Jagdeep Singh

Great Ideas in Computer Science

Professor: Zhaojun Bai

Future of the CPU

Humans, in the last century have progressed further in technology than all previous years of mankind combined. Perhaps the invention of the transistor was one of mankind's greatest innovations. The transistor has played a vital role in the development and advancement of computers. The transistor was shrunken down to nanometers and put onto silicon die creating CPUs. It's hard to imagine how a simple switch that controls current became one of the foundations for computers. Thanks to Moore's Law the number of transistors has doubled each year and continues to do so. But an interesting trend was brought up in 2005 in the article "The Free Lunch is over" by Herb Sutter. The number of transistors has been growing on CPUS, following Moore's law but the clock rate of CPUs has started to plateau. Meaning in the last few years the speed of processors has not improved. So that raised a question of what will the future of the CPU be if they aren't getting faster? Since manufacturers and engineers cannot increase the clock speed of processors they have worked around the issue to build more powerful CPUs of the future. Some of these innovations that engineers have created are already in today's market such as new Architecture, increased cache, multicore, and hyper threading.

Engineers found that making a more powerful and faster CPU requires an efficient Architecture. Architecture plays a crucial role on how a CPU handles an instruction set. The Architecture of a CPU is like the layout of a warehouse; if the items on a list are easy to grab and take out, the less time and effort it will require for the task to get done. CPU architecture works in the same manner, the CPU's control unit translates the instruction, plugs data from the memory into the Arithmetic Logic Unit, which in turn does calculations and spits out an answer. This is the most basic Architecture called the Von Neumann Architecture (Riley, 1987). All processors have some form of a von Neumann Architecture. How a CPU handles an instruction set greatly impacts its performance. Some examples of various architectures in the market are, ARM an architecture used for mobile computing it's usually used for a specific task and has less transistors, and the X86 architecture which is used for heavy tasks and desktops. The X86 architecture can handle heavy instruction sets compared to ARM and have numerous more transistors, as well as cache (Hectronic).

Now that the role of Architecture has been explained very briefly, one can see that cache is constantly being read every time a task is done. The processor can only pull memory as fast as the

speed of the memory or cache allows. Sutter notes that cache is a key component of faster processors of the future. Cache is the fastest form of memory; most commonly used data is stored on it and it is directly on the CPU. If the data a CPU needs is on the cache the calculation can be performed significantly faster. Cache can be up to 5 GHz in speed, most memory on the market can only reach 2.1 GHZ the speed is significantly valuable when the processor is doing millions of calculations. In short the more cache a processor has the more data it can access quicker.

After reaching a plateau in speed, engineers decided to incorporate multiple CPUs on one chip. Each core or CPU can execute a single thread of instructions, adding on multiple cores allowed multiple threads. This method allowed significant improvement and power to the cpu but it came at a cost. Multi core processors utilized more energy and as the number of cores increased it became more difficult to manage these cores. Adding more cores also made it more difficult to increase the clock rate of the cpu. Engineers did find a solution to this issue which Intel calls Hyper Threading. Normally a processor can only execute one instruction at a time in which each stream of data or thread must be executed individually. Hyper threading makes it possible for each processor core to schedule and assign resources to two threads at once. . The operating system sees four cores in a hyper threaded dual core CPU, in reality there are two logical cores and two physical cores. The software does not see the difference so it assigns multiple threads to the same core. Another way to explain Hyper Threading is thinking of the cpu as a worker in a factory with one conveyer belt. The conveyer belt brings the worker various tools to do his work. Sometimes there is a delay between the tools which causes the worker to idle. But if you add a second conveyer belt this gets rid of the delay and allows the worker to do more task and allocate his resources to two different conveyer belts. This is essentially what hyper threading does. Hyper threading has its benefits of allowing processors to handle heavier tasks, but it did not help in increasing clock speed. Hyper threading can be efficient but it does not bring the same amount of power as adding more cores. Its performance boost is about thirty percent over a non-hyper threaded CPU.

As one can see engineers are facing a tough challenge. They have thought of various ways to increase processor efficiency and speed all while working around the clock speed issue. They cannot work around it for long though, Joel Hruska brought up Amdahl's Law to explain why. Adding more cache and improving on architecture will continue out but adding multiple cores, which give the biggest boost in performance, will have no effect after a certain point. Amdahl's Law states that even if software is 95% programmed to run parallel it will only run about 20 times faster due to the limitation by the amount of serial code. In essence going beyond an X amount of cores for an X% of a program that can be run parallel will make no difference in performance (Hruska 2012).

The key to getting speedier processors as it turns out is not clock speed but what you can

accomplish in one cycle. The faster and more efficient you can zip data the more you can get accomplished in one cycle. Decreasing the size of the processors has given positive results but the movement of electrons in transistors becomes chaotic due to quantum mechanics (Strickland). Electrons can literally teleport through transistors once the transistors become less than ten atoms thick. This causes leakage of current resulting in all sorts of functional and structural issues in processors (Strickland). IBM has currently been researching in using lasers and fiber optics to transfer data up to 100x faster than traditional copper (Hodgin 2007). There are many issues and hurdles to overcome but processors still have the potential to get more powerful, it's just a matter of time.

Citations

Hectronic. "Enabling Your Next Step." *ARM versus X86*. N.p., n.d. Web. 17 Mar. 2013. <http://www.hectronic.se/website1/embedded/arm-versus-x86/arm-versus-x86.php>.

Hodgin, Rick C. "IBM Breakthrough, One Step Closer to Optical Computers | TG Daily." *TG Daily*. N.p., 6 Dec. 2007. Web. 17 Mar. 2013. <http://www.tgdaily.com/business-and-law-features/35171-ibm-breakthrough-one-step-closer-to-optical-computers>.

Hruska, Joel. "ExtremeTech." *ExtremeTech*. N.p., 1 Feb. 2012. Web. 17 Mar. 2013. <http://www.extremetech.com/computing/116561-the-death-of-cpu-scaling-from-one-core-to-many-and-why-were-still-stuck/2>.

Strickland, Jonathan. "How Small Can CPUs Get?" *HowStuffWorks*. N.p., n.d. Web. 17 Mar. 2013. <http://computer.howstuffworks.com/small-cpu2.htm>.

Sutter, Hurb. "The Free Lunch Is Over: A Fundamental Turn Toward Concurrency in Software." *The Free Lunch Is Over: A Fundamental Turn Toward Concurrency in Software*. N.p., 30 Mar. 2005. Web. 17 Mar. 2013.