Robot Firmware FRC 2024

Features

- Built-in Lua interpreter.
- Smart bot configuration with Lua bindings and negligible overhead.
- Fast code deploys when c++ doesn't need recompiled.
- Ultra low latency performance.
- Design is decoupled from wpilib making it portable to other robotics systems.

Clone

```
git clone --recursive git@github.com:snidercs/bot-2024.git
```

VScode Commands

There are shortcuts to command line tasks.

Press `shift + ctrl + p` to open a list of commands. Type `wpilib` to filter FRC specific actions.

Requirements

I recommend at least these two tools:

- Git
- VSCode
- Docker

Optional Windows tools

- wsl2 (Windows Subsystem for Linux)
- consider choco. It's the best package manager for windows.

Optional Mac Tools

- use brew. It's the best package manager for mac.

Building

If it's your first time, it takes a while. So get some lunch for the first time. On Mac M2:

```
BUILD SUCCESSFUL in 36m 10s
18 actionable tasks: 18 executed
```

Dependencies

The `gradle build` and `gradle deploy` tasks both need roboRIO libraries and headers in place. Most of them are handled by wpilib, but some need special attention.

Note: For *Windows* GitBash and the CMD prompt are both needed.

Using Docker

Docker is used to build and test the code in a consistent environment. It is required if building dependencies yourself on a non-Linux machine.

First build our image. Commands and scripts below all depend on it. If the `Dockerfile` changes, re-run this command.

```
docker build . -t snidercs/bot-2024
```

Without it you'll have to install java, gradle, a comiler, and other tools on your computer, and good luck keeping it all up to date. Make sure you give at least 4GB RAM and 4GB of swap space to docker. Gradle build is a memory hog.

LuaJIT

Linux

```
# multilib support is needed for cross building, install if needed
sudo apt-get install gcc-multilib g++-multilib
# Run the native/roborio build scripts
util/build-luajit-linux64.sh
util/build-luajit-roborio.sh
```

macOS

The Mac build script can produce `arm64` or `x86_64` binaries. It will select the system default if not specified.

```
# Choose one of these...
util/build-luajit-macos.sh # Use system default
util/build-luajit-macos.sh arm64 # force arm64 (M1/M2/..) build
util/build-luajit-macos.sh x86_64 # force an x86_64 (intel) build

# If docker is isntalled and running...
util/docker-run.sh util/build-luajit-roborio.sh
```

Windows

The build script can be run from a **GitBash** terminal:

```
# In a bash emulator
util/build-luajit-msvc.sh
```

The roboRIO binaries need docker to compile from Windows which *requires* a regular **CMD prompt**:

```
util\docker-run.bat util/build-luajit-roborio.sh
```

Firmware with WPIlib VSCode

Open a terminal and do:

```
./gradlew build
```

Firmware With Docker

```
./dockerbuild.sh
```

Testing

Run all unit tests.

```
./gradlew check
```

Deployment

Run the following command to deploy code to the roboRIO

```
./gradlew deploy
```

If it gives problems, cleaning the project could help. The `--info` option could give more information too.

```
./gradlew clean
./gradlew deploy --info
```

Other Useful Terminal Commands

Docker

```
./util/docker-run.sh  # Run a shell in the docker container
```

util/docker-run.sh is a script that runs a shell in the docker container. It's a good way to run commands in the docker container.

Git

Useful Commands to Understand

```
git fetch        # Fetch changes from github (needed to check out new ones)
git status -a    # Shows the status of all git branches
git checkout .   # Resets current changes on the branch
git checkout new_branch_name    # Switches branches to new_branch_name from a different
git pull         # Pulls new code onto the branch
```

