HTL | MÖSSINGERSTRASSE

HÖHERE TECHNISCHE BUNDESLEHRANSTALT
KLAGENFURT, MÖSSINGERSTRASSE

ABTEILUNG ELEKTRONIK UND TECHNISCHE INFORMATIK

DIPLOMARBEIT

POTSConnect

JAHRGANG 5BHEL

eingereicht von Johannes Schlager
Thomas Niederkofler

Projektbetreuer Dipl.-Ing. Dr. Dieter Maier

Diese Diplomarbeit entspricht den Standards gemäß dem Leitfaden zur Umsetzung der Reife- und Diplomprüfung des BMBWF in der letztgültigen Fassung.

Klagenfurt, am 06.04.2026

HTL | MÖSSINGERSTRASSE

EIDESSTATTLICHE ERKLÄRUNG

Ich versichere an Eides statt, dass ich diese Diplomarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe. Alle Gedanken, die im Wortlaut oder in grundlegenden Inhalten aus unveröffentlichten Texten oder aus veröffentlichter Literatur übernommen, oder mit künstlicher Intelligenz generiert wurden, sind ordnungsgemäß gekennzeichnet, zitiert und mit genauer Quellenangabe versehen.

Verfasser

Johannes Schlager	Thomas Niederkofler

Klagenfurt, am 06.04.2026

Kurzbeschreibung

Ziel dieser Diplomarbeit ist es, ein Modul zu entwickeln, dass ein altes analoges Telefon in ein digitales Telefon umwandelt. Man sollte mit diesem Telefon dann normale Anrufe tätigen und empfangen können.

Aufgabenstellung

Ziel dieser Diplomarbeit ist die Entwicklung eines Moduls, das ein herkömmliches analoges Telefon in ein digitales Kommunikationsgerät umwandelt. Damit soll älteren Menschen ermöglicht werden, die vertraute Technik weiterhin zu nutzen und gleichzeitig von den Vorteilen moderner digitaler Kommunikation zu profitieren.

Das System soll analoge Telefonsignale digitalisieren und über eine Netzwerkverbindung weiterleiten. Zusätzlich wird ein einfach zu bedienendes Dadurch entsteht eine Verbindung zwischen klassischer Telefontechnik und heutiger digitaler Kommunikation.

Realisierung

Das primäre Alleinstellungsmerkmal des Projekts ist die nahtlose Integration moderner Digitaltechnik in ein Gehäuse, das direkt mit dem gewohnten analogen Telefon verbunden wird. Dadurch kann effektiv jene Hardware weitergenutzt werden, die den Anwendern seit Jahrzehnten vertraut ist. Das Modul wird einfach zwischen den Telefonstecker und die Stromversorgung geschaltet, wodurch einerseits ein hohes Maß an Interoperabilität mit alten Endgeräten erreicht wird und zusätzlich an der gewohnten Bedienung keinerlei Änderungen erforderlich sind. Das System bezieht seine Energie aus einer herkömmlichen Steckdose, welche einen wartungsfreien Dauerbetrieb ermöglicht und eine komplizierte Installation überflüssig macht.

Ein wesentlicher Teil für die Modernisierung des Telefons ist die Kommunikation zwischen der analogen Mechanik und dem digitalen Mobilfunknetz. Zum einen soll die Erfassung der Wählimpulse von der Wählscheibe sowie die Übertragung der Sprachdaten über das GSM-Netz möglichst verzögerungsfrei und stabil erfolgen. Aus diesem Grund wird im Modul ein ESP32-Mikrocontroller in Kombination mit einem SIM808-Mobilfunkmodul verwendet, was eine unabhängige Verbindung ohne vorhandenes WLAN ermöglicht. Ein weiterer wichtiger Teil ist die integrierte Rufstromschaltung, die eine ausreichend hohe Spannung erzeugt, um die mechanische Glocke im Inneren des Telefons bei eingehenden Anrufen zuverlässig zum Läuten zu bringen.

Das Alleinstellungsmerkmal des POTSConnect-Moduls ist es, dass alle notwendigen Elemente – von der Impulserkennung über die Signalverarbeitung bis hin zur Mobilfunkantenne – in einem einzigen Gerät platziert sind. Dies verhindert eine aufwendige Installation am Einsatzort, da keine Verbindung zum Internet-Router oder eine Verkabelung durch das Haus

notwendig ist. Das System ist durch die eingelegte SIM-Karte sofort einsatzbereit und ermöglicht den uneingeschränkten Zugang zur Telefonie, ohne dass das gewohnte Gefühl der Sicherheit oder die einfache Handhabung des Wählscheibentelefons verloren gehen.

Ergebnisse

Alle grundlegenden Funktionen des POTSConnect-Moduls konnten erfolgreich getestet und anschließend zu einem Gesamtsystem zusammengefügt werden. Die Kommunikation zwischen dem ESP32-Mikrocontroller und dem SIM808-Mobilfunkmodul funktioniert einwandfrei. Das Ermitteln der Wählimpulse direkt von der alten Wählscheibe wurde ebenfalls vollständig realisiert. Für einen finalen Aufbau müsste das Gehäuse noch optimiert werden, um alle Bauteile und die Antenne noch platzsparender unterzubringen.

Das Modul wurde als Prototyp fertiggestellt und muss lediglich noch mit dem analogen Telefon verbunden werden. Die Signalisierung eingehender Anrufe durch die integrierte Rufstromschaltung funktioniert zuverlässig, sodass die mechanische Glocke wie gewohnt läutet. Ein wesentlicher Teil der Ergebnisse ist die Sprachqualität: Die Umwandlung der analogen Töne in digitale Mobilfunksignale wird komplett durch das SIM808-Modul übernommen, was in den Tests zu einer stabilen und klaren Verständigung führte. Das Abspeichern der notwendigen Konfigurationen auf dem Mikrocontroller und das automatische Einwählen in das Mobilfunknetz wurden ebenfalls erfolgreich realisiert.

Alle grundlegenden Funktionen wurden also umgesetzt. Für eine finale Anfertigung müssten jedoch noch einige Feinheiten an der Platine geändert werden. Insgesamt zeigt der Prototyp jedoch, dass die Verbindung von jahrzehntealter Hardware und moderner Mobilfunktechnik ohne komplizierte Zwischenschritte für den Endnutzer funktioniert.

Kurztitel: POTSConnect

Schlüsselwörter: Analog, Digital, Kommunikation, Telefon, Modernisierung

Abstract

The aim of this diploma thesis is to develop a module that converts an old analog telephone into a digital telephone. It should then be possible to make and receive normal calls with this telephone.

Assignment of Tasks

The aim of this diploma thesis is to develop a module that converts a conventional analog telephone into a digital communication device. This is intended to enable older people to continue using familiar technology while also benefiting from the advantages of modern digital communication.

The system is designed to digitize analog telephone signals and transmit them via a network connection. In addition, it will feature a user-friendly interface. This creates a link between classic telephone technology and today's digital communication.

Implementation

The primary unique selling point of this project is the seamless integration of modern digital technology into a housing that is directly connected to a familiar analog telephone. This allows hardware that users have been accustomed to for decades to continue being used effectively. The module is simply placed between the telephone plug and the power supply, ensuring a high level of interoperability with older devices while requiring no changes to the usual operation. The system is powered by a standard electrical outlet, enabling maintenance-free continuous operation and eliminating the need for complex installation.

A key aspect of modernizing the telephone is the communication between the analog mechanics and the digital mobile network. On the one hand, the detection of dial pulses from the rotary dial and the transmission of voice data via the GSM network should be as delay-free and stable as possible. For this reason, the module uses an ESP32 microcontroller in combination with a SIM808 cellular module, enabling an independent connection without the need for existing Wi-Fi. Another important component is the integrated ringing circuit, which generates a sufficiently high voltage to reliably activate the mechanical bell inside the telephone during incoming calls.

The unique feature of the POTSConnect module is that all necessary components—from pulse detection and signal processing to the cellular antenna—are integrated into a single device. This avoids complex installation at the point of use, as no connection to an internet router or additional household wiring is required. With an inserted SIM card, the system is immediately ready for operation and provides full access to telephony, without sacrificing the familiar sense of reliability or the simple handling of a rotary dial telephone.

Results

All fundamental functions of the POTSConnect module were successfully tested and subsequently integrated into an overall system. Communication between the ESP32 microcontroller and the SIM808 cellular module works flawlessly. The detection of dial pulses directly from the old rotary dial was also fully implemented. For a final version, the housing would need to be optimized to accommodate all components and the antenna in a more space-efficient manner.

The module has been completed as a prototype and only needs to be connected to the analog telephone. The signaling of incoming calls via the integrated ringing circuit works reliably, allowing the mechanical bell to ring as usual. A key aspect of the results is the voice quality: the conversion of analog tones into digital cellular signals is fully handled by the SIM808 module, which resulted in stable and clear communication during testing. The storage of necessary configurations on the microcontroller and the automatic connection to the cellular network were also successfully implemented.

All core functions have therefore been realized. However, for a final production version, some refinements to the circuit board would still be required. Overall, the prototype demonstrates that combining decades-old hardware with modern cellular technology is possible without complex steps for the end user.

Short title: POTSConnect

Keywords: Analog, Digital, Communication, Phone, Modernization

Vorwort

Diese Diplomarbeit entstand im Rahmen meiner Ausbildung an der HTL Mössingerstraße im Zweig Elektronik und technische Informatik. Ziel der Arbeit war es, ein bestehendes analoges System zu modernisieren und digitalisieren und dabei sowohl theoretisches Wissen als auch praktische Fähigkeiten anzuwenden.

Die Idee zu diesem Projekt entstand durch meine Mutter. Besonders spannend war für mich die Herausforderung, ein klassisches POTS-Telefon mit aktueller Technologie zu kombinieren und funktional umzusetzen.

Während der Umsetzung konnten viele neue Erkenntnisse gewonnen werden. Besonders lehrreich waren dabei die praktischen Probleme, die im Laufe des Projekts aufgetreten sind und eigenständig gelöst werden mussten.

Ein besonderer Dank gilt Herrn Dipl. Ing. Dr. Dieter Maier für seine Geduld und seine Ruhe, sowohl auch für die Unterstützung und fachliche Begleitung während der gesamten Projektphase. Ebenso möchte ich mich bei Felix Mikosch und Nico A. Roth bedanken, die mich bei der Umsetzung unterstützten.

Insgesamt kann ich sagen, dass diese Arbeit nicht nur eine technische Herausforderung war, sondern auch eine wertvolle Erfahrung für meine persönliche und berufliche Entwicklung. Dieses Projekt hat mich auch darin bestärkt, in Zukunft eine andere berufliche Richtung zu verfolgen, die besser zu meinen Interessen und Fähigkeiten passt.

Johannes Schlager

Inhaltsverzeichnis

1. Einleitung

In der heutigen, schnelllebigen digitalen Welt haben viele ältere Menschen Schwierigkeiten, mit modernen Technologien wie Smartphones und Tablets Schritt zu halten. Die Bedienung eines Handys kann für sie eine Herausforderung darstellen, sei es durch die komplizierte Benutzeroberfläche, kleine Bildschirme oder die Vielzahl an Funktionen, die nicht immer intuitiv sind. Dennoch sind sie auf die Vorteile der Kommunikation angewiesen, sei es für den Kontakt zu Familienmitgliedern, Freunden oder für Notfälle.

Viele ältere Menschen sind mit der einfachen, aber zuverlässigen Technologie der klassischen analogen Telefone aufgewachsen, die es ihnen ermöglichten, mit einfachem Drehen an der Wählscheibe zu telefonieren. Diese Technologie ist robust, überschaubar und vermittelt ein Gefühl von Vertrautheit. Doch im Zeitalter von Smartphones und digitalen Kommunikationsmitteln verliert das analoge Telefon zunehmend an Bedeutung.

Die vorliegende Diplomarbeit verfolgt das Ziel, diese alte Technik zu modernisieren und zu erhalten. Hierfür wird ein Modul entwickelt, das es ermöglicht, ein herkömmliches analoges Telefon in ein modernes digitales Telefon umzuwandeln. Auf diese Weise können auch ältere Menschen, die mit der Bedienung eines Smartphones überfordert sind, weiterhin auf die bewährte Technik zurückgreifen, aber gleichzeitig von den Vorteilen der digitalen Kommunikation profitieren. So wird der Zugang zur modernen Kommunikation ohne den Verlust der gewohnten Benutzerfreundlichkeit ermöglicht.

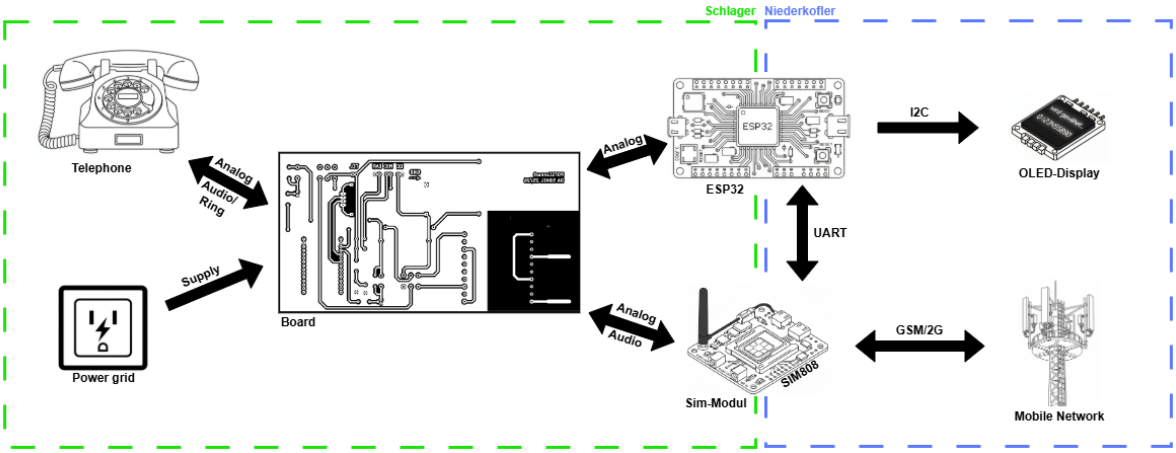

Abbildung 1.1: Systemstrukturplan

2. Grundlagen und Methoden

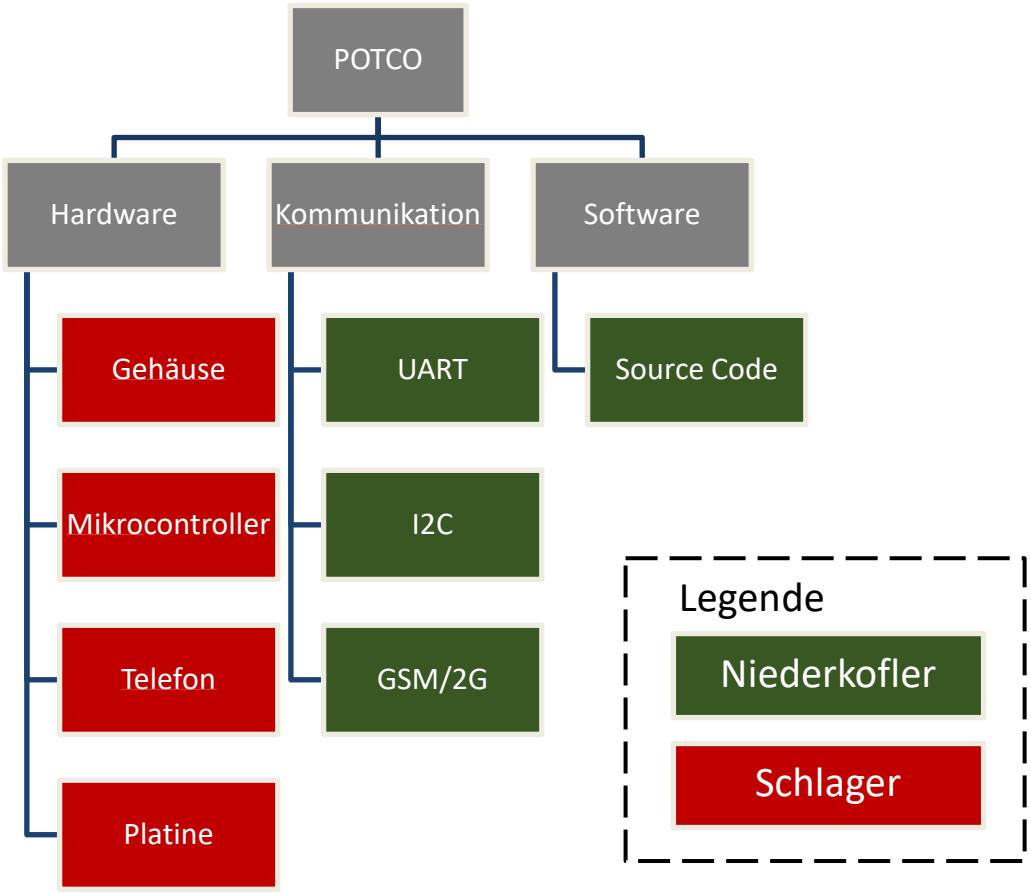

Abbildung 2.1 Produktstrukturplan

2.1. Unterstützung von künstlicher Intelligenz

In der vorliegenden Diplomarbeit wurden ChatGPT und Google Gemini als unterstützendes Tool eingesetzt. Sämtliche Texte wurden zur Gänze eigenständig verfasst und von ChatGPT lediglich korrigiert.

3. Entwicklung, Versorgung, Analyse (Johannes Schlager)

3.1. Aufgabenstellung individueller Teil

Das Ziel dieser Diplomarbeit ist die Entwicklung und der Bau eines speziellen Elektronik-Moduls. Dieses Modul dient als Brücke, um ein klassisches analoges Telefon mit Wählscheibe an das moderne digitale Kommunikationsnetz anzuschließen. Die Hardware muss dabei so gestaltet sein, dass die ursprüngliche Bedienung des Telefons komplett erhalten bleibt, während im Hintergrund die Umwandlung der Signale stattfindet.

Zunächst muss eine Schnittstelle geschaffen werden, die das Telefon mit Strom versorgt und alle analogen Signale verarbeitet. Dazu gehört eine Schaltung, die eine ausreichend hohe Spannung erzeugt, um die mechanische Glocke des Telefons bei einem eingehenden Anruf läuten zu lassen. Ebenso muss die Hardware in der Lage sein, die Impulswahl der alten Wählscheibe präzise zu erfassen und diese Klick-Signale in digitale Wahlinformationen zu übersetzen.

Als Herzstück der Hardware wird ein Mikrocontroller eingesetzt, der die gesamte Steuerung übernimmt. Dieser verarbeitet die Wahlsignale und leitet die Sprachdaten über ein Sim-Modul drahtlos in das Mobilfunknetz weiter. Die gesamte Elektronik soll auf einer kompakten Platine Platz finden, die durch ein handelsübliches Netzteil über eine Steckdose versorgt wird. Abschließend ist ein passendes Gehäuse zu entwerfen, das die Technik schützt und eine einfache, unauffällige Platzierung direkt neben dem Telefon ermöglicht.

3.1.1. Funktionen POTS-Telefon

Ein POTS-Telefon, ausgeschrieben **P**lain **O**ld **T**elephone **S**ervice Telefon, ist ganz simpel aufgebaut besteht aus ein paar Kondensatoren, Widerstände und sämtlichen Elektronik Kleinbauteilen. Ein POTS-Telefon benötigt hauptsächlich nur 2 Anschlüsse. [1]

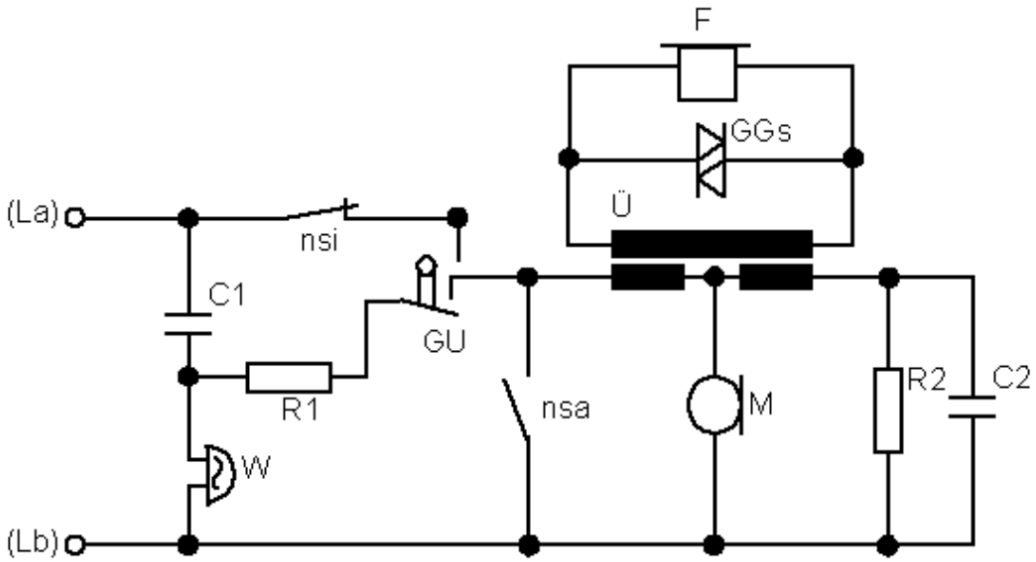

Abbildung 3.1: Schaltung eines alten Telefons (Hörer aufgelegt)[1]

3.1.1.1. Die Bezeichnung POTS

POTS ist ein englischer Fachbegriff, der übersetzt so viel wie „der gute alte Telefondienst“ bedeutet. Es ist die Abkürzung für Plain Old Telephone Service. POTS bezeichnet den ursprünglichen analogen Telefondienst. Man hat diesen Begriff erfunden, um die einfachen, traditionellen Telefondienste von den moderneren, digitalisierten Netzen (wie dem heutigen Internet oder erweiterten Telefonnetzen) klar abzugrenzen.

[1]https://elektronikbasteln.pl7.de/wp-content/uploads/2018/11/1telefon-schaltbild-ohne-erdtaste-klingel.png

3.1.1.2. Telefonieren mit POTS

1. **Wählen**
    a. Die Nummer wird entweder über eine Wählscheibe oder über einen Tastenwahlblock gewählt.
    b. Das Drehen der Wählscheibe/drücken der Taste sorgt für einen Kurzschluss, was den Wählvorgang signalisiert.
    c. Nach loslassen der Taste, werden Impulse über die Leitung gesendet, gewählte Zahl = Impulsanzahl (Abbildung 3.2).
2. **Klingeln**
    a. Das Telefon wird ganz einfach mit Wechselspannung versorgt, welche einen Hammer zwischen zwei Glocken zum Schwingen bringt.
3. **Telefonieren**
    a. Das Sprachsignal wird mittels Mikrofones aufgenommen und in elektrische Wellen umgewandelt.
    b. Diese Wellen verändern sich je nach Lautstärke und Stimmenhöhe. Je höher die Stimme, desto höher die Frequenz. Je lauter die Stimme, desto höher die Spannung.
    c. Diese Wellen kommen bei dem Gesprächspartner an und werden dort wiederum verstärkt und über einen Lautsprecher ausgegeben.
4. **Auflegen**
    a. Durch das Auflegen des Hörers, wird der Schaltkreis unterbrochen und es fließt kein Strom mehr. [2]

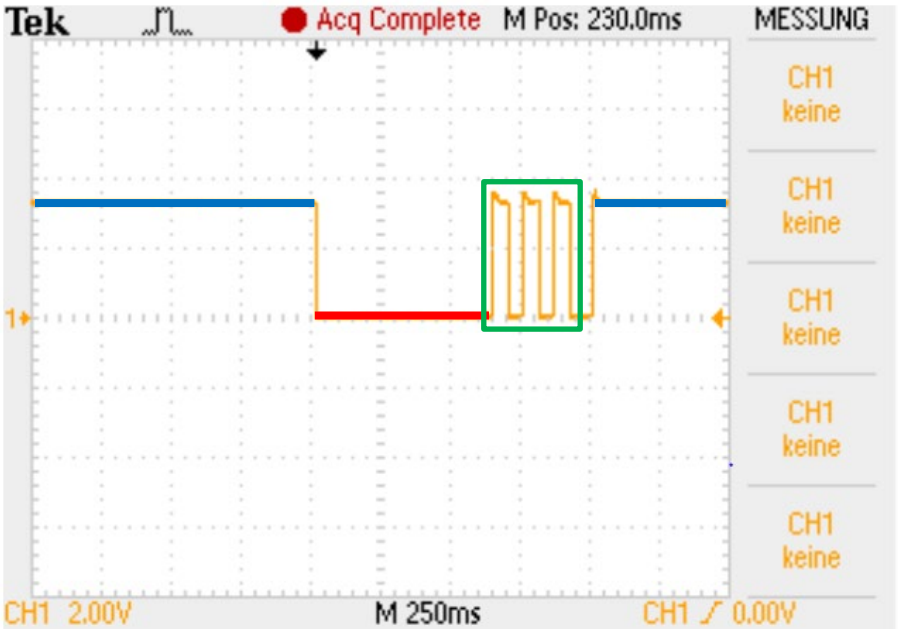

Abbildung 3.2: Beispiel Nummerneingabe

In Abbildung 3.2 wurde als Beispiel die Zahl 3 gewählt.

- Blau: Ruhezustand
- Rot: Aufziehen der Wählscheibe
- Grün: Impulse = Zahl

3.1.1.3. POTS-Anschluss

Der POTS-Anschluss, besser bekannt unter dem Begriff TAE-Anschluss (Telekommunikations-Anschluss-Einheit) ist der „normale“ standardisierte Anschluss für alte Telefone. [3]

Abbildung 3.3: TAE, Stecker und Buchse[2]

Modernere Telefone besitzen meistens schon einen RJ12 oder auch RJ45 Stecker. Der RJ45 Stecker wird auch in der Netzwerktechnik verwendet. Zusätzlich gibt es auch noch einen RJ11 Stecker.

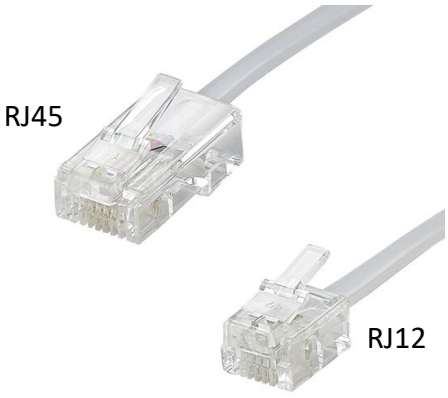

Abbildung 3.4: RJ45 und RJ12 Stecker[3]

[2]https://telefonmanufaktur.de/158-large_default/adapter-tae-f-steckertae-nff-und-6p4c-buchse.jpg
[3]https://www.sunany.com/wp-content/uploads/2022/07/Difference-Between-RJ45-and-RJ12-of-POS-System.jpg

Der RJ45 Stecker besitzt acht, der RJ12 sechs und der RJ11 vier Pins.

Wichtig sind nur zwei Pins bei einem POTS-Anschluss, die a- und b-Ader. Die a-Ader kann man sich als der Pluspol und die b-Ader als der Minuspol vorstellen. Die gesamte Sprachübertragung sowie das Impulswahlverfahren werden nur über diese beiden Adern übertragen.

Man unterscheidet auch zwischen verschiedenen TAE-Steckern. TAE-Stecker und -Buchsen besitzen jeweils entweder eine F-, N oder Z-Kodierung.

- Das **F** steht für **F**ernsprechen und ist für Telefone vorgesehen.
- Die Kodierung **N** steht für **n**icht Fernsprechen (umgangssprachlich auch **N**ebengerät) und damit für alle Endgeräte außer Telefonen, dazu zählen zum Beispiel Anrufbeantworter, Faxgeräte, Modems und Gebührenanzeiger, aber auch z. B. Telefon-Fax-Kombigeräte.
- Die Kodierung **Z** steht für **Z**usatzgeräte und wurde für serielle Datenschnittstellen (Modem, auch an Standleitungen) verwendet.
- An Telefonanlagen und Terminaladaptern finden sich gelegentlich **U**-kodierte Buchsen („**u**niversal“), in die man wahlweise einen N- oder F-Stecker einführen kann, widerspricht jedoch dem Sinn des TAE-Prinzips. [3]

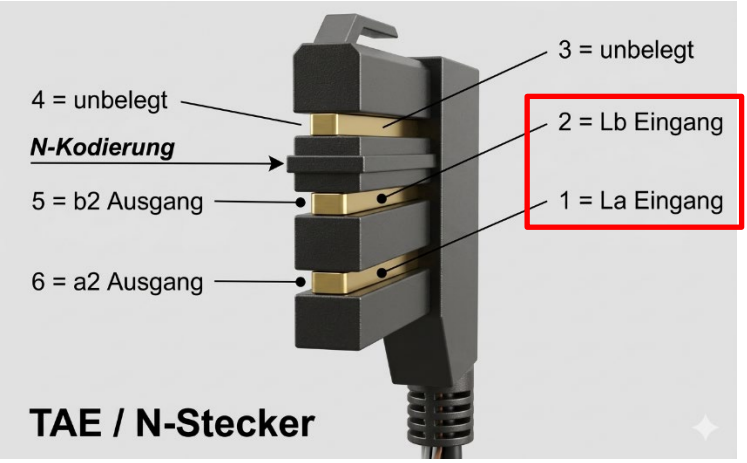

Abbildung 3.5: TAE-Stecker Pinbelegung[4]

[4] Generiert mit Gemini 3, 27.03.2026, Prompt: verschärfe dieses Bild: https://www.husvedvf.de/its/mat/tae/telsn1b.jpg

3.1.1.4. Versorgung

Ein altes Telefon benötigt sowohl Gleich- als auch Wechselspannung. Die Wechselspannung wird hauptsächlich zum Läuten verwendet; auch die Sprachübertragung erfolgt über Wechselspannung. Um das Telefon klingeln zu lassen, wird eine Spannung von 40-90 V AC bei 25Hz benötigt. Je geringer die Spannung, umso leiser, und je höher die Frequenz, desto schriller klingelt das Telefon. Ist die Spannung jedoch zu gering, so klingelt das Telefon gar nicht. Für die Sprachübertragung reichen bereits 150mV. Je lauter man spricht, desto höher ist die Spannung.

Um die Abnahme des Hörers zu erkennen, wird eine Gleichspannung von etwa 48 V DC angelegt. Sobald der Hörer abgehoben wird, fließt ein Strom von ca. 200 mA. Die Gleichspannung wird außerdem genutzt, um das Wählen an der Wählscheibe zu erkennen. [1]

3.1.2. Spannungsversorgungskonzept/Blockschaltbild

In Abbildung 3.6 ist das Spannungsversorgungskonzept von POTSConnect zu sehen. Für die Versorgung wurde sich für einen Transformator und ein Netzteil entschieden.

