Table of Contents

Overview

complx-tools is a suite of tools for learning lc3 assembly. It includes both a gui and cli based simulator (named complx and comp respectively), an assembler (as2obj), a very simple program that runs lc3 assembly files and spits out whats printed to the console (lc3runner), and a framework for testing lc3 assembly code (lc3test). Complx-tools also be extended with plugins that add additional functionality to the LC3. The tools also come with a C++ interface to the LC3 (liblc3).

These tools are mainly used in CS2110 at Georgia Tech.

Setup

Manual steps to compile the program are as follows

From the root directory of where you have the source code...

1. Ensure you have a C++ compiler (sudo apt-get install build-essential)
2. Install dependency wxWidgets 3.0 (sudo apt-get install libwxgtk3.0-dev)
3. Build the program (make)
4. Install the program (sudo make install)
5. Run ldconfig (sudo ldconfig)

If you'd rather a script to setup the program for you see the following files

To use this make sure you are in the root directory of the source code...

1. Make sure the script is executable (chmod u+x *filename.sh*)
2. Run the script (sudo ./*filename.sh*)

install.sh - Performs the above operations

install_script_fedora_21.sh – Install script specifically for Fedora, thanks Thomas Coe

install_script_ubuntu_13_10.sh - Install script for Ubuntu 13.10 and below, thanks Abhijit Murthy

TODO Talk about Arch Linux here...

Compiling on Windows and Mac is not supported and untested.

General Usage

Assembly Files

An assembly file (with the .asm extension) is just a normal text document. You can create .asm files in any text editor of your choosing. I recommend gedit. As a sample assembly program to get you started copy and paste this into a file named helloworld.asm

```
.orig x3000
     LEA R0, HELLOWORLD
     PUTS
     HALT
     HELLOWORLD .stringz “Hello World”
.end
```

As2obj

This program just assembles files and produces an object file (.obj) and a symbol table file (.sym). I will explain the formats of these two files later. Invoking the assembler is easy (note the []'s indicate optional parameters)

as2obj *filename.asm* [-all_errors] [-disable_plugins] [outfile]

an explanation of the command line parameters

- -all_errors Report all errors encountered by the assembler if possible
- -disable_plugins Disable use of all lc3 plugins
- outfile output filename

Lc3runner

This program is a simple command line program that just runs your assembly program until it halts spitting out any output that is written to the console. Running this program is very simple, but it doesn't have many options you can't view the contents of any address unless your program prints it out.

lc3runner filename.asm [-zeroed] [-input=some_text_file] [-count=num]

an explanation of the parameters

- -zeroed if passed in will zero out all memory if not passed in then all memory is randomized.
- -input=some_text_file use this file as stdin any reads from keyboard will read from the file instead
- -count=num from forr num instructions the default is run until HALT is encountered

Comp

Explain this once its finalized

Complx

This program is the main program you will probably want to run. It is an lc-3 simulator, that allows you to play through instructions, undo instructions, and modify the state of the lc3 while a program is running. Also supports a plethora of debugging tools. To start the program you can either find it in the “Start Menu” (Should be under programming), or start it through the terminal,

