big.LITTLE Architecture: Heterogeneous Multicore Processing

Shubham Kamdar
B-Tech Scholar
Institute of Technology,
Nirma University, Ahmedabad

Neha Kamdar
Assistant Professor
Malwa Institute of Technology,
Indore

ABSTRACT
ARM's big.LITTLE architecture introduces a solution to optimize power consumption by selecting the core type most suitable for a level of processing load along with high performance. Using heterogeneous multi-core processing having high performance cortex-A15 with power efficient cortex-A7 is what big.LITTLE architecture follows to have power efficient system on chip which is demanded by today's smart phones. For smooth functioning and effective data transferring big.LITTLE uses GIC-400 and CCI-400 interfaces to interface between two processors. Present smart phones using big.LITTLE technology having cluster migration as their operating mode. But using global task scheduling method as operating mode of smart phones in which cores gets individually activated or deactivated according to workload can give even further improvement in power consumption.

General terms
GIC-400, CCI-400

Keywords
Cortex-A7, Cortex-A15, GIC-400, CCI-400, GTS, HMP

1. INTRODUCTION
As the need for higher performance is increasing day by day, the battery life (capacity) of that system must also be increased accordingly. But the actual problem comes in the size of the battery which cannot be altered a lot because of the small size of the mobile phones. So to overcome this problem we need an architecture which can provide us with high performance as well lower power consumption. Previously in most of the mobile phones, a processor giving higher performance or fast processing (Big processors)are not power efficient like ARM cortex A-15 .And processors consuming less power (Little processor)do not provide fast processing like ARM cortex A-7.Thus our goal cannot be achieved by using processors of same type. To overcome this problem ARM series launched a technology termed as "big.LITTLE technology" which is a heterogeneous processing architecture which uses two types of processor giving performance driven by big processor and power efficient little processor.

2. big.LITTLE TECHNOLOGY
The first big.LITTLE processing pair consists of the ARM Cortex-A15 and Cortex-A7 processors. Since both processors support the same ARMv7-A ISA, the same instructions or program can be run in a consistent manner on both processors. Differences in the internal Micro architecture of the processors allow them to provide the different power and performance characteristics that are fundamentals to the big.LITTLE processing concept. Applying these on a system we can divide the work load according to performance, getting optimum power consumption. This becomes more clear after exploring [2] characteristics of the smart phones such as Samsung galaxy S4, there is less need of powerful CPU when running low –intensity tasks such as emailing or MP3 player which a smaller more efficient CPU is able to complete proficiently consuming less power. Conversely, a powerful CPU is best candidate for heavy duty processing required for gaming or multipage web browsing, where using lower-power CPU could lead to delay of work.

3. ARM cortex -A7 and ARM cortex -A15
The most important point for big.LITTLE to work coherently is that both big and little processors should be architecturally identical. This is fulfilled by ARM cortex-A7 and cortex-A15 processors both of which are working on ARMv7A architecture. Also cortex-A7 is 100% [9] compatible with the cortex A15. Which means any code running on an A15 can run on a cortex A7, but slower than A15. This is very important for switching of loads depending on workload requirements.

But the reason which differentiates them as big and little lies in their micro-architecture which is quite different. The cortex-A7 processors have a pipeline length of 8-stages and 10-stages which is relatively simple and is designed to be extremely power efficient, whereas cortex-A15 has pipeline length of between 15-stages to 24-stages which is designed to achieve high performance by running more instructions in parallel on a bigger and more complex pipeline.

And the energy consumed by the execution of an instruction is partially related to the number of pipeline stages [1] it must traverse. Also more the pipeline stages better will be the performance as more number of instructions will be executed in parallel. This is the reason which came after analysis [1] that cortex-A15 delivers roughly 2x the performance of Cortex-A7 per unit MHz, and Cortex-A7 is roughly 3x as efficient as that of Cortex-A15 in completing a workload.