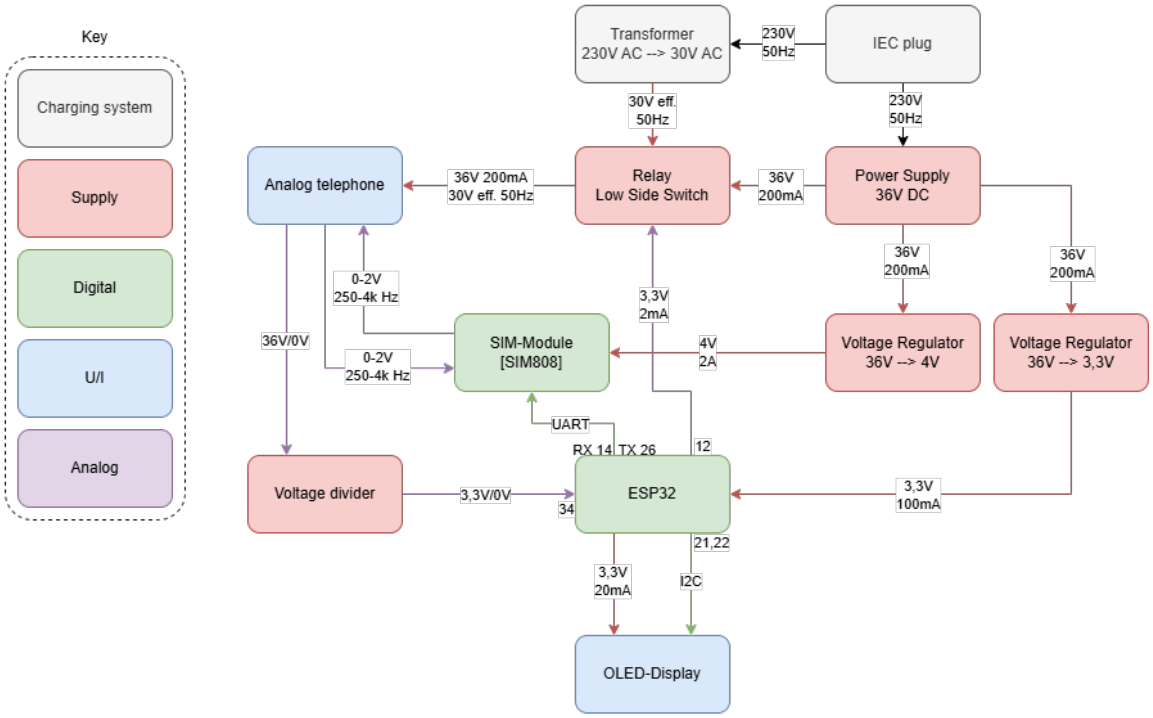

Abbildung 3.6: Blockschaltbild/Spannungsversorgungskonzept

3.1.2.1. Betriebszustand

Das System wird dauerhaft in Betrieb sein, da das Telefon ständig mit Spannung versorgt werden muss, um zu erkennen, ob der Hörer abgenommen wird. Außerdem muss der ESP32 das Telefon bei einem eingehenden Anruf zum Läuten bringen. Da das System somit permanent läuft und vergleichsweise hohe Spannungen benötigt, ist ein Akkubetrieb nur schwer bis kaum umsetzbar.

3.1.2.2. Stromverbrauch

Situation	Bauteil	Strom [mA]
Standby	Ausgang Netzteil	**90**
	Telefon	**0,3**
	SIM-Modul	**200**
Läuten	Ausgang Netzteil	**90**
	Telefon	**70**
	SIM-Modul	**200**
Telefonieren	Ausgang Netzteil	**300**
	Telefon	**80**
	SIM-Modul	**500**
Kommunikation AT-Netz	SIM-Modul	**Bis zu 2000**

Tabelle 3.1: Stromverbrauch Gesamtsystem

In Tabelle 3.1 ist der Stromverbrauch des Gesamtsystems aufgelistet. Diese Übersicht zeigt, wie viel Strom die verschiedenen Teile des Geräts in unterschiedlichen Situationen benötigen. Dabei fällt auf, dass der Stromverbrauch stark davon abhängt, was das System gerade macht:

- **Im Standby:** Das System befindet sich im Ruhezustand. Während das Telefon selbst inaktiv ist (0 mA), benötigen das Netzteil und das SIM-Modul zusammen 460 mA, um die Betriebsbereitschaft und die Netzverbindung aufrechtzuerhalten.
- **Beim Läuten:** Sobald ein Anruf eingeht, wird das Telefon aktiv (70 mA). Interessanterweise sinkt der Verbrauch am Netzteilausgang in diesem Moment auf 70 mA ab, während das SIM-Modul weiterhin konstant 160 mA zieht.
- **Während des Telefonierens:** In diesem Modus steigt die Last deutlich an. Besonders das SIM-Modul benötigt mit 500 mA viel Energie, um die Sprachdaten kontinuierlich zu senden und zu empfangen.
- **Spitzenlast (Kommunikation AT-Netz):** Der höchste Wert tritt bei der aktiven Kommunikation mit dem Mobilfunknetz auf. Hier entstehen kurze Stromspitzen von bis zu **2000 mA (2 A)**. Wichtig ist hierbei, dass es sich nicht um einen dauerhaften Verbrauch handelt, sondern um sehr kurze Impulse (sogenannte Peaks), die immer dann auftreten, wenn das Modul mit voller Leistung Daten an den Sendemast überträgt.

Zusammenfassend lässt sich sagen: Solange sich das Gerät im Standby befindet, fließt wenig Strom. Sobald aber Signale verarbeitet oder gesendet werden müssen – besonders bei der direkten Verbindung mit dem Handynetz –, steigt der Stromverbrauch deutlich an.

3.2. Ansteuerung Telefon

3.2.1. Läuten des Telefons

Ein Telefon benötigt wie in Kapitel 3.1.1.4 besprochen ca. 40 bis 90V AC, um zu läuten. Als erster Test wurde ein Funktionsgenerator mit einer Spannung von 10Vpp und 25Hz verwendet. Da 20Vpp zu wenig sind wurde noch ein Leistungsverstärker hinzugefügt. Nach 10 Minuten testen, wurde festgestellt, dass der Leistungsverstärker kaputt war, und deswegen kein Klingeln zu hören war.

Abbildung 3.7: Funktionsgenerator mit Leistungsverstärker

Als nächstes wurde eine Gleichspannung von 48V direkt auf die Klingel angeschlossen, um zu testen, ob die Klingel überhaupt funktionstüchtig war. Befand sich der Hammer der Klingel auf der richtigen Position, so war ein kurzes „Ding“ zu hören. Damit bestätigte sich, dass die Klingel funktionierte. Um ein Zurückziehen des Hammers zu verursachen, wurde eine negative Spannung benötigt.

Nach Absprache mit dem Betreuer Dipl.-Ing. Dr. Dieter Maier, wurde uns ein Modul aus der Werkstätte zur Verfügung gestellt. Dort waren bereits passende Anschlüsse gefunden, um ein Telefon klingeln zu lassen. Nach ein paar Tests wurde festgestellt, dass das getestete Telefon bereits bei 27V AC-Effektivwert läutet.

3.2.2. Erstellung einer Versorgungsschaltung

Um das Projekt POTSConnect umzusetzen, muss ein funktionables Versorgungskonzept mit Wechsel- und Gleichspannung erfolgen. Zu Beginn wurde ein Akkusystem geplant, welches aber sofort wieder verworfen wurde, da dies zu komplex für die zeitlichen Möglichkeiten gewesen wäre. Bei einem Akkusystem müsste man die Gleichspannung in Wechselspannung umwandeln.

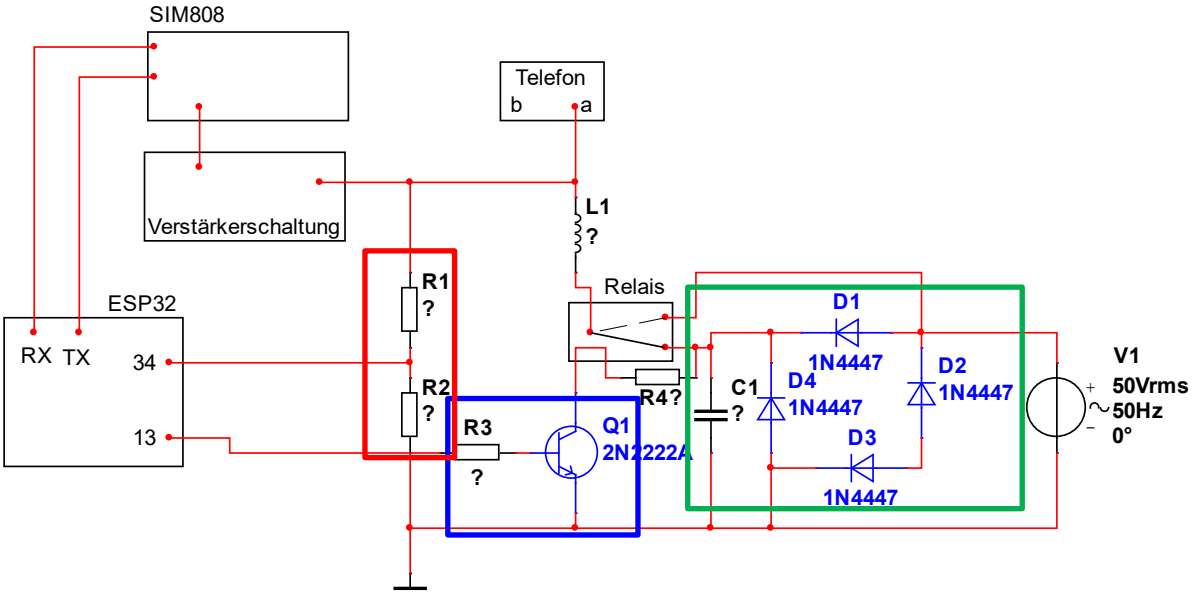

Abbildung 3.8: Erstes Versorgungskonzept

Abbildung 3.8 stellt das erste Versorgungskonzept grob dar. Diese „Skizze“ wurde mit dem Betreuer Herrn Dieter Maier erstellt. Die Berechnungen der Bauteile sind erst nach Erstellung erfolgt.

Der Gleichrichter (grün markiert) wird benötigt, um die Wechselspannung in Gleichspannung umzuwandeln. Der Kondensator C1 ist für das Glätten der Spannung zuständig.

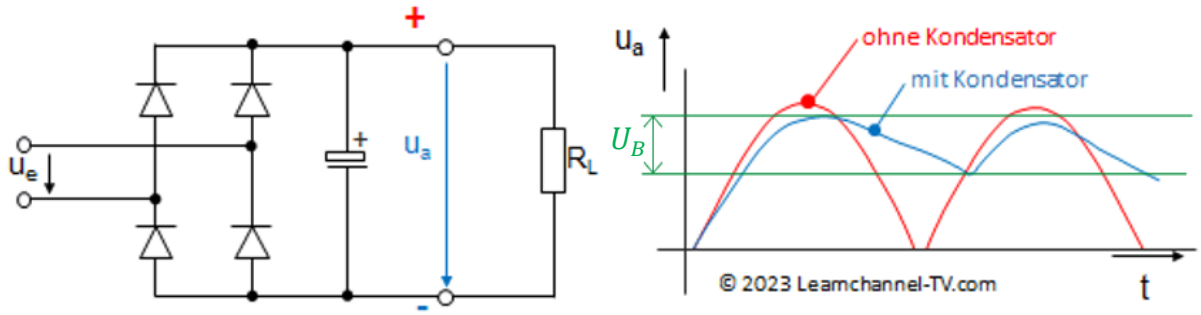

Abbildung 3.9: Gleichrichterschaltung[5]

[5] https://learnchannel-tv.com/de/elektronik/gleichrichterdiode/gleichrichter/

In Abbildung 3.9 ist schön zu sehen, wie so ein Gleichrichter funktioniert. Die maximale Gleichspannung kann nur ~1,4V der Wechselspannung betragen, da pro Diode ~0,7V abfallen. Je größer der Wert des Kondensators, desto weniger Brummspannung tritt auf. Die Brummspannung ist die Differenz der Maximal- und Mindestspannung des Kondensators, in Abbildung 3.9 ebenfalls grün eingezeichnet. Formel, um **C** zu berechnen:

$$C_{min} = I * \frac{\Delta t}{\Delta U}$$

Formel 3.2-1: Formel zur Berechnung der Kapazität

$\boldsymbol{\Delta t}$ $\frac{T}{2} \rightarrow T = \frac{1}{f}$

$\boldsymbol{\Delta U}$ 10% von U

Das Relais dient zum Schalten der Wechsel und Gleichspannung. Die Idee ist, dass das Relais mittels Low-Side-Switch (Abbildung 3.8, blau markiert) über den ESP32 geschalten wird.

Ein Low-Side-Switch ist ein elektronischer Schalter, bei dem der Transistor zwischen dem Verbraucher und dem Minuspol platziert wird. Der Verbraucher ist dabei dauerhaft mit der Versorgungsspannung verbunden und wird durch den Transistor von der Masse getrennt. Sobald der ESP32 eine Steuerspannung an die Basis des Transistors liefert, schließt dieser die Verbindung zum Minuspol. Dadurch wird der Stromkreis geschlossen, sodass der Strom durch den Verbraucher abfließen kann und das Relais schaltet.

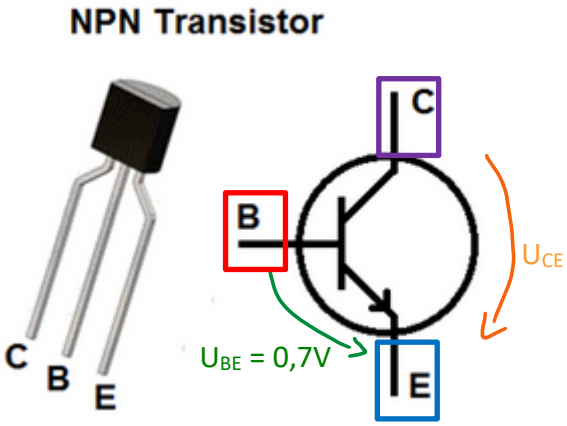

Abbildung 3.10: NPN-Transistor[6]

[6] https://shoptransmitter.com/blog/what-is-the-difference-between-pnp-and-npn/

Um einen Low-Side-Switch zu berechnen, geht man wie folgt vor:

U_B	Spannung an der Basis **B**, wenn ein µC verwendet wird dann die max. Spannung am Pin
U_{BE}	Spannungsabfall von der Basis **B** zum Emitter **E** -> 0,7V -> Diode
I_C	Strom am Collecter **C** – Strom, der durch den Verbraucher fließt
h_{fe}	Verstärkung des Transistors, steht im Datenblatt (Abbildung 3.11). Je nach Strom anders!
S	Sicherheitsfaktor – wird verwendet, da der Strom nicht immer exakt ist
I_B	Strom an der Basis **B**
R_B	Widerstand an der Basis **B**

Gegeben: $U_B,\ U_{BE}, I_C,\ h_{fe}, S$

Berechnungen:

$$I_B = \frac{I_C}{hfe} * S$$

$$R_B = \frac{U_B - U_{BE}}{I_B}$$

Formel 3.2-2: Formeln zur Berechnung eines Low-Side-Switches

DC Current Gain		h_{FE}				-
(I_C = 10 µA, V_{CE} = 5.0 V)	BC547A		–	90	–	
	BC546B/547B/548B		–	150	–	
	BC548C		–	270	–	
(I_C = 2.0 mA, V_{CE} = 5.0 V)	BC546		110	–	450	
	BC547		110	–	800	
	BC548		110	–	800	
	BC547A		110	180	220	
	BC546B/547B/548B		200	290	450	
	BC547C/BC548C		420	520	800	
(I_C = 100 mA, V_{CE} = 5.0 V)	BC547A/548A		–	120	–	
	BC546B/547B/548B		–	180	–	
	BC548C		–	300	–	

Abbildung 3.11: Beispiel hfe eines Transistors(BC547C)

Um zu verhindern, dass das Sprachsignal des Telefons von der Gleichspannung wieder geglättet wird, wurde eine Spule eingebaut.

Der Spannungsteiler (Abbildung 3.9, rot markiert) ist dafür da, um die Nummernwahl am ESP32 zu erkennen. Da der EPS32 maximal 3,3V an seinen Pins schafft, muss ein Spannungsteiler eingebaut werden, um den ESP32 nicht zu zerstören.

3.3. Sprechen über das Telefon

3.3.1. Messen des Sprachsignales

Um ein Sprachsignal zu messen, muss das Telefon mit Gleichspannung versorgt werden. Das Mikrofon wandelt die gesprochene Sprache ganz einfach in elektrische Spannung um. Diese Spannung – ~100mV bis ~1V AC – wird dann auf die Gleichspannung aufsummiert.

Zuerst wurden die Tests direkt an dem Mikrofon durchgeführt und geprüft, ob dieses überhaupt funktioniert. Dazu wurde ein Oszilloskop direkt an der Quelle angeschlossen.

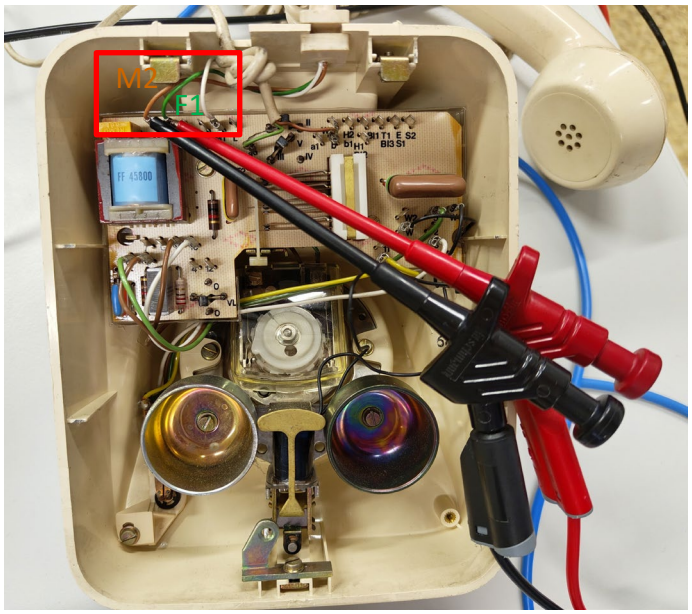

Abbildung 3.12: Messvorrichtung

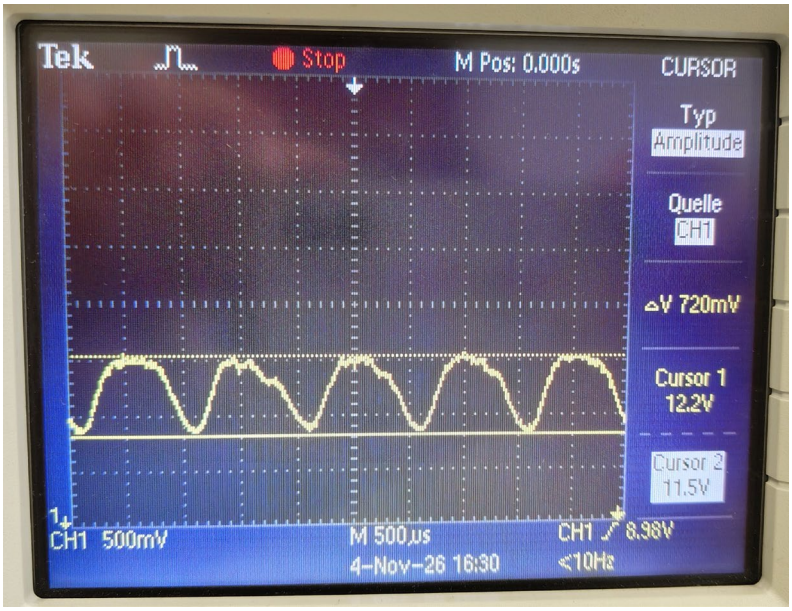

Abbildung 3.13: Gemessenes Signal

In Abbildung 3.13 ist deutlich zu erkennen, wie ein solches Sprachsignal aussieht. Das „gesprochene“ Signal wurde durch ein Mobiltelefon simuliert (Frequenz = 1kHz).

Dieses Signal war aber nur direkt an der „Quelle“ messbar. Um an der a- und b-Ader ein solches Signal zu messen, was das Ziel ist, wurde eine andere Messschaltung benötigt (Abbildung 3.14).

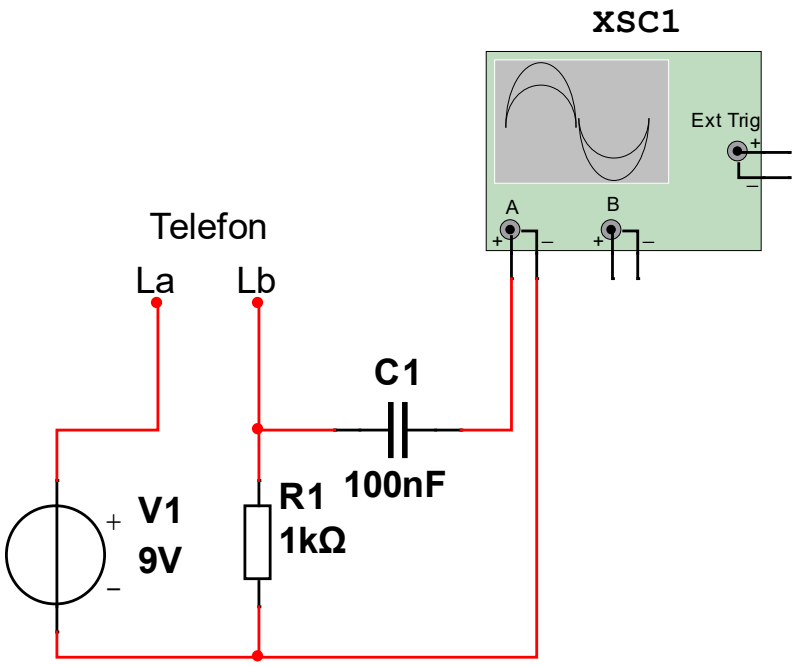

Abbildung 3.14: Messschaltung für das Sprachsignal

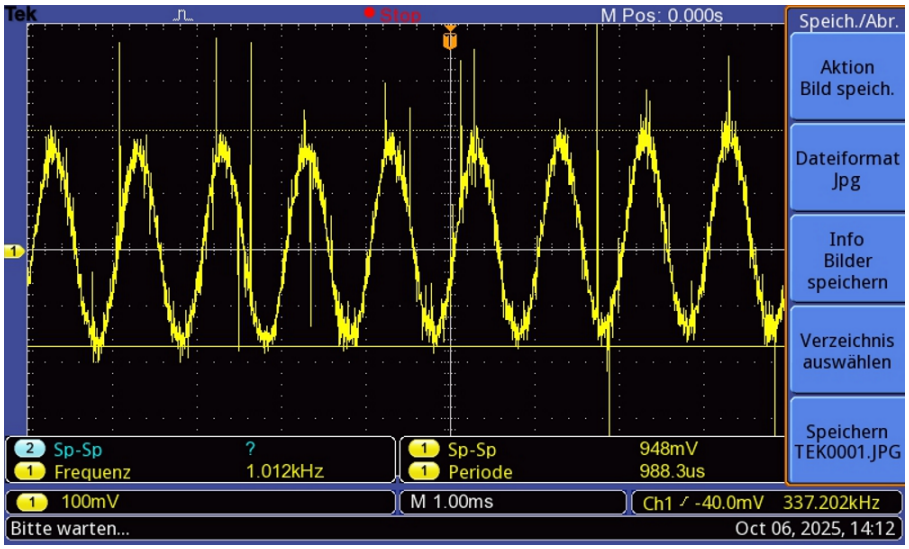

Abbildung 3.15: Messergebnis

Wie in Abbildung 3.15 zu sehen, war das Messen des Sprachsignals an den a- und b-Adern erfolgreich. Hier wurde wieder eine Frequenz von 1kHz mit dem Mobiltelefon simuliert. Danach wurde sogar in den Hörer gesprochen und es war deutlich eine Veränderung des Signales zu erkennen.

Nach Absprache mit dem Betreuer Herrn Dipl.-Ing. Dr. Dieter Maier wurde eine simplere Methode, um das Sprachsignal zu messen, gefunden.

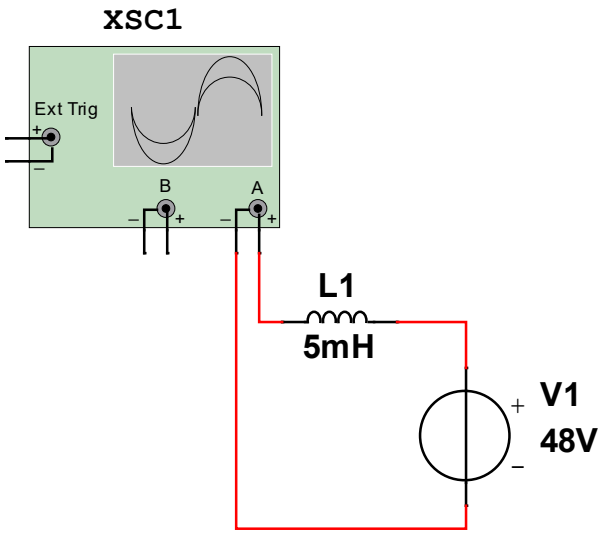

Abbildung 3.16: Messschaltung mit Spule

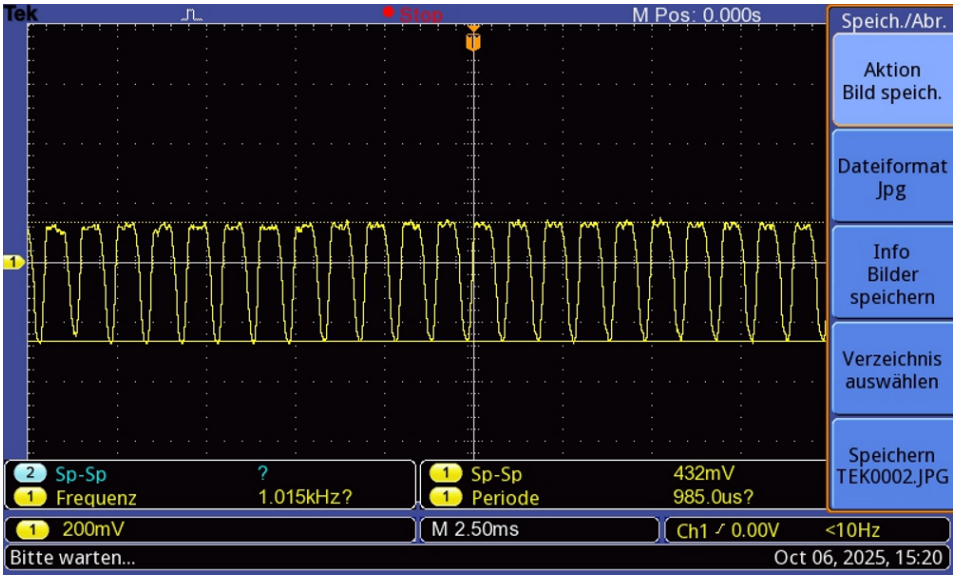

Abbildung 3.17: Messergebnis mit Spule

Der Wert der Spule wurde durch eine L-Dekade durch Testung ermittelt, es wird eine Spule von 10mH verwendet.

3.3.2. Sprachsignal verstärken

Um das Sprachsignal auswerten zu können muss dieses verstärkt werden, da dieses sonst viel zu schwach ist. Dazu wurde eine Schaltung in Multisim entworfen.

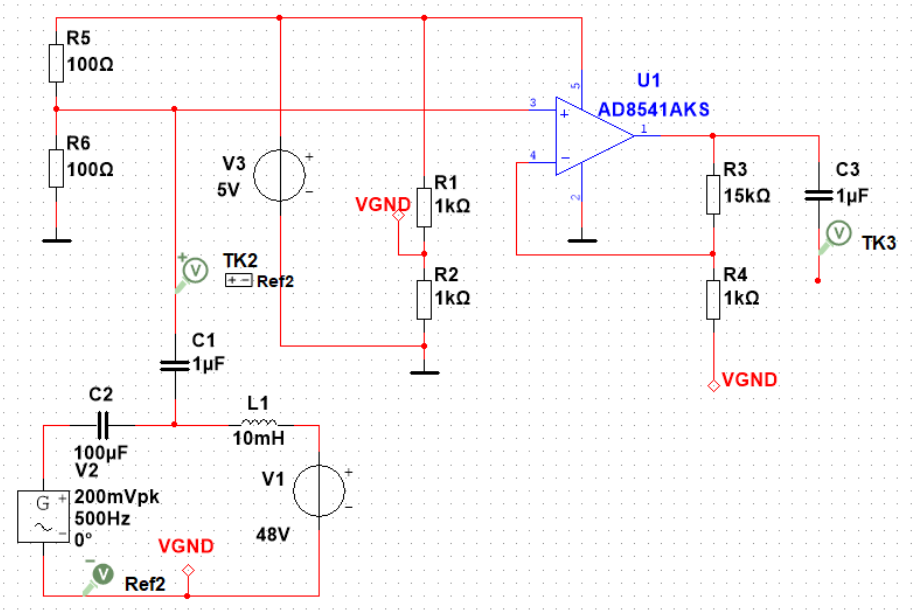

Abbildung 3.18: Schaltung in Multisim

Da wir den Operationsverstärker (OPV) nur mit einer einzigen Spannungsversorgung betreiben, müssen wir einen Single-Supply-OPV verwenden. Um dies zu ermöglichen, wird ein sogenannter Virtual Ground benötigt (VGND).

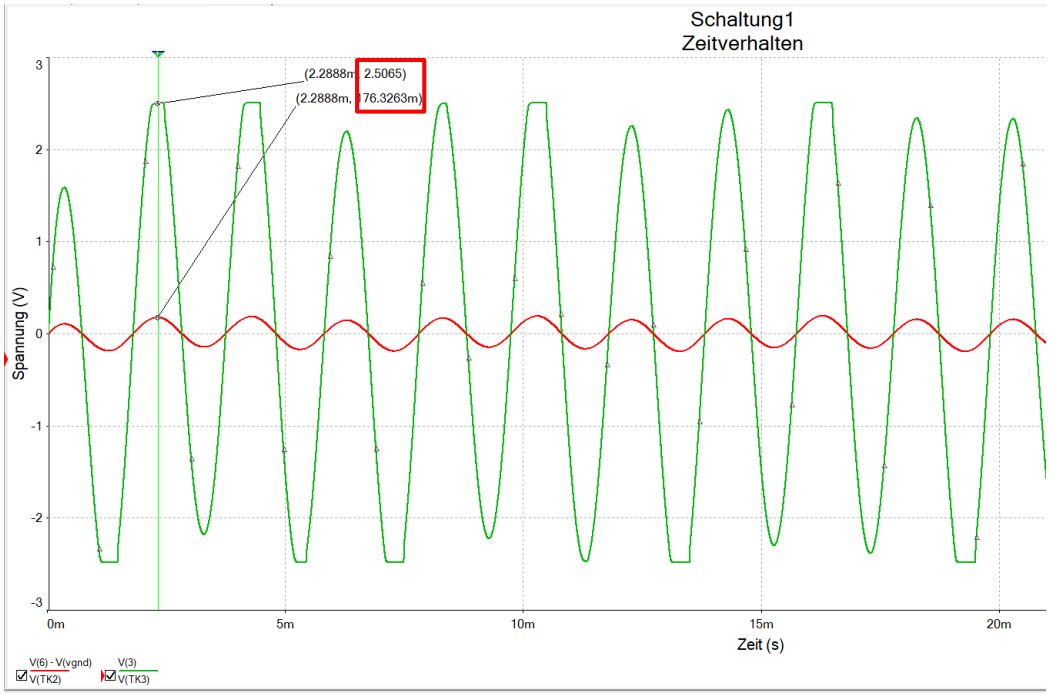

Abbildung 3.19: Simulationsergebnis

In Abbildung 3.19 ist deutlich zu erkennen, wie das Signal verstärkt wird. Von 176mV → 2,5V.

Nach weiterer Recherche wurde festgestellt, dass das Audiosignal selbst bereits einen Peak von 1,5V erreicht und das Sim-Modul keine Probleme hat. Der Entwurf der Verstärkerschaltung wurde also wieder verworfen.

