

AUTOTUNER

Gabriel Unterweger

Projektleiter: Martin Detomaso
Klasse: 5BEL
Schuljahr 2023/2024

Table of Contents

1 Einleitung.....	3
1.1 Vorwort.....	4
1.2 Abstract.....	5
1.2.1 Deutsch.....	5
1.2.2 Englisch.....	5
1.2.3 Italienisch.....	5
2 Projektüberblick.....	6
2.1 Projektbeschreibung.....	7
2.2 Projektantrag.....	8
2.3 Pflichtenheft.....	11
2.4 Gantt-Diagramm.....	13
3 Hardware.....	15
3.1 Mikrocontroller.....	16
3.2 ST – Link.....	16
3.3 Quarz.....	17
3.4 USB.....	18
3.5 Batterie.....	20
3.6 DCDC-Wandler.....	21
3.7 Klinkenstecker.....	22
3.8 Operationsverstärker.....	23
3.9 LCD – Display.....	24
3.10 LEDS.....	25
3.11 Buzzer.....	25
3.12 Motor.....	25
4 Hardware.....	26
4.1 Vorwort.....	27
4.2 Autodesk Eagle.....	27
4.3 Schaltplan.....	28
4.4 PCB.....	33
5 Software.....	34
5.1 Allgemeines.....	35
5.2 Flussdiagramm.....	35
5.3 Programmbeschreibung.....	36
5.3.1 Interrupt.....	36
5.3.2 ADC.....	37
5.3.3 Berechnung.....	38
5.4 Motorsteuerung.....	39
5.5 Menüstruktur.....	40
6 Gehäuse.....	41
6.1 Fusion 360.....	42
6.2 Schachtel und Deckel.....	42
6.3 Motoraufsatz.....	42
7 Projektmanagement.....	43
7.1 Arbeitsstunden.....	44

7.2 Beraterstunden.....	46
7.3 Materialkosten.....	47
7.3.1 PCB und Bestückte Bauteile.....	47
7.3.2 Nicht Bestückte Bauteile.....	48
7.4 Gesamtkostenrechnung.....	48
8 Schlusswort.....	49
9 Anhänge.....	50
9.1 Quellen.....	51
9.2 Datenblätter.....	55

1 Einleitung

1.1 Vorwort

An der TFO Meran hatten wir, statt ein 2-Wöchiges Praktikum, die Möglichkeit über das Ganze Schuljahr verteilt die Aufgabe ein Projekt selbst zu entwickeln. Im Fach TPS konnten wir 5 Stunden je Woche daran arbeiten.

Die Idee mich mit diesem Projekt außeinander zu setzen entstand durch 2-3 Wochen langen Überlegungen. Da ich sehr musikalisch bin passt ein Stimmgerät für eine Gitarre sehr zu mir. Ich habe sehr viel im Bereich Wechselstrom sowie Audientechnik gelernt und mich mit Frequenzberechnungen befasst.

Entstanden ist das Projekt als ein Mix mehrerer Projekte und Inspirationsquellen. Zuerst hatte ich die Idee die Frequenz der Gitarre Analog auszuwerten. Jedoch stellte ich gleich fest, dass ich dabei auf einige Schwierigkeiten stoßen könnte, wo mir klar wurde, dass die Digitale Auswertung mittels Microcontroller die bessere Lösung wäre.

Die Entwicklung des Projektes vollstreckte sich auf ein gesammtes Schuljahr was um die 35 Wochen waren. Dabei bin ich vor allem auf ein Problem gestoßen: Wie verarbeite ich das Signal, das aus der Gitarre kommt? Schwierig war es, die Störsignale des Netzbrummens sowie der Lichter, welche vom PickUp der Gitarre aufgenommen wurden, zu filtern.

Außerdem war der Mechanische Part, vor allem der Aufstecker für den Motor, eine relativ schwierige Aufgabe.

1.2 Abstract

1.2.1 Deutsch

Das Projekt "AutoTuner" ist ein E-Gitarren-Stimmgerät, das in ein Gehäuse verpackt ist und die Gitarrensaiten mithilfe eines Motors selbst stimmt. Das Wechselspannungssignal der angezupften Saite wird, wie üblich bei einer Gitarre, über einen 6-Millimeter-Audio-Jack entnommen und von einem Operationsverstärker so verstärkt, dass der Mikrocontroller ein gut gestrecktes Signal einlesen kann. Ein Algorithmus filtert mögliche Störungen heraus und ermittelt die Frequenz des Eingangssignals. Am Display kann man auswählen, welche Saite gestimmt werden soll. Dabei sieht man die erwartete und die aktuelle Frequenz der Saite.

1.2.2 Englisch

The “AutoTuner” project is an electric guitar tuner that is packaged in a housing and tunes the guitar strings itself using a motor. The alternating voltage signal of the plucked string is taken via a 6-millimeter audio jack, as is usual with a guitar, and amplified by an operational amplifier so that the microcontroller can read in a well-stretched signal. An algorithm filters out possible interference and determines the frequency of the input signal. You can select which string is to be tuned on the display. You can see the expected and current frequency of the string.

1.2.3 Italienisch

Il progetto “AutoTuner” è un accordatore per chitarra elettrica racchiuso in un involucro che accorda le corde della chitarra stessa mediante un motore. Il segnale di tensione alternata della corda pizzicata viene prelevato tramite un jack audio da 6 millimetri, come avviene di solito con la chitarra, e amplificato da un amplificatore operazionale in modo che il microcontrollore possa leggere un segnale ben disteso. Un algoritmo filtra le possibili interferenze e determina la frequenza del segnale in ingresso. Sul display è possibile selezionare la corda da accordare. È possibile visualizzare la frequenza prevista e quella attuale della stringa.

2 Projektüberblick



2.1 Projektbeschreibung

Der AutoTuner ist ein Stimmgerät, speziell entwickelt für elektrische Gitarren und für den Bass. Dieses Gerät stimmt die Saiten mithilfe eines Motors. Es funktioniert, indem es ein Signal von der Gitarre verarbeitet, das über den 6 mm Audio-Jack ausgegeben wird.

Die Operationsverstärkerschaltung, eine Schlüsselkomponente des AutoTuners, blockiert die negative Spannung des Wechselspannungssignals. Gleichzeitig verstärkt und spreizt sie das Signal, um eine optimale Auflösung zu gewährleisten. Die komplexen Algorithmen zur Signalauswertung wurden sorgfältig auf dem STM32 über das PlatformIO Framework programmiert.

Die Benutzerfreundlichkeit des AutoTuners wird durch ein intuitives Interface gewährleistet, das über drei einfach zu bedienende Tasten gesteuert und auf einem klaren LCD-Display dargestellt wird. Neben seiner Hauptfunktion, der Stimmung, bietet das Einstellungsmenü des AutoTuners verschiedene personalisierbare Optionen. Dazu gehören die Änderung der Grundstimmfrequenz und die Anpassung der Toleranz des Stimmvorgangs, um den individuellen Bedürfnissen jedes Musikers gerecht zu werden.

Darüber hinaus ist das Stimmgerät kompakt und tragbar, was es zu einem idealen Begleiter für Musiker macht, die häufig unterwegs sind. Es ist batteriebetrieben, was bedeutet, dass es keine ständige Stromquelle benötigt und daher auch unterwegs oder bei Auftritten verwendet werden kann.



2.2 Projektantrag

Projektantrag

Seite 1/2
Ansuchen für ein Maturaprojekt, Fassung September 2023

Projekttitel:	AutoTuner	
Projektleiter:	Gabriel Unterweger	
Klasse und Schuljahr:	5B EL	Schuljahr 2023 / 2024

Logo des Projektes:



Kurze Beschreibung des Projektes:

Ein Stimmgerät für die Gitarre, das die Saiten selbst stimmt.

Das Signal wird über einen 6-mm-Audio-Jack in das Stimmgerät eingespeist. Das Wechselspannungssignal wird verstärkt und anschließend von einem STM32 aufgenommen und mittels Schwellenerkennung ausgemessen. Die Saiten werden mithilfe von Motoren an der Gitarre selbst gestimmt.

Das Stimmgerät kann entweder batteriebetrieben oder direkt über ein Kabel mit Spannung versorgt werden.

Mit Hilfe von zwei Tastern kann die Referenzfrequenz, auf der das A gestimmt wird, eingestellt werden.

Die Auswahlmenü wird auf dem LCD-Display angezeigt. Der zu stimmende Ton wird außerdem auf einer analogen Anzeige (Zeigerinstrument / Voltmeter) angezeigt.

Das PCB (Leiterplatte) wird selbst hergestellt und enthält die Schaltung, den Operationsverstärker (OPV) und die Motorsteuerung. Außerdem wird ein Gehäuse selbst entworfen und gedruckt. Die Programmstruktur wird ebenfalls selbst entwickelt, allerdings wird das LCD-Display mithilfe einer Library gesteuert.

ungsrelevante Meilensteine des Projektes:

- Meilenstein 1: Verstärktes Signal auf dem Steckbrett (PW 3)
- Meilenstein 2: Verstärktes Signal im µC einlesen (PW 5)
- Meilenstein 3: Motorsteuerung auf dem Steckbrett funktioniert (PW 6)
- Meilenstein 4: Bauteile bestellen (PW 6-8)
- Meilenstein 5: PCB Prototyp (PW 10)
- Meilenstein 6: Signalauswertung mit µC auf PCB (PW 12)
- Meilenstein 7: Motoransteuerung mit µC (PW 15)
- Meilenstein 8: Benutzerinterface (PW 16)
- Meilenstein 9: Software Komplett (PW 18)
- Meilenstein 10: Redesign vom PCB (PW 24)
- Meilenstein 11: Motorenhalterung (PW 29)
- Meilenstein 12: Gehäuse fertiggestellt (PW 29)
- Meilenstein 13: Abgabe (PW 31)

Projektantrag

Seite 2/2

Anhänge:	<input type="checkbox"/> JA	() Seiten	<input checked="" type="checkbox"/> NEIN
----------	-----------------------------	---	----------	--

Projekt genehmigt:	<input checked="" type="checkbox"/> JA	<input type="checkbox"/> NEIN
--------------------	--	-------------------------------

Maximal erreichbare Note bei 100% Funktionalität: (unter Berücksichtigung der unten genannten Präzisierungen)	10 (Zehn)
---	-----------

Präzisierungen des PAG:

Note je nach Projekt - Fertigstellung

PAG 1: Martin De Tomaso

PAG 2: Ivan Huber



Der Projektleiter
Gabriel Unterweger

2.3 Pflichtenheft

Pflichtenheft

AutoTuner



Funktion meines Projektes:

Der AutoTuner ist ein Stimmgerät für Gitarren, welches die Saiten selbst stimmt.

Eingespeist wird ein von der Gitarre kommendes Sinus-Signal, welches über einen 6 mm Audio-Jack gespeist wird. Das Wechselspannungssignal wird mit einem Operationsverstärker verstärkt und anschließend von einem Mikrocontroller ausgelesen.

Die Berechnung der Schwellen erfolgt wie folgt:

Der Mikrocontroller liest eine gewisse Zeit X und berechnet aus den Eingelesenen Daten die Schwelle, welche zur Berechnung der Frequenz hergenommen wird. Die berechnete Frequenz wird mit einem Lookup Table verglichen und daraus der Ton ausgelesen. Am Stimmmechanismus der Gitarre werden Motoren angebracht, welche die Gitarre automatisch stimmen.

Komponenten:

- stm32f103c8t6
- LCD - Display
- 6mm - Audio - Jack
- Zeigerinstrument
- 3,7 V - Batterie
- 6 Schrittmotoren
- TPS63031DSKR - Voltage Converter
- USB - C
- TP4054 - Batterie Charger
- OPV
- Oscillator
- Battery Charger Mosfet
- Taster
-
-
-
-
-
-
-
-
-
-

I/O:

Das Stimmgerät kann entweder batteriebetrieben oder direkt über ein Kabel mit Spannung versorgt werden.

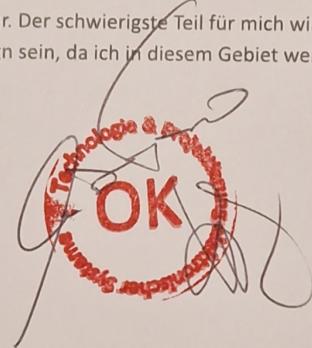
Mit den Tasten kann durch das Programm Interface navigiert werden. Das Interface, so wie der Ton, der erklingt, wird auf dem LCD- Display angezeigt. Dabei kann die gewünschte Referenzfrequenz eingestellt werden. Außerdem wird mit einer analogen Voltanzeige die Feinstimmung veranschaulicht.

Testen:

Alle Grundfunktionen werden getrennt am Steckbrett aufgebaut und getestet, um zu verifizieren, dass es funktionieren könnte.

Fazit:

Ich denke, das Projekt ist schaffbar. Der schwierigste Teil für mich wird die Planung der mechanischen Halterung und das Gehäuse-Design sein, da ich in diesem Gebiet wenig Ahnung habe.

