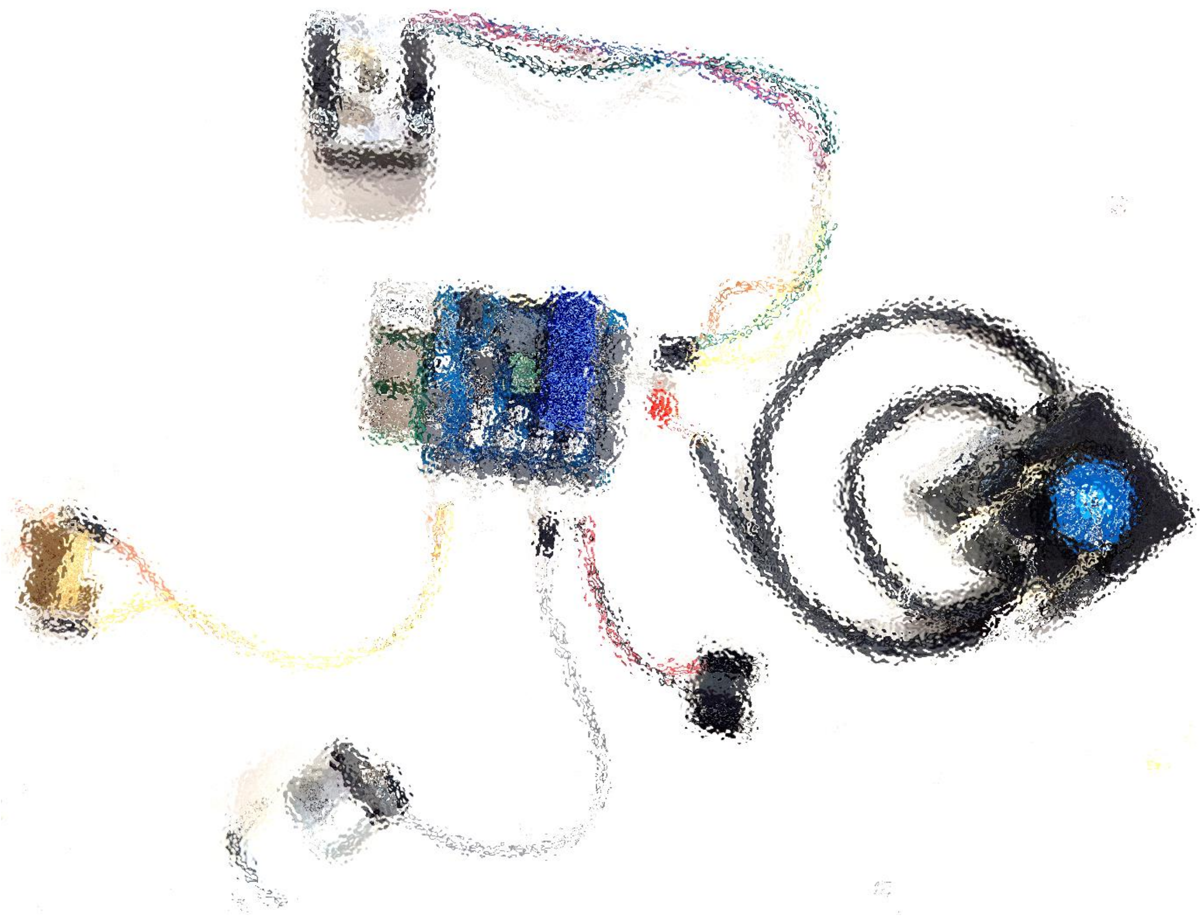
Designing and producing a test platform for fluid monitoring

Redesigning a fluid monitoring setup's hardware and software
for increased functionality, reliability, and durability.



Internship documentation
Embedded Systems Engineering

Casper R. Tak

HAN_UNIVERSITY OF APPLIED SCIENCES

1 General Information

Document
Version: V1.0
Date: 24-01-2023
Word count: 17166
Page count main report: 47
Page count attachments: 21

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 until 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

2 Document log

Table 1 Document history

Version	Date	Edits/notes
1.0	25-01-2023	Finalized main report. Changed cover photo.
0.4	25-01-2023	Incorporated Jeroen's feedback on V0.2 added hyperlinks for some figures and tables. Added word count and page count.
0.3	24-01-2023	Added reflection, added results and overall improvements.
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.

3 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 tested to ensure accurate and reliable measurements by testing both hardware and software components separately.

4 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?

List of contents

List of tables

List of figures

Glossary

Rastaban name of the current prototype that is being developed (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/IDE Industrial product design(er) or Industrial Design Engineer(ing)

ESE Embedded Systems Engineer(ing)

PMS Power management system

ESD Electrostatic discharge

IC Integrated circuit

PWM Pulse width modulation

5 Preface

5.1 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 analyzing 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 analyzing 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.

5.2 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.

6 Background information

6.1 Water quality monitoring background

The ultimate goal of this system is to analyze 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.

6.2 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 an 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.

7 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?
   a. Sub: What do I think is achievable in this timespan?
3. Main: What is water quality?
   a. Sub: how can we measure water quality?
4. Main: What is the current process on this project
   a. Sub: what did previous groups contribute?
   b. Sub: what can be used or build upon in my project?

Answers:

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.
   a. 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)
   a. 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.
   a. 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.
   b. 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)

8 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 videos 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.

9 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 guideline 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:

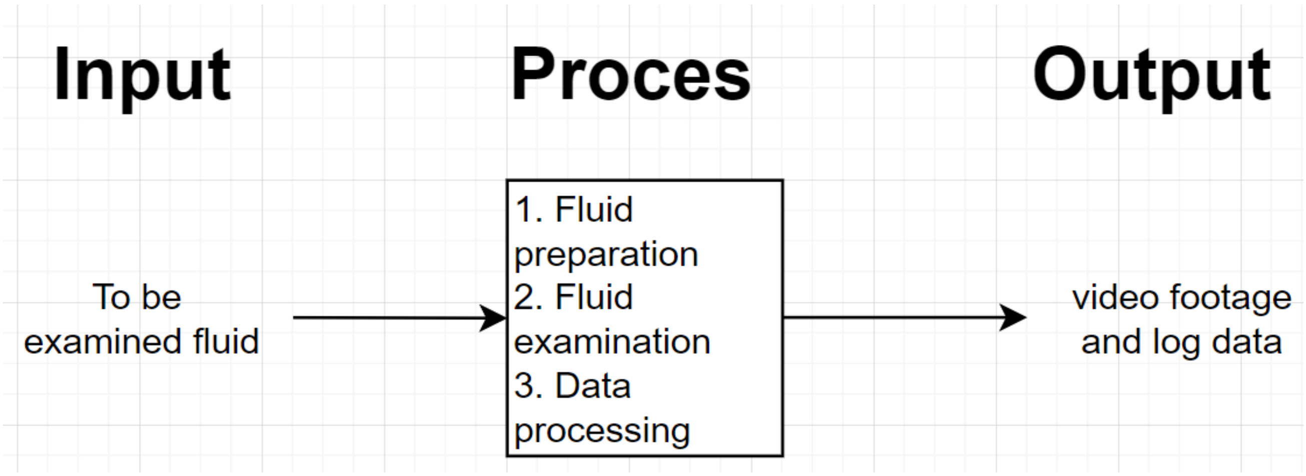


Figure 1 Input output diagram

9.1 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.

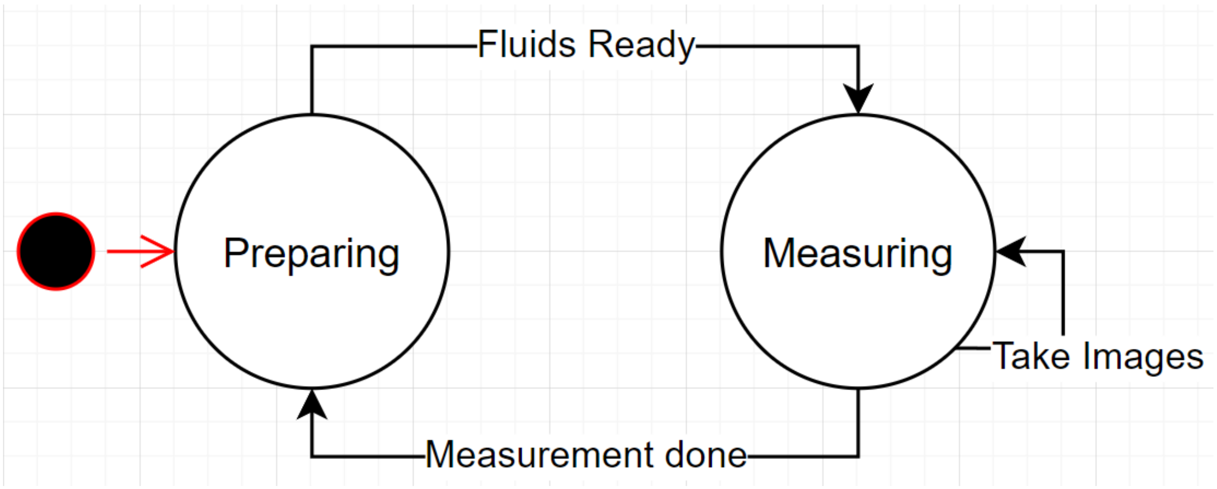


Figure 2 State machine

9.2 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 come in contact with a server solution.

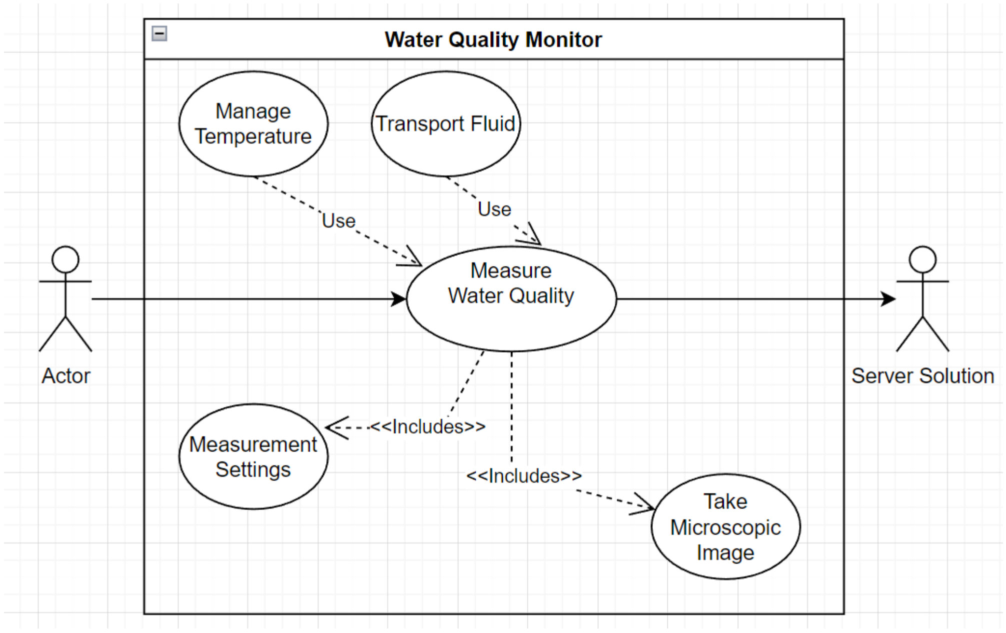


Figure 3 Use Case Diagram

9.3 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.

