Inverted Pendulum II

Electronic Product Development II (MECH-B-5-AED-EPE2-ILV)

Bachelor: Mechatronik Design und Innovation

5th Semester

Lehrveranstaltungsleiter: Matthias Gfall

Gruppe: BA-MECH-23

Verfasser: Stefan Hörtnagl

January 3, 2026

Introduction

The project "Inverted Pendulum" was developed, assembled, and commissioned as part of the lectures "Electronic Product Development I & II". The following report covers the development process from digital conceptualization through the assembly of the individual components to commissioning.

All necessary information is provided so that the entire project can be handed over to a new group for the subsequent lecture after the end of the course. This includes the functional principles and procedures used during development as well as the hardware. In addition, recommendations for further improvements and simplifications are given in order to reduce the level of complexity and optimize the project for final use as a demonstrator in the lecture "Control Engineering".

As a basis for this report, the documentation on the development of the PCB from the previous lecture must be taken into account [1]. There, the circuit diagram of the PCB is documented in detail and further completed in the following chapters. The electronic components of the periphery remained unchanged, and no major modifications to the system were required for commissioning.

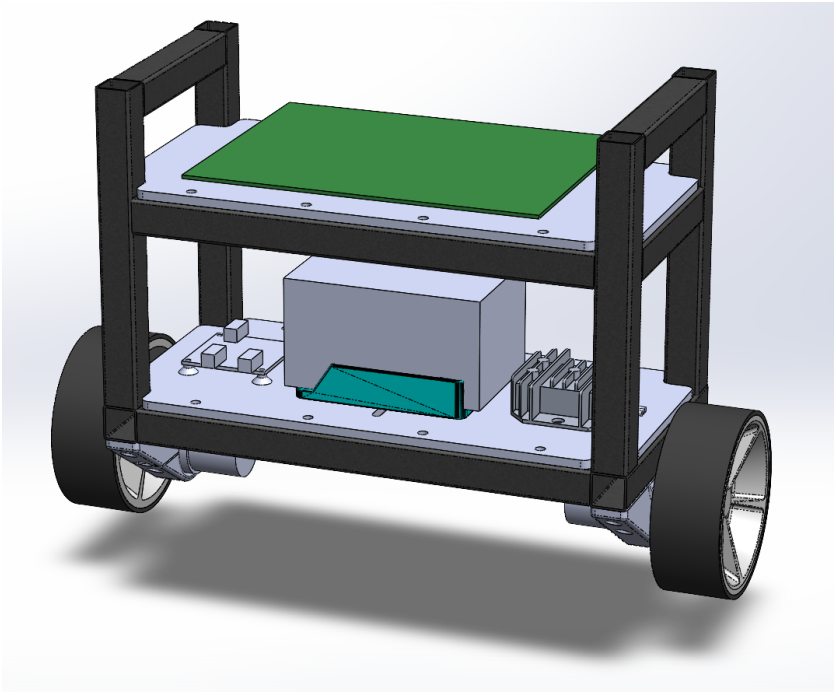


Figure 0.1: Final version of the hardware

To support further development, suggestions for improvements and incentives for optimization were provided in order to give the subsequent group a direction in which the product can be further improved. Figure 0.1 shows the final version of the mechanical development.

To translate the laboratory report and improve the writing style without changing the content, translation tools were used [2].

All project data are available in the following GitHub repository [3].

Contents

1 Project Objective and Project Status

1.1 PROJECT OBJECTIVE

The objective of the project is the development of a demonstrator for the lecture "Control Engineering". It is intended to enable control algorithms to be programmed and tested live from MATLAB via a USB connection.

The control algorithms developed in MATLAB are transferred to the microcontroller, processed, and executed in real time in order to actively stabilize the pendulum against gravity. The required measurement data are acquired via a gyroscope sensor.

In addition, it is planned to transmit relevant system and sensor data wirelessly via WLAN during operation, making live measurement values available for analysis and visualization.

1.2 PROJECT STATUS

The mechanical development of the project is fully completed. The frame, including all intended attachments, has been assembled and tested. Likewise, the printed circuit board is fully populated and mounted.

The microcontroller can be reliably controlled and programmed via the USB interface. All required power supplies have been verified and operate stably within the specified tolerances.

Thus, the mechanical and hardware-related prerequisites are fulfilled. The system is now in a functional basic state and ready for further development and implementation of the software.

In addition, the hardware has been designed in such a way that later extensions and adaptations of the control algorithms are possible. The parameterization of the control system, the implementation of WLAN data transmission, and the evaluation of sensor data are part of the remaining software development.

The project is therefore in a defined handover state: the hardware is verified and operational, allowing further work to focus entirely on software and control development. If required, the project can be continued without any hardware modifications and the software can be implemented.

This is one of two possible ways to continue with the project. The other option results from optimization of the PCB and a redesign to implement a number of improvements. More on this later.

2 Assembly of the Printed Circuit Board

Before assembly began, all components were inspected and labeled to save time during placement. The assembly of the printed circuit board was divided into two steps. In the first step, the first third of the board was coated with solder paste using a stencil, and all components were placed.

Since these are not suitable for oven reflow soldering, all THT components were soldered by hand in a third assembly step.

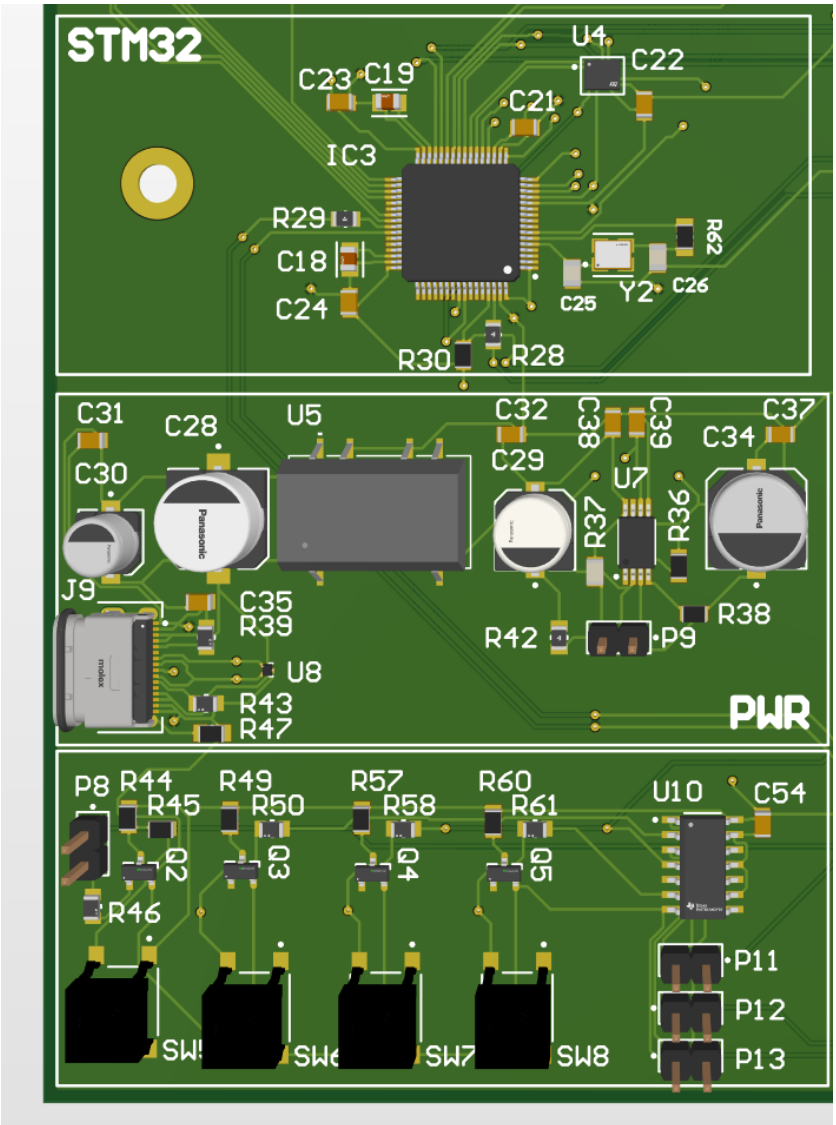


Figure 2.1: First part of the assembly

The components shown in Figure 2.1 were reflow soldered in the oven during the first step. The placement of the microcontroller and other smaller components was carried out using a pick-and-place machine.

In the second step, the remaining part of the board was assembled and reflow soldered, as shown in Figure 2.2. For this purpose, the stencil was cut into two parts, since components from the first step were already mounted on the board.

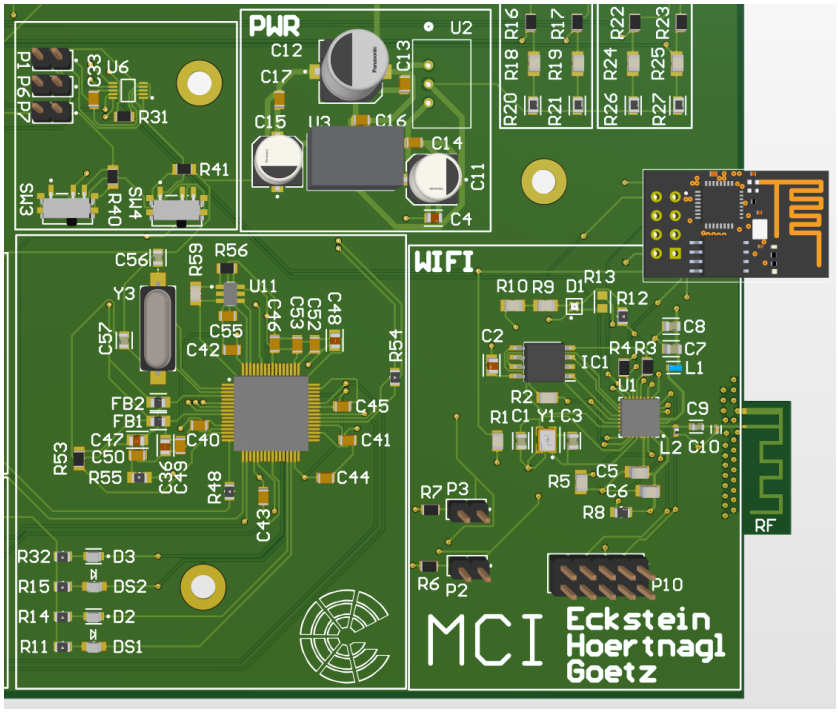


Figure 2.2: Second part of the assembly

In the third step, all THT components were soldered by hand. This mainly concerned the breakout pins and terminal blocks, as shown in Figure 2.3.

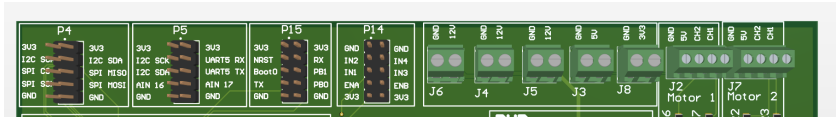


Figure 2.3: Third part of the assembly

3 Commissioning Documentation

During the commissioning of the printed circuit board, several challenges occurred, which were successfully resolved. The identified problems, the applied procedure, and the developed solutions are structured and documented in this chapter.