Simulation

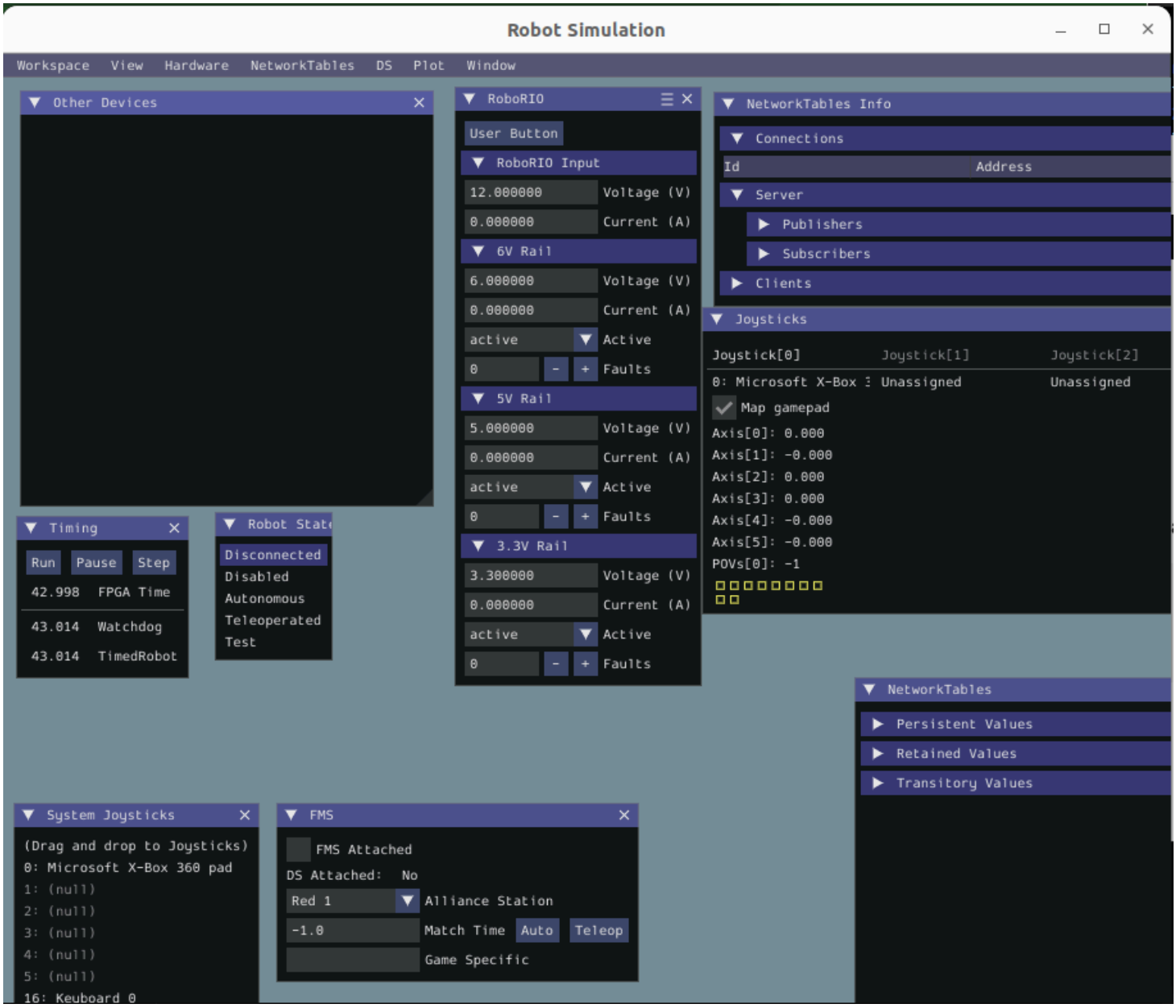
Bot simulation is available in a few ways...

IMPORTANT: The `Map gamepad` in the controllers list must be checked to mock the real bot.

Basic Simulator

This is the data-based simulator panel.

```
# run this from the command line to start the basic simulator
./gradlew simulateNative
```



Extern Sim With Real Driver Station

```
./gradlew simulateExternalNativeDebug
```

or

```
./gradlew simulateExternalNativeRelease
```

Feedforward and Feedback Control

Picking a Control Mechanism

Driver Station Tuning & Deployment

- Flashing the OS
- Deployment Info
- Driver Station Best Practices

Camera Example

When CameraServer opens a camera, it creates a webpage that you can use to view the camera stream and view the effects of various camera settings. To connect to the web interface, use a web browser to navigate to http://roboRIO-TEAM-frc.local:1181. There is no additional code needed other then Simple CameraServer Program. If running in the *simulator* navigate to http://localhost:1181

- Example Code

Example Code

FRC provides several examples of how to utilize WPIlib on GitHub. It is easier, and faster to copy and paste from these rather than generate new projects all the time.

- All cpp Examples

Differential Drive Help

- PWM Controllers
- Drive classes