Whitepaper

NVIDIA Tegra 4 Family CPU Architecture

4-PLUS-1 Quad core

Table of Contents

Introduction

NVIDIA led the industry by introducing the Tegra® 2, the first multi-core mobile system-on-a-chip (SoC) processor in early 2011, changing the face of mobile computing. As the world's first dual-core mobile processor, Tegra 2 started shipping in tablets and smartphones delivering higher performance, longer battery life, and a better user experience. With the rapid adoption of high-end smartphones and tablets, consumers are relying more and more on such devices for their daily computing and entertainment needs.

With the introduction of the Tegra 3, NVIDIA shipped the world's first **4-PLUS-1™ quad core mobile processor.** This unique 4-PLUS-1 architecture utilizes four CPU cores for performance-intensive applications, and one **Battery Saver core** for processing low performance applications and background tasks when the device is idling. This unique combination delivers increased performance to consumers to enable full-featured Web browsing, console class gaming, extreme multitasking, photo editing and video editing, while at the same time increasing battery life.

The next generation of multi-core mobile processors is expected to deliver significantly higher performance, enabling PC-class productivity, social networking, multimedia, and console-quality gaming experiences on smartphones and tablets. In addition, operating systems continue to leverage the increasing performance of mobile processors to deliver new and improved features, faster, richer, and more efficient user interfaces, and enhanced application performance.

With the introduction of powerful SoC devices like Tegra 4, the processing capabilities of smartphones, tablets, and notebook PCs are converging, raising consumer expectations for PC-class performance and uncompromised battery life in their mobile devices. Higher resolution displays, wireless display solutions such as Miracast™, and console-class mobile games will have consumers using their tablets and smartphones as both portable gaming devices and family room gaming consoles.

NVIDIA's new **Tegra 4** family of SoC processors is architected to meet the challenges of this new crop of mobile use cases.

NVIDIA Tegra 4 Family of Mobile Processors

NVIDIA Tegra 4 is the world's first **ARM Cortex-A15**-based **quad core CPU** architecture and delivers superior performance for demanding mobile applications and improved battery life. Plus, with the addition of a 2nd generation **Battery Saver CPU** core and **Variable Symmetric Multiprocessing (vSMP**) technology, Tegra 4 further improves both performance and battery life.

Tegra 4 is both significantly faster than Tegra 3, as well as more power efficient, delivering faster performance and richer graphics while consuming roughly the same amount of power.

Figure 1 NVIDIA Tegra 4 Family

NVIDIA Tegra 4i is based on the Tegra 4 architecture and it brings the Tegra 4 super phone experiences into the mainstream smartphone, and integrates NVIDIA's LTE i500 SDR modem, all on a single chip. Tegra 4i includes ARM's newest and most efficient core— the **Cortex-A9 r4** CPU— in a quad-core implementation at 2.3 GHz with a fifth battery saver core, and is half the size of the competing quad-core integrated LTE mobile processors.

Tegra 4i's new CPU was designed by ARM with help from NVIDIA and sets new standards for efficiency and performance while being 2x the performance of the previous revision of Cortex-A9 found in Tegra 3. Tegra 4i delivers super-fast Web browsing and quick load time for apps, along with great battery life.

Benchmarking CPU Performance

Benchmarks are typically used to evaluate the performance of a device or a design such that the benchmark results correlate well with the real world performance of the design. Therefore when evaluating the performance of a CPU architecture, it is essential to ensure that the workloads used to benchmark the performance of the CPU is similar to those presented by real world applications and programs.

Over the years several benchmarks have been developed to evaluate the performance of CPUs. Two of the well-known benchmarks that have been used for CPU benchmarking are

Dhrystone MIPS and Coremark. Dhrystone MIPS, also known as DMIPS, is a synthetic benchmark that was developed about 30 years ago primarily intended to be representative of CPU integer processing performance.

Unfortunately while DMIPS was relevant 30 years ago when CPU architectures were much simpler and tied closely to the speed of external memory, it is now outdated and does not accurately reflect the performance required of real world applications. There is a significant issue with DMIPs as a metric – the benchmark code fits entirely in the L1 caches of modern mobile processors. Thus it does not exercise the L1 cache miss handling logic, L2 cache, or SoC memory system. The author of the original Dhrystone benchmark has this to say even way back in 1999:

> *" DMIPS cannot claim to be useful for modern workloads as it is so short, it fits in on-chip caches, and fail to stress the memory system "*
>
> *Author of DMIPS benchmark (Weicker), EDN Magazine, 10/28/1999*

Similarly, other widely used benchmarks such as Coremark and Linpack are written to test only one specific portion of the CPU performance. Coremark tests only the ALU and FPU performance while Linpack is only a measure of the performance of the CPU's Floating Point Unit.