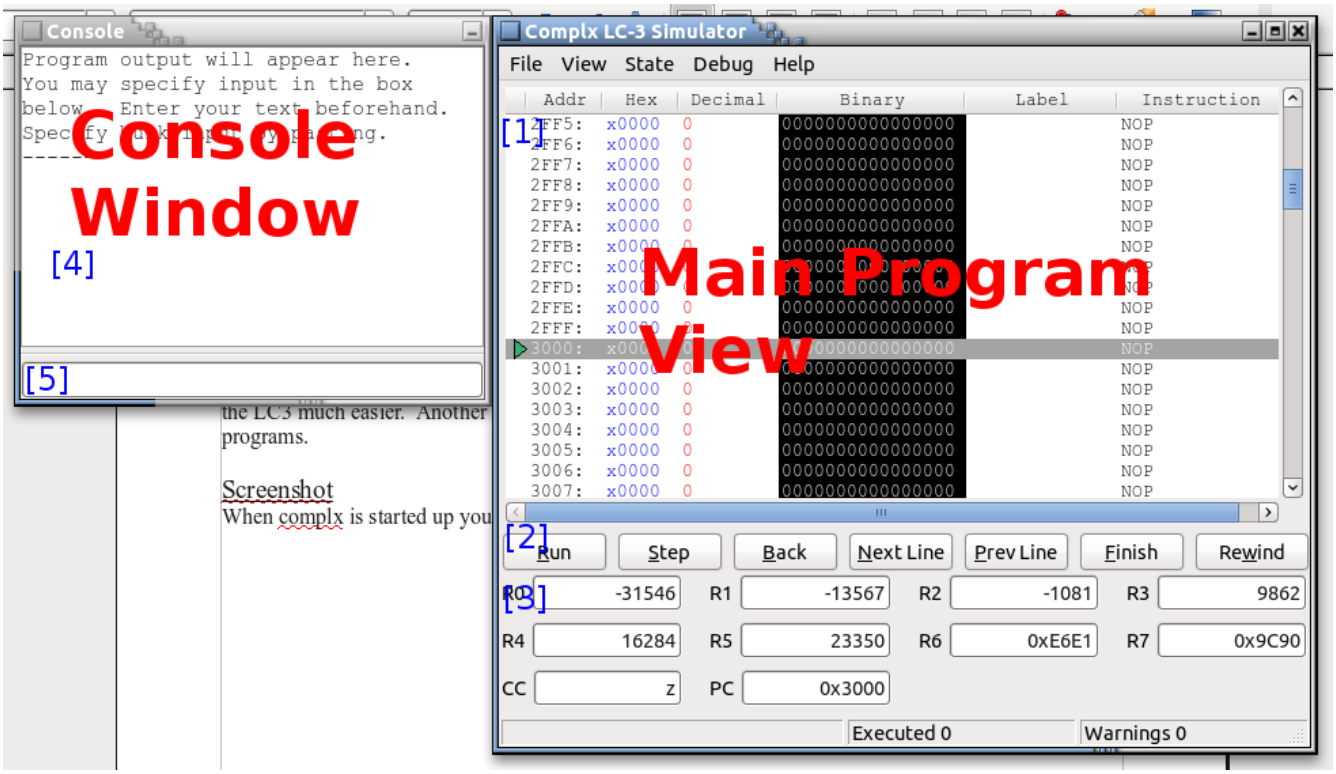
complx filename.asm

Command line parameters

The command line parameters as shown below are all optional.

- --unsigned=bool Sets if decimal representations are displayed in unsigned (default 0)
- --disassemble=num Sets the disassemble level 0: basic 1: normal 2: high level (default 1)
- --stack_size=num Sets the Undo stack size (default 65536 instructions)
- --call_stack_size=num Sets the Call stack size (default 10000 subroutine/trap calls)
- --address=hex Sets the PC's starting address (default 0x3000)
- --true_traps=num Enable true traps (see section on True Traps) (default 0)
- --interrupts_enable=num Enable interrupts (default 0)
- --highlight=num Enable instruction highlighting (default 1)

Interface tour



As a brief tour of the interface

[1] Is the main memory view. This shows the value each address contains in the LC-3 interpreted in hexadecimal, decimal, binary, and as an LC-3 Instruction. Remember that all instructions can be represented as a 16 bit value, So if I had the instruction ADD R0, R0, R0 then I should see the following: under the hex column I should see x1000, under the decimal column I should see 4096, in binary I should see 0001000000000000 and in instruction I should see ADD R0, R0, R0 (if the disassemble level is set to normal).

[2] The main control buttons. You will use these buttons to run your program (Run) or step through your program one instruction at a time (Step), you may also undo instructions using Back or rewind the entire program undoing everything in the undo stack (Rewind).

[3] The current state of all registers. You can enter values in binary, hexadecimal, or decimal. You can also change the base the registers contents is displayed in by either double clicking the text box or right clicking and selecting a new base for the register. By default R5-R7 are displayed in hexadecimal since these registers usually contain an address.

[4] Console window output will be displayed here. Also any warnings generated from your program will also be spit out here.

[5] Console window input will be typed in here. Its best to type your text before the program is ran.

Basic Workflow

Debugging

Credits