

**HTBLuVA Salzburg**

**Höhere Lehranstalt für**

**Elektronik und Technische Informatik**

HTL_Logo

**DIPLOMARBEIT**

Gesamtprojekt

**C-Track 2.0**

Entwurf einer Lokalisierungseinheit für Nutztiere

Samuel Putz 5BHEL Betreuer: Prof. Dipl.-Ing. Siegbert Schrempf

Simon König 5BHEL

Kooperationspartner: Z\_GIS - Fachbereich Geoinformatik

ausgeführt im Schuljahr 2023/24

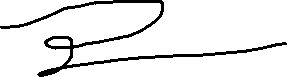
Abgabevermerk:

Datum: 02.04.2024 übernommen von: Siegbert Schrempf

**Eidesstattliche Erklärung**

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbständig und ohne fremde Hilfe verfasst, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Salzburg, am 02.04.2024 Verfasserinnen / Verfasser:



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Samuel Putz

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Simon König

**DIPLOMARBEIT**

**DOKUMENTATION**

|  |  |
| --- | --- |
| Namen der  Verfasserinnen / Verfasser | Samuel Putz  Simon König |
| Jahrgang  Schuljahr | 5BHEL  2023/24 |
| Thema der Diplomarbeit | C-Track 2.0 - Entwurf einer Lokalisierungseinheit für Nutztiere |
| Kooperationspartner | Z\_GIS - Fachbereich Geoinformatik |

|  |  |
| --- | --- |
| Aufgabenstellung | Die genaue Lokalisierung von Nutztieren in alpinen Lagen ist von großem praktischen als auch von wissenschaftlichem Interesse. Derzeit am Markt befindliche Lösungen sind technisch unausgereift oder dementsprechend teuer. Deshalb sollte eine einfache und robuste Lokalisierungseinheit für Herdetiere entwickelt werden. |

|  |  |
| --- | --- |
| Realisierung | Mithilfe des LoRa Netzwerks werden die GPS-Daten von der Platine an der Kuh an einen Empfänger auf der Alm übermittelt und zusätzlich in einem kürzeren Zeitintervall auf einer SD-Karte gespeichert. Von dort aus werden die Daten an eine REST-API gesendet und in eine SQL-Datenbank gespeichert. Von dieser werden die Daten an den Nutzer übermittelt. |

|  |  |
| --- | --- |
| Ergebnisse | Das Endergebnis ist eine ausfallsichere Lokalisierungseinheit für Nutztiere, welche die gesammelten Daten für die Analyse des sozialen Herdenverhaltens auf einer SD-Karte speichert und dem Landwirt bei der Suche der Tiere hilft.. |

|  |  |
| --- | --- |
| Typische Grafik, Foto etc.  (mit Erläuterung) | Abbildung 1: Projekt Logo |

|  |  |
| --- | --- |
| Teilnahme an Wettbewerben,  Auszeichnungen | Keine Teilnahmen |

|  |  |
| --- | --- |
| Möglichkeiten der Einsichtnahme in die Arbeit | Schulbibliothek der HTBLuVA Salzburg |

|  |  |  |
| --- | --- | --- |
| Approbation  (Datum / Unterschrift) | Prüferin / Prüfer | Direktorin / Direktor  Abteilungsvorständin / Abteilungsvorstand |

**DIPLOMA THESIS**

**Documentation**

|  |  |
| --- | --- |
| Author(s) | Samuel Putz  Simon König |
| Form  Academic year | 5BHEL  2023/24 |
| Topic | C-Track 2.0 - Design of a localization unit for livestock |
| Co-operation Partners | Z\_GIS – department of geoinformatics |

|  |  |
| --- | --- |
| Assignment of Tasks | The precise location of farm animals in alpine locations is of great practical and scientific interest. Solutions currently on the market are technically immature or correspondingly expensive. Therefore, a simple and robust localization unit for herd animals should be developed. |

|  |  |
| --- | --- |
| Realisation | Using the LoRa network, the GPS data is transmitted from the circuit board on the cow to a receiver on the mountain pasture and is also stored on an SD card at a shorter time interval. From there, the data is sent to a REST API and stored in a SQL database. From here the data is transmitted to the user. |

|  |  |
| --- | --- |
| Results | The result brings a fail-safe farm animal locator that stores the collected data in an SD card for both social herd behaviour analysis and can help the farmer locate and check the animals' well-being. |

|  |  |
| --- | --- |
| Illustrative Graph, Photo  (incl. explanation) | Abbildung 2: Project Logo |

|  |  |
| --- | --- |
| Participation in Competitions  Awards | No participations |

|  |  |
| --- | --- |
| Accessibility of  Diploma Thesis | Library of the Higher Technical College in Salzburg |

|  |  |  |
| --- | --- | --- |
| Approval  (Date / Sign) | Examiner | Head of College  Head of Department |

**Vorwort**

C-Track wurde im vergangenen Jahr initiiert, jedoch blieb es unvollendet. Die Bedeutung dieses Projekts für die Forschung am Zentrum für Geoinformatik (ZGIS), sowie für Bauern in alpinen Regionen ist von großer Tragweite. Aus diesem Grund haben wir es uns zur Aufgabe gemacht, dieses Projekt unter dem Namen C-Track 2.0 fortzusetzen, zu restrukturieren und zu optimieren. Die Idee hinter C-Track erwies sich als äußerst vielversprechend, jedoch stellten sich zahlreiche Herausforderungen auf unserem Weg. Um die Anforderungen der Kooperationspartner und unsere eigenen Ideen umzusetzen, wurden viele Hürden überwunden und neues Wissen angeeignet. Jedes Teammitglied arbeitete dabei in dem Fachgebiet, das er am besten beherrschte.

**Danksagung**

Wir möchten uns sehr herzlich bei allen Personen, die uns bei der Realisierung unserer Diplomarbeit unterstützt haben, bedanken.

Besonderen Dank möchten wir an folgende Personen aussprechen:

* Prof. Dipl.-Ing. Siegbert Schrempf für die Unterstützung als unser Projekt Betreuer
* Der ZGIS, insbesondere Frau Dr. Assoc. Prof. Gudrun Wallentin als Ansprechpartnerin für das Projekt.
* Oliver Gollenz für die Unterstützung beim Platinenentwurf
* Bei unseren Vorgängern Samuel Neureiter und Marvin Klabacher
* Arno Krenslehner für Unterstützung im Bereich der drahtlosen Übertragung

Ohne der Unterstützung dieser Personen wäre das Projekt in diesem Ausmaß nicht möglich gewesen.

Inhalt

[1 Einleitung 11](#_Toc163449179)

[2 Systemspezifikation 12](#_Toc163449180)

[2.1 Zielbestimmungen 12](#_Toc163449181)

[2.1.1 Musskriterien 12](#_Toc163449182)

[2.1.2 Wunschkriterien 12](#_Toc163449183)

[2.1.3 Abgrenzungskriterien 12](#_Toc163449184)

[2.2 Produkteinsatz 13](#_Toc163449185)

[2.2.1 Anwendungsbereiche 13](#_Toc163449186)

[2.2.2 Zielgruppen 13](#_Toc163449187)

[2.2.3 Betriebsbedingungen 13](#_Toc163449188)

[2.3 Entwicklungsumgebung 14](#_Toc163449189)

[2.3.1 Software 14](#_Toc163449190)

[2.3.2 Hardware 14](#_Toc163449191)

[2.3.3 Orgware 14](#_Toc163449192)

[2.4 Produktfunktionen 15](#_Toc163449193)

[*2.5* Produktdaten 16](#_Toc163449194)

[2.6 Produktleistungen 17](#_Toc163449195)

[2.7 Qualitätszielbestimmungen 18](#_Toc163449196)

[2.8 Globale Testszenarien und Testfälle 19](#_Toc163449197)

[3 Organisation - Projektmanagement 20](#_Toc163449198)

[3.1 Projektteam 20](#_Toc163449199)

[3.2 Individuelle Aufgabenstellungen inkl. Arbeits- und Terminplan 21](#_Toc163449200)

[3.2.1 Samuel Putz 21](#_Toc163449201)

[3.2.2 Simon König 22](#_Toc163449202)

[4 Grundlagen und Methoden 23](#_Toc163449203)

[4.1 Platine 24](#_Toc163449204)

[4.1.1 Energie Versorgung 24](#_Toc163449205)

[4.1.2 Spannungsregler 24](#_Toc163449206)

[4.1.3 GPS-Modul 30](#_Toc163449207)

[4.1.4 Mikrocontroller 33](#_Toc163449208)

[4.1.5 SD-Karte 34](#_Toc163449209)

[4.1.6 Gehäuse 38](#_Toc163449210)

[4.1.7 Programmablauf 39](#_Toc163449211)

[4.2 LoRa 41](#_Toc163449212)

[4.2.1 Anforderungen 41](#_Toc163449213)

[4.2.2 Rechtliche Rahmenbedingungen 42](#_Toc163449214)

[4.2.3 Modulation 42](#_Toc163449215)

[4.2.4 Codierung 43](#_Toc163449216)

[4.2.5 Antennen 46](#_Toc163449217)

[4.2.6 SX1276 47](#_Toc163449218)

[4.3 Basisstation 48](#_Toc163449219)

[4.3.1 TTGO T-Beam 48](#_Toc163449220)

[4.3.2 PlatformIO 48](#_Toc163449221)

[4.3.3 Programmablauf 48](#_Toc163449222)

[4.4 Server 51](#_Toc163449223)

[4.4.1 Laravel 51](#_Toc163449224)

[4.5 Smartphone Applikation 55](#_Toc163449225)

[4.5.1 Anforderungen 55](#_Toc163449226)

[4.5.2 Flutter 55](#_Toc163449227)

[4.6 Verbesserungen 62](#_Toc163449228)

[4.6.1 Platine 62](#_Toc163449229)

[4.6.2 Server 62](#_Toc163449230)

[4.6.3 Smartphone Applikation 62](#_Toc163449231)

[5 Ergebnisse – Abnahme 63](#_Toc163449232)

[6 Literaturverzeichnis 65](#_Toc163449233)

[7 Abkürzungen 68](#_Toc163449234)

[8 Abbildungen 69](#_Toc163449235)

[9 Begleitprotokoll gemäß § 9 Abs. 2 PrO 71](#_Toc163449236)

[9.1 Begleitprotokoll Samuel Putz 71](#_Toc163449237)

[9.2 Begleitprotokoll Simon König 72](#_Toc163449238)

[10 Anhang 73](#_Toc163449239)

# Einleitung

Die genaue Lokalisierung von Nutztieren in alpinen Lagen ist von großem praktischen, sowie wissenschaftlichem Interesse. Derzeit am Markt befindliche Lösungen sind technisch unausgereift oder entsprechend teuer. Deshalb sollte eine einfache und robuste Lokalisierungseinheit für Herdentiere entwickelt werden.

Ein Bild, das Clipart, Diagramm enthält.

Automatisch generierte Beschreibung

Abbildung 3: Blockschaltbild

Mithilfe der LoRa Funkübertragung werden die GPS-Daten von der Sendeplatine an der Kuh an einen Empfänger auf der Alm übermittelt und zusätzlich in einem kürzeren Zeitintervall auf einer SD-Karte gespeichert. Von dort aus werden die Daten an eine REST-API gesendet und in eine SQL-Datenbank gespeichert. Von dieser werden die Daten an den Nutzer übermittelt.

# Systemspezifikation

## Zielbestimmungen

### Musskriterien

* Outdoorfähiges Gehäuse
* GPS-Daten müssen auf SD-Karte gespeichert werden
* Standort auf wenige Meter genau
* Hohe Reichweite der Übertragung
* Geringer Leistungsverbrauch
* Benutzerfreundliche Handyapplikation

### Wunschkriterien

* Webseite
* Lange Akkulaufzeit
* Login für User

### Abgrenzungskriterien

* Ausschließlich für Almbetrieb vorgesehen
* Die Applikation ist nur mit Android kompatibel
* Basisstation muss eine stabile Internetverbindung vorhanden sein
* Das Mobiltelefon muss einen Internetzugriff haben

## Produkteinsatz

### Anwendungsbereiche

Das Produkt soll in alpinen und schwer zugänglichen Regionen den Viehbetrieb für Landwirte erleichtern, sowie Informationen über das Herdeverhalten bringen

### Zielgruppen

Dieses Projekt dient als Hilfestellung für Landwirte in alpinen Regionen sowie eine Erleichterung in der Forschung

### Betriebsbedingungen

* Platine:

Die Platine soll bei jeder Wetterbedingung GPS-Daten aufnehmen und an die Basisstation senden. Die Stromversorgung per Batterie soll dabei über einen langen Zeitraum standhalten. Zielumgebung ist eine offene Weidefläche.

* Basisstation:

Die Basisstation soll in einer Almhütte geschützt von Wind und Wetter mit einer stabilen Internetverbindung stationiert werden. Zudem soll ein geeigneter Anbringungsort für die Antenne vorhanden sein.

* Smartphone Applikation:
  1. Die Applikation ist für Android Mobiltelefone ausgelegt.
  2. Es wird eine Internetverbindung benötigt, um die aktuellen Daten abzurufen.

## Entwicklungsumgebung

### Software

* EAGLE: Entwurf der Platine (V: 9.6.2)
* Autodesk Fusion: Entwurf des Gehäuses
* Visual Studio Code: Programmierung von Handy App, Backend und Basisstation (V: 1.87.0)
* Microchip Studio: Programmierung des Mikrocontrollers (V: 7.0.2594)

### Hardware

* ATmega644 PA (Mikrocontroller)
* Elara-1 (GPS-Modul)
* RFM95W (Lora Breakoutmodul)
* TTGO T-Beam 1.1 (Basisstation)
* LPRS-ANT-868-DP-N-F (Antenne Bodenstation)
* ISM 868/915MHz Dipole (Antenne Sender)
* Android Smartphone

### Orgware

* OneDrive: Cloud Speicher (V: 24.025.0204.0003)
* Draw.io: Diagramme (V: 24.1.0)

## Produktfunktionen

**/F0010/ Empfang der Standortdaten**

Die Sendeeinheit soll mittels einem GPS-Moduls den Standort ermitteln können.

**/F0020/ Speichern auf SD-Karte**

Die GPS-Daten sollen in einem 5-Minuten Intervall auf einer SD-Karte gespeichert werden.

**/F0030/ Datenübertragung**

Die GPS-Daten sollen mittels LoRa an den Empfänger übertragen werden.

**/F0040/ Datenverwaltung**

Die GPS-Daten sollen in einer Datenbank gespeichert und dem Nutzer zur Verfügung gestellt werden.

**/F0050/ Daten abrufen**

Die GPS-Daten sollen graphisch in einer Applikation für Mobiltelefone und einer Website angezeigt werden können.

## Produktdaten

**/D0010/ Benutzerdaten**

* Name
* Passwort
* E-Mail

**/D0020/ Standortdaten**

* Zeit
* Längengrad
* Breitengrad
* Kuh Id

## Produktleistungen

**/L0010/ Benutzerfreundlichkeit**

Die Bedienung der Smartphone Applikation soll möglichst einfach und selbsterklärend sein

**/L0020/ Wetterbeständigkeit**

Die Sendeeinheit soll wind- und wasserfest sein

**/L0030/ Genauigkeit**

Die GPS-Daten sollen auf ca. 2m genau sein

**/L0040/ Laufzeit**

Die Stromversorgung der Sendeeinheit soll ca. 100 Tage ohne Batteriewechsel funktionieren

## Qualitätszielbestimmungen

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Sehr wichtig** | **Wichtig** | **Weniger wichtig** | **Unwichtig** |
| **Benutzerfreundlichkeit** |  | **x** |  |  |
| **Wetterbeständigkeit** | **x** |  |  |  |
| **GPS- Genauigkeit** |  | **x** |  |  |
| **Laufzeit** | **x** |  |  |  |

## Globale Testszenarien und Testfälle

**/T0010/ Empfang der Standortdaten**

Die Sendeeinheit kann eine Verbindung zu einem Satelliten aufstellen und die Standortdaten ermitteln

**/T0020/ Speichern auf SD-Karte**

Die GPS-Daten werden in einem bestimmten Intervall auf einer SD-Karte gespeichert.