3.4. Erster Prototyp

3.4.1. Dimensionierung der ersten Versorgungsschaltung

Nach Absprache mit dem Betreuer wurde mit dem Dimensionieren der Schaltung aus Abbildung 3.8 begonnen.

3.4.1.1. Berechnen des Gleichrichters:

Für die Berechnung von C benötigen wir die Formel aus 3.2.2:

$$C_{min} = I * \frac{\Delta t}{\Delta U}$$

$$\Delta t = \frac{\frac{1}{50Hz}}{2} = 0{,}01s$$

$$\Delta U = 50 * 0{,}1 = 5V$$

$$I = max.Strom \approx 250mA$$

$$C_{min} = 250mA * \frac{0{,}01}{5} = 500\mu F$$

Formel 3.4-1: Formeln zur Berechnung des Gleichrichters

3.4.1.2. Berechnen der Spule:

Der Wert der Spule wurde nicht berechnet, sondern durch diverse Tests und Abgleichungen evaluiert, hierzu wurde eine L-Dekade verwendet. Nach längerer Suche einer Spule wurde sich aufgrund der spezifischen Anforderungen und mangels passender Bauteile dazu entschieden, eine eigene Spule wickeln.

Da bei 10mH die Länge des Drahtes und die Wicklungen in die Höhe schossen, wurde sich auf einen Wert von 5mH geeinigt. Natürlich wurde dies zuerst auch mit einer L-Dekade getestet und es funktionierte weiterhin einwandfrei. Als Prototyp wurde ein PVC-Rohr mit einer Länge von 30mm und einem Durchmesser von 20mm verwendet.

Für die Berechnung der Wicklungen wird folgende Formel benötigt:

$$N = \sqrt{\frac{L * l}{\mu_0 * \mu_r * A}}$$

Formel 3.4-2: Formel zur Berechnung der Wicklungszahl

$\boldsymbol{N}$	Wicklungsanzahl
$\boldsymbol{L}$	Gewünschte Induktivität
$\boldsymbol{l}$	Länge des Rohres (in m)
$\boldsymbol{A}$	Fläche des Rohres (in m²)
μ_0	Fester Wert ($4\pi * 10^{-7}$)
μ_r	Wert für das Material im Inneren des Rohres (Luft = 1)

$$A = \pi * \left(\frac{d}{2}\right)^2$$

$$A = \pi * \left(\frac{0{,}02}{2}\right)^2 = 0{,}000314m^2$$

$$N = \sqrt{\frac{0{,}005 * 0{,}03}{(4\pi * 10^{-7}) * 1 * 0{,}000314}} = 616{,}561$$

Formel 3.4-3: Berechnung der Wicklungszahl

Um auf Nummer sicher zu gehen wurde ein eigener Rechner zur Kontrolle verwendet.

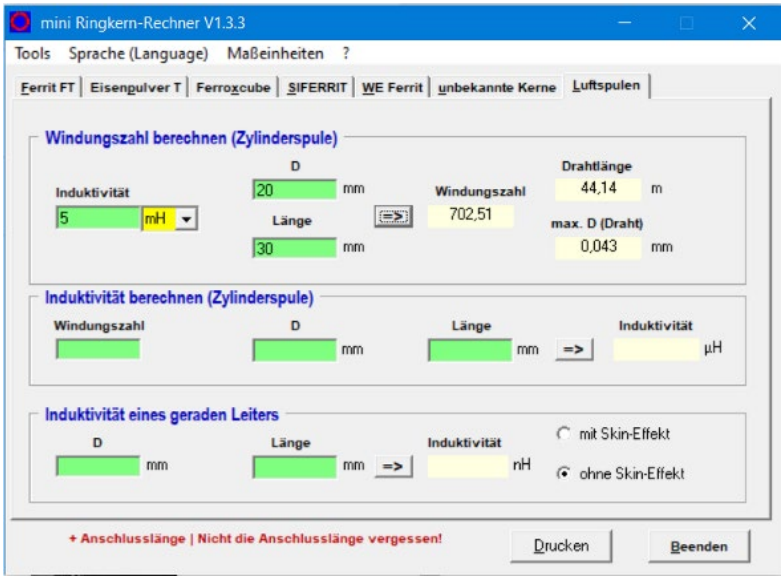

Abbildung 3.20: Rechner zum Berechnen der Spulendaten [4]

Die einfache Formel gilt nur für eine „unendlich lange“ Spule mit einem idealen, verlustfreien Magnetfeld. Da unsere Spule mit 30mm Länge eine kurze Spule ist, bricht das Magnetfeld an

den Enden nach außen aus (Streufeld). Um diesen Effizienzverlust durch die Geometrie auszugleichen, berechnet der Rechner eine höhere Windungszahl für dieselbe Induktivität.

Um eine Induktion von 5mH zu erreichen, werden für das PVC-Rohr folgende Werte benötigt:

Anzahl an Wicklungen: ~700
Länge des Drahtes: ~44m
Dicke des Drahtes: ~0,043mm

Im weiteren Verlauf wurde ein eigene Spule designed (Abbildung 3.21). Damit die Spule nicht davonrollt, und guten Halt später auf der Platine hat, wurde zusätzlich noch ein Gehäuse konstruiert (Abbildung 3.22).

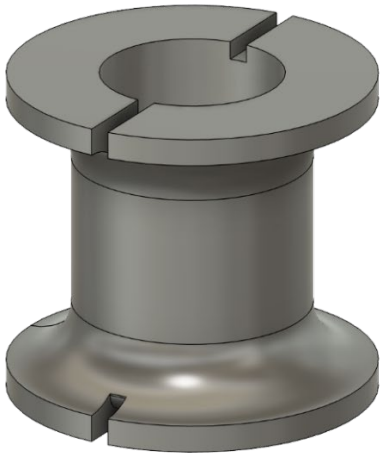

Abbildung 3.21: Spule

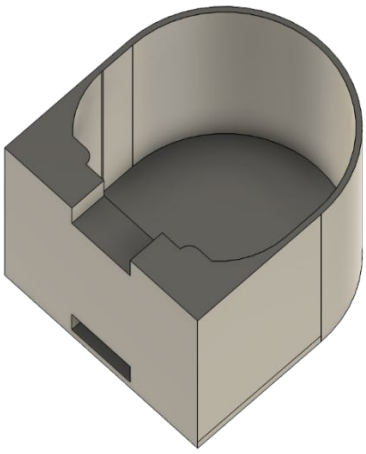

Abbildung 3.22: Gehäuse für die Spule

Nach dem Wickeln der Spule, wurde der Wert gemessen und es wurde ein tatsächlicher Wert mittels Bauteiltester von 6,63mH gemessen

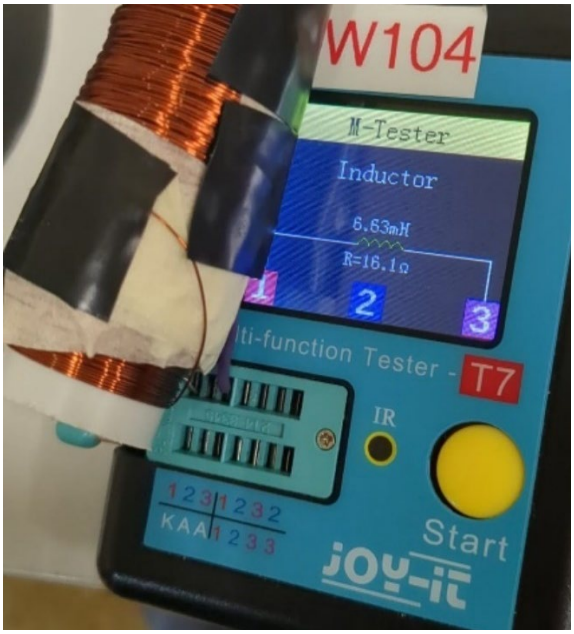

Abbildung 3.23: Messung mittels Bauteiltester

3.4.1.3. Berechnen des Low-Side-Switches

Zum Berechnen des Low-Side-Switches werden die Schritte, wie in 3.2.2 beschrieben, angewendet. Dabei sind die Daten des verwendeten Relais entscheidend.

Das verwendete Relais ist das Finder 36.11-4001, da es sich gut für den Einsatz auf Platinen eignet und die Anforderungen der Schaltung erfüllt. Die Schaltspannung des Relais liegt im Bereich von 9 bis 18 V, wobei die Nennspannung 12 V beträgt. Der benötigte Schaltstrom des Relais beträgt 30 mA.

Die angegebenen Werte stammen aus dem Datenblatt des Relais.

Spulendaten

DC Ausführung

Nenn-spannung U_N	Spulen-code	Arbeitsbereich U_{min}	Arbeitsbereich U_{max}	Widerstand R	Bemessungs-strom I
V		V	V	Ω	mA
3	9.003	2,2	4,5	25	120
5	9.005	3,7	7,5	70	72
6	9.006	4,5	9	100	60
9	9.009	6,7	13,5	225	40
12	9.012	9	18	400	30
24	9.024	18	36	1.600	15
48	9.048	36	72	6.400	7,5

Abbildung 3.24: Auszüge aus dem Datenblatt

Da wir eine Gleichspannung von 69,3V haben, unser Relais aber nur 12V aushält, benötigen wir einen Vorwiderstand. Da dieser Widerstand viel Spannung regulieren muss, wird ein Leistungswiderstand benötigt.

Berechnen des Vorwiderstandes R_R:

$$R_R = \frac{69{,}3 - 12}{0{,}03} = 1{,}91k\Omega$$

$$P_{R_R} = 57{,}3 * 0{,}03 = 1{,}719W$$

Formel 3.4-4: Berechnungen Vorwiderstand Relais

Auf dem Widerstand fallen 57,3V Spannung ab. Das heißt an diesem Widerstand liegt eine Leistung von ~ 1,7W an. Hingegen hält ein herkömmlicher Widerstand nur eine Leistung von bis zu 0,5W aus. Nachdem wir unseren Vorwiderstand eingebaut haben, liegen nur noch die gebrauchten 12V am Relais an (Abbildung 3.25).

Abbildung 3.25: Spannung am Relais

Genauso wichtig wie die Daten des Relais sind die Daten des Transistors. Als Transistor wird ein einfacher NPN-Transistor verwendet. Wir verwenden den BC547B, da dieser unseren Ansprüchen perfekt entspricht und außerdem auch kostengünstig ist.

MAXIMUM RATINGS (T_A = 25°C unless otherwise noted)

Characteristic	Symbol	Value	Unit
Collector – Emitter Voltage	V_{CEO}	40	Vdc
Collector – Base Voltage	V_{CBO}	75	Vdc
Emitter – Base Voltage	V_{EBO}	6.0	Vdc
Collector Current – Continuous	I_C	600	mAdc
Total Device Dissipation @ T_A = 25°C Derate above 25°C	P_D	625 5.0	mW mW/°C
Total Device Dissipation @ T_C = 25°C Derate above 25°C	P_D	1.5 12	W mW/°C
Operating and Storage Junction Temperature Range	T_J, T_{stg}	−55 to +150	°C

Abbildung 3.26: Eigenschaften des Transistors

Die Collector-Emitter Voltage (U_{CE}) ist die Spannung, die zwischen Kollektor und Emitter eines Transistors anliegt. Sie darf einen bestimmten Maximalwert nicht überschreiten, da der Transistor sonst beschädigt wird. In unserem Fall liegen 12V an, während der Transistor bis zu 40V aushält – das ist also sicher. Der Collector Current (I_C) beschreibt den Strom, der durch den Kollektor fließt. Auch hier gibt es einen maximal zulässigen Dauerstrom, der nicht überschritten werden darf. Mit 30mA liegen wir deutlich unter den erlaubten 600mA, wodurch der Transistor ebenfalls im sicheren Bereich betrieben wird.

ON CHARACTERISTICS

DC Current Gain	h_{FE}			–
(I_C = 0.1 mAdc, V_{CE} = 10 Vdc)		35	–	
(I_C = 1.0 mAdc, V_{CE} = 10 Vdc)		50	–	
(I_C = 10 mAdc, V_{CE} = 10 Vdc)		75	–	
(I_C = 10 mAdc, V_{CE} = 10 Vdc, T_A = −55°C)		35	–	
(I_C = 150 mAdc, V_{CE} = 10 Vdc) (Note 1)		100	300	
(I_C = 150 mAdc, V_{CE} = 1.0 Vdc) (Note 1)		50	–	
(I_C = 500 mAdc, V_{CE} = 10 Vdc) (Note 1)		40	–	
Collector–Emitter Saturation Voltage (Note 1)	$V_{CE(sat)}$			Vdc
(I_C = 150 mAdc, I_B = 15 mAdc)		–	0.3	
(I_C = 500 mAdc, I_B = 50 mAdc)		–	1.0	
Base–Emitter Saturation Voltage (Note 1)	$V_{BE(sat)}$			Vdc
(I_C = 150 mAdc, I_B = 15 mAdc)		0.6	1.2	
(I_C = 500 mAdc, I_B = 50 mAdc)		–	2.0	

Abbildung 3.27: h_{fe} aus dem Datenblatt

Berechnen des Basiswiderstandes:

$$I_B = \frac{0{,}03}{100} * 4 = 1{,}2mA$$

$$R_B = \frac{3{,}3 - 0{,}7}{0{,}0012} = 2{,}17k\Omega$$

Formel 3.4-5: Berechnungen des Low-Side-Switches

3.4.1.4. Berechnen des Spannungsteilers

Um das Signal des Telefons (Impulswahl) für den ESP32 nutzbar zu machen, wird ein Spannungsteiler verwendet. Der Spannungsteiler muss die Spannung von 69,3V auf 3,3V reduzieren.

Berechnen der Gleichspannung:

$$50 * \sqrt{2} = 70{,}7V$$

$$70{,}7V - 1{,}4V = 69{,}3V$$

Formel 3.4-6: Berechnen der Spitzenspannung

Der Effektivwert einer Wechselspannung ist der Wert, der bei einem Widerstand die gleiche Leistung erzeugt wie eine gleich große Gleichspannung. Er wird deshalb oft als „wirksamer“ Spannungswert betrachtet und ist der Wert, den man auch mit normalen Messgeräten meist misst. Bei einer sinusförmigen Wechselspannung ist der Effektivwert kleiner als die maximale

Spannung (Spitzenspannung). Der Zusammenhang entsteht durch die mathematische Mittelwertbildung über eine Sinusperiode, wodurch sich ergibt, dass die Spitzenspannung dem Effektivwert multipliziert mit Wurzel 2 entspricht. Die 1,4 V entstehen durch den Spannungsabfall an zwei Dioden, wobei an jeder Diode etwa 0,7 V abfallen. $2 * 0{,}7V = 1{,}4V$

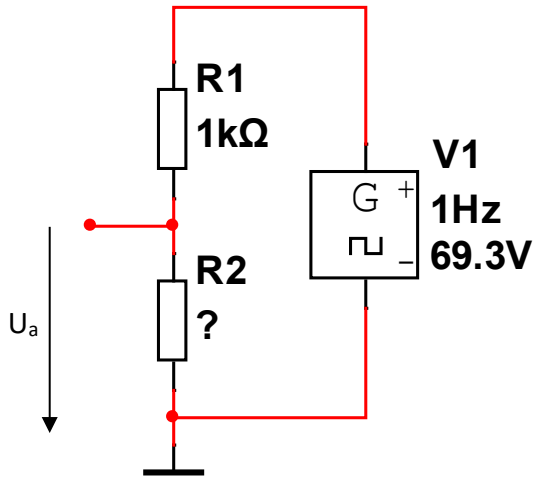

Abbildung 3.28: Schaltung Spannungsteiler

R1 wird mit 1kΩ angenommen. Um R2 zu berechnen wird diese Formel benötigt:

$$R2 = R1 * \frac{U_a}{U_e - U_a}$$

$$R2 = 1000 * \frac{3{,}3}{69{,}3 - 3{,}3}$$

$$R2 = 50\Omega$$

Formel 3.4-7:Berechnungen des Spannungsteilers

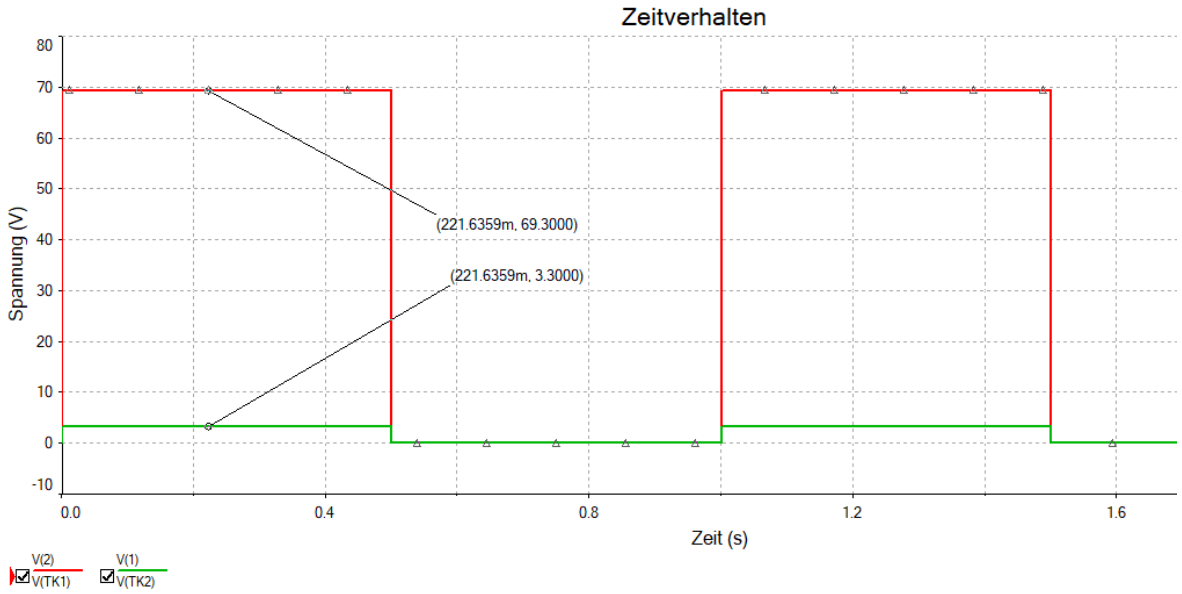

Abbildung 3.29: Simulation Spannungsteiler

Wie in Abbildung 3.29 zu sehen ist, funktioniert der Spannungsteiler einwandfrei und das Signal wird von 69,3V auf 3,3V reduziert.

Als vorzeitige Spannungsquelle zum Testen hätte ein eigener, spezieller Funktionsgenerator verwendet werden sollen. Doch nachdem wir uns auf die Suche nach diesem Gerät begaben, wurden wir nicht fündig. Nach längerer Absprache einigten wir uns darauf, eine Spannungsquelle von 30V eff. zu verwenden, da diese Spannung bereits ausreicht. Zum Testen verwendeten wir dafür das Modul, welches wir bereits bekommen hatten (27V eff.).

Nach der Neuberechnung der Bauteile stellte sich ein weiteres Problem heraus. Durch unseren selbstgebauten Gleichrichter ist die Gleichspannung sehr „unschön“.

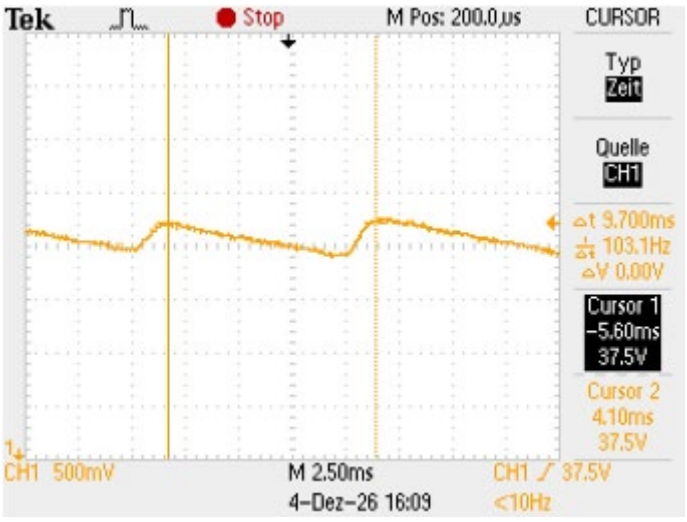

Abbildung 3.30: Beispiel für eine „unschöne“ Gleichspannung

Dieses Signal hat die Messung am ESP32 verfälscht, und die Funktion des Telefons beeinträchtigt. Da es keine Auswirkungen auf die anderen Komponenten des Systems hat, und uns die Arbeit erleichtert, haben wir uns für ein externes Power-Supply entschieden. Wir einigten uns darauf, dass 36V völlig ausreichen. Ein externe Power-Supply ist vorteilhaft, da die Gleichspannung auch wirklich sehr schön ist.

Abbildung 3.31: Power-Supply 36V[7]

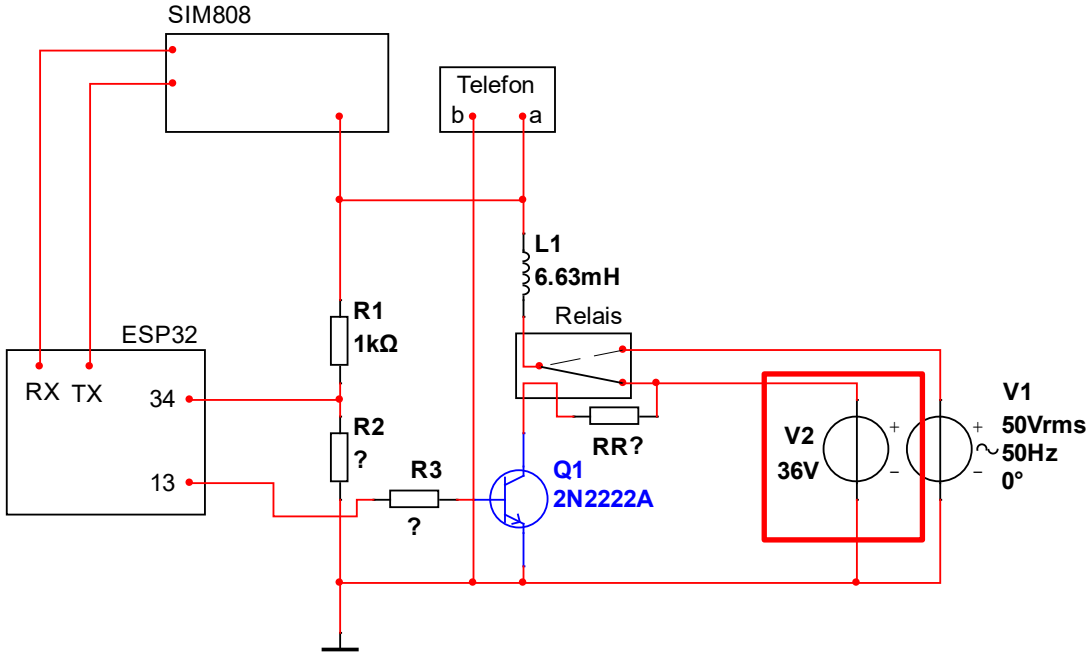

Abbildung 3.32: Versorgungsschaltung mit externem Power-Supply

In Abbildung 3.32 ist jetzt unser externes Power-Supply zu sehen (rot markiert). Der oben verwendete Gleichrichter kann und wurde auch ganz einfach ersetzt.

[7] https://www.reichelt.at/at/de/shop/produkt/schaltnetzteil_geschlossen_50_w_36_v_1_45_a-202964

Während der Planungsphase haben wir uns für ein anderes Relais entschieden. Der Grund für diesen Wechsel war, dass das neue Modell robuster und effizienter arbeitet. Ein entscheidender Vorteil ist zudem, dass es über zwei Wechsler (Umschaltkontakte) verfügt, nicht nur über einen. Entschieden wurde sich für das Relais Finder 40.52.9.012

Spulendaten

DC Ausführung - Standard 0.65 W (Typ 40.31/51/52/61/62)

Nenn-spannung	Spulen-code	Arbeitsbereich		Widerstand	Bemessungs-strom
U_N		U_{min}	U_{max}	R	I
V		V	V	Ω	mA
5	9.005	3.65	7.5	38	130
6	9.006	4.4	9	55	109
7	9.007	5.1	10.5	75	94
9	9.009	6.6	13.5	125	72
12	9.012	8.8	18	220	55
14	9.014	10.2	21	300	47
18	9.018	13.1	27	500	36
21	9.021	15.3	31.5	700	30
24	9.024	17.5	36	900	27
28	9.028	20.5	42	1200	23
36	9.036	26.3	54	2000	18

Abbildung 3.33: Datenblatt Relais Finder 40.52.9.012

Finale Berechnungen:

$$R_R = \frac{36 - 12}{0{,}055} = 436{,}36\Omega$$
$$P_{R_R} = (36 - 12) * 0{,}055 = 1{,}32W$$

Formel 3.4-8: Erneute Berechnung des Vorwiderstandes des Relais

Wir haben bereits einen Leistungswiderstand für 7W mit einem Wert von 470Ω zur Verfügung, somit passt dieser perfekt.

Der Low-Side-Switch ist natürlich auch neu zu berechnen. Obwohl sich der Strom am Kollektor geändert hat, wird die gleiche h_{fe} verwendet.

$$I_B = \frac{0{,}055}{100} * 4 = 2{,}2mA$$

$$R_B = \frac{3{,}3 - 0{,}7}{0{,}0022} = 1{,}18k\Omega$$

Formel 3.4-9: Erneute Berechnungen des Low-Side-Switches

Auch der Spannungsteiler ist neu zu berechnen:

$$R2 = 1000 * \frac{3{,}3}{36 - 3{,}3} = 100{,}9\Omega$$

Formel 3.4-10: Erneute Berechnung des Spannungsteilers

Um diverse Störungen am ESP32 zu filtern verwenden wir parallel zu R_2 einen Kondensator, der uns mögliche Störsignale filtert (C = 100nF).

Für die praktische Umsetzung der Schaltung werden Widerstände aus der international genormten E12-Reihe verwendet. Da Bauteile aufgrund von Fertigungstoleranzen nicht in jedem beliebigen Wert produziert werden können, gibt diese Normreihe feste Standardwerte vor, die den gesamten Widerstandsbereich sinnvoll abdecken. Bei der Dimensionierung der Schaltung ist es daher notwendig, die theoretisch berechneten Idealwerte auf den jeweils nächsten verfügbaren Wert dieser Reihe anzupassen, da nur diese Komponenten im Handel standardmäßig erhältlich sind. Neben der E12 Reihe gibt es auch noch die E3, E6, E24, E48. E96 und die E192 Reihe. Die Zahl hinter dem Buchstaben gibt an, wie viele verschiedene Werte es pro Bereich gibt. Während die E6-Reihe nur sechs grobe Abstufungen bietet, unterteilt die E192-Reihe denselben Bereich in 192 winzige Schritte. Das bedeutet: Je höher die Zahl der Reihe, desto feiner sind die Sprünge und desto genauer kann man einen berechneten Wert in der Praxis treffen. Da bei vielen Werten die Lücken kleiner werden, müssen diese Bauteile auch viel präziser gefertigt werden, damit sie sich nicht gegenseitig überschneiden. Man wählt also die E6 oder E12 für einfache Aufgaben und die hohen Reihen wie E192 nur dann, wenn die Schaltung extrem exakt arbeiten muss.

Die Widerstandsreihe E6 besteht aus den folgenden Widerständen

1 Ω	10 Ω	100 Ω	1 kΩ	10 kΩ	100 kΩ	1 MΩ	10 MΩ
1.5 Ω	15 Ω	150 Ω	1.5 kΩ	15 kΩ	150 kΩ	1.5 MΩ	15 MΩ
2.2 Ω	22 Ω	220 Ω	2.2 kΩ	22 kΩ	220 kΩ	2.2 MΩ	22 MΩ
3.3 Ω	33 Ω	330 Ω	3.3 kΩ	33 kΩ	330 kΩ	3.3 MΩ	33 MΩ
4.7 Ω	47 Ω	470 Ω	4.7 kΩ	47 kΩ	470 kΩ	4.7 MΩ	47 MΩ
6.8 Ω	68 Ω	680 Ω	6.8 kΩ	68 kΩ	680 kΩ	6.8 MΩ	68 MΩ

Abbildung 3.34: Widerstandsreihe E6[8]

Die Widerstandsreihe E24 besteht aus den folgenden Widerständen

1 Ω	10 Ω	100 Ω	1 kΩ	10 kΩ	100 kΩ	1 MΩ	10 MΩ
1.1 Ω	11 Ω	110 Ω	1.1 kΩ	11 kΩ	110 kΩ	1.1 MΩ	11 MΩ
1.2 Ω	12 Ω	120 Ω	1.2 kΩ	12 kΩ	120 kΩ	1.2 MΩ	12 MΩ
1.3 Ω	13 Ω	130 Ω	1.3 kΩ	13 kΩ	130 kΩ	1.3 MΩ	13 MΩ
1.5 Ω	15 Ω	150 Ω	1.5 kΩ	15 kΩ	150 kΩ	1.5 MΩ	15 MΩ
1.6 Ω	16 Ω	160 Ω	1.6 kΩ	16 kΩ	160 kΩ	1.6 MΩ	16 MΩ
1.8 Ω	18 Ω	180 Ω	1.8 kΩ	18 kΩ	180 kΩ	1.8 MΩ	18 MΩ
2 Ω	20 Ω	200 Ω	2 kΩ	20 kΩ	200 kΩ	2 MΩ	20 MΩ
2.2 Ω	22 Ω	220 Ω	2.2 kΩ	22 kΩ	220 kΩ	2.2 MΩ	22 MΩ
2.4 Ω	24 Ω	240 Ω	2.4 kΩ	24 kΩ	240 kΩ	2.4 MΩ	24 MΩ
2.7 Ω	27 Ω	270 Ω	2.7 kΩ	27 kΩ	270 kΩ	2.7 MΩ	27 MΩ
3 Ω	30 Ω	300 Ω	3 kΩ	30 kΩ	300 kΩ	3 MΩ	30 MΩ
3.3 Ω	33 Ω	330 Ω	3.3 kΩ	33 kΩ	330 kΩ	3.3 MΩ	33 MΩ
3.6 Ω	36 Ω	360 Ω	3.6 kΩ	36 kΩ	360 kΩ	3.6 MΩ	36 MΩ
3.9 Ω	39 Ω	390 Ω	3.9 kΩ	39 kΩ	390 kΩ	3.9 MΩ	39 MΩ
4.3 Ω	43 Ω	430 Ω	4.3 kΩ	43 kΩ	430 kΩ	4.3 MΩ	43 MΩ
4.7 Ω	47 Ω	470 Ω	4.7 kΩ	47 kΩ	470 kΩ	4.7 MΩ	47 MΩ
5.1 Ω	51 Ω	510 Ω	5.1 kΩ	51 kΩ	510 kΩ	5.1 MΩ	51 MΩ
5.6 Ω	56 Ω	560 Ω	5.6 kΩ	56 kΩ	560 kΩ	5.6 MΩ	56 MΩ
6.2 Ω	62 Ω	620 Ω	6.2 kΩ	62 kΩ	620 kΩ	6.2 MΩ	62 MΩ
6.8 Ω	68 Ω	680 Ω	6.8 kΩ	68 kΩ	680 kΩ	6.8 MΩ	68 MΩ
7.5 Ω	75 Ω	750 Ω	7.5 kΩ	75 kΩ	750 kΩ	7.5 MΩ	75 MΩ
8.2 Ω	82 Ω	820 Ω	8.2 kΩ	82 kΩ	820 kΩ	8.2 MΩ	82 MΩ
9.1 Ω	91 Ω	910 Ω	9.1 kΩ	91 kΩ	910 kΩ	9.1 MΩ	91 MΩ

Abbildung 3.35: Widerstandsreihe E24[8]

Die Widerstandsreihe E12 besteht aus den folgenden Widerständen

1 Ω	10 Ω	100 Ω	1 kΩ	10 kΩ	100 kΩ	1 MΩ	10 MΩ
1.2 Ω	12 Ω	120 Ω	1.2 kΩ	12 kΩ	120 kΩ	1.2 MΩ	12 MΩ
1.5 Ω	15 Ω	150 Ω	1.5 kΩ	15 kΩ	150 kΩ	1.5 MΩ	15 MΩ
1.8 Ω	18 Ω	180 Ω	1.8 kΩ	18 kΩ	180 kΩ	1.8 MΩ	18 MΩ
2.2 Ω	22 Ω	220 Ω	2.2 kΩ	22 kΩ	220 kΩ	2.2 MΩ	22 MΩ
2.7 Ω	27 Ω	270 Ω	2.7 kΩ	27 kΩ	270 kΩ	2.7 MΩ	27 MΩ
3.3 Ω	33 Ω	330 Ω	3.3 kΩ	33 kΩ	330 kΩ	3.3 MΩ	33 MΩ
3.9 Ω	39 Ω	390 Ω	3.9 kΩ	39 kΩ	390 kΩ	3.9 MΩ	39 MΩ
4.7 Ω	47 Ω	470 Ω	4.7 kΩ	47 kΩ	470 kΩ	4.7 MΩ	47 MΩ
5.6 Ω	56 Ω	560 Ω	5.6 kΩ	56 kΩ	560 kΩ	5.6 MΩ	56 MΩ
6.8 Ω	68 Ω	680 Ω	6.8 kΩ	68 kΩ	680 kΩ	6.8 MΩ	68 MΩ
8.2 Ω	82 Ω	820 Ω	8.2 kΩ	82 kΩ	820 kΩ	8.2 MΩ	82 MΩ

Abbildung 3.36: Widerstandsreihe E12[8]

[8] https://www.electronicsplanet.ch/Widerstand/Widerstandsreihe-E12.php

Somit gilt:

Widerstand	Idealer, berechneter Wert	Widerstandsreihe	Gewählter Wert
R_B	1,18kΩ	E12	1,2kΩ
R_2	100,9Ω	E12	100Ω

Tabelle 3.2: Anpassung der Idealwerte an die E12-Normreihe

3.4.2. Testen der ersten Versorgungsschaltung

Nach Fertigstellung der Berechnungen, wurde die Schaltung aufgebaut und getestet.

