CIS 5220 Research Report Fall 2015

Analysis of the ARM® Cortex®-A9 Architecture

Bernice Templeman

Professor David LeVan

23 November 2015

2015

CIS 5220 Research Report Fall 2015

Title: Analysis of the ARM® Cortex®-A9 Architecture

# Introduction

While researching computer architectures, both past and present, I found some architectures that are used on mobile devices [1]. These computer architectures have specialized features that are geared toward mobile devices [1]. However, while these different architectures are used on mobile devices, there is one architecture that is used on the majority of mobile devices [1]. The ARM architecture is used by almost 60% of mobile devices [2]. The ARM Cortex-A9 is one of ARM’s architectures that is currently widely used and stable [3]. My goal in this paper is to provide an analysis and summary of the ARM Cortex-A9 architecture by discussing the architecture’s hardware features to explain how each feature operates and its capabilities.

To achieve this goal, I have organized my paper into six main sections, one of which has sub-sections. The first section is the introduction. In the second section I explain my computer architecture selection. The third section is an overview of the ARM Cortex-A9 architecture. The fourth section is an analysis of each hardware feature: Central Processing Unit (CPU) and its registers, Arithmetic Logic Unit (ALU), Floating Point Unit (FPU), Cache, Memory Management Unit (MMU), the Main memory architecture and the Instruction Fetch and Decode Unit. In the fifth section I discuss three special and interesting features of ARM processors: the MPCore Technology [3], the GPU Compute (Graphics Processing Unit) [3], and the big.LITTLE™ Technology [4]. In the sixth section I conclude my paper with a summary of the ARM Cortex-A9 architecture analysis. Before I begin my analysis of the architecture’s hardware features, however, I need to provide an explanation of my hardware selection, and this is where I go to next.

2. Computer Architecture Selection

My computer architecture selection needed to satisfy two requirements: it is a computer that I want to learn more about and information about the computer must be widely available. I started my selection process by looking at mobile device computer architectures. I began with mobile devices for several reasons. When I decided to go back to school to update my computer skills, my advisor recommended Mobile Application Development. I researched the career outlook for software developers and I found that software developers have one of the best rated jobs [5]. Developers are in high demand, employment growth is higher than average, it pays well [6] and I enjoy software development.

Then I researched mobile application development. I found that there is a high demand for Mobile Application Developers [7]. There has been an increase of mobile devices in the world, and now there are more mobile devices in the world than people in the world. [8][9][10]. There are also more mobile devices than desktops in the world [11]. Therefore, I decided to focus on Mobile Application Development.

To narrow down my search for a computer architecture, I looked at the worldwide use of mobile devices. The majority of devices are Android [12] with the majority of devices in the world using an ARM architecture [13]. ARM Ltd (the company) began in 1990 and its headquarters is located in Cambridge England [14]. The original ARM chip was first developed at a company called Acorn Computers in the 1985 [15]. The processor IP (Intellectual Property) became ARM (Advanced RISC Machine). There is a wide range of ARM processors and ARM (with their partners) has now sold over 60 Billion Chips [14]. ARM does not sell silicon, instead it sells licenses to their partners [14].

The ARM processor has been evolving and ARM currently has several profiles available [16]. I decided to focus on the Application profile since I am interested in Application Development. Next I did some research and concluded that there is information for the Cortex-A9 that is widely available for use. I have selected the ARM Cortex-A9 architecture and now it is time to turn to an overview of the ARM Cortex-A9.

3. Overview of the Architecture

The ARM website gives an overview of the Cortex-A9 architecture and its specifications:

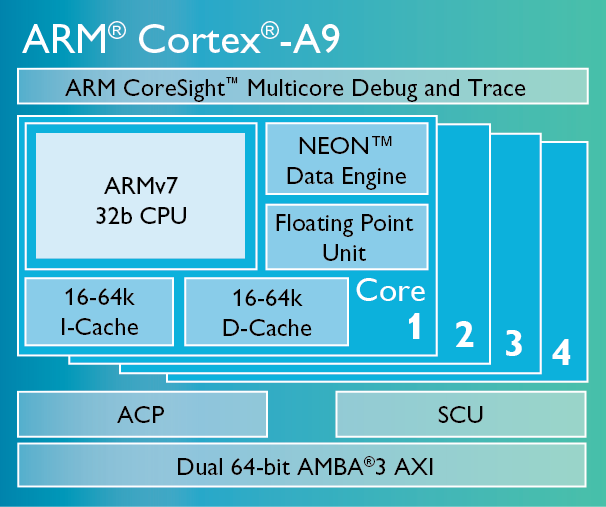
**

Figure 1: ARM®Cortex®-A9 Chip Diagram [3]

From ARM’s chip diagram and chip description on their website:

* The CPU architecture of the ARM Cortex-A9 is ARMv7 32-bit processor
* It is a Multicore Processor that can have 1-4 cores [3].

According to the ARM website, the Main Features of the ARM Cortex A-9 and a brief description are:

* ARM 32-bit Instructions
* Thumb®-2 Technology : 16/32-bit Instruction Set
* NEON™ Data Engine (optional): Advanced SIMD. SIMD is Single Instruction Multiple Data. This improves multimedia performance.
* FPU(optional): Floating Point Unit
* Optimized Level 1 Caches: Level 1 Cache includes both I-cache & D-cache
* Optimized Level 2 Cache Controller (optional) [3].

ARM lists the following as the Multicore Features of the ARM Cortex-A9 and a brief description:

* Snoop Control Unit (SCU): ARM’s Control Unit
* Accelerator Coherency Port (ACP):[AMBA® 3 AXI](http://www.arm.com/products/system-ip/amba/index.php) compatible slave interface on the SCU provides an interconnect point for a range of system masters that for overall system performance, power consumption or reasons of software simplification are better interfaced directly with the Cortex-A9 MPCore processor
* AMBA®: Advanced Microcontroller Bus Architecture
* Generic Interrupt Controller (GIC): Interrupts are hardware prioritized and can be routed between the operating system and [TrustZone](http://www.arm.com/products/processors/technologies/trustzone/index.php)® software management layer
* [TrustZone](http://www.arm.com/products/processors/technologies/trustzone/index.php)® : hardware based system security
* Jazelle® RCT (Runtime Compilation Target) & DBX (Direct Bytecode eXecution) Technology: additional instruction set to ARM & Thumb. This can be used by Java Virtual Machines.
* Debug & Trace [3].

4. Analysis of the Architecture:

In this section is an analysis of the ARM Cortex-A9 hardware features. There are two basic distinguishing architectural features of the ARM® Cortex®-A9. First, According to Steve Furber, who was a principle engineer of the ARM microprocessor, the ARM chip is based on the Reduced Instruction Set Computer [15]. Second, ARM says they have moved from a von Neumann Architecture to a Harvard Architecture with separate instruction and data caches [16]. These features affect the performance, size and power. High performance, small size and low power [3].



Figure 2: Cortex-A9 Microarchitecture [17 ]

## 4.1 Central Processing Unit (CPU) and Registers

From our class and textbook, we learned that a CPU architecture is defined by the basic characteristics and major features of the CPU and it also can be called the Instruction Set Architecture (ISA) [18]. ARM’s website says the processor is based on the RISC (reduced instruction set computer) Architecture because it has:

* A uniform register file load/store architecture, where data processing operates only on

the register contents, not directly on memory contents

* Simple addressing modes, with all load/store addresses determined from register contents and instruction fields only [19].

