

A Modern Open-Source Stack for Robotics Side Projects

How we used Open-Source tools for our robotics side project

Vincent Chapuis 沙文森
Arthur Amalvy 艾亞瑟

Introduction

We wanted to make a robot.

For the Micromouse Competition! (TMIRC)



Introduction

■ Problem

- What should we use for this robotics project?
 - To manage code and files
 - To create and build the project
- And that is open-source <3

▤ Disclaimer

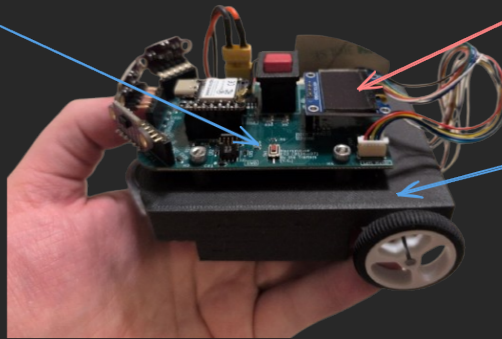
- This is a tour of everything you can do by yourself
 - There are other solutions that might be simpler to use!

Software

Electronics

Software

Mechanics



Infrastructure

■ First: Manage the code

▨ What we need

- Collaborative
- Version management
- Compatible with CI

Infrastructure

■ Gitea / Forgejo

- Git VCS
- Issue and PR Management
- Has its own CI runner
- Can be hosted on NixOs in a LXC container within Proxmox



Gitea

Infrastructure

IssuesPull RequestsMilestonesExplore

InsightLabs / macromousePrivate

Unwatch2Star0Fork0

<> CodeIssues8Pull Requests1ActionsPackagesProjectsReleasesWikiActivitySettings

32 Commits5 Branches0 Tags

mainGo to fileAdd File<> Code

vchapus07e9202c74	feat(firmware): Add telemetry support test and Working ES2 (#70)	yesterday
gitea/workflows	feat(firmware): Add telemetry support test and Working ES2 (#70)	yesterday
.vscode	feat(algo): add behaviour (#27)	6 months ago
crates	feat(firmware): Add telemetry support test and Working ES2 (#70)	yesterday
design	feat(firmware): Add telemetry support test and Working ES2 (#70)	yesterday
nix	tests(firmware): Add integration and unit tests (#44)	3 months ago
webots	Control (#69)	2 months ago
.editorconfig	feat(workspace): Initialize-Repository (#16)	last year
.env.template	feat(firmware): Add telemetry support test and Working ES2 (#70)	yesterday
.gitignore	Fix(git): Keep the webots/controllers/webots-tester directory in order to c...	6 months ago
.taplo.toml	feat(workspace): Initialize-Repository (#16)	last year
Cargo.lock	feat(firmware): Add telemetry support test and Working ES2 (#70)	yesterday

Search code...

Description

Micromouse project. Firmware, Electrical and Hardware design.

Manage Topics

Readme

100 MiB

Languages

Rust91%

Nix9%

Infrastructure

■ Second: Build and Test the code

▨ What we need

- Integrated with Gitea
- That's pretty much it

Infrastructure

■ Gitea-Runner (previously act-runner)

- Integrated in Gitea workflow
- Syntax compatible with github actions
- Hosted: NixOs/LXC/Proxmox



Gitea Actions

Infrastructure

✔

feat(firmware): Add telemetry support test and Working ES2 (#70)

ci.yaml: Commit 07e9202c74 pushed by vchapuis

main

Re-run all jobs

☰ Summary

All jobs

✔ build-check-test

3m1s

Run Details

 Workflow file

build-check-test

Success



> ✔ Set up Job

1s

> ✔ Checkout to the repository

2s

> ✔ Spell check with 'typos'

15s

> ✔ Execute the task 'format' from Makefile.toml

8s

> ✔ Execute the task 'taplo' from Makefile.toml

11s

> ✔ Create dummy .env file

0s

> ✔ Check for test and code coverage

29s

> ✔ Execute the task 'clippy' from Makefile.toml

22s

> ✔ Execute the task 'build' from Makefile.toml

27s

> ✔ Execute the task 'build-firmware' from Makefile.toml

42s

10 / 36

Infrastructure

■ Third: Reproducibility

How to make sure my code builds the same on:

- My machine
- His machine
- The CI

Infrastructure

■ Nix/Guix

- Declarative dependency management
- Reproducible builds



Infrastructure

Declaring and installing dependencies as simple as:

```
let
  # Download specific version of nixpkgs
  pkgs_dl_2605 = fetchTarball "
https://github.com/nixos/nixpkgs/tarball/nixos-26.05";
  pkgs_2605 = import (pkgs_dl_2605) {};

  # Put our pkgs into scope
in with pkgs_2605;
mkShell {
  nativeBuildInputs = [
    presenterm # Used for this presentation
    ghostty # Also used for this presentation
  ];
}
```

