Single Instruction, Multiple Data (SIMD)

The following figure shows how one instruction is operated on multiple data where the central processing unit (CPU) will carry out the same single transaction on multiple instances of data stream:

The idea of SIMD was introduced in the late 1980s. It was a breakthrough in utilizing the CPU to its full capability and maximum performance so that it can process the same instruction on multiple different instances of data. The idea behind it is to use more than one single instruction, single data (SISD) architecture and serve the data stream or packets into more than one processing procedure. The SISD exists in any computer, but the application of the SISD into a SIMD implementation can and will enforce the use of the processor into a multiprocessing, multitasking, and multidata stream service and application of instruction. The old computer configuration based on Flynn’s taxonomy clearly is centered toward SIMD. As people move to the Duo Core, Quad Core, i3, i5, and i7 computers, instruction–CPU interaction and utilization on data is based on MIMD. It is very important to realize that the hardware is always evolving, which forces the software to evolve in return. Gone are the days where you had to write a cryptic command on the command line. Everything now is based on Windows, and it forces the execution of multiple instruction sequences of different commands to operate on different data at the same time. It is clear that people are buying into the concept of multiprocessing because they have enjoyed the concept of multitasking.
SIMD Versus MIMD

Multiple Instruction, Multiple Data (MIMD)

The idea of MIMD was also introduced in the late 1980s (Tanenbaum, 2012). However, it has forced the architecture, implementation, and use in the last 5 years to achieve parallelism. The days have passed where you have only one CPU on your motherboard. The CPUs will share memory and utilize the concept of multiple sequences of instructions on multiple data streams. Nowadays, users buy computers with anywhere from 2–8 CPUs in them.

It is very complex to microprogram the control signals in a computer. It is a hard procedure to hardwire the controls in a processor. The invention of microprogramming has added the elements of more flexibility, productivity, controllability, and discretionability (Tanenbaum, 2012). In reality, microprogramming interfaces with the firmware, which are the burnt instructions on the CPU to handle the hardware. Firmware is the software and hardware combination that is sometimes referred to as the Kernel, and it handles the control and use of the hardware. Once you are able to use the microprogramming of the CPU, designing the control unit (CU) becomes very easy and simple. It is highly logical to go that route. Hardwired is very difficult, mundane, complex, and not user friendly.

Reference