Designing and producing a test platform for fluid monitoring

Afbeelding met elektronica, circuit

Automatisch gegenereerde beschrijving

Redesigning a fluid monitoring setup’s hardware and software

for increased functionality, reliability and durability.

****

**Internship documentation**

**Embedded Systems Engineering**

**Casper R. Tak**

# General Information

**Document**

Version: V1.0

Date: 20-01-2023

Word count: X

Page count main report: X

Page count attachments: X

**Student**

Name: Casper R. Tak

Studentnumber: 657313

Email: cr.tak@student.han.nl

Phone number: 0612292537

**Institution**

High School:Hogeschool van Arnhem en Nijmegen (HAN)

Faculty: Academic Engineering and Automotive (AEA)

Education: Embedded Systems Engineering (ESE)

Start and end of internship: 1 September 2022 till 27 January 2023

**Client**

Name: Jeroen Veen

Email: jeroen.veen@han.nl

**Coaches**

Name: Rudie van den Heuvel

Email: R.vandenheuvel@han.nl

Name: Jeroen Veen

Email: jeroen.veen@han.nl

Source logo HAN (Wikipedia)

All illustrations are self made unless specifically mentioned

# Document log

Table 1 Document history

|  |  |  |
| --- | --- | --- |
| Version | Date | Edits |
| 0.2 | 23-01-2023 | Many edits done. Added front page and incorporated most of Jeroen’s feedback on V0.1. |
| 0.1 | 09-01-2023 | Start of report laid down basic layout and style. |

# Summary

In this report, I present the hardware and software design and implementation of the Rastaban water quality monitoring system.

The system uses a Raspberry Pi running Raspbian as the main controller and computer, and a PCB for circuit layout. Control of the hardware, GUI, and data processing is done using Python. The design process involved selecting suitable hardware, designing the PCB, and programming the firmware in Python. The system was rigorously tested to ensure accurate and reliable measurements by testing both hardware and software components separately.

# Introduction

This document is written for my client, Jeroen Veen and potentially others (students) who want in insight in my journey to creating the hardware for the so called Rastaban project where fluids like water are being monitored. This document contains my design path from software and hardware for my Rastaban PCB.

I wrote this document for my embedded systems internship at the HCL. I have always had my interests in technology and this became apparent when I was little. I always moved trains and gears to see how the worked. I disassembled many, many devices, like Gameboys, Mixers, DVD players etc. I also broke many by doing so. Learning how to repair devices and what component does what was one of the things that I wanted to know when I was in middle school. After that finding an education that could teach me that and teach me how to create my own devices became my goal. I later found a small job as a care taker of elderly people and found so many subjects where machines could assist them in their lives. That’s how I found myself in my first project in the HCL and after 2 projects I still wanted more health related projects. That’s how I ended in the HCL for my internship.

While working on this project I have learned so much when it comes to general improvements in my design course. I used the V-model as a guideline for the first time and I found that it really helped me stay on track. I also found that I improved my scheduling and log skills. Apart from that I have developed myself by exploring different subjects like industrial product designing and even teaching and coaching ESE and IPD students.

I want to thank Jeroen Veen for all his hard work and patience when teaching me how to approach this project and teaching me about software and hardware. I know that Jeroen and I do not always think or communicate the same way and that sometimes resulted in misunderstanding, but I found this challenging and an excellent exercise. Thankfully, we also could fantasize and discuss very interesting subjects like morals on some technical aspects and I think we learned a thing or two from each other by doing so. I really enjoyed that. I think that Jeroen and I can be a real brainstorm machine sometimes and I hope that we can one day.

Rudie van den Heuvel was (on paper) my client, but we quickly figured that the WQM was a more suitable project for me and that was fine. Rudie helped me very much by talking about some more personal subjects like human-human relationships and how to communicate with people who are different. He sometimes thought me a lot by saying nothing at all. I learned from his positive mindset and wise stickers and quotes. When teaching the IPD students I saw how Rudie approached his students and this was inspiring to me. I really hope that Rudie and I will be working together in the future, maybe IPD related?

# Preface

## Personal Goals

I recently completed projects S3 and S4 at the Health Concept Lab (HCL) at the HAN and decided to do my internship at the same location. During my internship, I plan to further develop my planning skills and improve my PCB design skills using KiCad6. Additionally, I hope to improve my programming skills in C, C++, and Python. I am also interested in exploring and expanding my knowledge and understanding of various topics like industrial product design.

The fluid analysing device project will be a key part of my internship as it will allow me to achieve these learning goals. The project involves creating a smaller, improved version of a fluid analysing device that was developed by previous S6 and S4 students. The device is being developed for Jeroen Veen, who is the client for this project. I will later write a reflection where I will conclude if I achieved my goals.

## Personal Motivation

My interest in health and care has grown significantly since completing my S3 project. I have become interested in anything related to health care technology. I was fortunate enough to come into contact with Rudie van den Heuvel, who informed me that the Health Concept Lab (HCL) was still seeking students to assist with various projects, including a water monitoring system. Rudie mentioned the opportunity to grow my PCB and Python skills, which motivated me to join the HCL. I believe that the HCL and its teachers can provide me with the support and knowledge I am seeking to gain.

# Background information

## Water quality monitoring background

The ultimate goal of this system is to analyse and monitor water quality. As people age and take more medications, the remains of these medications can end up in urine and feces, leading to polluted water that can harm organisms. The Water Quality Monitor (WQM) will be a relatively cheap and useful device to use for testing the quality of water or other fluids. For now the focus is on water but the long-term goal is to create a portable device that can continuously monitor water quality and potentially even test for diseases.

## Stakeholders

**Jeroen Veen** is the project leader. He will be guiding me mostly through all the ESE related questions, regarding programming, but also regarding the overall process and development skills. Since Jeroen is the project leader and the client, I will be working very closely to him.

**Rudie van den Heuvel** is a industrial product design (IPD) teacher and knows a lot about materials, mechanics and designing in general. He may be able to provide me with knowledge on how to approach this project and whenever I want to do something with materials, he can assist.

**Health Concept Lab** is the lab that has been created around 3 years ago. The goal of this lab is to create a pool of knowledge on health on both mechanical as electrical ground. The lab is populated by students from S3 and above and there some teachers, who are well known in the subjects, as well.

**HAN** teachers can be consulted as well. Johan Brussen or Francesco Ursino for example are well known with power electronics. They can help me when it comes to supplying bigger loads with power. Other ESE or ELT teachers can also be of help. For EMC I can probably go to Ico van Diemen De Jel.

List of contents

[1 General Information 1](#_Toc125380331)

[2 Document log 2](#_Toc125380332)

[3 Summary 3](#_Toc125380333)

[4 Introduction 4](#_Toc125380334)

[5 Preface 4](#_Toc125380335)

[5.1 Personal Goals 4](#_Toc125380336)

[5.2 Personal Motivation 5](#_Toc125380337)

[6 Background information 5](#_Toc125380338)

[6.1 Water quality monitoring background 5](#_Toc125380339)

[6.2 Stakeholders 5](#_Toc125380340)

[7 Main and sub questions 12](#_Toc125380341)

[8 Objective and end product 13](#_Toc125380342)

[9 Functional Design 14](#_Toc125380343)

[9.1 State machine 14](#_Toc125380344)

[9.2 Use Case Diagram 15](#_Toc125380345)

[9.3 State diagram 15](#_Toc125380346)

[10 Technical design 18](#_Toc125380347)

[10.1 System architecture 18](#_Toc125380348)

[11 Hardware 20](#_Toc125380349)

[11.1 Microscope led 20](#_Toc125380350)

[11.1.1 Testing plan 21](#_Toc125380351)

[11.2 Stepper motor drivers 22](#_Toc125380352)

[11.2.1 UART on the TMC2209 23](#_Toc125380353)

[11.3 Thermal Control 24](#_Toc125380354)

[11.3.1 The modules 24](#_Toc125380355)

[11.3.2 Schematics 24](#_Toc125380356)

[11.4 Pi Hat i2c EEPROM interface 25](#_Toc125380357)

[11.5 12V Power GPIO 25](#_Toc125380358)

[11.6 PowerManagementSystem 26](#_Toc125380359)

[11.6.1 Power Supply 26](#_Toc125380360)

[11.6.2 Preventing Supply Noise 26](#_Toc125380361)

[11.6.3 EMC 27](#_Toc125380362)

[11.6.4 ESD protection 27](#_Toc125380363)

[11.6.5 Power regulators 28](#_Toc125380364)

[11.6.6 12V vs 24V 28](#_Toc125380365)

[11.6.7 Trace width 29](#_Toc125380366)

[11.6.8 2 VS 4 layer PCB 30](#_Toc125380367)

[11.6.9 Coil whine 30](#_Toc125380368)

[11.7 Backpower protection circuit 31](#_Toc125380369)

[11.8 PCB versions 31](#_Toc125380370)

[11.8.1 PCB V0.1 31](#_Toc125380371)

[11.8.2 PCB V0.2 35](#_Toc125380372)

[11.8.3 PCB V0.3 37](#_Toc125380373)

[11.9 Focusing the lens 39](#_Toc125380374)

[12 Software 40](#_Toc125380375)

[12.1 Components 40](#_Toc125380376)

[12.1.1 Microscope LED 40](#_Toc125380377)

[12.1.2 TMC2209 Stepper motor driver 40](#_Toc125380378)

[12.1.3 PowerGPIO, Cooling Fan, Heating resistor 40](#_Toc125380379)

[12.1.4 Voicecoil 40](#_Toc125380380)

[12.1.5 Pigpio 41](#_Toc125380381)

[12.1.6 Init function (PIGPIO daemon) 41](#_Toc125380382)

[12.2 Doxygen 41](#_Toc125380383)

[12.3 GUI 42](#_Toc125380384)

[12.3.1 GUI design 43](#_Toc125380385)

[12.3.2 Final design sketch 43](#_Toc125380386)

[13 Appendix A: ChatGPT 44](#_Toc125380387)

[13.1 ChatGPT usage in this report 44](#_Toc125380388)

[13.2 Review of using ChatGPT 45](#_Toc125380389)

[13.3 Notes 45](#_Toc125380390)

[14 Appendix B: KiCad 46](#_Toc125380391)

[14.1 What is KiCad 46](#_Toc125380392)

[Why KiCad? 46](#_Toc125380393)

[14.2 Plugins 46](#_Toc125380394)

[14.3 How to learn KICAD6 47](#_Toc125380395)

[The design approach 48](#_Toc125380396)

[15 Appendix C: Technical Considerations 49](#_Toc125380397)

[16 Appendix D: TMC2209 UART debugging 52](#_Toc125380398)

[16.1 UART Problem with multiple drivers 52](#_Toc125380399)

[16.1.1 First Conclusion/solution 53](#_Toc125380400)

[16.2 Further testing UART communication TMC2209 54](#_Toc125380401)

[16.2.1 Second conclusion and solution for the UART error 55](#_Toc125380402)

[16.2.2 Final fix and conclusions 56](#_Toc125380403)

[16.2.3 Motor stutter issue 56](#_Toc125380404)

List of tables

[Table 1 Document history 2](#_Toc125380405)

[Table 2 Functional specifications 17](#_Toc125380406)

[Table 3 Technical specifications 19](#_Toc125380407)

[Table 4 Estimated maximum power usage 26](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380408)

[Table 5 V0.2 notes 32](#_Toc125380409)

[Table 6 Current usage 34](#_Toc125380410)

[Table 7 V0.3 notes 35](#_Toc125380411)

[Table 8 V0.4 notes 37](#_Toc125380412)

[Table 9 Testing configurations 52](#_Toc125380413)

List of figures

[Figure 1 Input output diagram 14](#_Toc125380414)

[Figure 2 State machine 14](#_Toc125380415)

[Figure 3 Use Case Diagram 15](#_Toc125380416)

[Figure 4 State diagram 16](#_Toc125380417)

[Figure 5 System Architecture 18](#_Toc125380418)

[Figure 6 Typical application TPS61158 driver 20](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380419)

[Figure 7 pcb recommendations 20](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380420)

[Figure 8 finalized led driver circuit 20](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380421)

[Figure 9 Micro stepping settings table 22](#_Toc125380422)

[Figure 10 Thermal control components circuit 24](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380423)

[Figure 11 optional EEPROM circuit 25](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380424)

[Figure 12 PowerGPIO circuit 25](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380425)

[Figure 13 ESD protection using TVS diodes 27](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380426)

[Figure 14 Power Management System circuit (simplified) 28](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380427)

[Figure 15 Trace Width Calculator tool 29](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380428)

[Figure 16 5V power circuit 30](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380429)

[Figure 17 PCB V0.1 31](#_Toc125380430)

[Figure 18 ESD TVS diode option 33](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380431)

[Figure 19 PCB V0.2 35](#_Toc125380432)

[Figure 20 PCB V0.3 37](#_Toc125380433)

[Figure 21 PCB V0.1-V0.3 38](#_Toc125380434)

[Figure 22 Application of GUI in the system 42](#_Toc125380435)

[Figure 23 Windows sketch GUI 42](#_Toc125380436)

[Figure 24 GUI widget options in one window 43](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380437)

[Figure 25 Final design sketch GUI 43](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380438)

[Figure 26 KiCad Design workflow (Kicad like a pro 3th) 48](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380439)

[Figure 27 UART control (Trinamic TMC2209 datasheet) 52](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380440)

[Figure 28 error output 52](#_Toc125380441)

[Figure 29 Addressing multiple slave (Trinamic TMC2209 driver datasheet) 53](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380442)

[Figure 30 Pin configuration 54](#_Toc125380443)

[Figure 31 successful transmission 54](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380444)

[Figure 32 Deviating TX and RX 54](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380445)

[Figure 33 RX and TX matching occasionally 54](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380446)

[Figure 34 successful transmission 55](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380447)

[Figure 35 pin configuration 2 55](#_Toc125380448)

[Figure 36 receive and transmit messages 55](https://hannl-my.sharepoint.com/personal/jeroen_veen_han_nl/Documents/Research/Effluent/07%20Studentprojects/Casper%20Tak/Rastaban%20Project/Documentation/Main%20report%20Internship%20Casper%20Tak.docx#_Toc125380449)

[Figure 37 stepper motor schematic 55](#_Toc125380450)

[Figure 38 successful initialisation 56](#_Toc125380451)

Glossary

**Rastaban** name of the current prototype (PI HAT)

**HAT** An attachment for the Raspberry Pi 40 pin connector header. Usually consists of specific hardware for a specific task like a 3d printing HAT.

