big.LITTLE technology

- 1. Introduction to the big.LITTLE technology
- 2. Exclusive cluster migration (not discussed)
- 3. Inclusive cluster migration (not discussed)
- 4. Exclusive core migration (not discussed)
- 5. Global task scheduling (GTS)
- 6. Supporting GTS in OS kernels (partly discussed)
- 7. References

1. Introduction to the big.LITTLE technology

- 1.1 The rationale for big.LITTLE processing
- 1.2 Principle of big.LITTLE processing
- 1.3 Implementation of the big.LITTLE technology

1.1 The rationale for big.LITTLE processing

1.1 The rationale for big.LITTLE processing (1)

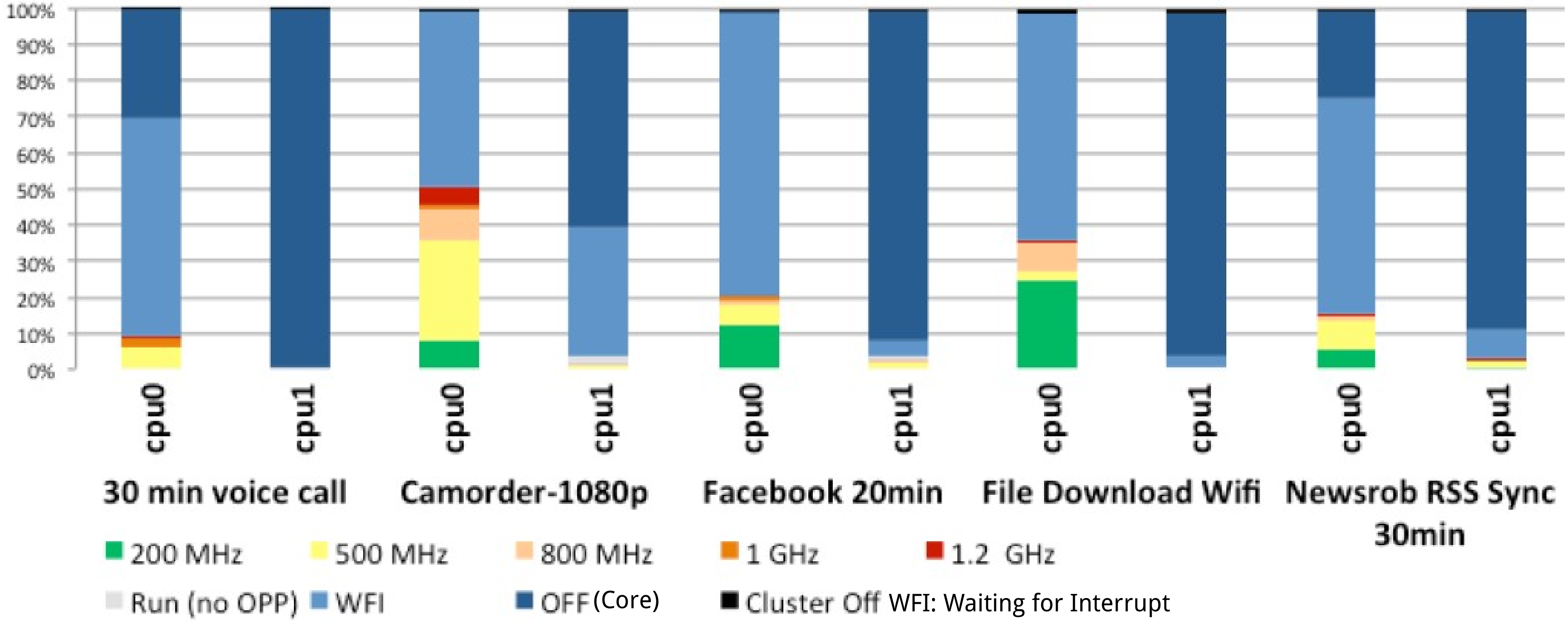
1.1 The rationale for big.LITTLE processing [1]

- big.LITTLE technology aims primary at power reduction.
- The technology is based on the recognition that the workload of smartphones and tablets is dynamically changing, that is tasks with different processing intensities alternate, such as high intensity tasks, like games and low intensity tasks, like audio, e-mail etc.
- When now the processor has two different kind of cores implementing the same ISA. i.e.
  - a cluster of powerfull and power hungry "big" cores and
  - another cluster of less powerfull but less power hungry "LITTLE" cores

  and low intensity tasks will run on less powerful but less power hungry cores the entire power consumption of the chip could be reduced.

1.1 The rationale for big.LITTLE processing (2)

Example: Percentage of time spent in DVFS states and further power states in a dual core mobile device for low intensity applications [9] -1



WFI: Waiting for Interrupt

- The mobile device is a dual core Cortex-A9 based mobile device.
- In the diagram, the red color indicates the highest, green the lowest frequency operating point whereas colors in between represent intermediate frequencies.
- In addition, the OS power management idles a CPU for Waiting for Interrupt (WFI) (light blue) or even shuts down a core (dark blue) or the cluster (darkest blue).

1.1 The rationale for big.LITTLE processing (3)

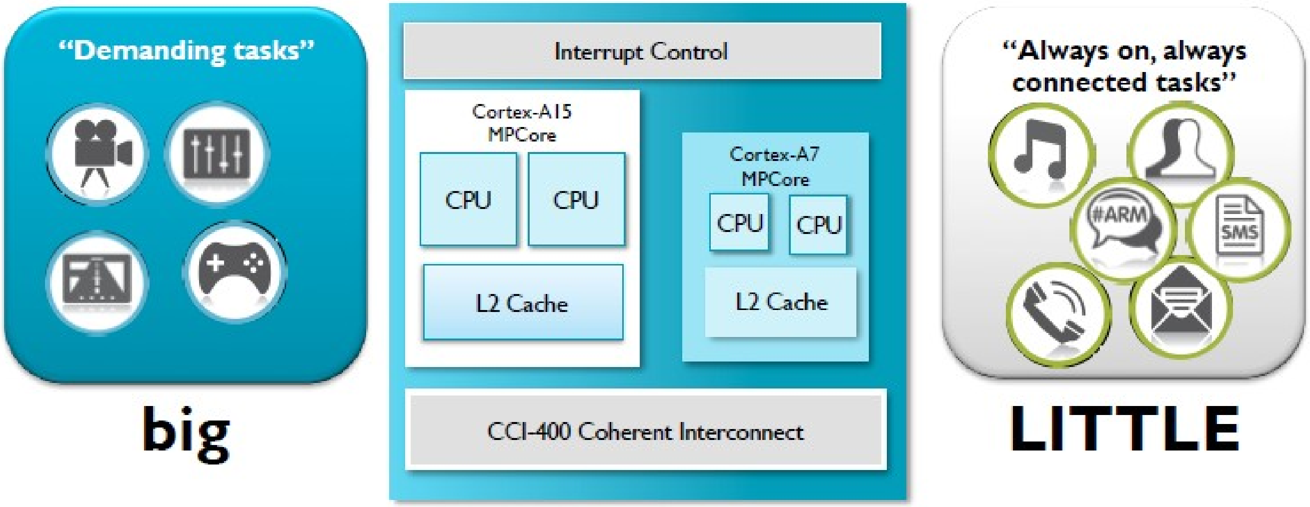
Example: Percentage of time spent in DVFS states and further power states in a mobile device for low intensity applications [9] -2

As seen in the diagram, both cores spend most on their time in idle, shut down or light intensity operating points, thus there is a large headroom for power saving through the big.LITTLE technology.

1.1 The rationale for big.LITTLE processing (4)

Expected results of using the big and LITTLE technology [2]

- Uses the right processor for the right job
- Up to 70% energy savings on common workloads
- Flexible and transparent to apps – importance of seamless software handover



1.1 The rationale for big.LITTLE processing (5)

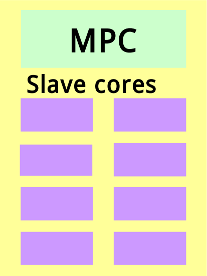
big.LITTLE technology as an option of task distribution policy in heterogeneous multicore processors

Task distribution policies in heterogeneous multicore processors

Master/slave processing

Heterogeneous master/slave processing

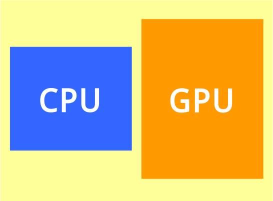
There is a master core (MCP)
and a number of slave cores.
The master core organizes
the operation of the slave cores
to execute a task



Task forwarding to a dedicated accelerator

Heterogeneous attached processing

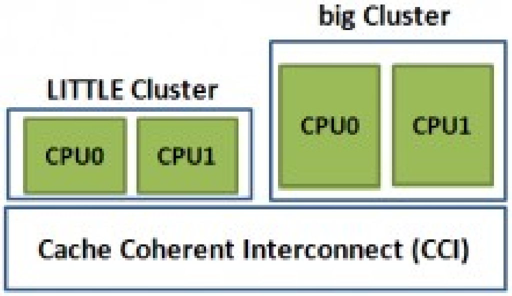
Beyond a CPU there are
dedicated accelerators,
like a GPU available.
The CPU forwards an instructon
to an accelerator
when it is capable to execute
this instruction more efficiently
than the CPU.



Task migration to different kind of CPUs

Heterogeneous big.LITLE processing

There two or more clusters of cores,
e.g. two clusters; a LITTLE and a big one.
Cores of the LITTLE cluster execute
less demanding tasks and consume
less power, whereas cores of the big
cluster execute more demanding tasks
with higher power consumption.



1.2 Principle of big.LITTLE processing

1.2 Principle of big.LITTLE processing (1)

1.2 Principle of big.LITTLE processing [6]

1.2 Principle of big.LITTLE processing (2)

Assumed platform

Let's have two or more clusters of architecturally identical cores in a processor.
As an example let's take two clusters;

- a cluster of low performance/low power cores, termed as the LITTLE cores and
- a cluster of higher performance higher power cores, termed as the big cores, as seen in the Figure below.

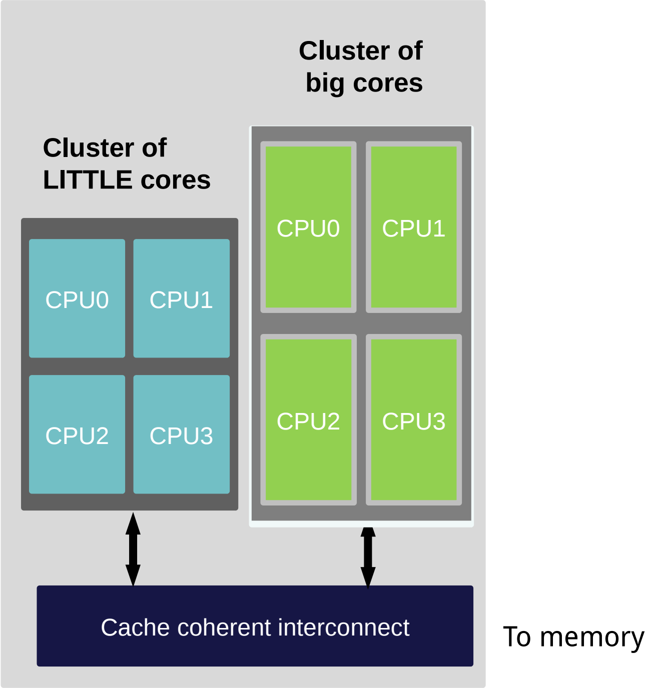


Figure: A big.LITTLE configuration consisting of two clusters

- Let's interconnect these clusters by a cache coherent interconnect to have a multicore processor, as indicated in the Figure.

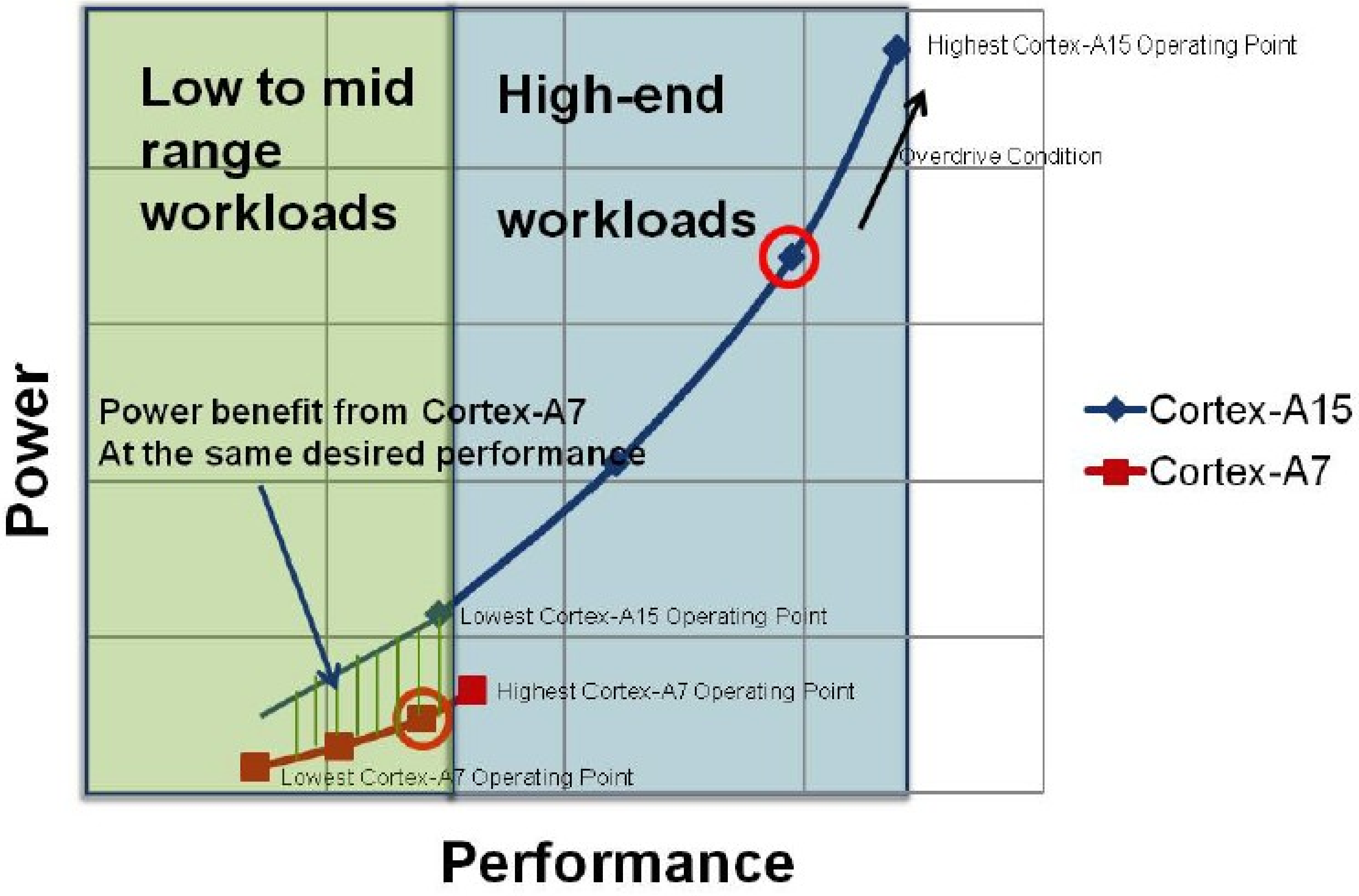
1.2 Principle of big.LITTLE processing (3)

Principle of operation -1 [4]

- There are a number of different alternatives how big.LITTLE technology can be implemented, as discussed in the next Section.
- Nevertheless, to give a conceivable description of the principle of operation of the big.LITTLE technology, we will restrict ourselves to a specific implementation, as detailed subsequently.
- We take for granted that there are two clusters of cores, one with less powerful but more power efficient LITTLE cores and another one with more powerful
- big cores that consume more power.
- From a number of fesible task scheduling algorithms, to be discussed in Section 1.3, here we outline the so called exclusive cluster switching one.
- Furthermore let us suppose that all cores of a cluster operate at the same operating point (i.e. at the same clock frequency and core voltage) while there are a few operating points available for both the LITTLE and the big core clusters, as indicated in the next Figure.

1.2 Principle of big.LITTLE processing (4)

Example: Operating points of a multiprocessor built up of two core clusters: one of LITTLE cores (Cortex-A7) and one of big cores (Cortex-A15), as described above [3]



1.2 Principle of big.LITTLE processing (5)

Example big (Cortex-A15) and LITTLE cores (Cortex-A7) [3]

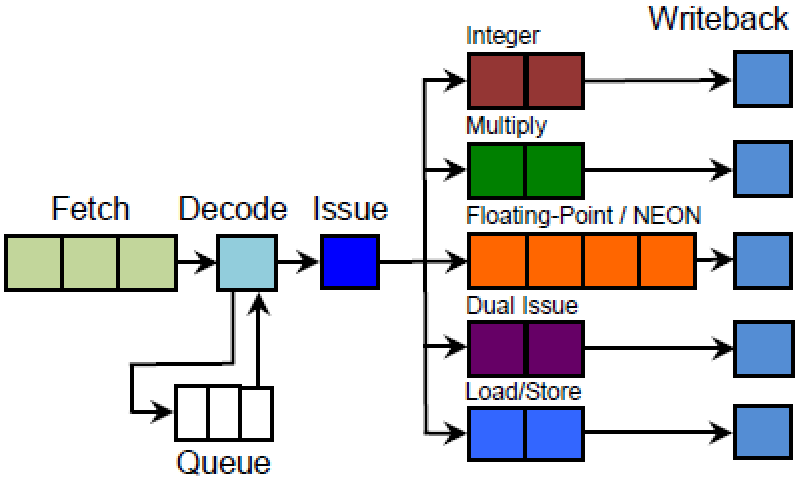


Figure 1 Cortex-A7 Pipeline

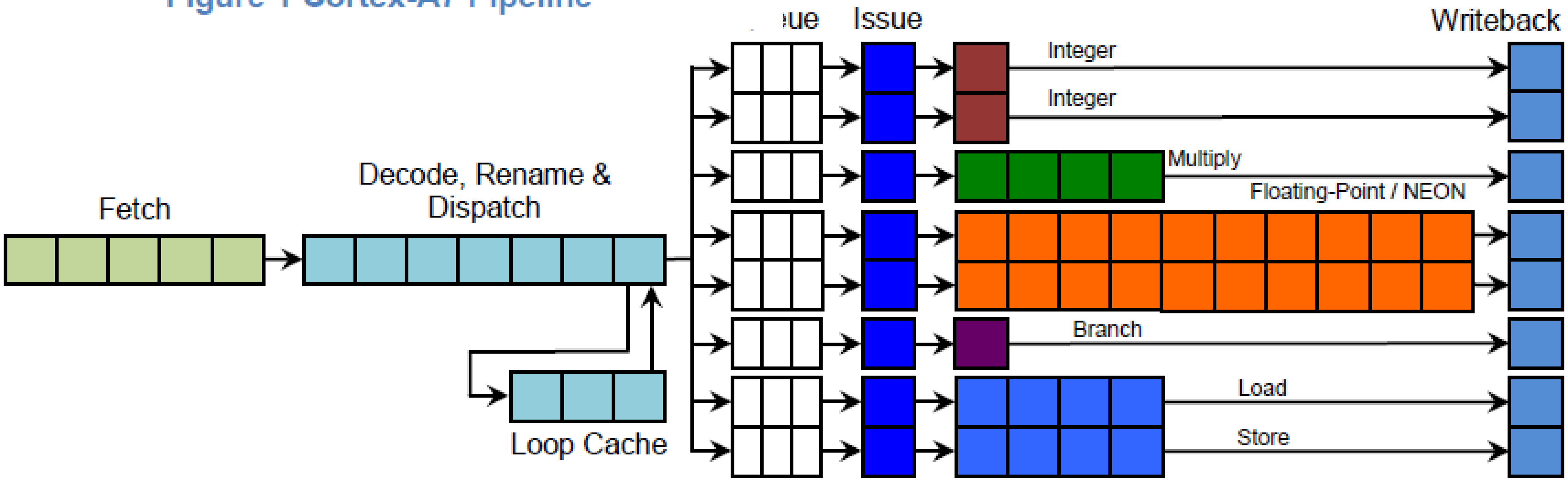
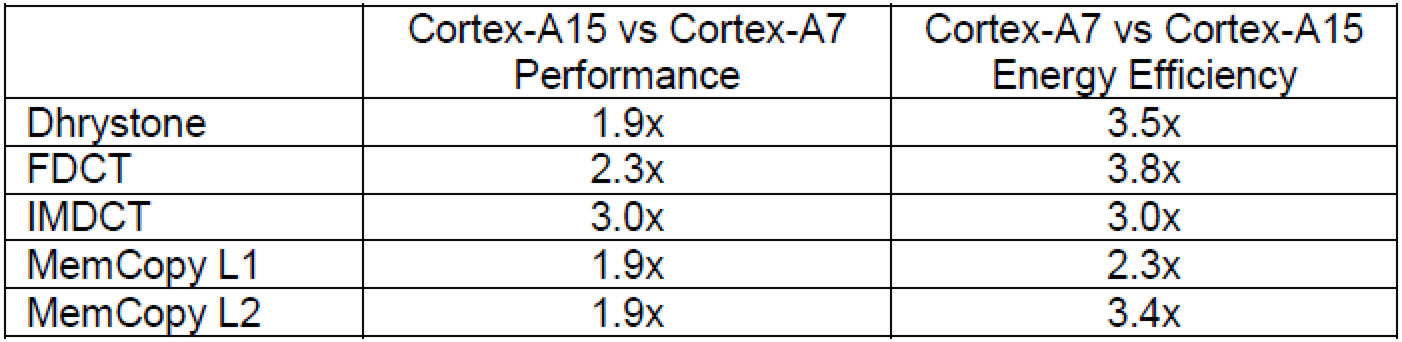


Figure 2 Cortex-A15 Pipeline

1.2 Principle of big.LITTLE processing (6)

Performance and energy efficiency comparison of the Cortex-A15 vs. the Cortex-A7 cores [3]



	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

1.2 Principle of big.LITTLE processing (7)

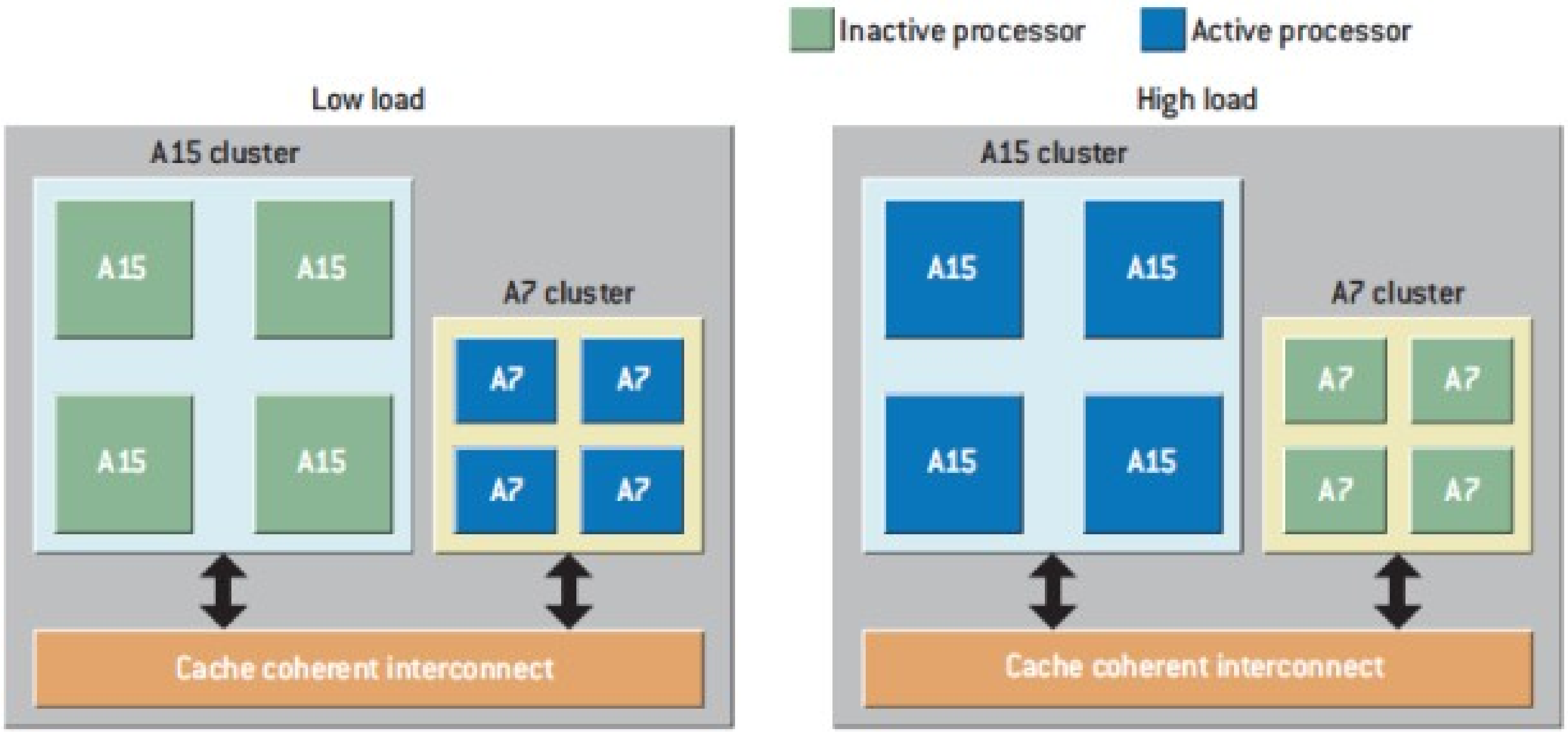
Principle of operation -2 [4]

- Let’s suppose that an appropriate OS routine (e.g. the cpufreq kernel routine of Linux) tracks the workload and sets the operating point accordingly.
- In the assumed task scheduling mode, the scheduler activates only the cores of the low power, low performance „ LITTLE” cluster as long as this cluster can tackle the actual workload.

  If however the workload exceeds the performance capability of the LITTLE core cluster, an appropriate switching routine activates the cluster of high performance high power “big” cores, performs a cluster switch by migrating the actual workload to the cores of the big cluster and switches off the cores of the LITTLE cluster.
- In this way, at any given time only cores of one cluster can be active.
- Again, the working point of the big cores will be chosen according to the workload demand.
- When any time the actual workload becomes less then a given lower limit of the big core cluster, the scheduler activates again the cluster of the LITTLE cores etc.

1.2 Principle of big.LITTLE processing (8)

Illustration of the described model of operation [7]



Note that at low load the LITTLE (A7) and at high load the big (A15) cluster is operational.

1.3 Implementation of the big.LITTLE technology

1.3 Implementation of the big.LITTLE technology (1)

1.3 Implementation of the big.LITTLE technology

Hardware requirements of big.LITTLE systems [1]

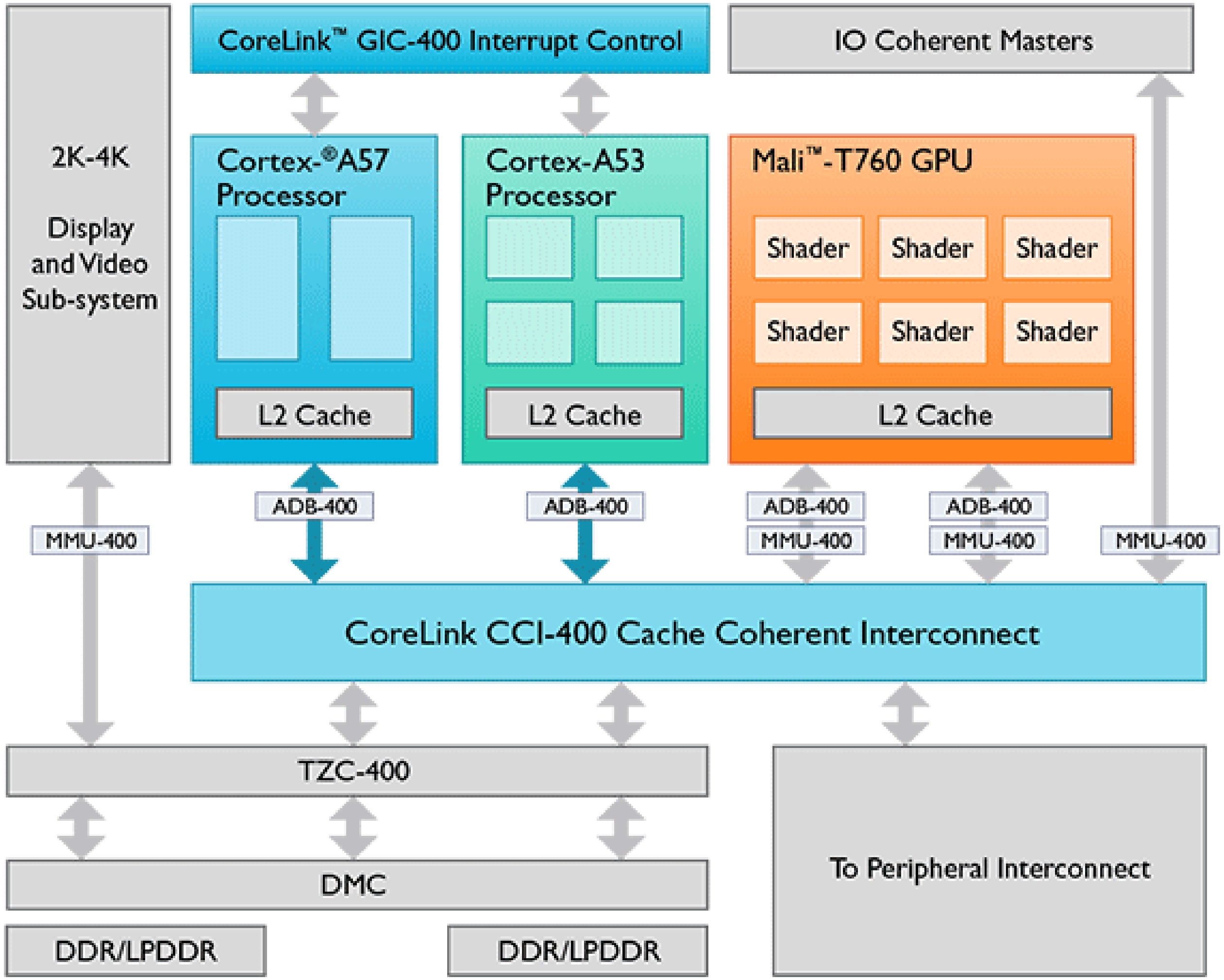
- big and LITTLE cores of a big.LITTLE system need to satisfy two key requirements for a seamless operation, such as
  - the big and LITTLE CPU cores must be architecturally identical, i.e. they must have the same ISA, i.e. run the same instructions and support the same ISA extensions, like virtualization, address space etc. and
  - both core clusters must be fully cache coherent to support task migration.
- Recommended core pairings are

	• 1st Generation: ARMv7 (32-bit, 40-bit physical)	• 2nd Generation: ARMv8 (32-bit/64-bit)
• High-performance CPU core (big)	• Cortex-A15, Cortex-A17	• Cortex-A57, Cortex-A72
• High-efficiency CPU core (LITTLE)	• Cortex-A7	• Cortex-A53, Cortex-A35

- In each of the core combinations above, the big and LITTLE clusters can each have up to four cores.

1.3 Implementation of the big.LITTLE technology (2)

Example block diagram of a two cluster big.LITTLE SOC design [1]



ADB-400: AMBA Domain Bridge
AMBA: Advanced Microcontroller Bus Architecture

MMU-400: Memory Management Unit
TZC: Trust Zone Address Space Controller

1.3 Implementation of the big.LITTLE technology (3)

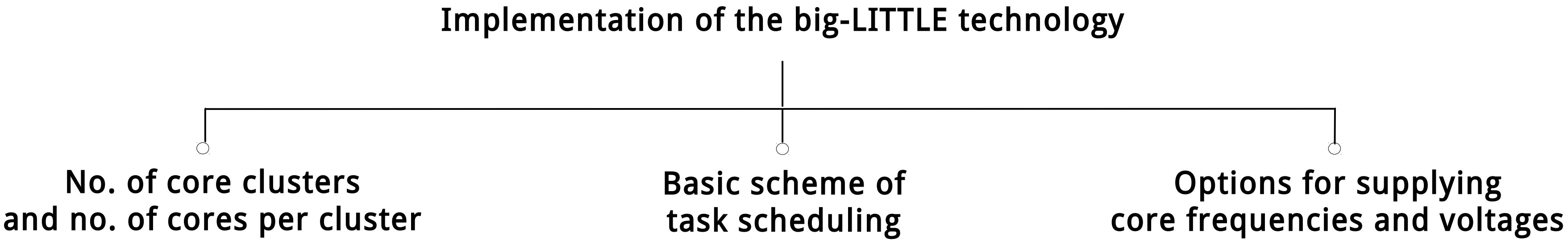
Required software support of the big.LITTLE technology

- The big.LITTLE technology needs appropriate software support, e.g. to monitor the workload, task scheduling, perform task migrations from one core to another, etc.
- The software needed will be provided by ARM/Linaro or will be developed by the SOC designer, usually in form of kernel patches.
- In this chapter we will not go into details of the software required by the big.LITTLE technology, but refer to the Section 6. and the related literature.

1.3 Implementation of the big.LITTLE technology (4)

Design space of the implementation of the big.LITTLE technology

In our discussion of the big.LITTLE technology we take into account three basic design aspects, as follows:



Implementation of the big-LITTLE technology

No. of core clusters and no. of cores per cluster

Basic scheme of task scheduling

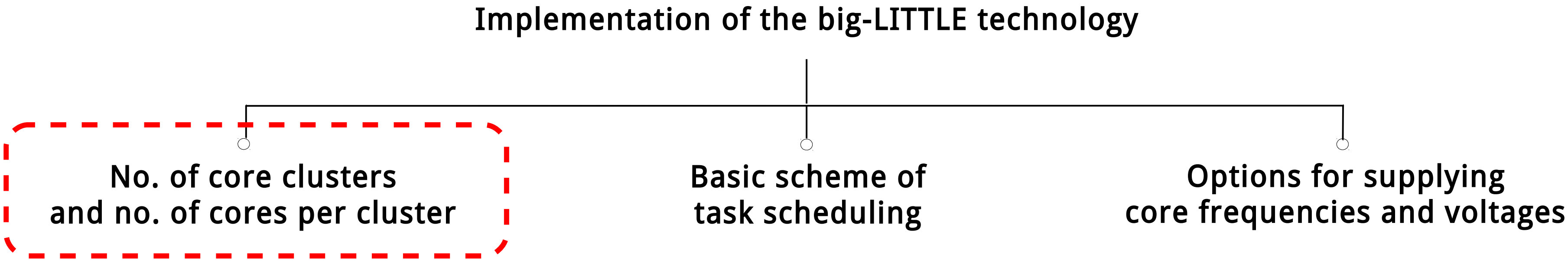
Options for supplying core frequencies and voltages

These aspects will be discussed subsequently.

1.3 Implementation of the big.LITTLE technology (5)

Design space of the implementation of the big.LITTLE technology

In our discussion of the big.LITTLE technology we identify three basic design aspects, as follows:



Implementation of the big-LITTLE technology

No. of core clusters
and no. of cores per cluster

Basic scheme of
task scheduling

Options for supplying
core frequencies and voltages

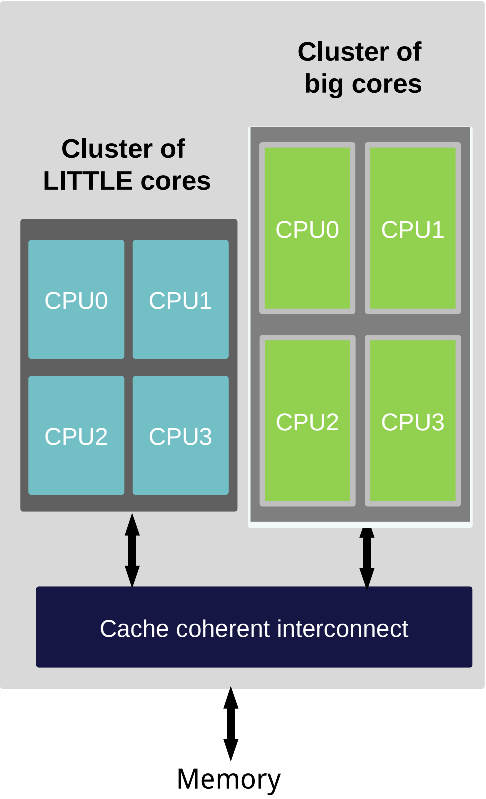
These aspects will be discussed subsequently.

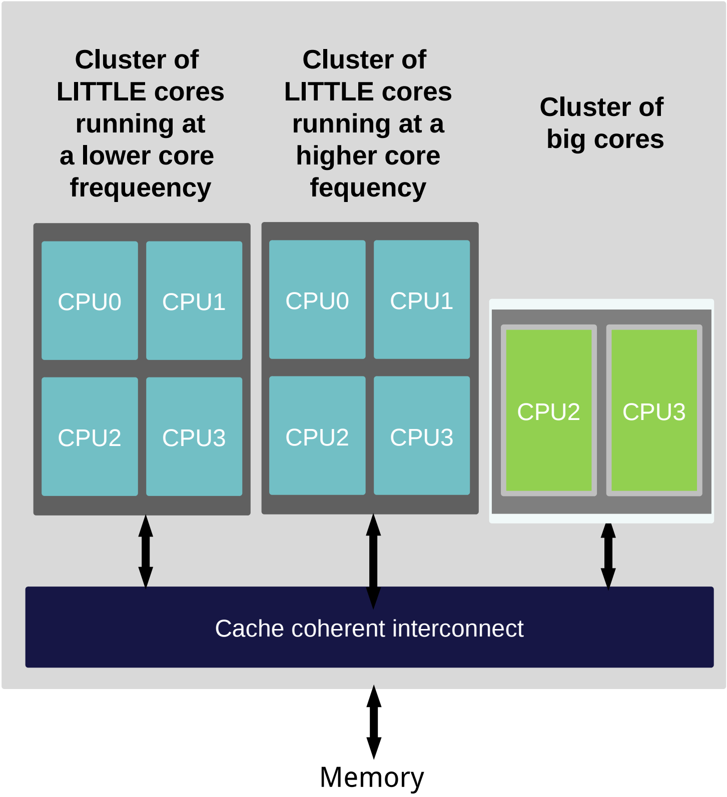
1.3 Implementation of the big.LITTLE technology (6)

Number of core clusters and number of cores per cluster

Number of core clusters and number of cores per cluster

	Dual core clusters used	Three core clusters used
Example configurations	4 + 4 2 + 2 1 + 4	4 + 4 + 2 4 + 2 + 4





Cluster of LITTLE cores running at a lower core frequeency
Cluster of LITTLE cores running at a higher core fequency
Cluster of big cores
CPU0
CPU1
CPU2
CPU3
CPU0
CPU1
CPU2
CPU3
CPU2
CPU3
Cache coherent interconnect
Memory

1.3 Implementation of the big.LITTLE technology (7)

Example 1: Dual core clusters, 4+4 cores: Samsung Exynos Octa 5410 [11]

- It is the world’s first octa core mobile processor.
- Announced in 11/2012, launched in some Galaxy S4 models in 4/2013.

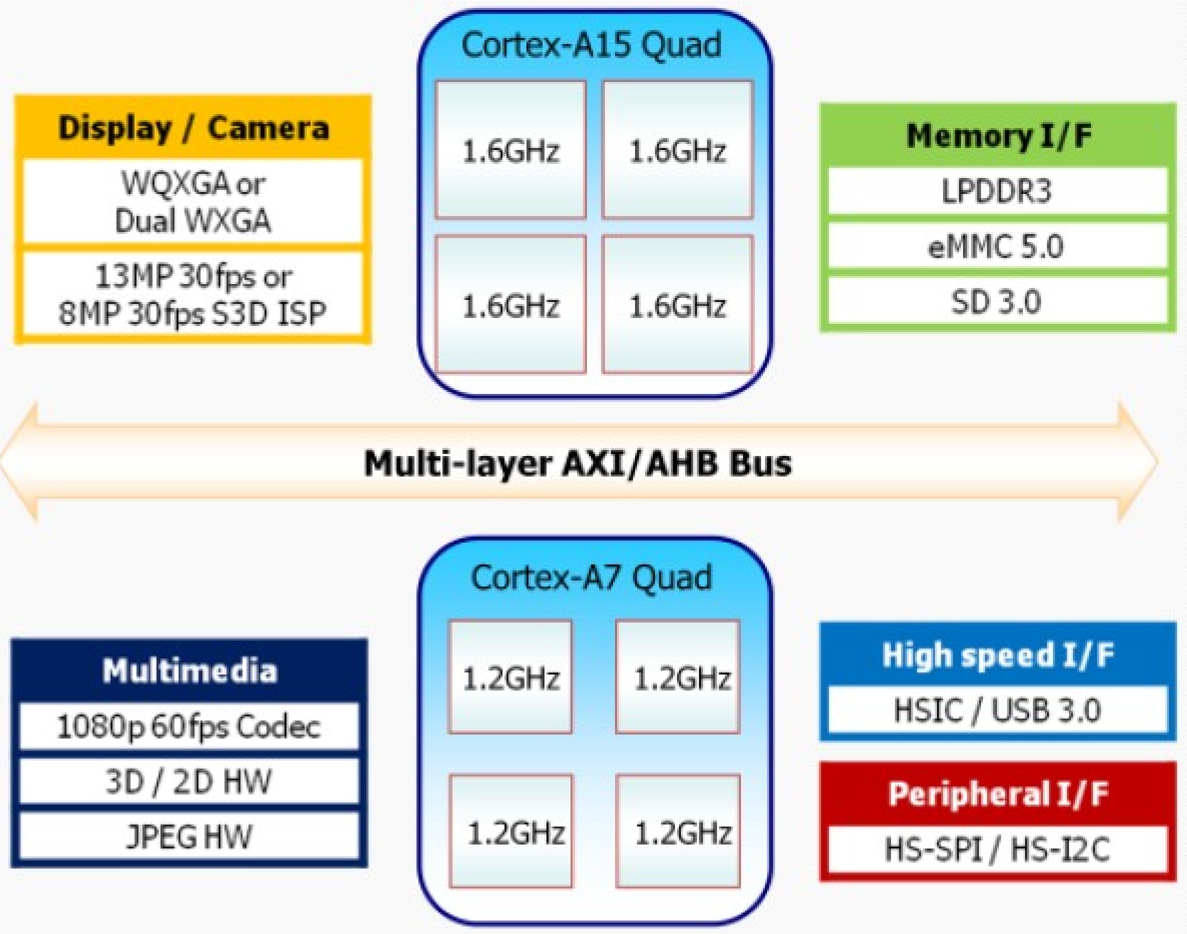


Figure: Block diagram of Samsung’s Exynos 5 Octa 5410 [11]

1.3 Implementation of the big.LITTLE technology (8)

Example 2: Three core clusters, 2+4+4 cores: Helio X20 (MediaTek MT6797)

- In this case, each core cluster has different operating characteristics, as indicated in the next Figures.
- Announced in 9/2015, to be launched in HTC One A9 in 11/2015.

