULTICORE DESIGN SIMPLIFIED
Imperas

Software is executed unchanged, such that the software does not know that it is not executing on the hardware

- The virtual platform is a set of models that reflects the hardware on which the software will execute
  - Subset / subsystem of a single device
  - Processor chip
  - Board
  - System
- 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, …

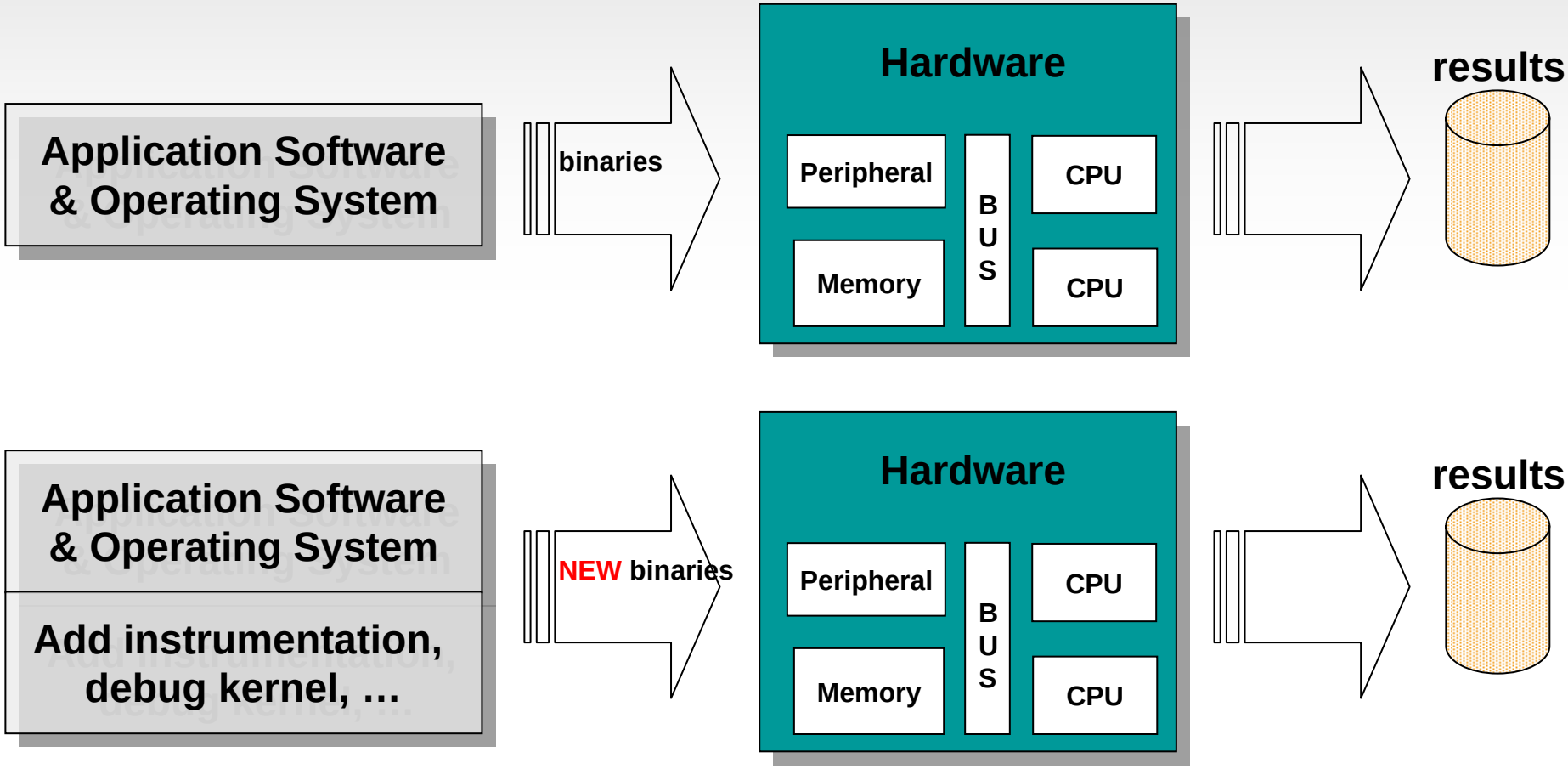


Virtual Platforms Simulate the Software Running on the Hardware



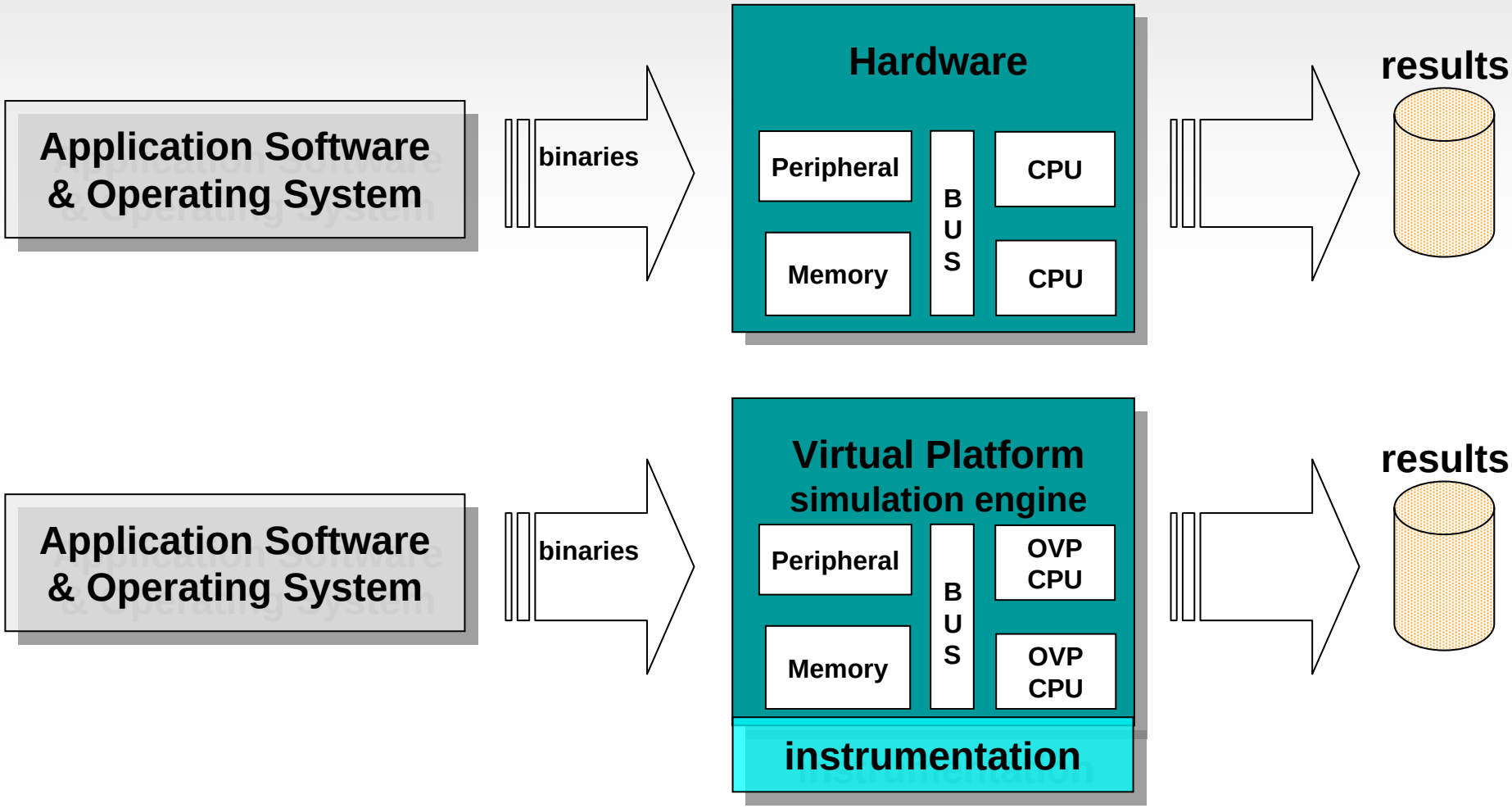
results(VP) = results(HW)