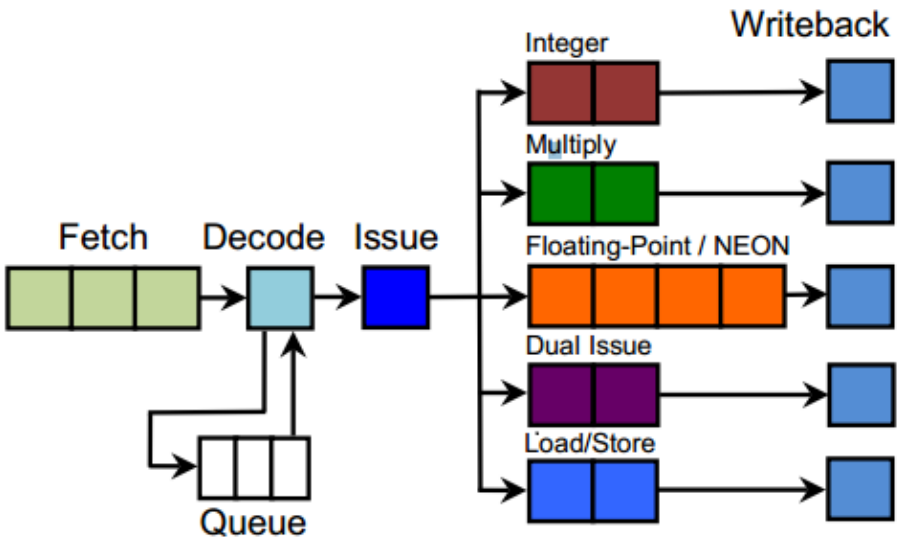

Fig.1: - pipelining in cortex-A7 [1]

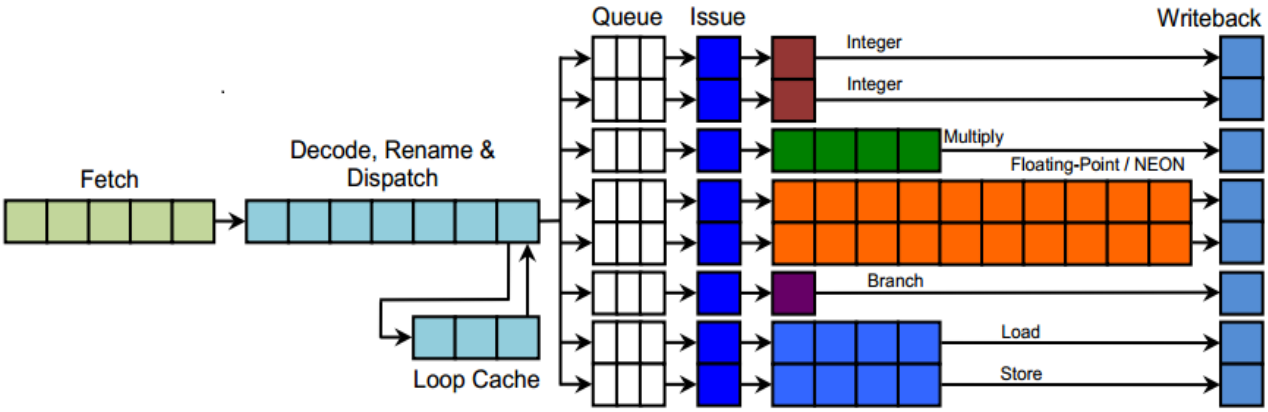

Fig.2: - pipelining in cortex-A15 [1]

4. PROCESSOR INTERFACING

The most important point for big.LITTLE architecture to work efficiently is both the processors should be interfaced coherently so that they can transfer information to each other. So in big.LITTLE they are tied together via two interfaces called CoreLink CCI-400 and GIC-400 interrupt control, provided by ARM, in which "cache coherent interconnect" provides seamless data transfer between clusters. And CoreLink provides dynamically configurable interrupt distribution to all the cores through which it can share 480-interrupts to cortex-A15 and cortex-A7. The programmable nature of GIC-400 allows interrupts to be migrated between any cores in the two clusters

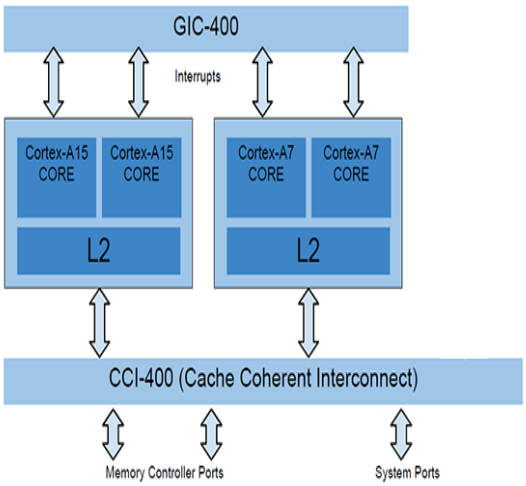

Fig.3: - big.LITTLE Coherency [2]

.