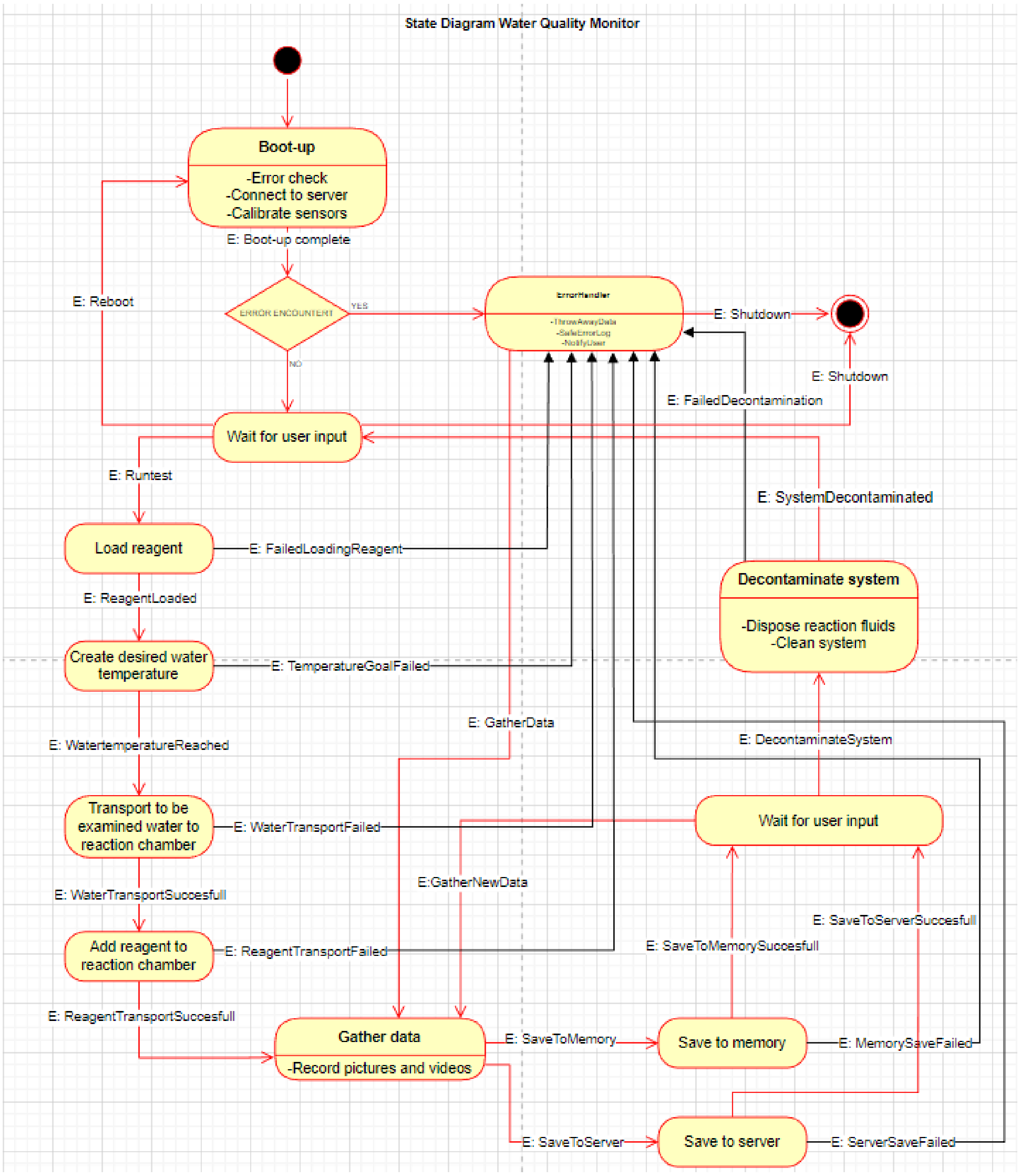


Figure 4 State diagram

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 Its 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

10 Technical design

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

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

10.1 System architecture

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

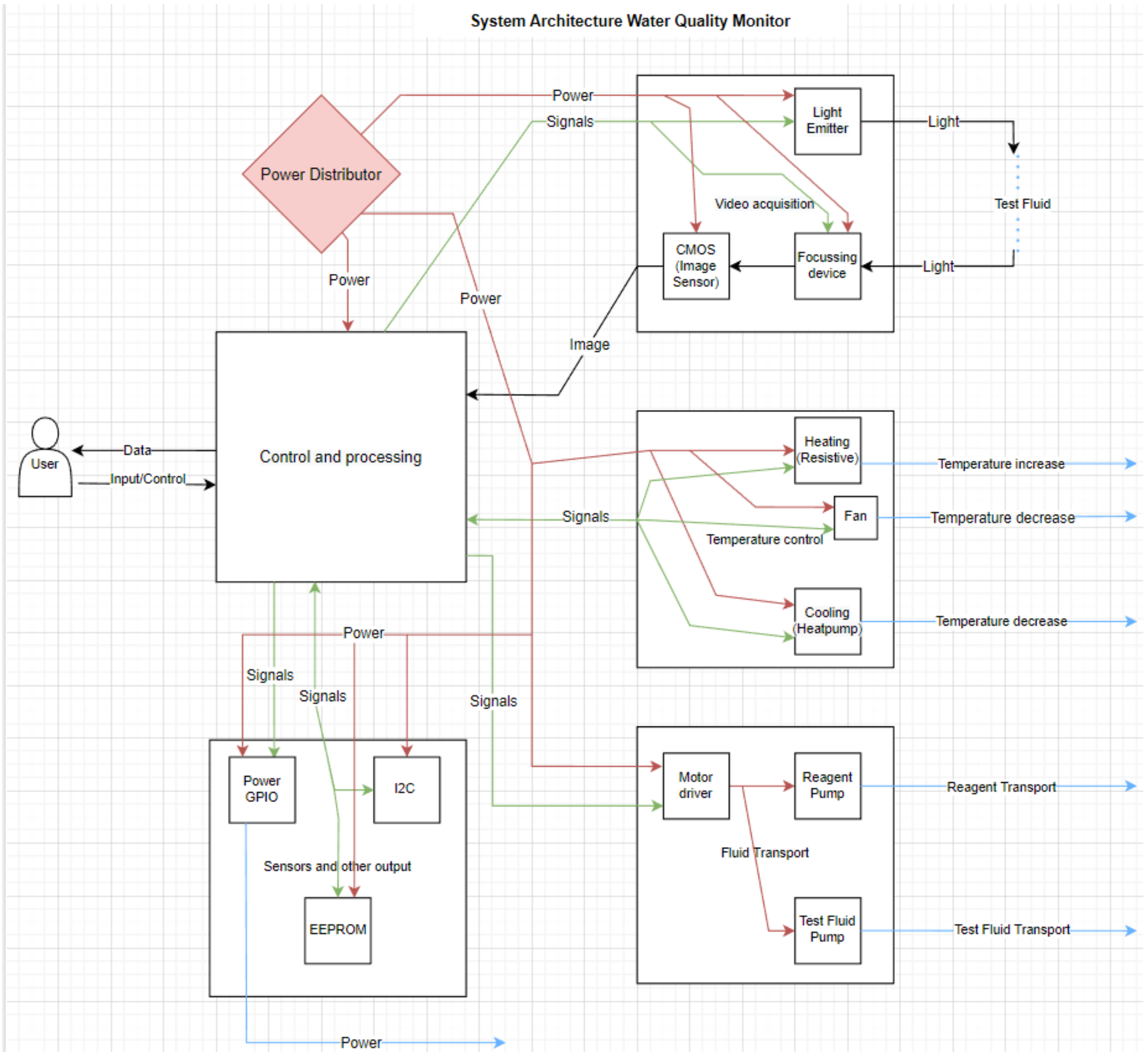


Figure 5 System Architecture

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 (voice coil)
T3.1.2	S	When using magnets (voice coil), 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 (nonproprietary)
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

Table 3 Technical specifications

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

11 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.

11.1 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.

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.

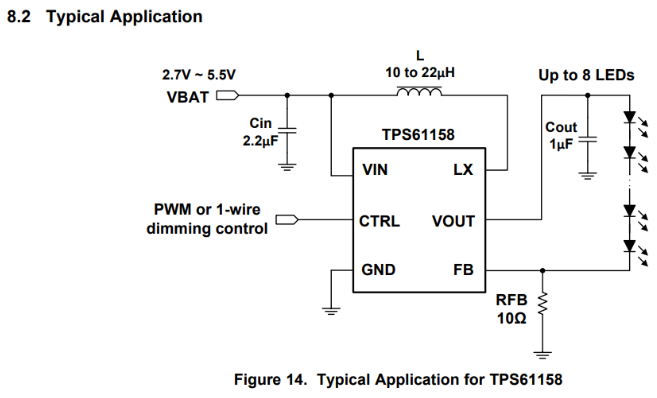


Figure 6 Typical application TPS61158 driver

It is recommended to take advice from the pcb layout recommendations to avoid unsuspected behavior 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.

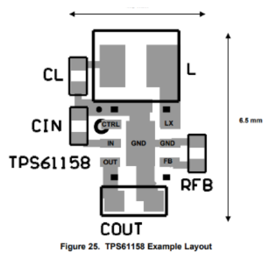


Figure 7 pcb recommendations

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.

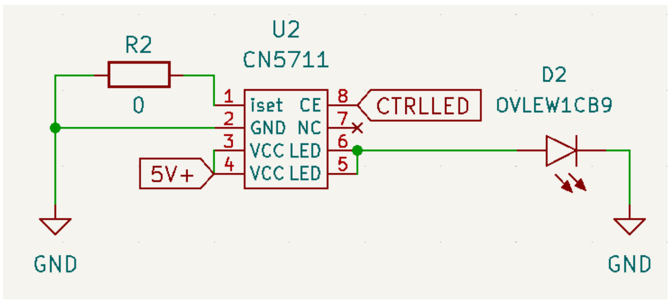


Figure 8 finalized led driver circuit.

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.

11.1.1 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.

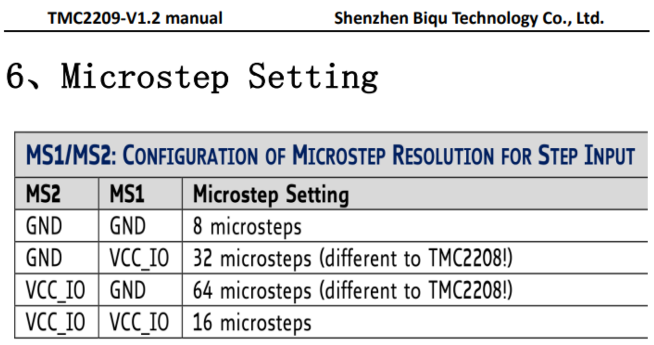
11.2 Stepper motor drivers

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


TMC2209-V1.2 manual Shenzhen Biqu Technology Co., Ltd.

6、Microstep Setting

MS1/MS2: CONFIGURATION OF MICROSTEP RESOLUTION FOR STEP INPUT

MS2	MS1	Microstep Setting
GND	GND	8 microsteps
GND	VCC_IO	32 microsteps (different to TMC2208!)
VCC_IO	GND	64 microsteps (different to TMC2208!)
VCC_IO	VCC_IO	16 microsteps

Figure 9 Micro stepping settings table

11.2.1 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.

11.3 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.

11.3.1 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.

11.3.2 Schematics

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.

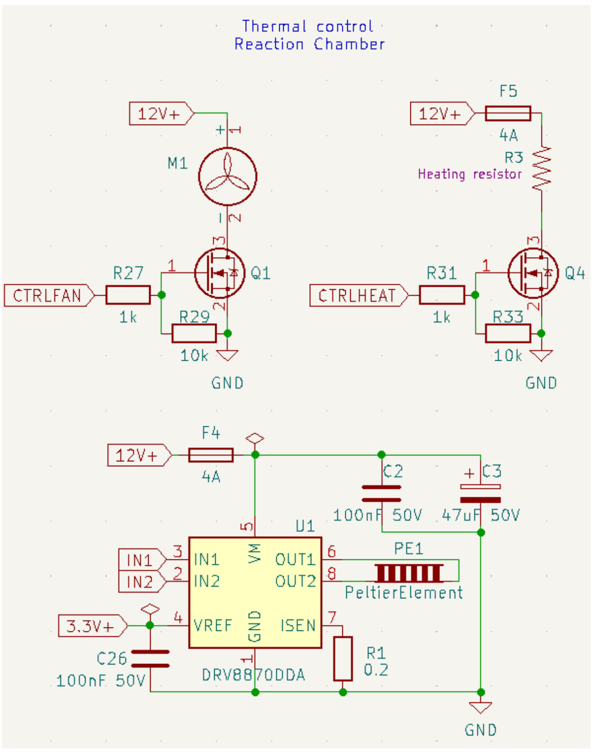


Figure 10 Thermal control components circuit

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.

11.4 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.

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.

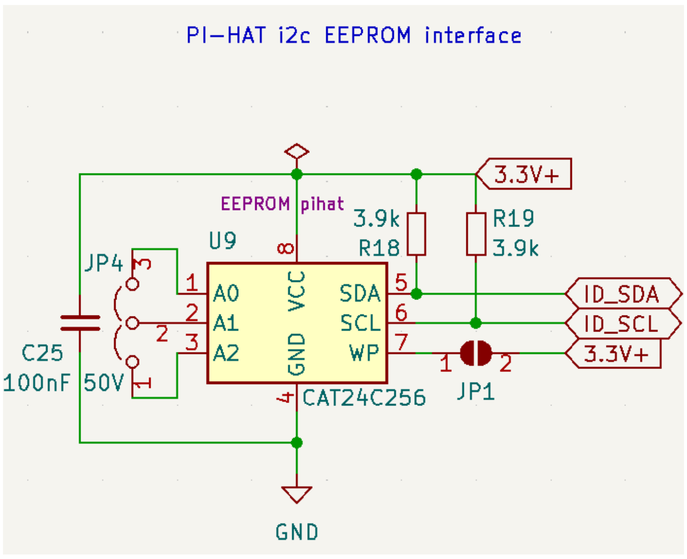


Figure 11 optional EEPROM circuit

11.5 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.

