All about multi-core processors: what they are, how they work, and where they came from

Source: http://www.pcmech.com/article/all-about-multi-core-processors-what-they-are-how-they-work-and-where-they-came-from/

OCTOBER 2, 2015 BY CHRISTIAN DE LOOPER

While in the past most computers have only had one single processor core to do all the work, nowadays it's not uncommon to see computers, phones, and other devices with multiple cores. These cores reside in the same, single, CPU, or Central Processing Unit.

Having multiple cores is a big advantage. With only one core, a computer can only work on one task at a time, having to complete a task before it moves onto another. With more cores, however, a computer can work on multiple tasks at once, which is especially useful for those who do a lot of multitasking.

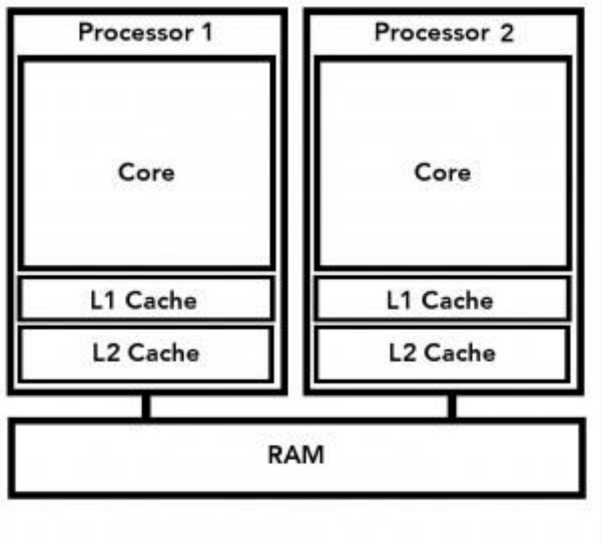
Before diving into exactly how multi-core processors work, it's important to talk a little about the backstory of processing technology, after which we will discuss what multi-core processors do.

Some History

Before processors with multiple cores were built, people and companies such as Intel and AMD tried to build computers with multiple CPUs. What this meant was that a motherboard with more than one CPU socket was needed. Not only was this more expensive, because of the physical hardware needed for another CPU socket, but it also increased latency because of the increased communication that was needed to take place between the two processors. A motherboard had to split data up between two completely separate locations in a computer rather than simply sending all of it to the processor. Physical distance does in fact mean that a process is slower. Putting these processes on one chip with multiple cores not only means that there is less distance to travel, but it also means that different cores can share resources to perform particularly heavy tasks. For example, Intel's Pentium II and Pentium III chips were both implemented in versions with two processors on one motherboard.

After a while, processors needed to be more powerful, so computer manufacturers came up with the concept of hyper-threading. The concept itself came from Intel, and it was first conceived in 2002 on the company's Xeon server processors, and later on its Pentium 4 desktop processors. Hyper-threading is still used today in processors, and is even the main difference between Intel's i5 chips and its i7 chips. It basically takes advantage of the fact that there are often unused resources in a processor, specifically when tasks don't require much processing power, which could be put to use for other programs. A processor that uses hyper-threading basically presents itself to an operating system as though it has two cores. Of course, it doesn't really have two cores, however for two programs that use half of the processing power available or less, there may as well be two cores because of the fact that together they can take advantage of all the power that the processor has to offer. Hyper-threading will, however, be slightly slower than a processor with two cores when there isn’t enough processing power to share between the two programs using the core.

You can find an insightful video giving a brief, more detailed explanation of hyper-threading here.



Multi-Processors

After much experimentation, CPUs with multiple cores were finally able to be built. What this meant was that one single processor basically had more than one processing unit. For example, a dual-core processor has two processing units, a quad-core has four, and so on.

So why did companies develop processors with multiple cores? Well, the need for faster processors was becoming more and more apparent, however developments in single core processors were slowing down. From the 1980s until the 2000s, engineers were able to increase processing speed from several megahertz to several gigahertz. Companies like Intel and AMD did this by shrinking the size of transistors, which allowed for more transistors in the same amount of space, thus improving performance.

Because of the fact that processor clock speed is very linked to how many transistors can fit on a chip, when transistor shrinking technology began to slow, development in increased processor speeds also began to slow. While this is not when companies first knew about multi-core processors, it is when they started experimenting with multi-core processors for commercial purposes. While multi-core processors were first developed in the mid 1980s, they was designed for large corporations, and were not really revisited until single-core technology began to slow.

The first multi-core processor was developed by Rockwell International, and was a version of the 6501 chip with two 6502 processors on one chip.

What Does A Multi-Core Processor Do?

Well, it’s really all pretty straightforward. Having multiple cores allows for multiple things to be done at once. For example, if you’re working on emails, have an Internet browser open, are working on an excel spreadsheet, and are listening to music in iTunes, then a quad-core processor can work on all of these things at once. Or, if a user has a task that needs to be completed right away, it can be split up into smaller, easier to process tasks.

Using multiple cores is also not just limited to multiple programs. For example, Google Chrome renders each new page with a different process, meaning that it can take advantage of multiple cores at once. Some programs, however, are what’s called single-threaded, which means they were not written to be able to use multiple cores and as such cannot do so. Hyper-threading again comes into play here, allowing Chrome to send multiple pages to two “logical cores” on one actual core.