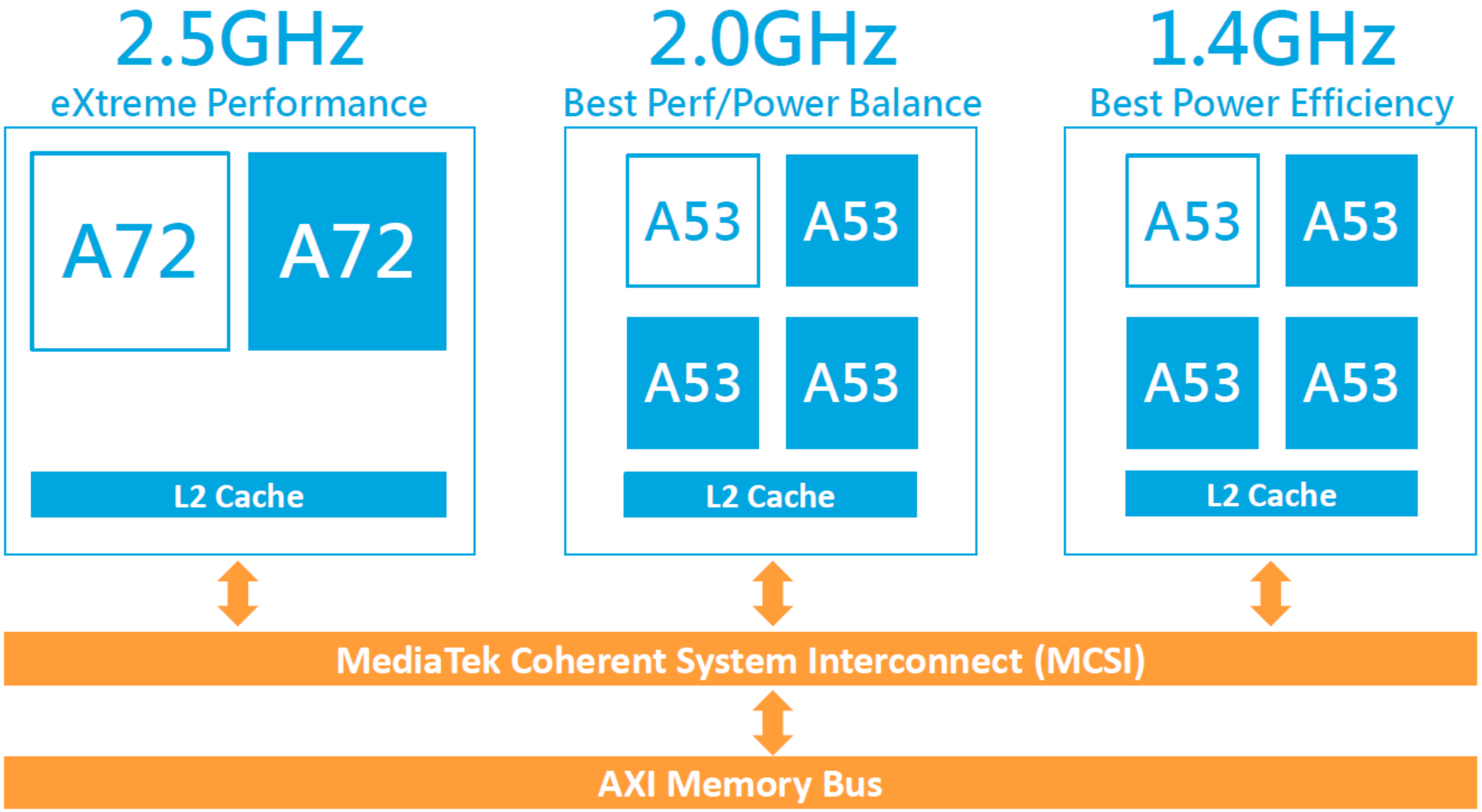
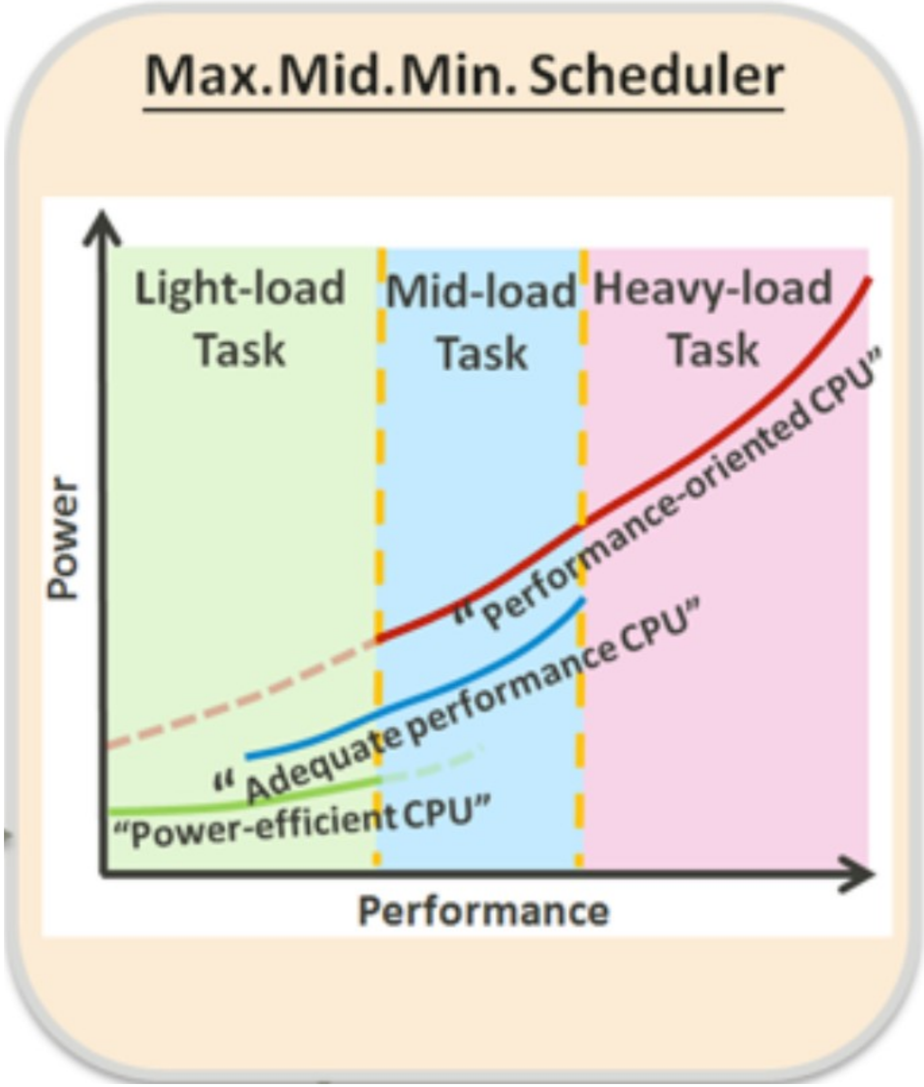


Figure: big.LITTLE implementation with three core clusters (MT6797) [46]

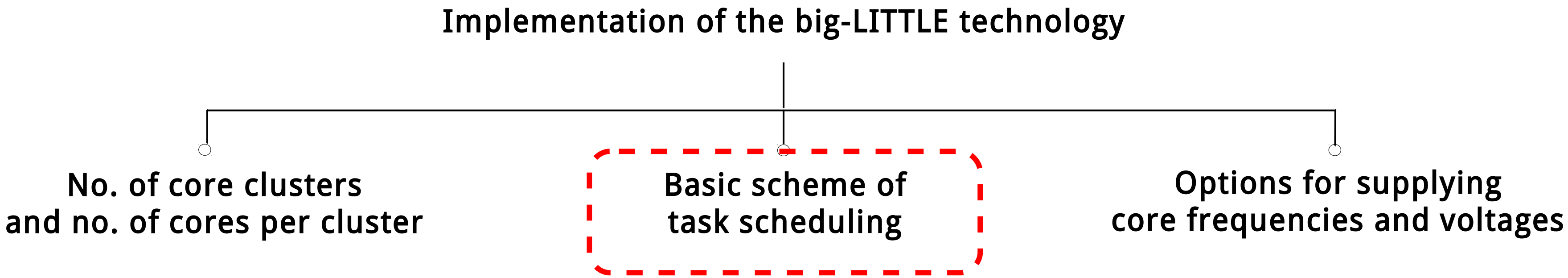
1.3 Implementation of the big.LITTLE technology (9)

Power-performance characteristics of the three clusters [47]



1.3 Implementation of the big.LITTLE technology (10)

Basic scheme of task scheduling



Implementation of the big-LITTLE technology

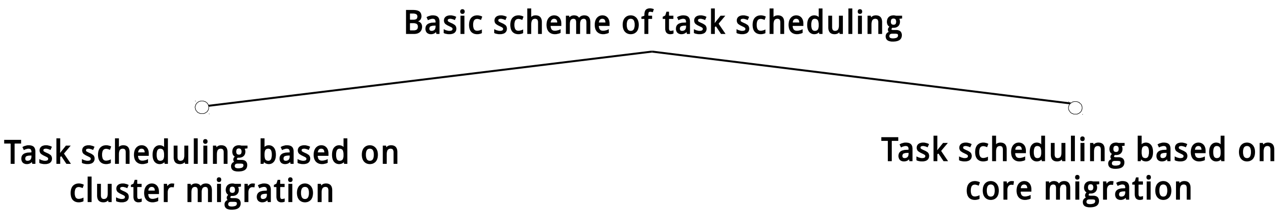
No. of core clusters
and no. of cores per cluster

Basic scheme of
task scheduling

Options for supplying
core frequencies and voltages

1.3 Implementation of the big.LITTLE technology (11)

Basic scheme of task scheduling



Basic scheme of task scheduling

Task scheduling based on cluster migration

Task scheduling based on core migration

1.3 Implementation of the big.LITTLE technology (12)

Task schedulinng based on cluster migration (assuming two clusters)

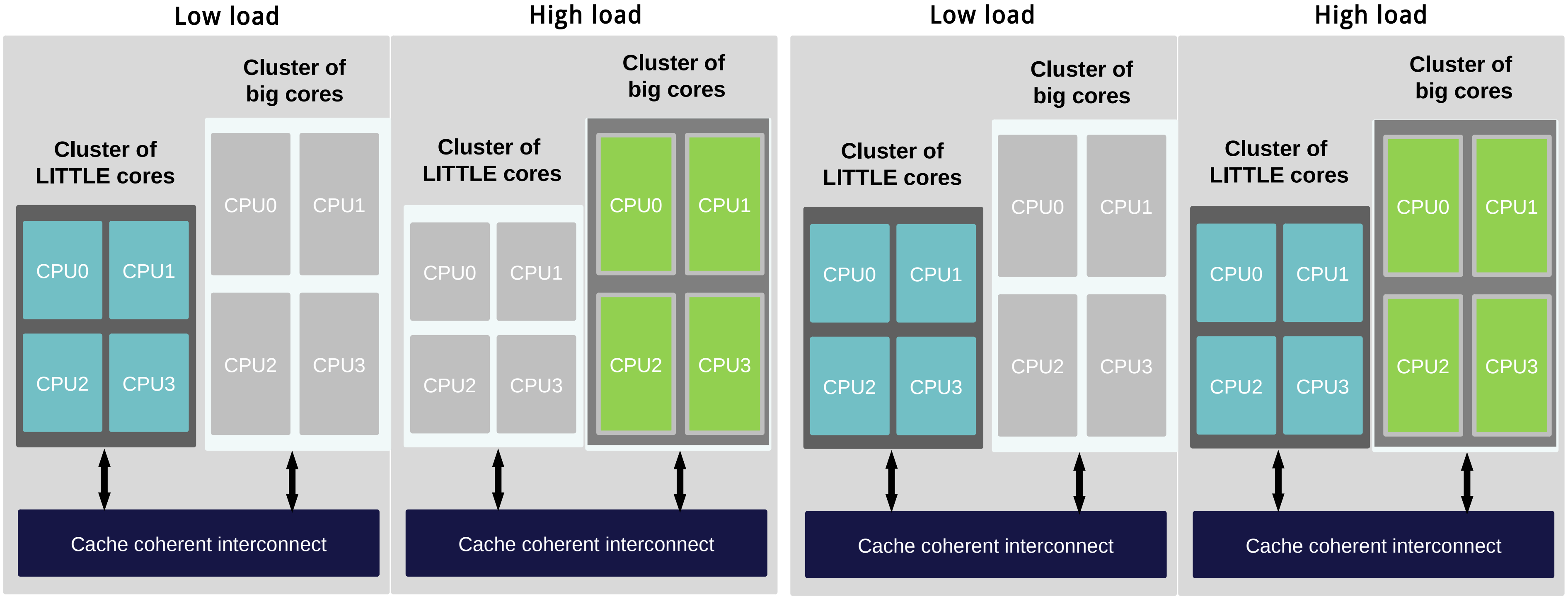
Task scheduling based on cluster migration

Exclusive use of the clusters

At any time
either the big or the LITTLE cluster is in use

Inclusive use of the clusters

For low workloads only the LITTLE
but for high workloads both the
big and the LITTLE clusters are in use



Exclusive cluster migration

Inclusive cluster migration

1.3 Implementation of the big.LITTLE technology (13)

Task scheduling based on core migration (assuming two clusters)

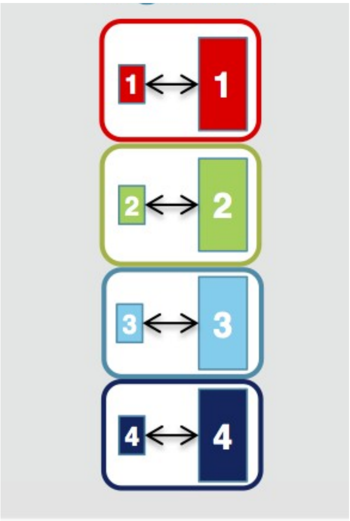
Task scheduling based on core migration

Exclusive use of cores in big.LITTLE core pairs

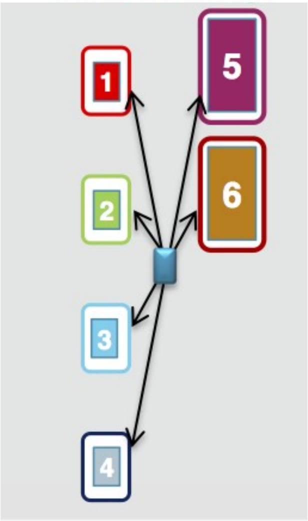
big and LITTLE cores are ordered in pairs.
In each core pair
either the big or the LITTLE core is in use.

Inclusive use of all big and LITTLE cores

Both big and LITTLE cores may be used
at the same time.
A global scheduler allocates the workload
appropriately for all available
big and the LITTLE cores.



Exclusive core migration [48]

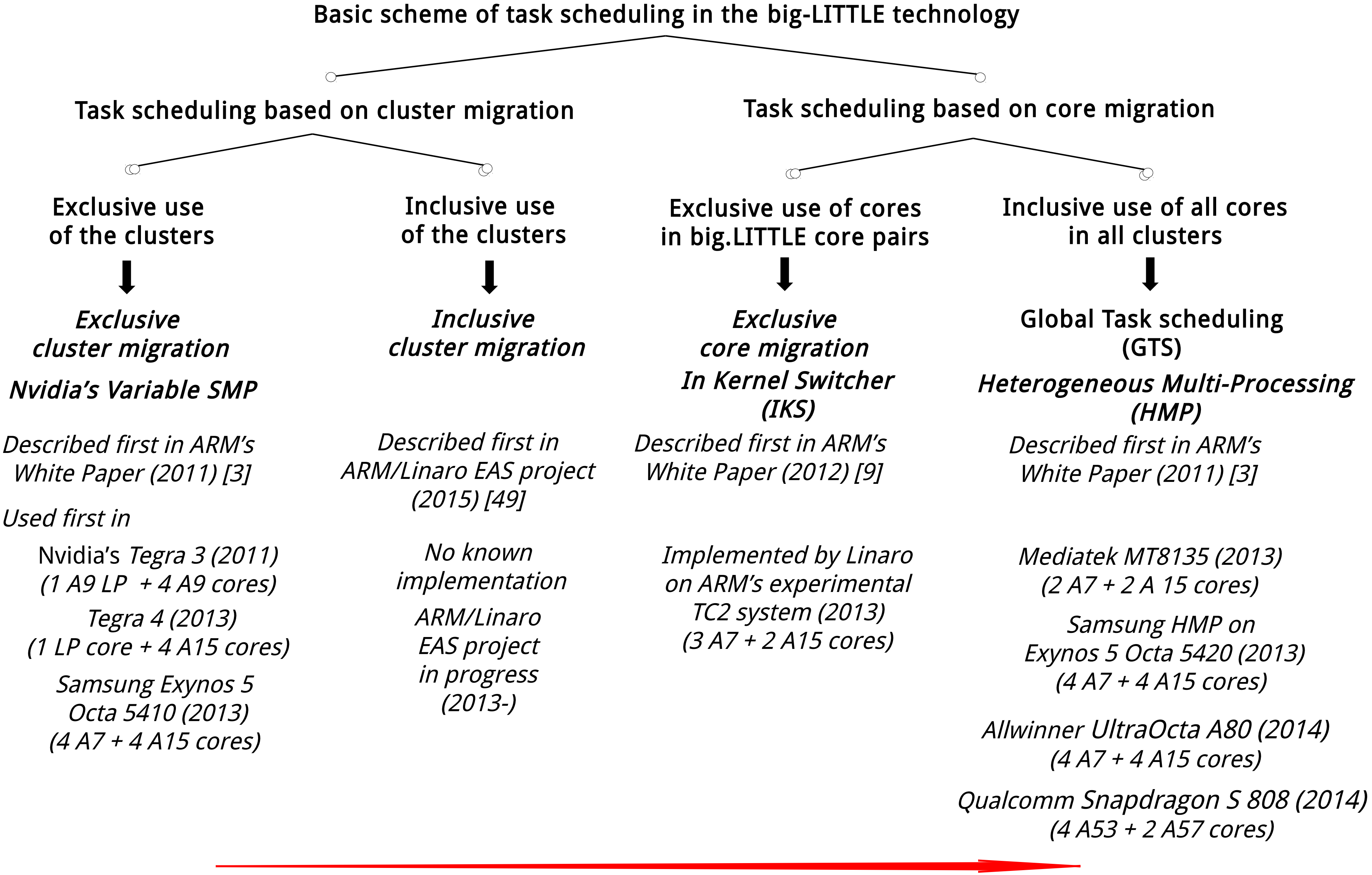


Global scheduler

Global Task Scheduling [48]

1.3 Implementation of the big.LITTLE technology (14)

Basic design space of task scheduling in the big.LITTLE technology



Basic scheme of task scheduling in the big-LITTLE technology

Task scheduling based on cluster migration

Exclusive use of the clusters

Exclusive cluster migration

Nvidia's Variable SMP

Described first in ARM's White Paper (2011) [3]

Used first in

Nvidia's *Tegra 3 (2011)*
(1 A9 LP + 4 A9 cores)

Tegra 4 (2013)
(1 LP core + 4 A15 cores)

Samsung Exynos 5 Octa 5410 (2013)
(4 A7 + 4 A15 cores)

Inclusive use of the clusters

Inclusive cluster migration

Described first in ARM/Linaro EAS project (2015) [49]

No known implementation

ARM/Linaro EAS project in progress (2013-)

Task scheduling based on core migration

Exclusive use of cores in big.LITTLE core pairs

Exclusive core migration

In Kernel Switcher (IKS)

Described first in ARM's White Paper (2012) [9]

Implemented by Linaro on ARM's experimental TC2 system (2013)
(3 A7 + 2 A15 cores)

Inclusive use of all cores in all clusters

Global Task scheduling (GTS)

Heterogeneous Multi-Processing (HMP)

Described first in ARM's White Paper (2011) [3]

Mediatek MT8135 (2013)
(2 A7 + 2 A 15 cores)

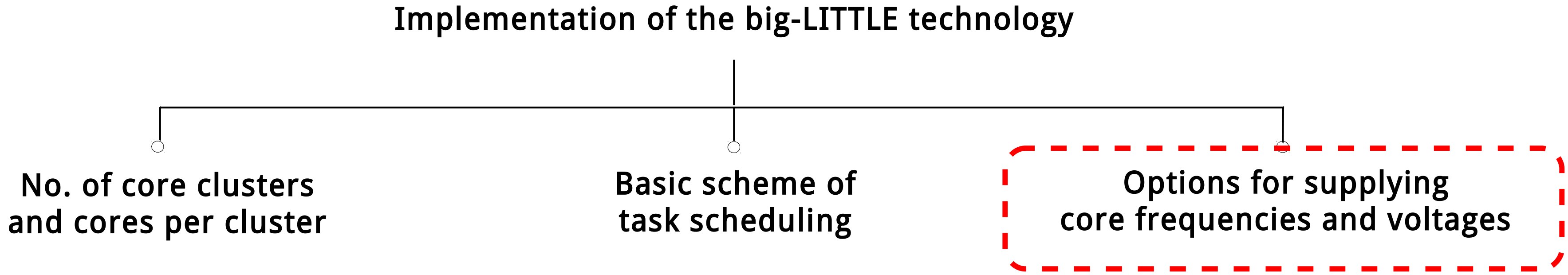
Samsung HMP on Exynos 5 Octa 5420 (2013)
(4 A7 + 4 A15 cores)

Allwinner UltraOcta A80 (2014)
(4 A7 + 4 A15 cores)

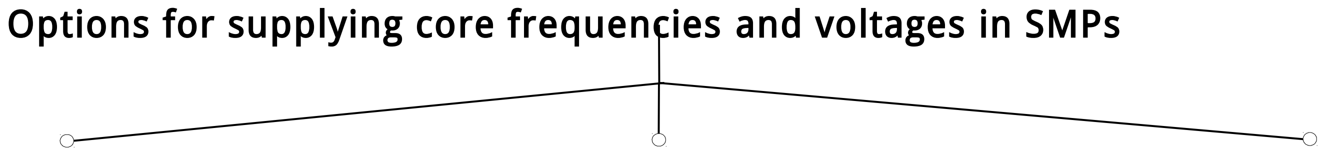
Qualcomm Snapdragon S 808 (2014)
(4 A53 + 2 A57 cores)

1.3 Implementation of the big.LITTLE technology (15)

Options for supplying core frequencies and voltages


Implementation of the big-LITTLE technology
No. of core clusters and cores per cluster
Basic scheme of task scheduling
Options for supplying core frequencies and voltages

Options for supplying core frequencies and voltages-1



Options for supplying core frequencies and voltages in SMPs

Synchronous CPU cores

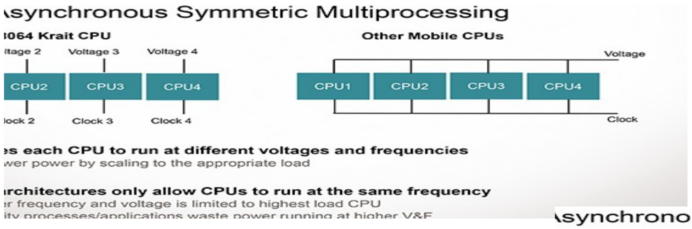
The same core frequency
and core voltage
for all cores

Semi-synchronous CPU cores

Individual core frequencies
but the same core voltage
for the cores

Asynchronous CPU cores

Individual core frequencies
and core voltages
for all cores



synchronous Symmetric Multiprocessing

064 Krait CPU

Other Mobile CPUs

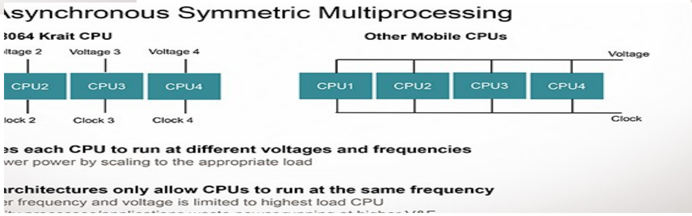
s each CPU to run at different voltages and frequencies

wer power by scaling to the appropriate load

rchitectures only allow CPUs to run at the same frequency

r frequency and voltage is limited to highest load CPU

ity processes/applications waste power running at higher V&F



synchronous Symmetric Multiprocessing

064 Krait CPU

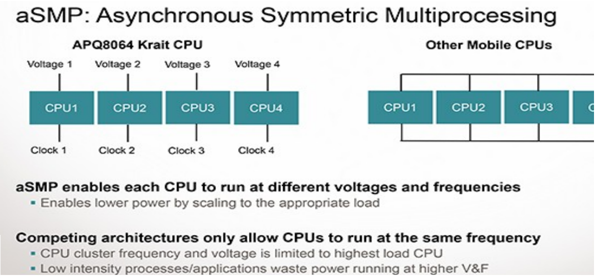
Other Mobile CPUs

s each CPU to run at different voltages and frequencies

wer power by scaling to the appropriate load

rchitectures only allow CPUs to run at the same frequency

r frequency and voltage is limited to highest load CPU



aSMP: Asynchronous Symmetric Multiprocessing

APQ8064 Krait CPU

Other Mobile CPUs

aSMP enables each CPU to run at different voltages and frequencies

- Enables lower power by scaling to the appropriate load

Competing architectures only allow CPUs to run at the same frequency

- CPU cluster frequency and voltage is limited to highest load CPU
- Low intensity processes/applications waste power running at higher V&F

Examples in mobiles

Used within clusters of big.LITTLE configurations, e.g. ARM's big.LITTLE technology (2011)

Nvidia's vSMP technology (2011)

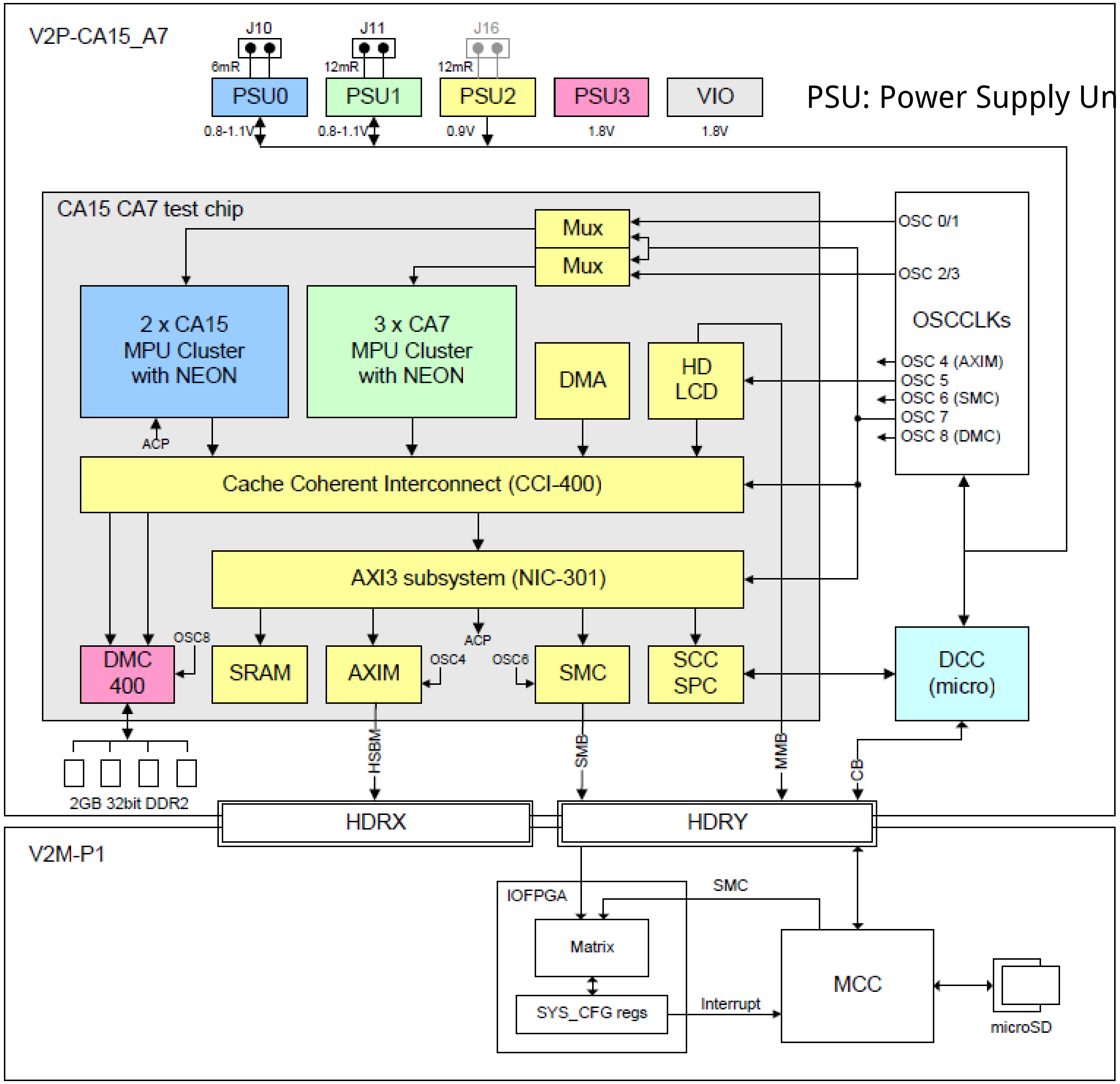
No known implementation

Typical in ARM's design in their Cortex line

Qualcomm Snapdragon family with the Scorpion and then the Krait and Kryo cores (since 2011)

1.3 Implementation of the big.LITTLE technology (17)

Example 1: Per cluster core frequencies and voltages in ARM's test chip [10]

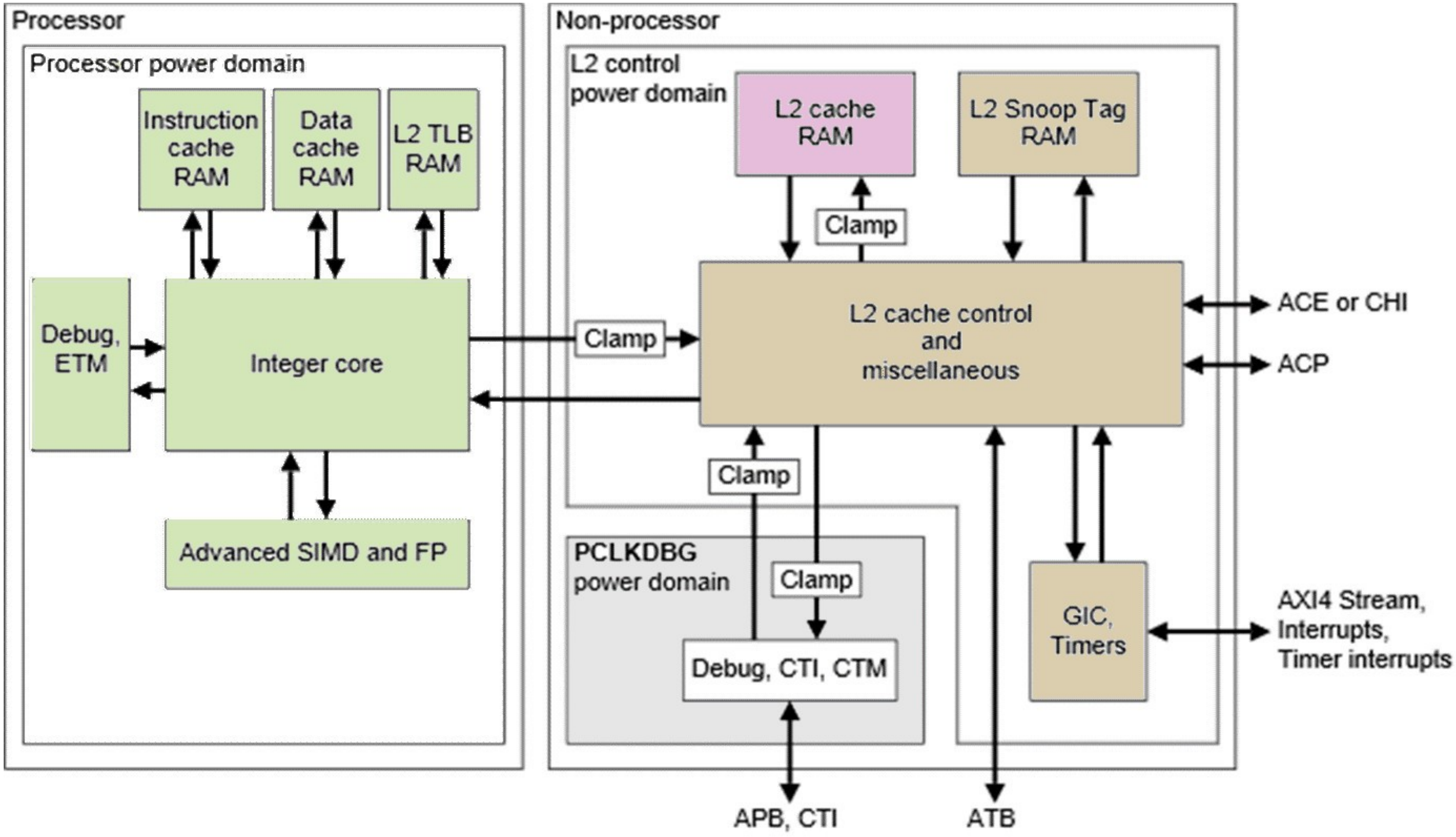


PSU: Power Supply Unit

DCC: Cortex-M3

1.3 Implementation of the big.LITTLE technology (18)

Example 2: Per core power domains in the Cortex A-57 MPcore [50]

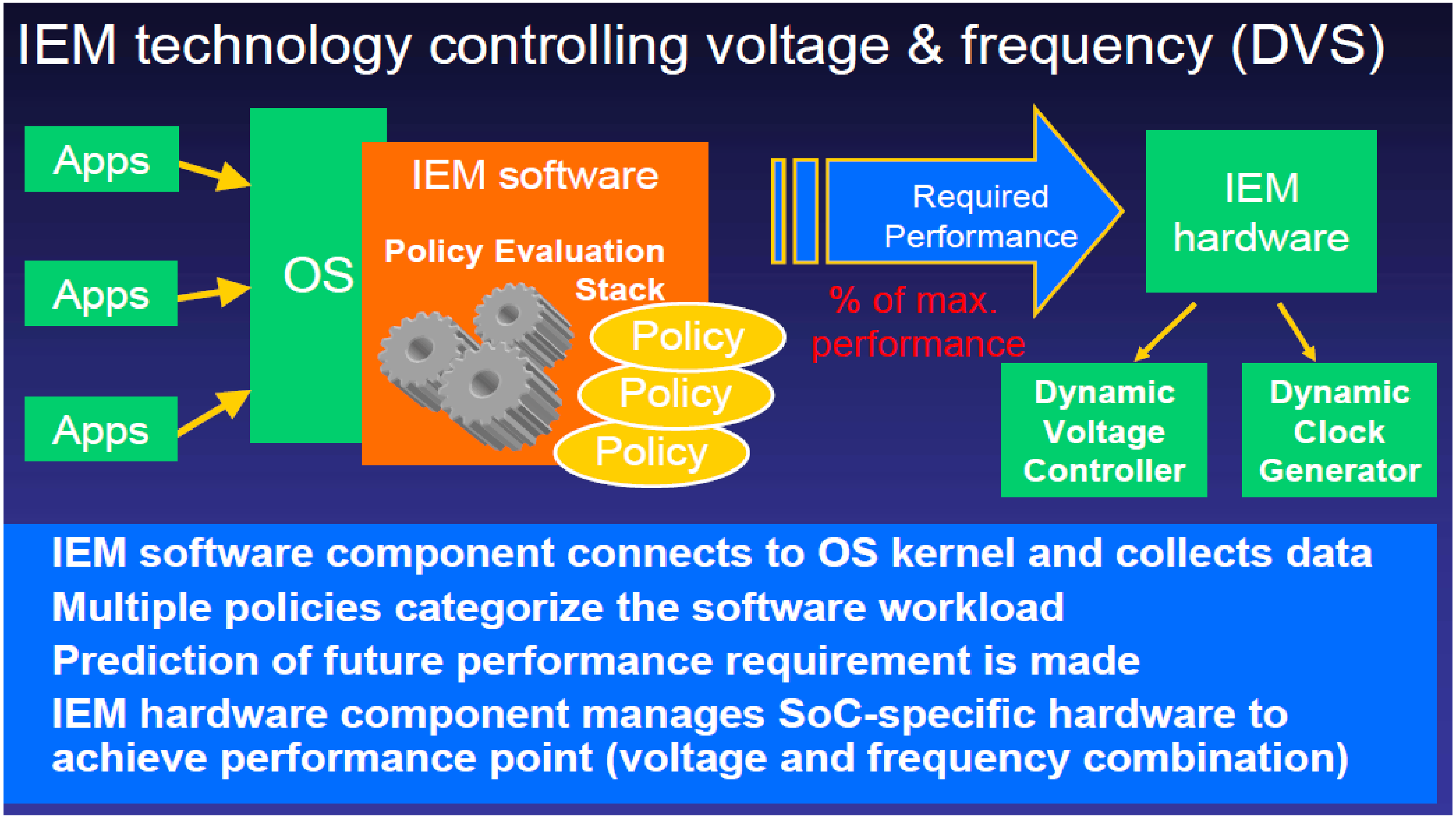


Note: Each core has a separate power domain nevertheless, actual implementations often let operate all cores of a cluster at the same frequency and voltage.

1.3 Implementation of the big.LITTLE technology (19)

Remark: Implementation of DVFS in ARM processors

- ARM introduced DVFS relatively late, about 2005 first in their ARM11 family.
- It was designed as IEM (Intelligent Energy Management) (see Figure below).



IEM technology controlling voltage & frequency (DVS)

IEM software component connects to OS kernel and collects data

Multiple policies categorize the software workload

Prediction of future performance requirement is made

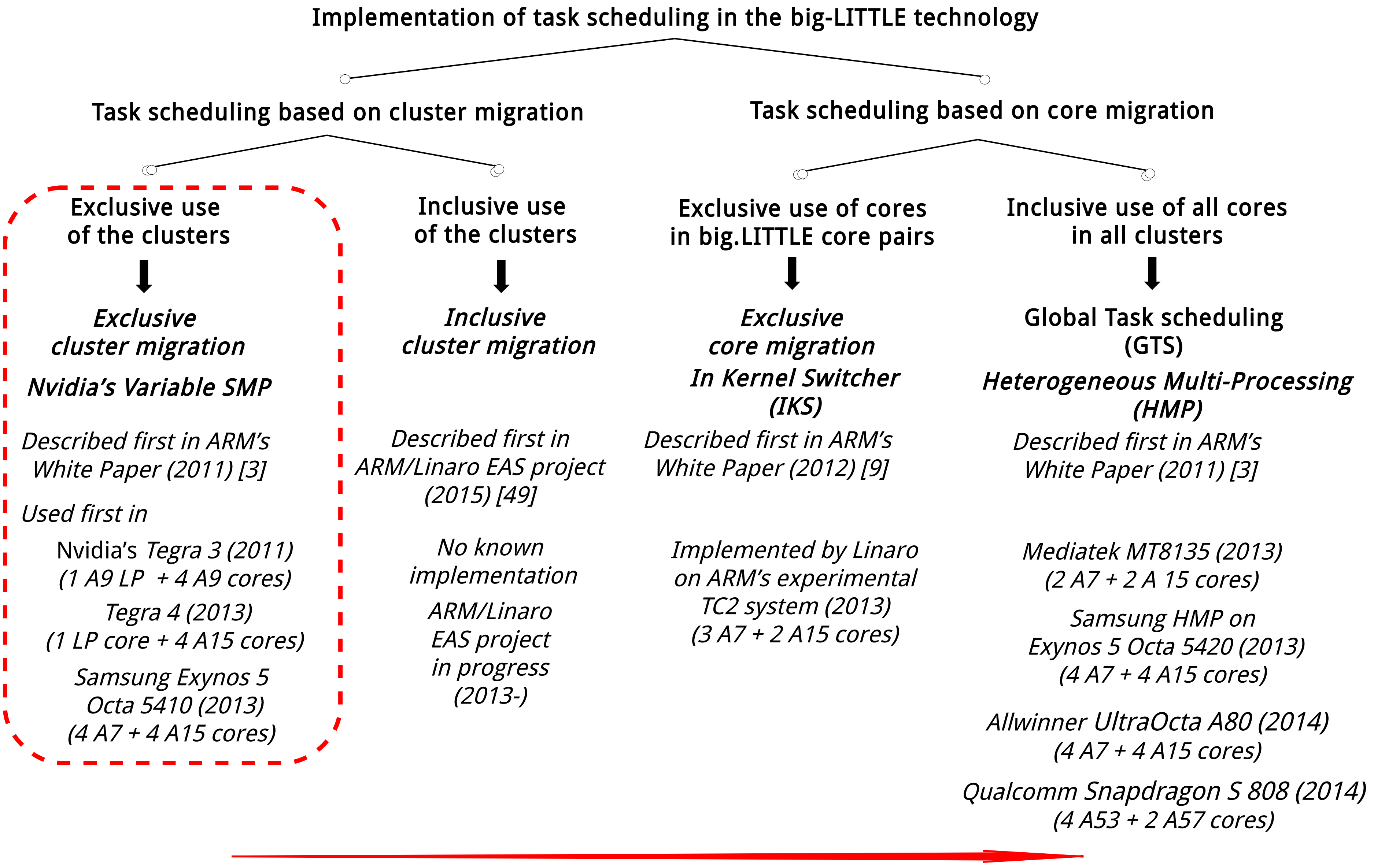
IEM hardware component manages SoC-specific hardware to achieve performance point (voltage and frequency combination)

Figure: Principle of ARM’s IEM (Intelligent Energy Management) technology [51]

2. Exclusive cluster migration

2. Exclusive cluster migration (1)

2. Exclusive cluster migration



Implementation of task scheduling in the big-LITTLE technology

Task scheduling based on cluster migration

Task scheduling based on core migration

Exclusive use of the clusters

Exclusive cluster migration

Nvidia's Variable SMP

Described first in ARM's White Paper (2011) [3]

Used first in

Nvidia's *Tegra 3 (2011)*
(1 A9 LP + 4 A9 cores)

Tegra 4 (2013)
(1 LP core + 4 A15 cores)

Samsung Exynos 5 Octa 5410 (2013)
(4 A7 + 4 A15 cores)

Inclusive use of the clusters

Inclusive cluster migration

Described first in ARM/Linaro EAS project (2015) [49]

No known implementation

ARM/Linaro EAS project in progress (2013-)

Exclusive use of cores in big.LITTLE core pairs

Exclusive core migration

In Kernel Switcher (IKS)

Described first in ARM's White Paper (2012) [9]

Implemented by Linaro on ARM's experimental TC2 system (2013)
(3 A7 + 2 A15 cores)

Inclusive use of all cores in all clusters

Global Task scheduling (GTS)

Heterogeneous Multi-Processing (HMP)

Described first in ARM's White Paper (2011) [3]

Mediatek MT8135 (2013)
(2 A7 + 2 A 15 cores)

Samsung HMP on Exynos 5 Octa 5420 (2013)
(4 A7 + 4 A15 cores)

Allwinner UltraOcta A80 (2014)
(4 A7 + 4 A15 cores)

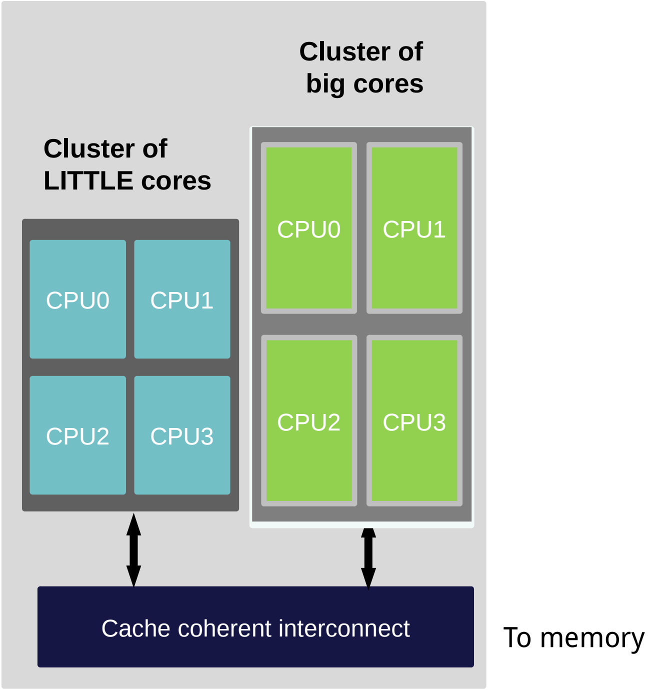
Qualcomm Snapdragon S 808 (2014)
(4 A53 + 2 A57 cores)

2. Exclusive cluster migration (2)

Principle of the exclusive cluster migration-1

- For simplicity, let’s have two clusters of cores, as usual, e.g. with 4 cores each;
  - a cluster of low power/low performance cores, termed as the LITTLE cores and
  - a cluster of high performance high power cores, termed as the big cores,

as indicated below.

