

Projekt-Bericht

Isabell Albrecht, Erik Engelhardt, Oliver Kochan, Florian Steffens

Solarbetriebene, mobile Wetterstation

Betreuung durch: Prof. Dr. Franz Schubert
Eingereicht am: 14. Januar 2019

*Fakultät Technik und Informatik
Department Informations- und Elektrotechnik*

*Faculty of Computer Science and Engineering
Department Information and Electrical Engineering*

Inhaltsverzeichnis

1 Einführung	1
1.1 Die Wetterstation	1
1.2 Anforderungen	2
1.3 Vorüberlegungen	2
2 Spannungsversorgung	3
2.1 Anforderungen	3
2.2 Power-Board	3
2.3 Sensor-Board	5
3 Sensoren	6
3.1 Kompassmodul QMC5883L	6
3.1.1 Montage	7
3.2 GPS	7
3.2.1 Montage	7
3.3 Anemometer und Windfahne	8
3.3.1 Montage	8
3.4 Neigungssensor MPU6050	8
3.4.1 Montage	8
3.5 BME280	9
3.5.1 Montage	9
3.6 Anschlagssensor	9
3.6.1 Montage	9
4 Mikrocontroller	9
4.1 Auswahl des Mikrocontrollersystems	10
4.2 Pin-Belegung	11
5 Firmware	13
5.1 Auswertung der Sensoren (<code>sensorlib</code>)	14
5.1.1 BME280	14
5.1.2 CPU-Temperatursensor	14
5.1.3 QMC5883L	14
5.1.4 MPU6050	14
5.1.5 Anemometer und Windfahne	14

5.1.6	Strom- und Spannungsmessung	14
5.2	GPS: NMEA 0183 Protokollverarbeitung	14
5.3	Bluetooth: Aktivierung und AT-Befehlssatz	14
6	Ausrichtung des Solarpanels	14
6.1	Berechnung der Sonnenposition	15
7	Benutzeroberfläche	16
7.1	Funktionen	16
7.2	Entwicklung	21
8	3D gedruckte Komponenten	24
8.1	Hauptgehäuse	25
8.2	Nebengehäuse	27
8.3	Adaptor	29
8.4	Herstellung	31
9	Fazit	32
Literatur		33

1 Einführung

Der folgende Bericht gibt einen Überblick über den Entwurf und die Umsetzung einer solarbetriebenen Wetterstation im Rahmen der Sensorikvorlesung. Hierbei sollen zunächst die Anforderungen und bereits vorhandene Konzepte/Komponenten kurz vorgestellt und dann sowohl auf die Soft- als auch die Hardwareseitige Umsetzung eingegangen werden.

1.1 Die Wetterstation



Abbildung 1: Gegebener Aufbau der Wetterstation

In Abbildung 1 ist der gegebene Aufbau der Wetterstation dargestellt. Dieser besteht aus einem kippbaren Solarpanel, das auf einer drehbaren Platte montiert ist. Sowohl Dreh- als auch Kippbewegung wird über jeweils einen Getriebemotor ermöglicht. Des Weiteren ist bereits eine Windfahne zur Erfassung der Windrichtung und -geschwindigkeit montiert.

1.2 Anforderungen

Die Wetterstation soll folgende Werte erfassen:

- Temperatur
- Luftdruck
- Luftfeuchte
- Höhe über NN
- Windgeschwindigkeit
- Windrichtung
- Standort

Weiterhin soll die Wetterstation über das Solarpanel mit Strom versorgt werden, wobei aber dieses durch einen 12V Akku gepuffert werden soll, um z.B. die Nachtstunden ohne Ausfall der Spannungsversorgung überbrücken zu können. Für eine optimale Energieausbeute soll das Panel abhängig vom Sonnenstand über die beiden Getriebemotoren ausgerichtet werden. Außerdem soll der Akkuzustand erfasst und die drahtlose Kommunikation mit einem PC ermöglicht werden. Schließlich sollen die erfassten Sensordaten auf einer SD-Karte gespeichert werden.

1.3 Vorüberlegungen

Bevor die beschriebenen Anforderungen umgesetzt werden, sind einige Vorüberlegungen und die Auswahl der geeigneten Sensorik nötig. Letztere werden ausführlich in Abschnitt 3 dargestellt. Die Drahtloskommunikation soll über ein Bluetooth-Modul realisiert und um eine eigene GUI, die die Darstellung aktueller Sensordaten auf einem PC ermöglicht, erweitert werden. Des Weiteren wird die bisher vorhandene Windfahne ausgetauscht und ersetzt, da diese nur über eine unzureichende Dokumentation verfügt. Ebenfalls ist eine wasserdichte Ausführung der Wetterstation wünschenswert, die Umsetzbarkeit muss überprüft werden. Um Energie zu sparen, soll die Ausrichtung der Wetterstation sowie die Abfrage der Sensordaten nicht kontinuierlich sondern in noch festzulegenden Zeitabständen erfolgen.

2 Spannungsversorgung

In diesem Abschnitt soll auf die hardwareseitige Umsetzung der Spannungsversorgung für die Wetterstation eingegangen werden. Ziel ist der Entwurf einer Platine, auf der sämtliche Anforderungen umgesetzt werden.

2.1 Anforderungen

Zunächst sollen in diesem Abschnitt die Anforderungen, die sich aus der Aufgabenstellung ableiten lassen, sowie solche, die sich aus den weiteren Überlegungen zur Umsetzung der Wetterstation ergeben.

- Messung des Ladestroms
- Messung der Batteriespannung (Ladezustand)
- Messung des Stromverbrauchs der Wetterstation

Des Weiteren soll der Stromverbrauch der Wetterstation so niedrig wie möglich sein, um die Puffer-Batterie zu schonen und sonnenarme Phasen bzw. die Nacht ohne Stromausfall überbrücken zu können. Die verwendete Batterie hat eine Ladeschlussspannung von 12 V. Da für den Mikrocontroller und die Sensoren allerdings Spannungspegel von 3,3 V und 5 V benötigt werden, müssen diese auf der Platine erzeugt werden.

Aus den mechanischen Anforderungen, dass Mikrocontroller, Platine und Sensoren möglichst in einem Gehäuse untergebracht werden sollen, ergibt sich, dass die entworfene Platine auf die Pinheader des Mikrocontrollers gesteckt werden soll. Auf Grund des Umfangs werden zwei Platinen Layoutet: Ein Powerboard zur Erzeugung der benötigten Spannungen und ein Sensor-Board, auf dem sich der Schaltungsteil für die Sensoren befindet.

2.2 Power-Board

In Abbildung 2 wird das finale Layout des Power-Boards gezeigt. Im folgenden sollen einige wichtige Details des Stromlaufplans dargestellt.

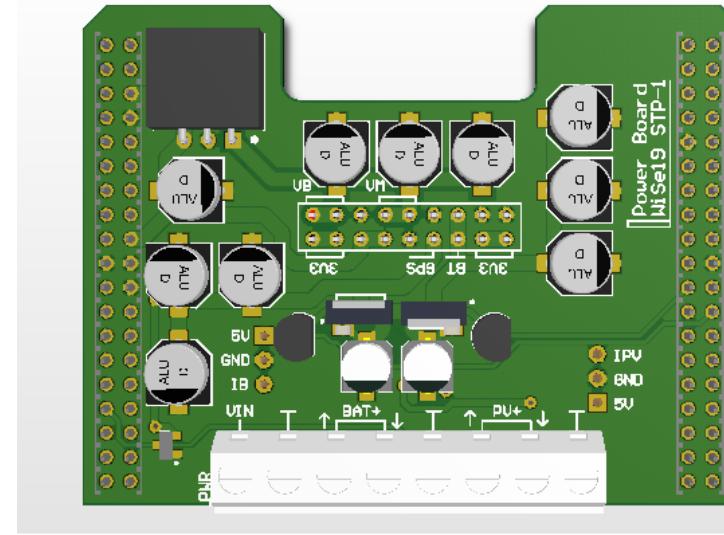


Abbildung 2: Power-Board

Wesentliche Komponenten des Power-Boards sind die Erzeugung der 3V3 und 5V Spannungen. Die 5 V werden von einem Mornsun Festspannungsregler mit einer V_IN Range von 6,5 bis 30 V. Die 3V3 werden direkt vom Mikrokontroller, der mit 5 V versorgt wird, erzeugt. Über zwei Spannungsteiler wird die Batterie- bzw. Solarpanelspannung gemessen (s. Abb. 3):

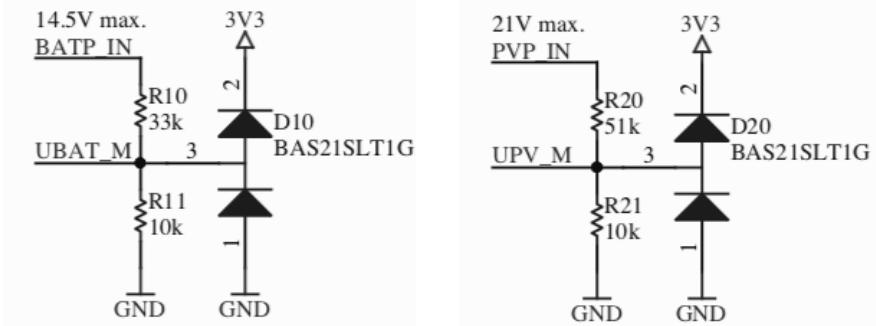


Abbildung 3: Spannungsteiler

Im Zuge der Überlegungen bezüglich möglicher Energiesparmaßnahmen wurden sowohl das Bluetooth- als auch das GPS-Modul als groÙe Verbraucher ermittelt. Da beide Module auch nicht dauerhaft benötigt werden – das GPS-Modul nur alle 15 Minuten zur

2 Spannungsversorgung

Neuausrichtung des Panels und das Bluetooth-Modul nur nach Bedarf – ist es sinnvoll, die Betriebsspannungen beider Module schaltbar zu machen. Eine Möglichkeit dafür ist die Verwendung eines p-Kanal-Mosfets, der von einem n-Kanal-Mosfet getrieben wird. Diese Schaltung wird in Abbildung 4 dargestellt.