**/T0030/ Datenübertragung**

Die GPS-Daten werden mittels LoRa an den Empfänger übertragen.

**/T0040/ Datenverwaltung**

Die GPS-Daten werden in einer Datenbank gespeichert und sind für die Applikation und die Website zugänglich.

**/T0050/ Daten abrufen**

Der Standort wird graphisch in einer Karte sowohl in der Applikation für Mobiltelefone als auch in der Website angezeigt.

# Organisation - Projektmanagement

## Projektteam

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Individuelle Themenstellung** |  | **Klasse** | **Arbeitsaufwand** |
| **Samuel Putz** | Entwurf der Hardware-Komponenten |  | 5BHEL | 180 Stunden |
| **Simon König** | Entwurf der Systemsoftware und der Datenbank |  | 5BHEL | 180 Stunden |

## Individuelle Aufgabenstellungen inkl. Arbeits- und Terminplan

### Samuel Putz

Ein Bild, das Text, Screenshot, Zahl, parallel enthält.

Automatisch generierte Beschreibung

Abbildung 4: GANT-Diagramm Putz

### Simon König

Ein Bild, das Text, Screenshot, Zahl, Quittung enthält.

Automatisch generierte Beschreibung

Abbildung 5: GANTT-Diagramm König

# Grundlagen und Methoden

Das System kann in vier grundlegende Teile aufgeteilt werden. Die Sendeeinheit ist auf der Kuh befestigt. Diese empfängt die Standortdaten und speichert sie auf einer SD-Karte. Ist ein LoRa-Modul vorhanden, übermittelt diese die Daten an die Basisstation. Die Basisstation empfängt die Daten und schreibt diese mit http-Requests auf den Server. Der Server speichert die Daten und stellt sie für die Applikation wieder zur Verfügung. Der Ablauf und die dafür verwendeten Komponenten wurden in einem Blockdiagramm (siehe Abb 6.) zusammengefasst.

Ein Bild, das Text, Screenshot, Diagramm, Schrift enthält.

Automatisch generierte Beschreibung

Abbildung 6: Blockdiagramm des C-Track Systems

## Platine

Die Sendeeinheit, wird am Hals der Kuh angebracht und in einem wasserfesten Gehäuse sicher verschraubt. Das Elara-1 GPS-Modul stellt eine Verbindung mit einem Satelliten her, um GPS-Daten zu empfangen. Diese Daten werden sowohl auf der SD-Karte gespeichert als auch über das RFM95W LoRa Breakoutmodul an die Basisstation übertragen. Die zentrale Steuereinheit der Platine ist ein energieeffizienter ATmega644 PA Mikrocontroller. Zur Gewährleistung einer zuverlässigen Stromversorgung der Bauteile wurden mehrere Spannungsregler integriert.

### Energie Versorgung

Das Ziel des Projekts ist es, einem Landwirt auf der Vierkaseralm während der Sommersaison von 3 Monaten dabei zu unterstützen, die Kühe rund um die Uhr zu überwachen und dem Landwirt deren Position zu übermitteln. Um die Funktionsfähigkeit für diesen Zeitraum zu gewährleisten sowie eine gute und stabile Stromversorgung sicherzustellen, musste die Entscheidung zwischen einem Akkumulator und einer herkömmlichen Batterie getroffen.

Ein Akkumulator bietet den Vorteil, dass er wiederaufladbar ist und somit die Notwendigkeit eines regelmäßigen Austauschs vermeidet. Aus folgenden Gründe wurde eine Batterie jedoch für unser Projekt als die effektivere Variante angesehen.

Zum einen bietet eine Batterie eine höhere Energiedichte als ein Akkumulator, was für die kurzeitig hohen Stromspitzen des GPS-Moduls und des Lora Breakout Boards erforderlich ist. Zum anderen ist eine Batterie nicht so anfällig für Temperaturschwankungen wie ein Akkumulator. [1]

Ein Akkumulator benötigt zusätzliche Regler, um vor Tiefenentladung geschützt zu sein. Zum Aufladen wird zudem eine zusätzliche Ladeschaltung sowie ein Port benötigt. Dies erhöht die Komplexität der Schaltung durch eine höhere Anzahl an Bauteilen und steigert dadurch wieder die Kosten. Ein Ladeport erschwert zudem die Wasserdichtheit des Gehäuses, wenn dieser von außen zugänglich sein sollte.

### Spannungsregler

Eine Batterie verliert im Laufe ihrer Lebensdauer immer mehr an Spannung. Die Bauteile auf der Platine müssen jedoch gemäß den Spezifikationen mit einer konstanten Spannung von 3,3 V versorgt werden. a die Batterie im geladenen Zustand einen Spannungsbereich von 3,6 V und im entladenen Zustand 2,7 Volt aufweist, wird ein Spannungsregler benötigt. Für unsere Zwecke wurde ein Buck-Boost-Konverter als die optimale Lösung angesehen, da diese Schaltung die Spannung bei Bedarf sowohl herunter- als auch hinaufregeln kann.

#### Buck Converter

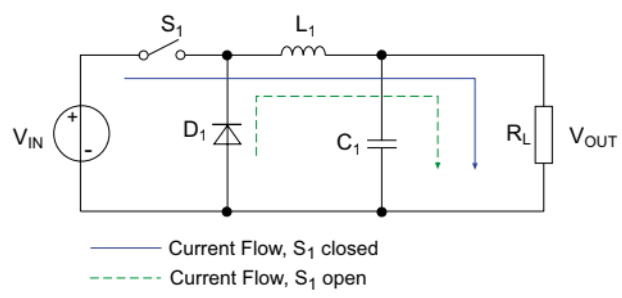
Der Buck Converter ist ein DC - DC - Wandler, um höhere Eingangsspannungen in kleinere Ausgangsspannungen umzuwandeln. Der Vorteil dieser Schaltung liegt in ihrer einfach gehaltenen Bauweise und einer dadurch verringerte Bauteilanzahl. Zudem besitzt der Buck Converter einen höheren Wirkungsgrad als Linearregler.

Abbildung 7: Buck Converter Schaltung

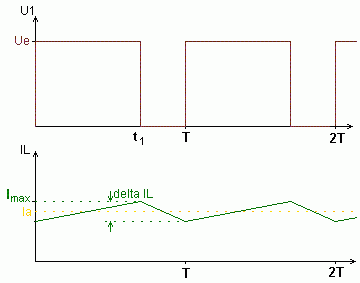
Typischerweise ist der Schalter ein MOSFET. Wenn der Schalter sich schließt, fließt Strom über die Spule und Kondensator von Plus nach Minus. Wird der Schalter geöffnet dreht sich die Spannung an der Spule durch die Selbstinduktion um. Der Strom fließt über den Kondensator weiter über die Diode, worüber der Strom zirkulieren kann und sich abbaut. Durch schnelles wiederholtes Öffnen und Schließen des Schalters kann so die Spannung auf die gewünschte Ausgangsspannung herunter geregelt werden.

Abbildung 8: Strom und Spannungsverlauf des Buck Converters

Es ist zu erkennen, dass ein sägezahnähnlicher Spulenstrom entsteht. Dies ist auf die Funktionsweise des Kondensators zurückzuführen, da dieser sich gemäß einer e-Funktion lädt und entlädt. Durch das schnelle Ein- und Ausschalten des Schalters befindet sich die Ladekurve jedoch in einem Bereich, in dem die Änderung so klein ist, dass sie quasi linear wirkt und somit den sägezahnähnlichen Stromverlauf erzeugt.

Der Wirkungsgrad liegt typischerweise bei um die 90 Prozent. Verluste in Form von Leitungsverlusten treten bei Bauteilen wie der MOSFET, der Diode und der Spule auf. Beim MOSFET entstehen zusätzlich weitere Verluste durch den wechseln zwischen Ein- und Ausgangszuständen. Am Kondensator treten sogenannte Kapazitive Verluste auf, da beim Laden und Entladen Energie abgegeben wird.

##### Dimensionierung:

Bei der Berechnung spielt die Spule eine entscheidende Rolle, da sie so dimensioniert werden muss, dass sie ausreichend Energie speichern kann, um die gewünschte Ausgangsspannung zu erreichen.

Die Auswahlkriterien beziehen sich dabei auf die Schaltfrequenz ( in der der Schalter ein und ausschaltet, dem Ausgangsstrom () und der gewünschten Ausgangsspannung (), sowie der Eingangsspannung ().

Die Stromwelligkeit wird üblicherweise mit ungefähr 20% des Ausgangsstroms dimensioniert.

Wählt man die Stromwelligkeit zu klein erhöht es den Wert der Spule, was zu höheren Kosten führen kann.

Die Spule wird mit Folgender Formel dimensioniert:

#### Boost Converter

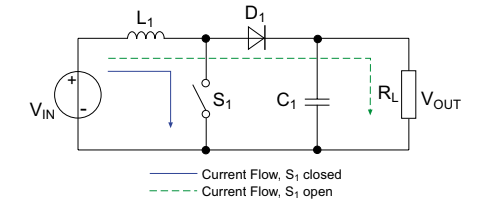
Der Boost Converter ist wie der Buck Converter ein DC-DC-Wandler, welcher diesmal jedoch eine kleinere Eingangsspannung zu einer höheren Ausgangsspannung transformiert. Er besitzt die gleichen Baukomponenten eines Buck Converters und bringt somit dieselben Vor- und Nachteile wie dieser.

Abbildung 9: Boost Converter

Der Schalter ist hier ebenfalls typsicherweise ein MOSFET. Wenn der Schalter geschlossen wird, fließt der Strom über die Spule und den geschlossenen Schalter von Plus nach Minus. Wird der Schalter geöffnet so fließt der Strom durch die Diode und weiter über den Kondensator zurück nach Minus. Durch diesen Wechsel dreht sich die Spannungsrichtung der Spule um und addiert sich zur Eingangsspannung. Je kürzer der Abstand zwischen diesen Wechsel ist, desto höhere Ausgangsspannungen können erreicht werden. Dies führt jedoch zu einem geringeren Strom. Die Diode ist in diesem Fall wesentlich, um das Entladen des Kondensator zu verhindern.

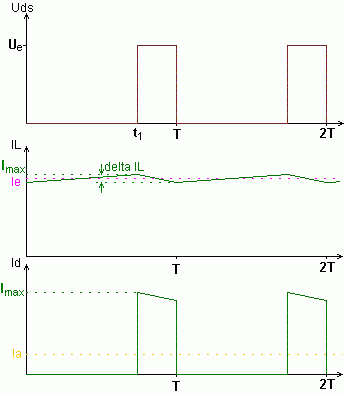
Wie beim Buck Converter lässt sich auch in diesem Fall der Sägezahn Verlauf des Induktionsstrom auf dasselbe Phänomen der Lade und Entladekurve des Kondensators zurückführen.

Abbildung 10: Strom und Spannungsverlauf des Boost Converters

Durch die gleichen Bauteile wie in einem Buck Converter ergibt sich ein ähnlicher Wirkungsgrad von rund 90 Prozent, erklärbar durch dieselben Verluste.

##### Dimensionierung:

Ähnlich zum Buck Converter beziehen sich die Auswahlkriterien erneut auf die Schaltfrequenz ( in der der Schalter ein und ausschaltet, dem Ausgangsstrom () und der gewünschten Ausgangsspannung (), sowie der Eingangsspannung ().

Auch hier wird die Stromwelligkeit mit 20% Prozent des Ausgangsstromes gewählt (1)

Mit Folgender Formel wird die Spule des Boos Converters dimensioniert:

#### Buck Boost Converter

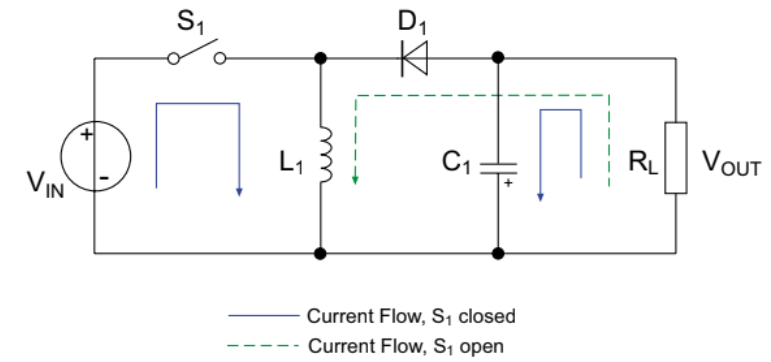
Will man die Vorteile und Effizienz beider Schaltungen in einer Kombinieren entsteht der Buck - Boost Converter. Mit der gleichen Bauteilanzahl eines normalen Buck oder Boost Converters kann dieser sowohl eine niedrige Eingangsspannung hochregeln als auch eine hohe Eingangsspannung herunterregeln.

Abbildung 11: Buck - Boost - Converter

Wird der Schalter geschlossen fließt der Strom über die Spule da die Diode in Sperrichtung ist, von Plus nach Minus. Wird der Schalter geöffnet wird die Spule durch ihre Selbstinduktion von einem Verbraucher zu einem Generator und die Spannung dreht sich um. Diesmal fließt der Strom weiter über den Kondensator und über die in Durchlassrichtung betriebene Freilaufdiode, um den Stromfluß aufrecht zu erhalten. Dabei lädt sich der Kondensator so auf das am Ausgang eine negative Spannung abgegriffen werden kann.

Der Buck Boos Converter besitzt zwei Operationsmodi [2]:

* **Continuous Conduction Mode**: In diesem Modus wird der Induktionsstrom nie Null und bleibt positiv. Dies bringt eine höhere Effizienz sowie eine glattere Wellenformern. Der Nachteil bei diesem Modus liegt darin, dass ein höherer Induktivitätswert benötigt, wird
* **Discontinuous Conduction Mode:** Bei diesem Modus kann es dazu kommen, dass, während den Schaltzyklen kurz kein Induktionsstrom fließt. Höhere Spannungsripple, höhere Schaltverluste und insgesamt weniger Effizienz ist dadurch die Folge. Jedoch wird ein niedrigerer Induktionswert benötigt.

#### TPS 63001 DRCT

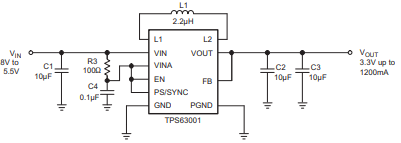
Für unsere Platine entschieden wir uns für einen TPS 63001 Buck Boost Converter. Bei einer Eingangsspannung von 1,8 – 5,5V und einer Ausgangsspannung von 3,3 V entsprach dieser Chip genau unseren Ansprüchen. Zudem bietet er einen Wirkungsgrad von bis zu 96% und liefert einen hohen Ausgangstrom der sowohl für das GPS-Modul als auch den Microcontroller benötigt wird.

Abbildung 12: Typische Applikation des TPS 63001 [3]

Bei der TPS 6300x Reihe kann man mit Widerständen am Ausgang die gewünschte Ausgangsspannung nach Belieben konfigurieren. Für unsere Zwecke wurde der TPS 63001 gewählt da dieser schon so eingestellt ist das er immer auf die 3,3 V regelt und zudem einen Ausgangsstrom von bis zu 1,2 A ausgibt. Die weiteren Bauteilwerte wurden aus dem Datenblatt entnommen [3].

Ein Bild, das Text, Reihe, Diagramm, Schrift enthält.

Automatisch generierte BeschreibungEin Bild, das Text, Screenshot, Reihe, Diagramm enthält.

Automatisch generierte BeschreibungUm die Funktionalität unseres Buck Boost Converters zu überprüfen, wurde eine Messung der Ausgangripples unternommen.

Abbildung 13: Messung der Ausgangripples an der Platine

Abbildung 14: Messungen des Datenblattes [3]

Bei einer Eingangsspannung von 5V ergab sich ein Spannungsripple am Ausgang mit einem Wert von 9,2mV. Aus der Spezifikation ergibt sich ein Spannungsripple am Ausgang von um die 10mV bei einer Eingangsspannung von 4,2V. Vergleicht man nun unsere Messung mit der Spezifikation ist zu erkennen das unsere Buck Boost Schaltung der Spezifikation nahezu ident ist und somit die Funktionalität gegeben ist

### GPS-Modul

Für die Standortermittlung wurde das von Würth Elektronik entwickelten Elara-I GPS-Modul verwendet. Das Modul kann aufgrund der integrierten Antenne ohne aufwendige Beschaltung in die Platine integriert werden und ist aufgrund des kleinen Formfaktors gut für Tracker geeignet.