The ARM instruction sets are described in the University of Washington’s presentation slides on the ARM Architecture:

* The ARM instruction set is a set of 32-bit instructions.
* The Thumb instruction set was developed as a 16-bit instruction set with a subset of the functionality of the ARM instruction set
* Thumb-2 is 16/32-bit set allowing both sizes to increase performance
* Jazelle: 8-bit instructions to execute Java Bytecode

The processor can execute in ARM state, Thumb state or Jazelle state [20].

The ARM processor modes are described in The ARM University Program video, “ARM Architecture Fundamentals” as follows:

There are seven operating modes with their own stack and subset of registers. User mode is the only unprivileged mode. All modes except System & User are Exception modes. These are the modes:

* SVC (supervisor) is entered on reset and when a SVC instruction is executed.
* FIQ is entered when a high priority (fast) interrupt is raised.
* IRQ is entered when a normal priority interrupt is raised.
* Abort is used to handle memory access violations.
* Undef is used to handle undefined instructions.
* System is the Privileged mode using the same registers as User mode.
* User is the mode under which most Applications / OS tasks run [21].

The ARM Cortex-A9 are pictures in the Register Organization Summary from the “ARM Architecture Fundamentals” video [21].

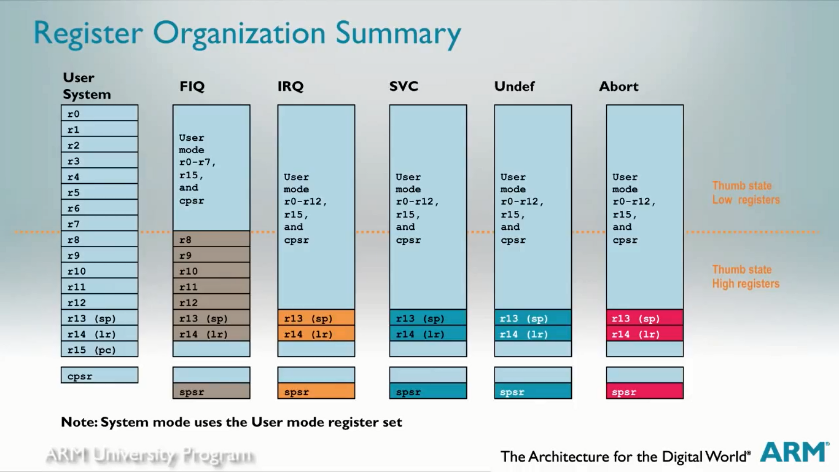
******

Figure 3: Register Summary [21]

Figure 3 is from ”ARM Architecture Fundamentals” which shows there are37 registers [21]. A subset of these registers is accessible in each mode.They are banked [21].

ARM Cortex-A9 Registers described in the ”ARM Architecture Fundamentals” are:

* r0 –r12 are the general purpose registers
* r13 is the sp (stack pointer)
* r14 is the lr (link register)
* r15 is the pc (program counter)
* CPSR (Current Program Status Register)
* SPSR (Saved Program Status Register) [21].

4.2 Arithmetic Logic Unit (ALU)

In our class text book, the ALU is described as the part of the CPU where data is stored temporarily and calculations are made. The ARM’s Data processing instructions are: Arithmetic, Logical, Comparisons and Data Movement [22]. The ARMKeil website says the ARM ALU has a 32-bit barrel shifter that can do shift and rotate operations [23]. The second argument is sent to the ALU via the barrel shifter [22].

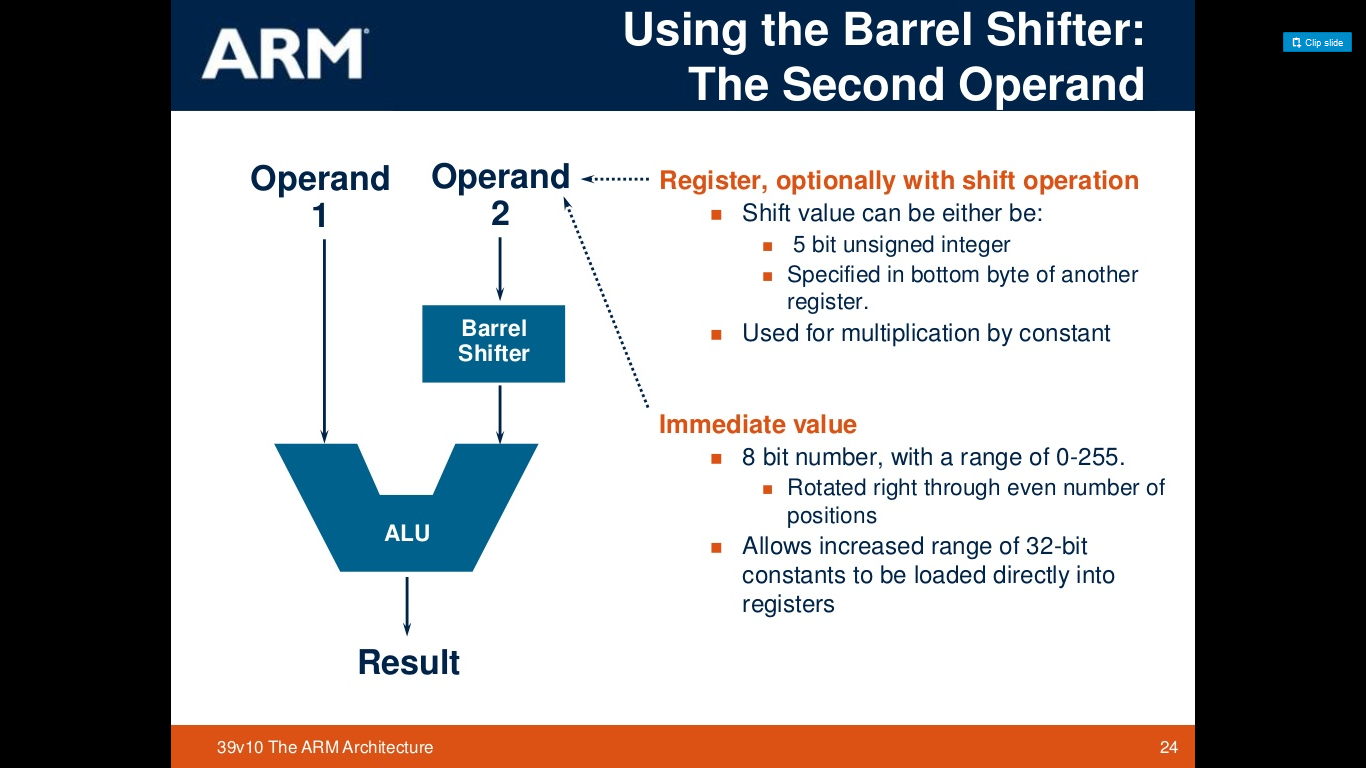
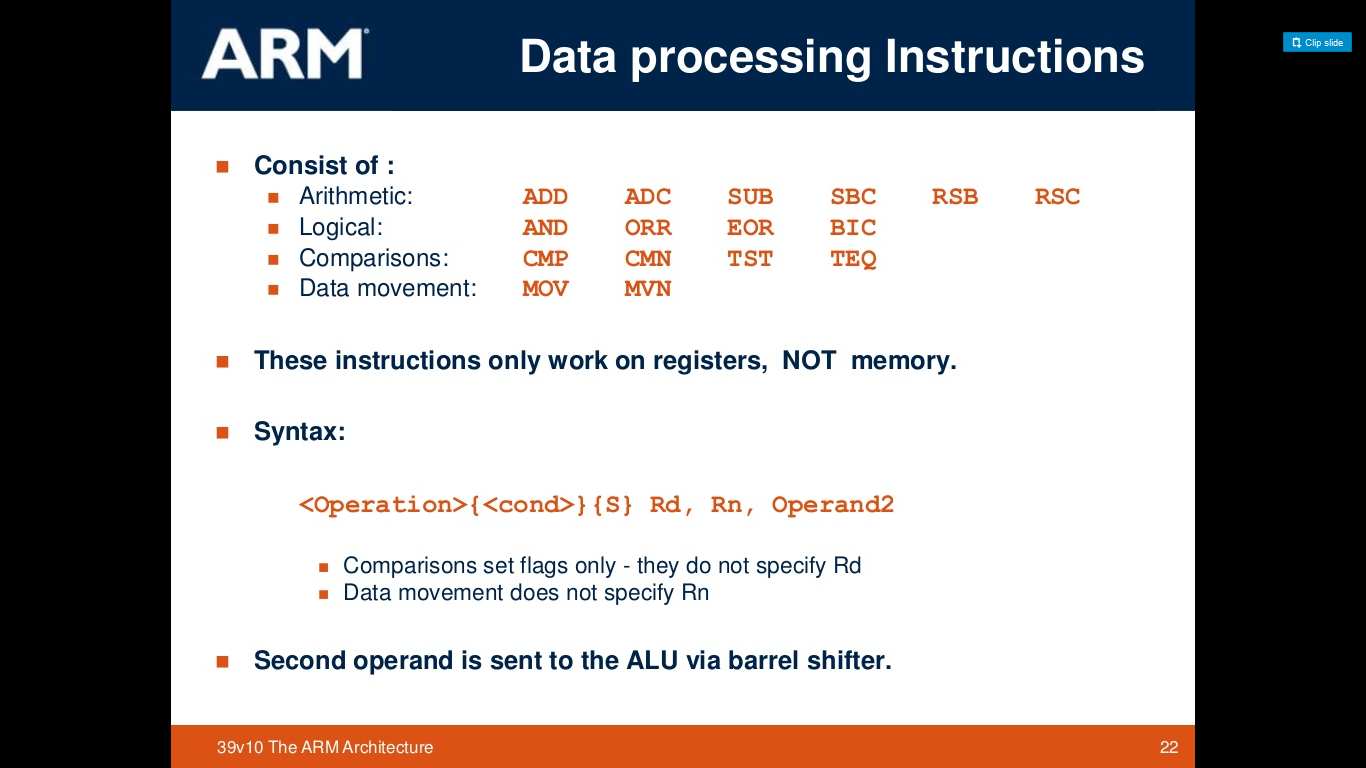