3.1 PROBLEMS

3.1.1 Power Supply

The printed circuit board has three essential voltage levels required for the operation of the various functional blocks:

- 12 V supply of the hardware from the battery.
- 5 V supply via the USB-C interface.
- 3.3 V supply for the board components.

A central issue was that all three voltage levels had to be present correctly and stably, but should not be initialized simultaneously. For later use, it is intended that the battery and USB-C voltages can be connected in parallel, for example during programming.

During commissioning, the following errors were identified:

- Incorrectly connected common pin on the switches.
- Resistor populated with an incorrect resistance value.
- FTDI chip soldered in a rotated orientation.

These errors led to short circuits on the board, causing the power supply to collapse.

3.2 PROCEDURE

Initial commissioning was carried out using a laboratory power supply in combination with the existing USB-C connection. The current was limited to 50 mA. This limitation is sufficient to verify the voltage levels and identify possible short circuits.

Since no stable voltage level was achieved during the first tests, the problem was isolated step by step. The following measures were taken:

1. Inspection of all resistors and capacitors.
2. Removal of the ESP8266 chip.
3. Removal of the FTDI chip.

During inspection of the resistors, an incorrect resistor was found and replaced. The installed resistor had a value of 10 Ω, although 10 kΩ was required. The excessively small resistance caused a short circuit, which was resolved. This concerned resistor R31, as shown in Figure 3.1.

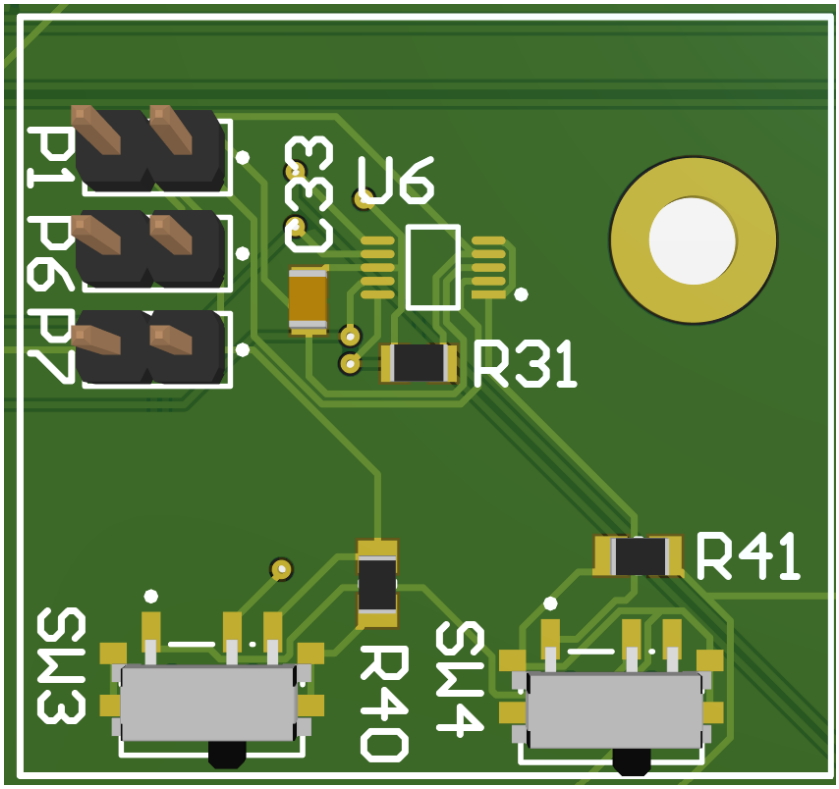


Figure 3.1: Incorrect resistor

In further steps, the ESP8266 and a new FTDI chip were gradually reinstalled. Subsequent commissioning of the power supply was then carried out stepwise according to the reinstalled components. Initially, the board was operated exclusively via the 5 V USB-C supply. All relevant voltages were checked using a multimeter, in particular the correct generation of the 3.3 V supply. Additionally, checks for short circuits and increased current consumption were performed. During further inspection, the voltage collapsed again, especially when actuating the switches for signal switching shown in Figure 3.2.

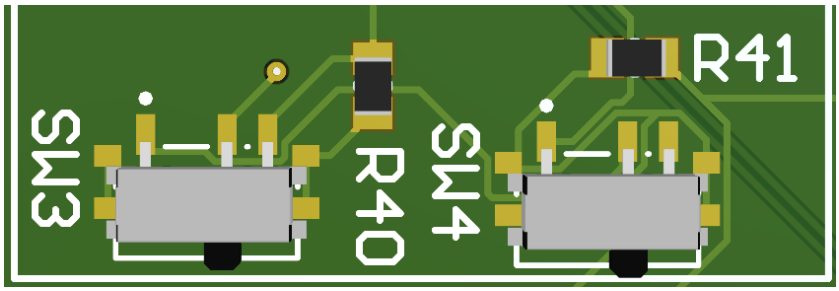


Figure 3.2: Signal switching switches

This behavior is due to the fact that the common pin is not located on the first pin as assumed in the schematic symbol in Figure 3.3, but on the second pin, as stated in the datasheet [4].

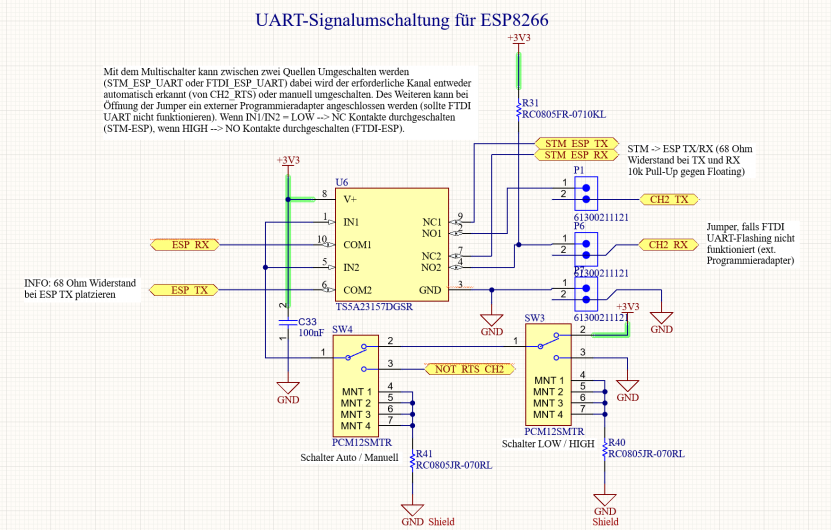


Figure 3.3: Switches schematic

This caused a short circuit when switching, since the voltage level was directly pulled from +3.3 V to ground. For further commissioning, this could initially be ignored, as the first switch position establishes communication between the microcontroller and the FTDI chip, which was sufficient for commissioning the microcontroller. Switching the position establishes the connection to the ESP8266 in order to flash the required firmware.

Thanks to the step-by-step and structured procedure, each voltage level could be checked and validated independently. This allowed faults in the power supply to be identified without endangering other assemblies.

After completion of the described measures, a stable power supply was available. The board could then be operated reliably and was ready for further functional tests.

4 Test of the Communication Interface

After the successful assembly of the printed circuit board, the USB-C serial communication of the FT2232H with the microcontroller is first tested. The goal is to ensure that both channels are correctly recognized, that data can be reliably sent and received, and that the connected target system responds.

It was found that the reset pin of the microcontroller exhibits a floating state. This is due to the open-drain MOSFET (Q2) driven via the SN74LVC14AD, which does not pull the reset pin clearly to either high or low in the case of an undefined input state. After removing Q2 (Figure 4.1), the reset pin was again clearly defined via the pull-up and became stable.

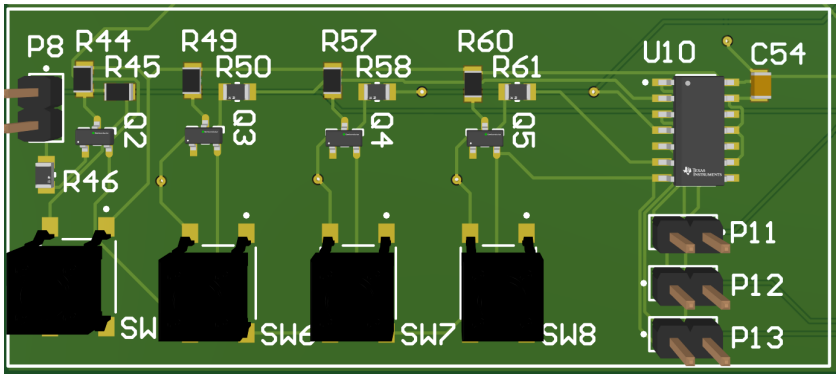


Figure 4.1: Q2

As a result, the reset pin of the microcontroller was at a defined high level, allowing a normal startup when the power supply is connected. Initially, the ports of the FTDI chip were not recognized by the computer. This was resolved by manually resetting the chip, which is described later in the chapter "Operation" 6.

Note: Manually reset the MCU after the USB connection has been established.

4.1 REQUIRED SOFTWARE

The following programs are used and required for the tests:

- Driver for the FTDI chip.
- Python and the package "pyserial" for simple verification of the communication.

4.2 BASIC COMMUNICATION TEST VIA A TERMINAL

1. Open STM32 Cube Programmer.
2. Select the appropriate FT2232H COM port (A or B).
3. Establish the connection.

4. Send characters and check whether a response or an echo appears.

A visible echo or response confirmed the basic functionality of the USB connection and the FT2232H. In this process, the installed version of the microcontroller bootloader is returned. This also indicates that communication with the microcontroller is functioning and that it is ready to flash software.

4.3 QUICK TEST WITH PYTHON

After installation of the corresponding package, a systematic functional test can be performed using the following Python script:

```
pip install pyserial

import serial

ser = serial.Serial("COM20", 115200, timeout=1)

while True:
    print(ser.readline().decode(errors="ignore"))
```

Periodic responses or echoes appear, indicating that the microcontroller is operating and that the interface is functioning reliably.

4.4 VISUAL CONFIRMATION OF COMMUNICATION

As soon as the RX and TX lines actively send or receive signals, the LEDs light up according to the schematic [5]. As shown in Figure 4.2, the two LEDs DS1 (green) and D2 (red) light up when an active connection is present.

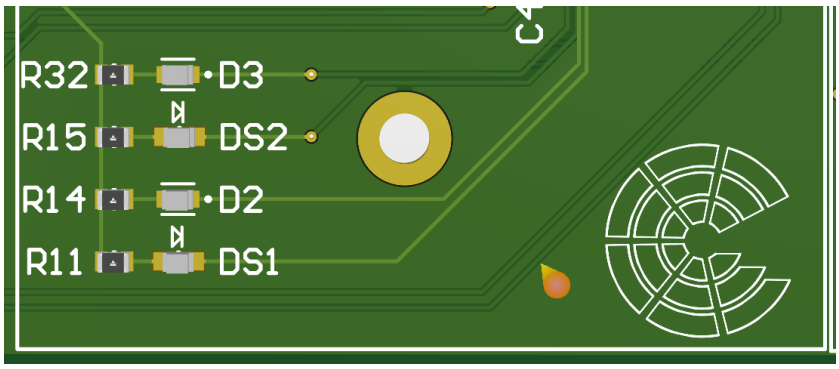


Figure 4.2: LEDs illuminated

Communication has been commissioned and verified up to this point. In the subsequent steps, all further communication paths must be tested.

