POA Internship

**Main report internship assignment Health Concept Lab**

**Student:** Casper R. Tak

**Studentnumber:** 657313

**Client**: Rudie van den Heuvel

**Coach:** Jeroen Veen

**High School:** HAN Arnhem

**Education:** Embedded Systems Engineering

**Date:** 15-12-2022

# Preface

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 KiCad V6. 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.

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 students. The device is being developed for Jeroen Veen, who is my client for this project.

The ultimate goal of this system is to analyze 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 long-term goal is to create a portable device that can continuously monitor water quality and potentially even test for diseases.

Inhoud

[Preface 2](#_Toc122596743)

[Motive 5](#_Toc122596744)

[Companies and individuals involved 5](#_Toc122596745)

[Main and sub questions 5](#_Toc122596746)

[Goal and end product 5](#_Toc122596747)

[Theory 6](#_Toc122596748)

[Schedule 6](#_Toc122596749)

[Design 6](#_Toc122596750)

[GUI 12](#_Toc122596751)

[Statemachine 12](#_Toc122596752)

[State diagram 13](#_Toc122596753)

[Use Case Diagram 14](#_Toc122596754)

[System Architecture 15](#_Toc122596755)

[Kesselring 16](#_Toc122596756)

[KiCad 17](#_Toc122596757)

[Why KiCad6? 17](#_Toc122596758)

[Plugins 17](#_Toc122596759)

[How to learn KICAD6 19](#_Toc122596760)

[The design approach 19](#_Toc122596761)

[Hardware 20](#_Toc122596762)

[Microscope led 20](#_Toc122596763)

[Testing plan 21](#_Toc122596764)

[Driver circuit decision table 21](#_Toc122596765)

[Stepper motor driver 23](#_Toc122596766)

[UART on the TMC2209 24](#_Toc122596767)

[UART Problem with multiple drivers 24](#_Toc122596768)

[Conclusion and solution for the UART error 28](#_Toc122596769)

[The modules 30](#_Toc122596770)

[Schematic 30](#_Toc122596771)

[PowerManagementSystem 31](#_Toc122596772)

[Power Supply 31](#_Toc122596773)

[Preventing Noise 31](#_Toc122596774)

[Planes on PCB 31](#_Toc122596775)

[EMC 32](#_Toc122596776)

[ESD protection 32](#_Toc122596777)

[Power regulators 33](#_Toc122596778)

[12V vs 24V 33](#_Toc122596779)

[Trace width 33](#_Toc122596780)

[2 VS 4 layer PCB 35](#_Toc122596781)

[Coil whine 35](#_Toc122596782)

[Backpower protection circuit 36](#_Toc122596783)

[Pi Hat i2c EEPROM interface 37](#_Toc122596784)

[12V Power GPIO 37](#_Toc122596785)

[Focusing the lens 37](#_Toc122596786)

[Software 38](#_Toc122596787)

[Doxygen 38](#_Toc122596788)

[Components 38](#_Toc122596789)

[Microscope LED 38](#_Toc122596790)

[TMC2209 Stepper motor driver 38](#_Toc122596791)

[PowerGPIO, Cooling Fan, Heating resistor 39](#_Toc122596792)

[Peltier driver 39](#_Toc122596793)

[Pigpio 39](#_Toc122596794)

[Init function (PIGPIO daemon) 39](#_Toc122596795)

# Motive

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.

# Companies and individuals involved

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

**Rudie van den Heuvel** is a 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 ELC teachers can also be of help. For EMC I can probably go to Ivo van Diemen De Jel.

# Main and sub questions

The soul reason of creating this POA is to gain an 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 things, it should be possible for me to create a planning.

To get a understanding of the project’s size and the current challenges, I need to know what the problems are.

Problem: there is currently no all in one compact and functional driver board for the device.

# Goal and end product

The goal is to create a water analysing device. My goal would be to create a printed circuit board along with software for the raspberry pi that will control the monitoring system and retrieve all the data collected by it.

# Theory

See OneDrive documents.

I’m going to approach this project via the V-model

# Schedule

See schedule document.

