ARM© big.little™ processing Task migration and integration into Linux Power Management

CE Linux – Japan Technical Jamboree

3/23/2012

Sylvain Bayon de Noyer

Synopsys

Agenda

- big.LITTLE processing introduction
  - In a nutshell
  - Background, motivation & technology
- Multi-core task migration software layer
  - Integration into Linux DVFS framework
  - Results
- Outlook

Agenda

big.LITTLE processing introduction

- In a nutshell
- Background, motivation & technology

Multi-core task migration software layer

- Integration into Linux DVFS framework
- Results

Outlook

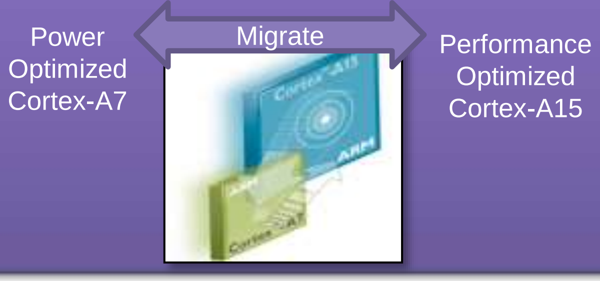
big.LITTLE processing in a nutshell – high level

Motivation

Optimal performance & battery mileage

Technology

ARM’s new big.LITTLE processing



Performance on demand

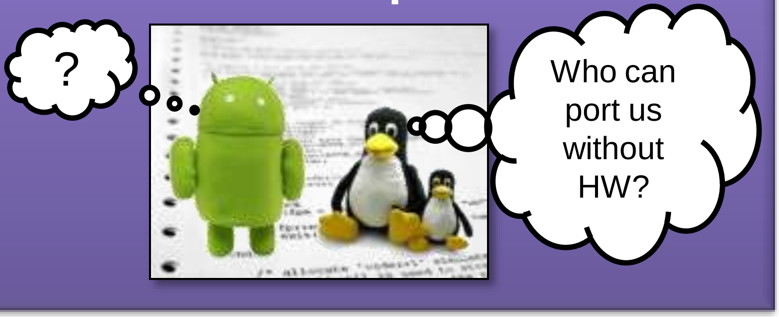
Power Optimized Cortex-A7

Migrate

Performance Optimized Cortex-A15

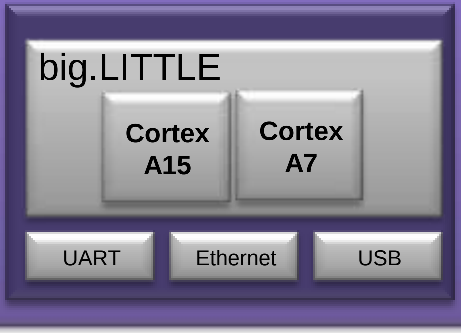
Challenges

No hardware! How to start SW development?