Acronyms

**WQM** Water quality monitor

**PCB** Printed Circuit board, a thin board that provides footprints and wires for electrical components to be connected to.

**RPI** Raspberry Pi

**IPD** Industrial product design(er)

**ESE** Embedded Systems Engineer(ing)

**PMS** Power management system

**ESD** Electro static discharge

**IC** Integrated circuit

# Main and sub questions

In this paragraph I will list everything I need to know to get this project started. I need a basic understanding of the project, it’s size and the required approach to work as efficient as possible. By knowing what is expected, what is not, and how I think I can approach problems, it should be possible for me to create an approach plan.

1. Main: What is the goal of this product?
2. Main: What will be my contribution to this project?
   1. Sub: What do I think is achievable in this timespan?
3. Main: What is water quality?
   1. Sub: how can we measure water quality?
4. Main: What is the current process on this project
   1. Sub: what did previous groups contribute?
   2. Sub: what can be used or build upon in my project?

Answer:

1. The goal is to create a prototype within 5 months that is able to monitor water quality.
2. I will be designing and creating this prototype’s hardware and software.
   1. I think designing a PCB in combination with software that controls it should be possible.
3. Water quality is measured by several factors, such as the concentration of dissolved oxygen, bacteria levels, the amount of salt (or salinity), or the amount of material suspended in the water (turbidity). In some bodies of water, the concentration of microscopic algae and quantities of pesticides, herbicides, heavy metals, and other contaminants may also be measured to determine water quality. (Sanctuary, 2023)
   1. Water quality parameters include chemical, physical, and biological properties and can be tested or monitored based on the desired water parameters of concern. Parameters that are frequently sampled or monitored for water quality include temperature, dissolved oxygen, pH, conductivity, ORP, and turbidity. However water monitoring may also include measuring total algae, ISEs (ammonia, nitrate, chloride), or laboratory parameters such as BOD, titration, or TOC. (YSI, 2023)
4. The previous groups have presented prototypes and Jeroen Veen has created some software himself to control some aspects of the WQM.
   1. The previous groups S4 and the S6, have contributed by designing PCBs and software for the WQM. S4 has focused on making the control board that can be attached to the raspberry pi. The raspberry pi would use a camera attachment and a microscope to observe water quality. The S4 setup was relatively small, but also limited in its functionality. The S6 group has created a fully functioning setup with heaters, pumps for (precision) fluid movement and a spectrometer. They focused on this sensor instead of a camera attachment for the pi. Sadly the setup is rather large, prone to getting detached cables, since not everything is soldered and its control interface is not intuitive. The problem the client is now facing in short: the current prototypes lack in functionality, are prone failing hardware problems or they are not easily controlled.
   2. I think I can build upon the work of both groups. I should consider some of the choices they made. Connector types, software libraries, components etc. Maybe I find that some of their design choices are a good starting point for my project or I find that it can better be archived and I will create my own design. I do know that I would like to work on the idea of a HAT (attachable PCB for the pi)

# Objective and end product

The goal is to design and develop a stable and sturdy prototype device, called Rastaban, for monitoring water quality. The device will be equipped with a printed circuit board (PCB), we will call this the HAT. Further there will be software created for the Raspberry Pi and this PCB, which will control the whole system and retrieve data collected by it.

Rastaban is a project that aims to improve water quality monitoring in various locations. The device will be used to monitor water quality by taking optical images and possibly video’s of the chemical reaction that emerges when a water sample is mixed with the reaction fluid in the reaction chamber.

The device will be used by environmental scientists, engineers, students from the Wageningen University and other professionals who are interested in monitoring and improving water quality. The device should be able to operate in a range of aquatic environments, including freshwater and saltwater. It should also be durable enough to withstand harsh weather conditions and rough handling.

The project will be divided into several phases, including research and development, prototyping, testing, and deployment. We aim to complete the first compact Rastaban prototype within the next 5 months. The first version of the Rastaban will be designed and developed specifically for measuring water quality. However, we will also keep DNA research capabilities in mind during the design process, and incorporate any necessary components or features suggested by Jeroen.

My contribution to this project will be the design and creation of the HAT and it’s software to control it. After this period I deliver the hardware, software and documents. This way the device will be usable, expendable and serviceable by anyone with basic knowledge on the subject.

# Functional Design

Before creating anything, as the V-model shows, it is important to gather specifications. I started by creating a list of functional specifications: what should the device be able to do? Should it be able to move fluids? Should it be able to heat up the fluids? Etc. These specifications will be the guide line to follow while making decisions. The goal is to create a sturdy device that will be used for monitoring water quality. The basic input/output of the WQM will be as follows:



Figure 1 Input output diagram

## State machine

The device will mostly be working in this continuous simplified manner.

1. Prepare system for testing (flush system clean, reach required fluid temperature etc.).
2. When the fluids are ready, pump them to the reaction chamber for measuring.
3. Capture optical images/video’s.
4. Store the captured data on a storage medium or cloud based storage.

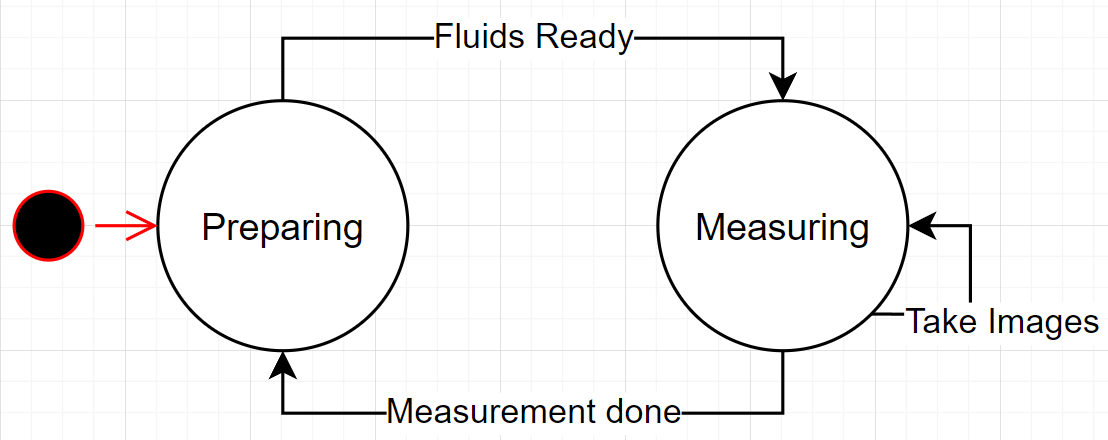


Figure 2 State machine

## Use Case Diagram

In the use case diagram we show the normal usage of the system in a static way. Here it is explained how the system functions in a block and how to Actor (user) provides input to the WQM and how the WQM may com in contact with a server solution.

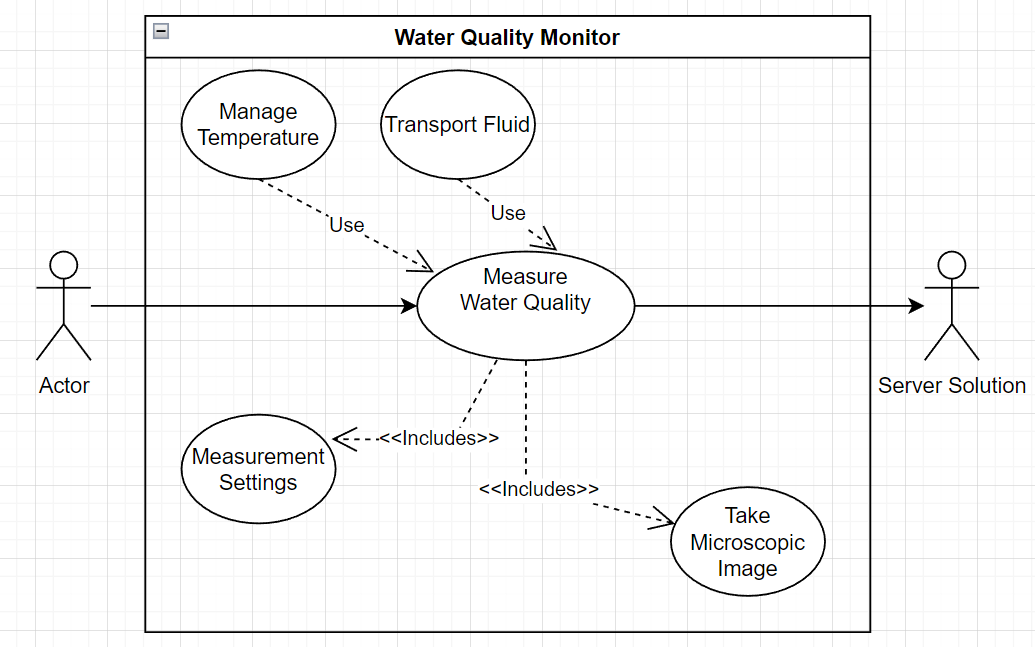


Figure 3 Use Case Diagram

## State diagram

To make this device usable for now and in the foreseen future, there will be functional and technical requirements that reflect this. For example, the device may later be used as an DNA monitoring device, and that requires heating and cooling elements in the reaction chamber to function.

Therefore some specifications will be chosen for future use accordingly. But to know how a WQM would work and what specifications are required, we need to know its usual operation style. Therefore I created a simple state diagram.

This diagram gives the overall view of how the state machine will work if Rastaban would be used in the field, functioning automatically or if it would be controlled manually.

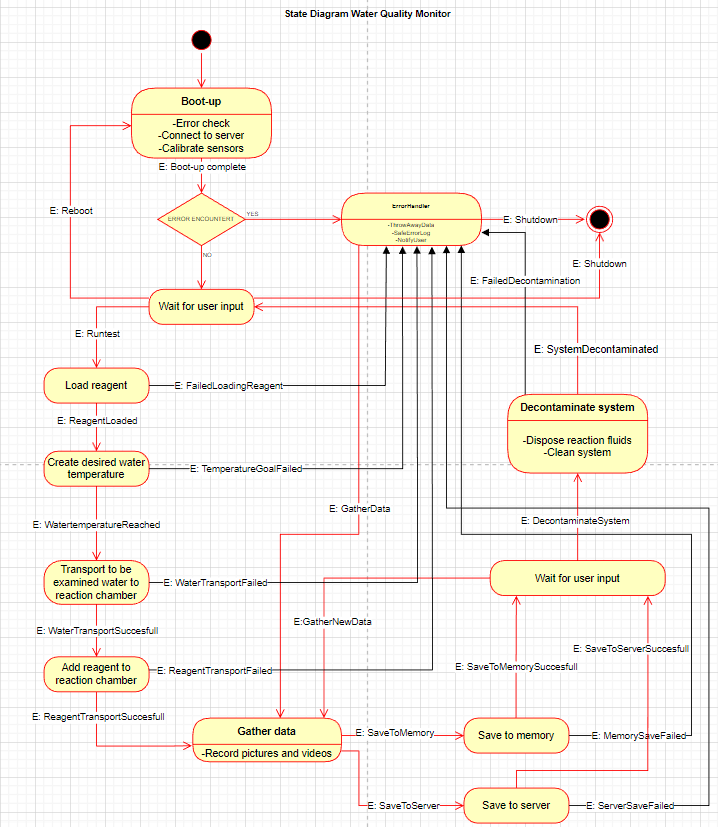


Figure 4 State diagram

After having created the diagrams indicating the expected functionality required by the WQM, I created a list with functional specifications.

Table 2 Functional specifications

|  |  |  |
| --- | --- | --- |
| **Functional specifications** | | |
| **#** | **MoS CoW** | **Description** |
| F1 | **M** | **The device is collecting data with the biosensor setup** |
| F1.1 | **M** | The device can detect particles |
| F1.2 | **M** | The device has a light source that provides sufficient light for the camera |
| F2 | **M** | **The device can control the temperature of the examination chamber** |
| F2.1 | **M** | The device can heat the examination chamber |
| F2.2 | **M** | The device can cool the examination chamber |
| F2.3 | **M** | The device can measure the temperature of the examination chamber |
| F3 | **M** | **The device can manipulate fluid movement** |
| F3.1.0 | **M** | The device can add fluids and remove fluids from the examination chamber |
| F3.1.1 | **S** | The fluid should be able to be moved forward and backwards. |
| F4 | **M** | **The device needs to be able to connect to external sensors** |
| F5 | **M** | **The device must be portable** |
| F6 | **W** | **The device will not be battery powered** |
| F7 | **C** | **The device could have It’s own identification** |
| F8 | **M** | **The device must be sturdy** |
| F8.1 | **M** | The device must be able to be moved without connections between electrical components being lost (lose cables) |
| F8.2 | **S** | The device should be protected against ESD to some extend |
| F8.3 | **S** | The device should have some EMC precautions |

# Technical design

After finishing the functional specifications, it is now possible to create a technical design upon our needs. The technical design is still rather inconclusive. I have to read datasheets to conclude what components are suitable for our prototype. My technical considerations can be found in appendix C.

A technical design shows the technical requirements and specifications that must be met in order to ensure that the system performs as expected.

## System architecture

The system architecture gives a static view of how to system components are connected to each other.