5 Connection Plan

To complete the hardware, the entire wiring of the demonstrator must still be installed. This essentially includes the following components:

- Battery [6]
- DC/DC converter [7]
- Motor driver [8]
- PCB [5]
- Motors [9]
- Battery holder [10]

5.1 BATTERY CONNECTION

First, the connections between the holder, the DC/DC converter, and the PCB are established. At the end of the entire wiring process, the battery is inserted into the designated holder. The wiring is to be carried out as follows:

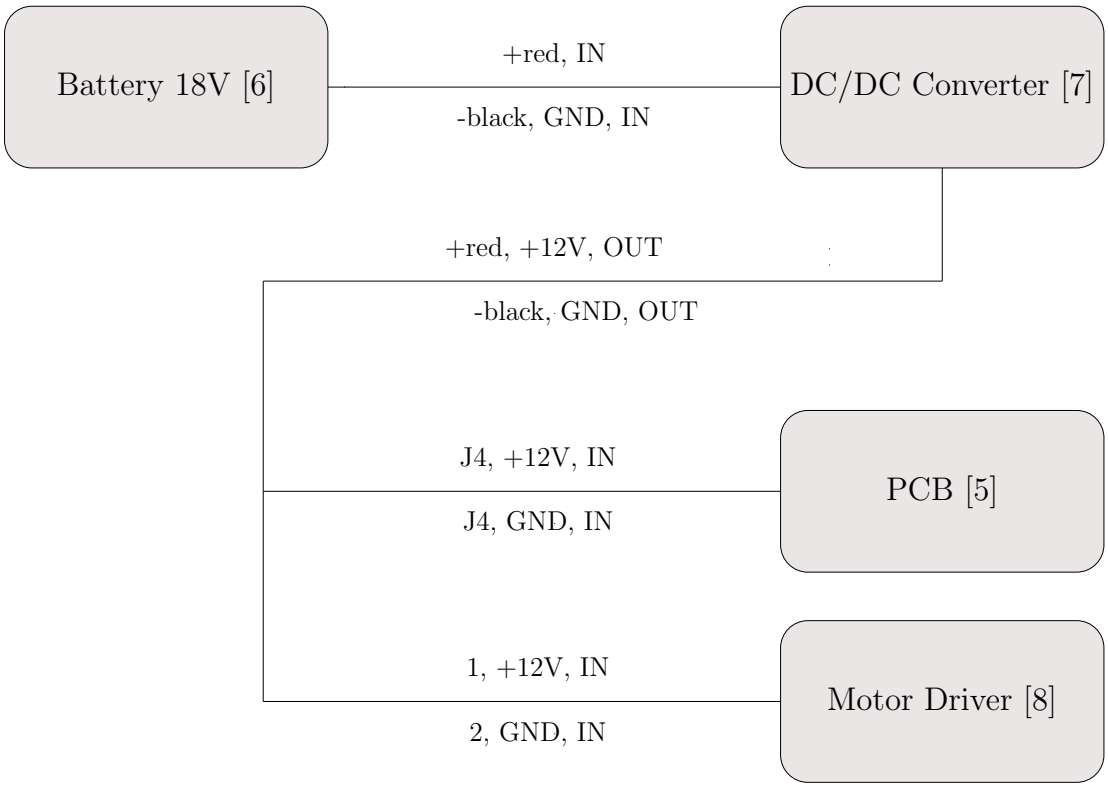


Figure 5.1: Battery to PCB

After wiring, the battery can be inserted. Alternatively, it is also possible to supply the system via a laboratory power supply and limit the current. Subsequently, the following voltage values must be checked:

- 12 V output at the DC/DC converter
- 3.3 V PCB supply
- 5 V PCB outputs

The DC/DC converter [7] must be adjusted to the corresponding 12 V. This must be verified. Afterwards, remove the battery again for further wiring.

5.2 MOTORS

For connecting the motors, Figure 5.2 is used. This must be carried out for both the first and the second motor with the corresponding pins. In addition, it must be ensured that the assignment of the control signals (IN, EN) to the respective motor channel is correct so that control is clearly assigned to the desired motor. Before commissioning, all connections should be checked to avoid malfunctions or damage to the components.

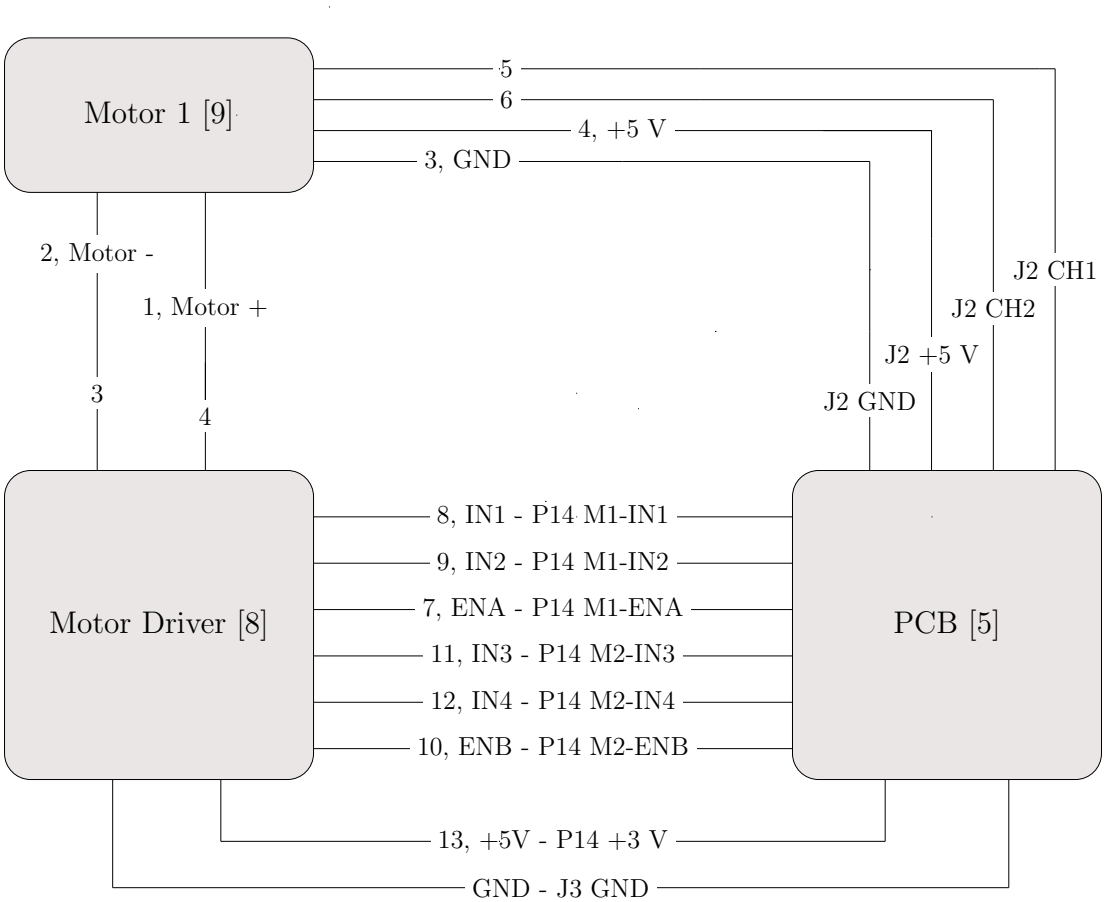


Figure 5.2: Motors

5.2.1 Important Notes on Wiring

When wiring, it must be ensured that all components share a common ground (GND), otherwise the control signals between the PCB and the motor driver will not be correctly recognized. The motor

driver is supplied via the 12 V rail, while the logic reference of the driver must be connected to the 3.3 V logic voltage of the PCB. The encoders may only be supplied with 5 V and never with 12 V. The motor cables should be routed as separately as possible from the encoder and signal cables in order to avoid interference. If a motor rotates in the wrong direction, the two motor connections at the driver can simply be swapped.

The required connections are marked in Figure 5.3. The exact wiring can be found in the respective diagrams for the power supply (Figure 5.1) as well as the motor diagram (Figure 5.2). All proposed wiring configurations can be found in the datasheets referenced in Section 5 and must be verified.

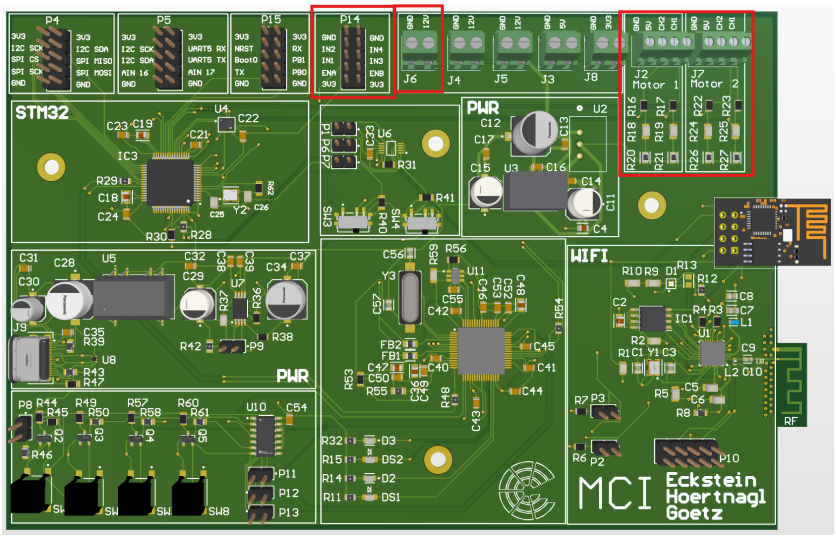


Figure 5.3: PCB Motors

6 Operation

The board can be operated both via software and via several hardware push buttons. These allow the reset and boot modes of the integrated microcontrollers to be selected. An ESP8266 and an STM32 are located on the board, each of which can be controlled and programmed independently.

6.1 SOFTWARE

Depending on the microcontroller, the software is flashed via different interfaces. The STM32 can either be programmed via the internal bootloader (UART) or via a debugger. For this purpose, the required pins for SWM were routed out during the initial assembly. The ESP8266 is typically flashed via the serial UART interface. Selection of the respective flash mode is performed via hardware buttons and UART switches or later via software.

6.2 MANUAL OPERATION

Several push buttons are available for hardware operation, allowing the reset and boot states of the microcontrollers to be set. The following sections describe the function of these buttons and their effects on the respective modes.

6.2.1 ESP8266 Buttons SW5 (GPIO0) & SW6 (EN/Reset)

The ESP8266 uses the state of the GPIO0 pin in combination with Enable to distinguish between normal operation and bootloader mode.

Table 6.1: Boot and reset modes of the ESP8266

SW5 (GPIO0)	SW6 (EN)	Result	Description
not pressed	not pressed	Normal start	ESP8266 boots firmware from flash memory.
not pressed	briefly pressed	Restart	Reset of the ESP8266, then normal boot from flash.
pressed	briefly pressed	Flash mode	GPIO0 is low during reset, the ESP switches to the UART bootloader and can be programmed.
pressed	not pressed	No change	-

6.2.2 STM32 Buttons SW7 (BOOT0) & SW8 (NRST)

The STM uses the state of the BOOT0 pin in combination with a reset to distinguish between normal operation and bootloader mode.