Tools

Virtualizer Development Kit for big.LITTLE



big.LITTLE

Cortex A15

Cortex A7

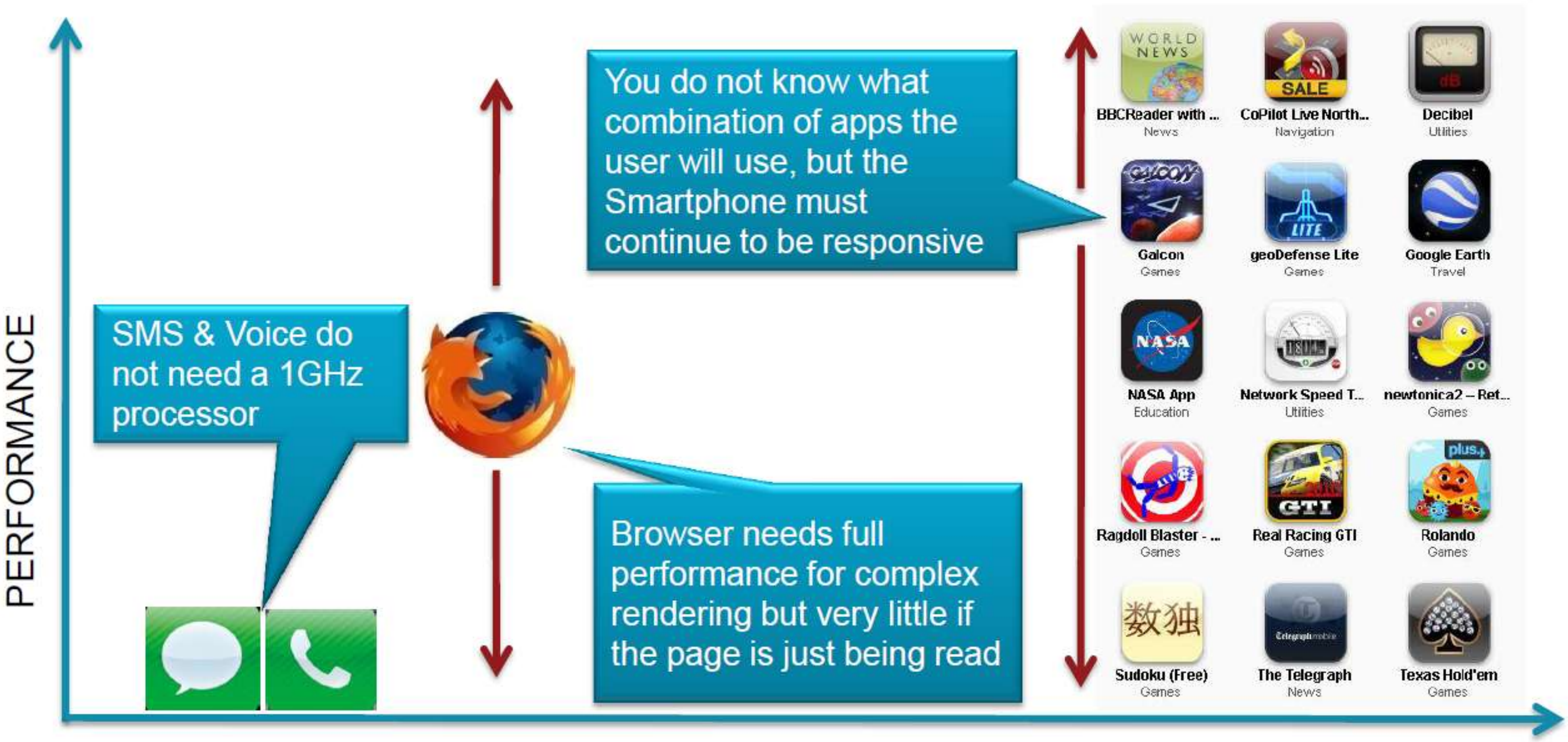
UART

Ethernet

USB

Modern Mobile System

Trade off: Performance – User Experience – Battery life



SMS & Voice do not need a 1GHz processor

You do not know what combination of apps the user will use, but the Smartphone must continue to be responsive

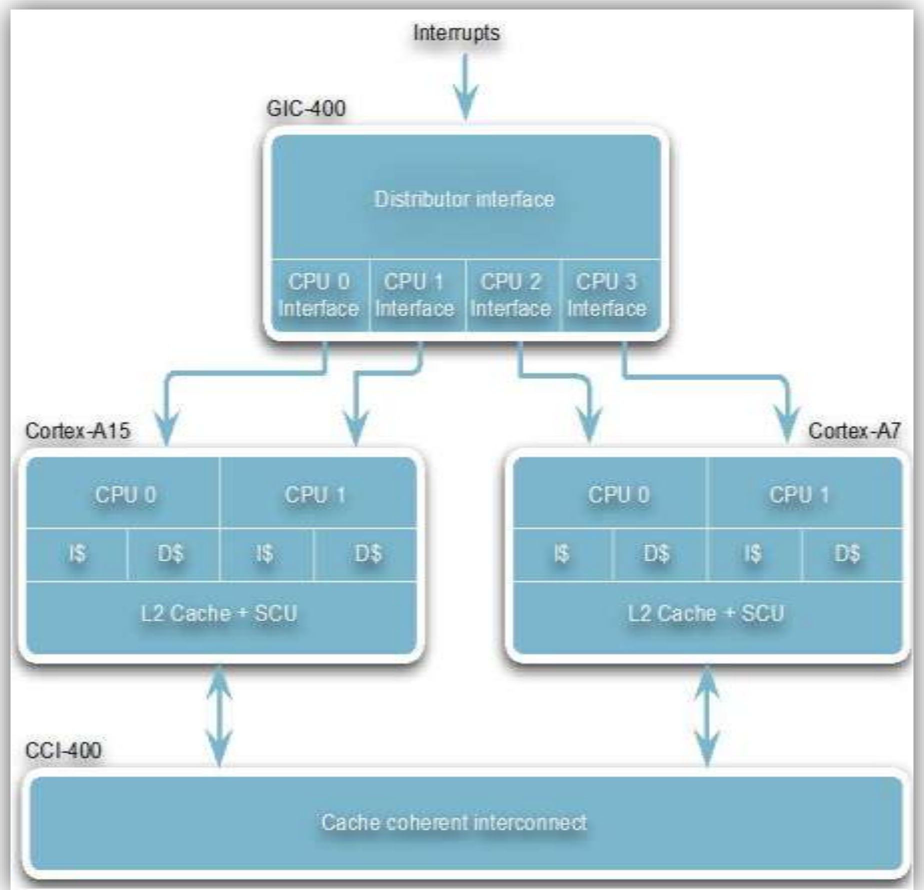
Browser needs full performance for complex rendering but very little if the page is just being read

Source: Hardware accelerated Virtualization in the ARM Cortex™ Processors, John Goodacre Director, Program Management ARM Processor Division

http://xen.org/files/xensummit_seoul11/nov2/2_XSAsia11_JGoodacre_HW_accelerated_virtualization_in_the_ARM_Cortex_processors.pdf