Programming Language

■ Rust

- Modern language
- Memory Safe
- Cost-free abstractions
- Integrated build and documentation tools
- Convenient Embedded Systems support



Embedded Framework

■ What we need

- Real-time capabilities
 - Multi-task with priorities
- HAL (Hardware Abstraction Layer)
- STM32 support
- Convenient

Embedded Framework

■ Embassy

- embassy-stm32 HAL
 - Type Safe Peripherals usage
 - Most convenient
- Mock time driver (great for simulation)
- Interrupt and threaded executors (multi-prio)
- Async



Embedded Framework

```
/// LED Blink Example
#![no_std]
#![no_main]

#[embassy_executor::main]
async fn main(_spawner: Spawner) {
    // One line peripherals and clock initialization
    let p = embassy_stm32::init(Default::default());

    // Compiler tracks pin ownership & peripherals configurations
    let mut led = Output::new(p.PC13, Level::High, Speed::Low);

    loop {
        // Toggle a LED (High/Low)
        led.set_toggle();

        // Non-blocking delay
        Timer::after_millis(500).await;
    }
}
```

Simulation

- Testing things in the real world is time consuming and hazardous.
- We needed something to simulate our system in a virtual environment.
- ROS is great (many good tools!) but...
 - Only works in a specific version of Ubuntu.
 - Can be quite complex.
 - Need to do everything in the ROS ecosystem.

Simulation

■ Webots

- An open source simulation engine
- Can simulate many things!
 - physics ✓
 - actuators (motor, speaker...) ✓
 - sensors (lidar, camera, distance sensors...) ✓
- Write your robot controller in many languages: C, C++, Python, Java, MATLAB...
 - Rust... with webots-rs.
- See also: Gazebo, Copper, MuJoCo...

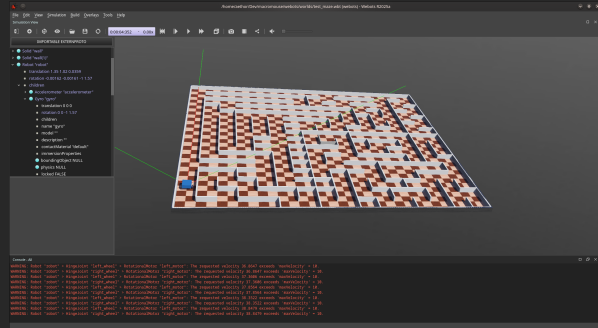
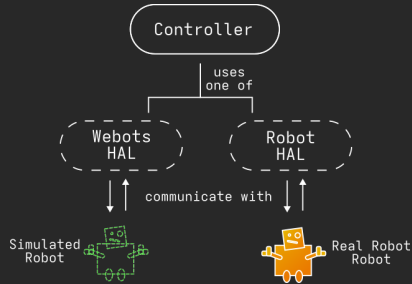


WebotsTM
robot simulation

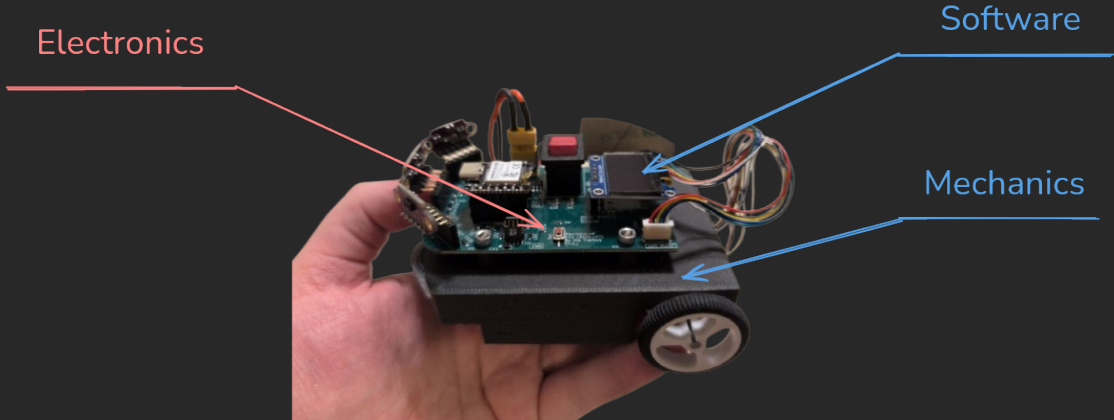
Simulation

Webots: In Practice

- You can reuse the same controller in the simulation and in your robot!



