CSC 317 Project 1 Report

Nathaniel Fagrey

October 2019

Introduction

This document outlines how to run the T34 Emulator. It also lists all the functions used and how they were tested.

Running the Program

To run the program:

$ python3 t34.py <objectFile>

If an object file is specified on the command line, it must contain a program in Intel Hex format. Also, the program will only work if python3 is used. Earlier versions of python will not work.

Functions

There is one main class used in this project phase: the Memory class. In this class there were 6 methods:

- parseUserData - parses the user data entered on the command line and determines what action needs to be done.
- storeData - stores the given data in the correct address in memory.
- getMultipleData - returns data given a start address and an end address.
- storePrograms - stores a program given in a file in a memory address.
- checkMemory - continually checks memory to make sure that the memory bounds are not exceeded or that the address given is out of bounds.
- getData - given an address, it returns a single value for that address. Right now not necessary, but it will be helpful in the future.

There were two other classes used. The RunProgram class is only called with the run program command is specified. As of now, it simply, initializes the registers to 0 and sets the program counter to the correct address. The Testing class was created to make sure the Memory class runs correctly.

Testing

The program and functions were tested in two main ways. One was just manually entering data and programs on the command line and printing out that data. This was a quick way to make sure that the store data, print data, and store program functions worked.
However, to test the code more rigorously, a Testing mode was implemented. This mode can be activated by specifying the file TESTING.txt on the command line as follows:

$ python3 t34.py TESTING.txt

TEXTING.txt is previously filled with random data that was then stored in programs "memory". That data was printed out and checked with the original data to make sure all data was in the correct place and was preserved. To fill TESTING.txt with random data run following command:

$ python3 fillTestingFile.py

Specific tests were also done (most of these were to check edge cases):

- Made sure the values stored were actually stored and remained in memory even after multiple access to memory.
- Made sure that my error checking method would prevent memory allocated over our specified amount or that our addresses could not go beyond 2^{16}.
- Made sure that multiple programs can be given in a file in Intel hex format and stored in memory.