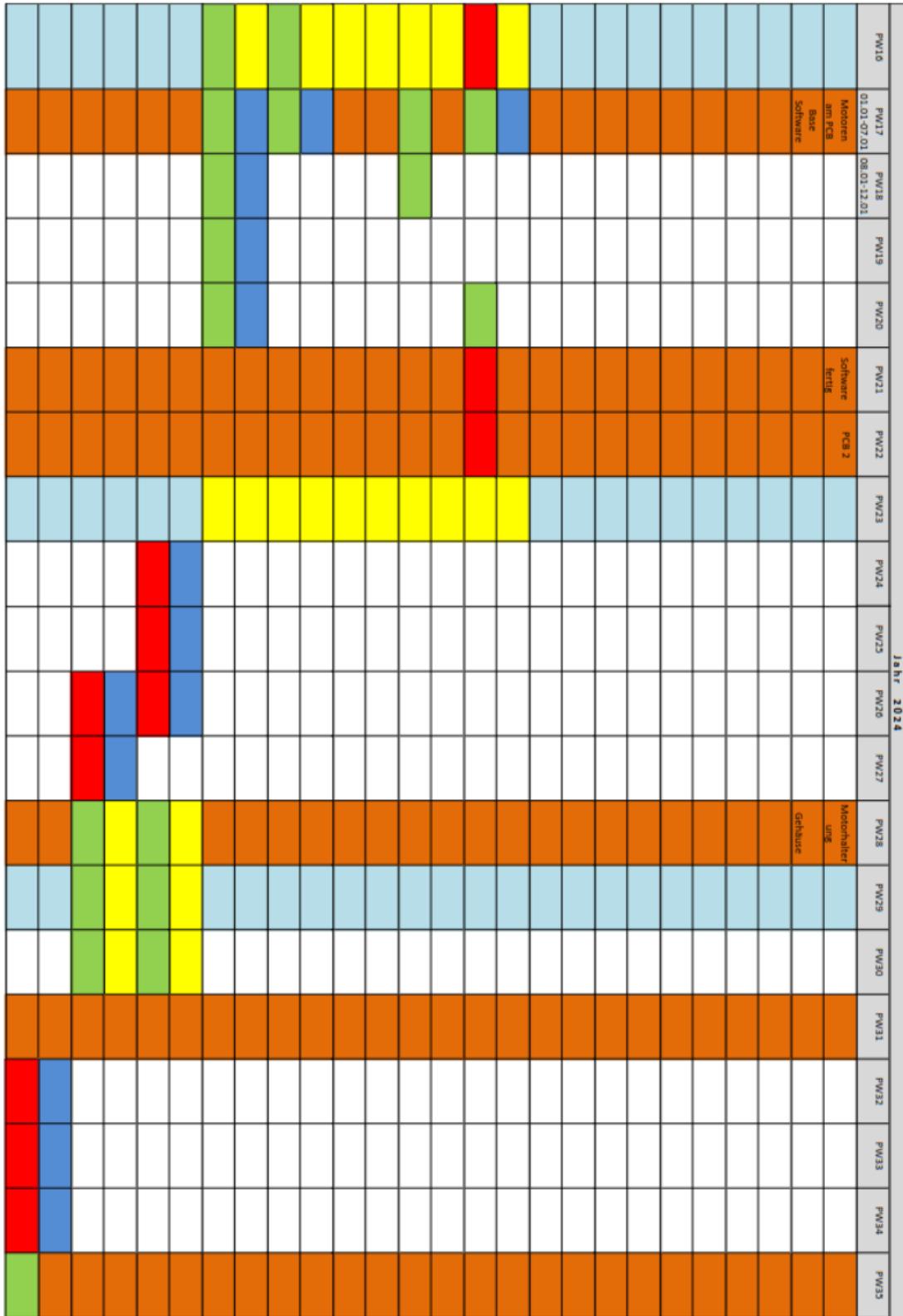


Projekt "AutoTuner" - 5BEL - Gabriel Unterweger

	PW1	PW2	PW3	PW4	PW5	PW6	PW7	PW8	PW9	PW10	PW11	PW12	PW13	PW14	PW15
	Jahr 2023														
WP1: Projektsuche							Stckbrett		Signal im MC		PCB 1				
WP2: Projektantrag															
WP3: Pflichtenheft/GANT Diagramm erstellen															
WP4: Signalverarbeitung (Steckbrett)															
WP5: Motorsteuerung (Steckbrett)															
WP6: PCB erstellen															
WP7: PCB Bestücken und Fehlersuche															
WP8: Signalberechnung Programmieren															
WP9: Motortsteuerung Programmieren															
WP10: Menüstruktur Programmieren															
WP11: Motorkalibrierung															
WP12: Gehäuse erstellen															
WP13: Dokumentation															



Ober Zelle: In Blau wird die Zeit eingezeichnet, die für den jeweiligen Working Package vorgesehen ist.
Pufferzeit (Reserve)
Untere Zelle: Vom zu handeln nachstrafen. Grün: Ok, im Zeitplan Rot: Über dem Zeitplan
Meilenstein
Fertigen



3 Hardware

3.1 Mikrocontroller

Der STM32F103 ist ein Mikrocontroller, der auf dem ARM Cortex-M3-Kern basiert. Er hat eine maximale CPU-Geschwindigkeit von 72 MHz. Sein Portfolio deckt einen Flash-Speicherbereich von 16 KByte bis 1 MByte ab.

Die Modellbezeichnungen des Herstellers folgen einem bestimmten Schema. Der verwendete STM32F103C8T6 hat 48 Pins, 64 kB Flash, 16kB RAM, ein QFP-Gehäuse und ist für Temperaturen von -40 bis +85 °C ausgelegt.

Die STM32F103-Mikrocontroller sind in verschiedenen Modellen erhältlich, die sich in der Anzahl der Pins, der Größe des Flash-Speichers, der Größe des RAMs, der Art des Gehäuses und dem Temperaturbereich unterscheiden. Die Modellnummer gibt Aufschluss über diese Eigenschaften.

Die STM32F103-Mikrocontroller sind Teil der Mainstream-Familie, zu der auch die STM32F0-Serie (ARM Cortex M0 bis 48 MHz), die STM32G0-Serie (ARM Cortex M0+ bis 64 MHz) und die STM32F1-Serie (ARM Cortex M3 bis 72 MHz) gehören.

Die STM32F103-Mikrocontroller verfügen über eine 12-Bit-ADC, eine Motorsteuerung, USB und CAN.

Für die Programmierung der STM32F103-Mikrocontroller ist das Reference Manual am wichtigsten, da es die I/O-Funktionen und Register beschreibt. Im Errata Sheet beschreibt der Hersteller überraschende Einschränkungen und Fehler der Mikrochips, teilweise mit konkreten Workarounds.



Abbildung 1: STM32 Microcontroller-Chip

3.2 ST – Link

Der ST-Link ist ein In-Circuit-Debugger und Programmierer, der speziell für die STM8- und STM32-Mikrocontroller-Familien entwickelt wurde, einschließlich des STM32F103. Er ermöglicht es den Entwicklern, ihre Anwendungen direkt auf der Zielhardware zu debuggen und zu programmieren.

Der ST-Link unterstützt sowohl das Single-Wire-Interface-Modul (SWIM) als auch JTAG/Serial Wire Debugging (SWD) Protokolle. SWIM wird für STM8-Mikrocontroller verwendet, während JTAG/SWD für STM32-Mikrocontroller verwendet wird.



Abbildung 2: ST-LINK Programmiergerät

Der ST-Link ist in verschiedenen Formen erhältlich, einschließlich eigenständiger Programmierer, eingebettet in STM32 Discovery- und Nucleo-Boards, und als Add-On-Board für bestehende Hardware.

Der ST-Link ist mit der ST-LINK/V2 in-circuit debugger/programmer Firmware kompatibel, die eine einfache Upgrade-Funktion bietet. Dies ermöglicht es den Benutzern, ihre Hardware auf dem neuesten Stand zu halten und von Verbesserungen und neuen Funktionen zu profitieren.

Zusätzlich zur Hardware bietet STMicroelectronics auch die ST-LINK-Server-Software an. Diese Software ermöglicht den Fernzugriff auf den ST-Link über TCP/IP. Dies ist besonders nützlich für Entwickler, die ihre Anwendungen aus der Ferne debuggen und programmieren möchten.

3.3 Quarz

Ein Quarz ist ein Mineral, das in der Natur vorkommt und in der Industrie als Rohstoff für die Keramik-, Glas- und Zementindustrie weltweite Bedeutung hat. In Quarzuhrnen und elektronischen Schaltungen wird jedoch nicht das Mineral selbst, sondern sogenannte Schwingquarze als Taktgeber verwendet.



Ein Schwingquarz ist ein wesentlicher Bestandteil in der Welt der Mikrocontroller und dient als Taktgeber, der die Geschwindigkeit *Abbildung 3: Schwingquarz in THT Form* der Prozesse im Mikrocontroller bestimmt. Die Verwendung eines Quarzes in einem Mikrocontroller-System bietet eine Reihe von Vorteilen, darunter eine hohe Genauigkeit, eine geringe Temperaturabhängigkeit und eine Langzeitstabilität besser als 0,0001%.

Ein Schwingquarz erzeugt eine Schwingung, wenn eine elektrische Spannung angelegt wird. Diese Schwingung ist sehr stabil und genau, was sie ideal für die Verwendung als Taktgeber in Mikrocontrollern macht. Die Frequenz dieser Schwingung, auch als Resonanzfrequenz bezeichnet, ist fast unabhängig von Umgebungseinflüssen wie Temperatur oder Amplitude.

In Ihrem speziellen Fall erwähnen Sie die Verwendung eines 8 MHz und eines 32,768 kHz Quarzes. Beide Quarze haben ihre spezifischen Anwendungen und Vorteile.

Ein 8 MHz Quarz wird oft in Mikrocontrollern verwendet, da er eine hohe Taktfrequenz bietet, die für viele Anwendungen ausreichend ist. Diese "runden" Frequenzen sind meist leichter erhältlich und haben den Vorteil, dass man Verzögerungsschleifen und Rechendauern relativ leicht errechnen kann. Es ist wichtig zu beachten, dass die Kapazitäten von C1 und C2, die mit dem Quarz verbunden sind, nicht der Lastkapazität des Quarzes entsprechen.

3.4 USB

Die USB-Schnittstelle, auch bekannt als Universal Serial Bus, ist ein bit-serielles Datenübertragungssystem, das zur Verbindung eines Computers mit externen Geräten dient. Sie wurde 1996 mit einer maximalen Datenübertragungsrate von 12 Mbit/s als USB 1.0 eingeführt. Im Jahr 2000 wurde die Version USB 2.0 mit 480 Mbit/s spezifiziert. Bei dem 2014 eingeführten Standard USB 3.1 Gen 2 beträgt die maximale Brutto-Datenübertragungsrate für SuperSpeed+ 10 Gbit/s. 2017 wurde USB 3.2 mit einer Übertragungsrate bis zu 20 Gbit/s spezifiziert.

Einige der Hauptmerkmale der USB-Schnittstelle sind:

Hot Plugging: USB-Geräte können im laufenden Betrieb miteinander verbunden und erkannt werden.

Kompatibilität: Fast alle früheren Schnittstellenvarianten wurden durch USB ersetzt, was für die Anwender Vereinfachungen mit sich brachte.

Baum-Topologie: Trotz des Begriffs „Bus“ in der Bezeichnung „Universal Serial Bus“ verwendet USB eine Baum-Topologie mit dem Root-Hub als Wurzel.

In Bezug auf die Spannungsversorgung versorgt der USB einfache Geräte, wie Maus, Tastatur, USB-Sticks und Festplatten mit Strom. Die Leistung reicht von 100 mW bis 15 Watt, abhängig vom USB-Standard.

Es ist wichtig zu beachten, dass die Spannung auf USB beim Anschließen eines Gerätes stets 5 V beträgt, kann aber nach Verhandlungen zwischen Gerät und Host mittels seriellen Protokolls auf 12 V oder 20 V erhöht werden. Ein Endgerät, z. B. ein Smartphone muss dem Netzteil immer aktiv mitteilen, welche Kombination aus Spannung und Stromstärke gefordert ist. Sonst ist nur USB 2.0 mit 5 Volt und 0,5 Ampere (500 mA) erlaubt.



Abbildung 4: USB-B Kabel

3.5 Batterie

Eine Batterie ist ein elektrochemischer Energiespeicher. Sie speichert elektrische Energie auf elektrochemischer Basis. Bei der Entladung wird gespeicherte chemische Energie durch die elektrochemische Redoxreaktion in elektrische Energie gewandelt. Die umgewandelte Energie kann von einem vom Stromnetz unabhängigen elektrischen Verbraucher genutzt werden.

In nicht wiederaufladbaren Primärbatterien sind die Reaktionen bei der Entladung nicht oder nur teilweise umkehrbar. Dagegen sind in wiederaufladbaren Sekundärbatterien (Akkumulatoren) die Entladereaktionen weitgehend umkehrbar, sodass eine mehrfache Umwandlung von chemischer in elektrische Energie und zurück möglich ist.

Die Elektrodenmaterialien legen die Nennspannung der Zelle fest. Höhere Spannungen erhält man durch ein Hintereinanderschalten (Reihenschaltung) mehrerer Zellen. Die Kapazität einer Batterie wird als theoretisch entnehmbare Ladungsmenge in Amperestunden (Einheit: Ah) angegeben.