Software Analysis Tools on HW Platforms Have Validity Questions



results(HW+ instrumentation) ? = results(HW)

Software Analysis Tools Using Binary Interception Techniques are Non-Intrusive



results(VP + instrumentation) = results(HW)

And Virtual Platform simulators can be very fast

MULTICORE DESIGN SIMPLIFIED
imperas

	Altera Nios II			ARM32			Imagination MIPS32		
Benchmark	Simulated Instructions	Run time	Simulated MIPS	Simulated Instructions	Run time	Simulated MIPS	Simulated Instructions	Run time	Simulated MIPS
linpack	3,075,857,171	2.52s	1225	6,105,766,856	4.79s	1277	9,814,621,392	5.31s	1852
Dhrystone	1,810,082,387	1.18s	1547	2,250,079,359	2.32s	974	1,795,088,667	1.27s	1414
Whetstone	5,850,887,389	3.28s	1789	1,185,959,501	1.04s	1140	1,890,420,892	0.93s	2033
peakSpeed2	22,000,013,458	3.11s	7097	22,400,008,766	4.67s	4807	22,800,009,853	4s	5714
	Xilinx MicroBlaze			ARM AArch64			Imagination MIPS64		
Benchmark	Simulated Instructions	Run time	Simulated MIPS	Simulated Instructions	Run time	Simulated MIPS	Simulated Instructions	Run time	Simulated MIPS
linpack	6,386,275,159	3.77s	1699	594,945,589	1.01s	594*	1,558,856,686	0.83s	1901
Dhrystone	3,770,115,740	2.61s	1450	3,030,061,475	2.79s	1086	1,590,094,345	1.23s	1293
Whetstone	27,108,532,655	13.23s	2054	488,724,620	0.64s	759*	2,133,926,552	0.99s	2156
peakSpeed2	22,000,023,433	5.76s	3826	11,200,003,894	3.73s	3011	17,100,018,075	4.23s	4052
	PowerPC			Renesas v850			Synopsys ARC		
Benchmark	Simulated Instructions	Run time	Simulated MIPS	Simulated Instructions	Run time	Simulated MIPS	Simulated Instructions	Run time	Simulated MIPS
linpack	3,163,966,113	2.95s	1076	4,991,344,159	4.76s	1051	4,184,162,664	3.67s	1143
Dhrystone	2,205,068,239	1.75s	1260	6,410,133,101	4.01s	1603	3,155,082,476	2.75s	1148
Whetstone	6,424,865,755	3.97s	1622	10,296,940,591	7.41s	1393	7,883,567,047	4.4s	1796
peakSpeed2	22,400,002,937	5.6s	4007	22,400,007,569	3.53s	6364	22,000,002,100	4.05s	5446

All measurements on 3.40GHz Intel i7-3770, Linux, OVPsim 20140127.0

* Hardware Floating Point Instructions

- Example speed of Imperas simulation models

And booting OS can be fast too

```
C:\Windows\system32\cmd.exe
Info   Final program counter : 0x8001d668
Info   Simulated instructions: 1,131,123,567
Info   Simulated MIPS        : 20.0
Info ----------------------------------------------------
Info
Info ----------------------------------------------------
Info CPU 'ArmVersatileExpress-CA15/cpu_CPU2' STATISTICS
Info   Type                  : arm (Cortex-A15MPx4)
Info   Nominal MIPS          : 1000
Info   Final program counter : 0x8001d668
Info   Simulated instructions: 17,224,484,756
Info   Simulated MIPS        : 304.2
Info ----------------------------------------------------
Info
Info ----------------------------------------------------
Info CPU 'ArmVersatileExpress-CA15/cpu_CPU3' STATISTICS
Info   Type                  : arm (Cortex-A15MPx4)
Info   Nominal MIPS          : 1000
Info   Final program counter : 0x8001d668
Info   Simulated instructions: 1,110,697,906
Info   Simulated MIPS        : 19.6
Info ----------------------------------------------------
Info
Info ----------------------------------------------------
Info TOTAL
Info   Simulated instructions: 22,568,501,091
Info   Simulated MIPS        : 398.5
Info ----------------------------------------------------
Info
Info ----------------------------------------------------
Info SIMULATION TIME STATISTICS
Info   Simulated time        : 3264.16 seconds
Info   User time             : 55.61 seconds
Info   System time           : 1.01 seconds
Info   Elapsed time          : 56.89 seconds
Info   Real time ratio       : 57.37x faster
Info ----------------------------------------------------

CpuManagerMulti finished: Mon Mar 03 20:50:39 2014


CpuManagerMulti (64-Bit) v20140224.0 Open Virtual Platform simulator from www.IMPERAS.com.
Visit www.IMPERAS.com for multicore debug, verification and analysis solutions.

Press any key to continue . . .
```