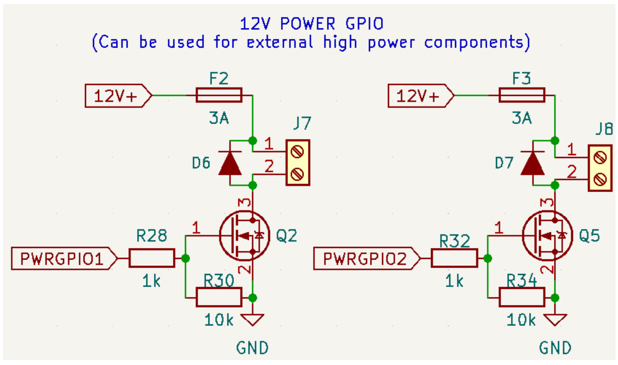


Figure 12 PowerGPIO circuit

This prevents a floating pin and therefor unpredictable behavior.

11.6 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:

Component	Quantity	Max Voltage (V)	Max Current (A)	Max Power (W)	Notes
Raspberry Pi 4B	1	5.2	3	15.6	
Peltier Module	1	12	4	48	
Microscope LED	1	3.2	0.05	0.16	
Focussing lens motor driver	3	12	2	24	Unknown Stepper motor
Stepper motor driver	3	12	2	24	
TOTALS:	9	44.4	11.05	111.76	

Table 4 Estimated maximum power usage

11.6.1 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.

11.6.2 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.

11.6.2.1 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 ics 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.

11.6.3 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.

11.6.4 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.

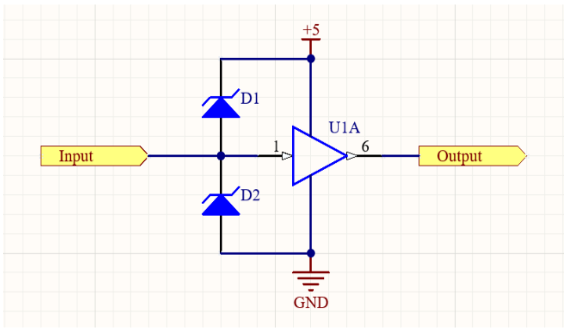


Figure 13 ESD protection using TVS diodes

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

11.6.5 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.

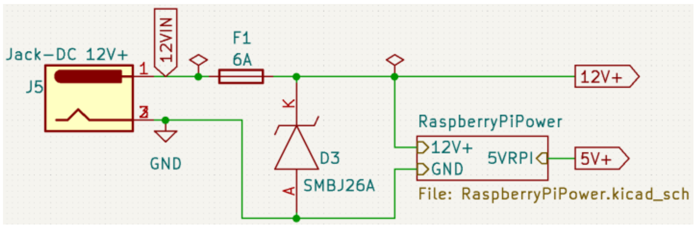


Figure 14 Power Management System circuit (simplified)

Other components that also run on 5.0V, such as the microscope LED and the voice coil 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).

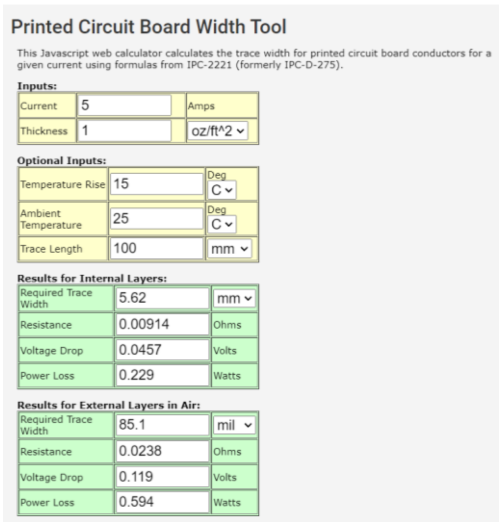
11.6.6 12V vs 24V

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.

11.6.7 Trace width

Since 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.

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.



Printed Circuit Board Width Tool

This Javascript web calculator calculates the trace width for printed circuit board conductors for a given current using formulas from IPC-2221 (formerly IPC-D-275).

Inputs:

Current	5	Amps
Thickness	1	oz/ft^2

Optional Inputs:

Temperature Rise	15	Deg C
Ambient Temperature	25	Deg C
Trace Length	100	mm

Results for Internal Layers:

Required Trace Width	5.62	mm
Resistance	0.00914	Ohms
Voltage Drop	0.0457	Volts
Power Loss	0.229	Watts

Results for External Layers in Air:

Required Trace Width	85.1	mil
Resistance	0.0238	Ohms
Voltage Drop	0.119	Volts
Power Loss	0.594	Watts

Figure 15 Trace Width Calculator tool

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.

11.6.8 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.

11.6.9 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.

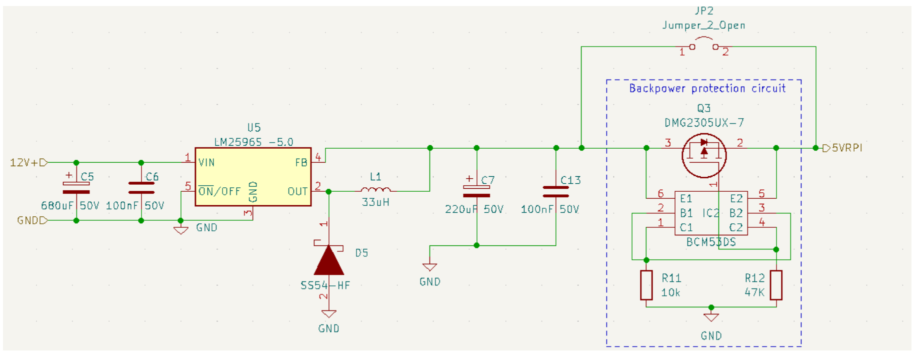


Figure 16 5V power circuit

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.

11.7 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.

11.8 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 behavior 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.

11.8.1 PCB V0.1

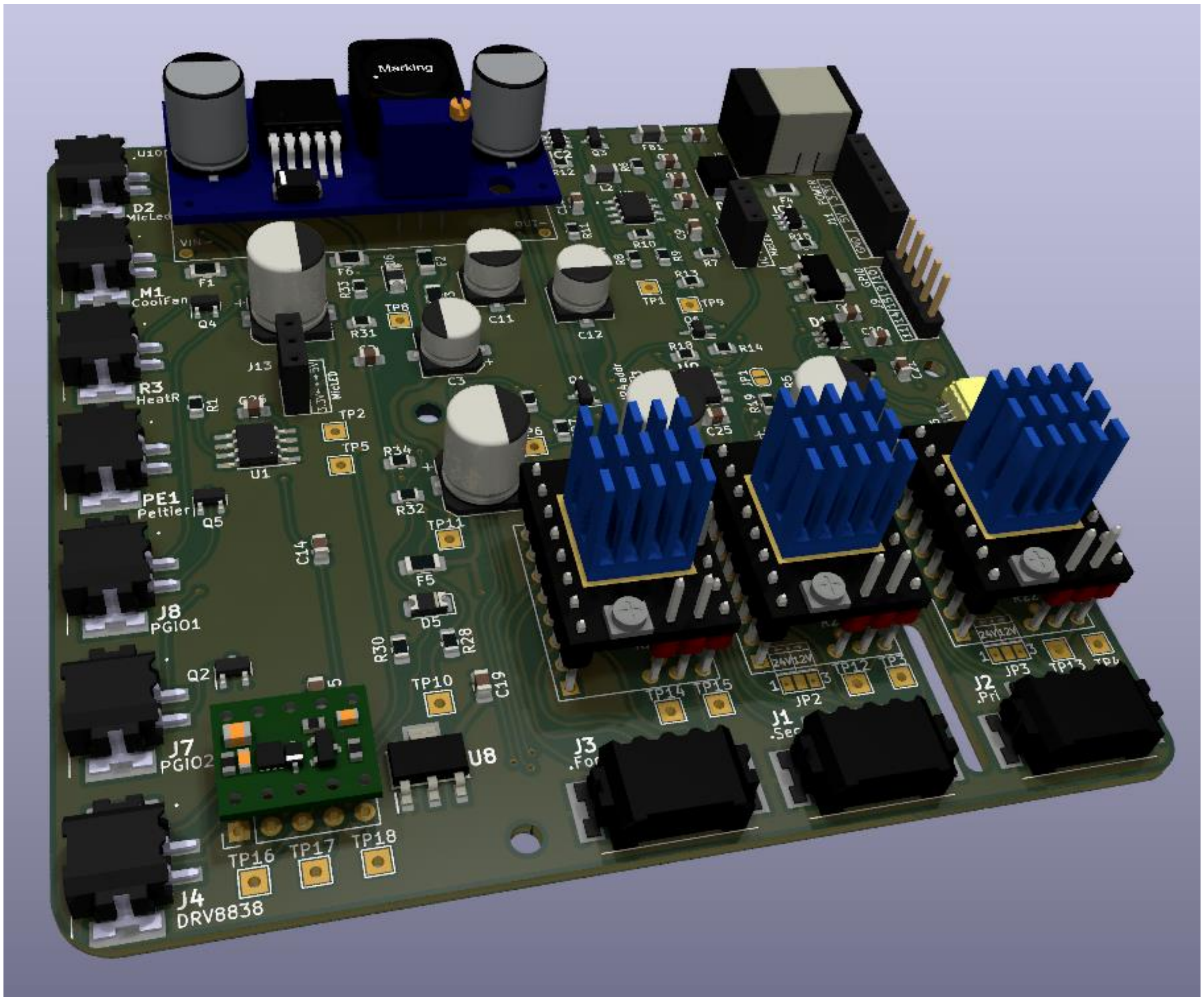


Figure 17 PCB V0.1

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:

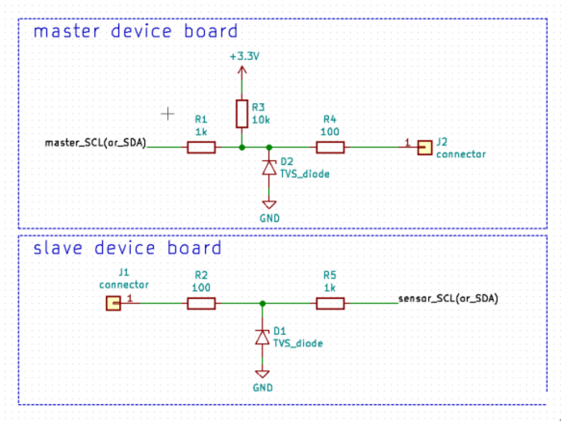
- Most of the functions/ elements work
- This layout seems to be a good starting point, maybe some small refinements.
- The mounting holes and dimensions are all correct.

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 build in esd for gpio. Raspberry pi is connected to ground via power connector. Leave ESD footprints, but do not 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. Do not 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 mosfet 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 evaluated (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 evaluated. 3.3V from u2 (tlv1117-33) works fine.
ESD protection not evaluated.

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?)
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:  master device board +3.3V R3 10k R1 1k R4 100 master_SCL(or_SDA) 1 J2 connector D2 TVS_diode GND slave device board J1 connector 1 R2 100 R5 1k sensor_SCL(or_SDA) D1 TVS_diode GND *Figure 18 ESD TVS diode option*

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:

- Do not 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 double check the most important things (footprints).
- Do some life testing so create mockups for PCBs 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.