Figure 4: Data Processing [22] Figure 5: Barrel Shifter [23]

4.3 Floating Point Unit (Optional)

The ARM Floating Point architecture (VFP) is optional [3]. The FPU is IEEE 754 compliant [24]. It has its own registers & instruction set [24]. The ARM website says the FPU provides floating-point computation suitable for a wide spectrum of applications such as:

* personal digital assistants and smartphones for graphics, voice compression and decompression, user interfaces, Java interpretation, and Just In Time (JIT) compilation
* games machines for three-dimensional graphics and digital audio
* printers and MultiFunction Peripheral (MFP) controllers for high-definition color rendering
* set-top boxes for digital audio and digital video, and three-dimensional user interfaces
* automotive applications for engine management and power train computations[24][25].

4.4 Cache

The L1 memory system is described in the “Cortex-A9 Technical Reference Manual” as having:

* separate instruction and data caches each with a fixed length of 32 bytes
* The instruction cache is virtually indexed and physically tagged.
* The data cache is physically indexed and physically tagged.
* Store buffer has four 64-bit slots with data merging capability
* 64-bit data paths throughout the memory system
* support for four sizes of memory pages
* export of memory attributes for external memory systems
* support for Security Extensions

The power consumption is decreased by reducing the number of full cache reads because they take advantage of the sequential pattern of many cache operations [26].

The L2 Cache Controller is described in the “Cortex-A9 Technical Reference Manual” as:

* consisting of two 64-bit wide AXI bus masters that ensure high data throughput and maximum performance:
* M0 is the data side bus
* M1 is the instruction side bus and has no write channels.
* L2 Cache is shared, unified, Off-chip, 128KB to 8 MB and 4 to 16-way associative.

This provides low-latency and high-bandwidth access to up to 8MB of cached memory in high-frequency designs, or designs needing to reduce the power consumption associated with off-chip memory access [26].

4.5 Memory Management Unit (MMU)

Steve Furber describes the MMU as:

Modern computer systems typically have many programs active at the same time. A single processor can, of course, only execute instructions from one program at any instant, but by switching rapidly between the active programs they all appear to be executing at once, at least when viewed by the human timescale.

The rapid switching is managed by the operating system, so the application programmer can write his or her program as though it owns the whole machine. The mechanism used to support this illusion is described by the term memory management unit (MMU) [15 ].

The “Cortex-A9 Technical Reference Manual”, states that the MMU:

* maps virtual addresses to physical addresses in L1 & L2
* It also controls accesses to and from external memory
* It has a translation lookaside buffer (TLB) which is a memory cache that stores recent translations of virtual memory to physical addresses for faster retrieval. When a virtual memory address is referenced by a program, the search starts in the CPU
* Instruction side micro TLB
  + 32 fully associative entries
* Data side micro TLB
  + 32 fully associative entries
* Unified main TLB
  + unified, 2-way associative, 2x32 entry TLB
  + support for 4 lockable entries using the lock-by-entry model
  + pseudo round-robin replacement policy
  + supports hardware page table walks to perform look-ups in the L1 data cache [26].

4.6 Main Memory Architecture

ARM moved from a von Neumann to a Harvard Architecture. According to ARM,

The name **Harvard Architecture** comes from the Harvard Mark I relay-based computer. The most obvious characteristic of the Harvard Architecture is that it has physically separate signals and storage for code and data memory. It is possible to access program memory and data memory simultaneously. Typically, code (or program) memory is read-only and data memory is read-write. Therefore, it is impossible for program contents to be modified by the program itself [16].

4.7 Instruction Fetch and Decode Unit

The “Cortex-A9 Technical Reference Manual” states that:

* the L1 instruction side memory system provides an instruction stream to the Cortex-A9 processor
* to increase overall performance and to reduce power consumption, it contains dynamic branch prediction and instruction caching [26].