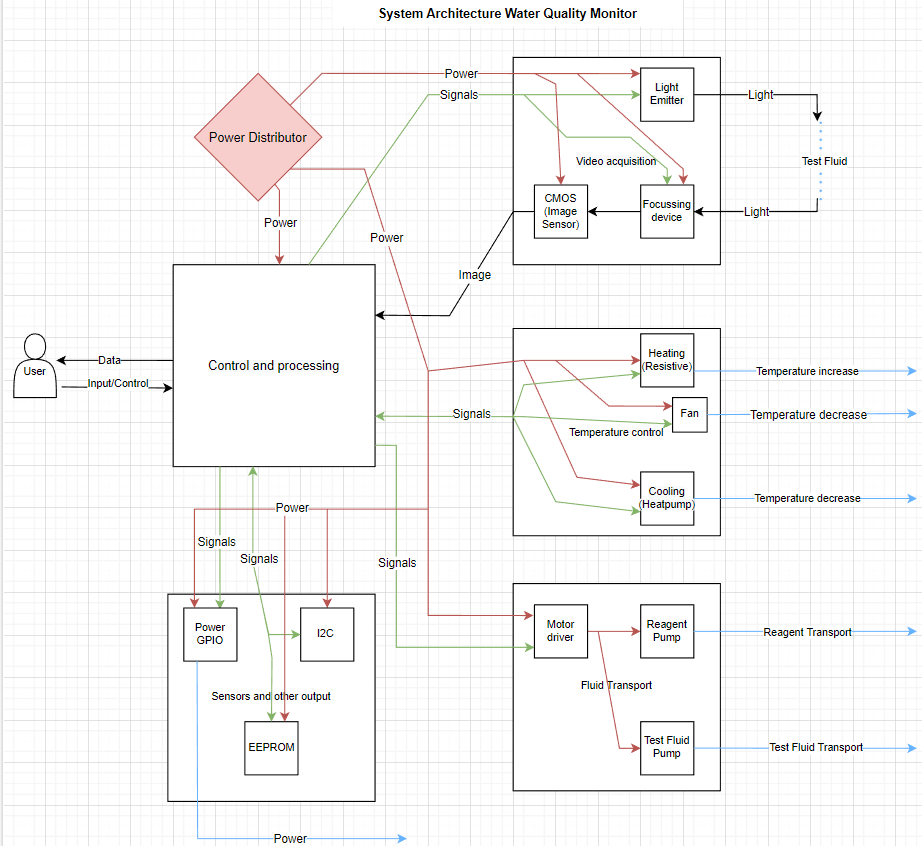


Figure 5 System Architecture

Table 3 Technical specifications

|  |  |  |
| --- | --- | --- |
| **Technical specifications** | | |
| **#** | **MoSCoW** | **Description** |
| **T1** | **M** | **A Raspberry Pi will be used as the computer and controller** |
| **T2** | **M** | **The Raspberry Pi will connect to a PCB hat (PI-HAT)** |
| **T3** | **M** | **The device will use a camera connected directly to the pi via a flat cable to take pictures and videos** |
| **T.3.1.0** | **M** | **The camera will be stationary, the lens itself will move** |
| **T3.1.1** | **M** | **The device will be able to focus and reposition the lens via motors or moving magnets (voicecoil)** |
| **T3.1.2** | **S** | **When using magnets (voicecoil), the drv8838 should be utilized** |
| **T4** | **M** | **The device will have a LED to create light for the microscope** |
| **T4.1** | **S** | **The led driver that should be utilized is the cn5711** |
| **T5** | **M** | **There will be stepper motors used** |
| **T5.1** | **S** | **The motors should be chosen according to the required strength** |
| **T5.2** | **S** | **The motor resolution should be below x degrees** |
| **T6** | **C** | **Some motors and sensors could be attachable via connectors** |
| **T6.1** | **S** | **All connectors should be standard (non proprietary)** |
| **T7** | **M** | **The device will cool the examination chamber with a Peltier module** |
| **T7.1** | **M** | **The Peltier module must be actively cooled to function and prevent damage** |
| **T7.2** | **M** | **The device will heat the examination chamber via resistive heating** |
| **T8** | **C** | **A flat cable could be used to connect the data lines and power to the PI-HAT** |
| **T8.1** | **S** | **The PI-Hat should distribute power to all devices including the Raspberry Pi** |
| **T8.2** | **W** | **The PI-HAT will not have any LEDs or status indicators** |
| **T8.3** | **C** | **The PI-HAT may provide a fan connector to cool the Raspberry Pi** |

The technical design is elaborated no further in chapter 13: Hardware. There I will show in detail the hardware choices that have been made.

# Hardware

The Rastaban hardware consists of multiple components and categories. It controls, light intensity, temperature, movement and it has room for other sensors and output. All hardware will be described below

## Microscope led

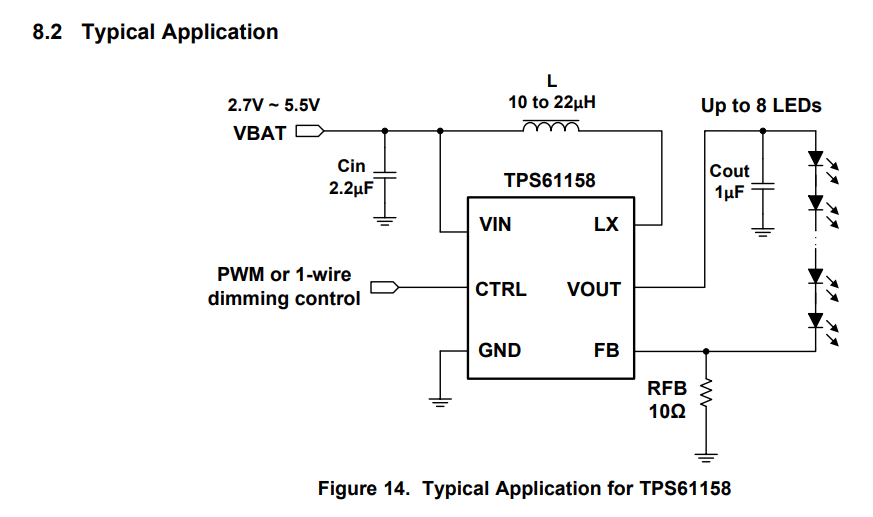
****The microscope led is the light source required for the camera to see an image in the reaction chamber. One of the options for driving the led was the TPS61158 driver.

Figure 6 Typical application TPS61158 driver

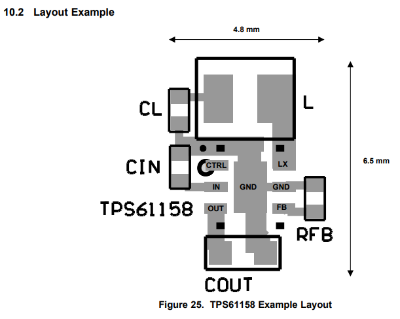
In this typical application circuit the input capacitor is to filter ripples from the power source and is there to function as a tiny buffer. The output capacitor is responsible for delivering current to the leds when the required current cannot be drawn continuously from the IC itself due to switching or power supply reasons. The inductor is there to function within the boost circuitry and has a high influence on the ripple and on maximum current and efficiency.

Figure 7 pcb recommendations

It is recommended to take advice from the pcb layout recommendations to avoid unsuspected behaviour due to unwanted resistance or capacitances.

However most of the component placements are common rules for PCB layout, so this layout recommendation can be considered trivial.

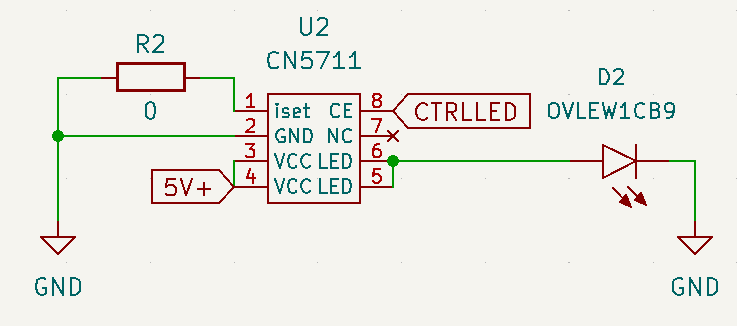
We later concluded that it was rather difficult to get our hands on the chips we wanted to use. This is why we chose to go with the CN5711 driver. This one is widely available on LCSC, AliExpress and similar shops.

Figure 8 finalized led driver circuit

The driver was used in the testing setup for the LED. We never intended to use this chip since I didn’t feel comfortable investing in a “no name/brandless” chip that may become obsolete sooner than later. However for this prototype it is sufficient. Driving the chip is done via a PWM signal, which is not proprietary, the code will always stay the same even with a new driver. R2 may be set if current limiting the driver is desired.

### Testing plan

In order to accurately capture images or videos of the reaction chamber, it is essential that the LED light used has a switching frequency that is higher than the camera's shutter speed. If the switching frequency is lower than the shutter speed, the resulting image or video may appear dark or flickering due to the incomplete exposure of the image sensor. To avoid this issue, it is necessary to determine the shutter speeds of the camera and select an appropriate LED switching frequency, particularly when dimming the light. This will ensure that the image or video is accurately captured with minimal artifacts.

Due to time limitations and other priorities I decided that testing this extensively is not smart. I tested the led by using a raspberry pi camera and my smartphone’s high speed camera starting a video capture and started dimming the led. I saw no direct flickering issues at most frequencies. I suspect that by further tweaking of the shutter speed of led switching frequencies it is possible to get even better results.

## Stepper motor drivers

There are many different stepper driver models available. Since we opt to go for a compact PCB and since we don’t need to supply high currents it is safe to say that industrial grade or high current drivers aren’t going to be suitable for us. There are small drivers like the like the A4988 or the DRV8425 which are very simple to control and do not have any special functions except for micro stepping (Max 16uSteps). When searching for drivers, I found that 3D printing stepper drivers are very capable and well tested and documented by the community. So I found the TMC2208 and the TMC2209 stepper drivers.

The TMC2209 and TMC2208 are both stepper motor drivers that have identical pinouts, meaning they can be used interchangeably in terms of their physical footprint on a printed circuit board (PCB). For considerations regarding PCB layout, refer to Chapter 19 of the TMC2208 datasheet. It should be noted, however, that the TMC2209 is the recommended replacement for the TMC2208 and TMC2130 in newer designs, as it is the more current and up-to-date version of these drivers.

A typical stepper motor operates at a resolution of 200 steps per complete revolution. Through the use of micro stepping, the resolution of a stepper motor can be increased to as many as 51200 steps per revolution (depending on the specific motor). This results in a resolution of 1/256 steps. The implementation of micro stepping has been shown to reduce noise levels, improve the smoothness and accuracy of motor operation, and potentially increase energy efficiency. It is worth noting, however, that micro stepping can also decrease torque, particularly at higher speeds, which may lead to stalling. Some drivers are equipped with the ability to adjust the stepping mode based on speed in order to mitigate this issue.

Source on Micro stepping <https://www.youtube.com/watch?v=G8oGa2mawKk&t=68s>

Afbeelding met tafel

Automatisch gegenereerde beschrijving

Figure 9 Micro stepping settings table

### UART on the TMC2209

The UART (Universal Asynchronous Receiver/Transmitter) interface is a serial communication protocol that allows for the transfer of data between devices. The TMC2209 stepper motor driver supports UART communication in addition to the traditional step/dir interface. Using UART to communicate with the TMC2209 can provide several benefits compared to using step/dir:

* Higher data transfer rates: UART allows for faster data transfer compared to step/dir, which can be useful for applications that require high-speed communication or precise control of the motor.
* More flexible control: UART allows for more advanced control of the motor, such as micro stepping, automatic load compensation, and stealthChop mode. These features are not available using the step/dir interface.
* Enhanced diagnostics and monitoring: UART allows for the monitoring of various internal parameters of the TMC2209, such as the temperature and current draw, which can be useful for debugging and performance optimization.
* Ease of use: UART can be easier to implement than step/dir, as it does not require the use of external pulse generators or counters.
* Using UART also removes the connection needed to step/dir. Using one RX line per driver and an enable pin to select the driver (with our current tmc2209 software).

Overall, using UART to communicate with the TMC2209 can provide improved performance, flexibility, and ease of use compared to using the step/dir interface. That’s why I settled for using this driver as our main stepper motor driver.

## Thermal Control

Thermal control will be in charge of providing energy and controlling all thermal regulating components of the Water Quality Monitoring (Rastaban) device fluids.

### The modules

The Thermal control consists of three power components:

1. Power Resistor
2. Peltier module
3. Fan (for Peltier)

These components are here to increase or lower the temperature of fluid that would be examined. The first two modules will be drawing very high currents (2A and higher) and therefore these powerlines need to be controlled and protected with care.

### Schematics

Figure 10 Thermal control components circuit

To control and protect the module and PCB from damage we need to building some passive protection. That’s why I chose some fuses that are roughly 1.25 times the value of the maximum current the module is supposed to use. When this value Is exceeded, the fuse will “blow” and the current will stop flowing.

The power mosfet used is the PMV15ENE**.** This NPN mosfet is designed to be controlled by a 3.3V logic level signal, which the raspberry pi uses as well. The mosfet is able to switch 6A and that’s why the component may seem overspecced. However, by using a higher current mosfet (lower RDS-on) there is higher efficiency because less energy is converted into resistive energy (heat). The component will also have a longer lifetime expectancy. Apart from this it also means there are no extra cooling components (heatsinks) required for this mosfet.

For the Peltier module we used a H-bridge since the Peltier module can cool or heat the same side when current is reversed. The module can help reaching higher temperatures in combination with the power resistor. When the current is reversed cooling the same surface can be achieved.

## Pi Hat i2c EEPROM interface

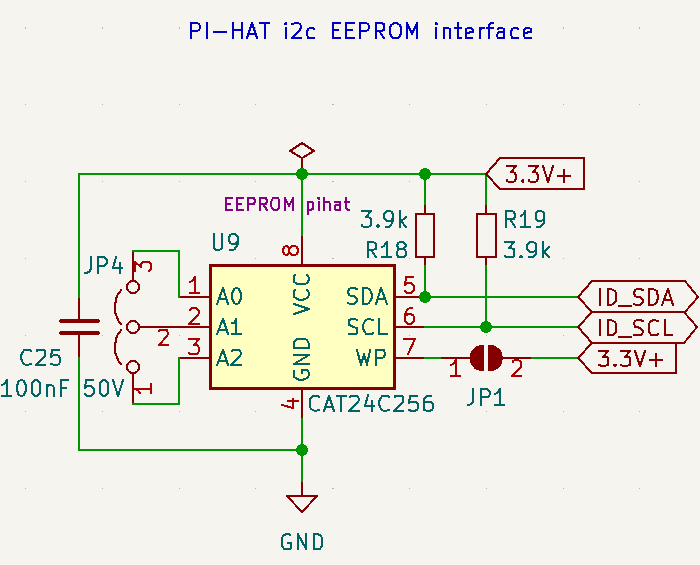
The i2c EEPROM interface was intended to create a script that would set the Raspberry Pi (RPI) in "Rastaban mode." The Sense Hat, developed by the Raspberry Pi Foundation, uses the EEPROM to identify the Sense Hat version and ID in the software used for the Hat. This allows the RPI to recognize different Hats and adjust settings, run appropriate code, and so on accordingly.