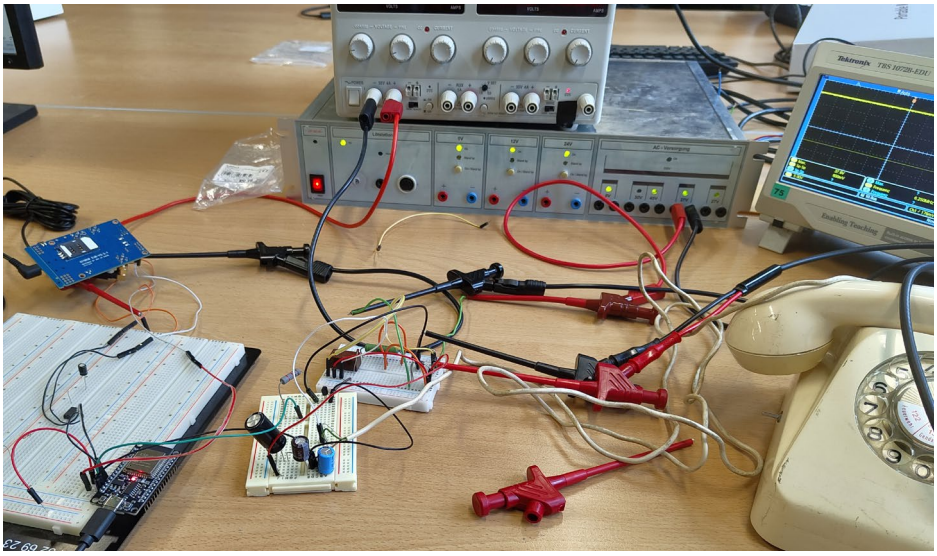

Abbildung 3.37: Aufbau der ersten Schaltung

Wie auf der Abbildung 3.37 zu erkennen ist, wurde für den ersten, grundlegenden Test bewusst auf eine fest verlötete Platine verzichtet. Stattdessen haben wir die gesamte Schaltung vorerst nur provisorisch auf sogenannten Steckbrettern aufgebaut. Der Versuchsaufbau erfolgte bewusst auf einem Steckbrett, um eine flexible Anpassung der Bauteile zu ermöglichen.

Zunächst verlief der Versuch auch sehr erfreulich. Im normalen Ruhezustand verhielt sich der Aufbau genauso, wie wir es vorher auf dem Papier berechnet hatten. Der Strom floss wie geplant, und alle grundlegenden Dinge haben fehlerfrei funktioniert.

Allerdings zeigte sich dann bei der eigentlichen Benutzung des alten Telefons ein unerwartetes Problem. Sobald man das Wählrad (die Wählscheibe) mit dem Finger aufzog, passierte ein Fehler im System. Durch das Aufziehen der Wählscheibe schließt das Telefon bewusst einen Kontakt, wodurch ein Kurzschluss entsteht (Kapitel 3.1.1.2).

Die Folge dieses Fehlers war sofort spürbar: Unsere Stromversorgung war für diese plötzliche Belastung nicht ausgelegt und ist sofort komplett eingebrochen. Die Spannung fiel ab, und der ganze Testaufbau hat sich abgeschaltet. Dieser Vorfall zeigt sehr gut, wie wichtig solche frühen Tests mit losen Kabeln sind, bevor man viel Zeit in eine fertige Platine steckt.

Für den ersten Test wurde das SIM-Modul noch nicht mit dem Telefon verbunden, stattdessen nutzten wir externe herkömmliche Kopfhörer.

Lösung:
Um dieses Problem zu beheben, haben wir eine einfache, aber sehr effektive Lösung umgesetzt: Wir haben einen Leistungswiderstand mit 470 Ohm in den Stromkreis eingebaut.

Dieser Widerstand dient nun als Schutz für unsere Spannungsquelle. Wenn das Wählrad aufgezogen wird und das Telefon den Kontakt schließt, fließt der Strom nun nicht mehr ungehindert ab. Der Widerstand begrenzt den Stromfluss auf einen Wert, den unser Netzteil problemlos bewältigen kann. Man kann sich das wie eine Bremse vorstellen, die verhindert, dass zu viel Energie auf einmal abfließt.

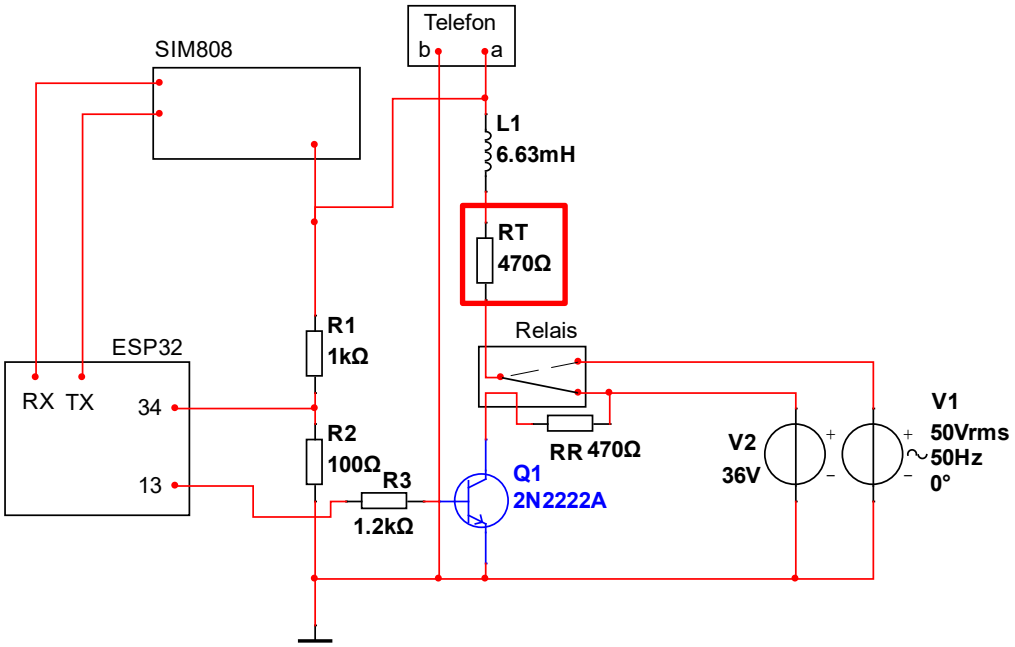

Abbildung 3.38: Versorgungsschaltung mit neuem Widerstand

Nachdem wir dieses Bauteil eingefügt haben (Abbildung 3.38, rot markiert), konnten wir die Wählscheibe beliebig oft betätigen, ohne dass die Spannung einbrach. Der restliche Teil der Schaltung blieb stabil und arbeitete fehlerfrei weiter. Damit war der Test erfolgreich abgeschlossen und wir hatten die Gewissheit, dass das Konzept funktioniert. Durch den Einsatz eines Leistungswiderstandes ist zudem sichergestellt, dass das Bauteil nicht zu heiß wird, falls der Wählvorgang länger dauert oder häufig hintereinander erfolgt.

3.4.3. SIM-Modul Spracheingabe

Nachdem die Spannungsversorgung des Telefons stabilisiert wurde, erfolgte die Anbindung an das SIM-Modul. In diesem ersten Schritt wurde die Übertragung des Sprachsignals implementiert.

Da das Telefonsystem mit einer Spannung von ca. 36V DC arbeitet, der das analoge Sprachsignal überlagert ist, ist eine direkte Verbindung zum SIM-Modul nicht möglich. Um den empfindlichen Mikrofoneingang des Moduls vor der hohen Gleichspannung zu schützen, wird ein Einkoppelkondensator zur DC-Entkopplung verwendet. Dieser wird in Serie in den Signalweg geschaltet und fungiert als Hochpassfilter: Er blockiert die Gleichspannung und lässt lediglich das Wechselspannungssignal der Sprache passieren.

Die Dimensionierung des Kondensators basiert auf dem relevanten Frequenzband der Telefonie, welches sich von 250Hz bis 4000Hz erstreckt. Um niederfrequente Störanteile (wie z.B. Netzbrummen) effektiv zu dämpfen, wurde die untere Grenzfrequenz (f_g) auf 250 Hz festgelegt.

Parameter		Min	Typ	Max	Unit
Mic biasing voltage			1.9	2.2	V
Working Current				2	mA
Input impedance(differential)		13	20	27	KΩ
Idle channel noise				-67	dBm
SINAD	Input level:-40dBm0	29			dB
	Input level:0dBm0		69		dB

Abbildung 3.39: Mikrofoneingang Spezifikationen SIM808-Modul

Gemäß den Spezifikationen des SIM808-Moduls (Abbildung 3.39) weist der Mikrofoneingang eine Eingangsimpedanz von 20kΩ auf. Die benötigte Kapazität lässt sich mit der folgenden Formel berechnen:

$$C = \frac{1}{2 * \pi * f_g * R}$$

Formel 3.4-11: Formel zur Berechnung von C

Durch Einsetzen der Parameter ergibt sich:

$$C = \frac{1}{2 * \pi * 250\text{Hz} * 20000\Omega} = 31.8nF$$

Formel 3.4-12: Berechnung Entkopplungskondensator

Da 31,83nF kein Standardwert in der Fertigung ist, wurde für den Schaltungsaufbau der nächstgelegene verfügbare Standardwert von 33nF gewählt. Diese geringfügige Abweichung

verschiebt die Grenzfrequenz nur minimal und hat keinen negativen Einfluss auf die Sprachqualität.

Das Sprechen über das Telefon hat so einwandfrei funktioniert, und der gegenüber auf dem Mobil-Telefon konnte einen gut verstehen.

3.4.4. Anpassung der Berechnungen

Durch den Einbau des 470-Ohm-Lastwiderstandes haben wir die Stromversorgung stabilisiert, aber gleichzeitig einen neuen Störfaktor in das System gebracht. Nach dem Ohm'schen Gesetz fällt an unserem Widerstand eine gewisse Spannung ab. Das bedeutet, dass am eigentlichen Rest der Schaltung jetzt etwas weniger Spannung ankommt als ursprünglich geplant.

R_1 wird weiterhin mit 1kΩ angenommen, nun müssen wir R_2 mit der neuen Spannung berechnen. Durch Tabelle 3.1 wissen wir, dass unser Telefon im Standby kaum Strom verbraucht. Der ESP32 kann mittels analogem Pin, genau die Spannung zwischen 3,3V und 0V erkennen. Unser Ziel ist es somit, dass die Spannung unter 3,3V abfällt, wenn der Hörer abgehoben wird. Somit würde der EPS ein Abheben erkennen. Beim Aufziehen des Wählrades entsteht ein Kurzschluss, womit 0V am ESP anliegen.

Strom im Standby: 0,3mA
Spannungsabfall:

$$U = R * I = 470 * 0{,}0003 = 0{,}141V$$

Formel 3.4-13: Berechnung des Spannungsabfalles am Vorwiderstand

Spannung am Spannungsteiler: 35,86V

$$R_2 = 1000 * \frac{3{,}3}{35{,}86 - 3{,}3} = 101{,}35\Omega$$

Formel 3.4-14: Berechnungen des Spannungsteilers mit Vorwiderstand

Im Standby-Zustand fließt mit nur 0,3 mA ein sehr geringer Strom durch die Schaltung. Dadurch entsteht am 470-Ohm-Lastwiderstand nur ein kleiner Spannungsabfall von etwa 0,14 V. Diese Spannungsdifferenz ist im Vergleich zur gesamten Versorgungsspannung sehr gering und hat somit kaum Einfluss auf den Spannungsteiler.

Daraus folgt, dass sich die Verhältnisse im Spannungsteiler praktisch nicht verändern. Die berechneten Widerstandswerte, insbesondere R_2, können daher unverändert beibehalten werden, ohne die Funktion der Schaltung zu beeinträchtigen.

3.4.5. Platine des ersten Prototyps

Nach erfolgreichem Testen auf dem Steckbrett, wurde die allererste Platine in Fusion entworfen. Vorerst wurde der Schaltplan gezeichnet und anschließend wurde die Platine designed.

In den ersten Prototyp wurde auch ein OLED-Display eingebaut (Abbildung 3.40, rot markiert). Das Display dient ausschließlich als Anzeige der gewählten und eingehenden Rufnummern. Zusätzlich werden auch Statusinformationen angezeigt (siehe Kapitel 4.3.1.15). Das OLED-Display wird einfach über 3,3V durch den ESP32 versorgt.

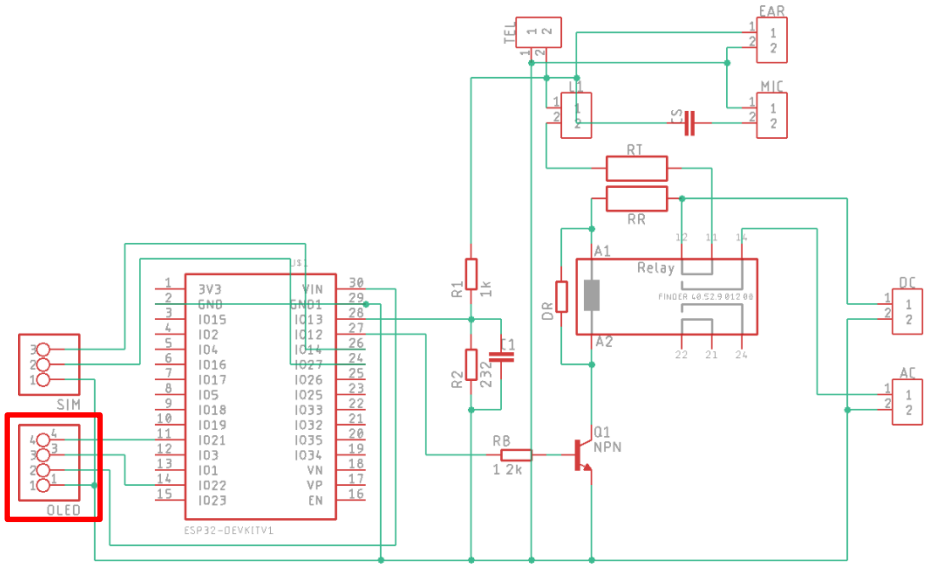

Abbildung 3.40: Schaltplan Prototyp 1

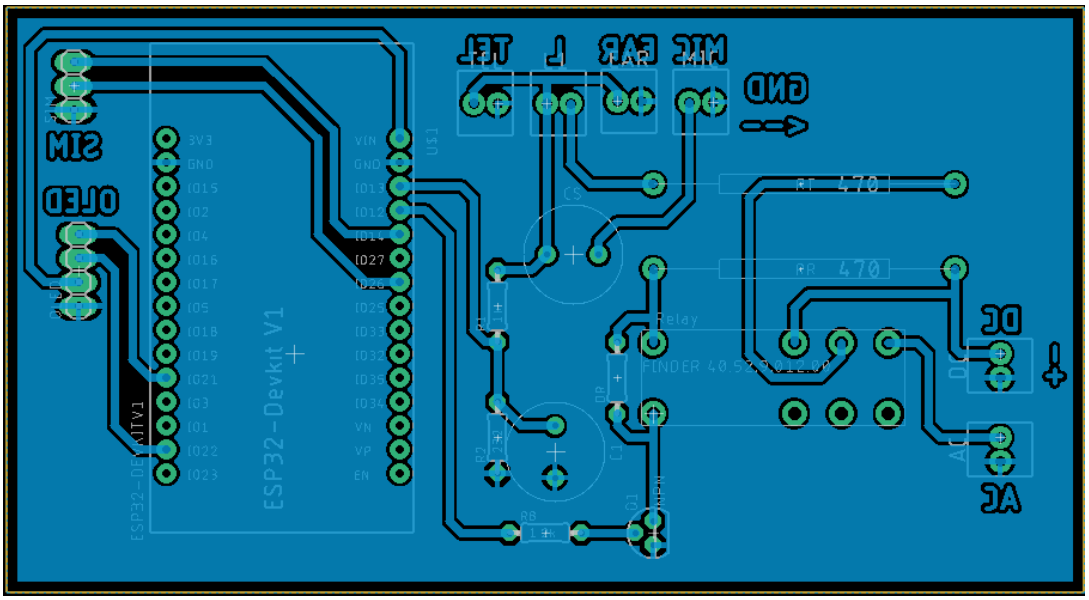

Abbildung 3.41: Platine Prototyp 1

Nachdem alle Berechnungen abgeschlossen waren und die Schaltung auf dem Steckbrett funktionierte, haben wir das gesamte Modul fest auf einer Platine aufgebaut. Dabei wurden

alle Bauteile ordnungsgemäß eingelötet, um eine dauerhafte und stabile Verbindung zu garantieren und Fehler durch Wackelkontakte auszuschließen.

Nach dem Zusammenbau haben wir das Modul einem umfassenden Funktionstest unterzogen. Dabei wurde geprüft, ob die Rufspannung das Telefon zum Klingeln bringt und ob die Wahlimpulse beim Wählen einer Nummer korrekt verarbeitet werden.

Das Ergebnis: Die Schaltung hat einwandfrei funktioniert. Alle Tests verliefen erfolgreich, und das Modul arbeitet genauso, wie wir es in der Planung vorgesehen haben. Für das finale Produkt ist vorgesehen, das Modul unabhängig von externen Netzgeräten zu betreiben. Ziel ist eine integrierte Lösung, bei der lediglich ein Kaltgerätekabel für die direkte Stromversorgung aus der Steckdose benötigt wird. Dies bildet die Grundlage für die Entwicklung des zweiten Prototyps und der finalen Platine.

3.5. Zweiter/finaler Prototyp

Ein wesentlicher Bestandteil für die finale Spannungsversorgung war der Transformator mit einer Ausgangsspannung von 30V eff. Nach Erhalt dieses Bauteils konnte die Schaltung wie geplant vervollständigt werden. Durch die Integration des Transformators ist es nun möglich, die benötigte Rufspannung intern zu generieren, wodurch das System die angestrebte Unabhängigkeit von externen Wechselspannungsquellen erreicht.

Nach dem Testen des ersten Prototyps stellte sich ein weiteres Problem heraus. Wenn das Telefon läutete, schaltete das Relais um und es lag Wechselspannung an. Diese Wechselspannung lag allerdings auch an unserem ESP32 an. Wenn wir jetzt wieder die Formel für die Spitzenspannung einer Wechselspannung hernahmen: $U = U_{eff} * \sqrt{2} = 30 * \sqrt{2} = 42{,}43V$

Bei unserem Spannungsteiler haben wir aber mit den 36V des Netzteiles gerechnet. Somit liegt am ESP32, wenn das Telefon klingelt, zu viel Spannung an. Dadurch dass es aber nicht dauerhaft läutet, ist dies nicht direkt schädlich für den ESP32. Das Problem ist, dass durch das Anheben des Hörers (abheben) die Spannung für den ESP32 noch immer zu hoch ist.

3.5.1. Berechnungen Prototyp 2

Wenn das Telefon läutet, liegt eine Spannung von insgesamt 42,43V am Spannungsteiler an. Dadurch, dass es Wechselspannung ist, müssen wir eine Diode einbauen, um die negativen Halbwellen zu filtern.

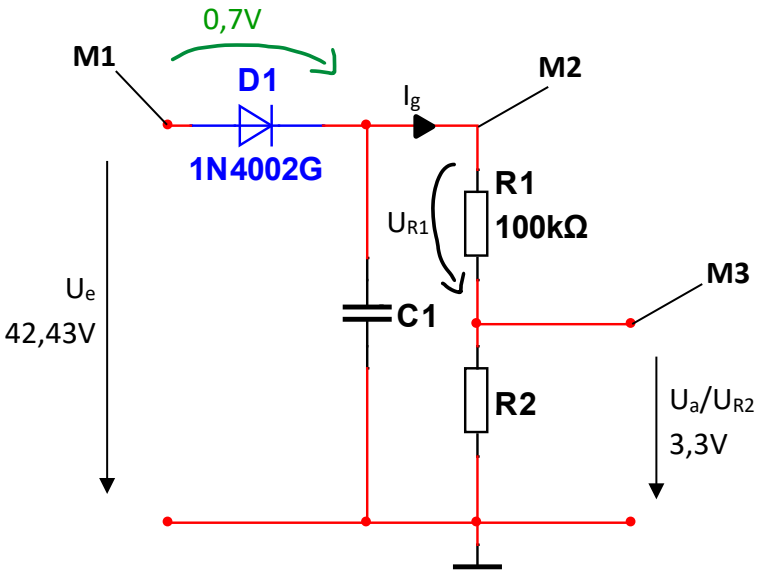

Abbildung 3.42: Aufbau Spannungsteiler

An der Diode fallen ~0,7V ab. Somit liegen nach der Diode 41,73V an. Nicht nur die Spannung hat sich geändert, sondern auch das Signal. Durch die Diode wird die negative Halbwelle blockiert, und es ist nur noch die positive Halbwelle zu sehen. Durch den Kondensator wird das Signal auch noch zusätzlich geglättet.

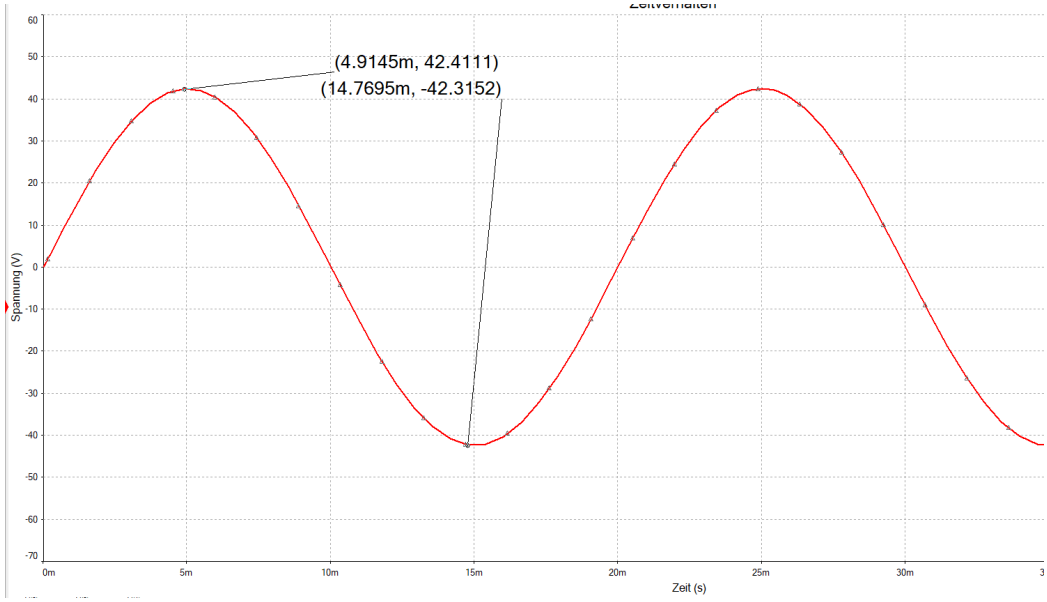

Abbildung 3.43: Signal an Messpunkt 1 (M1)

In Abbildung 3.43 ist das Signal an Messpunkt M1 aus der Schaltung in Abbildung 3.42 zu sehen. Das ist genau die Spannung, die am Telefon anliegt, wenn dieses läutet.

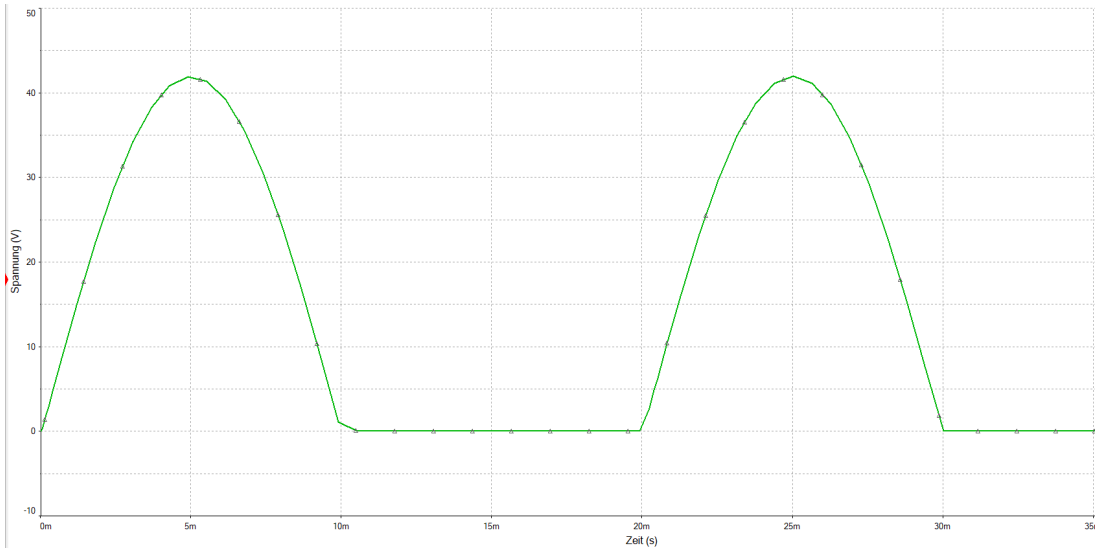

Abbildung 3.44: Signal nach der Diode ohne Kondensator C1

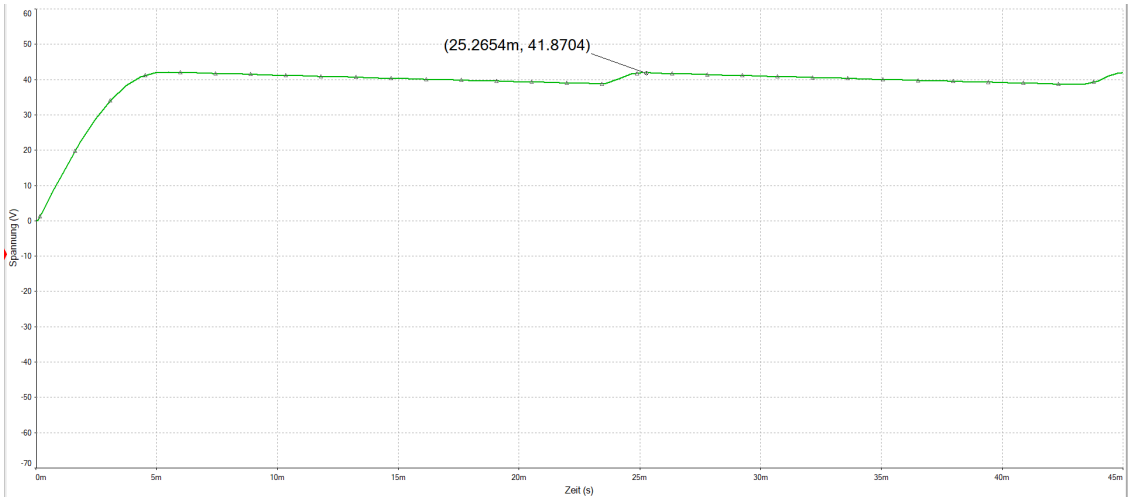

Abbildung 3.45: Signal an Messpunkt 2 (M2)

In Abbildung 3.45 ist das Signal an Messpunkt M2 aus der Schaltung in Abbildung 3.42 zu sehen. Hier ist schön zu sehen, wie die Diode die negativen Halbwellen blockiert. Hier ist auch die Wirkung des Kondensators C1 sehr gut erkennbar. Er glättet das Signal, somit entsteht eine beinahe perfekte Gleichspannung.

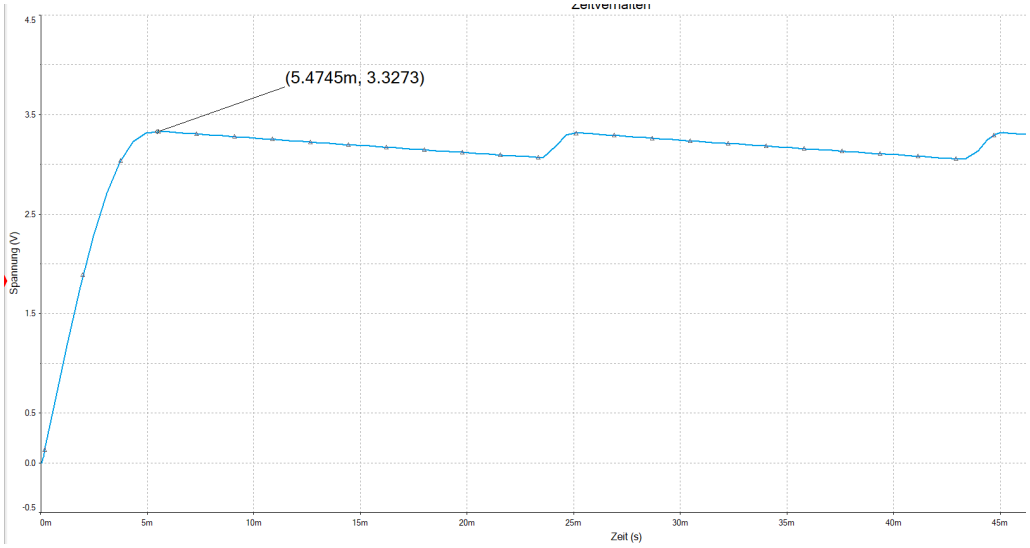

Abbildung 3.46: Signal an Messpunkt 3 (M3)

Messpunkt M3 ist das Signal, das endgültig an unserem ESP32 ankommt.

