SicSim: A simulator of the Educational SIC/XE Computer for a System-Software Course

System Software is a relatively difficult course to teach due to its complexity and comprehensive explanations required. Since real-life circumstances (such as x86) are too complex, professors rely on stimulations to teach the concepts to students. Which brings us to SIC and SIC/XE – both are hypothetical computers specifically designed for teaching purposes. SIC stands for Simplified Instructional Computer and it comes in a standard model (SIC) and a more equipped version (SIC/XE). The only con about using a hypothetical computer in an educational environment is that it cannot run programs. To improvise, a program can be run on a dedicated simulator. In this case, the simulator is called “SicSim” and it simulates the SIC/XE computer. Since the Systems Software course is usually taken by junior/senior level computer science students, they should have a strong background in computing that sets them up for the topics discussed. Such topics include: run-time systems, process virtual machines, assembling, macro processing, static and dynamic linking, and loading. As part of the course, students are even asked to make their own simulator for SIC/XE!

When creating a SIC/XE simulator, a separate program is needed to assemble the source code. Therefore, a user of SicSim must execute the assembler manually before stimulating it. For example, the original simulator was created by Leland L. Beck. He implemented his SIC/XE simulator using Pascal. In order to get to the simulator, “sicasm” is needed to assemble the source code then “sicasm” can be used. SicSim is used to load and execute machine code and displays the state of the hidden virtual machine. SicSim displays information such as the registers, instruction, and values of the selected memory locations.

At first, I was unsure why SIC and SIC/XE were taught in a Systems Software course instead of x86. Reading this article and doing some research on x86, I understand why. Although x86 seems very complex, I believe that SIC/XE has provided a solid basis for me. After learning my first assembly language, picking up another one should not be too difficult. During my research of assembly languages, I came across the “Motorola 6800” assembly language from the 1970s and it looks very similar to SIC/XE; this makes me proud to say I understand it! My only concern with SIC/XE is that it is outdated and has very few resources which makes it harder for students to learn – besides that, I think it is a great foundation for assembly language.