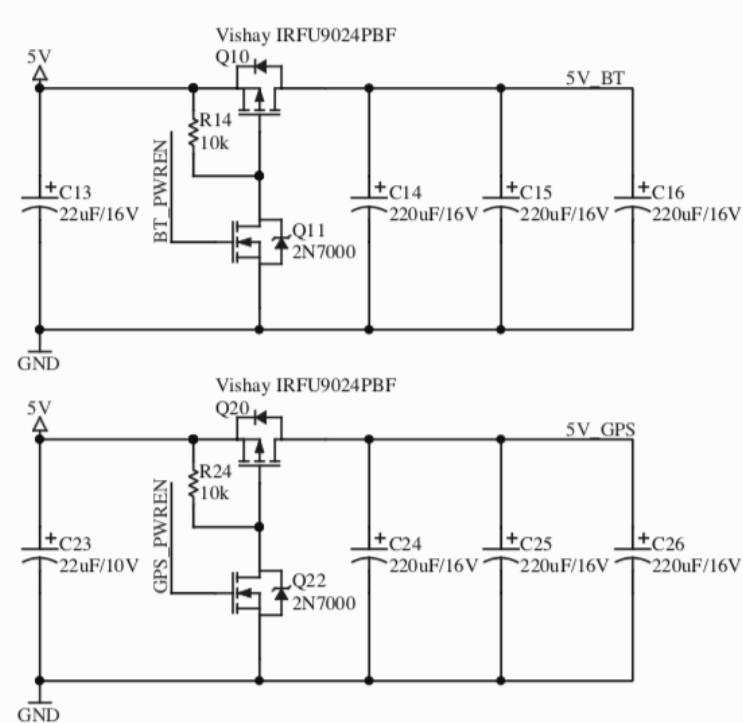


Abbildung 4: Spannungsabschaltung 5VBT und 5VGPS

Die Messung des Ladezustands wird über einen einfachen Spannungsteiler realisiert, der so dimensioniert ist, dass maximal 3,37 V bei vollgeladenem Akku ausgegeben werden. Die genauen Werte sind dem angehängten Stromlaufplan zu entnehmen. Die Ströme werden mittels zweier ACS712 Stromsensoren, die von 5 V versorgt werden, gemessen und vom Mikrokontroller ausgewertet.

Die Pins des Mikrocontrollers werden über das Powerboard zum im folgenden erläuterten Sensor-Board durchgeschliffen.

2.3 Sensor-Board

In Abbildung 5 wird das finale Layout des Sensor-Boards gezeigt:

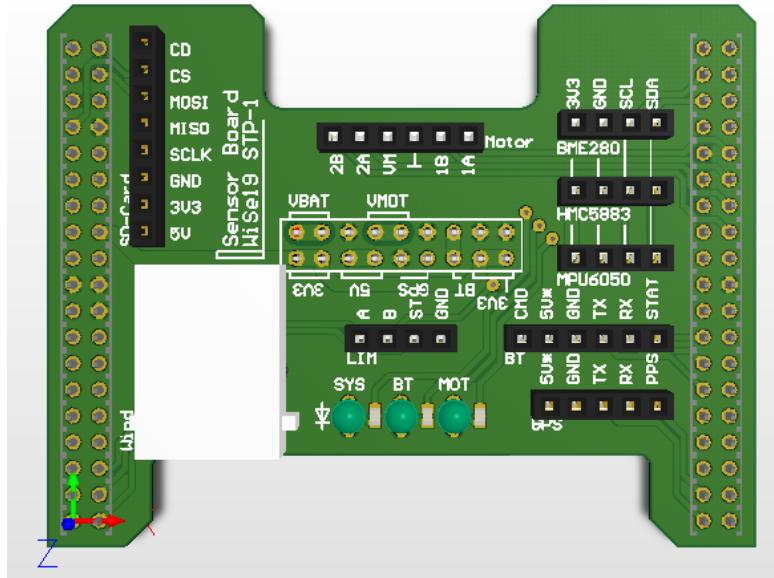


Abbildung 5: Sensor-Board

Das Sensor-Board beschränkt sich im Wesentlichen auf die Verdrahtung von Signal- und Spannungsleitungen der Sensoren mit dem Mikrokontroller und den benötigten Spannungen. Die Sensoren werden dabei über Pinheader mit dem Board konnektiert. Drei Status-LEDs, die im späteren Aufbau im Hauptgehäuse nach außen geführt werden, geben Aufschluss über den Status des Systems, Bluetooth-Verbindung und Motoransteuerung.

3 Sensoren

3.1 Kompassmodul QMC5883L

Als Kompass dient uns der **QMC5883L**. Verwendet wird er, um die Ausrichtung der Wetterstation gen Norden zu messen. Diese Information wird benötigt, um das Solarpanel korrekt zur Sonne auszurichten zu können.

Mit einer Genauigkeit von 1° bis 2° ist er für unsere Zwecke ausreichend genau. Ebenfalls für den Sensor sprechen seine geringe Stromaufnahme von $75 \mu\text{A}$ und die Möglichkeit ihn mittels I²C anzusprechen.

3.1.1 Montage

Das Kompassmodul hat 5 Anschlüsse, von denen wir vier verwenden: V_{CC}, GND, SDA und SCL. Der fünfte Anschluss wird nur bei einer häufigen Datenübertragung benötigt, er sendet die Information, dass Daten zum Abruf bereit stehen. Dargestellt ist die Verschaltung in Abbildung 6.

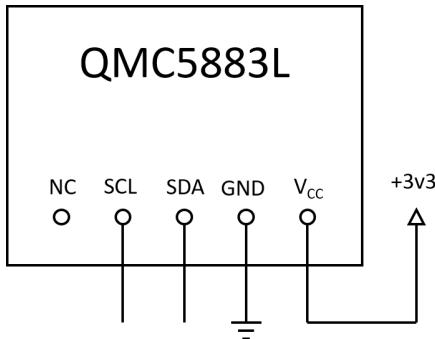


Abbildung 6: Verschaltung des QMC5883L

Da der Sensor das Magnetfeld misst, muss dieser entfernt von der Windfahne und dem Anemometer platziert werden, da beide mit magnetischen Bauteilen arbeiten. Andernfalls könnte es zu Fehlmessungen kommen. Umgesetzt wurde dies so, dass an der einen Stütze des Solarpanels das Kompassmodul und an der anderen die Windfahne, das Anemometer und das gegenüber magnetischen Beeinflussungen relativ unempfindliche GPS-Modul angebracht wurden.

3.2 GPS

Um den Standort des Geräts zu bestimmen und auch um die Berechnung der korrekten Auslenkung des Solarpanels auf Grund des Standorts zu ermöglichen, musste ein GPS-Modul genutzt werden. Wir haben uns für den

3.2.1 Montage

Da das GPS-Modul in Einheit mit einer Antenne funktioniert, musste das Modul entfernt von der Elektronik und damit außerhalb des Gehäuses platziert werden. Dazu bot sich an, das GPS-Modul an der zweiten Stütze des Solarpanels zu platzieren. Das Modul

wurde dafür ebenso in einem Nebengehäuse untergebracht (s. Kap. 8.2). Die Antenne des Moduls wurde dabei mittels Kabelbinder an der Stütze angebracht. Dieses Vorgehen war einer dauerhaften Anbringung, etwa durch Kleber, vorzuziehen, da die Bauteile so leichter ausgetauscht werden können.

3.3 Anemometer und Windfahne

3.3.1 Montage

3.4 Neigungssensor MPU6050

3.4.1 Montage

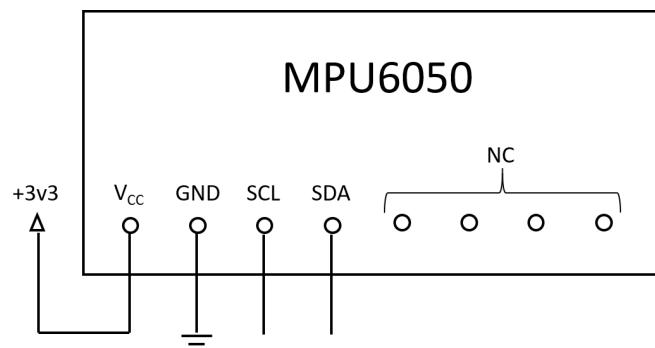


Abbildung 7: Verschaltung des MPU6050

3.5 BME280

3.5.1 Montage

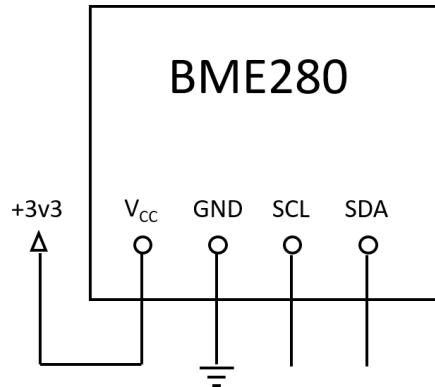


Abbildung 8: Verschaltung des BME280

3.6 Anschlagssensor

3.6.1 Montage

4 Mikrocontroller

Die zentrale Steuereinheit der Wetterstation wird durch einen *STM8L152* Mikrocontroller von ST Microelectronics gebildet. Dieser ist zusammen mit einer 3.3V Spannungsversorgung, einem Quarzoszillator und einem Debug- und Programmieradapter auf einer kommerziell erhältlichen Entwicklungsplatine montiert. Die Verbindung zu den anwendungsspezifischen Modulen der Wetterstation geschieht über zwei zweireihige 38-pin Stiftleisten.