Figure 11 optional EEPROM circuit

However, we later realized that this may be unnecessary as we are programming the SD card to work specifically with this Hat and there will not be a need to support other boards. It is possible that the i2c EEPROM chip could be repurposed for safer data logging storage, as micro SD cards tend to become corrupt in RPI systems after extensive reading and writing.

## 12V Power GPIO

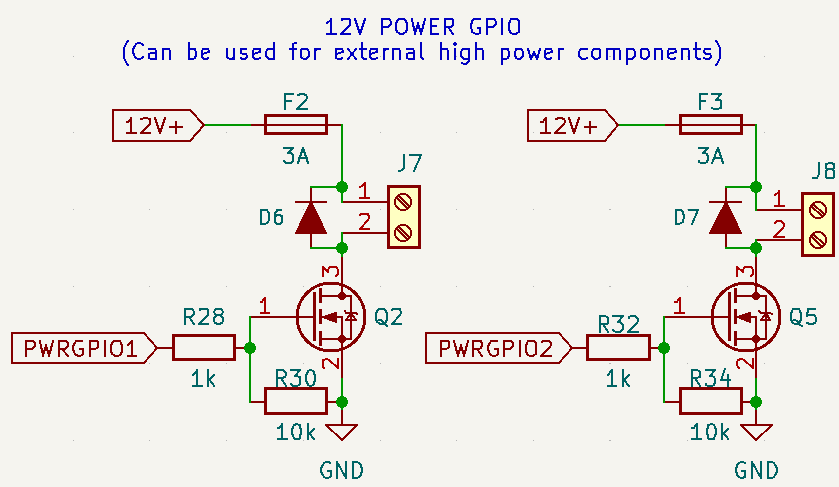
I added two power outputs for the user to connect high power loads to. The devices could theoretically draw to up to 6A, but I limited the fuses to 3A and the loads should be chosen accordingly or controlled with PWM to avoid tripping the fuse. The diodes are there to prevent back EMF destroying the mosfet or 12V supply when suddenly disconnecting an inductive load such as an motor. The 1k resistor on the PWRGPIO pin is for ESD protection of the pin and the 10k resistor is to define a state for the mosfet at all times even if the raspberry pi is disconnected physically.

Figure 12 PowerGPIO circuit

This prevents a floating pin and therefor unpredictable behaviour.

## PowerManagementSystem

The PowerManagementSystem (PMS) will be in charge of providing energy to all components of the Water Quality Monitoring (Rastaban). Rastaban has the following components that require power:

Table 4 Estimated maximum power usage



### Power Supply

This table is just an indication of the maximum power rating and it was made to give some insights. After seeing this table I think I can conclude that a 24V 10A (240W) power supply should be sufficient to run the device when it’s working in its normal consumption parameters. I don’t see the device ever drawing 11Amps. This power supply will be an external one for the time being, since it eliminates the need for a much bigger PCB and extra EMC and safety precautions. It also gives us the possibility to easily swap the PSU out if it gets damaged.

### Preventing Supply Noise

It is important to prevent noise that may be caused by some more power hungry components in the device. That’s the reason why I will try create power lines for 12V, 5.0V and 3.3V. The 12V rail will use a lot of power where the 5.0V and 3.3V are on the lower side of power usage. The use of capacitors will also smooth out the voltage dips that may occur while load becomes high.

#### Planes on PCB

There are multiple benefits to using ground planes, something that is already widely known. It improves thermals for heat inducing chips and it helps preventing EMC issues. It may however be wise to keep analog and digital grounds separated, to prevent ground loops and the noise it creates. We may use positive 12V or 5.0V planes as well, so we can transfer high currents to some loads without heating up the PCB, but this may introduce interference (EMC) with signal lines, something that is not tolerable.

Later in the project we figured that it is not possible to keep analog and digital grounds separated, since some ic’s have analog inputs ground that are connected to digital ground. If even one ic has this ground setup, than trying to separate the grounds won’t work. Adding the face that it makes routing the pcb more difficult, we will keep this idea in mind, but we won’t implement it for now. It is not smart to use 12V high power planes on the pcb since this can create magnetic fields that interrupt data lines or components.

### EMC

EMC (Electromagnetic Compatibility) refers to the ability of electronic devices and systems to function properly in their intended electromagnetic environment without causing interference to other devices or systems. In the context of PCB (Printed Circuit Board) design, EMC refers to the design practices and measures that are taken to ensure that a PCB does not emit or receive electromagnetic interference (EMI) that could affect the performance or reliability of other electronic devices. There are several strategies that can be used to prevent unwanted EMC issues on a PCB:

1. Use proper grounding: Proper grounding is essential for EMC compliance. Use a solid ground plane and ensure that all ground connections are made as direct and low-impedance as possible.
2. Use proper power distribution: Proper power distribution helps to prevent voltage drops and noise on the power supply lines, which can cause EMI. Use a dedicated power plane and ensure that the power supply lines have low impedance.
3. Use proper decoupling: Decoupling capacitors help to filter out noise on the power supply lines and should be placed as close as possible to the power pins of all active components.
4. Use shielded components: Shielded components, such as shielded inductors and transformers, can help to reduce EMI.
5. Use proper routing: Proper routing helps to reduce the length and proximity of high-frequency signals, which can reduce EMI. Use a differential pair routing technique for high-speed signals and avoid routing signals near sensitive components or traces.

By following these design practices, it is possible to prevent unwanted EMC issues on a PCB and ensure that the electronic system functions properly in its intended environment. I have tried to implement a few of these strategies like point 1, 3 and possibly 5 if the pcb size allows this. Some strategies are not viable, cost effective or mandatory to get the system working properly.

I used a ground plane on the front and back of all my 2 layer PCB designs. This makes connecting al ground connections easier but I suspect it also creates a “shield” for underlaying layers when using 4 layer PCB’s where the second layer is a ground layer. I further made sure that my coil for the buck converter is properly shielded with the same ground plane and I routed no data lines beneath the coil. Since the coil produces an electric magnetic field it is wise to do so.

### ESD protection

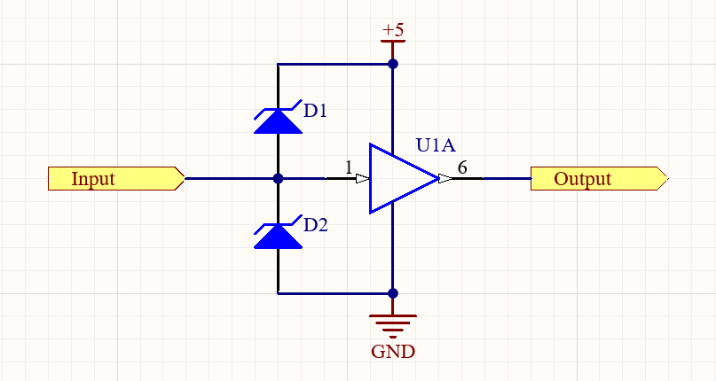
A voltage clamping diode circuit is used to prevent excessive voltages from accumulating at the input terminal of a buffer or differential input of an operational amplifier (op-amp). The circuit consists of two diodes, D1 and D2, which are reverse-biased under normal conditions. When the input voltage exceeds the supply rail voltage, D1 becomes forward-biased and conducts, limiting the voltage at the input. Similarly, when the input voltage falls below ground, D2 becomes forward-biased and conducts, again limiting the voltage at the input. This circuit is useful for protecting sensitive input stages, such as those found on I2C data lines.

Figure 13 ESD protection using TVS diodes

We later decided to implement this, as we can always omit the diodes on the PCB if it may not be necessary after all.

### Power regulators

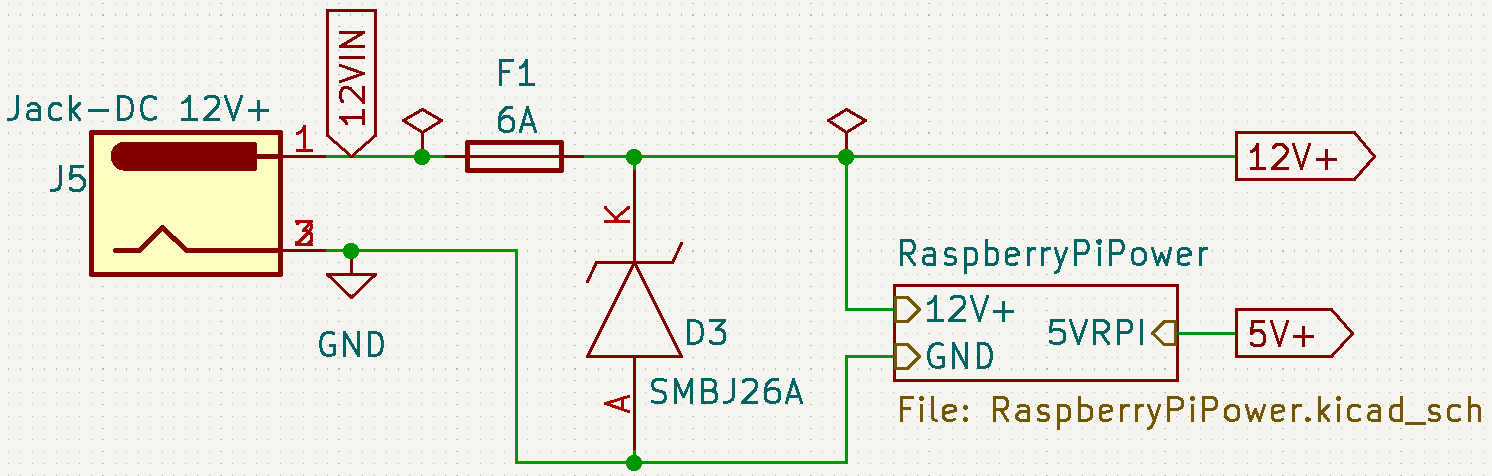
There are two main methods for adjusting the voltage in a circuit: Buck/Boost converters and linear regulators. Buck/Boost converters are more efficient and produce less heat, but they require more components and can cause noise on the supply line due to switching. Linear regulators, on the other hand, waste a lot of energy as heat but produce less noise on the output voltage. However, in cases where the difference between the input and output voltage is not significant, linear regulators can be more efficient than switching regulators. The choice between these two options depends on the specific requirements of the device. In this case, we prioritize functionality and repairability, so we would likely choose a linear regulator. We ultimately decided to use two step-down converters, both of which are switching regulators. The first one is used to power the Raspberry Pi at 5.0V and can deliver up to 3A if needed. I chose this because the Raspberry Pi will eventually be used with a camera and that can lead to higher current usage when compared to using just the raspberry without peripherals. Other components that also run on 5.0V, such as the microscope LED and the voicecoil driver (with a buck converter IC), are powered by this converter as well. The 5.0V converter also supplies the 5.0V breakout of the breakout power pin header. It is unlikely that protection will be necessary, as the LM2596 5.0 has built-in thermal and overvoltage protection. The second converter is used to provide 3.3V and is based on the TLV1117-33 IC, which can deliver up to 3A. The output of this converter is connected to the voice coil motor driver (DRV8838).

Figure 14 Power Management System circuit (simplified)

### 12V vs 24V

Since the primary and secondary stepper motors are designed to run on 12V we think it may be wise to choose a 12V supply. 24V gives us more headroom for voltage dips, but the motors can probably not handle the voltage difference. It is also less efficient to buck a higher voltage to a lower voltage, so keeping the difference lower is better. We do need to compensate for the possible voltage dips with capacitors and a PSU with a rather high current (probably 10A). I think that a 120W power supply would suffice for this prototype. As an addition we found that the Peltier module usually work on lower voltages like 12V, 5.0V and even 2.2V when they are really small (1cm by 1cm). bucking the voltage from 24 to 12 or lower and still having >3A currents will be very difficult when it comes to sizes of IC’s. using 12V is again beneficial in this case.

### Trace width

Afbeelding met tafel

Automatisch gegenereerde beschrijvingSince high currents will run through the board to some components, we need to create thicker wires other wise the resistance will be too high and the board will heat drastically, possibly leading to traces burning up or degrading over time due to temperature fluctuations.

Figure 15 Trace Width Calculator tool

I used an online calculator to get a overall indication of the track width required. Sadly it seems impossible to create a 5.62 mm trace for every high power net. I decided to go with a trace with that was still possible to route and that roughly resembled the width of the package pins of the power IC use. This may not be the most professional way to test and see if this would work, but it is the only way to go. It later seemed to work perfectly fine using 0.6mm wide traces. It would be a great experiment to check the temperature of the board with a high resolution thermal camera.

I included rounded tracks in the PCB design for two reasons:

1. Rounded tracks serve a functional purpose when signals are being transmitted at above 1Ghz.
2. Rounded tracks give the PCB a visually distinct appearance that can make it stand out. I found it nice to use since the rounded tracks remind me of water.

Overall, the inclusion of rounded tracks in the design serves mostly a aesthetic purpose.

### 2 VS 4 layer PCB

A PCB (Printed Circuit Board) consists of one or more layers of conductive material, typically copper, separated by insulating layers known as core and prepreg. The number of layers in a PCB can range from one to many, depending on the complexity of the circuit and the required performance.