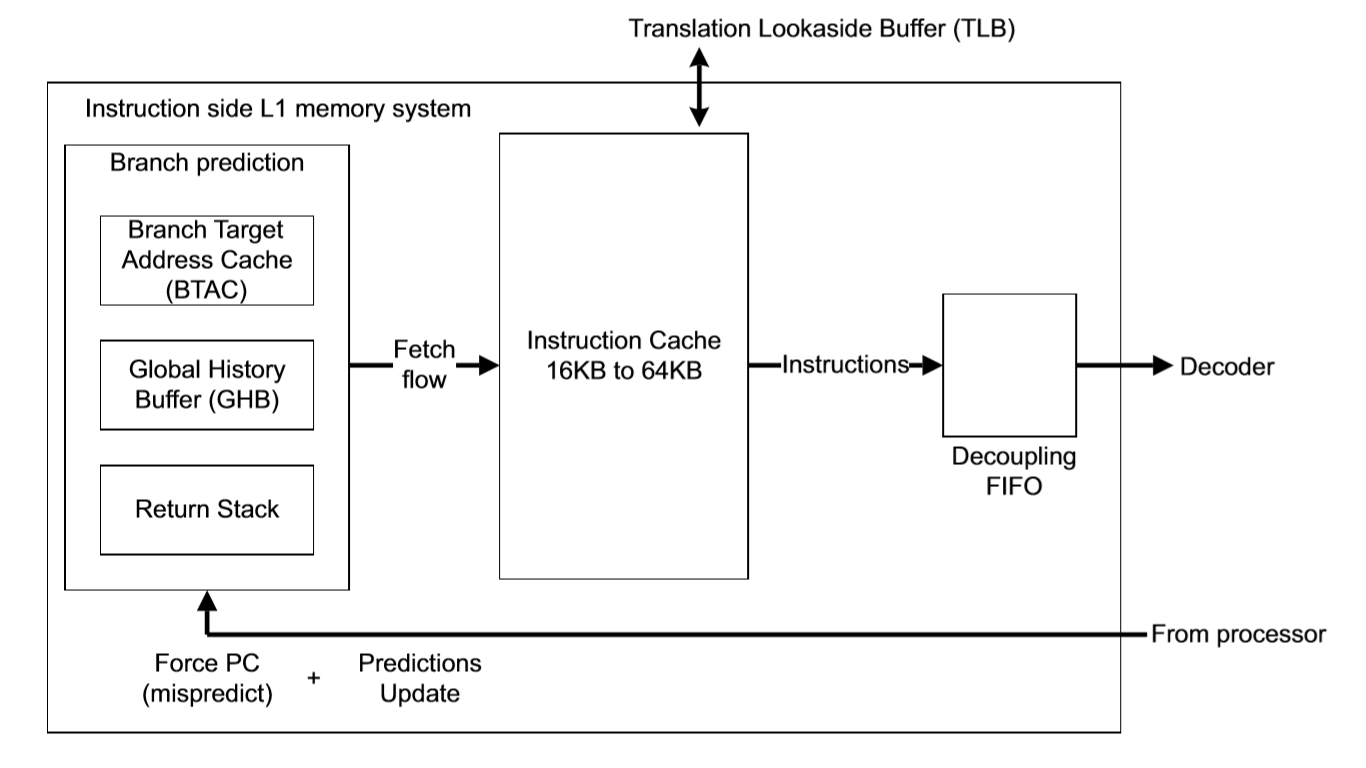


Figure 6: Instruction Fetch Flow [26]

The ARM Cortex-A9 pipeline is a dual-issue superscalar, and out-of-order, dynamic length (8 – 11 stage) pipeline [3].

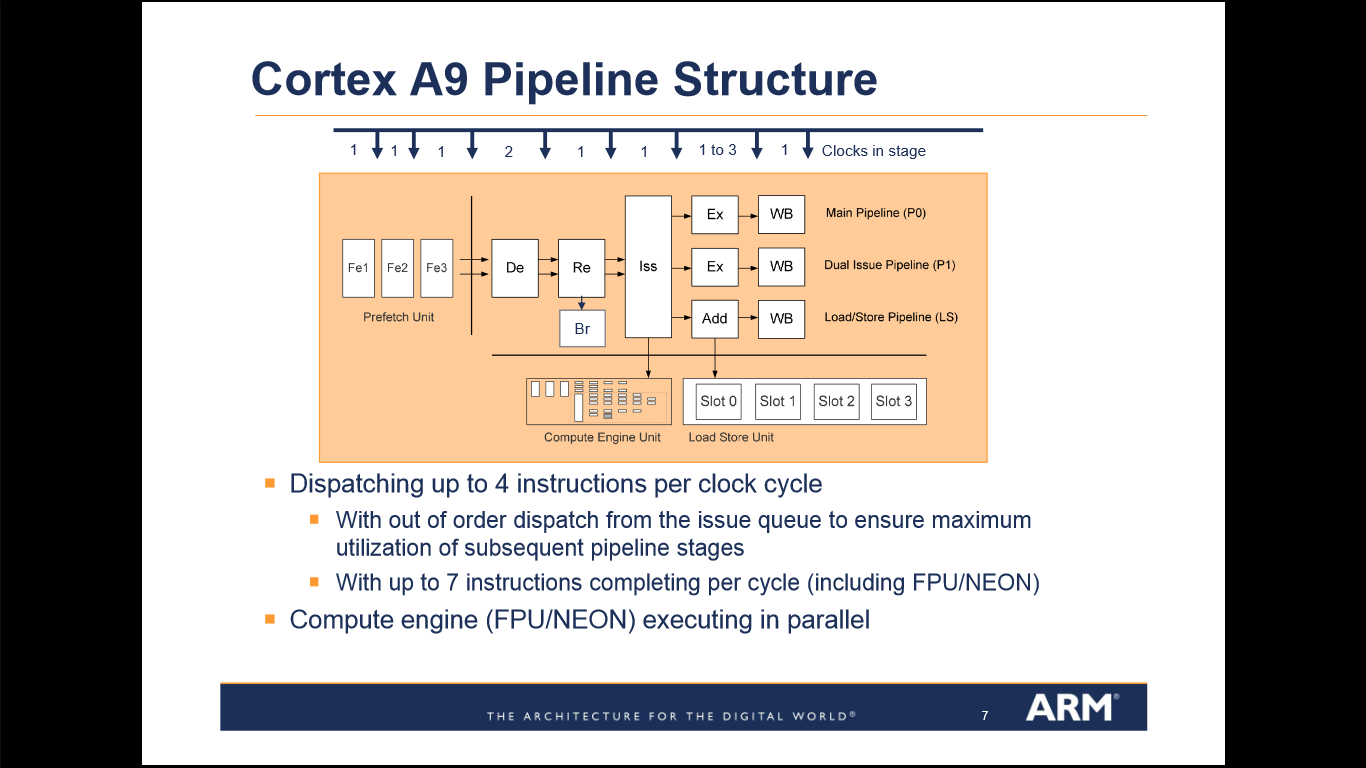


Figure 7:Cortex-A9 Pipeline [17]

From a slide presentation on the Cortex-A9 MicroArchitecture [17] and [27]:

Out-of-order (OoO) instruction dispatch and completion

* Leverages OoO without traditional power/resource hungry reorder buffers
* Supports dispatch of 4 instructions and completion of 7 instruction per clock cycle
* Provides optimal performance from binaries compiled for previous ARM processors

Multiple outstanding memory transactions with out of order completion

* Low load-use penalty of a single clock cycle optimizes benefit from OoO dispatch

Optimized for high performance and low cost [27].

A Pipeline description from Design & Reuse article “The ARM Cortex A9 Processors” says the following:

* Advanced processing of instruction fetch and branch prediction - unblocks branch resolution from potential memory latency-induced instruction stalls.
* Up to four instruction cache line prefetch-pending - further reduces the impact of memory latency so as to maintain instruction delivery.
* Between two and four instructions per cycle forwarded continuously into instruction decode - ensures efficient superscalar pipeline utilization.
* Fast-loop mode - provides low power operation while executing small loops.
* Superscalar decoder - capable of decoding two full instructions per cycle.
* Speculative execution of instructions - enabled by dynamic renaming of physical registers into an available pool of virtual registers.
* Increased pipeline utilization - removing data dependencies between adjacent instructions and reducing interrupt latency.
* Virtual renaming of registers - accelerating code through an effective hardware based unrolling of loops without the additional costs in code size and power consumption.
* Any of the four subsequent pipelines can select instructions from the issue queue - providing out of order dispatch further increasing pipeline utilization independent of developer or compiler instruction scheduling. Ensures maximum performance from code optimized for previous generation of processors and maintains existing software investments.
* Concurrent execution across full dual arithmetic pipelines, load-store or compute engine, plus resolution of any branch each cycle.
* Dependent load-store instructions can be forwarded for resolution within the memory system - further reduces pipeline stalls and significantly accelerating the execution of high level code accessing complex data structures or invoking C++ functions.
* Support for four data cache line fill requests - further reduces stalls due to memory latency with either automatic or user driven prefetching to ensure the availability of critical data.
* Out of order write back of instructions - enables the pipeline resources to be released independent of the order in which the system provides the required data [28].