Die Anschlussbelegung der *Nucleo-64* Entwicklungsplatine ist standardisiert [12]; neben dem hier verwendeten *STM8* Mikrocontroller bietet ST Microelectronics Entwicklungsplatten in diesem Formfaktor für eine Reihe von Controllerfamilien an.

4.1 Auswahl des Mikrocontrollersystems

In der Voruntersuchungsphase des Projektes wurden unterschiedliche Mikrocontrollersysteme für den Einsatz in der Wetterstation untersucht. Hierbei konnte auf bereits vorliegende Hardware zugegriffen werden. Folgende Controllerfamilien wurden ausgewertet:

- Atmel, 8-bit AVR: ATmega328P (Arduino Uno)
- Atmel, 8-bit AVR: ATmega2560 (Arduino Mega)
- Atmel, 32-bit ARM Cortex-M3: SAM3X8E (Arduino Due)
- ST Microelectronics, 32-bit ARM Cortex-M4F: STM32F446 (NUCLEO-F446RE)
- ST Microelectronics, 8-bit STM8: STM8L152R8 (NUCLEO-8L152R8)

Für die Auswahl wurde neben den Hardwarespezifikationen auch die Qualität der Software-Werkzeuge und der Entwicklungskomfort in Betracht gezogen. Die Ergebnisse sind nachfolgend in der Entscheidungsmatrix (s. Tab. 1 und 2) aufgezeichnet.

Controller	UART	SPI	I ² C	Analog In	Sleep I _{CC}
ATmega328P [3]	1	1	1	6	4 mA
ATmega2560 [2]	4	1	1	16	7 mA
SAM3X8E [4]	5	4	2	16	8 mA
STM32F446RE [11]	6	4	4	16	10 mA
STM8L152R8 [10]	3	2	1	28	4 mA

Tabelle 1: Entscheidungsmatrix für die Auswahl des Mikrocontrollersystems (Hardwarespezifikationen)

Controller	IDE	Debugger	MCU Library	Periph. Library	Sleep	Aufwand
ATmega328P	Arduino IDE	nein	Arduino	ja	nein	sehr einfach
ATmega328P	Atmel Studio	ja (ext)	ASF	nein	ja	einfach
ATmega2560	Arduino IDE	nein	Arduino	ja	nein	sehr einfach
ATmega2560	Atmel Studio	ja (ext)	ASF	nein	ja	hoch
SAM3X8E	Atmel Studio	ja (ext)	ASF	nein	ja	sehr hoch
STM32F446RE	TrueStudio	ja	CMSIS	nein	ja	sehr hoch
STM8L152R8	STVD	ja	STM8-SPL	nein	ja	einfach

Tabelle 2: Entscheidungsmatrix für die Auswahl des Mikrocontrollersystems (Entwicklungsgrundlage)

Ausgehend hiervon wurde die Entscheidung getroffen, den *STM8L152R8* Mikrocontroller für das Projekt einzusetzen. Der primäre Kandidat, *ATmega328P* konnte aufgrund der zu geringen Anzahl von UART-Schnittstellen nicht eingesetzt werden.

4.2 Pin-Belegung

Für die Planung der Pin-Belegung wurde das Tool *STM8CubeMX* eingesetzt (s. Abb. 9), welches neben der Pin-Zuordnung auch die Konfiguration des Clock Tree dokumentieren kann und ein Analysewerkzeug für die Leistungsaufnahme bereitstellt.

Aus der grafischen Visualisierung des Controller-IC ist zu erkennen, dass die Anzahl der verfügbaren I/Os gerade optimal für diesen Anwendungsfall ist – der Controller ist also weder überdimensioniert, noch mussten Abstriche an den I/Os gemacht werden.

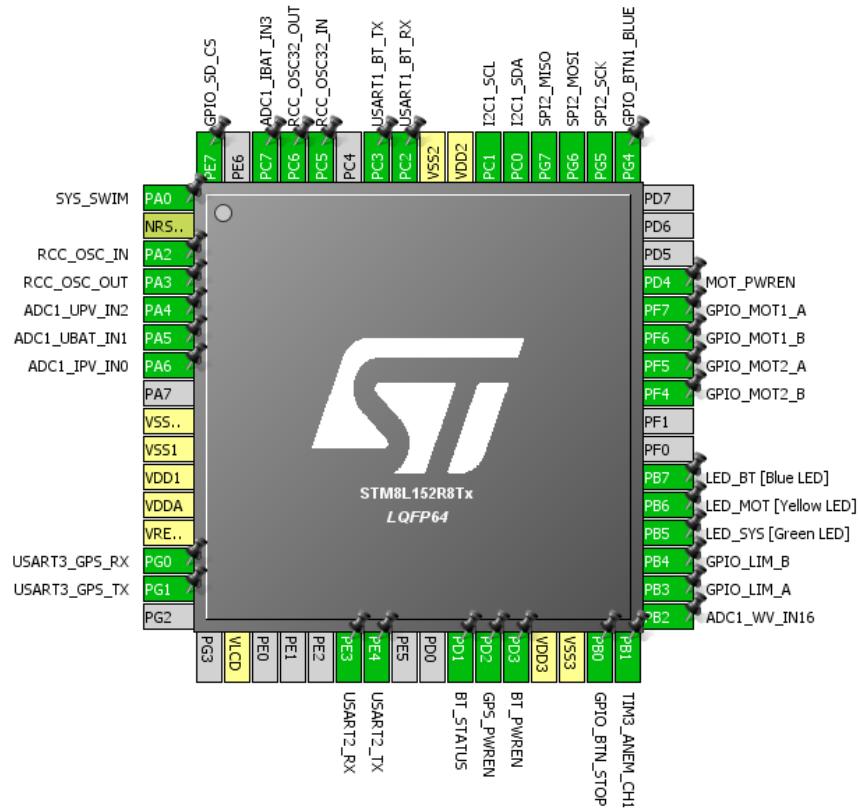


Abbildung 9: Pin-Zuordnungen für den STM8-Mikrocontroller

Peripheral	Verwendung
ADC1	CPU-Temperatursensor, Strom- und Spannungssensoren, Richtungssignal Windfahne
GPIO	Taster, LEDs, Endlagenschalter, Motor- treiber Steuersignale
I2C	Digitale Sensoren
SPI2	SD-Karte
TIM3	Zähler für Anemometer-Geberimpulse
UART1	UART-over-Bluetooth Umsetzer
UART2	printf()-Debugausgaben
UART3	NMEA 0183 Sentences vom GPS-Modul

Tabelle 3: Mikrocontroller-Peripheriemodule und ihr Einsatzzweck für die Wetterstation

5 Firmware

Die entwickelte Mikrocontroller-Firmware definiert den Funktionsumfang der Wetterstation: neben der Auswertung und Aufzeichnung von Messdaten erfolgt die Ausrichtung des Solarpanels softwaregesteuert.

Das Softwareprojekt ist dabei modular aufgebaut – voneinander unabhängige Softwarekomponenten sind in eigene Module aufgeteilt. Funktionen zur Umrechnung der rohen Sensordaten in die gewünschten Darstellungen werden durch entsprechende Unterteilung in verschiedene Headerdateien vor anderen Teilen der Software “versteckt”.

Hieraus wurde eine die folgende Projektstruktur entwickelt:

- `main` – Initialisierung des Systems beim Start und zyklische Datenverarbeitung im Hintergrund-Task
- `commlib/` – Module zur Ansteuerung der seriellen Kommunikationsschnittstellen (UART, SPI, I²C)
- `fslib/` – Implementierung des FAT-Dateisystems für die SD-Karte, basierend auf der Open-Source Bibliothek “FatFS” [5]
- `motorlib/` – Motorsteuerung mit Reaktion auf Betätigung der Endlagenschalter in einer Interruptserviceroutine
- `powerlib/` – C-Präprozessormakros für den Wechsel in den “wait-for-interrupt” Wartezustand
- `sensorlib/` – Module zur Auswertung der digitalen und analogen Sensoren
- `stm8lib/` – STM8 Standard Peripheral Library vom Mikrocontroller-Hersteller ST Microelectronics
- `userlib/` – Module zur Steuerung des Systems auf einer höheren Abstraktionsebene

Die Software wurde dabei für eine interruptgesteuerte Verarbeitung ausgelegt. Der Mikrocontroller wird durch Timer-Interrupts in regelmäßigen Zeitabständen “aufgeweckt”, um Steuerungsaufgaben auszuführen. Nach Abschluss der Ausführung wird er wieder in den Wartezustand “wait-for-interrupt” versetzt, um die Stromaufnahme während der Leerlaufzeit zu verringern.

6 Ausrichtung des Solarpanels

Nachfolgend wird auf einige der oben gelisteten Module näher eingegangen.