Die integrierte Antenne hat eine hohe Empfindlichkeit und kann auch schwache Signale auswerten. Die Position wird entweder über das GPS oder das GLONASS-Satellitensystem auf bis zu 1,5m genau ermittelt. [4]

#### Startvorgang

Nach Versorgung mit Spannung beginnt das Modul mit dem Startvorgang. Ist dieser beendet, sendet das Modul einen kurzen Puls am WAKE\_UP-Pin und wechselt in den Stand-by-Modus. Um mit der Kommunikation zu beginnen, muss Puls, welcher mindestens 100µs lang ist, an den ON\_OFF-Pin gesendet werden. [4]

#### Kommunikation

Der Datenaustausch mit dem Elara-1 Modul erfolgt über eine UART-Schnittstelle. Die Daten für die Kommunikation werden standardmäßig im NMEA-Sentences übertragen. Für komplexere Funktionen muss die Kommunikation auf den OSP-Sentences umgeschaltet werden. Nach dem Start beginnt das Elara-1 Modul Daten in NMEA-Sentences zu senden.

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

Automatisch generierte BeschreibungNMEA-Sentences sind in ASCII codiert und Folgen in der Struktur aus Abbildung 15 aufgebaut. NMEA-Sentences enthalten die Daten in Feldern. Die einzelnen Felder sind mit Beistrichen getrennt.

Abbildung 15: Aufbau einer NMEA-Satz [4]

Das Elara-Modul versendet Daten über die Position, über die verwendeten Satelliten, über die Uhrzeit und das Datum und über die Geschwindigkeit und Richtung. Die Daten werden in die Felder aufgeteilt und in einzelnen Sätzen übertragen. Welche Informationen im übertragenen Satz sind, ist vom Data-Format abhängig. Jedes Datenformat liefert bestimmte Informationen. Mit dem ZDA-Format erhält man nur Uhrzeit und Datum, das GLL-Format liefert Längen- und Breitengrad. Eine Liste mit allen unterstützten Formaten findet man im Datasheet. Je nach benötigten Informationen können die Formate ein- und ausgeschalten werden.

Um alle Funktionen und Einstellungen, des Moduls nutzen zu können, muss das Modul in den OSP-Modus versetzt werden. OSP-Sentences sind ein von Qualcomm entwickeltes Format, um direkt mit dem Chip zu kommunizieren. Statt mit ASCII Werten arbeitet das System mit Hexadezimalzahlen. Daten werden in binärer Form übertragen. [4]

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

Automatisch generierte Beschreibung

Abbildung 16: Aufbau eines OSP-Satz [4]

OSP-Sätze beginnen mit einer Startsequenz. Die Daten sind bei OSP in Messages aufgeteilt. Die Messages sind mit den Datenformaten aus NMEA vergleichbar. Ähnlich wie bei NMEA wird der Satz mit einer Prüfsumme und einer Endsequenz beendet.

Der OSP-Modus sendet detailliertere Daten über das System und die Positionsermittlung. Anders als bei NMEA können die nicht benötigten Informationen nicht deaktiviert werden. Durch den höheren Verarbeitungsaufwand entstanden Probleme bei der Datenverarbeitung. Deshalb wurde für die Kommunikation der NMEA-Modus gewählt. [4]

Alle für das Projekt relevanten Informationen sind im RMC-Format enthalten. Alle anderen Sätze werden daher deaktiviert. Aus dem RMC-Format erhält man Datum, Uhrzeit, Längen- und Breitengrad, sowie Bewegungsgeschwindigkeit und Richtung. Ein typischer NMEA-Satz mit Daten im RMC-Format sieht wie der Satz in Abbildung 17 aus.



Abbildung 17: NMEA-Sentence mit RMC-Format

Mit der Tabelle aus Abbildung 18 kann man die benötigten Informationen aus einer NMEA-Sentece mit Daten im RMC-Format auslesen. [4]

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Abbildung 18: RMC-Format [4]

Um die Information über Längen- und Breitegrad in Anwendungen wie Google Maps, oder der flutter\_map Bibliothek nutzen zu können, werden diese in Dezimalgrad umgerechnet. Dazu teilt man den gegebenen Wert in Grad und Minuten auf. Die Minuten können mit einer Division durch 60 in Grad umgerechnet werden und können dann zum Gradwert addiert werden. Dezimalgrad berücksichtigen die Information über die Halbkugel. Ist der Längengrad auf der Südhalbkugel oder der Breitengrad auf der Westhalbkugel muss man der Zahl ein negatives Vorzeichen geben.

Für den Satz aus Abbildung 17 ergeben sich dadurch folgende Berechnungen.

Da der Punkt auf der Nordhalbkugel und auf der Osthalbkugel liegt, sind beide Werte positiv.

#### Trickle Power mode

Das Elara-1 verfügt über spezielle Modi für das Sparen von Energie. Für das Projekt interessant ist der Trickle Power mode. In diesem wechselt das Modul in einem festgelegten Zeitintervall automatisch aus dem Stand-By in den Empfangsmodus. Nach dem Empfangen der Daten wechselt das Modul in den CPU-Modus, um die Informationen auszuwerten. Wurden alle gesammelten Informationen gesendet, wechselt das Modul wieder in den Stand-By Modus. [4]

Ein Bild, das Text, Screenshot, Diagramm, parallel enthält.

Automatisch generierte Beschreibung

Abbildung 19: Trickle Power mode [4]

Der Modus funktioniert nur, wenn mit OSP-Sentences kommuniziert wird. Da die Kommunikation mit OSP-Sentences nicht zufriedenstellend implementiert werden konnte wird dieser Modus nicht verwendet.

#### Design Rules

Das wichtigste beim Elara 1 Modul ist die Keep Out Area, die im Datenblatt beschrieben wird. Diese ist in einem 3mm Radius um das Modul herum. Um eine symmetrische Ground Flächen Verteilung zu gewährleisten, soll das Modul in der Mitte der Platine platziert werden. Um eine bestmögliche Erdung zu erreichen, wird geraten eine reine Ground Plate am Bottomlayer einzufügen [4].

### Mikrocontroller

Als Steuerungselement unserer Platine wurde ein ATmega644 PA gewählt. Da er eine vielseitige Auswahl an Schnittstellen bietet, leistungsfähig ist und sehr stromeffizient arbeitet, ist dieser Chip die perfekte Wahl. Zudem wurde der ATmega644 im Schulunterricht intensiv behandelt, was den Einstieg wesentlich erleichterte.

#### Schnittstellen

Für unser Projekt wurde die SPI und UART-Schnittstelle verwendet

* **SPI:** Diese Schnittstelle wurde für das LoRa Breakout Modul sowie die SD-Karte verwendet. SPI wird grundsätzlich für den Datenaustausch zwischen Microcontroller und anderen Peripherie Geräten wie etwa Sensoren oder Speicherchips verwendet. Der Vorteil von SPI liegt an ihrer hohen Geschwindigkeit und der einfachen Konfiguration
* **UART:** Die UART-Schnittstelle wird in unserem Projekt für die Kommunikation und Ansteuerung des Elara 1 Moduls verwendet. UART wird typischerweise für die Kommunikation zwischen Microcontroller und externe Geräte wie Computer oder andere Microcontroller verwendet. Es bietet eine flexible und zuverlässige Datenübertragung über lange Strecken.

#### Interner Quarz Oszillator

Der ATmega644 verfügt über einen internen Quarz, wodurch die Notwendigkeit eines externeren Quarzes oder eines Oszillators entfällt. Dadurch verringert sich die Komplexität der Schaltung, was die Zuverlässigkeit erhöht und durch die verringerte Bauteilanzahl auch stromsparend wirken kann, sowie Preislindert wirkt.

#### Stromverbrauch

Der ATmega644 ist für den Betrieb mit Batterien optimiert und verfügt dadurch über verschieden Stromsparmodi. Dies erhöht die Lebensdauer der Batterie und maximiert die Effizienz. Stromsparmodi wie etwa der Power-down-, Standby-, oder Sleep Modus können den Stromverbrauch verringern, indem sie interne Schaltkreise oder Funktionen deaktivieren. Zusätzlich ist es möglich Peripheriegeräte selektiv zu aktivieren, was wiederum einen stromsparenden Effekt hervorruft. Die Taktfrequenz des Microcontrollers kann dynamisch angepasst werden, was ebenso stromsparend wirkt. Besonders hervorhebend ist der niedrige Ruhestrom des Standby Modus, welcher mit dem schon ohnehin niedrigen Stromverbrauch perfekt für unser Projekt ist.

### SD-Karte

Mit dem Tracker sollte das Herdenverhalten von Kühen analysiert werden. Dafür sollte der Standort lückenlos alle 5 Minuten gespeichert werden. Ausgewertet werden die Daten am Ende der Almsaison.

Damit die Daten gespeichert werden, selbst wenn keine Funkverbindung besteht, ist eine lokale Speicherung erforderlich. Als Speichermedium wurden MicroSD-Karten gewählt. MicroSD-Karten sind die weit verbreitet und lassen sich mit SPI gut in das System einbinden. Durch den kleinen Formfaktor sind MicroSD-Karten gut für einen Tracker geeignet. Der einzige Nachteil ist die kurze Lebensdauer der Speicherzellen.

Für die leichte Auswertung der Daten vom Nutzer ist es erforderlich, die Daten in eine Datei auf einem Dateisystem zu speichern. Aufgrund der Kompatibilität mit vielen Betriebssystemen wurde das Fat32 Dateisystem gewählt.

Da bei jedem Punkt die gleichen Daten entstehen, werden diese in tabellarischer Form im CSV-Format gespeichert. Es werden die Daten von einem Zeitpunkt in einer Zeile gespeichert. Die einzelnen Daten werden mit einem Beistrich getrennt. Aufgrund der einfachen Struktur können die Daten leicht in eigenen Programmen ausgewertet werden.

#### Abnutzung

Das von Microsoft entwickelte FAT-Dateisystem. Arbeitet mit einer Tabelle, die in den ersten Zellen des Speichers liegt. In dieser sind Informationen über Speicherort, Größe und Änderungszeiten der Dateien. Sobald in eine Datei geschrieben wird, muss diese Tabelle aktualisiert werden. [5] Dadurch werden die ersten Zellen stärker beansprucht, was bei Flash Speicher ein Problem ist.

Die Flash-Speicherzellen einer SD-Karte können, je nach Bauform, 1-3 Bit speichern. Singe Level Cells, kurz SLC, können nur 1 Bit speichern. Dafür bieten diese höhere Lese- und Schreibgeschwindigkeiten und einen niedrigeren Leistungsverbrauch. Multi- oder Triple Leve Cells bieten mehr Speicherplatz auf der gleichen Fläche, können aber nicht so oft beschrieben werden. Während SLCs oft bis zu 100.000 Schreibzyklen überstehen können TLCs schon nach 3000 Schreibvorgängen ausfallen. [6] [7]

Um dem entgegenzuwirken, verbauen viele Speicherhersteller einen sogenannten wear leveling Algorithmus. Dieser unterteilt den Speicher in Blöcke. Der Nutzer schreibt nicht direkt auf den Speicher, sondern in einen logischen Block. Werden Daten in einen Block geschrieben, wird der Speicherbereich mit den wenigsten Schreibzyklen beschrieben und eine Verknüpfung vom logischen Block zu dem physischen Speicherblock erstellt. Dadurch werden die Schreibzyklen auf alle Zellen verteilt und die Lebensdauer erhöht. [6]

Damit die Daten sicher auf die SD-Karte geschrieben werden können, sollte eine SLC SD-Karte mit wear-leveling verwendet werden. Diese Karten werden oft mit dem Zusatz „Für industrielle Anwendungen“ verkauft.

Um die Abnutzung zu reduzieren, wurde geplant, die Standortdaten in einen Buffer zu speichern und nur einmal pro Tag auf die SD-Karte zu schreiben. Aus zeitlichen Gründen konnte dies nicht implementiert werden.

#### Initialisierung

Die einfachste Möglichkeit eine SD-Karte anzusteuern ist der SPI-Modus. Für diesen werden nur die 4 SPI-Leitungen und eine Versorgung benötigt. Um mit allen SD-Karten kompatibel zu sein, muss für den Startprozess eine Frequenz zwischen 100kHz-400kHz verwendet werden. Nach dem Start kann die Frequenz angehoben werden. Die maximale Frequenz variiert nach Hersteller und kann aus dem CRD-Register ausgelesen werden.

Die Kommunikation erfolgt mit Befehlen. Die ersten beiden Bits müssen zuerst 0, dann 1 sein. Ein Befehl ist, mit Ausnahme der ACMD-Befehle und des CMD55 Befehls, ein Byte lang. Diesem Befehl werden die benötigten Argumente mitgegeben. Die wichtigsten Befehle für die Kommunikation werden in Abbildung 20 zusammengefasst.

Ein Bild, das Text, Screenshot, Zahl, Schrift enthält.

Automatisch generierte Beschreibung

Abbildung 20: Befehle für die Kommunikation mit SD/MMC-Karten (vgl. [8])

Als Antwort kommt in den meisten Fällen eine R1-Response. Diese enthält Informationen über die Verarbeitung des Befehls und den Status des Speichermediums und ist ein Byte lang (vgl. Abbildung 21). Bei fehlerfreier Verarbeitung sind alle Bits 0.

SD-Karte R1 Antwort


Abbildung 21: R1-Response [8]

Eine Sonderform der R1-Antwort ist die R1b Antwort. Diese wird versendet, wenn die Verarbeitung noch nicht abgeschlossen wurde. Für die Dauer des Prozesses wird MISO auf LOW gesetzt. Nach Erhalt von 0xFF auf der Datenleitung ist die Verarbeitung beendet.

Werden Daten von der SD-Karte ausgelesen kommt eine R3 oder R7 Antwort. Diese besteht aus einer R1-Antwort und 32 zusätzlichen Datenbits.

Nachdem die SD-Karte mit Spannung versorgt wurde, dauert es ca. 1ms., bis sich der interne Zustand stabilisiert und alle Speicherzellen initialisiert sind. Zu beachten ist, dass die Karte zu dem Zeitpunkt noch kein SPI kompatibles Gerät ist und sich nicht an Chipselect hält. Bevor die Kommunikation gestartet werden kann, benötigt die SD-Karte mindestens 74 Clock Signale. Chipselect und MOSI müssen dabei HIGH sein. Die Karte sollte danach dazu bereit sein, die native Befehle zu verarbeiten.

Um die SD-Karte zu resetten und alle Einstellungen für die Kommunikation zu löschen wird der Befehlt GO\_IDLE\_STATE gesendet. Zu beachten ist, dass der SPI-Modus noch nicht aktiviert ist und eine CRC-Prüfsumme mitgeschickt werden muss. Nach dem Reset testet die Karte das Chipselect Signal. Ist dieses LOW wird der SPI-Modus aktiviert. Als Antwort auf den GO\_IDLE\_STATE Befehl kommt eine R1-Antwort. Wichtig ist, dass das Idle-Bit gesetzt ist.

Um den vollen Funktionsumfang zu nutzen, muss noch, abhängig der Version des SD-Standards, ein SEND\_OP\_COND-Befehl oder ein APP\_SEND\_OP\_COND-Befehl gesendet werden. Um mit allen Versionen kompatibel zu sein, wird empfohlen einen Befehlt zu senden und die Antwort auszuwerten. War der Befehlt nicht erfolgreich muss der andere Befehl verwendet werden. Die SD-Karte verlässt den Idle-Zustand wieder und kann ab jetzt beschrieben und gelesen werden. [8]

#### Dateisystem

Damit die Dateien auf Möglichst vielen Geräten ausgewertet werden können wird das Fat32 Dateisystem verwendet. Da das Dateisystem bereits 1996 existiert, gibt es einige Nachteile. Im Dateisystem sind keine Schutzmechanismen gegen ausfallende Zellen verbaut. Daten sind bei defekten Speicherzellen verloren. Zudem kann eine Datei kann maximal 4GB groß sein. [9]

Eine Im CSV-Format gespeicherte Zeile mit Informationen zu Datum, Uhrzeit, Längen- und Breitengrad ist 39 Zeichen lang. Die Textdatei ist im UTF-8 Format kodiert. Ist das zu speichernde Zeichen ein ASCII-Zeichen benötigt man ein Byte um dieses zu Speichern. Alle Zeichen, die für die Speicherung der Daten verwendet werden, sind aus dem ASCII-Zeichensatz. Die Daten werden alle 5 Minuten über die Saison, die meistens um die 100 Tage lang ist, gespeichert. Dadurch lässt sich die Dateigröße berechnen.

