Systems II
Spring 2026
Tools

1 Introduction

The projects for this course require some tools that you likely already have and a few that you probably don't. To ensure that everyone has all the tools, and that everyone is using the same versions of those tools, this document provides instructions for installing and configuring the environment needed.

Keep in mind that this course will involve significant group work, and that you will be using a system simulator that is bespoke for this course. It will be important to use the same environment as everyone else, since differences in versions of the underlying tools may create subtle problems that will be difficult to diagnose. So please, don't try to wing it with your own installation or setup; **install the tools as described here**, even if you already have some of them. The tools you install for this class are easily separated from tools you already have, and can later be removed if you don't want this installation to interfere with work from other courses or projects.

Here are the tools that you will need:

1. **Micromamba:** A tool for creating and managing *Mamba environments*, within which we will install the software that we need.
2. **The Bourne-Again Shell** (`bash`) **:** The most common command-line interpreter. It is included with Linux and macOS systems, and there are instructions for installing it on Windows in Section 2.
3. **Java Development Kit (JDK):** This software package will provide both the *Java compiler* (`javac`) and the *Java virtual machine (JVM)* (`java`).
4. **The LLVM Compiler Infrastructure:** A set of tools and used by many compilers, providing us the `llc` command to generate RISC-V assembly.
5. **The Clang compiler:** A compiler for the C language, which is built upon LLVM.
6. **`sed`:** An old-school command-line ***s****tream* ***ed****itor*, used to edit files via shell script.

Once you have these tools installed, you will be able to use *Fivish*, our simulated RISC-V system, and its tools.

2 Special preparation for Windows

If you use Microsoft Windows, you must prepare your computer before installing the rest of the software described below in Section 3. Specifically, Windows does not normally have `bash`, a common shell available on other systems, and a number of standard command-line tools that go with it. Our projects for this course will require `bash` and some of those basic tools, so we must add them to Windows.

Git for Windows provides `bash` and these standard command-line tools, allowing Windows to behave like any other UNIX-like system. To get it, visit the Git for Windows site, and click the `Download` button to obtain the installer. After you install it, you can open the *Start* menu and type `git bash` to find and open the `bash` shell.

For the curious and adventurous: Another alternative, for those with Windows, is to install *Windows Subsystem for Linux (WSL)*. This tool makes it easy to install a Linux distribution within a *virtual machine*, allowing you to use Windows and Linux simultaneously. It is a good way to have available an additional constellation of software that is often useful for computer scientists. To install it, you can simply follow Microsoft's instructions, *How to install Linux on Windows with WSL*.

3 Installing *Micromamba*

For our projects, you are going to create a *Mamba environment* that will contain the software we are installing. You will activate this environment whenever you work on these projects; the software we install for these projects are isolated from the rest of your software and will only be used when you activate this environment.

To manage this environment, you will install and use *Micromamba*. Follow these steps, taken from the Micromamba installation guide:

1. **Open a terminal:** For macOS users, open the *Terminal* app; Windows users, open *Terminal* (for WSL) or *Git Bash*.[1]

[1] Linux users: You're on your own; figure it out.

2. **Install *Micromamba*:** Enter the following at the shell prompt:[2]

   ```
   $ "${SHELL}" <(curl -L micro.mamba.pm/install.sh)
   ```

   Issuing this command will download *Micromamba* and start a script to install it. Press `Enter` at any prompts to accept the default options.

3. **Re-initialize your shell:** In order to be sure that your current shell uses the newly installed *Micromamba*, enter the following command to restart your shell:[3]

   ```
   $ bash
   ```

4. **Update:** Make sure that your new installation is at the newest version:

   ```
   $ micromamba self-update
   ```

4 Install software packages

With *Micromamba* installed, you can create a *Mamba environment* for these projects and install the needed software. To do so, following these steps:

1. Create the Mamba environment:

   ```
   $ micromamba create -n sys2
   ```

2. Activate the environment:

   ```
   $ micromamba activate sys2
   ```

   After this step, your prompt should appear with a prefix showing the active environment's name.

3. Install the needed software (with no line-break; it's all one command):

   ```
   (sys2) $ micromamba install -y openjdk=25.0.1
            clang=19.1.6 llvm-tools=19.1.6
            python=3.14.2
   ```

[2]When I show commands at a shell prompt, the leading `$` symbol represents the prompt itself, which often ends with that character or the `%` symbol. **Do not type that part of the command**; the leading `$` simply represents that what follows is a shell command.

[3]You only have to do this once, at this moment. Henceforth, every shell that you start will automatically be ready to use *Micromamba*.

4. **For macOS users only:** macOS comes with *BSD sed*, which is a bit different from *GNU sed*, which is the variant of the sed utility that we rely on. So, macOS users need to install one more package that provides the *GNU sed* that we need:

   ```
   (sys2) $ micromamba install -y sed=4.9
   ```

5. Validate the install. Try the following commands, making sure that what you see looks something like what is shown here (with your username in place of USERNAME):

   ```
   (sys2) $ which clang
   /home/USERNAME/micromamba/envs/sys2/bin/clang
   ```

5 Using the environment

Having created the sys2 environment, you will need to activate it each time you start a shell in order to use the software that you installed. Activating it looks like this each time:

```
$ micromamba activate sys2
(sys2) $
```

As noted above, you will see the (sys2) prefix on the shell prompt whenever the environment is active. Then, when your work is done and you no longer want the environment to be active:

```
(sys2) $ micromamba deactivate
$
```