5.1 Auswertung der Sensoren (sensorlib**)**

5.1.1 BME280

5.1.2 CPU-Temperatursensor

5.1.3 QMC5883L

5.1.4 MPU6050

5.1.5 Anemometer und Windfahne

5.1.6 Strom- und Spannungsmessung

5.2 GPS: NMEA 0183 Protokollverarbeitung

5.3 Bluetooth: Aktivierung und AT-Befehlssatz

6 Ausrichtung des Solarpanels

Um die Leistungsaufnahme des Solarpanels zu optimieren ist es notwendig dieses direkt auf die Sonne auszurichten und diese Ausrichtung auch in geeigneten Zeitabständen zu korrigieren. Im Vergleich mit einem fest ausgerichteten Solarpanel konnten J. Rizek *et al.* mit einem nachgeführten Solarpanel beispielsweise die Leistungsaufnahme um durchschnittlich 30% erhöhen [7].

Hierfür kommen grundsätzlich verschiedene Methoden in Frage. In diesen Fall soll die Position der Sonne relativ zur Wetterstation auf Grundlage des Längen- und Breitengrades, der Uhrzeit und des Kalendertages berechnet werden. Diese Information werden über das GPS-Modul bereitgestellt. Anschließend wird das Solarpanel mit Hilfe der Motoren, des Kompass-Moduls und des am Panel befestigten Neigungssensors auf die Sonne ausgerichtet.

6.1 Berechnung der Sonnenposition

Da die Formeln zur Berechnung der Sonnenposition in diesem Projekt lediglich benutzt werden, wird an dieser Stelle auf eine Herleitung verzichtet. Die verwendeten Formeln können beispielsweise im *Astronomical Almanac* [1] gefunden werden. *M. L. Roderick* beschreibt die nötigen Berechnungen in seinem Report [9] und liefert zudem compilierbaren C-Code. Dieser wird im folgenden zur Berechnung der Sonnenposition verwendet.

Die Position der Sonne, im Bezug auf einen Beobachter auf der Erde, lässt sich durch die Werte *Zenith* und *Azimut* eindeutig beschreiben. Der Zenith beschreibt den Winkel zwischen einer Linie vom Beobachter zur Sonne und der Vertikalen. Der Azimut den Winkel zwischen der Horizontalen und Norden. Dabei stehen beispielsweise ein Azimut von 90° für Osten, 180° für Süden und 270° für Westen. Eine Veranschaulichung kann in Abbildung 10 gefunden werden.

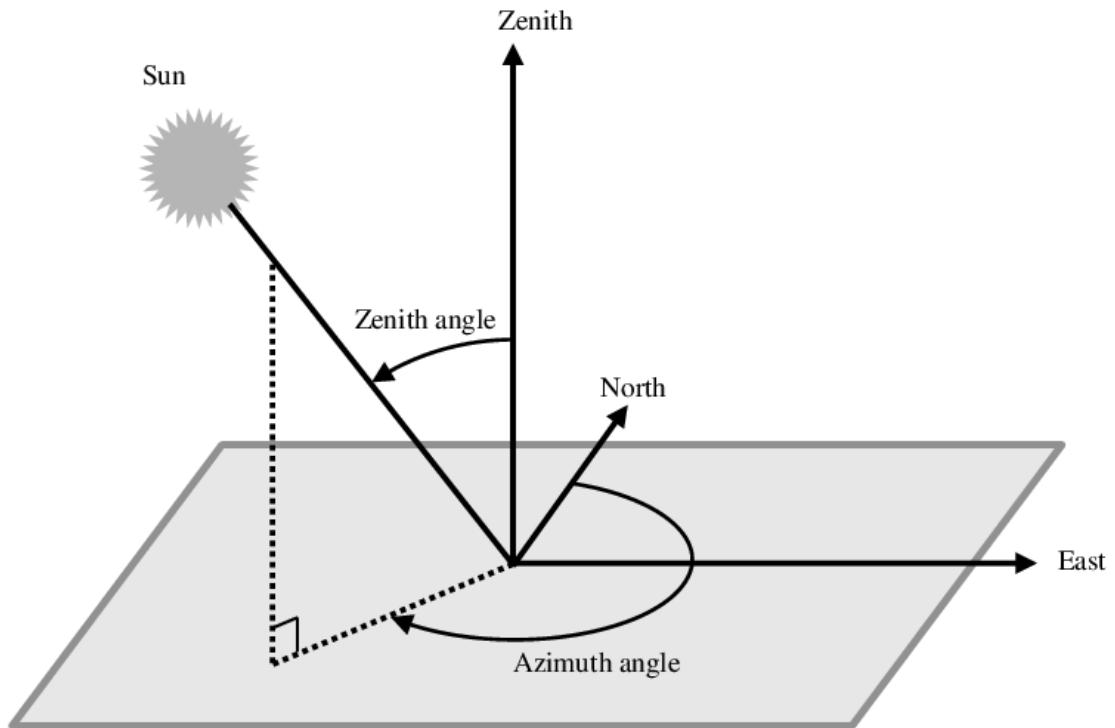


Abbildung 10: Beschreibung der Sonnenposition durch Zenith und Azimut [8]

Die Genauigkeit der verwendeten Formeln beträgt laut dem *Astronomical Almanac* 0.01° bezogen die auf die Position und 0.1 min bezogen auf die Zeit.

7 Benutzeroberfläche

Um eine leichte Handhabung der Wetterstation zu ermöglichen wird eine, auf dem PyQt5 [13] Framework basierende, Benutzeroberfläche verwendet. Für die Kommunikation mit der Wetterstation muss der Computer über eine Bluetooth-Schnittstelle verfügen

In diesem Kapitel werden die zur Verfügung stehenden Funktionen sowie die Entwicklung der Benutzeroberfläche beschrieben.

7.1 Funktionen

Die Benutzeroberfläche (s. Abb. 11) verfügt über zwei Hauptansichten: eine graphische (s. Abb. 12) und eine tabelarische (s. Abb. 13).

Diese unterscheiden sich jeweils nur durch die Anzeigevariante (s. Abb. 11 und 13 (1)).

Über das Konsolenfenster (s. Abb. 12 (2)) können die im Kapitel (HIER KAPITELLINK EINFÜGEN) beschrieben AT-Befehle an die Wetterstation gesendet werden. Sowohl die gesendeten als auch die empfangenen Daten werden im Konsolenfenster angezeigt. Über die Schaltfläche *Clear* lässt sich der Inhalt des Konsolenfensters löschen. Die Größe des Konsolenfensters und des Anzeigefensters kann beliebig verändert werden.

Das Menü (s. Abb. 12 (3)) enthält die zur Bedienung notwendigen Befehle. Es sind nicht alle AT-Befehle implementiert. Die AT-Befehle welche nicht implementiert sind, können manuell über das Konsolenfenster an die Wetterstation gesendet werden. Einige der Menübefehle wurden aufgrund von Zeitmangel nicht implementiert und sind daher ausgegraut. Das Menü enthält die Folgenden Befehle:

- File
 - Save (nicht implementiert): Speichert die empfangenen Daten in einer CSV-Datei ab.
 - Open (nicht implementiert): Lädt die Daten aus einer CSV-Datei in das Programm.
- Control
 - Set Time

- * UTC: Setzt die Uhrzeit und das Datum der Wetterstation auf die Koordinierte Weltzeit. Die Uhrzeit wird der Computeruhr entnommen.
- * Custom (nicht implementiert): Öffnet ein Fenster in welches eine beliebige Uhrzeit und Datum eingegeben werden kann. Diese wird anschließend auf die Wetterstaion gespielt.
- Set Position
 - * Hamburg: Setzt die Postition der Wetterstation auf (53.556354, 10.022650) (HAW Hamburg).
 - * Custom (nicht implementiert): Öffnet ein Fenster in welches beliebige Koordinaten eingegeben werden können. Diese werden anschließend auf die Wetterstation übertragen.
- Adjust Orientation (nicht implementiert): Gibt der Wetterstation den Befehl sich sofort neu auszurichten.
- Set Update-Intervall: Bestimmt das Intervall, in welchem eine Verbindung mit der Wetterstaion aufgebaut wird. Nach dem Aufbau der Verbindung werden die neuen Messdaten angefordert, empfangen und dargestellt. Anschließend wird die Verbindung, um Energie zu sparen, wieder geschlossen.
 - * 5 sec: fünf Sekunden
 - * 1 min: eine Minute
 - * 15 min: fünfzehn Minuten
 - * 1 h: eine Stunde
 - * Manuel: Kein automatisches Anfordern der Messdaten. Zum Anfordern der Messdaten muss die *Update*-Schaltfläche betätigt werden.
- Set Measuring-Intervall: Schickt einen Befehl an die Wetterstation, welcher vorgibt in welchem Intervall Messungen durchzuführen sind.
 - * 5 sec: fünf Sekunden
 - * 15 sec: fünfzehn Sekunden
 - * 1 min: eine Minute

- View: Bestimmt das Zeitintervall, welchem in der graphischen Anzeige dargestellt wird.
 - * Last Minute: letzte Minute
 - * Last 15 Minutes: letzten fünfzehn Minuten
 - * Last Hour: letzte Stunde
 - * Last Day: letzter Tag
 - * Last Week: letzte Woche
 - * Last Month: letzter Monat
 - * Custom (nicht implementiert): Der dargestellte Bereich wird über die Elemente zur Darstellung der Start- und Stopzeit (s. Abb. 12 (4)) eingestellt. Der Bereich wird im Gegensatz zu den anderen Optionen nicht automatisch mit voranschreiten der Zeit geupdatet.
- Mode:
 - Enable Debug: Aktiviert den Debug-Modus. Debugnachrichten werden angezeigt.
 - Disable Debug: Deaktiviert den Debug-Modus. Debugnachrichten werden nicht angezeigt.
 - Enable Com: Baut manuell eine Bluetooth-Verbindung zur Wetterstation auf. Dies ist notwendig, wenn Befehle zur Wetterstation gesendet werden sollen. Für das automatische, periodische Einlesen der Messdaten ist dies nicht notwendig. Es gibt momentan keine Möglichkeit den für die Verbindung verwendeten Com-Port über die Benutzeroberfläche anzupassen. Stattdessen muss dieser manuell in der Datei *main_ctrl.py* geändert werden.
 - Disbale Com: Schließt die Bluetooth-Verbindung zur Wetterstation.