11.8.1.1 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 stepper motors	0.5 (3 motors)
Stepper motor (large)	Primary stepper motor	1.7
Stepper motor (medium)	Secondary stepper motor	1.5

11.8.2 PCB V0.2

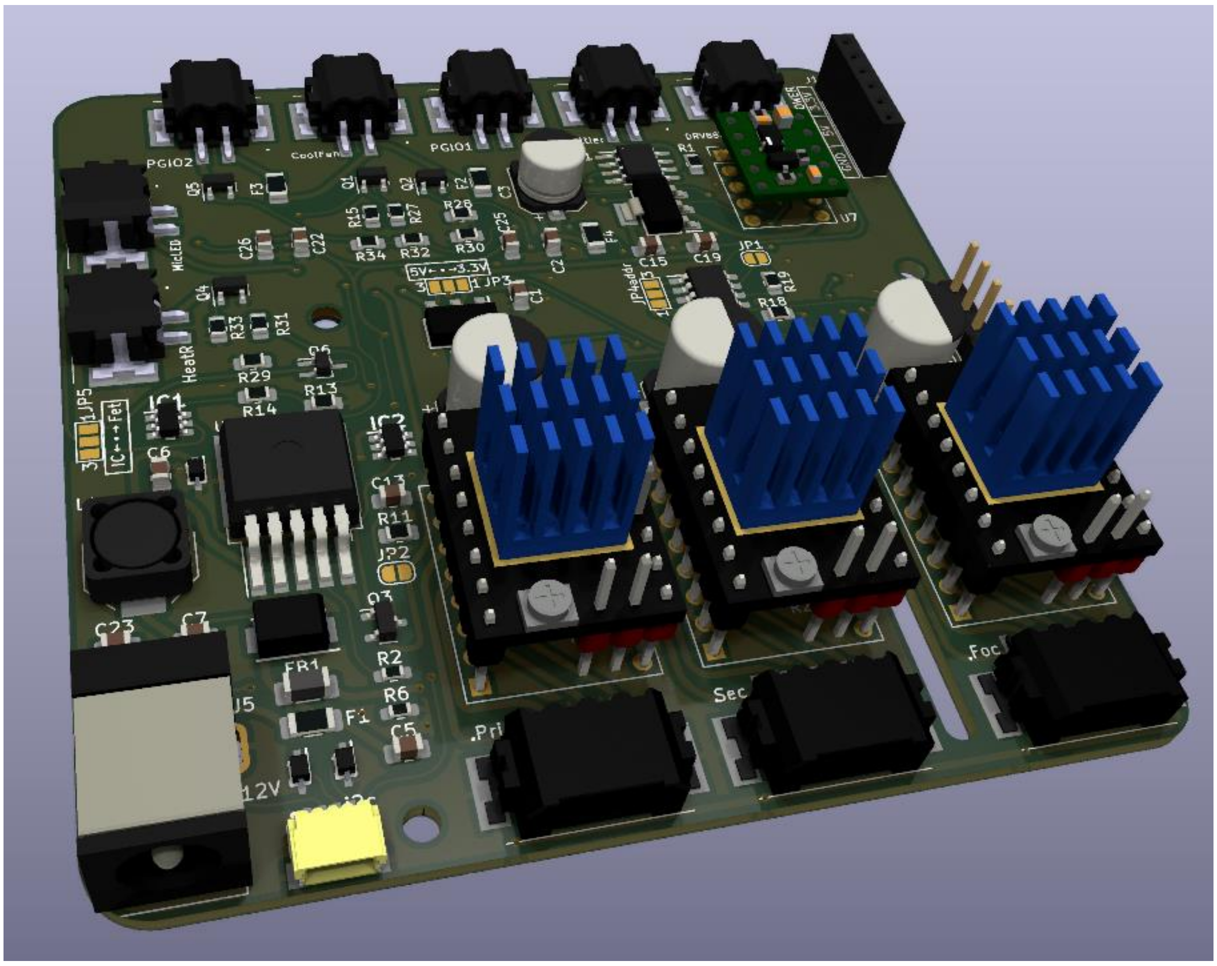


Figure 19 PCB V0.2

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/focus motor (drv8838)	NO
2	TMC 2209 uses different ms1 ms2 configuration for micro stepping than the tmc 2208! Keep this in mind.	DONE, modern 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

11.8.3 PCB V0.3

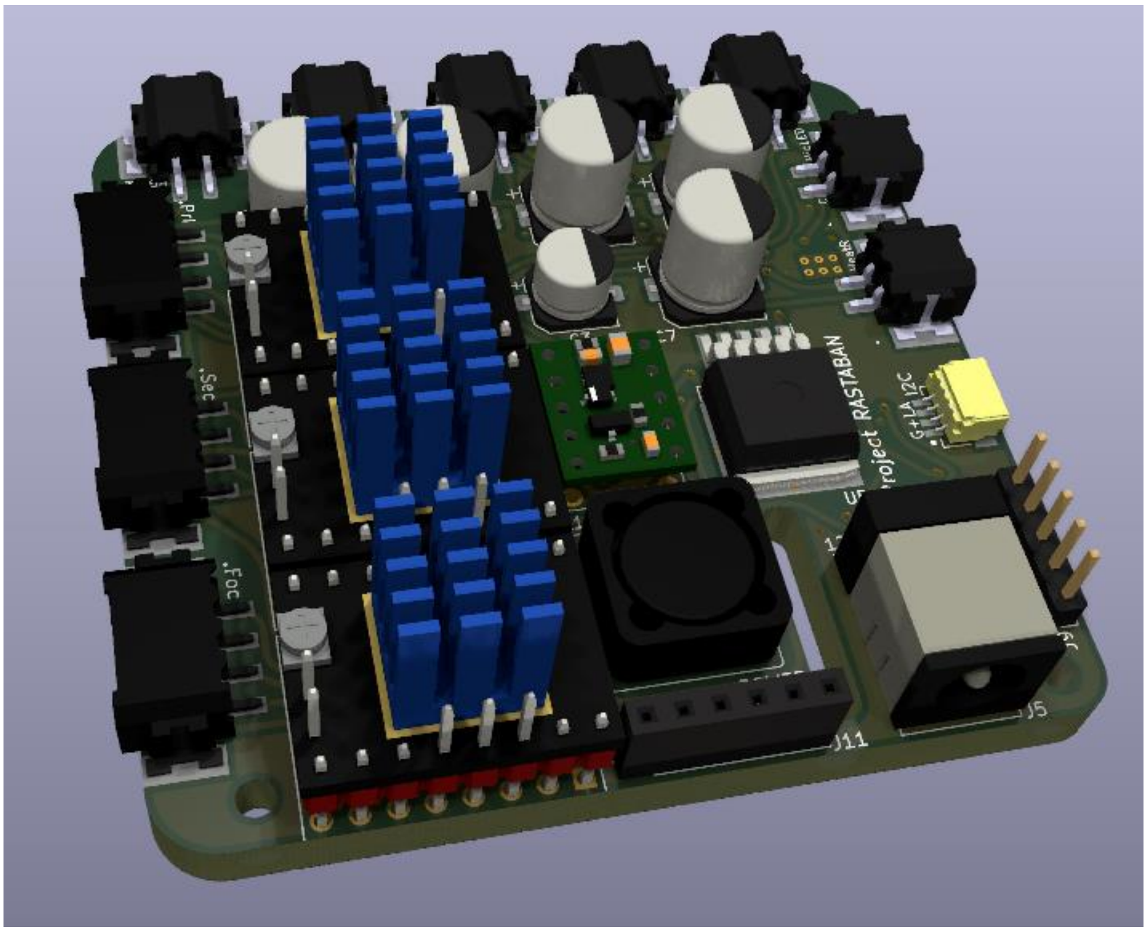


Figure 20 PCB V0.3

#	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	

Successes:

- All functions/elements are still working.
- This version is smaller than its 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

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.

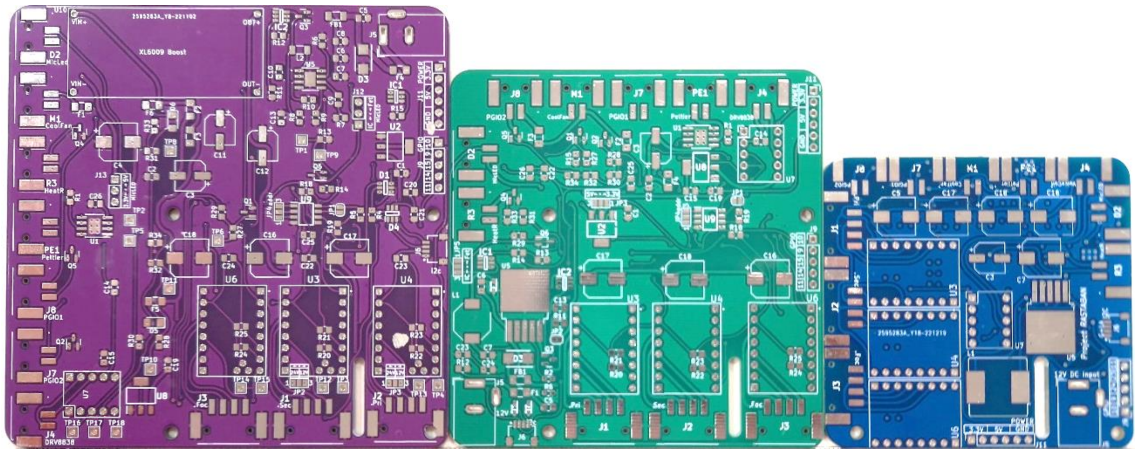


Figure 21 PCB V0.1-V0.3

11.9 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.

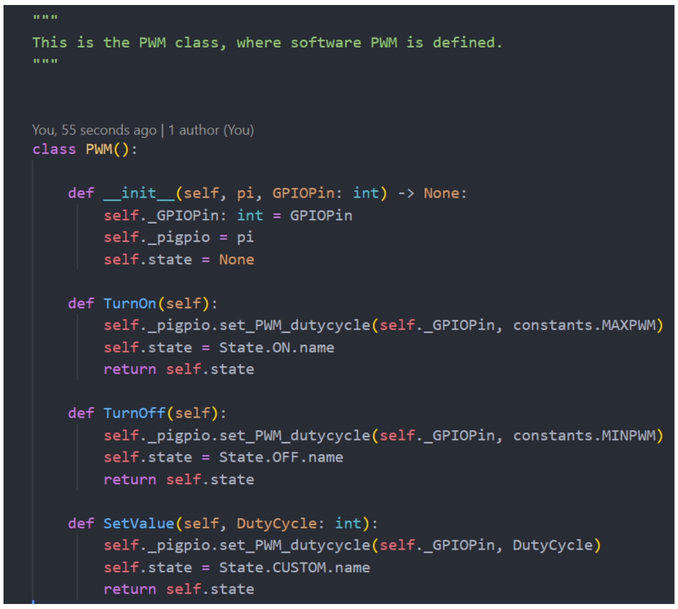
12 Software

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 see Figure 22 Example of object oriented written code (PWM) for reference.

12.1 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
- Voice coil
- PIGPIO
- Init (PIGPIO)



```
"""
This is the PWM class, where software PWM is defined.
"""


You, 55 seconds ago | 1 author (You)
class PWM():

    def __init__(self, pi, GPIOPin: int) -> None:
        self._GPIOPin: int = GPIOPin
        self._pigpio = pi
        self.state = None

    def TurnOn(self):
        self._pigpio.set_PWM_dutycycle(self._GPIOPin, constants.MAXPWM)
        self.state = State.ON.name
        return self.state

    def TurnOff(self):
        self._pigpio.set_PWM_dutycycle(self._GPIOPin, constants.MINPWM)
        self.state = State.OFF.name
        return self.state

    def SetValue(self, DutyCycle: int):
        self._pigpio.set_PWM_dutycycle(self._GPIOPin, DutyCycle)
        self.state = State.CUSTOM.name
        return self.state
```

Figure 22 Example of object oriented written code (PWM)

12.1.1 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.

12.1.2 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

12.1.3 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.

12.1.4 Voicecoil

The Voice coil code controls the voice coil for focusing the lens. The code was designed for the DRV8838 motor driver.

12.1.5 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.

12.1.6 Init function (PIGPIO daemon)

I created an init function for the Raspberry Pi to address unexpected behavior 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.

12.2 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.

12.3 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.

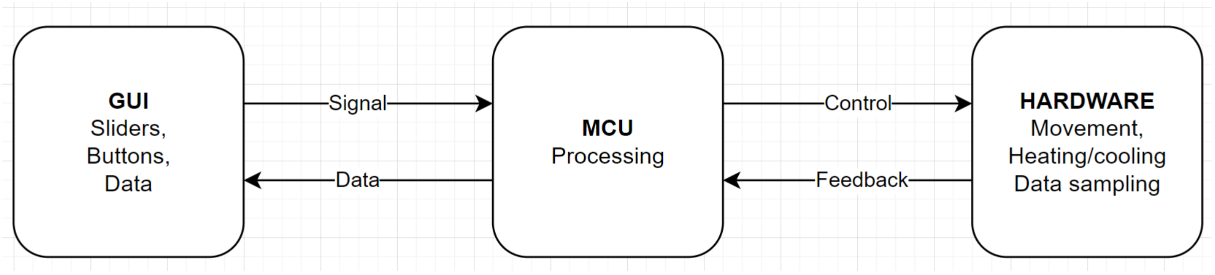


Figure 23 Application of GUI in the system

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

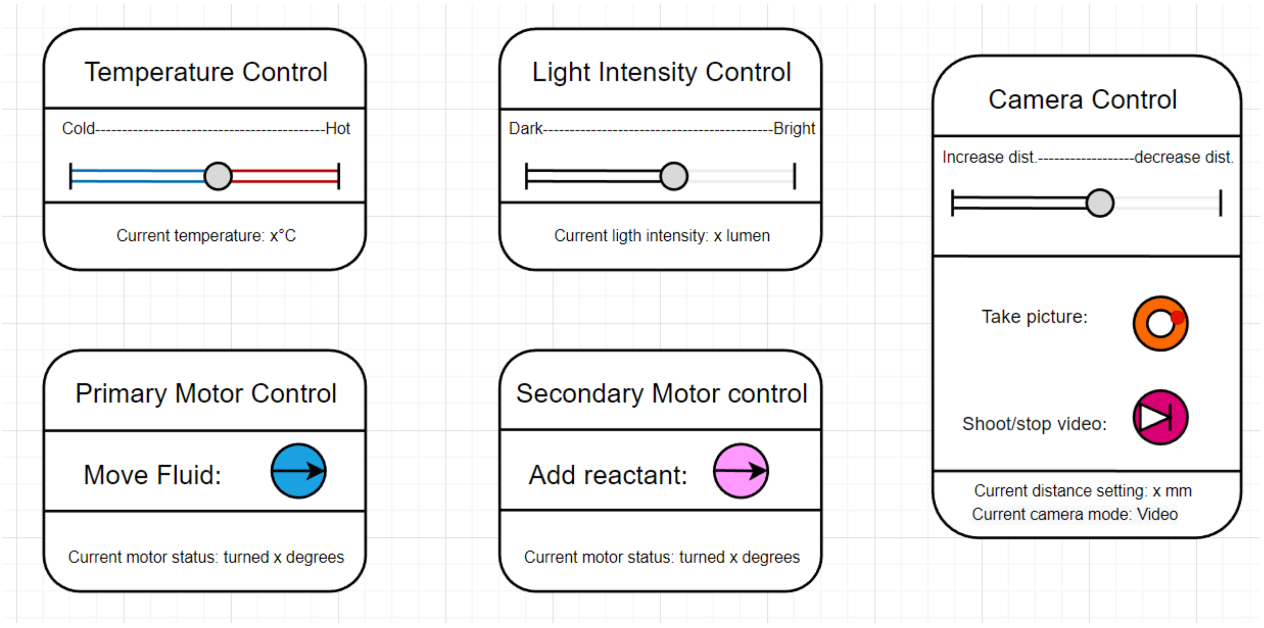


Figure 24 Windows sketch GUI

12.3.1 GUI design

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.