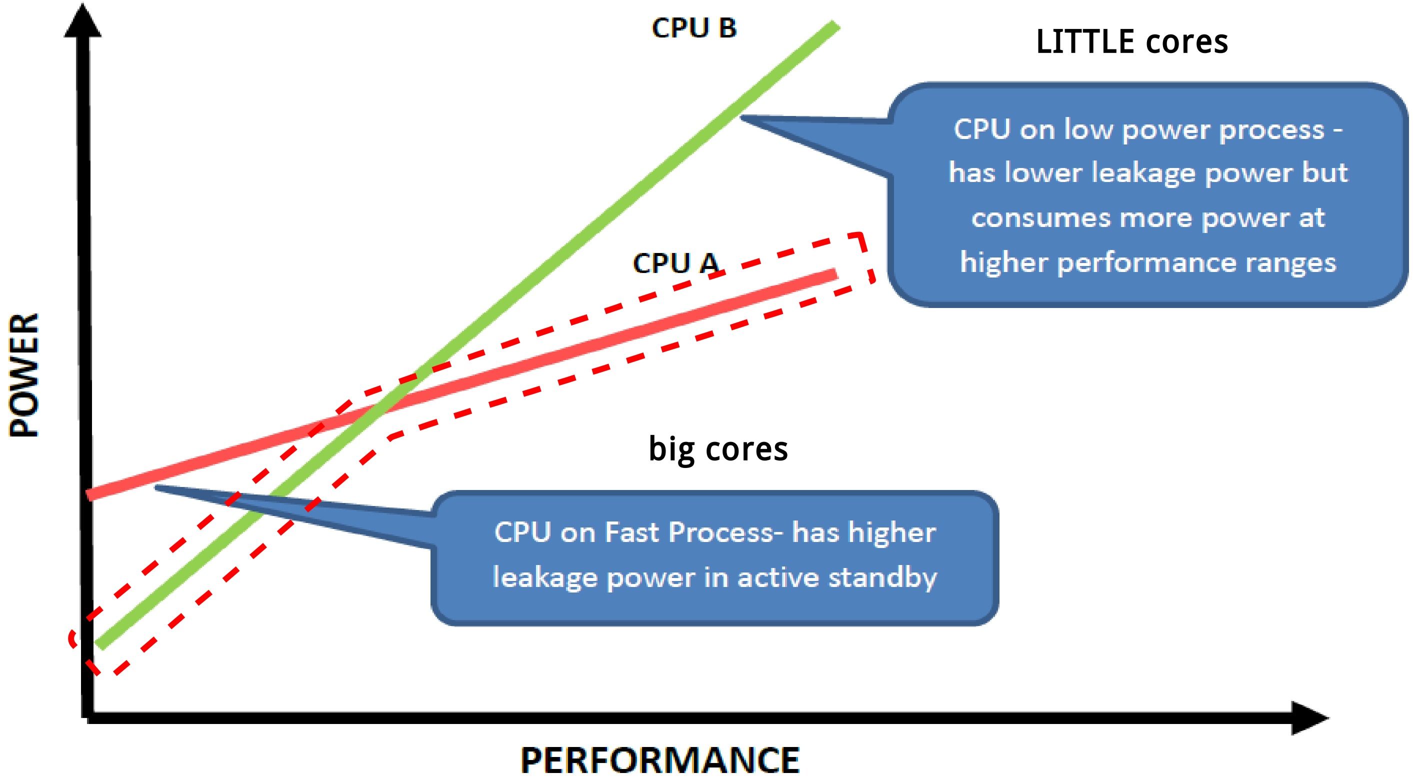


- Use the cluster of “LITTLE” cores for less demanding workloads, whereas the cluster of “big” cores for more demanding workloads, as indicated in the next Figure.

2. Exclusive cluster migration (3)

Principle of the exclusive cluster migration-2 [4]

- The OS (e.g. the Linux cpufreq routine) tracks the load for all cores in the cluster.
- As long as the actual workload can be executed by the low power, low performance cluster, this cluster will be activated.
- If however the workload requires more performance than available with the cluster of LITTLE cores (CPU A in the Figure), an appropriate routine performs a switch to the cluster of high performance high power "big" cores (CPU B).



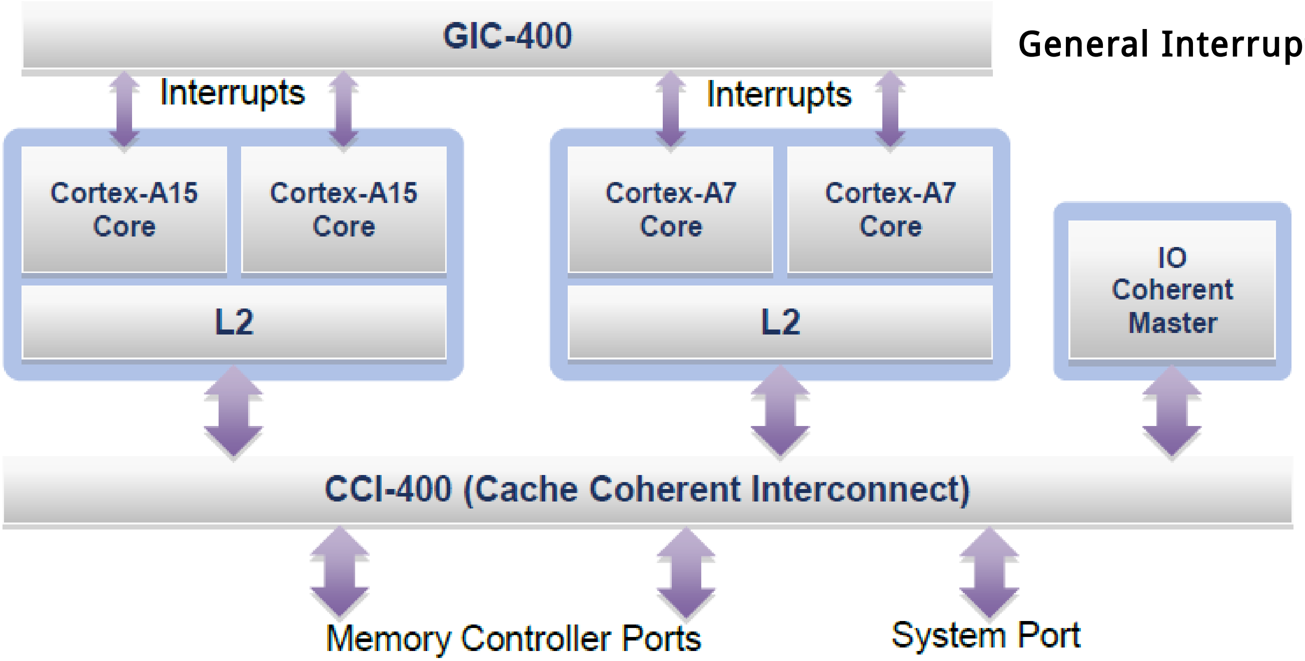
CPU on low power process - has lower leakage power but consumes more power at higher performance ranges

CPU on Fast Process- has higher leakage power in active standby

2. Exclusive cluster migration (4)

Main components of an example system

- The example system includes a cluster of two Cortex-A15 cores, used as the big cluster and another cluster of two Cortex-A7 cores, used as the LITTLE, cluster, as indicated below.



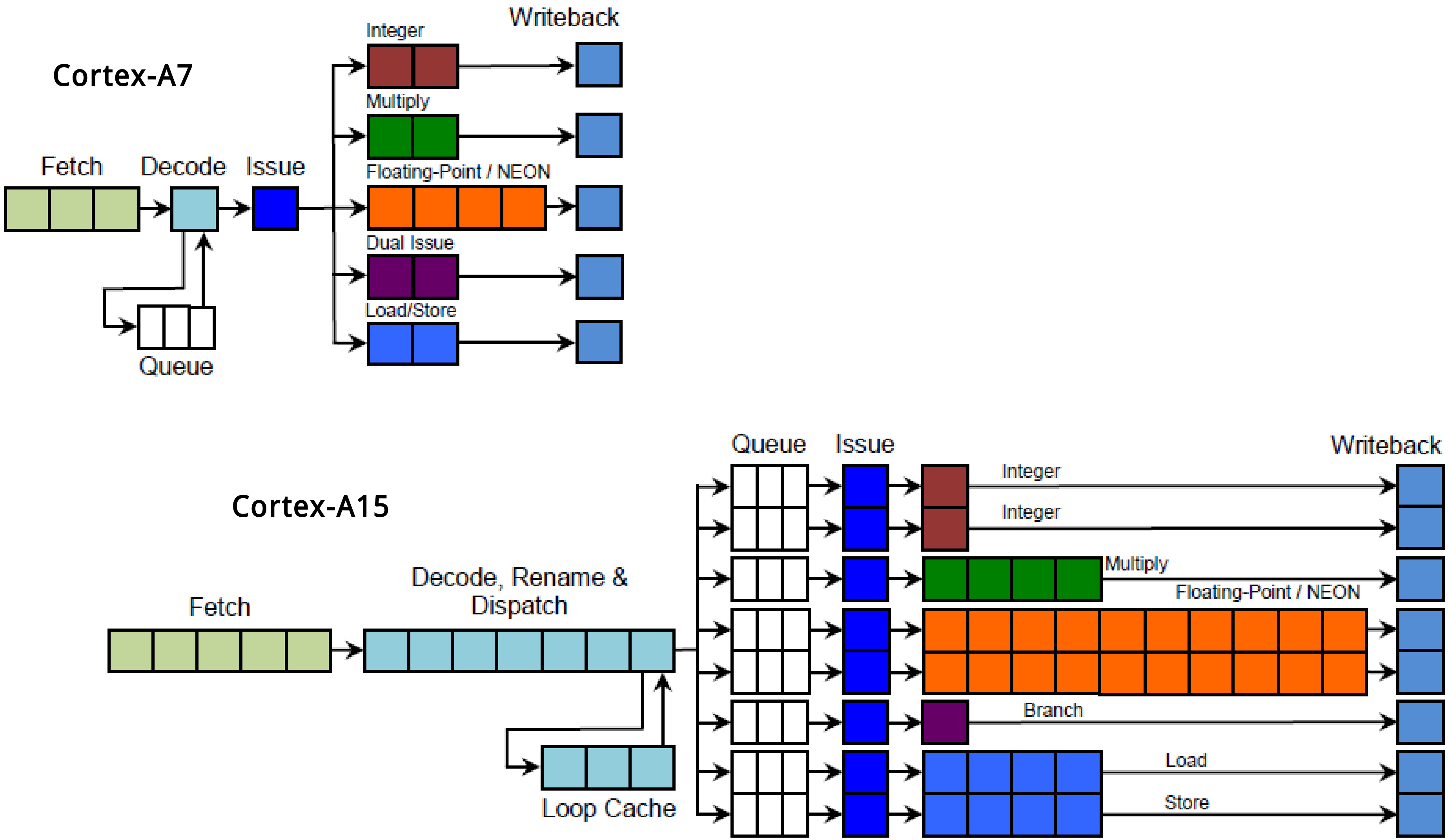
General Interrupt Controller

Figure: An example system assumed while discussing exlusive cluster migration [3]

- Both clusters are interconnected by a Cache Coherent Interconnect (CCI-400) and are served by a General Interrupt Controller (GIC-400), as shown above.

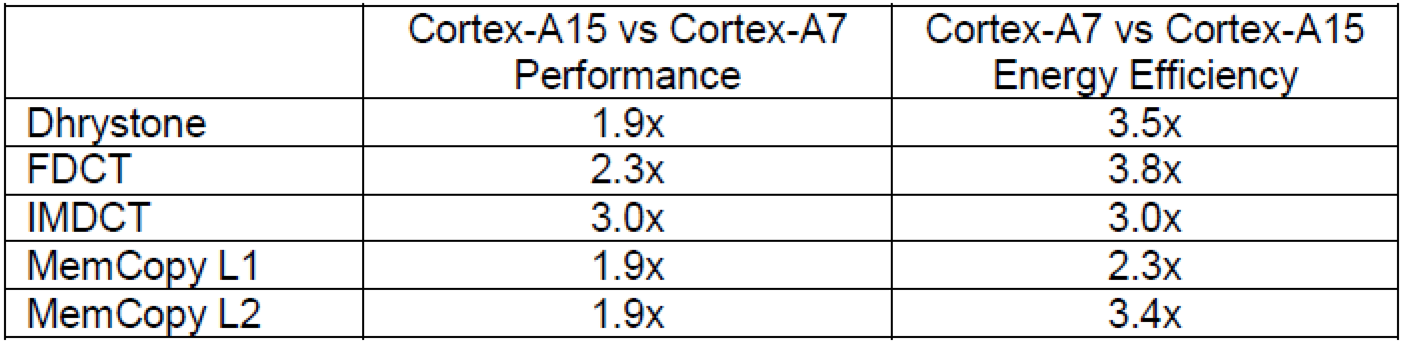
2. Exclusive cluster migration (5)

Pipelines of the “big” Cortex-15 and the “LITTLE” Cortex-A7 cores [3]



2. Exclusive cluster migration (6)

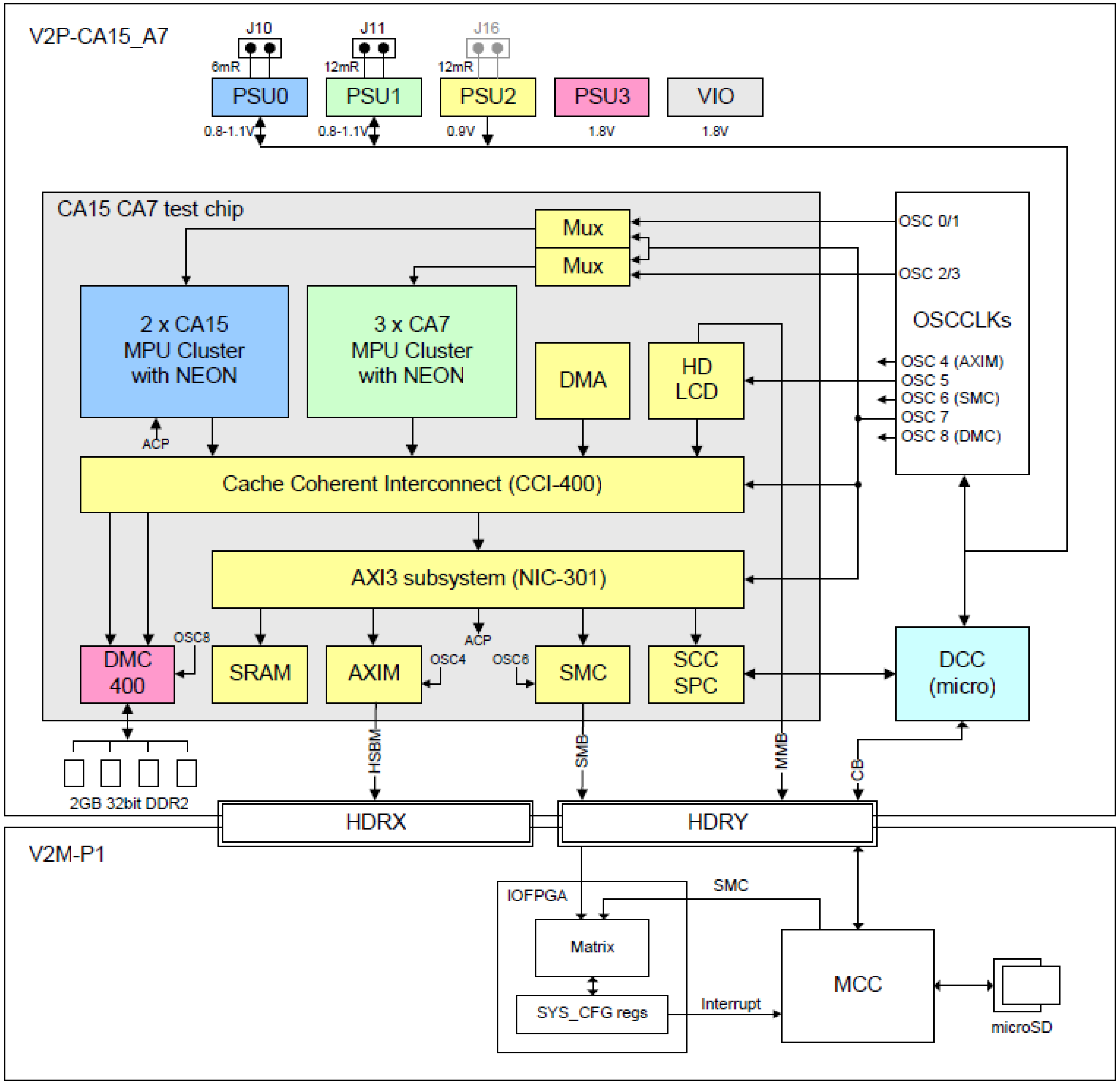
Contrasting performance and energy efficiency of the Cortex-A15 and Cortex-A7 cores [3]



	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

2. Exclusive cluster migration (7)

Voltage domains and clocking scheme of the V2P-CA15_CA7 test chip [10]



DCC: Cortex-M3

2. Exclusive cluster migration (8)

Implementation of DVFS in case of Cortex-A15/Cortex-CA7 clusters [10]

Based on the referred Application Note, it can be assumed that both the Cortex-A15 and Cortex-A7 processor cores have global DVFS (Dynamic Voltage and Frequency Scaling), i.e. all cores of a cluster run at the same scalable operating point, i.e. at the same but variable clock frequency and core voltage.

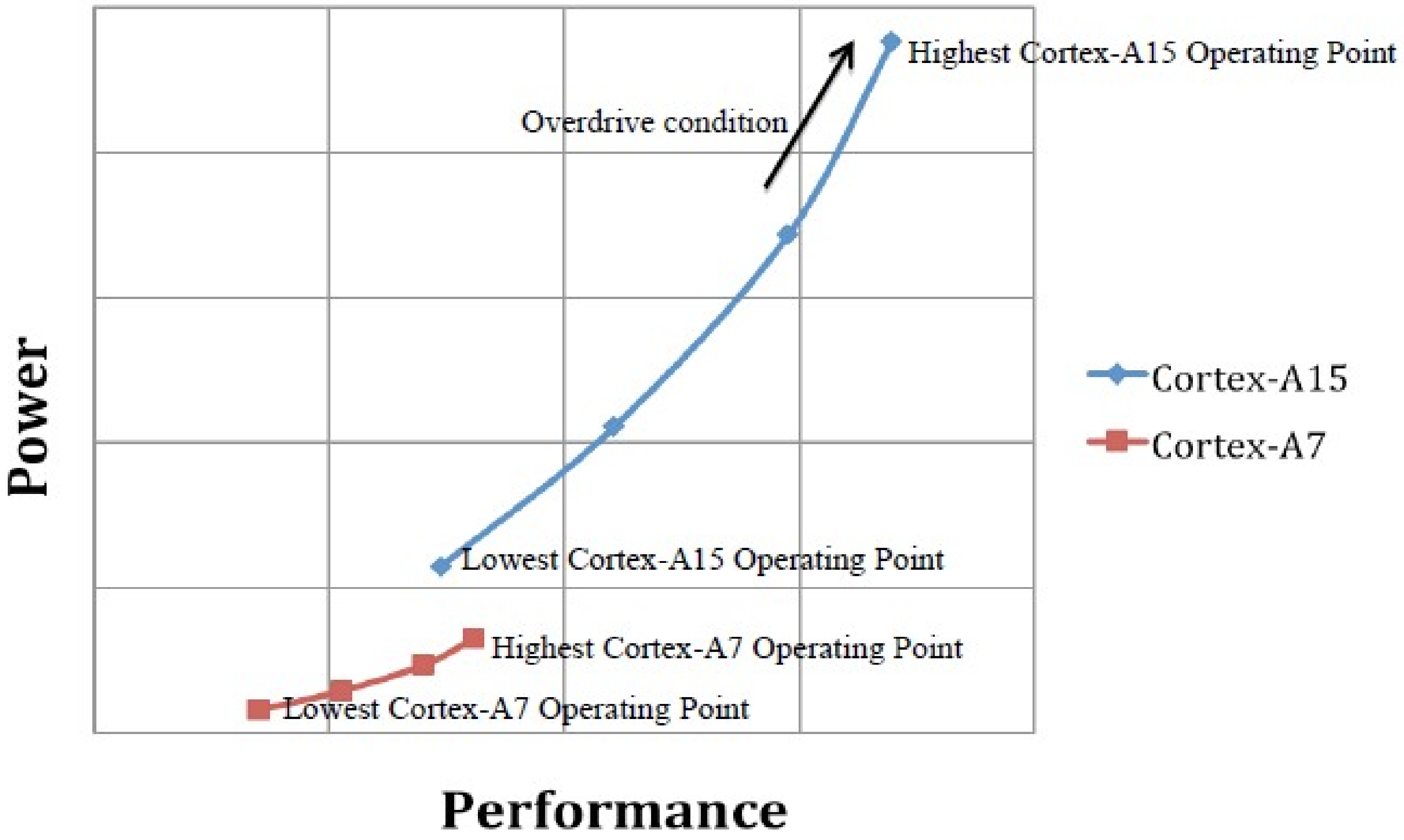
2. Exclusive cluster migration (9)

Running applications with a given set of operating points [3]

- The operating points of both clusters are used as a continuum during DVFS.
- While the Cortex-A7 cluster is active the OS can tune the operating points as it would do in case of a single application processor.
- Once the Cortex-A7 cluster is at its highest operating point and the OS requires more performance, a task migration will be invoked to the Cortex-A15 cluster.
- In this way, low and medium intensity applications will run with high energy on the Cortex-A7 cluster whereas high intensity applications will be executed on the high performance Cortex-A15 cluster.
- While the Cortex-A15 cluster is active the OS can tune the operating points as it would do in case of a single application processor.

2. Exclusive cluster migration (10)

Operating points of the Cortex-A15 and Cortex-A7 cores [3]

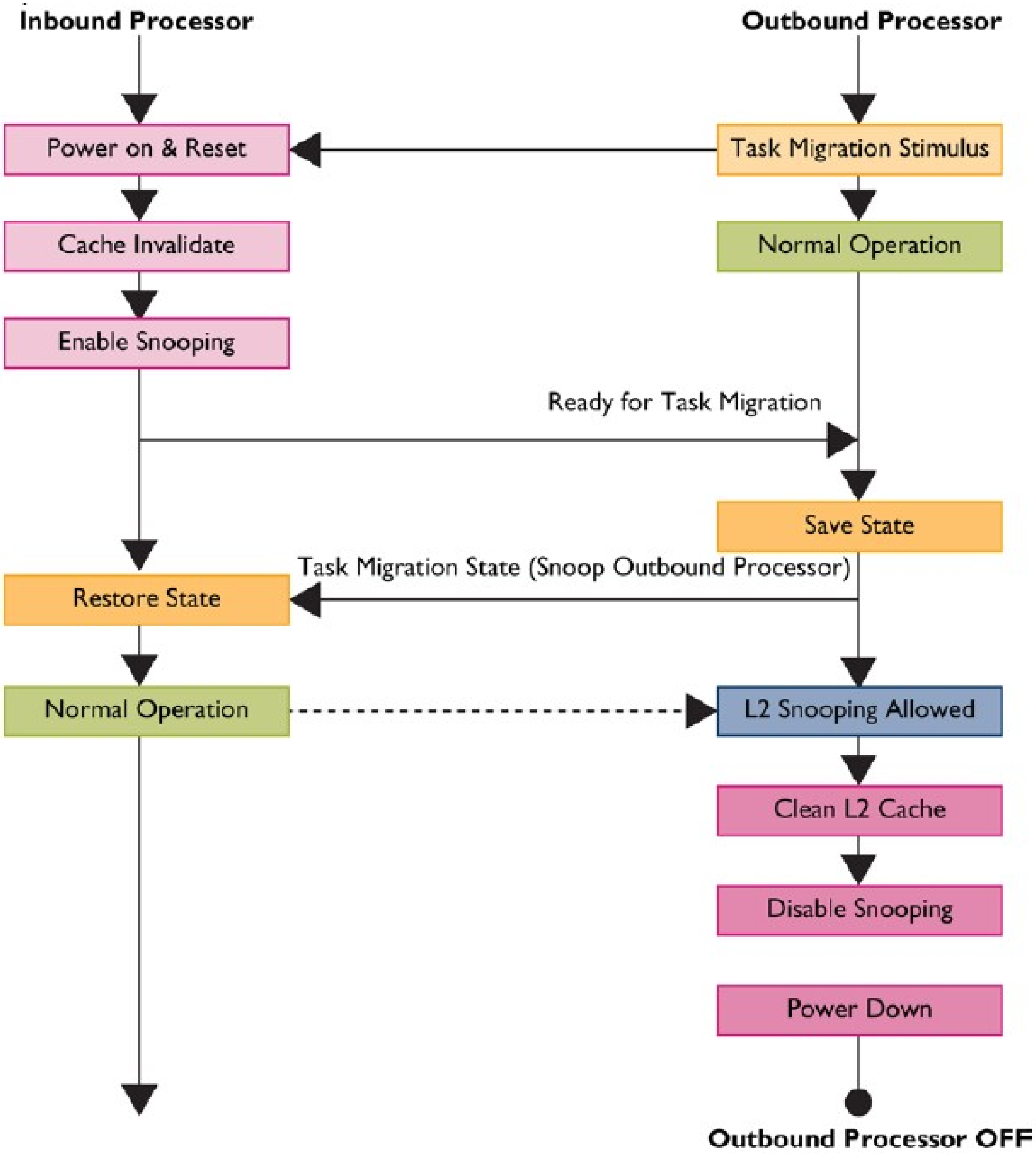


2. Exclusive cluster migration (11)

The process of cluster switching [3]

Currently inactive cluster

Currently active cluster



2. Exclusive cluster migration (12)

The time needed for task migration [3]

According to ARM [3] the time needed to migrate tasks between the two clusters is about 20 μs, i.e. about 20 000 cycles while the processor operates at 1 GHz.

2. Exclusive cluster migration (13)

Implementation example: Samsung Exynos Octa 5410 [11]

- It is the world's first octa core processor.
- Announced in 11/2012, launched in some Galaxy S4 models in 4/2013.

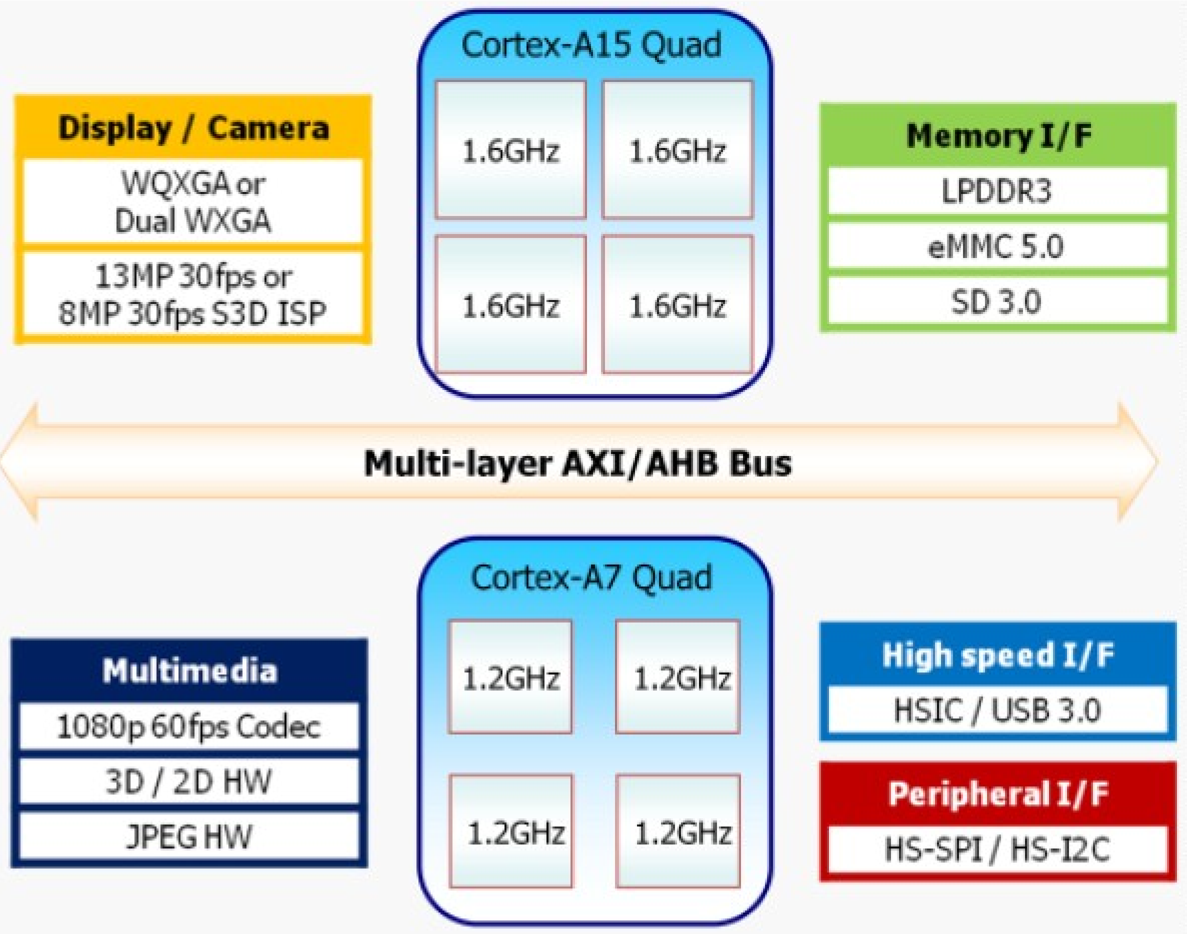
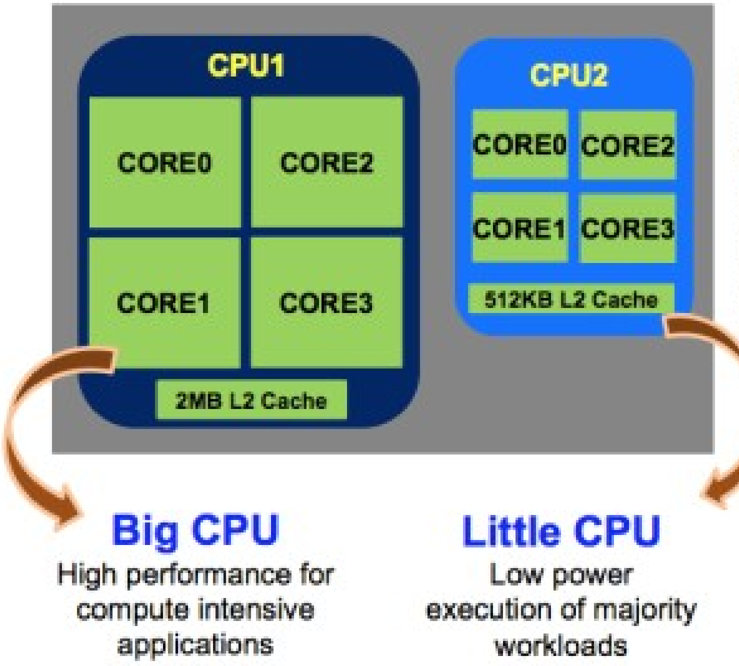


Figure: Block diagram of Samsung's Exynos 5 Octa 5410 [11]

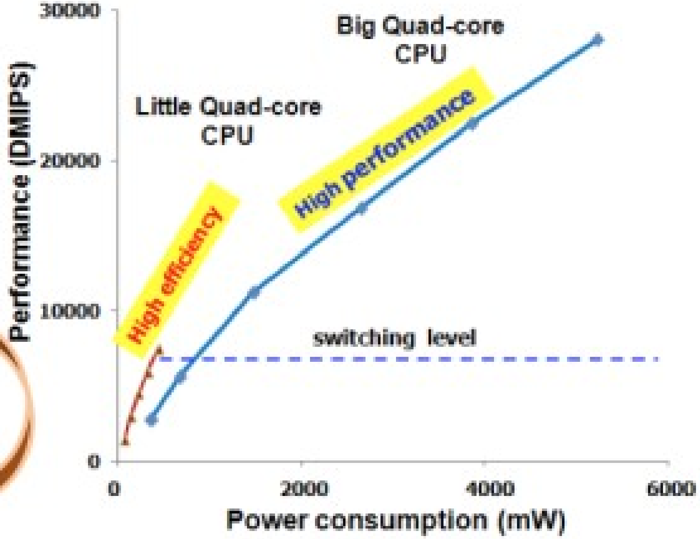
2. Exclusive cluster migration (14)

Operation of the Exynos 5 Octa 5410 using exclusive cluster switching [12]



Big CPU
High performance for compute intensive applications

Little CPU
Low power execution of majority workloads



It was revealed at the International Solid-State Circuit Conference (ISSCC) in 2/2013, without specifying the chip designation.

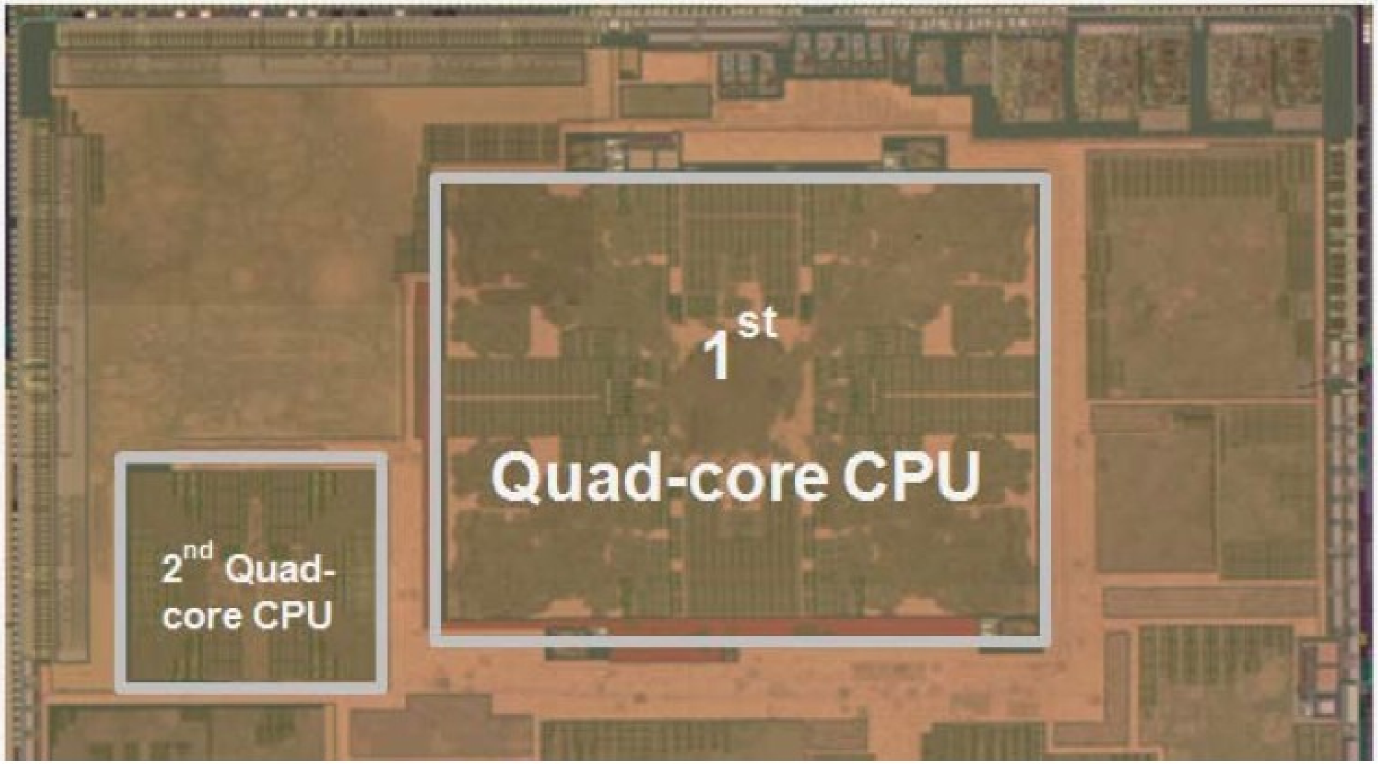
2. Exclusive cluster migration (15)

Remark

- According to sources there was a troublesome bug in the CCI-400 coherent bus interface [13].
- Thus, Samsung disabled the coherency between the two clusters, and as a consequence after cluster switches they need to invalidate all caches.
- Obviously, this has impeded performance and battery life.

2. Exclusive cluster migration (16)

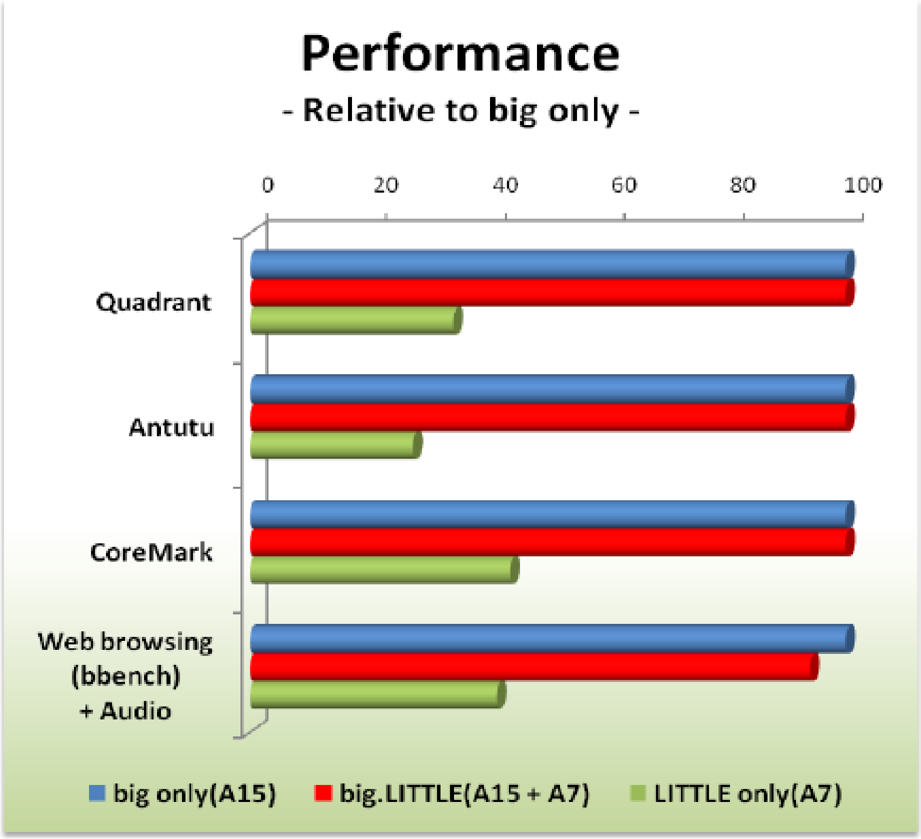
Assumed die photo of Samsung's Exynos 5 Octa 5410 [12]

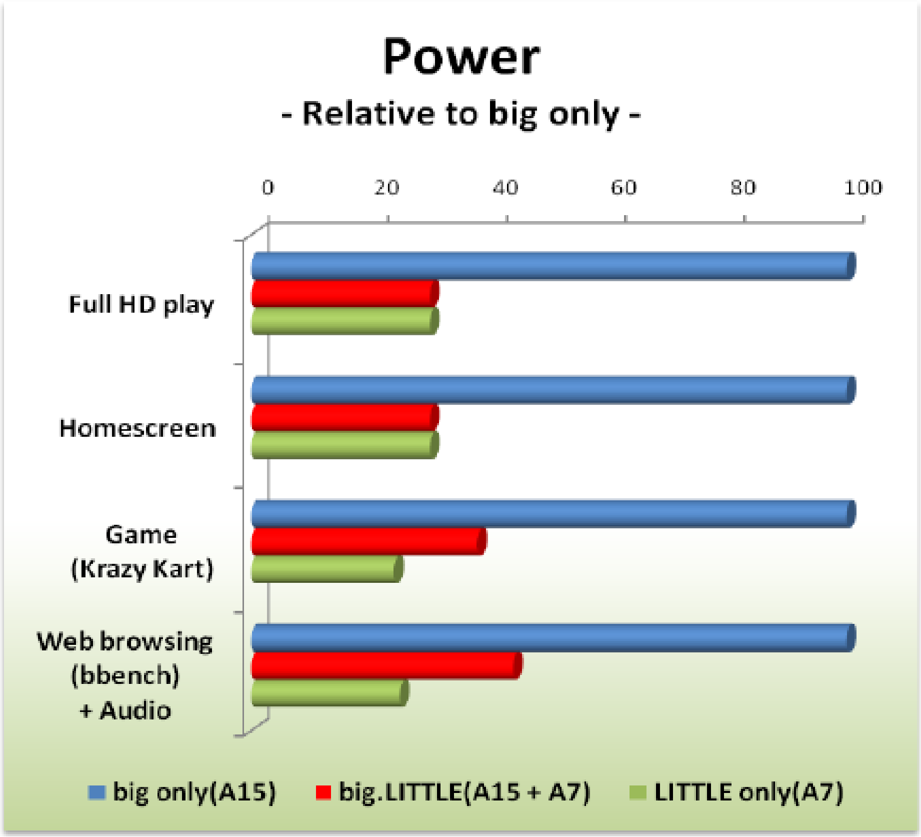


It was revealed at the International Solid-State Circuit Conference (ISSCC) in 2/2013 without specifying the chip designation.

2. Exclusive cluster migration (17)

Performance and power results of the Exynos 5 Octa 5410 [11]





2. Exclusive cluster migration (18)

Nvidia's variable SMP

- Nvidia preferred to implement exclusive cluster migration with a "LITTLE" cluster including only a single core, and a "big" cluster with four cores, as indicated in the next Figure.

E.g.