5.Special or Interesting Features

Moving from a single core to multiple cores becomes more interesting and adds more specialization. There are three special and interesting features that I am mentioning in this section. First is MPCore Technology, which improves performance by combining cores. Each processor has its own L1 cache, but the L2 Cache is shared [27]:

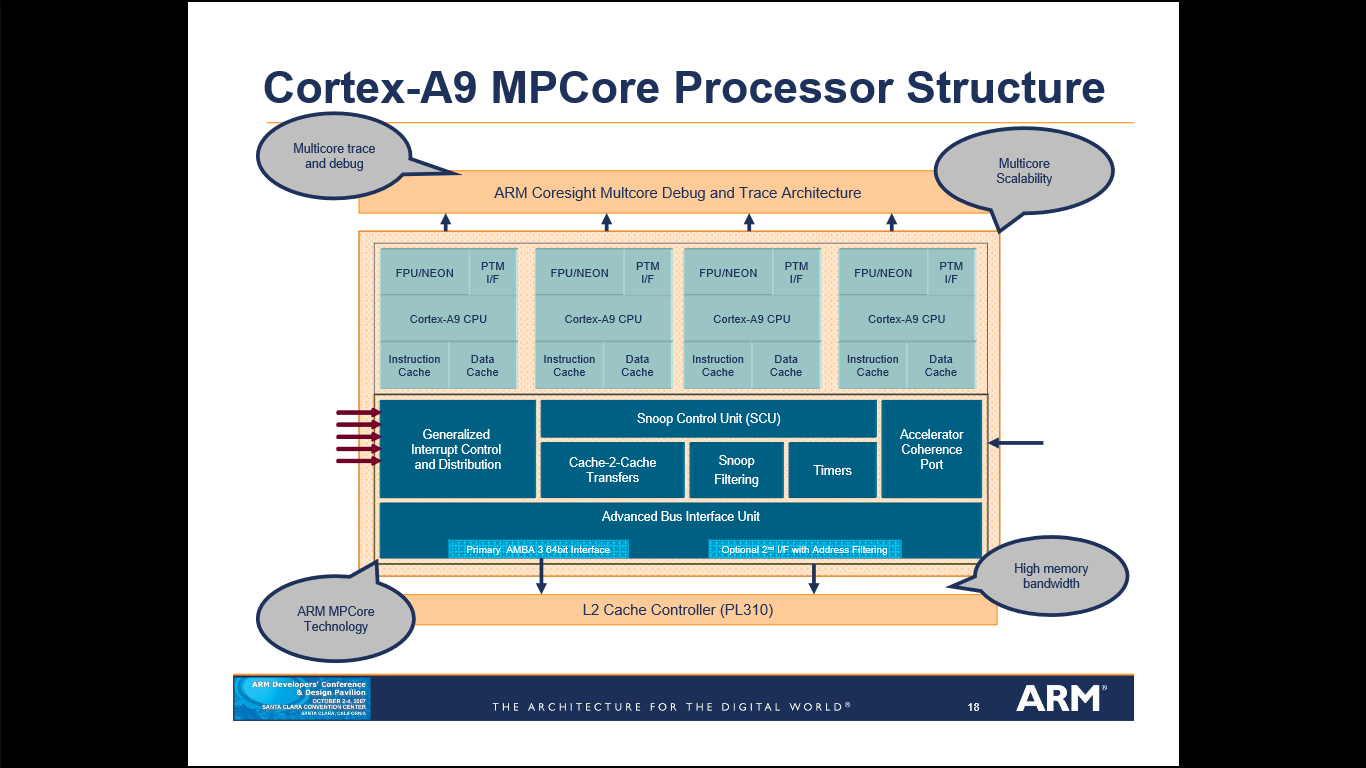


Figure 8:MPCore Processor Structure [27]

The Accelerator Coherence Port enables the sharing of benefits [27].

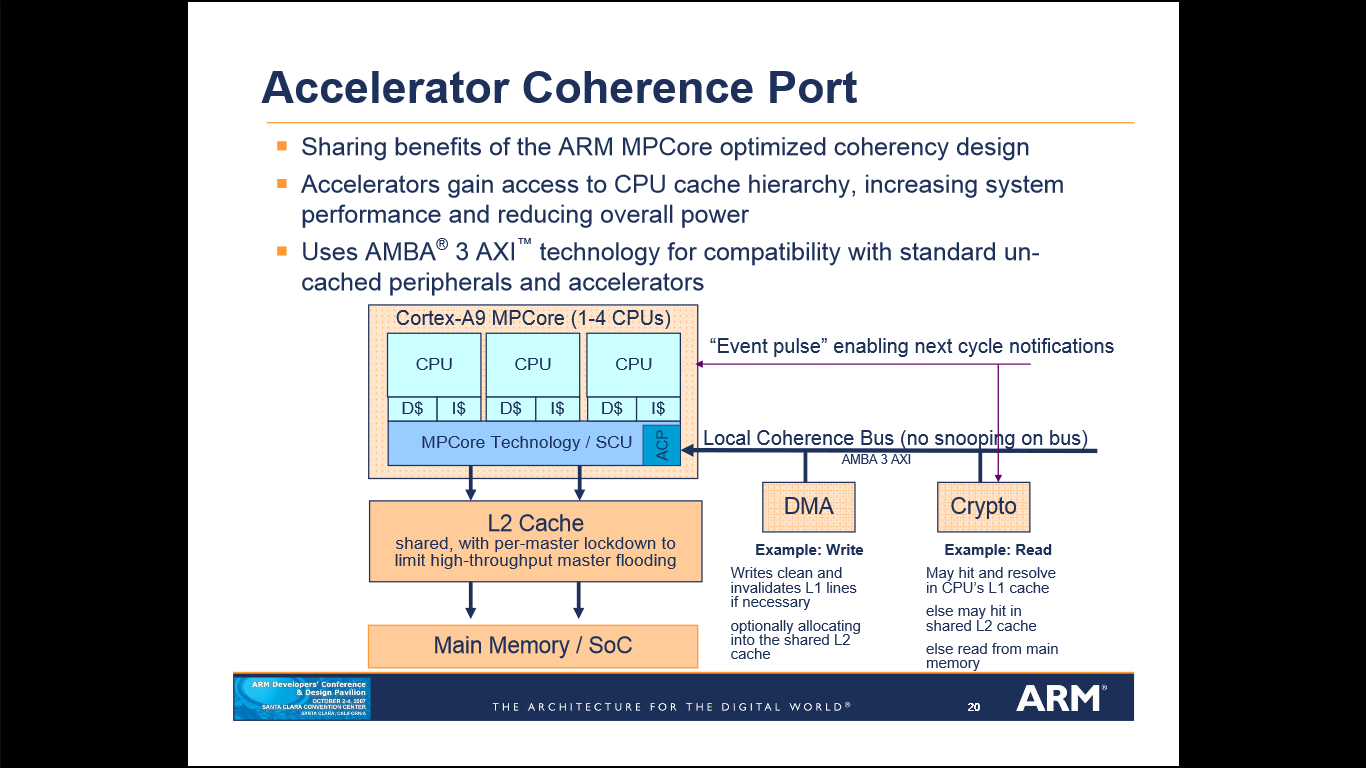


Figure 9:Accelerator Coherance Port [27]

The second special and interesting feature is ARM’s GPU Compute Technology. This is also stated on the ARM website as “Mali™” [3].