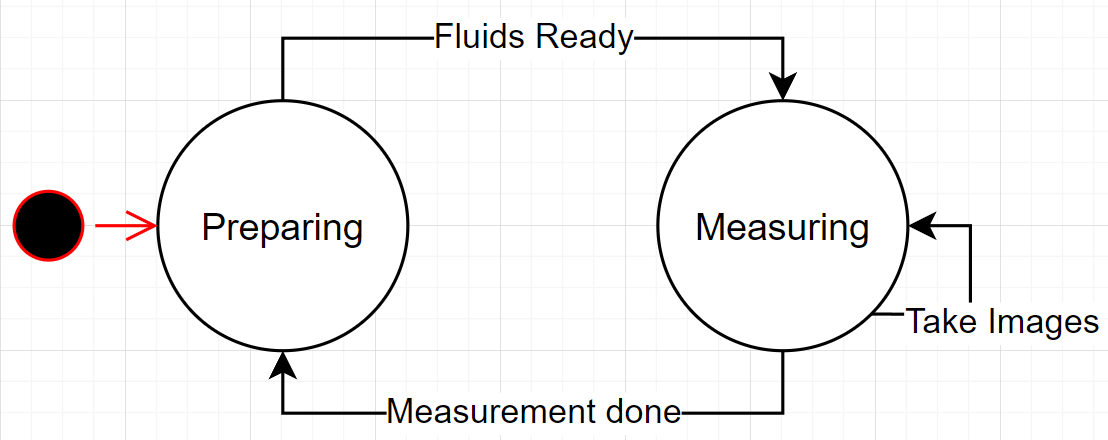
# Design

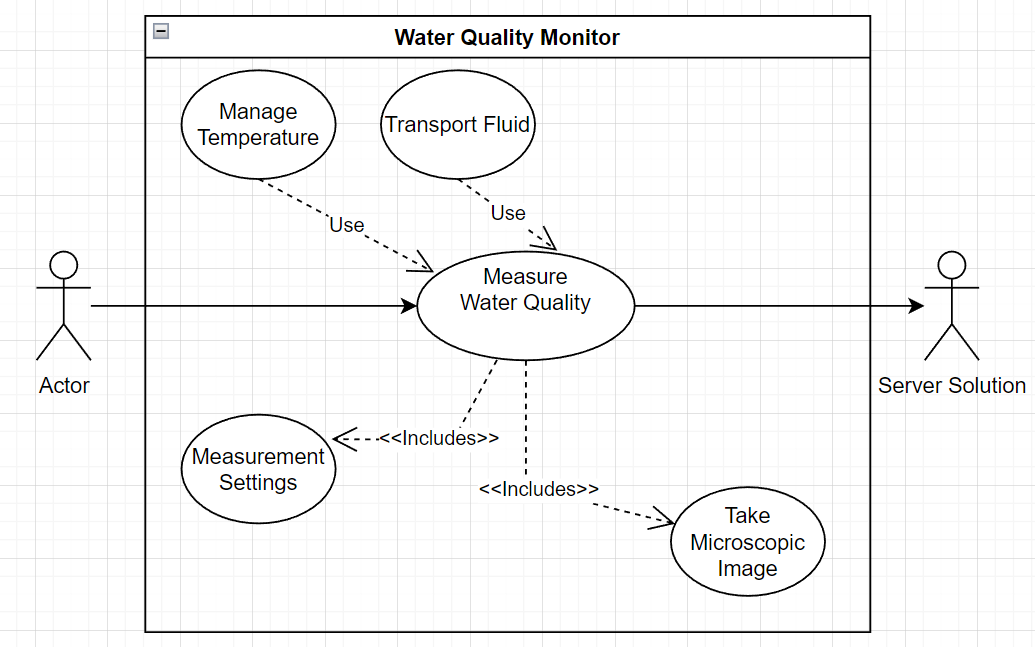
|  |  |  |
| --- | --- | --- |
| **Functional specifications** | | |
| **#** | **MoS CoW** | **Description** |
| F1 | **M** | **The device is collecting data with the biosensor setup** |
| F1.1 | **M** | The device can detect particles |
| F1.2 | **M** | The device has a light source that provides sufficient 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 is able to decontaminate the examination chamber** |
| F5 | **M** | **The device needs to be able to connect to external sensors** |
| F6 | **M** | **The device must be able to be carried by men** |
| F6.1 | **S** | The device should be as small as possible |
| F6.2 | **S** | The device should be as light as possible |
| F6.3 | **W** | The device will not be battery powered |

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

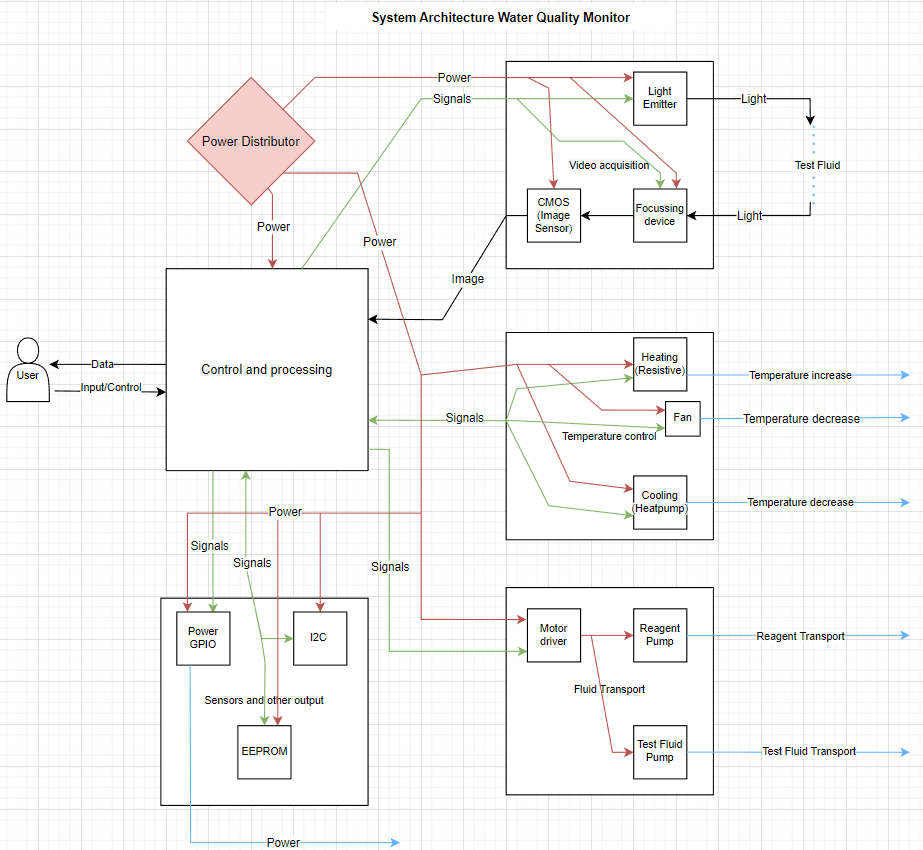
|  |  |  |
| --- | --- | --- |
| **Technical specifications** | | |
| **#** | **MoSCoW** | **Description** |
| **T1** | **M** | **A Raspberry Pi will be used as the computer and controller** |
| **T2** | **M** | **The Raspberry Pi will connect to a PCB hat (PI-HAT)** |
| **T3** | **M** | **The device 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** |
| **T3.1.2** | **S** | **When using magnets, 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 heati** |
| **T8** | **S** | **A flat cable should connect the sensors and motors data to the PI-HAT** |
| **T8.1** | **C** | **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** |