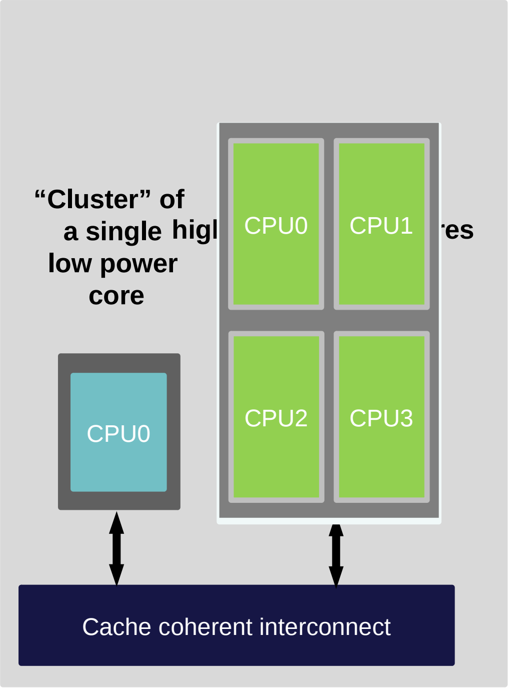
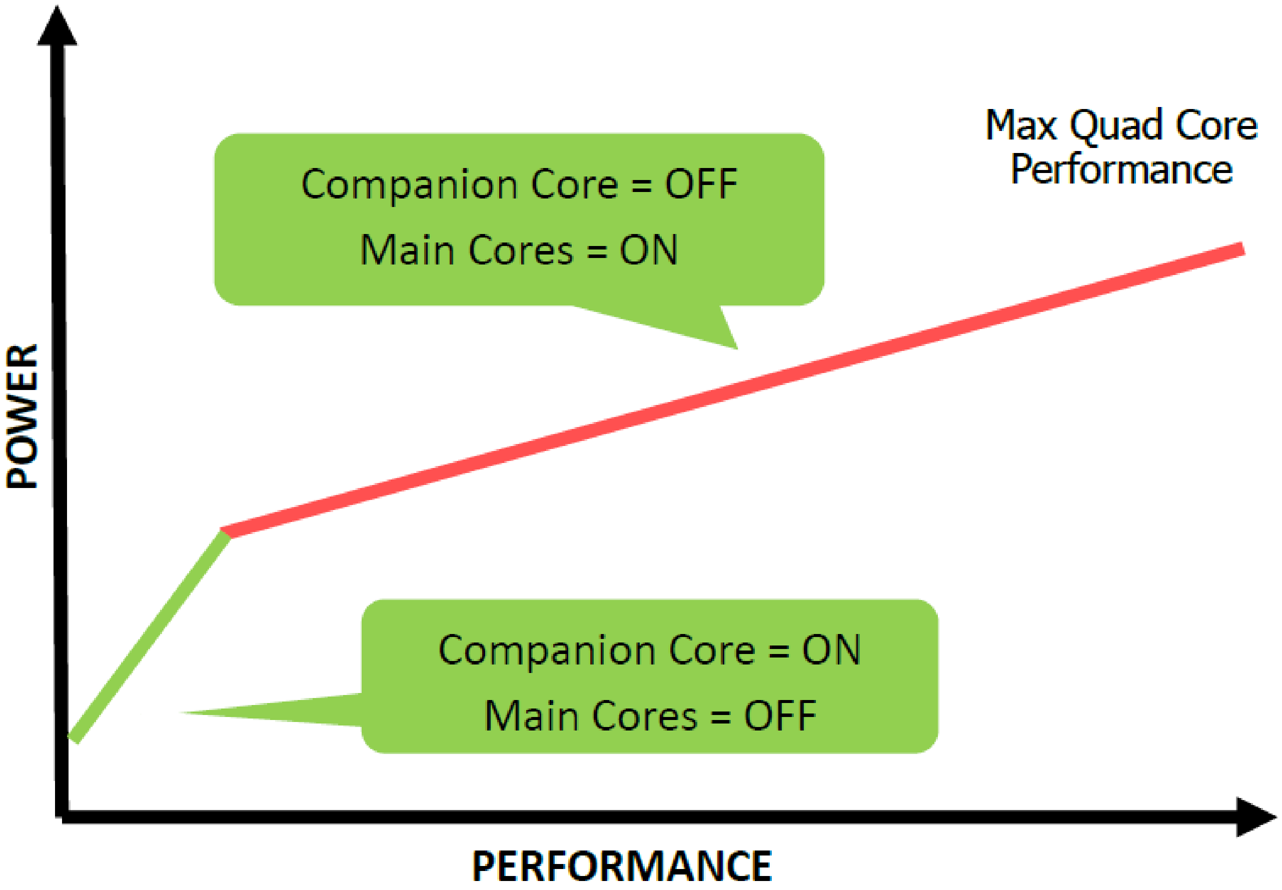


Figure: Example layout of Nvidia's variable SMP

- Nvidia designates this implementation of big-LITTLE technology as variable SMP.
- It was implemented early in the Tegra 3 (2011) with one A9 LP + 4 A9 cores and subsequently in the Tegra 4 (2013) with one LP core + 4 A15 cores.

2. Exclusive cluster migration (19)

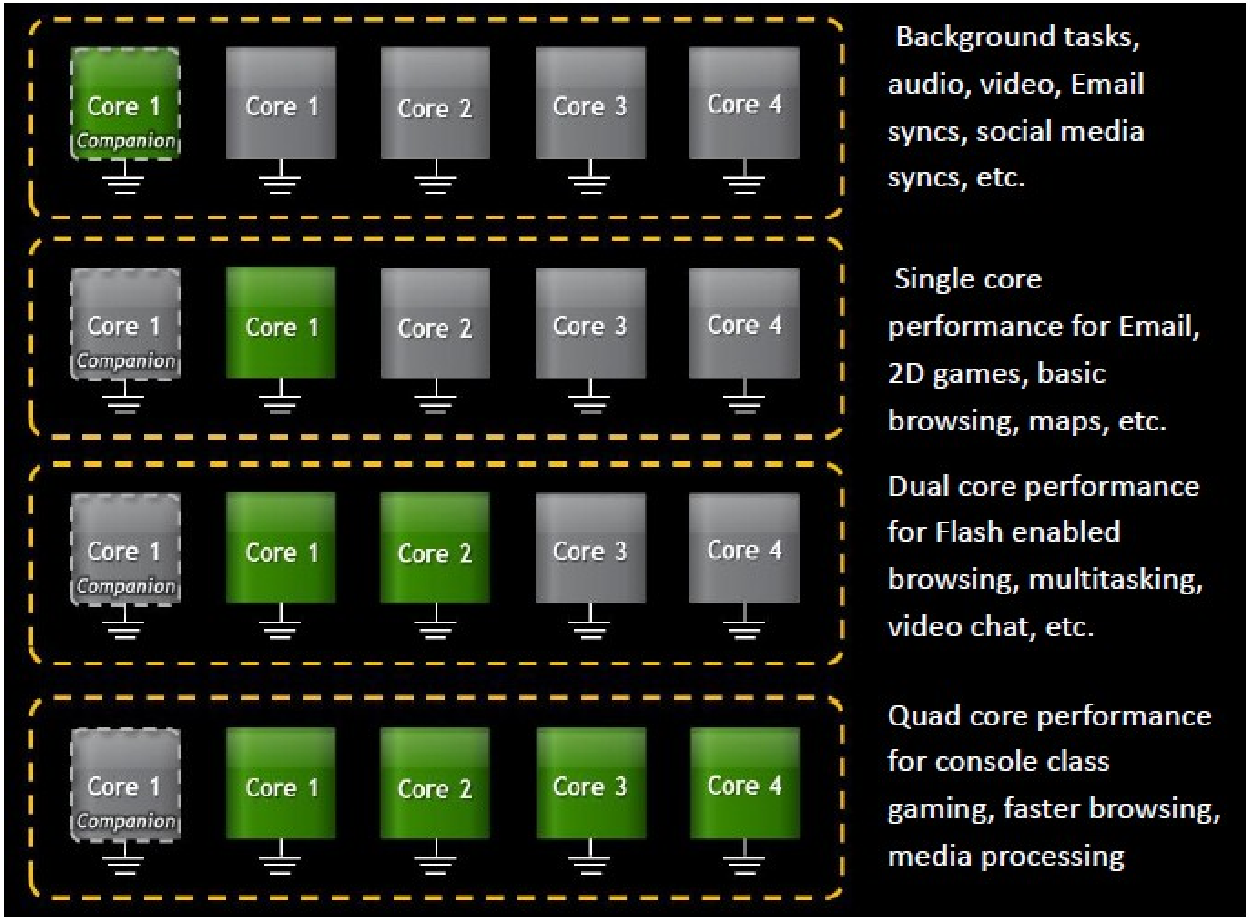
Power-Performance curve of Nvidia's variable SMP [4]



Note that in the Figure the "LITTLE" core is designated as "Companion core" whereas the "big" cores as "Main cores".

2. Exclusive cluster migration (20)

Illustration of the operation of *Nvidia's Variable SMP* [4]



Background tasks, audio, video, Email syncs, social media syncs, etc.

Single core performance for Email, 2D games, basic browsing, maps, etc.

Dual core performance for Flash enabled browsing, multitasking, video chat, etc.

Quad core performance for console class gaming, faster browsing, media processing

Implemented in the Tegra 3 (2011) and Tegra 4 (2013).

2. Exclusive cluster migration (21)

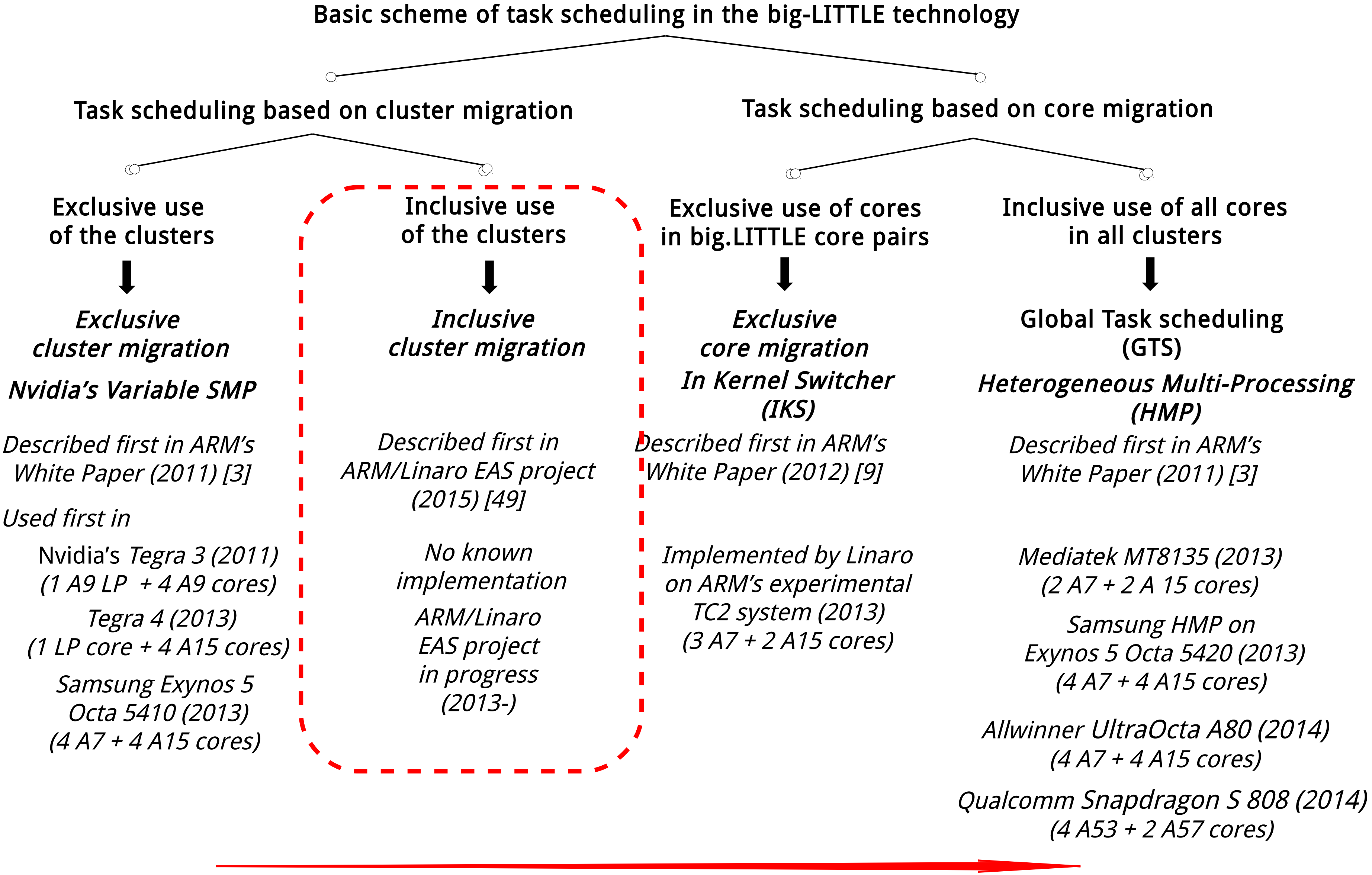
Remark [52]

The cluster switch approach was the default scheduler in Android 4.2.2.

3. Inclusive cluster migration

3. Inclusive cluster migration (1)

3. Inclusive cluster migration



Basic scheme of task scheduling in the big-LITTLE technology

Task scheduling based on cluster migration

Task scheduling based on core migration

Exclusive use of the clusters

Exclusive cluster migration

Nvidia's Variable SMP

Described first in ARM's White Paper (2011) [3]

Used first in

Nvidia's *Tegra 3 (2011)*
(1 A9 LP + 4 A9 cores)

Tegra 4 (2013)
(1 LP core + 4 A15 cores)

Samsung Exynos 5 Octa 5410 (2013)
(4 A7 + 4 A15 cores)

Inclusive use of the clusters

Inclusive cluster migration

Described first in ARM/Linaro EAS project (2015) [49]

No known implementation

ARM/Linaro EAS project in progress (2013-)

Exclusive use of cores in big.LITTLE core pairs

Exclusive core migration

In Kernel Switcher (IKS)

Described first in ARM's White Paper (2012) [9]

Implemented by Linaro on ARM's experimental TC2 system (2013)
(3 A7 + 2 A15 cores)

Inclusive use of all cores in all clusters

Global Task scheduling (GTS)

Heterogeneous Multi-Processing (HMP)

Described first in ARM's White Paper (2011) [3]

Mediatek MT8135 (2013)
(2 A7 + 2 A 15 cores)

Samsung HMP on Exynos 5 Octa 5420 (2013)
(4 A7 + 4 A15 cores)

Allwinner UltraOcta A80 (2014)
(4 A7 + 4 A15 cores)

Qualcomm Snapdragon S 808 (2014)
(4 A53 + 2 A57 cores)

3. Inclusive cluster migration (2)

3. Inclusive cluster migration

- Until now no commercial implementation became known using inclusive core migration.
- Nevertheless, since 2013 ARM and Linaro pursue a development project, called Energy Aware Scheduling (EAS) to provide a generic energy model based scheduling approach that supports a broad range of current and future CPU topologies, including SMP, multi-cluster SMP (e.g. 8-core Cortex-A53 products), as well as traditional ARM big.LITTLE configurations.
- EAS assumes a platform with inclusive cluster migration, as indicated in the next Figure.

3. Inclusive cluster migration (3)

Assumed platform for EAS (Energy Aware Scheduling) [49]

The assumed platform would have the following voltage and frequency domains:

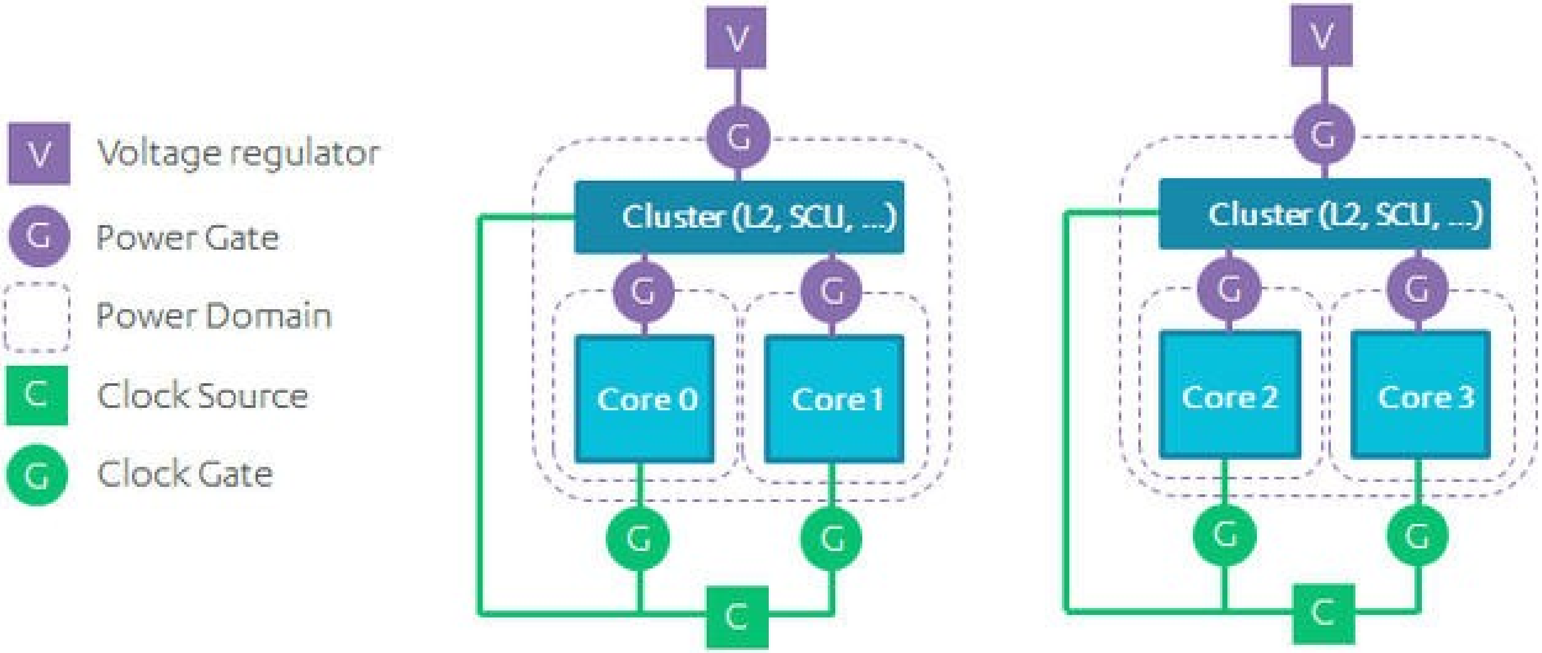


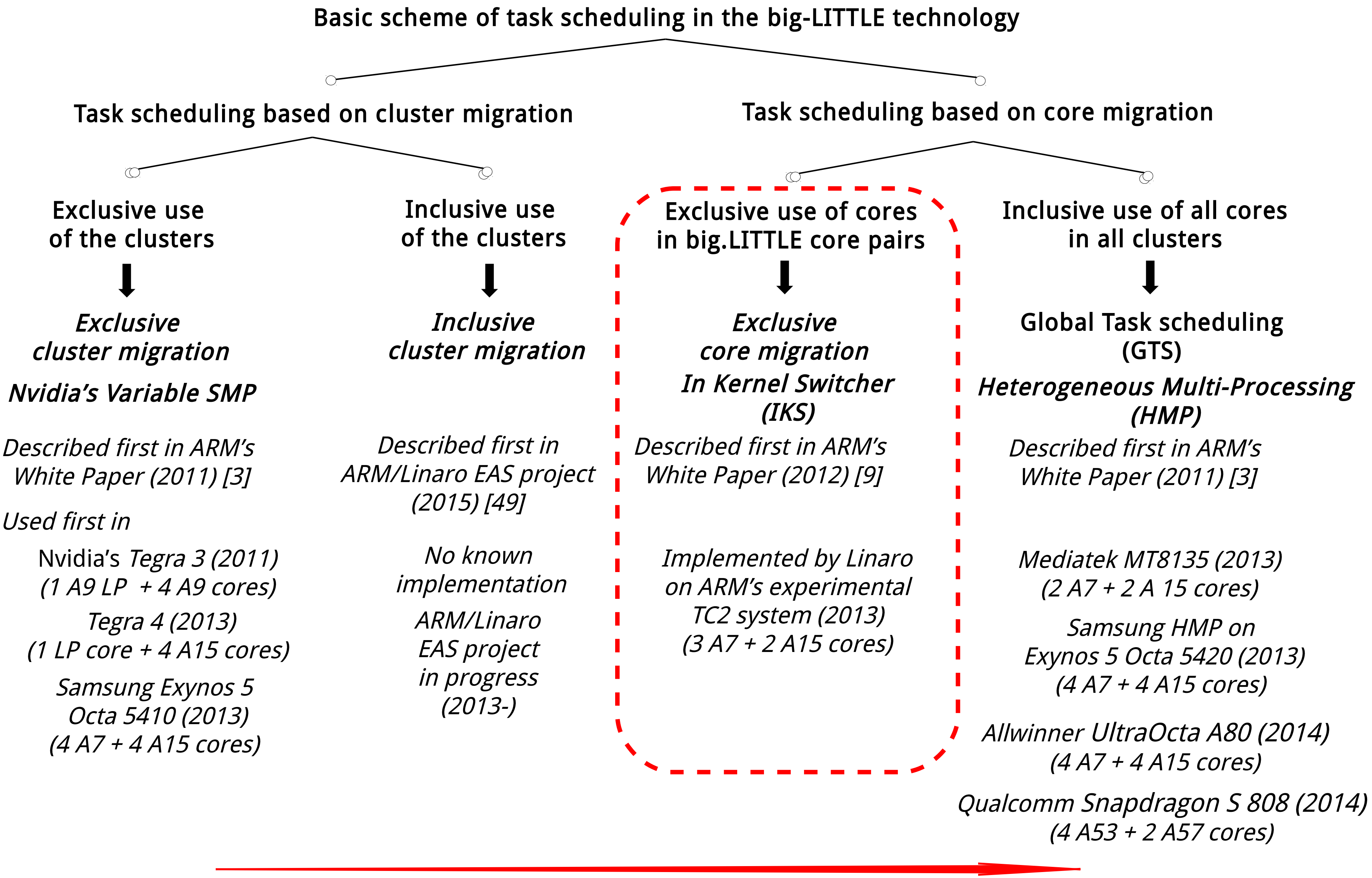
Figure: Assumed plaform for EAS (Energy Aware Scheduling)

- Ideally, each cluster will operate at its own separate independent frequency and voltage.
- By lowering the voltage and frequency, there is a substantial power saving.
- This allows the per-cluster power/performance to be accurately controlled, and tailored to the workload being executed.

4. Exclusive core migration

4. Exclusive core migration (1)

4. Exclusive core migration



Basic scheme of task scheduling in the big-LITTLE technology

Task scheduling based on cluster migration

Exclusive use of the clusters

Exclusive cluster migration

Nvidia's Variable SMP

Described first in ARM's White Paper (2011) [3]

Used first in

Nvidia's *Tegra 3 (2011)*
(1 A9 LP + 4 A9 cores)

Tegra 4 (2013)
(1 LP core + 4 A15 cores)

Samsung Exynos 5 Octa 5410 (2013)
(4 A7 + 4 A15 cores)

Inclusive use of the clusters

Inclusive cluster migration

Described first in ARM/Linaro EAS project (2015) [49]

No known implementation

ARM/Linaro EAS project in progress (2013-)

Task scheduling based on core migration

Exclusive use of cores in big.LITTLE core pairs

Exclusive core migration

In Kernel Switcher (IKS)

Described first in ARM's White Paper (2012) [9]

Implemented by Linaro on ARM's experimental TC2 system (2013)
(3 A7 + 2 A15 cores)

Inclusive use of all cores in all clusters

Global Task scheduling (GTS)

Heterogeneous Multi-Processing (HMP)

Described first in ARM's White Paper (2011) [3]

Mediatek MT8135 (2013)
(2 A7 + 2 A 15 cores)

Samsung HMP on Exynos 5 Octa 5420 (2013)
(4 A7 + 4 A15 cores)

Allwinner UltraOcta A80 (2014)
(4 A7 + 4 A15 cores)

Qualcomm Snapdragon S 808 (2014)
(4 A53 + 2 A57 cores)

4. Exclusive core migration (2)

Principle of exclusive core migration-1

- Linaro developed a model for task scheduling on big.LITTLE SOCs, called IKS (In Kernel Switcher) and designed an appropriate Linux kernel patch (LSK 3.10 (Linaro Stable Kernel) for an experimental system.
- IKS builds core pairs from the cores of the big and LITTLE core clusters, e.g. from Cortex-A15 and Cortex-A7 cores, and treats each core pair, consisting of a big and a LITTLE core, as a single virtual core, as indicated in the next Figure.

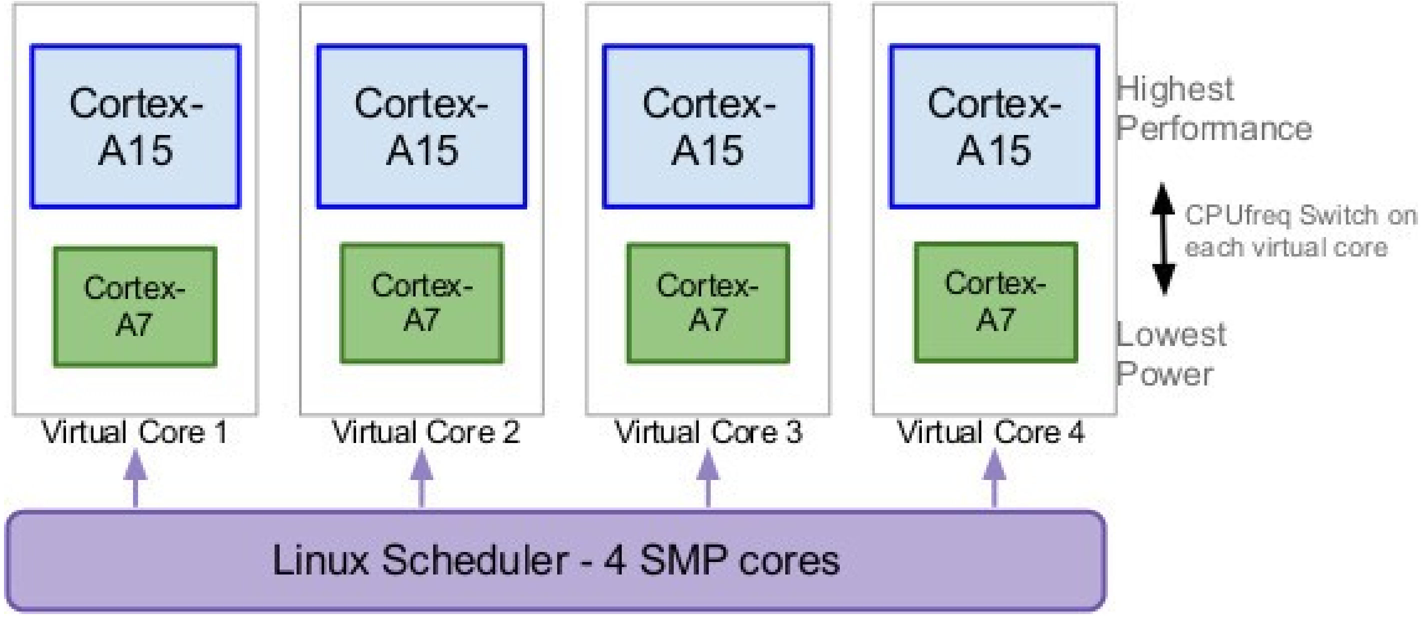


Figure: Virtual cores of a 4x Cortex-A15 and 4x Cortex-A7 big.LITTLE SOC [15]

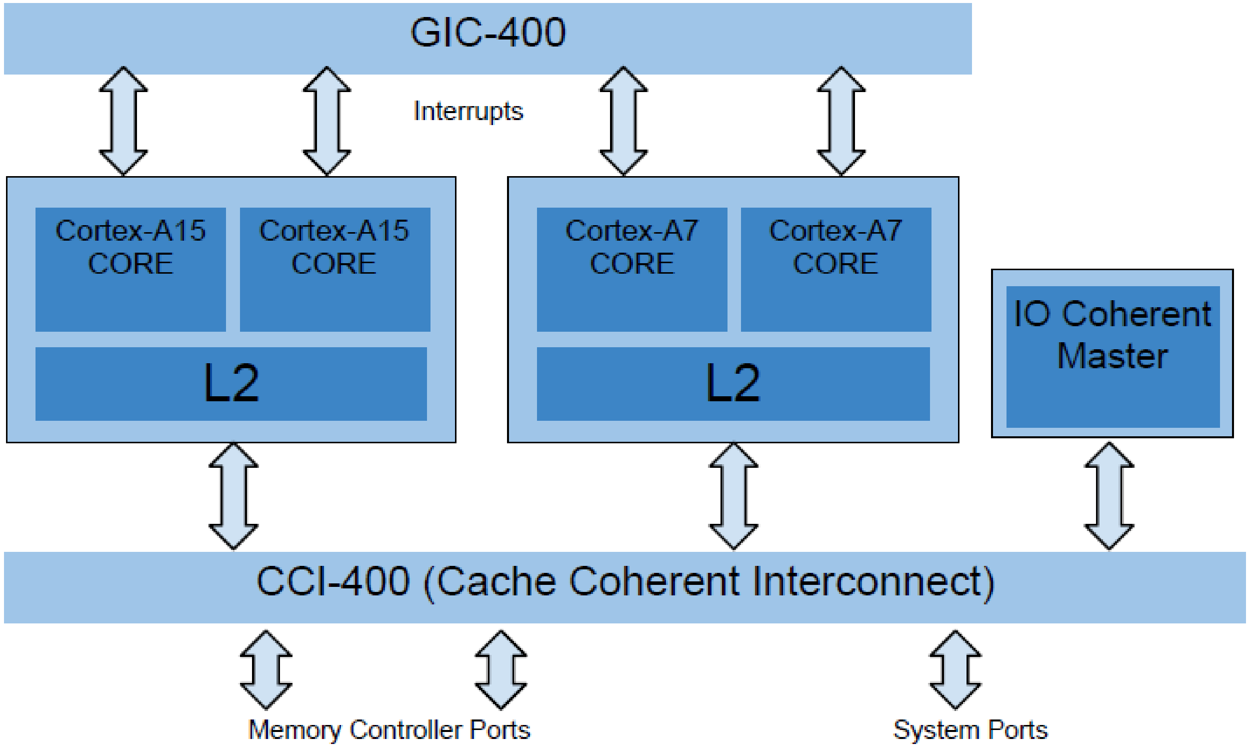
4. Exclusive core migration (3)

Principle of exclusive core migration-2

- The Linux kernel schedules then the tasks to the virtual cores.
- For each core pair the cpufreq driver of the Linux kernel controls whether the LITLE core (for low power) or the big core (for maximum performance) should be activated.
- An important feature of IKS is that it relies on existing Linux kernel scheduling mechanisms and thus is easy to implement.

4. Exclusive core migration (4)

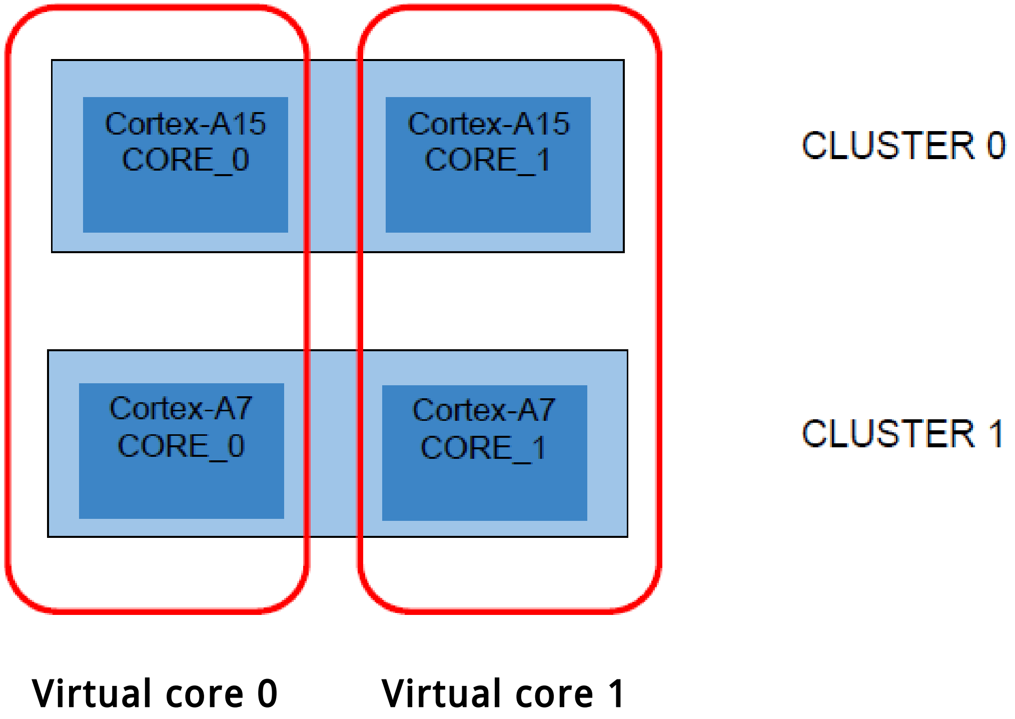
Experimental implementation of IKS on a 2x Cortex-A15 and 2x Cortex-A7 big.LITTLE configuration [16]



* Picture by ARM LTD.

4. Exclusive core migration (5)

Virtual cores of the experimental implementation of IKS on a 2x Cortex-A15 and 2x Cortex-A7 big.LITTLE configuration [16]



4. Exclusive core migration (6)

Operating points of the virtual cores-1 [16]

The Cortex-A15 and Cortex-A7 SOCs have originally the following operating points:

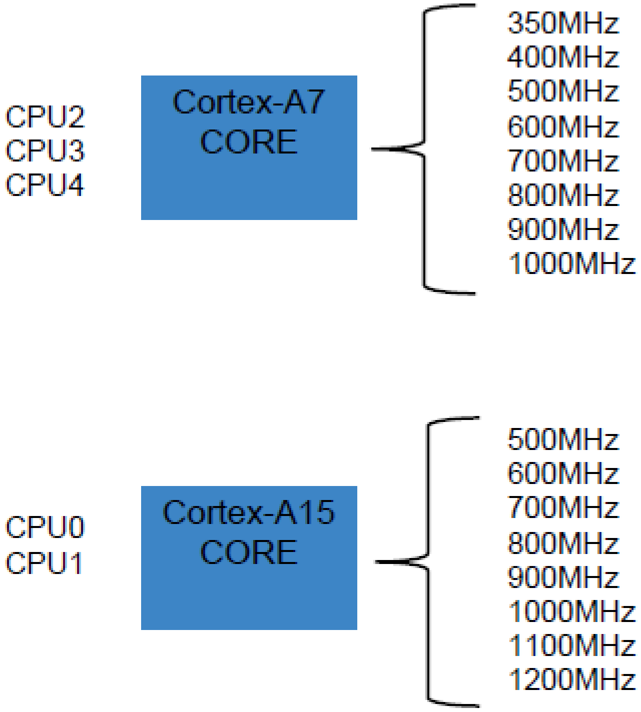
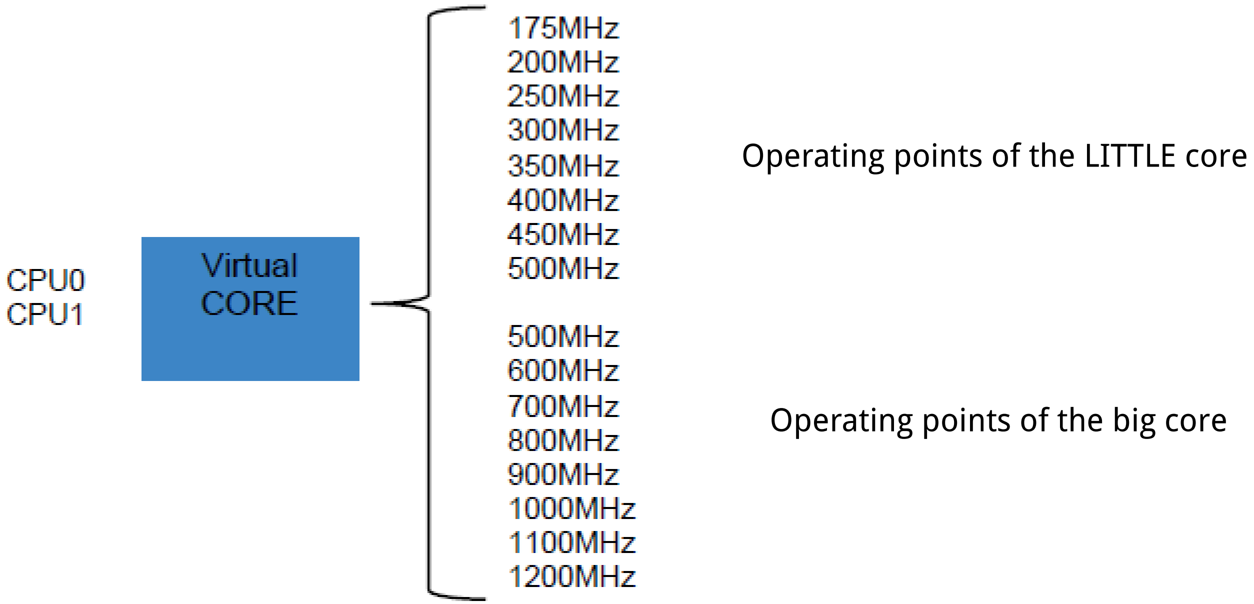


Figure: Operation points of the Cortex-A15 and Cortex-A7 SOCs [16]

4. Exclusive core migration (7)

Operating points of the virtual cores-2 [16]

For a seamless continuation of the operating points of both SOCs the original operating points of the Cortex-A7 will be modified, actually halved, during the initialization of the IKS, as shown below.



CPU0
CPU1

Virtual CORE

175MHz
200MHz
250MHz
300MHz
350MHz
400MHz
450MHz
500MHz

Operating points of the LITTLE core

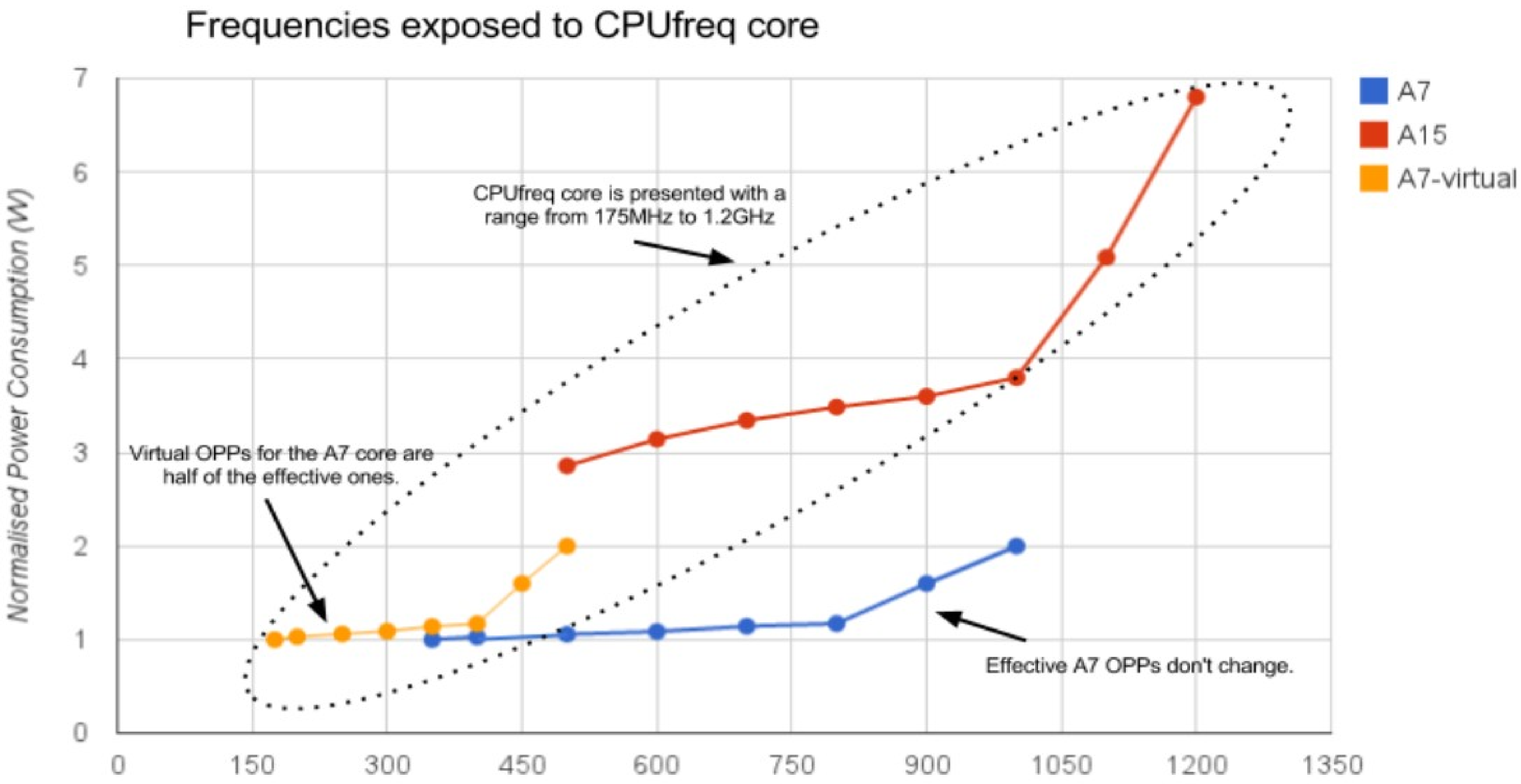
500MHz
600MHz
700MHz
800MHz
900MHz
1000MHz
1100MHz
1200MHz

Operating points of the big core

4. Exclusive core migration (8)

Operating points of the virtual cores-3 [16]

As a result the Linux kernel sees the following operating points of the virtual cores:



4. Exclusive core migration (9)

Switching between big and LITTLE cores [16]

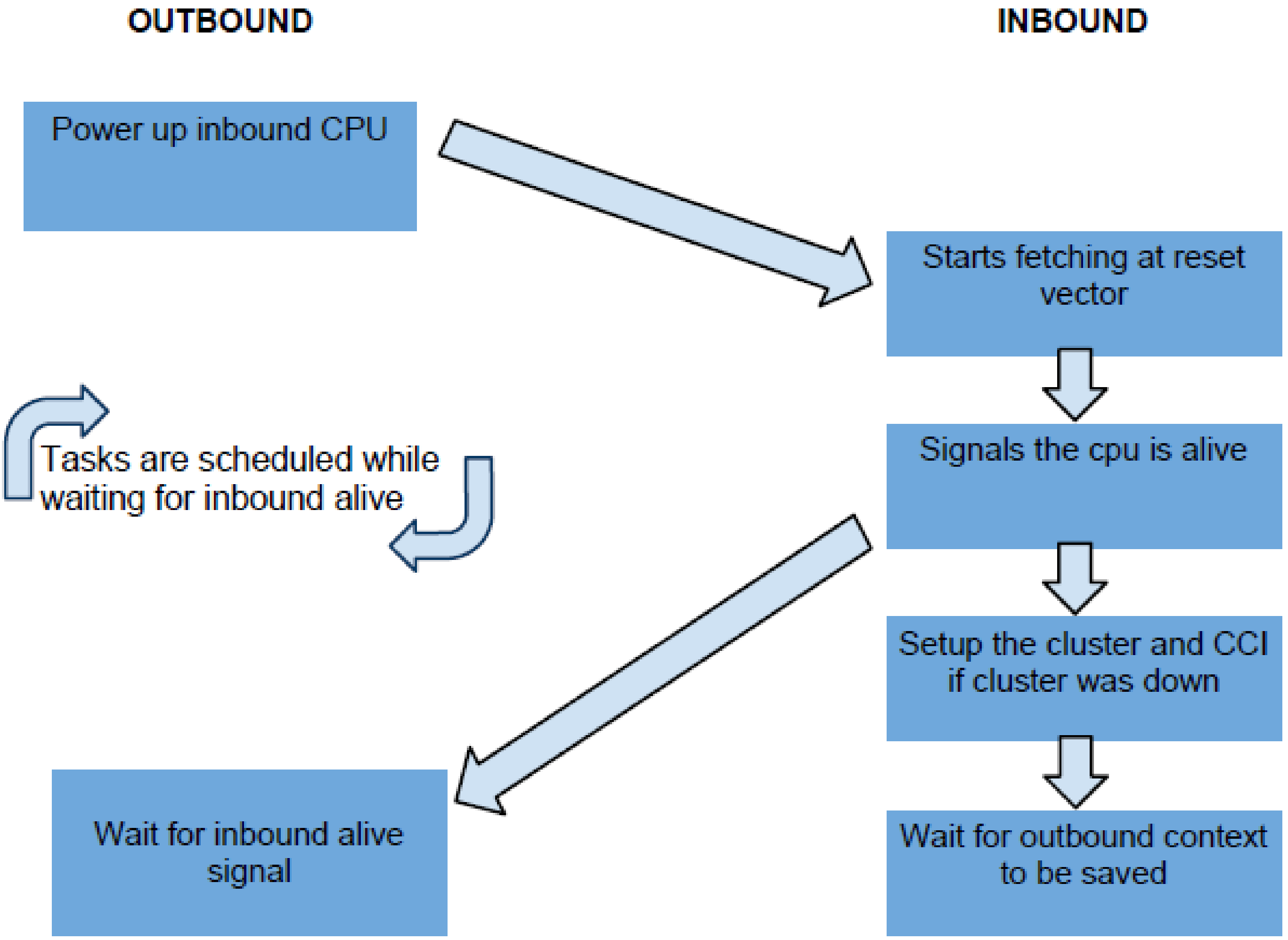
- Let's suppose that at a given time a virtual core runs at 200 MHz.