Berechnungen:

Gesucht: I_g, R_2 und C_1

Gegeben: U_e, U_a und R_1

$$I_g = \frac{U}{R} = \frac{U_{R1}}{R_1} = \frac{41{,}73 - 3{,}3}{100\,000} = 0{,}384mA$$

$$R_2 = \frac{U_{R2}}{I_g} = \frac{3{,}3}{0{,}384m} = 8{,}594k\Omega$$

Formel 3.5-1: Berechnungen des Spannungsteilers mit Kondensator

Um sicherzustellen, dass unser Signal gut geglättet wird, muss der Kondensator groß genug sein. Mit der folgenden Formel wird berechnet, welche Mindestkapazität dieser Kondensator haben muss, damit die Spannung nicht zu stark schwankt:

$$C_{min} = I_g * \frac{\Delta t}{\Delta U}$$

Formel 3.5-2: Formel zur Berechnung des Kondensators

Δt Zeit, in der der Kondensator entladen wird.

ΔU Maximale Spannung, die am Kondensator theoretisch abfallen darf, auch Brummspannung genannt

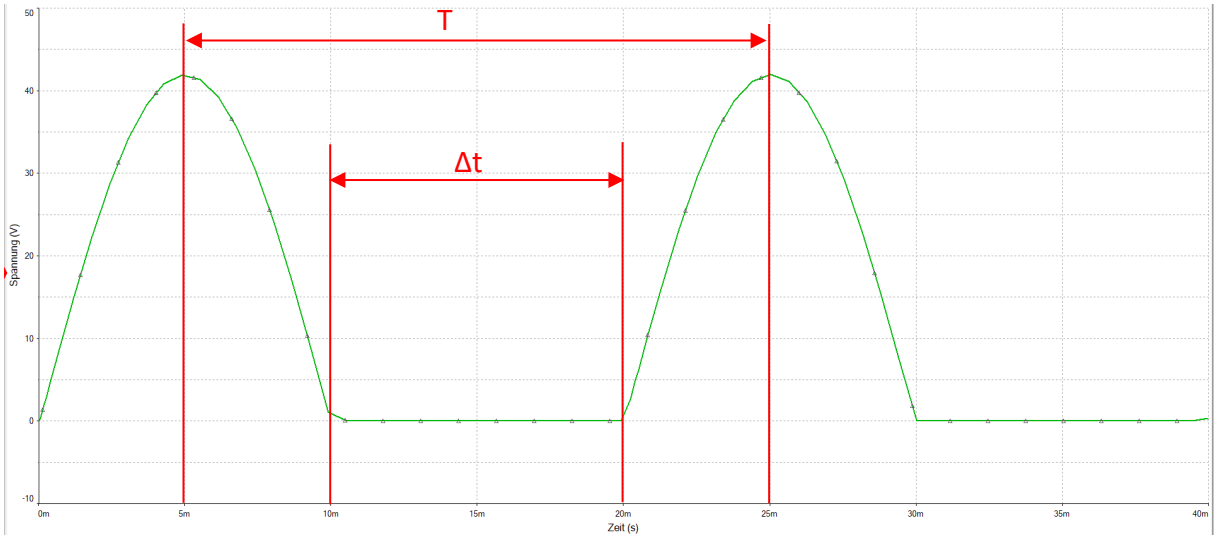

Abbildung 3.47: Beispiel Periodendauer T und Δt

Δt kann allgemein mit folgender Formel berechnet werden:

$$\Delta t = \frac{\frac{1}{f}}{2}$$

Formel 3.5-3: Formel zur Berechnung von Δt

Bei einer Netzfrequenz von 50Hz ergibt sich:

$$\Delta t = \frac{\frac{1}{50Hz}}{2} = \frac{20ms}{2} = 10ms$$

Formel 3.5-4: Berechnung von Δt

Die Formel für C_{min} wird üblicherweise für Gleichrichterschaltungen herangezogen. In unserem Fall wurde jedoch lediglich eine Einweggleichrichtung mit einer Diode verwendet, weshalb sich ein größeres Entladeintervall (Δt = T) ergibt und dieses entsprechend in der Berechnung berücksichtigt werden muss.

Δt = T = 20ms
ΔU = 10% von 41,73V = 4,173

Wenn wir jetzt alle Werte in unsere C_{min}-Formel einsetzen, erhalten wir folgendes Ergebnis:

$$C_{min} = 0{,}384mA * \frac{20ms}{4{,}173} = 1{,}84\mu F$$

Formel 3.5-5: Berechnung von C_{min}

Da dieser Wert nur ein Minimalwert ist, kann der Kondensator auch größer gewählt werden. Wenn der Kondensator allerdings zu groß ist, dann braucht er zu lange, um sich zu entladen,

wenn der Hörer abgenommen wird. Daraus folgt, dass der ESP32 den Spannungsabfall erst spät erkennt, wodurch nicht direkt abgehoben wird. Ein verzögertes Erkennen des Spannungsabfalles ist für den Nutzer sehr unerfreulich.

Der nächste Kondensatorwert, der zur Verfügung steht, ist ein 2,2µF Kondensator. Für diesen wurde sich dann auch endgültig entschieden.

C_1 = 1,84µF → 2,2µF
R_2 = 8,594kΩ → 8,2kΩ + 330Ω

3.5.2. Aufbau finaler Prototyp

Nachdem nun auch der Spannungsteiler der richtigen Spannung angepasst wurde, können wir mit dem finalen Schaltplan und der finalen Platine beginnen.

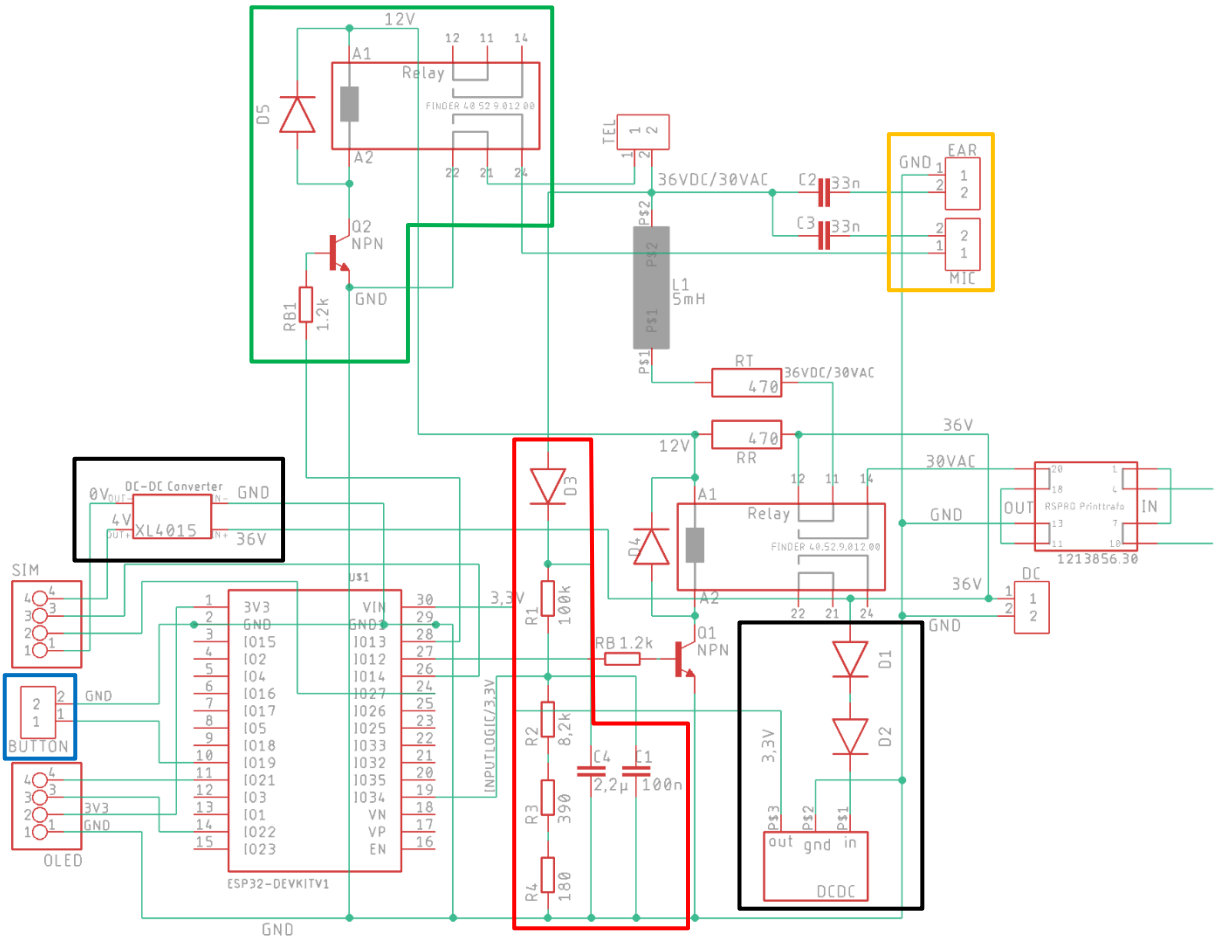

Abbildung 3.48: Schaltplan Prototyp 2

In Abbildung 3.48 ist der finale Schaltplan der Diplomarbeit POTSConnect zu sehen. In roter Umrandung ist unser vorhin berechneter Spannungsteiler zu sehen. In blauer Umrandung ist der Button, der am Ende noch hinzugefügt wurde, erkennbar. Der Button dient zur Speicherung von Telefonnummern und auch um das Favoritenmenü aufzurufen (siehe Kapitel 4.3.1.8). Grün umrandet ist unser zweites Relais mit Low-Side-Switch als Ansteuerung.

3.5.2.1. Relais Spracheingabe

Die b-Ader des Telefons ist standardmäßig mit der Masse der Gesamtschaltung verbunden. Für die Dauer eines Gesprächs muss diese Verbindung jedoch getrennt und die Ader auf den Ausgang des SIM-Moduls geschaltet werden. Zur Umsetzung dient ein Relais. Da das gleiche Relais, wie zuvor in Kombination mit demselben NPN-Transistor verwendet wird, entfällt eine Neuberechnung des Low-Side-Switches. Der Basiswiderstand von R_B = 1,2kΩ wird unverändert übernommen.

Um das analoge Telefon an die getrennten Audio-Anschlüsse des SIM-Moduls anzupassen, teilt die Schaltung die Signale auf: Die a-Ader wird genutzt, um das Sprachsignal des Telefons an den Mikrofon-Eingang (Abbildung 3.48 EAR, orange) des SIM-Moduls zu leiten. Die b-Ader hingegen empfängt das Signal aus dem Lautsprecher-Ausgang (Abbildung 3.48 MIC, orange) des SIM-Moduls, sobald das Relais schaltet.

Wichtig ist dabei die Entkopplung durch die beiden Kondensatoren (C2 und C3). Sie filtern die hohe Gleichspannung (36 V) der Telefonleitung heraus und lassen nur die reinen Audiosignale passieren, sodass das SIM-Modul störungsfrei und sicher arbeiten kann.

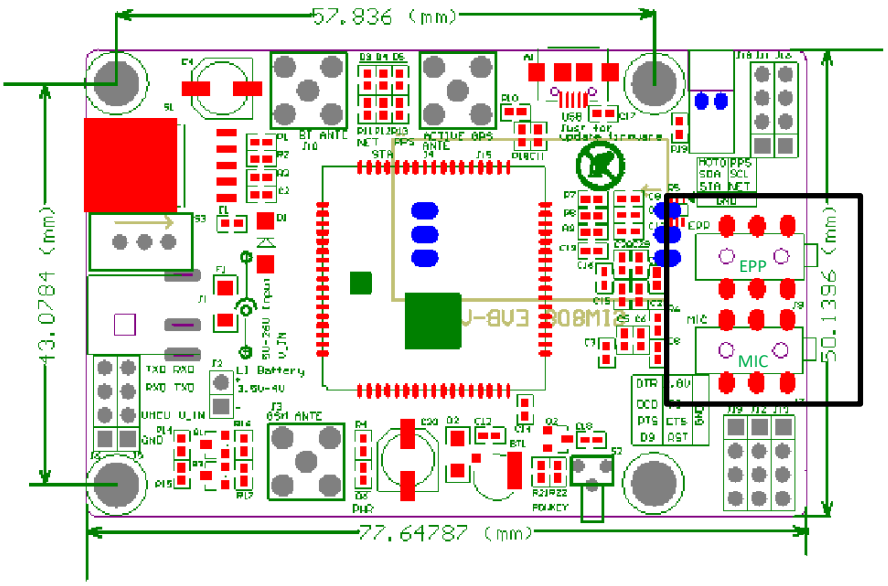

Abbildung 3.49: SIM808-Modul Aufbau

In Abbildung 3.49 ist der Aufbau des SIM808-Moduls dargestellt. Die schwarz umrandeten Bereiche kennzeichnen die Audio-Ein- und Ausgänge. Die Verbindung erfolgt wie folgt: Der MIC-Anschluss der Platine wird mit dem EPP-Anschluss des SIM-Moduls verbunden. Der EAR-Anschluss der Platine ist mit dem MIC-Anschluss des SIM-Moduls verbunden.

3.5.2.2. ESP32-SIM-Modul Versorgung

Bis jetzt wurden der ESP32 und das SIM-Modul über externe Power-Supplys versorgt. Wir haben uns für einfache DC-DC Konverter entschieden (Abbildung 3.48, schwarz markiert), da das Modul unabhängig von externen Netzteilen betrieben werden soll. Für den ESP32 kommt der Tarco TSR 1-2433 zum Einsatz, für das SIM-Modul der XL4015. So ist eine stabile und autarke Stromversorgung direkt auf der Platine gewährleistet.

Der Tarco TSR 1-2433 erfüllt unsere Anforderungen optimal. Er unterstützt eine maximale Eingangsspannung von 36 V, was genau unserer Systemspannung entspricht. Die Ausgangsspannung beträgt exakt 3,3 V, bei einer Effizienz von 91 % – gemessen bei einer Eingangsspannung von 4,75 V. Für den ESP32 reicht es auch, dass der DC-DC einen maximalen Ausgangsstrom von 1A hat.

Models

Order Code	Output Current max.	Input Voltage Range	Output Voltage nom.	Efficiency typ.
TSR 1-2412		4.6 - 36 VDC (9 VDC nom.)	1.2 VDC	74 % (at Vin min.)
TSR 1-2415			1.5 VDC	78 % (at Vin min.)
TSR 1-2418			1.8 VDC	82 % (at Vin min.)
TSR 1-2425			2.5 VDC	87 % (at Vin min.)
TSR 1-2433	1'000 mA	4.75 - 36 VDC (9 VDC nom.)	3.3 VDC	91 % (at Vin min.)
TSR 1-2450		6.5 - 36 VDC (12 VDC nom.)	5 VDC	94 % (at Vin min.)
TSR 1-2465		9 - 36 VDC (12 VDC nom.)	6.5 VDC	93 % (at Vin min.)
TSR 1-2490		12 - 36 VDC (24 VDC nom.)	9 VDC	95 % (at Vin min.)
TSR 1-24120		15 - 36 VDC (24 VDC nom.)	12 VDC	95 % (at Vin min.)
TSR 1-24150		18 - 36 VDC (24 VDC nom.)	15 VDC	96 % (at Vin min.)

Abbildung 3.50: Ratings Tarco TSR 1-2433

Zur Sicherheit wurden noch zwei Dioden in Serie mit dem DC-DC Konverters geschaltet. Durch den Spannungsabfall von jeweils etwa 0,7V an den Dioden, reduziert sich die effektive Eingangsspannung auf ca. 34,6V

Das SIM-Modul benötigt eine Eingangsspannung von 4 V und einen Strom von bis zu 2 A. Der XL4015 deckt diese Anforderungen problemlos ab und sorgt für eine stabile sowie zuverlässige Versorgung des Moduls.

Absolute Maximum Ratings（Note1）

Parameter	Symbol	Value	Unit
Input Voltage	Vin	-0.3 to 40	V
Feedback Pin Voltage	V_{FB}	-0.3 to Vin	V
Output Switch Pin Voltage	V_{Output}	-0.3 to Vin	V
Power Dissipation	P_D	Internally limited	mW
Thermal Resistance (TO263-5L) (Junction to Ambient, No Heatsink, Free Air)	R_{JA}	30	°C/W
Operating Junction Temperature	T_J	-40 to 125	°C
Storage Temperature	T_{STG}	-65 to 150	°C
Lead Temperature (Soldering, 10 sec)	T_{LEAD}	260	°C
ESD (HBM)		>2000	V

Abbildung 3.51: Maximum Ratings XL4015

Um einen Ausgangsstrom von 2 A zu erreichen, muss der erforderliche Eingangsstrom bestimmt werden. Zur Berechnung kann folgende Formel verwendet werden:

$$I_{in} = \frac{U_{out} * I_{out}}{\eta * U_{in}}$$

Formel 3.5-6: Formel zur Berechnung des Eingangsstromes

I_{in} Strom, den der DC-DC-Wandler vom Eingang zieht
U_{out} Ausgangsspannung
I_{out} gewünschter Ausgangsstrom
η Wirkungsgrad
U_{in} Eingangsspannung

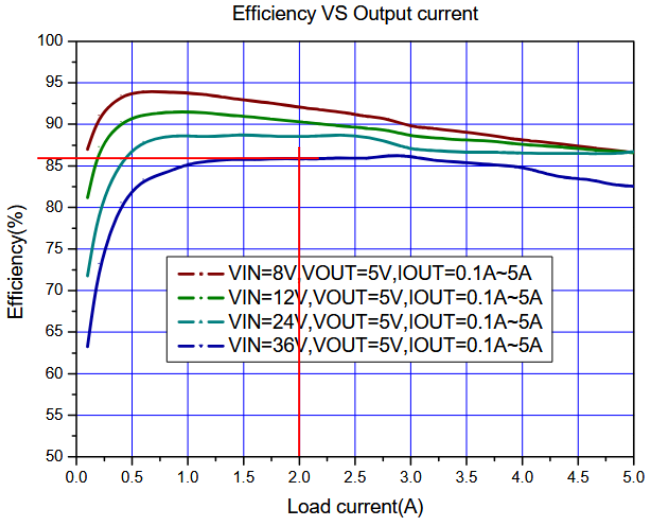

Abbildung 3.52: XL4015 Wirkungsgrad Kurve

Aus Abbildung 3.52 können wir unser η ablesen. Bei einer Eingangsspannung von 36V und einem Ausgangsstrom von 2A beträgt der Wirkungsgrad etwa 0,86. Die Werte können nun in die Formel eingesetzt werden:

$$I_{in} = \frac{4 * 2}{0{,}86 * 36} = 0{,}258A = 258mA$$

Formel 3.5-7: Berechnung von I_{in}

Das Ergebnis zeigt, dass bei einer Eingangsspannung von 36V und einem Wirkungsgrad von 86% ein Eingangsstrom von ca. 0,26 A ausreicht, um am Ausgang die geforderten 2 A bei 4 V bereitzustellen. Dies bestätigt, dass der eingesetzte XL4015 Wandler die Stromversorgung des SIM-Moduls zuverlässig übernehmen kann, ohne dass das Eingangssystem überlastet wird.

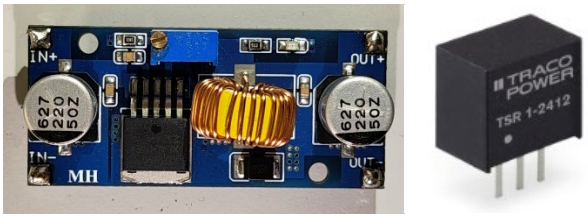

Abbildung 3.53: XL4015 und Tarco TSR 1-2433

3.5.2.3. Platine

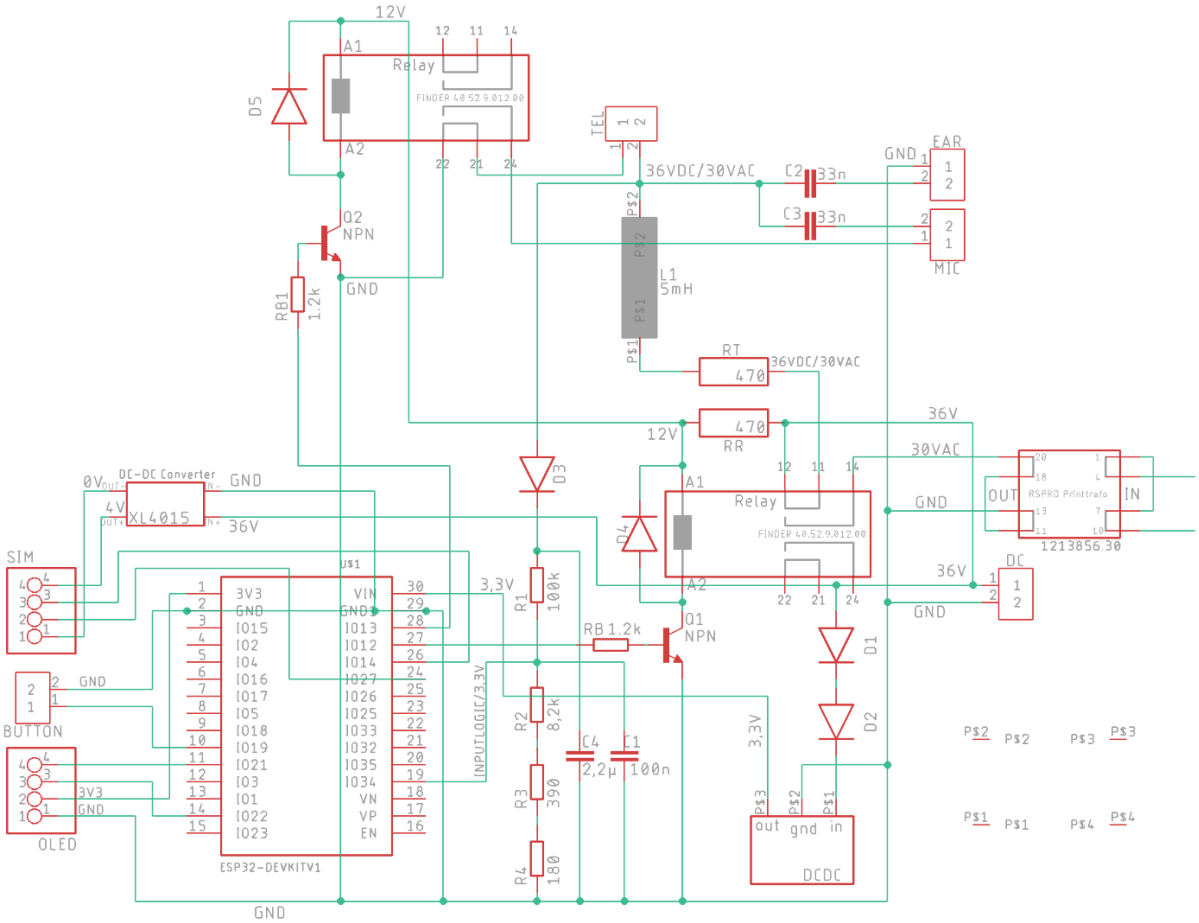

Abbildung 3.54: Schaltplan POTSConnect

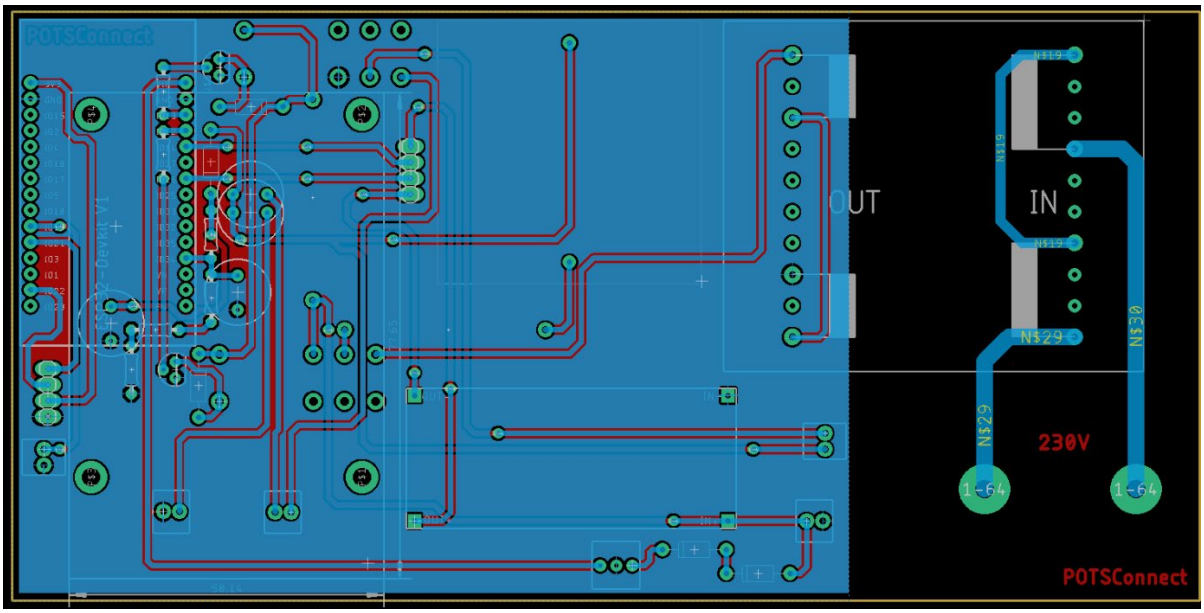

Abbildung 3.55: Platine POTSConnect Bottom-View

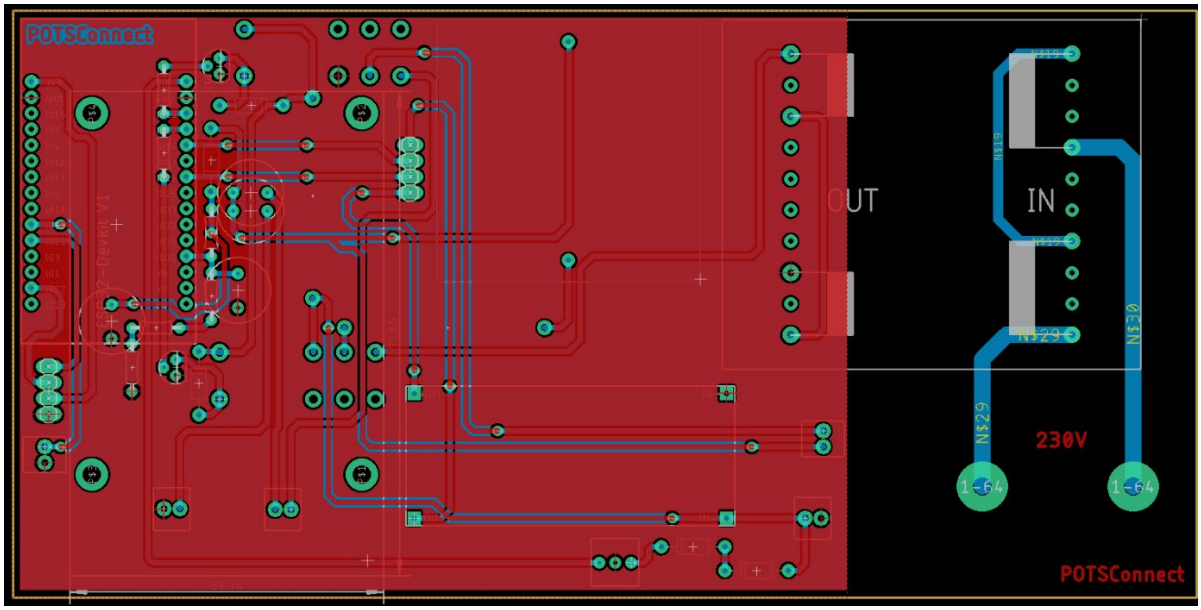

Abbildung 3.56: Platine POTSConnect Top-View

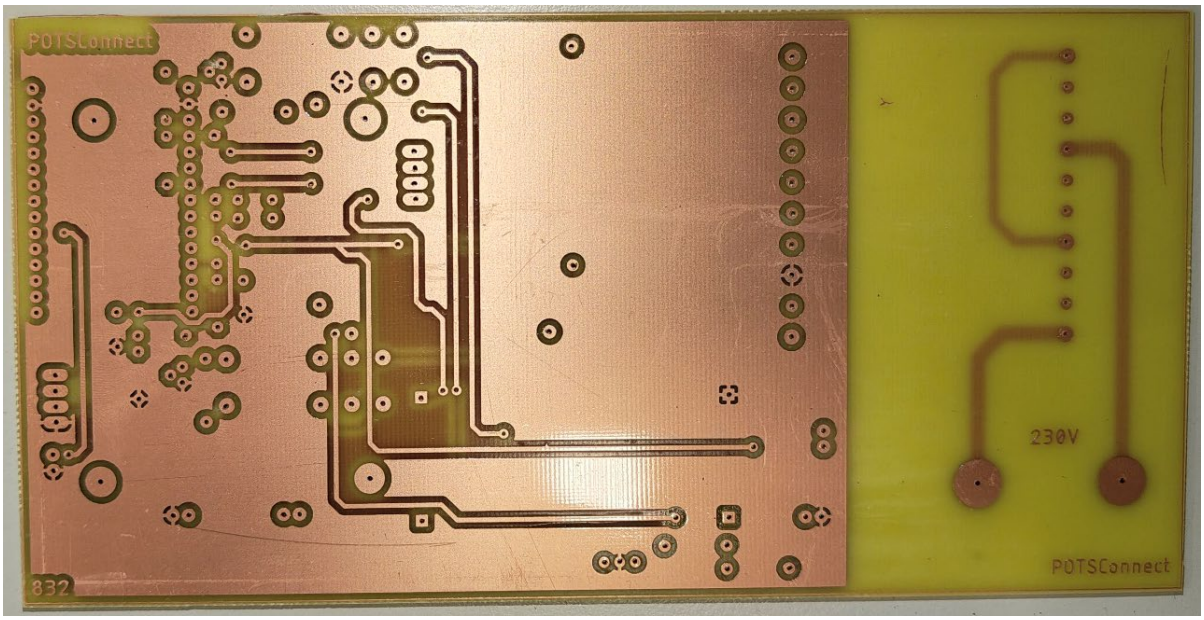

Abbildung 3.57: Angefertigte Platine

Durch die gewählte Anordnung der Bauteile konnten alle wichtigen Module – wie der ESP32, das SIM808-Modul und die DC-DC-Konverter – kompakt und funktional integriert werden. Besonderes Augenmerk lag auf der Trennung von digitalen und analogen Signalen, um Störungen zu minimieren und eine stabile Stromversorgung zu gewährleisten.

Die fertige Platine bildet somit die Grundlage für den Prototypenbetrieb und ermöglicht einen autarken Betrieb ohne externe Netzteile, während alle Verbindungen und Schnittstellen klar strukturiert umgesetzt wurden.