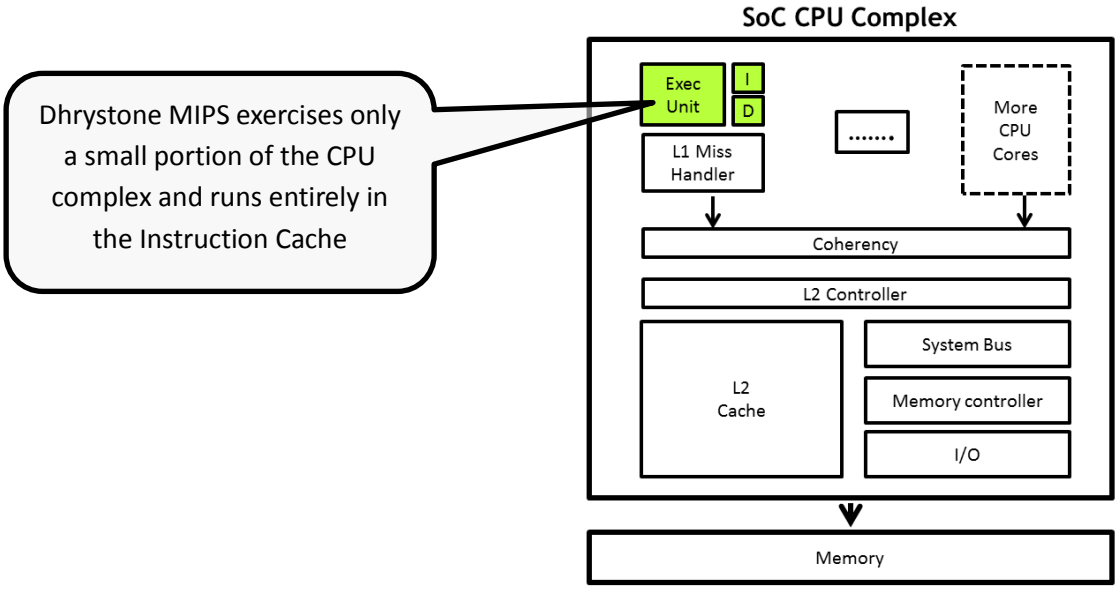

Figure 2 Dhrystone MIPS benchmarks only a small section of the CPU complex

A more representative benchmark of true CPU performance is the SPECint benchmark. It was written to specifically use the kernels of real world applications such as file compression, word processing, gaming, database management, and many more instead of using synthetic tests.

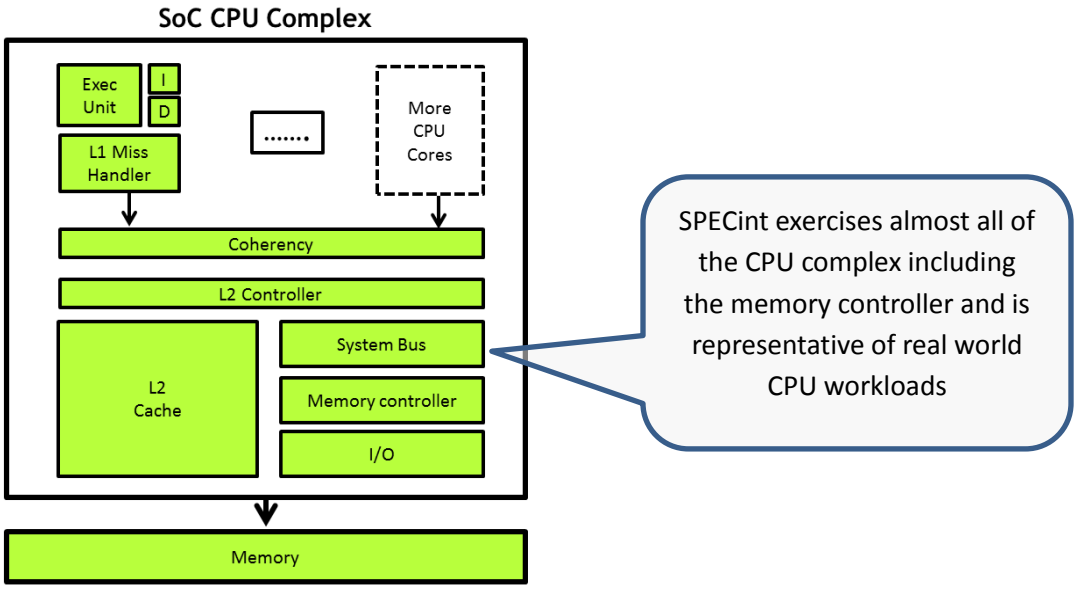

Figure 3 SPECint exercises the entire CPU complex and has workloads that are similar to real world applications

Tegra 4 Family CPUs Architected for High Performance and Power Efficiency

Current generation mobile applications are stretching the performance capabilities of today's mobile processors. The transition to multi-core CPU architecture and higher operating frequencies have increased processor capabilities to meet growing performance needs. But, the next generation of PC-class mobile applications requires a new CPU architecture that not only has sufficient headroom to meet growing performance needs, but also continues to stay within mobile power budgets.

Both the Cortex-A15 CPU architecture in Tegra 4 and the Cortex-A9 r4 CPU architecture in Tegra 4i have several enhancements over the current generation architectures, providing up to two times the performance of current solutions in their respective device classes. Some of the key enhancements in these architectures are described below.

