Simulator Setup

Table of Contents

[1. Installing Linux for Windows 10 (WSL) 2](#_Toc67904895)

[2. Download project files from GitHub 3](#_Toc67904896)

[3. Installing libraries for WSL 5](#_Toc67904897)

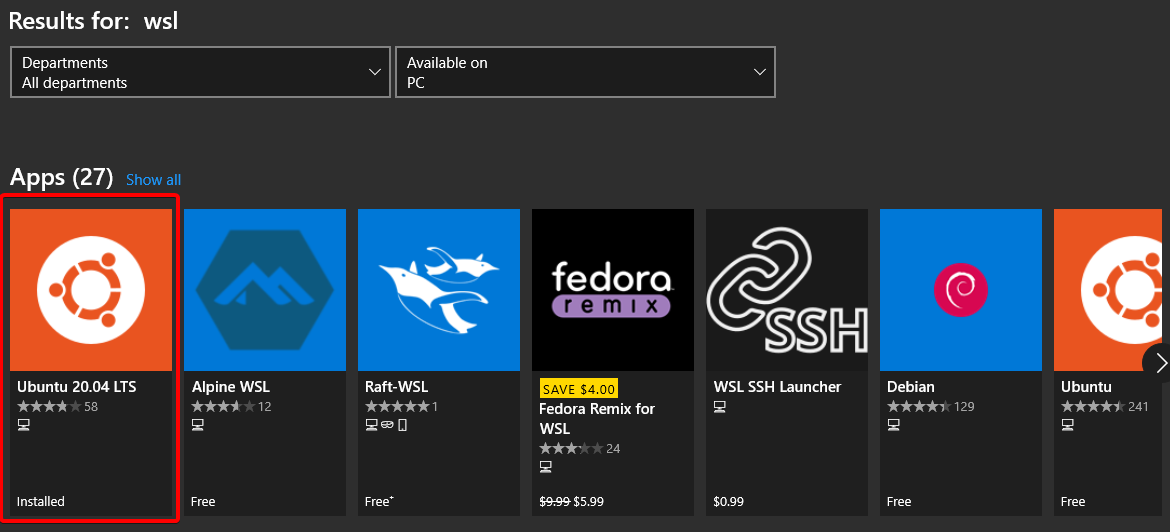
[4. Installing VcXsrv Windows X Server 6](#_Toc67904898)

[5. Configuring XLaunch 8](#_Toc67904899)

[6. Compiling the code with makefile and running the simulator 11](#_Toc67904900)

# Installing Linux for Windows 10 (WSL)[[1]](#footnote-1)

1. Follow steps 1 to 6 in this article: <https://docs.microsoft.com/en-us/windows/wsl/install-win10> (These steps may require multiple PC reboots)
2. For step 6, install the latest version of Ubuntu LTS from the Windows store. (currently 20.04)

