Lab 0: Getting Started on the Labs – Version 1.1 (9/7/14)

Throughout this course, you will use shared machines for working on the labs. These machines include `vlsifarm-03.mit.edu` through `vlsifarm-08.mit.edu`. You can log into these machines through ssh using your Athena username and password.

This document will instruct you how do some things required for lab such as obtaining initial code for each lab. Begin by using an ssh client to log into one of the servers named above.

1 Setting up the Tool-Chain

Execute the following commands to set up your environment and gain access to the tool-chain:

```
$ add 6.175
$ source /mit/6.175/setup.sh
```

The first command gives you access to the course locker `/mit/6.175` and only needs to be run once per computer. The second command configures your current environment to include tools required for the lab and needs to be run every time you log in to work on classwork.

2 Using Git to Get and Submit Lab Code

The reference designs are provided in Git repositories. You can clone them into your work directory using the following command (substitute `labN` with the lab number, such as lab1 and lab2):

```
$ git clone $GITROOT/labN.git
```

This command creates a `labN` directory in your current directory. The `$GITROOT` environment variable is unique to you, so this repository will be your personal repository. Inside that directory, the test benches can be run using the directions specified in the lab handouts.

Discussion questions should be answered in `discussion.txt` file supplied with the rest of the code.

If you want to add any new files in addition to what has been supplied by the TAs, you need to add the `newFile` in git using:

```
$ git add newFile
```

You can locally commit your code whenever you hit a milestone using

```
$ git commit -am "hit milestone"
```

And submit your code by adding any necessary files and then using

```
$ git commit -am "Finished Lab"
$ git push
```

You can submit multiple times before the deadline if necessary.

3 Writing BSV for the Labs

3.1 On Vlsifarm-0x

If you are not familiar with working in a Linux environment, this is a great opportunity to learn. In order to test the BSV code you have written, you need to use a Linux environment to run `bsc`, the BSV compiler. It makes sense to go ahead and write the BSV code on the same machine.

While there are many text editors you can use, there is only Bluespec-provided BSV syntax highlighting for Vim and Emacs. The Vim syntax highlighting files can be installed by running

```
$ /mit/6.175/vim_copy_BSV_syntax.sh
```

The Emacs syntax highlighting files can be found on the resources tab of the course website:

`http://csg.csail.mit.edu/6.175/resources.html`

Your TA does not use Emacs, so he has no idea how to install the files, or even if they work. If you want to write some instructions for your classmates or setup a script like the one for Vim, contact the TA and he will post it here.

3.2 On any Athena Machine

Your home directory on the Vlisfarm machines is the same as your home directory on any Athena machine. Therefore you can write code on an Athena machine using gedit or another graphical text editor and log into a Vlsifarm machine to run it.

3.3 On your own Machine

You can also use file transfer programs to move files between your Athena home directory and your own machine. MIT has help for securely transfering files between machines on the web at:

`http://ist.mit.edu/software/filetransfer`

4 Compiling BSV on Other Machines

BSV can be also be compiled on non-Vlsifarm machines. This may be useful when the Vlsifarm machines are busy near lab deadlines.

4.1 On any Athena Machine

The instructions used for Vlsifarm will also work for Linux based Athena machines. Just open a terminal and run the commands you would run on Vlsifarm machines.

4.2 On your own Linux Machine

To run the 6.175 labs on your own Linux machine, you will need the following items installed on your computer:

- `OpenAFS` to access the course locker
- `git` to access and submit the labs
- `libgmp.so.3` to run the BSV Compiler
- `python` to run build scripts
- support for executing 32-bit executables to run `smips-gcc`

A similar setup may be able to work for OS X. If you get it working, please provide details to the TA so it can be included here.

4.2.1 OpenAFS

Installing OpenAFS on your local machine will give you access to the directory `/afs/athena.mit.edu` that contains all of the course lockers. You will have to create your own `/mit` folder with symlinks within to point to the necessary course lockers.

CSAIL TIG has some information about how to install OpenAFS for Ubuntu on their website:

`http://tig.csail.mit.edu/wiki/TIG/OpenAFSOnUbuntuLinux`

These instructions are for accessing `/afs/csail.mit.edu`, but you need access to `/afs/athena.mit.edu` for the lab, so replace `csail` with `athena` wherever you see it. When you install OpenAFS on your machine, it gives you a `/afs` folder with many domains within. This website also contains the instructions for logging in with your username and password to gain access to the files that require authentication. You will need to do this every day you work on the lab, or every time you reset your computer, in order to access the 6.175 course locker.

Next you need to make a folder named `mit` in your root directory and populate it with a symlink to the course repository. On Ubuntu and similar distributions, the commands are:

```
$ cd /
$ sudo mkdir mit
$ cd mit
$ sudo ln -s /afs/athena.mit.edu/course/6/6.175 6.175
```

You can now access the course locker in the folder `/mit/6.175`.

4.2.2 Git

On Ubuntu and similar distributions, you can install `git` with

```
$ sudo apt-get install git
```

4.2.3 libgmp.so.3

The BSV Compiler uses `libgmp` for unbounded integers. To install it on Ubuntu and similar distributions, use the command

```
$ sudo apt-get install libgmp3-dev
```

If you have `libgmp` installed on your machine, but you do not have `libgmp.so.3`, you can create a symlink named `libgmp.so.3` that points to a different version.

4.2.4 Python

On Ubuntu and similar distributions, you can install `python` with

```
$ sudo apt-get install python
```

4.2.5 Support for executing 32-bit executables

The C compiler for the SMIPS architecture, `smips-gcc`, was originally built for 32-bit machines, so to use it on a modern 64-bit machine you need support for running 32-bit programs. Many Linux distributions have multiarch support to allow for 32-bit libraries and 64-bit libraries to coexist, but the exact procedure of doing so changes from distribution to distribution, so you should search the web for your particular distribution. Running `smips-gcc` is not necessary until much later in the course.

4.2.6 Setting up the toolchain

The original `setup.sh` script will not work on your machine, so instead you will have to use

```
$ source /mit/6.175/local_setup.sh
```

to set up the toolchain. Once you have done this, you should be able to use the tools as usual on your own machine.

Change Log

Version 1.1 (9/7/14)

Separated the "Doing the Labs" section into a writing BSV section and a compiling BSV section. Added a lot of information about how to run the labs on your own machine including a new setup script.