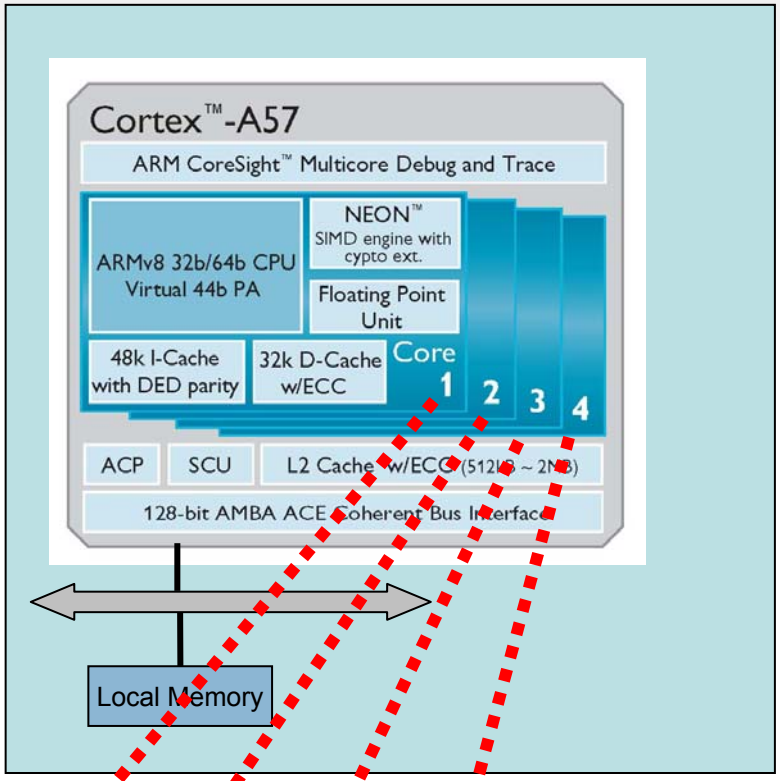
```
ArmVersatileExpress-CA15/uart0
o0/input/input0
usbcore: registered new interface driver usbhid
usbhid: USB HID core driver
oprofile: no performance counters
oprofile: using timer interrupt.
TCP: cubic registered
NET: Registered protocol family 17
VFP support v0.3: implementor 41 architecture 3 p
rtc-pl031 1c170000.rtc: setting system clock to 2
000)
ALSA device list:
  No soundcards found.
Freeing init memory: 188K
input: ImExPS/2 Generic Explorer Mouse as /device
00.kmi/serio1/input/input1

This root FS contains most basic linux utilities
and the Lynx web browser.

Kernel config is available through /proc/config.g

Welcome to OVP simulation from Imperas

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

- Boot Linux on ARM Cortex-A15x4 = 6 seconds on Win7 laptop
- Runs simulated Linux applications at 100s of MIPS

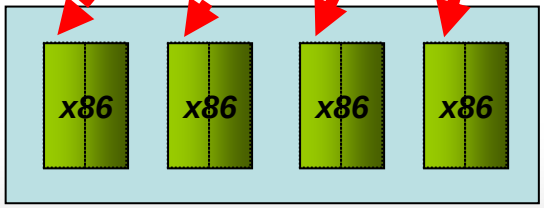
ARMv8 simulation using parallel host-cpu resources

MULTICORE DESIGN SIMPLIFIED
imperas

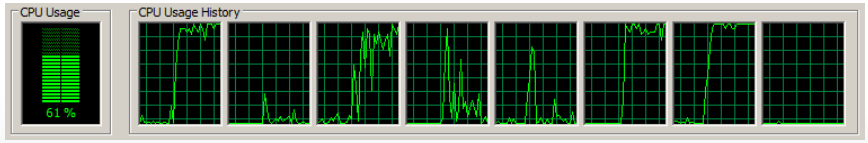
Simulation



- Advanced parallel synchronization algorithm for SMP, AMP and hardware accelerators
- Transparent operation to user: No model, tool, software changes
- Total performance on benchmarks recorded up to 16 Billion ins/sec
- Performance advantage 15x over nearest commercial alternative



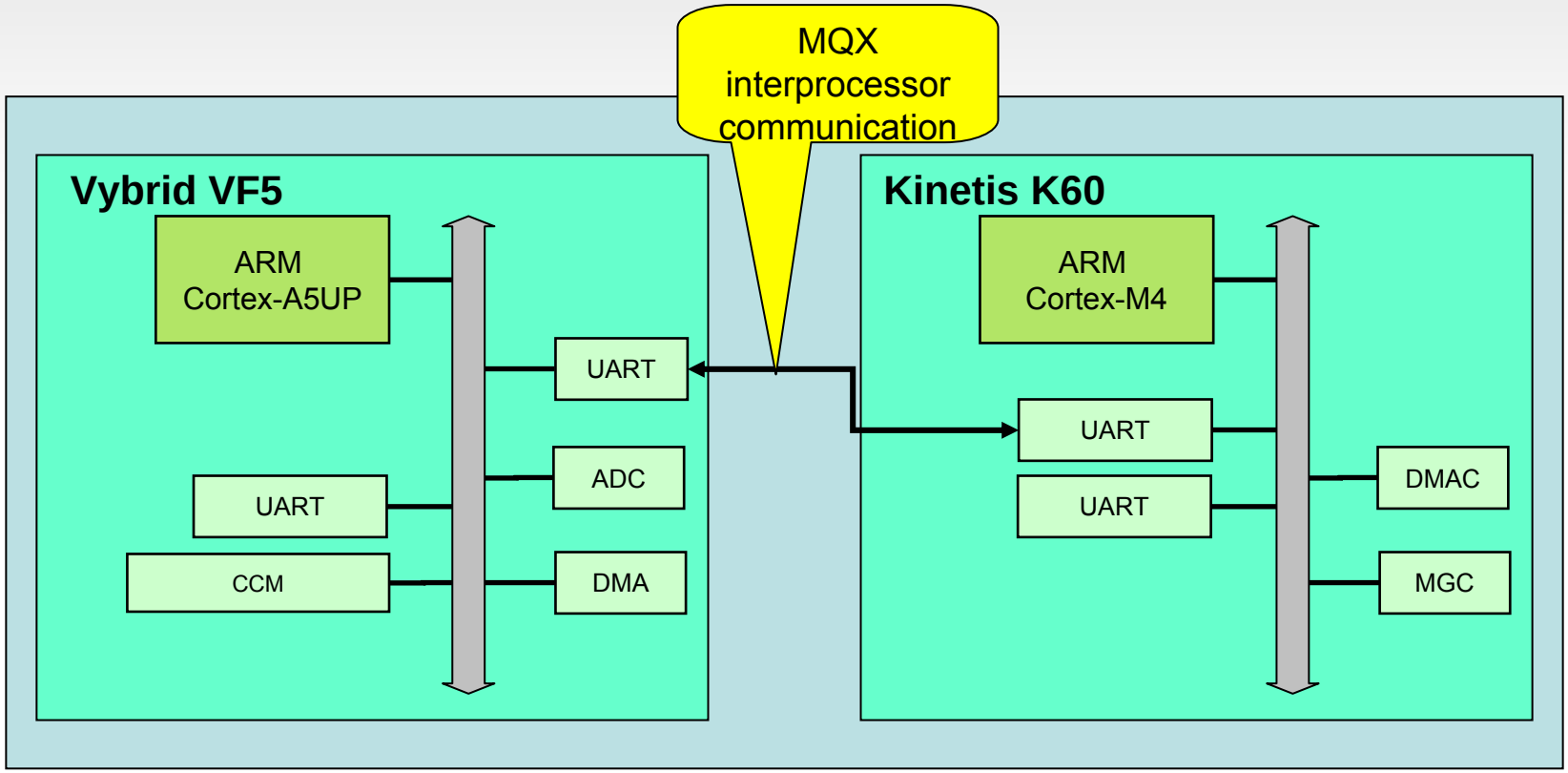
Host Processors



Agenda

- Programming on heterogeneous platforms
- Limitations of hardware-based software development, debug and test
- Software simulation (virtual platform) advantages for operating system porting, bring up and verification
- Case study 1: Interprocessor communications with Freescale Vybrid-Kinetis platform
- Case study 2: OS porting, bring up and verification on Altera Cyclone V SoC FPGA
- Summary

Case Study 1: Interprocessor Communication on Freescale Vybrid – Kinetis Virtual Platform