Table 6.2: Boot and reset modes of the STM32

SW7 (BOOT0)	SW8 (NRST)	Result	Description
not pressed	not pressed	Normal start	The STM32 starts the program stored in flash.
not pressed	briefly pressed	Normal reset	Restart of the microcontroller, boot from flash.
pressed	briefly pressed	Bootloader mode	BOOT0 is high during reset, the STM32 starts the bootloader.
pressed	not pressed	No change	-

6.2.3 Default State Without Button Press

If no buttons are pressed during power-up or operation, the board is in the normal state. Both microcontrollers start the respective installed firmware from the internal flash memory.

Table 6.3: System state without user interaction

Component	State	Result
ESP8266	GPIO0 high, EN high	Normal boot from flash
STM32	BOOT0 low, NRST high	Normal boot from flash
UART-SW3/SW4	Mechanical	According to switch position

6.2.4 UART Switches (SW3 & SW4)

The UART switches SW3 and SW4 define the routing of the ESP UART. Independently of this, the STM32 is permanently connected to the FTDI interface via the USB-C connector.

Table 6.4: UART switching configurations (ESP UART)

SW3	**SW4**	**UART connection**	**Usage**
Auto	Low	STM32 ↔ ESP	Standard operation: communication between STM32 and ESP.
Auto	High	FTDI/USB ↔ ESP	Flashing the ESP via USB.
Low	Low	forced STM32 ↔ ESP	Manually fixed connection between STM32 and ESP.
High	High	forced FTDI ↔ ESP	Manual connection between FTDI and ESP.

Note: The STM32 is permanently connected to the FTDI via USB-C and can be programmed and debugged independently of SW3 and SW4.

7 Jumpers

7.1 UART CONNECTION ESP

Jumpers P1 and P6 are used to switch the UART communication lines of the ESP. P1 is the TX jumper and affects the transmit line, while P6 controls the RX line. Together, they allow the ESP to be operated either via the integrated USB-UART interface or via an external UART programming adapter.

Table 7.1: Jumpers P1 and P6 – UART signal switching of the ESP

Jumper	Signal	Function and effect
P1	ESP_TX	P1 switches the TX line of the ESP between the internal USB-UART interface (FTDI) and an external signal source.
P6	ESP_RX	P6 switches the RX line of the ESP between the internal USB-UART interface (FTDI) and an alternative or external signal source.

Note: In normal operation, the jumper must be closed.

7.2 BOOTLOADER ACTIVATION: AUTOMATIC AND MANUAL OPTIONS

For automatic operation, jumpers P11, P12, and P13 must be closed via software. Jumper P8 is controlled directly by the FTDI chip and does not require inversion. It must also be closed for automatic operation.

Table 7.2: Function and state of jumpers P8, P11–P13

Jumper	Signal	Closed	Open
P8	BOOT0 STM	Boot level can be controlled via software.	Start in normal operation, boot pin is manually controlled via push button SW5
P11	NRST STM	Enable or reset signal is controlled via software. STM32 is reliably released or reset.	Start in normal operation, reset pin is manually controlled via push button SW6
P12	GPIO0 ESP	Boot level is controlled via software. ESP: flash mode possible (GPIO0 = low during reset).	Start in normal operation, pin is manually controlled via push button SW7
P13	EN ESP	The reset pin can be controlled via software.	Reset pin is manually controlled via push button SW8

Jumpers P8 as well as P11 to P13 together form the central reset and boot control logic of the board. They define whether the reset and boot signals for the ESP and STM32 are generated automatically by the onboard logic or whether they are disconnected from it and must be actuated manually.

Note: When closed, the jumpers enable automatic flashing and resetting of the microcontrollers.

7.3 ONBOARD LOGIC FOR RESET AND BOOT (ESP & STM32)

The board features an onboard logic 7.1 for automatic control of the reset and boot signals of the ESP and STM32. The goal is to enable flashing via USB without manual button sequences and to ensure reproducible startup states.

The logic evaluates the UART control signals of the USB-UART interface (DTR/RTS). Since these signals are not directly suitable for the microcontrollers in terms of polarity and electrical characteristics, they are inverted, decoupled, and temporally conditioned by a logic component.

The control of the signals is performed on the software side. However, the required software for targeted control of DTR and RTS still needs to be implemented. Subsequently, the onboard logic converts these signals into valid reset and boot signals for the microcontrollers.

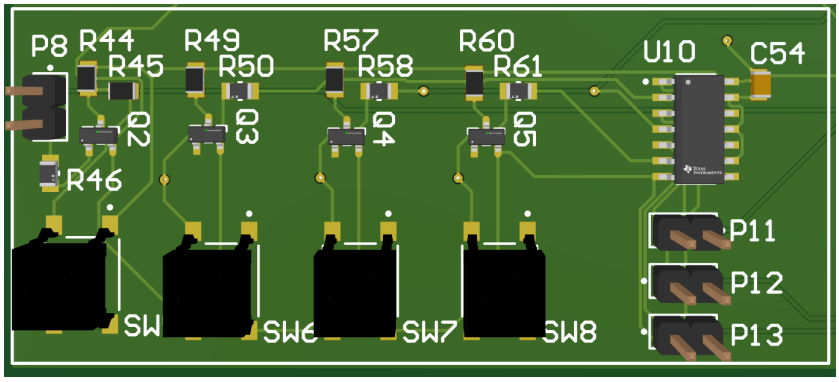


Figure 7.1: Onboard logic

Note: The FTDI chip is controlled by software, the onboard logic conditions the signals, and the microcontrollers switch the operating mode.

8 Design and Assembly

8.1 HOUSING

To simplify the project, the second mounting position of the battery was removed. However, by using aluminum profiles with dimensions of 20 x 20 mm, this position can be added again at any time.

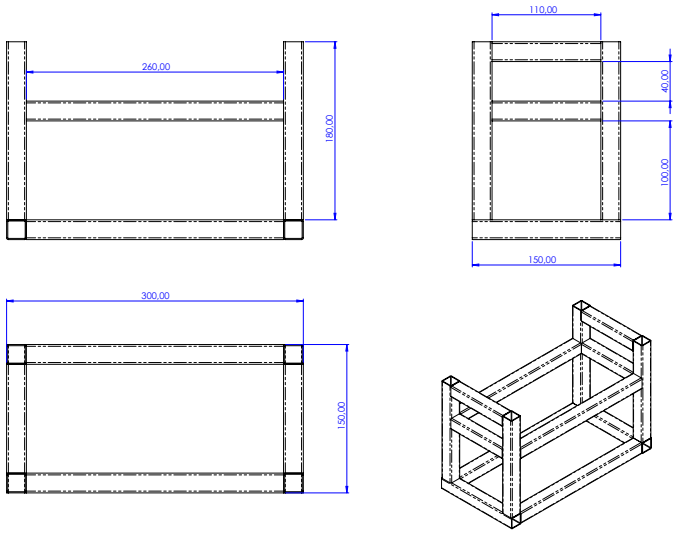


Figure 8.1: Frame

The individual cut profiles were connected using the system-specific fastening system. This allowed the connecting screws to be cleanly integrated and hidden. In addition, the aluminum profile leaves room for further improvements as well as a clean option for cable routing along the profiles.

8.2 MOTOR MOUNTS

The motor mounts were manufactured using a 3D printing process. Additive manufacturing enabled a targeted design that is perfectly adapted to the mechanical requirements.

The mount encloses the motors and takes into account the geometric pin of the motor shaft, which also serves as an anti-rotation feature. The enclosure was designed with a slot in order to clamp the motor using an M4x20 socket head screw.

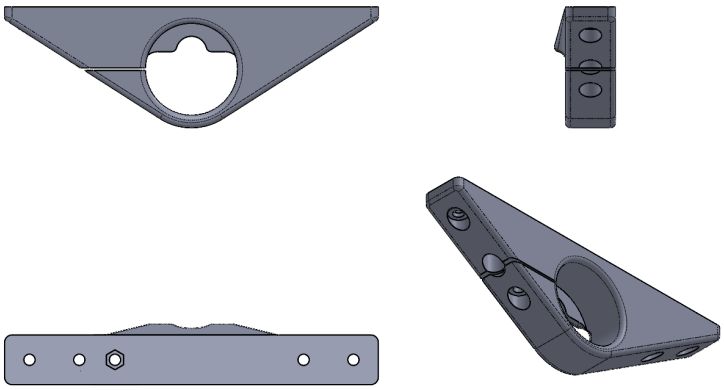


Figure 8.2: Motor mounts

The mount itself is attached to the frame using four M4 screws. The system-specific T-slot nuts of the profile manufacturer are used as nuts. This ensures a robust connection in the event of an impact with the ground of the demonstrator.

8.3 ELECTRONICS INSTALLATION LEVEL

The mount for the electronic components was manufactured using a 3D printing process. This allowed individual positioning of the components to be implemented very easily. The underside of the level was reinforced with ribs.

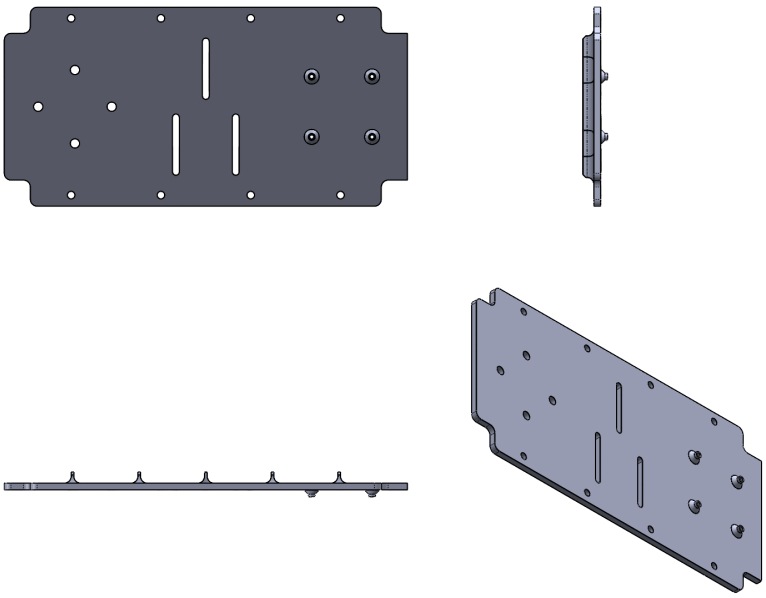


Figure 8.3: Electronics level

The battery holder was provided with elongated holes. This serves to adjust the center of mass of the battery and to align it centrally with respect to the frame. Through this variable positioning, imbalances in the overall structure can be compensated by correcting the position of the battery.

The four through-holes are used to mount the DC/DC converter. Two holes are required for installation. The unused mounting holes can be used for an alternative position of the converter. Thus, the converter can be rotated if necessary, for example if required by the wiring.

8.4 PCB INSTALLATION LEVEL

The mount was designed similarly to the electronics installation level. The four standoffs mirror the mounting points of the PCB. By removing the alternative battery position, easy access to the PCB is also ensured.