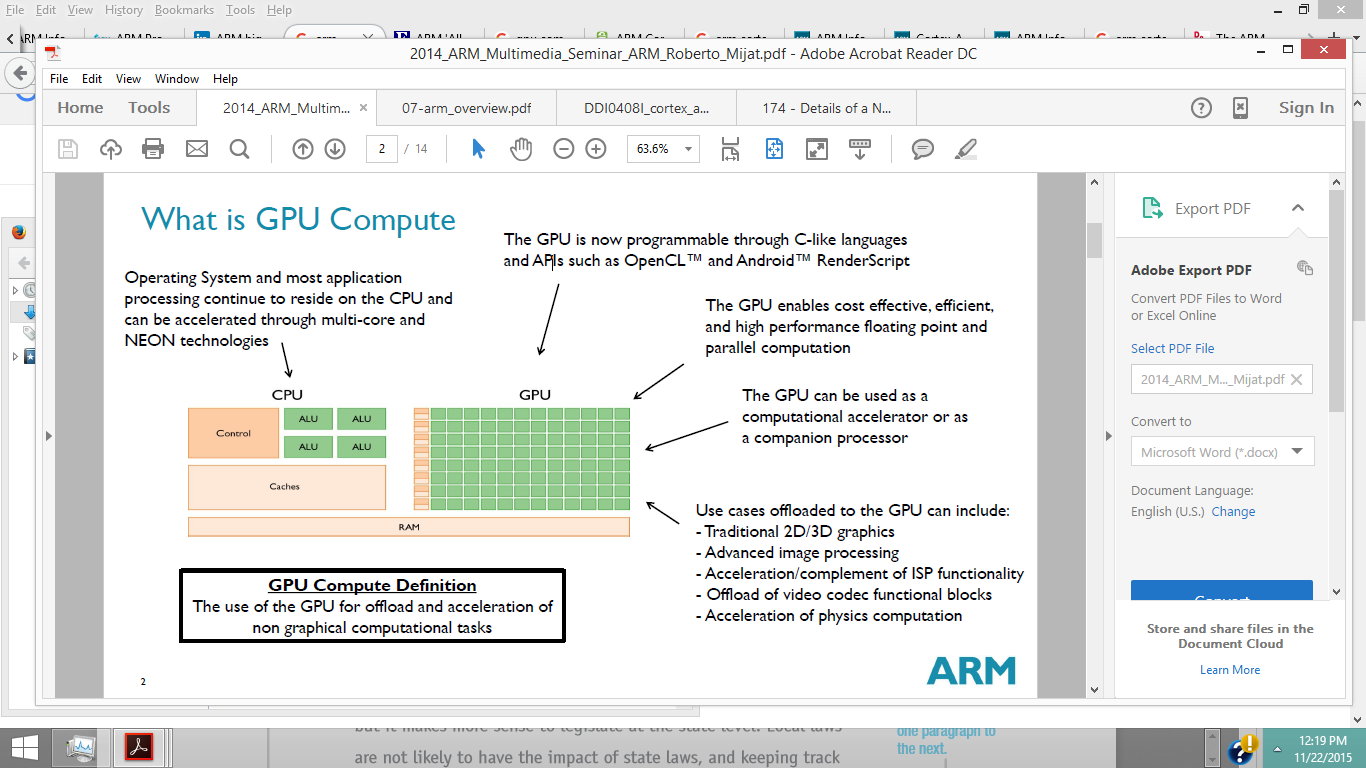


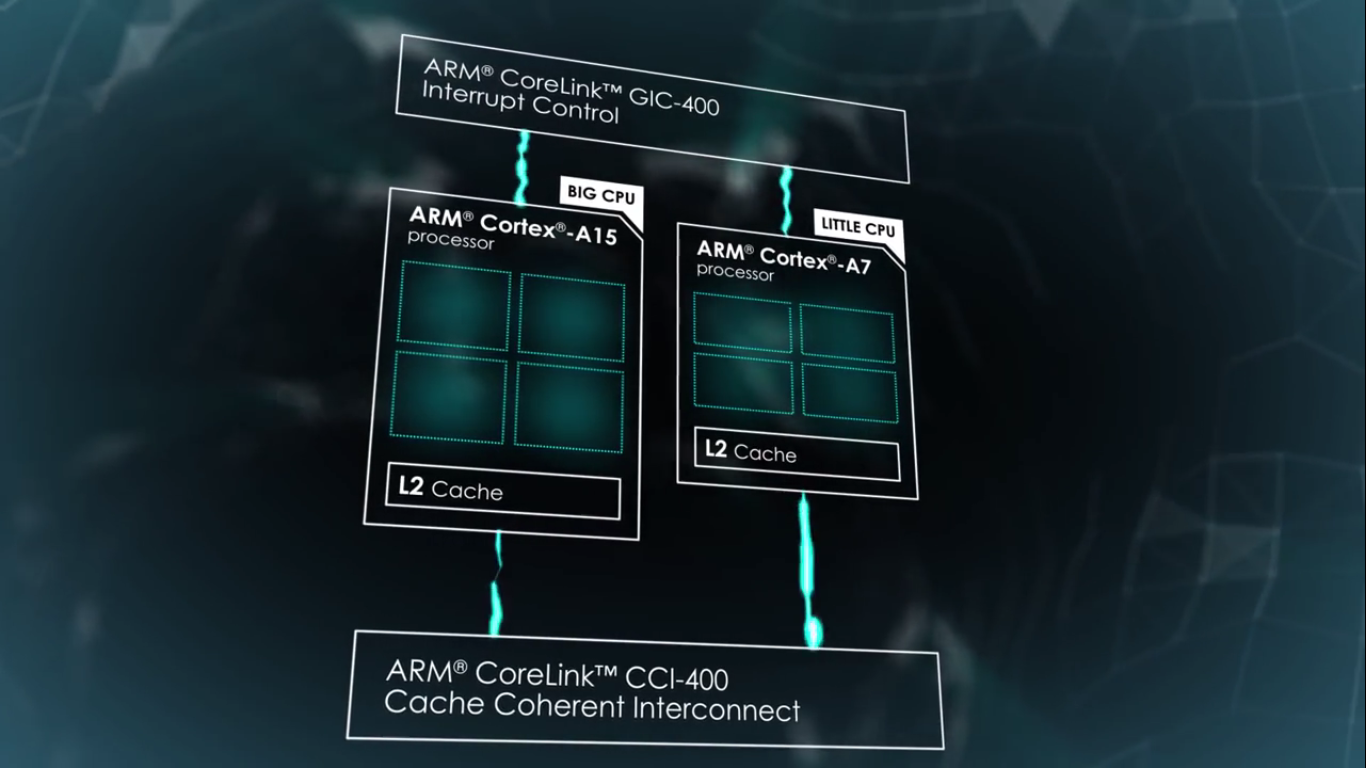
Figure 10: GPU Compute [29]

On Forbes.com, Patrick Moorhead said in his article, “ARM ‘All-In’ with Mobile GPU Compute”:

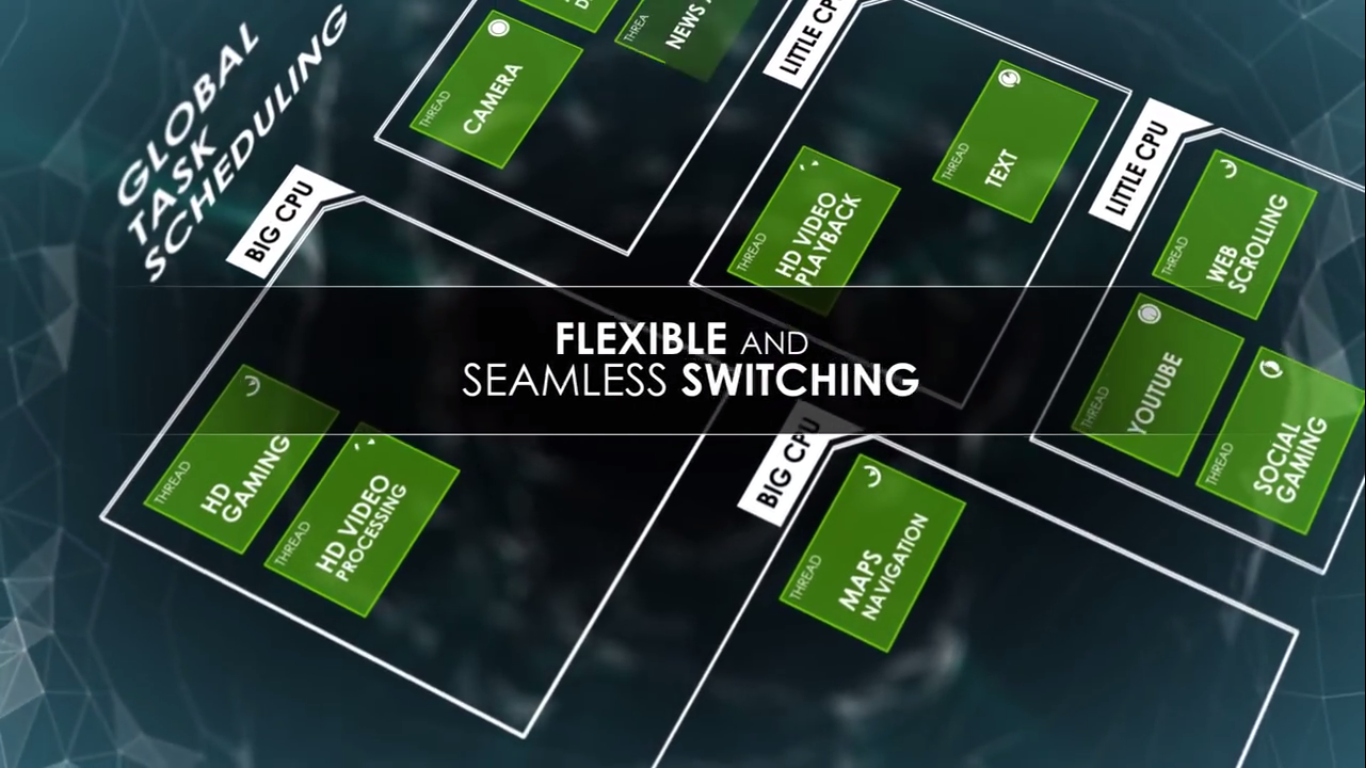
While GPU compute is being done today on the smartphones you are using today, it only gets better in the future.  Today, GPU compute is used today primarily to make games play better and to make pictures look better by assisting with color correction, lens aberration correction, distortion compensation, and noise filtering.  In the next few years, GPU computing will get even better as standards gel, programming becomes more standard and simple, and the performance rises.   This will enable even more powerful and quicker image processing, 4K video editing and effects, super-resolution, improved voice control, security and thoroughly enhanced augmented reality, all without decreasing battery life. [30]

The third special and interesting feature is the big.LITTLE™ Technology [4].ARM® describes big.LITTLE™ processing as:

a power-optimization technology where high-performance ARM CPU cores are combined with the most efficient ARM CPU cores to deliver peak-performance capacity, higher sustained performance, and increased parallel processing performance, at significantly lower average power. The underlying big.LITTLE software, big.LITTLE MP, automatically and seamlessly moves workloads to the appropriate CPU core based on performance needs [4].



big.LITTLE technology takes advantage of the dynamic usage pattern for smartphones and tablets. Periods of high processing intensity tasks such as initial web page rendering and game physics calculation alternate with typically longer periods of low processing intensity tasks such as scrolling or reading a web page, waiting for user input in a game, and lighter weight tasks like texting, e-mail and audio [4].



According to the ARM Connected Community blog “Ten Things to know about big.Little”:

Global Task Scheduling, software can enable all cores to be active at once because the OS is aware of the big and LITTLE cores in the system and is in direct control of thread allocation among the available cores. With Global Task Scheduling the OS power management mechanisms will continue to idle unused cores in the same way it does in standard multi-core systems today [32].

6. Summary

From analyzing the ARM Cortex-A9 architecture by discussing the architecture’s hardware features to explain how each feature operates and its capabilities, I can say that both the

RISC Architecture and Harvard Architecture have had a strong influence on the processors success. Both these architectures have enabled the Cortex-A9 to deliver high performance with low power constraints. ARM says that the following are Energy efficiency features:

* accurate branch and return prediction, reducing the number of incorrect instruction fetch and decode operations
* the use of physically addressed caches, reducing the number of cache flushes and refills, saving energy in the system
* the use of micro TLBs reduces the power consumed in translation and protection lookups for each cycle
* caches that use sequential access information to reduce the number of accesses to the tag RAMs and to unnecessary accesses to data RAMs
* instruction loops that are smaller than 64 bytes often complete without additional instruction cache accesses, so lowering power consumption [26].

ARM also says they have continued to evolve and add low power techniques which include:

* Clock gating
* DVFS: Dynamic Voltage and Frequency Scaling
* Core gating: multiple cores
* Specialized accelerators: Accelerator Coherency Port
* GPU Compute: graphics processing unit
* Big.LITTLE Technology: pairing big processors with small processors [31].