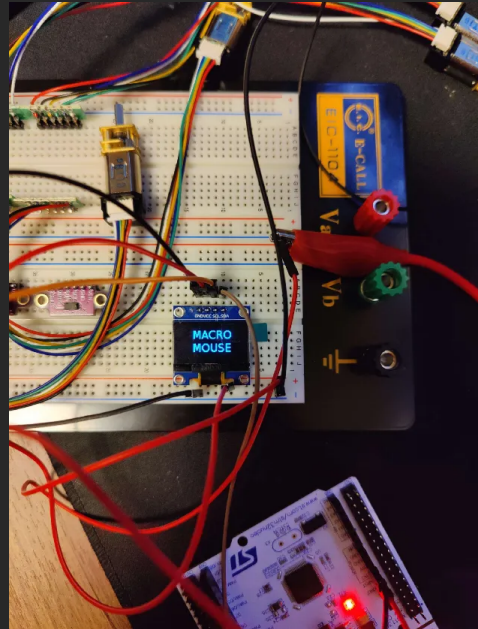
Electronics



PCB Design

After Breadboard/Devboard:

- PCB (Printed Circuit Board) Development
 - Smaller size factor
 - More reliable connections
 - Integrate chips that do not have dev board



PCB Design

■ KiCAD

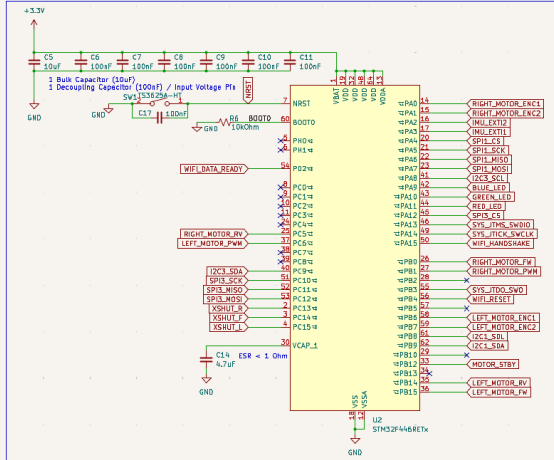
- Schematic Capture
- PCB Layout
- 3D Viewer
- Export to manufacture (JLCPCB) with 3rd party addons
 - Fabrication Toolkit