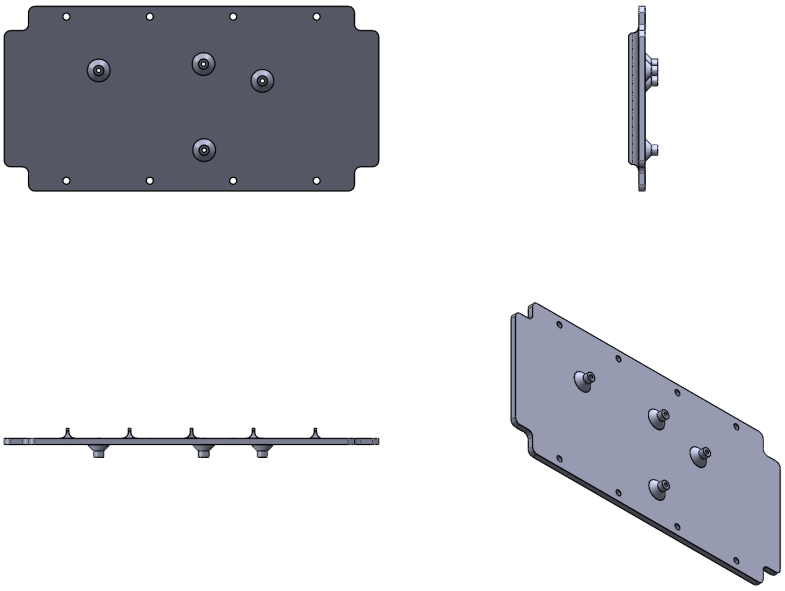


Figure 8.4: PCB level

The underside of the mount was also designed with reinforcing ribs. The material thicknesses and the material used (PETG) are sufficient for the mechanical loads that occur.

8.5 WHEELS

The wheels were manufactured using additive manufacturing, just like the installation levels. The wheel mount was designed to match the motor shaft. The tolerances were selected such that the wheel and motor shaft are connected via an interference fit.

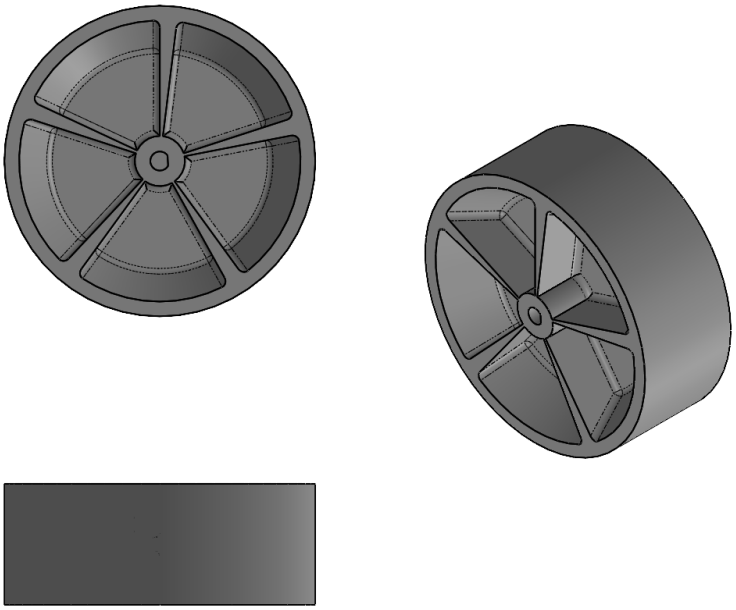


Figure 8.5: Wheels

Both the mechanical connection between the wheel and the motor shaft and the mechanical loads are sufficient for operation. PLA was selected as the material for the wheels, as it offers sufficient mechanical strength limits for this application.

9 Further Procedure and Open Points

Within the scope of the work carried out so far, a functional demonstrator has been built, which, however, does not yet include any software. In addition, not all functions have been thoroughly tested. For stable operation and further development of the system, there are still optimization opportunities on both the mechanical and electronic levels :contentReference[oaicite:0]index=0.

9.1 HARDWARE IMPROVEMENTS

The mechanical design of the setup is generally functional, but offers potential for improvement with regard to stability, maintainability, and cable routing.

1. Implementation of the wiring.
2. To improve traction and reduce slip, rubber elements were added to the wheels.
3. The frame can be extended by a second level in order to spatially separate electronics and mechanics and to create an alternative battery position (optional).
4. Damping elements are integrated to prevent or reduce the system tipping over in unstable control states.
5. Cable holders for the ITEM profiles are retrofitted to enable structured cable routing.

9.2 PCB IMPROVEMENTS

There are also several starting points in the PCB design to simplify commissioning and increase robustness.

1. Conversion to a ready-made WiFi module, such as the ESP8684-MINI-1-H4 [11], in order to use a more modern, compact, and better-supported WLAN platform.
2. Removal of the existing signal switching for the ESP and replacement with a dedicated programming adapter with externally routed pins.
3. Addition of test points for relevant signals (UART, reset, supply voltage) to facilitate debugging and troubleshooting.
4. Routing the missing pin for the SWD connection of the ST-Link to a dedicated header.
5. Simplification of the voltage conversions, for example by using the Pololu 3.3V, 1A Step-Down Voltage Regulator D24V10F3 [12].
6. Integration of solder bridges, jumpers, and 0 Ω resistors to allow sectional commissioning of the PCB.
7. Correction of the voltage dividers from the encoders to the STM32. The expected 3.2 V is not present at the STM32, only 1.8 V. Different resistors must be installed here.

The points listed represent reasonable next steps to make the demonstrator more robust, maintenance-friendly, and better expandable for both lecture use and further development by subsequent project groups.

As described in the list, the section converting 5 V to 3.3 V can be significantly simplified by using modules.

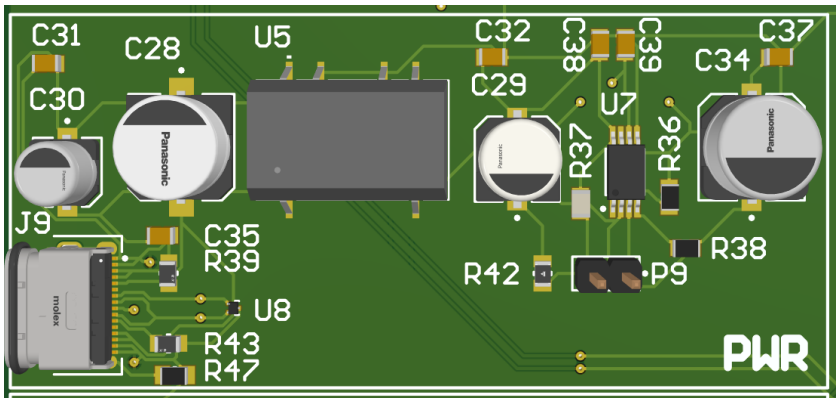


Figure 9.1: 5 V to 3.3 V

The same module can also be used in the 12 V to 3.3 V section and would allow the focus to be placed on the remaining parts of the PCB.

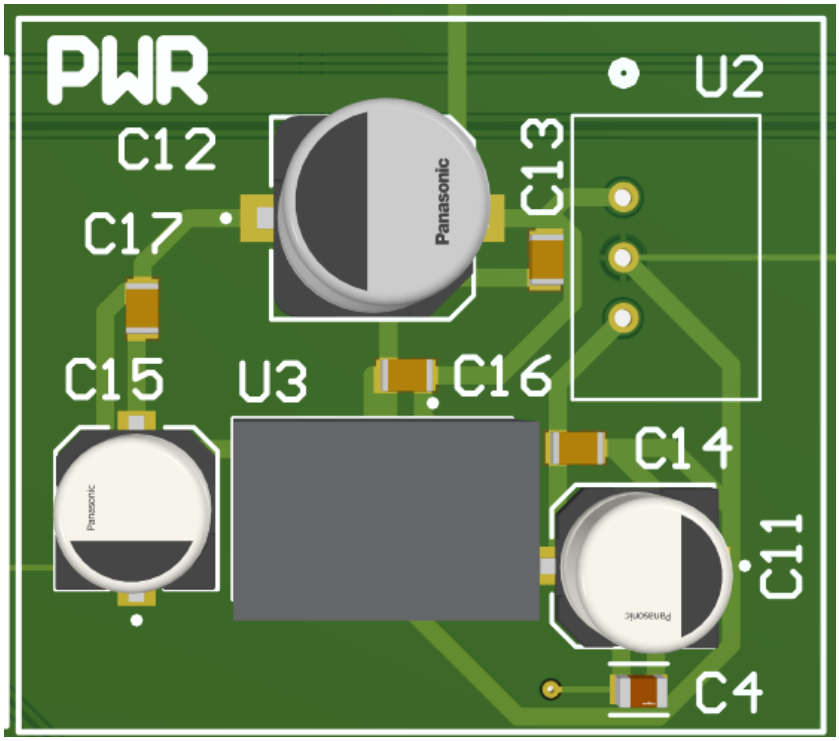


Figure 9.2: 12 V to 3.3 V

The UART switching between the ESP, STM, and FTDI chips can be removed by routing the required pins for external programming of the ESP to a dedicated header. As a rule, the WLAN module only needs to be programmed during functional tests and no longer requires the ability to switch the communication connection once the software has been correctly configured.

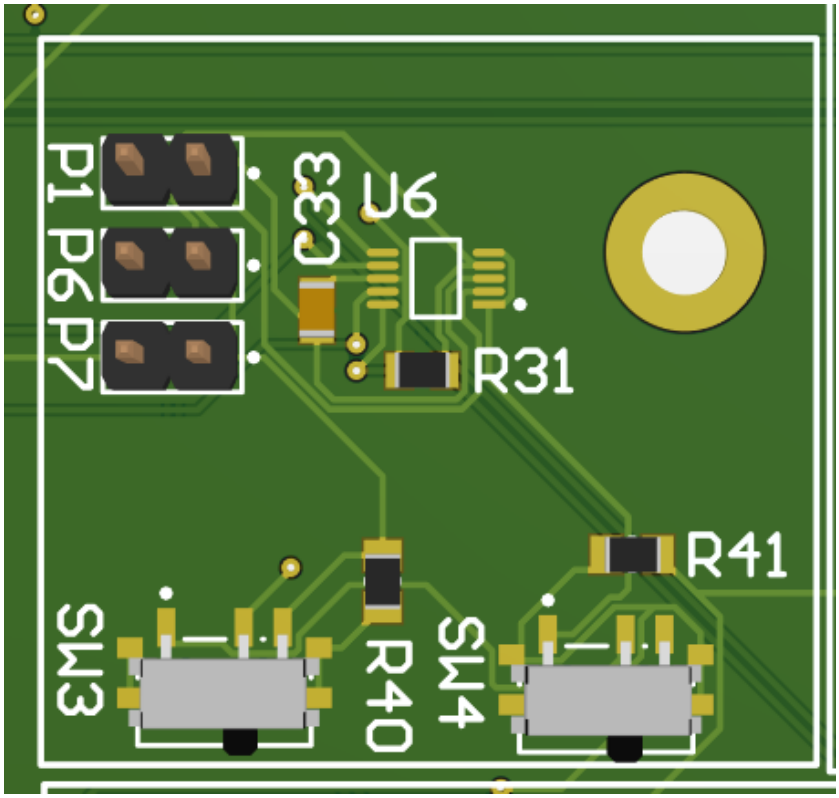


Figure 9.3: UART switching

The entire WLAN module including the PCB antenna should be significantly simplified. For this purpose, ready-made modules such as the ESP8684-MINI-1-H4 [11] can be used. This is also recommended by the manufacturer and additionally offers the expansion to Bluetooth.