Aufgrund des engen Zeitplans konnte für das Gehäuse keine Umsetzung mehr erfolgen. Daher sind im nachfolgenden Bild die Komponenten provisorisch auf einer Matte angeordnet, um den Aufbau und die Funktionsweise des Systems zu veranschaulichen.

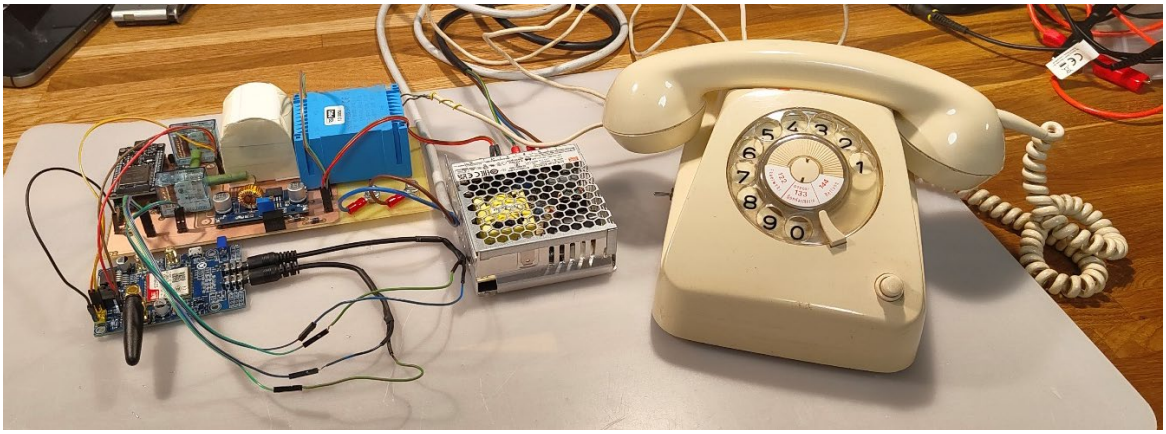

Abbildung 3.58: Aufbau POTSConnect

4. Analyse-Mikrokontroller, Sim-Module, Mobilfunksysteme (Thomas Niederkofler)

Im Rahmen des Projekts POTSConnect wurde analysiert, wie ein klassisches analoges Wählscheibentelefon in ein digitales Kommunikationsgerät überführt werden kann. Da ein solches Telefon ursprünglich auf mechanischen Impulswahlsystemen basiert, ist zur Digitalisierung eine Zusatzhardware notwendig. Diese übernimmt sowohl die Signalaufnahme als auch deren Verarbeitung und Weiterleitung über moderne Mobilfunknetze.

Um eine zuverlässige und alltagstaugliche Umsetzung zu ermöglichen, wurden unterschiedliche Mikrocontroller, SIM-Module sowie verfügbare Mobilfunktechnologien vergleichend untersucht. Die folgenden Abschnitte beschreiben die Kriterien, Parameter und Überlegungen, die zur finalen Systemauswahl geführt haben.

4.1.1. Übersicht verwendbarer Mikrocontrollersysteme

Für die Umsetzung der Signalerfassung, Datenverarbeitung und Kommunikation wurde eine Reihe gängiger Mikrocontroller verglichen. Die Bewertung erfolgte anhand folgender Kriterien:

- **Rechenleistung und Architektur**
  Die Auswertung von Wählimpulsen erfordert präzise Zeitmessungen und eine gewisse Rechenleistung. Ein moderner Controller mit effizienten Timerfunktionen sowie paralleler Aufgabenverarbeitung ist hier von Vorteil.

- **Stromverbrauch**
  Ein zu hoher Energiebedarf würde den praktischen Einsatz erschweren, insbesondere wenn spätere Varianten als Low-Power-System vorgesehen sind.

- **Kommunikationsschnittstellen**
  Für die Anbindung eines SIM-Moduls sind UART-Schnittstellen essenziell. Zusätzliche Bussysteme wie I2C und SPI ermöglichen spätere Modul- oder Sensorerweiterungen.

- **Preis und Verfügbarkeit**
  Da das Projekt auch für den schulischen sowie prototypischen Bereich relevant ist, müssen die verwendeten Komponenten kostengünstig und gut verfügbar sein.

- **Unterstützung durch Community und Software**
  Eine große Nutzerbasis erleichtert Entwicklungsarbeit erheblich, insbesondere wenn Bibliotheken für GSM-Module oder WLAN-Funktionen bereits existieren.

Die betrachteten Mikrocontroller sind in Tabelle 1 zusammengefasst.

µC	Chip	Architektur	Spannung	Strom	Preis	Kommunikation	Anwendung
ESP32	Dual-Core 32bit Xtensa	RISC	3.3V	260mA	4–10€	UART, I2C, SPI, Wi-Fi, BT	ideal für SIM-Integration [5]
Arduino Uno	8bit ATmega328P	RISC	5V	40mA	20€	UART, I2C, SPI	einfach, aber limitiert
Arduino Nano	8bit ATmega328P	RISC	5V	20mA	10€	UART, I2C, SPI	Kompakt, eingeschränkt
Raspberry Pi	64bit ARM Cortex-A53+	CISC	5V (USB)	900mA	35–60€	UART, I2C, SPI, Wi-Fi, Ethernet, BT	Webserver möglich [6]

Tabelle 4.1: Mikrokontroller im Vergleich

4.1.1.1. Vergleich und Bewertung der Systeme

Arduino Uno / Arduino Nano

Diese beiden Systeme verwenden den 8-Bit-Mikrocontroller ATmega328P. Er ist sehr gut dokumentiert und für einfache Aufgaben hervorragend geeignet, stößt aber bei der Verarbeitung paralleler Prozesse schnell an seine Grenzen. Die geringe Rechenleistung erschwert insbesondere die gleichzeitige Auswertung der Impulswahl und die Steuerung eines SIM-Moduls. Zudem besitzen sie keine integrierten Funktechnologien, wodurch externe Module notwendig werden [7].

Raspberry Pi

Mit seinem 64-Bit ARM-Prozessor bietet der Raspberry Pi eine beträchtliche Rechenleistung. Für die aufgebürdeten Aufgaben ist er jedoch überdimensioniert. Stromverbrauch (600–900 mA), Preis und der deutlich komplexere Aufbau sprechen gegen seinen Einsatz. Der Pi wäre eher geeignet, wenn serverseitige Anwendungen oder datenintensive Aufgaben geplant wären [6] .

4.1.1.2. Begründung für den ESP32

Der ESP32 erwies sich als optimaler Kompromiss zwischen Funktionsumfang, Leistung und Kosten:

- Dual-Core-Architektur mit 32-Bit-Xtensa-Kernen
- vielseitige Kommunikationsschnittstellen (WLAN, Bluetooth, UART, I2C, SPI)
- besonders günstiger Preis (4–10 €)
- gute Timer-Hardware zur Erfassung der Impulswahl
- große Community und exzellente Softwareunterstützung
- stabiler Betrieb bei niedrigem Stromverbrauch

Insbesondere die Möglichkeit, parallel laufende Aufgaben auf mehrere Prozessorkerne zu verteilen, ist ein deutlicher Vorteil gegenüber 8-Bit-Systemen. Dadurch wird eine präzise und störungsarme Auswertung des Wählscheibenmechanismus möglich.
Auf Grundlage dieser Merkmale wurde der ESP32 als Mikrocontroller für das Projekt ausgewählt [5].

4.1.2. Übersicht verwendbarer SIM-Module

Für die Mobilfunkanbindung des Systems wurden mehrere SIM- und GSM-Module untersucht. Im Fokus standen:

- Stromaufnahme in Sendephasen
- Unterstützung verschiedener Mobilfunkstandards (2G/3G/4G/NB-IoT)
- Preis
- Kompatibilität mit dem ESP32 (v. a. über UART)
- Funktionsumfang (Telefonie, SMS, GPS, Datenübertragung)

Die analysierten SIM-Module sind in Tabelle 2 dargestellt.

SIM-Modul	Mobilfunk-standarts	Span-nung	Strom	Preis	Kommu-nikation	Anwendungsfälle
SIM808	GSM (2G), GPRS, GPS	4.4V	2A	5–25€	UART, GPS	einfach, Anruf/SMS, GPS [8] [9]
SIM800L	GSM	4.2V	2A	5–10€	UART	nur SMS/Anrufe
SIM7000	GSM, LTE Cat-M/NB-IoT	4.2V	1.6A	15–75€	UART	LTE-fähig, perfekt für Raspi [10]
A7670E	LTE (4G)	4.2V	2A	15–30€	UART, USB	aufwendig für ESP32

Tabelle 4.2: Sim-Module im Vergleich

4.1.2.1. Analyse der einzelnen Module

SIM800L

Günstiges und kompaktes 2G-Modul. Es eignet sich für einfache Telefon- und SMS-Funktionen, bietet aber keinen GPS-Empfänger. Für IoT-Projekte wird es häufig verwendet, ist jedoch funktional begrenzt. [8] [9]

SIM7000-Serie

Unterstützt moderne Übertragungsstandards wie LTE Cat-M oder NB-IoT. Diese Technologien

sind energieeffizient, aber nicht flächendeckend verfügbar. Für den Zweck des Projekts – einfache Mobiltelefonie – wäre das Modul technisch überdimensioniert und deutlich teurer. [10]

A7670E
Ein modernes LTE-Modul, das umfangreiche Funktionen bietet, allerdings zu einem höheren Preis und mit komplexerer Integration. Zudem wäre für VoLTE-Telefonie zusätzliche Softwarekonfiguration notwendig.

4.1.2.2. Begründung für das SIM808

Das **SIM808-Modul** vereint mehrere Vorteile:

- zuverlässige Unterstützung von GSM/GPRS (2G)
- integrierter GPS-Empfänger
- robuste UART-Schnittstelle
- niedriger Preis
- große Menge an dokumentierten Beispielen und Bibliotheken
- einfache Integration mit dem ESP32

Die Stromaufnahme ist ähnlich wie bei anderen GSM-Modulen, jedoch stellt das SIM808 ein besonders gutes Verhältnis aus Preis, Funktionsumfang und Komplexität dar. Aus diesen Gründen wurde es als Mobilfunkmodem für POTSConnect ausgewählt [8] [9].

Abbildung 4.1: Sim808-Modul

Das Sim-Modul umfasst folgende Komponenten:

- SIM-Karten-Slot
- GSM-Antenne
- GPS-Antenne-wird in dem Projekt nicht unbedingt benötigt
- Status-LEDs
- Serielle Schnittstelle (UART)
- Klinkenanschluss für Mikrofon und Audioausgabe [11]

4.1.3. Übersicht von Mobilfunksystemen

Da das verwendete SIM-Modul direkt von den unterstützten Mobilfunkstandards abhängig ist, war eine Analyse der gegenwärtigen Netztechnologien notwendig. Die Bewertung orientierte sich an Reichweite, Energieeffizienz, Latenz, Geschwindigkeit und Zukunftssicherheit.

Tabelle 3 zeigt die relevanten Parameter.

System	Band-breite	Latenz	Energiever-brauch	Abdeckung	Status
GSM (2G)	0.1 Mbit/s	hoch	gering	hoch	Immer noch aktuell (Notruf)
UMTS (3G)	2 Mbit/s	mittel	mittel	abnehmend	abgeschaltet
LTE (4G)	150 Mbit/s	gering	mittel	sehr gut	aktuell
5G	10 Gbit/s	sehr gering	hoch	im Ausbau	noch nicht notwendig

Tabelle 4.3: Mobilfunknetze im Vergleich

4.1.3.1. Analyse der einzelnen Standards

- **GSM (2G)**
  Obwohl technisch über 30 Jahre alt, bietet GSM eine sehr hohe Netzabdeckung und wird nach wie vor für sicherheitsrelevante Systeme genutzt. Durch die geringe Datenrate ist es ideal für SMS, Anrufe und einfache IoT-Anwendungen geeignet.
- **UMTS (3G)**
  3G wurde in vielen Ländern vollständig abgeschaltet, auch in Österreich-nicht sinnvoll
- **LTE (4G)**
  4G ist aktuell der wichtigste Mobilfunkstandard. Für reine Telefonie ist jedoch VoLTE erforderlich, was nur von speziellen Modulen unterstützt wird. Für POTSConnect wäre LTE nur relevant, wenn später datenintensive Funktionen integriert würden.
- **5G**
  Extrem leistungsfähig, aber sowohl preislich als auch technisch überdimensioniert. Für Telefonie-basiertes IoT ist 5G weder notwendig noch wirtschaftlich sinnvoll.

4.1.3.2. Begründung für GSM

Da das Projekt im Kern einfache Telefonie umsetzt, reicht 2G vollkommen aus. Zudem wird 2G viele Jahre bestehen bleiben, da:

- große Teile der öffentlichen Infrastruktur weiterhin darauf basieren

- Notrufsysteme teilweise 2G nutzen
- 2G von IoT-Anbietern weiterhin verlangt wird

Daher wurde **GSM (2G)** als Mobilfunknetz für POTSConnect ausgewählt [12].

4.1.4. Fazit: Systemauswahl für POTSConnect

Die technische Analyse zeigt, dass die Kombination aus **ESP32**, **SIM808-Modul** und **GSM-Netz** die sinnvollste Lösung darstellt. Diese Systemkonfiguration ermöglicht:

- präzise Erfassung der Impulswahl
- stabile digitale Weiterleitung über Mobilfunk
- kostengünstige und energieeffiziente Realisierung
- einfache Integration durch vorhandene Bibliotheken
- potenzielle Erweiterungen über WLAN, Bluetooth oder GPS

Die Entscheidung basiert auf den untersuchten Parametern und erfüllt die Anforderungen des Projekts optimal. Langfristig ist ein Umstieg auf 4G- oder 5G-basierte Module problemlos möglich, ohne das Grunddesign des Systems verändern zu müssen.

Komponente	Verwendet im Projekt	Anmerkung / Begründung
µC	ESP32	Preis/Leistung, UART, energiesparend [5]
SIM-Modul	SIM808	Kombiniert GSM/GPRS/GPS [8] [9]
Mobilfunknetz	GSM(2G)	Derzeit aktuell, durch Notruf noch länger aktiv

Tabelle 4.4: Ausgewählte Systeme für das Projekt

4.2. Sim-Test

Testaufbau:

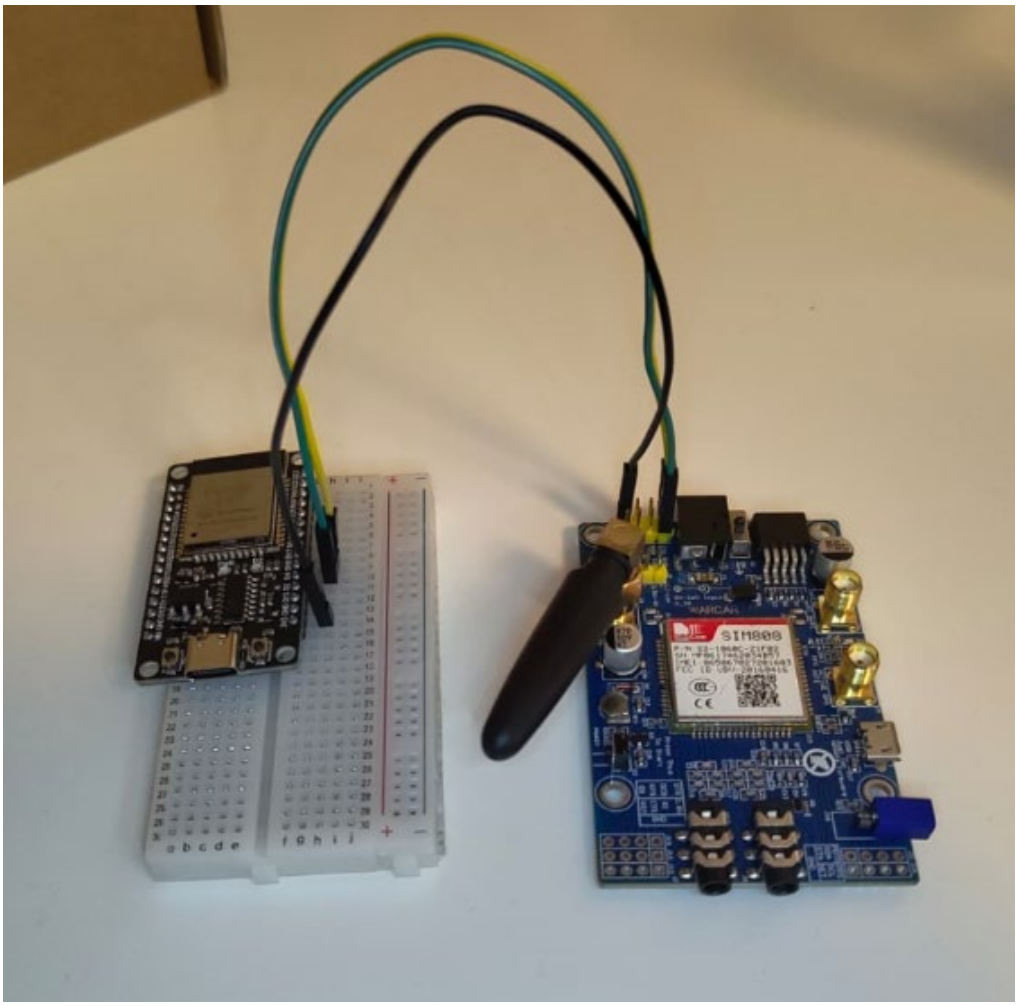

Abbildung 4.2: Testaufbau

a. Verbindung des ESP32 mit dem SIM808 GSM/GPS-Modul
b. Benötigte Hardware
c. SIM808 Modul
d. ESP32 Mikrocontroller
e. SIM-Karte (mit aktiviertem GSM- und Datenplan) – wird für erste Funktionstests nicht benötigt
f. Verkabelung (TX, RX für serielle Kommunikation)

Eine Versorgung über VMCU/GND des ESP32 ist nicht möglich, da das SIM808-Modul

kurzfristig hohe Stromspitzen (bis zu 2 A) benötigt. Zum Testen ist noch keine SIM-Karte erforderlich. Eine zu kleine SIM-Karte (z. B. Nano-SIM) kann mit einem passenden Adapter eingesetzt werden.

a. Firmware

Das SIM808-Modul besitzt eine eigene Firmware, die die interne Kommunikation mit GSM-, GPRS- und GPS-Funktionalitäten steuert. Eine aktuelle Firmware ist entscheidend, um die Kompatibilität mit modernen Netzwerken und stabilen Betrieb sicherzustellen.

Die Firmware kann bei Bedarf mit der folgenden Anleitung aktualisiert werden:

https://www.programmingboss.com/2024/08/update-sim800-gsm-module-firmware-using-SIMCom-firmware-download-tool.html

Empfehlung:

- Ein Firmware-Update sollte nur durchgeführt werden, wenn Verbindungsprobleme, fehlerhafte AT-Kommandos oder GPS-Probleme auftreten.
- Vor dem Update sollte die aktuelle Version mit dem Befehl AT+CGMR überprüft werden.
- Das Update erfolgt über ein serielles USB-UART-Kabel (nicht über den ESP32).

b. Überprüfung der Kommunikationsfähigkeit

c. Der Befehl `sim808.println("AT");` liefert mit `sim808.read()` "OK" zurück [13]

1. Beispiel für erfolgreiche Verbindung

Serielle Ausgabe:

```
ESP32 SIM808 Test Starting...
Waiting for SIM808 to initialize...
Sent AT command, waiting for response...
SIM808 response: A
SIM808 is responding!
T
OK
```

2. Beispiel für fehlerhafte Verbindung

Serielle Ausgabe:

```
ESP32 SIM808 Test Starting...
Waiting for SIM808 to initialize...
```

Sent AT command, waiting for response...
No response, retrying...
Sent AT command, waiting for response...
No response, retrying...
In diesem Fall reagiert das SIM808-Modul nicht auf die AT-Kommandos. Ursachen können sein:

- Falsche TX/RX-Belegung
- Unzureichende Spannungsversorgung (zu wenig Strom)
- Falsche Baudrate (Standard: 9600)
- Defekte oder nicht richtig angeschlossene GSM-Antenne

d. Sim-Karte

Für spätere Tests mit GSM-oder GPRS-Funktionalität wird eine aktive SIM-Karte benötigt.

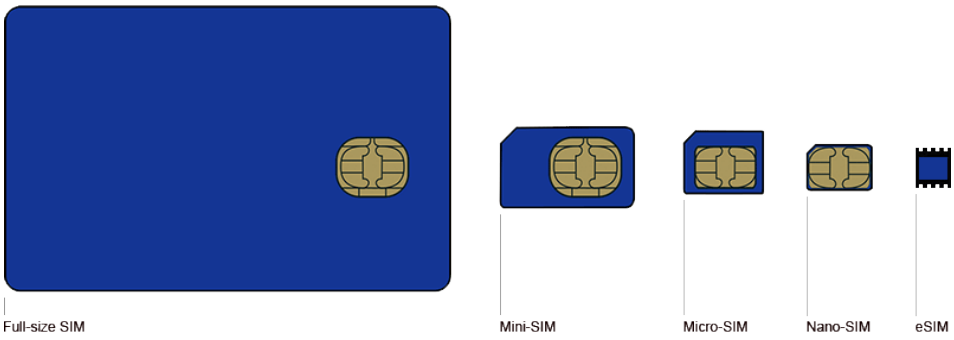

Abbildung 4.3: Sim-Größen[1]

Wichtige Punkte:

- Für das Sim808-Modul wird eine Karte der Größenordnung Mini-Sim benötigt
- Die PIN-Abfrage der SIM-Karte sollte deaktiviert sein.
- Ein Datentarif ist erforderlich, falls Internetverbindungen (HTTP, MQTT, etc.) getestet werden sollen.
- Für reine SMS- oder Anruf-Funktionalität genügt ein einfacher GSM-Tarif.

[1] https://www.skyynet.de/sim.php

4.2.1.1. Kommunikationstests

Um mögliche Fehler und Schäden vorzeitig festzustellen, sollte vorm in Betrieb setzen des Moduls ein einfacher Kommunikationstest durchgeführt werden. Dies erfolgt durch das Senden einer einfachen Test-SMS.

```
void sendSMS(const String& phoneNumber, const String& message) {
  sendAT("AT+CMGF=1");
  sim808.print("AT+CMGS=\"");
  sim808.print(phoneNumber);
  sim808.println("\"");


  unsigned long startTime = millis();
  while (millis() - startTime < 5000) {
    if (sim808.available()) {
      String response = sim808.readStringUntil('\n');
      response.trim();
      if (response.indexOf(">") != -1) {          break;
      }
    }
  }

  sim808.print(message);
  sim808.write(26);

  sendAT("");
}
```

Listing 1: Methode zum Senden von SMS

Ergebnis:

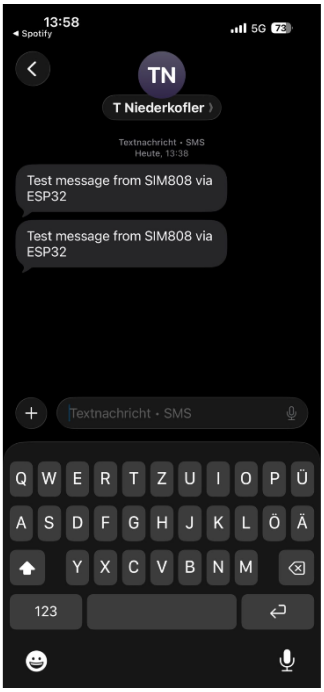

Abbildung 4.4: Test SMS gesendet vom Esp32

4.3. Sim Calling System

Das entwickelte System stellt eine Kombination aus klassischer analoger Telefontechnik und moderner digitaler Mikrocontrollersteuerung dar. Ziel ist es, ein herkömmliches Wählscheibentelefon mithilfe eines ESP32 und eines GSM-Moduls zu einem voll funktionsfähigen Mobiltelefon umzurüsten.

4.3.1.1. Zustandsmodell

Das System kann logisch in folgende Zustände unterteilt werden:

Ruhezustand

- Kein aktiver Anruf
- Hörer liegt auf
- System wartet auf Eingaben oder eingehende Anrufe

Eingehender Anruf

- GSM-Modul sendet „RING“
- Klingel wird aktiviert
- System wartet auf Abheben

Aktiver Anruf

- Verbindung wurde hergestellt
- Sprachübertragung läuft
- System überwacht Auflegen

Wählmodus

- Nutzer gibt Nummer über Wählscheibe ein
- Impulse werden gezählt und interpretiert

Speicher-/Menümodus

- Nummern können gespeichert oder abgerufen werden

Dieses Zustandsmodell verhindert Konflikte (z. B. gleichzeitiges Wählen und Annehmen eines Anrufs) und sorgt für ein störungsbefreites Anrufs Erlebnis.

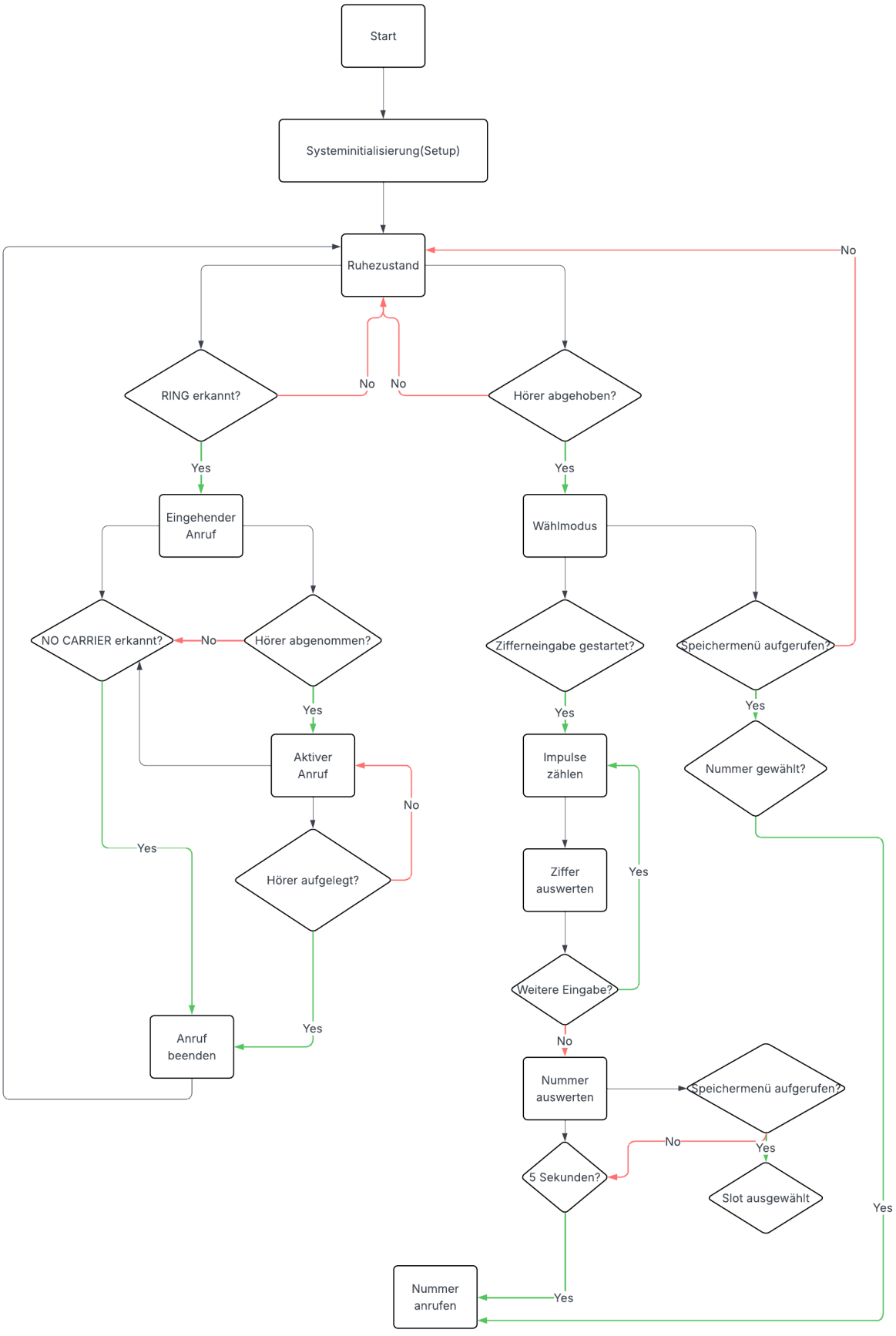

Abbildung 4.5: Flussdiagramm

4.3.1.2. Einbinden der nötigen Libraries

Zum Beginn des Programms werden die nötigen Bibliotheken initialisiert.

```
#include <Arduino.h>           //Import von Libraries
#include <vector>              //-----|------
#include <Wire.h>              //-----|------
#include <Adafruit_SSD1306.h> //-----|------
```

Listing 2: Library Import

- Arduino.h- Grundlegende Funktionen der Arduino-Plattform (z.B. pinMode, digitalWrite, millis).
- Vector-Grundlegende Speicherlogik(Arrays, Lists…)
- Wire.h-I²C Kommunikation wird für OLED-Display benötigt
- Adafruit_SSD1306.h- Ansteuerung eines OLED-Displays über I²C. Das Display dient als Debug- und Benutzerinterface, wodurch das System deutlich benutzerfreundlicher wird.

Als Entwicklungsebene wurde PlatformIO verwendet und folglich können die Libraries einfach über PIO-Home implementiert werden.

