University of New Hampshire
Department of Electrical and Computer Engineering
ECE 562 – Computer Organization
Fall 2019
Lab #1

Introduction to Assembly Development and Tools

In this assignment, you will learn about assembly development, toolchains (assembler, linker, disassembler), emulators, and debuggers.

Step 1 – Getting the Tools

Install the following on your system: `make`, `qemu`, and the cross-compiler `aarch64-elf-gcc` toolchain. At least `aarch64-elf-as`, `aarch64-elf-ld`, `aarch64-elf-objdump`, and `aarch64-elf-gdb` should be available. A *Linux* environment is suggested, but *macos* with *Homebrew*, or *Windows* with *WSL* or *mingw32* would also work. See the course website discussions for guides and ask your questions through the “Discussions” section.

Step 2 – A Simple Assembly File

Download the `lab1.s` file and read it carefully. There is a missing line in it for you to complete.

Step 3 – Linker Script and Makefile

Download `link.ld` and `Makefile`, and read them carefully. Perform necessary changes, if any. Run `make` to create the executable `lab1`.

Step 4 – Disassembly

Run `aarch64-elf-objdump -D lab1` to see the assembly instructions reconstructed from the machine code. Submit the output as `step4.txt` through the course website.

Step 5 – Emulation

Run `make qemu` to emulate your program under `qemu`, which will wait for a debugger. Leave it open. As you step through instructions in the debugger in the next step, you will them appear on this terminal window.

Step 6 – Debugging

On a separate terminal window, run `aarch64-none-elf-gdb`. Submit the output as `step6.txt` through the course website.

- First, enter `target remote :1234`, then, enter `set scheduler-locking step`.
- Then, enter `display/i $pc`. This will display the next instruction every time the debugger halts the program.
- Then, enter `si` to execute a single instruction. You can enter `info registers` to see the contents of all registers, or `info register x0` to see the contents of X0
- Keep executing instructions one-by-one and observe the changes to the registers.
- Then, enter `si 50` to execute 50 instructions. Print the X0 register again.

Enter `q` to leave the debugger. You can then terminate the emulator on the other terminal window by pressing `Ctrl+C`.

Questions

Submit your answers to the following questions as “answers.txt” through the course website.

1. How many non-comment lines do you have in the assembly source code?
2. Do these lines match the disassembly output from Step 4?
3. How many bytes are used for each instruction?
4. Check the file size of the executable `lab1`. You can use `wc –c lab1` or `ls –l lab1`. How do you explain the size difference? Beside the few instructions, what else is there in the `lab1` file?
5. You can see the contents of the lab1 file, printed word-by-word using the following command: `hexdump –v –e '1/4 "%08x " "\n"' lab1`. Can you spot the actual instructions?
6. Create `lab1.bin` the following way: `aarch64–none–elf–objcopy –O binary lab1 lab1.bin`. Then look at the `hexdump` of `lab1.bin`. How about now? Change the parameter filename in the `Makefile` to run `qemu` with `lab1.bin` instead. Does it still work?