Die entnehmbare Kapazität hängt vom Entladestrom und der Entladeschlussspannung der Batterie ab. Generell nimmt die entnehmbare Kapazität einer Batterie mit zunehmendem Entladestrom ab. Die Verringerung der entnehmbaren Kapazität mit zunehmendem Entladestrom ist stark vom Typ der Batterie abhängig.

Die im praktischen Gebrauch entnehmbare Ladungsmenge hängt ab vom Batterietyp, der Höhe des Entladestroms, der Restspannung bei Entladungsende, des Batteriealters und der Temperatur. Die Batteriekapazität oder der Maximalstrom bei gegebener Spannung lassen sich durch größer gebaute Zellen oder durch Parallelschaltung von Zellen oder Batterien erhöhen.

In meinem Fall schließe ich zwei LIPO Batterien mit jeweils 3,7V Maximalspannung in Serie, um auf maximal 8,4V hinaufzukommen.



Abbildung 5: 3.7V 500mA Batterie

3.6 DCDC-Wandler

Der Step-Down-Converter, auch als Buck Converter bekannt, ist ein DC-DC-Wandler, der die Eingangsspannung verringert und gleichzeitig den Strom erhöht. Er besteht aus mindestens zwei Halbleitern (einer Diode und einem Transistor) und mindestens einem Energiespeicherelement (einem Kondensator, einer Induktivität oder einer Kombination aus beiden).

Der Boost-Converter ist eine Art DC-DC-Wandler, dessen Ausgangsspannung größer als die Eingangsspannung ist. Er ist äquivalent zu einem Flyback-Wandler, verwendet jedoch anstelle eines Transformators eine einzige Induktivität.

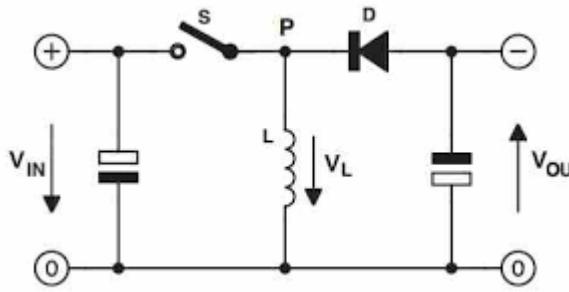


Abbildung 7: Buck Boost als Schaltprinzip

benötigt.

Um die Batteriespannung von maximal 8.4V auf 5V zu reduzieren und von nur einer Batterie (3.7V) auf 5V zu erhöhen, habe ich mich für einen Buck-Boost-Wandler entschieden. Des Weiteren benötigt der Microcontroller eine Spannung von 3V3 weshalb ich auf ein zusätzlichen Linearregler von 5V auf 3V3 Volt gesetzt habe.

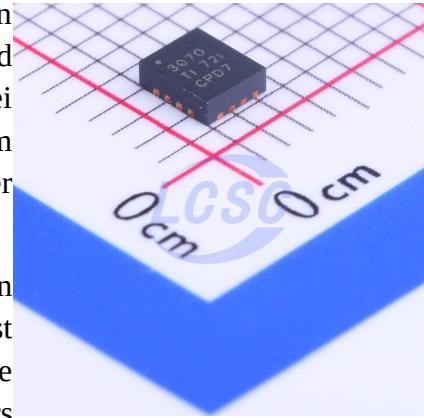


Abbildung 6: Buck Boost Converter

Der Webench Power Designer von Texas Instruments ist ein Online-Tool, mit dem man DC-DC sowie AC-DC Schaltungen erstellen kann.

Als Eingangsspannung kann entweder der USB-B mit den jeweiligen 5V oder die Batterie, welche zwei 3.7V LIPO-Zellen parallel sind, angeschlossen werden. Die Batteriespannung wird für die Motorsteuerung benötigt.

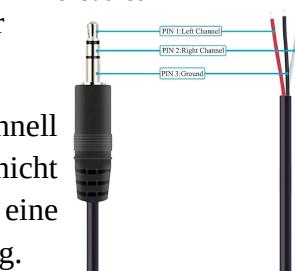
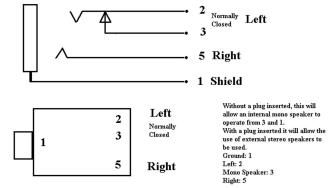
3.7 Klinkenstecker

Ein Klinkenstecker ist ein weit verbreiteter elektrischer Steckverbinder, der zur Übertragung von kleinen elektrischen Signalen und großen Leistungen bei Endstufen in der Bühnentechnik oder einer reinen Wechsel- oder Gleichspannung im Kleinspannungsbereich dient.

Die Spitze und die Hülle des zweipoligen Klinkensteckers entsprechen dem Innenleiter (Signal) und dem Außenleiter (Masse) des Kabels. Bei mehrpoligen Steckern entspricht jeder Ring einem weiteren Innenzylinder als Kontakt.

Klinkenstecker werden mit verschiedenen Schaftdurchmessern hergestellt: 2,5 mm für besonders kleine Geräte, wie Headsets für Mobiltelefone. 3,5 mm wird häufig an Kopfhörern und Handys gefunden. 6,35 mm wird für Audiogeräte oder HiFi-Anlagen verwendet.

Einige Vorteile von Klinkensteckern sind die einfache Handhabung, die schnell lösbare Verbindung und die platzsparende Bauform. Nachteile sind die nicht berührungssicheren Kontakte, bei Handhabung unter Spannung eine Kurzschlussgefahr und die Kontaktmängel bei nachlassender Federspannung.



3.8 Operationsverstärker

Ein Operationsverstärker (OPV oder Op-Amp) ist ein elektronisches Bauteil, das in analogen Schaltungen vielseitig einsetzbar ist. Ein OPV ist ein integrierter Schaltkreis, der eine hohe Verstärkung und vielseitige Funktionalität bietet. Er verfügt über zwei Eingänge, den invertierenden (-) und den nicht-invertierenden (+) Eingang, sowie einen Ausgang. Die Differenzspannung zwischen den beiden Eingängen wird verstärkt und am Ausgang ausgegeben. OPVs werden in verschiedenen Bereichen eingesetzt, darunter Signalverstärkung, Filterung, Oszillatoren, Spannungsregelung und mathematische Operationen wie Addition, Subtraktion, Integration und Differentiation. Sie sind grundlegende Bausteine in der Analogelektronik und finden sich in vielen elektronischen Geräten und Systemen. Für die Schaltung verwende ich eine externe Beschaltung, die den nicht-invertierenden Verstärker ergibt. Dabei ist eine Verstärkung von $V_u = 1,1$ bis theoretisch ∞ möglich.

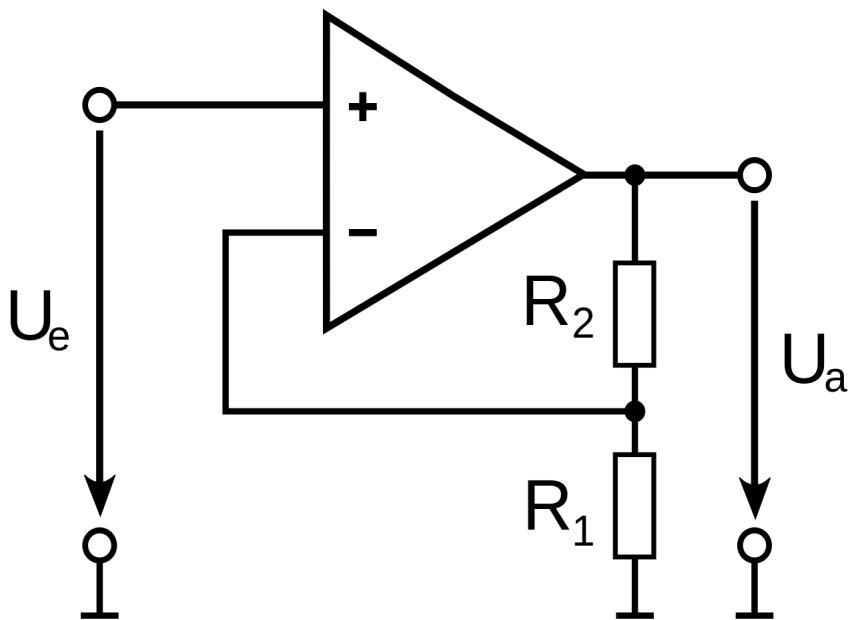


Abbildung 11: Nicht-Invertierender Verstärker

3.9 LCD – Display

Ein LCD-Display, auch bekannt als Liquid Crystal Display, basiert auf sogenannten Flüssigkristallen. Diese ändern ihre Polarisationsrichtung, wenn eine elektrische Spannung angelegt wird, und werden dadurch undurchsichtig.

Die Flüssigkristalle in einem LCD-Display sind in Segmenten angeordnet und können unabhängig voneinander ihre Transparenz ändern. Sie finden Anwendung in vielen Geräten wie digitalen Armbanduhren, Drucker-Displays, Thermometern oder im Auto.



Abbildung 12: LCD-Display

Ein Element der Anzeige besteht aus einer Flüssigkristallschicht zwischen zwei transparenten Elektroden und zwei Polarisationsfiltern. Die Filter lassen nur die Lichtwellen mit einer bestimmten "Ausrichtung" durch.

LCD-Displays können in verschiedenen Formen und mit unterschiedlichen Segmentanordnungen vorliegen, häufig als Pixelraster. Im uC-Bereich sind mehrzeilige alphanumerische LCD-Displays mit 16 Pin Connector beliebt. Oft wird ein Treiber-IC wie z.B. der HD44780 Chip von Hitachi verwendet, um über 4 Pins und mit der Arduino-Library "LiquidCrystal" das Display anzusprechen.

Solch ein LCD-Display kann im 4 oder 8 Bit Modus angesprochen werden. Je nach Display-Typ gibt es einen Anschluss für die Hintergrundbeleuchtung und ein Potentiometer, um den Anzeigekontrast zu regeln.

3.10 LEDs

LEDs (Light-emitting diode) sind Halbleiterbauelemente, die Licht erzeugen, wenn elektrischer Strom durch sie fließt. Sie sind energieeffizient, langlebig und in verschiedenen Farben erhältlich. Für die Leds wurden 5mm Leds aus dem Arduino Kit genommen.



Abbildung 13: 5mm LED

3.11 Buzzer

Ein passiver Buzzer erzeugt einen Ton durch die Anregung einer internen piezoelektrischen Scheibe oder Membran, die zu Schwingungen gebracht wird. Im Gegensatz zu einem aktiven Buzzer, der eine eingebaute Schaltung zur Erzeugung eines Tons besitzt, benötigt ein passiver Buzzer ein PWM Signal, um zu vibrieren und Schallwellen zu erzeugen. Diese externen Signale werden typischerweise von einer Mikrocontroller-Schaltung oder einem anderen Oszillatoren bereitgestellt. Die Frequenz des angelegten Signals bestimmt die Tonhöhe des erzeugten Tons.



Abbildung 14: Passiver Buzzer

Für den passiven Buzzer, der über eine Transistor-Schalterschaltung gesteuert wird, wird ein einfacher Buzzer aus dem Arduino-Kit verwendet.

3.12 Motor

Ein Gleichstrommotor (DC-Motor) wandelt elektrische Energie in mechanische Bewegung um. Er besteht aus einem festen Teil (Stator) und einem drehbaren Teil (Rotor). Wenn Strom durch den Rotor fließt, erzeugt er ein Magnetfeld, das mit dem Magnetfeld des Stators interagiert. Diese Interaktion erzeugt eine Kraft, die den Rotor dreht. Ein Kommutator sorgt dafür, dass der Stromfluss ständig umgeschaltet wird, sodass der Rotor sich kontinuierlich dreht. So entsteht eine gleichmäßige Rotationsbewegung.

Als Motor des Projekts wurde ein speziell ausgewählter 6V Gleichstrommotor mit einem integrierten Getriebe verwendet. Der Motor ist mit einer H-Brücke gesteuert, was eine Kontrolle über die Drehrichtung des Motors ermöglicht.



Abbildung 15: DC Motor mit Getriebe

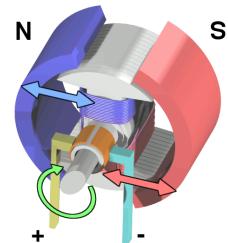


Abbildung 16: DC Motor - Innenaufbau

4 Hardware

4.1 Vorwort

Um das Projekt zu realisieren, wurde eine Platine selbst entworfen und bei der chinesischen Firma JLCPCB bestellt. Die Schaltung des Mikrocontrollers wurde direkt auf der Platine gezeichnet, wobei ich den Schaltplan vom Hersteller STMicroelectronics als Vorlage nutzte. Dieser wurde nachgezeichnet, um eine korrekte Funktion zu gewährleisten. Die Anschlüsse für die Batterien und den Motor wurden so gestaltet, dass sie ein- und ausgesteckt werden können. Auch das LCD-Display wurde steckbar gemacht.

Bei der Gestaltung des DC-DC-Wandlers auf einer Platine gibt es viele Aspekte zu berücksichtigen. Die Erstellung eines Platinenlayouts ist ein kreativer und technischer Prozess. Der Entwickler legt die physische Anordnung der Komponenten und Leiterbahnen auf der Leiterplatte fest. In diesem Fall wurde der Power Wrench Designer von Texas Instruments als Hilfsmittel verwendet. Dabei sollten zusammengehörige Bauteile möglichst nebeneinander platziert werden. Die Leiterbahnen sollten möglichst kurz und kreuzungsfrei gehalten werden.