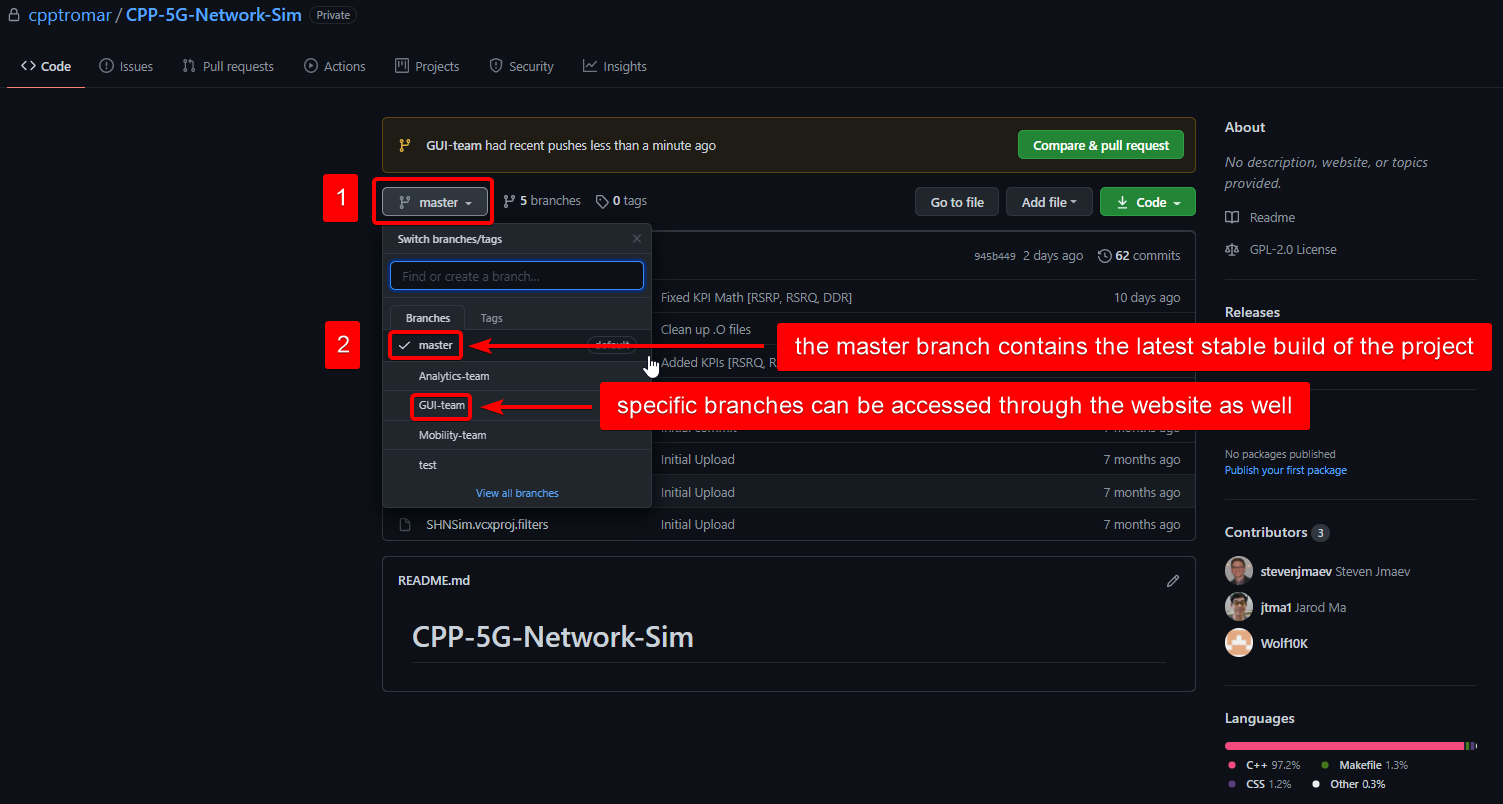


1. After successfully installing Ubuntu 20.04 LTS, open the program through the app store. It will prompt the user to create a password. This password will be used in a future step so be sure to write it down somewhere.

