Program 1 Write-up Jennifer Kulich

Jennifer.Kulich

October 11 2019

1 Introduction

This program is to create an emulator that can store and print memory from an array. The user can either store memory in the list, print a range of addresses, print one address, or 'run' th program. This is all done through the terminal.

1.1 How To Run

This should be run from the command line and requires IntelHex. " It is recommended to use the included Pipfile and Pipfile.lock with pipenv install (which can be installed with pip install pipenv). This will setup the virtualenv for you, which can be invoked with pipenv shell," (Matthew Krohn- because I couldn't word it well enough). Then, you can type 'python3 t34.py' in the command line for the program to run and input to be taken in. If you wish to run a file, you can add the file name to the command line after t34.py.

1.2 How the Program Works

It is assumed that there will be correct input, so there is no checking to make sure that the input is correct. However, there is checking to make sure that the memory being stored or accessed is not outside the bounds of the array. If that happens, a friendly message will be output to the screen, and the program will continue. If the user wishes to exit the program, they can type 'exit', ctrl-C, or ctrl-D.

1.2.1 Storing Memory

For storing memory, the user will enter the starting location (in Hex) where they want to start storing memory, a ':' and then what they are storing. In the code, it will separate the starting position in the memory and the items that are being input. I used the .split on the : and then the ' ' so a list could be returned. Then, the list of what to store is put into the memoryList.

1.2.2 Printing Memory

The user can either choose to print a range from the memory or just a single address. If they want a single address from memory, only that address and what is in the memoryList is printed. If they give a range, the program will print everything in that range from the memoryList. The format for this is to start by printing the current place in memory first, the next 8 bits, and then repeating on a new-line until everything is printed. If the starting position is not an 8-bit multiple, the output will be indented to indicate that.

1.2.3 Loading From a File

The program can also load storing memory from a file. This is done using the IntelHex library. This makes things super easy because it can take the file, separate the addresses and what goes into that place. This makes it a simple for-loop to store the memory from the file.

1.2.4 Running the Program

If the user inputs a memory address with an 'R' at the end, the program will print out the format and then print out the memory address. The memory address will also be stored in a variable that is called programCounter.

1.3 How I Tested

I tested this by using the input provided to us in the write-up along with the files given to us. I also put in my own input along the way and after to make sure that is was correct. Testing it was not terribly hard, and it was clear when the program was correct.