Die Masseführung war beim PCB-Design ebenfalls ein wichtiger Aspekt, da ich den Motor als induktiven Verbraucher auf der Platine habe. Dieser kann leicht Störungen verursachen, insbesondere Massestörungen, die den Mikrocontroller beeinträchtigen können, der empfindlicher ist. Daher habe ich die einzelnen Baugruppen getrennt und jeweils mit einer einzigen Masseleitung verbunden, um die Störungen zu minimieren.

4.2 Autodesk Eagle

Autodesk EAGLE (Easily Applicable Graphical Layout Editor) ist eine Software für das Design von Leiterplatten (PCBs, Printed Circuit Boards), die von Ingenieuren, Elektronikdesignern und auch von Hobbyisten verwendet wird. Sie bietet einen Schematic Editor zum Erstellen und Bearbeiten von Schaltplänen und einen Layout Editor zur Platzierung und Verdrahtung von Bauteilen auf der Leiterplatte. EAGLE enthält umfassende Bauteilbibliotheken und unterstützt die Erstellung eigener Bauteile. Ein Design Rule Check (DRC) stellt sicher, dass das Design den festgelegten Entwurfsregeln entspricht, um Herstellungsfehler zu vermeiden. Die Software ermöglicht die Ausgabe von Gerber-Dateien, die für die Fertigung von Leiterplatten benötigt werden, und unterstützt Skripte sowie User Language Programs (ULPs) zur Erweiterung des Funktionsumfangs und Automatisierung spezifischer Aufgaben. Mit einer intuitiven Benutzeroberfläche ist EAGLE sowohl für Anfänger als auch erfahrene Designer geeignet und wird oft für Prototyping und Kleinserienfertigung verwendet. Es ist in verschiedenen Lizenzmodellen erhältlich, einschließlich einer kostenlosen Version für eingeschränkte Nutzung und kostenpflichtigen Versionen mit erweitertem Funktionsumfang.

4.3 Schaltplan

Die einzelnen Funktionsgruppen wurden sorgfältig auf vier verschiedene Sheets aufgeteilt, um eine strukturierte und übersichtliche Anordnung zu gewährleisten. Sheet 1 spielt eine zentrale Rolle, da es alles rund um den Mikrocontroller beinhaltet, der das Herzstück des Systems bildet. Sheet 2 ist ebenso wichtig und beinhaltet alle Spannungswandler, die für die Energieversorgung des Systems unerlässlich sind. Darüber hinaus findet sich auf diesem Sheet auch die speziell entworfene Schaltung zum Laden der Batterien sowie die verantwortliche Schaltung für die Spannungsauswahl. Auf Sheet 3 befinden sich die Ein- und Ausgänge der Platine, die eine Schnittstelle nach außen darstellen und für die Kommunikation des Systems mit anderen Geräten unerlässlich sind. Schließlich beinhaltet Sheet 4 die H-Brücke der Motorsteuerung, eine kritische Komponente für die präzise Steuerung der Motoren, sowie die Anpassungsschaltung des Gitarreneingangssignals, die die korrekte Interpretation des Eingangssignals der Gitarre gewährleistet.

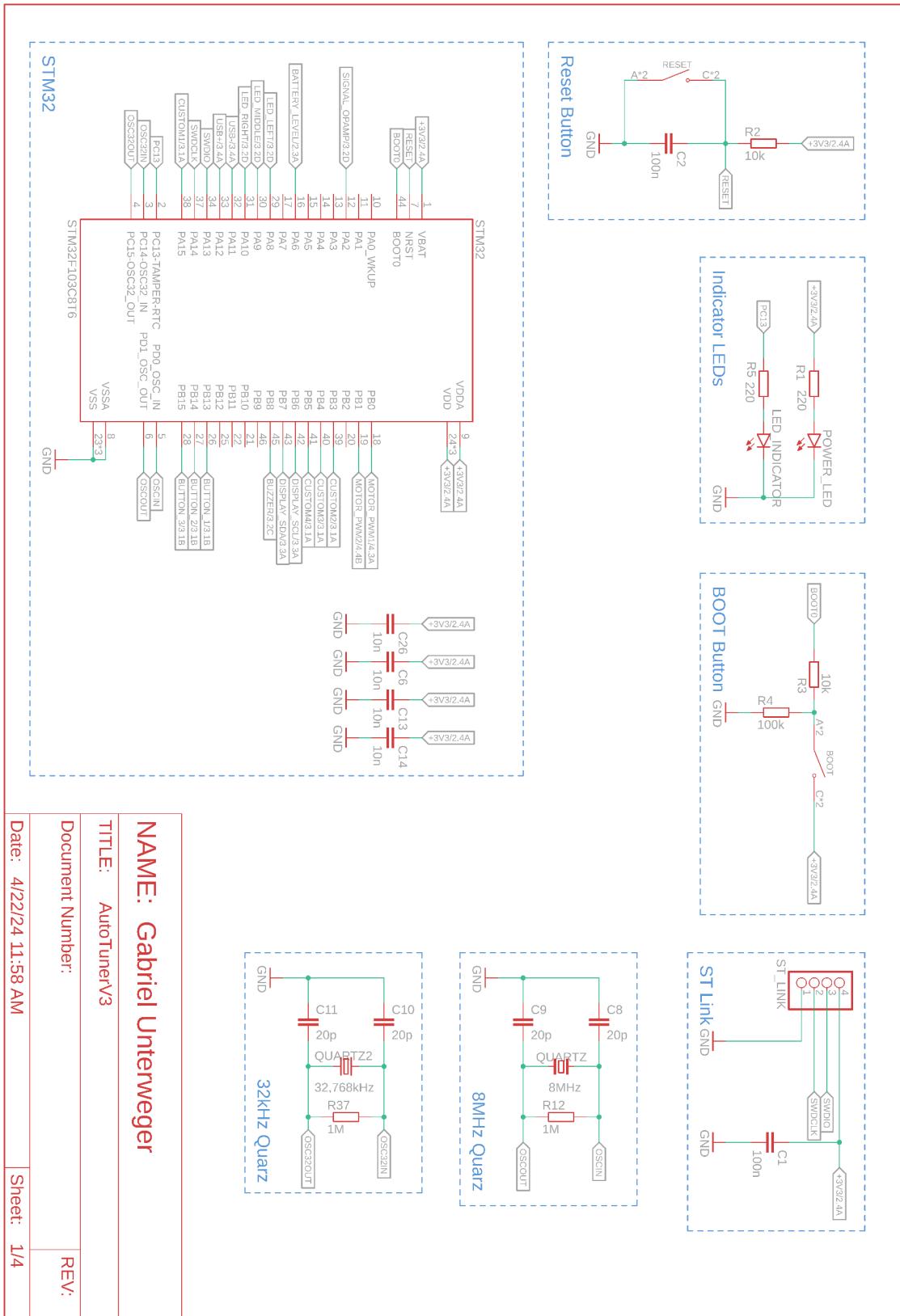


Abbildung 17: Schematic - Sheet 1

NAME:	Gabriel Unterweger
TITLE:	AutoTunerV3
Document Number:	
REV:	
Date:	4/22/24 11:58 AM
Sheet:	1/4