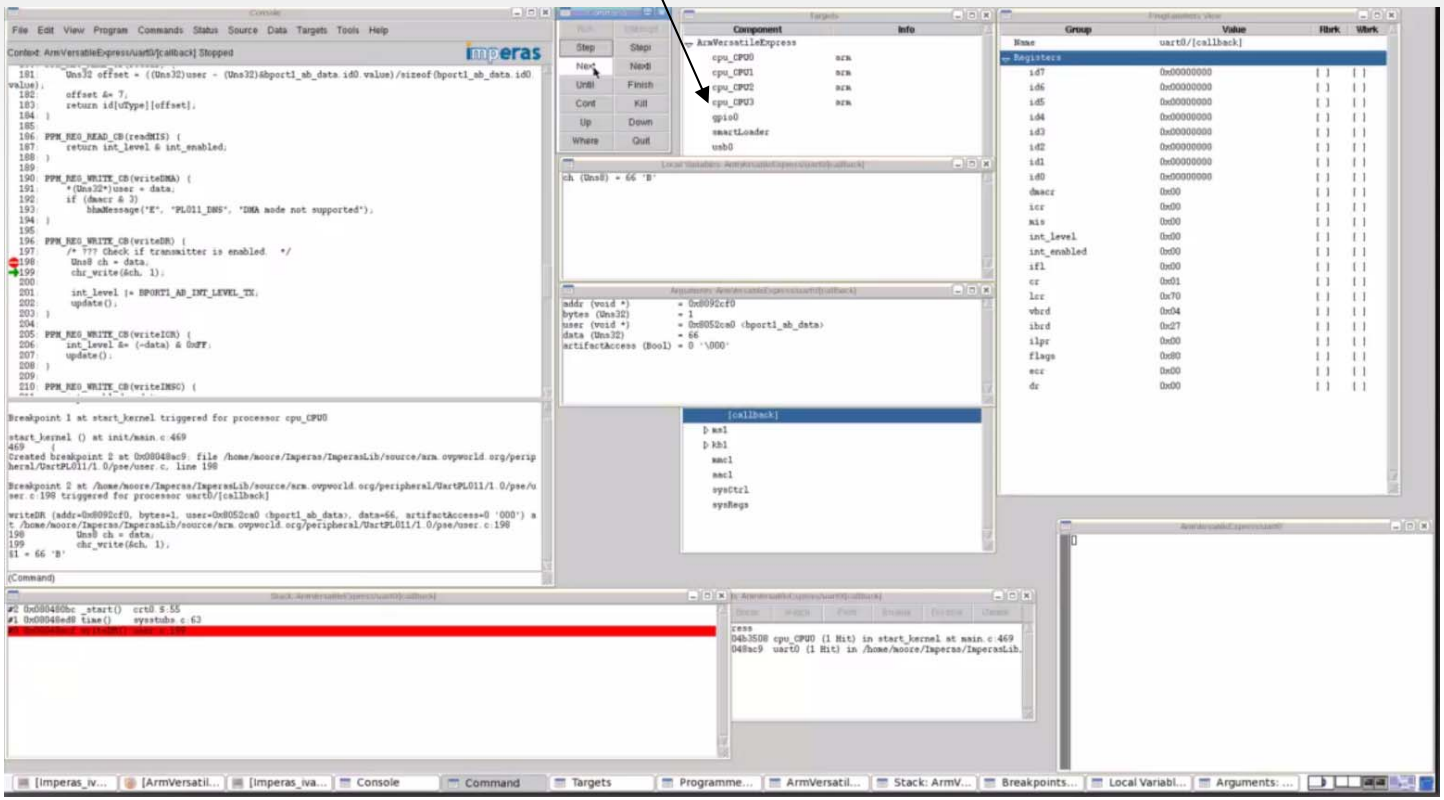


- Freescale MQX RTOS running on each device
- Uses MQX interprocessor communication capability
- Open Virtual Platforms (OVP, www.OVPworld.org) models

Multicore heterogeneous debug is needed

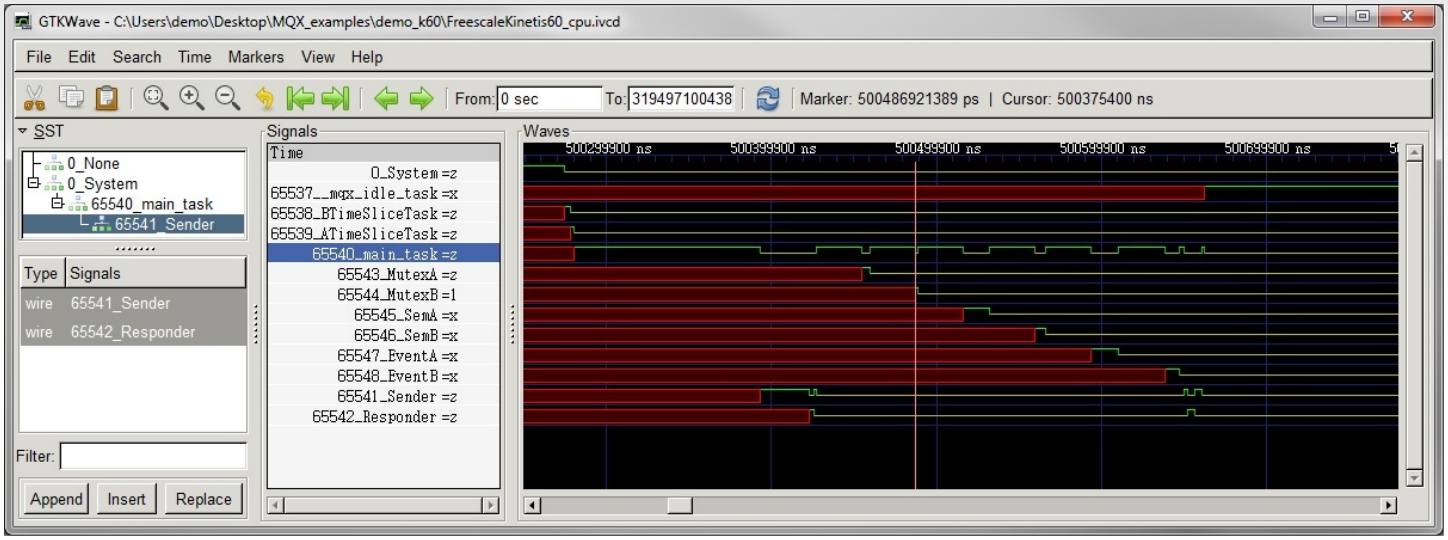
MULTICORE DESIGN SIMPLIFIED
imperas

Select target context



- Single coherent debug of cpus, peripherals, homogeneous, heterogeneous, AMP, SMP

Advanced tools needed too MQX RTOS Scheduler Analysis



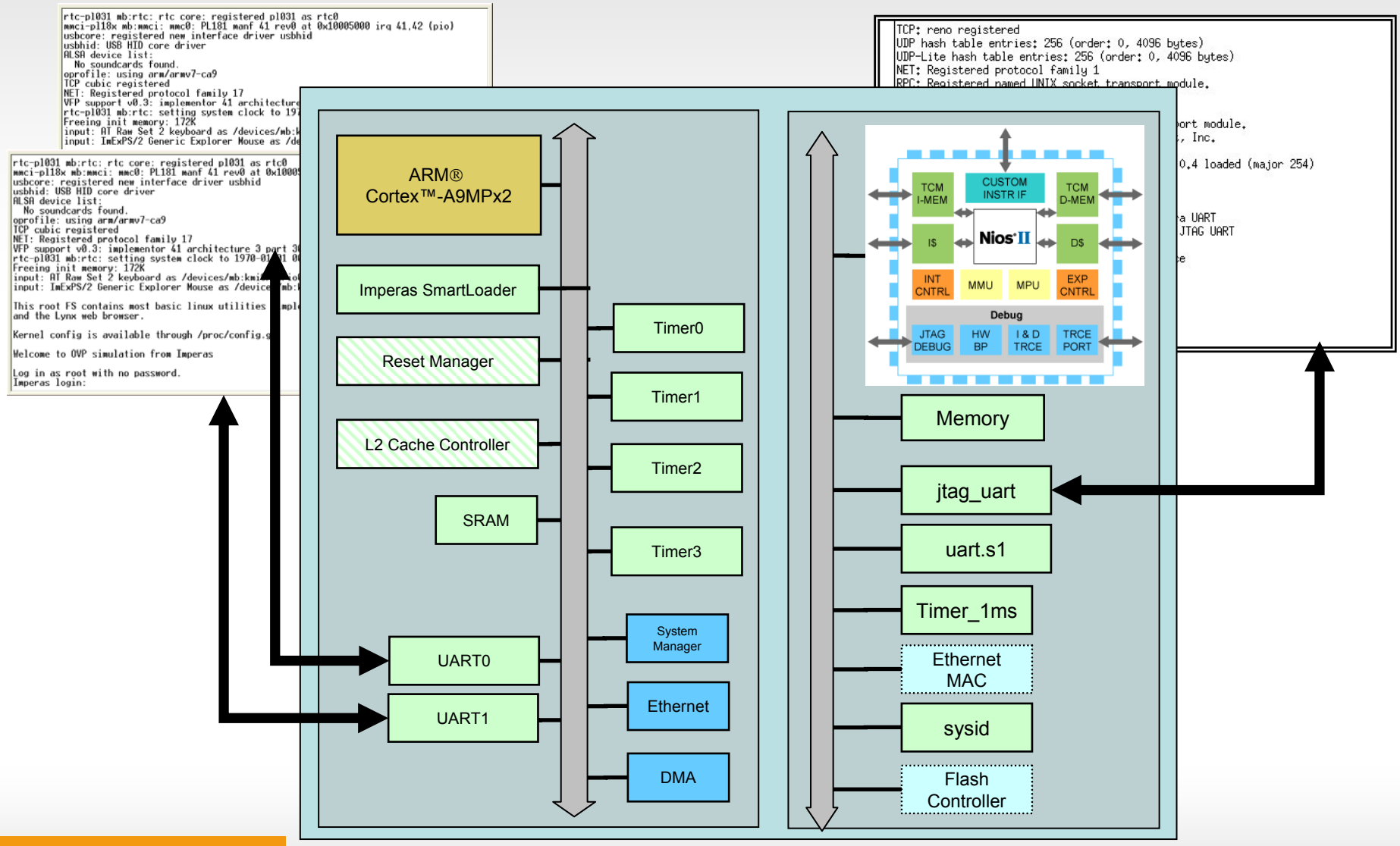
Scheduler analysis helps to identify RTOS task scheduling issues

- Imperas tools are non-intrusive
- Tools have understanding of CPUs, OSs
  - CPU-aware: code coverage, profiling, function tracing, fault injection, …
  - OS-aware: task tracing, event tracing, scheduler analysis, …
- Pre-defined and user-defined tools use same API
- Software assertions can be added to the virtual platform simulation environment

Agenda

- Programming on heterogeneous platforms
- Limitations of hardware-based software development, debug and test
- Software simulation (virtual platform) advantages for operating system porting, bring up and verification
- Case study 1: Interprocessor communications with Freescale Vybrid-Kinetis platform
- Case study 2: OS porting, bring up and verification on Altera Cyclone V SoC FPGA
- Summary

Altera Cyclone V SoC FPGA



Case Study 2: OS Porting, Bring Up and Verification on Altera Cyclone V SoC FPGA

MULTICORE DESIGN SIMPLIFIED imperas

Software use cases (challenges):

1) Linux boot on single core ARM Cortex-A9
2) SMP Linux boot on dual core ARM Cortex-A9
3) RTOS boot on single core ARM Cortex-A9
4) AMP boot on dual core ARM Cortex-A9
5) Linux boot on single core Nios II
6) SMP Linux boot on dual core ARM Cortex-A9 plus Linux boot on Nios II

Cyclone V SoC FPGA Virtual Platform