Die Anzeigen (4) (s. Abb. 12) zeigen, wie bereits erwähnt, das Intervall an welches dargestellt wird. Ist dieses Intervall nicht auf *Custom* gestellt, so ist die Endzeit immer die aktuelle Zeit. Die Startzeit ergibt entsprechend des eingestellten Zeitintervalls.

Über die Schaltflächen (5) (s. Abb. 12) können zwei Messkurven ausgewählt werden, welche dargestellt werden sollen. Die Schaltfläche *Update* ist für das manuelle Anfordern

7 Benutzeroberfläche

der Messdaten notwendig. Unter den Schaltflächen werden die, über das Fadenkreuz (6) (s. Abb. 12) ausgewählten, Messpunkte schriftlich dargestellt.

Im aktuellen Softwarestand wird die Kommunikation mit der Wetterstation nicht in einem separaten Task ausgeführt. Dies führt zu Problemen in der Bedienbarkeit der Anwendung. Während das Programm Daten mit der Wetterstation austauscht kann nicht, durch den Nutzer ausgelöste, Ereignisse reagiert werden. Um den Bedienkomfort zu erhöhen sollte die Kommunikation in einen eigenen Task ausgelagert werden. Dies konnte aufgrund der zeitlichen Beschränkung des Projekts nicht realisiert werden.

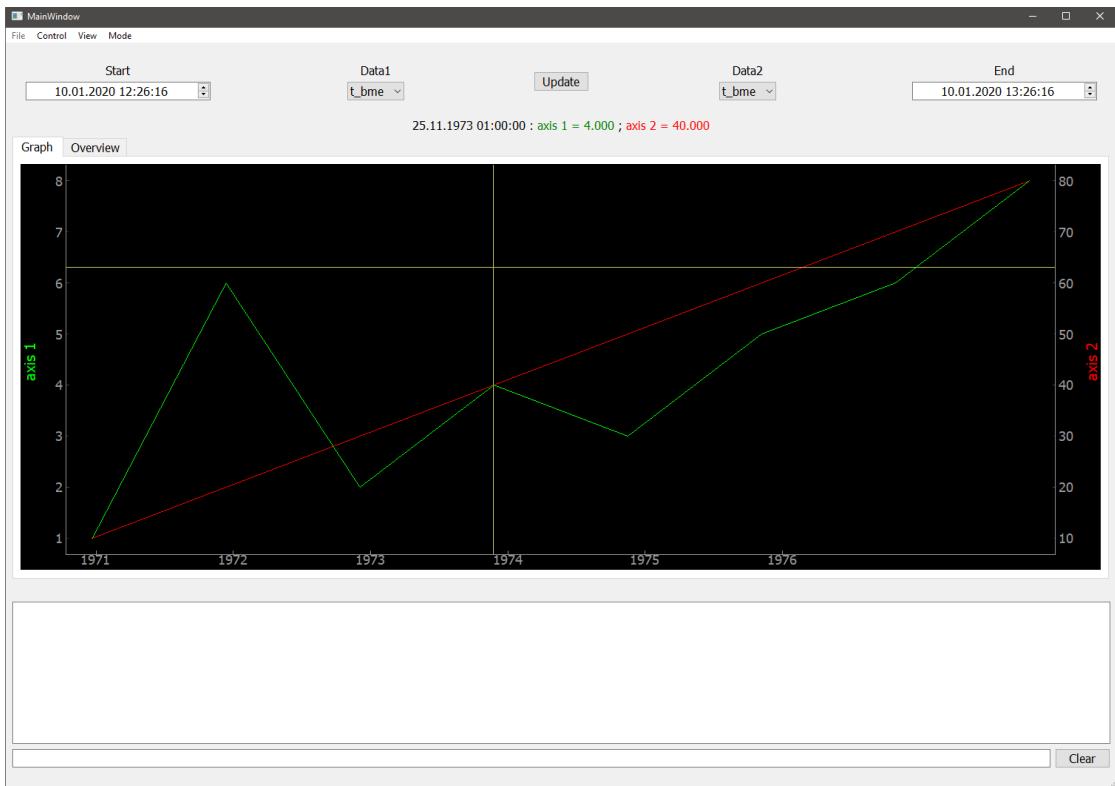


Abbildung 11: Benutzeroberfläche nach dem Öffnen. Dargestellt sind zwei vordefinierte Testkurven, welche nach Empfang des ersten Messwertes gelöscht werden.

7 Benutzeroberfläche

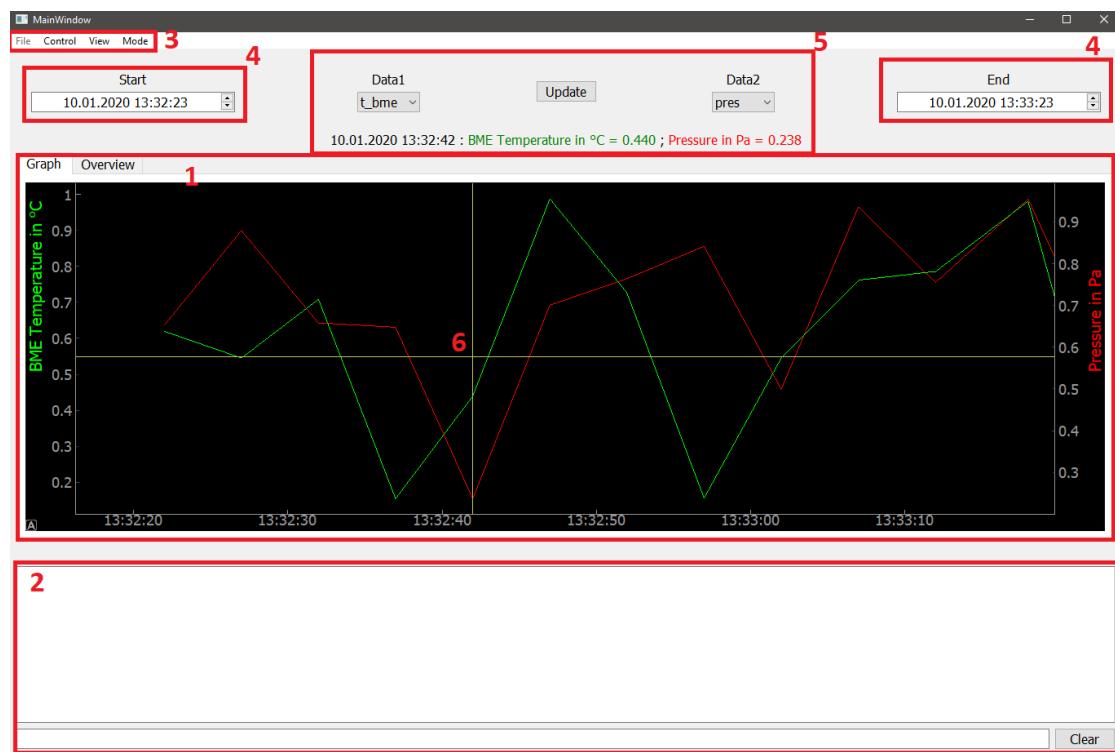


Abbildung 12: Benutzeroberfläche: (1) Graphische Darstellung der Messwerte. (2) Konsolefenster. (3) Kontrollmenü mit den wichtigsten Befehlen. (4) Start- und Endzeit der graphischen Darstellung. (5) Auswahl der dargestellten Messwerte, Schaltfläche zum manuellen Update der Messwerte und Anzeige der Messwerte an der aktuellen Fadenkreuzposition. (6) Fadenkreuz zum Anzeigen spezifischer Messwerte.