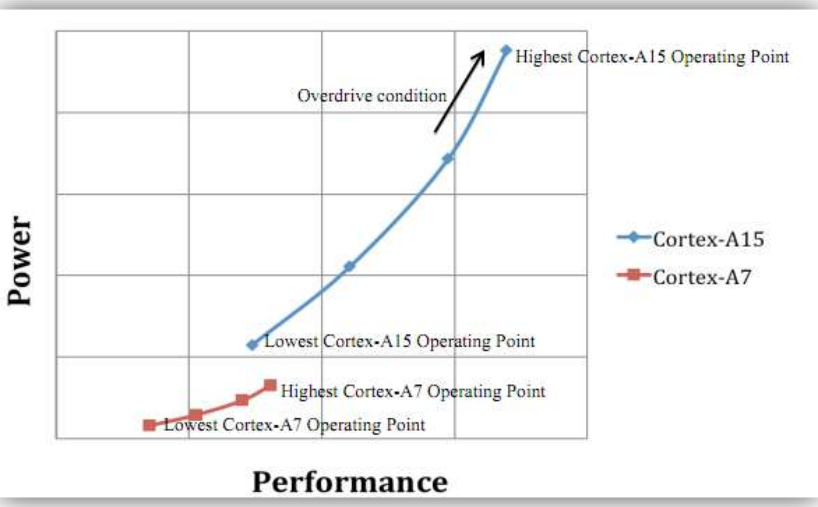
big.LITTLE Concept

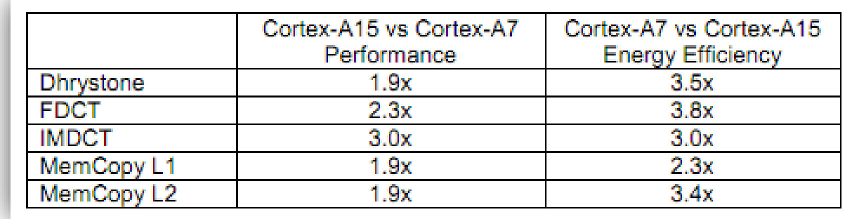
- High performance Cortex™-A15 cluster
- Energy efficient CortexTM-A7 cluster
- CCI-400 provides cache coherency between clusters
- Shared GIC-400 interrupt controller



Why big.LITTLE?

- Cortex-A15 and Cortex-A7 are ISA Identical
- Cortex-A15 efficiently achieves high performance
- Cortex-A7 is highly energy efficient
  - Offers performance approaching the latest high-end smartphones





	Cortex-A15 vs Cortex-A7 Performance	Cortex-A7 vs Cortex-A15 Energy Efficiency
Dhrystone	1.9x	3.5x
FDCT	2.3x	3.8x
IMDCT	3.0x	3.0x
MemCopy L1	1.9x	2.3x
MemCopy L2	1.9x	3.4x

Software Execution Models

- Multi-processing
  - Load Balancing on an asymmetric architecture
- Task migration
  - Only one cluster is active at a time.
    - The OS continues to load balance on an MP cluster as always
  - When the cluster cannot service the load requirement, migrate across clusters
  - When the cluster is underutilized, migrate across clusters

Open material: … Press Releases

- Press: Samsung confirmed to use ARM's big.LITTLE chip architecture for frugal Exynos in 2012 12/20/2011 5:17 AM
- TI Blog: "Best core for the chore" evolves to maximize mobile user experience 12/20/2011 5:24 AM
- Press: ARM Unveils its Most Energy Efficient Application Processor Ever; Redefines Traditional Power And Performance Relationship With big.LITTLE Processing 12/20/2011 5:27 AM
- Linaro Blog: big.LITTLE Technology – Two Usage Models 12/20/2011 5:44 AM
- ARM Blog: big.LITTLE and AMBA 4 ACE keep your cache warm and avoid flushes 12/20/2011 5:47 AM
- ARM Blog: Combining large and small compute engines - ARM Cortex-A7 12/20/2011 5:51 AM
- Interview: ARM CTO Mike Muller on big.LITTLE and power 12/21/2011 2:16 AM
- ARM Presentation: Hardware accelerated Virtualization in the ARM Cortex™ Processors 1/9/2012 6:53 AM
- ARM video: big.LITTLE Processing from ARM - CES 2012 1/31/2012 1:36 AM

- Where multiple names are used:
  - Hypervisor
  - ARM Virtualizer or Virtualisor
  - **Switcher**