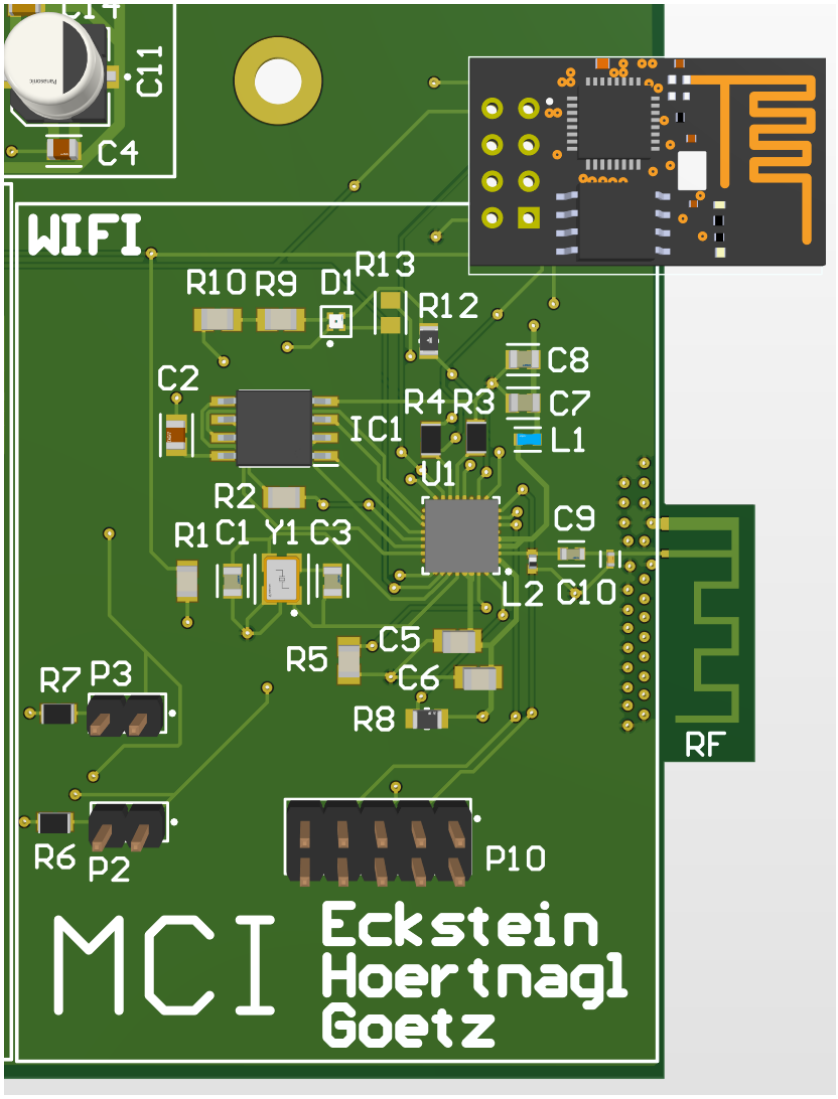


Figure 9.4: WiFi section

9.3 FURTHER PROCEDURE FOR THE SOFTWARE

Independent of the hardware-related adjustments, the development of the overall software is a central step in making the demonstrator fully operational. The software forms the interface between control algorithms, sensors, and actuators and is therefore of decisive importance for the functional operation of the system.

First, stable firmware must be implemented for the microcontrollers, handling basic tasks such as hardware initialization, sensor acquisition, and actuator control. Building on this, a communication interface must be realized to connect the demonstrator to an external computer.

A key objective is the connection to MATLAB in order to transfer control parameters or complete control structures from the PC and adjust them during operation. In addition, measurement data, such as position or velocity information, are to be continuously sent back to MATLAB to enable live analysis and visualization.

Finally, tests and validations are required to ensure the reliability of the software as well as the interaction between hardware and software under real operating conditions.

List of Figures

List of Tables

References

[1] L. G. Hoertnagl Stefan, Maximilian Eckstein, "Elektronische produktentwicklung," Dokumentation, Abschlussbericht, Bericht Inverses Pendel, 2025, begleitdokumentation.

[2] DeepL SE. (2025) Deepl translator. DeepL SE. Online translation service, accessed on January 3, 2026. [Online]. Available: https://www.deepl.com/translator

[3] S. Mci, "Inverted pendulum – electronic product development," https://github.com/Stefmci/INV_Pendel_Elektronische_Produktentwicklung, 2024, gitHub repository.

[4] CK Switshes, *PCM Series Ultraminiature Surface Mount Slide Switches*, CK Switshes, 2022, zugriff: 27.12.2025. [Online]. Available: https://datasheet.octopart.com/PCM12SMTR-C%26K-Components-datasheet-175508494.pdf?src-supplier=Newark

[5] L. G. Hoertnagl Stefan, Maximilian Eckstein, *Schaltplan Inverses Pendel*, 2025, eigene Schaltungsentwicklung, interne Dokumentation.

[6] Makita, *BL1840B Li-Ion Akku 18V 4.0Ah*, Makita, 2025, lXT Lithium-Ionen Akku. [Online]. Available: https://www.makita.at/product/bl1840b.html

[7] Unbekannt, *DC-DC Step-Down Wandler 18V auf 5V/3.3V*, 2023, schaltregler zur Spannungsreduktion für Mikrocontroller und Peripherie. [Online]. Available: https://bauer-united.com/products/dc-dc-18v-36v-zu-12v-10a-spannungswandler

[8] DFRobot, *7A Dual DC Motor Driver*, DFRobot, 2016, sKU DRI0041. [Online]. Available: https://wiki.dfrobot.com/7A_Dual_DC_Motor_Driver_SKU__DRI0041

[9] ——, *Metal DC Geared Motor with Encoder 12V 122RPM*, DFRobot, 2024, gB37Y3530-12V-90EN. [Online]. Available: https://www.dfrobot.com/product-1210.html

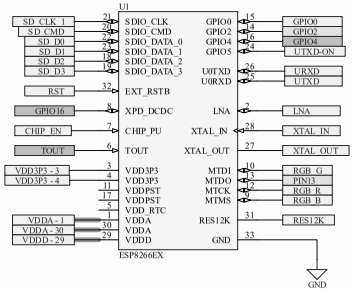
[10] QUPERR, *Akku-Adapter fuer 18V Makita LXT Akkus*, QUPERR, 2023, akku-Aufnahme mit Sicherung und Anschlusskabeln für Makita LXT Akkus. [Online]. Available: https://www.amazon.de/QUPERR-Sicherung-Akku-Adapter-Lithium-Akku-Power-Converter/dp/B0BPRYRGVF

[11] Digi-Key Electronics, "Esp8684-mini-1-h4 wlan/bluetooth modul," https://www.digikey.at/de/products/detail/espressif-systems/ESP8684-MINI-1-H4/21572303, 2025, zugriff am 29.12.2025.

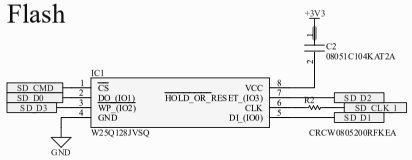
[12] Pololu Corporation, "3.3v, 500ma step-down spannungsregler d24v5f3," 2025, zugriff am 29.12.2025. [Online]. Available: https://www.pololu.com/product/2830

A Schematics and printed circuit board

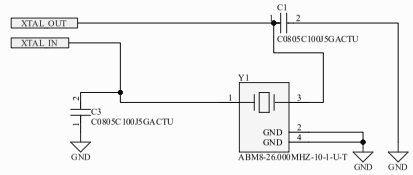
ESP8266



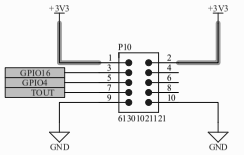
Flash



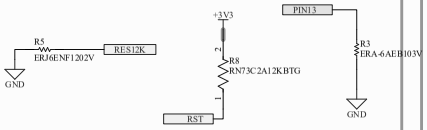
Oszi



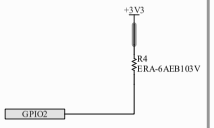
Breakout



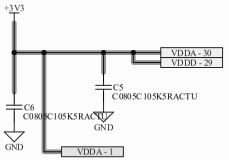
Kleinteile



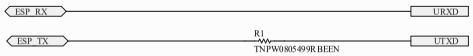
GPIO2 High bei Boot



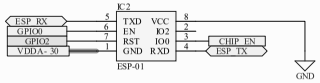
Power



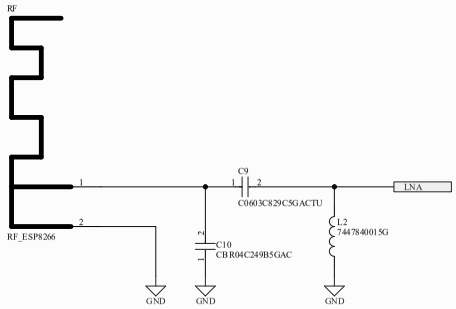
UART



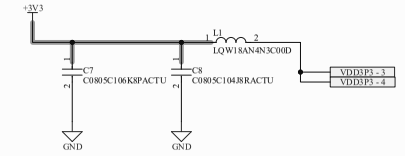
Plan B



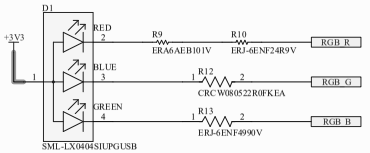
RF



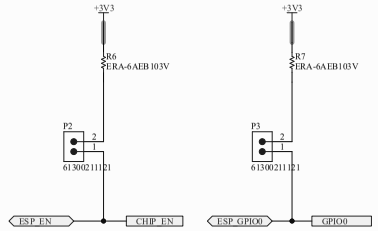
Power PA&LNA



LED



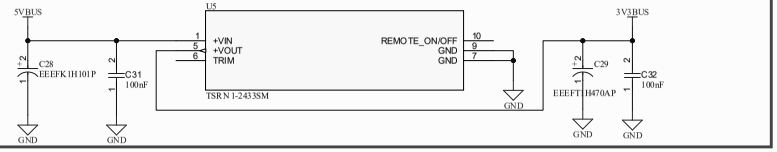
Jumper



Title		
Size: A2	Number	Revision
Date: 12.10.2025		Sheet of
File: C:\Users\...\ESP8266_final.SchDoc		Drawn By:

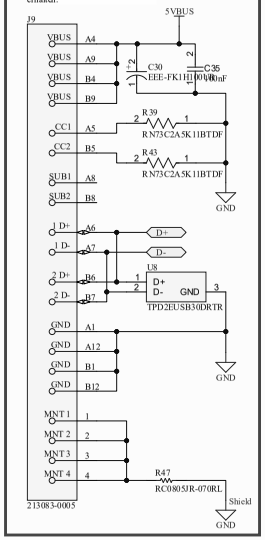
DC-DC Konverter 5V USB-Spannung auf 3V3

Es muss ein sehr leistungsstarker DC/DC Wandler verwendet werden, damit auch mit den 5V Eingangsspannung 1A am Ausgang ausgegeben werden kann.