Wider Issue Execution Units for Higher Throughput

The Cortex-A15 CPU architecture has eight execution units, and is capable of issuing eight instructions per clock compared to five on Cortex-A9. However, the execution units are of little value unless they can be effectively used. In order to fully benefit from the higher number of

execution units, the architecture needs to support deeper Out-Of-Order execution for better Instruction Level Parallelism (ILP), and for earlier detection of cache misses to increase Memory Level Parallelism (MLP). It should also have accurate branch prediction capabilities to improve work efficiency.

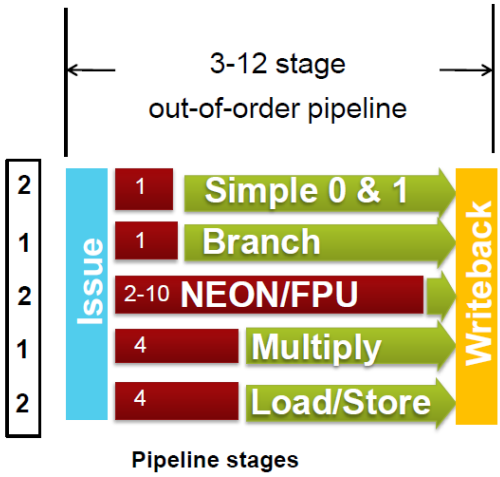

Figure 4 Eight Execution Units on ARM Cortex-A15 CPU

Better Memory Level Parallelism from a Larger Instruction Window for Out-of-Order Execution

CPU performance is heavily dependent on reducing the latency to memory. The performance efficiency of a CPU drops significantly if the CPU is idle and waiting for data to be fetched from memory. Therefore, it is important to not only reduce the number of cache misses (that trigger data fetches from memory) with larger and more effective CPU caches, but also to increase the Memory Level Parallelism (MLP - the number of data fetches from system memory happening concurrently). The effective latency in fetching data from system memory back to the CPU is equal to the latency for one memory fetch, divided by the number of memory fetches that can be executed in parallel (MLP).

To improve CPU performance efficiency it is therefore important to look ahead many instructions, and issue Load instructions that may miss in the caches. The larger the Out-Of-Order lookup window, the greater are the chances of finding such a Load.

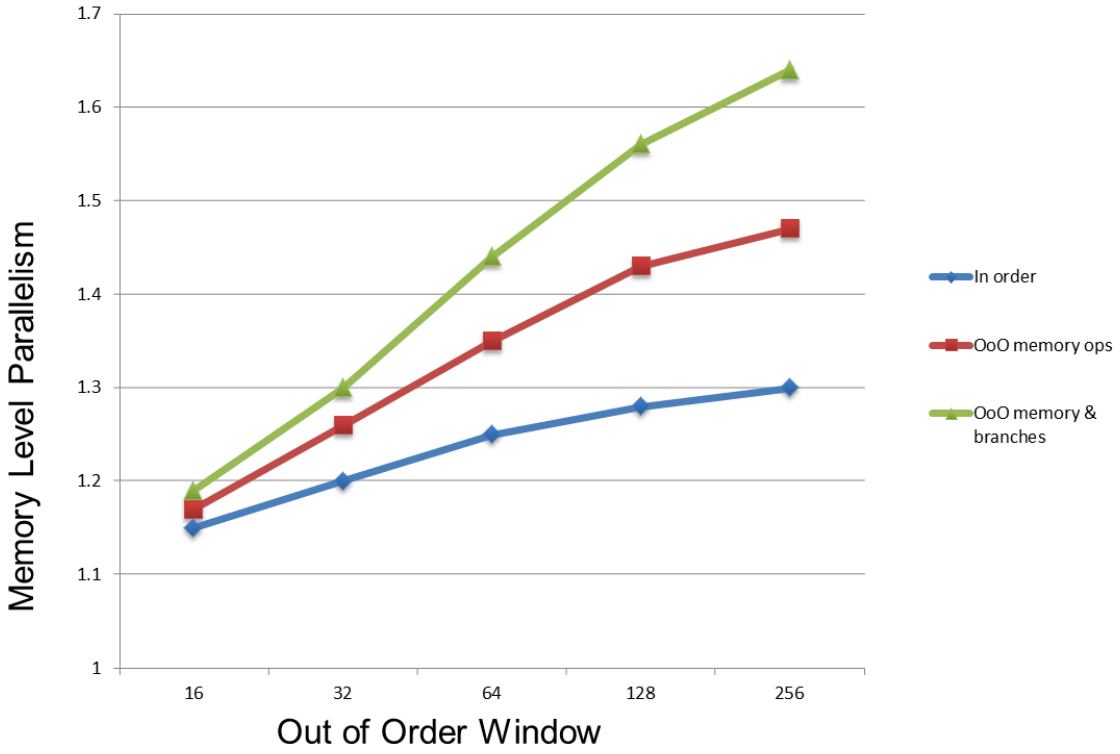

Figure 5 Relationship between Memory Level Parallelism and OoO Window lookup size

Figure 3 shows the importance of being more aggressive in CPU architecture when using a larger OoO window. Both memory operations and branches should be executed out-of-order in order to maximize the available MLP in an application.