Open material: … an example

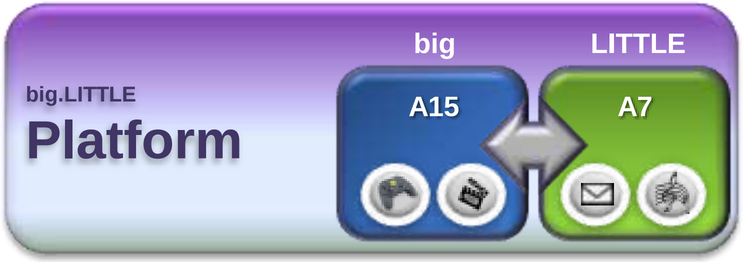
- Example task migration software freely available from Linaro
  - ~3400 lines of code including comments
  - http://git.linaro.org/gitweb?p=people/dmart/arm-virt-bl.git
  - Statically performs a task migration every 12M cycles

Agenda

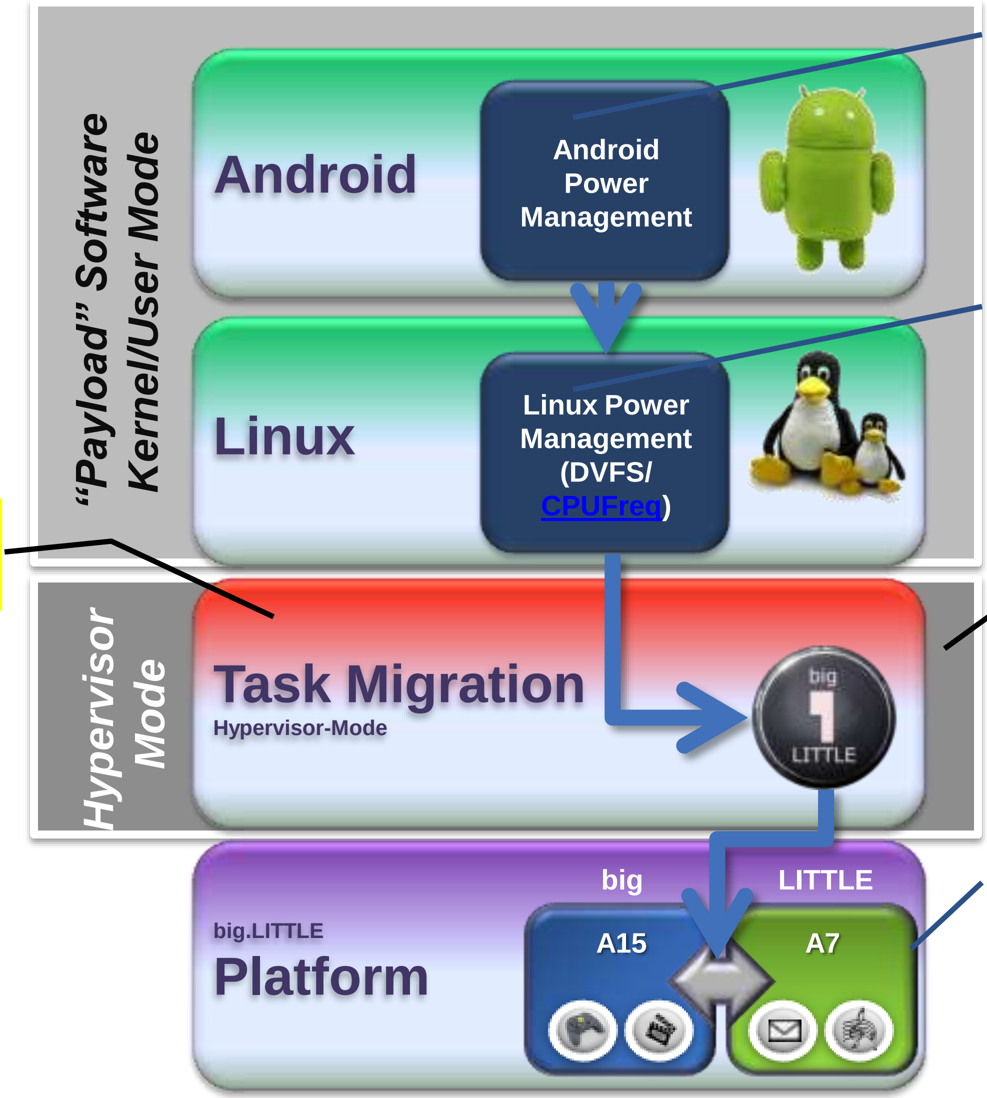
- big.LITTLE processing introduction
  - In a nutshell
  - Background, motivation & technology
- Multi-core task migration software layer
  - Integration into Linux DVFS framework
  - Results
- Outlook

big.LITTLE processing? No Hardware available yet! But…

- Synopsys provides Virtual Prototypes for ARM Cortex CPUs
  - Matching the well known ARM Versatile Express uATX Motherboard
  - With reference software: Linux, Android
  - With environment: touch-screen, console terminal, etc.
- We just needed to upgrade it to include a Cortex-A15 + Cortex-A7 "virtual" coreTile daughter board



Task Migration – Software Integration



"Payload" Software Kernel/User Mode

Android

Android Power Management

Linux

Linux Power Management (DVFS/ CPUFreq)

Hypervisor Mode

Task Migration

Hypervisor-Mode

big LITTLE

big.LITTLE

Platform

big

LITTLE

A15

A7

New!