  Then the LITTLE core is active and its big core pair (A15) is shut down.
- When subsequently, the frequency governor requests a core frequency of, let say 1000 MHz, this request cannot be satisfied by the LITTLE core (A7) and the CPUfreq driver of the kernel instructs the Switcher to perform a core switch from the LITTLE core to the big core.
- The switching process is detailed in the next three Figures.

  Here we note that in the Figures the currently active core is termed as the "outbound" core whereas the target core as the "inbound" core.

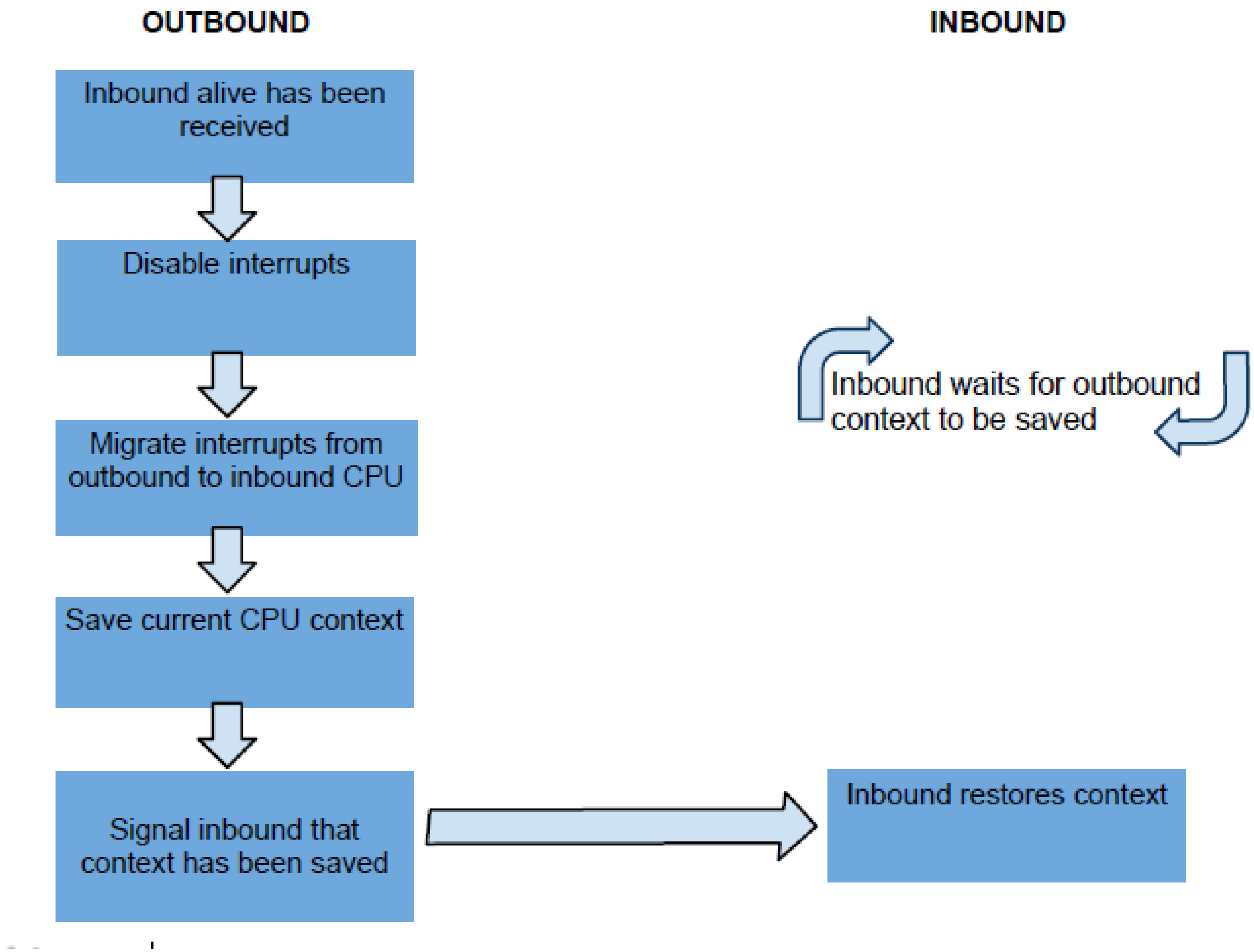
4. Exclusive core migration (10)

The core switching process-1 [16]


OUTBOUND
INBOUND
Power up inbound CPU
Starts fetching at reset vector
Tasks are scheduled while waiting for inbound alive
Signals the cpu is alive
Setup the cluster and CCI if cluster was down
Wait for inbound alive signal
Wait for outbound context to be saved

4. Exclusive core migration (11)

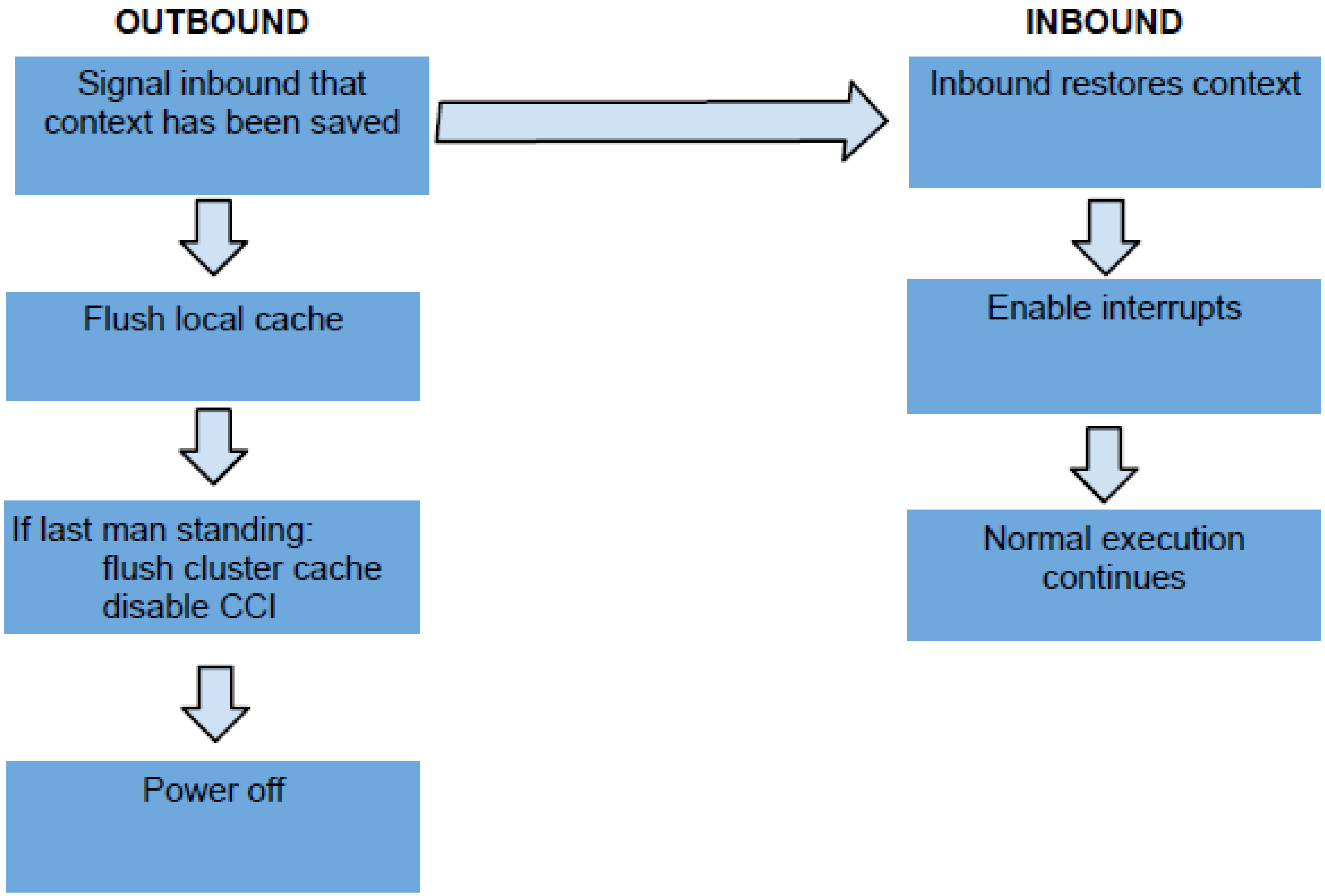
The core switching process-2 [16]



OUTBOUND
INBOUND
Inbound alive has been received
Disable interrupts
Migrate interrupts from outbound to inbound CPU
Save current CPU context
Signal inbound that context has been saved
Inbound waits for outbound context to be saved
Inbound restores context

4. Exclusive core migration (12)

The core switching process-3 [16]



OUTBOUND

Signal inbound that context has been saved

Flush local cache

If last man standing:
flush cluster cache
disable CCI

Power off

INBOUND

Inbound restores context

Enable interrupts

Normal execution continues

4. Exclusive core migration (13)

Measured results of IKS-1 [16]

- Performance/power results of the experimental IKS system are shown below.
- The data contrasts performance/power values of IKS (implemented in three configurations) with a system including only two Cortex-A15 or two Cortex-A7.

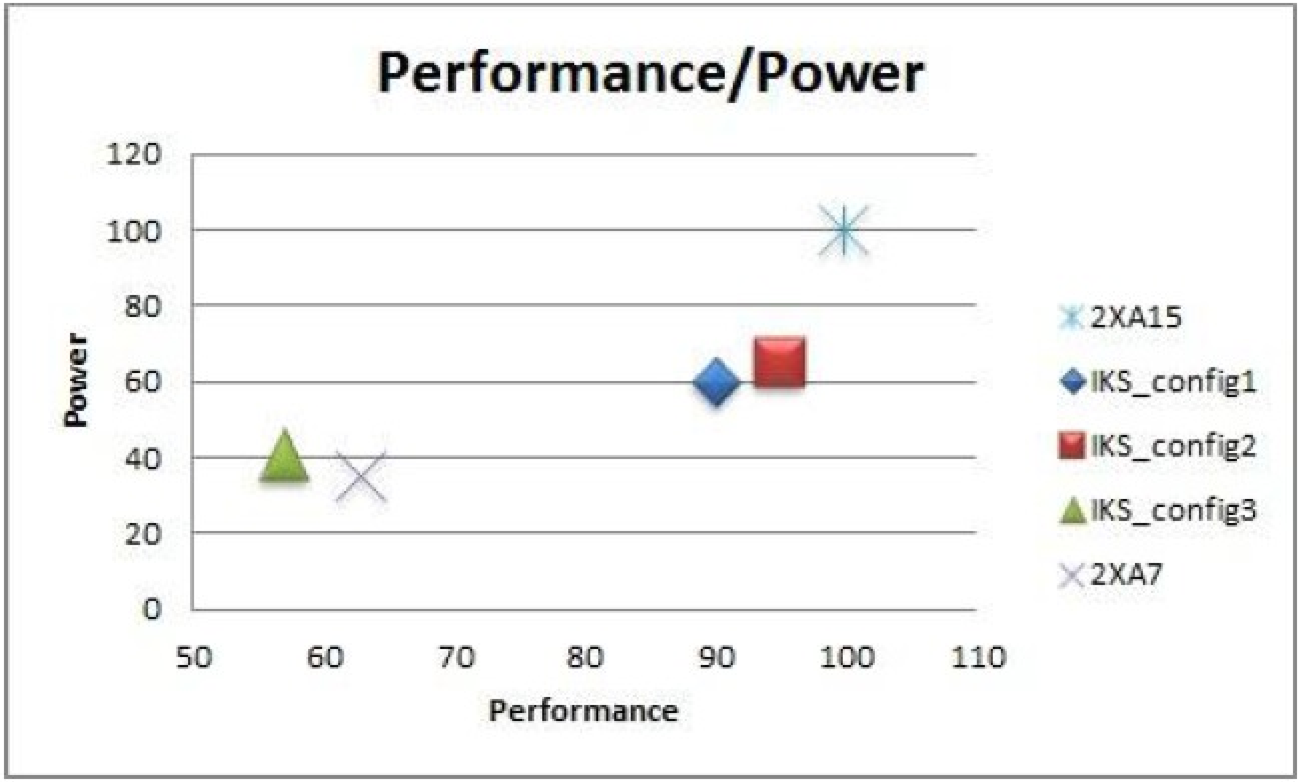


Figure: Measured performance/power results of IKS [16]

4. Exclusive core migration (14)

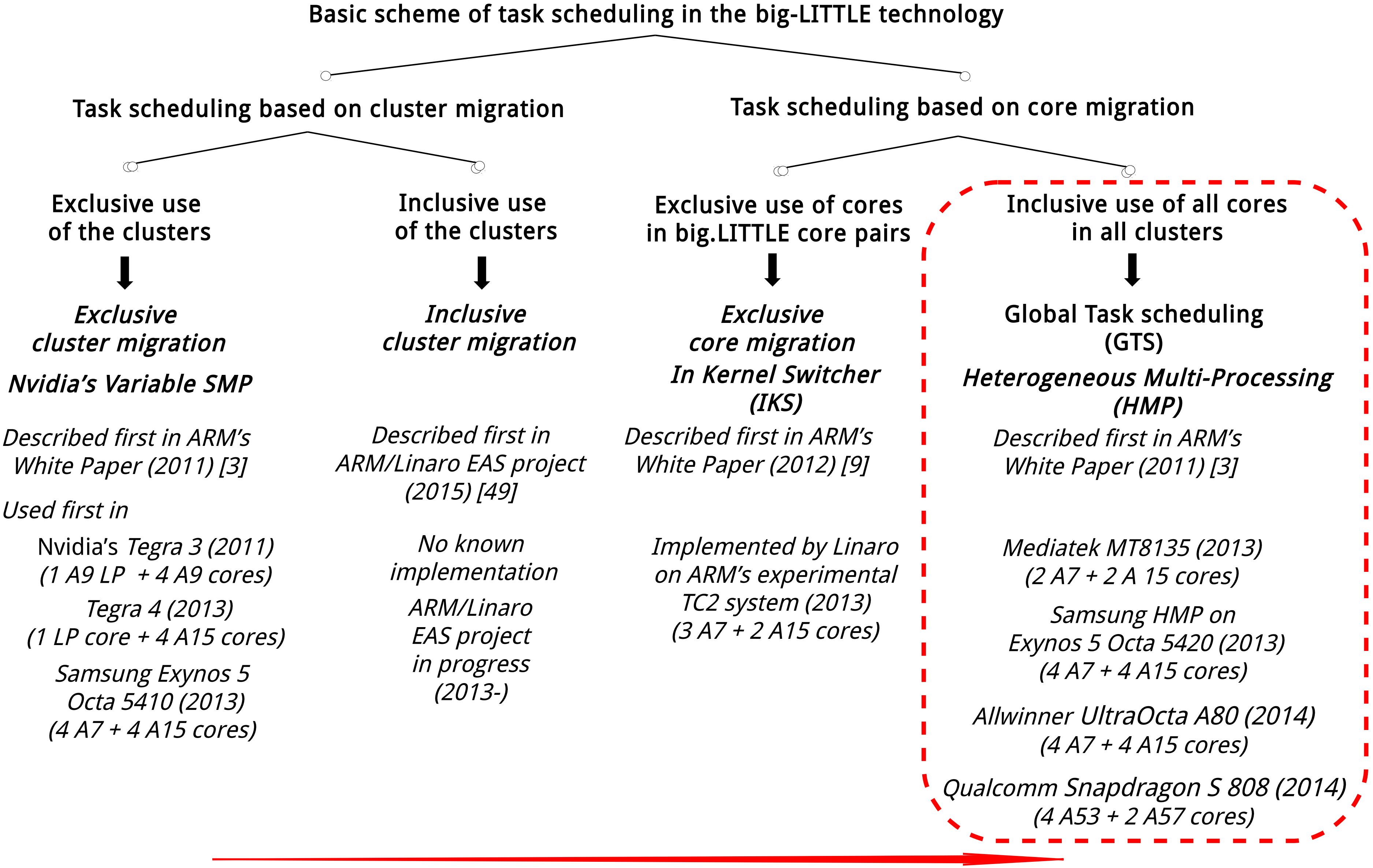
Measured results of IKS-2 [16]

- As the Figure shows, achieved results are not very convincing.
- This can be the reason why no commercial implementation became known using IKS yet.

5. Global task scheduling (GTS)

5. Global Task Scheduling (GTS) (1)

5. Global taks scheduling (GTS) -1



Basic scheme of task scheduling in the big-LITTLE technology

Task scheduling based on cluster migration

Exclusive use of the clusters

Exclusive cluster migration

Nvidia's Variable SMP

Described first in ARM's White Paper (2011) [3]

Used first in

Nvidia's *Tegra 3 (2011) (1 A9 LP + 4 A9 cores)*

Tegra 4 (2013) (1 LP core + 4 A15 cores)

Samsung Exynos 5 Octa 5410 (2013) (4 A7 + 4 A15 cores)

Inclusive use of the clusters

Inclusive cluster migration

Described first in ARM/Linaro EAS project (2015) [49]

No known implementation

ARM/Linaro EAS project in progress (2013-)

Task scheduling based on core migration

Exclusive use of cores in big.LITTLE core pairs

Exclusive core migration

In Kernel Switcher (IKS)

Described first in ARM's White Paper (2012) [9]

Implemented by Linaro on ARM's experimental TC2 system (2013) (3 A7 + 2 A15 cores)

Inclusive use of all cores in all clusters

Global Task scheduling (GTS)

Heterogeneous Multi-Processing (HMP)

Described first in ARM's White Paper (2011) [3]

Mediatek MT8135 (2013) (2 A7 + 2 A 15 cores)

Samsung HMP on Exynos 5 Octa 5420 (2013) (4 A7 + 4 A15 cores)

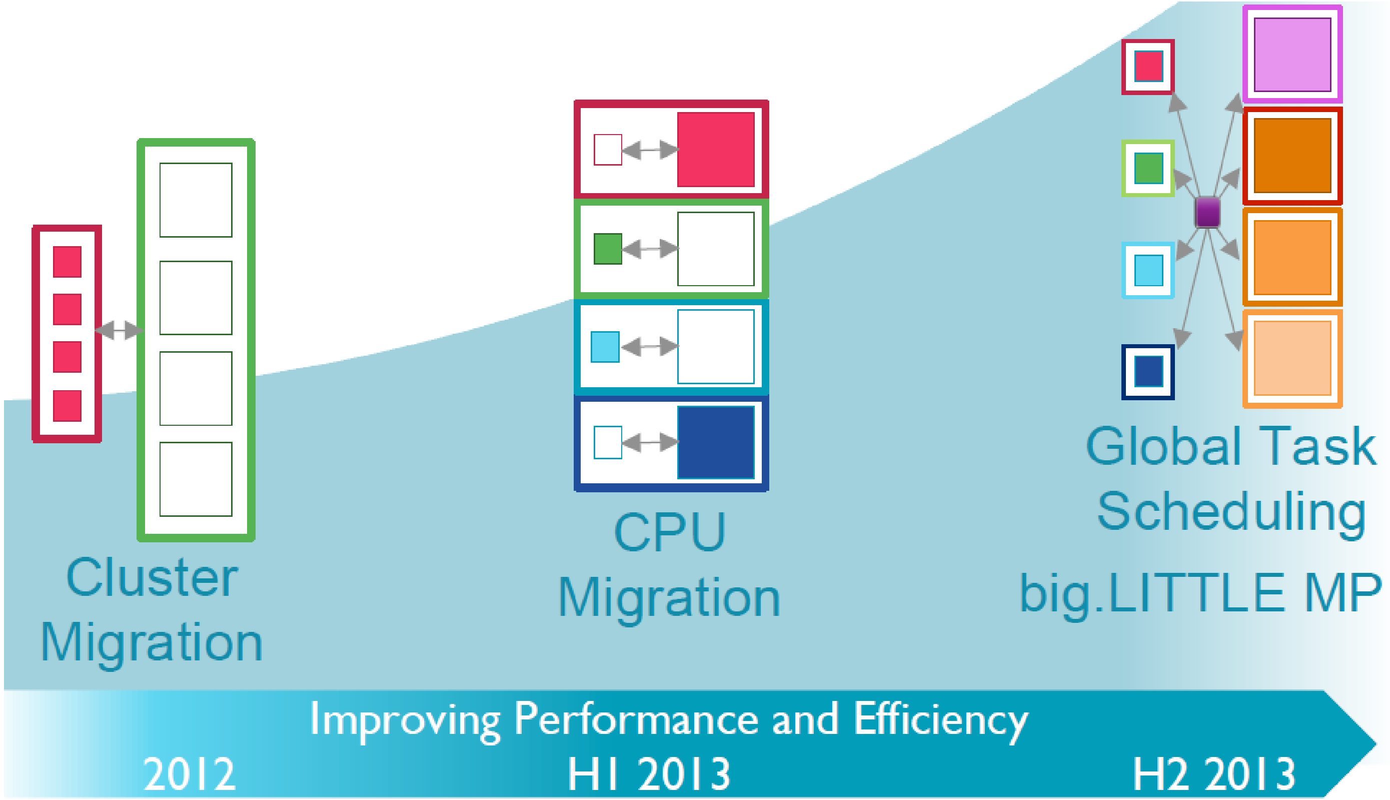
Allwinner UltraOcta A80 (2014) (4 A7 + 4 A15 cores)

Qualcomm Snapdragon S 808 (2014) (4 A53 + 2 A57 cores)

5. Global Task Scheduling (GTS) (2)

Global taks scheduling (GTS) -2

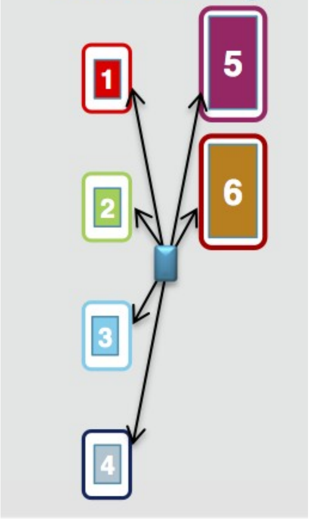
Global taks scheduling (GTS) or big.LITTLE MP in ARM's terminology, can be considered as the final step of the evolution of the big.LITTLE technology, as indicated below [17].



5. Global Task Scheduling (GTS) (3)

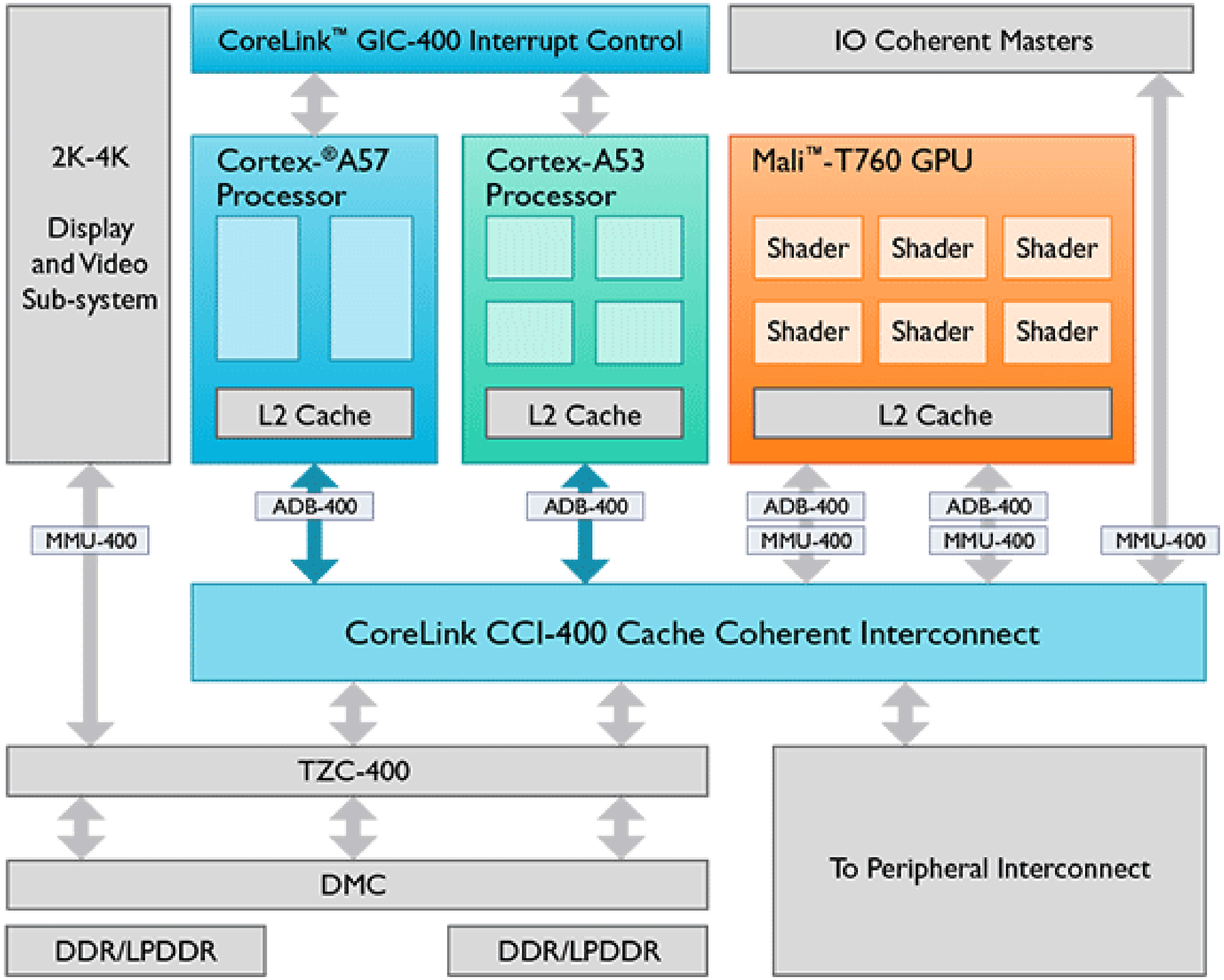
Principle of GTS [8], [5]

- OS (e.g. a modified Linux scheduler) tracks the average load of each task, e.g. in time-windows.
- The processor has at least two clusters of architecturally identical cores at its disposal, e.g. a big cluster including two cores, and a LITTLE cluster with four cores, as shown in the Figure on the right.
- The OS scheduler has all cores of both clusters or of all three clusters at its disposal and can schedule tasks to any core at any time.
- There are many options for the layout of the scheduling policy, to be discussed later in Section 6.



5. Global Task Scheduling (GTS) (4)

Example block diagram of a big.LITTLE SOC with GTS [1]



Taken from ARM's presentation of the big.LITTLE technology [1]

ADB: AMBA Domain Bridge

AMBA: Advanced Microcontroller Bus architecture

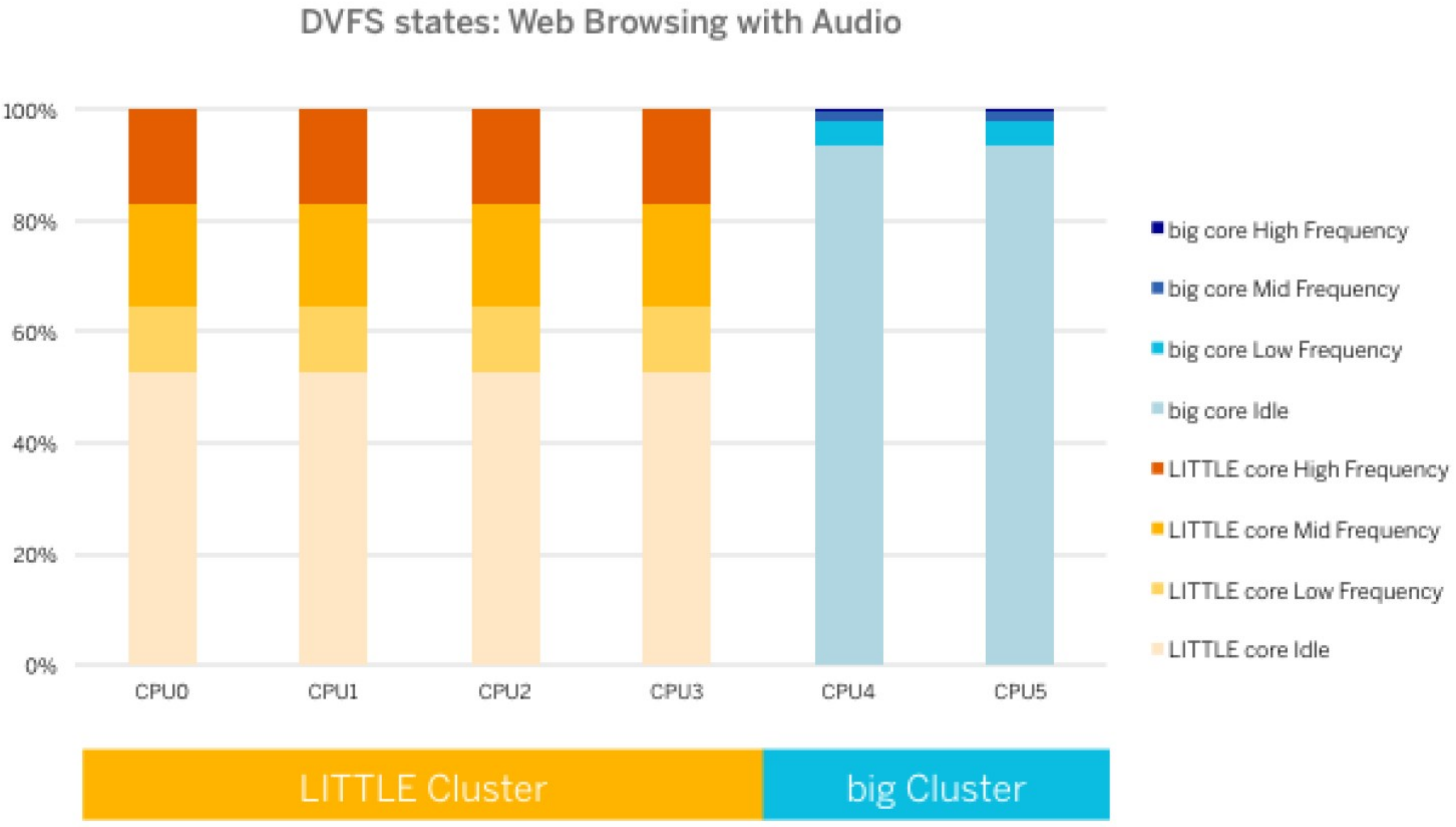
MMU: Memory Management Unit

TZC: TrustZone Address Space Controller

DMC: Dynamic Memory Controller

5. Global Task Scheduling (GTS) (5)

Core residency at various DVFS frequency states of a 2 big/4 LITTLE GTS configuration for web browsing with audio [19]



5. Global Task Scheduling (GTS) (6)

Key benefits of GTS over IKS or exclusive cluster migration [18]

- Finer grained scheduling of workloads than achievable by cluster migration.
- Ability to easily support non-symmetric configurations, such as (2+4) ones, vs IKS.
- Higher peak performance through the ability to use all cores simultaneously.
- Higher performance/Watt vs. IKS.

E.g. ARM reported about 10 % higher performance/Watt figures over IKS on a range of benchmarks.

But performance and energy consumption is greatly dependent on the task scheduler.

5. Global Task Scheduling (GTS) (7)

Achieved power saving of a big.LITTLE configuration with GTS vs. a traditonal configuration

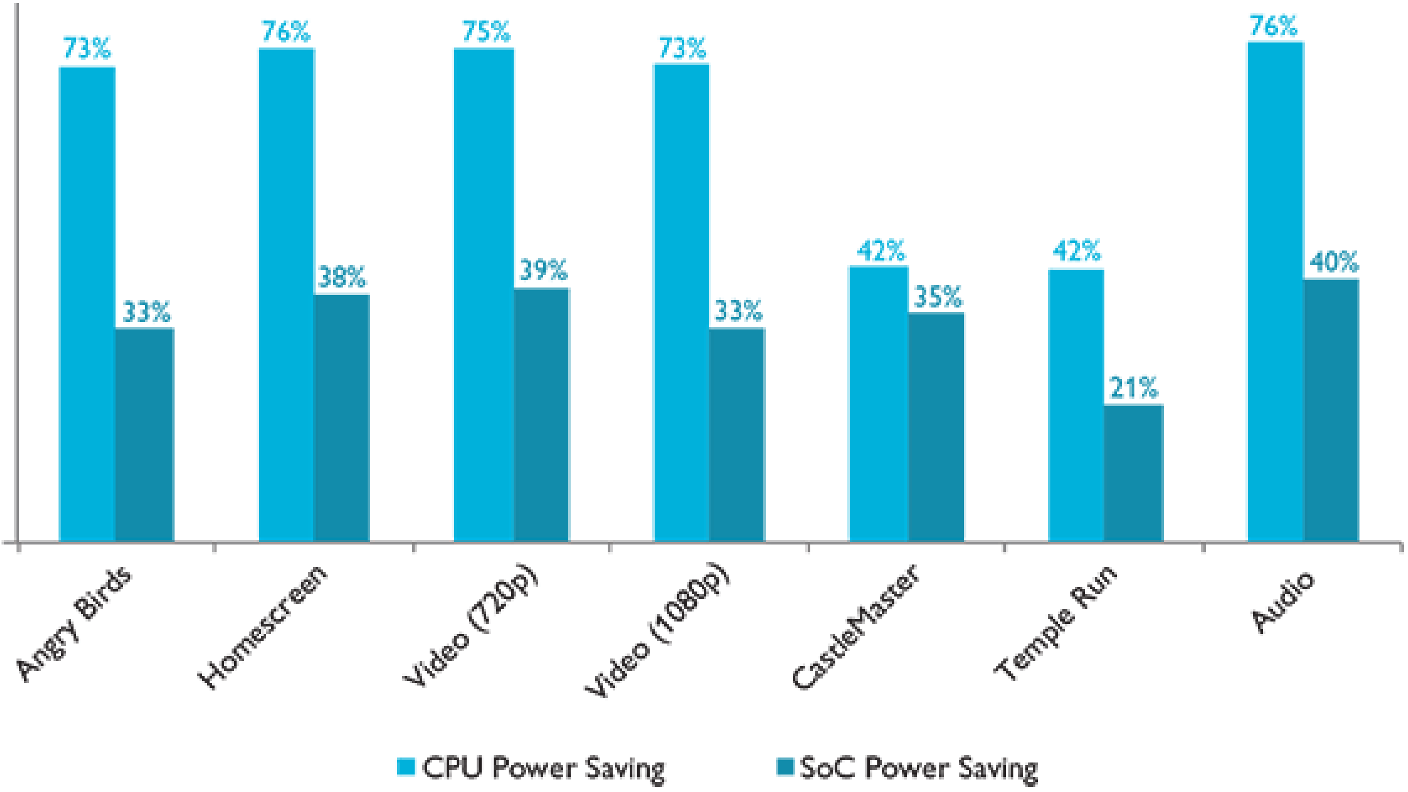


Figure: Measured CPU and SoC power savings on a 4x Cortex-A15 4x•Cortex-A7 big.LITTLE MP SoC relative to a 4x Cortex-A15 SoC for different applications [1]

5. Global Task Scheduling (GTS) (8)

Overview of big.LITTLE implementations with GTS

Model	Year	Cores	Techn.	Integrated modem
Samsung Exynos 5 Octa 5420	2013	4x A7 + 4x A15	28 nm	no
Samsung Exynos 5 Octa 5422	2014	4x A7 + 4x A15	28 nm	no
Samsung Exynos 5 Octa 5260	2014	4x A7 + 2x A15	28 nm	no
Samsung Exynos 5 Octa 5430	2014	4x A7 + 4x A15	20 nm	no
Samsung Exynos 5 Octa 5433	2014	4x A53 + 4x A57	20 nm	no
Samsung Exynos 7 Octa 7420	2015	4x A53 + 4x A57	14 nm	no
Samsung Exynos 8 Octa 8890	2015	4x A53 + 4x M1	14 nm	yes
Qualcomm Snapdagon S 808	2014	4x A53 + 2x A57	20 nm	no
Qualcomm Snapdagon S 810	2015	4x A53 + 4x A57	20 nm	no
Qualcomm Snapdagon S 820	2016	2x Kryo 1.7 GHz + 2x Kryo 2.2 GHz	14 nm FnFET	no
MediaTek MT8135	2013	2x A7 + 2x A15	28 nm	no
MediaTek MT6595	2014	4x A7 + 4x A17	28 nm	yes
MediaTek MT6797	2015	8x A53+ 2x A57	20 nm	yes
Renesas MP 6530	2013	2x A7 + 2x A15	28 nm	yes
Allwinner UltraOcta A80	2014	4x A7 + 4x A15	28 nm	no

5. Global Task Scheduling (GTS) (9)

Main features of Samsung's mobile SOCs in big.LITTLE configuration

SoC			CPU			GPU	Memory technology	Availability	Utilizing devices (examples)
Model number	fab	Instr. set	Microarch.	cores	fc (GHz)				
Exynos 5 Octa *(Exynos 5410)*	28 nm HKMG	ARMv7	Cortex-A15+ Cortex-A7[1]	4+4	1.6 1.2	IT PowerVR SGX544MP3 @ 480 MHz 49 GFLOPS	32-bit DCh LPDDR3-800 (12.8 GB/sec)	Q2 2013	Samsung Galaxy S4 I9500, ZTE Grand S II TD,
Exynos 5 Octa *(Exynos 5420)*	28 nm HKMG		Cortex-A15+ Cortex-A7(2)	4+4	1.8–1.9 1.3	ARM Mali-T628 MP6 @ 533 MHz; 109 GFLOPS	32-bit DCh LPDDR3e-933 (14.9 GB/sec)	Q3 2013	Samsung Chromebook 2 11.6", Samsung Galaxy Note 3/Note 10.1/Note Pro 12.2, Samsung Galaxy Tab Pro/Tab S
Exynos 5 Octa *(Exynos 5422)*	28 nm HKMG		Cortex-A15+ Cortex-A7[2]	4+4	1.9-2.1 1.3-1.5	ARM Mali-T628 MP6 @ 533 MHz (109 Gflops)	32-bit DCh LPDDR3/DDR3-933 (14.9 GB/sec)	Q2 2014	Samsung Galaxy S5 (SM-G900H),
Exynos 5 Octa *(Exynos 5800)*	28 nm HKMG		Cortex-A15+ Cortex-A7[2]	4+4	2.1 1.3	ARM Mali-T628 MP6 @ 533 MHz	32-bit DCh LPDDR3/DDR3-933 (14.9 GB/sec)	Q2 2014	Samsung Chromebook 2 13,3"
Exynos 5 Hexa *(Exynos 5260)*	28 nm HKMG		Cortex-A15+ Cortex-A7[2]	2+4	1.7 1.3	ARM Mali-T624 @ 600 MHz	32-bit DCh LPDDR3-800 (12.8 GB/sec)	Q2 2014	Galaxy Note 3 Neo, Samsung Galaxy K zoom
Exynos 5 Octa *(Exynos 5430)*	20 nm HKMG		Cortex-A15+ Cortex-A7[2]	4+4	1.8-2.0 1.3-1.5	ARM Mali-T628 MP6 @ 600 MHz; 122 GFLOPS	32-bit DCh LPDDR3e/DDR3-1066 (17.0 GB/sec)	Q3 2014	Samsung Galaxy Alpha (SM-G850F),
Exynos 5 Octa (Exynos 5433	20 nm HKMG	ARMv8-A	Cortex-A57+ Cortex-A53[2]	4+4	1.9 1.3	Mali-T760 MP6 @ 700 MHz; 206 GFLOPS (FP16)	32-bits DCh LPDDR3-825 (13.2 GB/s)	Q3/Q4 2014	Samsung Galaxy Note 4 (SM-N910C)
Exynos 7 Octa (Exynos 7420)	14 nm FinFET	ARMv8-A	Cortex-A57+ Cortex-A53[2]	4+4	2.1 1.5	Mali-T760 MP8 @ 772 MHz; 227 GFLOPS (FP16)	32-bits DCh LPDDR4-1552 (25.6 GB/s)	Q2 2015	Samsung Galaxy S6/ S6 Edge
Exynos 8 Octa (Exynos 8890)	14 nm	ARMv8-A	Mongoose Cortex-	4+4	2.3 n.a.	Mali-T880 MP12 @	n.a.	Q4 201	Smsung Galaxy S7

[1] big.LITTLE configuration with exclusive core migration

[2] big.LITTLE configuration with GTS

5. Global Task Scheduling (GTS) (10)

Leaked Geekbench scores of latest mobile processors [65]

Model	Introduced	Cores	Clock frequency	Single core performance	Multicore performance
Samsung Exynos 8890	2015	2x Mongoose+ 2x Mongoose	2.3 GHz ? GHz	2294	6908
Samsung Exynos 7420	2015	4x Cortex A57+ 4x Cortex A53	2.1 GHz 1.5 GHz	1486	4970
Apple A9	2015	2x Cyclone	1.9 GHz	2487	4330

Remark [66]

"Geekbench is a cross-platform processor benchmark, with a scoring system that separates single-core and multi-core performance, and workloads that simulate real-world scenarios." (Source: Wikipedia)

As a comparison the Geekbench score of the Intel Core m7-6Y75 (Skylake processor at 1.512 GHz with a TDP of 4.5 W) is about 2500. (http://www.primatelabs.com/)

6. Supporting GTS in OS kernels

- 6.1 Overview
- 6.2 OS support for GTS provided by ARM/Linaro
- 6.3 MediaTek's CorePilot releases
- 6.4 Quacomm's big.LITTLE schedulers

6.1 Overview

6.1 Overview (1)

6.1 Overview

- big.LITTLE technology needs suitable OS support to schedule tasks to the right computing resources for achieving the least power consumption.
- It is stated that "software represents the Achilles' heel of the technology and severely limits its potential [53].
- ARM and Linaro jointly develop OS support fo GTS, these will be made available first as Linux or Android patch sets, later also they can be included into the mainstream Linux or - Android kernel.
- As an example, ARM/Linaro's IPA (Intelligent Power Management) became first available as a Linaro patch set in 09/2014 and then it was included into Linux 4.10 in 8/2015.
- Accordingly, in this section we give an overview of the OS support of GTS.