# State machine

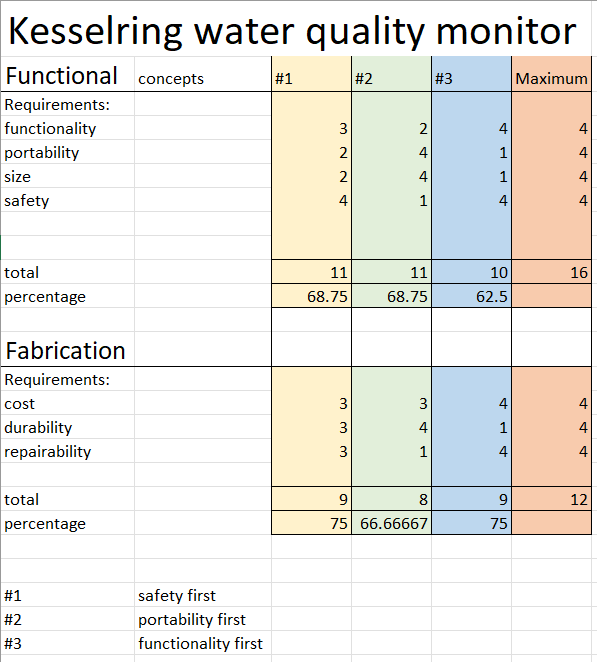


Use Case Diagram

System Architecture



# Kesselring



# KiCad

## Why KiCad6?

Afbeelding met tekst

Automatisch gegenereerde beschrijving

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. KICAD 6 also has plugins that allow you to easily obtain LCSC part numbers for PCB assembly with JLCPCB.

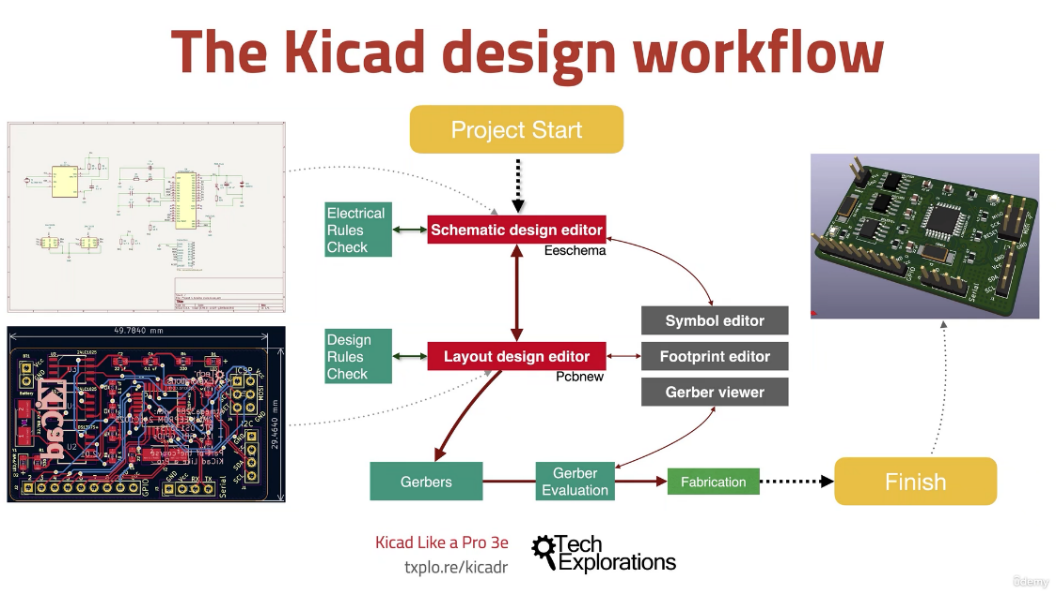
## Plugins

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

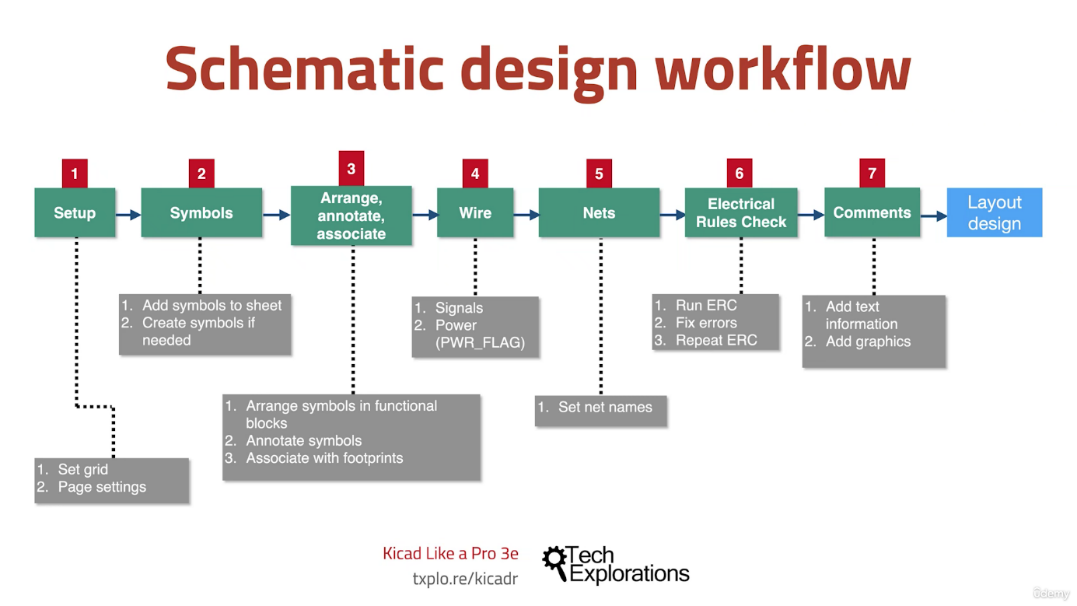
|  |  |  |
| --- | --- | --- |
| Name | Function | Download location |
| KiCAD JLCPCB tools | This plugin allows you to search the JLCPCB parts database, assign LCSC article numbers to your parts, generate production files for JLCPCB and much more. | <https://github.com/Bouni/kicad-jlcpcb-tools> |
| Interactive HTML BOM | This plugin generates convenient BOM listing with ability to visually correlate and easily search for components and their placements on the pcb. | Build into KICAD6 |
| PCB action tools | Annular Ring Checker, Snap Selected Footprint(s) to Grid, Fabrication Footprint Position, Move Selected Drawings to chosen Layer, Export pcb technical layers to DXF, Checking 3D missing models | Build into KICAD6 |
| Archive 3D models | Copies footprint models to the project local subfolder and remaps all the links within the used footprints. | Build into KICAD6 |
| Place Footprints | Arrange sequentially numbered footprints or footprints from multiple hierarchical sheets in linear, circular or matrix arrangement. This plugin works on footprints already present in the layout, so that layout and schematics stay in sync. | Build into KICAD6 |
| Round Tracks | Algorithmically smooth tracks in a predictable manner. Useful for flex PCBs, or just because it looks cool. | Build into KICAD6 |
| Length matching | Track Length Calculator | Build into KICAD6 |
| Freerouting | Auto router for Kicad. It draws all the connections between components for you. Be warned: Auto routing should never be used carelessly, always check the results. | Build into KICAD6 (requires Java) |