Android can guide Linux to provide "better" A7/A15 usage through device use context awareness

Task migration control via existing Linux performance scaling framework.

Task migration operates in highest privilege mode. Traps accesses to HW and presents one cluster to the OS.
Virtualizes memory and interrupts

Only one cluster is active at a t time through a controller block that holds/releases reset.

A look at the software side

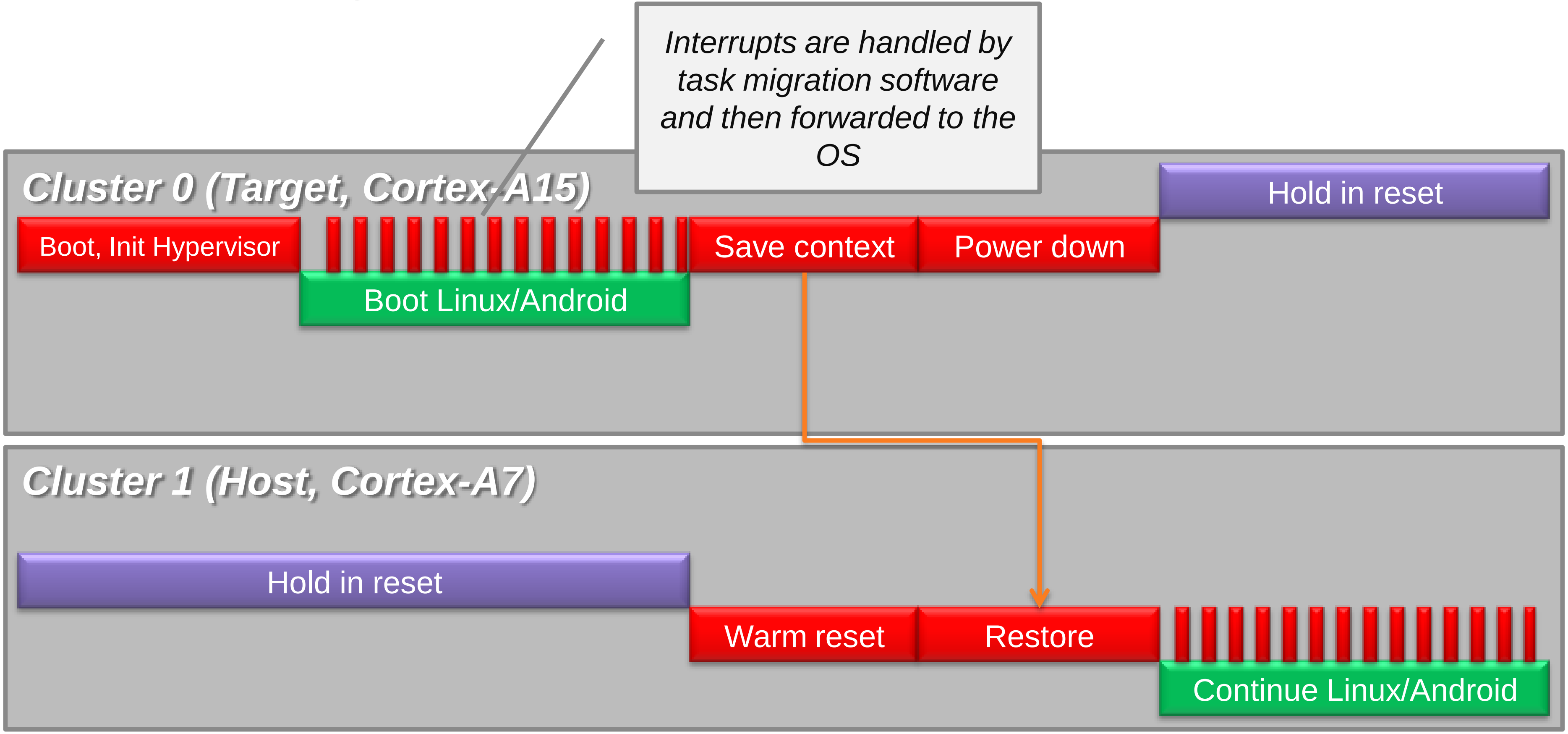
What we have done:

- Updated the task migration software layer to interface to Linux
  - Not only every 12M cycles
- Integrated with Linux DVFS/CPUFreq PM
  - In the coreTile code tree (in ./arch/arm/mach-vexpress)
    - cpufreq
  - Overloaded function to change the CPU frequency
    - Cortex-A7 and Cortex-A15 are just treated as different power states
- Play with use case scenarios
  - Play with android

What we have not done:

- Modified Governors of Linux standard Linux power management
  - i.e.: the DVFS/ CPUFreq PM framework
- Modified Android Power Manager or created “User space” governors