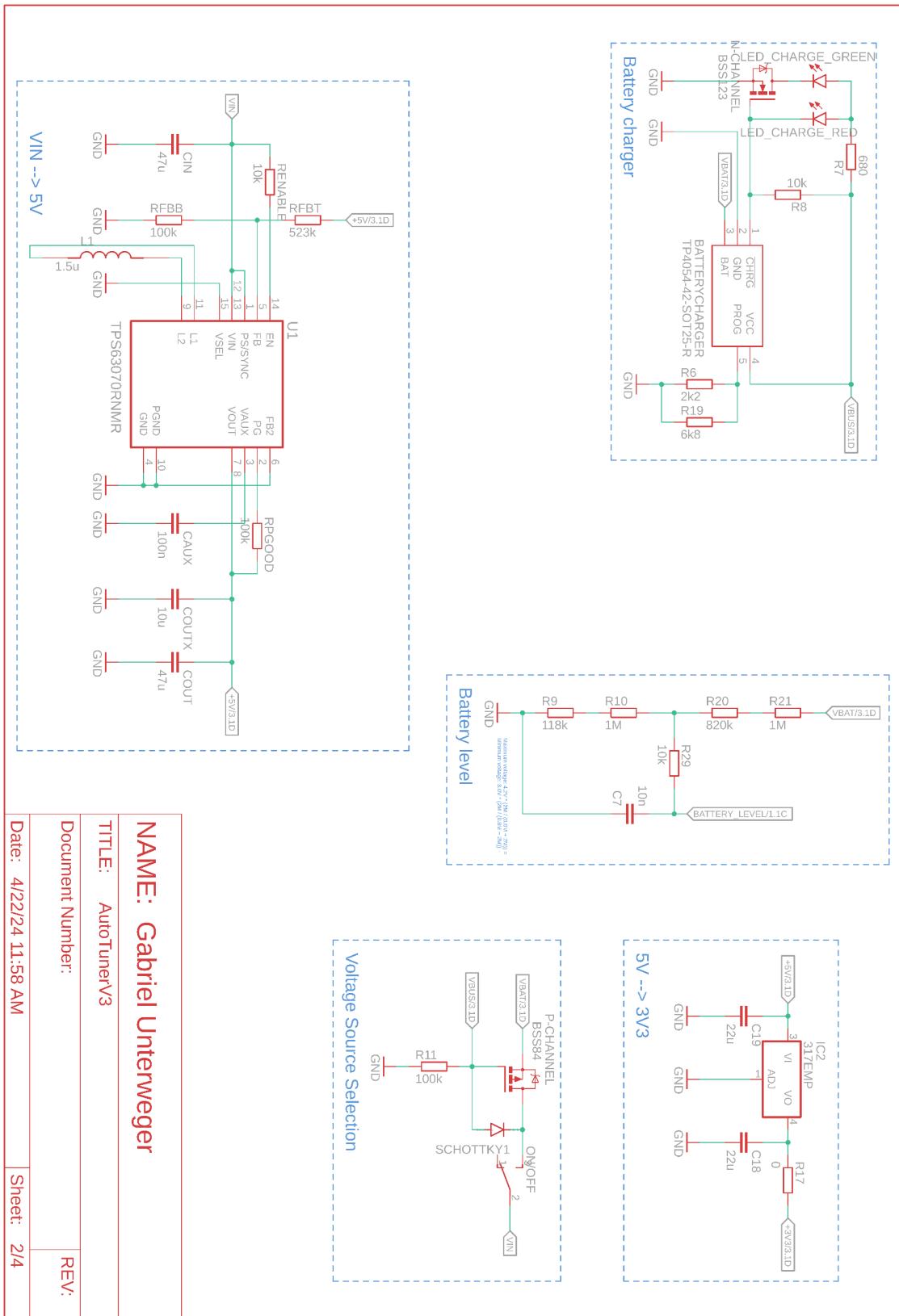


Abbildung 18: Schematic - Sheet 2

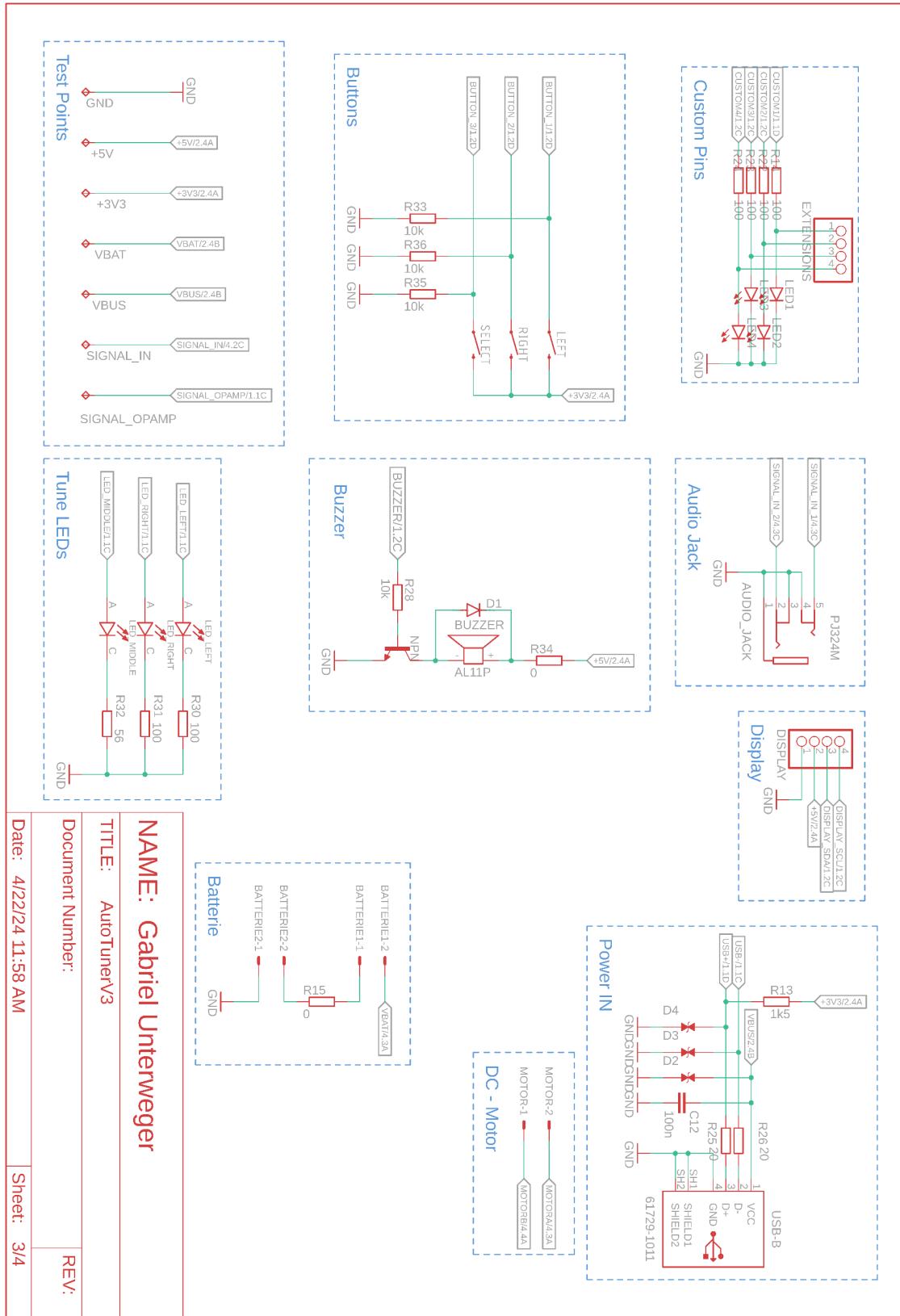


Abbildung 19: Schematic - Sheet 3

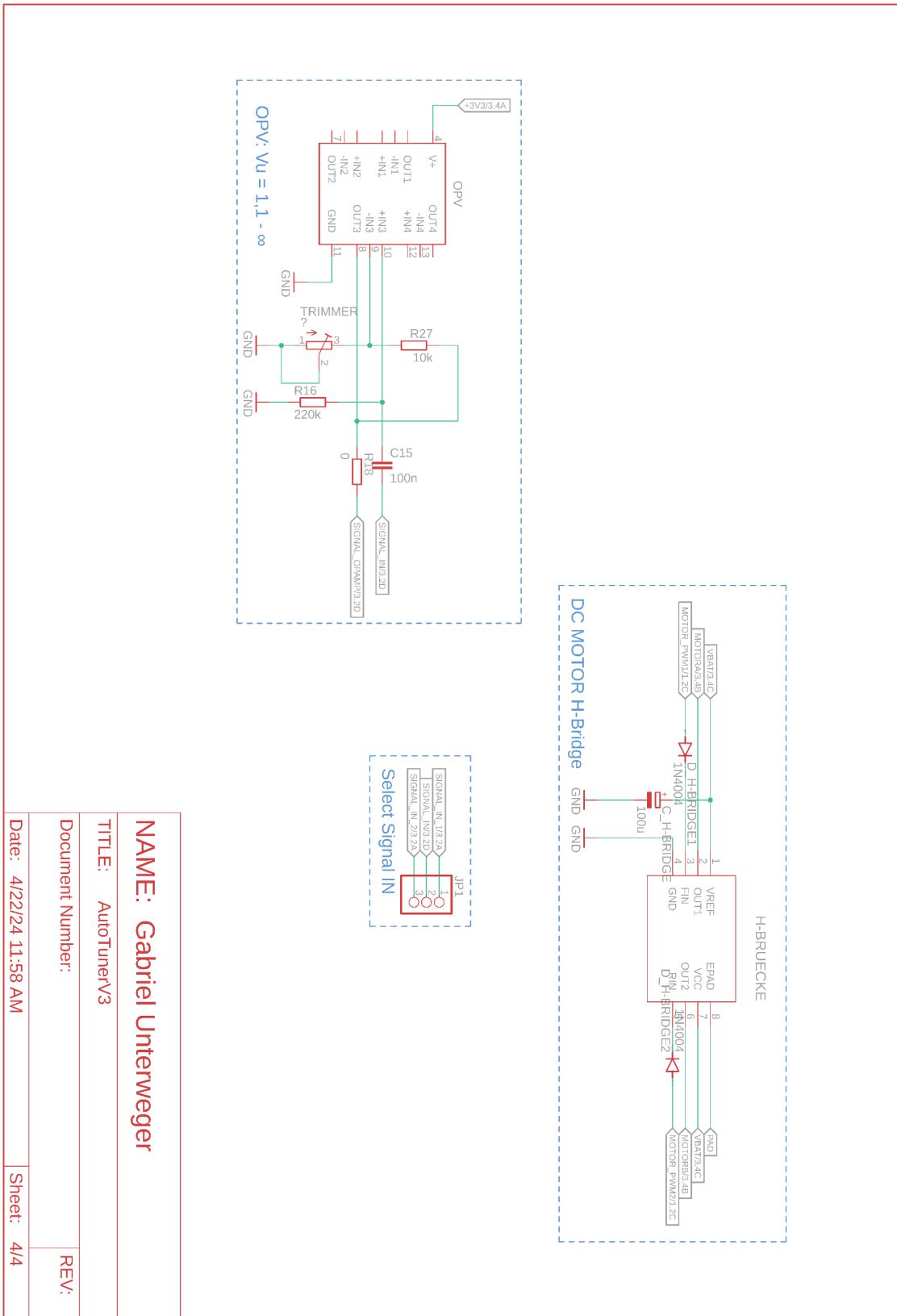


Abbildung 20: Schematic - Sheet 4

4.4 PCB

Für mich war seit Anfang klar, dass ich die Platine bestücken lassen möchte, was bedeutet, dass ich alle Komponenten auf TOP plazieren muss.

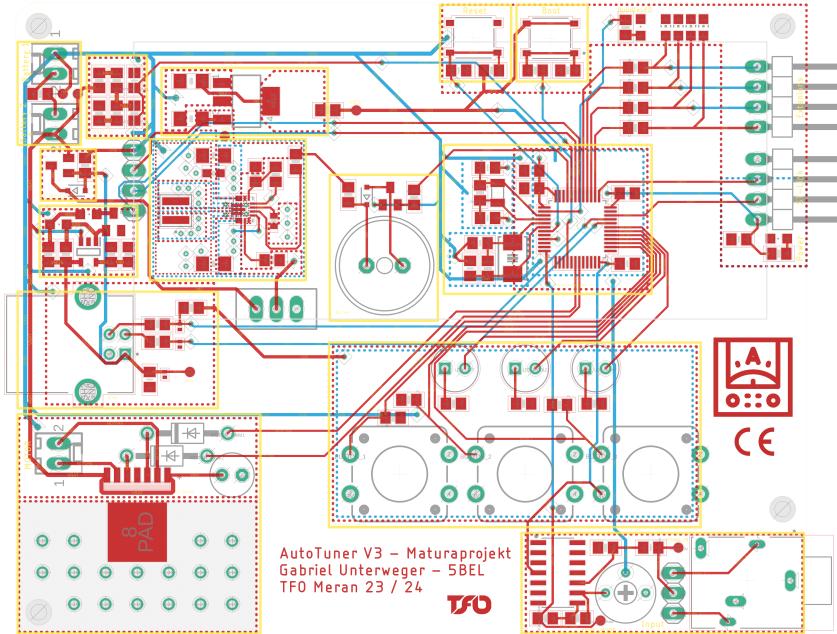


Abbildung 21: PCB Layout der 3. Version der Platine

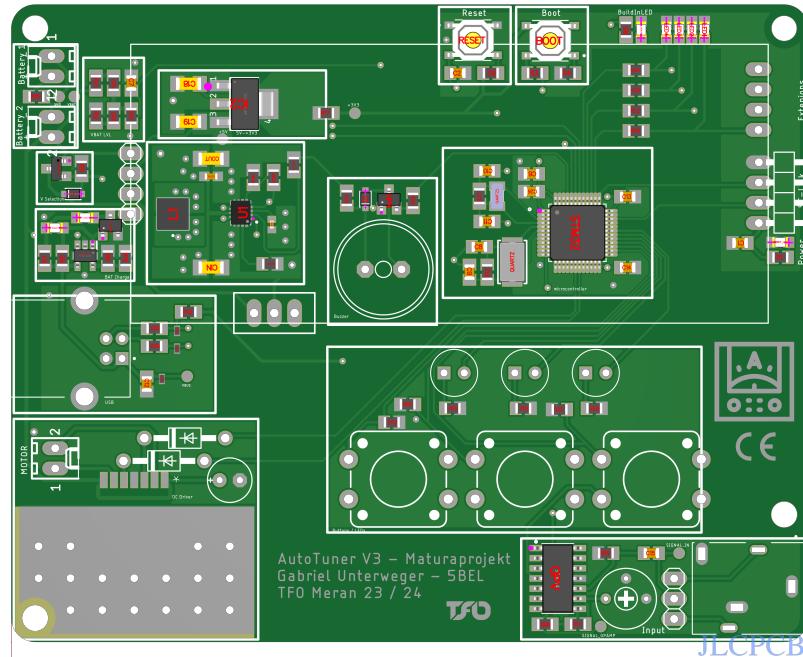


Abbildung 22: JLCPCB Part Placement der 3. Version der Platine

5 Software

5.1 Allgemeines

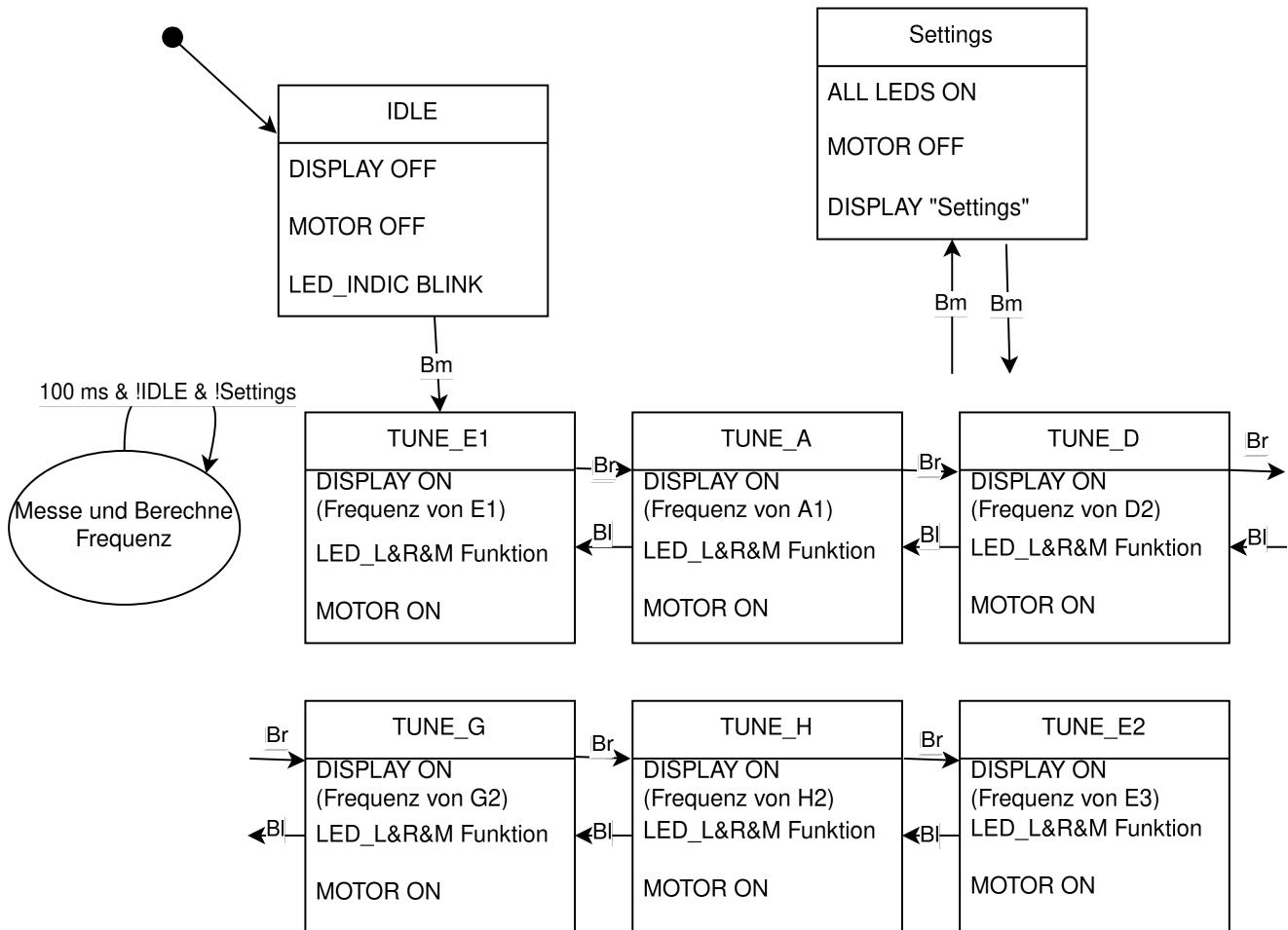
Das auf den STM32-Mikrocontroller programmierte Programm wurde mit Hilfe des PlatformIO-Frameworks erstellt, einer leistungsstarken Entwicklungsumgebung für eingebettete Systeme, in der Programmiersprache C++. Als integrierte Entwicklungsumgebung (Integrated Development Environment, IDE) wurde CLion von JetBrains verwendet, eine hochmoderne, plattformübergreifende IDE, die für Effizienz und Benutzerfreundlichkeit entwickelt wurde.