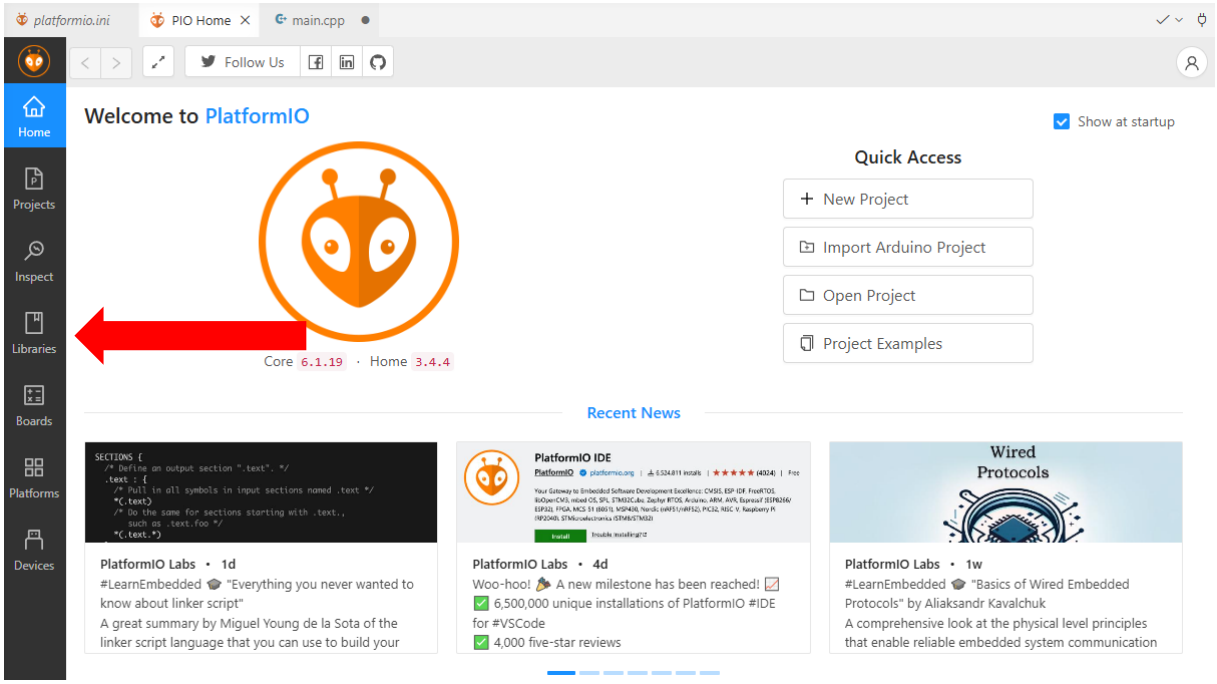

Abbildung 4.6: Libraries in PlatformIO

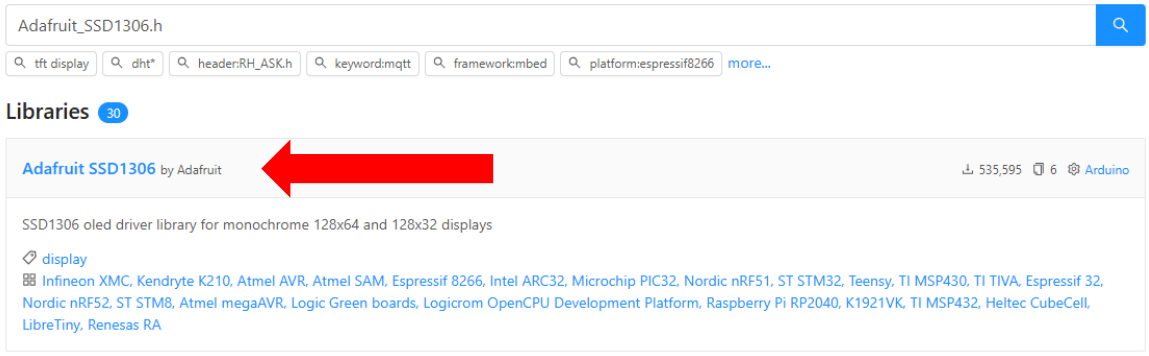

Abbildung 4.7: PlatformIO Library Suchfeld

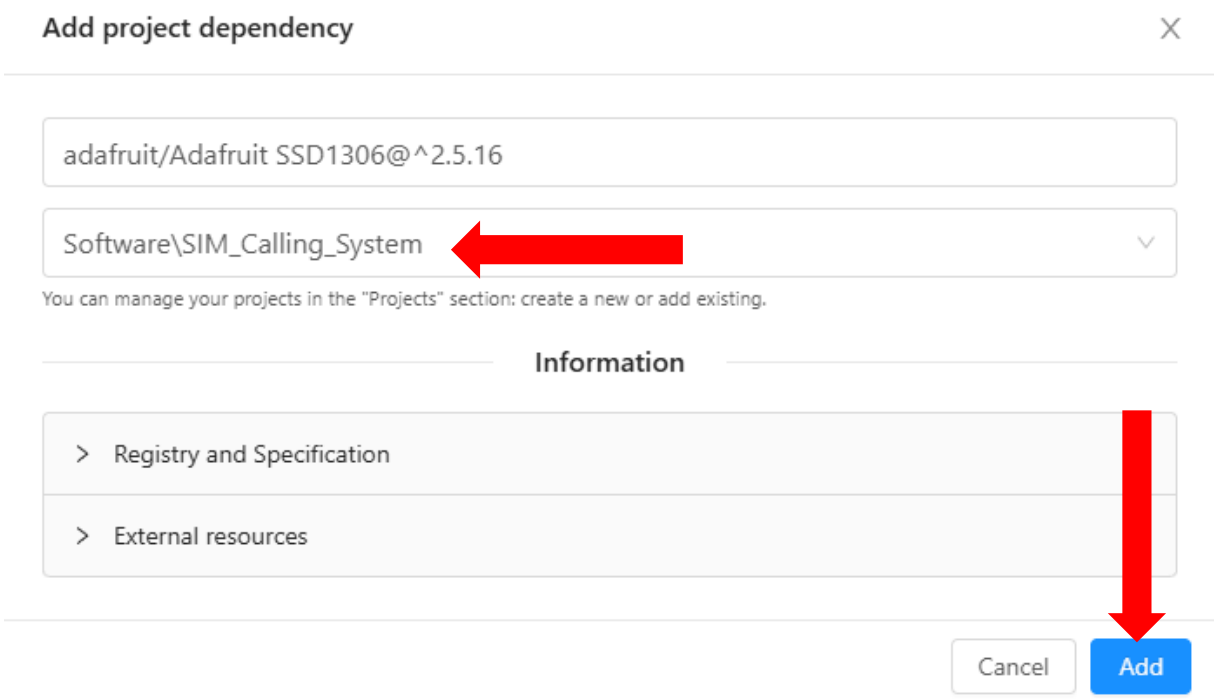

Abbildung 4.8: Einbinden von Library in Projekt

4.3.1.3. Projektkonfiguration(Platform.ini)

Zur Konfiguration des Projekts wird die Datei platformio.ini verwendet. Diese definiert die Entwicklungsumgebung, das Zielsystem sowie alle benötigten Bibliotheken.

```
[env:esp32doit-devkit-v1]
platform = espressif32
board = esp32doit-devkit-v1
framework = arduino
monitor_speed = 115200
lib_deps =
    plerup/EspSoftwareSerial@^8.2.0
    adafruit/Adafruit SSD1306@^2.5.16
```

Listing 3: Platform.ini

Die .ini-Datei dient als zentrale Konfigurationsstelle für:

- Compiler-Einstellungen
- Zielhardware
- Bibliotheken
- Upload- und Monitorparameter

Genauere Erklärung:

`platform = espressif32`

- Legt die verwendete Plattform fest, in dem Fall Esp32

`board = esp32doit-devkit-v1`

- Legt Pinbelegung, Speicher und Taktfrequenz fest

`framework = arduino`

- Legt das Programmier-Framework als Arduino Framework fest

`monitor_speed = 115200`

- Baudrate für den Serial Monitor, welcher für Debugging-Funktionen verwendet wird
- Muss übereinstimmen mit `Serial.begin(115200);`

```
lib_deps =
    plerup/EspSoftwareSerial@^8.2.0
    adafruit/Adafruit SSD1306@^2.5.16
```

- Bibliotheksverwaltung
- PlatformIO lädt Bibliotheken automatisch herunter

- Versionen werden fest definiert, was häufige Versionsfehler verhindert falls eine veraltete Version implementiert wurde
-

4.3.1.4. Initialisierung von serieller GSM-Kommunikation

Die Kommunikation mit dem Sim-Modul erfolgt über UART und benötigt so eine serielle Kommunikationsschnittstelle und je einen Reciever bzw. Transmitter-Pin.

```
HardwareSerial sim808(1); // Anlegen von serieller Kommunikation
#define SIM808_RX 14       // Festlegen einer Variable für RX (UART)
#define SIM808_TX 26       // Festlegen einer Variable für TX (UART)
```

Listing 4: Serielle Kommunikation(UART)

Hiermit kann der Esp32 AT-Befehle an das Sim-Modul senden.

4.3.1.5. Systeminitialisierung (Setup)

```
void setup
{
  pinMode(RING_PIN, OUTPUT);         // Definieren der Pins - INPUT/OUTPUT
  pinMode(Call_active_PIN, OUTPUT); //-----|------
  pinMode(AnalogInput, INPUT);       //-----|------
  pinMode(SAVEBUTTON_PIN, INPUT_PULLUP);

  digitalWrite(RING_PIN, LOW);
  digitalWrite(Call_active_PIN, LOW);

  Serial.begin(115200);
  sim808.begin(9600, SERIAL_8N1, SIM808_RX, SIM808_TX);
}
```

Listing 5: Setup

Hier werden hauptsächlich die Pins und deren Funktionen definiert bzw. die serielle Schnittstelle sowie der Serial Monitor gestartet.

4.3.1.6. Befehlsübermittlung zum Sim-Modul

Die Kommunikation mit dem Sim-Modul erfolgt durch die zuvor bereits kurz erwähnten AT-Befehle. Diese sind ein standardisierter Befehlssatz zur Steuerung und Konfiguration von Kommunikationsmodulen wie Modems oder eben auch Sim-Modulen. Der Begriff „AT“ steht für das englische Wort *attention* (deutsch: Aufmerksamkeit) und signalisiert dem Gerät, dass ein Steuerungskommando folgt. Jeder Befehl beginnt demnach mit „AT“ gefolgt von einem spezifischen Befehl. [14]

Die Kommunikation wurde, wie in folgender Kurzform realisiert:

```
void sendAT(const String &cmd) // Methode zur Kommunikation mit dem Sim-Modul
```

```
{
  if (cmd.length() > 0) // Präventiv gegen leere Befehle
  {
    sim808.println(cmd); // Senden des Befehls
  }

      String response = sim808.readStringUntil('\n');
      response.trim();
      Serial.println(response);
      if (response.indexOf("OK") != -1 || response.indexOf(">") != -1)
        return;
      if (response.indexOf("ERROR") != -1)
        return;
}
```

Listing 6: Befehlsübertragung

4.3.1.7. Erkennen eingehender Anrufe

```
if (response.indexOf("RING") != -1)
    {
      activeCall = true;
      ringing = true;

      if (averageAnalogValue < ANALOG_LOW_THRESHOLD && !recieverState)
      {
        answerCall();
        recieverState = true;
        ringing = false;
      }
    }
```

Listing 7: Anruferkennung

Wenn das Sim-Modul einen eingehenden Anruf erkennt, sendet es über die serielle Kommunikationsverbindung die Nachricht „RING“. Folglich reagiert das System entsprechend und steuert die Telefonklingel an. Zeitgleich wird mit Hilfe der Variable „activeCall“ sichergestellt, dass die Leitung besetzt ist und das System sich nicht durch mehrere erkannten Anrufe selbst erhängt. Insofern der Nutzer abheben möchte und den Hörer abnimmt, wird das durch einen Spannungszusammenbruch am analogen Pin erkannt und dementsprechend der Anruf mit der Methode „answerCall()“ entgegengenommen.

```
void answerCall() // Methode zur Anrufbeantwortung
{
  digitalWrite(RING_PIN, LOW); // Klingeln wird unterbunden
  sendAT("ATA");               // AT + Answer - Anruf wird beantwortet

}
```

Listing 8: Anrufbeantwortung

Wenn der Nutzer abhebt, hört das Telefon auf zu klingeln und der Befehl „ATA“ (AT+ANSWER) beantwortet den Anruf. [13]

4.3.1.8. Initiieren von Anrufen

```
if (!recieverState)
{
  currentState = averageAnalogValue < ANALOG_LOW_THRESHOLD ? HIGH : LOW;
  if (currentState != lastState)
  {
    if (currentState == HIGH && lastState == LOW && !activeCall)
    {
      if (!reading)
      {
        reading = true;
        Impulsecount = 1;
        readStartTime = millis();
        pulseStartTime = millis();
      }
      else if (millis() - pulseStartTime > 50)
      {
        Impulsecount++;
        pulseStartTime = millis();
        readStartTime = millis();
      }
    }
    lastState = currentState;
  }
}
if (reading && (millis() - readStartTime > 200))
{
  reading = false;
  int digit;
  if (Impulsecount >= 10)
    digit = 0;
  else
    digit = Impulsecount;
  currentPhoneDigits += String(digit);
  Serial.print("Ziffer erkannt: ");
  Serial.println(digit);
  lastDigitTime = millis();
  Impulsecount = 0;
}
if (currentPhoneDigits.length() > 0 && millis() - lastDigitTime > 5000)
{
  Serial.print("Nummer: ");
  Serial.println(currentPhoneDigits);
  Call(currentPhoneDigits);
  currentPhoneDigits = "";
```

```
  }
```

Listing 9:Nummerneingabe

Um Anrufe zu initiieren, muss zuerst der Hörer abgenommen und im Anschluss die Nummer mit dem Wählrad gewählt werden. Der ESP32 empfängt vom Telefon ein gleichbleibendes analoges Signal, welches beim Abheben etwas einbricht und beim Drehen des Wählrades abhängig von der gewählten Zahl eine bestimmte Anzahl von Unterbrechungen aufweist. Somit muss man, um die gewählte Nummer auszulesen, die Flanken im Signal erkennen und entsprechend auswerten.

Nach jeder eingelesen Ziffer ist ein sehr generöses Zeitfenster von 5 Sekunden gegeben, bis der Anruf für die gegebene Nummer getätigt wird. Alternativ kann auch ein Taster betätigt werden, um ins Speichermenü zu gelangen. Die nächste eingegebene Ziffer bestimmt den Slot in dem die Nummer gespeichert wird. Falls der Taster im neutralen Zustand betätigt wird gelangt man ins Anrufmenü, wo die nächste eingegebene Ziffer für die anzurufende Nummer steht. Eine erneute Betätigung des Taster bricht den Prozess ab.

```
 if (reading == false && Impulsecount == 0 && currentPhoneDigits.length() > 0)
  {
    int digit = currentPhoneDigits.toInt();
    selectedNumberSlot = digit;
  }
  int buttonState = digitalRead(SAVEBUTTON_PIN);
  if (buttonState == LOW && lastSaveButtonState == HIGH && millis() - but-
tonDebounce > 200)
  {
    buttonDebounce = millis();
    if (currentPhoneDigits.length() > 0 && millis() - lastDigitTime < 5000)
    {
      Serial.println("Speichermodus");
      saveMenu = true;
    }
    else if (!activeCall)
    {
      Serial.println("Anrufmenü");
      callMenu = true;
    }
    else if (callMenu || saveMenu)
    {
      Serial.println("Menü abgebrochen");
      callMenu = false;
      saveMenu = false;
    }
  }
  lastSaveButtonState = buttonState;
  if (saveMenu && selectedNumberSlot >= 0 && selectedNumberSlot < 10)
  {
```

```
    storednumbers[selectedNumberSlot] = currentPhoneDigits;

    Serial.print("Nummer gespeichert in Slot ");
    Serial.println(selectedNumberSlot);
    currentPhoneDigits = "";
    saveMenu = false;
  }
  if (callMenu && selectedNumberSlot >= 0 && selectedNumberSlot < 10)
  {
    if (storednumbers[selectedNumberSlot].length() > 0)
    {
      Serial.print("Calling Slot ");
      Serial.println(selectedNumberSlot);
      Call(storednumbers[selectedNumberSlot]);
    }
    else
    {
      Serial.println("Slot emty");
    }
    callMenu = false;
  }
```

Listing 10: Nummernspeicherlogik

Anrufe erfolgen mittels der folgenden Methode:

```
void Call(const String &phoneNumber)
{
  sim808.print("ATD"); // AT + "Dial"
  sim808.print(phoneNumber);
  sim808.println(";");
  activeCall = true;
}
```

Listing 11: Methode zum initieren von Anrufen

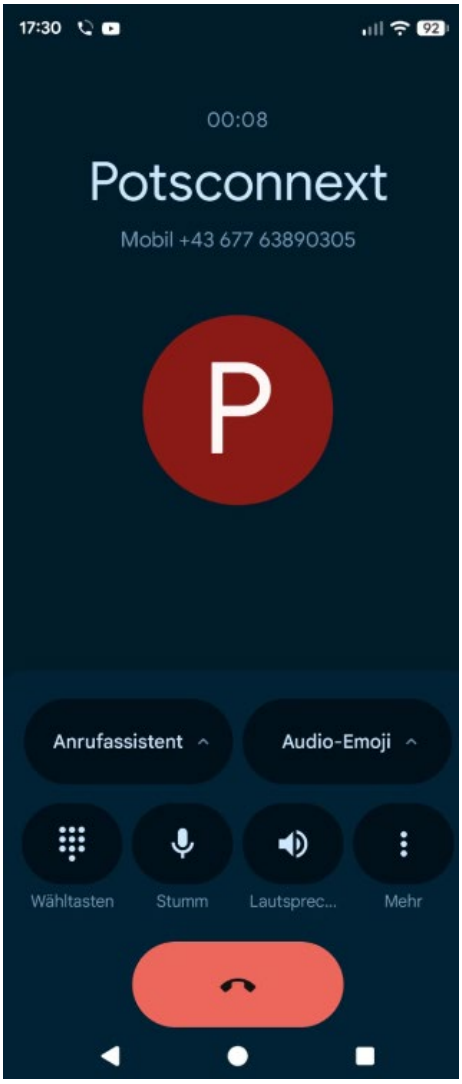

Abbildung 4.9: Anruf mit POTSConnect

4.3.1.9. Stabilisieren des analogen Signals

```
analogReadStarttime = millis();
static unsigned long lastRead = 0
  if (millis() - lastRead > 1
  {
    lastRead = millis
    analogValue = analogRead(AnalogInput
    analogValueSum = analogValueSum + analogValue
    analogValueCount += 1
  }
  if (analogValueCount > 99)
  {
    averageAnalogValue = analogValueSum / 100;
    analogValueSum = 0;
    analogValueCount = 0;
  }
```

Listing 12: Analog Stabilisation

Um etwaigen Störungen beim Signal, welches vom Telefon übertragen wird zu filtern wird mit der obrigen Funktion, alle 100ms der Durchschnittswert der in diesem Zeitraum eingelesenen analogen Werte bestimmt. Dies ermöglicht klares Einlesen des Hörerzustandes und entprellt zeitgleich die Nummerneingabe und reduziert so Rauschen und Fehlinterpretationen welche das Programm aufhängen könnten.

4.3.1.10. Verarbeitung eingehender Daten vom Sim-Modul

Sobald Daten vom SIM-Modul vorliegen, werden diese ausgelesen und analysiert.

```
 if (sim808.available())
  {
    String response = sim808.readStringUntil('\n');
    response.trim();
```

Listing 13: Einlesen von Daten vom Sim-Modul

- Verbindungsabbruch

```
 if (response.indexOf("NO CARRIER") != -1)
```

Listing 14: Anrufabbruch durch Externe

Die Response `"NO CARRIER"` bedeutet, dass die Verbindung mit dem Anrufer bzw. dem Angerufenen unterbrochen wurde, oder dass die Person am anderen Ende aufgelegt hat. Dementsprechend soll das System in dem Fall den Anrufszustand beenden und auflegen

4.3.1.11. Klingelsteuerung

```
if (ringing)
  {
    digitalWrite(RING_PIN, HIGH);
    ringStarttime = millis();
    if (ringStarttime > ringEndtime)
    {
      ringImpulse = !ringImpulse;
      ringEndtime += 500;
    }
    if (ringImpulse)
    {
      digitalWrite(RING_PIN, HIGH);
    }
    else
    {
      digitalWrite(RING_PIN, LOW);
    }
  }
```

Listing 15: Klingelsteuerung

Diese Funktion dient primär dem Energiesparen, um die Klingel nicht permanent versorgen zu müssen. Zusätzlich könnten hier Klingeltöne oder ähnliche Funktionen festgelegt werden.

4.3.1.12. Zeitsteuerung

Im gesamten Programm wird ausgenommen von einer Ausnahme konsequent auf delay() verzichtet und stattdessen auf non-blocking-Code gesetzt. Dies erfolgt durch die Abfrage des aktuellen Zeitpunktes mit dem Befehl millis() und darauffolgend wird dieser gespeicherte Zeitpunkt mit dem Startwert einer gewünschten Verzögerung verglichen.

Gründe dafür:

- delay() blockiert das gesamte Programm
- keine parallele Verarbeitung möglich
- schlechtere Reaktionsfähigkeit

Mit non-blocking-Code hingegen können mehrere Prozesse parallel ablaufen:

- Klingelsteuerung
- Signalanalyse
- Kommunikation
- Benutzereingaben

Dies ermöglicht ein kooperatives Multitasking-System.

```
unsigned long ringStarttime = 0;
unsigned long ringEndtime = 500;
unsigned long analogReadStarttime;
unsigned long pulseStartTime = 0;
unsigned long buttonDebounce = 0;
unsigned long readStartTime = 0;
unsigned long lastDigitTime = 0;
```

Listing 16: Zeitvariablen

Hier finden sich die Zeitvariablen, welche über das ganze Programm verteilt verwendet werden.

4.3.1.13. Allgemeines Funktionsprinzip der Zeitvariablen

Alle Zeitvariablen basieren auf demselben Prinzip:

```
if (millis() - startTime > Intervall)
```

Listing 17: Zeitfunktionsprinzip

Funktionsweise:

- startTime speichert den Zeitpunkt eines Ereignisses
- millis() liefert die aktuelle Zeit
- Die Differenz beschreibt die vergangene Zeit seit dem Ereignis
- Wird das Intervall überschritten, wird eine Aktion ausgelöst

4.3.1.14. Zeitliche Bestimmung der Wählradimpulse

```
unsigned long analogReadStarttime;
```

Listing 18: Leseprozessstartzeit

Diese Variable dient der zeitlichen Organisation des Analog-Ausleseprozesses.

Funktion:

- markiert den Startzeitpunkt eines Messzyklus
- ermöglicht kontrolliertes und gleichmäßiges Sampling

Bedeutung im System:

Die kontinuierliche Abtastung des analogen Signals ist notwendig, um:

- den Zustand des Hörers zu erkennen

- Wählimpulse korrekt auszuwerten
- stabile Durchschnittswerte zu berechnen

```
unsigned long pulseStartTime = 0;
```

Listing 19: Pulsstartzeitpunkt

Diese Variable ist entscheidend für die korrekte Interpretation der Wählscheibenimpulse.

- **Funktion:**
- speichert den Zeitpunkt des letzten erkannten Impulses
- dient zur Unterscheidung zwischen gültigen Impulsen und Störungen

Erklärung:

- Ein neuer Impuls wird nur akzeptiert, wenn mindestens 50 ms vergangen sind
- verhindert Mehrfachzählung durch Prellen oder Störungen

Technische Bedeutung:

- Signalentprellung
- Erhöhung der Messgenauigkeit
- zuverlässige Ziffernerkennung

```
unsigned long readStartTime = 0;
```

Listing 20: Einlesestartzeitpunkt

Diese Variable dient zur Erkennung des Endes einer Ziffer.

Funktion:

- speichert den Zeitpunkt des letzten Impulses
- definiert ein Zeitfenster zur Gruppierung von Impulsen

Erklärung:

- Wenn 200 ms kein neuer Impuls kommt → Ziffer abgeschlossen
- Impulse werden zu einer Zahl zusammengefasst

Bedeutung:

- Trennung einzelner Ziffern

- korrekte Interpretation der Wählscheibe

```
unsigned long lastDigitTime = 0;
```

Listing 21: Zeitpunkt der letzten Ziffer

Diese Variable bestimmt, wann eine vollständige Telefonnummer vorliegt.

Funktion:

- speichert den Zeitpunkt der letzten eingegebenen Ziffer
- dient zur Erkennung von Eingabepausen
- **Erklärung:**
- Wenn 5 Sekunden keine Eingabe erfolgt: Nummer gilt als vollständig Anruf wird automatisch gestartet
- **Technische Bedeutung:**
- Benutzerfreundliche Eingabe ohne Bestätigungstaste
- automatische Ablaufsteuerung

4.3.1.15. OLED Display

Zur visuellen Ausgabe von Systemzuständen wird ein OLED-Display verwendet. Dieses dient sowohl als Debugging-Werkzeug als auch als Benutzerinterface.

Initialisierung des Displays:

```
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64

Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);
```

Listing 22: Initialisierung des Displays

Erklärung:

- Das Display besitzt eine bestimmte Größe und diese muss entsprechend angegeben werden um klar festlegen zu können wie groß Schrift und andere Parameter des
- &Wire verweist auf die I²C-Schnittstelle
- -1 bedeutet, es kein Reset-Pin notwendig

Initialisierung im Setup:

```
display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
```

```
display.clearDisplay();
display.setTextColor(WHITE);
display.setTextSize(1);
display.setCursor(0, 0);
```

Listing 23: Initialisierung in Setup

Erklärung der einzelnen Schritte:

- `display.begin(…);`
  - Startet das Display mit I²C-Adresse 0x3C
- `display.clearDisplay();`
  - Löscht bereits vorhandene Bildschirmdaten aus dem Speicher
- `display.setTextColor(WHITE);`
  - Bestimmt die Textfarbe
- `display.setTextSize(1);`
  - Bestimmt die Schriftgröße
- `display.setCursor(0, 0);`
  - Legt die Startposition des ausgegebenen Textes fest.

Funktionsprinzip des Display-Systems:

Das OLED-Display arbeitet mit einem Frame-Buffer:

Ablauf:

1. Inhalte werden in Speicher geschrieben
2. Display bleibt unverändert

Vorteile dieses Systems:

- flimmerfreie Darstellung
- mehrere Elemente gleichzeitig darstellbar
- effizient

Nachteil:

- man muss immer aktiv aktualisieren

Anzeigen mit dem OLED-Display:

```
display.print("Beispiel");
display.println("text");
display.display();
```

Listing 24:Beispiel für OLED-Ausgabe

- `display.println()/display.print()`
  - Speichert den in Klammer als Parameter übergebenen Text and der festgelegten Cursorstelle des Displays im Framebuffer. Der Unterschied zwischen print und println ist, dass println zusätzlich einen Zeilenumbruch ausgibt.
- `display.display();`
  - Dieser Befehl gibt nun tatsächlich den im Buffer gespeicherten Text aus.

4.4. Fehlerbehandlung und Systemrobustheit

Ein wesentliches Ziel des Systems ist ein stabiler und zuverlässiger Betrieb auch unter nicht idealen Bedingungen. Daher wurden mehrere Mechanismen zur Fehlererkennung und -behandlung implementiert.

Erkennung typischer Fehlerzustände

Das SIM-Modul liefert standardisierte Rückmeldungen, welche aktiv ausgewertet werden:

- **„OK“**-Befehl erfolgreich ausgeführt
- **„ERROR“**-Fehler bei der Ausführung
- **„NO CARRIER“**-Verbindungsabbruch
- **„RING“**-eingehender Anruf

Timeout-Mechanismus

Innerhalb der Funktion sendAT() wird ein Timeout verwendet:

- Maximale Wartezeit: 5 Sekunden
- Falls keine gültige Antwort erfolgt Abbruch der Wartephase

Vorteile:

- Verhindert das „Einfrieren“ des Systems
- Stellt sicher, dass der Programmfluss erhalten bleibt

Schutz vor ungültigen Zuständen

Mehrere Variablen verhindern logische Fehler:

- activeCall-verhindert parallele Anrufe
- recieverState-verhindert mehrfaches Abheben
- reading-verhindert doppelte Impulserkennung

Diese Schutzmechanismen sind essenziell für ein fehlerfreies Systemverhalten.

4.4.1.1. Energieeffizienz und Ressourcenmanagement

Da das System planweise im Dauerbetrieb läuft, wurde besonderer Wert auf Resourceneffizienz gelegt.

Optimierung der CPU-Auslastung

- Non-blocking-Code und verzicht auf delay() im weiterführenden Programmablauf

Dadurch:

- parallele Abläufe möglich
- schnellere Reaktion auf Ereignisse
- bessere Skalierbarkeit

Optimierung der Hardware-Nutzung

- Klingel wird gepulst statt dauerhaft betrieben
- Display wird nur bei Änderungen aktualisiert
- Analoge Messwerte werden gefiltert statt ständig ausgewertet

Serielle Kommunikation

- Baudrate bewusst auf 9600 gesetzt
- ausreichend stabil für GSM-Kommunikation
- reduziert Fehleranfälligkeit

5. Anhang

In diesem Kapitel werden Projektmanagement, Aufgabenstellung, Scrum und andere Anhänge abgebildet.

5.1 Poster

Abbildung 5.1: Poster POTSConnect

5.1. Projektmanagement

Das Projekt wird mittels Scrum in GIT gemanagt, in Kapitel 5.1.2 ist der Scrum-Projektplan zu sehen.

5.1.1. Aufgabenstellung des Gesamtprojekts

Ein Modul, das alte analoge Telefone ins moderne Zeitalter bringen soll. Ziel ist, mit dem angeschlossenem Telefon später über eine SIM-Karte und 2G-Netz telefonieren zu können.

5.1.2. Scrum-Projektplan

Das Projekt verwendet Scrum zur Projektplanung und zur zeitlichen Einteilung der Handlungen. In den nachfolgenden Abbildungen wird dieser Projektplan dargestellt. Die Tasks sind in die jeweiligen Sprints, passend zu den Meilensteins Präsentationen eingeteilt.