7 Benutzeroberfläche

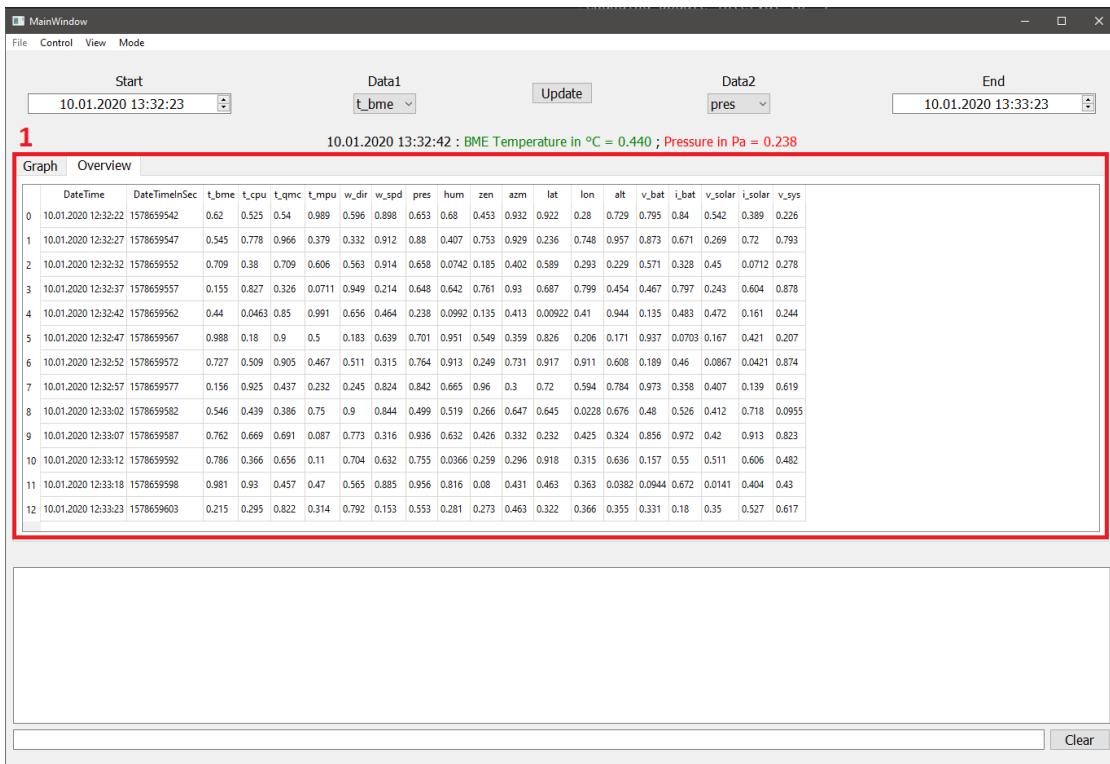


Abbildung 13: Benutzeroberfläche: (1) Tabellarische Darstellung von simulierten Werten.

7.2 Entwicklung

In diesem Kapitel werden die, für die Benutzeroberfläche implementierten Funktionen, *nicht* im Detail beschrieben. Es soll lediglich ein Überblick über die verwendeten Technologien und den Aufbau der Projekts gegeben werden. Bei der Entwicklung wurde versucht den Code in einer Art und Weise zu schreiben, welcher es anderen Personen ermöglicht, ohne eine ausführliche Beschreibung des gesamten Codes, an diesem weiterzuarbeiten.

Als Programmiersprache wurde Python 3 verwendet. Der verwendete Quellcode befindet sich im Ordner UI. Die Oberfläche wurde nach dem Model-View-Controller Entwurfsmuster (s. z.B. [6], s. Abb. 14) entwickelt. Für einige der Funktionalitäten wurde Code aus den, im PyQt5-Package enthaltenen, Beispielen als Vorlage verwendet.

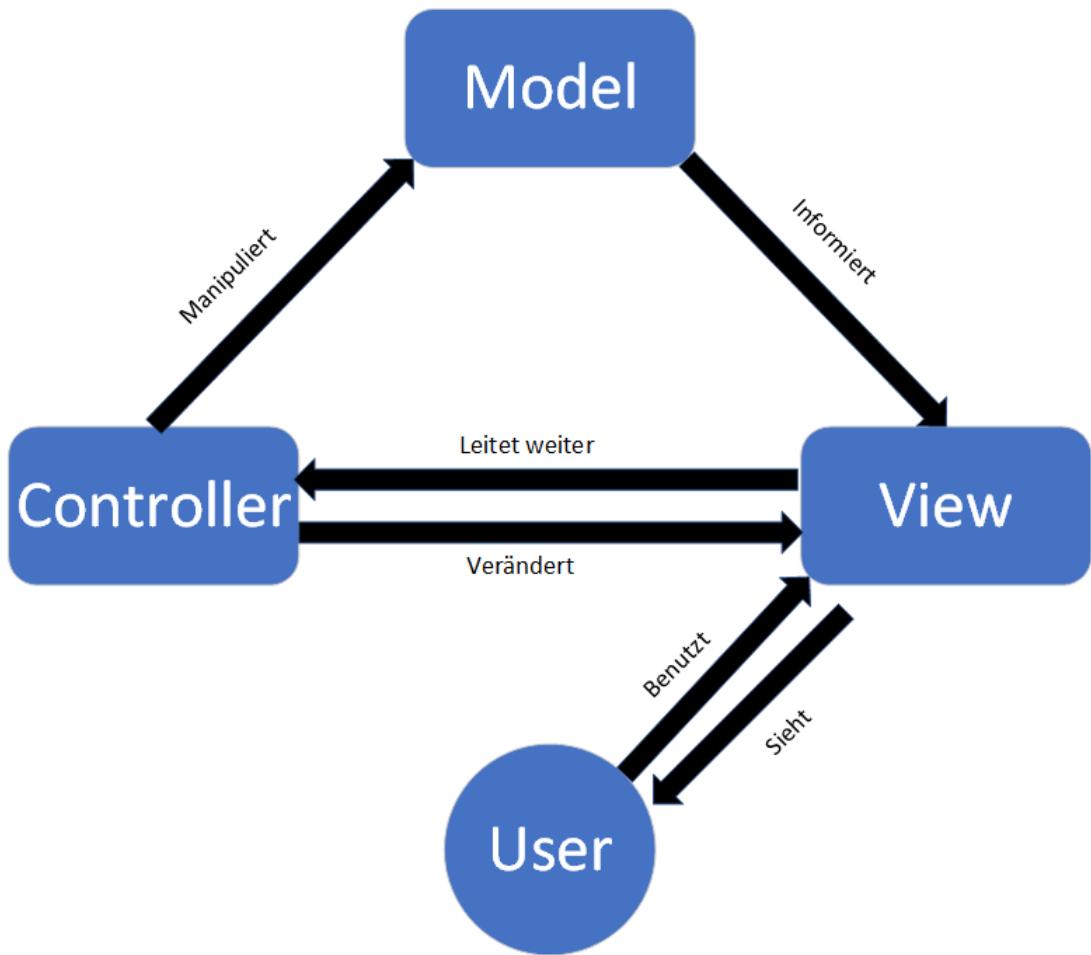


Abbildung 14: Model-View-Controller Entwurfsmuster, symbolische Darstellung

Projektstruktur

Das Projekt ist wie folgt strukturiert:

```
UI
├── .git
└── controllers
    └── main_ctrl.py
├── model
    └── model.py
└── mywidgets
    └── dateaxis.py
```

```
    └── dateaxisitem.py
    └── mplwidget.py
    └── mygraphicswidget.py
    ├── resources
    │   └── main_view.ui
    └── mvc_app.qrc
    ├── views
    │   └── main_view.py
    └── main_view_ui.py
    └── .gitignore.txt
    └── build.bat
    └── mvc_app.py
    └── requirements.txt
```

Die Hauptdatei des Projekts ist *mvc_app.py*. In ihr werden das Model, der Controller und das View initialisiert und verbunden. Das Model (*model.py*) beinhaltet alle Datenstrukturen, also sowohl die Messwerte als auch Variablen, welche den Programmablauf steuern. Das View (*main_view.py* und *main_view_ui.py*) beinhaltet die Benutzeroberfläche und bestimmt welche Funktionen beim Aktivieren welcher Schaltflächen aufgerufen werden. Diese Funktionen befinden sich im Controller (*main_ctrl.py*). Im Ordner *mywidgets* befinden sich die im View implementierten, benutzerdefinierten graphischen Anzeigen.

Die Verbindung zwischen Model, View und Controller ist über *Signale* und *Slots* realisiert.

Die Datei *requirements.txt* beinhaltet eine Liste aller benötigten Python-Packages. Mit dem Befehl „`pip install -r requirements.txt`“ können diese automatisch installiert werden.

Im Ordner *resources* befinden die mit *PyQt5-Designer* erstellten UI-Dateien.

Erstellung der Oberfläche

Zum Erstellen der Oberfläche wurde das graphische Tool *PyQt5-Designer* (s. Abb. 15) verwendet. Das Übersetzen der mit diesem Tool erstellten *.ui*- und *.qrc*-Dateien in ausführbaren Python-Code erfolgt über das Tool *pyuic5*. Zum Automatisieren dieses Pro-

8 3D gedruckte Komponenten

zesses wurde das Skript *build.bat* angelegt. Nach Bearbeitung der Oberfläche im *PyQt5-Designer* sollte dieses ausgeführt werden um die Änderungen zu übernehmen.