5. big.LITTLE WORKING

The flow of command given by the user to the fulfilment of the task starts from the activation of interrupt corresponding to that command. This interrupt of GIC-400 then sends this information to processors to get executed .This execution for good performance can be executed according to the operating mode of that architecture. big.LITTLE can work on mainly three operation modes such as cluster migration, CPU migration, Global Task scheduling depending on the combination and work distribution of ARM cortex-A7 and ARM cortex-A15 processor.

5.1 Cluster Migration

When we run a particular application on the most appropriate CPU/cluster requires that a device's operating system and power-management algorithms, called dynamic voltage and frequency scaling (DVFS) [5], monitor and allocate workload correctly. The operating system is in charge of deciding which cluster and CPUs are active, depending upon load, and ARM has fine-tuned the DVFS profiles to ensure that, in our examples, migration to the Cortex-A15 occurs once the application(s) requires more processing power than the Cortex-A7 can provide when operating at its highest frequency. With big. LITTLE's initial software implementation, called Cluster Migration, only one processor cluster is active at one time; the other cluster - be it Cortex A15 or Cortex-A7 - is powered down. The primary focus is on reducing overall power consumption when compared to a single performance-orientated CPU. In Cluster Migration, the operating system sees cluster blocks instead of individual cores in the Cortex-A15 and Cortex-A7 big.LITTLE configuration. Cluster-level switching can be inefficient if, for example, there is high load on a single core contained within a Cortex-A7, yet other cores within cluster are running at low load. In this case, even though the Cortex-A7 has the multi-core ability to handle the task, the entire cluster may be switched off and migrated to the power-hungry, faster Cortex-A15.So to overcome this problem big.LITTLE came with a new operating mode that is CPU migration.

5.2 CPU Migration

In CPU Migration single Cortex-A7 and Cortex-A15 cores are coupled together to form a virtual CPU. Only one core is active at a time depending on the work load, but the load can be moved from one core to another .This transferring of data is controlled by what is known as the cpufreq driver, which is responsible for telling the OS' kernel the required voltage and frequency.

It takes around 30 microseconds [2] for the big.LITTLE software to move the workload from one of the paired cores to another, based on load requirements, but the trade-off between lost time and improved energy efficiency is worth it. This paired approach of CPU Migration, requires that the ARM Cortex processors have an equal number of cores. What this also means is that a maximum of four cores can be run at any time. But there are many situations where high performance cores or big cores are required less in number compared to power efficient cores depending upon different application. This issue is also solved by "Global Task scheduling" mode.

5.3 Global Task Scheduling

The best method of smooth transfer of load between two big.LITTLE processors is called Global Task Scheduling (GTS) and is a type of Heterogeneous Multi-Processing (HMP) [6]. Here, all cores can be active at any time. Further, any combination of cores can be used simultaneously. The operating-system scheduler can then dynamically allocate workload on any core in either processor and manufacturers can also use non-core-matching configurations - two-core Cortex-A15s paired with four-core Cortex-A7s, for example. Unused processors are automatically turned off to conserve power, by the Operating System.

Using the scheduler instead of the cpufreq driver provides a performance boost of up to 10 per cent [2] in many benchmarks, according to ARM, while there's also the

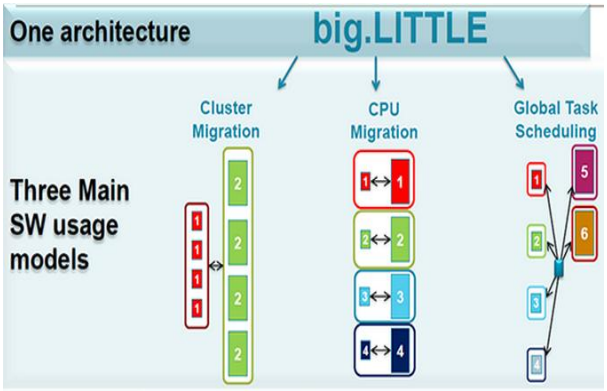

Fig.4: - Operating Modes of big.LITTLE

Possibility of further reduced power which is the most important factor in mobile devices and can be achieved by reacting more quickly to load changes and allocating work in a more fine-grained manner.