- Top level virtual platform built using Open Virtual Platforms (OVP, www.OVPworld.org) ICM API
- ARM Cortex-A9MPx2 and Altera Nios II processor core models from the OVP Library
- Peripheral models
  - Some models available in the OVP Library
  - Remaining models of peripheral components developed using OVP APIs
- OVP APIs written for C language
- Simulation engine: Imperas M*SDK

- All OVP processor and peripheral models include both native OVP and native SystemC/TLM2 interfaces, so all the following results could have been achieved using the OSCI SystemC simulator plus Imperas M*SDK product
  - Peripheral models could have been written in SystemC
  - M*SDK tools require OVP processor core models for ToolMorphing capability

1a) Linux Boot on Single Core ARM Cortex-A9

- Use Linux from Altera: Altera-3.4
- Use default configurations
- Use default device trees
  - Comment out a few peripherals not yet modeled
- Bug found in Linux kernel preemptive scheduling
  - Running multiple applications under Linux part of standard Imperas bring up testing
  - Linux boots and runs, but does not switch tasks properly
  - Not observed in previous virtual platform (different virtual platform vendor) using much slower model of ARM Cortex-A9MPx2
    - Could not run multiple applications for long enough simulation to observe the bug

- Approximately 2 man weeks effort to build virtual platform able to boot Linux
- Virtual platform boots Linux in under 5 sec on standard PC, Windows or Linux

1b) OS-Aware Tools Used to Find the Bug

- Use OS tracing [task, execve, schedule, context, …] to trace at the OS level, not instruction level
  - Higher level of abstraction makes debug easier: ~700,000,000 to boot Linux, however, only ~700 tasks
- OS-aware tools debug in hours, once the bug was observed
- Simulation overhead due to OS-aware tools < 10%

Imperas Multiprocessor Debugger