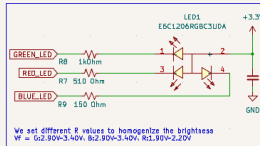
PCB Design: Schematics Capture

MICROCONTROLLER

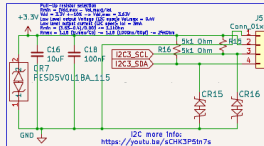
MCU - STM32F446RETE



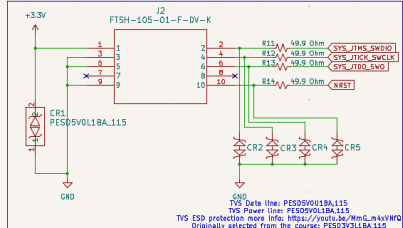
RGB LED



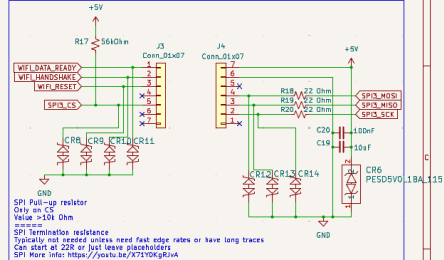
Display Header



SWD Header



WiFi Header



The Traitors

Sheet: Microcontroller/
Title: microcontroller_sch

Title: Macromouse

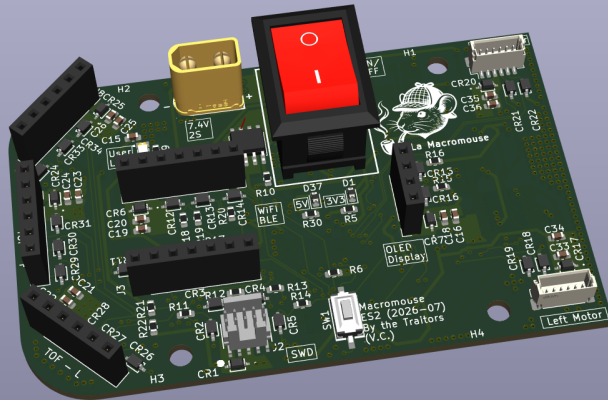
Size: A4 Date: 2026-05-25

KiCad E.D.A. 9.0.6

Rev: 0.1

Id: 3/6

PCB Design: 3D View



PCB Design: Manufacturing

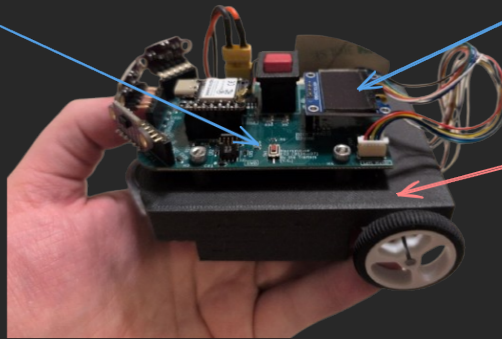


Mechanics

Electronics

Software

Mechanics



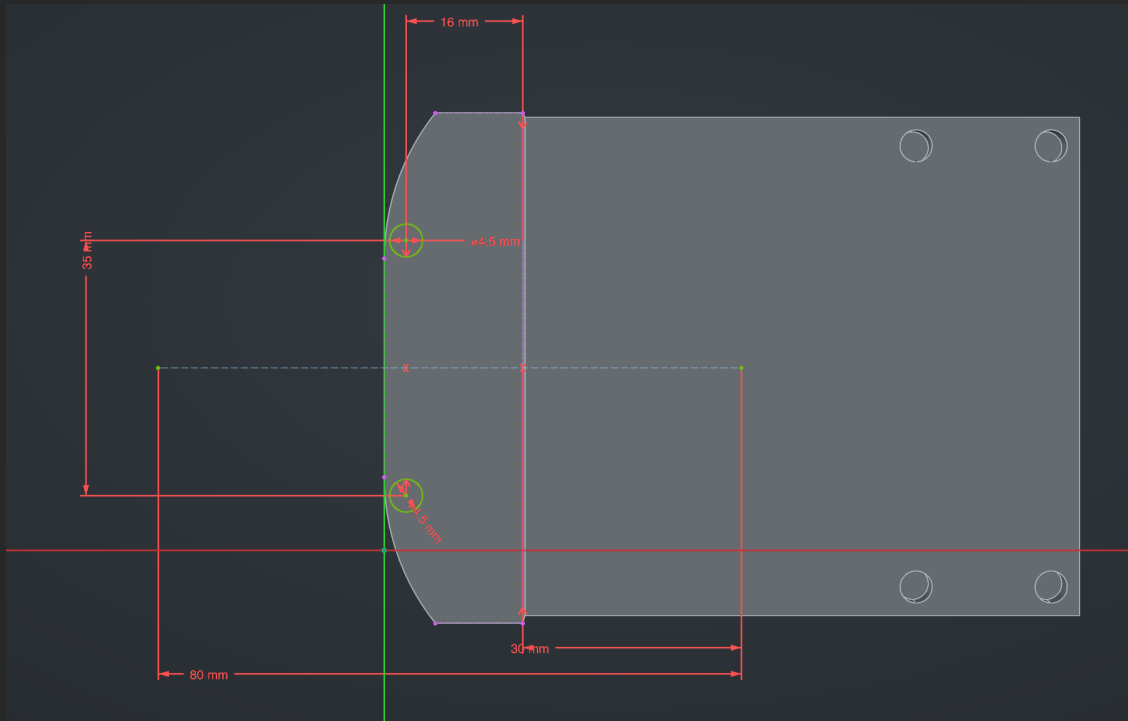
Mechanical Design

■ FreeCAD

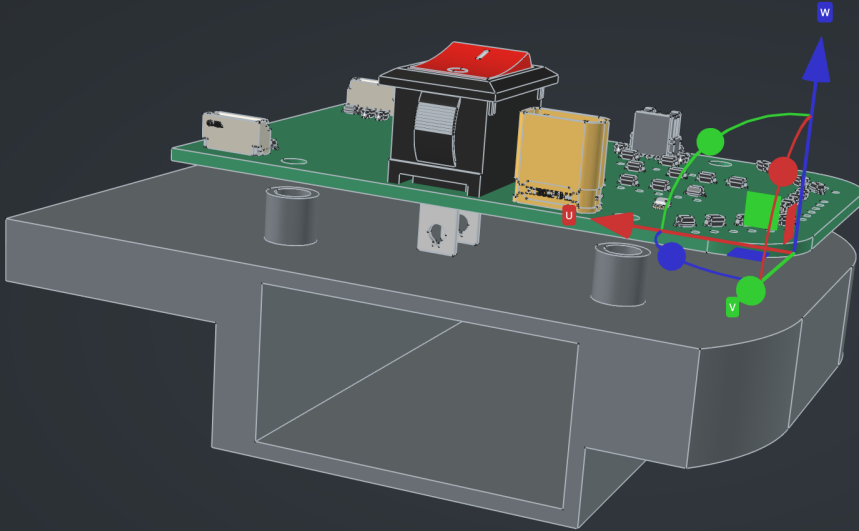
- 2D Schematics
- 3D Parametric



Mechanical Design



Mechanical Design



Mechanical Design



Mechanical Design



Conclusion

Don't be afraid to start your own project! If you didn't do it yet, start tomorrow!

And don't forget:

Have FUN!

5 Minutes Q&A

Ask your questions