Remark

Linaro is a non-profit foundation of interested firms to foster open source Linux packages optimized for ARM architectures.

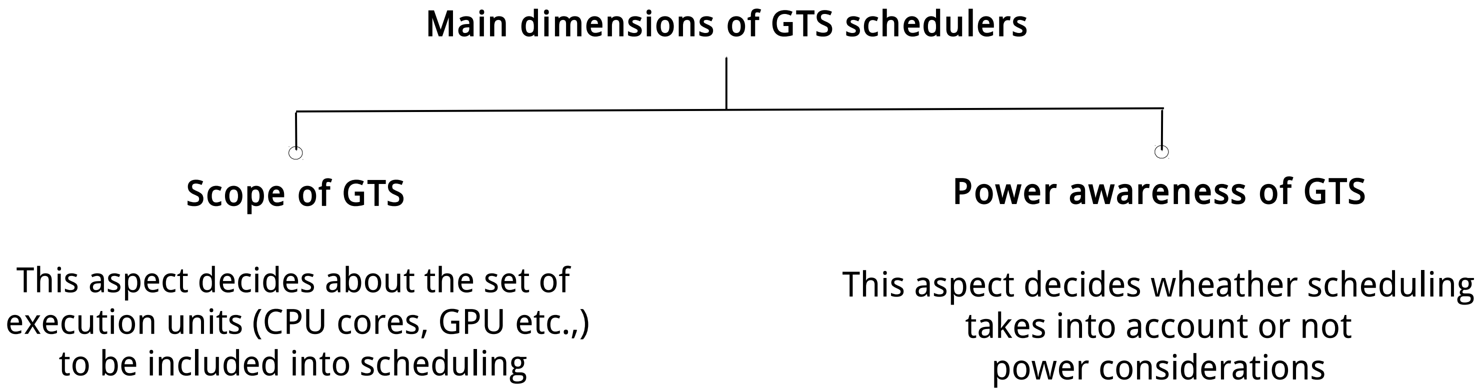
6.1 Overview (2)

Overview of supporting GTS in the OS kernel (announced or used)

ARM/Linaro	ARM big.LITTLE MP (Global Task Scheduling) (~06/2013)	ARM IPA (Inteligent Power Allocation) (10/2014)	ARM/Linaro EAS (Energy Aware Scheduling) (development yet in progress)
MediaTek	MediaTek CorePilot 1.0 (on MT8135) (07/2013)		MediaTek CorePilot 2.0 (on Helio X10 (MT6595) (03/2015) MediaTek CorePilot 3.0 (on Helio X20 (MT6797) (05/2015)
Qualcomm		Qualcomm's Energy Aware Scheduling (on Snapdragoon 610/615 (02/2014))	Qualcomm Symphony System Manager (on Snapdragoon 820) (11/2015)
Samsung	Samsung's big.LITTLE HMP (≈ ARM's big.LITTLE MP) (on Exynos 5 models) (09/2013)		
	2013	2014	2015

6.1 Overview (3)

Main dimensions of GTS schedulers



Main dimensions of GTS schedulers

Scope of GTS

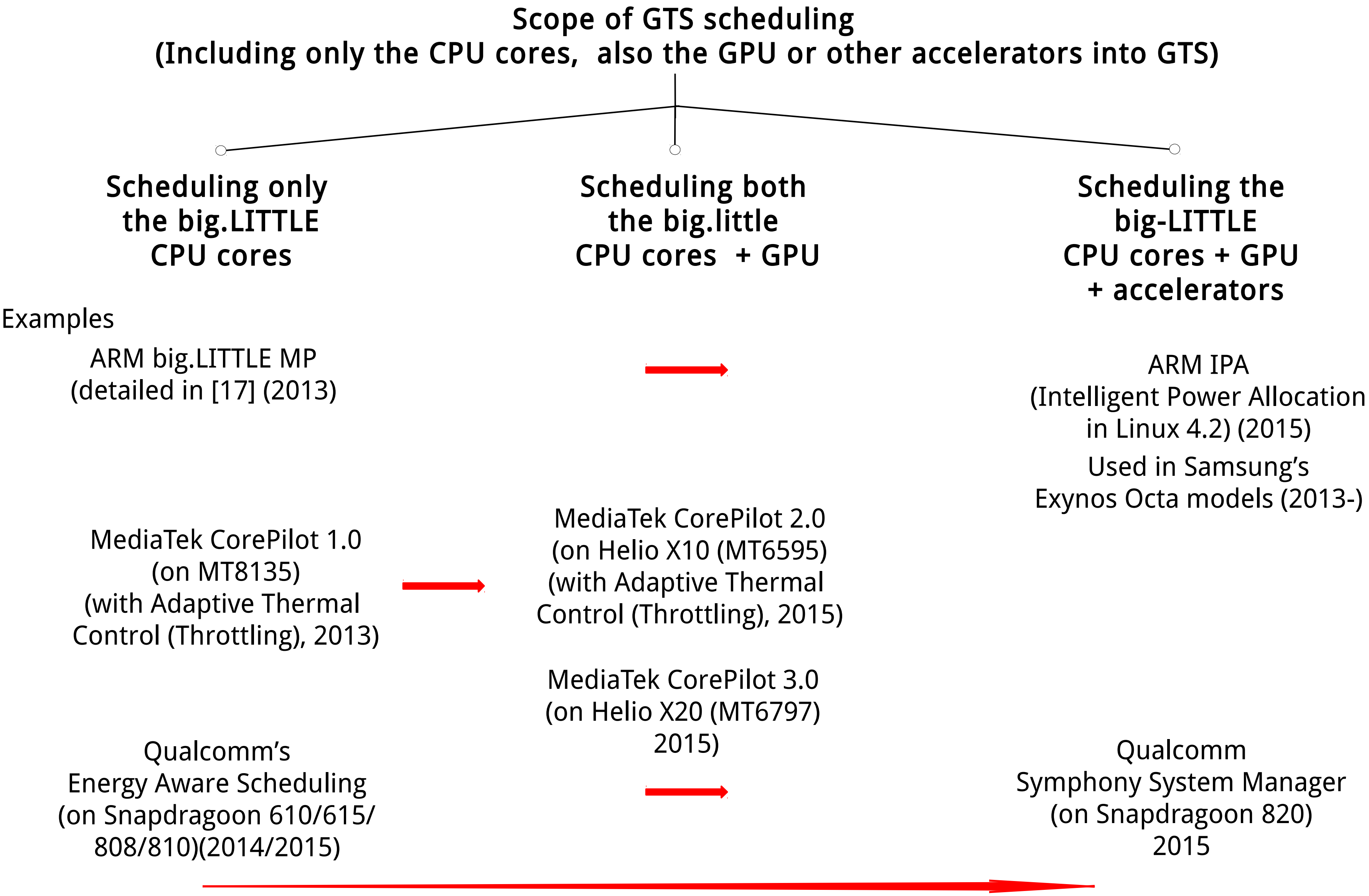
This aspect decides about the set of execution units (CPU cores, GPU etc.,) to be included into scheduling

Power awareness of GTS

This aspect decides wheather scheduling takes into account or not power considerations

6.1 Overview (4)

Scope of GTS scheduling



Scope of GTS scheduling
(Including only the CPU cores, also the GPU or other accelerators into GTS)

Scheduling only the big.LITTLE CPU cores

Scheduling both the big.little CPU cores + GPU

Scheduling the big-LITTLE CPU cores + GPU + accelerators

Examples

ARM big.LITTLE MP
(detailed in [17] (2013)

ARM IPA
(Intelligent Power Allocation
in Linux 4.2) (2015)

Used in Samsung's
Exynos Octa models (2013-)

MediaTek CorePilot 1.0
(on MT8135)
(with Adaptive Thermal
Control (Throttling), 2013)

MediaTek CorePilot 2.0
(on Helio X10 (MT6595)
(with Adaptive Thermal
Control (Throttling), 2015)

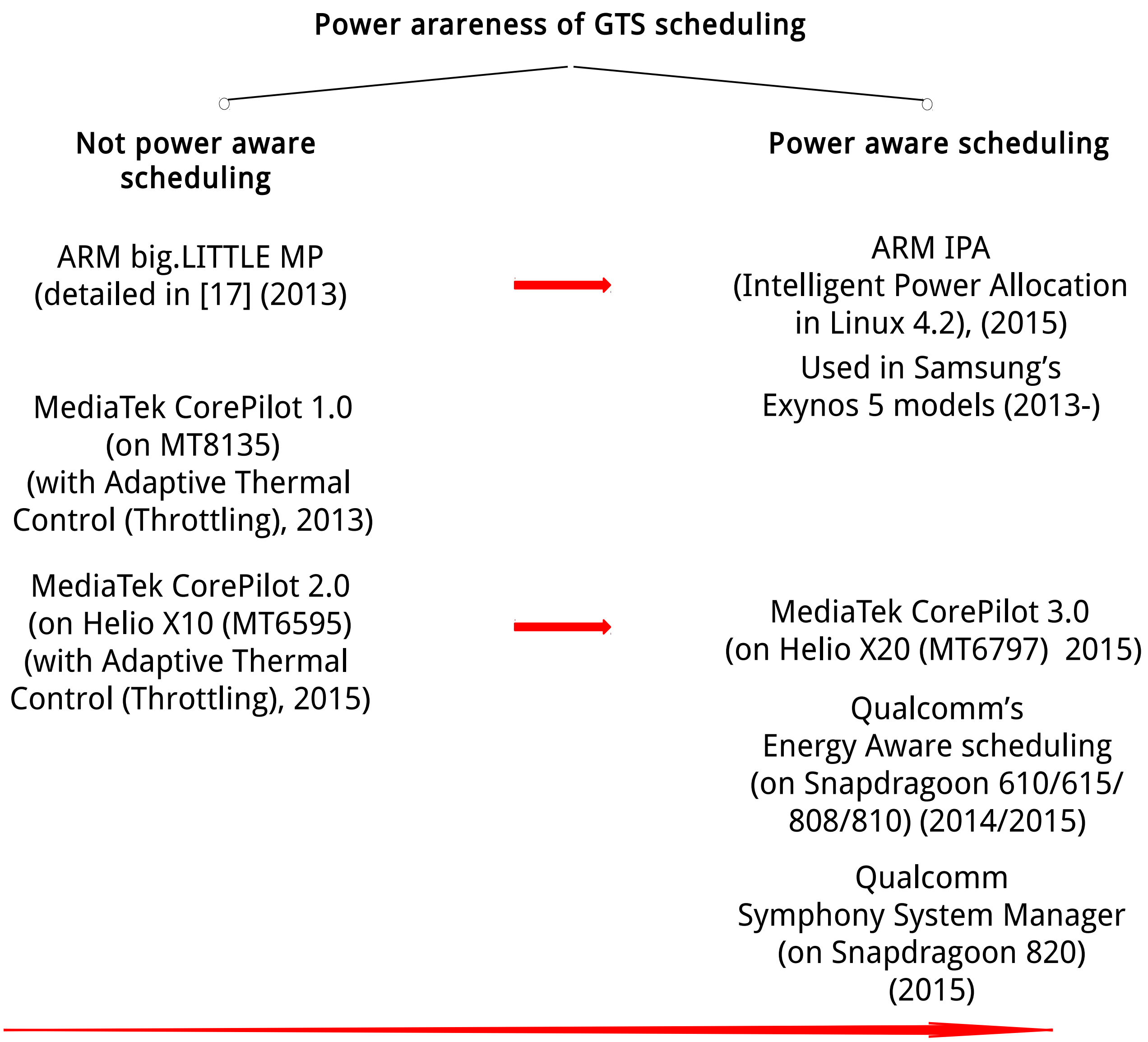
MediaTek CorePilot 3.0
(on Helio X20 (MT6797)
2015)

Qualcomm's
Energy Aware Scheduling
(on Snapdragoon 610/615/
808/810)(2014/2015)

Qualcomm
Symphony System Manager
(on Snapdragoon 820)
2015

6.1 Overview (5)

Power awareness of GTS scheduling



Power arareness of GTS scheduling

Not power aware scheduling

Power aware scheduling

Examples

ARM big.LITTLE MP
(detailed in [17] (2013)

MediaTek CorePilot 1.0
(on MT8135)
(with Adaptive Thermal
Control (Throttling), 2013)

MediaTek CorePilot 2.0
(on Helio X10 (MT6595)
(with Adaptive Thermal
Control (Throttling), 2015)

ARM IPA
(Intelligent Power Allocation
in Linux 4.2), (2015)

Used in Samsung's
Exynos 5 models (2013-)

MediaTek CorePilot 3.0
(on Helio X20 (MT6797) 2015)

Qualcomm's
Energy Aware scheduling
(on Snapdragoon 610/615/
808/810) (2014/2015)

Qualcomm
Symphony System Manager
(on Snapdragoon 820)
(2015)

6.1 Overview (6)

Scope and power awareness of GTS schedulers

Scope of GTS scheduling
(Including only the CPU cores, also the GPU or other accelerators into GTS)

		Scheduling only the big.LITTLE CPU cores	**Scheduling both the big.LITTLE CPU cores + GPU**	**Scheduling the big-LITTLE CPU cores + GPU + accelerators**
Power arareness of GTS scheduling	**Not power aware scheduling**	ARM big.LITTLE MP (detailed in [17] (2013) MediaTek CorePilot 1.0 (on MT8135) (with Adaptive Thermal Control (Throttling), 2013)	MediaTek CorePilot 2.0 (on Helio X10 (MT6595) (with Adaptive Thermal Control (Throttling), 2015)	
	Power aware scheduling	Qualcomm's Energy Aware Scheduling (on Snapdragoon 610/615/ 808/810) (2014/2015)	MediaTek CorePilot 3.0 (on Helio X20 (MT6797) 2015)	ARM iPA (Intelligent Power Allocation) (in Linux 4.2, 2015) Used in Samsung's Exynos Octa (2014-) Qualcomm Symphony System Manager (on Snapdragoon 820) (2015)

6.2 OS support for GTS provided by ARM/Linaro

6.2 OS support for GTS provided by ARM/Linaro (1)

6.2 OS support for GTS provided by ARM/Linaro

	2013	2014	2015
ARM/Linaro	ARM big.LITTLE MP (Global Task Scheduling) (~06/2013)	ARM IPA (Inteligent Power Allocation) (10/2014)	ARM/Linaro EAS (Energy Aware Scheduling) (development yet in progress)
MediaTek	MediaTek CorePilot 1.0 (on MT8135) (07/2013)		MediaTek CorePilot 2.0 (on Helio X10 (MT6595) (03/2015) MediaTek CorePilot 3.0 (on Helio X20 (MT6797) (05/2015)
Qualcomm		Qualcomm's Energy Aware Scheduling (on Snapdragoon 610/615 (02/2014))	Qualcomm Symphony System Manager (on Snapdragoon 820) (11/2015)
Samsung	Samsung's big.LITTLE HMP (≈ ARM's big.LITTLE MP) (on Exynos 5 models) (09/2013)		

6.2 OS support for GTS provided by ARM/Linaro (2)

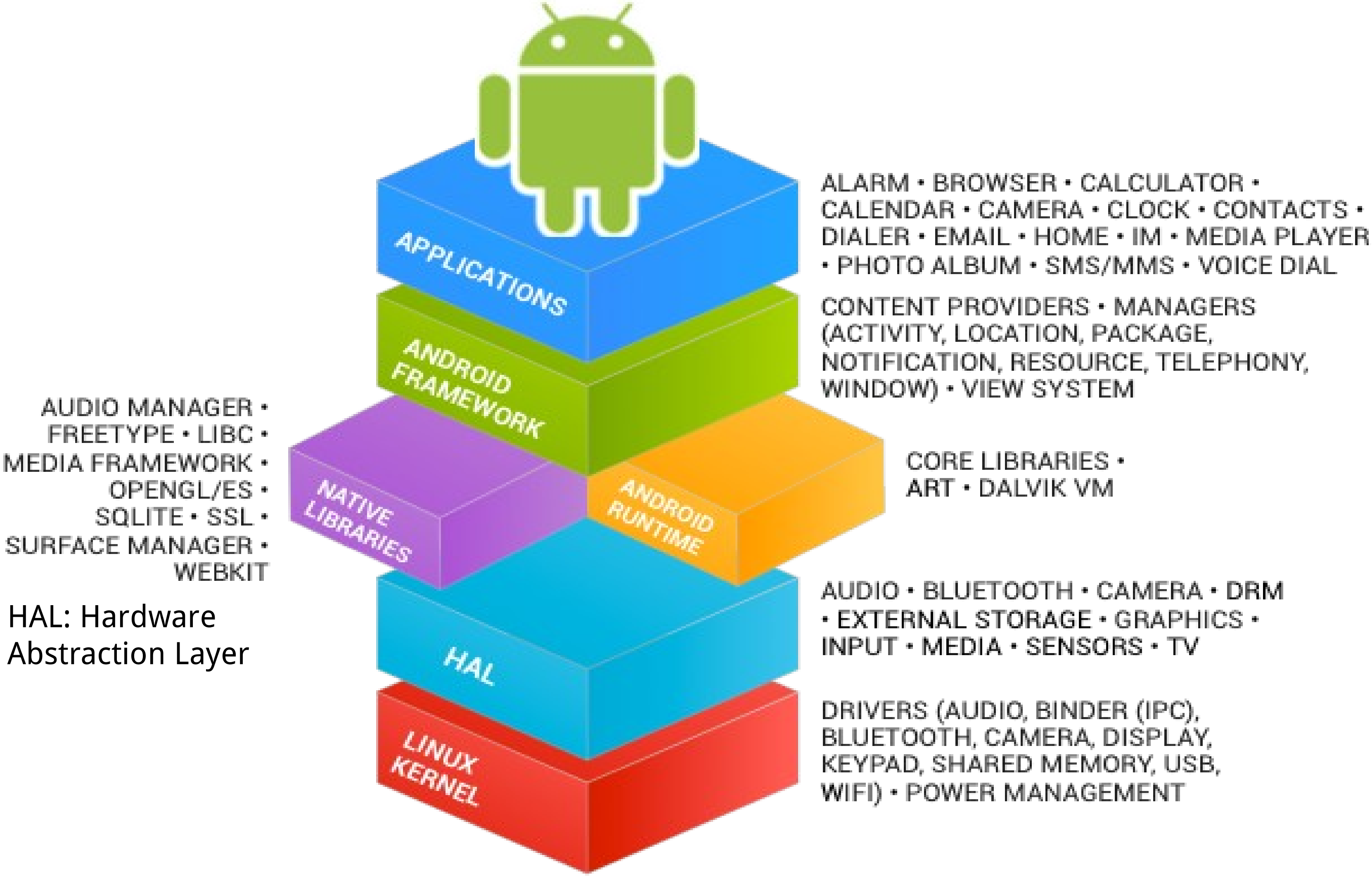
6.2.1 ARM big.LITTLE MP (Global Task Scheduling)

- For scheduling tasks on big.LITTLE mobile systems ARM developed a Linux kernel modification, called the ARM MP patch set, that became subsequently included into the Linux 3.10.33 (03/2014).
- There is also an Android v3.10 patch set for GTS from AOSP.

AOSP: Android Open Source Project (Led by Google.
This site offers the information and source code to customize Android).

6.2 OS support for GTS provided by ARM/Linaro (3)

The Android software stack [66]



ALARM • BROWSER • CALCULATOR • CALENDAR • CAMERA • CLOCK • CONTACTS • DIALER • EMAIL • HOME • IM • MEDIA PLAYER • PHOTO ALBUM • SMS/MMS • VOICE DIAL

CONTENT PROVIDERS • MANAGERS (ACTIVITY, LOCATION, PACKAGE, NOTIFICATION, RESOURCE, TELEPHONY, WINDOW) • VIEW SYSTEM

AUDIO MANAGER • FREETYPE • LIBC • MEDIA FRAMEWORK • OPENGL/ES • SQLITE • SSL • SURFACE MANAGER • WEBKIT

CORE LIBRARIES • ART • DALVIK VM

AUDIO • BLUETOOTH • CAMERA • DRM • EXTERNAL STORAGE • GRAPHICS • INPUT • MEDIA • SENSORS • TV

DRIVERS (AUDIO, BINDER (IPC), BLUETOOTH, CAMERA, DISPLAY, KEYPAD, SHARED MEMORY, USB, WIFI) • POWER MANAGEMENT

HAL: Hardware Abstraction Layer

6.2 OS support for GTS provided by ARM/Linaro (4)

Principle of operation of GTS -1

- Since release 2.6.23 (2009) Linux's scheduler is the Completely Faire Scheduler (CFS), which tries to spit runtime equally between runnable tasks.
- The ARM developed patch set disables the classic load balancing between the CPU CPU cores (done by CFS) and substitutes it by a big.LITTLE specific routine, as indicated below.

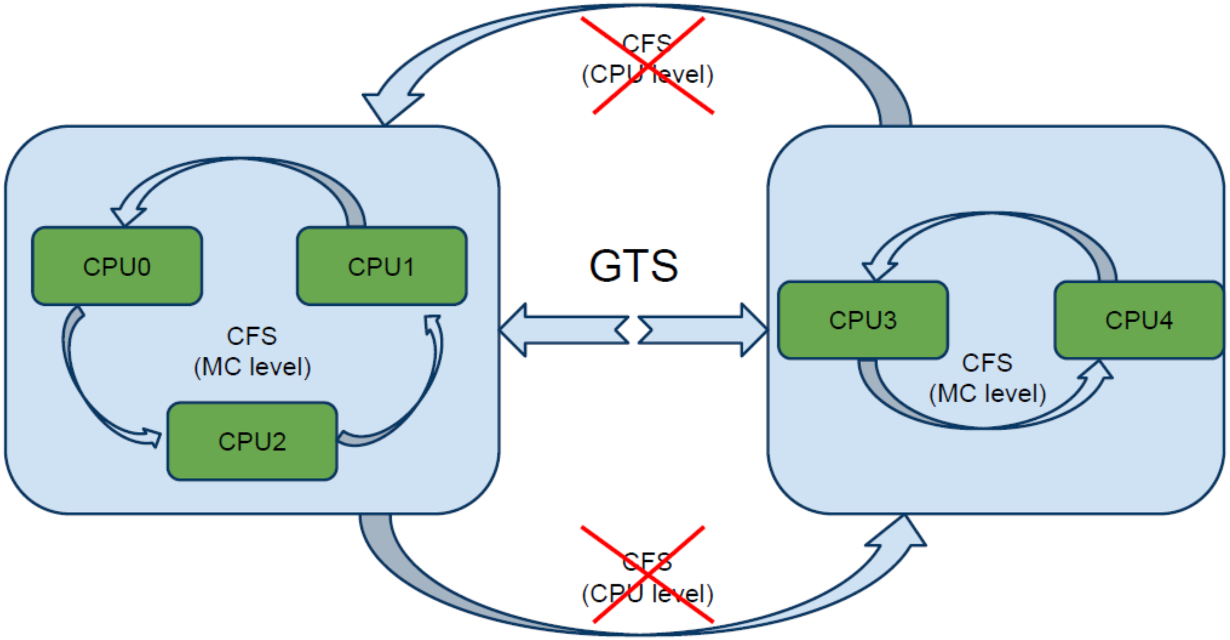
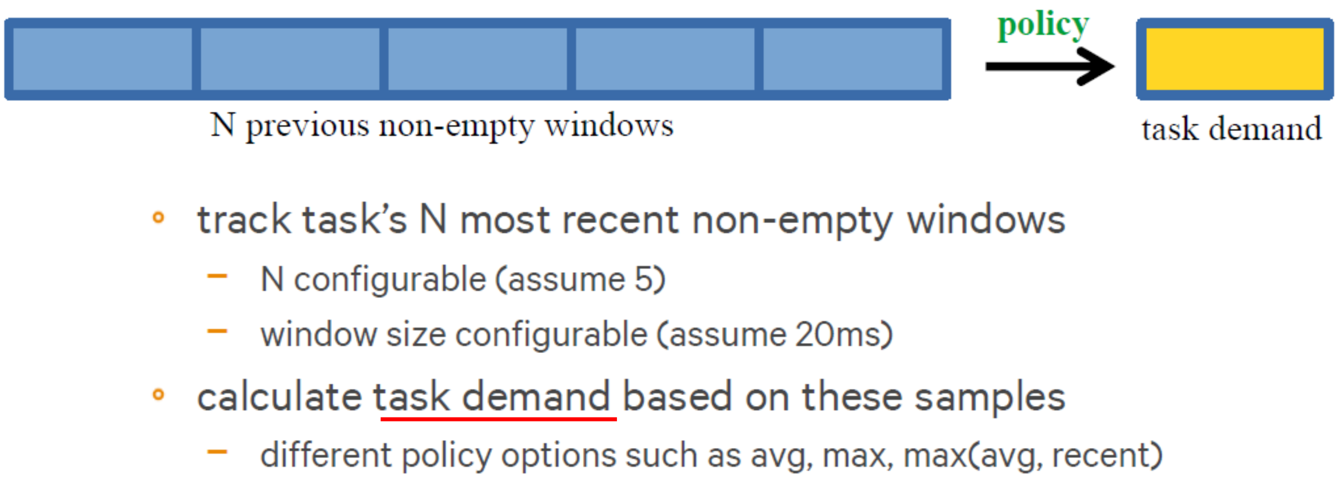


Figure: Disabling the classic load balancing in Linux and substituting it by a big.LITTLE specific routine [54]

6.2 OS support for GTS provided by ARM/Linaro (5)

Principle of operation of GTS -2

- The scheduling is based on a load tracker, that is the scheduling decisions will be accomplished on the sensed load.
- The load tracker performs a per-entity (task), window based load tracking and calculates the load as outlined below.



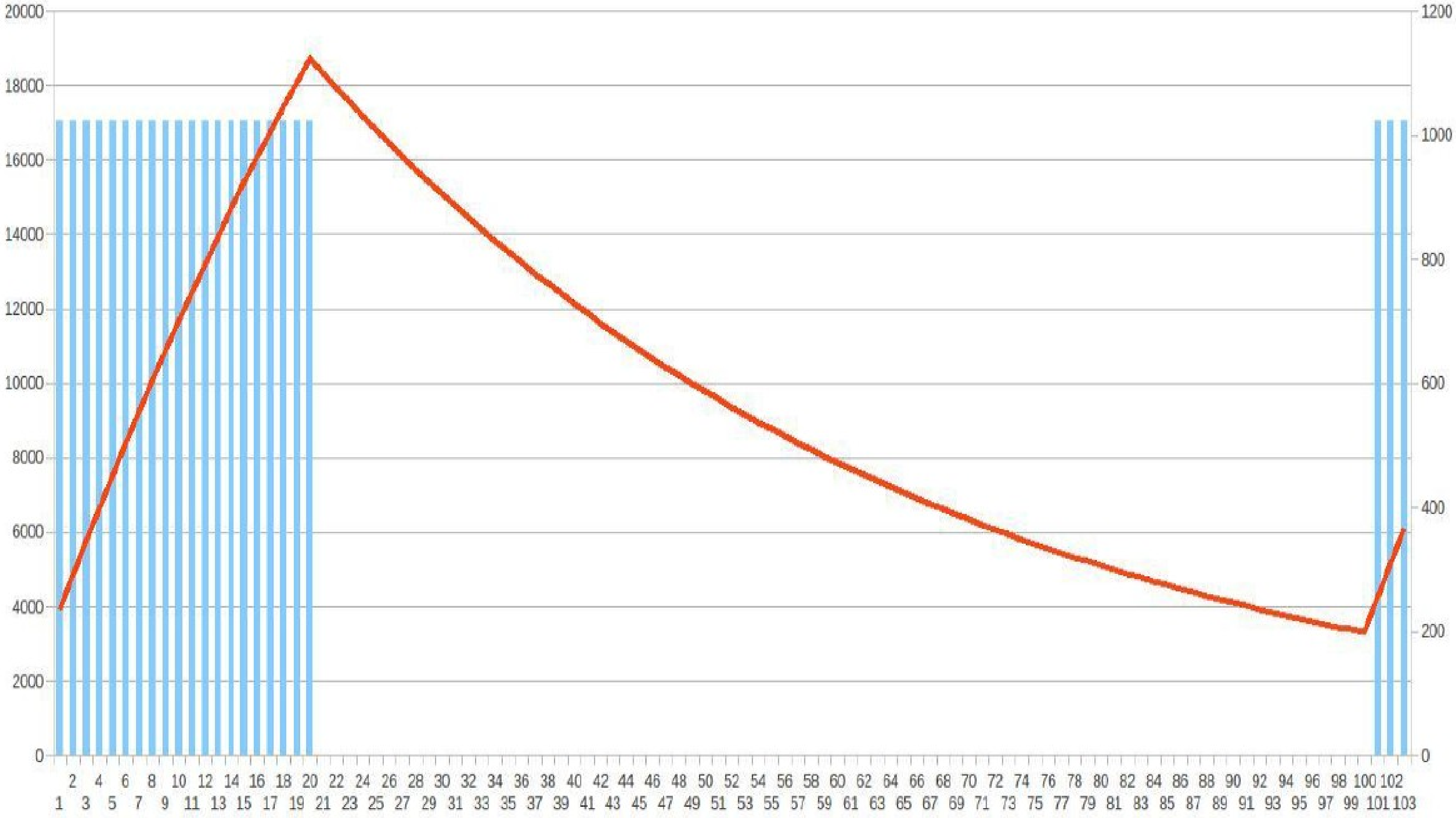
- track task's N most recent non-empty windows
  - N configurable (assume 5)
  - window size configurable (assume 20ms)
- calculate task demand based on these samples
  - different policy options such as avg, max, max(avg, recent)

Figure: Widow based per entitiy load tracking [55]

- The load (task demand) over the windows is weighted such that that the last window is weighted highest and previous loads by a given decay factors.

6.2 OS support for GTS provided by ARM/Linaro (6)

Illustration of calculated avarage load [54]



6.2 OS support for GTS provided by ARM/Linaro (7)

Principle of operation of GTS -3

There are two migration thresholds on the task load and the scheduler operates accordingly, as indicated in the Figure.

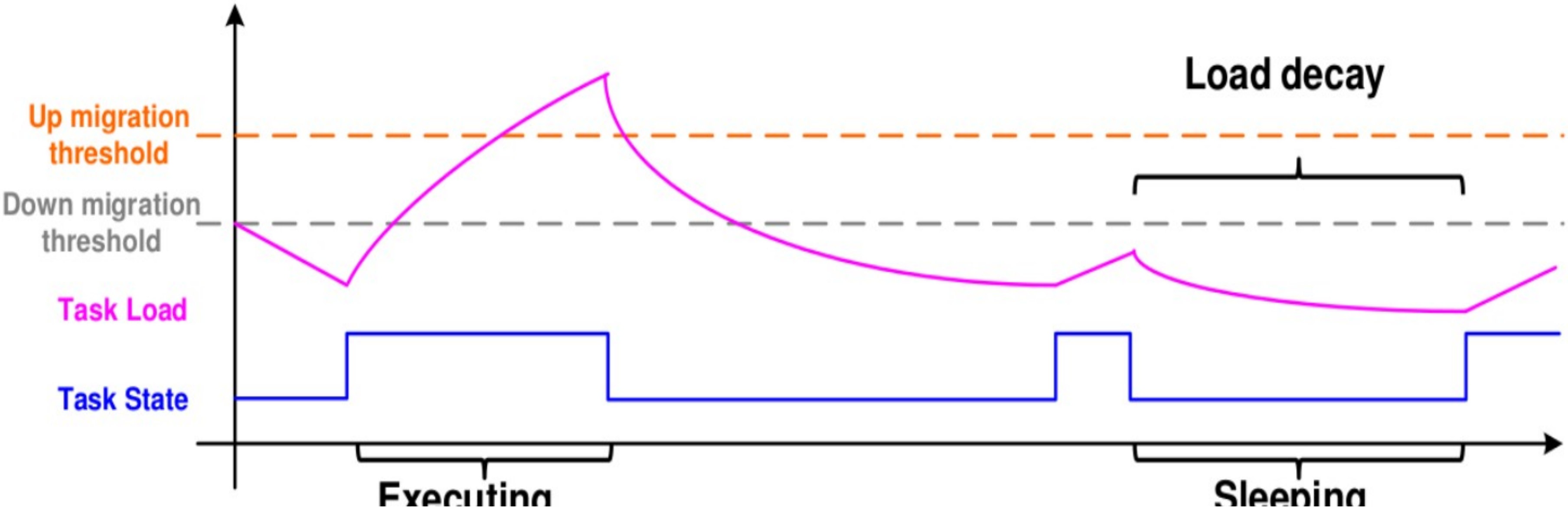


Figure: Basic principle of migrating tasks in GTS [54]

6.2 OS support for GTS provided by ARM/Linaro (8)

6.2.2 ARM IPA (Intelligent Power Allocation)

It is a thermal management based scheduling approach for big-LITTLE mobil topologies covering

- all the big.LITTLE CPU-cores,
- the GPU and
- available data accelerators.

6.2 OS support for GTS provided by ARM/Linaro (9)

Principle of operation of IPA -1

- IPA tracks the performance requests of the actors (everything that dissipates heat, like the CPU cores, the GPU, the modem etc.) derived from clock frequency and utilization, as indicated in the Figure below.

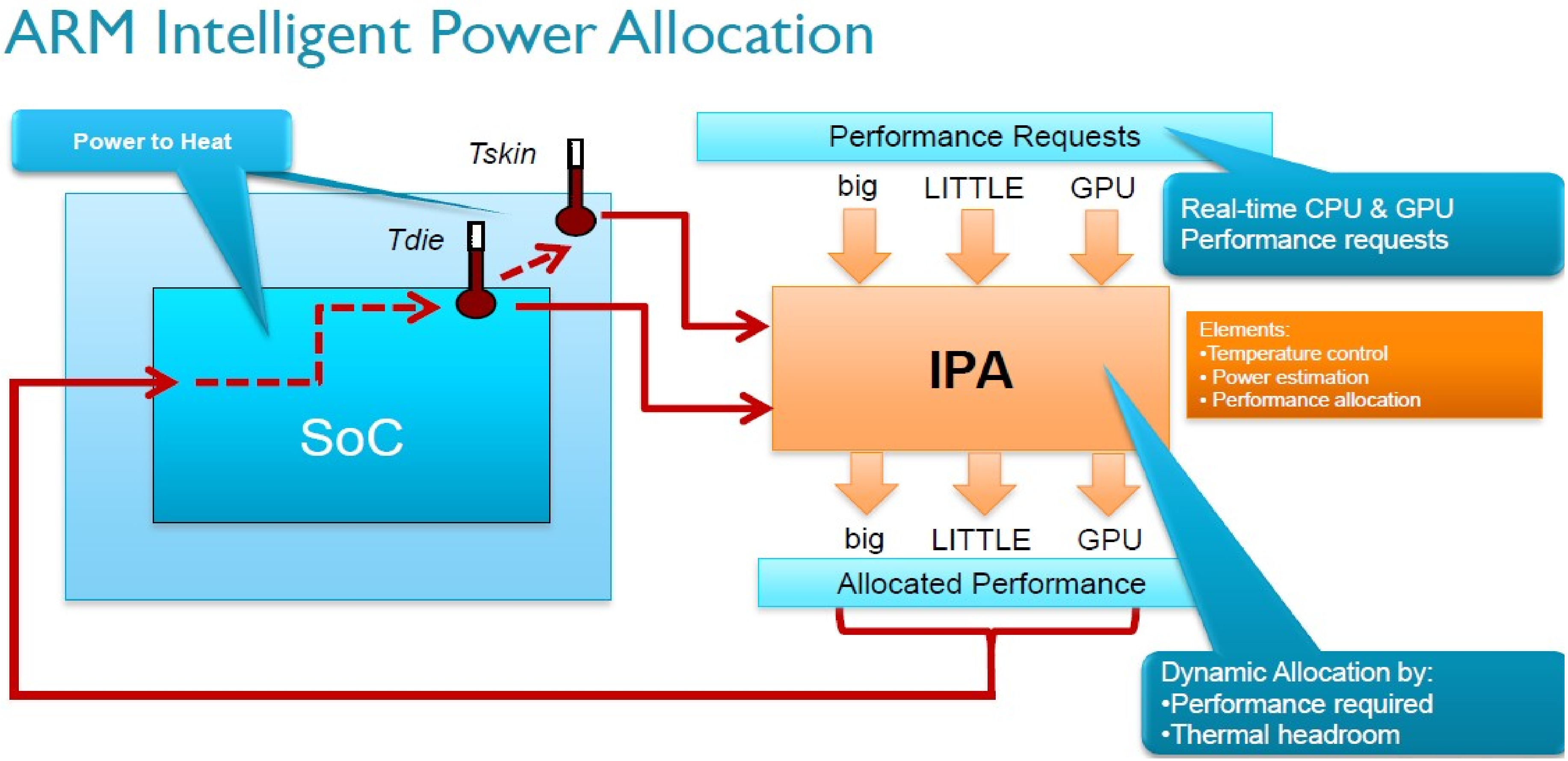


Figure: Principle of operation of IPA [56]

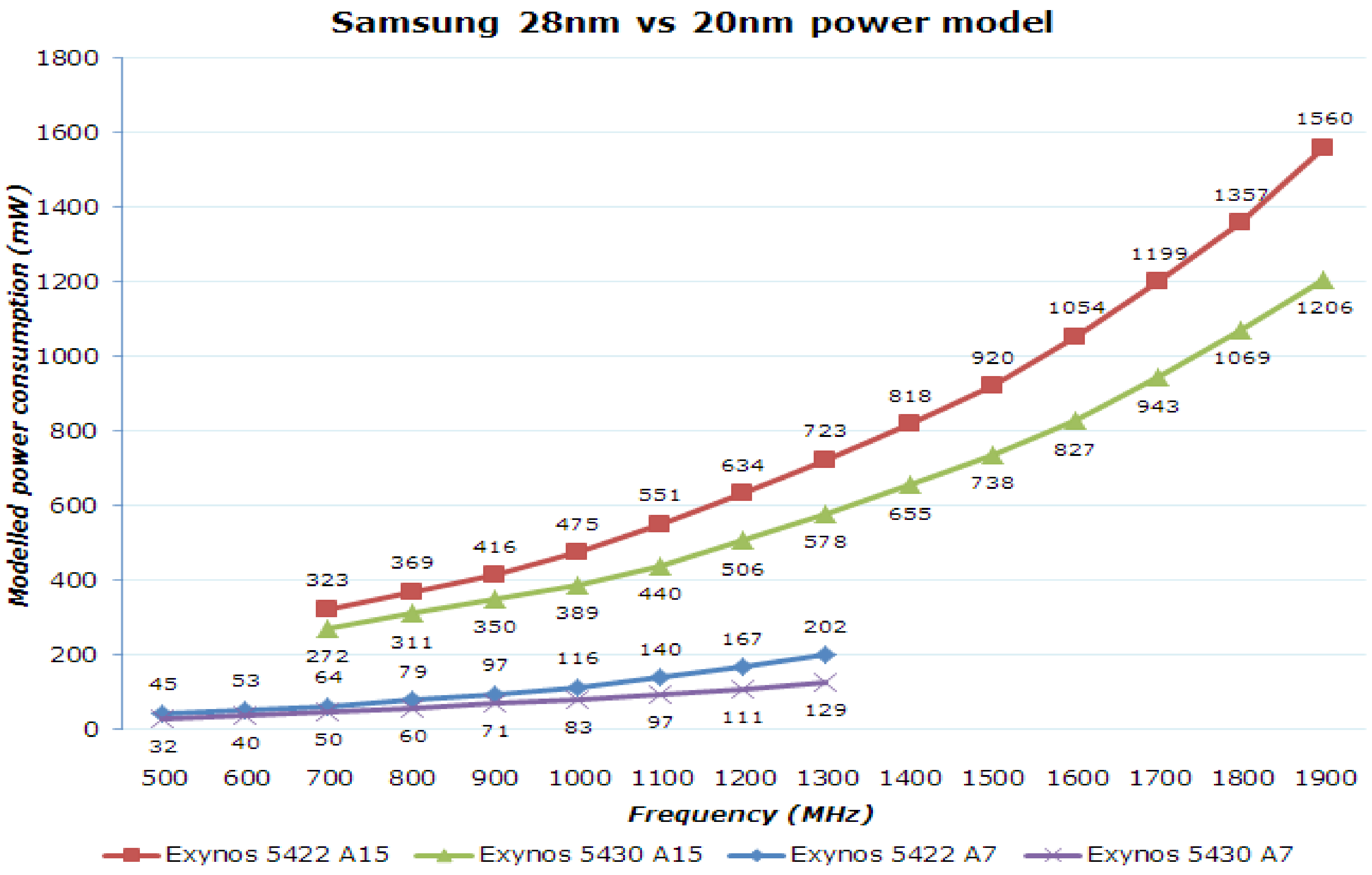
6.2 OS support for GTS provided by ARM/Linaro (11)

Principle of operation of IPA -2 [58]

- The performance requests of the threads are first converted into power requests based on a power model.
- The power model used gives an interrelationsship between the clock frequency and the related power consumption.

6.2 OS support for GTS provided by ARM/Linaro (12)

Example: Power models of the Samsung Exynos 5422 and 5433 SOCs [67]



6.2 OS support for GTS provided by ARM/Linaro (13)

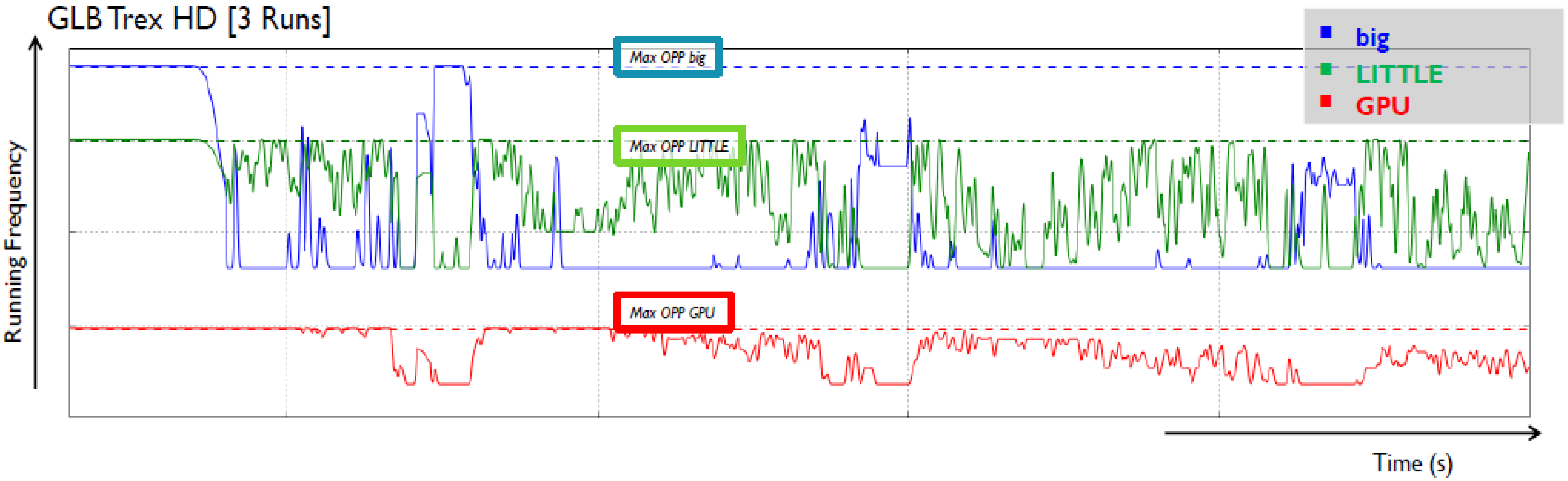
Principle of operation of IPA -3 [58]

- Based on the readings of the available temperature sensors (Tdie, Tskin), IPA estimates the available power budget.
- IPA then allocates the available power budget to each actor based on the requested performance.