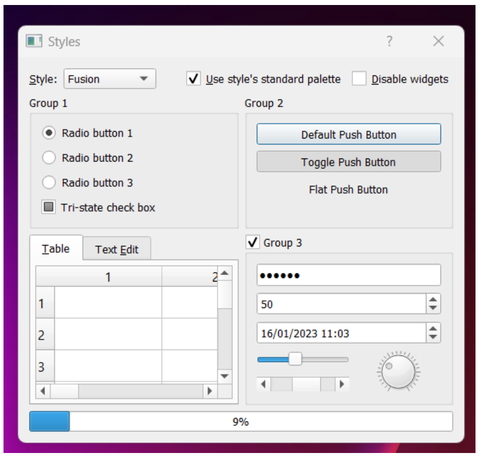


Figure 25 GUI widget options in one window

12.3.2 Final design sketch

Ultimately 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.

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.

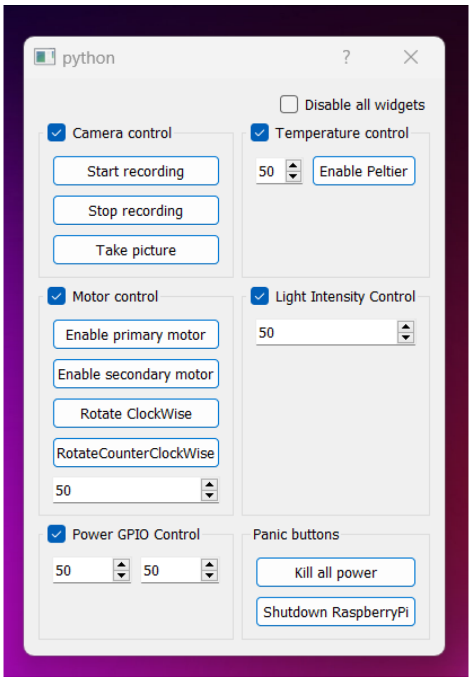


Figure 26 Final design sketch GUI

13 Results

Having spend hours working on this project working on both refining the hardware and software of the Rastaban prototype. I have successfully created the first fully functional prototype.

13.1 Hardware

The Rastaban HAT prototype has the following functionality (in short)

- Can control 3 stepper motors.
- Has two power GPIO pins (providing 12V 6A each)
- Can control an LED using a constant current IC.
- Can control the voice coil using the drv8838 driver.
- Can control a 12V 3.6A Peltier module using the DRV8870DDA H-bridge.
- Can control power to a heating resistor.
- Can control a fan for removing heat from the Peltier module heatsink.
- Has optional EEPROM ID configuration on PCB.
- Has a backpower protection circuit, fuses and ESD + EMC precautions.
- Has GPIO breakout pins for external control.
- Has power breakout pins for external power.
- It's HAT formfactor has been achieved.

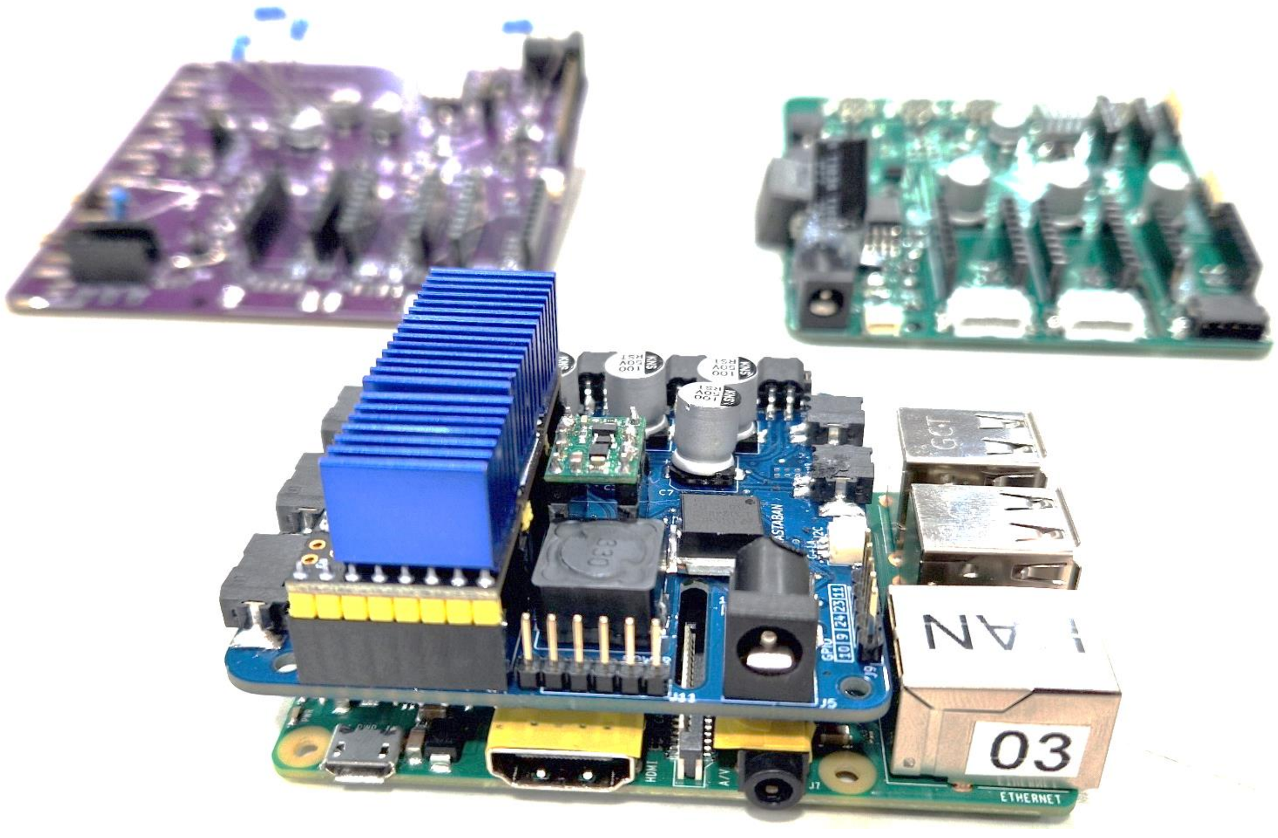


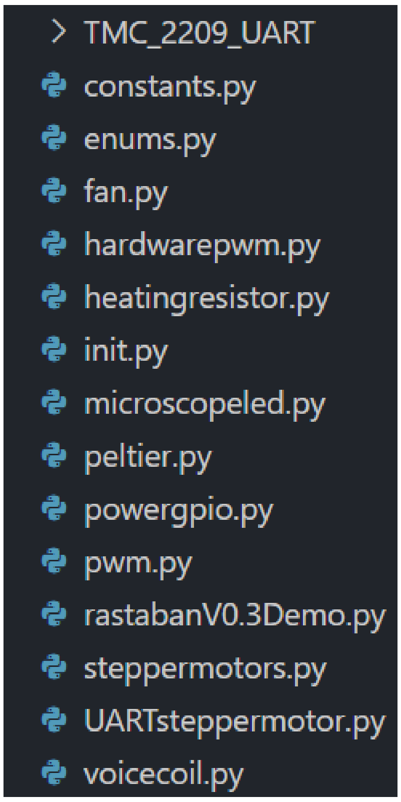
Figure 27 All PCB's and final result on foreground

13.2 Software

The software control the Rastaban HAT is written in python and in the object oriented style. With the software you have full control of the outputs of the Rastaban HAT. V0.3 uses UART to control its stepper motors but for legacy control of V0.1 and V0.2 steppermotors.py is still present.

In short, the software functions:

- Fan: fan control using PWM.
- Heatingresistor: control power to resistor using PWM.
- Init: code to initialize and see current process id of the PIGPIO daemon.
- Microscopeled: control the microscope LED using hardware PWM.
- Peltier: control the H-bridge that controls power to the Peltier element.
- Powergpio: control power output on powergpio using PWM.
- Pwm: the pwm class.
- RastabanV0.3Demo: this code has not been realized yet, but could be used for demonstration purposes. All code has a build in test function (unit test).
- Steppermotors: legacy stepper motor control code using step/dir and enable pins. This is for backwards compatibility with older PCB versions.
- UARTsteppermotor: the new stepper motor control method that is used in V0.3 of the Rastaban PCB. This code gives is a encapsulated version of the TMC_2209_UART code. It is created to make testing easier in some situations.
- Voice coil: controls the voice coil by controlling the DRV8838 driver.



```
> TMC_2209_UART
constants.py
enums.py
fan.py
hardwarepwm.py
heatingresistor.py
init.py
microscopeled.py
peltier.py
powergpio.py
pwm.py
rastabanV0.3Demo.py
steppermotors.py
UARTsteppermotor.py
voicecoil.py
```

Figure 28 All scripts and supporting code for the Rastaban HAT

Note: Figure 24 Windows sketch GUI that I created can be used as an starting point to connect this code to.

14 Conclusion and recommendations

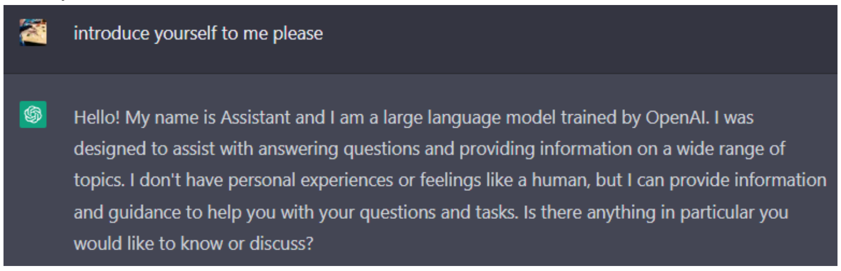
The hardware and software that I created is suitable for most of the WQM purposes and for the DNA research in the future. However there are many improvements that could be made, especially when it comes to the software. The GUI for example could be extended on and integrated into the control code of the Rastaban. There is also no fault detection in the software. Sometimes this is due to hardware limitations (no flags present), but often there could be error codes like: “pin not found, is everything connected correctly?” for example.

The hardware is functional and the motor driver modules are easily replaceable if something brakes. However it would be far cheaper to place the components directly on the PCB. There should be flyback protection added in that case (for accidental motor BackEMF). By integrating everything on the main PCB, production time and costs would be greatly reduced. It is also not always certain which driver module version you receive, which could sometimes lead to unpredictable behavior. Another improvement could be to place the connectors vertically, but this depends on the preference of the user. For more notes see Table 8 V0.4 notes.