Abbildung 5.2: Scrum-Plan der User Stories von POTSConnect

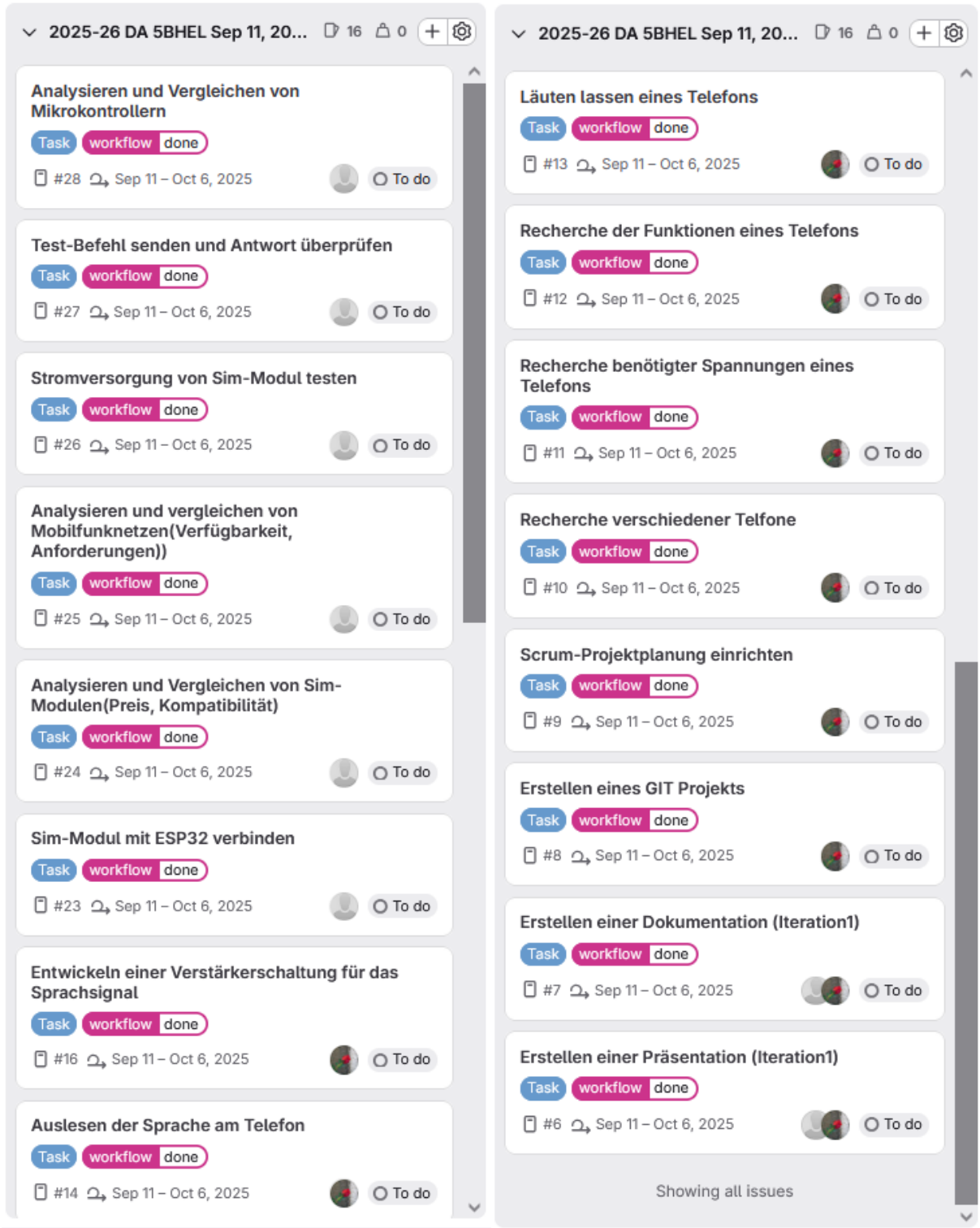

Abbildung 5.3:Scrum-Plan der Tasks aus Iteration 1

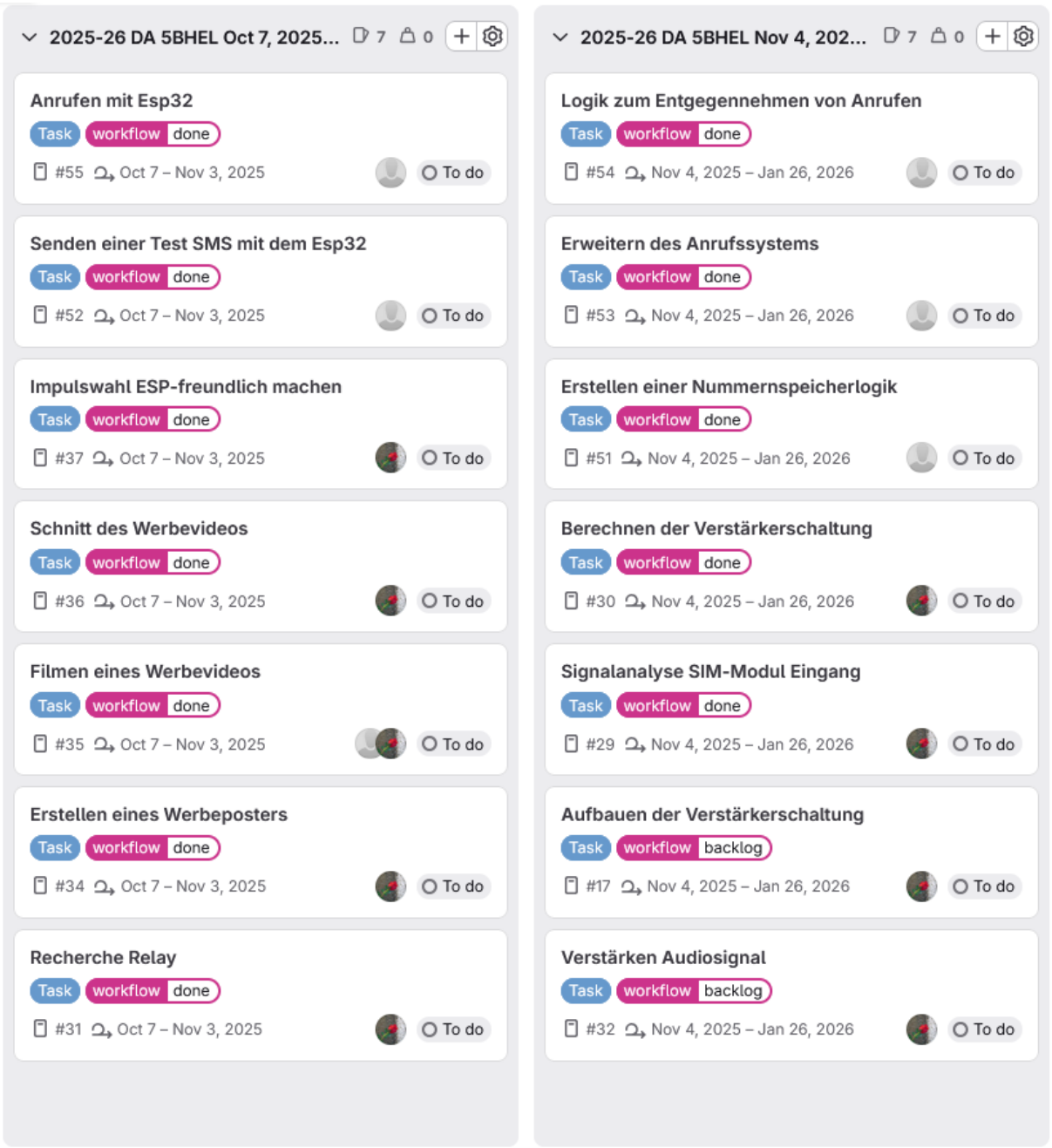

Abbildung 5.4: Scrum-Plan der Tasks aus Iteration 2 und 3

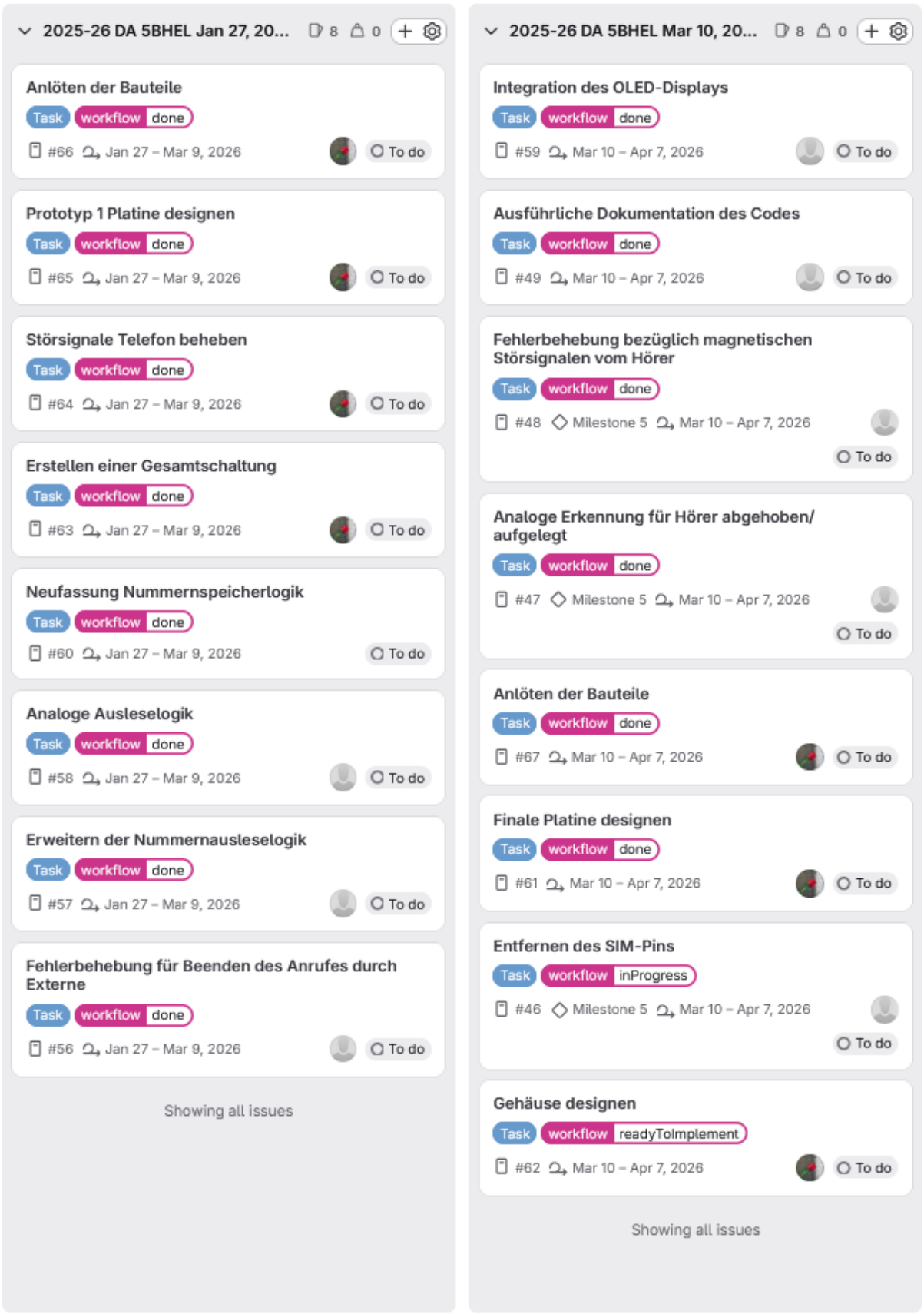

Abbildung 5.5: Scrum-Plan der Tasks aus Iteration 4 und 5

5.2. Kostenaufstellung

Diplomarbeit: POTSConnect, 06.04.2026		
Kostenart	**Stückpreis / Einheiten**	**Kosten in €**
SIM808 Modul	29,06 / 2	58,12
Analoges Telefon	25,00 / 1	25,00
Printtrafo 2x 15V	52,75 / 2	105,50
Power-Supply	14,22 / 1	14,22
Widerstand 1kΩ	0,07 / 10	0,7
	Gesamtkosten	**203,54**

Tabelle 5.1: Kostenaufstellung

Die oben aufgeführten Beträge beziehen sich ausschließlich auf die Materialkosten im Rahmen der Diplomarbeit (Prototypenbau). Diese enthalten einmalige Anschaffungen sowie Bauteile in geringen Stückzahlen. Diese Aufstellung ist daher nicht mit den Stückpreisen einer späteren Serienfertigung gleichzusetzen, da dort durch Mengenrabatte und optimierte Auswahl der Komponenten deutlich geringere Einzelkosten pro Modul erreicht werden.

5.2.1. Kosten für eine Serienanfertigung

Hier ist zu sehen, wie viel eine Serienanfertigung von ca. 1000 Stück an Modulen kosten würde.

Kostenart	**Stückpreis [€]**	**Anzahl/Modul**	**Kosten in € a 1000**	**Quelle**
SIM-Modul	24,07	1	24 070,00	amazon.de
Printtrafo	35,61	1	35 610,00	rs-online.com
Elektronik Kleinbauteile	10,00	1	10 000,00	diverse Anbieter
Relais	4,89	2	9 780,00	reichelt.at
Spule	5,32	1	5 320,00	mouser.at
DC/DC SIM-Modul	1,79	1	1 790,00	roboter-bausatz.de
DC/DC ESP32	10,89	1	10 890,00	reichelt.at
Power-Supply	14,22	1	14 220,00	reichelt.at
ESP32	7,36	1	7 360,00	reichelt.at
Gesamtkosten	**119,04**		**119 040,00**	

Tabelle 5.2: Gesamtkosten für ~1000 Stück

Bei den „Elektronik-Kleinbauteilen“ (wie Widerständen, Kondensatoren und Transistoren) handelt es sich um einen geschätzten Pauschalbetrag für die Bestückung eines Moduls. Die Materialkosten für ein einzelnes fertiges Modul belaufen sich somit auf insgesamt **~119,04 €**, woraus sich bei einer Serienfertigung von ~1000 Stück die Gesamtsumme von **~119.040,00 €** ergibt. Hinzu kommen dann noch die Platine und das Gehäuse des Moduls.

5.3. Besprechungsprotokolle

HTL | MÖSSINGERSTRASSE

AN

Teilnehmer und Unterrichtete

ERGEBNISPROTOKOLL ZUR BESPRECHUNG

Eingangs-vermerke

Veranstalter: HTL-Mössingerstraße

Protokollführer: Johannes Schlager

Projektbetreuer: Dipl.-Ing. Dr. Dieter Maier

Erstellungsdatum: 2025-10-13

am Tag, Datum: Montag, 13. Oktober 2025

Ort/Raum: Klagenfurt/W101

Thema:

Projektstatusgespräch Iteration 1

Projektname: POTSConnect

Teilnehmer (alphabetisch)		Unterrichtete (alphabetisch)		
Name	Klasse, Standort	Name		Klasse, Standort
Hr. Thomas Niederkofler	5BHEL, Klagenfurt	Hr. MSc	Peter Gigler	Lehrkraft, Klagenfurt
Hr. Johannes Schlager	5BHEL, Klagenfurt	Hr. Dipl.-Ing.	Herwig Guggi	Lehrkraft, Klagenfurt
		Hr. Dipl.-Ing.	Johann Leitner	Lehrkraft, Klagenfurt
		Hr. Dipl.-Ing.	Dr. Dieter Maier	Lehrkraft, Klagenfurt

Thomas Niederkofler:

Aktueller Stand:

- Verbindung Sim-Modul
- Kommunikationsstruktur

Plan für Iteration 2

- SMS senden(Test für Verbindung)
- Anrufe tätigen/entgegennehmen

Feedback:

- Kein senden von SMS – zu viel Zeitaufwand
- zu viel Dialekt
- Keine unnötigen Folien über Stoff von der 2.Klasse
- Schnelleres Arbeiten!

Johannes Schlager:

Aktueller Stand:

- Blockschaltbild
- Ansteuern Telefon
- Auslesen Audiosignal

Plan für Iteration 2:

- Aufbau Verstärkerschaltung für das Audiosignal
- Audiosignal für Sim808 nutzbar machen
- Wahlsignal mit DCDC Konverter für ESP32 nutzbar machen

Feedback:

- Blockschaltbild erneuern
- Keine Zeichnungen von der Tafel in die Präsentation
- Schnelleres Arbeiten!

Erledigung von, Termin

HTL Mössingerstraße
1_POTSConnect_Besprechungsprotokoll.docx

Seite 1 von 1
V01

Abbildung 5.6: Besprechungsprotokoll vom 13.10.2025

HTL | MÖSSINGERSTRASSE

AN

Teilnehmer und
Unterrichtete

ERGEBNISPROTOKOLL ZUR BESPRECHUNG

Veranstalter: HTL-Mössingerstraße

Protokollführer: Johannes Schlager

Projektbetreuer: Dipl.-Ing. Dr. Dieter Maier

Eingangsvermerke

Erstellungsdatum: 2025-11-24

am Tag, Datum: Montag, 24. November 2025

Ort/Raum: Klagenfurt/W101

Thema:

Projektstatusgespräch Iteration 2

Projektname: POTSConnect

Teilnehmer (alphabetisch)	
Name	Klasse, Standort
Hr. Thomas Niederkofler	5BHEL, Klagenfurt
Hr. Johannes Schlager	5BHEL, Klagenfurt

Unterrichtete (alphabetisch)		
Name		Klasse, Standort
Hr. MSc	Peter Gigler	Lehrkraft, Klagenfurt
Hr. Dipl.-Ing	Herwig Guggi	Lehrkraft, Klagenfurt
Hr. Dipl.-Ing	Johann Leitner	Lehrkraft, Klagenfurt
Hr. Dipl.-Ing. Dr.	Dieter Maier	Lehrkraft, Klagenfurt

Thomas Niederkofler: — **Erledigung von, Termin**

Aktueller Stand:

- Verbindung Sim-Modul
- Kommunikationsstruktur
- Anrufe tätigen/entgegennehmen
- Verbindung zum Telefon

Plan für Iteration 3

- Klingelfunktion Problembehebung
- Wählradfunktion verbessern

Feedback:

- Siehe Plan für Iteration 3

Johannes Schlager:

Aktueller Stand:

- Stand Iteration 1
- Ansteuern Telefon
- Auslesen Audiosignal
- Audiosignal für Sim808 nutzbar machen
- Wahlsignal mit Spannungsteiler für ESP32 nutzbar machen

Plan für Iteration 3

- Telefon reparieren
- Restlichen Bauteile bestellen
- Platine designen
- Hören über Telefon
- Projekt fit für Tag der offenen Tür machen

Feedback:

- Telefon versuchen zu reparieren
- Homepage mit Poster abgleichen (Retro-Stil)

HTL Mössingerstraße
2_POTSConnect_Besprechungsprotokoll.docx

Seite 1 von 1
V01

Abbildung 5.7: Besprechungsprotokoll vom 24.11.2025

HTL | MÖSSINGERSTRASSE

AN

Teilnehmer und
Unterrichtete

ERGEBNISPROTOKOLL ZUR BESPRECHUNG

Eingangs-vermerke

Veranstalter: HTL-Mössingerstraße

Protokollführer: Johannes Schlager

Projektbetreuer: Dipl.-Ing. Dr. Dieter Maier

Erstellungsdatum: 2026-02-02

am Tag, Datum: Montag, 02. Feber 2026

Ort/Raum: Klagenfurt/W101

Thema:

Projektstatusgespräch Iteration 3

Projektname: POTSConnect

Teilnehmer (alphabetisch)		Unterrichtete (alphabetisch)		
Name	Klasse, Standort	Name		Klasse, Standort
Hr. Thomas Niederkofler	5BHEL, Klagenfurt	Hr. MSc	Peter Gigler	Lehrkraft, Klagenfurt
Hr. Johannes Schlager	5BHEL, Klagenfurt	Hr. Dipl.-Ing	Herwig Guggi	Lehrkraft, Klagenfurt
		Hr. Dipl.-Ing	Johann Leitner	Lehrkraft, Klagenfurt
		Hr. Dipl.-Ing. Dr.	Dieter Maier	Lehrkraft, Klagenfurt

Thomas Niederkofler: — Erledigung von, Termin

Aktueller Stand:
- Analoges Einlesen
- Beheben eines Fehlers, welcher das System zum Neustarten zwang

Plan für Iteration 4
- Analogen Input stabilisieren
- Projekt in nur die nötigsten Funktionen zusammenfassen

Feedback:
- Bessere Git Struktur
- Organisieren von Code

Johannes Schlager:

Aktueller Stand:
- Restlichen Bauteile bestellen
- Platine designen
- Hören und Sprechen über Telefon

Plan für Iteration 4:
- Prototyp v2 erfolgreich zum Laufen bringen
- Bauteile bestellen
- Finale Tests durchführen
- Gehäuse design starten

Feedback:
- Besseres Vorbereiten (englische Wörter lernen/merken)
- Mehr Bilder in der Präsentation
- Bessere Beschreibung des Projektes zu Beginn

HTL Mössingerstraße
POTSConnect_Besprechungsprotokoll.docx

Seite 1 von 1
V01

Abbildung 5.8: Besprechungsprotokoll vom 02.02.2026

5.4. Arbeitszeitnachweis

5.4.1. Arbeitszeitnachweis Johannes Schlager

Datum	Anfang	Ende	Beschreibung	Dauer in h
13.09.2025	10:00	15:00	Recherche (Telefone)	5,00
14.09.2025	14:30	15:30	Recherche (Funktionen eines Telefons)	1,00
20.09.2025	16:00	18:00	Diskussion mit dem Betreuer (Schaltung)	2,00
24.09.2025	20:00	23:00	Telefon anschließen	3,00
01.10.2025	17:30	19:00	Sprachsignal messen (Hörer)	1,50
04.10.2025	15:00	17:30	Sprachsignal messen (a/b Ader)	2,50
05.10.2025	09:00	13:00	Entwicklung Verstärkerschaltung	4,00
08.10.2025	19:00	20:30	Entwicklung Verstärkerschaltung	1,50
01.10.2025	19:00	21:00	GIT überprüfen	2,00
01.10.2025	21:00	22:30	Scrum	1,50
08.10.2025	20:00	23:00	Dokumentation schreiben	3,00
09.10.2025	21:00	23:00	Präsentation erstellen Iteration 1	2,00
13.10.2025	20:00	21:00	Besprechungsprotokoll Iteration 1	1,00
15.10.2025	16:00	19:00	Spannungsversorgungskonzept Neuentwicklung	3,00
16.10.2025	14:00	15:00	Diskussion mit dem Betreuer	1,00
18.10.2025	22:00	00:00	Aufbau des neuen Spannungsversorgungskonzepts	2,00
19.10.2025	00:00	01:00	Aufbau des neuen Spannungsversorgungskonzepts	1,00
22.10.2025	16:00	18:00	Design Versorgung mit Relay	2,00
23.10.2025	16:00	16:30	Dokumentation schreiben	0,50
26.10.2025	13:30	14:00	Recherche Bauteile	0,50
26.10.2025	22:00	22:30	Spannungsversorgungskonzept Neuentwicklung	0,50
28.10.2025	12:00	12:30	Dokumentation schreiben	0,50
01.11.2025	14:00	17:30	Aufbau Versorgungsschaltung	3,50
04.11.2025	19:00	19:30	Scrum	0,50
04.11.2025	19:30	21:30	Homepage	2,00
10.11.2025	07:00	10:00	Spannungsversorgungskonzept	3,00
13.11.2025	16:00	17:00	Impulswahlsignal auslesen	1,00
15.11.2025	19:00	21:00	Spannungsversorgungskonzept	2,00
16.11.2025	10:00	14:00	Homepage	4,00
18.11.2025	16:00	18:00	Sprachsignal filtern	2,00
18.11.2025	18:00	19:00	Werbeplakat erstellen	1,00
20.11.2025	14:00	16:00	Werbevideo drehen	2,00
20.11.2025	16:00	17:00	Werbeplakat erstellen	1,00
22.11.2025	09:00	12:00	Werbevideo editieren	3,00
23.11.2025	07:00	09:00	Werbeplakat erstellen	2,00
23.11.2025	20:00	22:00	Dokumentation schreiben	2,00
23.11.2025	20:00	22:00	Präsentation erstellen Iteration 2	2,00
24.11.2025	17:00	18:00	Besprechungsprotokoll Iteration 2	1,00
27.11.2025	13:00	17:00	Platine designen	4,00
03.12.2025	15:00	20:00	Platine designen	5,00
08.12.2025	08:00	11:00	Testaufbau mit Kommunikation	3,00
09.12.2025	19:00	22:00	Platine designen	3,00
14.12.2025	08:00	14:00	Platine designen	6,00
20.12.2025	07:00	12:00	Auflöten der Bauteile auf der Platine	5,00
23.12.2025	12:00	15:00	Spule design	3,00
12.01.2026	16:00	20:00	Sprachsignal Probleme beheben	4,00
16.01.2026	08:00	15:00	Tag der offenen Tür	7,00
19.01.2026	19:00	23:00	Neudesign Spannungsversorgungskonzept	4,00

22.01.2026	20:00	00:00	neue Platine designen	4,00
23.01.2026	00:00	02:00	neue Platine designen	2,00
25.01.2026	10:00	15:00	Iteration 3 Präsentation	5,00
30.01.2026	11:00	12:00	GIT überprüfen	1,00
01.02.2026	10:00	15:00	Platine designen	5,00
05.02.2026	15:00	18:00	Schaltplan überarbeiten	3,00
10.02.2026	13:00	14:00	DC/DC Konverter Recherche	1,00
11.02.2026	12:00	15:00	Homepage	3,00
15.02.2026	10:00	13:00	Dokumentation	3,00
20.02.2026	18:00	19:00	Platine designen	1,00
21.02.2026	15:00	17:00	Testen der Schaltung	2,00
24.02.2026	13:30	15:00	Dokumentation	1,50
24.02.2026	15:00	15:30	Blockdiagramm bearbeiten	0,50
24.02.2026	15:30	16:00	Systemstrukturplan bearbeiten	0,50
02.03.2026	16:00	18:00	Änderungen der Platine im Layout	2,00
04.03.2026	12:00	15:00	Änderungen der Platine im Layout	3,00
12.03.2026	13:00	16:00	Dokumentation	3,00
16.03.2026	10:00	14:00	Testen der Schaltung	4,00
21.03.2026	19:00	00:00	Gehäuse designen	5,00
24.03.2026	14:00	15:00	Fertigstellung inkl. Zusammenbau	1,00
26.03.2026	13:00	14:30	Löten Platine	1,50
26.03.2026	14:30	17:00	Testen der Schaltung	2,50
27.03.2026	17:00	21:00	Dokumentation schreiben	4,00
30.03.2026	18:00	20:00	Dokumentation schreiben	2,00
31.03.2026	10:00	20:00	Dokumentation schreiben	10,00
01.04.2026	10:00	20:00	Dokumentation schreiben	10,00
02.04.2026	10:00	20:00	Dokumentation schreiben	2,00
03.04.2026	15:00	18:00	Dokumentation schreiben	3,00
04.04.2026	15:00	16:30	Dokumentation schreiben	1,50
			Gesamtarbeitszeit (Ist)	**204,00**
			Soll	160,00
			Offen (Ist-Soll)	-44,00

Tabelle 5.3 : Arbeitszeitnachweis Johannes Schlager

5.4.2. Arbeitszeitnachweis Thomas Niederkofler

Datum	Anfang	Ende	Beschreibung	Dauer in h
17.09.2025	14:00	16:00	Recherche (Mobilfunknetze)	2,00
19.09.2025	16:15	17:45	Recherche (Sim-Module)	1,50
20.09.2025	14:30	16:30	Recherche (Telefon)	2,00
25.09.2025	20:00	22:00	Erstellen des Git-Projekts und Recherche	2,00
01.10.2025	14:00	15:00	Recherche Sprachübertragung	1,00
01.10.2025	15:00	17:00	Informieren über Telefonie mit ESP32	2,00
05.10.2025	14:00	18:00	Testaufbau und Ansteuern des Sim-Moduls	4,00
08.10.2025	18:00	20:00	Recherche (Sim-Karten)	2,00
10.10.2025	10:30	12:45	Dokumentation (Iteration 1)	2,25
10.10.2025	14:35	17:35	Präsentation (Iteration 1)	3,00
12.10.2025	14:20	16:45	Scrum + Dokumentation	2,42
15.10.2025	14:30	18:40	Recherche AT-Befehle	4,17
18.10.2025	11:00	15:00	Entsperren der Sim-Karte-senden einer Test-SMS	4,00
19.10.2025	13:00	15:00	Fehlerbehebung und verbessern des Testaufbaus	2,00
22.10.2025	16:00	18:00	Statischer Test-Anruf	2,00
25.10.2025	13:40	16:30	Non-blocking Code	2,83
26.10.2025	15:20	16:20	Testanruf mit Mikrofon und Lautsprecher	1,00
27.10.2025	16:00	20:30	Taster für Abheben/Auflegen	4,50
28.10.2025	10:00	11:00	Problembehebung-Taster	1,00
30.10.2025	14:00	18:00	Ansteuern der Telefonklingel über den ESP32	4,00
05.11.2025	16:50	20:20	Verbesserungen und Fehlerbehebungen im Code	3,50
07.11.2025	14:00	18:00	Homepage	4,00
08.11.2025	10:00	12:00	Dokumentation (Iteration 1)	2,00
08.11.2025	15:40	20:30	Einleselogik für Wählrad	4,83
09.11.2025	16:20	18:00	Fehleranalyse	1,67
12.11.2025	14:00	18:00	Speicherlogik für Telefonnummern	4,00
19.11.2025	15:00	16:00	Werbeplakat	1,00
21.11.2025	16:00	17:00	Werbevideo	1,00
23.11.2025	15:00	16:40	Scrum + Dokumentation	1,67
23.11.2025	22:00	23:50	Präsentation (Iteration 2)	1,83
28.11.2025	14:00	18:00	Innovationspreis	4,00
03.12.2025	15:00	19:00	Neue Sim-Karte und Änderungen bei Klingel	4,00
06.12.2025	10:00	14:00	Funktionstests	4,00
12.12.2025	18:00	20:00	Fehlerbehebungen	2,00
24.12.2025	13:00	16:00	Lautstärke und Mikrofon und Website	3,00
10.01.2026	09:00	13:00	Tests and Pin Änderungen	4,00
17.01.2026	10:00	14:00	Auflegen von Anrufer Test	4,00
18.01.2026	16:00	20:00	Code Kommentierung und Taster erstetzen	4,00
24.01.2026	12:00	16:00	Wählrad auslesen Tests	4,00
25.01.2026	12:00	20:00	Dokumentation (Iteration 3) und Präsentation	8,00
15.02.2026	14:00	18:00	Platine	4,00
18.02.2026	16:00	17:00	Fehlerbehebung	1,00
21.02.2026	10:00	12:00	Fehlerbehebung	2,00
27.01.2026	14:00	22:00	Erstellen der fertigen Anrufverarbeitung	8,00
28.01.2026	16:00	20:00	Analoges Erkennen vom Zustand des Hörers	4,00
29.01.2026	13:00	15:00	Code Dokumentation	2,00
11.03.2026	16:00	23:00	Code Dokumentation	7,00
12.03.2026	10:00	20:00	Implementieren der Speicher-Nummernausleselogik	10,00
15.03.2026	12:00	20:00	Dokumentation	8,00
21.03.2026	14:00	16:00	OLED-Display	2,00

22.03.2026	16:00	18:00	Dokumentation	2,00
28.03.2026	17:00	19:00	Scrum	2,00
29.03.2026	16:00	19:00	Fehlerbehebung	3,00
04.04.2026	13:00	15:00	OLED-Display	2,00
05.04.2026	11:00	22:00	Dokumentation	11,00
06.04.2026	12:00	20:00	Dokumentation	8,00
			Gesamtarbeitszeit (Ist)	**192,17**
			Soll	160,00
			Offen (Ist-Soll)	-32,17

Tabelle 5.4: Arbeitszeitnachweis Thomas Niederkofler

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[1] V. Lange-Janson, „Elektronikbastler," 27 11 2018. [Online]. Available: https://elektronikbasteln.pl7.de/funktion-telefon. [Zugriff am 12 10 2025].

[2] „Wikipedia, Plain Old Telephone Service," [Online]. Available: https://de.wikipedia.org/wiki/Plain_Old_Telephone_Service. [Zugriff am 28 10 2025].

[3] „Wikipedia, Telekommunikations-Anschluss-Einheit," [Online]. Available: https://de.wikipedia.org/wiki/Telekommunikations-Anschluss-Einheit. [Zugriff am 12 10 2025].

[4] Klaus, „dl0hst.de," Hochschule Stralsund, 2021. [Online]. Available: https://www.dl0hst.de/mini-ringkern-rechner.htm. [Zugriff am 17 11 2025].

[5] Espressif Systems, „ESP32 Series Datasheet," 2024. [Online]. Available: : https://www.espressif.com/. [Zugriff am August 2025].

[6] Raspberry Pi Foundation, „Raspberry Pi Technical Documentation," 2024. [Online]. Available: https://www.raspberrypi.org/. [Zugriff am August 2025].

[7] Arduino, „Arduino Uno," [Online]. Available: https://store-usa.arduino.cc/products/arduino-uno-rev3?srsltid=AfmBOop2AM7V659rtnOvNNkXEtoILnwWvraLe8-d2T1IiVV13ty9DmTD. [Zugriff am August 2025].

[8] SIMCom Wireless Solutions, „SIM808 Hardware Design," 2023. [Online]. Available: https://simcom.com/. [Zugriff am August 2025].

[9] „Amazon Sim808," 2025. [Online]. Available: https://www.amazon.de/dp/B083M5VBQT. [Zugriff am August 2025].

[10] „Amazon Sim7000," 2025. [Online]. Available: https://www.amazon.de/SIM7000E-Erweiterungsschild-Arduino-GSM-Federantenne-GPS-Innenantenne/dp/B07MK25Z8L. [Zugriff am August 2025].

[11] „alldatasheet," [Online]. Available: https://www.alldatasheet.com/datasheet-pdf/download/1741390/SIMCOM/SIM808.html. [Zugriff am August 2025].

[12] „Wikipedia, Mobilfunkstandard," [Online]. Available: https://de.wikipedia.org/wiki/Mobilfunkstandard. [Zugriff am 20 08 2025].

[13] [Online]. Available: https://www.dfrobot.com.cn/images/upload/File/DFR0355/20160425154144zpbheu.pdf. [Zugriff am 2025].

[14] „Wikipedia,“ [Online]. Available: https://de.wikipedia.org/wiki/AT-Befehlssatz. [Zugriff am März 2026].

[15] „Wikipedia,“ [Online]. Available: https://de.wikipedia.org/wiki/AT-Befehlssatz. [Zugriff am März 2026].

Listings

Formelsammlung