Patrick V. Mediodia

Discussion and Essay

Most computers today run on the x86 or x64 architecture. Throughout the years, the x86 architecture has evolved to what we know today and is backwards compatible with older versions. This is the reason why we can run software written in x86 on many different computers. x86 computers have the same underlying architecture but have a different organization or how instruction sets are executed. Instruction sets are used to control and give instructions to the CPU which is the Assembly Language.

It is important to learn assembly so that we can understand the inner workings of our modern programming languages, this is because high-level languages used by programmers today such as C, Java, Python, etc. are all compiled down to assembly. Assembly is then converted to its corresponding machine code based on the instruction set. Machine language is the only language that computers understand. However, machine language is difficult for humans to understand as it is only composed of 1s and 0s. Because of this, early programmers have created assembly languages that map machine language to human-readable mnemonics. Understanding what goes under the hood is beneficial and important to realize the need to create efficient solutions. Programmers today are lucky to have high-level programming languages which apply the concept of abstraction, hiding the complexity of assembly and machine language so that it is easier to create and develop software applications.

Although assembly may be tedious to write, assembly is still being used today for direct hardware manipulation so that there is no need for translation of high-level languages. The benefit of this is faster program execution and is used for time sensitive tasks due to speed and memory optimization. According to Investopedia, assembly languages are used in high-frequency trading platforms in the finance industry due to the need for speed and accuracy of transactions in order to have an edge over their competitors. Another use for assembly language that I found was through device drivers. Device drivers allows the device to communicate properly with the operating system in place to function properly. Assembly was used to create device drivers as it can directly invoke instructions to the CPU for direct communication, but nowadays developers have moved on to using C or C++.

All modern computers today follow the Von Neumann Architecture. Learning the Von Neumann Architecture is important because by understanding it, you virtually understand how most modern computers work deep down. This is because x86 processors follow this architecture and its correlation can be seen in its instruction sets. In the Von Neumann Architecture, both data and instructions can be stored in memory and be accessed through memory addressing. An 8086 processor only has 4 registers namely: AX, BX, CX, DX which are 16-bit registers with their own uses. Trying to create a complex program using only these 4 registers can be a challenge. To fix this problem, memory addressing schemes are used. To store an address in memory, brackets are used to denote that you are trying to access an address in memory. For example, to access a value in the memory address 210H, you will use brackets [210H] which will return the 8-bit value stored in that address location. Note that in emu8086 instructions start at 100H, so normally we use addresses far away from 100H to store our “variables” in memory.

Assembly languages are a step away from machine language and it allows us to directly invoke instructions to the CPU using instruction sets defined by the CPU architecture. Assembly language code are executed sequentially. Sequential execution of code can be limiting on what a program can do. Like modern programming languages, there are control statements, looping and functions in assembly. Control statements can be represented using the CMP or the compare statement used to compare two values. After the compare statement, various JMP statements can be used depending on the result of the CMP statement to jump to a function. We primarily used JE which is jump if equal and JNE or jump if not equal. Iteration in assembly can be done using the LOOP command which uses the CX register as a counter for how many times to iterate through a function. And similar to modern programming languages, functions are used to group code that perform a specific functionality.

In addition to invoking instructions to the CPU, assembly languages also allow programmers to communicate with other computer hardware. To do this, the programmer must send an interrupt request command. Interrupts requests through hardware signals interrupt the CPU and let the hardware device function. After this is done, the CPU will continue its main process. x86 processors have command codes such as 01H which lets the user input a single character, 02H which lets the user output a single character, and 09H which allows the user to output strings terminated with the “$” symbol. These codes must be stored at the AH accumulator and are executed by calling the INT 21H command.

In summary, assembly lets you have deep control over the CPU and your computers resources. This gives the programmer the control to create efficient and optimized code. Programmers today might think that code optimization is not an issue since computers nowadays are much faster, but back then you had to utilize efficiently the limited resources you had. Knowing and appreciating assembly is a good way to look back on how far computers have come and what they can do today.

References:

https://www.investopedia.com/terms/a/assembly-language.asp

https://teachcomputerscience.com/device-drivers/