The Cortex-A15 architecture has a large 128 micro-op lookup window that significantly improves MLP, and execute branches and some memory operations out-of-order (for a majority of ARM instructions there is a one-to-one mapping between instructions and micro-ops, but some of the more complex CISC-like instructions include two or more operations that can be expanded). In comparison, the Qualcomm Krait processor architecture has only a 40 instruction OoO window, limiting its capability for MLP.

Note that all memory operations can complete out of order, the only restriction on OoO memory operation issue is that the address must be known for older stores, so A15 lies close to the fully OoO green line on the graph above.

Fast Load-To-Use Logic allows larger L1 Data Cache

A critical path in modern CPU design is the load-to-use path, where an operation is dependent on the result of a preceding load. This path has a number of complex steps:

- Add to compute address

- Translate address from virtual to physical
- Compare address to the tag contents
- Select the critical data from the resulting cache line
- Align that data to the required register

Cortex-A15 combines the logic for add and translate steps using a highly efficient logic design that generates only enough data to determine if the compare hits without the full (and slow) add operation. Combining these steps speeds up the load-to-use path, and permits implementation of a high speed 32KB L1 Data cache, even though large L1 caches typically must run at lower clock rates than smaller caches. Indeed, competing architectures like Krait are limited to a 16KB L1 Data cache. Figure 6 shows the effect of data cache size on SPECint tests, showing the importance of a 32K data cache.

Cortex-A9 r4 combines the same logic and 32K cache, and also increases the TLB size to 512 entries to match A15.

- Cortex-A15 and Cortex-A9 r4 support a highly efficient load-to-use path
- Cortex-A15 and Cortex-A9 r4 have 32KB L1 Data Cache
- Cortex-A9 r4 also has the larger TLB (translation lookaside buffer) of the Cortex-A15

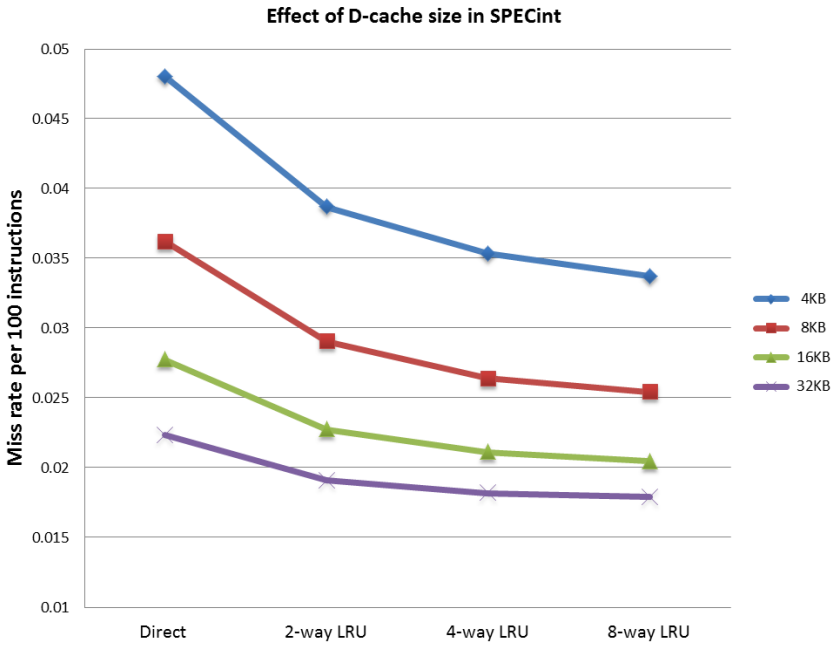

Figure 6 Impact of Data Cache size on cache miss rates and CPU performance

Enhanced branch prediction for higher efficiency

Accurate Branch Prediction is critical for OoO efficiency, both to extract MLP, and to avoid wasted work. Cortex-A15 has an advanced branch prediction mechanism with the following features:

- A bi-mode predictor to reduce interference between aliased branches that branch in opposite directions. Bi-mode predictors are broadly described on the Internet for those in search of further details.
- Branches resolve OoO to fix speculation errors sooner, to avoid wasted work.
- Return Address Stack correction from speculative pops and pushes. An aggressive out-of-order machine like A15 can corrupt the return address stack with speculative calls and returns, and therefore has a mechanism to correct mis-speculation.
- Indirect Branch Prediction mechanism. Indirect branches are common for the interpreters required for Web languages. To better predict such branches A15 has a separate structure that is accessed based on PC and history for the target.
- Finally, a static prediction mechanism is required to predict cold BTB misses.

The Cortex-A9 r4 architecture has the same larger history register as A15, and the same predictor structure sizes. These enhancements help the Cortex-A9 r4 CPU to deliver prediction accuracy that is similar to that of the Cortex-A15 architecture.