Während des gesamten Entwicklungsprozesses wurden alle kleineren Funktionsupdates und Entwicklungsfortschritte kontinuierlich und regelmäßig auf GitHub hochgeladen. Dies diente vor allem als Backup.

Anbei der GitHub-Link für den Code:

<https://github.com/TheMinestone/AutoTuner/tree/main/code/stm32>

5.2 Flussdiagramm



5.3 Programmbeschreibung

5.3.1 Interrupt

Ein Interrupt bei einem Mikroprozessor ist ein Ereignis, das den normalen Programmablauf unterbricht, um eine spezielle Aufgabe oder eine dringende Anforderung zu behandeln. Interrupts ermöglichen es dem Mikroprozessor, auf externe Signale zu reagieren oder interne Ereignisse zu verarbeiten, ohne dabei den laufenden Programmcode zu blockieren. Bei Mikrocontrollern können Interrupts ausgelöst werden, wenn sich der an einem bestimmten Eingangs-Pin anliegende Pegel ändert, eine vorher festgelegte Zeitspanne abgelaufen ist, eine serielle Übertragung abgeschlossen ist oder eine Messung des Analog-Digital-Wandlers abgeschlossen ist.

Die Registrierung eines Interrupts setzt ein entsprechendes Interruptflag. Das Anwendungsprogramm wird dann unterbrochen, das Interruptflag gelöscht und eine Interrupt Service Routine (ISR) aufgerufen. Nach Beendigung der ISR, läuft das Programm weiter. ISRs sollten kurz sein und komplexe Berechnungen oder Ausgaben vermeiden, stattdessen sollten sie Steuersignale oder Parameter an das Hauptprogramm übergeben.

Für mein Projekt wird der Interrupt durch mehrere Timer gesteuert. Alle 100µs stoppt der normale Programmablauf für eine ADC-Messung. Dabei wird eine bestimmte Anzahl an Zyklen des ADCs aufgenommen und gespeichert. Sobald diese Aufnahmen abgeschlossen sind, endet das Interrupt und das Programm fährt fort.

Darüber hinaus prüft eine weitere Interrupt-Routine alle 5 Sekunden, ob die Batterie noch ausreichend Spannung aufweist.

Sobald sich das Stimmgerät im “Tune” Modus befindet, wird der Timer Aktiviert

```
HardwareTimer *timer = new HardwareTimer(TIM3);           // Timer Initialisieren  
HardwareTimer *batteryLevel = new HardwareTimer(TIM2);    // Timer Initialisieren
```

```
timer->setOverflow(100, MICROSEC_FORMAT);                // 100 µs  
batteryLevel->setOverflow(5, SECONDS_FORMAT);            // 5 s
```

Sobald sich das Stimmgerät im “Tune” Modus befindet, wird der Timer Aktiviert.

```
timer->attachInterrupt(read);
```

5.3.2 ADC

Ein Analog-Digital-Wandler (ADC, Analog-to-Digital Converter) ist ein elektronisches Gerät, das analoge Signale in digitale Werte umwandelt. Dies ist notwendig, weil viele physikalische Größen wie Temperatur, Druck, und Lichtintensität analog sind, während digitale Systeme wie Mikrocontroller und Computer nur digitale Daten verarbeiten können.

Es gibt verschiedene Arten von ADCs, die sich in ihrer Architektur und ihren Eigenschaften unterscheiden:

Successive Approximation Register (SAR) ADC:

- Arbeitet iterativ, um den analogen Wert zu approximieren.
- Bietet eine gute Balance zwischen Geschwindigkeit und Genauigkeit.
- Weit verbreitet in Mikrocontrollern und allgemeinen Anwendungen.

Flash ADC:

- Sehr schnell, verwendet eine parallele Struktur von Komparatoren.
- Eignet sich für Anwendungen, die hohe Abtastraten benötigen, wie z.B. Radar.
- Benötigt viel mehr Hardware und ist daher teurer und energieintensiver.
-

STM32 Mikrocontroller, die von STMicroelectronics hergestellt werden, verwenden in der Regel SAR-ADCs. Diese ADCs sind gut geeignet für eine breite Palette von Anwendungen, da sie eine gute Balance zwischen Geschwindigkeit, Genauigkeit und Komplexität bieten. Die SAR-ADCs in STM32 Mikrocontrollern bieten in der Regel folgende Eigenschaften:

- Auflösungen von 10 bis 16 Bit, je nach Modell. Im meinem Fall 12-Bit
- Sampling-Raten, die typischerweise im Bereich von 1 bis 5 MSPS (Millionen Samples pro Sekunde) liegen.
- Mehrere Kanäle, die den Anschluss mehrerer analoger Sensoren ermöglichen.
- Unterstützung für verschiedene Trigger-Modi und Konfigurationen, um flexibel auf unterschiedliche Anwendungen reagieren zu können.

Die Ansteuerung des ADC erfolgt wie folgt

```
uint32_t Signal::readADC() const {
    return analogRead(_pin);
}
```

Das Signal wird zunächst sorgfältig ausgelesen und für die spätere Verarbeitung in einem Vektor gespeichert. Sobald die maximale Anzahl an Messzyklen erreicht ist, können wir zur nächsten Phase übergehen, der Berechnung der Frequenz.

```
_wave.push_back(static_cast<uint8_t>(readADC()));
_cycle++;
if (_cycle == _max_cycles) {
    _cycle = 0;
    _ready = true;
}
```

5.3.3 Berechnung

Die Berechnung basiert auf einer Methode, die die extremen Werte des eingelesenen Signals erkennt, genauer gesagt den maximalen und minimalen Wert. Diese beiden Punkte sind von besonderem Interesse, da sie uns erlauben, zwei Schwellenwerte zu berechnen, die wir als _pUP und _pDOWN bezeichnen.

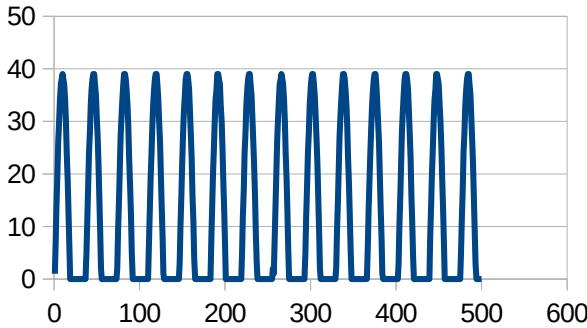


Abbildung 24: Eingangssignal des μ C



Abbildung 23: Signal des Pickups

```
auto max_value = *max_element(_wave.begin(), _wave.end());
_pUP = max_value * 0.75;
_pDOWN = max_value * 0.25;
```

Nachdem diese Schwellenwerte festgelegt sind, beginnen wir mit der Analyse des gesamten Vektors. Wir gehen jeden Punkt nacheinander durch und achten darauf, ob ein Punkt die Schwelle überschreitet. Wenn wir eine Schwelle erkennen, speichern wir die Position dieses Punktes, die wir xP nennen.

Dieser Prozess der Schwellerkennung und Positionsspeicherung wird so lange fortgesetzt, bis wir insgesamt zehn Schwellen gefunden haben. Das bedeutet, dass wir fünf Maxima und fünf Minima identifiziert haben.

```
for (int i = 0; i < _wave.size(); ++i) {
    if (_points == 0 && _wave[i] >= _pUP) {
        _xP1 = i;
        _points = 1;
    }
    if (_points == 1 && _wave[i] <= _pDOWN) {
        _xP2 = i;
        _points = 2;
    }
}
```

Sobald wir diese zehn Punkte haben, gehen wir zur nächsten Berechnungsphase über. Hier berechnen wir das Delta, also den Unterschied, zwischen zwei aufeinander folgenden Punkten. Dieses Delta wird dann auf alle zehn Punkte angewendet. Auf diese Weise können wir einen Durchschnittswert erzeugen, der uns ein Mittelwert der Frequenz des Signals gibt.

```
_sumDelta = _deltaT1T3 + _deltaT3T5 + _deltaT5T7 + _deltaT7T9 + _deltaT2T4 + _deltaT4T6 + _deltaT6T8 + _deltaT8T10;
frequency = 1 / (((_sumDelta) / 8) * 0.000001);
```

5.4 Motorsteuerung

Die Motorsteuerung erfolgt über zwei Digitale Pins welche je nach Kombination den Motor bewegen können.

FIN	RIN	Operation
0	0	Stillstand
1	0	Vorwärts
0	1	Rückwärts
1	1	Stop

Für die Motorsteuerung habe ich eine spezielle Klasse entwickelt, deren Hauptzweck es ist, die Steuerung des Motors zu vereinfachen. In Kombination mit der Frequenzüberwachung reguliert der Motor ständig. Wenn die Frequenz darüber oder darunter liegt, wird der Motor entsprechend aktiviert, um die Saite zu stimmen.

```
motor.standby();
motor.forward();
motor.reverse();
motor.break();
```

5.5 Menüstruktur

Um alles korrekt kontrollieren zu können, habe ich verschiedene Zustände für die jeweiligen Teilaufgaben programmiert. Die Menüstruktur habe ich als Switch-Case programmiert, um den Überblick zu behalten. Sie ist hauptsächlich in drei Teilbereiche gegliedert: der Hauptzustand "Main_State", der Zustand für die Signalverarbeitung "Signal_State" und der Zustand für die Einstellungen "Settings_State".

```
enum MAIN_STATE {
    STATE_IDLE = 0,
    STATE_TUNE_E2,
    STATE_TUNE_A2,
    STATE_TUNE_D3,
    STATE_TUNE_G3,
    STATE_TUNE_H3,
    STATE_TUNE_E4,
    STATE_SETTINGS,
    STATE_BATTERY_CRITICAL,
};

enum SIGNAL_STATE {
    SIGNAL_WAIT = 0,
    SIGNAL_READ,
    SIGNAL_PROCESS,
};

enum SETTINGS_STATE {
    SETTINGS_MAIN = 0,
    SETTINGS_CHANGE_TOLERANCE,
    SETTINGS_CHANGE_TUNE,
    SETTINGS_BATTERY,
    SETTINGS_BUZZER,
};
```

Die Tasten dienen als primäres Mittel, um durch das Menü zu navigieren und verschiedene Funktionen zu aktivieren. Man kann nicht nur einzelne Tasten verwenden, sondern es gibt auch eine Vielzahl von Tastenkombinationen, um schnell auf die Einstellungen zuzugreifen. Diese Tastenkombinationen sind eine praktische Funktion, um auf verschiedene Einstellungen zuzugreifen.

Sobald im Hauptmenü STATE_SETTINGS getroffen wird, wird SETTINGS_STATE aktiviert. Von dort aus kann die Toleranz des Tones, die Stimmung des Kammertons A4 (440Hz) geändert und der Buzzer ein- oder ausgeschaltet werden. Zudem kann man den aktuellen Batterieladestand einsehen.

Um die Signalmessung nicht zu verfälschen, kann während der Verarbeitung des Signals kein neues Signal aufgenommen werden.

6 Gehäuse

6.1 Fusion 360

Fusion 360 ist eine umfassende 3D-CAD-Software von Autodesk, die den gesamten Produktentwicklungsprozess unterstützt. Die Hauptfunktion im Bereich der 3D-Modellierung liegt in der Kombination von parametrischer und direkter Modellierung, wodurch detaillierte und präzise Modelle erstellt werden können. Parametrische Modellierung ermöglicht die Definition von Abhängigkeiten zwischen Modellkomponenten, während die direkte Modellierung schnelle Anpassungen erlaubt. Zudem bietet Fusion 360 integrierte Simulations- und Analysewerkzeuge, um Modelle auf strukturelle und thermische Belastungen zu prüfen.

6.2 Schachtel und Deckel

Das Gehäuse und der Deckel wurden eigenständig modelliert. Dabei wurde das fertige PCB in Fusion importiert und die Maße für das Gehäuse drumherum gezeichnet. Die 3D-Pakete des PCBs, die von Eagle auf Fusion übertragen wurden, waren dabei sehr nützlich, da man die Bauteile in der 3D-Ansicht sehen konnte. Wichtig war die Höhe der Halterung, auf die die Platine montiert wird, da darunter beide Batterien Platz haben müssen. Zusätzlich musste ich genug Platz zwischen der Platine und dem Deckel lassen, da das Display noch im Deckel untergebracht werden muss und eine bestimmte Tiefe hat. Die Aussparung der drei Knöpfe stellte eine Herausforderung dar, da der Kopf des Aufsatzes nicht als 3D-Objekt in Fusion enthalten war und ich nicht genau wusste, wie hoch und breit diese sind. Den 3D-Die Produktion des Gehäuses wurde in der Schule durchgeführt, was den Vorteil hatte, dass keine Kosten anfielen. Bei diesem Prozess kam das weit verbreitete Druckmaterial Acrylnitril-Butadien-Styrol zum Einsatz, das sich durch Vielseitigkeit auszeichnet.