The advantage of using GTS is, it give high performance with relatively less number of cores operating resulting in more power consumption. To explain this fact, an example would be best. Let's say a user is simultaneously playing a HD video while he is also playing Angry birds. In this HD video will be handled by cortex-A15 and angry birds will be handled by cortex-A7, in Comparision to A15 handling both tasks simultaneously, leading to more power consumption than the first case .Similarly if a user is playing a audio song while playing temple run then OS will again distribute work accordingly (temple run to cortex-A15 and cortex-A7 to audio). This power reduction is best shown using graph comparing power consumed by Mobile using only cortex-A15 with mobile using big.LITTLE GTS operating mode for distributing load. This shows that the audio power consumption is reduced to 77% in Comparison.

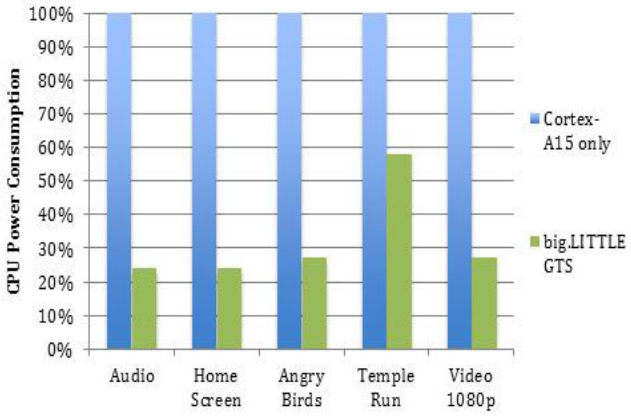

Fig.5: - big.LITTLE Power Savings compared to a Cortex-A15 processor-only based system

6. CONCLUSION

ARM big.LITTLE processing is a very good power optimizing technology which can save up to 75% [7] of CPU in low to moderate performance scenarios, and can increase performance by 40% in highly threaded workloads. This becomes much clear after analysing the graph comparing power consumption of cortex-A15 and big.LITTLE which clearly shows power efficiency of big.LITTLE HMP technology. The core reason behind reduced power consumption is due to the operating system working on global task scheduling and dynamically distributing data among cores in a way that task requiring high processing speed is sent to cores of cortex-A15 which is having 24 stages of pipelining which processes data faster than cortex-A7 . While on the other hand power is saved by sending low load task to cortex-A7 having 8-10 pipeline stages resulting in lower power consumption. Overall they all are assigned workload in a way that the net power consumption and performance meets the user requirement. This all is because of GIC-400 and CCI-400 interrupts where GIC-400 shares its 480 interrupts to manage the tasks between the processors and CCI-400 provides cache coherency between multi-core processors along with os operating system which is assigning workload working on global task scheduling (GTS) mode. The biggest advantage of using GTS is it reduces number of cores, as we can have any number of cores of big, LITTLE processors in a way that can fulfil our demands. Such as having two big cores instead of four along with four LITTLE cores can reduce both cost and power consumption in mobile phones.

7. REFRENCES

[1] Brian Jeff, ARM: Advances in big.LITTLE Technology for power and energy saving.

[2] Hexus: Tech explained –ARM big.LITTLE processing, 24 October 2013, 16:41 http://hexus.net/tech/tech-explained/cpu/48693-tech-explained-arm-biglittle-processing/.

[3] Advances in big.LITTLE Technology for Power and Energy Savings, Brian Jeff, ARM September 2012. http://www.arm.com/files/pdf/Advances_in_big.LITTLE_Technology_for_Power_and_Energy_Savings.pdf , 14/9/14 4:30 pm.

[4] White paper -2013 http://img.hexus.net/v2/press_releases/arm/big.LITTLE.Whitepaper.pdf.

[5] Peter Greenhalgh, ARM big.LITTLE processing with ARM cortex-A15 and A-7, September 2011.

[6] Kisoo Yu ; Donghee Han ; Changhwan Youn ; Seungkon Hwang ; Jaechul Lee, Power-aware task scheduling for big.LITTLE mobile processor Publication Year: 2013.

[7] Heterogeneous Multi-processing Solution of Exynos 5 Octa with ARM big.LITTLE technology, http://www.exynos.com.

[8] big.LITTLE technology. ARM: http://www.arm.com/products/processors/technologies/biglittleprocessing.php.

[9] Article : http://www.anandtech.com/tag/samsung.

[10] ARM's Cortex A7: Bringing Cheaper Dual-Core & More Power efficient High-end devices. By Anand Lal Shimpi. http://www.anandtech.com/show/4991/arms-cortex-a7-bringing-cheaper-dualcore-more-power-efficient-highend-devices