Die eingeschränkte Dateigröße stellt somit kein Problem dar.

#### FatFS

Aufgrund der hohen Komplexität von Dateisystemen wurde das FatFS-Modul für das Lesen und Schreiben in Dateien verwendet. Da es speziell für Microcontroller entwickelt wurde, benötigt es wenig Arbeitsspeicher. Durch Konfigurationen in der ffhconf.h kann der Verbrauch des Programmspeichers gesenkt werden.

Für das Projekt wurden der Parameter FF\_FS\_MINIMIZE auf drei gesetzt. Dadurch verliert man Funktionen zum Umbenennen, Löschen und Erstellen von Ordnern, sowie zur Suche von Dateien.

Um die Daten formatiert auf die SD-Karte schreiben zu können wurde FF\_USE\_STRFUNC auf zwei gesetzt. Dadurch kann man die *f\_printf* Funktion nutzen. Diese Funktion funktioniert wie die *printf* Funktion aus C und kann einen String formatiert in eine Datei speichern. Mit der FF\_PRINT\_FLOAT flag wurde die Unterstützung für Gleitkommazahlen aktiviert.

Das Modul hat jedoch keine Möglichkeiten mit Speichermedien zu kommunizieren. Die Funktionen dafür müssen bereitgestellt werden. Um den Prozess zu erleichtern, werden Beispielprojekte zur Verfügung gestellt. Als Ausgangspunkt wurde das Generic-SD Beispiel verwendet. In diesem wurden die Funktionen für die Kommunikation mit SD-Karten ausprogrammiert. Die Funktionen wurden an den Microcontroller angepasst und der Code für die Software-SPI Kommunikation durch Code für das SPI-Modul des Microcontrollers ersetzt.

### Gehäuse

Bevor das Layout der Platine entworfen wurde, mussten wir uns für ein Gehäuse entscheiden. Der ursprüngliche Plan sah vor, ein 3D-Gehäuse selbst zu entwerfen und zu drucken. Das Gehäuse muss wind- und wetterfest sein, um im Freien auf der Alm zu funktionieren. Daher wurde der Plan für ein selbstgedrucktes Gehäuse aufgegeben, da es einen erheblichen Zeit- und Kostenaufwand erfordern würde, um ein 3D-gedrucktes Gehäuse wasserdicht zu machen. Aus diesem Grund entschieden wir uns für ein gekauftes Gehäuse.

#### Polycarbonat Gehäuse

Ein Bild, das Text, Screenshot, Zahl, Schrift enthält.

Automatisch generierte Beschreibung Es wurde ein Polykarbonat-Gehäuse von RS Components mit der Schutzklasse IP65 ausgewählt. [10]

Abbildung 22: IP- Schutzklassen

Basierend auf den Informationen in der oben genannten Tabelle (Abb.: 22) entschieden wir uns dafür, dass die IP-Schutzklasse 65 für unsere Anforderungen perfekt geeignet ist. Somit ist das Gehäuse effektiv gegen Staub und andere Fremdkörper geschützt und bietet zudem Schutz vor Strahlwasser, was für den Betrieb auf der Alm ausreichend sein wird.

Nach der Auswahl des Gehäuses wurden die Maße der Platine so festgelegt, dass sie fest und sicher im Gehäuse verschraubt werden kann.

#### Luftöffnung [11]

Eine Luftöffnung für ein Gehäuse ist aus mehreren Gründen wichtig. Zum einen ermöglicht sie eine effektive Wärmeableitung und kühlt somit die Bauteile. Zum anderen reduziert sie Feuchtigkeit und verhindert Kondensation im Gehäuse, während sie gleichzeitig einen Druckausgleich zwischen Gehäuse und Umwelt ermöglicht. Wir haben uns für eine M12 Industrial Luftöffnung mit der Schutzklasse IP68 entschieden. Diese gewährleistet einen guten Luftaustausch und schützt gleichzeitig vor dem Eindringen von Wasser und Fremdkörpern.

### Programmablauf

Nach dem Start des Microcontrollers werden die benötigten Variablen erstellt und die Pins und die Seriellen Kommunikationsschnittstellen SPI und UART aktiviert konfiguriert.

Mit einem Puls am ON\_OFF-Pin wird das GPS-Modul eingeschalten. Über UART werden die nicht benötigten NMEA-Sätze deaktiviert.

/\*

The GPS Module is default in stand-by mode

wake up with pulse on PD4

\*/

PORTD |= (1 << PD4);

*\_delay\_ms*(100);

PORTD &= ~(1 << PD4);

uart\_sendStr("$PSRF103,1,0,0,1\*25\r\n");  
…  
uart\_sendStr("$PSRF103,9,0,0,1\*2D\r\n");

Um die SD-Karte und LoRa nutzen zu können, werden die beiden Geräte mit dem SPI-Bus initialisiert. Wichtig ist, dass die SD-Karte nach dem Start kein SPI-Gerät ist und sich nicht an Chipselect hält. Um Probleme bei der Datenübertragung an SPI-Geräte zu vermeiden, muss die SD-Karte als erstes initialisiert werden. (vgl. 4.1.5.2)

f\_mount(&FatFs, "", 0);

lora = lora\_init(); // LoRa init

lora\_setSpreadingFactor(12);

lora\_set\_bandwidth(0b0111); // 125 kHz

lora\_putd("Tracker started", 15);

In einer While-Schleife wartet das Programm auf Daten von der UART-Schnittstelle. Wird ein Satz empfangen, liest das Programm bis zum Satzende mit ´\r\n´.

while (1)

{

while ((buff[++i] = uart\_receive()) != '\n');

buff[i] = '\0';

Die Daten werden mit einer Funktion aus dem Satz ausgelesen und in ein dafür erstelltes Struct geschrieben. Ist der Satz gültig gibt die Funktion eine logisches True zurück und schreibt den Satz im CSV-Format in eine Datei.

if (parseRMC(buff, &p))

{

fr = f\_open(&Fil, "pos.txt", FA\_OPEN\_APPEND | FA\_WRITE);

if (fr == FR\_OK)

{

f\_printf(&Fil, "%02d/%02d/%02d %02d:%02d:%02d;%f;%f\r\n", p.hour, p.minutes, p.seconds, p.day, p.month, p.year, p.lat, p.lng);

fr = f\_close(&Fil);

}

Bei der Initialisierung wird geprüft, ob die Platine ein LoRa-Modul hat. Ist ein LoRa-Modul vorhanden werden die Daten mit der sprintf-Funktion in einen String geschrieben und mit LoRa an die Basisstation übermittelt.

if (lora)

{

int len = *sprintf*(buff, "%02d %02d %02d %02d %02d %02d %02d %" *PRId32* " %" *PRId32* "", T\_ID,

p.hour,

p.minutes,

p.seconds,

p.day,

p.month,

p.year,

(*int32\_t*)(100000\* p.lat),

(*int32\_t*)(100000\* p.lng));

lora\_putd(buff, len);

}

}

## LoRa

LoRa, kurz für Long Range, ist ein von Semtech entwickeltes, patentiertes Funkübertragungsverfahren, welches hohe Reichweiten bei wenig Stromverbrauch erreichen kann. Dadurch konnte sich diese Technik vor allem IOT-Bereich durchsetzen.

LoRa nutzt die Lizenzfreien Bereiche im ISM-Band. Dadurch fallen keine Gebühren an. Durch die Modulationstechnik sind Überlagerungen mit anderen Teilnehmern in diesem Frequenzbereich kein Problem.

Mithilfe von Parametern wie dem Spreadingfactor und der Coding Rate kann die Übertragung entweder auf hohe Reichweite und robuste Übertragung oder schnelle Übertragungsraten angepasst werden. [12]

### Anforderungen

Der Standort sollte für den Landwirt spätestens jede ½ Stunde aktualisiert werden. Da der Tracker die ganze Saison ohne Batteriewechsel laufen sollte, muss stromsparend gearbeitet werden. Die Kühe können sich auf dem Gebiet der Alm frei bewegen, daher sollten Daten auch über mehrere Kilometer Entfernung stabil übertragen werden.

Eine Möglichkeit wäre, die Daten über das GSM-Netz zu übertragen. Im Gegensatz zu moderneren Mobilfunkstandards wie UMTS wird dieser durch seine weite Verbreitung im Industriebereich nicht abgebaut werden. Die niedrige Datenübertragungsrate ist kein Problem. Allerdings benötigen Module, die mit diesem Netz kommunizieren können, während des Sendevorgangs oft 2A Strom, was bei der Versorgung mit Batterien oder Akkus zum Problem wird. Zudem benötigt jedes dieser Module eine eigene SIM-Karte mit gültigem Tarif, was den Aufwand und die Nutzungskosten erhöht. [13]

Eine im IOT-Bereich weit verbreitete alternative ist LoRa. Bekannt ist es Aufgrund der störsicheren Übertragung über hohe Reichweiten und dem geringen Stromverbrauch. Gesendet wird im für Short-Range Devices zugewiesenen Frequenzbereich von 868MHz. Dadurch fallen für das Senden keine weiteren Lizenzgebühren an. Die größte Einschränkung bei LoRa ist eine niedrige Datenrate, die durch den Duty-Cycle, einer gesetzlich vorgegebenen maximalen Sendezeit, noch weiter gesenkt wird.

Bei der Datenübertragung mit LoRa muss zwischen LoRa, der Funktechnik, und LoRaWAN, einem Netzwerkprotokoll, unterschieden werden. LoRaWAN basiert auf der LoRa Übertragung, bietet jedoch eine Netzwerkarchitektur mit Features für die Adressierung der Geräte und Verschlüsselung. In einem LoRaWAN-Netzwerk werden die Daten werden von einem Gateway empfangen und über das Internet an einen Server weitergeleitet. [14]

Diese Gateways können entweder selbst errichtet werden, oder von einem Anbieter gegen Bezahlung verwendet werden. Im Gebiet der Vierkaseralm gibt es noch keine öffentlichen LoRa Gateways, es müsste also ein eigenes Gateway errichtet werden.

Aufgrund der erhöhten Abstraktion von LoRaWAN müssen mehr Regeln zur Paketstruktur und zu den Einstellungen der Modulation eingehalten werden. Dadurch erhöht sich die Datenmenge und somit auch die benötige Leistung.

Da die Funktionen von LoRaWAN nicht benötigt werden, keine Gateways in der Nähe der Vierkaseralm sind und der Energieverbrauch am Tracker steigen würde, wurde auf LoRaWAN verzichtet und eine eigene Basisstation entwickelt mit der LoRa Funkübertragungstechnik entwickelt.

### Rechtliche Rahmenbedingungen

Für die Datenübertragung mit Funk benötigt man eine Lizenz für den Frequenzbereich, in dem die Übertragung stattfindet. Für private Anwendungen wie WLAN, Alarmanlagen oder andere Geräte mit Funkübertragung wurden Bereiche für short-range devices, kurz SRD, reserviert. Ausschlaggebend ist jedoch nicht die Reichweite, sondern die Sendeleistung und Sendedauer. LoRa verwendet das in Europa für SRD freigehaltene Frequenzband von 868MHz – 868,6MHz.

Um die Strahlungsbelastung gering zu halten und allen Funkteilnehmern die Kommunikation zu ermöglichen, wurde die Nutzung des Frequenzbereichs gesetzlich reguliert. Diese Regeln findet man in der „Entscheidung der Kommission vom 9. November 2006 zur Harmonisierung der Frequenznutzung durch Geräte mit geringer Reichweite“ unter dem Punkt „Funkgeräte mit geringer Reichweite für nicht näher spezifizierte Anwendungen“.

Es darf maximal mit einer Leistung von 25mW ERP gesendet für eine Dauer von 1% einer Stunde gesendet werden. Zusätzlich darf man in diesem Bereich keine Videoanwendungen betreiben. Die Reichweite der Funkverbindung spielt keine Rolle.

### Modulation

LoRa verwendet eine Chirp-Spread-Spectrum Modulation. Grundlage dafür ist ein Sinusförmiges Signal, dessen Frequenz sich linear ändert. Dieses Signal wird Chirp genannt. Abhängig von steigender oder fallender Frequenz wird in Up- und Down-Chirp unterschieden.

Ein Bild, das Reihe, Screenshot, Rechteck, Diagramm enthält.

Automatisch generierte Beschreibung

Abbildung 23: Chirp Signal im Zeitbereich [15]

Dieser Chirp kann mit dem Spreadingfactor und der Bandbreite beeinflusst werden. Mit dem Spreadingfactor kann berechnet werden, über wie viele Samples sich ein Chirp streckt. Die Bandbreite bestimmt den Frequenzbereich, in dem sich der Chirp bewegt. Je länger ein Chirp ist, desto besser lässt er sich auf der Empfängerseite auswerten. Mit einem Hohen Spreadingfactor erhält man somit eine gute Reichweite, muss aber eine langsamere Datenübertragung in Kauf nehmen.

Der Chirp bewegt sich zwischen und . Die Information wird mit einem Offset im Frequenzgang übertragen. Der Chirp beginnt bei der Offsetfrequenz. Erreicht der Chirp die maximale Frequenz springt er zu . Er endet wieder, sobald er die Offsetfrequenz erreicht hat. Durch die ständige Frequenzänderung sind Überlagerungen mit anderen Funkteilnehmern immer nur für einen kurzen Zeitraum. Das Signal ist daher sehr störsicher.

Die kleinste, nicht mehr teilbare, übertragbare Einheit nennt man ein Symbol. Ein Chirp überträgt jeweils ein Symbol. Die Zustände, die ein Symbol annehmen kann, sind aus der Menge . Nimmt man ein Symbol S aus dieser Menge kann mit der Formel 4 der Frequenzoffset für dieses Symbols berechnet werden.

Ein Bild, das Screenshot, Reihe, Farbigkeit, Text enthält.

Automatisch generierte Beschreibung

Abbildung 24.Frequenzverlauf eines LoRa-Signals [16]

Um das Signal zu demodulieren, werden die einzelnen Symbole mit einem Chirp ohne Offset und Frequenzgang in die entgegengesetzte Richtung multipliziert. Dadurch erhält man ein Signal, bei dem die auftretenden Frequenzen konstant sind. Aus diesem kann man mit der diskreten Fourier-Transformation die auftretenden Frequenzen analysieren. Die stärkste auftretende Frequenz ist die Offsetfrequenz. [17] [16]

### Codierung

Der für die Übertragung wichtigste Punkt ist die Codierung. Die Codierung hilft, Fehler erkennbar zu machen und zu korrigieren. Dadurch kann die Reichweite gesteigert werden. [17]

Semtech gibt nur wenig Details über den Codierungsprozess bekannt. Allerdings ist es Forschenden durch Analyse der Signale gelungen, die für die Codierung verwendeten Schritte zu rekonstruieren. Um die Fehler zu minimieren und auszugleichen werden mit LoRa übermittelte Daten vor der Übertragung in 4 Schritten codiert [17] [16]

Ein Bild, das Text, Reihe, Schrift, Diagramm enthält.

Automatisch generierte Beschreibung

Abbildung 25: Encoding/Decoding von LoRa [16]

#### Whitening

Bei Modulation sollten alle möglichen Symbole vertreten sein. Dadurch sind die übertragenen Symbole besser vom Grundrauschen unterscheidbar. [17]

Um möglichst viele unterschiedliche Symbole zu versenden, werden diese mit einer Pseudozufallssequenz logisch verknüpft. Die Zufallssequenz kommt aus einem Linear Feedback Shift Register, kurz LFRS. Um die Sequenz zu erzeugen, werden die Daten durch das Register geschoben. Die Ausgangssequenz des Registers wird wieder mit den Eingängen von den einzelnen Speicherzellen und den Daten verknüpft. Dadurch entsteht am Ausgang eine lineare Funktion, die abhängig von der Eingangskombination ist und im Empfänger rekonstruiert werden kann. [17] [18]

#### Hamming Code

Bei der Funkübertragung können durch Rauschen Fehler auftreten. Um die Fehler zu erkennen, arbeitet LoRa mit dem Hamming Code. Dafür müssen zusätzliche Bits eingebaut werden.