6.2 OS support for GTS provided by ARM/Linaro (14)

Example: Operation of IPA [68] -1

- **IPA dynamically allocates power to CPU clusters or GPU, based on load**
- **Temperature determines available power**
  - You set Temperature, IPA caps Frequencies
- **Load determines how power is divided between CPUs (big and LITTLE) and GPU**

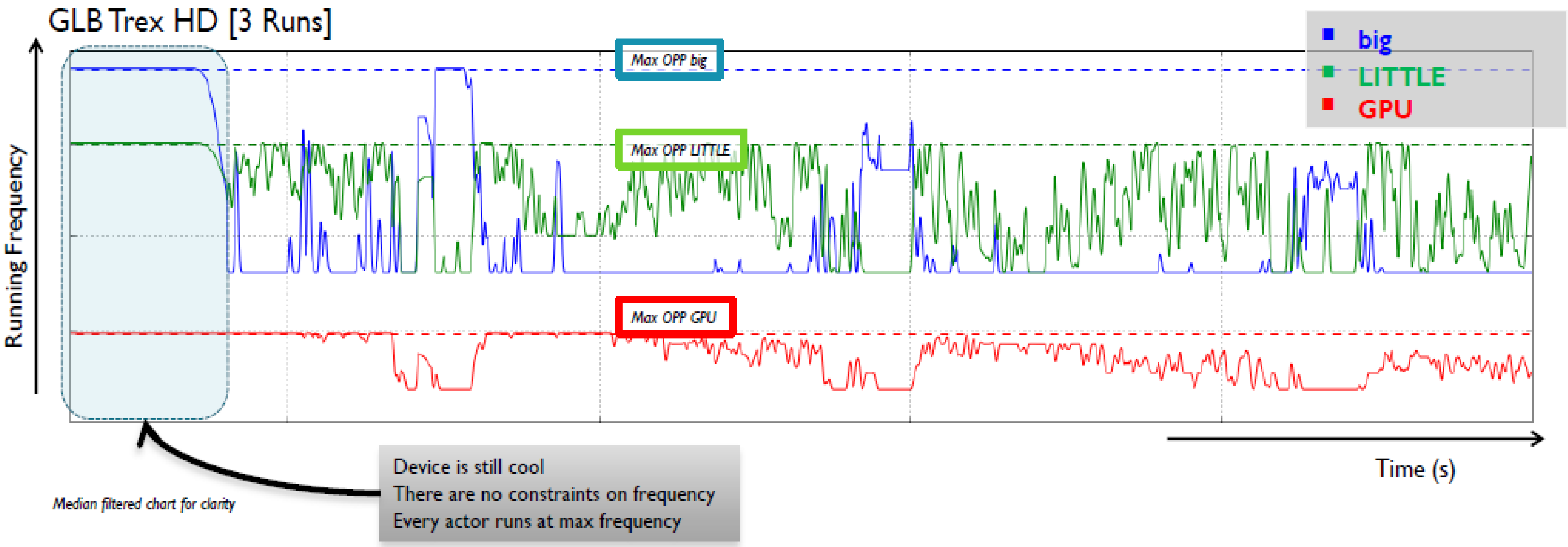


GLB T-Rex is a mobile benchmark based on OpenGL ES.
OpenGL ES is OpenGL for Embedded Systems
OpenGL is a computer graphics API (application Program Interface).

6.2 OS support for GTS provided by ARM/Linaro (15)

Example: Operation of IPA [68] -2

- **IPA dynamically allocates power to CPU clusters or GPU, based on load**
- **Temperature determines available power**
  - You set Temperature, IPA caps Frequencies
- **Load determines how power is divided between CPUs (big and LITTLE) and GPU**

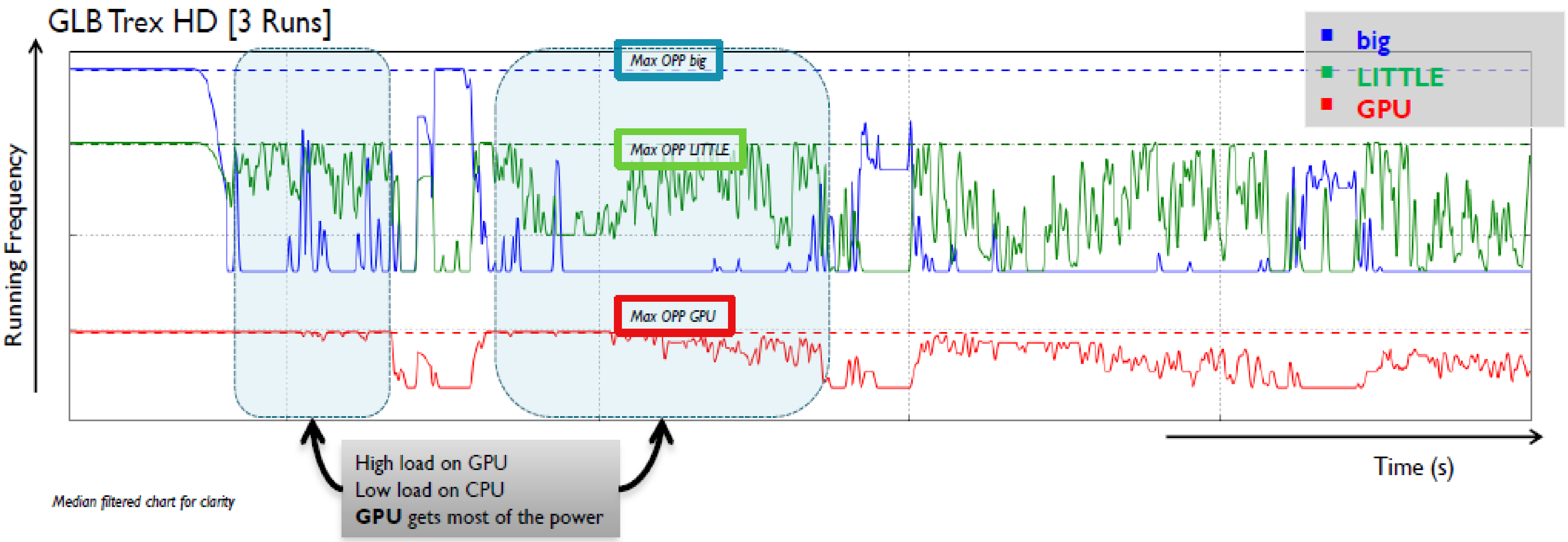


Device is still cool
There are no constraints on frequency
Every actor runs at max frequency

6.2 OS support for GTS provided by ARM/Linaro (16)

Example: Operation of IPA [68] -3

- **IPA dynamically allocates power to CPU clusters or GPU, based on load**
- **Temperature determines available power**
  - You set Temperature, IPA caps Frequencies
- **Load determines how power is divided between CPUs (big and LITTLE) and GPU**



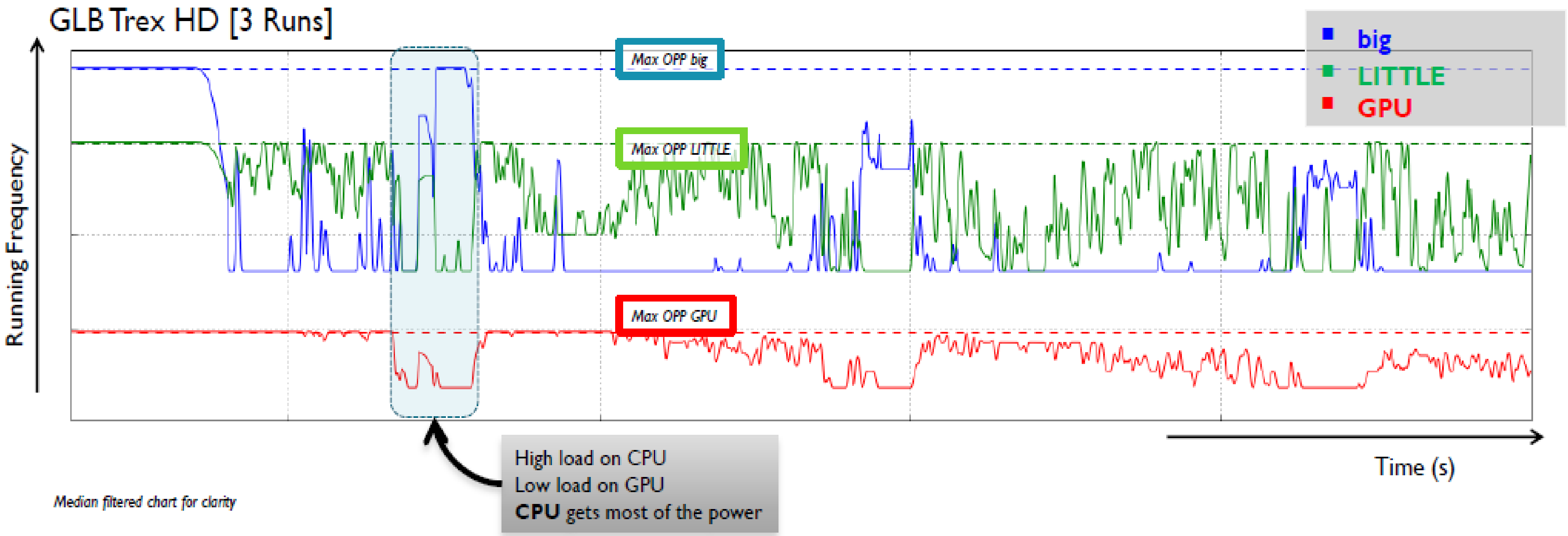
High load on GPU
Low load on CPU
GPU gets most of the power

Median filtered chart for clarity

6.2 OS support for GTS provided by ARM/Linaro (17)

Example: Operation of IPA [68] -4

- **IPA dynamically allocates power to CPU clusters or GPU, based on load**
- **Temperature determines available power**
  - You set Temperature, IPA caps Frequencies
- **Load determines how power is divided between CPUs (big and LITTLE) and GPU**



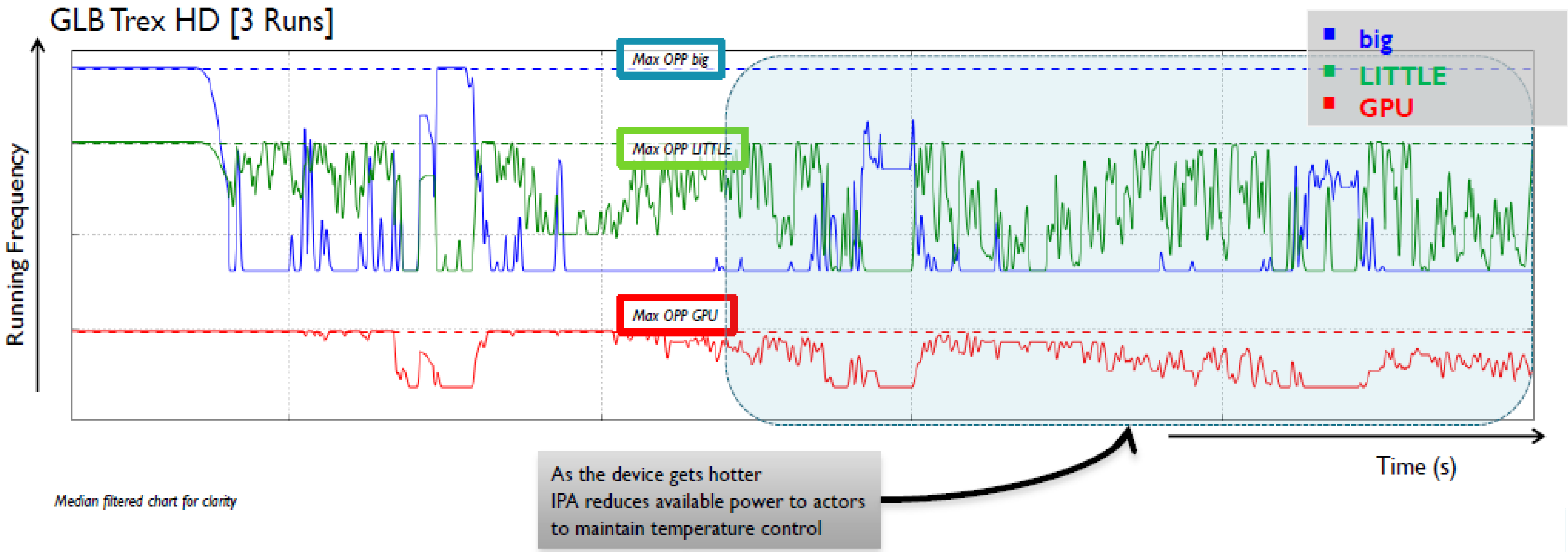
High load on CPU
Low load on GPU
CPU gets most of the power

Median filtered chart for clarity

6.2 OS support for GTS provided by ARM/Linaro (18)

Example: Operation of IPA [68] -5

- **IPA dynamically allocates power to CPU clusters or GPU, based on load**
- **Temperature determines available power**
  - You set Temperature, IPA caps Frequencies
- **Load determines how power is divided between CPUs (big and LITTLE) and GPU**



As the device gets hotter
IPA reduces available power to actors
to maintain temperature control

Median filtered chart for clarity

6.2 OS support for GTS provided by ARM/Linaro (19)

Availability and use of IPA

- As open source software IPA has been released as the big.LITTLE MP patch set from Linaro in 5/2013.

  It became later included also into mainline Linux 4.2 in 08/2015.
- It may easily be deployed in Linux-based devices including Android.
- IPA is typically used in Samsung's Exynos models, starting with the Exynos 5 5422 (2014).

6.2 OS support for GTS provided by ARM/Linaro (20)

6.2.3 Energy Aware Scheduling (EAS) from ARM and Linaro Not discussed

Motivation for the EAS project

It became recognized in 2013 that power management in Linux is implemented by different, largely uncordinated kernel routins, such as

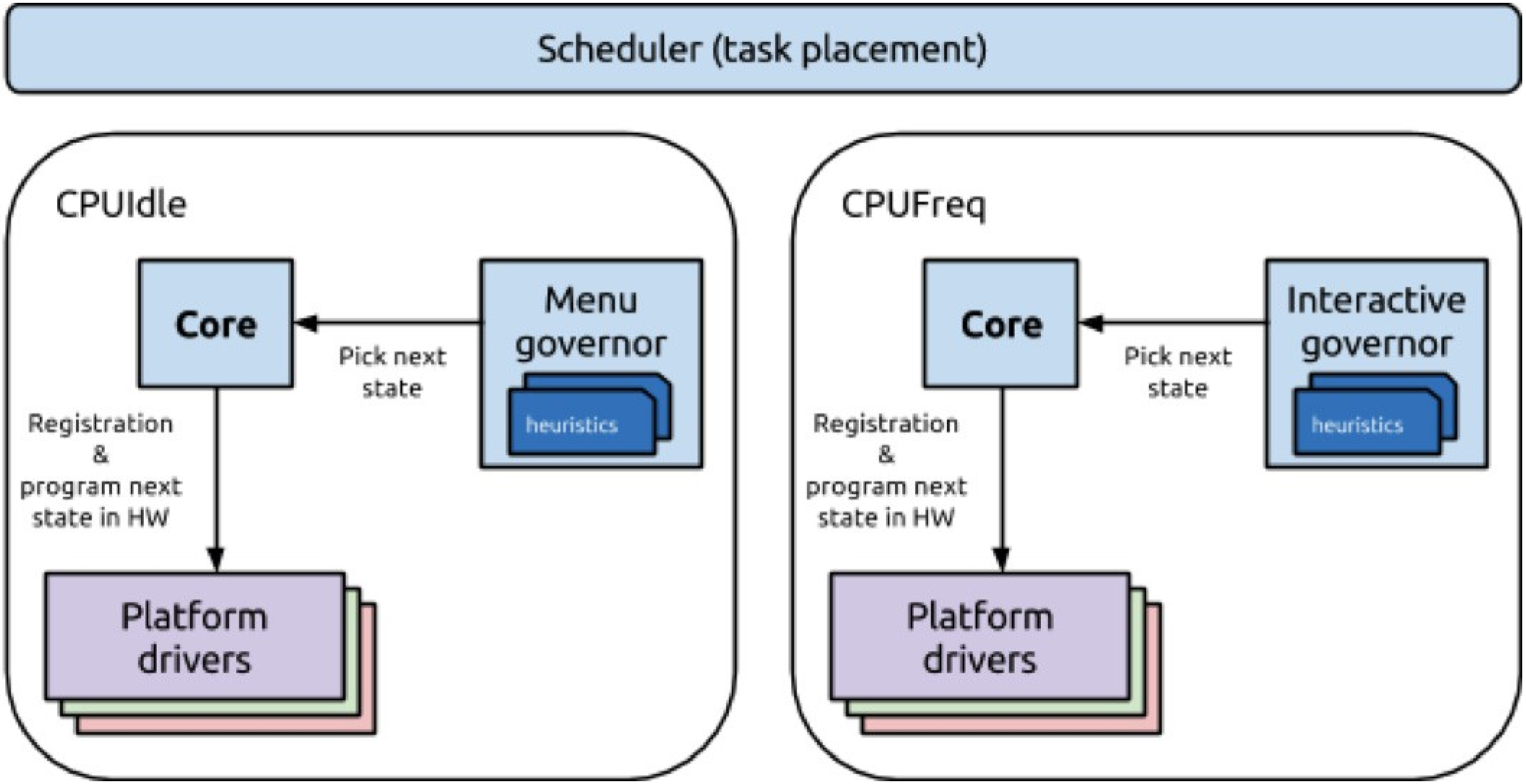
- the task scheduler
- the routine managing idle states (CPUidle) and
- the routine performing DVFS (CPUFreq),

as indicated in the next Figure.

This makes platform adaptation difficult and tuning difficult

6.2 OS support for GTS provided by ARM/Linaro (21)

Uncordinated operation of the Scheduler, CPUIdle and CPUFreq routines [59]



- As indicated in the Figure, the scheduler, CPUFreq and CPUIdle subsystems work in isolation, i.e. uncorrellated with each other.
- The scheduler tries to balance the load across all cores, unregarded the power costs, while the CPUFreq and CPUIdle subsystems are trying to save power by scaling down fc of the cores or idling them, respectively.

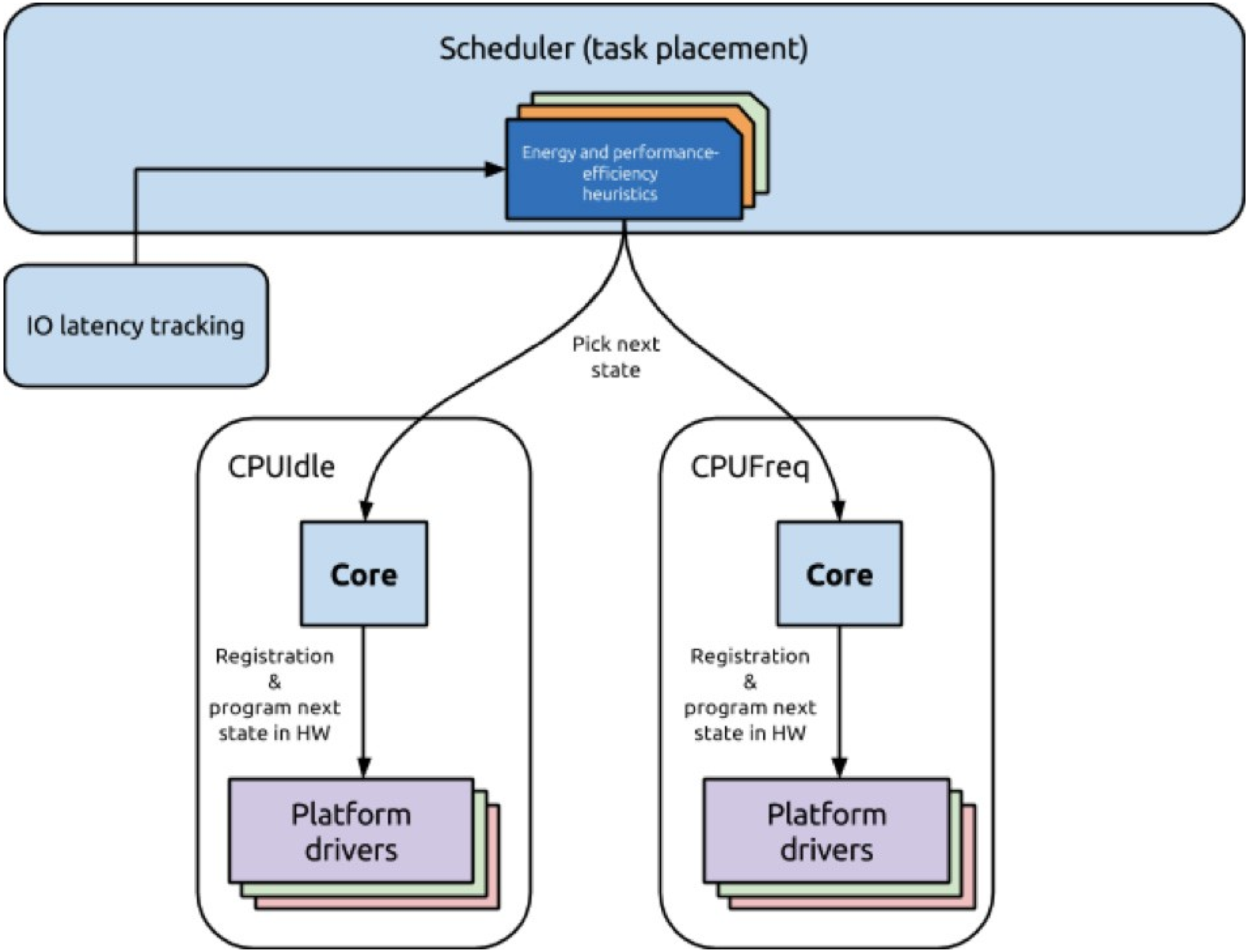
6.2 OS support for GTS provided by ARM/Linaro (22)

The aim of EAS

- EAS is aimed to provide a generic, energy aware task scheduling, based on a well grounded energy model rather than on magic tunables present in the recent power managenent framework (like upper and lower limits for migrating tasks between cores, etc..
- A generic energy model based approach is expected to support a broad range of current and future CPU topologies, including SMP, multi-cluster SMP (e.g. 8-core Cortex-A53 products), as well as traditional ARM big.LITTLE.
- In this model the scheduler directs both the CPUFreq and CPUIdle subsystems, as seen in the next Figure.

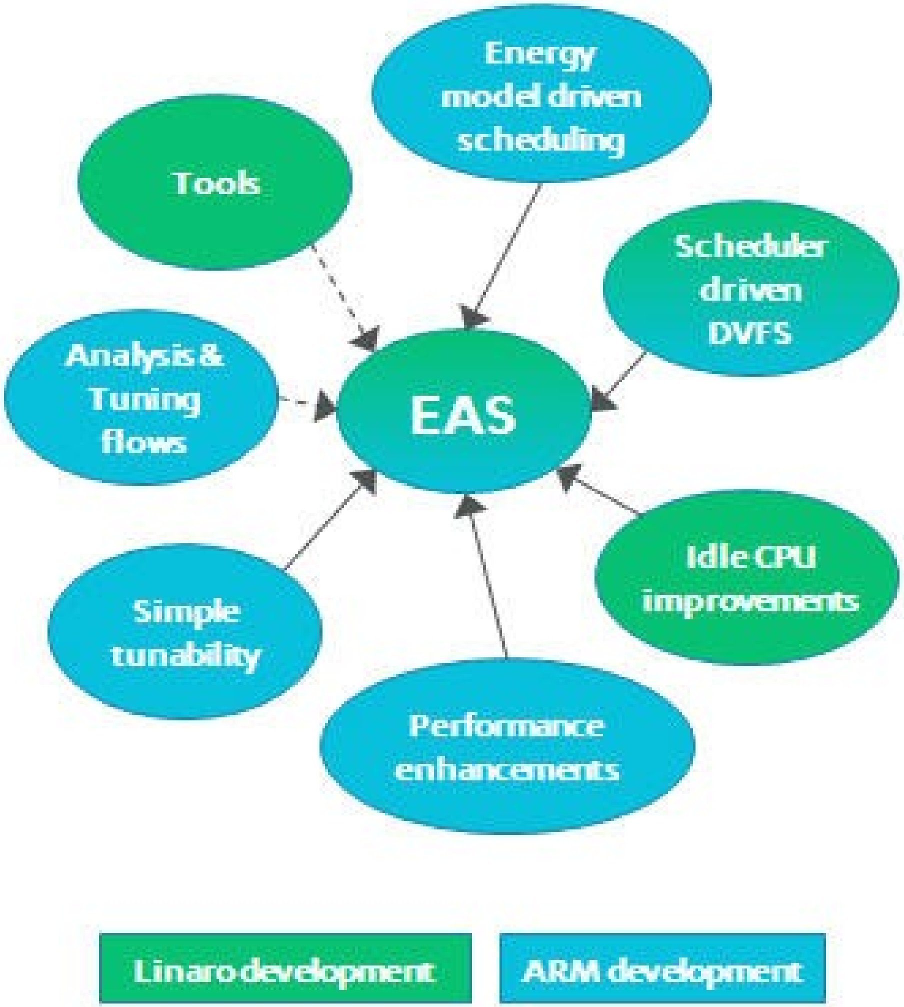
6.2 OS support for GTS provided by ARM/Linaro (23)

Coordinated operation of the scheduler, the CPUIdle and CPUFreq subsystems in EAS [59]


Scheduler (task placement)
Energy and performance-efficiency heuristics
IO latency tracking
Pick next state
CPUIdle
Core
Registration & program next state in HW
Platform drivers
CPUFreq
Core
Registration & program next state in HW
Platform drivers

6.2 OS support for GTS provided by ARM/Linaro (24)

Joint development of EAS subsystems by ARM and Linaro [59]



6.2 OS support for GTS provided by ARM/Linaro (20)

Status of the EAS development

At present (11/2015) EAS development is in progress yet.

6.3 MediaTek’s CorePilot releases

6.3 MediaTek's CorePilot releases (1)

6.3 Mediatek's CorePilot releases

	2013	2014	2015
ARM/Linaro	ARM big.LITTLE MP (Global Task Scheduling) (~06/2013)	ARM IPA (Inteligent Power Allocation) (10/2014)	ARM/Linaro EAS (Energy Aware Scheduling) (development yet in progress)
MediaTek	MediaTek CorePilot 1.0 (on MT8135) (07/2013)		MediaTek CorePilot 2.0 (on Helio X10 (MT6595) (03/2015) MediaTek CorePilot 3.0 (on Helio X20 (MT6797) (05/2015)
Qualcomm		Qualcomm's Energy Aware Scheduling (on Snapdragoon 610/615 (02/2014))	Qualcomm Symphony System Manager (on Snapdragoon 820) (11/2015)
Samsung	Samsung's big.LITTLE HMP (≈ ARM's big.LITTLE MP) (on Exynos 5 models) (09/2013)		

6.3 MediaTek's CorePilot releases (2)

6.3.1 CorePilot (1.) [33]

- It is based on the open-source GTS technology, designated big.LITTLE MP, developed by ARM.

  It was introduced along with the MT8135 in 07/2013.

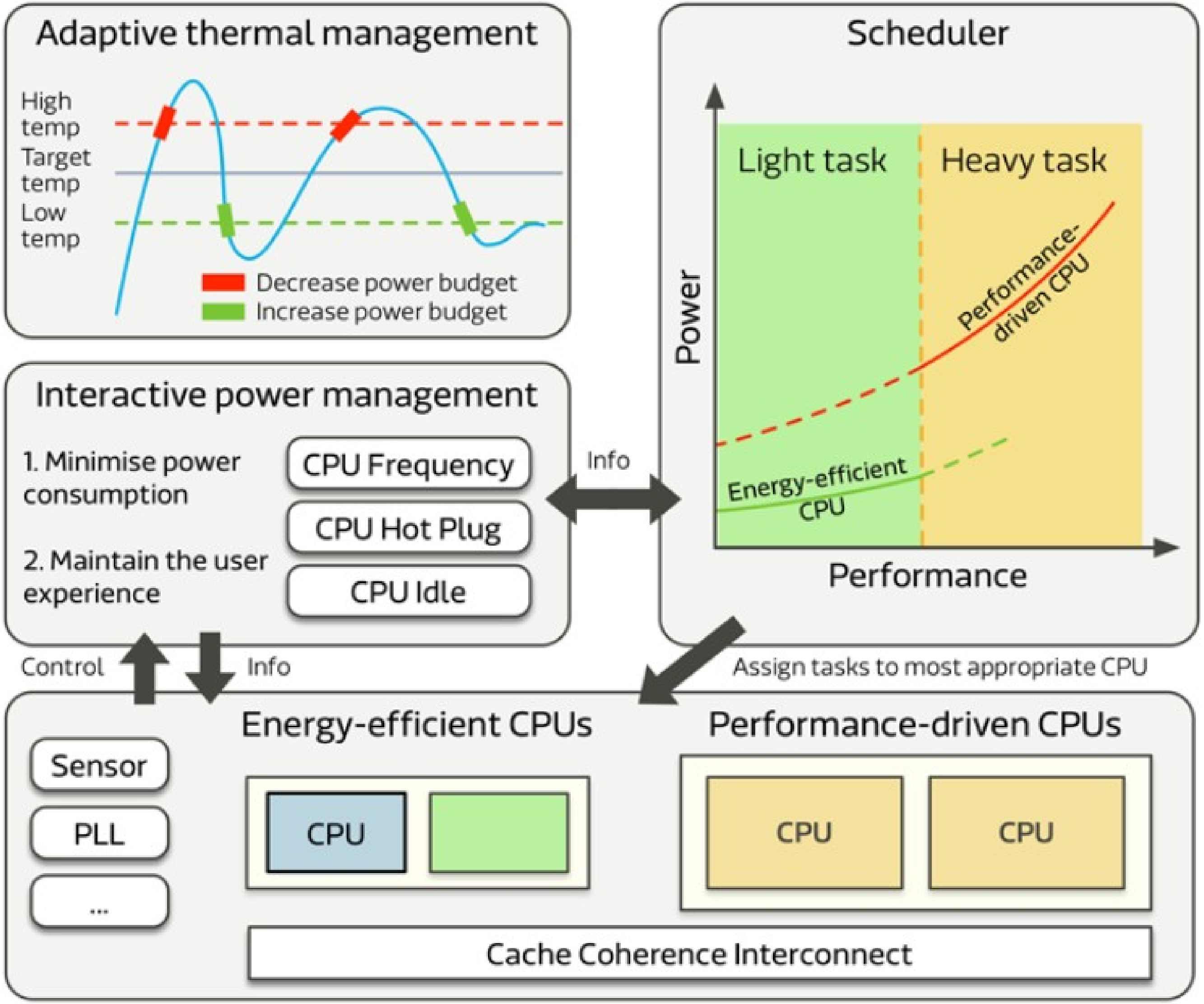
- CorePilot integrates thermal and power management into task scheduling and consists accordingly of the following three parts:

  - adaptive thermal management
  - interactive power management and
  - advanced scheduler algorithms,

  as indicated in the next Figure.

6.3 MediaTek's CorePilot releases (3)

Overview of the operation of CorePilot [33]



6.3 MediaTek's CorePilot releases (4)

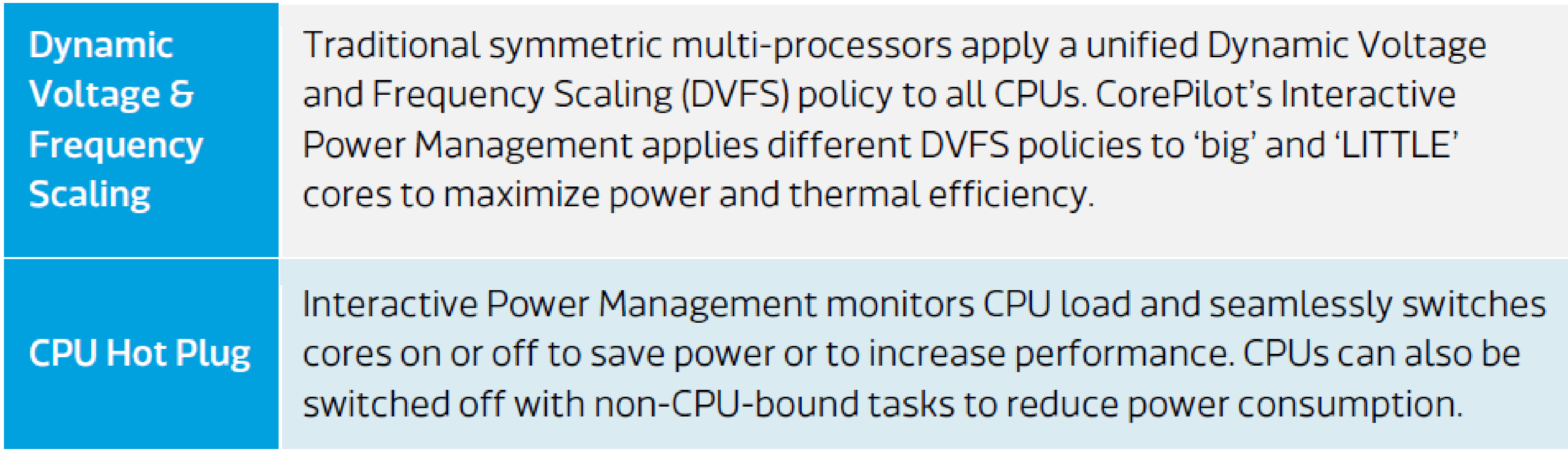
a) Adaptive Thermal Management [37]

- Power, thermal and performance (PTP) detectors are placed within the CPU to sense operating conditions.
- Adaptive Thermal Management monitors sensed data and allows the device to use the available voltage margin either to increase performance or lower power consumption when possible.
- According to MediaTek this technology allows for a 23% increase in clock speed or up to 41% power savings, depending on the SoC operating conditions.

6.3 MediaTek's CorePilot releases (5)

b) Interactive Power Management [33]

- CorePilot’s Interactive Power Manager reduces the amount of power and heat generated by the cores via two main modules.
- The DVFS (Dynamic Voltage and Frequency Scaling) module automatically adjusts the frequency and voltage of cores on the fly, while the CPU Hot Plug module switches cores on and off on demand, as summarized below.



Dynamic Voltage & Frequency Scaling	Traditional symmetric multi-processors apply a unified Dynamic Voltage and Frequency Scaling (DVFS) policy to all CPUs. CorePilot’s Interactive Power Management applies different DVFS policies to ‘big’ and ‘LITTLE’ cores to maximize power and thermal efficiency.
CPU Hot Plug	Interactive Power Management monitors CPU load and seamlessly switches cores on or off to save power or to increase performance. CPUs can also be switched off with non-CPU-bound tasks to reduce power consumption.

6.3 MediaTek's CorePilot releases (6)

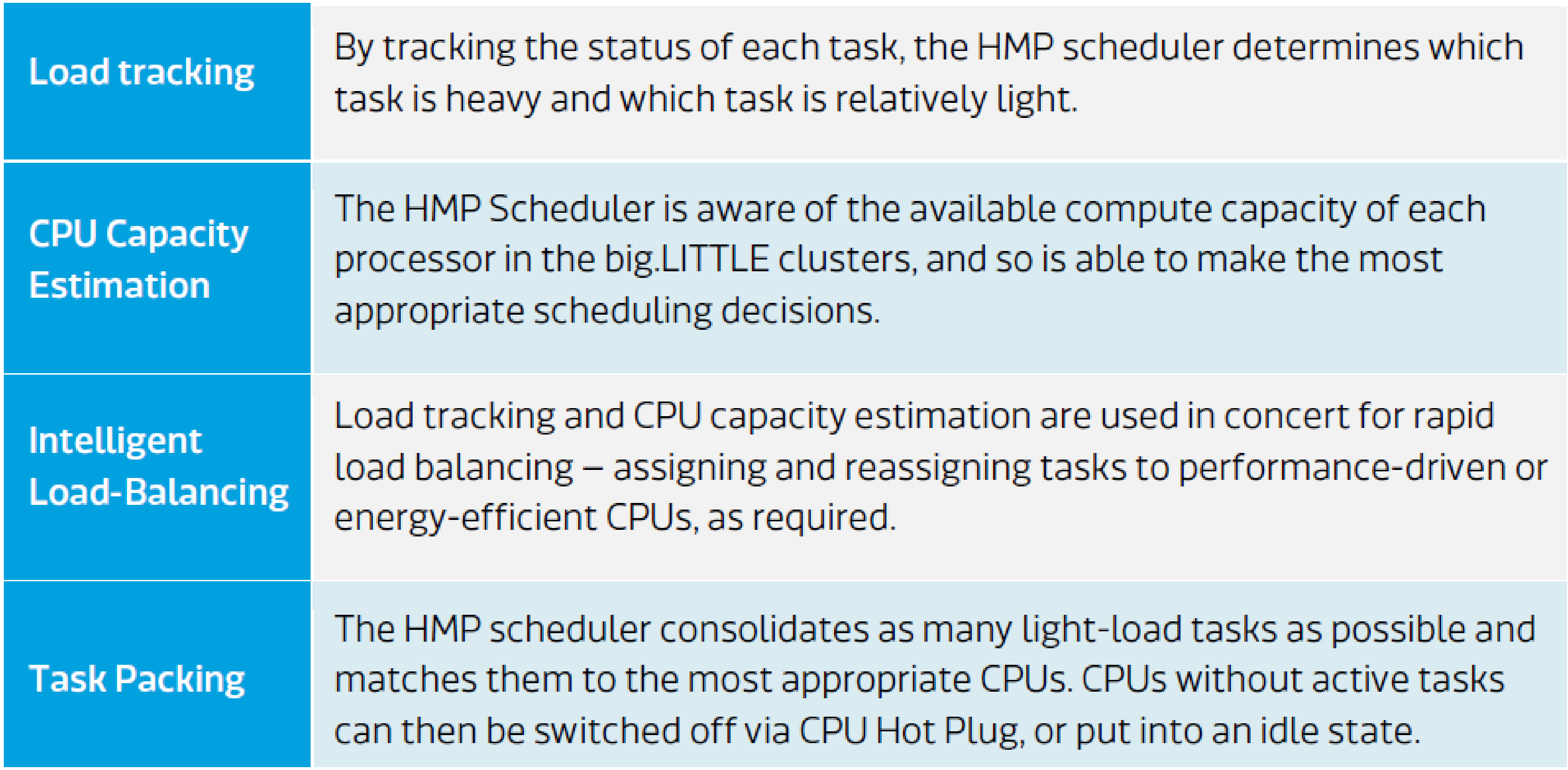
c) Scheduler Algorithms [33]

- With Symmetric Multi-Processing (SMP), the Completely Fair Scheduler (CFS) of the Linux kernel implements currently the most common scheduling algorithm, it distributes the workload equally among CPU cores.
- In case of Heterogeneous Multi-Processing, however, employing CFS can cause performance degradation, since tasks do not efficiently match to CPU core capabilities.
- MediaTek’s CorePilot, on the other hand, is based on a true heterogeneous compute model by using a scheduling algorithm that assigns tasks to two different schedulers, according to their priority — the Heterogeneous Multi-Processing (HMP) scheduler and the Real-Time (RT) scheduler.

6.3 MediaTek's CorePilot releases (7)

MediaTek’s HMP Scheduler

It is responsible for assigning normal-priority tasks to the big.LITTLE CPU core clusters and performs four main functions, as follows.



Load tracking	By tracking the status of each task, the HMP scheduler determines which task is heavy and which task is relatively light.
CPU Capacity Estimation	The HMP Scheduler is aware of the available compute capacity of each processor in the big.LITTLE clusters, and so is able to make the most appropriate scheduling decisions.
Intelligent Load-Balancing	Load tracking and CPU capacity estimation are used in concert for rapid load balancing – assigning and reassigning tasks to performance-driven or energy-efficient CPUs, as required.
Task Packing	The HMP scheduler consolidates as many light-load tasks as possible and matches them to the most appropriate CPUs. CPUs without active tasks can then be switched off via CPU Hot Plug, or put into an idle state.

Figure: Key components of MediaTek’s HPM scheduler [33]

6.3 MediaTek's CorePilot releases (8)

MediaTek's RT Scheduler [33]

- The RT scheduler assigns high-priority real-time tasks that require a fast response to the big.LITTLE cluster.
- The RT scheduler has a higher priority than the HMP scheduler.
- MediaTek has modified its design such that the highest priority tasks are assigned to high performance CPU cores whereas lesser priority real-time tasks to other available CPU cores.

6.3 MediaTek's CorePilot releases (9)

First use cases of MediaTek's CorePilot in big.LITTLE configurations

- MT8135 2xCortex-A15 + 2x Cortex-A7 (7/2013) for Android tablets
  World's first big.LITTLE chipset with inclusive core migration
- MT6592 8x Cortex-A7 (Q4/2013) for smartphones
  World's first octa core symmetrical multicore chipset with HSPA+ connectivity
- MT6595 4x Cortex-A17 + 4x Cortex-A7 (7/2014) for smartphones
  World's first octa core big.LITTLE 4G LTE chipset with inclusive core migration

6.3 MediaTek's CorePilot releases (10)

6.3.2 CorePilot 2.0

- Introduced along with MediaTek's first 64-bit SOC, the Helio X10 (MT6795) in 3/2015.
- It extends the scope of the scheduler also to the GPU by including the Device Fusion technology.
- With the Device Fusion technology CorePilot 2.0 decides which task will perform better on which computing device and dispatches workloads expressed in OpenCL to the suitable computing device (CPU cores or GPU) or to both types, as shown below.