15 Appendix A: ChatGPT

15.1 ChatGPT usage in this report

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

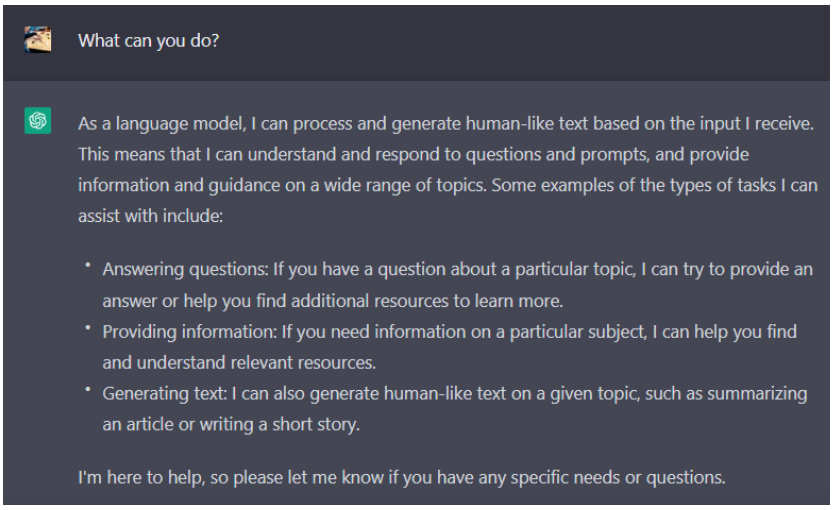


introduce yourself to me please

Hello! My name is Assistant and I am a large language model trained by OpenAI. I was designed to assist with answering questions and providing information on a wide range of topics. I don't have personal experiences or feelings like a human, but I can provide information and guidance to help you with your questions and tasks. Is there anything in particular you would like to know or discuss?

Figure 29 ChatGPT introduction

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.



What can you do?

As a language model, I can process and generate human-like text based on the input I receive. This means that I can understand and respond to questions and prompts, and provide information and guidance on a wide range of topics. Some examples of the types of tasks I can assist with include:

- Answering questions: If you have a question about a particular topic, I can try to provide an answer or help you find additional resources to learn more.
- Providing information: If you need information on a particular subject, I can help you find and understand relevant resources.
- Generating text: I can also generate human-like text on a given topic, such as summarizing an article or writing a short story.

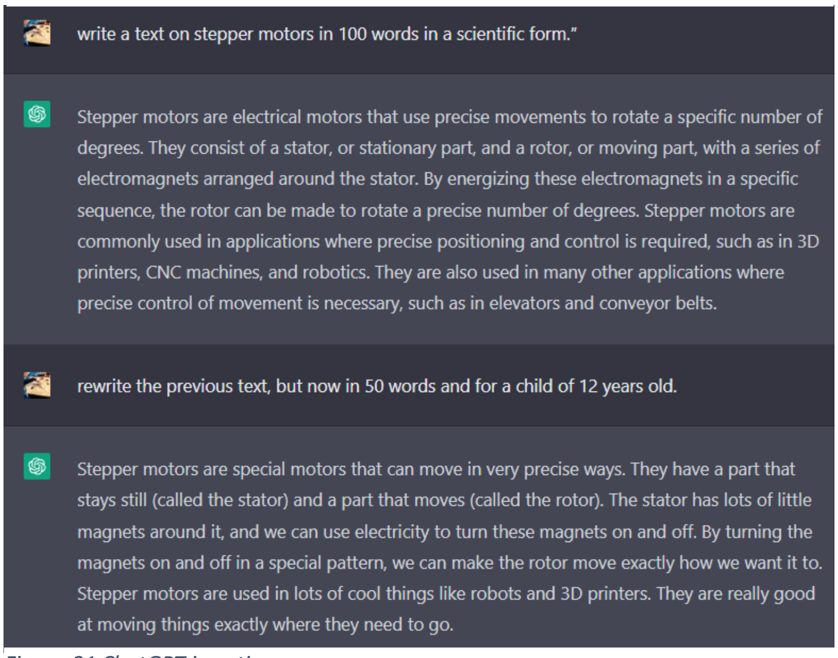
I'm here to help, so please let me know if you have any specific needs or questions.

Figure 30 ChatGPT functionality

I 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.

I 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.



write a text on stepper motors in 100 words in a scientific form.”

Stepper motors are electrical motors that use precise movements to rotate a specific number of degrees. They consist of a stator, or stationary part, and a rotor, or moving part, with a series of electromagnets arranged around the stator. By energizing these electromagnets in a specific sequence, the rotor can be made to rotate a precise number of degrees. Stepper motors are commonly used in applications where precise positioning and control is required, such as in 3D printers, CNC machines, and robotics. They are also used in many other applications where precise control of movement is necessary, such as in elevators and conveyor belts.

rewrite the previous text, but now in 50 words and for a child of 12 years old.

Stepper motors are special motors that can move in very precise ways. They have a part that stays still (called the stator) and a part that moves (called the rotor). The stator has lots of little magnets around it, and we can use electricity to turn these magnets on and off. By turning the magnets on and off in a special pattern, we can make the rotor move exactly how we want it to. Stepper motors are used in lots of cool things like robots and 3D printers. They are really good at moving things exactly where they need to go.

Figure 31 ChatGPT in action

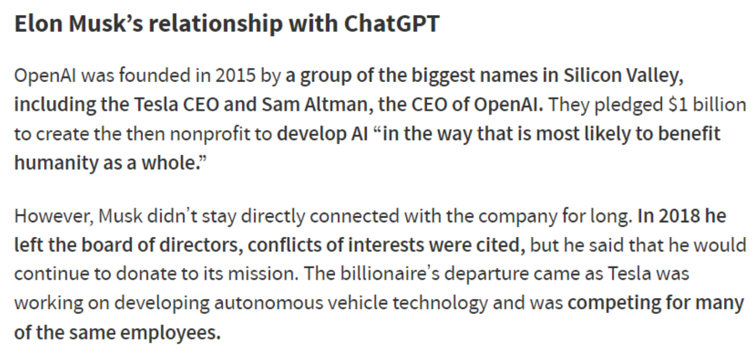
15.2 Review of using ChatGPT

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.

15.3 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." Figure 32 (as.com, 2001)



Elon Musk's relationship with ChatGPT

OpenAI was founded in 2015 by **a group of the biggest names in Silicon Valley, including the Tesla CEO and Sam Altman, the CEO of OpenAI.** They pledged $1 billion to create the then nonprofit to **develop AI "in the way that is most likely to benefit humanity as a whole."**

However, Musk didn't stay directly connected with the company for long. **In 2018 he left the board of directors, conflicts of interests were cited,** but he said that he would continue to donate to its mission. The billionaire's departure came as Tesla was working on developing autonomous vehicle technology and was **competing for many of the same employees.**

Figure 32 Elon Musk relation with ChatGPT

16 Appendix B: KiCad

16.1 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.

16.2 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.

16.3 Plugins

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

Table 9 KiCad Plugins

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	Build into KiCad6

	and remaps all the links within the used footprints.	
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)

16.4 How to learn KICAD6

I learned KICAD6 by following an online course. 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.

16.5 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.

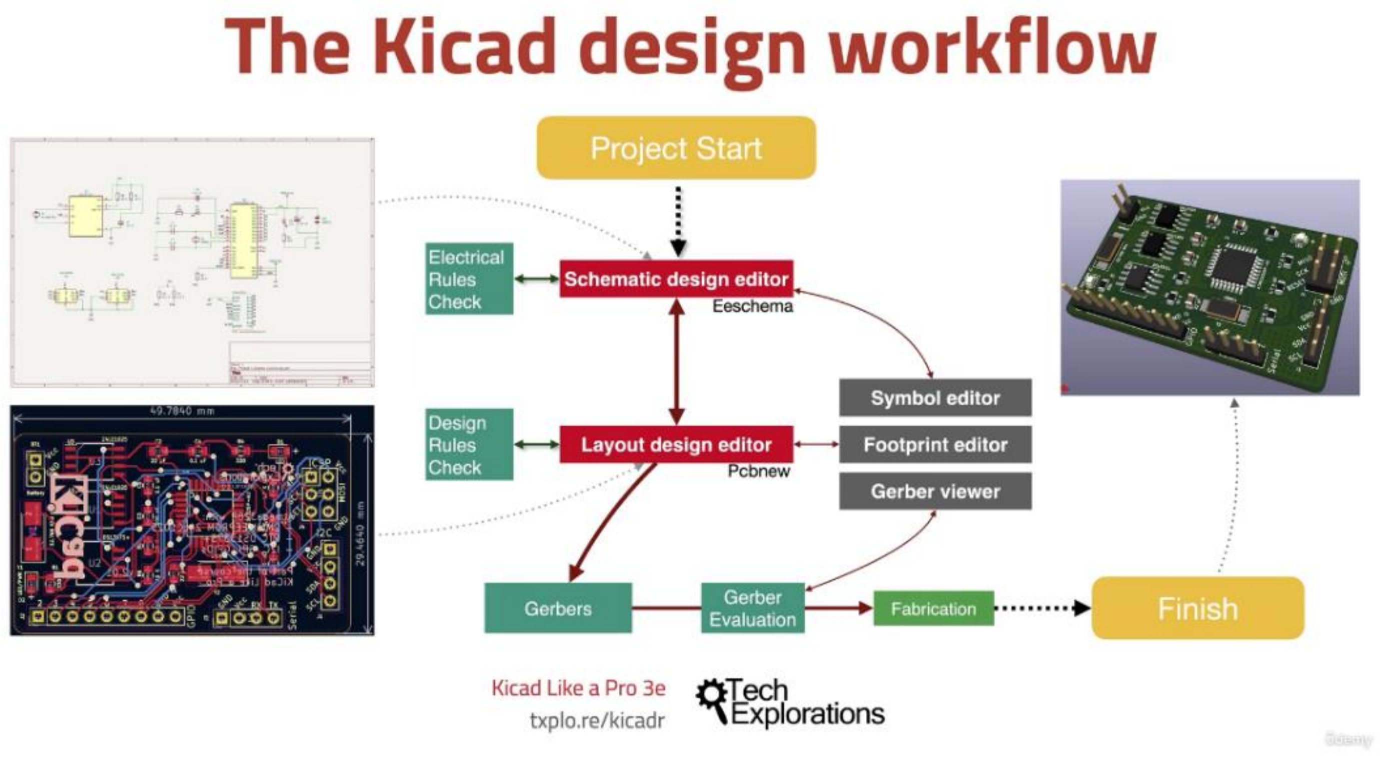


Figure 33 KiCad Design workflow (KiCad like a pro 3th edition)

17 Appendix C: Technical Considerations

To conclude what exact components to use in this prototype I have created a 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.

18 Appendix D: Reflection

18.1 Learning achievements internship

In my internship I have come in contact with many new experiences. I had to design hardware, software and had to carry this out alone, although I received guidance of Jeroen Veen. I also got introduced to new programming languages: Python and PYQT. This is a write down of the subjects that I learned or got introduced to:

- Python programming language
- PYQT programming language
- Draw.io diagram/drawing modelling software
- Autodesk Fusion 360
- Linux (Raspbian)
- PCB design in Kicad6
- 3D printing with PLA
- 3D printing with SLA
- Food grade silicon production
- General fabrication materials
- Educating students about electronics and the design process
- Trello
- GitHub Desktop
- Visual Studio Code (VSC)
- SSH (remote via VSC)
- VNC (remote desktop GUI)

Not all of these subjects are explored as extensively as others.

18.2 Overall reflection on my process

18.2.1 Project setup phase

In the first weeks of my internship I discovered that I was really focused on doing everything in the design process as it was shown in the V-model. I think that in the beginning of a project this is the best way to start. However I do think that planning or creating a very detailed plan of approach is a waste of time mostly. I noticed that I knew not enough about the subject to create a real plan of approach so next time I will start by doing a lot of research on the subject. And creating a Trello planning board accordingly. All gathered information can written down and with that information I plan to create my first functional specifications. This seems to me, the best approach for the setup phase next project.

18.2.2 Project general phase

18.2.2.1 Selecting components process

After finishing the functional and some of the technical design, I had to do a lot of research on components, chips and functionality in general. I noticed that I didn't have my goals and my specifications clear enough to select the right chips, drivers or other components. This resulted in choosing many chips with features I ought to be useful one day whilst they may have been overspecced heavily for the applications, even for future plans. This resulted in not being able to make choices and wasting time. I selected a chip, added complimentary components and figured out later that it was not a suitable chip to begin with, because I forgot X or Y, which was an specification required by the client.

Next time I will make sure that my technical specifications matchup with my functional specifications and I will ask my client to review the components I chose before implementing them into a design.

18.2.2.2 Testing on breadboard

I think that when I was creating my first PCB for Rastaban, I should have spend time testing things first on breadboard. This way I would have found a lot of problems that could have been fixed before ordering a PCB. Now the first PCB had many, many mistakes on it. Some which where unfortunately not really easy to spot and some mistakes could have been easily prevented if I had just tested it once on breadboard. Of course I was limited by the time to chip components and some components are not available in breadboard friendly packages (although there are workarounds) and testing them are time consuming.

It was not wrong from me to choose to order a PCB so soon without testing everything, it is not always wise to spend the time testing everything, since setting up tests can take much time too. But testing the bare minimum would have been smart. Don't implement something that is complex and may behave differently from how you think it might.

I can conclude that next time I will be testing all components that I can test easily on breadboard, will be tested and the ones that are not easily tested must be researched heavily before implementation.