Here are the main differences between 2-layer and 4-layer PCBs:

1. Number of layers: As the name suggests, a 2-layer PCB has two layers of conductive material, while a 4-layer PCB has four layers.
2. Layout complexity: 2-layer PCBs are typically limited to simple circuits with a small number of components, while 4-layer PCBs can accommodate more complex circuits with a larger number of components.
3. Signal routing: 2-layer PCBs have limited routing options and may require the use of vias (small holes that connect different layers) to route signals between layers. In contrast, 4-layer PCBs have more routing options and typically do not require the use of vias.
4. Performance: 4-layer PCBs can provide better performance compared to 2-layer PCBs, as they offer more routing options and can reduce the effects of crosstalk and noise.

In summary, 2-layer PCBs are suitable for simple circuits with a small number of components, while 4-layer PCBs are better suited for more complex circuits with a larger number of components and higher performance requirements.

In order to minimize the size of the PCB and facilitate its ease of transport and mounting on a Raspberry Pi, we chose to use a 4-layer PCB in the final design. This allows us to easily separate the power planes from the signal wires, which can improve the performance of the circuit. In future designs, it may be beneficial to include a ground plane between the signal and power planes to further improve the performance and EMC (Electromagnetic Compatibility) characteristics of the PCB.

### Coil whine

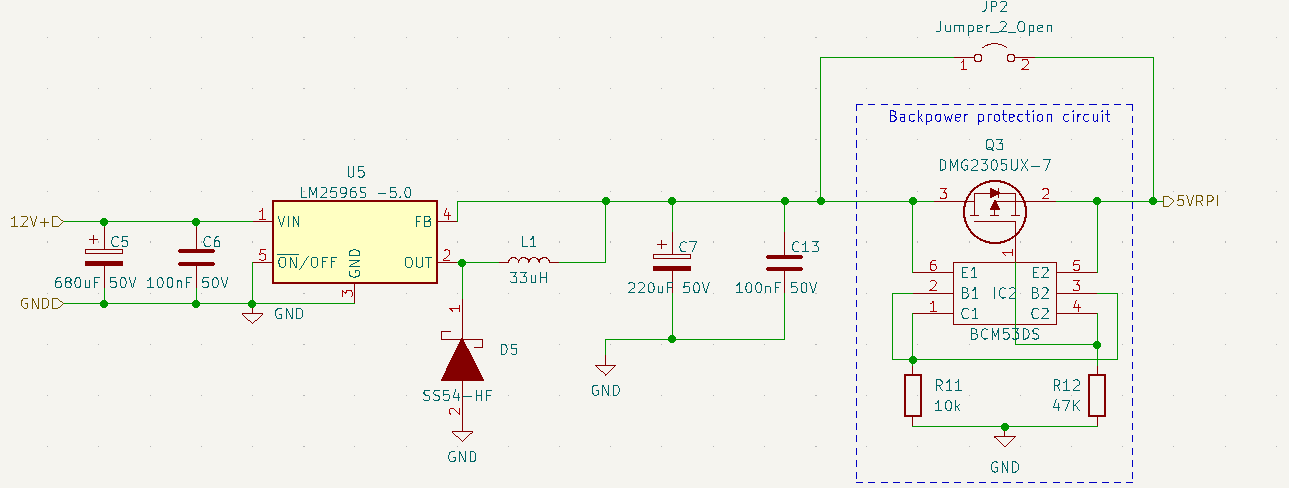
Upon connecting a minor load (300 mA) to the 5V Raspberry pi power circuit, I noticed that it generated a high-pitched whine. Upon further investigation, it was determined that the noise was likely emanating from the coil. Upon reviewing the circuit design, I realized that I had omitted the decoupling capacitor from the output of the LM2596 5.0 regulator. This likely resulted in the output producing pulses in the kHz range that were audible through the coil.

Figure 16 5V power circuit

To resolve this issue, I added an 220 uF capacitor to the output of the regulator. This effectively eliminated the whine and improved the performance of the circuit.

## Backpower protection circuit

The Backpower Protection Circuit (BPC) is designed to protect the Raspberry Pi from damage in the event that a 5V supply is connected to the micro USB port while a 12V supply is simultaneously connected to the barrel jack on the Pi Hat. However, due to a lack of necessary components, this circuit was not tested during the development of the Rastaban project. The BPC can be bypassed by using the solder jumper JP2. To further mitigate the risk of unintended power connections, the housing for the Rastaban project could be designed to cover the micro USB connector.

## PCB versions

When fabricating PCBs, it is common to go through multiple iterations before achieving a functional prototype. It is wise to test everything on a breadboard first, but this is not always feasible due to the size of certain components (not breadboard-friendly), the complexity of the circuit, and sometimes the parasitical behaviour of the breadboard, which can affect the performance of the prototype. During my internship, I designed three versions of the Rastaban HAT, as well as several simpler PCBs.

### PCB V0.1

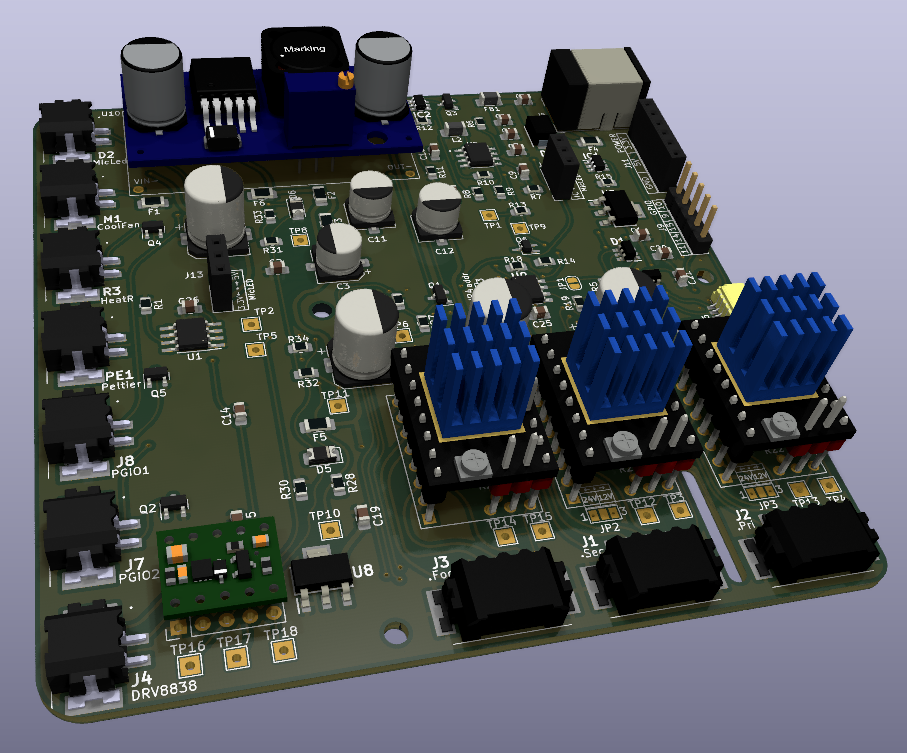


Figure 17 PCB V0.1

The first version of the PCB was designed to explore multiple options when it comes to driving the motors, led and power the components.

Successes:

* Most of the functions/ elements work
* This layout seems to be a good starting point, maybe some small refinements.
* The mounting holes and dimensions are all correct.

Sadly, the first version of the HAT had more problems than I expected, but I was able to make all the necessary improvements and get a fully functional prototype. I documented all the changes I made, so that I could easily apply them to future PCB’s.

Notes for V0.2 of the Rastaban PCB

Table 5 V0.2 notes

|  |
| --- |
| Improvement/note |
| CHECK THE ORIENTATION OF THE RASPBERRY PI HEADER FOOTPRINT(next time, double, double check) |
| VCC pin on the drv8838 driver must be connected to 3.3V or 5.0V for it to work. |
| Try to draw high power traces away from signal traces (move connectors?) |
| For the 2209 we could use UART port, so we can control clock, micro stepping etc, set parameters, delays, coolstepping, etc. |
| Buy the right parts 10uf 50V not 10uf 10V |
| FB1 will not work (made for 150mah), change for different inductor. |
| Order a lot of Phoenix ptsm connectors (both receptacle and plug) |
| Remove or find smd variants for fuses (remove some fuses/change values?) |
| Bypass backpower protection circuit (jumper) |
| Raspberry pi has decoupling capacitors (caps). Remove caps that are not necessary. Leave c11 and c8.  C5 should be 22uf. |
| Raspberry pi has buildin esd for gpio. Raspberry pi is connected to ground via power connector. Leave ESD footprints, but don’t solder them. |
| Simplify (or maybe even remove) raspberry pi power circuit. |
| Change inductor for rt7272A boost/buck circuit. Chip gets HOT, possibly switch to buck boost converter or worse case: external PSU for RPI (micro USB)? |
| Use a 40 pin FEMALE header on the pcb, not male |
| Make the schematic more readable. Don’t run wires through each other. |
| Fix u7 (footprint is wrong, partly flipped) |
| Flat cable for 40 pin to 40 pin connection for easier, safer? For testing purposes. |
| Better diode polarity indication on silkscreen. |
| J12 silkscreen error: ic and fet are flipped |
| Possibly add pinout (names) to bottom of HAT |
| ENABLE focus must be using a hierarchal flag! Now it’s global, it’s not consistent. |
| All stepper motors work fine with 12V, they have more than enough torque even while using 16bit micro stepping. Remove the 24V buck boost converter. |
| Peltier module has not been tested (0.2 sense resistor not received) |
| Micled IC not tested (ic BCR420UW6-7 not received). |
| I2c hat functionality (special eeprom chip) not tested. |
| 5.0V for mic led not tested. 3.3V from u2 (tlv1117-33) works fine. |
| ESD protection not tested. |
| Maybe use the xl6009 for the raspberry pi and 5V line (4A is possibly more stable than the 3A) or copy a known design ([df robot?](https://www.tinytronics.nl/shop/en/power/voltage-converters/buck-(step-down)-converters/dfrobot-dc-dc-buck-converter-7-24v-to-5v-4a)) |
| Remove some test points to make pcb smaller and routing easier? |
| I2C NOT working probably due to ESD protection ic. Remove the ic, it’s expensive and time consuming. Use diode alternative, which is optional:  Figure 18 ESD TVS diode option  evaluating ESD protection for 3.3V i2c - Electrical Engineering Stack  Exchange |

Small discussion with Jeroen:

* Is a sense resistor a special type of resistor -> figure this out.
* Maybe 5 1ohm resistors in parallel work as an alternative for this sense resistor -> try this to find out if this is something to continue.
* Get current h-bridge working or search for alternative (make your own h-bridge with mosfets maybe, higher currents?)
* Try to fix the power circuit to gain knowledge on the subject.
* Remember: 1 small value cap for low frequency, 1 larger nf cap for higher frequencies.
* Drv8838 keeps turning somehow after disabling it.

Conclusions on process:

* Don’t reflow large capacitors. Do it by hand after the small parts have been soldered.
* 230 degrees Celsius is usually hot enough to melt the solder paste.
* Stencils for small footprints diy soldering, otherwise solder connections will fail.
* Always three dubbel check the most important things (footprints).
* Do some life testing so create mockups for pcb’s to test fit (also for Industrial designers).
* Check solder connections! Visual inspection! A mosfet with only 2 pins soldered instead of all three does not work.
* Don’t unplug things while they are powered on (hot plug/swap), plug things or measure things while the system is powered.

#### Current usage

I tested what the current usage of the components. Some components could not be tested and have been estimated.

Table 6 Current usage

|  |  |  |
| --- | --- | --- |
| Component type | Name | Current (A) usage tot. |
| LED | Microscope led | 0.077 |
| FAN | Cooling FAN | 0.150 |
| HEATER | Power Resistor (10w) | 1.180 |
| Cooler/heater | Peltier | ? 0-3.6 |
| PGPIO | X | 0-6 |
| PGPIO | X | 0-6 |
| Stepper motor (small) | Focus steppermotors | 0.5 (3 motors) |
| Stepper motor (large) | Primary steppermotor | 1.7 |
| Stepper motor (medium) | Secondary steppermotor | 1.5 |
|  |  |  |

### PCB V0.2

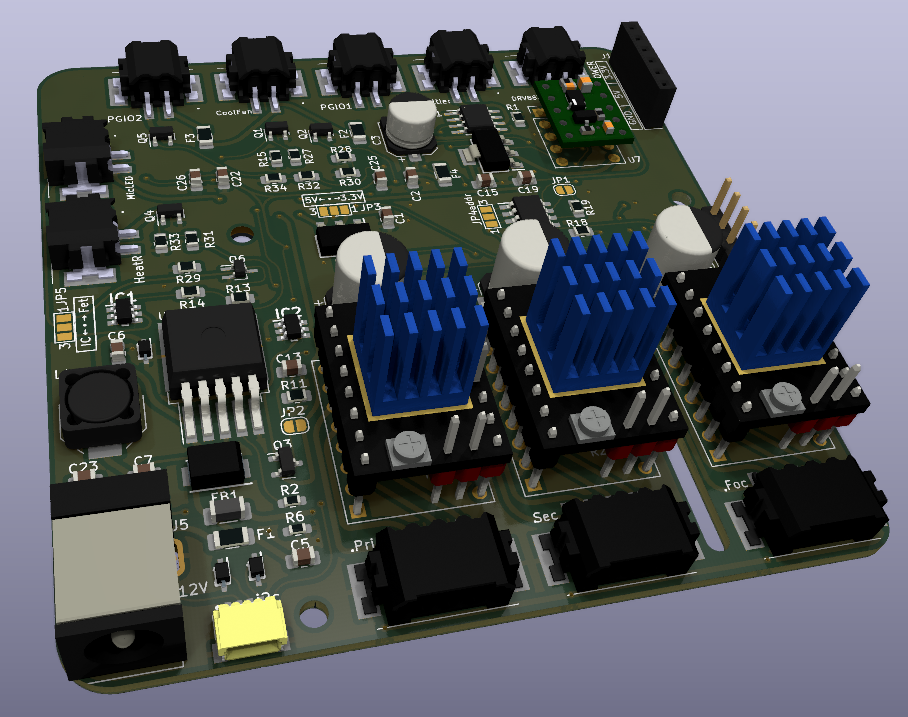


Figure 19 PCB V0.2

Successes:

* All functions/elements are working now, this version could be used for fluid testing purposes.
* This version is smaller than it’s predecessor L9,9CM; W9,9CM -> L8,4CM; W8,4CM.
* This version is capable of powering the raspberry pi through the GPIO header. External power for the raspberry pi Is no longer necessary.