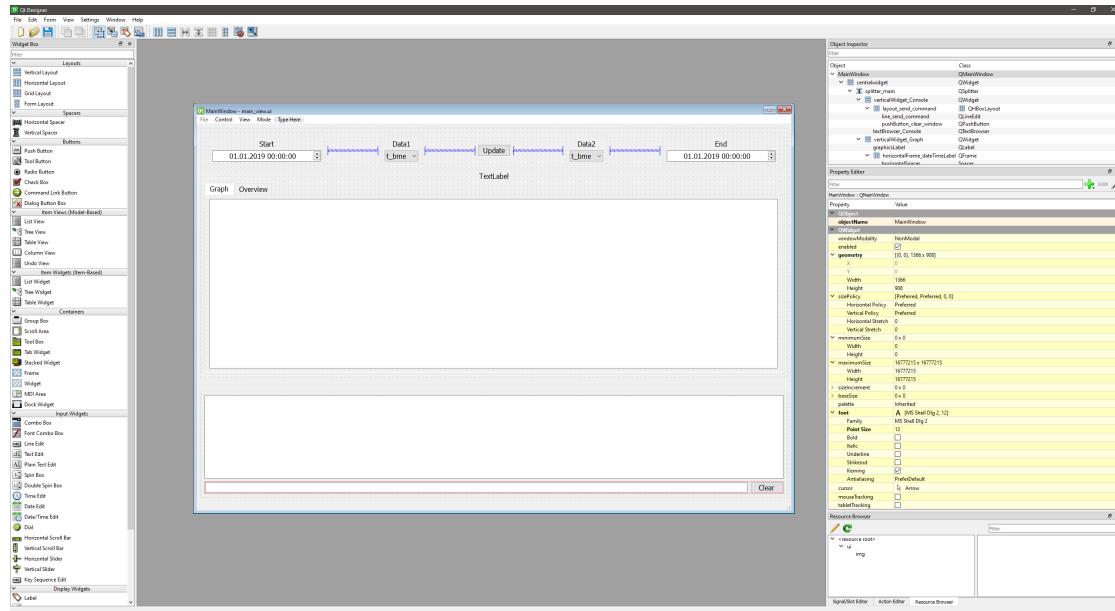


Abbildung 15: Erstellen der Oberfläche mit PyQt5-Designer.

Verwendete Python-Packages

Es werden die folgenden Python-Packages Programm verwendet:

- PyQt5: Als Framework für die Oberfläche.
- pyqtgraph: Für die graphische Darstellung der Messdaten.
- serial: Für die serielle Kommunikation, über Bluetooth, mit der Wetterstation.
- pandas: Für die Strukturierung der Messdaten.
- numpy: Für das Erstellen von Testdaten.

8 3D gedruckte Komponenten

Für das Unterbringen des Mikroprozessors und der Sensoren werden zwei verschiedene Gehäusevarianten verwendet. Dieses Kapitel beschäftigt sich mit deren Entwurf, sowie

dem Entwurf des Adaptors, mit welchem die Windfahne und das Anemometer an der Wetterstation befestigt sind.

Für den Entwurf der Komponenten wurde die Studentenversion von Autocad Fusion 360 verwendet.

8.1 Hauptgehäuse

Das Hauptgehäuse beinhaltet das Mikroprozessor-Board mit den zwei aufsteckbaren Platinen sowie den Sensoren. Drei der Sensoren befinden sich nicht im Hauptgehäuse (s. Kap. 8.2).

Das Gehäuse besteht aus zwei Teilen (s. Abb. 16), dem Hauptteil (links) und dem Deckel (rechts). Das STM8 Nucleo Board wird von drei Pins in Position gehalten (s. Abb. 16, 17). In der Vorderseite befinden sich zwei Aussparungen (s. Abb. 18). Über die Rechte ist der USB-Port des Mikroprozessor-Boards erreichbar. Hinter der Linken befindet sich der Motortreiber, welcher auf einer leicht erhöhten Plattform plaziert wird (s. Abb. 17). Über die Aussparung an der rechten Seite kann die SD-Karte erreicht werden. Die Aussparung an der hinteren Seite wird verwendet, um alle externen Komponenten mit dem Mikroprozessor-Board zu verbinden.

Im Deckel befinden sich drei Löcher, in welche Status-LEDs untergebracht werden.

Das Hauptteil und der Deckel werden lediglich aufeinander gesteckt. Bedingt durch die Geometrie entsteht ein relativ fester Halt. Optional können die Komponenten über die vier Aussparungen an den Seiten, mit beispielsweise Kabelbindern, zusätzlich gesichert werden.

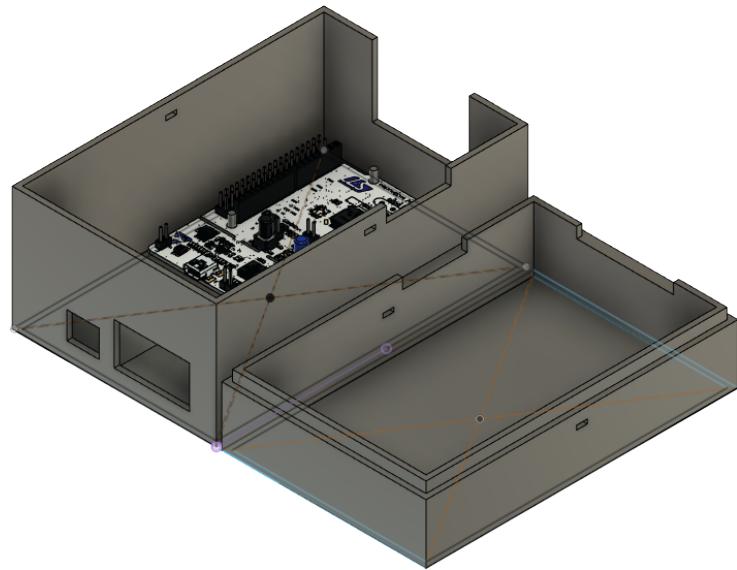


Abbildung 16: Hauptgehäuse

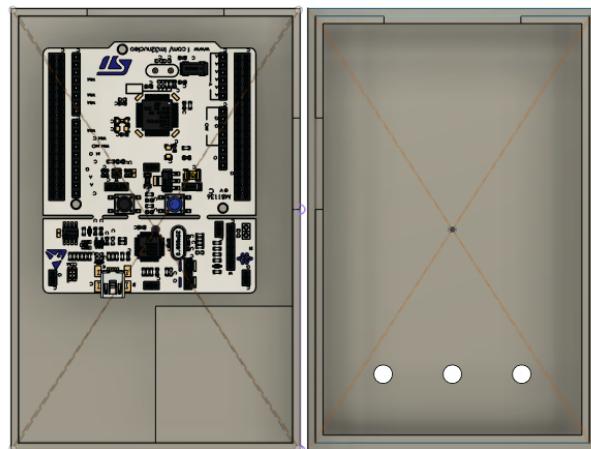


Abbildung 17: Hauptgehäuse, Ansicht von oben

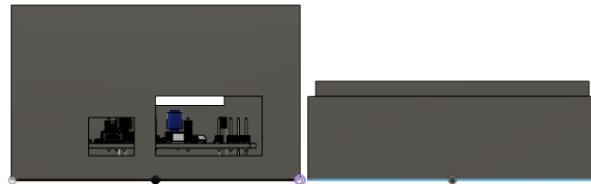


Abbildung 18: Hauptgehäuse, Ansicht von vorne

8.2 Nebengehäuse

Es wird jeweils ein Nebengehäuse für folgende Sensoren verwendet:

- der Neigungssensor, welcher an der Unterseite des Solarpanels befestigt werden muss
- der GPS-Empfänger, welcher für besseren Empfang am oberen Ende der Wetterstation befestigt wird
- das Kompass-Modul, welches empfindlich auf elektro-magnetische Störungen reagiert und aus diesem Grund auch am oberen Ende der Wetterstation untergebracht wird (auf der Seite gegenüber des GPS-Empfängers)

Die Nebengehäuse bestehen aus zwei Teilen, welche aufeinandergesteckt werden und zusätzlich, optional, mittels Kabelbinder gesichert werden (s. Abb. 19, 21).

Die Nebengehäuse werden mit einer Schraube und einer Mutter am Aluminium-Profil der Wetterstation befestigt (s. Abb. 20). Das Nebengehäuse, welches sich auf dem Solarpanel befindet, kann nicht geschraubt werden und wird entsprechend mit Klebstoff befestigt.

Die runden Aussparungen an der linken und rechten Seiten dienen dem Durchführen von benötigten Leitungen.

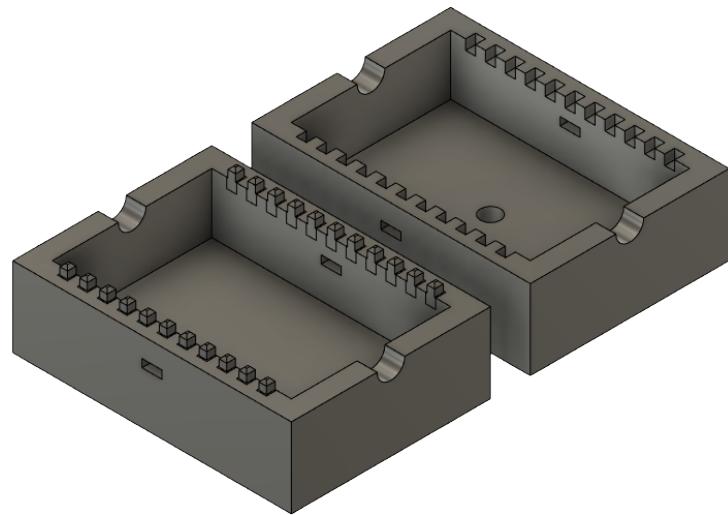


Abbildung 19: Nebengehäuse

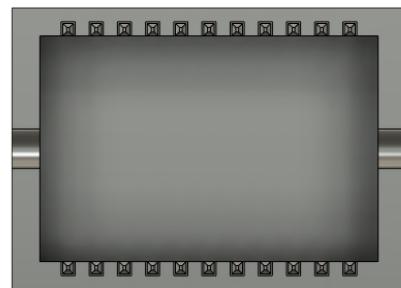
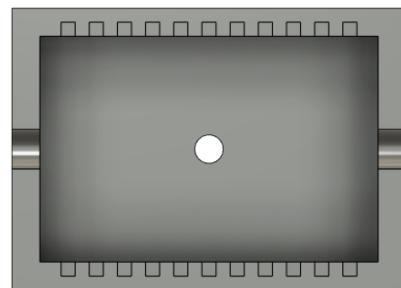


Abbildung 20: Nebengehäuse, Ansicht von oben

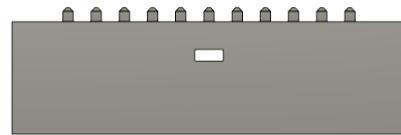


Abbildung 21: Nebengehäuse, Ansicht von vorne

8.3 Adaptor

Der Adaptor wird verwendet, um eine feste Verbindung zwischen dem Aluminium-Profil der Wetterstation und dem Mast mit Windfahne und Anemometer herzustellen.