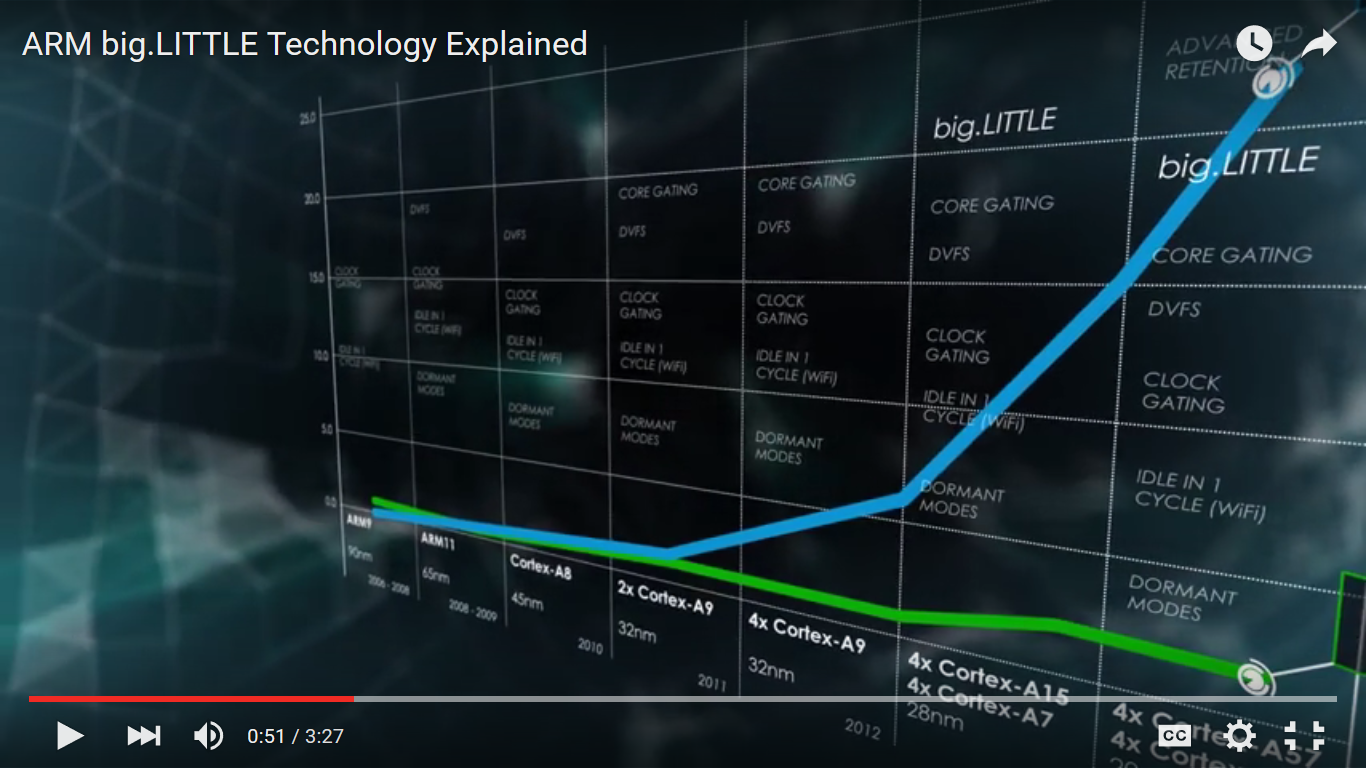


Figure 11:

Leigh Williamson, a Distinguished Engineer from IBM, says:

I really think that mobile is pretty quickly just going to be one channel of delivery for content and services. Already we are seeing a lot of interest in wearables, and some of the implementations related to those actually leverage the power of a mobile device. Mobile devices are getting more and more powerful with every release. They're almost being used now as an intermediate computing platform that bridges between some cloud-based backend services and a smaller wearable device like a wristwatch or an eyeglass device. The mobile phone or tablet becomes a powerful middle tier, with the ultimate endpoint being even more interesting and different than what we see today [7].

BIBLIOGRAPHY

[1] AndroidAuthority. ARM vs X86 – Key Differences Explained! http://www.androidauthority.com/arm-vs-x86-key-differences-explained-568718/

[2] Broadcom. ARM’s Reach: 50 Billion Chip Mileston [VIDEO] http://www.broadcom.com/blog/chip-design/arms-reach-50-billion-chip-milestone-video/

[3] ARM. Cortex-A9 Processor http://www.arm.com/cortex-a9.php

[4] ARM. Big.LITTLE Technology https://www.arm.com/products/processors/technologies/biglittleprocessing.php

[5] U.S.News Money. The 100 Best Jobs http://money.usnews.com/careers/best-jobs/rankings/the-100-best-jobs

[6] U.S.News Money. Best Technology Jobs: Software Developer http://money.usnews.com/careers/best-jobs/software-developer

[7] Keynote. The New Challenges of Managing the Mobile App Lifecycle: A conversation with IBM’s Leigh Williamson about mobile application management https://www.keynote.com/resources/articles/new-challenges-managing-mobile-app-lifecycle

[8] Independent. There are officially more mobile devices than people in the world http://www.independent.co.uk/life-style/gadgets-and-tech/news/there-are-officially-more-mobile-devices-than-people-in-the-world-9780518.html

[9] Census. U.S. and World Population Clock http://www.census.gov/popclock/

[10] GSMA Intelligence. Global Data https://gsmaintelligence.com/

[11] SmartInsights. Mobile Marketing Statistics 2015: Statistics on mobile usage and adoption to inform your mobile marketing strategy http://www.smartinsights.com/mobile-marketing/mobile-marketing-analytics/mobile-marketing-statistics/

[12] Marketing Land. Report: Apple, Android Now 96 Percent of Smartphones Globally http://marketingland.com/report-apple-android-now-96-percent-smartphones-globally

[13] Bloomberg. ARM Designs One of the World’s Most Used Products. So Where is the Money? http://www.bloomberg.com/bw/articles/2014-02-04/arm-chips-are-the-most-used-consumer-product-dot-where-s-the-money

[14] ARM. Company Profile http://www.arm.com/about/company-profile/index.php

[15] Furber, Steve. ARM System Architecture. New York: Addison Wesley Longman Inc., 1996. Print.

[16] ARM. HARVARD VS VON NEUMANN http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.faqs/ka11516.html

[17] ARM. Cortex-A9 Processor Microarchitecture http://web.archive.org/web/20110915171556/http://www.arm.com/files/downloads/Cortex-A9\_Devcon\_2007\_Microarchitecture.pdf

[18] Englander, Irv. The Architecture of Computer Hardware, Systems Software & Networking: An information technology approach. 5th Edition. Hoboken: John Wiley & Sons, 2014. Print.

[19] ARM. ARM Processor Architecture http://www.arm.com/products/processors/instruction-set-architectures/index.php

[20] University of Washington. The ARM Architecture https://courses.cs.washington.edu/courses/cse466/10au/pdfs/lectures/07-arm\_overview.pdf

[21] ARM. “The ARM University Program, ARM Architecture Fundamentals.” Online video clip. *YouTube.* YouTube, 24 Jun. 2013. Web. 15 Nov. 2015. https://www.youtube.com/watch?v=7LqPJGnBPMM

[22] SlideShare. The ARM Architecture http://www.slideshare.net/sreea4/arm-1

[23] ARMKeil: Embedded Development tools. Access to the Inline Barrel Shifter, 2015. http://www.keil.com/support/man/docs/armasm/armasm\_dom1359731140200.htm

[24] ARM. https://www.arm.com/products/processors/technologies/vector-floating-point.php

[25] ARM. Cortex A-9 floating point unit Technical Reference Manual, page 1-3 http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0388e/index.html

[26] ARM. Cortex A-9 Technical Reference Manual, http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0388e/index.html

[27] ARM Developers’ Conference Oct 2007. “Details of a New Cortex Processor Revealed: Cortex-A9” http://rtcgroup.com/arm/2007/presentations/174%20-%20Details%20of%20a%20New%20Cortex%20Processor%20Revealed%20Cortex-A9.pdf

[28] Design & Reuse. The ARM Cortex-A9 Processors http://www.design-reuse.com/articles/16875/the-arm-cortex-a9-processors.html

[29] ARM. GPU Compute http://www.arm.com/files/event/2014\_ARM\_Multimedia\_Seminar\_ARM\_Roberto\_Mijat.pdf

[30] Forbes. ARM ‘All-In’ With Mobile GPU Compute http://www.forbes.com/sites/patrickmoorhead/2013/07/15/arm-all-in-with-mobile-gpu-compute/

[31] ARM. ARM big.LITTLE Technology Explained https://www.youtube.com/watch?v=KClygZtp8mA

[32] ARM. Ten Things to Know About big.LITTLE https://community.arm.com/groups/processors/blog/2013/06/18/ten-things-to-know-about-biglittle