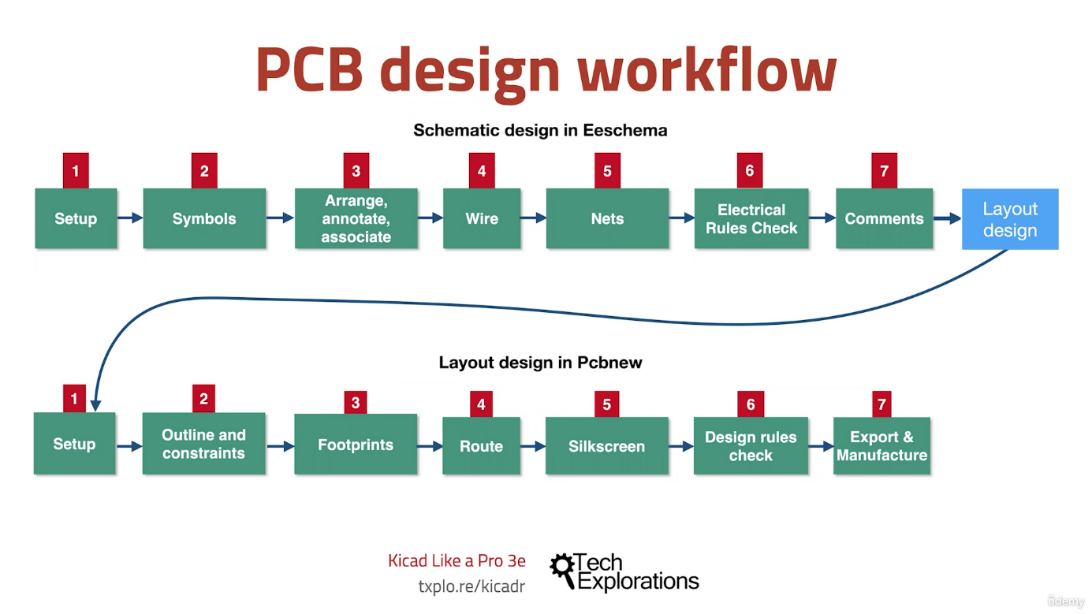
## How to learn KICAD6

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

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

1. Create the schematic
2. Create the PCB

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

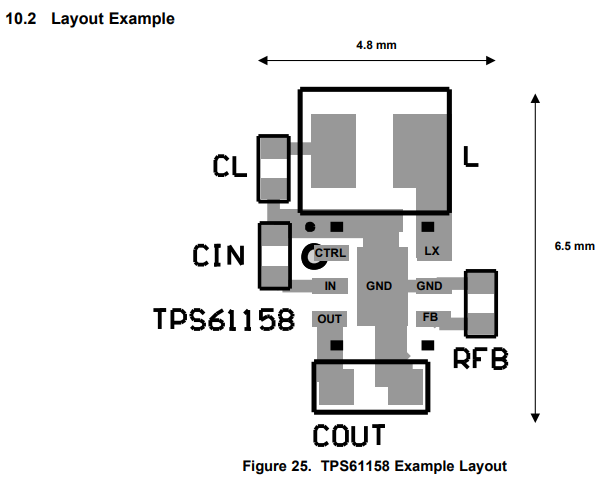


# Hardware

## Microscope led

****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.

It is important to follow the pcb layout recommendations to avoid unsuspected behaviour due to unwanted resistance or capacitances.



### 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.

Our expected shutter speed: 60

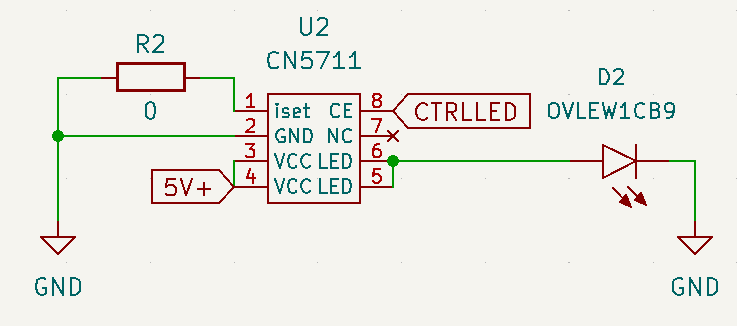
Minimal shutter speed: 300usecs (<https://forums.raspberrypi.com/viewtopic.php?t=323983>)

Python control camera: <https://raspberrypi.stackexchange.com/questions/99304/shutter-speed-and-exposure-time-of-picamera>

About shutter speed and fps: <https://camerajabber.com/what-shutter-speed-for-filming-moving-subjects/#:~:text=The%20180%2Ddegree%20Shutter%20Rule%20states%20that%20whatever%20the%20framerate,What%20is%20this%3F&text=Therefore%20if%20you%27re%20shooting,speed%20should%20be%201%2F120th>.