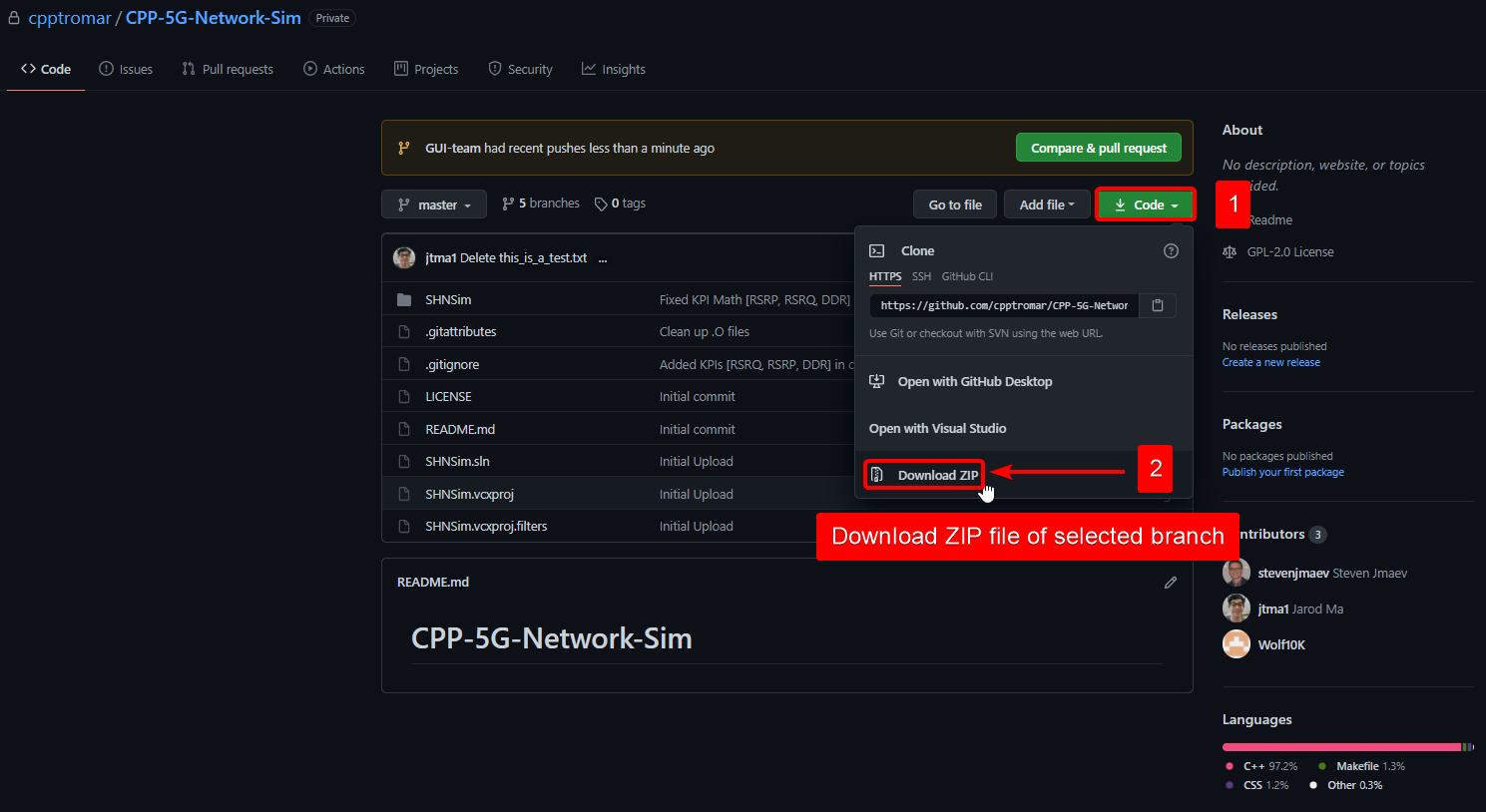
# Download project files from GitHub[[2]](#footnote-2)

<https://github.com/cpptromar/CPP-5G-Network-Sim>[[3]](#footnote-3)