```
TRC (TASK) 810691893: 'cpu_CPU0': free_pid called for pid=412 ('/sbin/mdev')
TRC (SCHD) 810749566: 'cpu_CPU0': scheduler switched from process 405 ('/bin/sh') to 3 ('ksoftirqd/0')
TRC (SCHD) 810753989: 'cpu_CPU0': scheduler switched from process 3 ('ksoftirqd/0') to 413 ('rcS')
TRC (TASK) 810805210: 'cpu_CPU0': do_execve called for pid=413 ('/bin/hostname')
TRC (EXEC) 810805210: 'cpu_CPU0': do_execve called for pid=413 with filename=/bin/hostname with:
TRC (EXEC) 810805210: 'cpu_CPU0':        argv virt=0x000dd24c "hostname"
TRC (EXEC) 810805210: 'cpu_CPU0':        argv virt=0x000dd264 "Imperas"
TRC (EXEC) 810805210: 'cpu_CPU0':        envp virt=0x7e943fe2 "USER=root"
TRC (EXEC) 810805210: 'cpu_CPU0':        envp virt=0x7e943f9f "HOME=/"
TRC (EXEC) 810805210: 'cpu_CPU0':        envp virt=0x7e943fa6 "TERM=vt102"
TRC (EXEC) 810805210: 'cpu_CPU0':        envp virt=0x7e943fb1 "PATH=/sbin:/usr/sbin:/bin:/usr/bin"
TRC (EXEC) 810805210: 'cpu_CPU0':        envp virt=0x7e943fd4 "SHELL=/bin/sh"
TRC (EXEC) 810805210: 'cpu_CPU0':        envp virt=0x000dd778 "PWD=/"
TRC (TASK) 810822171: 'cpu_CPU0': load_elf_binary('/bin/hostname') called for pid=413
TRC (TASK) 811285592: 'cpu_CPU0': do_exit called for pid=413 ('/bin/hostname')
TRC (SCHD) 811324531: 'cpu_CPU0': scheduler switched from process 413 ('/bin/hostname') to 405 ('/bin/sh')
TRC (TASK) 811326827: 'cpu_CPU0': free_pid called for pid=413 ('/bin/hostname')
TRC (TASK) 811341466: 'cpu_CPU0': do_exit called for pid=405 ('/bin/sh')
TRC (SCHD) 811388059: 'cpu_CPU0': scheduler switched from process 405 ('/bin/sh') to 3 ('ksoftirqd/0')
TRC (SCHD) 811397630: 'cpu_CPU0': scheduler switched from process 3 ('ksoftirqd/0') to 1 ('/init')
TRC (TASK) 811403745: 'cpu_CPU0': free_pid called for pid=405 ('/bin/sh')
TRC (SCHD) 811411242: 'cpu_CPU0': scheduler switched from process 1 ('/init') to 3 ('ksoftirqd/0')
TRC (SCHD) 811413283: 'cpu_CPU0': scheduler switched from process 3 ('ksoftirqd/0') to 414 ('init')
TRC (TASK) 811451482: 'cpu_CPU0': do_execve called for pid=414 ('/sbin/getty')
TRC (EXEC) 811451482: 'cpu_CPU0': do_execve called for pid=414 with filename=/sbin/getty with:
TRC (EXEC) 811451482: 'cpu_CPU0':        argv virt=0x7e8fdb24 "/sbin/getty"
TRC (EXEC) 811451482: 'cpu_CPU0':        argv virt=0x7e8fdb30 "-L"
TRC (EXEC) 811451482: 'cpu_CPU0':        argv virt=0x7e8fdb33 "38400"
TRC (EXEC) 811451482: 'cpu_CPU0':        argv virt=0x7e8fdb39 "ttyS0"
TRC (EXEC) 811451482: 'cpu_CPU0':        envp[] = virt=0x000dd008 <not in TLB>
TRC (TASK) 811469113: 'cpu_CPU0': load_elf_binary('/sbin/getty') called for pid=414
TRC (SCHD) 811480188: 'cpu_CPU0': scheduler switched from process 414 ('/sbin/getty') to 3 ('ksoftirqd/0')
TRC (SCHD) 811482266: 'cpu_CPU0': scheduler switched from process 3 ('ksoftirqd/0') to 1 ('/init')
TRC (SCHD) 811485826: 'cpu_CPU0': scheduler switched from process 1 ('/init') to 414 ('/sbin/getty')
TRC (SCHD) 812001979: 'cpu_CPU0': scheduler switched from process 414 ('/sbin/getty') to 3 ('ksoftirqd/0')
TRC (SCHD) 812006285: 'cpu_CPU0': scheduler switched from process 3 ('ksoftirqd/0') to 0 ('swapper')
TRC (SCHD) 824001960: 'cpu_CPU0': scheduler switched from process 0 ('swapper') to 1 ('/init')
```

2) SMP Linux Boot on Dual Core ARM Cortex-A9

- Use Linux from Altera: Altera-3.6
- Use default configurations
- Use default device trees
  - Comment out a few peripherals not yet modeled
- No problems in SMP Linux bring up on virtual platform

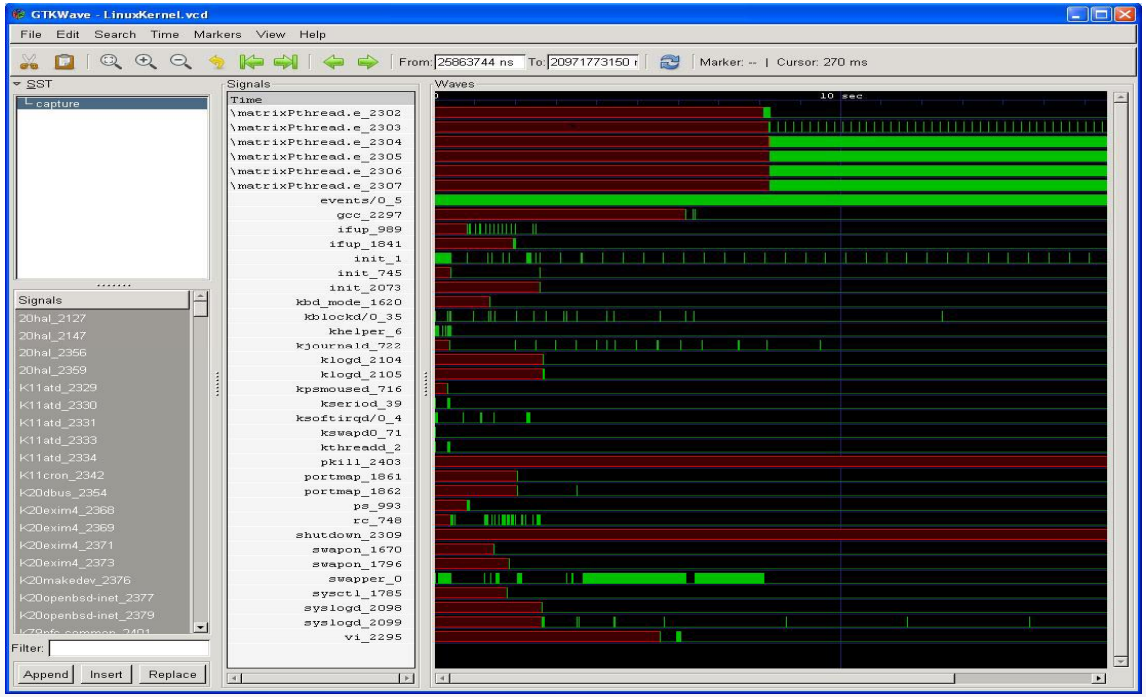
3a) Micrium µCOS-II Boot on Single Core ARM Cortex-A9

MULTICORE DESIGN SIMPLIFIED
imperas

- Use Altera µCOS-II release
- Bugs found and fixed in GIC register accesses using OS-aware tools
  - Access ICDICER 1 to 8 when only 0 to 7 exist
  - Access ICDIPTR 08 to 63 when only 00 to 55 exist
- Typically < 1 week effort to add support for new RTOS
- RTOS OS-aware tools include event scheduler viewing as waveform

3b) OS Porting and Bring Up

- Non-intrusive (no modification of OS source) trace of
  - process creation
  - context switch
  - process deletion
- Captures communications between processes
- Supported OS include Linux, FreeRTOS, Nucleus, µC/OS
  - < 1 week to support new RTOS
- View in waveform viewer



4a) AMP boot on Dual Core ARM Cortex-A9

- Linux booting on first core, µC/OS-II on second core
- Bug found in Linux accesses of GIC registers
- Virtual platform debug took 2 days versus 2 weeks on hardware platform (5x improvement)
- Also need to ensure that different operating systems do not access forbidden memory segments

4b) Custom Memory Access Monitor Accelerates AMP Platform Debug

MULTICORE DESIGN SIMPLIFIED imperas

- Memory access monitor is just C code, less than 350 lines, loaded into simulation environment
- When simulation is run, monitor produces warning if memory access rules are violated

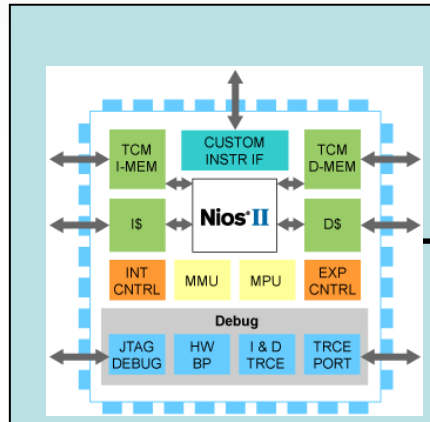
```
//
// Define watch areas for memory and peripherals defined in the platform
//
memWatchT amcWatch[] = {
//  name                          watchLow       watchHigh       allowedCPUs
    { "Linux memory",             0,             0x2fffffff,     LINUX_CPU   },
    { "uCOS memory",              0x30000000,    0x31ffffff,     UCOSII_CPU  },
    { "gmac0",                    0xff700000,    0xff700fff,     LINUX_CPU   },
    { "emac0_dma",                0xff701000,    0xff701fff,     LINUX_CPU   },
    { "gmac1",                    0xff702000,    0xff702fff,     LINUX_CPU   },
    { "emac1_dma",                0xff703000,    0xff703fff,     LINUX_CPU   },
    { "uart0",                    0xffc02000,    0xffc02fff,     LINUX_CPU   },
    { "uart1",                    0xffc03000,    0xffc03fff,     UCOSII_CPU  },
    { "CLKMGR",                   0xffd04000,    0xffd04fff,     LINUX_CPU   },
    { "RSTMGR",                   0xffd05000,    0xffd05fff,     LINUX_CPU   },
    { "SYSMGR",                   0xffd08000,    0xffd08fff,     LINUX_CPU   },
    { "GIC",                      0xfffec000,    0xfffedfff,     LINUX_CPU   },
    { "L2",                       0xfffef000,    0xfffeffff,     LINUX_CPU   },
    { 0 } /* Marks end of list */
};
```