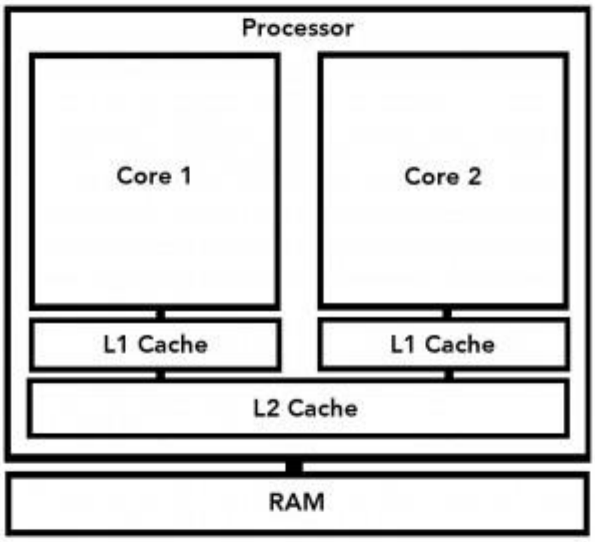
Going hand in hand with multi-core processors and hyper-threading is a concept called multithreading. Multithreading is essentially the ability for an operating system to take advantage of multiple cores by splitting code up into its most basic form, or threads, and feeding it to different cores simultaneously. This is, of course, important in multi-processors as well as multi-core processors. Multi-threading is a little more intricate than it sounds, as it requires operating systems to properly order code in a way that the program can continue to run efficiently.

Operating systems themselves do similar things with processes of their own – it’s not just limited to applications. Operating system processes are things that the operating system is always doing in the background, without the user necessarily knowing it. Because of the fact that these processes are always going on, having hyper-threading and/or multiple cores can be very helpful, as it frees up the processor to be able to work on other things like what is happening in apps.

How Do Multi-Core Processors Work?

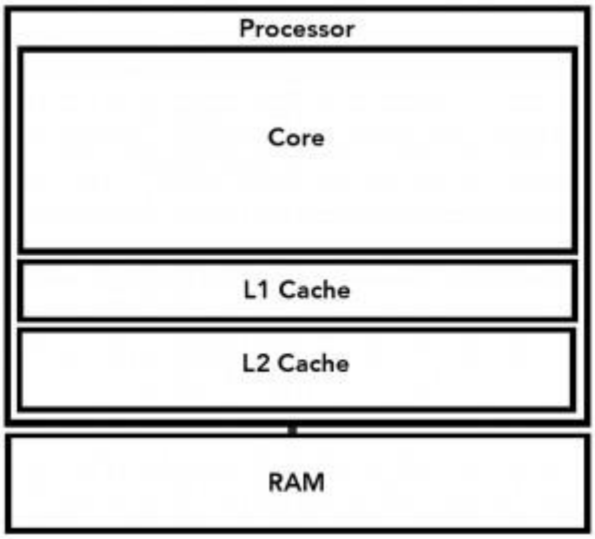
First, the motherboard and the operating system need to recognize the processor and that there are multiple cores. Older computers only had one core, so an older operating system might not work too well if a user tried to install it on a newer computer with multiple cores. Windows 95, for example, doesn’t support hyper-threading or multiple cores. All recent operating systems support multi-core processors, including the likes of Windows 7, 8, the newly released 10, and Apple’s OS X 10.10.

Put basically, the operating system then tells the motherboard that a process needs to be done. The motherboard then tells the processor. In a multi-core processor, the operating system can tell the processor to do multiple things at once. Essentially, through the direction of the operating system, data is moved from the hard drive or RAM, via the motherboard, to the processor.



Multi-Core Processor

Within a processor, there are multiple levels of cache memory that hold data for the processor’s next operation or operations. These levels of cache memory ensure that processor don’t have to look very far to find their next process, saving a lot of time. The first level of cache memory is the L1 cache. If the processor cannot find the data it needs for its next process in the L1 cache, it looks to the L2 cache. The L2 cache is larger in memory, but is slower than the L1 cache.



Single Core Processor

If a processor cannot find what it’s looking for in L2 cache, it continues down the line to L3, and if a processor has it, L4. After that, it will look in main memory, or the RAM of a computer.
There are also different ways in which different processors handle the difference caches. For example, some duplicate the data on the L1 cache on the L2 cache, which is basically a way to ensure that the processor can find what it is looking for. This does, of course, take up more memory in the L2 cache.

Different levels of cache also come into play in multi-core processors. Usually, each core will have its own L1 cache, but they will share L2 cache. This is different from if there were multiple processors, because each processor has its own L1, L2 and any other level cache. With multiple single-core processors, cache-sharing is simply not possible. One of the main advantages to having a shared cache is the ability to use a cache to its fullest, because of the fact that if one core isn’t using the cache, the other can.

In a multi-core processor, when searching for data a core can look through its own unique L1 cache, and will then branch out to shared L2 cache, RAM, and eventually the hard drive.
It is likely that we will continue to see the development of more cores. Processor clock speeds will surely continue to get better, albeit at slower rates than before. While it’s now not uncommon to see octa-core processors in things like smartphones, soon enough we could see processors with dozens of cores.