- Cortex-A9 r4 matches Cortex-A15 branch prediction performance by increasing the History Buffer and Saturating counters

Advanced Prefetcher for higher MLP and lower latency

Another method used in advanced processors to increase MLP is a hardware pre-fetcher. By predicting the data required in advance of its use, a pre-fetcher can decrease memory latency and increase MLP. In particular, there are common situations that that an out-of-order machine cannot detect. For example, if the program is following a linked list on a regular structure through memory, the successive loads cannot be issued until the previous load has completed to give the address. A stride detecting pre-fetcher will be able to load this data in advance.

Cortex-A15 has one hardware pre-fetcher per CPU in the L2 controller. Importantly, it uses both the program counter (PC) and the address bits to detect strided access, and to avoid aliasing between different threads that would occur without the use of the PC.

It will perform a programmable number of prefetches, and has a hardware throttling mechanism to avoid resource hogging.

Cortex-A9 r4 has a new L1 pre-fetcher in each CPU, similarly using both the PC and address. It has dedicated buffers so it does not allocate into the L1 D-cache until the data hits. It will check its own effectiveness and back off when ineffective, and can track multiple streams.

- Cortex-A9 r4 includes an L1 data Prefetcher
- The Cortex-A9 r4 Pre-fetcher has dedicated prefetch buffers and is capable of tracking multiple prefetch streams

Large Unified L2 Cache

Caches are used on CPUs to reduce the number of off-chip accesses to system memory. Caches store the most frequently used data on-chip enabling the CPU to access the data faster and improving the performance and efficiency of the CPU. Each core of the quad core ARM Cortex-A15 CPU complex on NVIDIA Tegra 4 has its own 32KB Instruction cache and 32KB of Data cache. All four cores share a common large 2MB L2 cache, which is 16-way set associative. The large 128 entry deep Out-off-order buffer allows the L2 cache latency to be largely hidden. Along with the 32KB L1 Caches, the 2MB L2 cache works to minimize off-chip fetches to system memory, both increasing performance and reducing power as DRAM fetches are more power intensive than on-chip SRAM fetches.

The quad core Cortex-A15 architecture is uniquely different from that of the Qualcomm Krait architecture in its implementation of the L2 cache. Unlike Krait, the entire 2MB cache on NVIDIA Tegra 4 is fully shared across all four cores. Depending on the workload, the allocated space for each core can be dynamically changed completely under hardware control. For example, on Tegra 4, one core could be allocated 1280KB while the other three cores could be assigned 256KB each for a total of 2MB. Similarly if a multi-thread application is sharing a common data structure, all four cores may access the same L2 cache lines. This workload-based allocation of L2 Cache space improves the efficiency of L2 cache and helps further reduce cache miss rates.

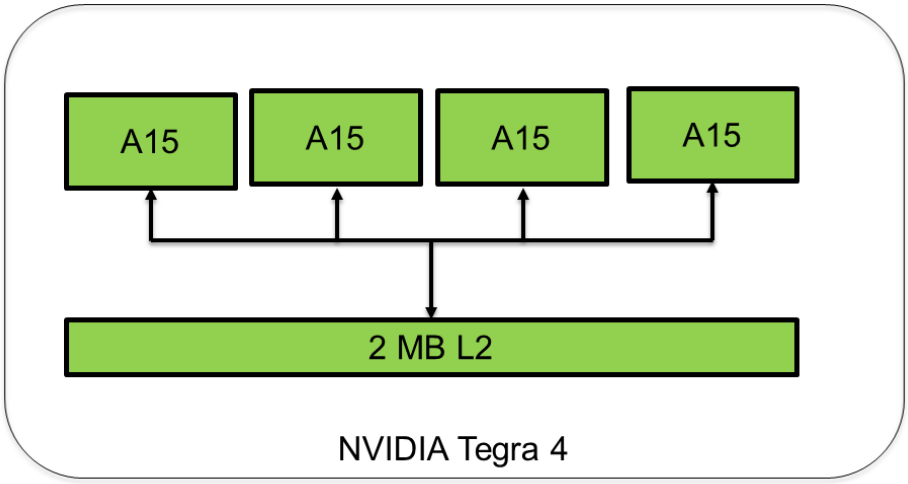

Figure 7 Entire 2MB L2 cache can be dynamically allocated to any CPU on NVIDIA Tegra 4

Krait on the other hand, due to the limitation of its Asynchronous SMP architecture, statically allocates a fixed 512KB of L2 cache for each CPU core. When a CPU core on Krait uses more than 512KB of data, it is forced to incur higher levels of L2 cache misses resulting in costly data fetches to system memory. The impact of the smaller static 512KB cache on Krait performance is easily seen on CPU benchmarks such as SPECint2K that mimic real world application performance.

Further, when a thread migrates from one core to another, the core needs to fetch data from another core L2 and migrate it to the local L2. This inter-core traffic is subject to a significant additional delay due to the clock domain-crossing synchronizers required between each core on the Asynchronous SMP boundary.