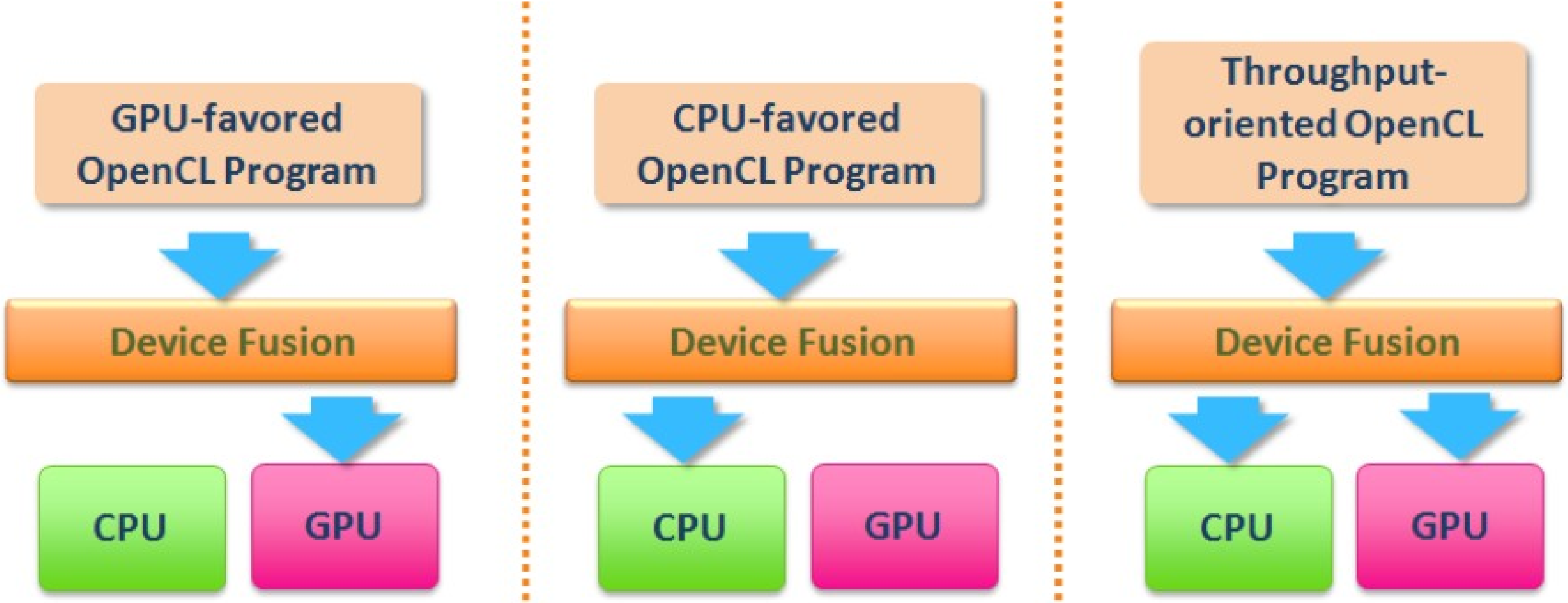

GPU-favored OpenCL Program
Device Fusion
CPU
GPU
CPU-favored OpenCL Program
Device Fusion
CPU
GPU
Throughput-oriented OpenCL Program
Device Fusion
CPU
GPU

Figure: Dispatch options in the Deive Fusion technology [60]

6.3 MediaTek's CorePilot releases (11)

Expected benefits of CorePilot 2.0

MediaTek states that CorePilot promises up to 146% performance increase and up to 18 % lower energy consumtion when compared to using CPU or GPU only architectures [60].

6.3 MediaTek's CorePilot releases (12)

6.3.3 CorePilot 3.0 [61]

- Introduced along with MediaTek's first three cluster SOC, the Helio X20 (MT6797) in 05/2015.
- CorePilot 3.0 enhances the scheduler to cope with three clusters of CPU cores as well as with the GPU while managing related power and temperature issues, as before (see the subsequent Figures).

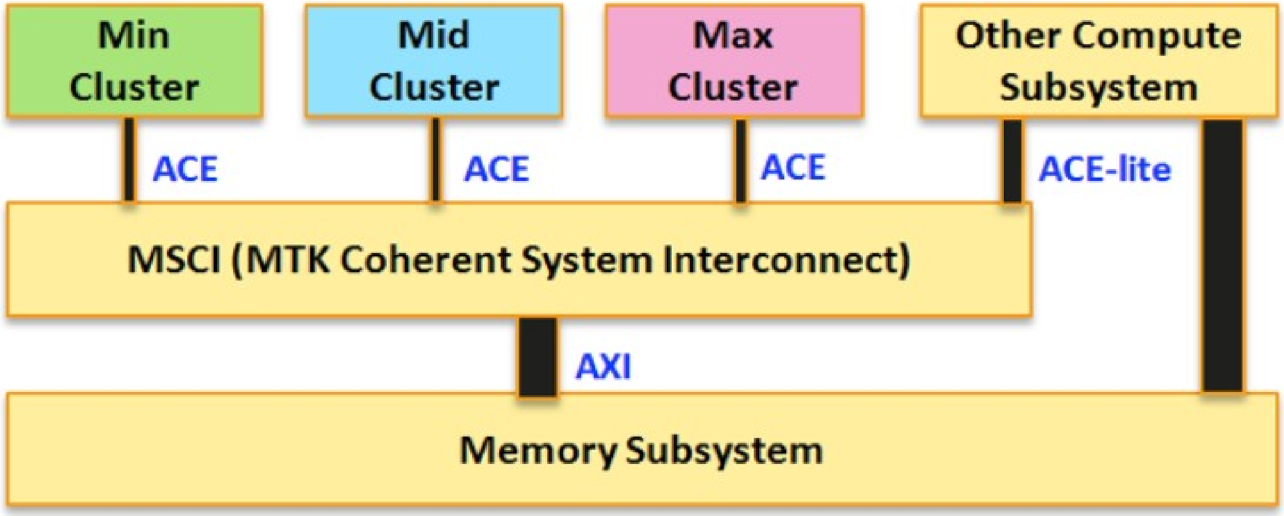
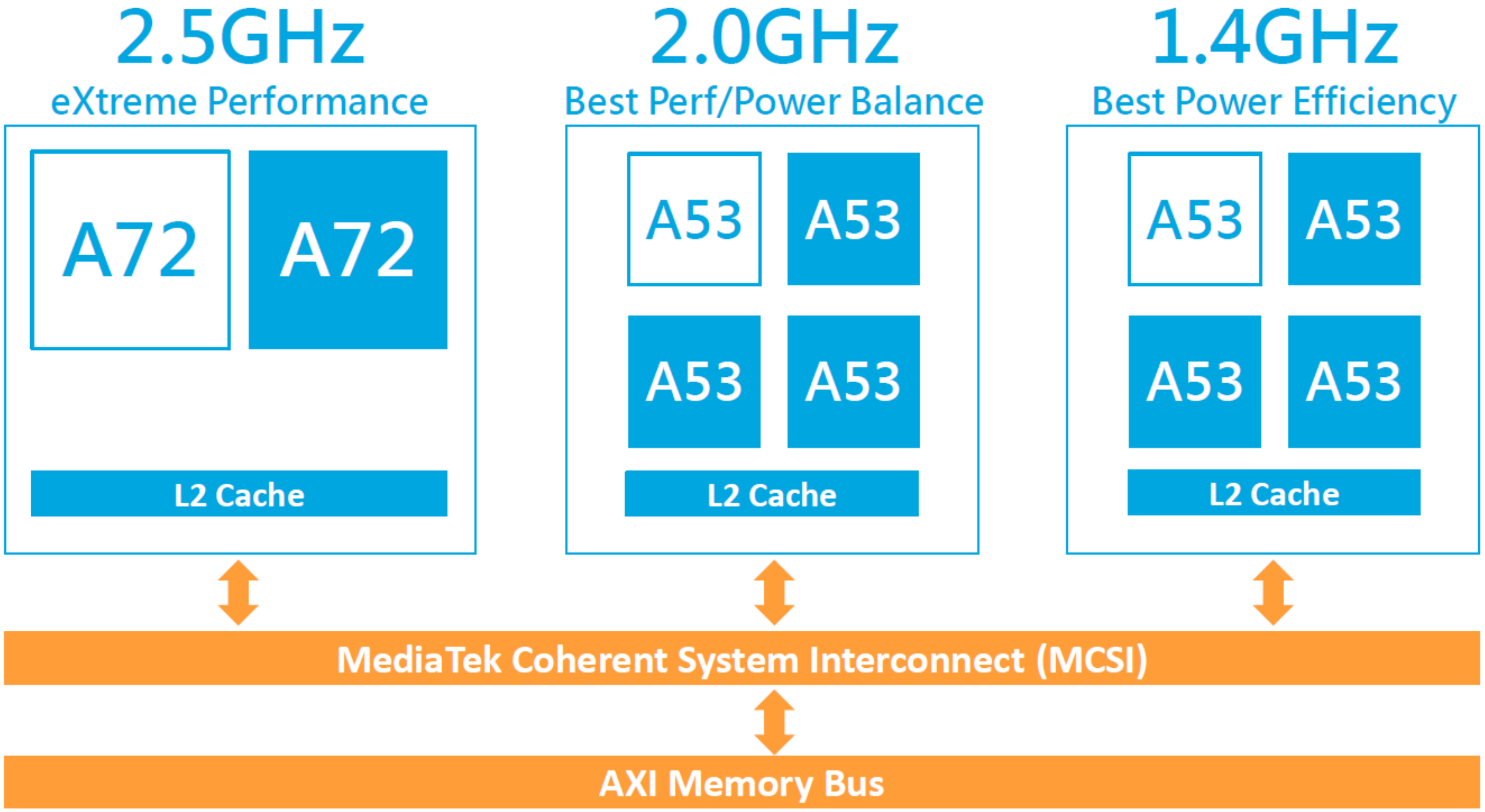


Figure MediaTek's three cluster big.LITTLE architecture [47]

6.3 MediaTek's CorePilot releases (13)

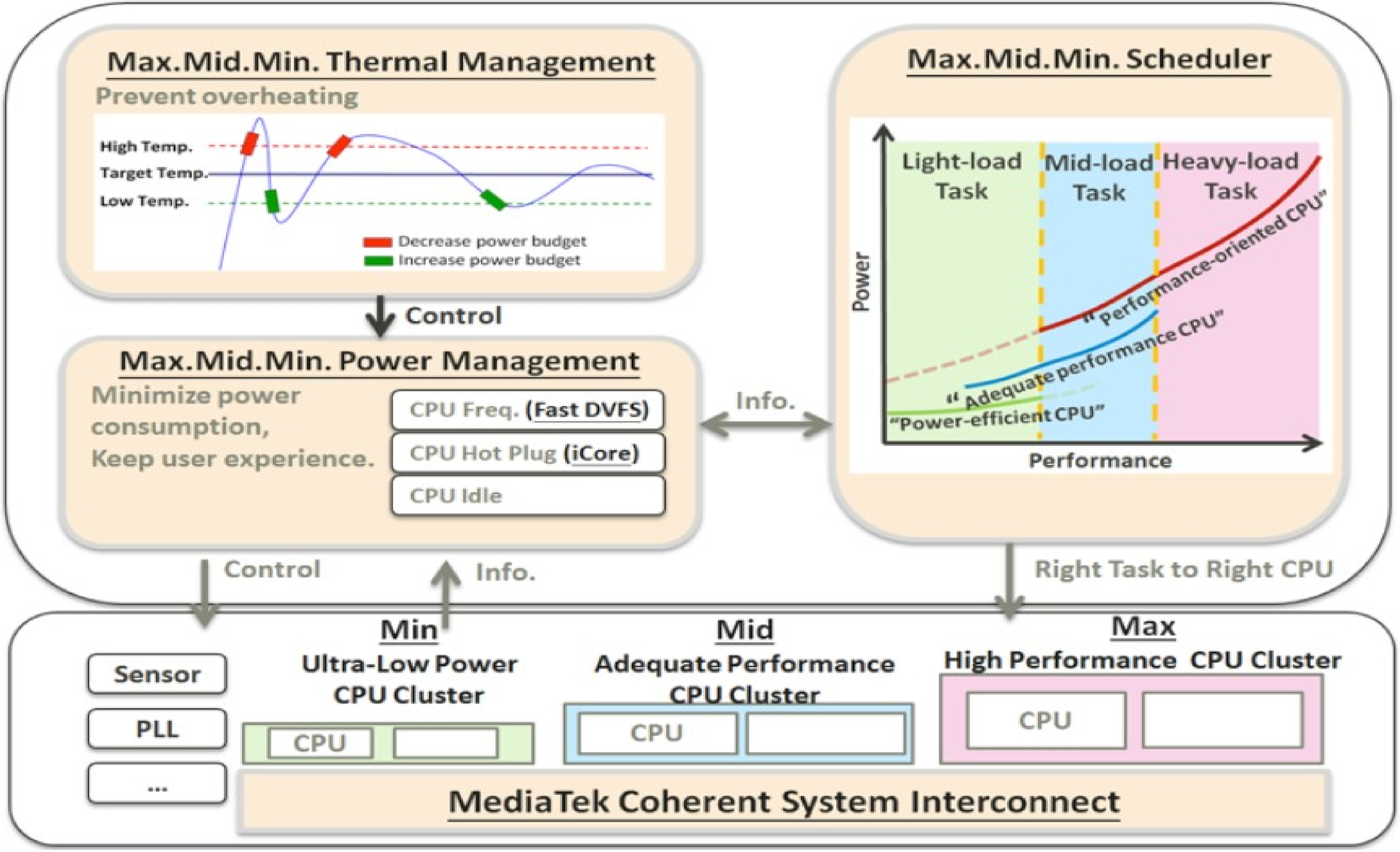
First implementation of MediaTek's 10 core (deka core) processor (the Helio X20 (MT6797)) [46]

Announced in 05/2015, first apppearance in smartphones in Q4/2015.



6.3 MediaTek's CorePilot releases (14)

Block diagram of CorePilot 3.0 [47]



6.3 MediaTek's CorePilot releases (15)

Principle of operation of CorePilot 3.0 [47]

- CorePilot 3.0 periodically calculates the allowable power budget of the SOC components based on temperature data and form factor (e.g. smartphone) heat up conditions and allocates it to the CPU cores available in three clusters and the GPU in a similar way, as before.
- In addition, thermal management prevents temperature spikes by proactively predicting temperature-rise bursts and limiting the temperature-rise speed.
- Furthermore, CorePilot 3.0 introduces the Fast DVFS technology by increasing the samplinng rate up to 40 times.

6.3 MediaTek's CorePilot releases (16)

The Fast DVFS technology

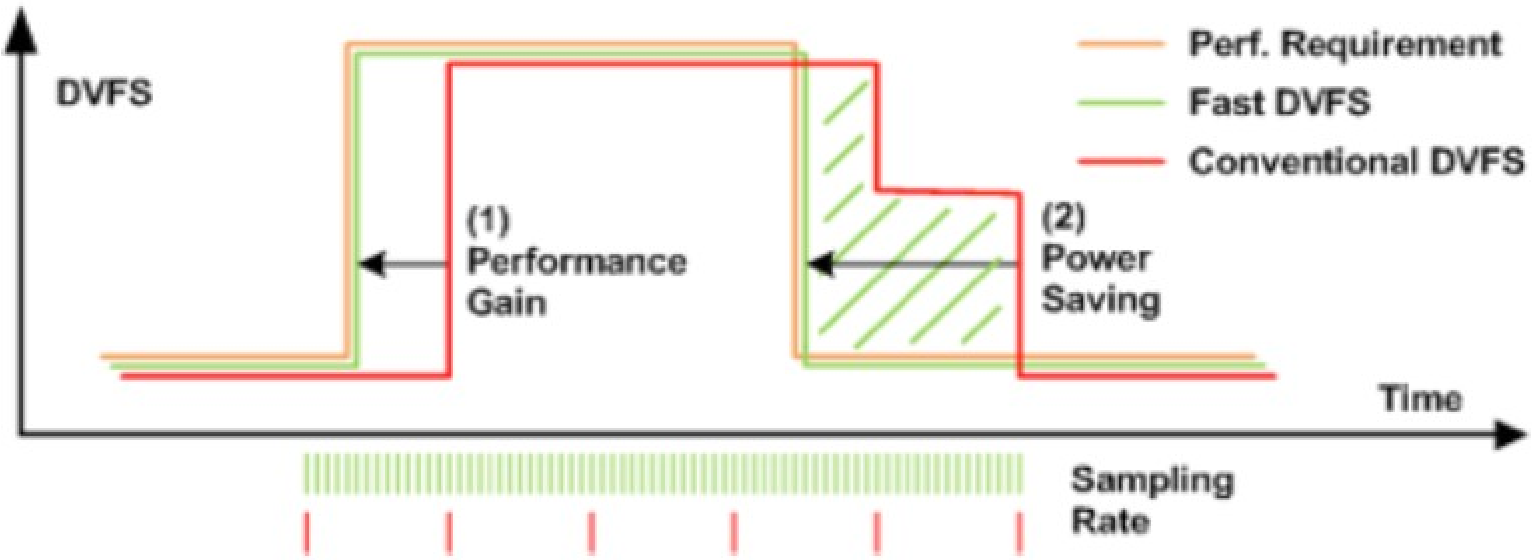


Figure: Benefits of the Fast DVFS technology [47]

Fast DVFS technology more rapidly increases clock frequency if needed to execute higher workload - providing better responsiveness, and more swiftly reduces clock frequency - if workload decreases - that results in power saving, as the above figure demonstrates.

6.4 Qualcomm's big.LITTLE schedulers

6.4 Qualcomm's big.LITTLE schedulers (1)

6.4 Qualcomm's big.LITTLE schedulers

ARM/Linaro	ARM big.LITTLE MP (Global Task Scheduling) (~06/2013)	ARM IPA (Inteligent Power Allocation) (10/2014)	ARM/Linaro EAS (Energy Aware Scheduling) (development yet in progress)
MediaTek	MediaTek CorePilot 1.0 (on MT8135) (07/2013)		MediaTek CorePilot 2.0 (on Helio X10 (MT6595) (03/2015) MediaTek CorePilot 3.0 (on Helio X20 (MT6797) (05/2015)
Qualcomm		Qualcomm’s Energy Aware Scheduling (on Snapdragoon 610/615 (02/2014))	Qualcomm Symphony System Manager (on Snapdragoon 820) (11/2015)
Samsung	Samsung's big.LITTLE HMP (≈ ARM's big.LITTLE MP) (on Exynos 5 models) (09/2013)		
	2013	2014	2015

6.4 Qualcomm's big.LITTLE schedulers (2)

6.4.1 Qualcomm's Power aware scheduler

- It was introduced with the Snapdragoon 610/615 processors in 2014.
- It is an enhancement of the ARM/Linaro GTS.
- Qualcomm's Power aware scheduler covers three tasks, as follows:

  a) load tracking

  b) power module

  c) hmp scheduler

6.4 Qualcomm's big.LITTLE schedulers (3)

a) Load tracking

- Tracking CPU demand is critical for an efficient scheduling.
- GTS determines per task CPU demand by tracking CPU load in the N most recent non-empty windows (for e.g. N = 5 with a window size of 20 ms) and calculates the CPU load by decaying subsequent CPU loads e.g. by geometric weights of $1/2^i$.
- The load calculation will be performed according to given policies, e.g. max. battery life, etc.

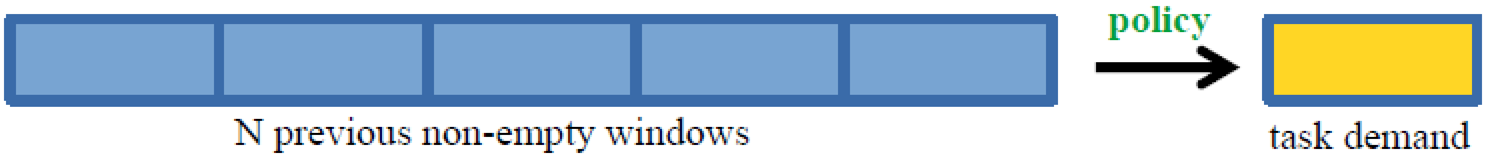


Figure: Principle of calculating task loads in N-subsequent windows in GTS [62]

- The drawback of this kind of load tracking is
  - too long ramp-up time for cpu-bound tasks and
  - too long decay time for idle tasks.
- For this reason Qualcomm modified load tracking as follows.

6.4 Qualcomm's big.LITTLE schedulers (4)

Load tracking in Qualcomm's Energy Aware Scheduler

Qualcomm's Energy Aware Scheduler does not make use of decaying loads measured in the windows, but calculates loads according to a number of policies, as indicated in the next Figure.

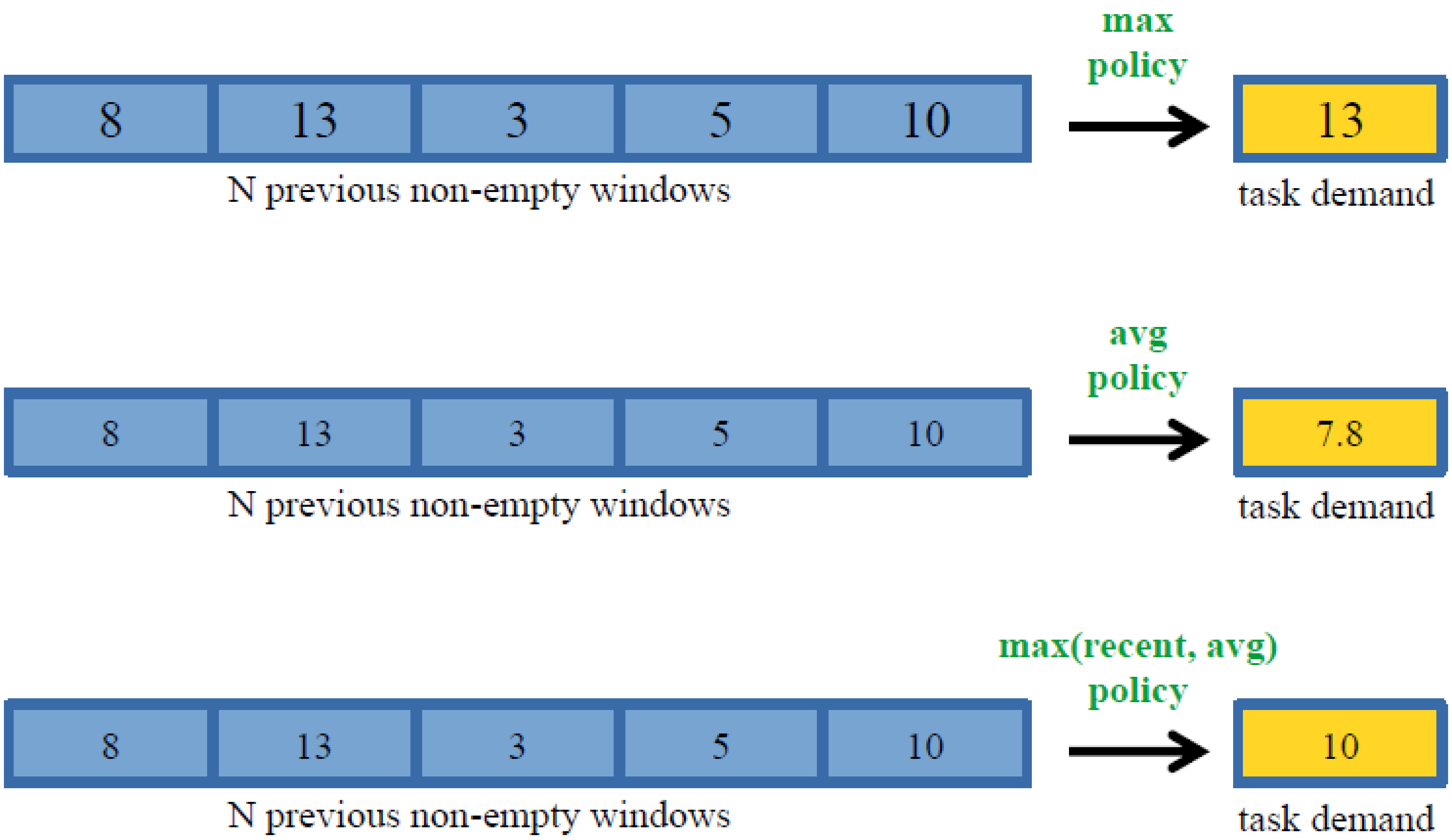
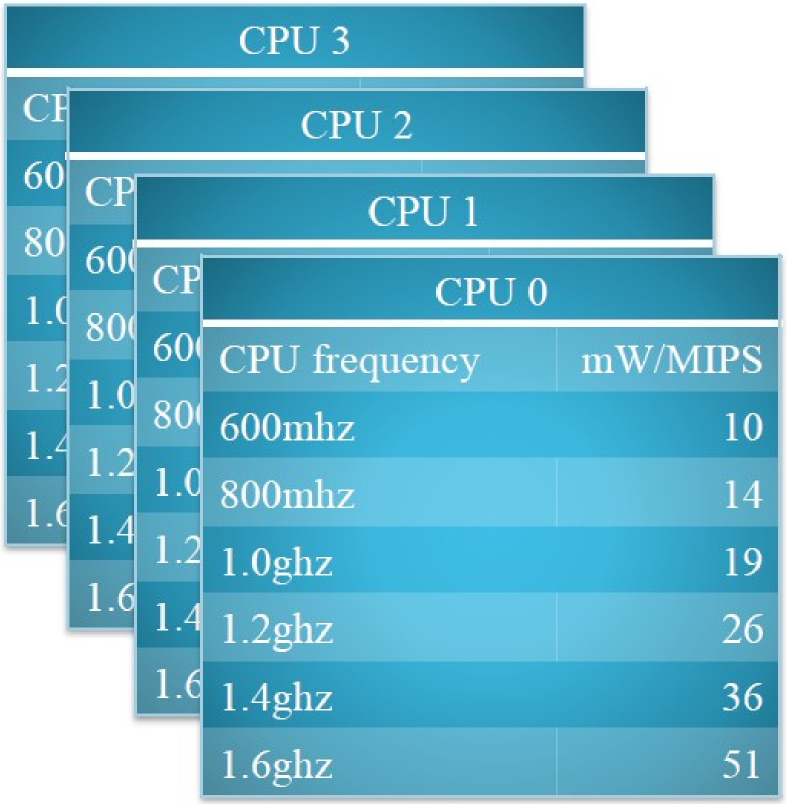


Figure: Load tracking in Qualcomm's Energy Aware Scheduler [62]

6.4 Qualcomm's big.LITTLE schedulers (5)

b) Power model

This model provides the interrelationsship between the core frequency and the execution efficiency in terms of mW/MIPS, as shown below.



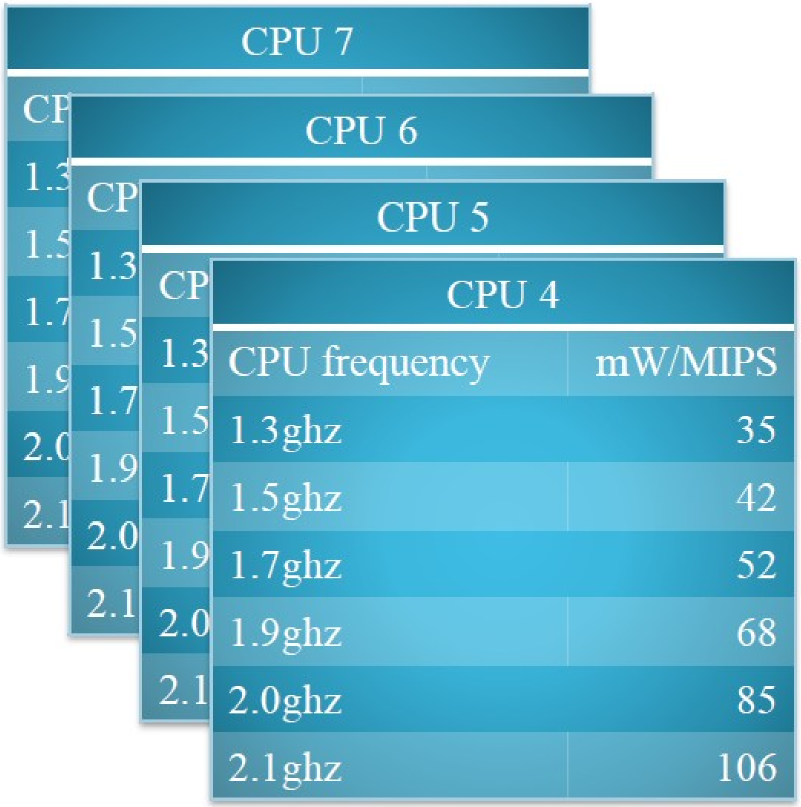
CPU 3

CPU 2

CPU 1

CPU 0

CPU frequency	mW/MIPS
600mhz	10
800mhz	14
1.0ghz	19
1.2ghz	26
1.4ghz	36
1.6ghz	51



CPU 7

CPU 6

CPU 5

CPU 4

CPU frequency	mW/MIPS
1.3ghz	35
1.5ghz	42
1.7ghz	52
1.9ghz	68
2.0ghz	85
2.1ghz	106

Figure: Power model in Qualcomm's Energy Aware Scheduler [62]

6.4 Qualcomm's big.LITTLE schedulers (6)

c) The hmc scheduler

Based on the available information on computing demand, energy impact on core performance etc. the hmc scheduler allocates tasks to the cores and migrates the tasks between cores if necssary.

6.4 Qualcomm's big.LITTLE schedulers (7)

6.4.2 Qualcomm's Symphony System Manager (SSM) -1 [63]

- It was introduced along with the Kryo core based Snapdragoon 820 in 11/2015.
- The Snapdragoon 820 is built up as a quad core big.LITTLE configuration including
  - 2 Kryo cores running at up to 2.2 GHz and
  - 2 Kryo cores running at up to 1.7 GHz.
- The 820 uses Qualcomm's SSM as an intelligent resource manager that spreads control of task scheduling and power management to the entire chip.

6.4 Qualcomm's big.LITTLE schedulers (8)

Qualcomm's Symphony System Manager (SSM) -2 [64]

- SSM achieves this by scheduling tasks to the right computing resources, such as (big or LITTLE Kryo cores, GPU or accelerators) by taking into account the required load and needed energy consumption to perform it.
- For example, when a user is taking a picture, SSM powers up the right components (CPU, Spectra ISP, GPU and memory system) and directs them to run at the needed frequency as long as required.
- In this way SSM chooses the most efficient and effective combination of cores and accelerators to perform a task as quickly as possible, with the least power consumption.
- At the time of writing this Section no detailed information is available of SSM.

7. References

7. References (1)

[1]: ARM, big.LITTLE Technology, 2014, http://www.arm.com/products/processors/technologies/biglittleprocessing.php

[2]: Flautner K., Heterogeneity to the rescue, 2011, https://www.bscmsrc.eu/sites/default/files/media/arm-heterogenous-mp-november-2011.pdf

[3]: Greenhalgh P., Big.LITTLE Processing with ARM Cortex-A15 & Cortex-A7, White Paper, Sept. 2011, http://www.arm.com/files/downloads/big.LITTLE_Final.pdf

[4]: Variable SMP – A Multi-Core CPU Architecture for Low Power and High Performance, Nvidia, Whitepaper, 2011, http://www.nvidia.com/content/PDF/tegra_white_papers/tegra-whitepaper-0911b.pdf

[5]: Klug B., Samsung Announces big.LITTLE MP Support in Exynos 5420, AnandTech, Sept. 11 2013, http://www.anandtech.com/show/7313/samsung-announces-biglittle-mp-support-in-exynos-5420

[6]: big.LITTLE MP, Linaro, https://wiki.linaro.org/projects/big.LITTLE.MP#Overview

[7]: Randhawa R., ARM's big.LITTLE systems provide more processing power for less energy, New Electronics, July 10 2012, http://www.newelectronics.co.uk/electronics-technology/arms-big-little-systems-provide-more-processing-power-for-less-energy/43563/

[8]: MediaTek Enables ARM big.LITTLE Heterogeneous Multi-Processing Technology in Mobile SoCs, http://www.mediatek.com/_en/Event/201307_TrueOctaCore/MediaTekEnablesARMbigLITTLEHMPTechnology.pdf

[9]: Jeff B., Advances in big.LITTLE Technology for Power and Energy Savings, White Paper, Sept. 2012, http://www.arm.com/files/pdf/Advances_in_big.LITTLE_Technology_for_Power_and_Energy_Savings.pdf

7. References (2)

[10]: CoreTile Express A15x2 A7x3 Power Management, Application Note 318, April 2013, http://infocenter.arm.com/help/topic/com.arm.doc.dai0318e/DAI0318E_v2p_ca15_a7_power_management.pdf

[11]: Kim M., Kim H., Chung H., Lim K., Samsung Exynos 5410 Processor – Experience the Ultimate Performance and Versatility, White Paper, 2013

[12]: Shin Y., Shin K., Kenkare P., Kashyap R., 28nm high- metal-gate heterogeneous quad-core CPUs for high-performance and energy-efficient mobile application processor, IEEE, 2013

[13]: Whitwam R., Samsung Exynos 5 Octa 5420 looks to correct past mistakes, shoddy graphics, Extreme Tech, July 23 2013, http://www.extremetech.com/computing/162090-samsung-looks-to-correct-past-mistake-with-updated-exynos-5-octa-5420-arm-chip

[14]: Smith C., These are the Galaxy S5's next-gen processors, BGR, Febr. 26 2014, http://bgr.com/2014/02/26/galaxy-s5-processor-snapdragon-801-exynos-5422/

[15]: Grey G., big.LITTLE Software Update, Linaro, July 10 2013, http://www.linaro.org/blog/hardware-update/big-little-software-update/

[16]: Poirier M., In Kernel Switcher: A solution to support ARM's new big.LITTLE technology, Linaro, Embedded Linux Conference, March 1 2013, https://events.linuxfoundation.org/images/stories/slides/elc2013_poirier.pdf

[17]: Jeff B., big.LITTLE Technology Moves Towards Fully Heterogeneous Global Task Scheduling, ARM TechCon, Nov. 2013, http://community.arm.com/servlet/JiveServlet/previewBody/7763-102-1-12076/big.LITTLE%20technology%20moves%20towards%20fully%20heterogeneous%20Global%20Task%20Scheduling_final%20%28pdf%29.pdf

7. References (3)

[18]: Boudra F., The Linaro IKS code now publicly available, Linaro, May 2 2013, http://www.linaro.org/blog/releases-blog/the-linaro-iks-code-now-publicly-available/

[19]: Enabling the Next Mobile Computing Revolution with Highly Integrated ARMv8-A based SoCs, ARM and Qualcomm, 2014, http://www.arm.com/files/pdf/ARM_Qualcomm_White_paper_Final.pdf

[20]: Exynos 5 Hexa, Samsung, http://www.samsung.com/global/business/semiconductor/product/application/detail?productId=7979&iaId=2341

[21]: Exynos 5 Octa, Samsung, http://www.samsung.com/global/business/semiconductor/minisite/Exynos/w/solution.html#?v=octa_5430

[22]: Jonnalagadda H., Samsung announces 64-bit Exynos 7 Octa with significant performance improvements, Android Central, Oct. 16 2014, http://www.androidcentral.com/samsung-officially-announces-64-bit-exynos-7-octa-significant-performance-improvements

[23]: Frumusanu A., Samsung's Exynos 5433 is an A57/A53 ARM SoC, AnandTech, Sept. 16 2014, http://www.anandtech.com/show/8537/samsungs-exynos-5433-is-an-a57a53-arm-soc

[24]: Sawant N., From 10 million chipsets to 350 million: Decoding MediaTek's amazing rise, Tech2, Nov. 25 2014, http://tech.firstpost.com/news-analysis/interview-dr-finbarr-moynihan-discusses-mediateks-future-plans-india-rd-centres-expanding-verticals-242436.html

[25]: Blancas J., MediaTek prepara arsenal para el 2015, incluye SoCs de 64-bits, Xataka, Aug. 18 2014, http://www.xataka.com.mx/celulares-y-smartphones/mediatek-prepara-arsenal-para-el-2015-incluye-socs-de-64-bits

7. References (4)

[26]: China Smartphone chips: LTE changes the balance, Global Markets Research, June 11 2014, https://www.nomura.com/events/china-investor-forum/resources/upload/China_Smartphone_chips.pdf

[27]: Tu F., An analysis of next-gen CPUs: MediaTek MT6732 MT6752 and MT6595, Gizmochina, June 23 2014, http://www.gizmochina.com/2014/06/23/an-analysis-of-next-gen-cpus-mediatek-mt6732-mt6752-and-mt6595/

[28]: MT6595 Octa-Core Smartphone Application Processor, Technical Brief, Dec. 31 2013

[29]: MT6589 HSPA+ Smartphone Application Processor, Technical Brief, Sept. 26 2012, http://www.datasheet4u.com/datasheet/M/T/6/MT6589_MediaTek.pdf.html

[30]: MediaTek True Octa-Core, Position Paper, http://multicorechina.files.wordpress.com/2013/07/mediatektrueocta-corepositionpaper.pdf

[31]: MT6592 Octa-Core Smartphone Application Processor, Technical Brief, July 6 2013, http://www.datasheet4u.com/datasheet-pdf/Mediatek/MT6592/pdf.php?id=844976

[32]: Lin T.Y., MediaTek MT8135 SoC - Heterogeneous big.LITTLE Processing for Mainstream, Aug. 7 2013, http://community.arm.com/groups/processors/blog/2013/08/07/mediatek-mt8135-soc--heterogeneous-biglittle-processing-for-mainstream

[33]: MediaTek CorePilot, Heterogeneous Multi-Processing Technology, http://cdn-cw.mediatek.com/MediaTek_CorePilot.pdf

[34]: Shivram R.V., e-MMC (Embedded MultiMedia Card), Richiervs, June 13 2012, http://richiervs.blogspot.hu/2012/06/e-mmc-embedded-multimedia-card.html

7. References (5)

[35]: 32 GB EMMC Memory Micro SD card Connection, 320 Volt, http://320volt.com/en/32-gb-emmc-bellege-sd-kart-baglantisi/

[36]: Englert J., SOC Royal Rumble! iPad Air 2 vs. Nexus 9 vs. China Tablet SOCS, Mobile Droid, 2014, http://www.mobiledroid.co.uk/blog/soc-royal-rumble-ipad-air-2-vs-nexus-9-vs-china-tablet-socs/

[37]: Wang, A., Lin T.Y., Ouyang S., Huang W.H., Heterogeneous multi-processing quad-core CPU and dual-GPU design for optimal performance, power, and thermal tradeoffs in a 28nm mobile application processor, ISSCC, 2014

[38]: Renesas Mobile Introduces Ground-Breaking Quad Core ARM Cortex-A15/Cortex-A7 CPU-based Communication Processor with Integrated LTE Cat-4 Modem, Febr. 14 2013, http://www.renesas.com/press/news/2013/news20130214b.jsp

[39]: A80 Octa-Core Block Diagram, Allwinner Technology, 2014, http://www.allwinnertech.com/en/clq/processora/A80.html

[40]: Walton J., State of the Part: SoC Manufacturers, AnandTech, Aug. 19 2014, http://www.anandtech.com/show/8389/state-of-the-part-soc-manufacturers

[41]: Wikipedia, Allwinner Technology, http://en.wikipedia.org/wiki/Allwinner_Technology

[42]: Demerjian C., Allwinner shows off 8-core A80 SoC for the mid-range, Semi Accurate, Febr. 7 2014, http://semiaccurate.com/2014/02/07/allwinner-shows-8-core-a80-soc-mid-range/

7. References (6)

[43]: Merritt R., Chinese SoC firm plans IPO to push mobile roadmap, EET Asia, Nov. 13 2013, http://www.eetasia.com/ART_8800691924_499489_NT_3a5a522f.HTM

[44]: Wikipedia, MediaTek, http://en.wikipedia.org/wiki/MediaTek

[45]: MediaTek is repositioning itself with the new MT6732 and MT6752 SoCs for the "super-mid market" just being born, plus new wearable technologies for wPANs and IoT are added for the new premium MT6595 SoC, Lazure, March 4 2014, http://lazure2.wordpress.com/2014/03/04/mediatek-is-repositioning-itself-with-the-new-mt6732-and-mt6752-socs-for-the-super-mid-market-just-being-born-plus-new-wearable-technologies-for-wpans-and-iot-are-added-for-the-new-prem/

[46]: Cunningham A., SoC, Ars Technica, May 12 2015, http://arstechnica.com/gadgets/2015/05/mediatek-escalates-the-multicore-madness-with-a-10-core-smartphone-soc/

[47]: MediaTek CorePilot 3.0, White Paper, http://cdn-cw.mediatek.com/White%20Papers/MediaTek%20CorePilot%203.0%20White%20Paper%20PDFCP3WP%200915.pdf

[48]: big.LITTLE Technology: The Future of Mobile, White Paper, 2013, https://www.arm.com/files/pdf/big_LITTLE_Technology_the_Futue_of_Mobile.pdf

[49]: Rickards I., Kucheria A., Energy Aware Scheduling (EAS) progress update, Linaro, Sept. 18 2015, https://www.linaro.org/blog/core-dump/energy-aware-scheduling-eas-progress-update/

7. References (7)

[50]: ARM Cortex-A57 MPCore Processor Technical Reference Manual, 2013, http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0488c/CACDDBBA.html

[51]: Watts C., Flynn D., Interoperability Developer's Forum, Oct. 21 2004, https://www.synopsys.com/Community/Interoperability/Documents/devforum_pres/2004oct/lpf/01-ARM-IEM.pdf

[52]: Anderson M., Scheduler Options in big.LITTLE Android Platforms, 2015, http://events.linuxfoundation.org/sites/events/files/slides/GTS_Anderson.pdf

[53]: Frumusanu A., Smith R., ARM A53/A57/T760 investigated - Samsung Galaxy Note 4 Exynos Review, AnandTech, Febr. 10 2015, http://www.anandtech.com/show/8718/the-samsung-galaxy-note-4-exynos-review

[54]: Poirier M., LCA14-104: GTS- A solution to support ARM's big.LITTLE technology, SlideShare, March 23 2014, http://www.slideshare.net/linaroorg/lca14-104-gtsasolutiontoarmsbiglittletechnology

[55]: Ho J., Frumusanu A., Understanding Qualcomm's Snapdragon 810: Performance Preview, AnandTech, Febr. 12 2015, http://www.anandtech.com/show/8933/snapdragon-810-performance-preview/4

[56]: Sims G., ARM's Intelligent Power Allocation adds some more clever to thermal management, Android Authority, Oct. 13 2014, http://www.androidauthority.com/arms-intelligent-power-allocation-536244/

[57]: ARM Intelligent Power Allocation, 2014, http://www.armtechforum.com.cn/2014/sz/C-1_ARMIntelligentPowerAllocation.pdf

7. References (8)

[58]: Agrawal P., Power allocation based thermal management, Gmane, Febr. 27 2014, http://article.gmane.org/gmane.linux.power-management.general/43243

[59]: Kucheria A., Energy-Aware Scheduling (EAS) Project, Linaro, Jan. 27 2015, https://www.linaro.org/blog/core-dump/energy-aware-scheduling-eas-project/

[60]: MediaTek CorePilot 2.0, White Paper, http://cdn-cw.mediatek.com/White%20Papers/MediaTek_CorePilot%202.0_Final.pdf

[61]: MediaTek Launches the MediaTek Helio X20: The World's First Mobile SoC Featuring Tri-Cluster CPU Architecture, May 12 2015, http://www.mediatek.com/en/news-events/mediatek-news/mediatek-launches-the-mediatek-helio-x20-the-worlds-first-mobile-soc-featuring-tri-cluster-cpu-architecture/

[62]: Muckle S., A QuIC Take on Energy-Aware Scheduling, Slideshare, Sept. 18 2014, http://www.slideshare.net/linaroorg/lcu14-406-a-quick-take-on-energyaware-scheduling

[63]: Snapdragon 820 Processor, Qualcomm, https://www.qualcomm.com/products/snapdragon/processors/820

[64]: Snapdragon 820 and Kryo CPU: heterogeneous computing and the role of custom compute, Qualcomm, Sept. 2 2015, https://www.qualcomm.com/news/snapdragon/2015/09/02/snapdragon-820-and-kryo-cpu-heterogeneous-computing-and-role-custom

[65]: Asif S., Samsung's Mongoose SoC benchmarks leak, shows incredible performance, SAMMOBILE, Sept. 2015, http://www.sammobile.com/2015/10/06/samsungs-mongoose-soc-benchmarks-leak-shows-incredible-performance

7. References (9)

[66]: The Android Source Code
https://source.android.com/source/index.html

[67]: ARM A53/A57/T760 investigated – Samsung Galaxy Note 4 Exynos Review, Cyber Parse, Feb. 10, 2015,
http://cyberparse.co.uk/2015/02/10/arm-a53a57t760-investigated-samsung-galaxy-note-4-exynos-review/

[68]: ARM Intelligent Power Allocation, ARM Techforum 2014,
http://www.armtechforum.com.cn/2014/sz/C-1_ARMIntelligentPowerAllocation.pdf