Task Migration – Execution Flow


Interrupts are handled by task migration software and then forwarded to the OS
Cluster 0 (Target, Cortex-A15)
Boot, Init Hypervisor
Boot Linux/Android
Save context
Power down
Hold in reset
Cluster 1 (Host, Cortex-A7)
Hold in reset
Warm reset
Restore
Continue Linux/Android

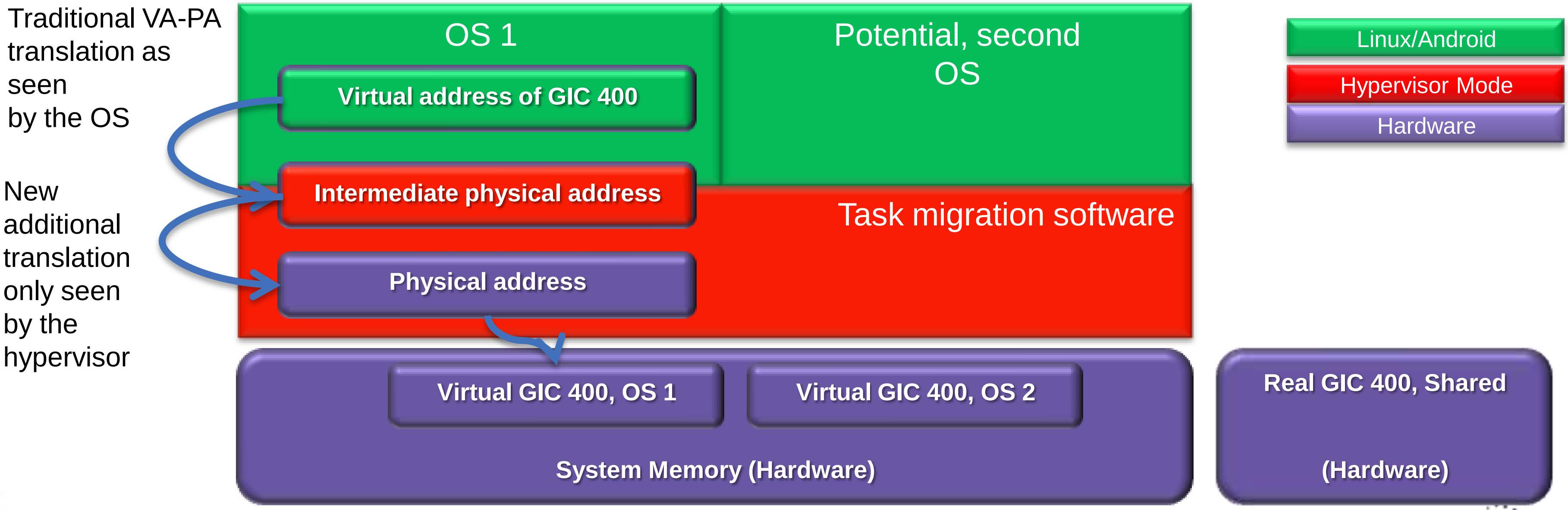
🟥 Hypervisor Mode

🟩 Linux/Android

🟪 Hardware

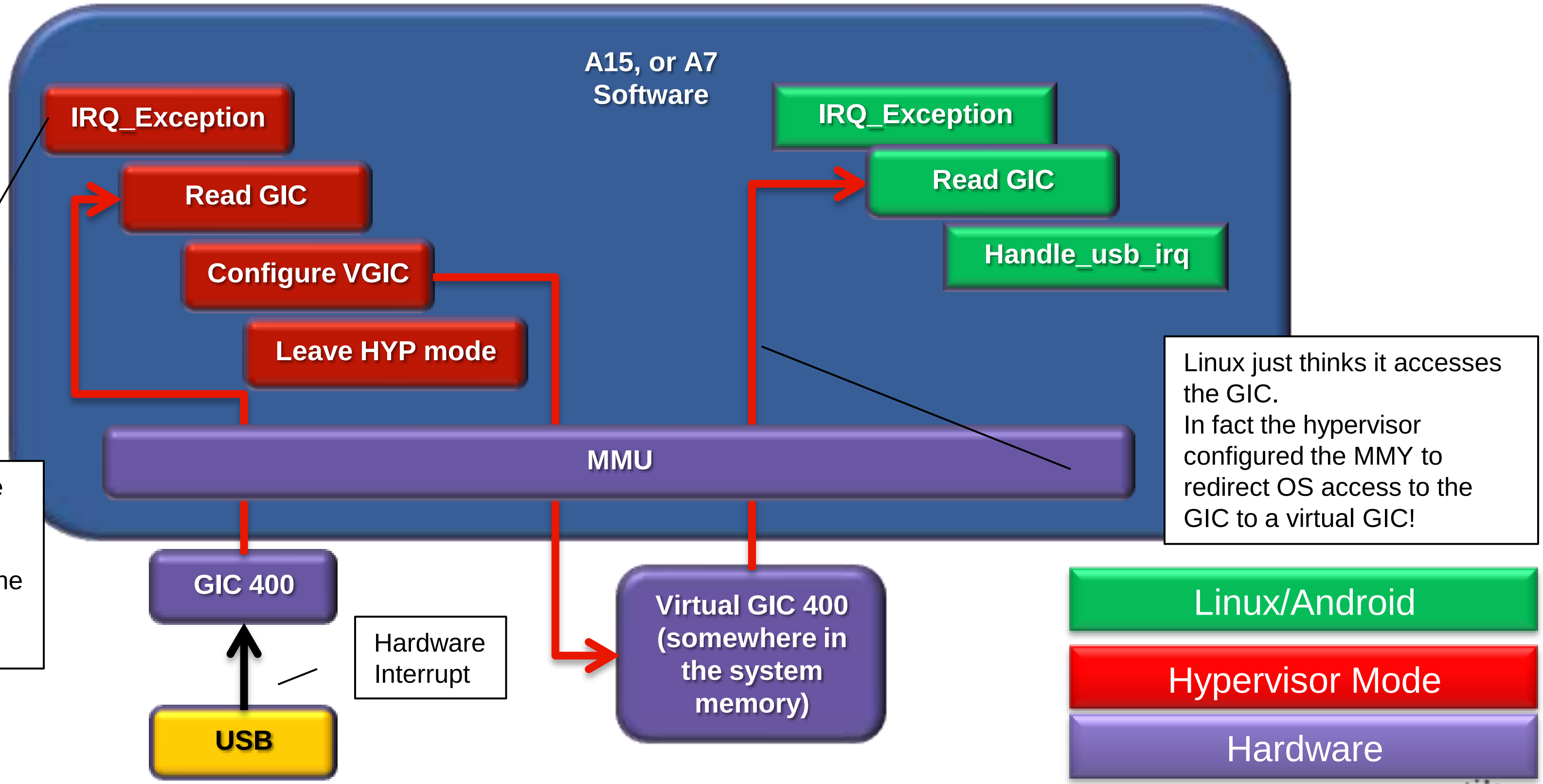
Memory Virtualization

- Without virtualization (direct translation):
  - The OS owns the memory
  - MMU translates from virtual (software) to physical (hardware)
- With virtualization (two stage translation):
  - MMU translates from virtual (software) to intermediate physical (virtual hardware) to physical addresses (hardware)
