Using MARS (CSCI 260)

Subash Shankar

Since few of us have a MIPS CPU, we can not directly execute MIPS programs. However, there are a few MIPS simulators that allow us to simulate a MIPS program on most common platforms. The most common one is MARS, and directions for using MARS are below.

Installing MARS

MARS is installed on the department Linux lab at `/usr/local/bin/Mars4_5.jar`. You may also install it on your personal machine from `http://courses.missouristate.edu/kenvollmar/MARS/download.htm`. You may need to install a latest java version (`https://www.java.com/en/download/`), but try without it first.

Regardless of which MARS installation you use, you **must** ensure that any project submissions run on the lab before you turn them in. Be aware that there are some features of 4.5 not in earlier versions, and we *might* need them in the future.

Using MARS

Directions are given below for using MARS on the lab machines; you will need to figure it out for yourself on your own machine though I expect its comparable on any Linux machine.

1. Login to one of the cslab machines (cslab1 – cslab23), through eniac, if remote. If you are using ssh, you should do `ssh -X` to enable X forwarding since MARS has a GUI, but it might be slow to run remotely.
2. Use your favorite editor (emacs, vi/vim, etc.) to enter your MIPS program in a file whose name ends in `.asm` (more info in the next section).
3. Open a terminal and navigate to the directory containing your .asm file. Type:
   `java -jar /usr/local/bin/Mars4_5.jar`
   I personally like making the window smaller first since it will open in full screen.
4. Use the File menu to open your file and then select Run→Assemble. Then, use Run→Go to run your program. You may also use Run→Step to single-step through your program when debugging, or click on the appropriate Bkpt checkbox (left column of text segment) to set a breakpoint before running.

You should get familiar with the different panes in the MARS interface before using it.

Creating a Source File

Your source file will contain MIPS code as well as other information needed to assemble your program. Consider the row-major example at:

`https://courses.missouristate.edu/KenVollmar/MARS/tutorial.htm`

The file has two parts delimited with the `.data` and `.text` assembler directives. The data segment reserves

storage space in memory for program variables, and also allows for referring to these variables by name instead of address (*e.g.*, the `data` label, not to be confused with the `.data` directive). The text segment contains the actual MIPS program. The last two lines of the text segment (`li` and `syscall`) are used to return to the operating system.

To create a source file, you probably want to edit an existing working file, modifying only the relevant sections.