18.2.2.3 Documentation

While working on my project I had to write documents of course. These documents should contain everything that reflects my design choices and a summary of my work.

I was very motivated this project to write good documentation on the design process and tests. I followed the V-model and I think that I have made good documents. I could however improve on the way I write down my specifications/requirements and I could work on my diagrams. I think that the tables that I am using for showing my requirements are a bit to simple, not detailed enough and difficult to read. I could improve it maybe by looking up a standard made by others. I do like the coloring I used for the tables. The diagrams should maybe be reviewed more often. Some of the diagrams may be created later in the project process, since loads of things get altered half way in the project.

18.2.3 Learning goals

During my internship, I planned 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.

18.2.3.1 Kicad6 and PCB workflow skills

I think that I succeeded in improving my kicad6 skills. I have designed 3 prototype PCBs and I think that I learned especially when to start designing and when to make improvements and how. I mean to say that it is not always a smart move to create a redesign, especially minor revisions, right away. It could be time consuming to reroute every wire on a pcb if some connections change. Next time I will be testing a PCB and note every improvement I see. After having tested the PCB extensively I will start a revision. I also learned implementing new footprints and librarys using SnapEda or Library Loader. The latter is really easy to use with the Mouser website and it's components.

Another thing I learned is to always check the footprints that you download from third parties. Some are fine, others are simply wrong. This can lead to non functional prototypes, long hours of debugging and damaged components. The last thing I would like to add is the importance of 3D models. I noticed that using 3D models in your design can potentially save you from real world problems. Especially connectors can be a real problem if you don't remember to position them correctly. Using 3D models also has another benefit: you can export your design as one .STEP file and import it to a program like Solidworks or Autodesk Fusion 360. You can easily design things like a casing around the PCB this way. Something that especially in multidisciplinary design teams like ours is a big win in time.

18.2.3.2 Programming skills

I learned new programming languages during my internship: Python and PYQT. I think I learned the most of object-oriented programming with Python. I learned the way to think will making classes. How functions give features to classes. The language itself really speaks to me. I feel more comfortable at this moment using Python than C++. I think this language gives me more confidence in programming and I expect that it will help me understand C++ and other languages better as well.

Designing software was a challenge for me. The goal was to create software that was easy to adapt to new hardware, pin configurations and make the software easy to understand and maintain.

I think that I learned much from the object orientated programming way I used with my Python programs. Next time I would like to implement the code more into a whole. I now left my code like many small scripts that all do their function, which is fine, but it feels like I created only testing scripts and not a fully functional product.

18.2.4 Other subjects I explored

Like I said in the beginning, I learned many things and to many to describe fully in this reflection. I want to talk about these learning experiences:

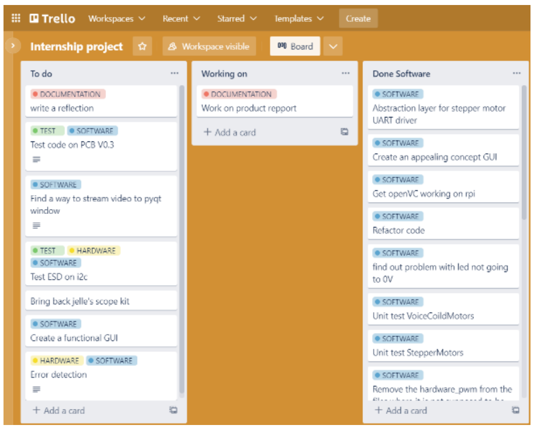
- Educating students about electronics and the design process
- Autodesk Fusion 360
- Trello
- GitHub Desktop

On request from Rudie, but also my own preference I decided to **teach** and help with the UCD2 lesson for the second year industrial product design students. This included helping students with their work by listening to their questions, forming a general idea of the problem, asking for validation on my conclusion and then generating an answer that is adjusted to the skill and understanding of the students. It was important that I wouldn't take over all the work of the students. They had to learn how to program and troubleshooting for themselves, but I did assist where needed or where I thought trouble shooting was beyond the scope of the learning goals.

Overall I would say that it was a useful learning experience and I think that I could see myself doing work as a teacher later when I'm a seasoned electrical engineer or industrial project designer. For now it is to soon, I want to explore my designing skills.

I used **Autodesk Fusion 360** to model a chocolate bar to gain more experience on 3D design, materials like Wood, PLA, SLA and Silicon. I learned some basic functions from Fusion 360, learned how to use fabrication machines and how to improve on my designs without spending much time and money on useless revisions. I also kept material usages and spillage in high consideration.

Trello is a tool that I used to create a general overview of what I had to do, what I was doing (I really do forget that) and what I already have done. I like that I can move the blocks from row to row and back. This way I can easily see what I may need to do again for example. Using the labels functions for software, hardware, documentation and tests (or other custom labels) gave me a quick overview of what I have been doing the most or what needs to be done in a specific category. I will absolutely use this tool again and it is really useful in a team settings as well. It is also easier for me to use in contrast to using a logbook in an excel sheet for example. I tried logging with a excel sheet, but after so many weeks I figured it was really time consuming and I didn't get much useful data out of it in my opinion. In Trello you can view your activity and using GitHub I can track my commit messages with short descriptions as well, so I think this is sufficient for my logging.+



To do
- DOCUMENTATION — write a reflection
- TEST, SOFTWARE — Test code on PCB V0.3
- SOFTWARE — Find a way to stream video to pyqt window
- TEST, HARDWARE, SOFTWARE — Test ESD on i2c
- Bring back jelle's scope kit
- SOFTWARE — Create a functional GUI
- HARDWARE, SOFTWARE — Error detection

Working on
- DOCUMENTATION — Work on product repport

Done Software
- SOFTWARE — Abstraction layer for stepper motor UART driver
- SOFTWARE — Create an appealing concept GUI
- SOFTWARE — Get openVC working on rpi
- SOFTWARE — Refactor code
- SOFTWARE — find out problem with led not going to 0V
- SOFTWARE — Unit test VoiceCoildMotors
- SOFTWARE — Unit test StepperMotors
- SOFTWARE — Remove the hardware_pwm from the

Figure 34 Trello page

I used **GitHub Desktop** to improve my version management. I found using GitHub Desktop really easy to use and I almost never had any problems with it. I only experienced one problem with GitHub and it was solved relatively quickly. A benefit of using GitHub Desktop is that I can download all the files to the raspberry pi with a few command lines.

I did notice that is it faster to use SSH in visual studio code to run python scripts, so for coding I still prefer that, but for moving files or big chunks of new code, using GitHub is great.

18.2.5 Writing things down
I learned to write things down in the middle of a conversation. This may sounds stupid but it really changed things up for me. By writing conversations down I am able to remember what we talked about and I don't have to worry about any details getting lost or forgetting that the conversation happened as a whole. I will write more conversations down in the future and possibly record conversations if that is agreed upon.

18.2.6 Troubleshooting
When troubleshooting the TMC2209 driver UART connection I was debugging for hours and was convinced that the problem lay in the hardware of my PCB. I was so convinced because I thought that I had seen the UART connection working with two boards on a breadboard of mine. I was so convinced that it has worked once, that I completely ruled out software or settings on the RPI as a problem. Of course my parameters could have been changed, but I just redownloaded the code I worked with the first time and everything "worked" somewhat. I think that I can conclude that seeing something "work" doesn't mean that it is working the way I think it is working and ruling things out that way is time consuming. I also figured that I need to grab my logic analyzer sooner than later in most situations. Just running tests with different values over and over again is time consuming and getting lucky with the right values is really not that scientific or effective.

18.3 Reaching a goal and documenting it
I only started looking at the BA intermediate internship (S5) grading form at the ending of my internship period. If I had looked at the beginning of my period, I would've been able to have a clearer view on what documentation and steps where required apart from the V-model.

I think I now lacked a feeling of process in the project. A begin, mid and end. I mostly spend my time in the middle of the project, tinkering and designing. This resulted in not having any clue where I needed to end and what I had to deliver to my school (and my client). I had a struggle getting my documentation to a good standard because I didn't know when to write what and I therefor couldn't use my documentation as guideline or fallback paper if things went wrong. It felt like going back to step 1 when something went wrong instead of step 5 out of the 10 steps from designing to completing a project.

Next time I will make sure that I have a clear understanding of how to project will be going, with great emphasis on how to documentation runs along the project instead of doing all the documents at the end of the project. This way I will have documentation to fall back on when a design doesn't function the way I expected. It also means that my project doesn't end with: now I need to write documents based on my results, but rather: the results are based on what I documented earlier.

I now know two ways to write documentation:

1. A product report, which you usually write after having created a functional prototype. (some documents are being made in the process like a schematic).
2. A process report, which describes how the product or prototype has become what is it now.

The first can arguably a more professional straight to the point way of telling what was made and how it can be reproduced. The latter describes more how things have come to be and therefor shows progress and considerations, which can be very useful for other teams that will work with the prototype in my opinion.

Both options seem useful and one is not per definition better than the other. Which I pick should be discussed with my client.

What I can say for certain and to end my reflection is that I still have many things to learn by doing and exploring and I really look forward to doing so.

19 Appendix E: 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 disconnect s during shutdown
TMC2209	Stepper driver	High quality, good document ation	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.	

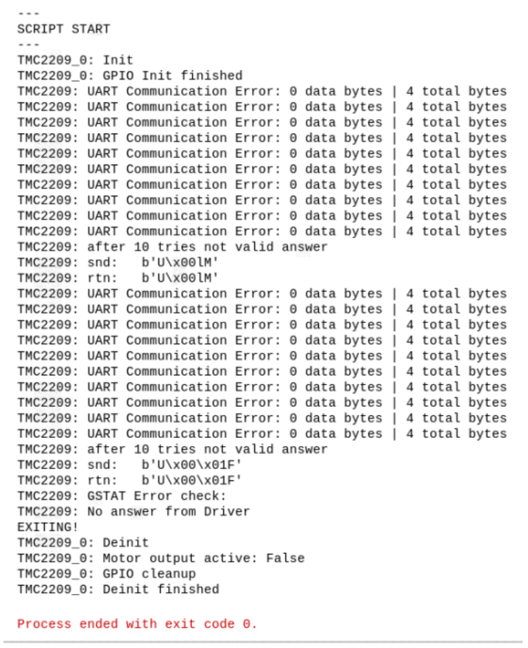
20 Appendix F: TMC2209 debugging

20.1 UART not connecting with multiple stepper motor 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.

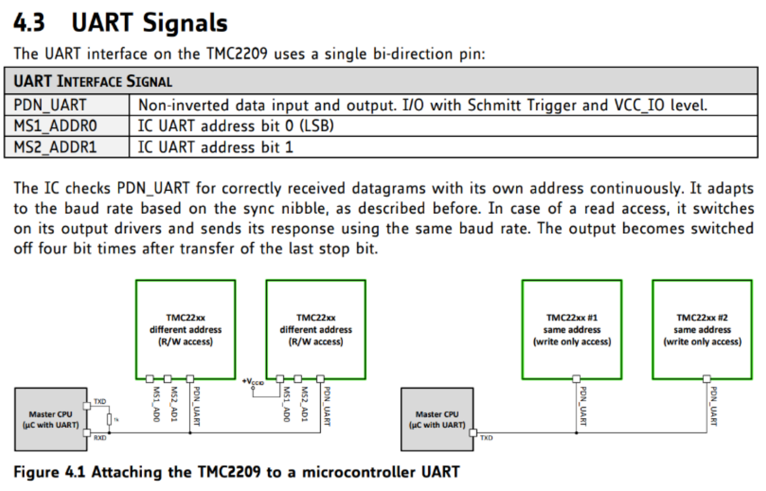
I configured one driver to be 01 and the other 00 by connection the ms1 and ms2 to GND and VCC the way it was shown in figure 4.1 above. Somehow, I always get this message:



```
---
SCRIPT START
---
TMC2209_0: Init
TMC2209_0: GPIO Init finished
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: after 10 tries not valid answer
TMC2209: snd:   b'U\x00lM'
TMC2209: rtn:   b'U\x00lM'
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: UART Communication Error: 0 data bytes | 4 total bytes
TMC2209: after 10 tries not valid answer
TMC2209: snd:   b'U\x00\x01F'
TMC2209: rtn:   b'U\x00\x01F'
TMC2209: GSTAT Error check:
TMC2209: No answer from Driver
EXITING!
TMC2209_0: Deinit
TMC2209_0: Motor output active: False
TMC2209_0: GPIO cleanup
TMC2209_0: Deinit finished

Process ended with exit code 0.
```

Figure 36 error output



4.3 UART Signals

The UART interface on the TMC2209 uses a single bi-direction pin:

UART Interface Signal	
PDN_UART	Non-inverted data input and output. I/O with Schmitt Trigger and VCC_IO level.
MS1_ADDR0	IC UART address bit 0 (LSB)
MS2_ADDR1	IC UART address bit 1

The IC checks PDN_UART for correctly received datagrams with its own address continuously. It adapts to the baud rate based on the sync nibble, as described before. In case of a read access, it switches on its output drivers and sends its response using the same baud rate. The output becomes switched off four bit times after transfer of the last stop bit.