Die Verbindung zur Wetterstation erfolgt durch Stecken der vier trapez-förmigen Steckkontakte am unteren Ende des Adaptors (s. Abb. 22, 24).

Der Mast wird von oben auf die angepasste Aussparung gesteckt. Optional kann dieser mittels Schraubverbindung fixiert werden (s. Abb. 22, 23).

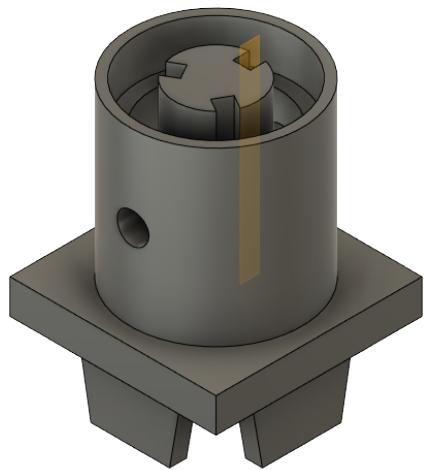


Abbildung 22: Adapter

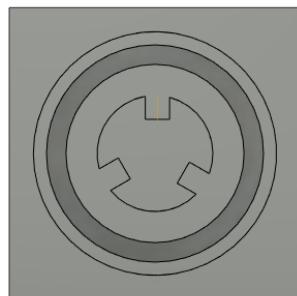


Abbildung 23: Adapter, Ansicht von oben

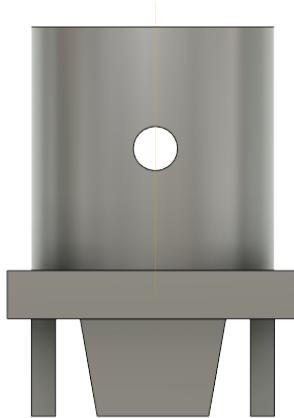


Abbildung 24: Adapter, Ansicht von vorne

8.4 Herstellung

Alle Komponenten wurden in PLA mit einer Schichtdicke von 0.2 mm gedruckt. Es wurden zwei bis drei Top-, Bottom- und Wandschichten gedruckt. Da die Komponenten keine großen Kräfte aushalten müssen wurden sie lediglich mit einem Infill von 10 % bis 20 % gedruckt.

9 Fazit



Abbildung 25: Fertiggestellte Wetterstation

In Abbildung 25 wird der fertige Aufbau der Wetterstation dargestellt. Die Anforderung der Erfassung von Temperatur, Luftdruck und -feuchte wird mit dem BME280, der im Hauptgehäuse platziert ist, umgesetzt. Für die Erfassung von Windrichtung und -geschwindigkeit konnte ein neues Anemometer gefunden und implementiert werden.

Im Hauptgehäuse ließen sich Mikrocontroller, Platinen und Sensoren sinnvoll unterbringen (Abbildung 26). Die Wasserfestigkeit des Aufbaus konnte leider, mit den verfügbaren Mitteln, in der vorgegebenen Zeit, nicht erreicht werden. Das verwendete 3D-Druck-Material ist nicht für einen abgedichteten Aufbau geeignet.

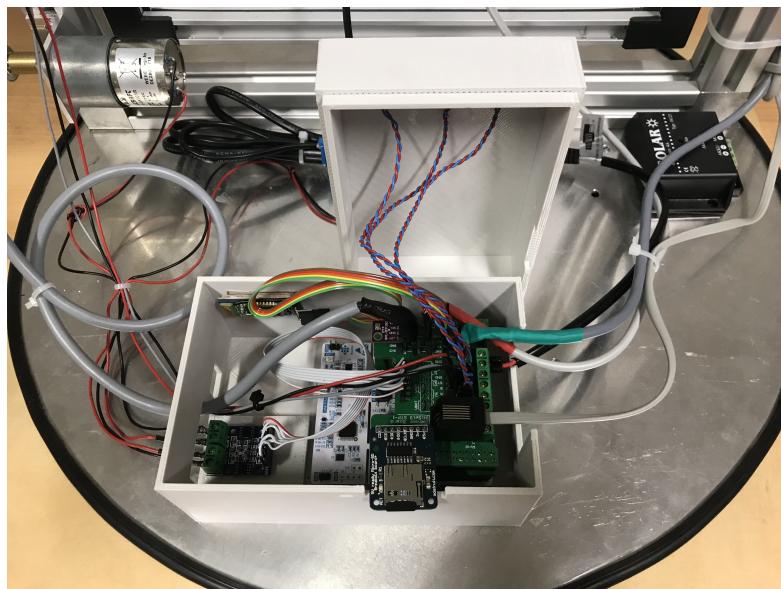


Abbildung 26: Blick in das Hauptgehäuse

Sowohl hardware- als auch softwareseitig wurden verschiedene Energiesparmaßnahmen implementiert. Die letztendliche Laufzeit müsste noch in einem Langzeittest ermittelt werden. Der verbaute Neigungssensor wird letztendlich nicht zur Kompensation bei der Ausrichtung des Panels benutzt, ist aber in diesem Fall auch nicht nötig, da die Funktionalität trotzdem gegeben ist. Die Aufhängung des Panels ließe sich in ihrer Mechanik noch verbessern, da sie in ihrer aktuellen Umsetzung relativ instabil ist, sodass das Panel leicht schief hängt. Diese Verbesserung war jedoch im vorgegebenen Zeitrahmen nicht möglich.

Grundlegend lässt sich sagen, dass alle gegebenen Anforderungen an das Projekt erfüllt wurden. Zusätzlich wurde eine Benutzeroberfläche zum Bedienen der Wetterstation und Anzeigen der Messwerte entwickelt.

Literatur

- [1] ANON.: *The Astronomical Almanac*
- [2] ATMEL CORPORATION: *ATmega2560 8-bit Atmel Microcontroller with 64KB In-System Programmable Flash.* : , 2014. – URL <https://ww1.microchip.com/downloads/en/devicedoc/>

- atmel-2549-8-bit-avr-microcontroller-atmega640-1280-1281-2560-2561_datasheet.pdf. – Zugriff: 11.01.2020
- [3] ATMEL CORPORATION: *Atmega328P 8-bit AVR Microcontroller with 32K Bytes In-System Programmable Flash.* : , 2015. – URL http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf. – Zugriff: 11.01.2020
- [4] ATMEL CORPORATION: *SAM3X/SAM3A Series SMART ARM based MCU.* : , 2015. – URL https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-11057-32-bit-Cortex-M3-Microcontroller-SAM3X-SAM3A_Datasheet.pdf. – Zugriff: 11.01.2020
- [5] CHAN: *FatFs - Generic FAT Filesystem Module.* – URL http://elm-chan.org/fsw/ff/00index_e.html. – Zugriff: 11.01.2020
- [6] DEACON, John: *Model-View-Controller (MVC) Architecture.* 2009. – URL <https://www.rareparts.com/pdf/MVC.pdf>. – Zugriff: 10.01.2020
- [7] J., Rizek ; Y., Chaiko: Solar Tracking System: More Efficient Use of Solar Panels. In: *World Academy of Science, Engineering and Technology* 41 (2008). – URL <http://www.mugla.edu.tr/data/03060010/belgeler/solar%20tracking.pdf>. – Zugriff: 17.10.2019
- [8] NOU, Julien ; CHAUVIN, Rémi ; THIL, Stéphane ; GRIEU, Stéphane: A new approach to the real-time assessment of the clear-sky direct normal irradiance. In: *Applied Mathematical Modelling* 40 (2016), 03
- [9] RODERICK, M.L.: Methods for calculating solar position and day length including computer programs and subroutines. In: *Resource Management Technical Reports* (1992). – URL <https://researchlibrary.agric.wa.gov.au/cgi/viewcontent.cgi?article=1122&context=rmtr>. – Zugriff: 17.10.2019
- [10] ST MICROELECTRONICS: *STM8L152x6/8 8-bit ultra-low-power MCU.* : , 2018. – URL <https://www.st.com/resource/en/datasheet/stm8l152r8.pdf>. – Zugriff: 11.01.2020
- [11] ST MICROELECTRONICS: *STM32F446xC/E Arm® Cortex®-M4 32-bit MCU+FPU.* : , 2019. – URL <https://www.st.com/resource/en/datasheet/stm32f446re.pdf>. – Zugriff: 11.01.2020

Literatur

- [12] ST MICROELECTRONICS: *UM1724, STM32 Nucleo-64 boards.* : , 2019.
 - URL https://www.st.com/resource/en/user_manual/dm00105823.pdf. – Zugriff: 11.01.2020
- [13] TOMPSON, Phil: *Python bindings for the Qt cross platform application toolkit.* 2019.
 - URL <https://pypi.org/project/PyQt5/>. – Zugriff: 28.12.2019

A Anhang