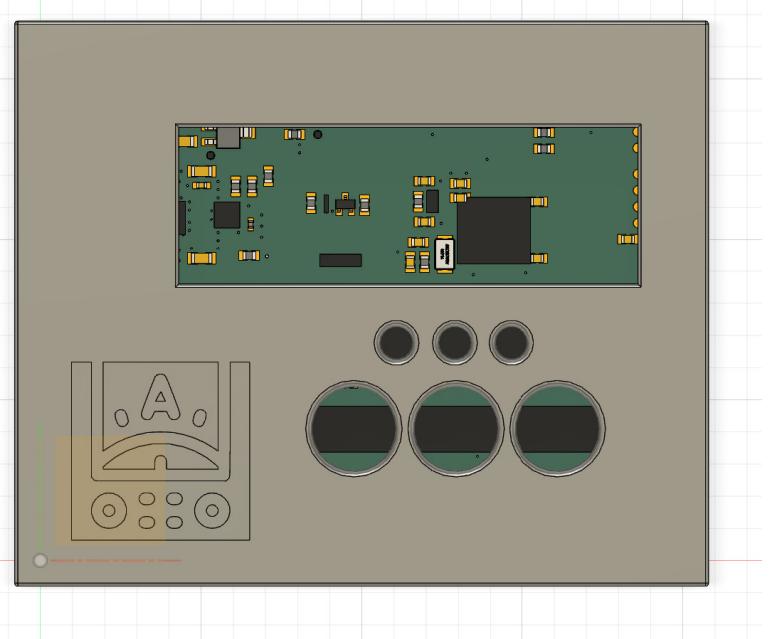


Abbildung 25: Vogelperspektive des Gehäuses

6.3 Motoraufsatz

Nicht nur beim Motoranbau, sondern auch beim Zeichnen der Box und des Deckels waren die Toleranzen des 3D-Drucks schwer einzuschätzen. Das liegt daran, dass sich das Filament nach dem Druck stark verformte und verbog. Beim Motoranbau ging es hauptsächlich darum, die Kraft des Motors auf die Drehfläche zu verteilen. Ausgehend vom Motorgewinde wurde eine zunehmend breiter werdende zylinderförmige Form gewählt, die dann mit der Ausbuchtung harmoniert, in die der Gitarrendreher passt.

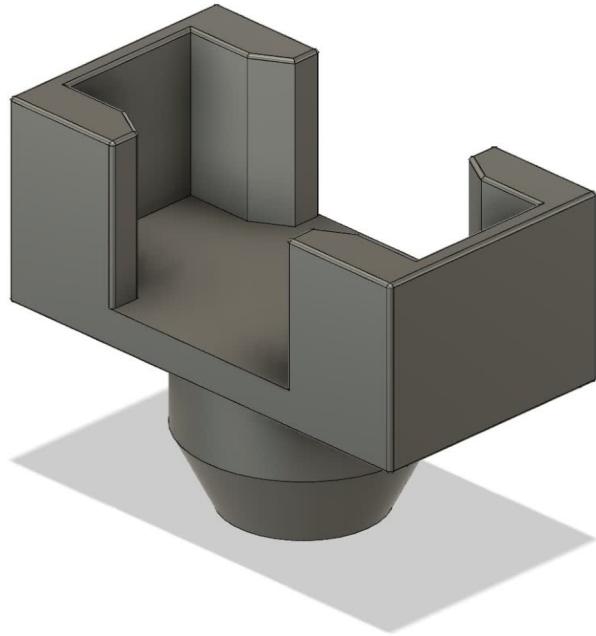


Abbildung 26: Aufsatz des Motors

7 Projektmanagement

7.1 Arbeitsstunden

Im Rahmen dieses Projekts wurde ein Stundenlohn von 40€ brutto vereinbart. Da ich derzeit als Student tätig bin und keine zusätzlichen Einkommensquellen habe, sind meine Einkünfte aus dem Projekt gemäß dem Einkommensteuertarif 2024 (nach § 32a des Einkommensteuergesetzes – EstG) der progressiven Besteuerung unterworfen. Dies bedeutet, dass meine Steuerbelastung in die Progressionszone I oder die untere Progressionszone fällt, die Steuersätze 23% umfasst.

Projektwoche	Stunden [h]				
	Schule	Offenes Labor	Zuhause	Gesammt	
1	5	0	2	7	€ 280.00
2	5	0	0	5	€ 200.00
3	5	0	0	5	€ 200.00
4	5	0	2	7	€ 280.00
5	5	0	0	5	€ 200.00
6	5	0	0	5	€ 200.00
7	5	0	5	10	€ 400.00
8	5	0	6	11	€ 440.00
9	5	0	12	17	€ 680.00
10	5	0	15	20	€ 800.00
11	5	0	25	30	€ 1,200.00
12	3	0	12	15	€ 600.00
13	5	0	8	13	€ 520.00
14	5	0	3	8	€ 320.00
15	0	0	2	2	€ 80.00
16	0	0	4	4	€ 160.00
17	2	0	0	2	€ 80.00
18	5	0	2	7	€ 280.00
19	5	0	0	5	€ 200.00
20	5	0	0	5	€ 200.00
21	5	0	0	5	€ 200.00
22	0	0	3	3	€ 120.00
23	2	0	2	4	€ 160.00
24	5	0	0	5	€ 200.00
25	1	0	0	1	€ 40.00
26	5	0	10	15	€ 600.00
27	5	0	12	17	€ 680.00
28	3	0	7	10	€ 400.00
29	2	0	8	10	€ 400.00
30	2	0	6	8	€ 320.00
31	5	0	5	10	€ 400.00
32	3	2.5	4	9.5	€ 380.00
33	2	0	3	5	€ 200.00
34	0	0	0	0	€ 0.00
35	0	0	2	2	€ 80.00
36	1	0	12	13	€ 520.00
Gesamt	126	2.5	172	300.5	€ 12,020.00
Steuerabgabe				23.00%	€ 2,764.60
Gesamt abzüglich Steuerabgabe					€ 9,255.40

7.2 Beraterstunden

Der Stundensatz für Beratungsleistungen wurde auf 80€ vereinbart. Dies bezieht sich auf Unterstützung von Lehrpersonen oder Mitschülern. Hilfeleistungen, die länger als 15 Minuten dauern, werden notiert. Diese werden schwarz abgerechnet und unterliegen keiner Steuer.

Datum	Stunden [h]	Berater	Kosten
26.09.2023	0.25	Huber Ivan	€ 20.00
28.09.2023	0.25	Detomaso Martin	€ 20.00
24.10.2023	0.25	Yegorov Vladislav	€ 20.00
21.11.2023	0.5	Yegorov Vladislav	€ 40.00
23.11.2023	0.5	Yegorov Vladislav	€ 40.00
23.11.2023	0.25	Detomaso Martin	€ 20.00
14.03.2024	0.25	Tibolla Fabian	€ 20.00
19.03.2024	3	Tibolla Fabian	€ 240.00
Gesamt	5.25		€ 420.00

7.3 Materialkosten

7.3.1 PCB und Bestückte Bauteile

Hier sind alle Bauteile aufgelistet, die von JLCPCB, der Firma, bei der die Platine bestellt wurde, bestückt wurden. Es ist zu beachten, dass dies nur die Bauteile der dritten Version der Platine sind. Diese wurde zweimal bestückt, daher beträgt die Anzahl oft "2". Kleine Bauteile wie zum Beispiel Dioden können nur in 5er oder 10er Mengen gekauft werden. Diese Anzahl steht also nicht in direktem Zusammenhang mit der effektiven Anzahl der bestückten Bauteile.

Art	Name	Anzahl	Einzelpreis	Gesamtpreis
Widerstände	R0805	152	€ 0.00	€ 0.24
Kondensatoren	C0805	38	€ 0.01	€ 0.23
Dioden	H5VL10B, 1N4148WS und 1N5819WS	24	€ 0.01	€ 0.24
MOSFET P-Channel	BSS84	10	€ 0.02	€ 0.20
Quarz	Q13FC13500004 und X50328MSB2GI	4	€ 0.16	€ 0.64
Spannungswandler	TPS63070RNMR	2	€ 0.60	€ 1.20
Leds	KT-0805	16	€ 0.01	€ 0.21
STM32	STM32F103C8T6	2	€ 1.25	€ 2.49
MOSFET N-Channel	AO3400A	2	€ 0.07	€ 0.14
Batterie Lader	TP4054-42-SOT25R	5	€ 0.13	€ 0.64
OPV	LM324DT	2	€ 0.12	€ 0.24
Spule	XFL4020-152MEC	2	€ 1.09	€ 2.18
Taster	TS-1187A-B-A-B	4	€ 0.02	€ 0.06
NPN Transistor	S8050-J3Y	10	€ 0.01	€ 0.08
Spannungswandler	AMS1117-3.3	2	€ 0.13	€ 0.27
EXTENDED PARTS				€ 26
PCB			Sconto	€ -9
Shipping				€ 20.42
Gesamt				€ 61.76

7.3.2 Nicht Bestückte Bauteile

Hier sind alle Bauteile aufgeführt, die nicht von JLCPCB bestückt wurden. Bauteile, die einen Preis von 0.00€ haben, stammen entweder von der Schule oder waren bereits in meinem Besitz (wie zum Beispiel das Arduino Kit).

Art	Name	Anzahl	Einzelpreis	Gesamtpreis
Batterie	6060100 3.7V 5000mAh Rechargeable LiIon Battery	2	€ 4.86	€ 9.72
THT Leds	5mm THT LEDs	3	€ 0.00	€ 0.00
Pin Heads	2,4mm Pinheads	2	€ 0.00	€ 0.00
Schalter	3 Pin Schalter	1	€ 0.00	€ 0.00
Trimmer	THT Trimmer	2	€ 0.00	€ 0.00
Display	4 PIN LCD Display	1	€ 0.00	€ 0.00
H-Brücke	ROHM H-BRIDGE 18V (BD6222HFP-TR)	2	€ 4.09	€ 8.18
Motor	6V DC Motor mit Getriebe	2	€ 2.35	€ 4.70
Buzzer	Passiver Buzzer	1	€ 0.00	€ 0.00
Taster	Taster mit Rundem Aufsatz	3	€ 0.00	€ 0.00
Audio Jack	3.5mm Audio Jack Female	2	€ 1.00	€ 2.00
USB-B	USB-B Female	2	€ 1.00	€ 2.00
Gesamt			€ 13.30	€ 26.60

7.4 Gesamtkostenrechnung

Die Gesamtkosten des AutoTuners bestehen aus den Arbeitsstunden, den Beraterstunden und den Bauteilkosten. Dies soll eine echte Kostenrechnung eines Projekts simulieren.

Kategorie	Kosten
Arbeitsstunden	€ 12,020.00
Beraterstunden	€ 420.00
Bestückt	€ 61.76
Nicht Bestückt	€ 26.60
Gesamt	€ 12,528.36

8 Schlusswort

Zum Abschluss möchte ich festhalten, dass das Projekt, welches mich durch das gesamte Schuljahr hindurch begleitet hat, mir sehr zugesagt hat. Es hat mir nicht nur gefallen, sondern es hat mich gelehrt, ich habe eine Fülle an Kenntnissen und Erfahrungen daraus geschöpft. Die emotionale Achterbahn während dieses Prozesses war ständig präsent, mit seinen Höhen und Tiefen. Auf der einen Seite gab es weniger erfreuliche Momente, die mich herausforderten und auf die Probe stellten. Auf der anderen Seite gab es zahlreiche freudige Augenblicke, die jede Herausforderung lohnenswert machten. Man kann sich den Verlauf dieses Projektes bildlich wie eine Berg- und Talfahrt vorstellen.

Des Weiteren spielte Stress eine sehr präsente und nicht zu unterschätzende Rolle in diesem Projekt. Es gab Zeiten, in denen die Dinge nicht so liefen, wie sie sollten, oder es gab Verzögerungen. Diese Situationen verursachten Stress, der jedoch auch eine treibende Kraft sein kann. Trotz dieser Herausforderungen ist es wichtig, immer im Hinterkopf zu behalten, dass die Entwicklung des Projekts stetig fortschreitet. Jedes Hindernis, jede Schwierigkeit ist letztendlich nur ein Schritt auf dem Weg zur Fertigstellung des Projekts und trägt dazu bei, das Endergebnis zu formen und zu verbessern.

Mir wurde die Wichtigkeit einer sorgfältigen Planung, Strukturierung und vor allem der Ordnung in der Zettelarbeit erst gegen Ende der Projektabgabe bewusst. In den ersten Projektwochen habe ich beträchtlich viel Zeit mit der Überlegung des Prinzips verbracht, vielleicht auch aufgrund mangelnder Protokollierung. Diese Verzögerung führte dazu, dass ich erst später mit der Planung und Zeichnung der Schaltung begann.

Diese Verzögerung wurde letztendlich zum Hindernis bei der Abgabe der ersten Platine. Trotz dieser Herausforderung habe ich mir die Zeit genommen, die Platine nach der Abgabe ausführlich zu überarbeiten und zu verbessern. Dadurch konnten die Grundfunktionen bereits mit der ersten Version der Schaltung erfolgreich umgesetzt werden.