Diese Bits ergänzen eine Reihe von Datenbits, sodass diese eine gerade Anzahl an logischen-1 Bits beinhalten. Da bekannt ist, welche Bits zusammen eine gerade Anzahl ergeben müssen kann man ermitteln, ob ein Fehler entstanden ist. Wird mehr als eines dieser Bits übertragen, werden die Daten in Muster, die sich stellenweise überschneiden, aufgeteilt. Entsteht ein Fehler kann mit den einzelnen Bits, die gerade sein müssten, der Schnittpunkt ermittelt und der Fehler korrigiert werden.

Wie viele dieser zusätzlichen Bits pro Bit von Nutzdaten übertragen werden ist in der Coding Rate festgelegt. Die Coding Rate wird als Verhältnis von 4 Datenbit zu 5-8 übertragenen Bit angegeben. Eine Coding Rate von 4/5 schützt die Daten nicht und kann nur einen Fehler pro Bitgruppe erkennen. Ab 4/7 Können mehrere Fehler erkannt und einzelne Fehler behoben werden. [17]

#### Interleaving

Während der Übertragung kann das Signal durch Rauschen beschädigt werden. Wird ein Symbol bei der Übertragung geschädigt, ist eine Reihe an Bits falsch. Fehlerkorrekturcodes wie der Hamming-Code sind darauf ausgelegt, zufällige Fehler wieder zu reparieren. Das funktioniert nur, wenn einzelne Bits falsch sind. Ist eine Reihe an Bits falsch, können die Daten nicht korrigiert werden. Damit die Fehler nicht auf zusammenhängenden Bits auftreten, werden diese vermischt. Dafür wird diagonal Interleaving verwendet. Dazu nimmt man, wie in Abbildung 26 erkennbar, aus jedem zu übertragendem Symbol ein Bit und wechselt nach jedem Symbol ein Bit weiter in Richtung des niederwertigsten Bits. [17]

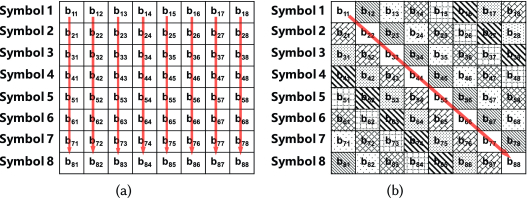


Abbildung 26: Interleaving (vgl. [17])

#### Gray Code

Ein Gray-Code weist numerischen Symbolen eine Bitfolge zu. Das Besondere am Gray-Code ist, dass sich zwischen zwei aufeinander folgenden Symbolen nur in einem Bit unterscheiden. Dadurch lassen sich falsch übertragene Daten leicht erkennen.

In den meisten LoRa-Modems sind ungenaue RC-Oszillatoren verbaut. Beim Abtasten kann es passieren, dass Teile vom falschen Symbol abgetastet werden. Mit der Gray-Codierung wird der dabei entstehende Fehler besser erkennbar. Dadurch ist es wahrscheinlich, dass der Fehler korrigiert werden kann. [17]

#### Paketierung

Um den Empfänger zu synchronisieren und die Daten sicher zu übertragen, versendet LoRa versendet die Daten in einer Paketstruktur. Der Aufbau eines Paketes ist in Abbildung 27 zu sehen.

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

Automatisch generierte Beschreibung

Abbildung 27: LoRa Paket [19]

Mithilfe des Preambles wird der Empfänger synchronisiert. Der Preamble ist standardmäßig 12 Symbole lang und kann nach Bedarf verkürzt oder verlängert werden. Der Header enthält Informationen über die Länge des Payloads, Informationen für die Korrektur von Fehlern und eine CRC-Prüfsumme, mit der die Informationen aus dem auf ihre Richtigkeit überprüft werden können. Der Header selbst wird immer mit der höchsten Coding Rate übertragen, damit die Informationen möglichst sicher ankommen.

Im Payload befinden sich die Daten, die Übertragen werden sollten. Abgeschlossen wird das Paket mit der Prüfsumme des Payloads. Die Coding Rate für diesen Teil kann vom Nutzer je nach Bedarf für hohe Reichweite oder hohe Übertragungsraten festgelegt werden. [19]

### Antennen

Für die Funkübertragung mit LoRa benötigt man Antennen. LoRa nutzt in Europa Frequenzen um 868MHz. Die Antennen müssen für diesen Frequenzbereich ausgelegt sein.

Bei der Übertragung mit hohen Frequenzen treten auf Leitungen Reflexionen auf, was die Übertragungsqualität verschlechtert. Um dem entgegenzuwirken, muss die Leitung mit dem Leitungswiderstand abgeschlossen sein. Dadurch verhält sich die Leitung, wie eine unendlich lange Leitung mit einem Eingangswiderstand gleich dem Wellenwiderstand. [20]

Sowohl das TTGO T-Beam als auch das RFM95W haben einen Wellenwiderstand von 50Ω. Deshalb ist es wichtig, dass die verwendeten Antennen einen Eingangswiderstand von 50Ω haben.

Antennen können nach ihrem Strahlverhalten eingeteilt werden. Die Stärke wird mit dem Isotropen Strahler verglichen. Der Isotrope Strahler ist eine nicht realisierbare Antenne, welche die Energie in alle Richtungen gleichmäßig abstrahlt. Die Verstärkung wird in alle Richtungen gemessen, verglichen und in einem Richtdiagramm aufgetragen. [21]

#### Sendeeinheit

Für die Sendeeinheit wurde die molex ISM 868/915MHz Dipol Antenne gewählt. Die Antenne hat ein gutes Rundstrahlverhalten. Durch die flexible Bauform lässt sie sich leicht in das Gehäuse integrieren. Durch den Wellenwiderstand von 50Ω kann die Antenne Problemlos verwendet werden. [22]

Der Dipol-Teil ist in einem Klebestreifen verbaut und wird an die Gehäusewand geklebt. Über den UFL-Anschluss wird die Antenne mit dem LoRa Breakout verbunden.

Ein Bild, das Diagramm, Text, Kreis, Reihe enthält.

Automatisch generierte Beschreibung

Abbildung 28: Richtdiagramm molex Dipol [22]

#### Basisstation

Als Antenne für die Basisstation wurde eine Dipolantenne des Herstellers LPRS verwendet. Die Antenne hat ein gutes Rundstrahlverhalte und empfängt Daten von allen Richtungen ungefähr gleich stark, wie aus dem Richtdiagramm in Abbildung 29 hervorgeht. Die Antenne wurde speziell für den Outdoor-Einsatz entwickelt und wird mit einer Wandhalterung an der Außenseite der Alm befestigt.

Ein Bild, das Kreis, Diagramm, Reihe, Symmetrie enthält.

Automatisch generierte Beschreibung

Abbildung 29: Richtdiagramm Antenne Basisstation [23]

### SX1276

Auf dem Adafruit RFM95W Breakout-Board auf der Sendeeinheit und auf dem TTGO T-Beam ist ein SX1276 LoRa Transceiver verbaut. Bei der Modulation mit LoRa können Signale mit einer Stärke bis zu -148dBm empfangen und ausgewertet werden. Die Kommunikation erfolgt mit einem SPI-Interface. [19]

#### Kommunikation

Mit dem SPI-Interface können einzelne Register beschrieben und gelesen werden. Für die Übertragung wird immer ein Byte mit dem ersten Bit für write-not-read und 7 Adressbits, gefolgt von einem Byte Daten übertragen. Schreiben ist nur zulässig, wenn das Modul im Stand-by-Modus ist. Um das Modul in den Stand-by-Modus zu versetzten muss ein Byte mit dem Wert 0 in das Operation Mode Register geschrieben werden. Die zu sendenden und empfangenen Daten werden in einem 256 Byte großem FiFo-Speicher abgelegt. Für senden und empfangen gibt es in diesem Speicher zwei frei bewegliche Pointer. [19]

#### Übertragung

Nachdem die Frequenz, die Bandbreite, die Coding Rate und der Spreadingfactor in die jeweiligen Register geschrieben wurden kann mit der Datenübertragung begonnen werden. Ist das Modul im Stand-By-Modus, können die Dateien in den FiFo geschrieben werden. Der FiFo-Pointer wird dabei automatisch erhöht. Um die Übertragung zu starten, muss im Modus-Register der Transmit Mode und in den Long Range Modus aktiviert werden. Nach dem Versenden der Daten wird ein Interrupt ausgelöst und das Modul wechselt zurück in den Stand-By Modus. [19]

## Basisstation

Um die Daten auf dem Server zu speichern, müssen diese wieder empfangen und anschließend mit einer http Anfrage weitergeleitet werden. Dafür benötigt die Basisstation eine stabile Anbindung an das Internet und ein LoRa-Receiver Modul.

Aufgestellt wird das Modul auf der Vierkaseralm. Die Verbindung zum Internet kann dort mit WLAN hergestellt werden. Versorgt wird die Basisstation mit einem Micro-USB Kabel, angesteckt an einem Netzteil. Die Antenne wird auf der Außenseite der Alm mit einer Wandhalterung befestigt.

### TTGO T-Beam

Für die Realisierung der Basisstation wurde die TTGO T-Beam Entwicklungsplatine gewählt. Als Controller ist ESP32 verbaut, wodurch kein extra Modul für WiFi benötigt wird. Die für WLAN-Antennen sind bereits auf der Platine verbaut. Versorgt und programmiert kann das Board mit einem Micro-USB Kabel werden. Für die Kommunikation mit LoRa bietet das Board einen SX1276 LoRa Transceiver. Für die LoRa-Antenne ist ein SMA-Anschluss verbaut. [24]

### PlatformIO

PlatformIO hat es sich zum Ziel gesetzt, die Entwicklung für Microcontroller möglichst plattformunabhängig und komfortabel zu gestalten. Die Basis ist ein Core, welcher Abhängigkeiten verwaltet und das Projekt kompiliert und auf den Microcontroller lädt. Um die Funktionen des Cores in einer Entwicklungsumgebung zu nutzen, werden die Funktionen mit einem Plugin in diese integriert. Dieses Plugin ist für viele gängigen Entwicklungsumgebungen verfügbar. Aufgrund der weiten Verbreitung und des einfachen Installationsprozesses wurde Visual Studio Code als Editor gewählt.

PlatformIO baut auf existierende Frameworks, wie dem Arduino Framework oder dem ESP-IDF auf, ist aber mit Boards und Microcontrollern von mehreren Herstellern kompatibel. Dadurch erhält man die Möglichkeit, viele Bibliotheken in sein Projekt zu integrieren, ohne von der Hardware der Hersteller und deren Tools Abhängig zu sein. [25]

Für die Kommunikation mit dem LoRa-Chip wurde die LoRa Bibliothek von Sandeepmistry verwendet. Diese ist mit den Chips der Semtech SX127X Familie und mit einigen Arduino Boards kompatibel und erleichtert die Kommunikation über LoRa. Zusätzlich diente der Code dieser als Hilfestellung bei der Programmierung der Sendeeinheit. [26]

Die empfangenen Daten werden mithilfe der ArduinoJSON Bibliothek in ein JSON-Objekt umgewandelt und mit dem für den ESP32 mitgelieferten HTTPClient an den Server übermittelt.

### Programmablauf

Zu Beginn des Programms die Pins für die Module definiert und alle Bibliotheken initialisiert. Das Hauptprogramm besteht aus einer Schleife, die wartet, bis Daten mit LoRa empfangen wurden. Werden Daten empfangen, werden diese zusammen mit der Empfangsstärke, kurz RSSI, auf der Konsole ausgegeben.

Mit der Funktion scanf werden die Daten aus dem empfangenen String wieder ausgelesen und ein struct geschrieben. Die sscanf-Funktion aus der Bibliothek des Compilers unterstützt keine Floating Point Zahlen. Deshalb wurden der Längen- und Breitengrad wurde als 32-Bit Integer übertragen Um die ursprünglichen Werte zu erhalten, muss man die beiden Variablen durch 100.000 dividieren.

sscanf(rec.c\_str(), "%d %d %d %d %d %d %d %" PRId32 " %" PRId32,

           &position->id,

           &position->hour,

           &position->minutes,

           &position->seconds,

           &position->day,

           &position->month,

           &position->year,

           lat,

           lng

           )

position->lat = (float)lat / 100000;

position->lng = (float)lng / 100000;

Das Datum wird aus den einzelnen Elementen wieder zusammengesetzt und die Daten werden in ein JsonDocument mit dem Namen Data geschrieben.

data["latitude"] = position->lat / 100000;

data["longitude"] = position->lng / 100000;

String date = "20" + String(position->year)

          + "/" + String(position->month)

          + "/" + String(position->day)

          + " " + String(position->hour)

          + ":" + String(position->minutes)

          + ":" + String(position->seconds);

data["date"] = date;

Das Objekt wird auf der Konsole ausgegeben, in einen String konvertiert und mit einer POST-Anfrage an den Server übermittelt.

String POST\_DATA;

serializeJson(data, POST\_DATA);

if (http.begin(url)) {

   http.addHeader("Content-Type", "application/json");

   int httpResponseCode = http.POST(POST\_DATA);

if (httpResponseCode > 0) {

    Serial.print("HTTP Response code: ");

       Serial.println(httpResponseCode);

       String response = http.getString();

       Serial.println(response);

   } else {

    Serial.print("Error in HTTP request. Fehlercode: ");

       Serial.println(httpResponseCode);

   }

  http.end();

}

In Abbildung 30 sind die einzelnen Schritte und Ausgaben aus dem Code auf dem Seriellen Monitor nach einem erfolgreichen Durchlauf erkennbar.

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

Automatisch generierte Beschreibung

Abbildung 30: Serieller Monitor Basisstation

## Server

Die Daten sollten von Überall über das Internet Abrufbar sein. Um das zu ermöglichen, benötigt es eine Serversoftware, welche die Daten in eine Datenbank speichert und nach Authentifizierung wieder zur Verfügung stellt.

Um den Server möglichst einfach zu entwickeln wurde das Laravel-Framework verwendet. Dadurch konnten Features wie der Login einfach eingebaut werden. Da eine Gemeinschaft an dem Framework arbeitet ist der Code oft fehlerfreier und sicherer als eine komplette Eigenentwicklung.

### Laravel

Laravel ist ein von einer Community entwickeltes und von Firmen unterstütztes open-Source PHP-Framework. Es erleichtert die Entwicklung von Webanwendungen und APIs. Für das Framework gibt es viele, fertige Bibliotheken um verschiedenste Features wie ein Bezahlsystem oder einen Login in seine Anwendung einzubauen. [27]

Laravel arbeitet mit dem Model-View-Control Muster. Dieses Muster unterteilt die Software in einen Modell Teil, in welchem ein Modell von den Daten definiert wird, einen View Teil, welcher sich um die Visualisierung und Interaktion mit dem Nutzer kümmert und einen Controller, welcher die Modelle auf Anfragen erstellt, ändert oder ausliest.

Modelle sind eine PHP-Klasse, die gespeichert, abgerufen und geändert werden kann. Durch diese wird in dem Framework mit Datenbanken kommuniziert. Der Controller Teil wird verwendet, um diese zu erstellen und zu verändern. Der View-Teil ist bei diesem Projekt die Smartphone-Applikation. [28]

#### Eloquent ORM

Ein Object-realtional mapper vereinfacht die Kommunikation mit Datenbanken. Für jedes Modell wird eine Tabelle in einer Relationalen Datenbank erstellt. Alle Modelle erben von der Model Klasse aus dem ORM. Dadurch hat jedes Modell bereits Funktionen für die Speicherung, Änderung und Abfrage aus einer Datenbank.

Mit Funktionen wie findAll und find können die Modelle aus der Datenbank zurück in eine Instanz der Klasse geladen werden.