TMC22xx different address (R/W access)
TMC22xx different address (R/W access)
TMC22xx #1 same address (write only access)
TMC22xx #2 same address (write only access)
Master CPU (µC with UART)
TXD
RXD
+VCCIO
MS1_AD0
MS2_AD1
PDN_UART
Master CPU (µC with UART)
TXD
PDN_UART
PDN_UART

Figure 4.1 Attaching the TMC2209 to a microcontroller UART

Figure 35 UART control (Trinamic TMC2209 datasheet)

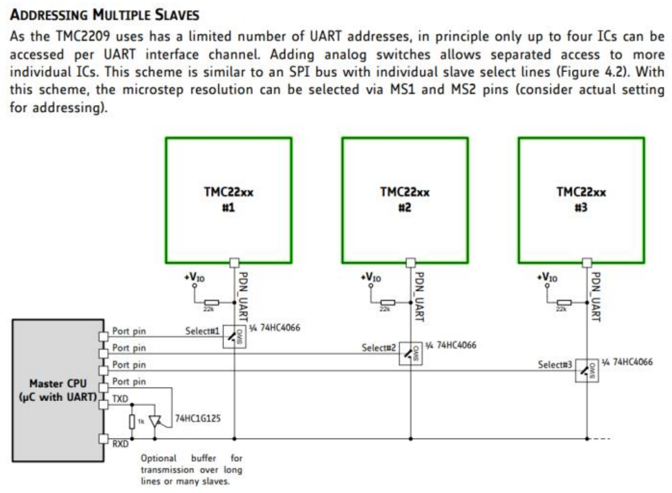
These are the configurations I evaluated with:

Table 10 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		

20.1.1 First Conclusion/solution

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.



Addressing Multiple Slaves

As the TMC2209 uses has a limited number of UART addresses, in principle only up to four ICs can be accessed per UART interface channel. Adding analog switches allows separated access to more individual ICs. This scheme is similar to an SPI bus with individual slave select lines (Figure 4.2). With this scheme, the microstep resolution can be selected via MS1 and MS2 pins (consider actual setting for addressing).

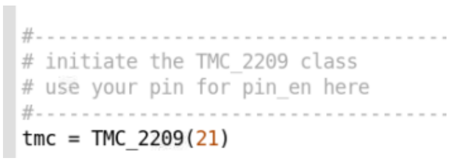
Figure 37 Addressing multiple slave (Trinamic TMC2209 driver datasheet)

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.

20.2 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:



```
#-----------------------------------
# initiate the TMC_2209 class
# use your pin for pin_en here
#-----------------------------------
tmc = TMC_2209(21)
```

Figure 38 Pin configuration

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

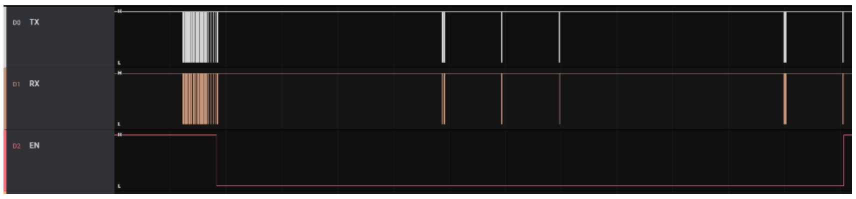


Figure 39 successful transmission

When zoomed in it is clear that the TX deviates from the RX line data.

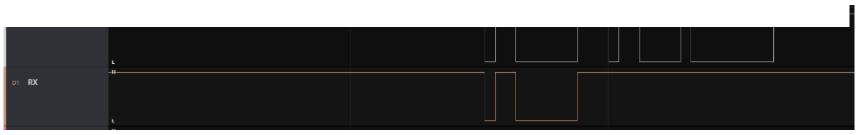


Figure 40 Deviating TX and RX

Further inspection shows that it does occasionally match the RX line

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

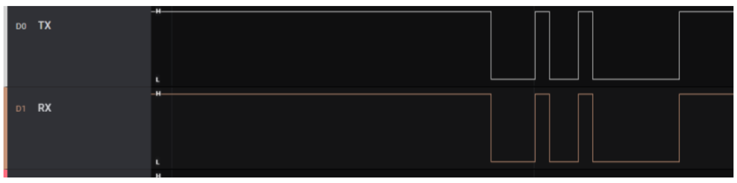
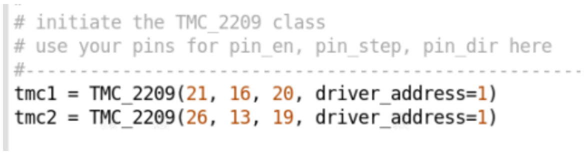


Figure 41 RX and TX matching occasionally

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:**



```
# initiate the TMC_2209 class
# use your pins for pin_en, pin_step, pin_dir here
#-----------------------------------------------------
tmc1 = TMC_2209(21, 16, 20, driver_address=1)
tmc2 = TMC_2209(26, 13, 19, driver_address=1)
```

Figure 43 pin configuration 2

This is what an unsuccessful transmission looks like:

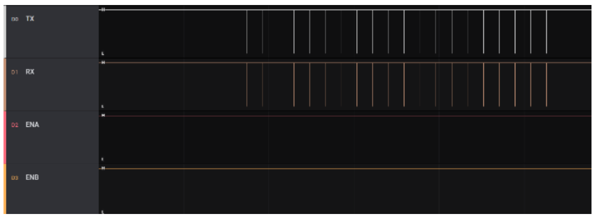


Figure 42 successful transmission

When 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.

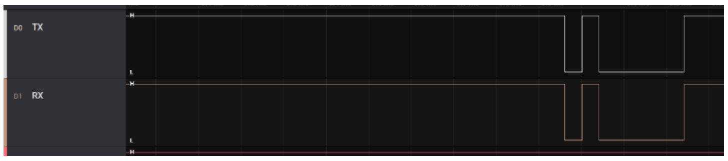


Figure 44 receive and transmit messages

20.2.1 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:

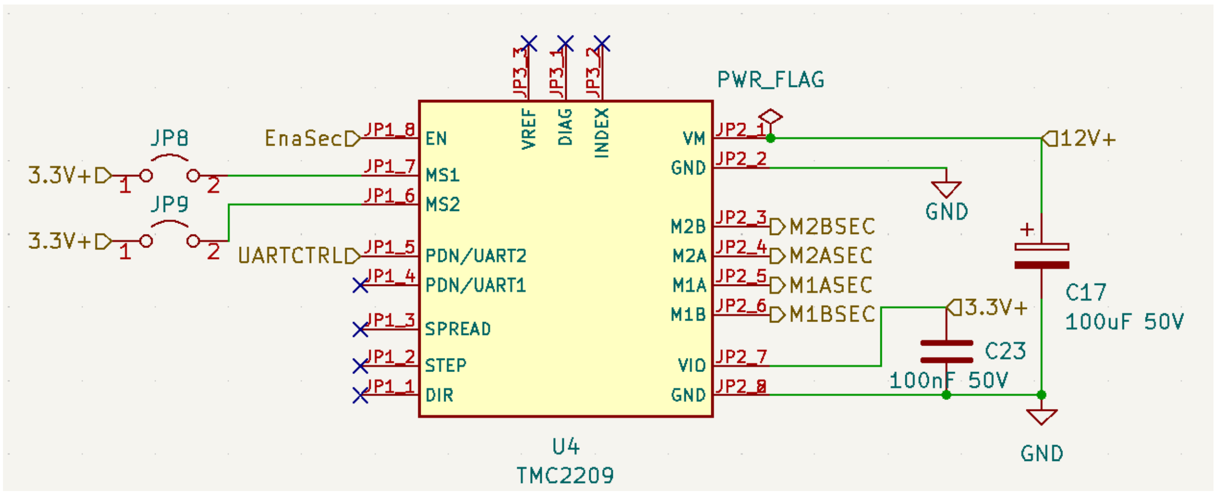


Figure 45 stepper motor schematic

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).

20.2.2 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.

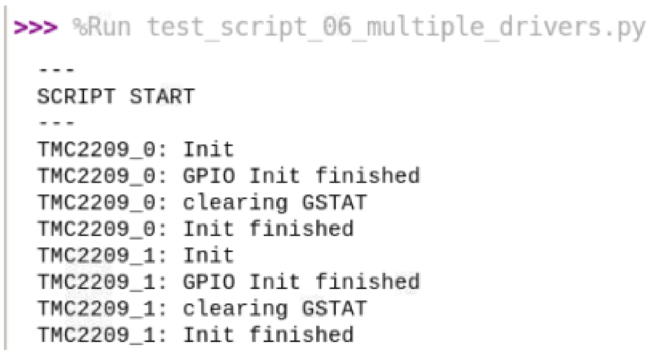
What I do know is that the TMC2208 driver that we used (the old driver) had a solder jumper that was not soldered. A strange decision by the manufacturer. When the solder connection was made, UART was functional. That was the only hardware problem relating UART. Now the software was where the real problem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



```
>>> %Run test_script_06_multiple_drivers.py

  ---
  SCRIPT START
  ---
  TMC2209_0: Init
  TMC2209_0: GPIO Init finished
  TMC2209_0: clearing GSTAT
  TMC2209_0: Init finished
  TMC2209_1: Init
  TMC2209_1: GPIO Init finished
  TMC2209_1: clearing GSTAT
  TMC2209_1: Init finished
```

Figure 46 successful initialization

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.

20.2.3 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. **Hypothesis:** there may be an open circuit. **Plan:** Check the hardware: is everything connected like it should be?
    a. 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.
    b. The motor is connected to the connector and there is continuity.
2. **Hypothesis:** there is no signal coming from the drivers. **Plan:** Measure signals: is there even a signal coming out of the driver?

    a. Yes, I connected my logic analyzer to the primary stepper motor connector and got 4 signals when I send the vactual movement test command.
3. **Hypothesis:** The signal coming from the driver is not suitable for the motor. **Plan:** Check outputs of drivers using logic an analyzer.
    a. 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 test sketch 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)
    b. Then I compared it to the output to 4 motor wires coming from the pcb. This looked vastly 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. **Hypothesis:** The motor connector on my PCB design is not matching the motor pinout. **Plan:** Check PCB design traces from driver to connector and then check motor coil wiring.
    a. I should have checked this first! The wiring was different from what I expected. 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. **Hypothesis:** If one motor driver works, multiple motor drivers will work as well. **Plan:** confirm by running software test.
    a. 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.

Results: After having concluded that point 5 was the cause and having addressed the issue, I can successfully control 3 motors apart from each other. The speed and resolution of stepping can be set in software, but are obviously restricted to the motor's physical build up.

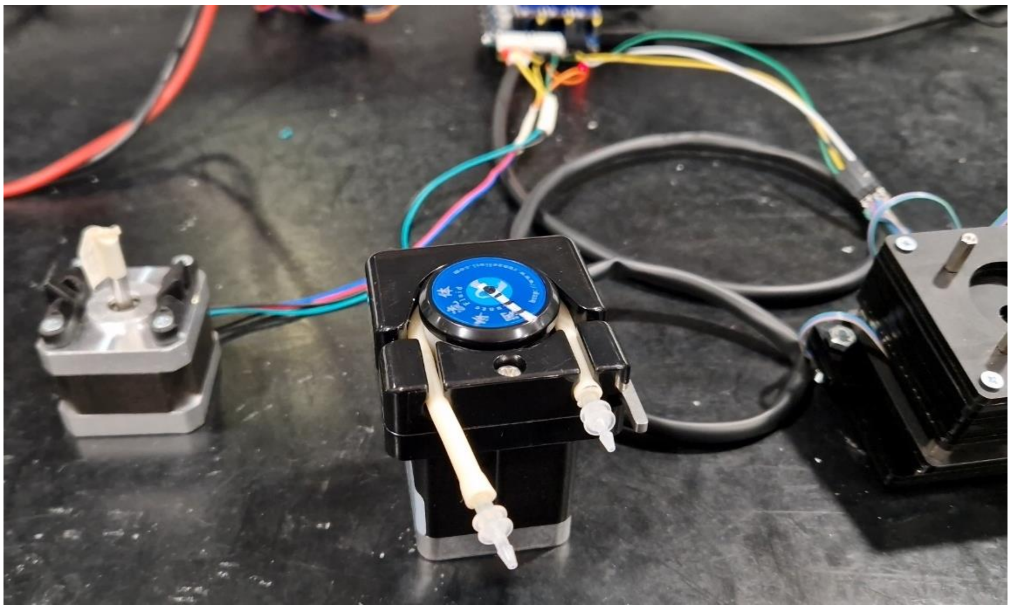


Figure 47 three functional drivers setup