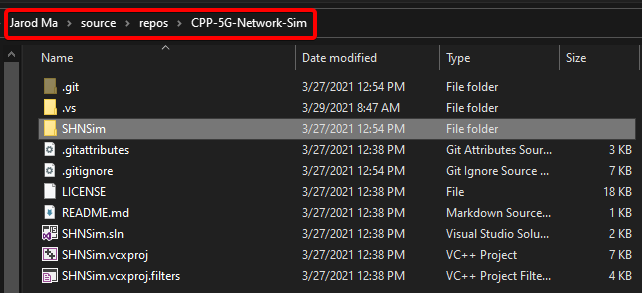
1. Choose version to download



1. Download ZIP file

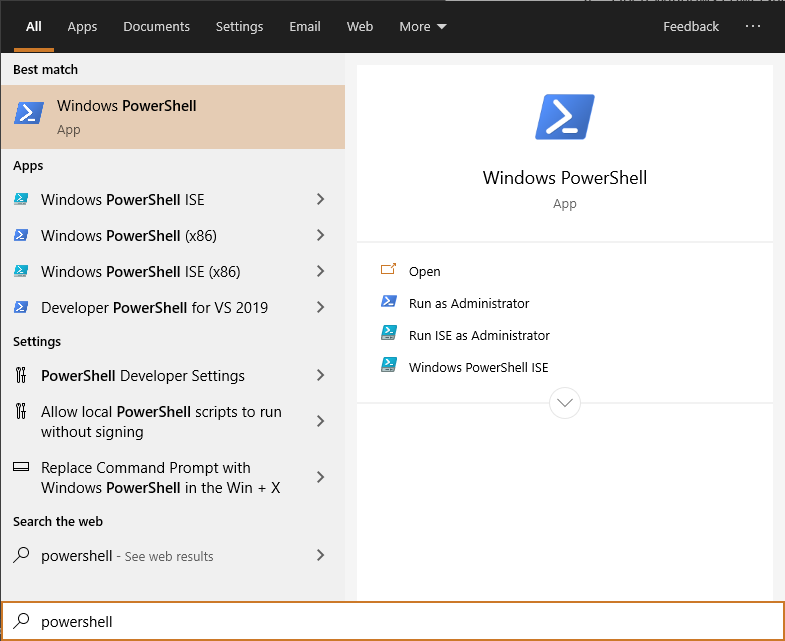


1. Extract ZIP file to a folder[[4]](#footnote-4)



# Installing libraries for WSL

1. Open Windows PowerShell[[5]](#footnote-5)



1. Change directory to wherever the SHNSim folder is located

|  |
| --- |
| cd [path\to\SHNSim\_folder] |

1. Run WSL

|  |
| --- |
| wsl |

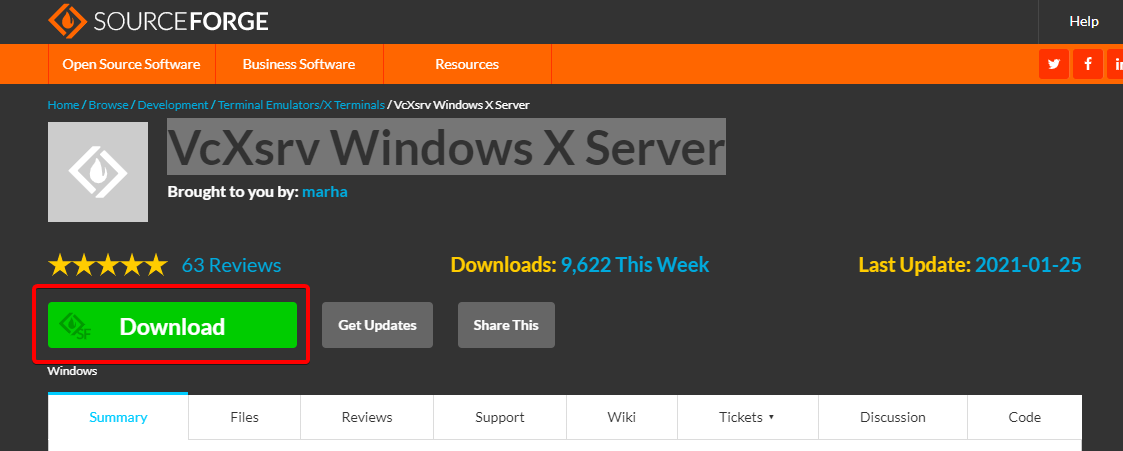
1. Install all required packages by running the following commands

|  |
| --- |
| **GTK & Dependency Libraries (for GUI)**  sudo apt-get update  sudo apt-get install libgtk-3-dev  sudo apt-get install libgtk-3-0  sudo apt-get install pkg-config  sudo apt-get install libcairo2-dev |
| **Makefile & G++ Libraries (for compiling)**  sudo apt install make  sudo apt install make-guile  sudo apt install g++ |