USB-C Schnittstelle

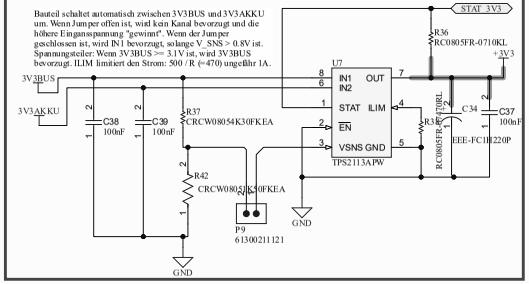
USB-C-Schnittstelle wurde anstelle der Mini-USB-Schnittstelle verwendet, um 5V @1.5A zu erhalten.



Auto-switching Power MUX

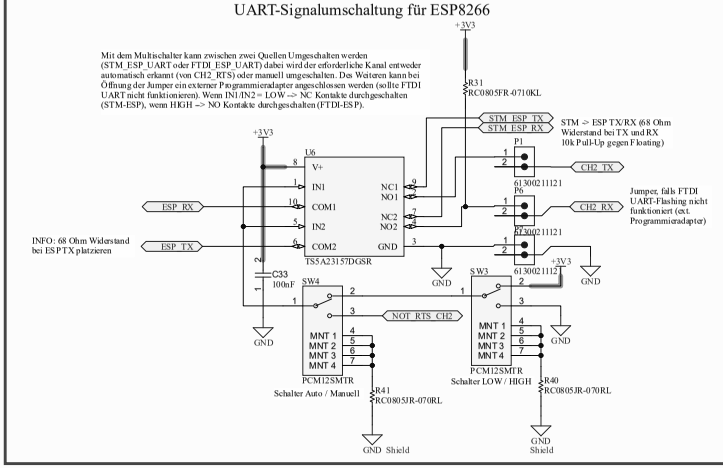
Bauteil schaltet automatisch zwischen 3V3BUS und 3V3AKKU um. Wenn Jumper offen ist, wird kein Kanal bevorzugt und die höhere Eingangsspannung "gewinnt". Wenn der Jumper geschlossen ist, wird IN1 bevorzugt, solange V_SNS > 0.8V ist. Spannungsteiler: Wenn 3V3BUS >= 3.1V ist, wird 3V3BUS bevorzugt. ILIM limitiert den Strom: 500 / R (=470) ungefähr 1A.

STAT_3V3 ist HIGH, wenn IN2 (Akku) Kanal aktiv ist.



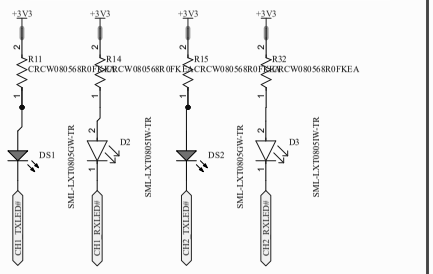
UART-Signalumschaltung für ESP8266

Mit dem Multischalter kann zwischen zwei Quellen Umgeschalten werden (STM_ESP_UART oder FTDI_ESP_UART) dabei wird der erforderliche Kanal entweder automatisch erkannt (von CH2_RTS) oder manuell umgeschalten. Das Weiteren kann bei Öffnung der Jumper ein externer Programmieradapter angeschlossen werden (sollte FTDI UART nicht funktionieren). Wenn IN1/IN2 = LOW --> NC Kontakte durchgeschalten (STM-ESP), wenn HIGH --> NO Kontakte durchgeschalten (FTDI-ESP).

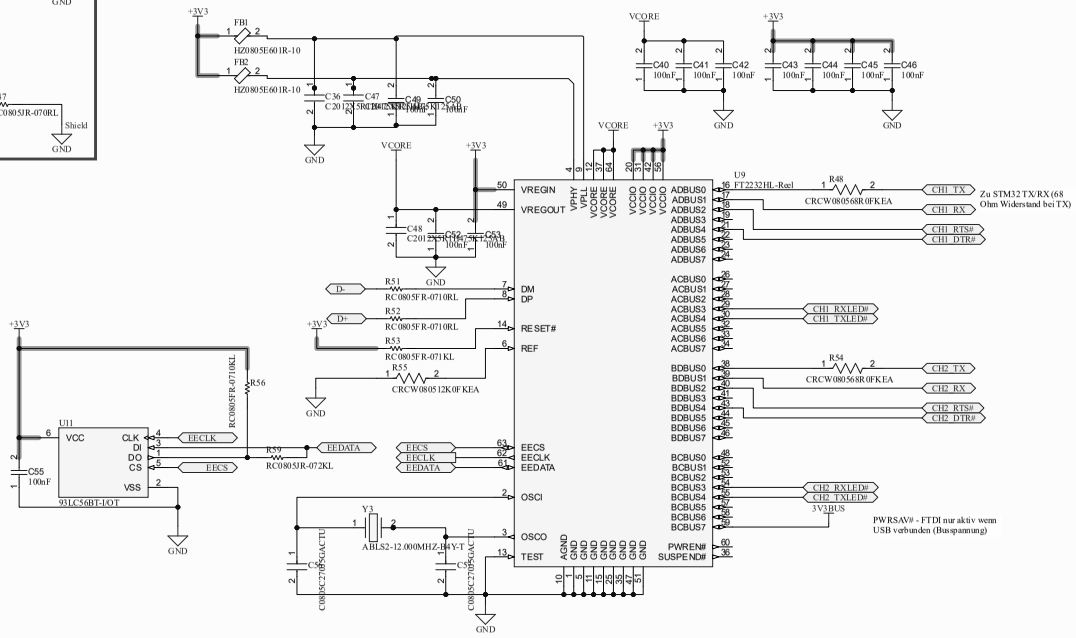


Status LEDs (UART Signal und 3V3 Akku IN2)

LEDs blinken bei UART Kommunikation (TX = Green, RX = Red)
Wenn 3V3 von Akku stammen (STAT_3V3 auf High gezogen) --> Rote LED an zur Statusanzeige



