José de Jesus Rodriguez Rivas

COMP 262-01

ARM vs x86 CPU Architecture

ARM and x86 are computer architectures, which is a level above the computer organization. Computer organization refers the hardware of the computer itself, and architecture is the level programmers interact with. CPUs with the same architecture can have widely different organizations. However, the design philosophy of an architecture has an impact on how the organization is designed. To understand the difference between the ARM and x86 architectures, and furthermore the processors that implement them, the difference of RISC and CISC architectures is key[[1]](#footnote-1).

ARM follows a RISC (Reduced Instruction Set Computer) architecture[[2]](#footnote-2). The basic idea of RISC is to reduce the complexity of each instruction in the instruction set. This allows each instruction to be executed in only one clock cycle. This is accomplished by only allowing ALU operations to be performed on registers. To perform operations on memory, the use of LOAD and STORE instructions is necessary. To add two numbers in memory addresses A and B, first LOAD must be used twice to load the two values into registers, then ADD can be used, and finally STORE is used on the result to wherever the value needs to be stored. Although each of the instructions took only one clock cycle, the overall operation took four instructions. The amount of instructions in a single program is increased, but the number of clock cycles per instruction is greatly reduced. It is up to the programmer (or compiler) to optimize the number of instructions used to decrease the execution time of the program. On top of less cycles per instruction, the organization can be made much simpler, which allows for a CPU that is more power efficient. The physical silicon area on the chip itself is smaller, which is why it is power efficient and produces less heat[[3]](#footnote-3).

The x86 architecture is a CISC (Complex Instruction Set Computers) architecture2. The design philosophy behind CISC is to have as few lines of assembly code possible per program. To do this, CISC processors can work on memory directly. So instead of having to LOAD and STORE before and after each call to an instruction, the CPU handles that itself. For example, to add numbers in memory addresses A and B and save the result in A, the only assembly instruction would be ADD A, B. The drawback is that each instruction takes multiple clock cycles. This is because ultimately, the CPU still must read from memory to pass the values through the ALU, and then finally write the result back to memory. To perform all these tasks, the organization of the chip is more complex, making the CPU run hotter, and use more power. However, the chip’s organization can be highly optimized with a much higher clock speed.

It may seem like the difference between the two architectures is arbitrary, but nothing can be further from the truth. The true difference comes in at the application level. ARM CPU’s are much more power efficient than x86, due to their lower clock speeds and simpler organizations. x86 processors are made with much faster clock speeds but require much more power and produce more heat.

# Works Cited

auselen. "How does the ARM architecture differ from x86?" 10 Feburary 2013. *StackOverflow.* Forum. 25 April 2019. <https://stackoverflow.com/questions/14794460/how-does-the-arm-architecture-differ-from-x86/14795541#14795541>.

Baltazar, Gino. "CPU vs GPU in Machine Learning." 13 September 2018. *Oracle + Data Science.* Document. 28 April 2019. <https://www.datascience.com/blog/cpu-gpu-machine-learning>.

Cahyadi, Willy Anugrah. "What is the difference between CPU architecture and GPU architecture?" 29 May 2016. *Quora.* Forum. 28 April 2019. <https://www.quora.com/What-is-the-difference-between-CPU-architecture-and-GPU-architecture>.

Choudhury, Abhinav. "What's the difference between ARM and x86 processors?" 15 June 2014. *Quora.* Forum Post. 25 April 2019. <https://www.quora.com/Whats-the-difference-between-ARM-and-x86-processors>.

Krewell, Kevin. "What's the Difference Between a CPU and a GPU?" 16 December 2009. *nvidia.* Document. 28 April 2019. <https://blogs.nvidia.com/blog/2009/12/16/whats-the-difference-between-a-cpu-and-a-gpu/>.

"risc vs cisc." n.d. *Standford.* Document. 25 April 2019. <https://cs.stanford.edu/people/eroberts/courses/soco/projects/risc/risccisc/>.