## Driver circuit decision table

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Nr.** | **Functionality** | **1** | **2** | **3** | **4** | **5** | **6** | **Points** | **Weight factor** |
| **1** | **Dim the led without visible flickering** | - | 1 | 1 | 1 | 1 | 1 | 5 | 5 |
| **2** | **Provide enough power to fully brighten LED** | 0 | - | 0 | 1 | 1 | 1 | 3 | 3 |
| **3** | **Enough brightness levels** | 0 | 1 | - | 1 | 1 | 0 | 3 | 3 |
| **4** | **PWM and digital control** | 0 | 0 | 0 | - | 0 | 1 | 1 | 1 |
| **5** | **PWM only control** | 0 | 0 | 0 | 1 | - | 1 | 2 | 2 |
| **6** | **Highest energy efficiency as possible** | 0 | 0 | 0 | 0 | 0 | - | 0 | 0 |

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.

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.

## Stepper motor driver

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 microstepping, 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 microstepping 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 microstepping can also decrease torque, particularly at higher speeds, which may lead to stalling. Some drivers are equipped with the ability to adjust the stepping mode based on speed in order to mitigate this issue.

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

Afbeelding met tafel

Automatisch gegenereerde beschrijving

### 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 microstepping, 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.

### Afbeelding met tekst Automatisch gegenereerde beschrijvingUART Problem with multiple drivers

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

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

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: Afbeelding met tafel

Automatisch gegenereerde beschrijving

These are the configurations I tested with:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 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 |  |  |

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

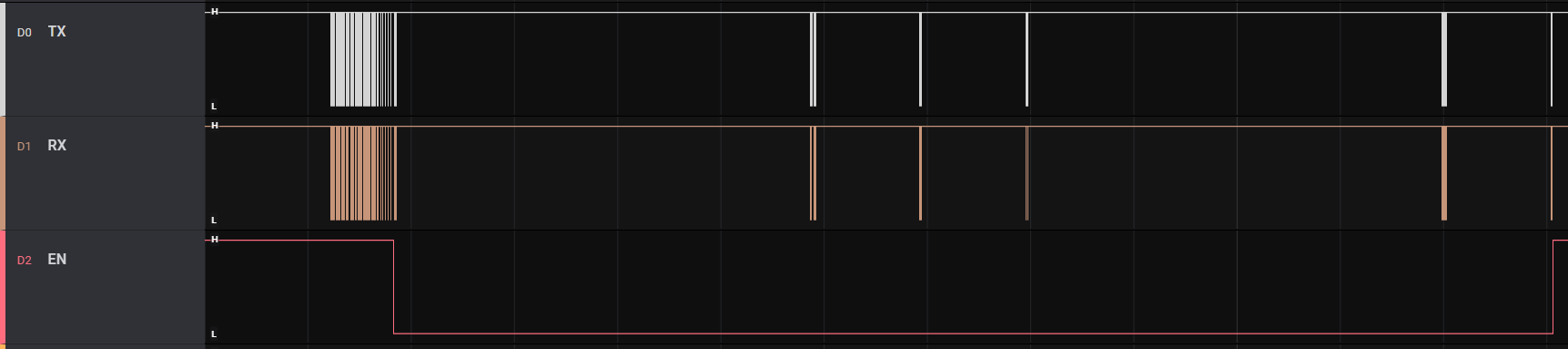
If reading from the driver is not required, using the write only function as seen in figure 4.1 of the datasheet is a viable option. This way we only need to use n+1 pins of the raspberry pi to control the motors, where n increases with every driver added (enable pin). This would treat all steppers as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

Configuration (test script 05: vactual.py) **RX and TX are flipped** **on the picture:**

Afbeelding met tekst

Automatisch gegenereerde beschrijving

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



When zoomed in it is clear that the TX deviates from the RX line data. Afbeelding met tekst, computer

Automatisch gegenereerde beschrijving

Further inspection shows that it does sometimes match the RX line

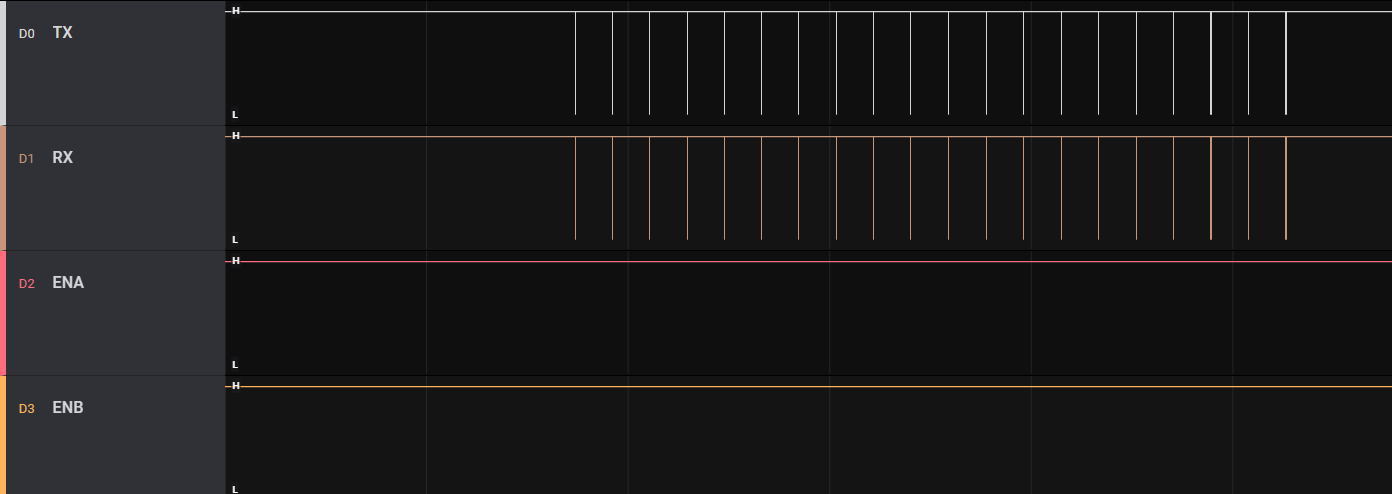
It is however always the case that TX continues sending after RX is done with a package.