- Allows creation of virtual devices such as the virtual GIC
  - OS thinks it talks to GIC, but talks to virtual GIC (VGIC)



Interrupt Virtualization

- Interrupts from the GIC are trapped in the hypervisor (not Linux)
- Task migration SW reads physical interrupt configures a virtual GIC
- Task migration SW configures MMU to redirect GIC accesses of the OS
- Linux accesses virtual GIC (without knowing it)



Before Linux will see this interrupt, it is trapped by the hypervisor through the new higher privilege HYP mode.

Hardware Interrupt

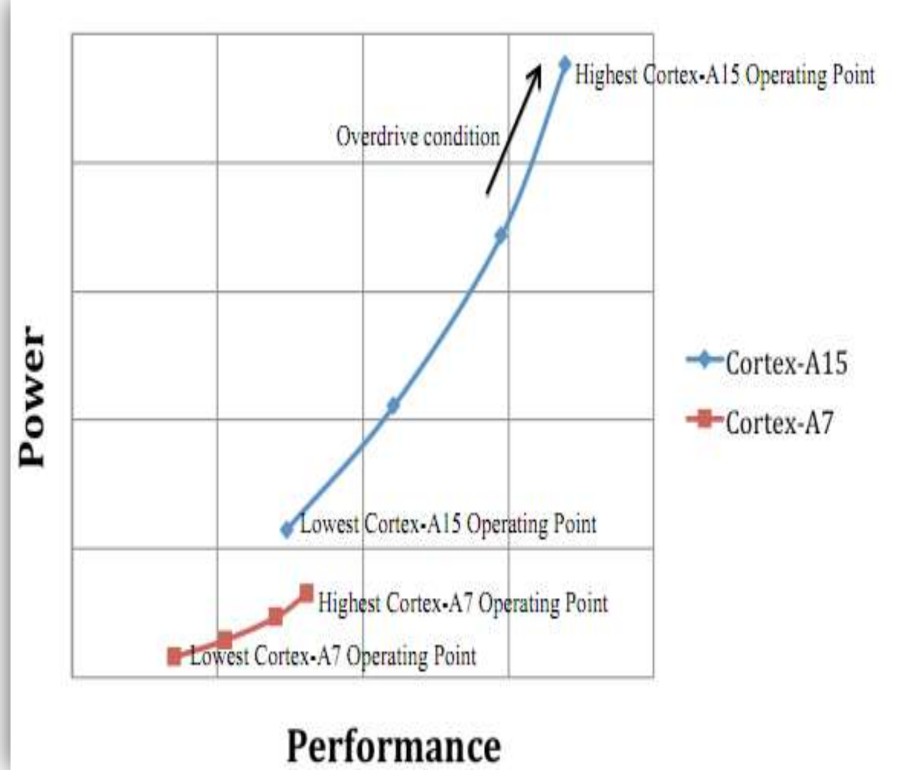
Linux just thinks it accesses the GIC.
In fact the hypervisor configured the MMY to redirect OS access to the GIC to a virtual GIC!

