Adam Deehring

Bazen Nega

**Suspicious Disassemblers - Disassembler Project**

**Exceptions Report**

While our disassembler is functional, it is not as efficient as it could be. We neglect using the stack as much as we probably should have. This is not inherently a problem but leaves a lot of room for improvement.

As we reflected on our disassembler, we realized that we neglected using Address Displacement often, having only used it a handful of times. In our implementation, we use a data register to hold the number of bytes that we will move along in the case of operation codes that are longer than a word. If we have a successful read from memory, we increment our current position in memory by adding that register to our pointer. Otherwise, we increment a word. This was initially not the case but was implemented when we were running into alignment issues. This could have greatly simplified if we used displacement, as we could use it as a wrapper that would grab the immediate or memory location that follow the current operation code.

In our many versions of our testing suite, we would run into issues when we read a MOVEM operation code that placed the registers into an absolute memory location. This would result in MOVEM being printed correctly to the output window (which we anticipated), but it would also be causing the operation code that followed to be skipped. We investigate that bug further and discover that it was probably caused by an errant addition to our memory location pointer. We had some trouble figuring out the source of this bug before the deadline, so we were never able to find it and fix it.

One small bug that we were never got around to fixing was the case of a trailing operand when being used separators, or more specifically a trailing Slash at the end of a MOVEM disassemble. Because we got MOVEM out of the way very early in the assignment, once we finished implementing it, we left it untouched, as to not disturb anything and accidentally break anything. We believe this is due to the very inefficient method in which we break down a MOVEM command. This seems to be related to our issue with how we programmed the entire program.

If we were to have been given a few more weeks to work on this, we most likely would have completely overhauled the entire program to solve our efficiency issue, which would lead to being able to overhaul how we break down a MOVEM command, thus getting rid of those to non-conformities.