# Installing VcXsrv Windows X Server[[6]](#footnote-6)

1. Download VcXsrv from their official website and run their installer (may require a PC reboot)

<https://sourceforge.net/projects/vcxsrv/>

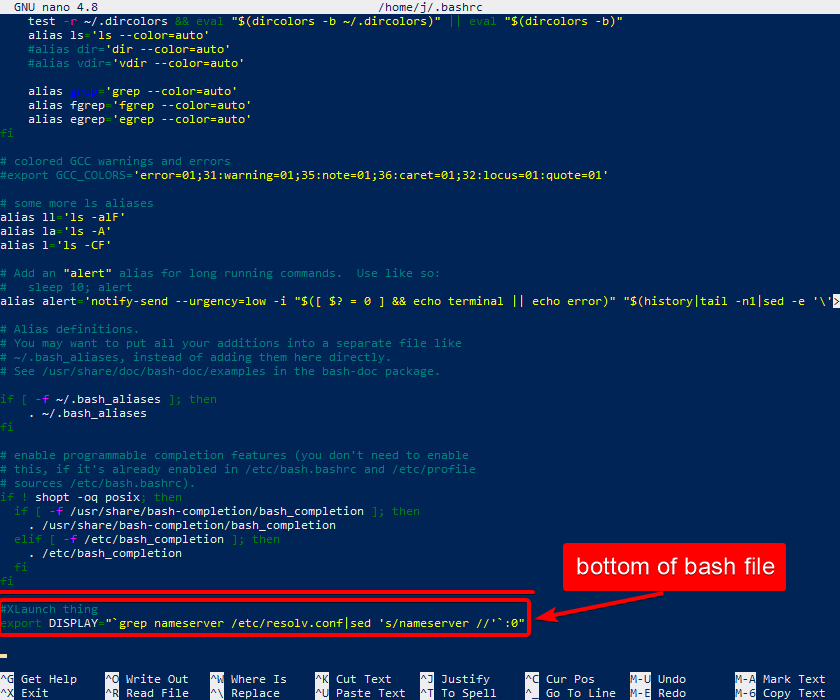


1. Run this command in PowerShell

|  |
| --- |
| sudo nano ~/.bashrc |

1. Enter your password that you set earlier when running WSL for the first time.
2. Use the down arrow key to navigate to the bottom of the bash file and add the following code:

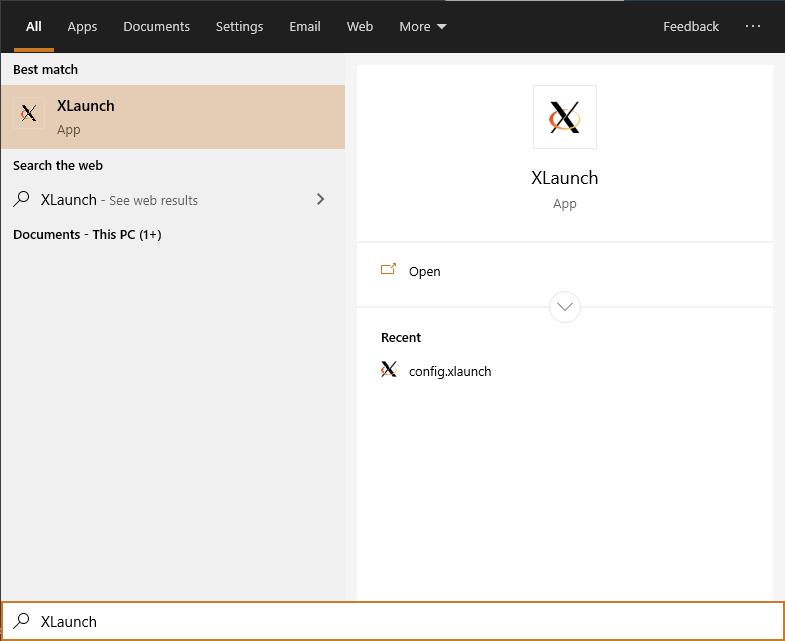
|  |
| --- |
| export DISPLAY="`grep nameserver /etc/resolv.conf|sed 's/nameserver //'`:0" |



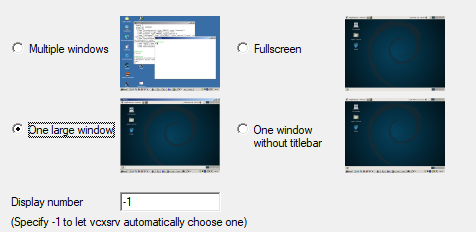
1. Save and close the bash file using these three key combinations in order: CTRL+O, Enter, CTRL+X
2. Close and re-open PowerShell

# Configuring XLaunch[[7]](#footnote-7)

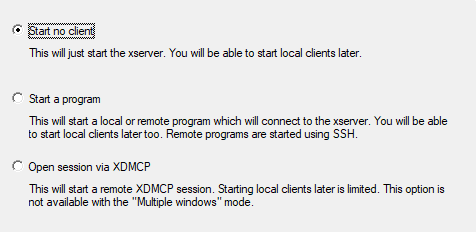
1. Open XLaunch through windows explorer



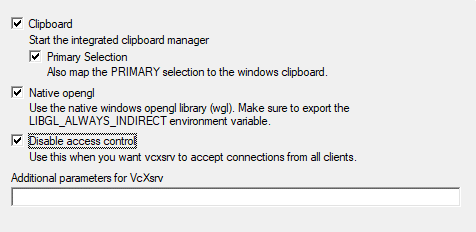
1. Select “One Large Window” and then press “Next”