```
Warning (AMPCHK_MWV) cpu_CPU0: AMP write access violation in uart1 area. PA: 0xffc03008 VA: 0xffc03008
Warning (AMPCHK_MWV) cpu_CPU0: AMP write access violation in uart1 area. PA: 0xffc0300c VA: 0xffc0300c
Warning (AMPCHK_MWV) cpu_CPU0: AMP write access violation in uart1 area. PA: 0xffc03010 VA: 0xffc03010
Warning (AMPCHK_MRV) cpu_CPU1: AMP read access violation in Linux memory area. PA: 0x00000020 VA: 0x00000020
```

5) Linux on Single Core Altera Nios II

- Altera Cyclone III 3c120
- Linux booting on Nios II processor core model
- No issues with Linux boot

```
TCP: reno registered
UDP hash table entries: 256 (order: 0, 4096 bytes)
UDP-Lite hash table entries: 256 (order: 0, 4096 bytes)
NET: Registered protocol family 1
RPC: Registered named UNIX socket transport module.
RPC: Registered udp transport module.
RPC: Registered tcp transport module.
RPC: Registered tcp NFSv4.1 backchannel transport module.
jffs2: version 2.2. (NAND) © 2001-2006 Red Hat, Inc.
msgmni has been set to 238
Block layer SCSI generic (bsg) driver version 0.4 loaded (major 254)
io scheduler noop registered
io scheduler deadline registered
io scheduler cfq registered (default)
ttyAL0 at MMIO 0x8004c80 (irq = 10) is a Altera UART
ttyJ0 at MMIO 0x8004d50 (irq = 1) is a Altera JTAG UART
loop: module loaded
mousedev: PS/2 mouse device common for all mice
TCP: cubic registered
NET: Registered protocol family 17
turn off boot console early0