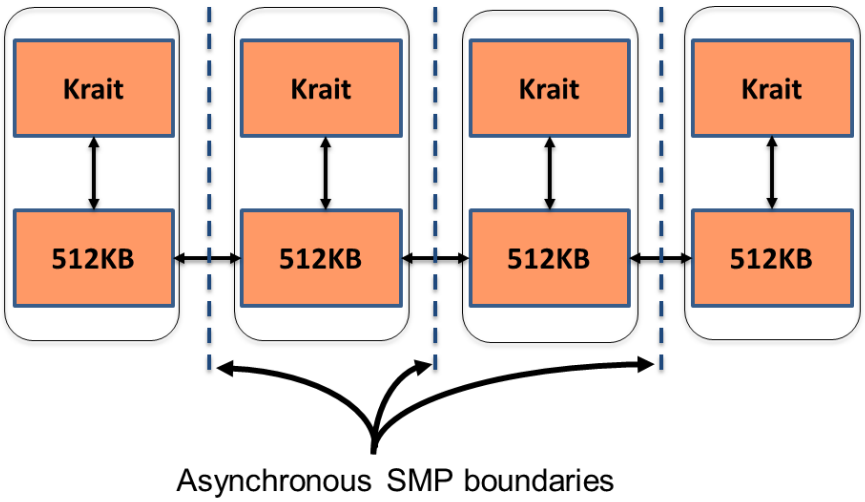

Figure 8 Asynchronous SMP prevents Krait from allocating the entire L2 to a single CPU core

Improved Cortex-A15 Power Efficiency

In addition to the features that deliver higher levels of performance, the Cortex-A15 CPU architecture includes several architectural enhancements that deliver higher power efficiency than current architectures.

The A15 architecture implements a large 32 entry Loop Buffer for active power management. Instruction fetches, Instruction Decodes, and Branch Prediction on Cortex-A15 consume about forty percent of CPU power. When the CPU is executing out of the Loop buffer these three elements are disabled and clock-gated off. Refer to the figure below.

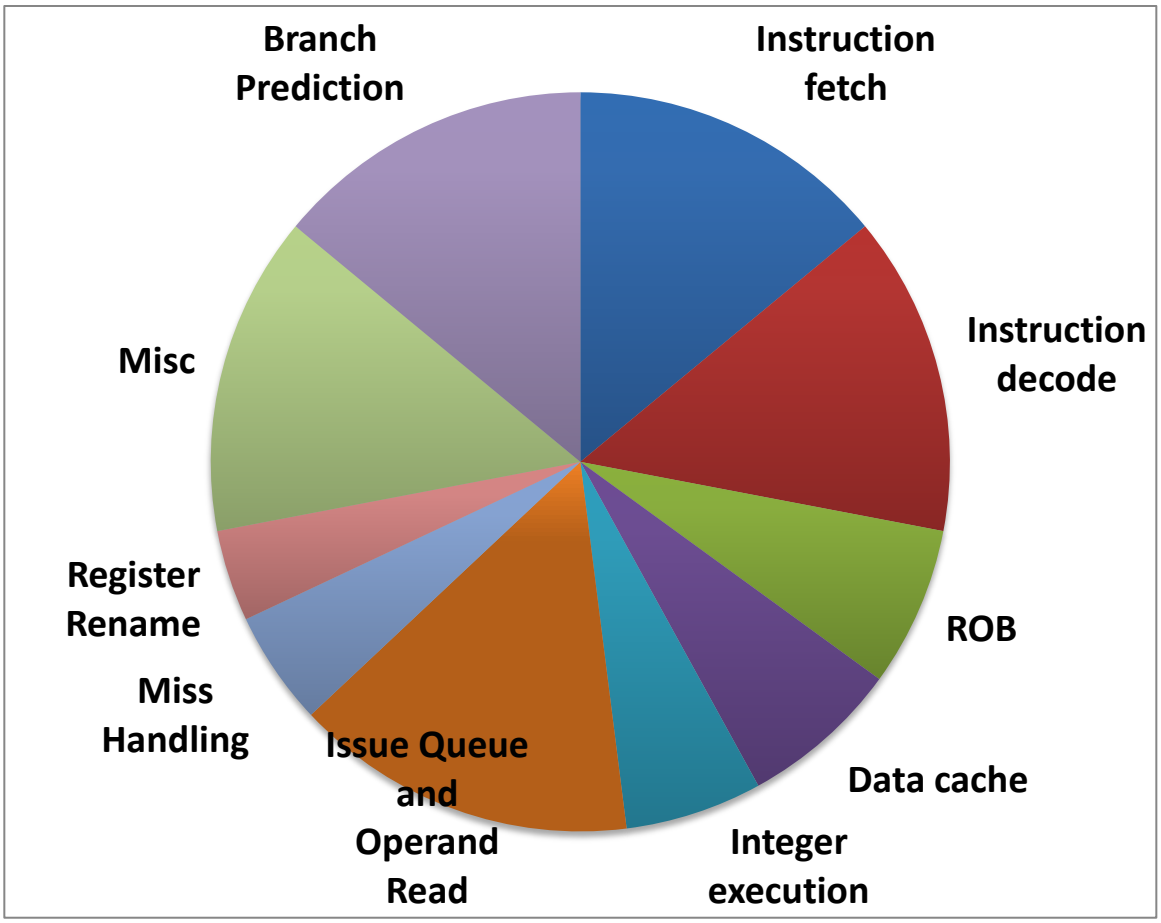

Figure 9 Deep Loop Buffer helps save almost fifty percent of front-end power consumed in a CPU core

Cortex-A15 balances performance and power by limiting Out-of-order instruction issue in cases where the power consumption is too high to justify the performance gain for mobile workloads. For example, a load instruction is not issued before older store instructions and new store instructions are not issued before older store instructions are executed. This avoids a power hungry memory disambiguation structure.