Ein weiteres Problem stellte die Planung des Motoraufsatzes dar, die ich erst zwei Wochen vor der Projektabgabe in Angriff nahm. Glücklicherweise wurden aufgrund schulischer Aktivitäten die Fristen für die Abgabe verschoben. Dies gab mir ausreichend Zeit, um die Halterung fertigzustellen.

Alles in allem bot mir dieses "Praktikum" einen wertvollen Einblick in die Arbeitswelt in diesem Bereich. Es war eine Erfahrung, die mich viele wichtige Lektionen in Bezug auf Projektmanagement und die Bedeutung von Struktur und Ordnung in der Arbeit lehrte.

9 Anhänge

9.1 Quellen

- (1) Introduction to STM32F103 - The Engineering Knowledge.
<https://www.theengineeringknowledge.com/introduction-to-stm32f103/>.
- (2) Datasheet - STM32F103xC, STM32F103xD, STM32F103xE - High-density
<https://www.st.com/resource/en/datasheet/stm32f103ze.pdf>.
- (3) STM32F103C8 - STMicroelectronics.
<https://www.st.com/en/microcontrollers-microprocessors/stm32f103c8.html>.
- (4) STMICROELECTRONICS STM32F103C8T6 - element14 Community.
<https://in.element14.com/stmicroelectronics/stm32f103c8t6/mcu-32bit-cortex-m3-72mhz-lqfp/dp/1447637>.
- (5) STM32F103 - PDF Documentation - STMicroelectronics. <https://www.st.com/en/microcontrollers-microprocessors/stm32f103/documentation.html>.
- (6) en.wikipedia.org. <https://en.wikipedia.org/wiki/STM32>.
 - (1) In-circuit debugger/programmer for STM8 and STM32 - st.com.
https://www.st.com/resource/en/data_brief/st-link.pdf.
 - (2) ST-LINK/V2 in-circuit debugger/programmer for STM8 and STM32.
https://media.distrelec.com/Web/Downloads/_t/ds/ST-LINK_V2_eng_tds.pdf.
 - (3) ST-LINK/V2 in-circuit debugger/programmer for STM8 and STM32.
<https://www.st.com/en/development-tools/st-link-v2.html>.
 - (4) stlink-org/stlink: Open source STM32 MCU programming toolset - GitHub.
<https://github.com/stlink-org/stlink>.
 - (5) ST-LINK/V2 in-circuit debugger/programmer - STMicroelectronics.
https://www.st.com/resource/en/user_manual/dm00026748-st-link-v2-in-circuit-debugger-programmer-for-stm8-and-stm32-stmicroelectronics.pdf.
- (1) Quarzoszillator – Wikipedia. <https://de.wikipedia.org/wiki/Quarzoszillator>.
- (2) Quarze und AVR – Mikrocontroller.net. https://www.mikrocontroller.net/articles/Quarze_und_AVR.
- (3) 32.768 kHz Quarze – Mouser Deutschland.
<https://www.mouser.de/c/passive-components/frequency-control-timing-devices/crystals/?frequency=32.768%20kHz&pg=3>.
- (4) Präzision und Frequenzstabilität: Die Stärken von 32,768 kHz
<https://www.jauch.com/blog/praezision-und-frequenzstabilitaet-die-staerken-von-32768-khz-oszillatoren/>.

- (5) 8 MHz Quarze – Mouser Deutschland. <https://www.mouser.de/c/passive-components/frequency-control-timing-devices/crystals/?frequency=8%20MHz>.
- (6) 8 MHz Crystals Quarze Datenblätter – Mouser Deutschland. <https://www.mouser.de/c/ds/passive-components/frequency-control-timing-devices/crystals/?frequency=8%20MHz&product=Crystals>.
- (7) 8 MHz SMD/SMT Quarze – Mouser Deutschland.
<https://www.mouser.de/c/passive-components/frequency-control-timing-devices/crystals/?frequency=8%20MHz&termination%20style=SMD%2FSMT>.
- (8) crystal 8 MHz - guloshop. <https://www.guloshop.de/shop/Quarze-und-Kondensatoren/Quarz-8-MHz-mit-Kondensatoren::39.html>.
- (9) Volumenwellenresonator ersetzt externen Quarz - Mikrocontroller
<https://www.elektroniknet.de/halbleiter/mikrocontroller/volumenwellenresonator-ersetzt-externen-quarz.164995.html>.
- (10) Kleine Taktgeber: Wo begegnen uns Quarze und Oszillatoren im Alltag
<https://www.jauch.com/blog/kleine-taktgeber-wo-begegnen-uns-quarze-und-oszillatoren-im-alltag/>.
- (11) 32.768kHz MICRO CRYSTAL Quarze | Farnell DE. <https://de.farnell.com/c/quarze-oszillatoren-resonatoren/quarze?brand=micro-crystal&nennfrequenz=32.768khz>.
- (12) Selecting and Testing 32.768 kHz Crystal Oscillators for AVR
https://www.microchip.com/content/dam/mchp/documents/MCU08/ApplicationNotes/ApplicationNotes/AN2648>Selecting_Testing-32KHz-Crystal-Osc-for-AVR-MCUs-00002648.pdf.
- (1) Universal Serial Bus – Wikipedia. https://de.wikipedia.org/wiki/Universal_Serial_Bus.
- (2) Stromversorgung mit USB - Elektronik-Kompendium.de.
<https://www.elektronik-kompendium.de/sites/com/2212141.htm>.
- (3) USB: Strom und Spannung messen (Volt und Ampere) - ganz einfach - GIGA.
<https://www.giga.de/extra/usb/specials/usb-strom-und-spannung-messen-volt-ampere/>.
- (4) USB - Universal Serial Bus (1.0 / 1.1) - Elektronik-Kompendium.de. <https://www.elektronik-kompendium.de/sites/com/0312021.htm>.
- (5) USB-Schnittstelle | Definition und Erklärung - DeLSt. <https://bing.com/search?q=USB+Schnittstelle+Eigenschaften>.
- (6) Wissen: Alles über USB und seine Stecker - Allround-PC.com.
<https://www.allround-pc.com/artikel/wissen/2015/wissen-alles-ueber-usb-stecker-und-standards>.
- (7) USB-Schnittstelle | Definition und Erklärung - DeLSt. <https://www.delst.de/de/lexikon/usb-schnittstelle/>.

- (8) USB einfach erklärt – Funktionen, Standards und Anschlüsse - BRACK.CH.
<https://www.brack.ch/ratgeber/computer-laptops-zubehoer/einfach-erklaert/usb-funktionen-standards-anschluesse>.
- (1) Batterie - chemie.de. <https://www.chemie.de/lexikon/Batterie.html>.
- (2) Batterie (Elektrotechnik) – Wikipedia. https://de.wikipedia.org/wiki/Batterie_%28Elektrotechnik%29.
- (3) Batterie in Physik | Schülerlexikon | Lernhelper. <https://bing.com/search?q=Was+ist+eine+Batterie+Physikalische+Chemische+Informationen>.
- (4) Batterie in Physik | Schülerlexikon | Lernhelper.
<https://www.lernhelper.de/schuelerlexikon/physik/artikel/batterie>.
- (5) Batterie – Klexikon – das Kinderlexikon. <https://klexikon.zum.de/wiki/Batterie>.
- (6) Getty Images. <https://www.gettyimages.com/detail/photo/battery-renewable-energy-innovation-ev-lithium-royalty-free-image/1392175897>.
- (1) Buck-Boost Converter | Analog Devices. <https://www.analog.com/en/resources/glossary/buck-boost.html>.
- (2) Buck converter - Wikipedia. https://en.wikipedia.org/wiki/Buck_converter.
- (3) Abwärtswandler – Wikipedia. <https://de.wikipedia.org/wiki/Abw%C3%A4rtswandler>.
- (4) Buck-boost converter - Wikipedia. https://en.wikipedia.org/wiki/Buck-%E2%80%93boost_converter.
- (5) Buck-Boost Converter: What is it? (Formula and Circuit Diagram).
<https://www.electrical4u.com/buck-boost-converter/>.
- (6) So geht EMV-gerechtes Leiterplattenlayout - Elektroniknet. <https://www.elektroniknet.de/e-mechanik-passive/passive/so-geht-emv-gerechtes-leiterplattenlayout.191213.html>.
- (7) DC/DC-Wandler für eine gute EMV optimieren - all-electronics.
<https://www.all-electronics.de/elektronik-entwicklung/emv-optimierte-dcdc-wandler.html>.
- (8) WEBENCH® Power Designer Power supply design made easy.
<https://www.ti.com/lit/ml/slyp708/slyp708.pdf?ts=1626989991697>.
- (9) Webench Power Designer | Stromversorgung entwickeln.
<https://www.we-online.com/de/support/design-tools/webench>.
- (10) How to use WEBENCH Power Designer | Video | TI.com.
<https://www.ti.com/video/5859518204001>.
- (11) DC to DC Buck Converter Tutorial & Diagram | Analog Devices.
<https://www.analog.com/en/resources/technical-articles/dc-to-dc-buck-converter-tutorial.html>.

(12) Schematic_Editor - webench.ti.com.

https://webench.ti.com/help/PowerDesigner/Schematic_Editor/Schematic_Editor.htm.

(13) DCDC Wandler Inp 18-36V auf 5V Printversion Eingang 24V typisch.

<https://www.netzgeraet.de/spannungswandler/dcdc-wandler/dcdc-wandler-pcb-platine/23865/dcdc-wandler-inp-18-36v-auf-5v-printversion-eingang-24v-typisch.html>.

(14) Vorteile von extrem kleinen PoL-DC/DC-Wandlern | DigiKey.

<https://www.digikey.de/de/articles/when-more-is-less-save-valuable-space-using-more-regulators>.

(15) Nachbau Schaltung.

https://files2.elv.com/public/13/1301/130118/Internet/130118_dcdc12_schaltplan.pdf.

(1) So steuert man ein LCD-Display mit Arduino - Arduino Tutorial. <https://starthardware.org/lcd/>.

(2) Wie funktioniert LCD - einfach erklärt - CHIP. https://praxistipps.chip.de/wie-funktioniert-lcd-einfach-erklaert_101313.

(3) Flüssigkristallanzeige – Wikipedia. <https://de.wikipedia.org/wiki/Fl%C3%BCssigkristallanzeige>.

(4) Funktion von LCD-Displays | LEIFIphysik.

<https://www.leifiphysik.de/optik/polarisation/ausblick/funktion-von-lcd-displays>.

(5) LCD Funktionsweise – anschaulich erklärt - COMPUTER BILD.

<https://www.computerbild.de/artikel/cb-Tipps-Notebooks-Netbooks-LCD-Die-Funktionsweise-der-Bildschirme-31496909.html>.

(6) github.com.

<https://github.com/T-systemsYoungsters/Emelie-LOGIST/tree/ec3496b9e20eda4ac6964e38fd73b03537233fe5/PlatformIO%2FProjects%2Fhello%20world%2Fsrc%2Fmain.cpp>.

(7) github.com. https://github.com/richelbilderbeek/arduino_voor_jonge_tieners/tree/478ccaf0c2d43d82c71241bae150fe41959fe684/hoofdstukken%2FLCD1%2FREADME.md.

(8) github.com. <https://github.com/YuRen-Su/Arduino-Classroom-learning-content/tree/e7ec24a42e5ede60d25324fec0a28e83eaa8b816/README.md>.

(1) Autodesk Fusion 360 for personal use. <https://www.autodesk.com/products/fusion-360/personal>.

(2) Download Autodesk Fusion for Free | Free Trial | Autodesk.

<https://www.autodesk.com/products/fusion-360/free-trial>.

(3) Fusion 360. <https://www.autodesk.in/products/fusion-360/free-trial>.

(4) Autodesk Fusion. https://play.google.com/store/apps/details/Autodesk_Fusion?id=com.autodesk.fusion&hl=en_US.

(5) Autodesk Fusion. <https://apps.apple.com/it/app/autodesk-fusion/id991074843>.

(1) Interrupts richtig einsetzen! – Edis Techlab. <https://edistechlab.com/interrupts/>.

- (2) Interrupt – Mikrocontroller.net. <https://www.mikrocontroller.net/articles/Interrupt>.
- (3) Interrupt - Microcontroller - Mikrocontroller Programmierung. <http://www.mikrocontroller-programmierung.de/interrupt.php>.
- (4) MCT: Interrupt: Was ist das? - xplore-dna. <https://www.xplore-dna.net/mod/page/view.php?id=1436&lang=en>.
- (5) de.wikipedia.org. <https://de.wikipedia.org/wiki/Interrupt>.

9.2 Datenblätter

STM32: <https://www.st.com/resource/en/datasheet/stm32f205rb.pdf>

LCD Display: <https://www.openhacks.com/uploadsproductos/eone-1602a1.pdf>

H-Brücke: <https://fscdn.rohm.com/en/products/databook/datasheet/ic/motor/dc/bd622x-e.pdf>

Buck Booster: https://www.ti.com/lit/ds/symlink/tps63070.pdf?ts=1716852682992&ref_url=https%253A%252F%252Fwww.google.com%252F

Batterie Lader: https://www.laskakit.cz/user/related_files/tp4054.pdf

OPV: <https://eu.mouser.com/datasheet/2/389/lm224a-1849701.pdf>

3V3 Linearregler: <http://www.advanced-monolithic.com/pdf/ds1117.pdf>