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

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

Afbeelding met tekst

Automatisch gegenereerde beschrijving

This is what a unsuccessful transmission looks like: 

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.Afbeelding met tekst

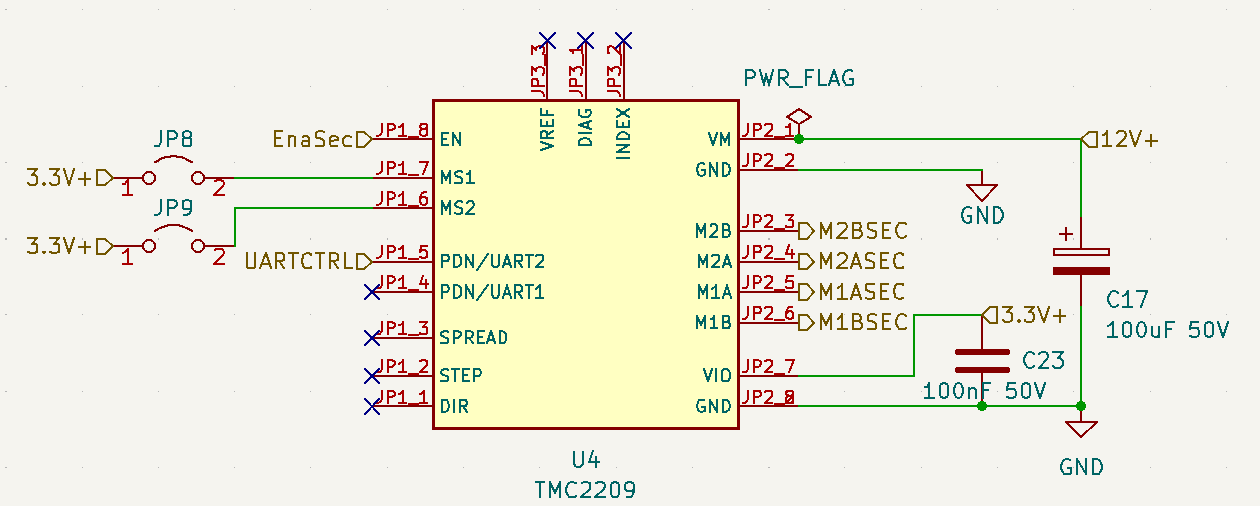
Automatisch gegenereerde beschrijving

### 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.

The fix for this problem was found by Thomas Ijsseldijk. 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:



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).Thermal Control

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

### The modules

The Thermal control consists of three power components:

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

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

### Schematic

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

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

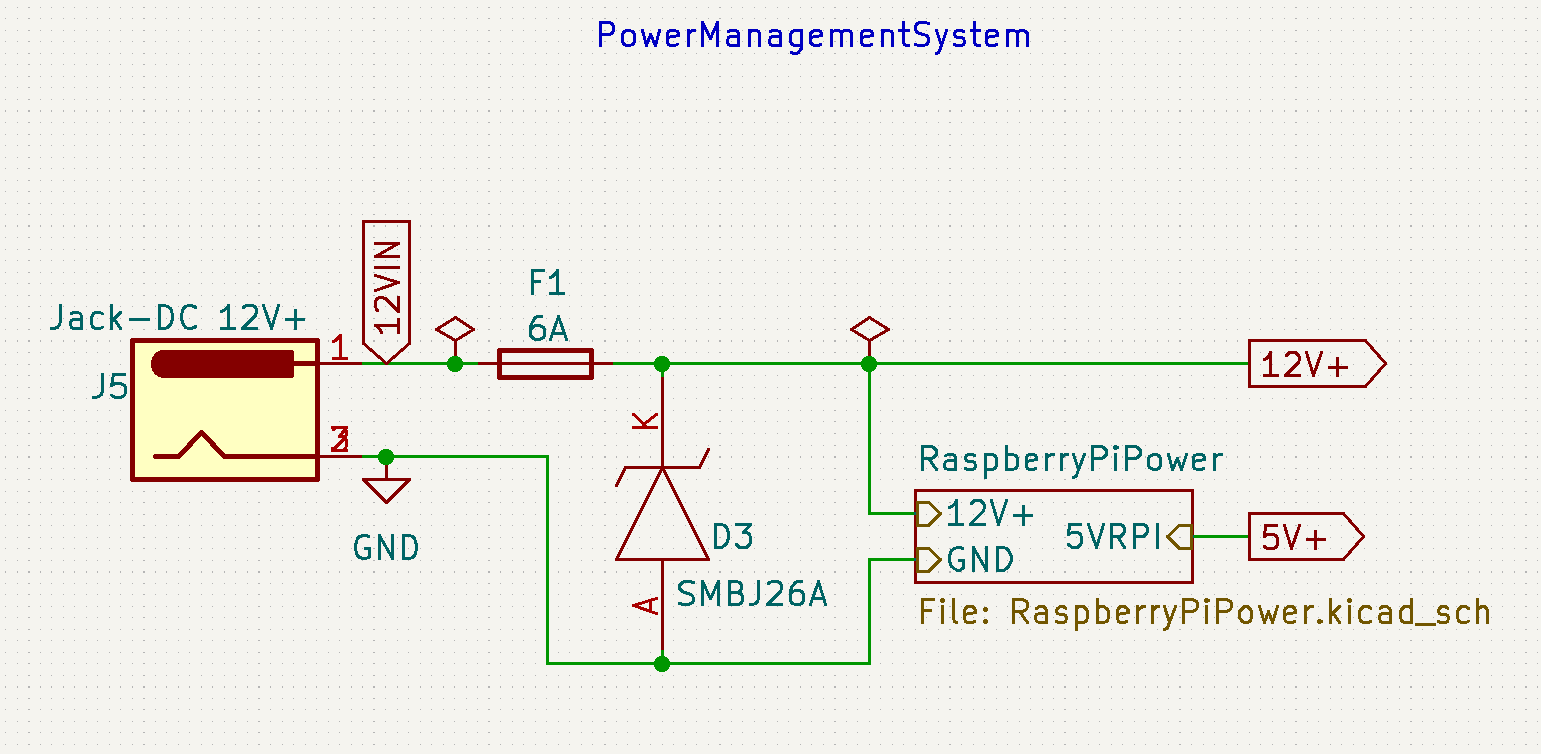
For the Peltier module we used a H-bridge since the Peltier module can cool or heat the same side when current is reversed. The module can help reaching higher temperatures in combination with the power resistor. When the current is reversed cooling the same surface can be achieved.