Um die Verbindung zwischen Modellen zu realisieren können die Funktionen has und hasMany und belongsTo genutzt werden. Dadurch werden die Daten in der Datenbank miteinander verknüpft und bei Änderung automatisch in allen Tabellen geändert. [29]

#### Sanctum Auth

Die Sanctum Bibliothek von Laravel vereinfacht das Erstellen von Projekten mit Nutzern und Nutzerverwaltung. Funktionen, um Nutzer zu erstellen, sich zu registrieren und zu authentifizieren werden bereitgestellt.

Sanctum arbeitet mit Tokens. Diese werden nach der Authentifizierung mittels E-Mail und Passwort an die App des Nutzers übermittelt. Diese schickt den Token bei jeder Anfrage an den Server im Header der Anfrage mit. Dadurch kann festgestellt werden, wer der Nutzer ist und worauf er Zugriff hat. [30]

#### Modelle und Datenbank

Aufgrund der Verwendung des ORMs ist die Wahl der Datenbank eingeschränkt. Der Laravel ORM ist mit dem Microsoft SQL Server, MariaDB, MySQL, PostgreSQL und SQLite kompatibel. Ein Wechsel zwischen den Datenbanken ist Aufgrund dieser Abstrahierung jederzeit ohne Probleme möglich.

Für das Projekt wurde MariaDB als Datenbank verwendet. MariaDB ist eine Quelloffene Datenbank, die auf Basis des MySQL SQL-Servers entwickelt wurde. Die Community, die den Datenbankserver entwickelt legt großen Wert auf Effizienz und durch die kostenlose Nutzung gibt es eine Vielzahl an Tutorials.

Wird aus den Modellen eine Tabelle erstellt nennt man das eine migration. Die an der Datenbank durchgeführten migrations werden zusammen mit der Anzahl wie oft die Tabelle verändert wurde in der migrations Tabelle gespeichert. Schlägt eine migration Fehl wird das in der failed\_jobs Tabelle festgehalten. [29]

Das Modell für den User wurde von Sanctum übernommen. Bei der Registrierung muss der Nutzer seinen Namen, ein Passwort und eine E-Mail-Adresse angeben. Für den Nutzer wird ein Account angelegt, mit welchem er sich anmelden, Tracker erstellen und Trackerdaten abrufen kann. In der personal\_access\_token Datenbank werden die Tokens, mit denen der Nutzer seine Identität bestätigen kann, gespeichert.

Der Name der Kuh soll in der App angezeigt werden. Dafür müssen den Trackern ein Namensfeld zugewiesen werden. Erstellt der Nutzer einen Tracker, muss ein dem Tracker einen Namen geben. Ist das Erstellen erfolgreich, bekommt er eine ID, die er im Tracker speichern muss. Der Nutzer, der den Tracker erstellt hat, wird aus dem Token in der Anfrage ausgewertet und in den Tracker gespeichert. Dadurch hat nur er Zugriff auf den Tracker.

Die Positionen, die vom Tracker an den Server geschickt werden, kommen in das Position Modell. Dieses bietet ein Feld für Längengrad, Breitengrad, den Zeitpunkt, an dem der Standort bestimmt wurde, sowie die ID des Trackers. Dadurch lässt sich bestimmen, zu welcher Kuh der Standort gehört.

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte Beschreibung

Abbildung 31: Datenbankstruktur

Aus den beschriebenen Modellen entstand die Datenbankstruktur in Abbildung 32. Die Modelle wurden mit dem mitgelieferten Tool artisan erstellt. Ein Modell kann mit dem Befehl *php artisan make:model Tracker -m* erstellt werden.

Der Befehl erstellt das benötigte File für das Modell und das File für die Datenbankmigration. Das File für die Datenbankmigration findet man im Ordner *database/migrations/2024\_01\_07\_110648\_create\_trackers\_table.php.* Dort müssen in der Create Funktion die benötigten Variablen angelegt werden. Als Beispiel wurde

Schema::create('trackers', function (Blueprint $table) {

$table->id();

      $table->unsignedBigInteger('user\_id');

      $table->string('name');

      $table->foreign('user\_id')->references('id')->on('users');

      $table->timestamps();

});

Das Modell findet man unter App/Models/Tracker.php. In diesem veränderbaren Daten und Beziehungen zu anderen Modellen festgelegt werden.

class Tracker extends Model

{

    use HasFactory;

    protected $fillable = ['user\_id', 'name'];

    public function user()

    {

        return $this->belongsTo(User::class);

    }

    public function positions()

    {

        return $this->hasMany(Position::class);

    }

}

#### Controller

Um die Tracker zu verwalten, wurde der TrackerController erstellt. Dieser kann Tracker erstellen und auflisten und Positionen zu einem Tracker hinzufügen und wieder abrufen. Als Beispiel für die Syntax und die Funktionsweise wird die Funktion zum Abrufen der Letzten Position eines Trackers beschrieben.

public function getPosition(Request $request, $trackerId)

    {

        // Find the tracker

        $tracker = Tracker::findOrFail($trackerId);

// check if user is creator of the tracker

        if ($tracker->user\_id == $request->user()->id)

        {

// return the last position

            $position = $tracker->positions->last;

            return response()->json(['position' => $position]);

        }

// if not, return no authorized

        return response('invalid Tracker id', '401');

    }

Die Funktion bekommt die Anfragedetails und die ID des Trackers als Parameter. Mit der findOrFail Funktion des ORMs wird nach dem Tracker gesucht. Existiert der Tracker wird überprüft, ob der Nutzer, der die Anfrage gemacht hat, den Tracker erstellt hat und auf die Daten zugreifen darf. Ist das der Fall, werden die Positionen des Trackers ausgelesen und der letzte Standort im JSON-Format zurückgegeben. Sonst wird der http-Fehlercode 401, welcher den Nutzer darauf hinweist, dass er keine Berechtigungen auf diese Ressource hat, zurückgegeben.

#### Routing

Routen legen fest, über welche Adresse über den Server auf eine Funktion zugegriffen werden kann. Um die Funktionen aus den Controllern nutzen zu können muss diese mit einer Route verknüpft werden. Die Anfragen können im Router mit einer sogenannten Middleware gefiltert werden. Sanctum bietet eine Middleware, die überprüft, ob der Nutzer angemeldet ist. Ist der Nutzer nicht angemeldet bekommt er den http-Statuscode 401 zurück. Ist er angemeldet und sendet einen gültigen Token lässt die Middleware die Kommunikation zu und gibt der aufgerufenen Funktion den Nutzer, der die Anfrage erstellt hat, mit. [31]

Die Route, an die die Basisstation die Kühe sendet, ist frei zugänglich und die empfangenen Daten werden nicht auf ihre Authentizität geprüft. Da Daten an die Route gesendet werden ist es eine POST-Anfrage. Die trackerID muss beim Aufrufen der Basisstation durch die ID des Trackers der Kuh ersetzt werden.

Route::post('/trackers/{trackerId}/positions', [TrackerController::class, 'addPosition']);

Die Routen, an denen ein Nutzer Tracker Daten von den Trackern aufruft, werden von der Sanctum Middleware kontrolliert. An dieser Stelle bekommt der Nutzer Daten, daher ist es eine GET-Anfrage.

Route::middleware('auth:sanctum')->get('/trackers/{trackerId}/position', [TrackerController::class, 'getPosition']);

## Smartphone Applikation

Nachdem die GPS-Daten in der Datenbank auf dem Server gespeichert wurden, ist eine visuelle Darstellung der Daten als letzter Schritt erforderlich. Wir haben uns dazu entschieden, sowohl eine Handy-Applikation als auch eine Website zu erstellen, die jederzeit mit einer stabilen Internetverbindung abgerufen werden können.

### Anforderungen

Die Hauptanforderung der Applikation ist, sehr simpel und übersichtlich gestaltet zu sein. Die GPS-Daten werden alle 5 Minuten aktualisiert und auf einer Karte dargestellt. Zudem soll die Applikation so programmiert werden, dass der Code leicht verständlich ist und Erweiterungen leicht möglich sind.

### Flutter

Als Framework für die C-Track haben wir uns für Flutter entschieden. Flutter ist ein Entwicklungs-Kit, welches von Google im Jahr 2017 veröffentlicht wurde. Flutter benutzt die Programmiersprache Dart und es können sogenannte Cross-Plattform-Applikationen damit erstellt werden. Dies bedeutet ein Flutter Programm kann ohne aufwendige Optimierung auf mehreren Plattformen wie Android, IOS, etc, laufen. Gegenüber anderen Frameworks wie React Native, hat Flutter den Vorteil, dass sie eine leicht zu verstehende Plattform bietet, eine sehr hilfsbereite Community dahintersteht und viele Tutorials, welche den Einstieg enorm erleichtern. Weiters bietet Flutter eine angenehme Testumgebung, um die Applikationen zu überprüfen. Wie bei allen anderen Frameworks auch, hat Flutter das Problem der Unüberschaubarkeit. Wird das Projekt größer und durch die Einbindung von mehreren Widgets kann man leicht die Übersicht verlieren. Zudem ist Flutter im professionellen Umfeld eine wenig verbreitetes Framework und wird hauptsächlich von „Hobby Programmierer“ verwendet.

#### Widgets

In Flutter sind Widgets die grundlegenden Bausteine für den Aufbau von Benutzeroberflächen. Sie repräsentieren alles in der Applikation, sei es ein Struktur Element wie eine Zeile oder ein Spaltenlayout, oder ein visuelles Element wie Schaltflächen, Text oder Bilder. Widgets können Zustände halten, interaktiv sein und sich, basierend auf Eingaben des Benutzers, Änderung von Daten oder anderen Faktoren, ändern. Zudem kann ein Widget wiederum aus mehreren anderen Widgets bestehen. Grundsätzlich gibt es in Flutter zwei Haupt Widget arten welche die Grundlage für das Erstellen einer Benutzeroberfläche bilden.

* Das Stateless Widget ist unveränderlich. Das heißt, es wird einmal eingestellt und kann nicht verändert werden. Benutzt wird dieses für statische Inhalte und nicht interaktive Elemente, wie Texte oder Icons.
* Das Statefull Widget kann im Gegensatz ihren Zustand ändern und somit dynamische Inhalte oder interaktive Elemente, wie Slider oder Buttons darstellen.

#### Packages [32]

Packages in Flutter sind externe Bibliotheken welche Tools und Funktionen bereitstellen, um die Entwicklung von Apps zu erleichtern und die Entwicklungszeit wesentlich verkürzen. Diese Packages können von Google als auch von der Community erstellt werden.

Grundsätzlich werden alle Packages in der zentralen Quelle für Packages, *pub.dev*, verwaltet. Um ein Package überhaupt verwenden zu können muss dieses zuerst in der *pubspec.yaml* Datei unter den *dependencies* hinzugefügt werden.

Ein Bild, das Text, Schrift, Screenshot enthält.

Automatisch generierte BeschreibungAls Beispiel sind hier die verwendeten Packages in unserem Projekt. Diese werden immer mit der im Projekt verwendeten Versionsnummer angezeigt

Abbildung 32: dependencies in pubspec.yaml

Danach wird der Befehl *flutter pub get* ausgeführt, um die Packages herunterzuladen. Anschließend können diese im Projekt mithilfe der Import Funktion importiert werden, um die benötigten Funktionen darin zu verwenden.

Sind Packages veraltet oder haben eine neuere Version können, die einfach mit dem *flutter pub upgrade* Befehl aktualisiert werden.

#### App Icon

Am Beginn jeder in Flutter programmierten Applikation hat diese standartmäßig das Flutter Logo als App Icon. Um dies zu ändern, muss zuerst eine neue Datei mit dem Namen *flutter\_launcher\_icons.yamal* erstellt werden

Anschließend wir dieser Code hineinkopiert. Im Projekt Verzeichnis wird nun ein Ordner Namens *assets* angelegt in dem alle für die Applikation benötigten Dateien oder Bilder eingefügt werden. Als letztes wird nun das gewünschte Bild als *icon.png* gespeichert und in den *dependencies* in der *pubspec.yaml* Datei die neuste Version von *flutter\_launcher\_icons* hinzugefügt.

Abbildung 33: Code für Launcher Icon

#### App Bar

Zuerst wird die Leiste (AppBar) am oberen Rand der Applikation angepasst und überarbeitet.

Mit dieser Code Zeile lässt sich das Debug Banner, welches bei jeder Flutter Applikation im rechten oberen Eck der AppBar befindet, ausschalten.



Abbildung 34: Code: Debug Banner deaktivieren

Ein Bild, das Text, Screenshot, Schrift, Zahl enthält.

Automatisch generierte BeschreibungAls nächstes wurde ein Text Widget erstellt, welches den Namen des Projekts anzeigt.

Wichtig ist die Funktion *textDirection.* Diese Funktion verhindert das beim Kompilieren des Programms die Fehlermeldung einer NullSafetyViolation aufkommt.

Abbildung 35:Code: Text in App Bar

Ein Bild, das Text, Karte, Atlas enthält.

Automatisch generierte BeschreibungDanach wurden noch die Ecken abgerundet, die Farbe geändert und die Schrift zentriert, um die AppBar anschaulicher zu gestalten. Das Endprodukt sieht danach so aus:

Abbildung 36: Fertige AppBar

#### Drop Down Menu

Als weiterer Schritt wurde ein Drop Down Menu in der AppBar erzeugt. In dieser befindet sich die Lizenz für die Benutzung von OpenStreetMap und eine kurze Übersicht über das Projekt.

Wird auf das Drop Down Menu Icon im linken Eck gedrückt, öffnet sich ein sogenanntes Drawer Widget. In diesem Widget befindet sich Knöpfe für die jeweiligen Optionen, sowie unser Projekt Plakat.

Ein Bild, das Text, Schrift, Screenshot, Reihe enthält.

Automatisch generierte BeschreibungDas Projekt Plakat wurde im *DrawerHeader* als *image* eingefügt,um die grundsätzliche Ästhetik und Anschaulichkeit der App zu verbessern.

Abbildung 37: Code: Drop Down Menu Header

Hierfür wurde das Plakat als png-Datei im assets Ordner gespeichert und mit folgender Funktion im Drop Down Menu angezeigt.

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

Automatisch generierte Beschreibung

Abbildung 38: Code: Button einfügen

Um einen Button einzufügen, wird ein ListTile Widget eingefügt in welchen das Icon, welches dieser Button haben soll, ausgewählt wird. Danach wird dieser Button benennt und im Anschluss wird gesagt was er beim Drücken machen soll. In unserem Fall wechselt er von der *HomePage* in die sogenannte *AboutUsPaige*. Möchte man weitere Buttons einfügen wird wieder genau gleich ein ListTile Widget erstellt und der Name, Icon und Funktion eingefügt.

Ein Bild, das Text, Screenshot, Design enthält.

Automatisch generierte Beschreibung

Am Ende sieht das ausgefahrene Drop Down Menu wie in Abb.: 40 aus. Dieses kann je nach Belieben leicht um weitere Funktionen erweitert und verbessert werden.

Abbildung 39: Drop Down Menu

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

Automatisch generierte BeschreibungKlickt man nun auf einen der zwei Knöpfe wird man auf eine neue Seite weitergeleitet. Als einfachste Lösung den gewünschten Text darzustellen, entschieden wir uns den Text in ein normales txt-File zu kopieren und dieses in den assets Ordner abzulegen und mit folgender Funktion aufzurufen.

Abbildung 40: Code: txt-File einlesen

Als erstes wird ein String mit dem Namen *data* initialisiert. Als nächstes wird in der *fetchFileData* Funktion der Inhalt des txt-Files herausgelesen und im Anschluss in den String hineingeschrieben.

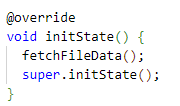
Nun wird diese Funktion im *initState* aufgerufen, sodass der Text angezeigt wird, sobald man von der HomePage in die *AboutUsPage* wechselt.

Abbildung 41: Code: initState

Ein Bild, das Text, Brief, Schrift, Screenshot enthält.

Automatisch generierte Beschreibung

Am Ende müssen nur noch Anpassungen über die Größe des Textes getroffen werden, um alles schön auf einer Seite anzeigen zu können. Möchte man nun wieder zurück zur *HomePage* muss man einfach nur den Pfeil im linken oberen Eck drücken.