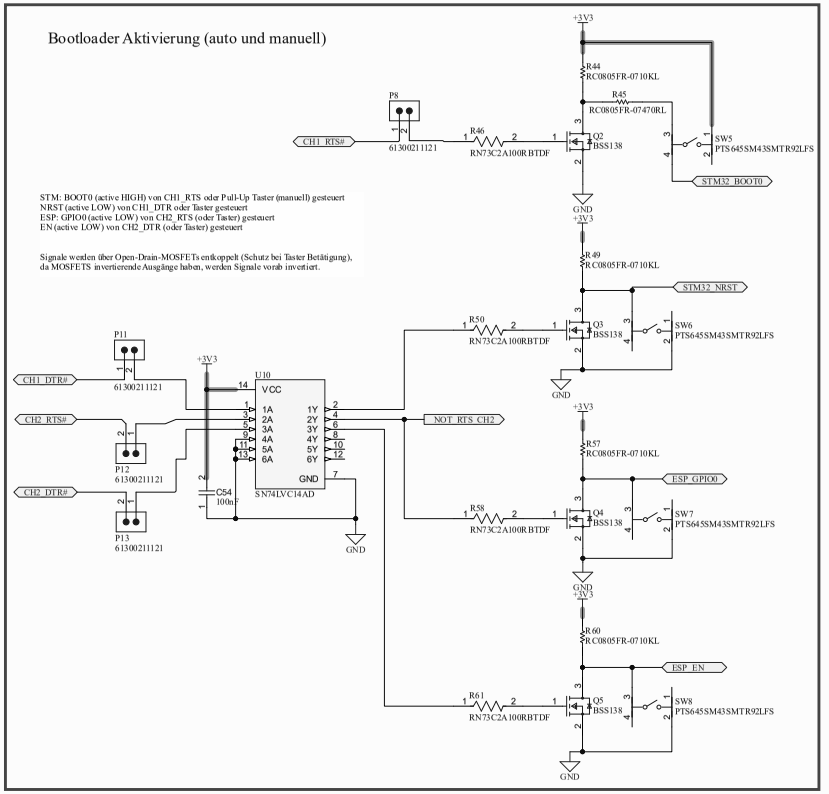
FTDI: USB-TTL/UART mit EEPROM

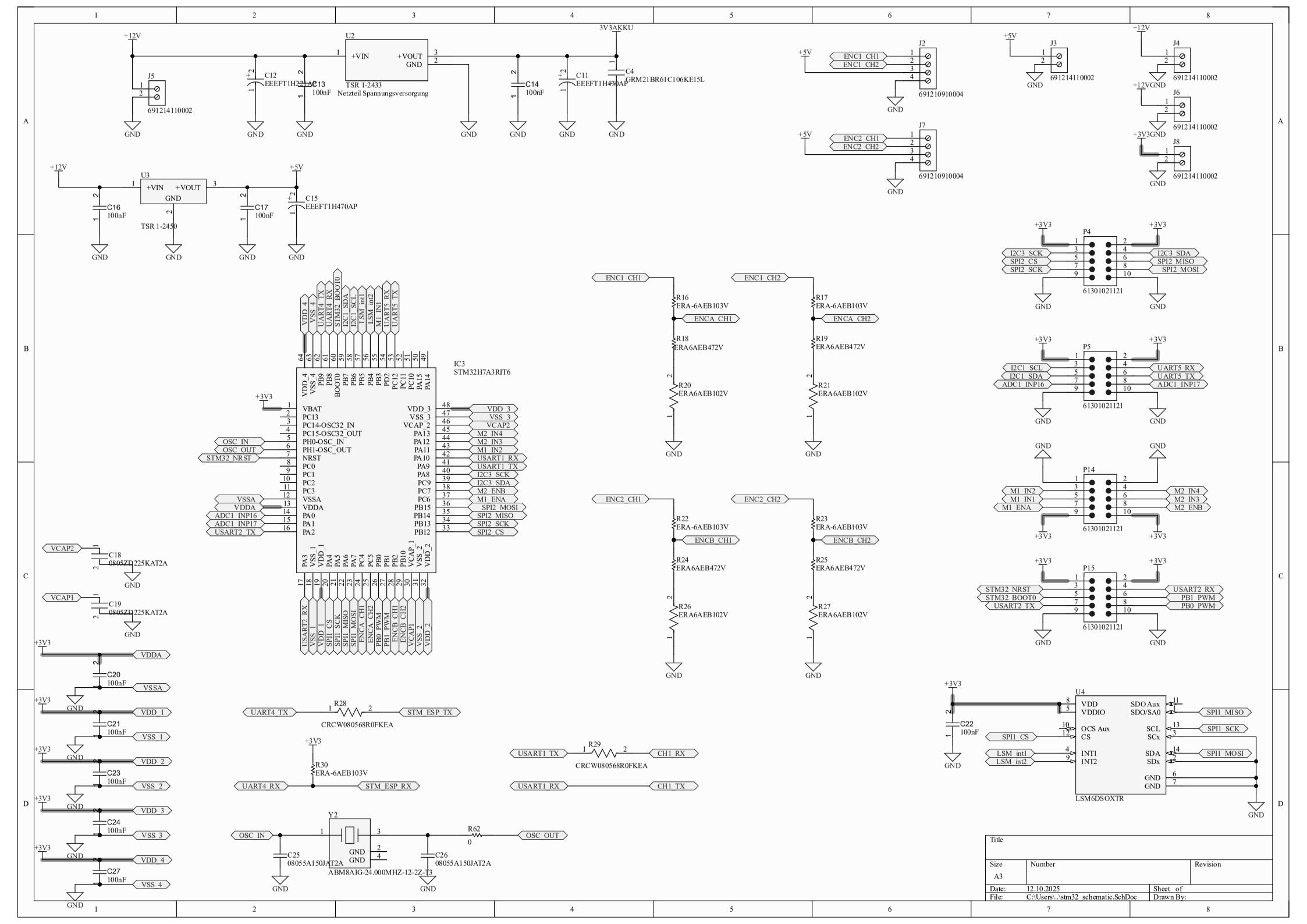


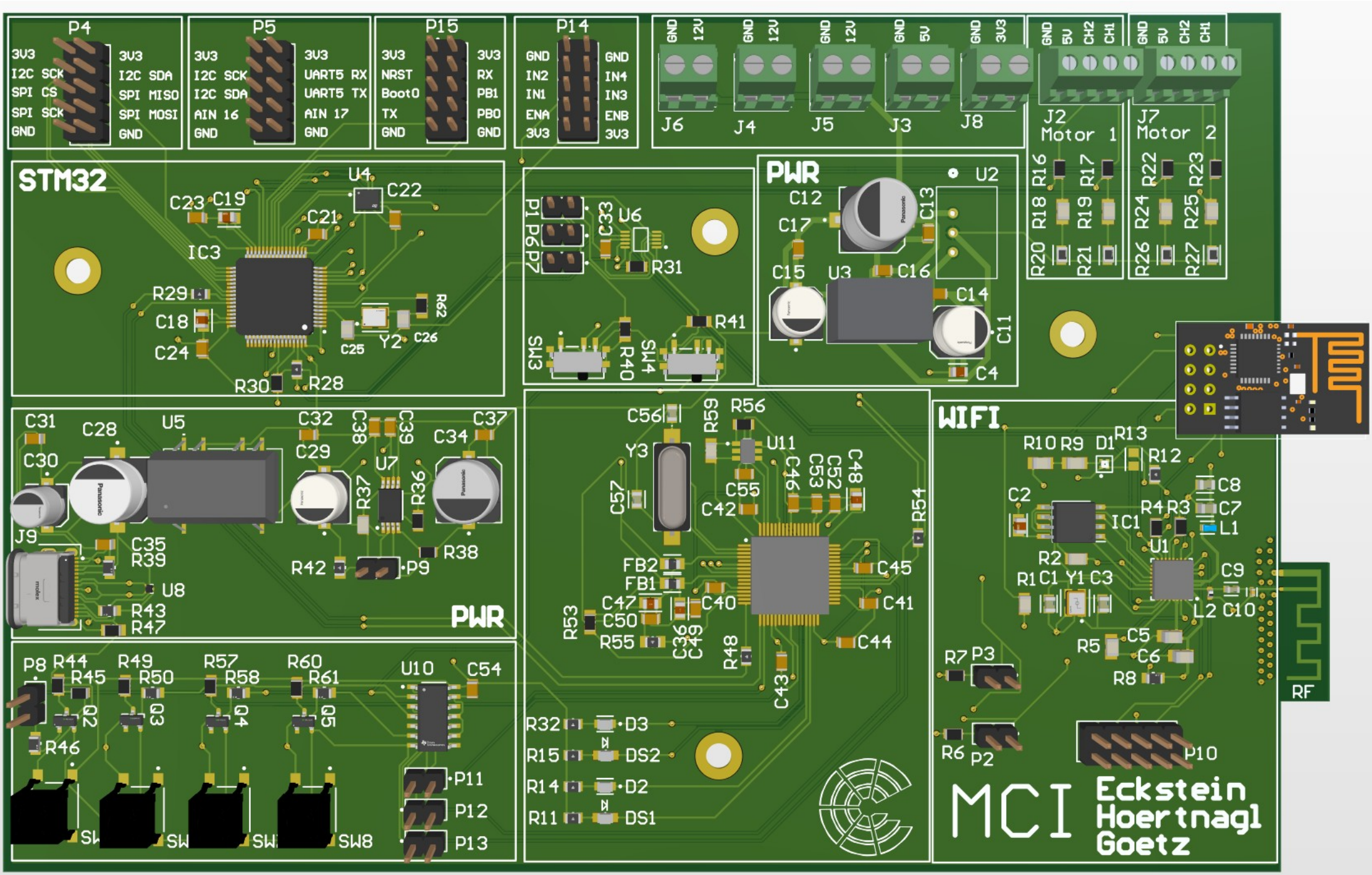
Bootloader Aktivierung (auto und manuell)

STM: BOOT0 (active HIGH) von CH1_RTS oder Pull-Up Taster (manuell) gesteuert
NRST (active LOW) von CH1_DTR oder Taster gesteuert
ESP_GPIO0 (active LOW) von CH2_RTS (oder Taster) gesteuert
EN (active LOW) von CH2_DTR (oder Taster) gesteuert

Signale werden über Open-Drain-MOSFETs entkoppelt (Schutz bei Taster Betätigung), da MOSFETS invertierende Ausgänge haben, werden Signale vorab invertiert.




+12V
U2
+VIN
+VOUT
GND
TSR 1-2433
Netzteil Spannungsversorgung
3V3AKKU
J5
691214110002
C12
EEEFT1H221AP
C13
100nF
C14
100nF
C11
EEEFT1H470AP
C4
GRM21BR61C106KE15L
GND
+5V
J2
ENC1_CH1
ENC1_CH2
691210910004
J3
691214110002
J4
J6
J7
ENC2_CH1
ENC2_CH2
J8
+3V3GND
U3
TSR 1-2450
C16
100nF
C17
100nF
C15
EEEFT1H470AP
IC3
STM32H7A3RIT6
VBAT
PC13
PC14-OSC32_IN
PC15-OSC32_OUT
PH0-OSC_IN
PH1-OSC_OUT
NRST
PC0
PC1
PC2
PC3
VSSA
VDDA
PA0
PA1
PA2
OSC_IN
OSC_OUT
STM32_NRST
ADC1_INP16
ADC1_INP17
USART2_TX
VDD_3
VSS_3
VCAP2
PA13
PA12
PA11
PA10
PA9
PA8
PC9
PC7
PC6
PB15
PB14
PB13
PB12
M2_IN4
M2_IN3
M1_IN2
USART1_RX
USART1_TX
I2C3_SCK
I2C3_SDA
M2_ENB
M1_ENA
SPI2_MOSI
SPI2_MISO
SPI2_SCK
SPI2_CS
P4
P5
P14
P15
61301021121
I2C3_SDA
I2C1_SCL
I2C1_SDA
UART5_RX
UART5_TX
ADC1_INP17
M1_IN2
M1_IN1
M1_ENA
M2_IN4
M2_IN3
M2_ENB
STM32_NRST
STM32_BOOT0
USART2_TX
PB1_PWM
PB0_PWM
ENC1_CH1
ENC1_CH2
R16
ERA-6AEB103V
R17
ERA-6AEB103V
ENCA_CH1
ENCA_CH2
R18
ERA6AEB472V
R19
ERA6AEB472V
R20
ERA6AEB102V
R21
ERA6AEB102V
ENC2_CH1
ENC2_CH2
R22
ERA-6AEB103V
R23
ERA-6AEB103V
ENCB_CH1
ENCB_CH2
R24
ERA6AEB472V
R25
ERA6AEB472V
R26
ERA6AEB102V
R27
ERA6AEB102V
VCAP2
C18
0805ZD225KAT2A
VCAP1
C19
0805ZD225KAT2A
VDDA
C20
100nF
VSSA
VDD_1
C21
100nF
VSS_1
VDD_2
C23
100nF
VSS_2
VDD_3
C24
100nF
VSS_3
VDD_4
C27
100nF
VSS_4
UART4_TX
R28
CRCW080568R0FKEA
STM_ESP_TX
R30
ERA-6AEB103V
UART4_RX
STM_ESP_RX
USART1_TX
R29
CRCW080568R0FKEA
CH1_RX
USART1_RX
CH1_TX
Y2
ABM8AIG-24.000MHZ-12-2Z-T3
C25
08055A150JAT2A
C26
08055A150JAT2A
R62
0
OSC_IN
OSC_OUT
U4
LSM6DSOXTR
C22
100nF
VDD
VDDIO
GND
OCS Aux
CS
SCL
SCx
SDA
SDx
INT1
INT2
SDO Aux
SDO/SA0
SPI1_MISO
SPI1_SCK
SPI1_CS
SPI1_MOSI
LSM_int1
LSM_int2
Title
Size
A3
Number
Revision
Date:
12.10.2025
Sheet of
File:
C:\Users\..\stm32_schematic.SchDoc
Drawn By:


P4
3V3
I2C SCK
SPI CS
SPI SCK
GND
3V3
I2C SDA
SPI MISO
SPI MOSI
GND
P5
3V3
I2C SCK
I2C SDA
AIN 16
GND
3V3
UART5 RX
UART5 TX
AIN 17
GND
P15
3V3
NRST
Boot0
TX
GND
3V3
RX
PB1
PB0
GND
P14
GND
IN2
IN1
ENA
3V3
GND
IN4
IN3
ENB
3V3
GND 12V
J6
GND 12V
J4
GND 12V
J5
GND 5V
J3
GND 3V3
J8
GND 5V CH2 CH1
J2
Motor 1
GND 5V CH2 CH1
J7
Motor 2
STM32
IC3
U4
C22
C23
C19
C21
R29
C18
C24
R30
R28
C25
Y2
C26
R62
P1
P6
P2
C33
U6
R31
SW3
R40
SW4
R41
PWR
U2
C12
C17
C13
C15
U3
C16
C14
C11
C4
R16
R17
R18
R19
R20
R21
R22
R23
R24
R25
R26
R27
C31
C28
U5
C32
C38
C39
C37
C34
C30
C29
U7
R37
R36
J9
C35
R39
U8
R43
R47
R42
P9
R38
PWR
C56
R59
R56
U11
Y3
C57
C55
C46
C53
C52
C48
C42
R54
FB2
FB1
C47
C50
R55
R53
C36
C49
C40
R48
C45
C41
C44
C43
R32
D3
R15
DS2
R14
D2
R11
DS1
WIFI
R10
R9
D1
R13
R12
C2
IC1
R4
R3
C8
C7
L1
U1
R2
R1
C1
Y1
C3
C9
L2
C10
R5
C5
C6
R8
R7
P3
R6
P2
P10
RF
P8
R44
R45
R49
R50
R57
R58
R60
R61
U10
C54
Q2
Q3
Q4
Q5
R46
P11
P12
P13
SW
SW
SW
SW8
MCI
Eckstein
Hoertnagl
Goetz