Notes for V0.3 of the Rastaban PCB

Table 7 V0.3 notes

|  |  |  |
| --- | --- | --- |
| # | Improvement | Finished |
| 1 | Figure out which hardware pins to use for what components led/focusmotor (drv8838) | NO |
| 2 | TMC 2209 uses different ms1 ms2 configuration for microstepping than the tmc 2208! Keep this in mind. | DONE, new design uses TMC2209 design config |
| 3 | Remove tmc2208 from design, focus on 2209 | DONE, no more tmc2208. |
| 4 | Use UART on 2209 and remove step/dir enable interface. | DONE, UART is only control way now |
| 5 | Use appropriate resistors for UART control on 2209 (resistance should decrease with increase in drivers). | DONE |
| 6 | Connect diag pin of 2209 for stall (stuck motor) indication for rpi? | NO |
| 7 | Checkout the problems with diag pin on tmc2209 (see pdf in datasheets) | NOT CHECKED |
| 8 | Change L1 footprint | Enlarged and changed to bourns square size |
| 9 | Copper zone for L1 | Copper ground zone added under L1 for MRI protection |
| 10 | Maybe move the camera connector to the left | YES, DONE |
| 11 | Corners in the hat header silkscreen. | Already satisfied in previous version |
| 12 | Raspberry pi power circuit coil whine should be resolved. | Solved, new values added to schematic |
| 13 | Footprints for all power circuit components should be enlarged/checked | Checked, DONE |
| 14 | Add logo v0.2 | DONE |

### PCB V0.3

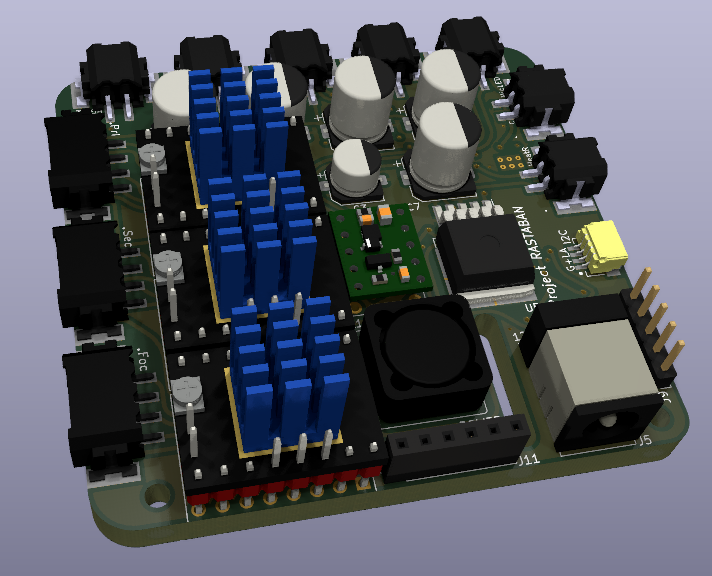


Figure 20 PCB V0.3

Successes:

* All functions/elements are still working.
* This version is smaller than it’s predecessor L8,4CM; W8,4CM -> L6.8CM; W6.5CM.
* Some footprints changed/enlarged.
* Removed step/dir connections and now using UART on TMC2209 driver.
* Added copper ground zone underneath L1 for MRI protection.
* Refined dimensions for camera cable hole.
* Newly created Rastaban logo added.

Notes for V0.4 of the Rastaban PCB

Table 8 V0.4 notes

|  |  |  |
| --- | --- | --- |
| # | Improvement/note | Implemented/finished |
| 1 | Make the H-bridge compatible with different Peltier module sizes (different voltages) |  |
| 2 | Possibly breakout more GPIO pins |  |
| 3 | Watch out for PEO connector on RPI (keep out zone possibly) R2 is now touching the PEO connector. |  |
| 4 | Heater and microscopeled are behind the USB port of the pi, this is not an option. Next time load in raspberry pi 3D model to catch this issue beforehand. Move connectors or use vertical connectors (easier). For now it is possible to move the hat a bit more up using a taller header or an extender possibly. |  |
| 5 | Write insertion method for module like the drv8838 and the tmc2209 driver boards (to avoid reversed polarity) |  |
| 6 | The I2c port should be vertical one as well to keep this small profile board. |  |
| 7 | It is possible to apply D4 and D1 footprints for D7 and D6 (smaller footprints) |  |
| 8 | CPU heatsink cannot be placed because of components on bottom side of PCB |  |
| 9 | Anode cathode of Diodes may be misleading. The “C” is not always the way the diode should be pointed to (D5, D3) |  |
| 10 | Remove Resistor R5 and R4. They are not necessary. One 1K resistor works for UART. The silkscreen can be removed as well. |  |
| 11 | Make sure all stepper motor outputs are wired the same from driver to the connector output. Some are switched up, this could be a problem. |  |
| 12 | Keep Step and Dir on all drivers, there are enough GPIO pins left. |  |
| 13 | VIO appears to not be required on the tmc2209 when VM is connected. Remove 3.3V from VIO |  |

For final reference I placed the three PCB designs for Rastaban side by side to accentuate the size reduction and overall component placement improvement over the versions.

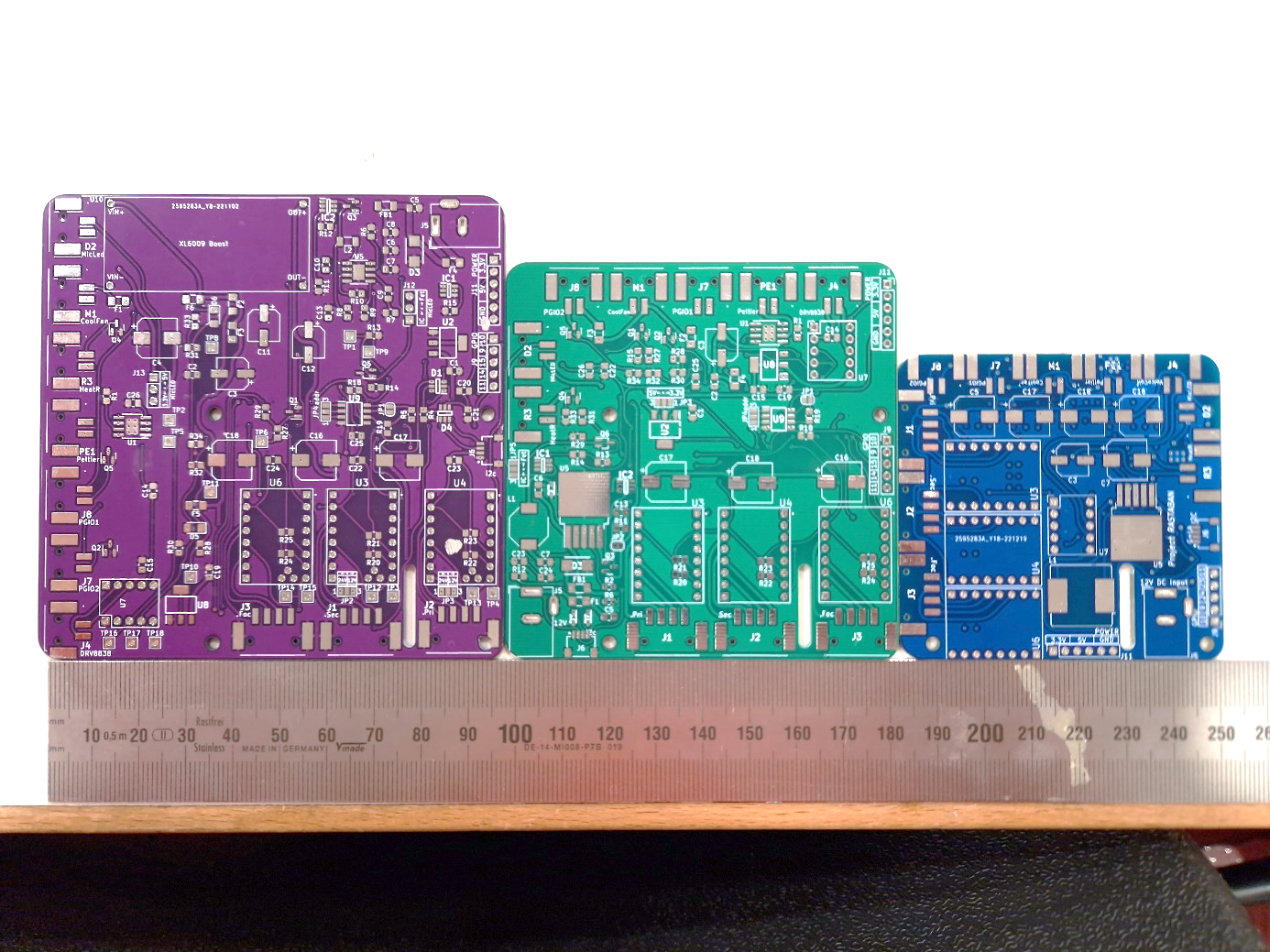


Figure 21 PCB V0.1-V0.3

## Focusing the lens

When imaging small objects through a lens, it is crucial to use a high-quality lens with sufficient magnification to clearly resolve the details of the objects. The working distance of the lens, or the distance between the lens and the object, should also be considered as it affects the ability to achieve focus.

There are several ways to adjust the distance between the lens and the object. One method is to use a voice coil, which works by magnetizing the coil to move it slightly closer or farther from the object, similar to the principle of a speaker coil. The Rastaban project can use a voice coil setup controlled by, for example, a DRV8838 driver. Another method is to use stepper motors to move the lens up and down. The design created by Jeroen uses three stepper motors in parallel for this purpose and will be controlled using TMC2209 drivers. It may be necessary to limit the current provided by the TMC2209 drivers through the UART settings in order to protect the small and vulnerable motors.

# Software

The software written for the Rastaban (HAT) has been made in the Python language. This language is chosen because it is widely used in the Raspberry pi community and it gives us the ability to use the PYQT GUI library. Other reasons are maybe for the ease of use and the reason that it is not compiled but interpreted code, which makes it faster for prototyping purposes. Most of the code is written in an object oriented style. I find this code looking cleaner and more maintainable than the regular coding style I know.

## Components

To give a short overview of the code, I will describe all components in short. I will not go into detail on the functions of the components, since the DoxyGen file is much more conclusive and is fluid in it’s versions, while this document may become obsolete sooner or later. Component list:

* Microscope LED
* Stepper motor driver
* PowerGPIO
* Cooling Fan
* Peltier Module
* Voicecoil
* PIGPIO
* Init (PIGPIO)

### Microscope LED

The CN5711 is utilized to control the microscope led. The device is driven by a Hardware PWM signal send by the raspberry pi. The code can control the LED’s brightness, switching frequency, duty cycle and on/off state.

### TMC2209 Stepper motor driver

We can use both step/dir commands to control the motor drivers for “legacy” stepper motor control, as we can use UART, the more modern way to control and set the stepper motor drivers. V0.3 of the Rastaban PCB only uses UART to control the stepper motor drivers.

The TMC2209 stepper motor driver can be controlled through its UART (Universal Asynchronous Receiver/Transmitter) interface. To use this interface, a UART-to-serial converter (such as a USB-to-serial adapter) is used to connect the TMC2209 to a microcontroller or computer. The microcontroller or computer can then send commands to the TMC2209 using a specific protocol, and receive status information from the TMC2209 in return. This allows for real-time customization and monitoring of the driver's behavior. We use the following library to control the drivers via UART: <https://github.com/Chr157i4n/TMC2209_Raspberry_Pi>

### PowerGPIO, Cooling Fan, Heating resistor

The power GPIO pins are driven using software PWM as well as the Cooling Fan and the Heating resistor. This code drives the mosfet that is responsible for switching the load.

The Peltier driver is controlled using two software PWM signals. Sending to IN1 and IN2 will control the output of the H-bridge.

### Voicecoil

The Voicecoil code controls the voicecoil for focusing the lens. The code was designed for the DRV8838 motor driver.

### Pigpio

Pigpiod or pigpio is a utility which launches the pigpio library as a daemon.  
Once launched the pigpio library runs in the background accepting commands from the pipe and socket interfaces.  
The pigpiod utility requires sudo privileges to launch the library but thereafter the pipe and socket commands may be issued by normal users. There are several reasons why we might choose to use the pigpio library:

* It provides a simple interface for controlling the GPIO pins: The pigpio library provides functions for setting the direction and level of individual pins, as well as for configuring PWM and detecting changes on the pins.
* It is lightweight and efficient: The pigpio daemon runs in the background and communicates with the pigpio library via a socket, which means that it has minimal overhead and does not require any additional libraries to be installed.
* It is flexible: The pigpio library allows users to control the GPIO pins from multiple programming languages, including C, Python, and Perl.
* It is well documented: The pigpio library comes with detailed documentation that explains how to use the library and troubleshoot any issues that may arise.

Overall, the pigpio library is a popular choice for controlling the GPIO pins on a Raspberry Pi because it is easy to use, efficient, flexible, and well documented. That is the reason why we choose to implement it in our Rastaban code.

### Init function (PIGPIO daemon)

I created an init function for the Raspberry Pi to address unexpected behaviour of the GPIO pins when using the pigpio daemon. Some pins were unable to go LOW or did not properly utilize PWM. If you wish to run Rastaban code that utilizes PIGPIO functions, you can run the Init function once to start the pigpio daemon. The function will indicate whether the daemon was already running and display its process ID. By restart or after a shutdown of the raspberry pi, the daemon would be stopped. It is possible to run this code on start-up of the RPI, but it may be better to run this code when the final Rastaban code has been created.