Abbildung 42: About Us Page

#### Flutter Map

Am wichtigsten ist die Landkarte selbst. Wir haben uns für Open Street Map entschieden da diese frei zu benutzen ist und die Lizenz dafür lediglich in der App einsehbar sein muss. Die Lizenz wurde ist unserer Applikation im Drop Down Menu unter dem Punkt *OSM License* abrufbar. Zudem ist die Landkarte sehr aktuell und teils genauer als Google Maps. Der Nachteil von Open Street Map ist, dass es nur mit einer stabilen Internetverbindung funktioniert.

Ein Bild, das Text, Schrift, Reihe, Screenshot enthält.

Automatisch generierte BeschreibungUm Open Street Map in die App einbinden zu können muss zuerst Flutter Maps importiert werden [33], sowie das *latlong2* Package [34].

Abbildung 43: Code: Flutter Map

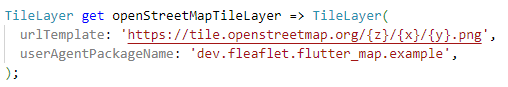
Als erstes wird nun ein Widget für Flutter Map erstellt in dem die grundsätzlichen Einstellungen der Karte vorgenommen werden. Diese sind etwa der Startpunkt, der angezeigt werden soll, wenn die Karte geöffnet wird oder auch den Startzoom.

Abbildung 44: Code: Tile Layer

Um die Karte von Open Street Map in der Applikation anzeigen zu lassen muss zuerst ein sogenanntes *TileLayer* Widget erstellt werden. Danach wird die URL von OpenStreetMap eingefügt, um die Karte in die Appliaktion zu implementieren.

Um die Position des Nutztieres auf der Karte darstellen zu können muss nun ein *MarkerLayer* erstellt werden. Hier werden die Höhe und die Weite , sowie die Position des Markers eingestellt. Die Position bekommt der MarkerLayer über die Verbindung mit dem Server. Als letztes wurde noch eine SVG-Datei einer Kuh aus dem assets Ordner als Icon des Markers verwendet.

Abbildung 45: Code: Marker Layer

#### Server Verbindung

Ein Bild, das Text, Schrift, Screenshot, Reihe enthält.

Automatisch generierte BeschreibungUm die Standortdaten der Kuh abzubilden, muss sich die Applikation mit dem Server verbinden. Für diese Funktion wird das http Package benötigt [35].

Abbildung 46: Code: Serververbindung

Um die Verbindung mit dem Server zu erstellen, wird dessen URL hinterleget sowie ein Token, welcher als Account Identifizierung dient.

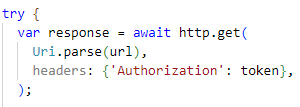


Abbildung 47. Code: Serververbindung

Über eine try/catch Funktion wird nun die Verbindung zum Server angefragt und überprüft.

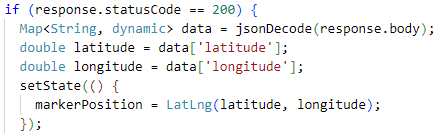


Abbildung 48: Code: Marker Position

Ein Bild, das Text, Screenshot, Schrift, Reihe enthält.

Automatisch generierte BeschreibungIst die Verbindung erfolgreich empfängt die Applikation eine json-Datei die Zeit, Längengrad und Breitengrad enthält. Da diese Datei jedoch unbrauchbar für das Einfügen der Marker Position ist, wird diese json-Datei in einen String umgewandelt und daraus werden die Werte für Längen – und Breitengrad extrahiert und in die *markerPosition* variable hineingeschrieben.

Abbildung 49: Code: initState

Die fetchMarkerPosition Funktion wird nun im initState aufgerufen, um die Position der Kuh beim Start der App sofort anzuzeigen. Um jetzt alle 5 Minuten die Position aktualisieren zu können wurde ein Timer hinzugefügt der in 5 Minuten abständen die Funktion erneut ausführt.

#### Fertige Handy App

Ein Bild, das Text, Karte, Atlas enthält.

Automatisch generierte Beschreibung

Die Fertige Handy App beinhaltet nun eine Landkarte von Open Street Map, welche beim Start der Applikation sich mit dem Server verbindet, um die Standortdaten der Kuh zu erhalten. Danach wird mit einem Kuh Symbol der Standort graphisch angezeigt und alle 5 Minuten aktualisiert.

Im Drop Down Menu befindet sich die benötigte Lizenz von Open Street Map sowie eine kurze Erklärung zu unserem Projekt

In diesem Bild ist der Startbildschirm unserer Smartphone Applikation zu sehen welche einen Testwert welcher vom Server gesendet wurde auf einer Landkarte darstellt.

Abbildung 50: Startbildschirm der Smartphone Applikation

## Verbesserungen

### Platine

Der Trickle Power Mode eignet sich perfekt für das Projekt, konnte aber Aufgrund der Komplexität von OSP-Sätzen nicht in das Projekt integriert werden.

Aus Stromspargründen wurde der interne RC-Schwingkreis des ATmega644 verwendet. Die Frequenz kann aber Temperatur- und Spannungsabhängig, wodurch Clock Frequenz, insbesondere bei niedrigen Temperaturen zu ungenau für UART wird. Daher muss entweder ein Quarz verbaut werden, oder die Kommunikation mit dem GPS-Modul auf eine andere Schnittstelle gelegt werden.

Zum Schutz der SD-Karte könnten die Daten in einen Buffer gespeichert werden und gesammelt auf die SD-Karte geschrieben werden. (vgl. 4.1.5.1)

### Server

Der Server prüft die Daten nicht auf Authentizität. Dadurch könnte ein Angreifer eigene Positionsdaten senden und speichern. Um den Entgegenzuwirken könnten die Daten mit Kryptografie signiert werden.

### Smartphone Applikation

In der Applikation wurden die Anmeldung des Servers, die Möglichkeit für mehrere Tracker und die Möglichkeit den Zeitverlauf zu betrachten nicht implementiert.

# Ergebnisse – Abnahme

Der Aktuelle Projektstand beinhaltet eine funktionstüchtige Platine, welche die GPS-Daten empfängt und diese auf der SD-Karte speichert und mittels LoRa diese and den Empfänger übermittelt. Über den Empfänger werden die Daten an eine Rest-API übermittelt, in eine Datenbank geschrieben und für den Nutzer wieder zur Verfügung gestellt. Die Anwendung für das Mobiltelefon verbindet sich zu dieser Rest-API und zeigt die GPS-Daten auf einer Landkarte an.

Ein Bild, das Text, Karte, Atlas enthält.

Automatisch generierte BeschreibungEin Bild, das Elektronik, Schaltung, Text, Elektronisches Bauteil enthält.

Automatisch generierte Beschreibung

Abbildung 51:Startbildschirm der Smartphone Applikation

Abbildung 52: Gelötete Platine V1

In der folgenden Tabelle werden die Ergebnisse der Testfälle, aus dem Lastenheft, veranschaulicht.

|  |  |  |
| --- | --- | --- |
| **Anforderung** | **Testergebnis** | **Freigabe (J/N)?** |
| Empfang der Standortdaten. (F0010) | Die Verbindung zu einem Satelliten wird hergestellt und die Standortdaten werden ermittelt. (T0010) | J |
| Speichern auf die SD-Karte. (F0020) | Die GPS-Daten werden auf der SD-Karte gespeichert. (T0020) | J |
| Daten sollen mittels LoRa übertragen werden. (F0030) | Die Daten werden von der Sendeeinheit zum Empfänger mittels Lora übermittelt. (T0030) | J |
| Daten sollen in Datenbank gespeichert werden (F0040) | Die Daten werden in der Datenbank gespeichert und für die App zugänglich gemacht (T0040) | J |
| Daten sollen über die App abgerufen werden können. (F0050) | Die Daten werden von der App abgerufen und graphisch in einer Landkarte dargestellt. (T0050) | J |

# Literaturverzeichnis

|  |  |
| --- | --- |
| [1] | „Akku vs Batterie,“ reichelt elektronik , 8 August 2016. [Online]. Available: https://www.reichelt.de/magazin/ratgeber/akku-vs-batterie-wann-welcher-stromspender-von-vorteil-ist/. [Zugriff am 2024 03 05]. |
| [2] | „Buck-Boost Converters,“ 15 03 2024. [Online]. Available: https://www.monolithicpower.com/en/power-electronics/dc-dc-converters/buck-boost-converters. |
| [3] | Texas-Instruments, „TPS63001,“ 2023. [Online]. Available: https://www.ti.com/product/TPS63001?keyMatch=TPS63001&tisearch=search-everything&usecase=GPN-ALT. |
| [4] | „Elara-I User Manual,“ 19 März 2024. [Online]. Available: https://www.we-online.com/components/products/datasheet/2613011037000.pdf. |
| [5] | B. Eager, „A tutorial on the FAT file system,“ 17 November 2017. [Online]. Available: http://www.tavi.co.uk/phobos/fat.html. |
| [6] | W. G. Wong, „The Fundamentals Of Flash Memory Storage,“ 20 März 2012. [Online]. Available: https://www.electronicdesign.com/technologies/embedded/digital-ics/memory/article/21789685/the-fundamentals-of-flash-memory-storage. |
| [7] | Kingston, „Was ist Nand?,“ Mai 2021. [Online]. Available: https://www.kingston.com/de/blog/pc-performance/difference-between-slc-mlc-tlc-3d-nand. |
| [8] | „How to Use MMC/SDC,“ 26 Dezember 2019. [Online]. Available: http://elm-chan.org/docs/mmc/mmc\_e.html. |
| [9] | IONOS, „Was ist FAT32,“ IONOS, 29 November 2021. [Online]. Available: https://www.ionos.de/digitalguide/server/knowhow/fat32/. [Zugriff am 1 April 2024]. |
| [10] | „Polycarbonat Gehäuse IP65,“ 25 10 2023. [Online]. Available: https://at.rs-online.com/web/p/universalgehause/2069086. |
| [11] | „Amphenol Luftöffnung,“ 7 März 2024. [Online]. Available: https://at.rs-online.com/web/p/gehausebeluftung/1749423. |
| [12] | Semtech, „LoRa™ Modulation Basics,“ 2 Mai 2015. [Online]. Available: https://semtech.my.salesforce.com/sfc/p/E0000000JelG/a/2R0000001OJk/yDEcfAkD9qEz6oG3PJryoHKas3UMsMDa3TFqz1UQOkM. |
| [13] | „Design Tips For Incorporating GSM Modules into Embedded Systems,“ 17 März 2024. [Online]. Available: https://resources.altium.com/p/design-tips-for-incorporating-gsm-modules-into-an-embedded-system. |
| [14] | Semtech, „LoRa BasicsN Modem Relay: A Low-Cost Battery Powered Network Extender,“ 8 März 2023. [Online]. Available: https://info.semtech.com/hubfs/LoRa-Basics%20Modem%20Relay%20A%20Low-Cost%20Battery%20Powered%20Network%20Extender-Whitepaper-F.pdf. [Zugriff am 16 März 2024]. |
| [15] | Georg-Johann, „Linear-chirp,“ 28 August 2010. [Online]. Available: https://de.wikipedia.org/wiki/Datei:Linear-chirp.svg. |
| [16] | J. Tapparel, „Complete Reverse Engineering ofComplete Reverse Engineering of LoRa PHY,“ [Online]. Available: https://www.epfl.ch/labs/tcl/wp-content/uploads/2020/02/Reverse\_Eng\_Report.pdf. |
| [17] | Z. Xu, S. Tong, P. Xie und J. Wang, „From Demodulation to Decoding: Toward Complete LoRa PHY Understanding and Implementation,“ 31 Jänner 2023. [Online]. Available: https://dl.acm.org/doi/10.1145/3546869. [Zugriff am 13 März 2024]. |
| [18] | M. Maxfield, „EETimes,“ 20 12 2006. [Online]. Available: https://www.eetimes.com/tutorial-linear-feedback-shift-registers-lfsrs-part-1/. [Zugriff am 2024 März 17]. |
| [19] | Semtech, „Datasheet SX1267-8-9,“ 7 Mai 2020. [Online]. Available: https://semtech.my.salesforce.com/sfc/p/E0000000JelG/a/2R0000001Rbr/6EfVZUorrpoKFfvaF\_Fkpgp5kzjiNyiAbqcpqh9qSjE. |
| [20] | Elektronik Kompendium, „Wellenwiderstand,“ Elektronik Kompendium, [Online]. Available: https://www.elektronik-kompendium.de/sites/grd/0301036.htm. [Zugriff am 15 März 2024]. |
| [21] | Clemen, „Antennen,“ [Online]. Available: https://www.hs-augsburg.de/~clemen/lehre/Skript\_Wellen/10Antennen.PDF. [Zugriff am 2024 Februar 28]. |
| [22] | Molex, „ISM 868/915MHz Dipole Flexible Antenna,“ 29 März 2024. [Online]. Available: https://www.molex.com/en-us/products/part-detail/2067640100?display=pdf. |
| [23] | LPRS, „LPRS-ANT-868-DP-N-F,“ 2017. [Online]. Available: https://lprs.co.uk/assets/files/LPRS-ANT-868-DP-N-F.pdf. |
| [24] | LILYGO, „T-Beam Meshtastic,“ [Online]. Available: https://www.lilygo.cc/products/t-beam-v1-1-esp32-lora-module. |
| [25] | PlatformIO, „What is PlatformIO?,“ 6 Februar 2023. [Online]. Available: https://docs.platformio.org/en/latest/what-is-platformio.html. |
| [26] | sandeepmistry, „Arduino LoRa,“ Github, [Online]. Available: https://github.com/sandeepmistry/arduino-LoRa. [Zugriff am 16 März 2024]. |
| [27] | „Laravel,“ 17 März 2024. [Online]. Available: https://laravel.com/. |
| [28] | jaydeepsathwara, „Introduction into Laravel and MVC Framework,“ 20 März 2024. [Online]. Available: https://www.geeksforgeeks.org/introduction-to-laravel-and-mvc-framework/. |
| [29] | Larvel, „Eloquent: Getting Started,“ [Online]. Available: https://laravel.com/docs/10.x/eloquent. |
| [30] | Laravel, „Laravel Sanctum,“ [Online]. Available: https://laravel.com/docs/10.x/sanctum. |
| [31] | Laravel, „Routing,“ [Online]. Available: https://laravel.com/docs/11.x/routing. |
| [32] | „pub.dev,“ Google, 2024. [Online]. Available: https://pub.dev/. [Zugriff am 20 03 2024]. |
| [33] | „flutter\_map,“ 2024. [Online]. Available: https://pub.dev/packages/flutter\_map. [Zugriff am 13 02 2024]. |
| [34] | „latlong2,“ 2024. [Online]. Available: https://pub.dev/packages/latlong2. [Zugriff am 14 02 2024]. |
| [35] | „http,“ 2024. [Online]. Available: https://pub.dev/packages/http. [Zugriff am 10 03 2024]. |
| [36] | „Flutter,“ 2024. [Online]. Available: https://flutter.dev/. [Zugriff am 23 03 2024]. |
| [37] | „Recom,“ 2024. [Online]. Available: https://recom-power.com/de/rec-n-an-introduction-to-buck,-boost,-and-buck!sboost-converters-131.html?0#:~:text=Ein%20Buck%2D%20oder%20auch%20Abw%C3%A4rtswandler,niedrigere%20Ausgangsspannung%20herunterregelt%20(reduziert).. [Zugriff am 20 3 2024]. |
| [38] | „Abwärtswandler,“ [Online]. Available: http://schmidt-walter-schaltnetzteile.de/smps/abw\_hilfe.html. [Zugriff am 21 03 2024]. |
| [39] | „Aufwärtswandler,“ [Online]. Available: http://schmidt-walter-schaltnetzteile.de/smps/aww\_hilfe.html. [Zugriff am 21 03 2024]. |
| [40] | „ATmega644 PA Datasheet,“ 24 März 2024. [Online]. Available: https://www.mouser.at/ProductDetail/Microchip-Technology/ATMEGA644PA-AU?qs=jWm4idrLpzylZp8jH09TIw%3D%3D. |
| [41] | „IP Schutzklassen,“ 25 03 2024. [Online]. Available: https://pinlight.eu/e/ip-schutzklassen/. |