Welcome to Nios II
nios2 login:
```



Memory

jtag_uart

uart.s1

Timer_1ms

Ethernet MAC

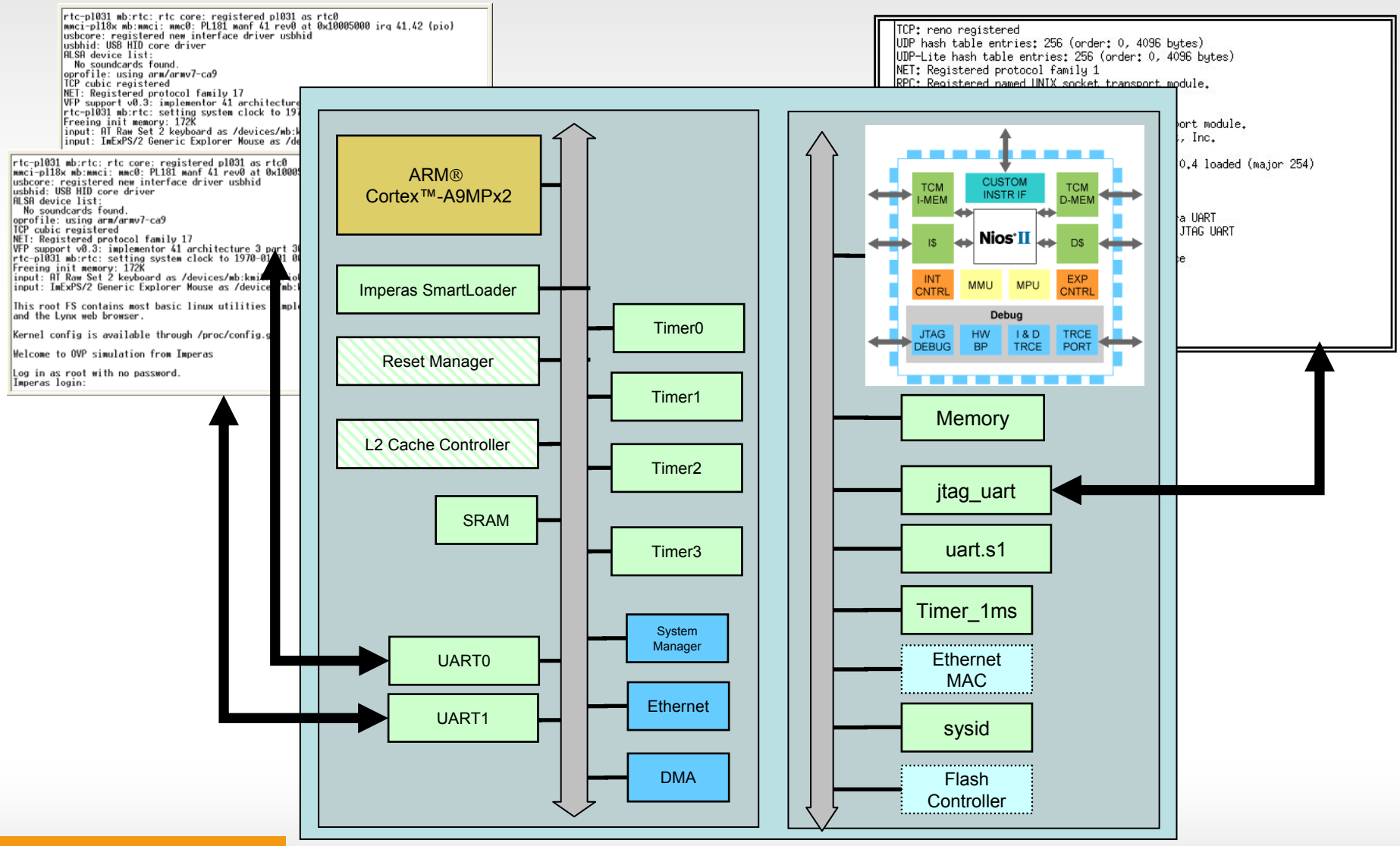
sysid

Flash Controller

Keyboard

6) Heterogeneous AMP Platform

Altera Cyclone V Cortex A9MPx2 (SMP Linux) and Nios II (Linux)



Summary

- Heterogeneous platforms require not only new programming tools but also new debug and analysis tools
- Hardware-based software testing has limitations in controllability and observability
- Instruction accurate software simulation – virtual platforms – has the required controllability and observability
- Use of virtual platform based tools can provide both higher quality and reduced schedules for heterogeneous systems
- Results were shown for heterogeneous ARM-ARM on Freescale Vybrid and Kinetis, and for ARM-Nios II AMP system on Altera Cyclone V SoC FPGA