## 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:



### Power Supply

This table is just an indication of the possible power consumption 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 whole device at full power, something that will not be typical but should be accounted for. This power supply will be an external one for the time being, since it eliminates the need for a much bigger PCB and extra EMC and safety precautions. It also gives us the possibility to easily swap the PSU out if it gets damaged.

### Preventing Noise

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

### Planes on PCB

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

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

### EMC

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

### 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.

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

### Power regulators

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

### 12V vs 24V

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

### Trace width

Afbeelding met tafel

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

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

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

1. Rounded tracks can help to reduce increases in wire resistance when making corners, which can be beneficial for maintaining low resistance and good signal integrity.
2. Rounded tracks also give the PCB a visually distinct appearance that can make it stand out.

Overall, the inclusion of rounded tracks in the design serves both a functional and aesthetic purpose.

## 2 VS 4 layer PCB

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

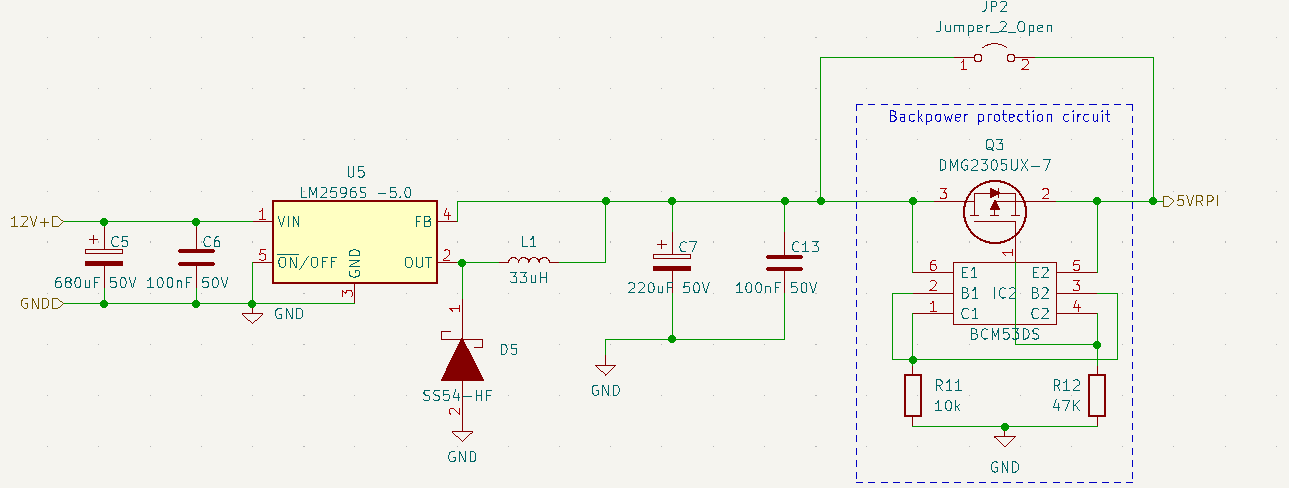
Here are the main differences between 2-layer and 4-layer PCBs:

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

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

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

### Coil whine

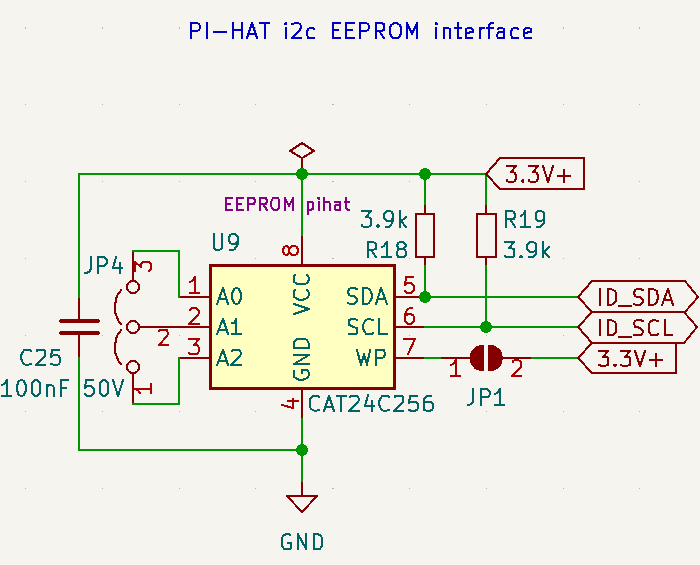
Upon connecting a minor load (300 mA) to the power circuit, I noticed that it generated a high-pitched whine. Upon further investigation, it was determined that the noise was likely emanating from the coil. Upon reviewing the circuit design, I realized that I had omitted the decoupling capacitor from 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.

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

## Backpower protection circuit

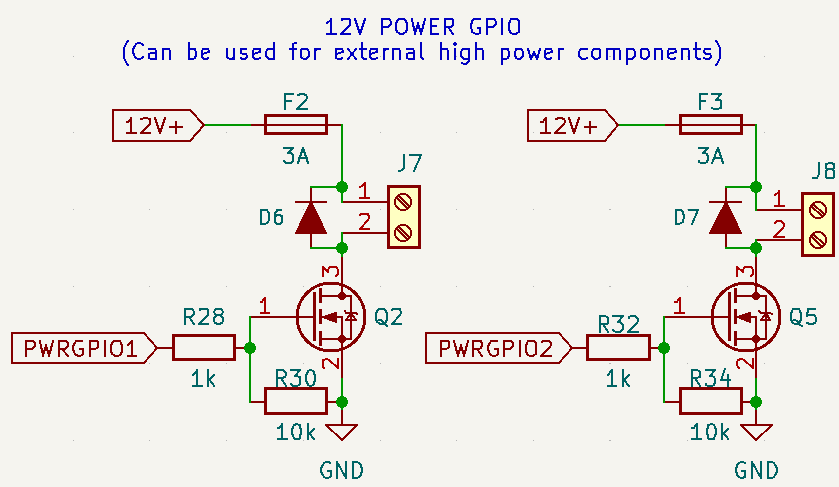
The Backpower Protection Circuit (BPC) is designed to protect the Raspberry Pi from damage in the event that a 5V supply is connected to the micro USB port while a 12V supply is simultaneously connected to the barrel jack on the Pi Hat. However, due to a lack of necessary components, this circuit was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

## 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.

## 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.

This prevents a floating pin and therefor unpredictable behaviour.