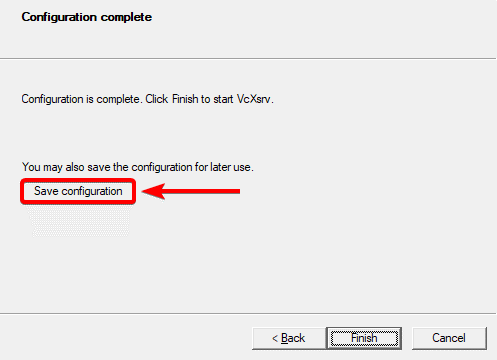
1. Select “Start no client” and then press “Next”

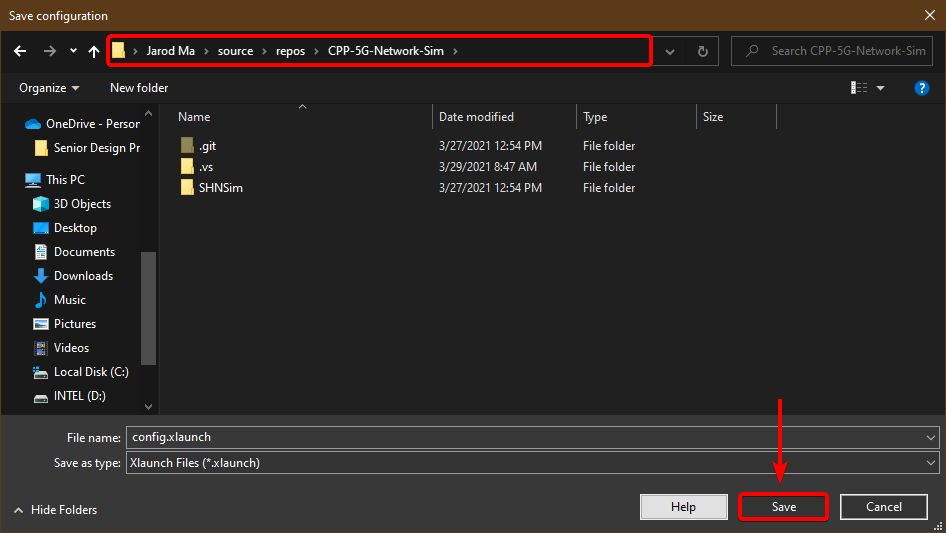


1. Check “Disable Access Control” and then press “Next”



1. Save configuration (.config file) to somewhere accessible. (e.g. Desktop or Project location)





# Compiling the code with makefile and running the simulator[[8]](#footnote-8)

1. Open XLaunch through the configuration file. This will open a blank window.
2. Change directory to wherever the SHNSim folder is located

|  |
| --- |
| cd [path\to\SHNSim\_folder] |

1. Run WSL

|  |
| --- |
| wsl |

1. Compile the code with makefile

|  |
| --- |
| make output  make clean |

1. Output to Display

|  |
| --- |
| ./output |

1. Linux machine or Mac, this step can be skipped. [↑](#footnote-ref-1)
2. A basic tutorial related to how GitHub works can be found here: <https://youtu.be/w3jLJU7DT5E> [↑](#footnote-ref-2)
3. This link requires permission to access the files and may change in the future depending on the owner of the repo. [↑](#footnote-ref-3)
4. We will be running all the Linux commands within the SHNSim folder. Be sure to extract the ZIP files to a folder instead of attempting to run the commands within the ZIP. [↑](#footnote-ref-4)
5. For Mac/Linux/Ubuntu, use the terminal application to run command lines. The “wsl” command does not need to be executed because it is used to start Windows Subsystem for Linux (WSL). [↑](#footnote-ref-5)
6. This application is used as a virtual display when running the “./output” command. Mac/Linux/Ubuntu users may require a different program to support displaying the simulator. Refer to the C++ GTK documentation for more information (<https://www.gtk.org/docs/language-bindings/cpp>). [↑](#footnote-ref-6)
7. The most efficient way to use XLaunch is to create a .config file that will be opened every time the simulator is opened. These following steps will show the process of generating a .config file using the XLaunch setup wizard. [↑](#footnote-ref-7)
8. These steps should be executed every time the simulator’s code is modified. To run the simulation without compiling with makefile: Opening Xserver, changing the directory, running WSL, and then using “./output” is sufficient. [↑](#footnote-ref-8)