# Abkürzungen

ASCII American Standard Code for Information Interchange

CSV Comma Seperated Values

FAT *File Allocation Table*

GPS *Global Positioning System*

LoRa *Long Range*

NMEA *National Marine Electronics Association*

ORM *Objet Relational Mapper*

OSP *One Socket Protocol*

RSSI *Received Signal Strength Indicator*

SD-Karte *Sichere Digitale Karte*

SLC Single Level Cell

SPI *Serial Peripheral Interface*

TLC *Triple Level Cells*

UART Universal Asynchronous Receiver Transmitter

# Abbildungen

[Abbildung 1: Projekt Logo 4](#_Toc163449123)

[Abbildung 2: Project Logo 6](#_Toc163449124)

[Abbildung 3: Blockschaltbild 11](#_Toc163449125)

[Abbildung 4: GANT-Diagramm Putz 21](#_Toc163449126)

[Abbildung 5: GANTT-Diagramm König 22](#_Toc163449127)

[Abbildung 6: Blockdiagramm des C-Track Systems 23](#_Toc163449128)

[Abbildung 7: Buck Converter Schaltung 25](#_Toc163449129)

[Abbildung 8: Strom und Spannungsverlauf des Buck Converters 25](#_Toc163449130)

[Abbildung 9: Boost Converter 26](#_Toc163449131)

[Abbildung 10: Strom und Spannungsverlauf des Boost Converters 27](#_Toc163449132)

[Abbildung 11: Buck - Boost - Converter 28](#_Toc163449133)

[Abbildung 12: Typische Applikation des TPS 63001 [3] 29](#_Toc163449134)

[Abbildung 13: Messung der Ausgangripples an der Platine 29](#_Toc163449135)

[Abbildung 14: Messungen des Datenblattes [3] 29](#_Toc163449136)

[Abbildung 15: Aufbau einer NMEA-Satz [4] 30](#_Toc163449137)

[Abbildung 16: Aufbau eines OSP-Satz [4] 31](#_Toc163449138)

[Abbildung 17: NMEA-Sentence mit RMC-Format 31](#_Toc163449139)

[Abbildung 18: RMC-Format [4] 31](#_Toc163449140)

[Abbildung 19: Trickle Power mode [4] 32](#_Toc163449141)

[Abbildung 20: Befehle für die Kommunikation mit SD/MMC-Karten (vgl. [8]) 35](#_Toc163449142)

[Abbildung 21: R1-Response [8] 35](#_Toc163449143)

[Abbildung 22: IP- Schutzklassen 38](#_Toc163449144)

[Abbildung 23: Chirp Signal im Zeitbereich [15] 42](#_Toc163449145)

[Abbildung 24.Frequenzverlauf eines LoRa-Signals [16] 43](#_Toc163449146)

[Abbildung 25: Encoding/Decoding von LoRa [16] 43](#_Toc163449147)

[Abbildung 26: Interleaving (vgl. [17]) 44](#_Toc163449148)

[Abbildung 27: LoRa Paket [19] 45](#_Toc163449149)

[Abbildung 28: Richtdiagramm molex Dipol [22] 46](#_Toc163449150)

[Abbildung 29: Richtdiagramm Antenne Basisstation [23] 47](#_Toc163449151)

[Abbildung 30: Serieller Monitor Basisstation 50](#_Toc163449152)

[Abbildung 31: Datenbankstruktur 52](#_Toc163449153)

[Abbildung 33: dependencies in pubspec.yaml 56](#_Toc163449154)

[Abbildung 34: Code für Launcher Icon 56](#_Toc163449155)

[Abbildung 35: Code: Debug Banner deaktivieren 57](#_Toc163449156)

[Abbildung 36:Code: Text in App Bar 57](#_Toc163449157)

[Abbildung 37: Fertige AppBar 57](#_Toc163449158)

[Abbildung 38: Code: Drop Down Menu Header 58](#_Toc163449159)

[Abbildung 38: Code: Button einfügen 58](#_Toc163449160)

[Abbildung 40: Drop Down Menu 58](#_Toc163449161)

[Abbildung 41: Code: txt-File einlesen 59](#_Toc163449162)

[Abbildung 42: Code: initState 59](#_Toc163449163)

[Abbildung 43: About Us Page 59](#_Toc163449164)

[Abbildung 44: Code: Flutter Map 60](#_Toc163449165)

[Abbildung 45: Code: Tile Layer 60](#_Toc163449166)

[Abbildung 46: Code: Marker Layer 60](#_Toc163449167)

[Abbildung 47: Code: Serververbindung 61](#_Toc163449168)

[Abbildung 48. Code: Serververbindung 61](#_Toc163449169)

[Abbildung 49: Code: Marker Position 61](#_Toc163449170)

[Abbildung 50: Code: initState 61](#_Toc163449171)

[Abbildung 51: Startbildschirm der Smartphone Applikation 62](#_Toc163449172)

[Abbildung 52:Startbildschirm der Smartphone Applikation 63](#_Toc163449173)

[Abbildung 53: Gelötete Platine V1 63](#_Toc163449174)

[Abbildung 54: Schaltplan V1 73](#_Toc163449175)

[Abbildung 54: Board V1 74](#_Toc163449176)

[Abbildung 56: Schaltplan V2 75](#_Toc163449177)

[Abbildung 57: Board V2 76](#_Toc163449178)

# Begleitprotokoll gemäß § 9 Abs. 2 PrO

## Begleitprotokoll Samuel Putz

**Name:** Hr. Samuel Putz

**Diplomarbeitstitel:** C-Track 2.0

|  |  |  |
| --- | --- | --- |
| **KW** | **Beschreibung** | **Zeitaufwand** |
| 38 | Betreuersuche und Besprechung/Planung des Projekts | 10h |
| 39 | Einarbeiten in altes Projekt | 10h |
| 40 | Besprechung mit Dr. Gudrun Wallentin | 5h |
| 41 | Diplomarbeitsantrag erstellen & absenden | 6h |
| 42 | Einarbeiten & testen des alten Projektes | 5h |
| 43 | Schaltplan | 10h |
| 44 | Board-Design | 5h |
| 45 | Herbstferien |  |
| 46 | Board-Design & Auswahl der Bauteile; Projektreview | 10h |
| 47 | Fertigstellung des Boards | 10h |
| 48 | Krankenstand Projektreview | 10h |
| 49 | Löten der Platine und erste Testungen | 10h |
| 50 | Testen des GPS-Moduls und Testen der LoRa Verbindung mit neuen Antennen | 10h |
| 51 | Aufsetzen und einlernen in Flutter | 10h |
| 52 | Weihnachtsferien |  |
| 1 | Weihnachtsferien |  |
| 2 | Einlernen in Flutter | 10h |
| 3 | Projektpräsentation ; Projektreview ; Tag der offenen Tür | 10h |
| 4 | Einlernen in Flutter | 10h |
| 5 | Präsentation | 10h |
| 6 | Platine V2 | 5h |
| 7 | Semesterferien |  |
| 8 | Diplomarbeit schreiben | 5h |
| 9 | Diplomarbeit schreiben | 5h |
| 10 | Programmierung der HandyApp | 10h |
| 11 | Programmierung der HandyApp | 10h |
| 12 | Testen des Projekts / schreiben der Diplomarbeit | 10h |
| 13 | Osterferien |  |

**KW …**Kalenderwoche

## Begleitprotokoll Simon König

**Name:** Hr. Simon König

**Diplomarbeitstitel:** C-Track 2.0

|  |  |  |
| --- | --- | --- |
| **KW** | **Beschreibung** | **Zeitaufwand** |
| 38 | Betreuersuche und Besprechung/Planung des Projekts | 10h |
| 39 | Einarbeiten in altes Projekt | 10h |
| 40 | Besprechung mit Dr. Gudrun Wallentin | 5h |
| 41 | Diplomarbeitsantrag erstellen & absenden | 6h |
| 42 | Einarbeiten & testen des alten Projektes | 5h |
| 43 | Datenbank | 10h |
| 44 | Datenbank | 5h |
| 45 | Herbstferien |  |
| 46 | Auswahl der Bauteile ; Projektreview | 10h |
| 47 | Rest API | 10h |
| 48 | Rest API ; Projektreview | 10h |
| 49 | Löten der Platine und erste Testungen | 10h |
| 50 | Testen des GPS-Moduls und Testen der LoRa Verbindung mit neuen Antennen | 10h |
| 51 | Aufsetzen und einlernen in Flutter | 10h |
| 52 | Weihnachtsferien |  |
| 1 | Weihnachtsferien |  |
| 2 | Low Powermode Konfiguration | 10h |
| 3 | Projektpräsentation ; Projektreview ; Tag der offenen Tür | 10h |
| 4 | GPS-Trickle Powermode | 10h |
| 5 | GPS-Trickle Powermode | 10h |
| 6 | Lora | 5h |
| 7 | Semesterferien |  |
| 8 | Lora | 5h |
| 9 | Diplomarbeit schreiben | 5h |
| 10 | Diplomarbeit schreiben | 10h |
| 11 | Optimieren der Sendeeinheit | 10h |
| 12 | Testen des Projekts / Diplomarbeit schreiben | 10h |
| 13 | Osterferien |  |

**KW …**Kalenderwoch

# Ein Bild, das Text, Diagramm, Plan, technische Zeichnung enthält. Automatisch generierte Beschreibung Anhang

Abbildung 53: Schaltplan V1

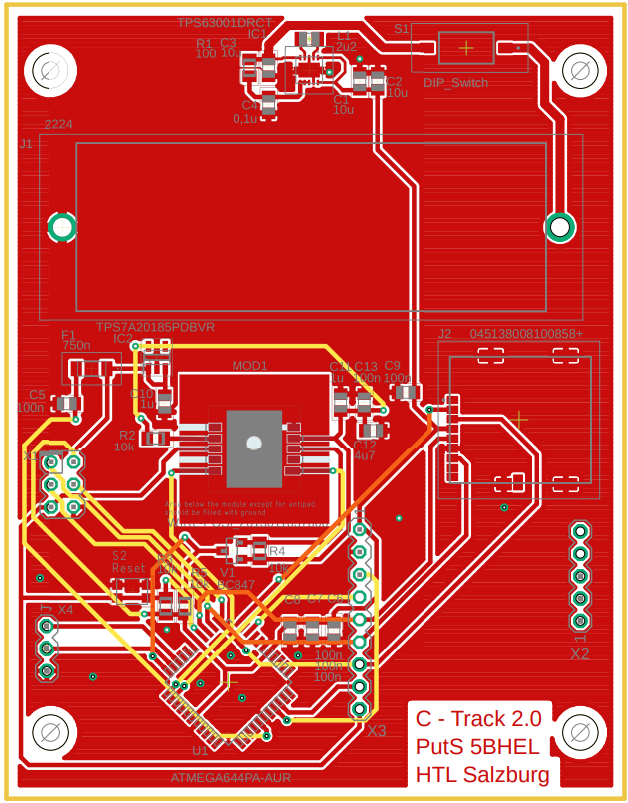


Abbildung 54: Board V1

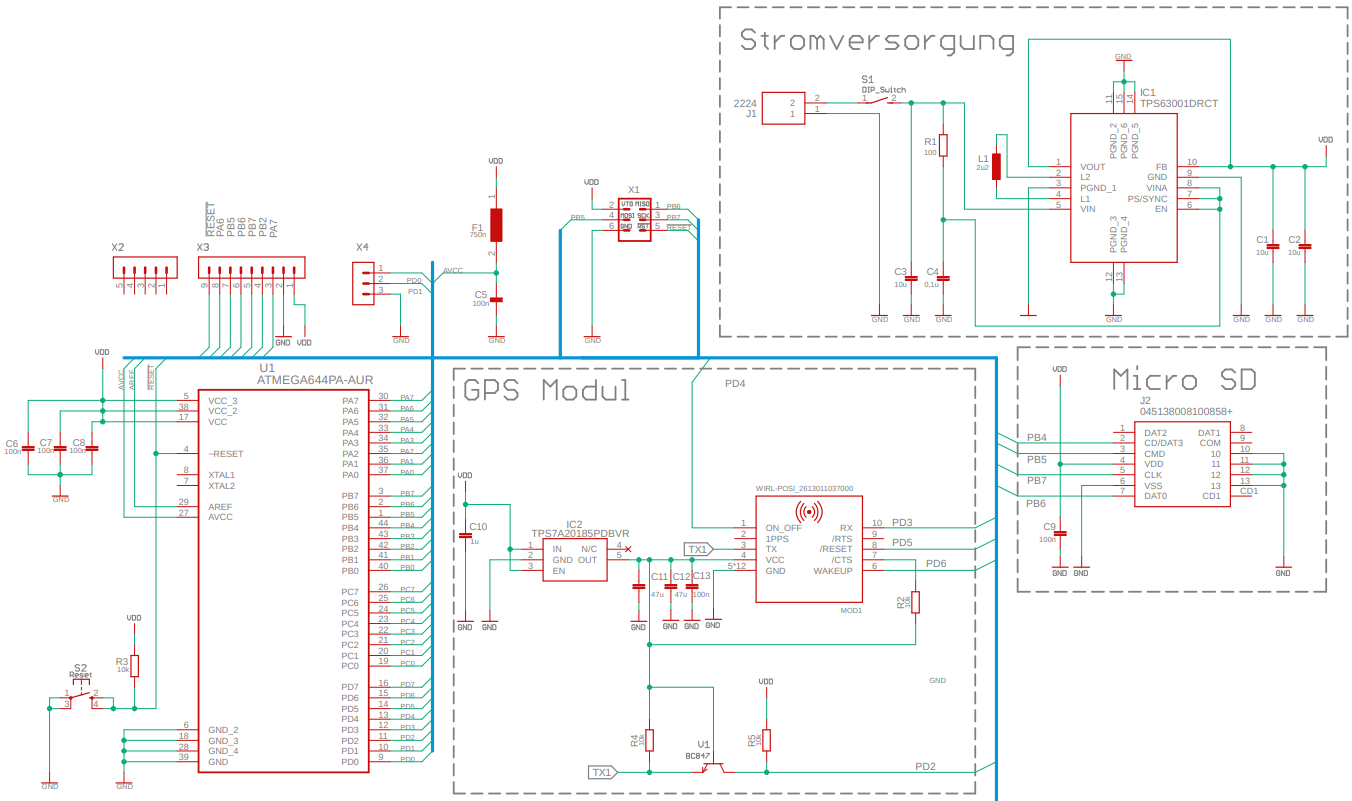


Abbildung 55: Schaltplan V2

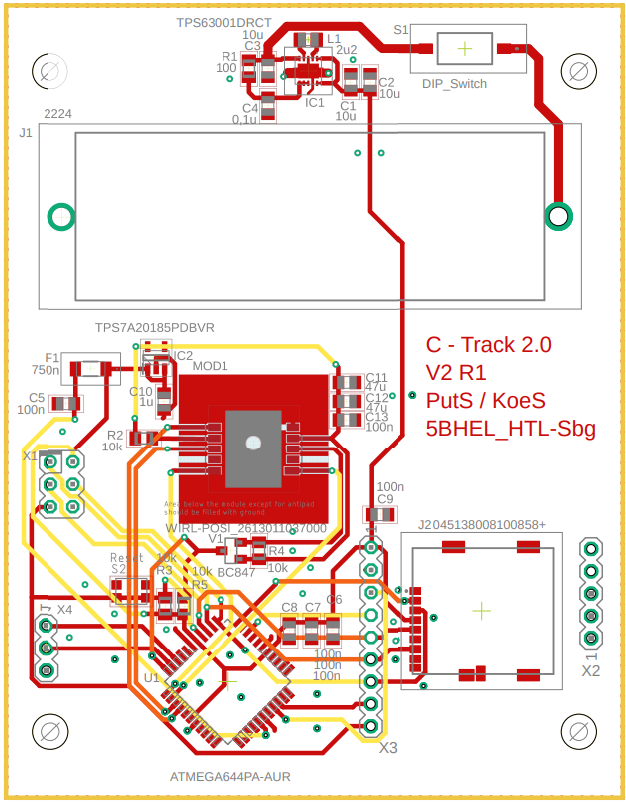


Abbildung 56: Board V2