## 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. In some cases, specialized lighting techniques, such as dark field illumination or oblique illumination, may be necessary to visualize the features of the small objects. Image processing techniques, such as deconvolution or super resolution, may also be used to improve the clarity and resolution of the images.

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

# Software

The software written for the Rastaban (HAT) has been made in the Python language. This language is chosen because it is widely used in the Raspberry pi community and it gives us the ability to use the PYQT GUI library. Other reasons are maybe for the ease of use and the reason that it is not compiled but interpreted code, which makes it faster for prototyping purposes.

## 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.

## Components

To give a short overview of the code, I will also describe all components in short.

### 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.

### TMC2209 Stepper motor driver

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

### PowerGPIO, Cooling Fan, Heating resistor

The power GPIO pins are driven using software PWM as well as the Cooling Fan and the Heating resistor.

### Peltier driver

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

### Pigpio

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

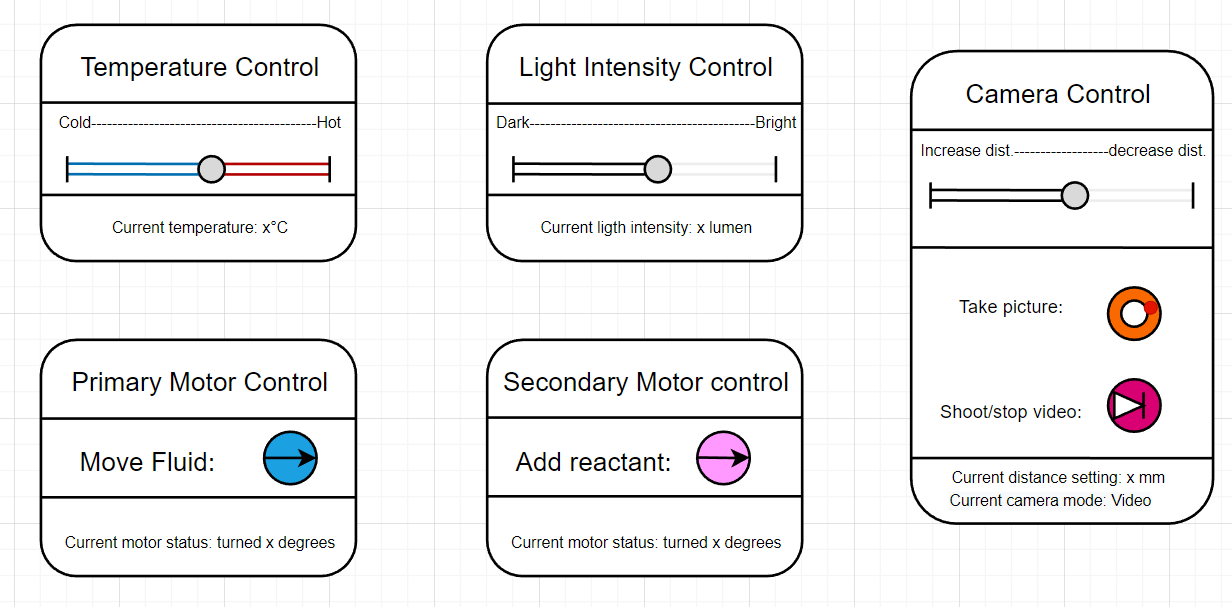
* It provides a simple interface for controlling the GPIO pins: The pigpio library provides functions for setting the direction and level of individual pins, as well as for configuring PWM and detecting changes on the pins.
* It is lightweight and efficient: The pigpio daemon runs in the background and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

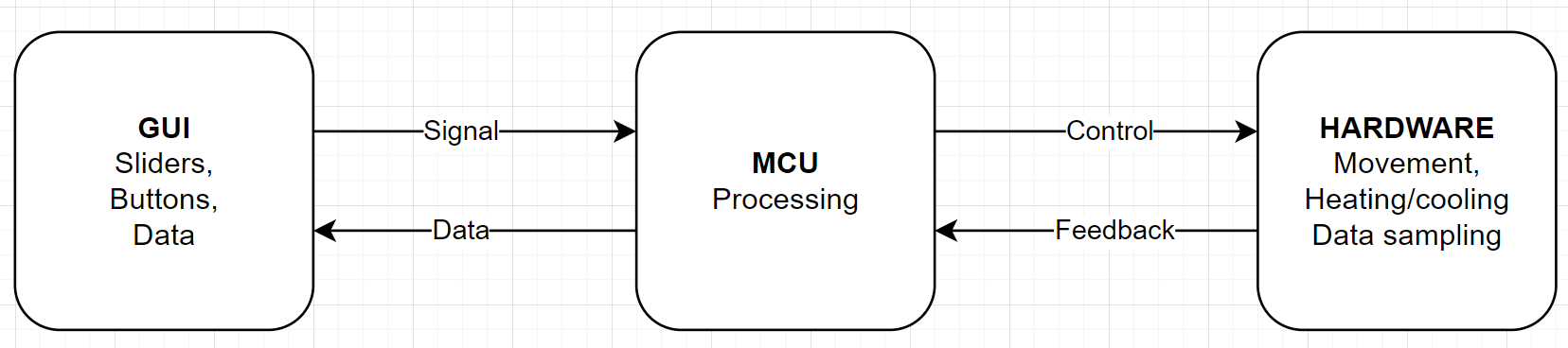
Overall, the pigpio library is a popular choice for controlling the GPIO pins on a Raspberry Pi because it is easy to use, efficient, flexible, and well documented. That is the reason why we choose to implement it in our Rastaban code.

### Init function (PIGPIO daemon)

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

# GUI





State diagram