Agenda

- big.LITTLE processing introduction
  - In a nutshell
  - Background, motivation & technology
- Multi-core task migration software layer
  - Integration into Linux DVFS framework
  - Results
- Outlook

Explore Energy/performance based on user Scenarios

1) Power ratio between Cortex-A15 and Cortex-A7



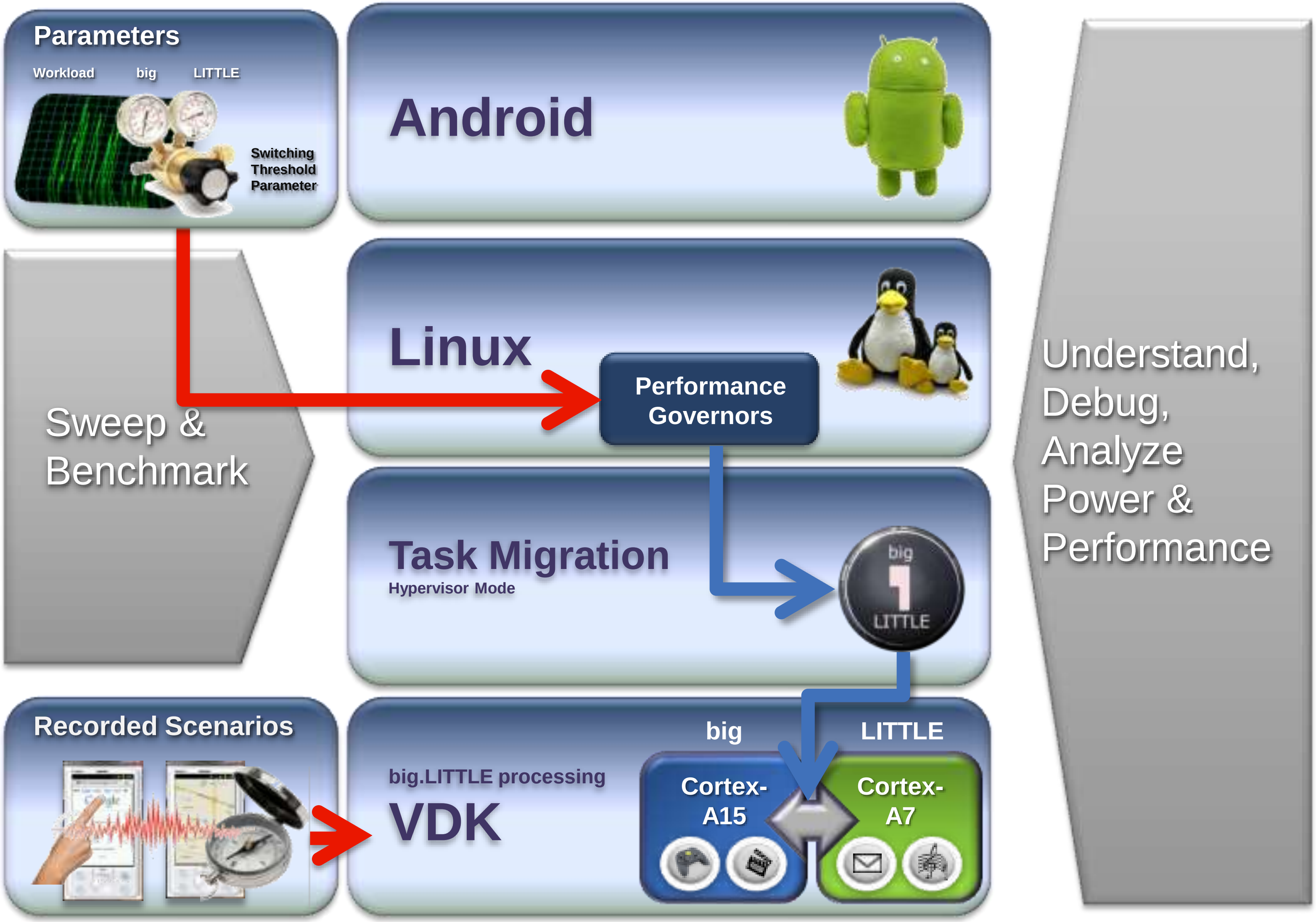
3) Software CPU load task migration threshold/strategy

via Linux CPUFreq driver

2) Performance ratio between A15 and A7

vi

Energy/performance optimizations



Thanks you!

Questions? Suggestions?

sylvainb (at) synopsys (dot) com

http://www.synopsys.com/VDK4big-LITTLE/

SYNOPSYS® 25 years