## Doxygen

I chose to use Doxygen because for the following reasons:

1. Improved code readability: Doxygen can help you create clear and concise documentation for your code, which can make it easier for other developers to understand and use your code.
2. Enhanced collaboration: Doxygen can help you document your code in a way that is more accessible to other developers, which can facilitate collaboration and make it easier for others to contribute to your project.
3. Increased code maintainability: By documenting your code with Doxygen, you can make it easier for other developers to understand and maintain your code over time. This can be particularly important for larger projects with many contributors.
4. Professional-quality documentation: Doxygen can generate professional-quality documentation in a variety of formats, including HTML, LaTeX, and PDF, making it easy to create high-quality documentation for your project.

Overall, using Doxygen can help create more readable, maintainable, and collaborative software projects by providing comprehensive documentation for my code. Therefore, all the code for the Rastaban has been processed using Doxygen and is primarily documented in this way.

## GUI

It was planned to control the Rastaban project via an GUI. This could make it easier to control the setup for the end user. As said before the user may be a student, teacher or possibly a researcher who most likely has not a very wide understanding of the Rastaban code or product as a whole. This GUI should make it easy for them to interact with the device. Below is shown the function of the GUI in the system.

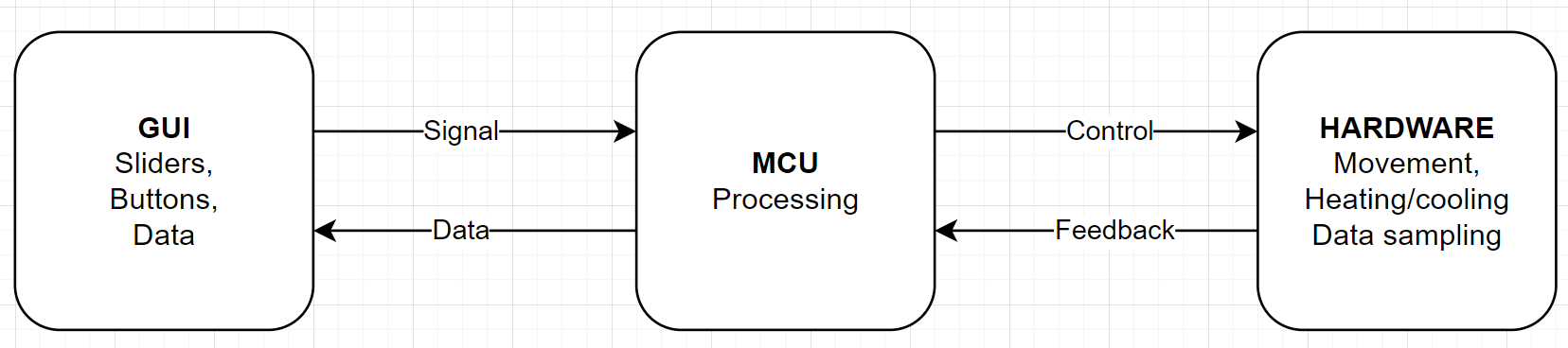


Figure 22 Application of GUI in the system

I created a crude window sketch for controlling the Rastaban project.

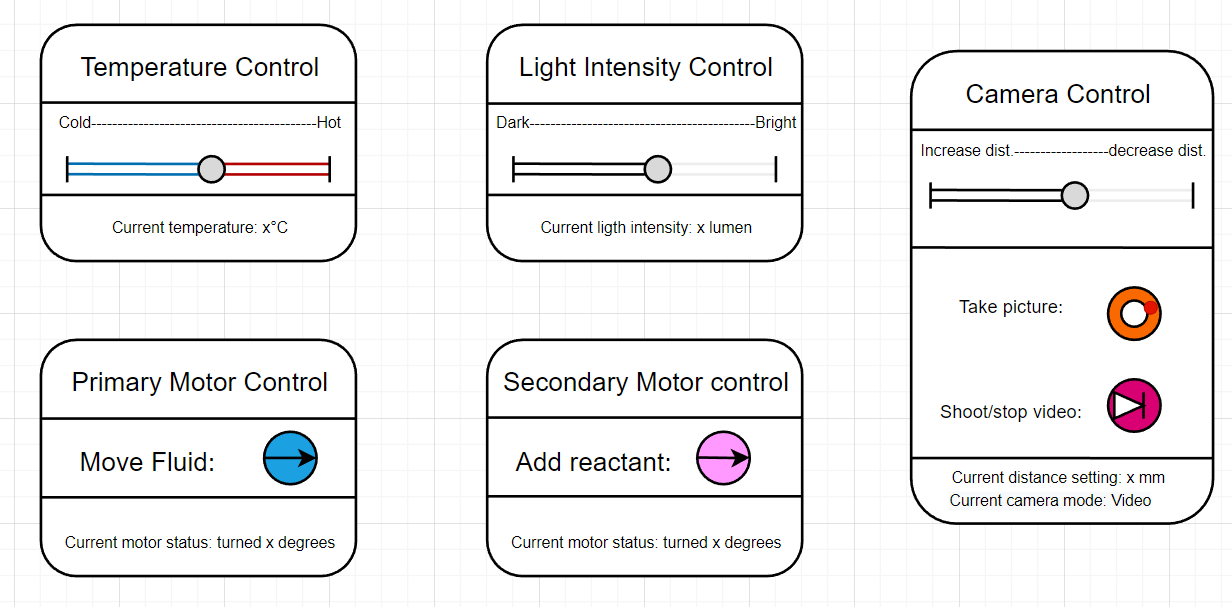


Figure 23 Windows sketch GUI

### GUI design

Figure 24 GUI widget options in one window

After having a general idea of how I wanted my GUI to function, I searched for possible input methods to put in my GUI. There are many options, sliders, dials etc. I ultimately decided to go with, in my opinion, easiest and safest option, by using only buttons and arrows that need to be clicked or scrolled on. I think that using a dial can be useful for an led for example for quick testing, but not for high power components like the Peltier or heating resistor. I think it would be dangerous to accidentally turn a dial to these high power components and a dial or slider doesn’t seem to be precise enough compared to setting numbers.

### Final design sketch

Afbeelding met tekst

Automatisch gegenereerde beschrijvingUltimately I think that functionality and safety in the GUI are of highest priority. That’s why in my final design sketch I made buttons to toggle enable/disable to specific blocks of control or the whole control window.

Figure 25 Final design sketch GUI

Too have full and precise control over the components, numbers can be set manually in the text boxes. The numbers can be changed by typing them in, scrolling on the box or clicking on the arrow, up/down.

This window/GUI may be build upon to give it functionality and add signals and slots to integrate it with the main Rastaban code.

# Appendix A: ChatGPT

## ChatGPT usage in this report

This year a new AI was introduced: ChatGPT. I will let it introduce itself: Afbeelding met tekst

Automatisch gegenereerde beschrijving

It however is very capable of doing a wide variety of things. It can give you code snippets, rewrite text, rewrite code etc. It is also capable of creating poems with the input of your choice. Or as it says itself.

Afbeelding met tekst

Automatisch gegenereerde beschrijvingI have used ChatGPT for the following:

1. Improve writing style: the AI rewrites sentences to sound more fluid, more efficient, conclusive and shorter.
2. Fetch facts on electrical components/theory: the AI analyses my question on a specific theory or component and gives me a list or short answer on how it works, how to use it, pros and cons on it etc.

Afbeelding met tekst

Automatisch gegenereerde beschrijvingI used the AI as an tool for this report. Al the text that is in this report is written by myself, but improved by the AI. This could be by adding text, changing words, changing grammar, removing unnecessary text or reformatting it. The AI can also write text on itself by giving it parameters. For example: “write a text on stepper motors in 100 words in a scientific form.” The AI will generate this within a minute. It is also possible to generate follow up questions like: “rewrite the previous text, but now in 50 words and for a child of 12 years old.” It wrote the text in 84 words, so as you can see, the AI is not perfect yet, but maybe rephrasing next time would help.

## Review of using ChatGPT

I found that using ChatGPT can be very beneficial when it comes to rewriting text that I already wrote myself to be more readable. ChatGPT is very good at formatting text in a chronological order and it really seems to have an understanding of your goal. Of course ChatGPT is using your input and the input of it’s database (A large part of the internet up to 2021) to extend on your requests. But this doesn’t mean that it always has the answer or text that you are looking for. A big part of using ChatGPT is therefor checking it’s output before using it in general. This is true for text, but also for software, since ChatGPT can create this too.

I would absolutely advice others to use ChatGPT as a text writing but mostly formatting tool for their reports. It has the power to rewrite your text so it is more readable, has a better progression or sounds more natural or professional for example. It is also very capable of helping you with software coding challenges and some basic questions, but the user should always be weary of copying any of its output into their report.

## Notes

By the time of writing, ChatGPT is a brand new AI tool from openAI that is free to use for everyone. You do need to create an account and provide your phone number and email address. This raises some questions for me. First of all, nothing is free, so why is there no subscription required? I suspect that the AI is being trained by everyone that uses it. You are able to give feedback relatively easy and the site tells you it is still experimental. I suspect that this tool will be behind a paywall sooner or later since so many people are using the tool at this moment. I also am a bit weary that this tool may influence the ability of mine to write texts will degrade. I can just ask ChatGPT to make it look “nicer” or let it write most of the text in general. Nothing is for free and I don’t think that the AI is made to “benefit humanity as a whole.”. as stated on <https://en.as.com/latest_news/what-is-elon-musks-relationship-with-chatgpt-the-famous-ai-chatbot-n/>

Afbeelding met tekst

Automatisch gegenereerde beschrijving

# Appendix B: KiCad

## What is KiCad

KiCad is a free and open source electronic design automation (EDA) software used for creating electronic schematics, printed circuit boards, and PCB layouts. KiCad is used mainly by engineers, hobbyists, and makers to design and document electronic circuits. It supports a wide range of components and features an easy-to-use graphical user interface. I used KiCad to create all the PCB’s featured in this project.

## Why KiCad?

The Rastaban project requires a printed circuit board (PCB) to be used and moved safely from one place to another. Additionally, a PCB is the best option for producing the product in larger quantities. I chose KiCad (6) for several reasons.

First, it is open-source and free, making it accessible to many hobbyists and professionals. Second, it is a good software to learn how to design PCBs. Previously, I used EasyEda, which is proprietary. However, this comes with ads, limited access to certain features, and increased vulnerability to losing or stealing designs. In addition, I was unable to install useful plugins like an interactive BOM file, 3D model archiver, and fabrication toolkits. These limitations led me to stop using EasyEda. However, it should be noted that using EasyEda's library may be the fastest and cheapest way to produce a fully assembled PCB. If you plan to assemble your PCBs yourself, this is no longer an issue. KiCad6 also has plugins that allow you to easily obtain LCSC part numbers for PCB assembly with JLCPCB.

## Plugins

To create an even better PCB and make my workflow easier, I downloaded a few plugins. The plugins I downloaded are as follows:

|  |  |  |
| --- | --- | --- |
| Name | Function | Download location |
| KiCad JLCPCB tools | This plugin allows you to search the JLCPCB parts database, assign LCSC article numbers to your parts, generate production files for JLCPCB and much more. | <https://github.com/Bouni/kicad-jlcpcb-tools> |
| Interactive HTML BOM | This plugin generates convenient BOM listing with ability to visually correlate and easily search for components and their placements on the pcb. | Build into KiCad6 |
| PCB action tools | Annular Ring Checker, Snap Selected Footprint(s) to Grid, Fabrication Footprint Position, Move Selected Drawings to chosen Layer, Export pcb technical layers to DXF, Checking 3D missing models | Build into KiCad6 |
| Archive 3D models | Copies footprint models to the project local subfolder and remaps all the links within the used footprints. | Build into KiCad6 |
| Place Footprints | Arrange sequentially numbered footprints or footprints from multiple hierarchical sheets in linear, circular or matrix arrangement. This plugin works on footprints already present in the layout, so that layout and schematics stay in sync. | Build into KiCad6 |
| Round Tracks | Algorithmically smooth tracks in a predictable manner. Useful for flex PCBs, or just because it looks cool. | Build into KiCad6 |
| Length matching | Track Length Calculator | Build into KiCad6 |
| Freerouting | Auto router for KiCad. It draws all the connections between components for you. Be warned: Auto routing should never be used carelessly, always check the results. | Build into KiCad6 (requires Java) |

## How to learn KICAD6

I learned KICAD6 by following [an online course.](https://techexplorations.com/so/kicad-like-a-pro-3rd-edition/) But I could also have used tutorials on YouTube or read books on it. KICAD itself is not difficult to use, but there are a lot of buttons, some of which are important, some are a bit redundant, which can be confusing. The best way to learn KICAD is to just start creating a schematic with some (maybe 4) parts and connecting them together. Then you can start creating a PCB in the “PCBNEW” section.

The design approach  
Creating a PCB is always following trough 2 stages

1. Create the schematic
2. Create the PCB

After having done this, your design is finished. Of course there are more details to both design stages an those are described in the following pictures.

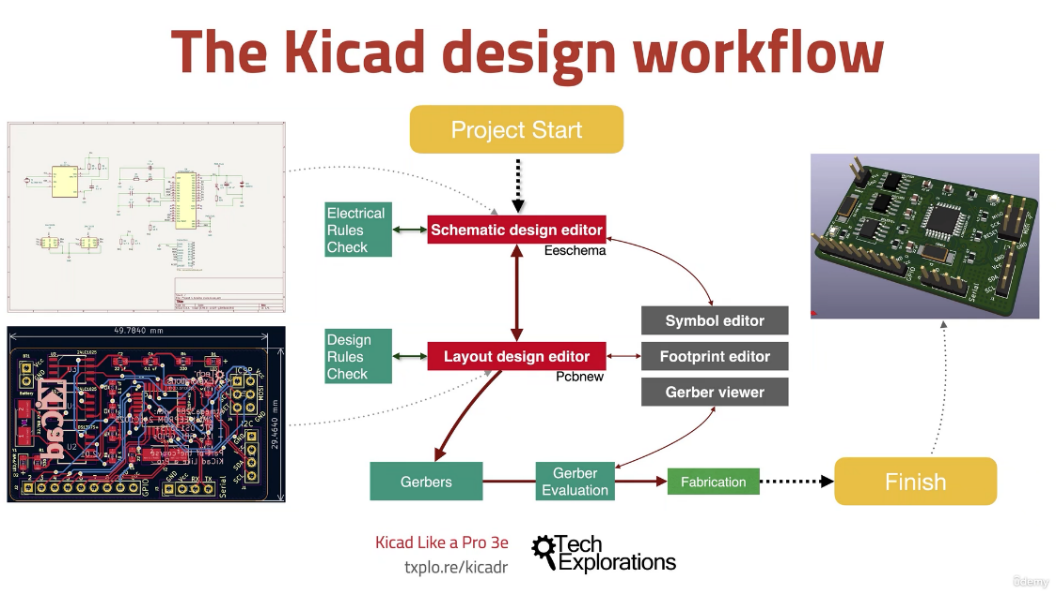


Figure 26 KiCad Design workflow (Kicad like a pro 3th)

# Appendix C: Technical Considerations

To conclude what exact components to use in this prototype I have created an technical considerations table. In this table I weigh the pros against the cons of each components. This way I hope to create an overview of what components there are available and which I should pick.