CPU vs GPU

The graphics processing unit (GPU) was born from the needs of the video games industry. As developers wanted to launch their games into 3D space, the general-purpose CPU with a couple cores and large ALU’s was no longer enough. They needed a chip that can perform hundreds of complex floating-point calculations at a time, and quickly. The CPU was born out of the need to make computing general-purpose.

The organization of a CPU consists of a small number of cores, which are their own chips that can perform their own instructions. CPU has a cache of shared memory that each of these cores can access, allowing for multithreaded programs to effectively allow the cores to communicate with each other. There is also a large control unit that handles each thread. Inside each core, is a large general-purpose ALU and some individual cache of memory. Though the multiple cores of the CPU do allow for parallel processing, the typical organization of a CPU is more optimized for sequential processing[[4]](#footnote-4). This is because of the high clock speed of a modern CPU.

GPU’s were designed to be able to perform hundreds of floating-point vector calculations all in parallel, so the standard organization of a CPU won’t cut it. The organization of a GPU consists of hundreds of smaller cores, with a handful of small caches and control units that work on a large amount of ALU’s[[5]](#footnote-5). This means that the GPU can run thousands of threads simultaneously, rather than the smaller amount that the CPU can handle[[6]](#footnote-6). GPU’s also have their own memory, known as VRAM, which the processor can perform read and write from much faster than it would be able to if it used the main system’s memory.

Because the control units within a GPU are much smaller, a GPU can perform a smaller set of operations. The idea is that out of the small set of operations that a GPU can perform, it can perform them at incredible speeds, with better efficiency than a CPU. GPU’s are also optimized to perform the same instruction repeatedly, unlike a CPU which will spend more time branching and predicting what instructions will come next.

GPUs have a slower clock speed than CPUs, so when a program cannot be parallelized, the GPU is not the best choice. In modern computing, programs must do both complex serial operations, and large batch data processing and computing. Because of this, programmers are tapping into the best of both worlds. They utilize the high clock speeds and large memory caches of the CPU to perform complex equations or other general-purpose serial processing. When they need to have a parallelized program that can run thousands of the same calculations repeatedly, they utilize the hundreds of ALU’s, and small data caches found in the organization of a GPU.

The reimagining of the GPU as more than just a chip made to render 3D video games has expanded the world of parallel processing. Programmers that write state of the art programs in fields such as artificial intelligence and super computing write their code to utilize all the features that both chips have to offer. One is not necessarily better than the other, because they need to be working together for the huge computing needs of the 21st century.

# Works Cited

Baltazar, Gino. "CPU vs GPU in Machine Learning." 13 September 2018. *Oracle + Data Science.* Document. 28 April 2019. <https://www.datascience.com/blog/cpu-gpu-machine-learning>.

Cahyadi, Willy Anugrah. "What is the difference between CPU architecture and GPU architecture?" 29 May 2016. *Quora.* Forum. 28 April 2019. <https://www.quora.com/What-is-the-difference-between-CPU-architecture-and-GPU-architecture>.

Krewell, Kevin. "What's the Difference Between a CPU and a GPU?" 16 December 2009. *nvidia.* Document. 28 April 2019. <https://blogs.nvidia.com/blog/2009/12/16/whats-the-difference-between-a-cpu-and-a-gpu/>.

1. <https://www.quora.com/Whats-the-difference-between-ARM-and-x86-processors> [↑](#footnote-ref-1)
2. <https://cs.stanford.edu/people/eroberts/courses/soco/projects/risc/risccisc/> [↑](#footnote-ref-2)
3. <https://stackoverflow.com/questions/14794460/how-does-the-arm-architecture-differ-from-x86> [↑](#footnote-ref-3)
4. <https://www.quora.com/What-is-the-difference-between-CPU-architecture-and-GPU-architecture> [↑](#footnote-ref-4)
5. <https://www.datascience.com/blog/cpu-gpu-machine-learning> [↑](#footnote-ref-5)
6. <https://blogs.nvidia.com/blog/2009/12/16/whats-the-difference-between-a-cpu-and-a-gpu/> [↑](#footnote-ref-6)