Project Journal: Weeks 1-4 (10/22 - 11/12)

BLARG! Systems - Ryan Kispert

Weekly Goals:

Week 1:

- Create and set up a basic website.
- Research and select an Assembler.

Week 2:

- Improve website visuals.
- Configure selected assembler.

Week 3:

- Make a "Hello World" program in Assembly.
- Compile program in x64.

Week 4:

- Research basic functions in Assembly.
- Create basic functions in Assembly.

Research:

Research during this period has entirely been focused on assembly, although the current course of this project has shifted (as determined in the Reflection). Initial research began on the selection, installation, and configuration of an environment for an assembler. The assembler converts assembly code to proper binaries (machine code). From initial research, it was decided to use the x64 Architecture as x32/x86 architectures are continually being deprecated. A paper available on Intel's website introduced the concepts of assembly and registers[1], mentioning several assemblers such as the Netwide Assembler (NASM), YASM, the Flat Assembler (FASM), and the Microsoft Macro Assembler (MASM). I eliminated YASM and MASM from my choices, as YASM is just a rewrite of NASM and MASM lacks some features that NASM includes[3] in addition to poor cross-platform support. Between NASM and FASM, I selected NASM (much to the dismay of my colleagues), as I found more guides than for FASM, and online forums had described NASM as having a simpler structure[2]. As NASM requires an external linker, essentially a tool that combines the binary generated by NASM into an executable program, I found MinGW[5] and attempted to utilize that. After issues with the configuration and utilization of MinGW, I instead opted for the linker included with Visual Studio. Using one of the more recent guides I found[4], I created a

“Hello World” program and *eventually* compiled it. Running the program, even though I was originally ahead of schedule, took me 2 weeks due to issues with the aforementioned linkers. After ensuring I was using an x64 version of the MinGW linker, and then eventually switching to the Visual Studio linker, I found the problem was actually in the configuration of the command line I was using[6]. Once using a properly configured command line, which included the x64 C libraries, the program successfully linked to an executable file, allowing me to test it.

Accomplishments:

- Created and improved the website
- Created a development environment
- Selected an assembler (NASM)
- Installed and configured NASM in the dev environment
- Followed a “Hello World” NASM Tutorial and successfully tested the program
- Gained a basic understanding of how Assembly and Registers work

Reflection:

The progress completed has been significantly less than originally intended, primarily due to the previously mentioned linking issues. Although this has been frustrating, it has resulted in a better understanding of the assembler and linking process. Luckily, after consulting with my advisors, I’ve elected to shift my current plan for this project, and this delay will not significantly affect my timeline. Rather than intending to create the entire compiler in assembly, which is one of the most complicated methods of doing so, I’ve decided to write my compiler in C. While I’m not particularly experienced in C, it’s much closer to other languages I already know, and I won’t be learning something completely foreign just to get started. To begin with my compiler, I will compile my language to C, however, I do still intend to compile my language to Assembly. This allows me to remain on track for my timeline and will continue to learn Assembly alongside the development of the compiler.

Bibliography:

1. *Introduction to x64 assembly. (n.d.). https://www.intel.com/content/dam/develop/external/us/en/documents/introduction-to-x64-assembly-181178.pdf*
2. *Jorido. (n.d.). FASM and NASM -- differences. flat assembler - Index. https://board.flatassembler.net/topic.php?t=19688*

3. *NASM. (n.d.). https://www.nasm.us/index.php*
4. *GitHub Pages. (n.d.). https://sonictk.github.io/asm_tutorial/*
5. *WinLibs. (n.d.). https://winlibs.com/*
6. *Hemaolle. (1966, December 1). Unresolved external symbol PRINTF in Windows x64 assembly programming with NASM. Stack Overflow. https://stackoverflow.com/questions/64413414/unresolved-external-symbol-printf-in-windows-x64-assembly-programming-with-nasm*