In addition, extensive clock and power gating is implemented across the design including the L1 and L2 caches to further reduce power consumption.

Fifth Battery Saver CPU Core

In addition to the four high-performance CPU cores, the NVIDIA Tegra 4 Family's Quad core CPU architecture includes a fifth low power Battery Saver CPU core. The fifth Battery Saver CPU core in this architecture is optimized for low power consumption and is built using transistors that require lower power to operate. The Battery Saver core is exceptionally power-efficient, and is focused on managing a wide variety of tasks that have low performance requirements, such as background email syncs, social media syncs, audio playback, video playback, book reading, and more.

The Battery Saver core is designed to handle low performance tasks and these tasks typically don't fully utilize the 2MB L2 cache. To further save power the Battery Saver core has its own 512KB L2 cache. When the main CPU cores are turned off, the 2MB L2 cache is flushed and power gated to save power. The Battery Saver core operates using the smaller and lower power 512KB L2 cache.

In cases where the performance requirements do not justify the use of the high performance quad core A15 CPU complex, Tegra 4 completely switches off the main quad CPU complex and switches to the Battery Saver core. The switching between the main quad core A15 CPU complex and the Battery Saver CPU core is managed by NVIDIA's second generation Variable Symmetric Multiprocessing (VSMP) technology that delivers high levels of power efficiency compared to the Qualcomm Krait architecture, which is based on Asynchronous SMP technology.

NVIDIA Tegra 4's 4+1 quad core processor based on vSMP technology delivers significant architectural benefits:

- **Cache Coherency:** vSMP technology does not allow both the Battery Saver Core and the main performance cores to be enabled at the same time, so there are no penalties involved in synchronizing caches between cores running at different speeds. ASMP incurs cache synchronization penalties when cores operating at higher frequencies attempt to synchronize data with cores running at much lower frequencies. This not only impacts the performance of the core(s) operating at higher frequency, but also adds additional synchronizing hardware between cores resulting in higher power consumption.
  Moreover, to accommodate asynchronous clocking, Krait architecture resorts to static L2 cache allocation, limiting the available L2 cache space for each CPU core to just 512KB. The reduced available cache space impacts the L2 cache hit percentage, and results in higher power consumption due to more off-chip memory fetches and higher memory latencies that impact CPU performance.

  OS Efficiency: Android OS and WinRT OS assume all available CPU cores are identical with similar performance capabilities. Based on this assumption, the OS schedules tasks to various CPU cores. When multiple CPU cores are each run at different asynchronous frequencies, it results in the cores having differing performance capabilities. This could lead to OS scheduling inefficiencies. In contrast, vSMP technology always maintains all

active cores at a similar synchronous operating frequency for optimized OS scheduling. Even when vSMP switches from the battery saver core to one or more of the main CPU cores, the CPU management logic ensures a seamless transition that is not perceptible to end users and does not result in any OS scheduling penalties.

For more details on NVIDIA Tegra's 4+1 CPU architecture and vSMP technology, please refer to the whitepaper titled *Variable SMP – A Multi-Core CPU architecture for High Performance and Low Power*.

Conclusion

Smartphones and tablets are increasingly being used as personal computing devices. Today's smartphones are no longer being used for just phone calls, messaging, and light Web browsing. Mobile applications for PC-class use cases such as photo editing, word processing, multi-tabbed Web browsing, modern graphics-rich gaming, and multi-tasking are now available for mobile devices and are pushing the performance requirements of these devices. The quad core CPUs in NVIDIA Tegra 4 family of mobile SoCs include several key enhancements that deliver higher performance for the next generation of mobile applications. The quad core Cortex-A15-CPU complex in Tegra 4 includes more execution resources for better instruction level parallelism, larger out-of-order windows, and smarter branch prediction for lower latencies and higher efficiencies. It also includes a large data cache, and larger L2 cache to improve overall data and instruction access speed and reduce penalties associated with cache misses.

The quad core Cortex-A9 r4 processor in Tegra 4i includes several Cortex-A15 class enhancements such as improved branch prediction with larger history buffers, larger TLB size, and a prefetcher unit to deliver much higher performance and power efficiencies than the standard Cortex-A9 architecture.

