https://link.springer.com/article/10.1007/s42514-023-00175-7

I read a paper on an algorithm called thSORT, which is a parallel sorting algorithm optimized to run on a processor called FT-M7032. This is a processor developed by the University of Defense Technology, and its main feature is that it's a hybrid chip featuring general usage CPU cores and additional DSP cores intended to be used in high performance computing. DSP (Digital Signal Processor) cores are mostly used in applications that require the processing of many mathematical operations based on analog inputs in real time. Common applications include the use in audio and video processing, and DSPs allow for conversion of the analog signal to digital, perform an operation such as filtering, and then convert back to analog. Its main advantage over traditional CPU cores is that their hardware architecture is optimized exclusively for signal processing, and as such can run these tasks with high power efficiency and require minimal cooling. thSort intends to utilize these characteristics of DSP cores to perform parallel sorting in a power efficient manner.

This paper was interesting because I had never thought about the impacts of hardware architecture on the usages of algorithms. We talked about in class the difference between theoretical time complexity and actual run time and how they might differ, for example, quick sort often being faster than merge sort due to its lower overhead required in most implementations. This introduces another perspective of how the underlying hardware executes the instructions, and how high-performance algorithms must take this into account. This could potentially mean that in some cases, an algorithm with a theoretically worse time complexity based on comparisons would run faster since it requires less instructions or utilizes hardware acceleration better.