I later found that there are so many options available sometimes that it makes sense to just choose the available and most popular option. Chances are high that documentation will be correct and support will be higher than with some “off brand” or no name components. For my led driver design it just happened so that the CN5711 was the most available and “tested and known” good chip, so that’s why I chose this one for this prototype. I would suggest choosing no name chips for large production designs.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Technical Considerations** | |  | | |  | |
| **Part** | **(Expected) Function** | | **Pros** | **Cons** | | **Notes** |
| **Raspberry Pi 4** | **Main computer** | | **All in one computer, good documentation, good software support** | **Chip shortage creates vulnerability for supply chain** | | **The compute model 4 is not an easy alternative since there is no build in camera connector. The Raspberry Pi 4 model B seems the most suitable** |
| **TPS61158** | **LED driver** | | **Flexible digital and pwm brightness control, 100:1 pwm dimming ratio, soft start build in** | **Datasheet unclear if there is a switching value of 750 MHz or KHz** | | **Mistakes in datasheet it seems, wrong frequency ratings** |
| **TPS6106x** | **LED driver** | | **Pwm brightness control, digital brightness control, 1mhz fixed switching frequency** | **Made for multiple leds it seems, only 80% efficient** | | **led disconnects during shutdown** |
| **TMC2209** | **Stepper driver** | | **High quality, good documentation** | **Not yet found, maybe over specked** | |  |
| **ST L297** | **Stepper driver** | | **Reputable brand, low cpu usage** | **Expensive** | |  |
| **Tmc2130** | **Stepper driver** | | **The flag ship version of the tmc series** | **Expensive and overkill** | |  |
| **Tmc2208** | **Stepper driver** | | **Just as good as the 2209** | **higher impedance and so lower output amperage than tmc2209. It also has less features than tmc2209** | |  |
| **DRV8870** | **H-bridge** | | **Is able to supply 3.6A of current, enough for the Peltier module most likely** | **Not the best option, a higher current version (6A) would have been ideal: DRV8874** | |  |
| **IRL540SPBF** | **High load switching mosfet (logic level)** | | **A logic level high mosfet that can handle very high currents (20A)** | **A little expensive** | |  |
| **TPS61169** | **LED driver** | | **LED current can be set with resistor** |  | |  |
| **TPS92360** | **LED driver** | | **LED current can be set with resistor** |  | | **Seems the same driver as the tps61169** |
| **P82B96** | **I2C ESD protection IC** | | **Galvanic separation of i2c lines which results in high ESD level protection. One package saves all** | **A little more complicated possibly than using passive components. Expensive.** | |  |

# Appendix D: TMC2209 UART debugging

## UART Problem with multiple drivers

When connection multiple drivers you are able to read/write commands to the drivers individually using the same bidirectional line.

Strangely enough I could not figure out why the above configuration did not work.

Afbeelding met tekst

Automatisch gegenereerde beschrijvingI configured one driver to be 01 and the other 00 by connection the ms1 and ms2 to GND and VCC the way it was shown in figure 4.1 above. Somehow I always get this message: Afbeelding met tafel

Automatisch gegenereerde beschrijving

Figure 27 UART control (Trinamic TMC2209 datasheet)

Figure 28 error output

These are the configurations I tested with:

Table 9 Testing configurations

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | PDN UART | TMC software address settings | Result | Additional notes |
| TMC driver A | Connected to RX | 1 | Functional | Changing one address to 0 also makes it NOT functional |
| TMC driver B | Floating | 1 |  |  |
|  |  |  |  |  |
| TMC driver A | Floating | 0 | Functional |  |
| TMC driver B | Connected to RX | 0 |  |  |
|  |  |  |  |  |
| TMC driver A | Connected to RX | 0 | NOT Functional |  |
| TMC driver B | Floating | 1 |  |  |
|  |  |  |  |  |
| TMC driver A | Floating | 0 | NOT Functional |  |
| TMC driver B | Connected to RX | 1 |  |  |

### First Conclusion/solution

Figure 29 Addressing multiple slave (Trinamic TMC2209 driver datasheet)

The driver UART control only works if there is one driver connected to RX and assigned address to this driver is selected in software (according to the ms1 + ms2 settings you chose). You can however use 2 drivers if you go with the “more than 4 drivers solution” which Trinamic provided. This could be achieved with transistors as well I suppose.

If reading from the driver is not required, using the write only function as seen in figure 4.1 of the datasheet is a viable option. This way we only need to use n+1 pins of the raspberry pi to control the motors, where n increases with every driver added (enable pin). This would treat all steppers as the same and would control them using the enable pin and sending data over the RX pin.

## Further testing UART communication TMC2209

Configuration (test script 05: vactual.py) and Logic Salae software used in combination with logic 8 hardware.

**RX and TX are flipped** **in the pictures:**

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Figure 30 Pin configuration

Afbeelding met toetsenbord

Automatisch gegenereerde beschrijvingThis is what a successful transmission, with a few rotations and direction changes should look like.

Figure 31 successful transmission

Afbeelding met tekst, computer

Automatisch gegenereerde beschrijvingWhen zoomed in it is clear that the TX deviates from the RX line data.

Figure 32 Deviating TX and RX

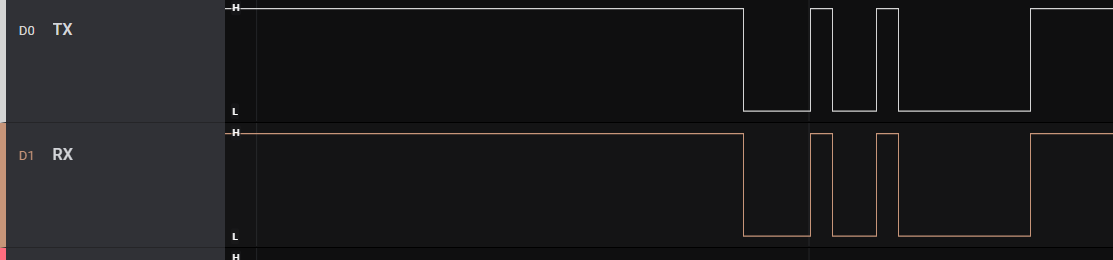
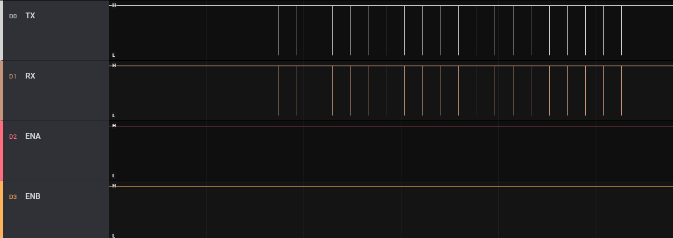
Further inspection shows that it does occasionally match the RX line

Figure 33 RX and TX matching occasionally

It is however always the case that TX continues sending after RX is done with a package.

It is however always the case that TX continues sending after RX is done with a package.

Using these settings (test script 06: multiple drivers.py) **RX and TX are flipped** **in the picture:**

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Figure 34 successful transmission

Figure 35 pin configuration 2

This is what a unsuccessful transmission looks like:

Afbeelding met tekst

Automatisch gegenereerde beschrijvingWhen zoomed in it seems that TX receives exactly the same information as it sends out. While, as stated earlier, it is expected that TX sends more data after RX is finished.

Figure 36 receive and transmit messages

### Second conclusion and solution for the UART error

I can conclude that somehow TX is getting the same information as it sends out on the RX line.

With the help of Thomas Ijsseldijk we found the cause and the fix for this problem. Lowering the TX resistor to 500ohms (from 1k) increases the voltage available for the IO of the TMC driver which results in the TMC being able to read the signal. It appears to be very important to choose the right resistor value. It is now possible to control the stepper motors via UART. This reduces the pins required on the raspberry pi with 7 pins (removing all step and dir pins)

The final configuration for the stepper motor drivers is as following:

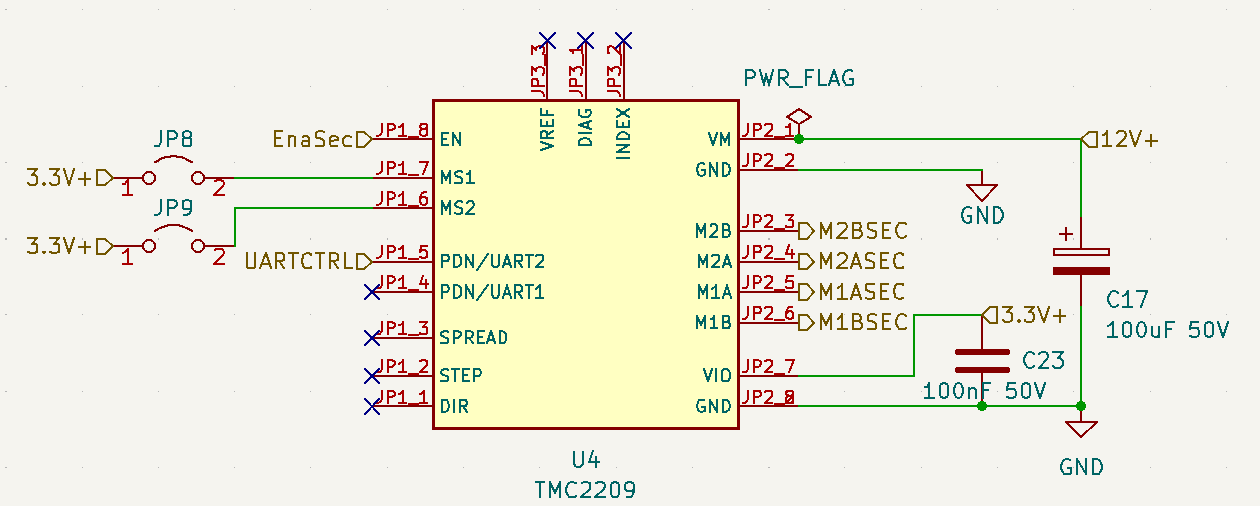


Figure 37 stepper motor schematic

We decided to go with the TMC2209 and we operate it in UART mode only. This means we need to connect to PDN UART and use one enable pin per driver to select it. The jumpers are there to select the driver address (up to 4 drivers in total).

### Final fix and conclusions

After having spend days debugging the hardware; checking continuity, isolating components, swapping resistors values, and rebuilding my original test setup I can conclude that I never had a UART connection to begin with. What exactly happened when the motors started driving is not clear to me. This answer lays in the library.

What I do know is that the TMC2208 driver that we used (the old driver) had a solder jumper that was not soldered. A really strange decision by the manufacturer. When the solder connection was made, UART was functional. That was the only hardware problem relating UART. Now the software was where the real problem was.

We needed to disable the system service that initializes the modem, so it does not connect to the first PL011 (UART0; /dev/ttyAMA0). In the command line:

*$sudo systemctl disable hciuart*

And further we (only!) disabled the login shell on the serial port in the interfacing options of "sudo raspi-config", and rebooted the system.

When we started the UART test script for the tmc2209, we immediately got everything working. We than tested 2 drivers and got the same result:

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Figure 38 successful initialisation

The extra resistors added to UART per driver was also proven to be a myth. 2 drivers can communicate perfectly fine with a 1K resistor only. This means that the previous theory that was found with the help of Thomas, is proven wrong now.

### Motor stutter issue

After having fixed the communication error with UART another problem occurred: the motors would start to stutter, switch direction randomly or stall and vibrate.

I tried debugging this following the following plan:

1. Check the hardware: is everything connected like it should be?
   1. This was the case, everything had power (since UART connections works) and I powered everything via the VM pin and I knew VIO is getting its power via this pin as well.
   2. The motor is connected to the connector and there is continuity
2. Measure signals: is there even a signal coming out of the driver?
   1. Yes, I connected my logic analyser to the primary stepper motor connector and got 4 signals when I send the vactual movement test command.
3. Is the signal that is transmitted to the motor the right one? Maybe the motor receives something it cannot work with.
   1. I had to check this I did this by using step and dir on an external tmc2209 and captured the signal. The motor rotated fine when using my old testsketch and I recorded the signal going to the 4 motor wires. This is the signal I require (or something that at least looks like it)
   2. Then I compared it to the output to 4 motor wires coming from the pcb. This looked very different. Later I figured out it may had something to do with a broken driver. 2 outputs where as expected, but the other 2 where not.
4. Is the motor connector wiring on the PCB to the connector the way I think it is.
   1. I should have checked this first! This was NOT the case! I expected the output from the connector to the driver to be M1A, M1B, M2A, M2B but it was not! The pairs on the connector where not AA,BB but ABAB This means that I connected the motor wires wrong and therefor the motor would not rotate correctly.
5. If one motor works, does the other work too?
   1. No, when connecting a second driver and a stepper motor to the second port and changing the enable pin to the second motor pin, I got an communication error (0,4) I quickly figured that I had to give an extra parameter to tmc object I created in my code: tmc = TMC\_2209(22, driver\_address=1) where I configured the jumpers on my PCB (address selectors) to be 0 for the primary driver, 1 for the secondary driver and 2 for the third driver.

After having concluded that point 5 was the cause and having addressed the issue, I can successfully control 3 motors apart from each other.