The benefits of these enhancements are easily seen in benchmark results of the industry-accepted SPECint2000 benchmark, that was designed to mimic the workloads presented by real applications such as word processing, gaming, databases, file compressions, and more. The results of SPECint2000 and other benchmarks highlight that the Tegra 4 family of quad core CPU architectures deliver higher performance than competing architectures on workloads that are expected to be seen in next generation of mobile applications.

More importantly, Tegra 4 family of processors delivers exceptional power efficiency and consumes thirty percent lower power than the previous generation Tegra 3 processor while executing the same workload[1] .

[1] Based on SoC plus memory power consumed on Tegra 3 and Tegra 4 development platforms at the same level of SPECint2000 performance.

APPENDIX

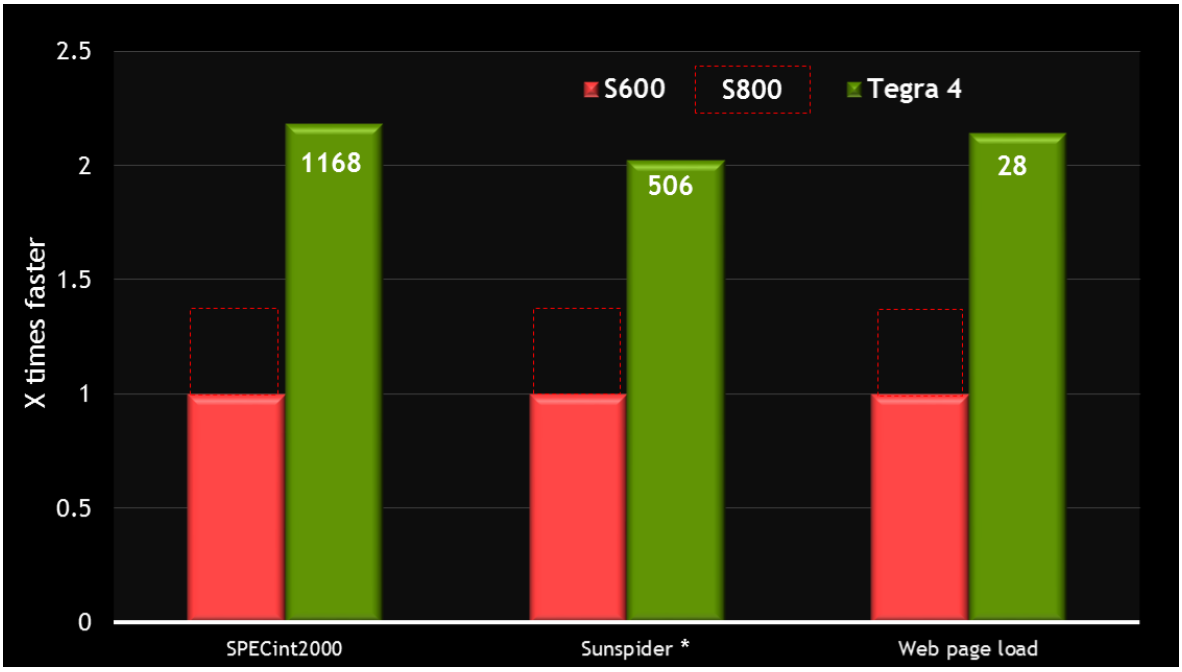

Figure 10 Performance of Tegra 4 vs Competition on CPU and Web Benchmarks[2]

	Benchmark	Result
CPU & System	SpecINT2000	1168
	Sunspider 0.91	506 milli-seconds
	Web Page Load	28 seconds
	WebGL Aquarium (50 fish)	60 fps
	Google Octane	4582
	Kraken 1.1	6799 milli-seconds
	Geekbench 1.0	4285
	Antutu 3.1.1	36127
	Quadrant Pro 2.0	16449
	CFBench 1.1	41227
GPU	GLBench 2.5 HD Egypt (1080p offscreen)	57 fps
	GLBench 2.5 HD Classic (720p offscreen)	274 fps
	Basemark ES 2 Hoverjet	59 fps

Table 1 Measured Tegra 4 Benchmark Results

[2] Tegra 4 scores measured on Tegra 4 development platform. Sunspider score for S600 is from published results. Other scores measured on S4 Pro retail device and scaled to S600 specs. S800 scores are scaled up from S600 based on higher clock frequency and IPC claimed by Qualcomm

Document Revision History

Revision Number	Notes

Notice

ALL INFORMATION PROVIDED IN THIS WHITE PAPER, INCLUDING COMMENTARY, OPINION, NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, “MATERIALS”) ARE BEING PROVIDED “AS IS.” NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE.

Information furnished is believed to be accurate and reliable. However, NVIDIA Corporation assumes no responsibility for the consequences of use of such information or for any infringement of patents or other rights of third parties that may result from its use. No license is granted by implication or otherwise under any patent or patent rights of NVIDIA Corporation. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. NVIDIA Corporation products are not authorized for use as critical components in life support devices or systems without express written approval of NVIDIA Corporation.

Trademarks

NVIDIA, the NVIDIA logo, Tegra and 4-PLUS-1 are trademarks or registered trademarks of NVIDIA Corporation in the United States and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright