

**HTBLuVA Salzburg**

**Höhere Lehranstalt für**

**Elektronik und Technische Informatik**

HTL_Logo

**DIPLOMARBEIT**

Gesamtprojekt **Gitcon**

**Entwicklung eines E-Gitarren zu MIDI-Converters**

Daniel Bräumann 5AHEL Betreuer: Prof. Dipl.-Ing. Siegbert

Schrempf

Simon Grundner 5AHEL

Laurenz Hölzl 5AHEL

ausgeführt im Schuljahr 2022/23

Abgabevermerk:

Datum: übernommen von:

**Eidesstattliche Erklärung**

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

Salzburg, am TT.MM.JJJJ Verfasserinnen / Verfasser:

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

Daniel Bräumann

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

Simon Grundner

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

Laurenz Hölzl

**DIPLOMARBEIT**

**DOKUMENTATION**

|  |  |
| --- | --- |
| Namen der  Verfasserinnen / Verfasser | Daniel Bräumann  Simon Grundner  Laurenz Hölzl |
| Jahrgang  Schuljahr | 5AHEL  2022/23 |
| Thema der Diplomarbeit | Gitcon – Entwicklung eines E-Gitarren zu MIDI-Converters |

|  |  |
| --- | --- |
| Aufgabenstellung | Virtuelle Instrumente sind in der modernen Musik-Produktion aufgrund ihrer Vielseitigkeit weit verbreitet. Das Projekt macht es möglich, diese virtuellen Instrumente auch mit einer E-Gitarre zu spielen. Hierfür wird das analoge Audiosignal direkt von der Ausgangsbuchse der Gitarre abgegriffen, in MIDI-Noten umgewandelt und via USB an eine Digital Audio Workstation (DAW) übertragen. |

|  |  |
| --- | --- |
| Realisierung |  |

|  |  |
| --- | --- |
| Ergebnisse |  |

|  |  |
| --- | --- |
| Typische Grafik, Foto etc.  (mit Erläuterung) |  |

|  |  |
| --- | --- |
| Teilnahme an Wettbewerben,  Auszeichnungen |  |

|  |  |
| --- | --- |
| Möglichkeiten der Einsichtnahme in die Arbeit |  |

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

**DIPLOMA THESIS**

**Documentation**

|  |  |
| --- | --- |
| Author(s) | Daniel Bräumann  Simon Grundner  Laurenz Hölzl |
| Form  Academic year | 2022/23 |
| Topic | Gitcon - Development of an electric guitar to MIDI converter |

|  |  |
| --- | --- |
| Assignment of Tasks | The present project enables the use of an electric guitar as a MIDI device. The device should reliably convert individual notes and chords into MIDI format with the lowest possible latency. |

|  |  |
| --- | --- |
| Realisation |  |

|  |  |
| --- | --- |
| Results |  |

|  |  |
| --- | --- |
| Illustrative Graph, Photo  (incl. explanation) |  |

|  |  |
| --- | --- |
| Participation in Competitions  Awards |  |

|  |  |
| --- | --- |
| Accessibility of  Diploma Thesis |  |

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

# Vorwort

Die vorliegende Diplomarbeit ermöglicht die Verwendung einer E-Gitarre als MIDI-Device. Das Gerät soll einzelne Noten und Akkorde zuverlässig mit möglichst geringer Latenz in das MIDI-Format umwandeln. Die MIDI-Signale werden anschließend an die USB-Schnittstelle eines PCs übertragen.

Der Name „Gitcon“ ist eine Abbreviatur von „Guitar Converter“, der Aufgrund der Funktion der Platine, nämlich Gitarrensignale in MIDI-Noten zu konvertieren, gewählt wurde.

Die Projektidee kam von Simon Grundner und Laurenz Hölzl. Da sich beide in ihrer Freizeit viel mit Musik befassen und durch ihr eigenes Interesse an digitaler Audio-Verarbeitung, war das Projekt schnell überlegt. Moderne Musikstücke werden üblicherweise in einer digitalen Umgebung aus mehreren Tonspuren zusammen gemischt. Um mit dieser Produktions-Umgebung (DAW) zu interagieren, kommen oft MIDI-Controller zum Einsatz. MIDI-Controller verwenden ein, einer Norm-Entsprechendes, Protokoll, um sich gegenseitig Parameter auszutauschen. Die Idee war es nun, eine elektrische Gitarre mit diesem Protokoll kompatibel zu machen und sämtliche gespielte Noten zu erkennen und in MIDI-Signale umzuwandeln.

Die individuellen Aufgabenstellungen wurden jeweils anhand der Spezialgebiete jedes Teammitglieds gewählt. Simon beschäftigte sich mit der Entwicklung der Hardware Frontend-Platine für den Microcontroller, Daniel widmete sich dem Entwurf der Analogen Signalverarbeitungskette und Laurenz arbeitete an der Implementierung der Digitalen Signalverarbeitung in die Firmware.

Durch unser Projekt haben wir als Team gelernt, wie wichtig gute Kommunikation ist, um gemeinsam Ziele zu erreichen. Im Laufe der Projektarbeit traten immer wieder Schwierigkeiten auf, die wir mit Erfolg überwinden konnten. Dank der exzellenten Zusammenarbeit im Team und der großartigen Unterstützung unseres Projektbetreuers konnten wir alle Schwierigkeiten meistern und das Projekt umsetzen. Die Realisierung von Gitcon hat uns nicht nur beim Verständnis technischer Problemlösungen geholfen, es ist auch definitiv eine unschätzbare Bereicherung unserer persönlichen Erfahrungen.

An dieser Stelle bedanken wir uns bei allen, die diese Diplomarbeit unterstützt haben. Zunächst bei unserem Projektbetreuer Prof. Dipl.-Ing. Siegbert Schrempf, der uns nicht nur mit seiner Expertise unterstützt, sondern auch immer wieder motiviert hat. Für die Hilfe bei der Erstellung der Projektpräsentation, bedanken wir uns außerdem bei Prof. Mag. Paul Schwaiger, der mit seiner Kritik geholfen hat, die Präsentation weiter zu verbessern.

Inhaltsverzeichnis

[Vorwort 7](#_Toc129277167)

[Überblick 11](#_Toc129277168)

[1 Systemspezifikationen 12](#_Toc129277169)

[1.1 Zielbestimmungen 12](#_Toc129277170)

[1.1.1 Musskriterien 12](#_Toc129277171)

[1.1.2 Wunschkriterien 12](#_Toc129277172)

[1.1.3 Abgrenzungskriterien 12](#_Toc129277173)

[1.2 Produkteinsatz 12](#_Toc129277174)

[1.2.1 Anwendungsbereiche 12](#_Toc129277175)

[1.2.2 Zielgruppen 12](#_Toc129277176)

[1.2.3 Betriebsbedingungen 12](#_Toc129277177)

[1.3 Produktumgebung 12](#_Toc129277178)

[1.3.1 Software 12](#_Toc129277179)

[1.3.2 Hardware 12](#_Toc129277180)

[1.4 Produktfunktionen 13](#_Toc129277181)

[1.5 Produktleistungen 13](#_Toc129277182)

[1.6 Benutzungsoberfläche 14](#_Toc129277183)

[1.7 Entwicklungsumgebung 14](#_Toc129277184)

[1.7.1 Software 14](#_Toc129277185)

[1.7.2 Hardware 15](#_Toc129277186)

[1.7.3 Orgware 15](#_Toc129277187)

[1.8 Qualitätsziel Bestimmungen 15](#_Toc129277188)

[1.9 Globale Testszenarien und Testfälle 15](#_Toc129277189)

[2 Projektmanagement 16](#_Toc129277190)

[2.1 Arbeitsaufteilung 16](#_Toc129277191)

[2.2 GANTT-Diagramme 16](#_Toc129277192)

[2.3 Projektverwaltung 16](#_Toc129277193)

[2.3.1 Versionskontrolle 16](#_Toc129277194)

[2.3.2 Kollaboration 16](#_Toc129277195)

[2.3.3 Integrierte Tools 16](#_Toc129277196)

[2.3.4 Veröffentlichung 16](#_Toc129277197)

[3 Grundlagen und Methoden 17](#_Toc129277198)

[3.1 Grundlagen Filter 17](#_Toc129277199)

[3.1.1 Warum werden Filter benötigt? 17](#_Toc129277200)

[3.1.2 Analoge Filter 17](#_Toc129277201)

[3.1.3 Aktive Filter 18](#_Toc129277202)

[3.1.4 Unterschied zwischen aktiven und passiven Filtern 18](#_Toc129277203)

[3.1.5 Unterschied zwischen analoge und digitale Filter 19](#_Toc129277204)

[3.2 Aktiver Filter 20](#_Toc129277205)

[3.2.1 Sallen-Key-Filter 20](#_Toc129277206)

[3.2.2 Butterworth 22](#_Toc129277207)

[3.2.3 Mathematische Beschreibung 23](#_Toc129277208)

[3.3 Grundlagen der digitalen Signal-Verarbeitung (DSV) 27](#_Toc129277209)

[3.3.1 Einleitung 27](#_Toc129277210)

[3.3.2 Fouriertransformation 27](#_Toc129277211)

[3.3.3 Wie funktioniert eine FFT 29](#_Toc129277212)

[3.3.4 Erklärung der FFT in der Firmware 29](#_Toc129277213)

[3.4 C-Programmiersprache 37](#_Toc129277214)

[3.4.1 Agile Softwareentwicklung mit C 37](#_Toc129277215)

[3.5 ESP32 37](#_Toc129277216)

[3.5.1 Stromversorgungs-Kriterien 37](#_Toc129277217)

[3.5.2 Bootloader Brennen mit dem ESP-Tool 37](#_Toc129277218)

[3.5.3 Einrichten in PlatformIO (PIO) 37](#_Toc129277219)

[3.5.4 ESP IDF 37](#_Toc129277220)

[3.5.5 Direct Memory Access (DMA) Controller 37](#_Toc129277221)

[3.6 Realtime Operating-System (RTOS) 37](#_Toc129277222)

[3.7 Serial Peripheral Interface (SPI) 37](#_Toc129277223)

[3.8 Inter-IC-Sound (I2S) 37](#_Toc129277224)

[3.9 Musical Instrument Digital Interface (MIDI) 37](#_Toc129277225)

[3.9.1 Kommandos 37](#_Toc129277226)

[3.10 Universal Asynchronous Receive and Transmit (UART) 37](#_Toc129277227)

[4 Ergebnisse 38](#_Toc129277228)

[4.1 Blockschaltbild 38](#_Toc129277229)

[4.2 Hardware 39](#_Toc129277230)

[4.2.1 Versorgung 39](#_Toc129277231)

[4.2.2 Analogue-Frontend (AFE) 42](#_Toc129277232)

[4.2.3 Digital Frontend 43](#_Toc129277233)

[4.2.4 Layout der Platine 46](#_Toc129277234)

[4.3 Firmware Referenz Handbuch 50](#_Toc129277235)

[4.3.1 Toolchain 50](#_Toc129277236)

[4.3.2 Daten Strukturen Index 50](#_Toc129277237)

[4.3.3 File Index 50](#_Toc129277238)

[4.3.4 Daten Strukturen Dokumentation 50](#_Toc129277239)

[4.3.5 File Dokumentation 50](#_Toc129277240)

[4.4 Software 51](#_Toc129277241)

[4.4.1 Virtueller MIDI-Port 51](#_Toc129277242)

[4.4.2 MIDI Serial Bridge 51](#_Toc129277243)

[4.4.3 Ableton Live Setup 52](#_Toc129277244)

[4.5 CAD-Modelle und Gehäuse 52](#_Toc129277245)

[5 Fehlererfassung 54](#_Toc129277246)

[5.1 Platine 54](#_Toc129277247)

[5.2 Bestückung 54](#_Toc129277248)

[5.3 ADC Channel 2 auf ADC Channel 1 überbrücken 54](#_Toc129277249)

[5.4 Fehlende Features 55](#_Toc129277250)

[6 Glossar 56](#_Toc129277251)

[7 Abbildungsverzeichnis 57](#_Toc129277252)

[8 Quellenverzeichnis 58](#_Toc129277253)

[9 Anhang 59](#_Toc129277254)

[9.1 PCB-Fertigungsunterlagen 59](#_Toc129277255)

# Überblick

# Systemspezifikationen

## Zielbestimmungen

### Musskriterien

Einzelne Noten und Akkorde müssen zuverlässig erkannt und umgewandelt werden.

### Wunschkriterien

Noten sollen mit möglichst geringer Latenz[[1]](#footnote-2) übertragen werden.

### Abgrenzungskriterien

Projekt soll nicht auf verschiede E-Gitarren getestet und optimiert werden. Das Produkt soll nicht auf Basis anderer Musikinstrumente funktionieren, welche ähnliche elektrische Ausgänge haben.

## Produkteinsatz

### Anwendungsbereiche

Der Anwendungsbereich findet sich in der Musikproduktion als innovatives Notationstool und im Lehrbereich um Anfängern das Erlernen des Notenlesens zu erleichtern.

### Zielgruppen

Zielgruppen sin sowohl Musikproduktions-Neueinsteiger, welche Gitarre spielen als auch bereits Erfahrene Produzenten, welche auf der Suche nach Einzigartigen und Inspirierenden Eingabemethoden sind.

### Betriebsbedingungen

Die Versorgung sowie die Datenübertragung erfolgen über USB. Hierzu wird eine E-Gitarre via einer 6,3mm Buchse an die Platine angeschlossen. Da kein Überspannschutz vorliegt darf der Eingang nur mit einer geringen Leistung beschalten werden.

## Produktumgebung

### Software

* Silicon Labs VCP Driver
* Hairless MIDI (v0.4)
* LoopMIDI (v1.0.16)
* Ableton Live Suite (v11.2.7)
* Gitcon Firmware

### Hardware

* Selbst entwickelte ESP32 32-Bit Mikroprozessor Platine
* 6,3mm Mono Audio Klinkenstecker
* Micro USB (AB) Male to USB (A) Male
* Gitarre mit elektrischem Ausgang

## Produktfunktionen

/F0010/ Transienten Erkennung:

Es wird erkannt, ob eine neue Saite angeschlagen wird.

/F0020/ Noten Erkennung:

Die Note der angeschlagenen Saite wird erkannt und anschließend in das MIDI-Format konvertiert.

/F0030/ USB Kommunikation:

Das Gerät wird als MIDI Device in der DAW erkannt, uns sendet die mit der Gitarre Gespielten Noten an einen Kanal des virtuellen MIDI-Ports am PC.

/F0050/ Ausgabe:

Eingelesene Noten werden bei aktivierter Aufzeichnung in der DAW auf Pianorolls angezeigt und gespeichert.

## Produktleistungen

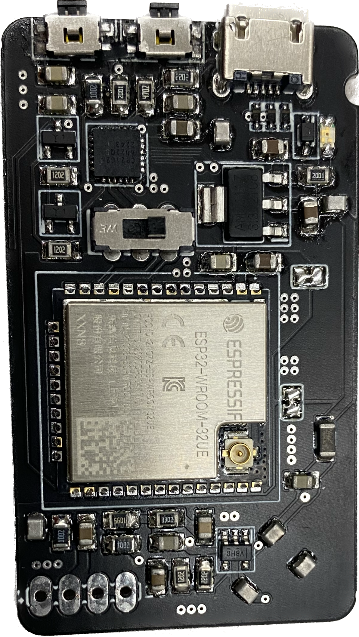
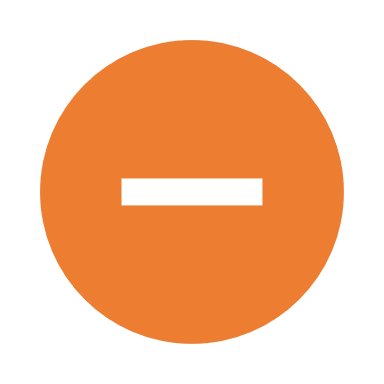
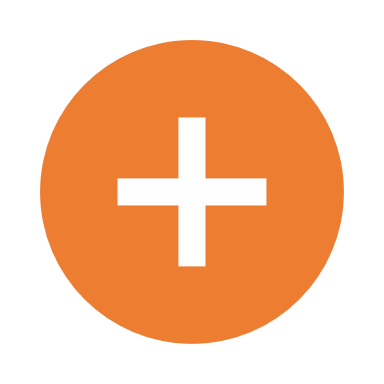
/L010/ Latenz:

Die Noten sollen ohne große Verzögerungen ankommen und so einen Liveeinsatz ermöglichen.

/L020/ Genauigkeit:

Das Signal soll zuverlässig in die richtigen Frequenzen aufgespalten werden.

## Benutzungsoberfläche



/B001/ USB-Buchse

An der Micro-USB Buchse werden die an den PC zu übertragenden Daten bereitgestellt. Die USB-Schnittstelle wird zum Programmieren, sowie zum Übertragen von MIDI-Daten benutzt.

/B002/ Boot-Mode Taster

Wird der ESP32, während der Boot-Mode Taster gedrückt ist, zurückgesetzt, so wechselt er in den Download-Modus. Im Download-Modus kann dann eine neue Firmware auf den ESP32 gespielt werden.

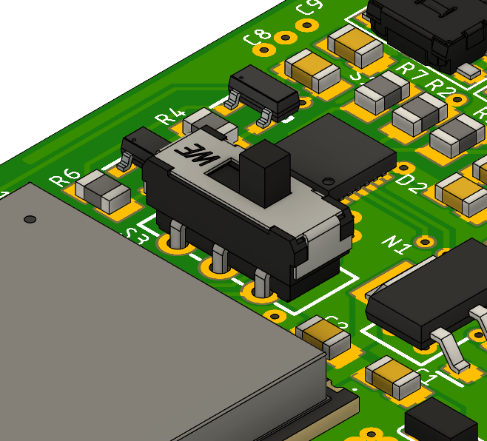
/B003/ Reset Taster

Der Reset-Taster setzt den ESP32 zurück.

/B004/ Analoger Eingang

Am analogen Eingang werden die beiden 6,3mm Mono-Audio-Buchsen, welche am Gehäuse befestigt sind, angeschlossen.

/B005/ Slider-Switch



R

L

Mit dem Slider Switch, wird der UART-Kanal ausgewählt. Ist der Slider-Switch in der Links-Stellung (L), so kann der ESP32 über die USB-Schnittstelle programmiert bzw. gedebuggt werden. Hat man die Rechts-Stellung (R) selektiert, ist man im MIDI-Übertragungsmodus. Auf die Schnittstelle werden nun sämtliche erkannte Noten Übertragen.

## Entwicklungsumgebung

### Software

* PlatformIO (Core v6.1.6, Home v3.4.3)
* ESP IoT Development Framework (v5.3.0)
* ESP-tool
* LTSpice XVII
* Saturn PCB Toolkit (v8.23)
* Ableton Live Suite (v11.2.7)
* Audacity

### Hardware

* Prototypen
  + Firmware Test-board
  + Filter Prototyp

### Orgware

* GitHub (<https://github.com/s-grundner/MTAP-MIDI-Guitar-Converter>)
* DrawIO/diagrams.net
* Obsidian (v1.1.9)
* Pro Create (v5.3.1)

## Qualitätsziel Bestimmungen

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | sehr wichtig | wichtig | weniger wichtig | unwichtig |
| Robustheit |  |  | x |  |
| Zuverlässigkeit | x |  |  |  |
| Korrektheit | x |  |  |  |
| Benutzungsfreundlichkeit |  |  | x |  |
| Effizienz | x |  |  |  |
| Portierbarkeit |  | x |  |  |
| Kompatibilität |  |  | x |  |

## Globale Testszenarien und Testfälle

# Projektmanagement

## Arbeitsaufteilung

Daniel Bräumann

Simon Grundner

Laurenz Hölzl

## GANTT-Diagramme

## Projektverwaltung

Um das Projekt während der Entwicklungsphase zu verwalten, wurde Github verwendet. Github ist eine weitverbreitete, webbasierte Plattform, mit welcher man das Projekt, in einer sogenannten „Repository“ ablegen und von dort aus verwalten kann. Besonders bei Softwareprojekten erleichtert Github den Arbeitsfluss immens, da die Plattform einige ausschlaggebende Benefits mit sich bringt.

### Versionskontrolle

Commits

Pullrequests

Branches

Rollback/Reset

### Kollaboration

### Integrierte Tools

Trello

Github Copilot

### Veröffentlichung

# Grundlagen und Methoden

## Grundlagen Filter

### Warum werden Filter benötigt?

Filter werden in der Elektronik verwendet, um unerwünschte Frequenzen oder Signale aus einem Signal zu entfernen oder zu reduzieren. Ein Filter kann in der Lage sein, ein bestimmtes Frequenzband durchzulassen und alles außerhalb dieses Bereichs zu unterdrücken, oder es kann in der Lage sein, bestimmte Frequenzen zu unterdrücken und den Rest durchzulassen.

Filter werden in vielen Bereichen der Elektronik eingesetzt, wie zum Beispiel in der Tontechnik, wo sie zur Entfernung von Rauschen und anderen Störungen verwendet werden, oder in der Datenkommunikation, wo sie zur Verarbeitung von Signalen und zur Entfernung von Interferenzen eingesetzt werden.

Insgesamt sind Filter ein wichtiges Werkzeug in der Elektronik, um die Qualität von Signalen zu verbessern und unerwünschte Störungen zu entfernen.

### Analoge Filter

#### Passive Filter

Ein Bild, das Text, Uhr enthält.

Automatisch generierte BeschreibungEin passiver Filter besteht ausschließlich aus passiven Bauelementen, wie Widerstand, Spule und Kondensator. Mithilfe dieser Bauteile kann man keine Leistungsverstärkung erzielen. Weiters benötigt dieser Filter keine externe Stromquelle für den Betrieb. Da die Eingangsimpedanz niedrig und die Ausgangsimpedanz hoch ist, wird eine Selbstregulierung der Spannungen ermöglicht, die die Lasten antreibt.

Der Lastwiderstand ist normalerweise nicht vom restlichen Netzwerk isoliert, dadurch kann die Charakteristik der Schaltung und des Filterprozess bei Veränderung der Last beeinflusst werden. Da ein passiver Filter keine Bandbreitenbeschränkungen aufweisen, ermöglicht dies einen zufriedenstellenden Betrieb bei sehr hohen Frequenzen. Allerdings tendiert der in der Schaltung verwendete Induktor bei niedrigeren Frequenzen größer zu sein, wodurch die gesamte Schaltung komplexer wird. Außerdem steigen die Kosten, wenn eine höhere Qualität und eine kleine Größe erwünscht ist. Weiters erzeugen passive Filter aufgrund des thermischen Rauschens in den Elementen ebenfalls ein hörbares Rauschen. Jedoch kann dies bei richtiger Auslegung der Bauteile minimiert werden.

Weil keine Verstärkung vorhanden ist, muss diese zu einem späteren Zeitpunkt durchgeführt werden. Dazu werden oft Pufferverstärker verwendet, um die Differenzen in der Ausgangsschaltung zu kompensieren.

### Aktive Filter

Im Gegensatz zu passiven Filtern, die nur aus passiven Bauelementen bestehen, kommen bei aktiven Filtern Transistoren oder Operationsverstärker zum Einsatz, außerdem werden keine Induktoren verwendet. Anders als bei passiven Filtern benötigen aktive Filter aufgrund der energieverbrauchenden/aktiven Elementen eine externe Stromquelle.

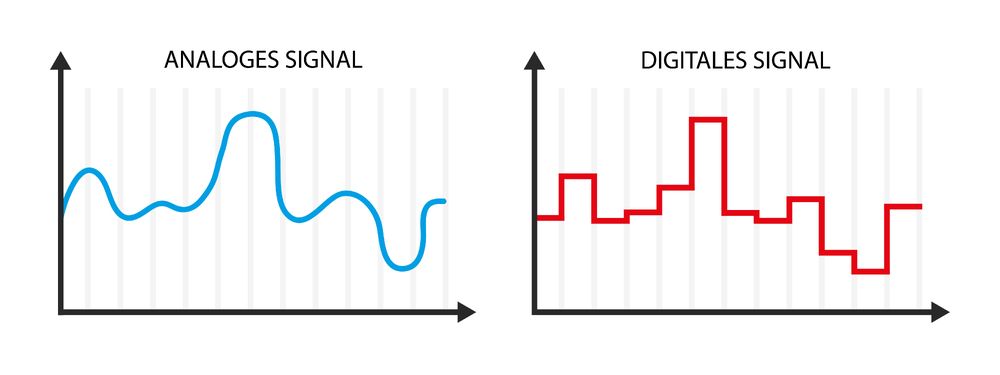
Da keine Induktoren zum Einsatz kommen, wird die Schaltung kompakter und weniger schwer. Die Eingangsimpedanz ist hoch und die Ausgangsimpedanz ist niedrig, so können niedrige Lasten am Ausgang angesteuert werden. Weiters ist die Last von der internen Schaltung isoliert, daher hat die Veränderung der Last keinen Einfluss auf die Charakteristik des Filters.

Das Ausgangssignal hat eine Leistungsverstärkung, auch können die Parameter wie Verstärkung und Grenzfrequenz beliebig angepasst werden. Probleme bei aktiven Filtern sind, dass eine Änderung in der Stromversorgung eine Änderung der Ausgangssignalgröße verursachen kann, weiters werden die Hochfrequenzbereiche durch die Eigenschaften der aktiven Elemente begrenzt. Außerdem können Rückkopplungsschleifen, die zur Regelung der aktiven Komponenten verwendet werden, zu Schwingungen und Rauschen beitragen.

### Unterschied zwischen aktiven und passiven Filtern

* Bei passiven Filtern wird die Energie des Signals verbraucht, es ist jedoch keine Leistungsverstärkung verfügbar ist, während bei aktiven Filtern eine Leistungsverstärkung zur Verfügung steht.
* Aktive Filter benötigen eine externe Stromquelle. Passive Filter arbeiten nur am Signaleingang.
* Ausschließlich passive Filter verwenden Induktivitäten.
* Aktive Filter verwenden Transistoren und Operationsverstärker, die aktive Bauelemente sind.
* Passive Filter haben theoretisch keine Frequenzbegrenzungen, während aktive Filter von den aktiven Elementen eine Einschränkung aufweisen.
* Außerdem sind passive Filter etwas stabiler und können auch großen Strömen standhalten.
* Passive Filter sind preiswerter als aktive Filter, jedoch sind aktive Filter meist kompakter.

### Unterschied zwischen analoge und digitale Filter

Analoge Filter sind leicht zu implementieren, da man das zu filternde kontinuierliche Signal am Eingang einspeisen kann. Digitale Filter hingegen, arbeiten zeitdiskret. Anstatt eine Analoge Schwingung, nehmen sie einen Datenstrom als Eingang, welcher aus **Samples** (Stichproben) des Signals besteht. Um das Signal in die Samples zu zerlegen, liest ein Analog-Digital-Umsetzer in jeder Abtastperiode den Momentanwert des Signals ein. Die Abtastfrequenz wird dabei so Hoch gewählt, dass das Signal aus den Samples möglichst genau rekonstruiert werden kann. (siehe Shannon-Nyquist Theorem)

Digitale Filter bieten einige Vorteile. Darunter zählt, dass ein einziger Filter mehrere Eingangssignale filtern kann, ohne etwas an der Hardware geändert zu haben. Weiters variiert die Leistung nicht mit den Umgebungsbedingungen, wonach immer eine konstante Leistung herrscht.

Die Hauptanwendungen für digitale Filter sein zum Beispiel die Trennung von kombinierten Signalen und die Wiederherstellung von verzerrten Signalen. Digitale Filter erzielen hier meist bessere Ergebnisse, aber diese Probleme sind auch mittels Analogfilter problemlos zu lösen.

Weiters unterscheiden sie sich bei der Bandbreite. Während bei Digitalfiltern die Bandbreite durch den Start der Aufnahme eingestellt wird, gilt diese bei Analogfiltern als unbegrenzt, wodurch die Auflösung nicht als statisch, sondern variabel gilt und jederzeit erhöht oder gesenkt werden kann, ohne Qualität einbüßen zu müssen.

Nachteile von digitalen Filtern sind, dass sie wesentlich teurer als Analogfilter sind und dazu eine höhere Latenz aufweisen. Dazu kommen noch eine geringere Bandbreite und ein störendes Quantisierungsrauschen.

Analoge Filter werden in einer Vielzahl von Anwendungen eingesetzt, insbesondere in der Elektronik, in der Signalaufbereitung und in der Tontechnik. Hier sind einige Beispiele für Anwendungen von analogen Filtern:

1. Tontechnik: Analogfilter werden in Audiogeräten wie Verstärkern, Equalizern und Lautsprechersystemen eingesetzt, um unerwünschte Rausch- und Störgeräusche herauszufiltern und die Klangqualität zu verbessern.
2. Telekommunikation: Filter werden in Telekommunikationssystemen verwendet, um das Signal-Rausch-Verhältnis zu verbessern und unerwünschte Signale zu unterdrücken. Zum Beispiel werden Filter in Mobilfunkgeräten eingesetzt, um Interferenzen von benachbarten Frequenzen zu reduzieren.
3. Leistungselektronik: Analogfilter werden in der Leistungselektronik eingesetzt, um Störungen und Störsignale aus den Schaltkreisen zu entfernen und die Leistungseffizienz zu verbessern.
4. Medizinische Geräte: Filter werden in medizinischen Geräten wie Elektrokardiogrammen (EKG) eingesetzt, um Rauschen aus dem Signal zu entfernen und klare Messergebnisse zu erhalten.
5. Radartechnik: Filter werden in Radarsystemen verwendet, um das Echosignal zu glätten und die Entfernung und Position von Zielen genauer zu bestimmen.

Diese sind nur ein paar Beispiele von vielen Anwendungen, in denen analoge Filter in der Elektronik eingesetzt werden. In der Regel werden analoge Filter dort eingesetzt, wo die Signale in kontinuierlicher Form vorliegen und es wichtig ist, spezifische Frequenzen herauszufiltern oder zu unterdrücken.

#### Shannon-Nyquist Theorem

Das Shannon-Nyquist Theorem ist ein Abtasttheorem, welches besagt, dass die Abtastfrequenz eines analogen Signals mindestens doppelt so hoch wie die höchste Frequenzkomponente des Signals sein muss. Dies ist aufgrund der sogenannten Nyquist-Frequenz möglich, die die höchste Frequenz angibt, die durch die Abtastung erfasst werden kann.

Dieses Abtasttheorem ist vor allem in der digitalen Signalverarbeitung und digitalen Kommunikation von Bedeutung, da es als Grundlage für die Abtastung, Kodierung und Quantisierung gilt.

## Aktiver Filter

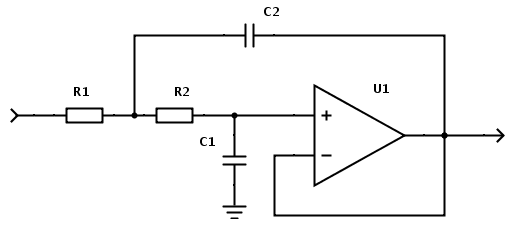
### Sallen-Key-Filter

Der Sallen-Key Filter ist einer der meistgenutzten Filter in der Signalverarbeitung, um Frequenzen in einem Signal zu verstärken oder abzuschwächen. Der Filter besteht lediglich aus einem Operationsverstärker und einigen passiven Bauelementen, weswegen er als aktiver Filter bezeichnet wird. Es ist wichtig zu beachten, dass der Sallen-Key eine Filtertopologie und keine Filtercharakteristik, wie Butterworth, Bessel usw. Jedoch können verschiedene Charakteristiken in verschiedene Topologien implementiert werden, je nach Änderung der Komponentenwerte verändert sich die Filtercharakteristik.

Sallen-Key werden meist als Tief- oder Hochpass verwendet, wobei bei Erweiterung auch ein Bandpass ermöglicht wird. Weiters weist der Sallen-Key eine gute Linearität und eine geringe Verzerrung auf, was es zu einer effektiven Lösung für die Signalverarbeitung macht. Dazu ist diese Topologie sehr leicht zu realisieren und auch in der Lage eine hohe Güte zu erreichen, was dazu führt, dass unerwünschte Frequenzen sich sehr stark unterdrücken lassen oder spezifische Frequenzen sehr genau verstärken lassen.

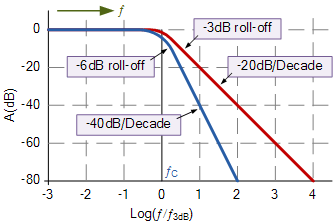
Jedoch gibt es auch Nachteile des Sallen-Key Filters. Dazu zählen eine begrenzte Bandbreite und hohe Empfindlichkeit bei Veränderung der passiven Komponenten, die zu einer Verschiebung der Filtercharakteristik führen können. Daher sollte die Überwachung der Filtercharakteristik an oberster Stelle stehen, um eine unerwünschte Veränderung zu bemerken.

#### Unterschiede in den Ordnungen

Ein Bild, das Antenne enthält.

Automatisch generierte BeschreibungFilter gibt es in diversen Ordnungen, mit jeweils anderen Eigenschaften. So ist es auch beim Sallen-Key der Fall. Eine Änderung, die sofort auffällt, ist der Aufbau der Schaltung. Während beim Sallen-Key 1. Ordnung nur jeweils ein Widerstand und Kondensator verwendet werden, benötigt ein Filter 2. Ordnung 4 passive Bauteile.

Abbildung : Filter 1. Ordnung

Weiters unterscheiden sich die Ordnungen auch in der Flankensteilheit. In erster Ordnung fällt die Kurve um 20dB/Dekade (6dB/Oktave), bei einem Sallen-Key 2. Ordnung beträgt die Flankensteilheit 40dB/Dekade (12db/Oktave).

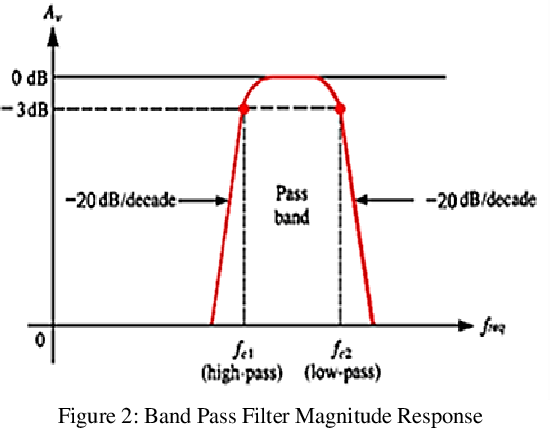
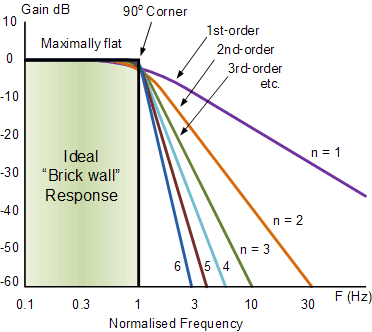
Während der Sallen-Key 1. Ordnung nur eine Resonanzfrequenz hat, hat der Sallen-Key 2. Ordnung zwei Resonanzfrequenzen. Die Resonanzfrequenzen beeinflussen die Filtercharakteristik maßgeblich in deren Umgebung. Auch die Dämpfung verändert sich je nach Ordnung. Das bedeutet, dass ein Filter 2. Ordnung eine höhere Dämpfungsfähigkeit aufweist und somit unerwünschte Frequenzen besser unterdrücken, kann als ein Filter 1. Ordnung.

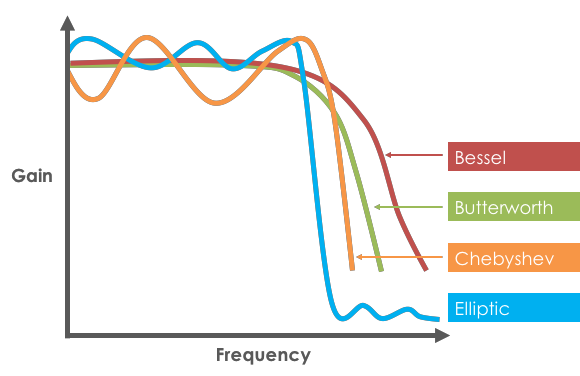
Abbildung : Bandpassfilter Übertragungsfunktion

### Butterworth

Die Butterworth Filtercharakteristik ist eine der beliebtesten Charakteristiken, da sie eine auf einer flachen Übertragungsfunktion basiert, was bedeutet, dass die Verstärkung im Frequenzbereich möglichst gleichmäßig ist. Weiters weist sie eine maximale lineare Phasenreaktion auf, was bedeutet, dass alle Frequenzen innerhalb des Durchlassbereichs des Filters ohne Phasenverschiebung passieren. Dies ist ein wichtiger Vorteil bei der Verarbeitung von Signalen, bei denen eine gleichmäßige Phasenreaktion erforderlich ist, wie zum Beispiel bei der Signalübertragung oder in der Audio- und Musikproduktion.

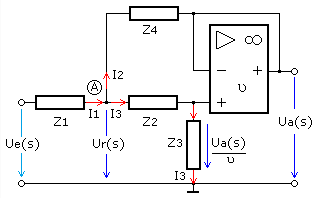
Die Übertragungsfunktion ist charakterisiert durch eine Glättungskurve, die zu einem sanften Roll-Off führt. Dies führt zu einem Kompromiss zwischen der Dämpfung von unerwünschten Frequenzen und der Aufrechterhaltung einer möglichst linearen Phasenreaktion innerhalb des Durchlassbereichs.

Außerdem ist die Butterworth Charakteristik für ihre hohe Stabilität bekannt. Anders als bei anderen Filtertypen, wie Chebyshev- oder Elliptischen-Filter, die ein Rippelverhalten in der Übertragungsfunktion aufweisen, weist ein Butterworth-Filter keine unerwünschten Spitzen oder Wellen auf. Dadurch wird die Filterleistung und Genauigkeit der Signalverarbeitung verbessert.

Allerdings benötigt die Butterworth-Topologie meist eine höhere Ordnung, um eine ausreichende Dämpfung für die unerwünschten Frequenzen zu erhalten, was meist zu einem erhöhten Komponentenbedarf führt. Weiters ist er beim Entfernen jener Frequenzen, welche nahe der Grenzfrequenz liegen weniger effektiv als zum Beispiel ein Chebychev-Filter.

### Mathematische Beschreibung

#### Allgemeine Herleitung der Übertragungsfunktion

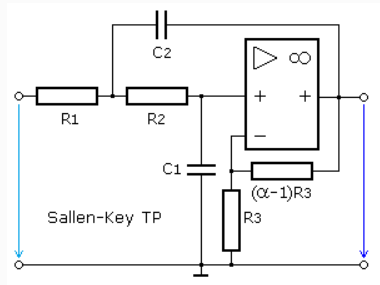
Bei der allgemeinen Herleitung der Übertragungsfunktion werden die Bauteile durch Impedanzen ersetzt. Außerdem wird der Operationsverstärker als ideal angenommen, deshalb hat diese Schaltung bei A nur einen Knotenpunkt. Es wird eine komplexe Frequenzvariable p = s = jω verwendet, um die Herleitung übersichtlicher zu gestalten. Diese Herleitung kann zur Weiteren Berechnung eines Tiefpasses oder Hochpasses verwendet werden.

durch Einsetzen den Teilgleichungen:

mit:

Allgemeine Übertragungsfunktion:

#### Sallen-Key-Tiefpass 2. Ordnung

Für die obige Schaltung wird bei einen Tiefpass 2. Ordnung die Impedanzen Z1 und Z2 ohmsche Widerstände eingesetzt. Für Z3 und Z4 werden kapazitive Blindwiderstände verwendet.

Sallen-Key TP mit Verstärkung:

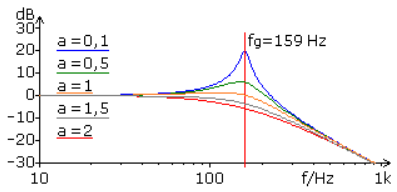
allgemein normierter TP 2. Ordnung:

Mit s = jω folgt:

Durch einen Vergleich sich entsprechender Komponenten im normierten allgemeinen TP-Filter 2.O. können Grenzfrequenzen und Dämpfungswerte für unterschiedliche Dimensionierungen von R und C ermittelt werden. Die Simulationsergebnisse für unterschiedliche Verstärkungen sind auf den Ausgangswert 0dB gesetzt. Die Amplitudenerhöhung nimmt bei Verstärkung größer 1,5 deutlich zu.

Nach Einsetzen von :

Mit und

Wenn Widerstände und Kondensatoren unterschiedliche Werte haben:

## Grundlagen der digitalen Signal-Verarbeitung (DSV)

### Einleitung

Schlägt man auf einer Gitarre eine Saite an, so schwingt diese, je nach Stimmung, mit einer gewissen Frequenz. Zur Erkennung der Frequenzen und daraus folgend der Noten, wird der Fast Fourier Transform (FFT) Algorithmus verwendet. Mithilfe des Algorithmus können einzelnen Frequenzen, aus welchen ein Signal besteht, ermittelt werden. Mit den ermittelten Frequenzen soll nun der Grundton des Signals bestimmt werden.

Aufgrund des Aufbaus einer E-Gitarre gibt es neben der Grundfrequenz aber noch eine Vielzahl anderer Schwingungen, welche beispielsweise durch das Kabel, Holz oder Pickup der Gitarre auftreten können.

Spielt man einen Ton auf einem Instrument so treten neben dem Grundton noch sogenannte Obertöne oder Teiltöne auf. Diese sind höher als der tatsächlich gespielte Grundton und stellen bei dessen Erkennung eine maßgebliche Herausforderung dar.

Um eine möglichst Oberton freies Signal zu gewährleisten ist es ratsam den Pickup, welcher am Hals sitzt zu wählen, da die Brücke einige Schwingungen reflektiert, was das Signal wiederum beeinflusst.

Ebenfalls ist es Ratsam kein Plektrum zu verwenden, da dies auch Auswirkungen auf die Auftretenden Obertöne hat.

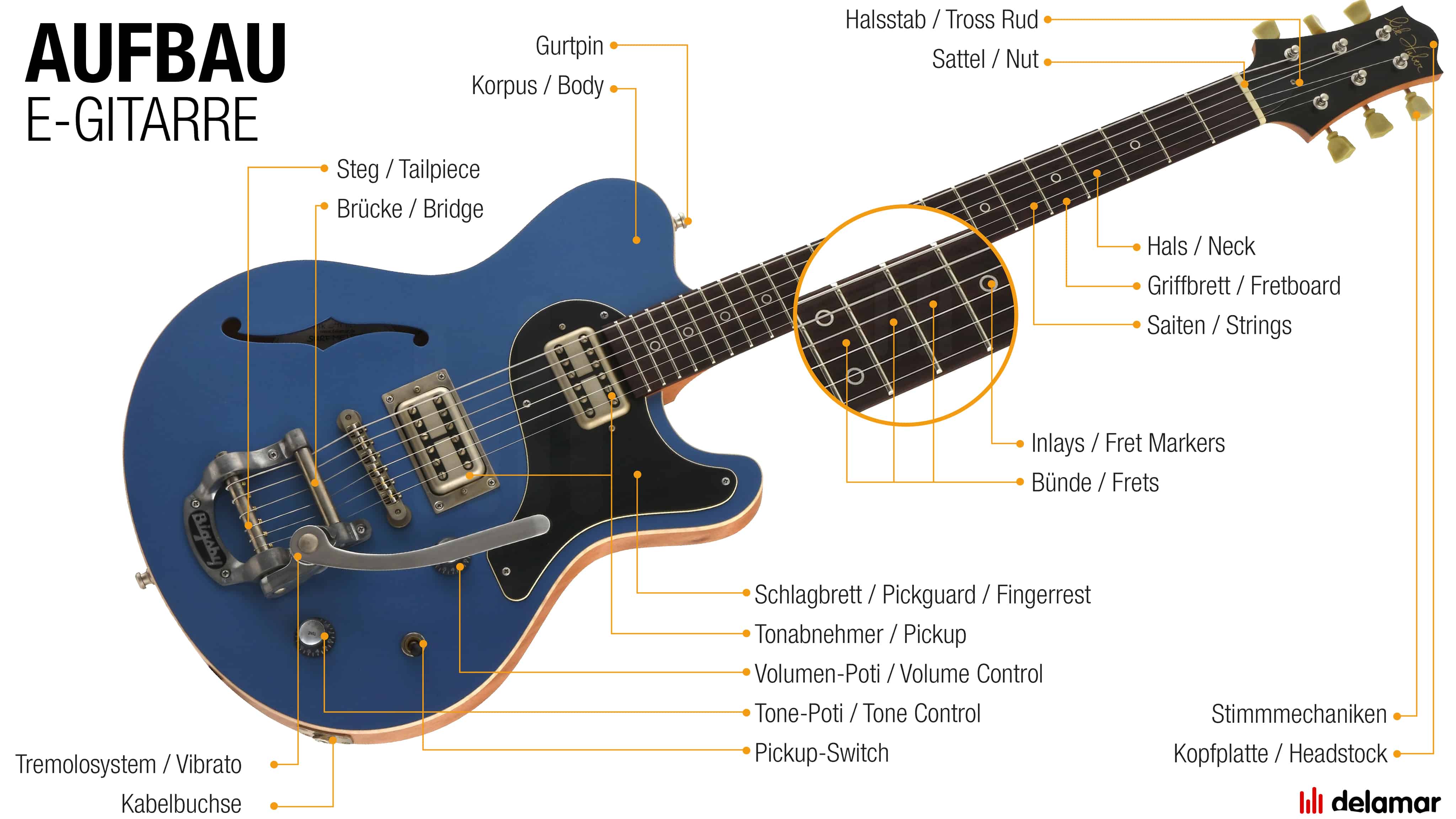


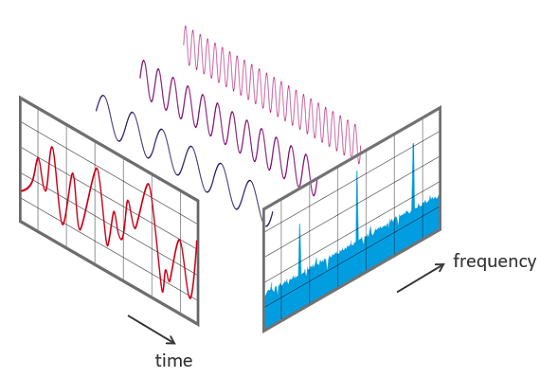
Abbildung : Aufbau einer E-Gitarre

### Fouriertransformation

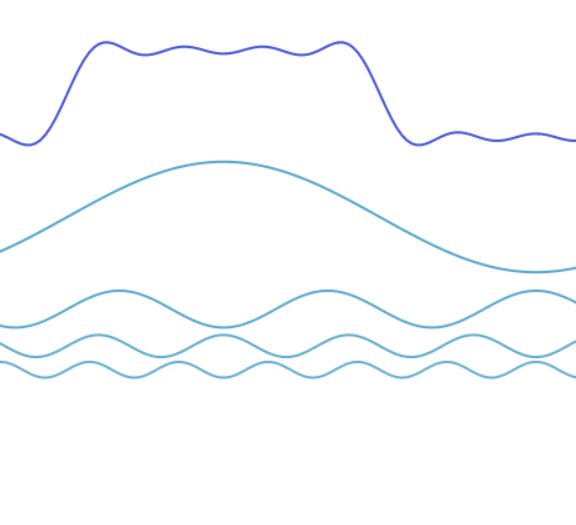
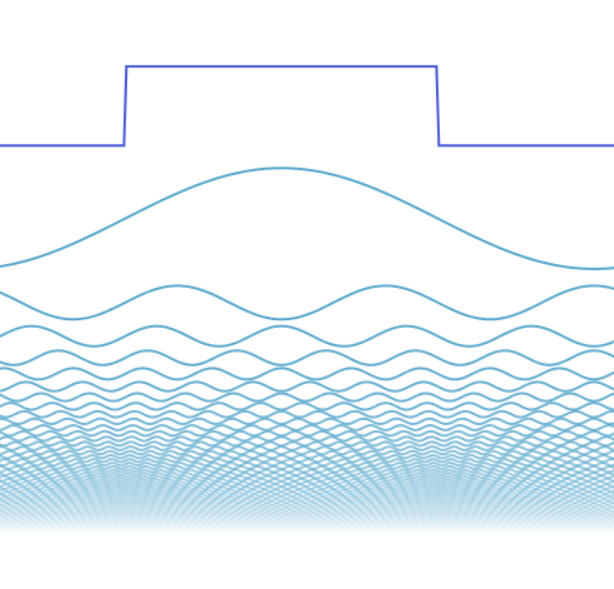
#### Was ist eine Fouriertransformation?

Die Fourier Transformation ist eine Integraltransformation, welche 1822 von Jean Baptiste Joseph Fourier eingeführt wurde und genutzt werden kann, um das diskrete Frequenzspektrum periodischer Signale zu ermitteln. Sie ist in vielen Bereichen von Wissenschaft und Technik unerlässlich und nahezu jeder nutzt sie täglich beim z.B. Streamen von Videos. Um sie von einem Computer ausführen zu lassen gibt es die Diskrete Fourier Transformation (DFT) und die Fast Fourier Transformation (FFT).

Eigentlich handelt es sich beim FFT-Algorithmus nur um eine Möglichkeit die DFT mit hoher Geschwindigkeit durchzuführen. James Cooley und John W. Tukey wiederentdeckten ihn im Jahr 1965. Wiederentdeckung deshalb, weil Carl Friedrich Gauß den Algorithmus in seiner ersten Form bereits 1805 entdeckte und verwendete, diese aber nie publizierte. Erst die hohe Durchführungsgeschwindigkeit machte eine so breit gefächerte Anwendung möglich.



Prinzipiell kann die Fouriertransformation genutzt werden, um ein Signal in eine Menge von Sinusschwingungen zerlegt zu werden.



Hier sieht man beispielsweiße ein zerlegtes Rechtecks Signal. Links mit sehr hoher und links mit niedriger Auflösung.

Nach diesem Prinzip lässt sich jedes Signal zerlegen. Das ist beispielsweise bei der Komprimierung von Audiodaten wichtig, da durch die Transformation nur die Formel für die Sinusse anstatt alle einzelnen Datenpunkte gespeichert werden müssen.

### Wie funktioniert eine FFT

### Erklärung der FFT in der Firmware

Das DSP-Programm, welches am ESP32 läuft, führt eine FFT mit dem Eingangssignal durch. Um ein kontinuierliches Einlesen der Gitarrensignale zu ermöglichen wird der DMA-Controller genutzt.

Wegen ihrer einfachen Anwendung, guten Dokumentation und ihrer hohen Performance wurde die FFT-Bibliothek von Robin Scheibler verwendet. Bei dieser Handelt es um eine Implementierung von radix-2, split-radix und einigen höheren Basen. Diese Basen limitieren die maximal mögliche Größe der FFT.

#### Testprogramm

#define NFFT 8192

#define F\_ABT 44100

Zuerst werden für NFFT die Größe des FFT-Buffers und für F\_ABT die in der Audiotechnik übliche Abtastrate von 44,1 kHz festgelegt.

#include <stdio.h>

#include <stdlib.h>

#include "esp\_log.h"

#include "freertos/FreeRTOS.h"

#include "freertos/task.h"

#include "driver/adc.h"

#include "esp\_adc\_cal.h"

#include <math.h>

#include "fft.h"

#include "processed-data.h" //enthält test\_buffer

Darauffolgend werden sämtliche benötigte Bibliotheken importiert. Die „freertos“-Bibliotheken werden von Task-System benötigt, welches das Betriebssystem des ESP32 mitliefert. (siehe 3.4…)  
Scheiblers FFT-Bibliothek ist in „fft.h“ enthalten.  
Der Header „processed-Data.h“ enthält die vorbereiteten Testdaten. (siehe 3.3.4.1.1 Testdatenfertigung)

float fft\_buffer[NFFT];

float magnitude[NFFT / 2];

float frequency[NFFT / 2];

float keyNR[NFFT / 2];

float ratio = (float)F\_ABT / (float)NFFT;

Dann werden all benötigten Buffer und Variablen initialisiert. Das Array „fft\_buffer“ wird nach den ausführen der FFT die Ergebnisse dieser beinhalte. Die drei Darauffolgenden beinhalten jeweils die Magnitude, die Frequenz und die am Piano korrespondierende Tastennummer. Der Float „ratio“ beinhaltet die Auflösung des Amplitudenspektrums in Hertz.

float getMaxMag();

float max;

Um unwesentliche Frequenzen relativ zur Lautstärke auszuschließen wird die Funktion getMaxMag() eingeführt welche die höchste auftretende Magnitude ermittelt.

float getMaxMag()

{

    float max = 0;

    for (int i = 0; i < NFFT / 2; i++)

    {

        if (magnitude[i] > max)

        {

            max = magnitude[i];

        }

    }

    return max;

}

Hierbei wird lediglich in einer For-Schleife überprüft, ob der nächste Wert höher ist als das momentan gespeicherte maximum. Sollte die der Fall sein wird der neue Wert als Maximum gespeichert.

Nun wird der „main-loop“ des Programms gestartet.

void app\_main(void)

{

fft\_config\_t \*real\_fft\_plan = fft\_init(NFFT, FFT\_REAL, FFT\_FORWARD, test\_buffer, fft\_buffer);

Als erstes wird hier die FFT mit den Argument Größe des Eingangsbuffers, Art der FFT, Eingangsbuffer und Buffer für die Ergebnisse konfiguriert.

Danach wird die FFT ausgeführt.

fft\_execute(real\_fft\_plan);

Die Ergebnisse werden anschließend verarbeitet.

for (int k = 1; k < NFFT / 2; k++)

{

magnitude[k] = 2 \* sqrt(pow(fft\_buffer[2 \* k], 2) + pow(fft\_buffer[2 \* k + 1], 2)) / NFFT;

frequency[k] = k \* ratio;

keyNR[k] = log2(frequency[k] / 440) \* 12 + 49;

}

max = getMaxMag();

Der abgebildete For-Loop ist für die Berechnung der Einzelnen Signifikanten Werte wie etwa Magnitude, Frequenz und die am Piano korrespondierende Tastennummer zuständig.

Zur Berechnung der Tastennummer wird die Formel

verwendet. Die 440Hz sind der Kammerton A. Hierbei ist zu beachten, dass, um die Midi-Notennummer zu erhalten noch 20 dazu addiert werden müssen, da Midi 128 mögliche Notennummern besitzt und die ersten 20 nicht belegt sind. Nach der Berechnung der Werte wird mit der zuvor beschriebenen Funktion getMaxMag() noch die höchste Magnitude ermittelt.

for (int k = 1; k < NFFT / 2; k++)

{

// printf("%d-th freq : %f+(%f)j\n", k, fft\_buffer[2\*k], fft\_buffer[2\*k+1]);

// printf("%f\t(%f)j\n", fft\_buffer[2\*k], fft\_buffer[2\*k+1]);

if (magnitude[k] > max\*0.5)

{

printf("%d-th magnitude: %f => corresponds to %f Hz\n", k, magnitude[k], frequency[k]);

printf("keyNR: %d\n", (int)round(keyNR[k]));

      }

      // printf("%f\n", magnitude[k]);

}

// printf("Middle component : %f\n", fft\_buffer[1]);

// N/2 is real and stored at [1]

Mit der If-Verzweigung wird überprüft, ob eine Magnitude einen Schwellenwert übersteigt, was bedeuten würde, dass diese Frequenz im Eingangssignal vorkommt.   
Die vereinzelten, auskommentierten „printf“ Funktionen dienen, um die einzelnen Zwischenwerte auszugeben um diese mit zum Beispiel Matlab zu Überprüfen. Im endgültigen Programm sind diese nicht mehr vorhanden da sie rein zum Debuggen dienen.

fft\_destroy(real\_fft\_plan);

Zuletzt wird die FFT-Konfiguration zerstört. Dies ist ein Wichtiger Schritt da für jeden Durchlauf des Algorithmus eine neue Konfiguration erstellt werden muss.

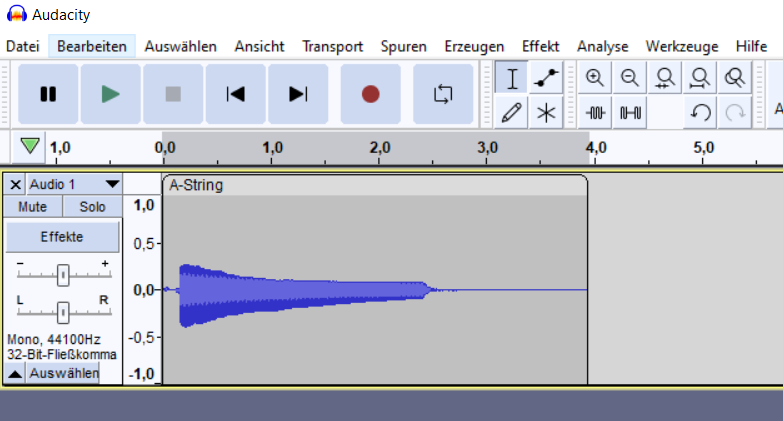
Hiermit ist das Ende des Testprogramms erreicht.

##### Testdatenfertigung

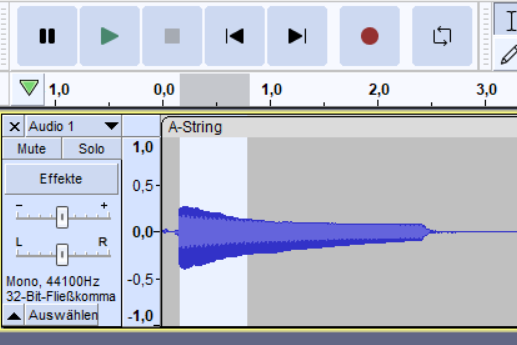
Zur Aufbereitung der Testdaten wurde das Audiobearbeitungsprogramm Audacity und ein eigens entwickeltes Python-Tool genutzt.

Für den Testdurchlauf wurde das Anschlagen einer A-Saite gewählt. Die Aufnahme erfolgt mit einer Samplerate von 44,1 kHz.

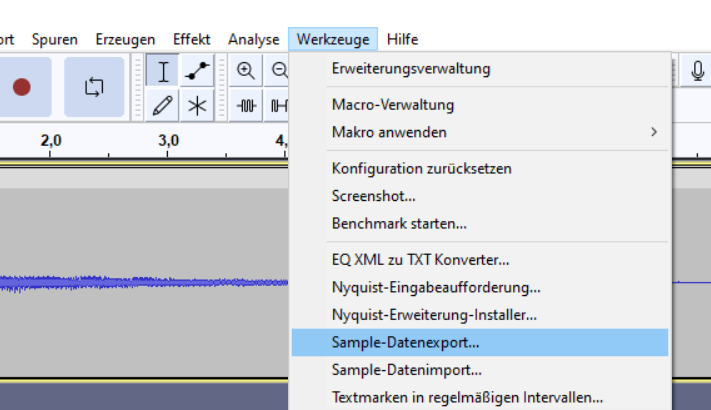
Im Audiobearbeitungsprogramm Audacity sieht das so aus:



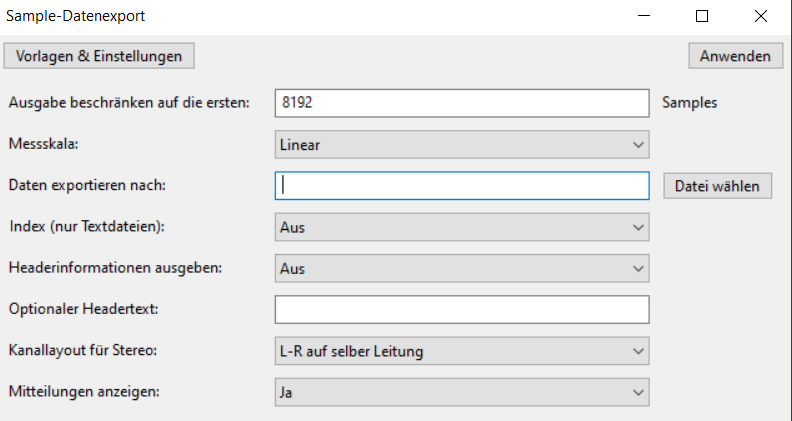
Um nun die einzelnen Punkte aus dem Signal zu extrahieren, muss ein Bereich im Signal ausgewählt werden.



Anschließend muss in der Menüleiste unter Werkzeuge die Option Sample-Datenexport gewählt werden.



Daraufhin öffnet sich das folgende Fenster.



Hier wird als erste Option die Anzahl der Samplepunkte gewählt. Da im Testprogramm für die maximale Buffergröße NFFT 8192 definiert wurde, wird dies hier ebenfalls gewählt. Die Option Messskala muss auf linear gestellt werden. Als Zielort wird der Pfad zum Python Tool angegeben.

Durch einen Klick auf Anwenden werden die Daten exportiert und das Fenster schließt sich.

###### Das Python Tool

Am zuvor festgelegten Zielort befindet sich nun die Datei „sample-data.txt“. Diese enthält einen Datenpunkt pro Zeile. Damit das Testprogramm damit arbeiten kann muss nun ein Array mit den Werten befüllt werden, welches in der Datei „processed-data.h“ erstellt wird.

Das Tool funktioniert folgendermaßen:

import shutil

Als erstes wird Pythons „shutil“-Modul importiert welche benötigt wird, um mit externen Dateien zu arbeiten.

arr=[]

arr = [0 for i in range(8192)]

i = 0

Danach wird ein Array mit der Größe 8192 erstellt in welches die Werte aus der „.txt“ Datei gespeichert werden. Die Variable „i“ dient hier als Zähler, welcher beim Befüllen noch wichtig wird.

with open('sample-data.txt') as f:

    while True:

        line = f.readline()

        if not line:

            break

        line = line.strip()

        data = float(line) \* 25

        #print(data)

        arr[i] = data

        i += 1

f.close()

Nun wird die Datei „sample-data.txt“ geöffnet. In der While-Schleife wird jede Zeile an die korrespondierende Stelle im Array gespeichert. Die If-Verzweigung beendet die Schleife sobald sie am Ende der Datei angelangt ist. Schlussendlich wird die Datei geschlossen.

file\_to\_delete = open("processed-data.h",'w')

file\_to\_delete.close()

Das Array soll in den Header „processed-data.h“ gespeichert werden, um es einfach in das Testprogramm integrieren zu können. Um sicherzustellen, dass keine Überschneidungen mit einer veralteten Version des Headers auftreten, wird diese kurzerhand gelöscht.

with open('processed-data.h', 'w') as d:

    d.write('float test\_buffer[] = {')

    for data in arr:

        d.write(str(data) + ',')

    d.write('};')

    d.close()

Im Anschluss wird der Header als leere Datei erstellt. Als erstes wird ein neues Array in c deklariert. Nach der geöffneten, geschwungenen Klammer wird der Inhalt des Python-Array in für die c-Syntax gültiger Form geschrieben und die geschlossene Klammer angehängt. Daraufhin wird die Datei geschlossen.

shutil.copyfile('processed-data.h', '/home/laurenz/Dokumente/GitHub/MTAP-MIDI-Guitar-Converter/firmware/ESP\_DSP/src/processed-data.h')

Im letzten Schritt wird die Datei noch in das Verzeichnis des Testprogramms kopiert.

Nach dem Ausführen des Tools ist der Test Buffer bereit und das Testprogramms kann ausgeführt werden

##### Interpretation der Ergebnisse des Testprogramms

Der Output nach einem Ausführen sieht folgendermaßen aus:

20-th magnitude: 2.373904 => corresponds to 107.666016 Hz

keyNR: 25

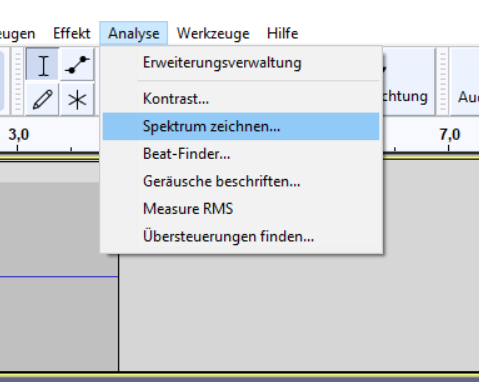
21-th magnitude: 2.073598 => corresponds to 113.049316 Hz

keyNR: 25

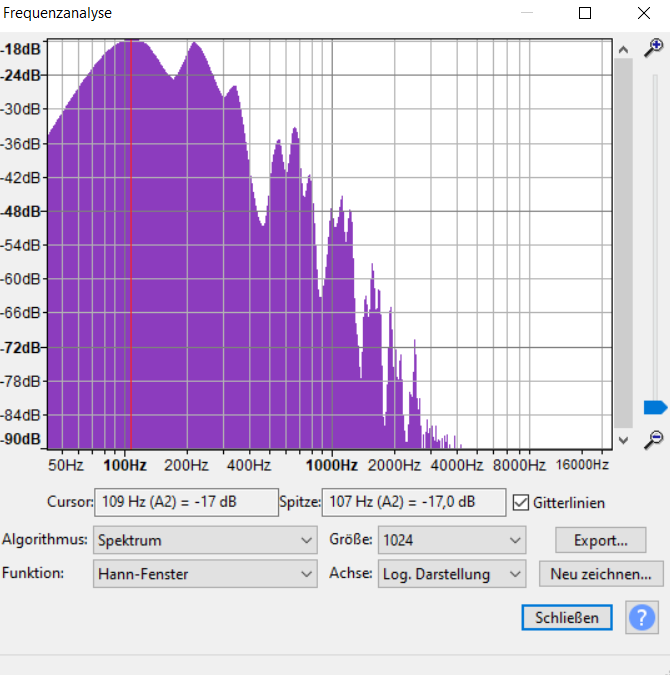
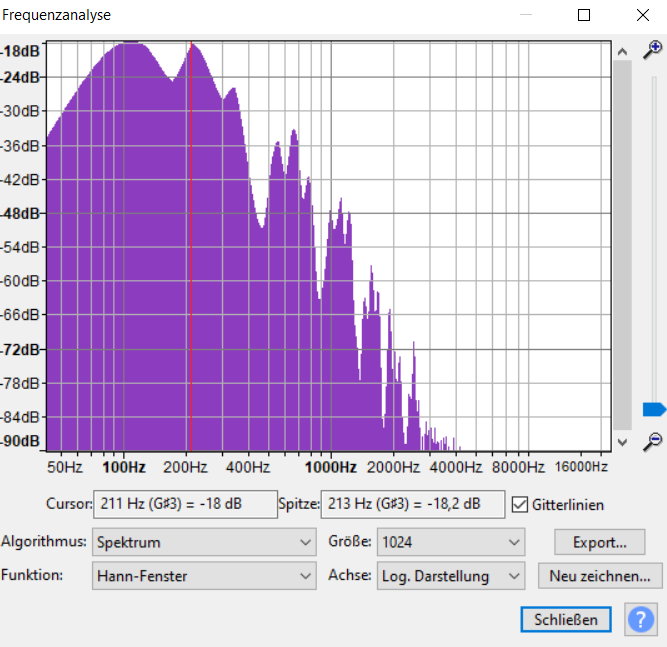
41-th magnitude: 3.630623 => corresponds to 220.715332 Hz

keyNR: 37

Die Richtigkeit des Outputs kann mit Audacity überprüft werden. Dafür muss in der Menüleiste unter Analyse, Spektrum zeichnen gewählt werden.



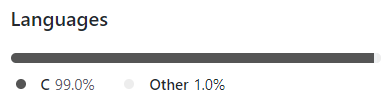
Im dadurch geöffneten Fenster kann nun das Frequenzspektrum untersucht werden.



Hierbei fällt auf, dass neben unserer Grundfrequenz von 107 Hz noch eine zweite sehr prominent heraussticht. Dabei handelt es um eine Harmonische, also einen Oberton, dessen Frequenz ein Vielfaches der Grundfrequenz ist. Diese haben nicht immer eine niedrigere Amplitude als der Grundton, weshalb sie bei der Notenerkennung eine besondere Herausforderung sind.

Um etwaige Erkennungsprobleme, welche durch die Harmonischen verursacht werden können zu umgehen, wird im Hauptprogramm die tiefste Frequenzspitze als Grundton angenommen. Stattdessen gibt es auch andere, zuverlässigere Möglichkeiten wie etwa Pitch-Detection mit Autocorrelation. Diese sind allerdings auch erheblich schwieriger zu implementieren, weshalb wir uns für die Umsetzung mithilfe des FFT-Algorithmus entschieden haben.

## C-Programmiersprache

Die Funktionen in der Firmware des Projektes wird beinahe ausschließlich mit der Programmiersprache C realisiert. C ist eine weitverbreitete imperative, prozedurale Programmiersprache, welche bekannt und beliebt für die Programmierung in Hardwarenähe ist. Das im Folgenden beschriebene ESP IoT Development Framework basiert auf der Programmiersprache C.

### Agile Softwareentwicklung mit C

In C lassen sich die Methoden der agilen Softwareentwicklung anwenden, auf welche beim Firmware-Development geachtet wurden. Drei bekannte Denkweisen im Extreme-Programming sind die KISS-, DRY- und YAGNI-Prinzipien.

KISS was so viel heißt wie „Keep it Simple, Stupid! “, ist eine im Extreme-Programming angewandte Methode, welche besagt, dass die kürzeste Lösung oft die Einfachste ist. Es sollen daher keine überflüssigen Methoden implementiert werden und Klammer Neste so seicht wie möglich gehalten werden.

DRY steht für „Don’t Repeat Yourself “. Mehrfach vorkommender Code soll in Funktionen gefasst werden, sodass bei Änderungen nur an einer Stelle gearbeitet werden muss. Daher soll eine Methode nur eine Funktion haben.

YAGNI ist die Abbreviatur der Phrase “You Ain’t Gonna Need It”. Aus dem Englischen übersetzt, bedeutet sie so viel wie „Du wirst es nicht brauchen“. Oftmals hat man während der Programmierung neue Einfälle, welche über die bisherigen Planungen hinausragen und erst viel später, wenn überhaupt, benötigt werden. Anstatt die Zeit in Arbeit zu investieren, welche möglicherweise überflüssig ist, sollte man sich auf die wesentlichen Problemstellungen fokussieren.

## ESP32

Ein Bild, das Text, Elektronik, Schaltkreis enthält.

Automatisch generierte BeschreibungDer ESP32 ist ein Leistungsstarker Dual-Core 32-bit RISC-V Prozessor des Herstellers Espressif, welcher oft aufgrund seiner Vielzahl an Peripherien, Protokollen und Sensorschnittstellen, im Internet of Things (IoT) Verwendung findet. Integrierte Schnittstellen sind beispielsweise UART, SPI, CAN, I2C, I2S, WLAN und Bluetooth.

Abbildung : ESP32 Prozessor (Mitte), PIF WLAN-antenne (links)

Für Prototypen sind ESP32-WROOM Module, halbfertige PCB-Module mit gekerbten Löchern als Pins, üblich. Module unterscheiden sich grundlegend im Footprint, welcher durch die Art der Antenne, PIF-Antenne (Planar Inverted F-Shaped Antenna) oder IPX Connector (Koaxial Steckverbinder) bestimmt ist. Eine vom Footprint unabhängige Kenngröße, ist die Größe des Flash-Memory.

Abbildung : ESP-WROOM Modul mit IPX Connector

### Stromversorgungs-Kriterien

### Bootloader Brennen mit dem ESP-Tool

### Einrichten in PlatformIO (PIO)

### ESP IoT Development Framework

### Direct Memory Access (DMA) Controller

## Realtime Operating-System (RTOS)

## Serial Peripheral Interface (SPI)

## Inter-IC-Sound (I2S)

## Musical Instrument Digital Interface (MIDI)

### Kommandos

## Universal Asynchronous Receive and Transmit (UART)

# Ergebnisse

Mit dem Wissen der Grundlagen, lassen sich nun die Entwicklungsergebnisse im Detail erklären. Hierbei wird darauf geachtet, dass jede Funktion hinterfragt und die Wahl der Komponenten genau begründet ist.

## Blockschaltbild

Mit dem folgenden Funktionsblockdiagramm wird die Prozesskette des Guitar Converter beschrieben.

Gestrichelte Linien visualisieren die drei primären Sektionen, in die die Prozesskette unterteilt ist:

* Analoges Frontend
* Digitalisierung des Signals
* Firmware, die für die DSV zuständig ist und die Daten auf den USB schickt.

Dicke Gelbe Pfeile stellen den Übergang einer Sektion in eine andere dar.

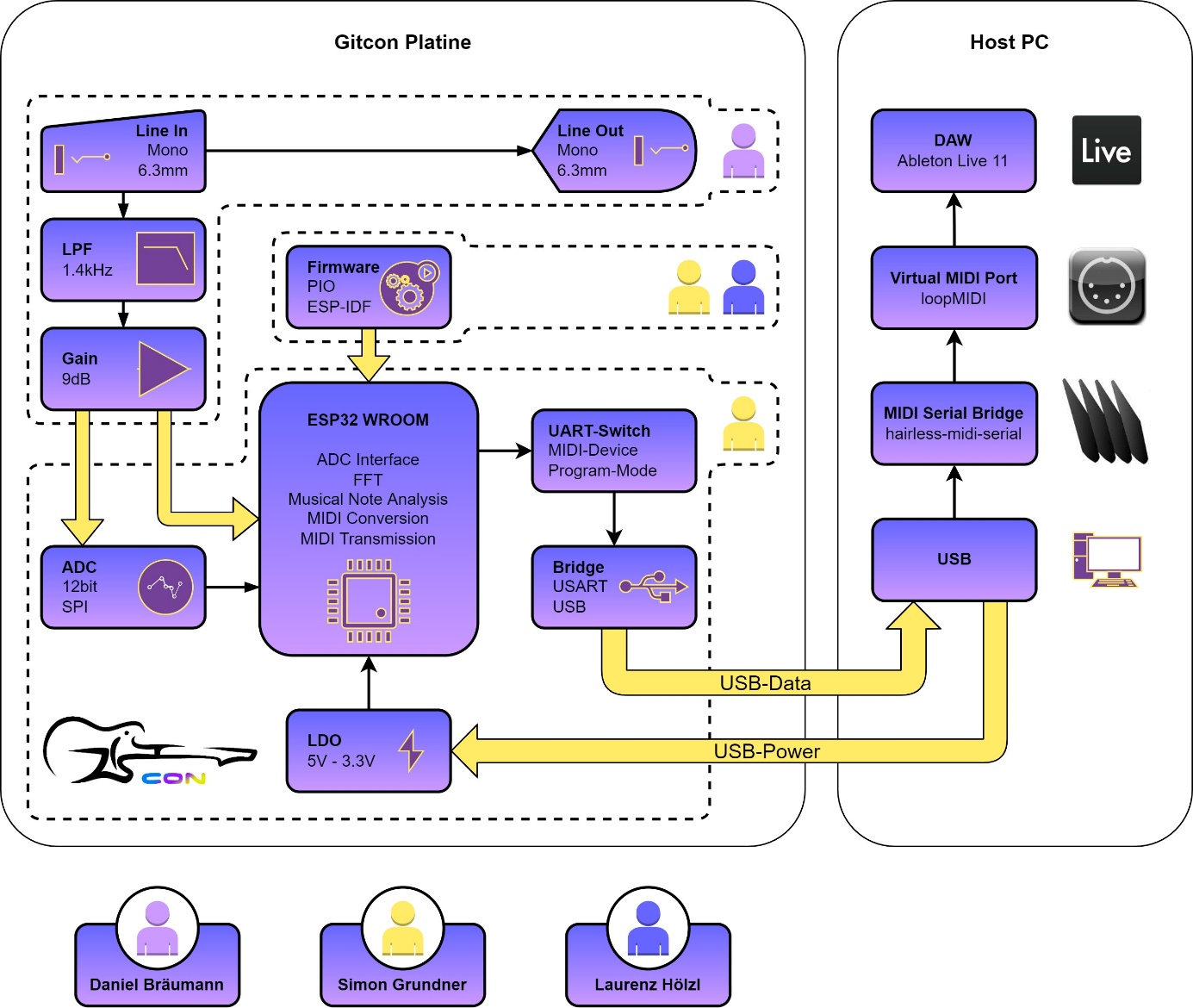
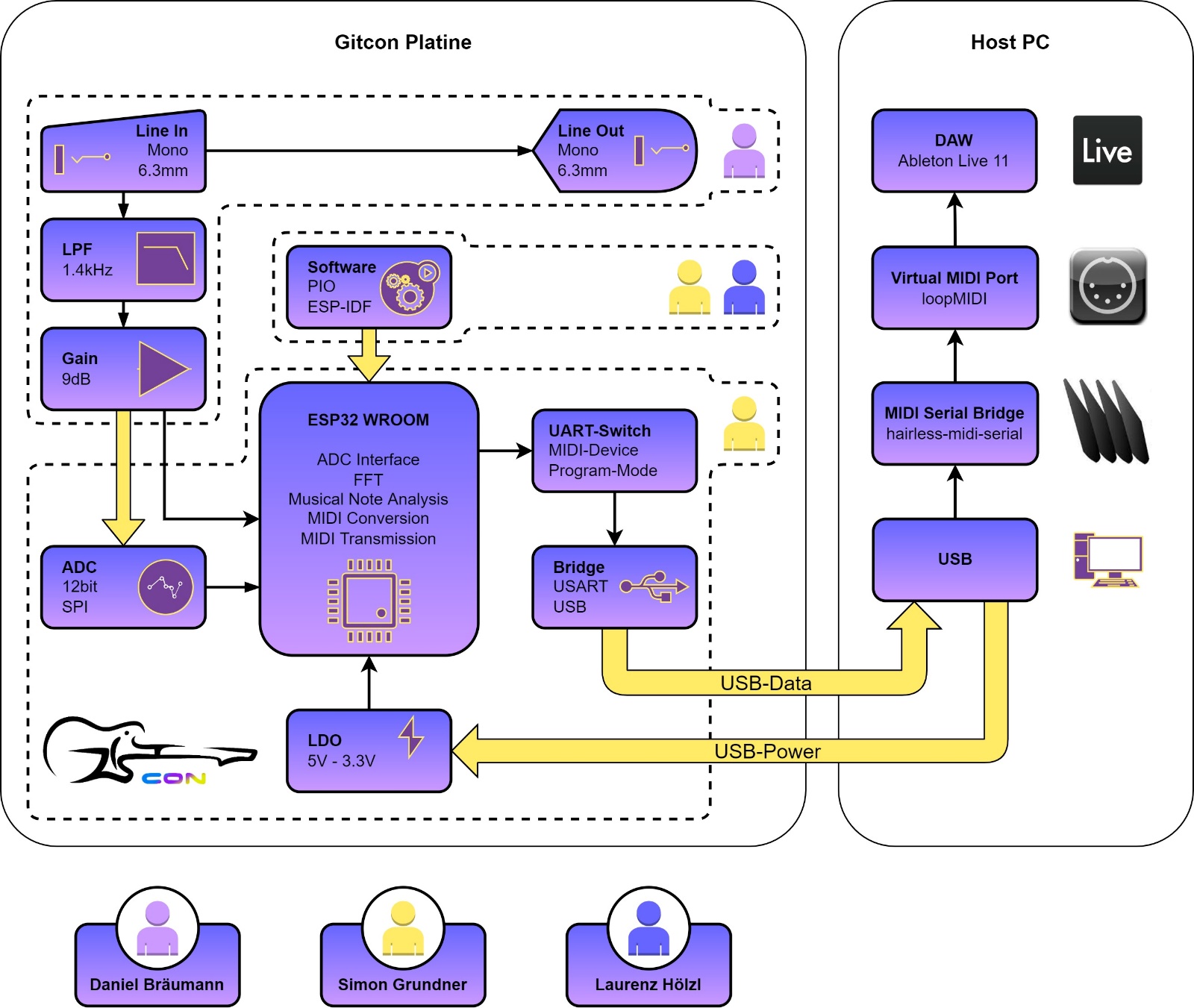


Abbildung : Funktionsblockdiagramm des Gitcon

Mit den Figuren innerhalb der strichlierten Linien, wird noch einmal zusätzlich die Arbeitsunterteilung veranschaulicht. Mit der Farbe der jeweiligen Figur, wird auf ein Teammitglied referiert.



## Hardware

Im folgenden Teil wird zunächst der Schaltplan des funktionalen Blockschaltbildes (Abbildung 6) dokumentiert, sowie die Wahl der Komponenten begründet. Darauffolgend werden die Layout Kriterien und Guidelines für das Leiterplatten-Design beschreiben.

### Versorgung

Die Komponenten, die zur Realisierung des Blockdiagrammes gewählt wurden, müssen mit 3,3 Volt versorgt werden. Da uns der USB-Bus jedoch einen Pegel von 5 Volt zur Verfügung stellt, muss dieser zuerst heruntergeregelt werden. Hierzu wird ein Low Drop-Out Regulator verwendet.

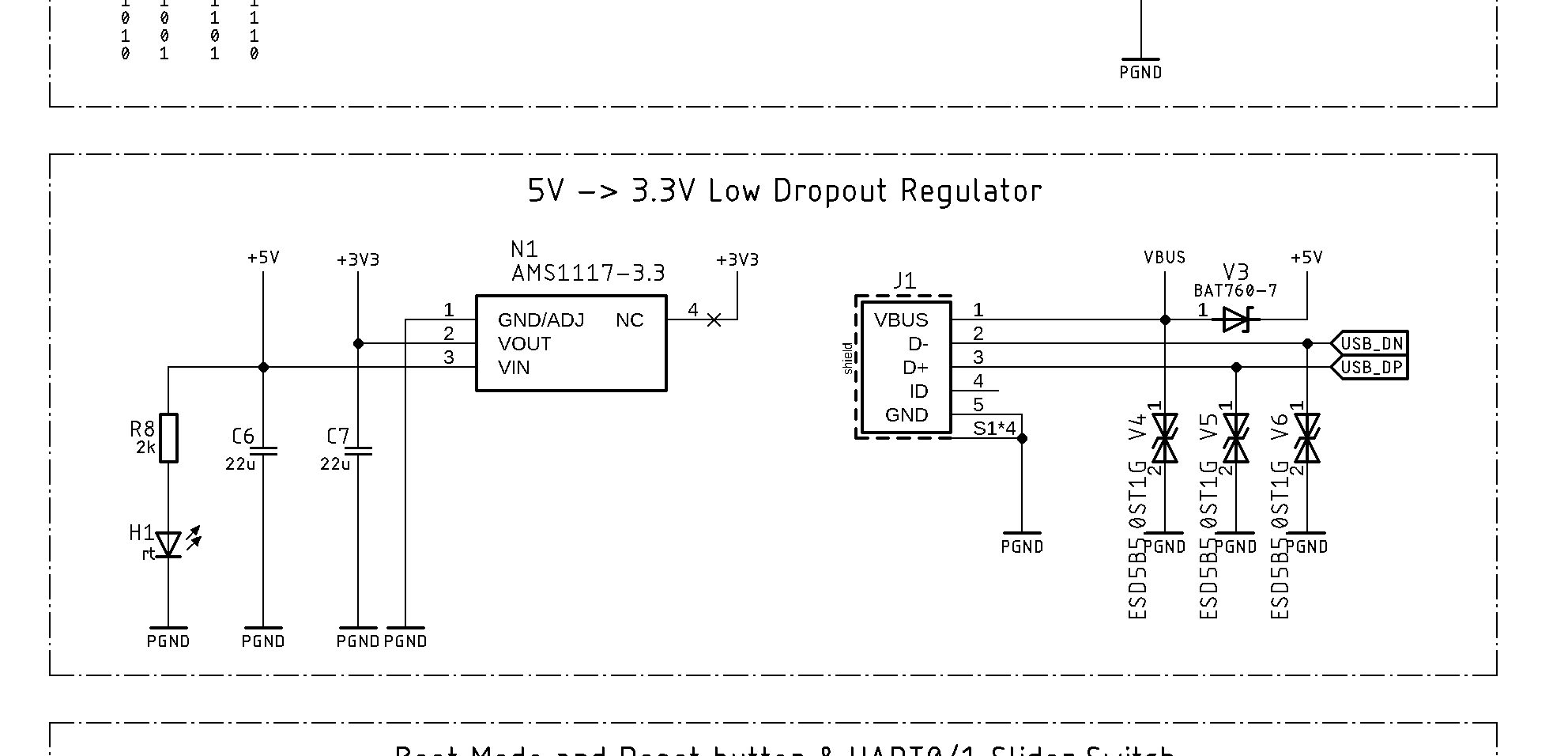


Abbildung : Spannungsregler und USB-Connector im Schaltplan

#### LDO (AMS1117-3.3)

Ein Low-Dropout Regulator ist eine Art linearer Spannungsregler mit einem möglichst geringen Spannungsabfall über dem Ein- und Ausgangskontakt. LDOs werden verwendet, um das Rauschen in Versorgungsleitungen zu eliminieren aber auch, um eine (etwas) höhere Spannung in eine niedrigere Spannung zu regeln. In diesem Fall werden die 5 Volt der USB-Schnittstelle, in die von den Peripherien am Board benötigten, 3,3 Volt heruntergeregelt.

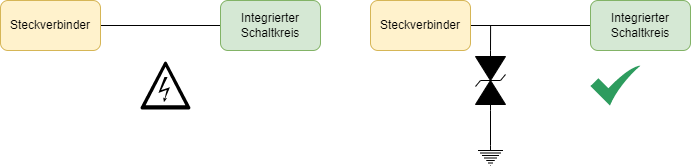
Gegenüber Schaltreglern haben sie den Vorteil, dass sie kompakt sind und kein Schaltrauschen verursachen. Auch der Wirkungsgrad ist aufgrund der niedrigen Dropout-Spannung sehr hoch. Mit einem maximalen Ausgangsstrom von 1 Ampere reicht er vollkommen aus, den Gitcon mit Strom zu versorgen. Jedoch ist zu beachten, dass, um die Stabilität zu gewährleisten, ein 22µF (Tantal-Elektrolyt-) Kondensator am Ausgang angebracht werden sollte. C6 und C7 halten hier die Ein- und Ausgangsspannung bei Spannungseinbrüchen möglichst konstant. Außerdem leuchtet eine rote LED leuchtet auf, wenn das Gerät eingesteckt ist.

#### ESD-Schutz

ESD ist die Abkürzung für electrostatic discharge und beschreibt die plötzliche Entladung eines Objekts auf ein anderes, wenn sie miteinander in Kontakt treten. Der Potentialunterschied zwischen den Objekten, kann je nach Material mehrerer **Kilovolt** erreichen.

Integrierte Schaltkreise, wie sie in diesem Projekt enthalten sind, sind typischerweise sehr empfindlich gegenüber statischer Entladung. Aufgrund des vielen Interagieren mit der Platine, zum Beispiel beim Ein- und Ausstecken, muss mittels TVS[[2]](#footnote-3)-Dioden V4-6 eine Abschirmung gegen statische Entladung am USB-Eingang vorhanden sein.

Abbildung : Warnung zu ESD-Empfindlichen Objekten



##### *Kennwerte der TVS-Diode*

Ein Bild, das Pfeil enthält.

Automatisch generierte BeschreibungUni- oder Bidirektional

Abbildung : Symbol der Bipolaren TVS-Diode

Je nachdem ob die zu schützende Leitung über oder unter dem Massepotential liegt, muss man bei unidirektionalen TVS-Dioden die Polung beachten. Bidirektionale arbeiten in beide Richtungen.

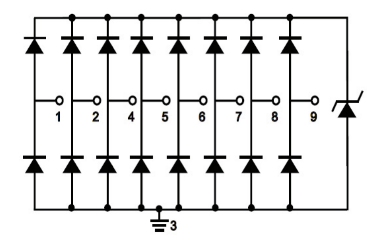
Anzahl von Kanälen

Abbildung : SC7538-08UTG Block Diagramm

Oft haben Konnektoren eine Vielzahl an Pole (z.B. HDMI) die geschützt werden müssen. Deshalb gibt es mehrere TVS-Dioden in einem einzigen Package.

Arbeitsspannung

Die Arbeitsspannung ist die maximale Spannung, welche im normalbetrieb auf der Leitung vorkommt. Hier 3,3 Volt.

Klemmspannung

Im Falle eines ESD-Spikes, wirkt die TVS-Diode niederohmig gegen Masse. Die Spannung, die dabei an der Diode abfällt, ist die Klemmspannung. Obwohl sie immer noch signifikant höher ist als die Betriebsspannung, ist sie aufgrund der kurzen Dauer nicht weiter gefährlich, trotzdem sollte sie in jedem Anwendungsfall möglichst gering gewählt werden. Einen genauen Wert dafür findet man nur sehr schwer. Man sollte im Datenblatt der TVS-Diode auf Verweise für typische Applikationen achten.

Intrinsische Kapazität

Wie bei jedem Bauteil, gibt es gewisse parasitäre Kenngrößen, die im inneren präsent sind. Bei der TVS-Diode ist die Kapazität eine störende, dennoch unvermeidbare, Größe. Bei TVS-Dioden mit hoher Kapazität besteht die Gefahr, dass sehr kurze ESD-Stöße nicht gefiltert werden können. Abgesehen von der intrinsischen Kapazität, ist zu berücksichtigen, dass auch die Leiterbahn von der Buchse zur Diode keine eigene signifikante Kapazität aufweisen darf. Dies wird durch eine möglichst kurze Leiterbahnführung sowie eine dünne Leiterbahnbreite realisiert.

IEC61000−4−2 Rating

Die TVS-Diode verfügt über eine Rating Stufe der IEC61000-4-2 Norm, welche Aussagen über die absoluten Höchstwerte des ESD-Spikes liefert.

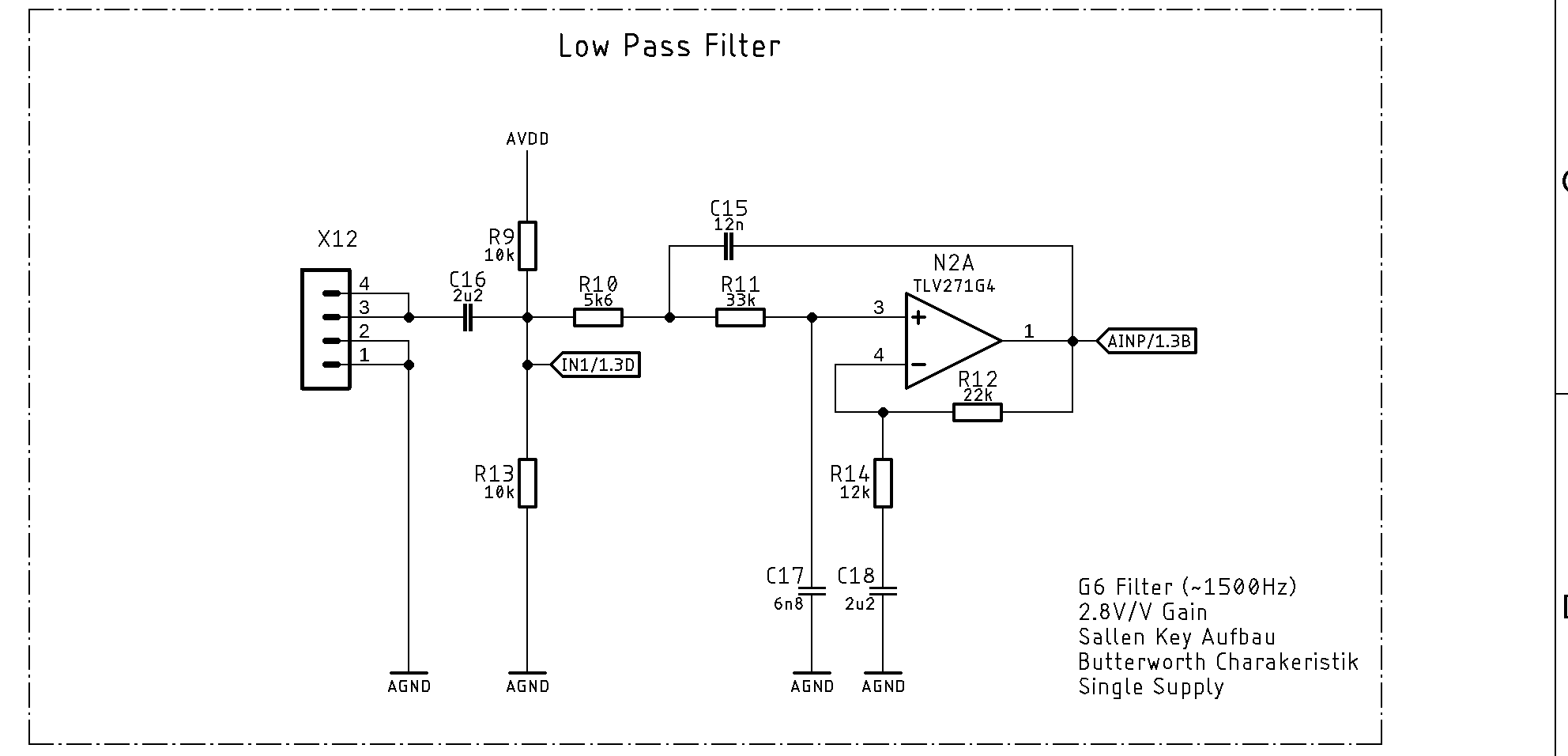
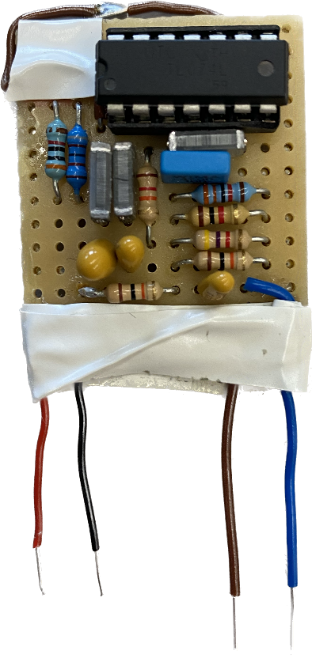
#### Verpolungsschutz

Obwohl es nur schwer möglich ist, einen USB-Stecker falschherum einzustecken, verfügt die Schaltung trotzdem über einen Verpolungsschutz. Sobald eine negative Spannung am Eingang anliegt, wirkt die Schottky-Diode V3 entgegen dem Stromfluss. Der Vorwärtsspannungsabfall an der Diode ist hierbei nicht problematisch, da die Spannung nach der Diode auf 3.3V heruntergeregelt wird.

### Analog-Frontend (AFE)

Warum wir einen Filter benötigen, lässt sich aus den Eigenschaften des Gitarren Pickups sowie das Schwingverhalten einer Saite erschließen.

#### Schaltung



##### Einkopplung des Audios

Da der Gitcon über keine negative Versorgung verfügt, muss das Audio zwischen Masse und +3.3 Volt eingekoppelt werden. Dies erfolgt über … (Daniel)

#### Filterdimensionierung

#### Simulation

#### Filterprototyp

### Digital Frontend

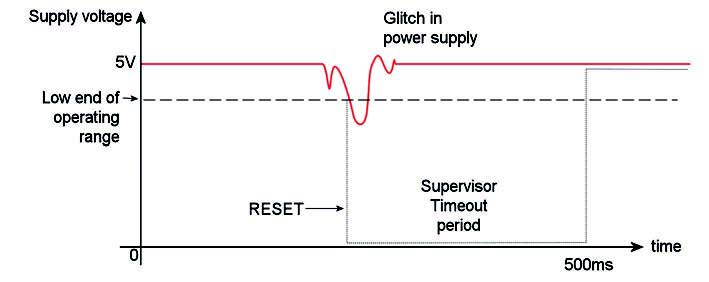
Die nötigen Peripherien des Gitcon werden mit einem ESP-WROOM-32UE-N8 Modul gesteuert. Die Namensgebung des Moduls liefert Informationen über die Serie des Chips:

|  |  |
| --- | --- |
| ESP-WROOM-32 | 32-Bit WROOM Modul |
| U | IPX Antennen Connector für externe Antennen |
| E | E-Serie |
| N8 | 8MB Flash Speicher |
| - | Kein PSRAM |

#### Beschaltung des ESP-WROOM-32UE Chips



Für den Betrieb des ESP32-WROOM Moduls, werden grundsätzlich nur zwei Entkopplungskondensatoren C1 und C2 benötigt. Sie müssen möglichst nah am Versorgungs-Kontakt platziert werden, wobei der HF-Entkopplungs-Kondensator C2 näher als der Bulk-Kondensator C1 platziert werden sollte. Dies versichert, dass der ESP32 stabil versorgt wird, da der Watchdog automatisch einen Reset bei Spannungseinbrüchen auslöst.



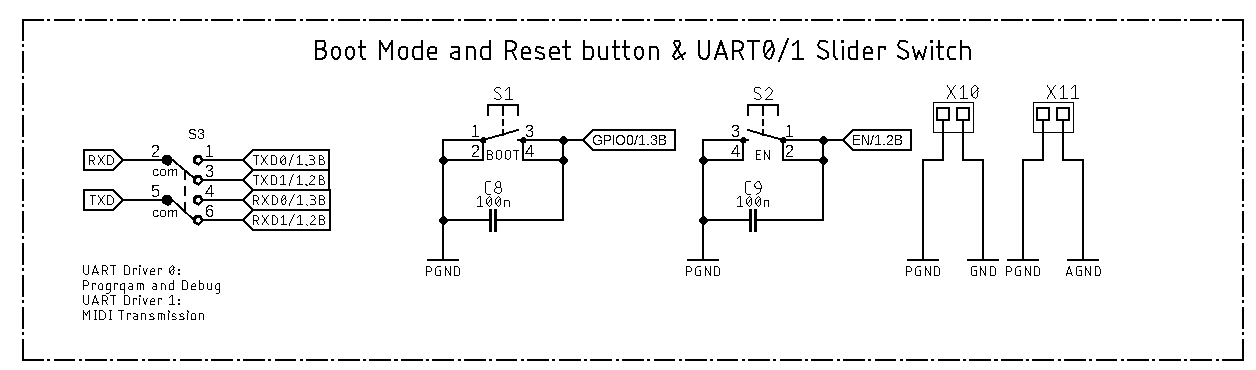
3.3V

Abbildung : Reset bei Spannungseinbruch

Für den manuellen Reset Eingang wird ein Pull-Up-Widerstand (R1) in Kombination mit einem Entstör-Kondensator (C3) benötigt, welcher ungewollte Resets-Verhindert und zusätzlich die natürliche Prellung des Tasters unterdrückt. Die beiden Komponenten sollten möglichst nah am Reset-Pin platziert werden.

##### Pin-Out

|  |  |  |
| --- | --- | --- |
| ESP32 GPIO | Label | Funktion |
| GND\_xx | GND | Digitale Masse Pins,  Heat-Sink vom Modul |
| 3V3 | +3V3 | Versorgungsspannung |
| EN | EN | Enable /Pin |
| IO26 | TXD1 | Transmit-Leitung des UART1 Treibers (MIDI TX) |
| IO27 | RXD1 | Receive-Leitung des UART1 Treibers (MIDI RX) |
| IO0 | GPIO0 | Boot-Mode Pin |
| IO5 | VSPI\_ | Chipselect für den ADC |
| IO18 | VSPI\_CLK | Taktleitung des VSPI-Busses |
| IO19 | VSPIQ\_MISO | Master In Slave Out des VSPI-Busses |
| RXD0 | RXD0 | Receive-Leitung des UART0 Treibers (Debug/Programmierung) |
| TXD0 | TXD0 | Receive-Leitung des UART0 Treibers  (Debug/Programmierung) |



#### Beschaltung der USB-Bridge

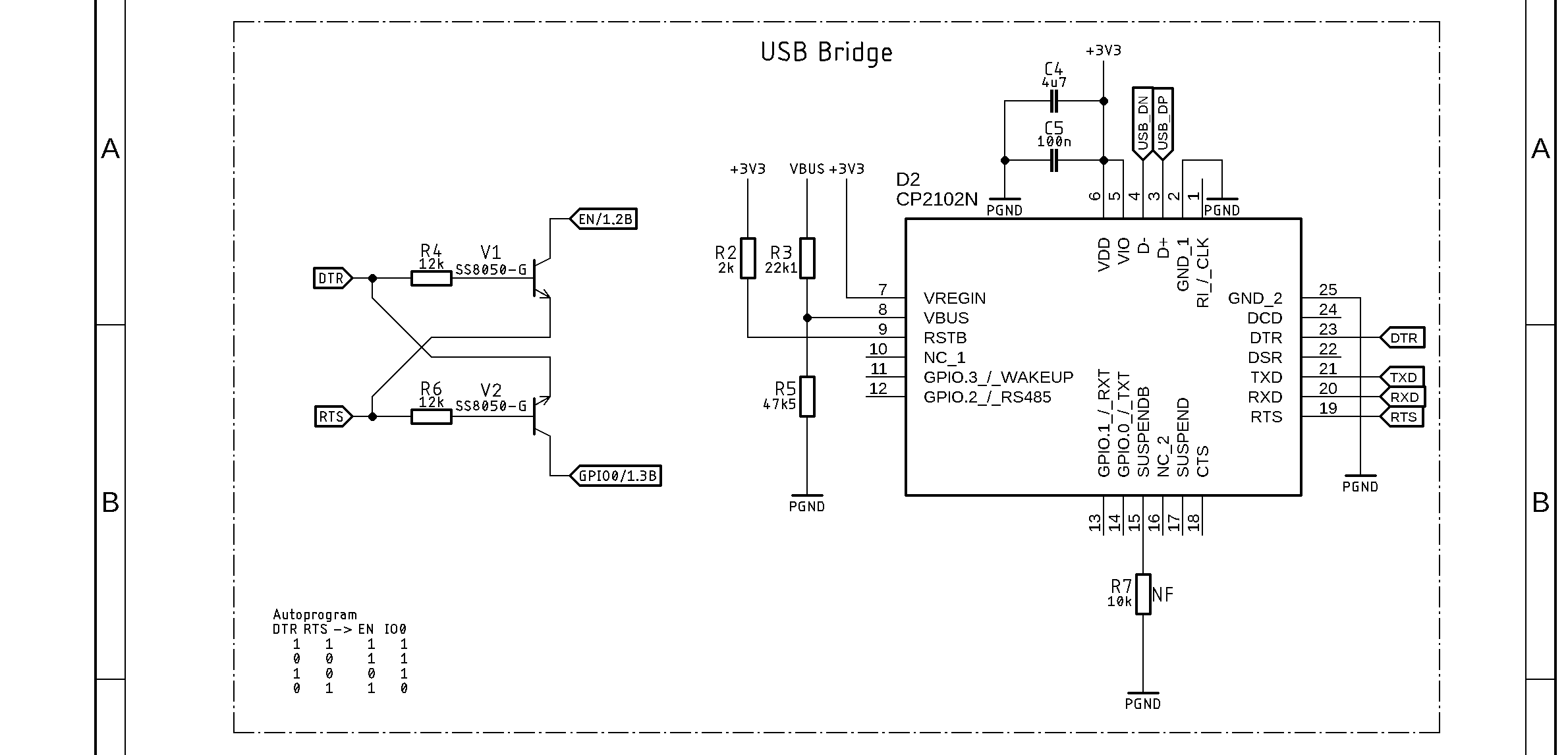


Abbildung : Beschaltung der USB-Bridge

Die Beschaltung der USB-Bridge wurde nach den Richtlinien im Datenblatt implementiert. Neben den Pull-Widerständen R2 und R7, wird sie mit einem Spannungsteiler, bestehend aus R3 und R5, beschalten. Dieser dient zur Konfiguration eines integrierten Linearreglers, welcher aber im Bypass Modus ist. Das heißt, dass er nicht zur Versorgung externer Komponenten verwendet wird, da sein Versorgungsstrom maximal 100mA ist. Abzüglich des Stromes, den die Bridge selbst benötigt, ist der Ausgangsstrom des internen LVR zu niedrig, um als Hauptversorgung verwendet zu werden. Die besagte in Konfiguration wird im Datenblatt als „USB Self-Powered“-Konfiguration bezeichnet.

#### Interner ADC

#### MCP3202 ADC

### Layout der Platine

Eine Zielsetzung bei dem Projekt ist es, sich von den Entwicklungsmodulen zu distanzieren und ein Gerät zu entwickeln, welches einem vermarktbaren Produkt ähnelt. Um dieses Ziel zu erfüllen, ist ein geeignetes Leiterplattendesign ausschlaggebend für die Funktionalität und einen kompakten Footprint des Gerätes. Bei dem Design der Leiterplatte wurden besonders auf die Elektro-magnetische Verträglichkeit geachtet.

#### Versorgungsleitungen

Die Versorgungsleiterbahnen sind leicht von den anderen Leiterbahnen zu unterscheiden, da sie im Vergleich viel dicker sind. Da über Versorgungsleitungen der meiste Strom fließt, ist es von Vorteil diese so breit wie möglich zu machen, um Verluste zu minimieren. Mit einer Breite von 32mil kann eine solche Leiterbahn bis zu 3 Ampere (mit einem Temperaturanstieg um ca. 30°C) standhalten.

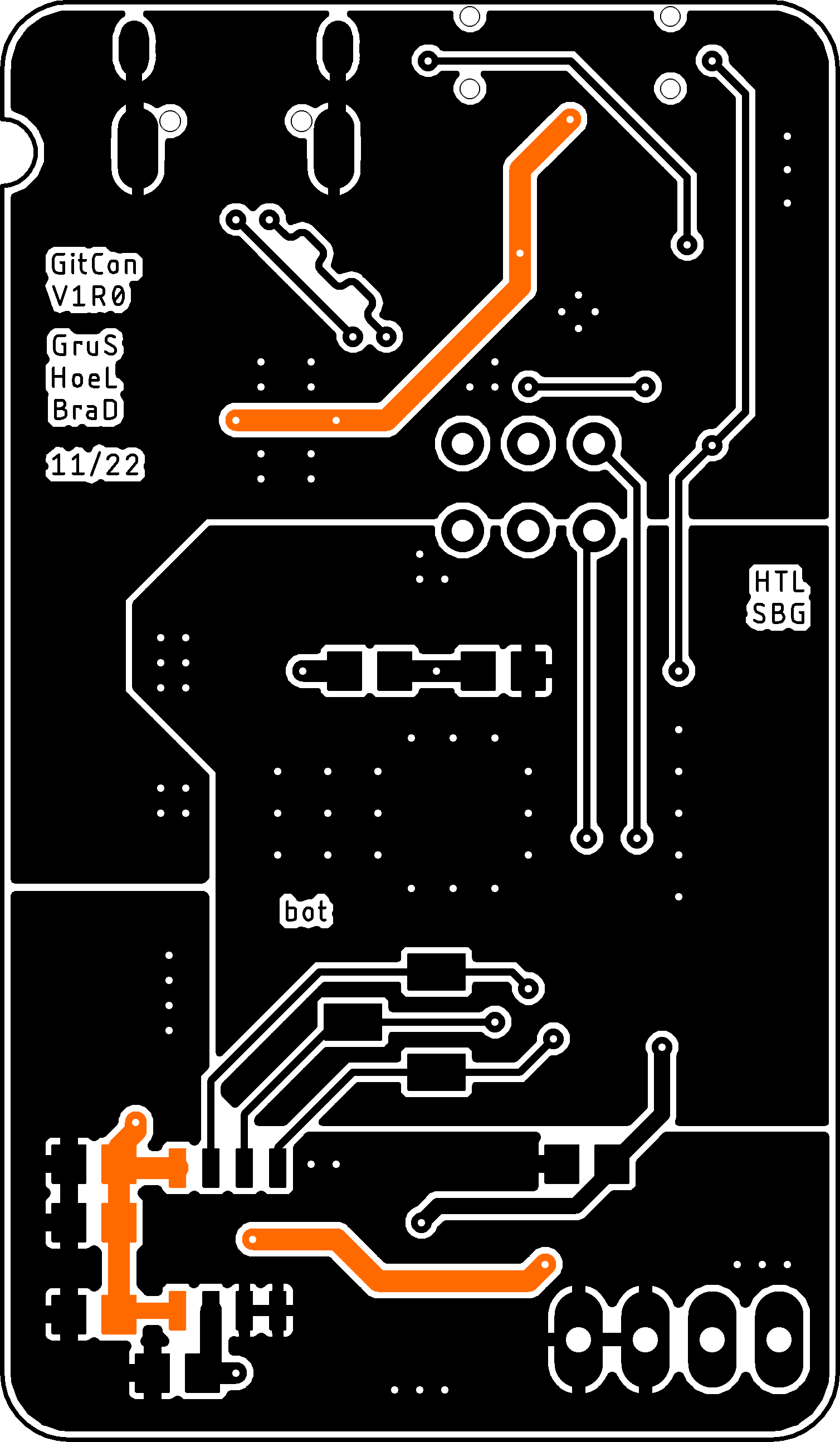
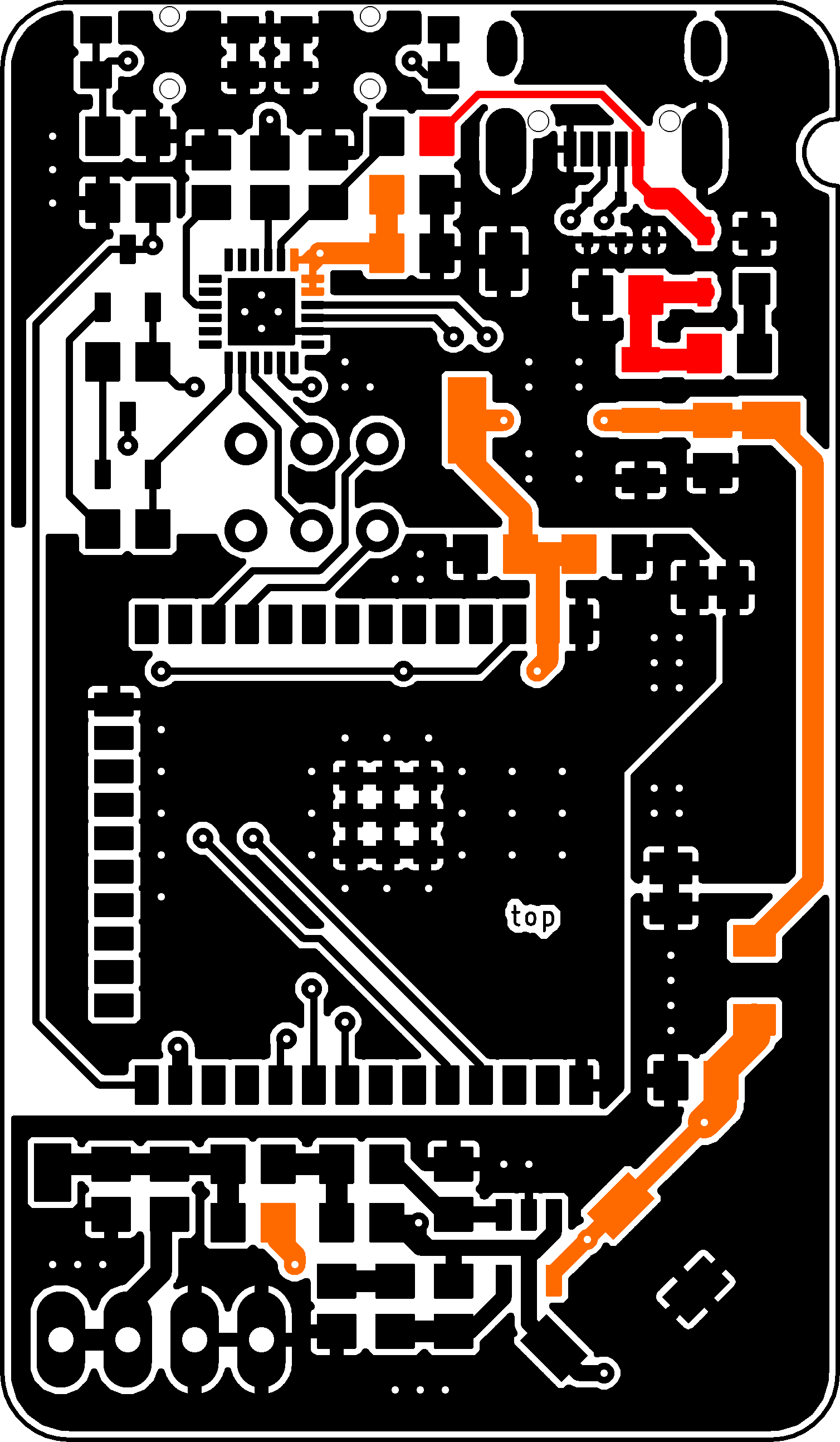


Abbildung :Versorgungsleitungen

Die Versorgung der USB-Bridge, des ESP32 und der Analog-Sektion sind Sternpunkt-förmig am LDO angeschlossen. Eine Besonderheit bei der Analog-Versorgung ist, dass sie zusätzlich mit einem passivem C-L-C Pi-Filter ausgestattet ist, um Mittel- bis Hochfrequentes Rauschen zu entfernen.

Abbildung : Komponenten des Pi-Filters



#### Massefläche

Die Masseanschlüsse der einzelnen Komponenten werden mit einer Massefläche verbunden. Die daraus resultierende induktionsarme Verbindung sorgt dafür, dass Stromrückflüsse einen nur sehr geringen Potentialunterschied verursachen.

##### Sternpunktmasse

Treffen sich die einzelnen Masseflächen in nur einem Punkt, so spricht man von einer Sternpunktmasse. Eine Sternpunktmasse hat den Vorteil, dass Masseschleifen verhindert werden.

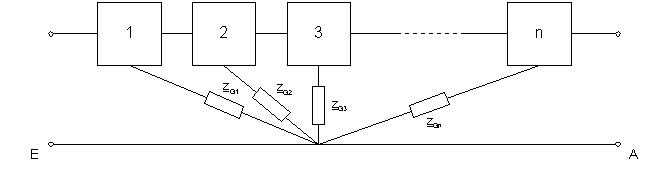


Abbildung : Visualisierung einer Sternpunktmasse

Eine Masseschleife entsteht, wenn sich mehrere Bauteile eine Impedanz (eine Leiterbahnenabschnitt) als Masse teilen. Durch den Stromrückfluss eines Bauteils auf diesem Abschnitt, kann das Massepotential an einem anderen Bauteil angehoben werden. Man würde dann von einer galvanischen Kopplung sprechen.

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungIn diesem Fall werden die Masse-Polygone möglichst nahe am LDO mit SMD-Überbrückungen verbunden. (siehe pink-markierte Felder in Abbildung 13)

Abbildung : SMD-Jumper

Auf der Platine werden die Komponenten auf drei verschiedene Arten klassifiziert: Digitale-, Analoge- und Leistungsbauteile.

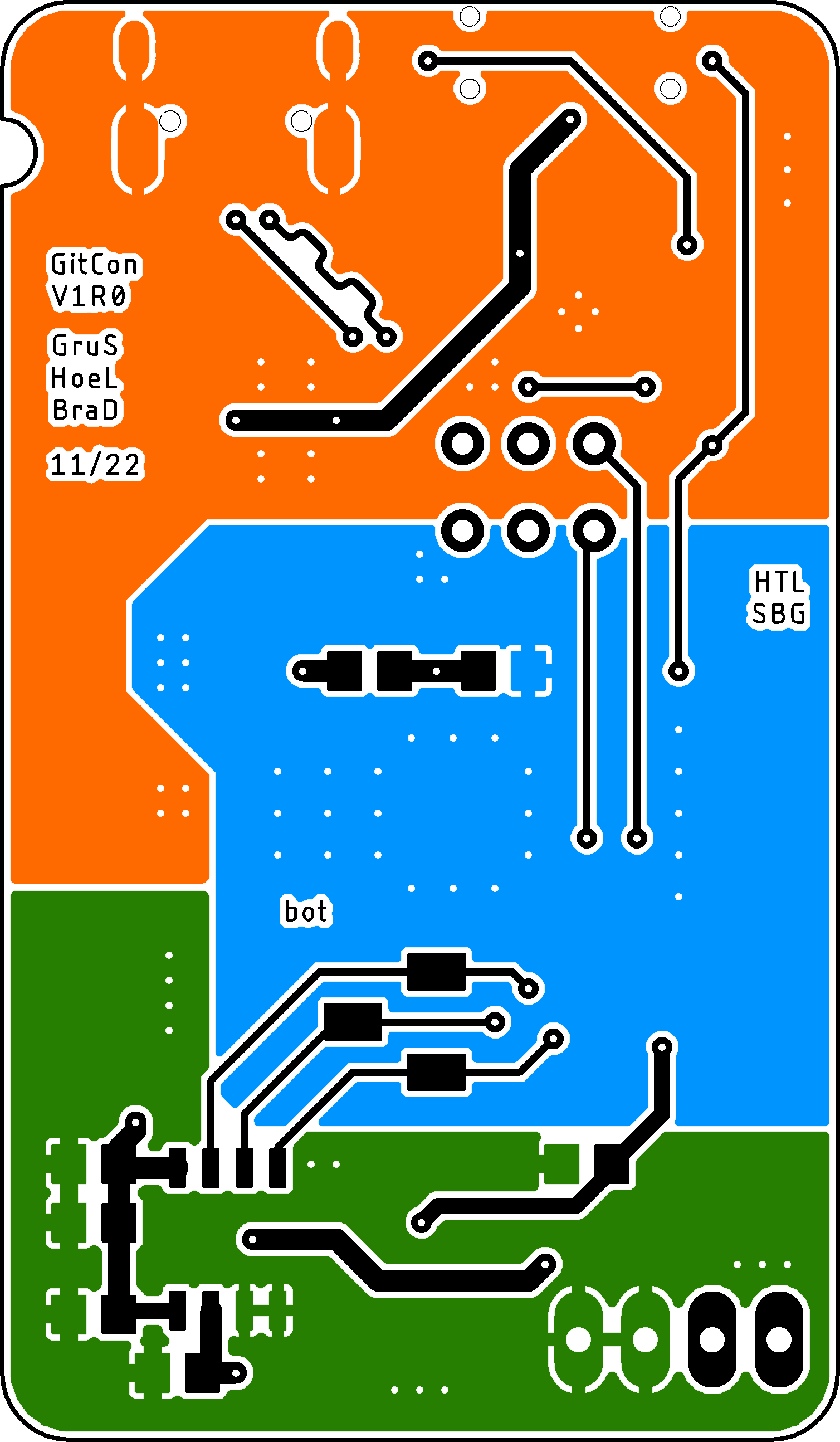
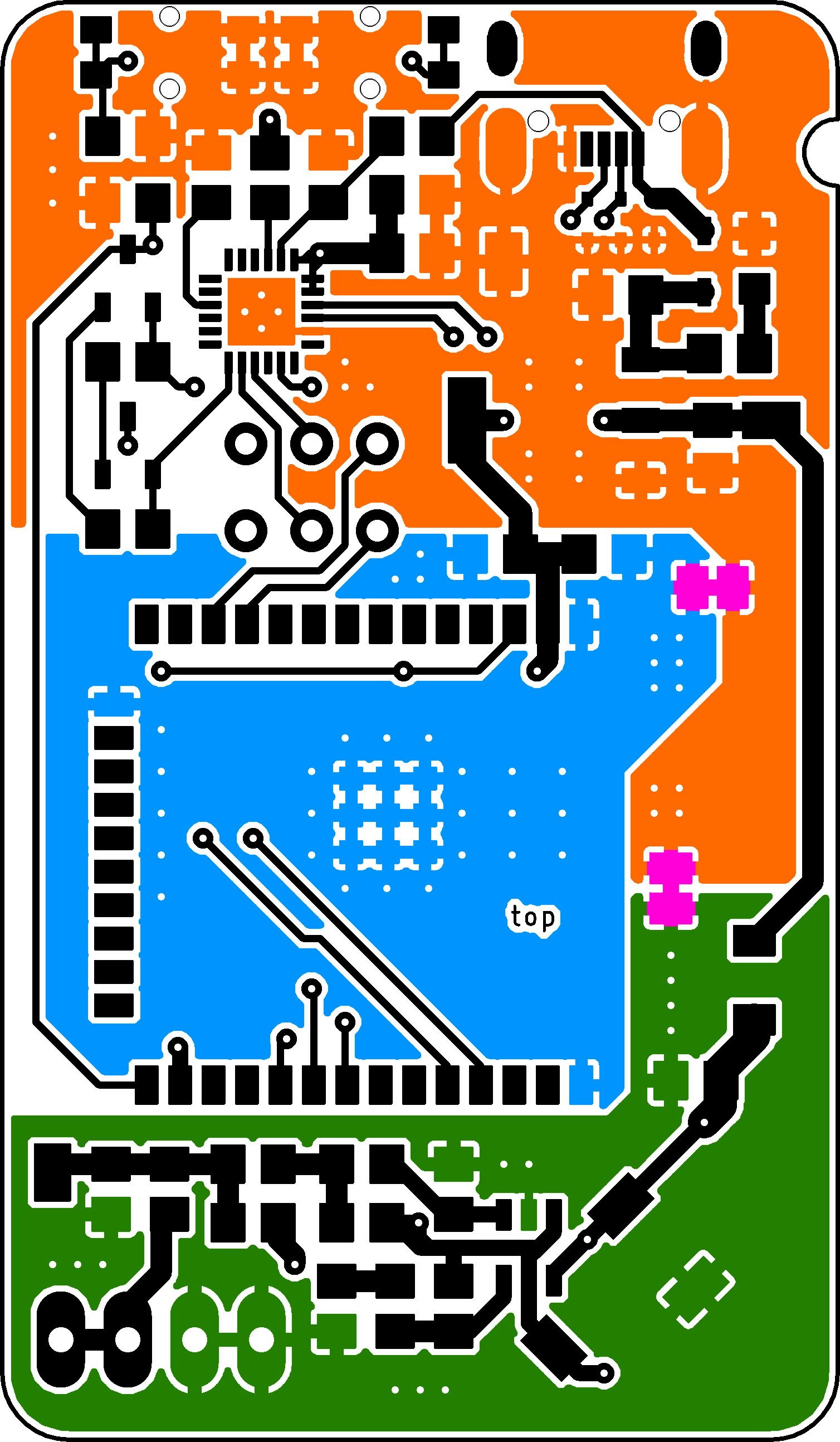
 

Abbildung : Bottom (links) und Top (rechts) Masse-Layer

PGND: Leistungsmasse (Orange)

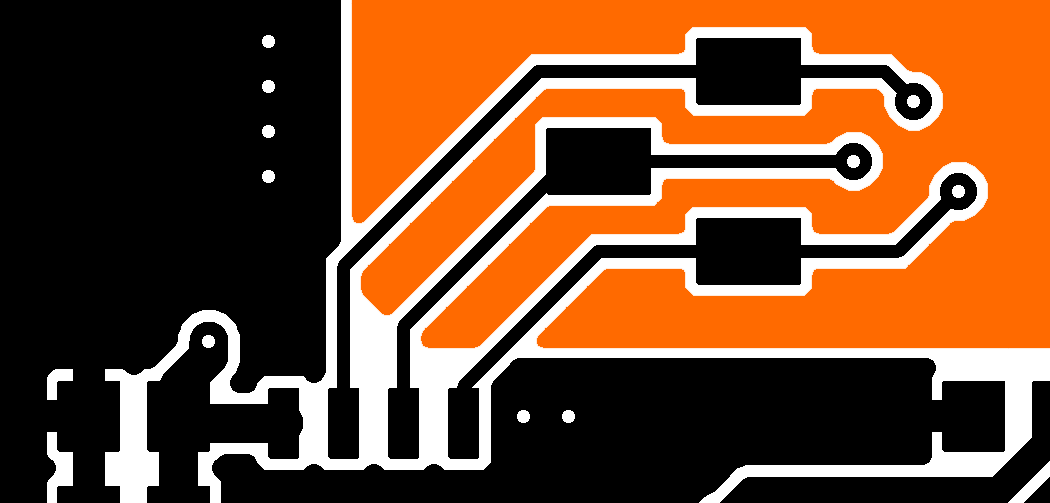
Leistungsbauteile trieben oft hohe Ströme, welche ungewollte Differenzen auf dem Massepotential hervorrufen können. Isoliert man nun diese Masse von den Signaltreibenden Bauteilen, ist nur das Leitungsbauteil von der galvanischen Kopplung betroffen.

GND: Digitalmasse (Blau)

Ein Bild, das Text, Uhr, Screenshot enthält.

Automatisch generierte BeschreibungDigitale Bauteile Arbeiten oft mit hochfrequenten Signalen (hier z.B. der SPI-Bus). Wenn HF-Leitungen frei liegen, können sie auf nahen Leiterbahnen, durch das sich schnell ändernde elektrische Feld Störungen verursachen. Diesen Effekt bezeichnet man als kapazitive Kopplung.

Um dieser Kopplung zwischen den Leiterbahnen entgegenzuwirken, muss zwischen besagten Leitungen eine Massefläche ausgeprägt sein.



MISO

CS

CLK

Da diese Störungen nun von der Masse absorbiert werden, muss die Fläche vom Rest isoliert werden. Dies wird erzielt, indem man sie nahe an der Masse der Versorgungseinheit (LDO) anschließt.

AGND: Analogmasse (Grün)

Bei analogen Signalen ist oft das Rauschen ein kritischer Faktor (SNR). Beispielsweise haben die Audiosignale, welche von den Pick-Ups der E-Gitarre induziert werden, eine sehr geringe Amplitude. Benötigt nun eine andere Komponente für kurze Zeit einen hohen Strom oder generiert durch Hohe Frequenzen ein Rauschen, so können Störungen auf der Masse auftreten, welches die Referenz für das Audiosignal verändert. Ein ADC kann dann das Signal nicht mehr korrekt erfassen und das Sample ist fehlerhaft. Eine Störungsbefreite Masse unterstützt somit die Integrität des Signales und es kann in möglichst originaler Form weiterverarbeitet werden.

##### Via-Stitching am Masse-Polygon

Um den Stromrückfluss-Weg von Komponenten so kurz wie möglich zu halten, wurden die Masseflächen an mehreren Stellen mit einer Vielzahl an Durchkontaktierungen (Vias) „vernäht“. Das Massepotential wird dadurch über die gesamte Fläche konstant gehalten.

Abbildung : Ground-Stitches auf der Platine



#### Platzierung der Entkopplungskondensatoren

Entkopplungskondensatoren haben grundsätzlich zwei Aufgaben. Zum einen sollen sie bei Spannungseinbrüchen die Versorgung möglichst aufrechterhalten, außerdem sollen Hochfrequente Störsignale gefiltert werden.

Ein einziger Kondensator kann beide Aufgaben nicht ideal erfüllen, da mit niedrigeren Kapazitäten zwar die HF-Entkopplung besser ist, jedoch die Bulk-Kapazität fehlt, um einen Einbruch der Spannung zu Kompensieren und umgekehrt. Deshalb ist es üblich, zwei Kondensatoren so nahe wie möglich an der Versorgung eines Integrierten Schaltkreises (IC) auf der Platine zu montieren.

#### Leitungslängenanpassung beim Differenziellen Paar des USB

#### Hochfrequenz Leitungen

#### Audio

Niederfrequente Audio-Signale, müssen von allen Stör- und Rauschquellen isoliert werden, sodass das Audio möglichst original aufgefasst werden kann. Auf der Gegenüberliegenden Seite der Leiterplatte sind daher **keine** Leiterbahnen verlegt, sondern nur eine Massefläche.

Abbildung : Masse unter Audio-Sektion

#### Abmessung und Kompakter Footprint

Ein Bild, das LEGO, Spielzeug enthält.

Automatisch generierte BeschreibungDie Platine verfügt über eine Einkerbung, mit welcher sich die Platine mit einer M3 Schraube an das Gehäuse befestigen lässt.

Abbildung : Einkerbung zur Befestigung

#### SMD-Testterminals

Um Systemtests zu vereinfachen, wurden wichtige Signale mithilfe von Testterminals leicht für Tastköpfe von Oszilloskopen und Multimetern zugänglich gemacht. Insgesamt wurden auf der Platine

## Firmware Referenz Handbuch

### Toolchain

### Firmware Diagramm

### Daten Strukturen Index

### File Index

### Daten Strukturen Dokumentation

### File Dokumentation

## Software

Um den Guitar-Conveter am PC auslesen zu können, werden bestimmte Treiber benötigt. Im Folgenden werden die nötigen Konfigurationen der Software beschrieben, um den Gitcon betriebsberiet zu machen und ihn in der digital Audio Workstation Ableton Live verwenden zu können.

### An Introduction to MIDI Connections — iConnectivityVirtueller MIDI-Port

Abbildung : MIDI-Port eines klassischen Controllers

Originale MIDI-Controller verfügen über zwei DIN 5-Pol Stecker, um die Parameter untereinander auszutauschen. Da am Mainboard eines PCs jedoch keine DIN-Stecker verbaut sind, werden die Daten über USB vermittelt. Damit die DAW nun MIDI-Geräte voneinander unterscheiden kann, gibt es Softwarebasierte Virtuelle Ports.

Die Software **loopMIDI** kommt daher zum Einsatz. In ihr kann man Instanzen von virtuellen MIDI-Ports erzeugen, sodass die DAW weiß, welche MIDI-Signale tatsächlich vom dem Gitcon-Device stammen.

Ein Bild, das Text, Screenshot, Monitor, Bildschirm enthält.

Automatisch generierte Beschreibung

### MIDI Serial Bridge

Eine MIDI Serial Bridge wird benötigt, um erhaltene Bytes von dem COM-Port an einen MIDI-Port weiterzuleiten, welcher als Schnittstelle zur DAW dient.

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungDas hier verwendete Programm, **hairless-midiserial**, ermöglicht es, einen Seriellen Port auszuwählen, von welchem die Daten, an ein MIDI-Port gesendet (MIDI Out), beziehungsweise empfangen (MIDI In) werden können. Ein zuvor erzeugter virtueller MIDI-Port kann nun in der Benutzeroberfläche selektiert werden

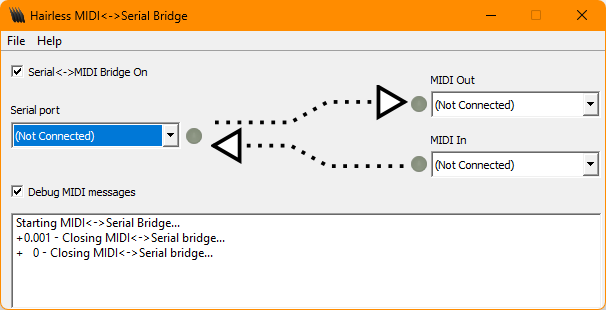


Abbildung : hairless-midiserial Benutzeroberfläche

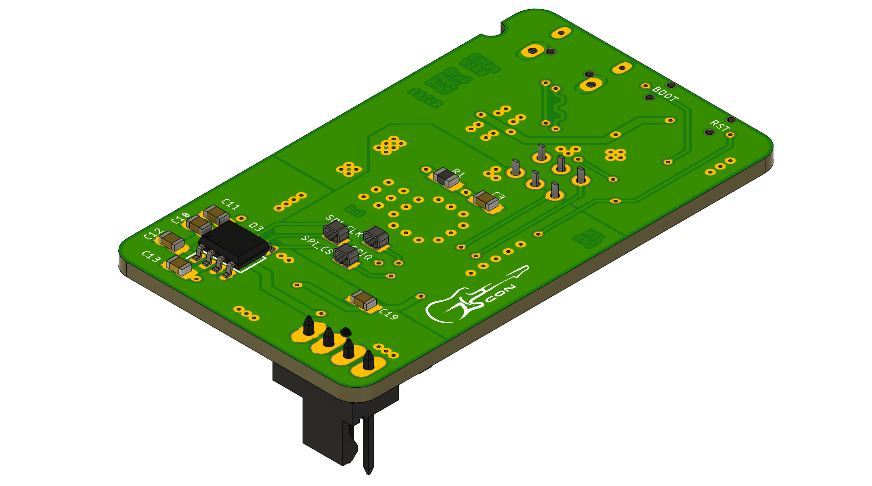
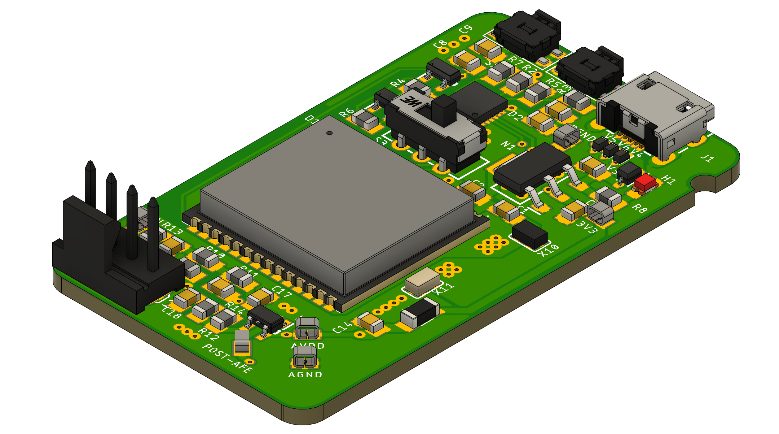
### Ableton Live Setup

#### Einbindung in die DAW

Mit dem Tastenkürzel **Ctrl + ,** werden die Voreinstellungen (Preferences) aufgerufen. Sofern zuvor ein virtueller MIDI-Port erzeugt worden ist, wird er von Ableton Live erkannt und er kann im Reiter **MIDI** selektiert werden.



## CAD-Modelle und Gehäuse



# Fehlererfassung

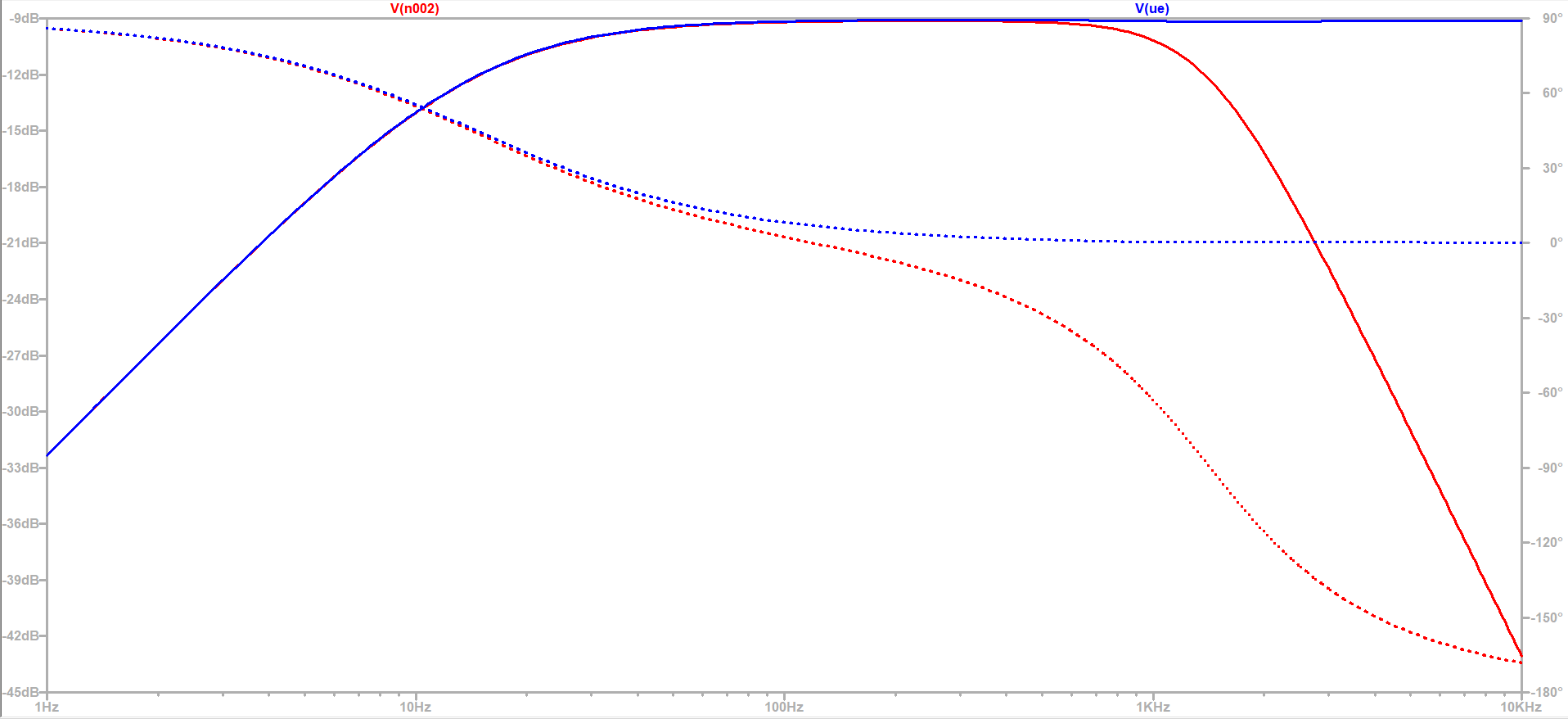
## Platine

* Taster Footprint falsch
* Teil des Doku-Layers wurde nicht auf den Silkscreen gedruckt.
* (Bild einfügen)

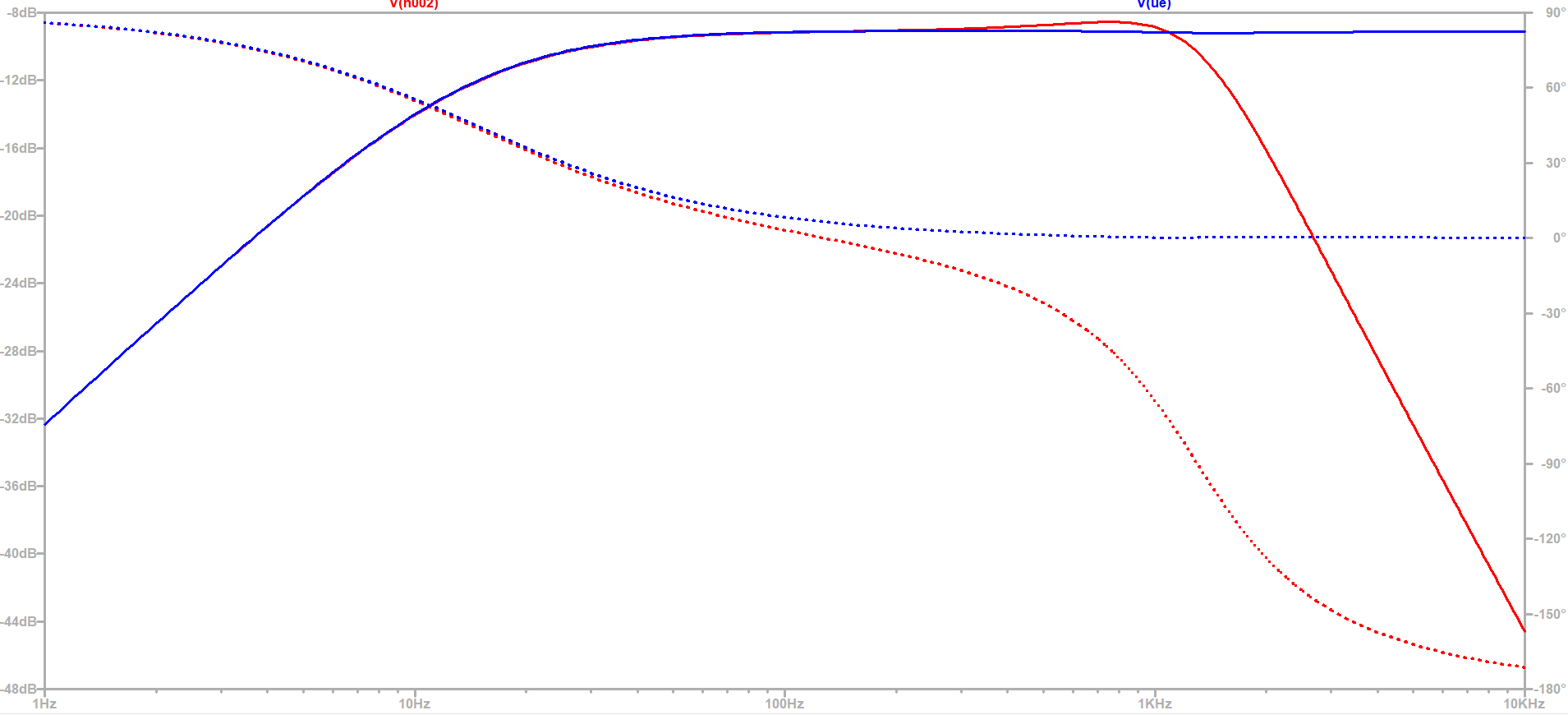
## Bestückung

* 10nF statt 12nF bei Tiefpass verfügbar
* Bild einfügen

Simulation mit 10nF (Ist)



Simulation mit 12nF (Soll)



Fazit: 10nF beeinflusst die Schaltung positiv, da die Resonanz an den Grenzfrequenzen geringer ist.

## ADC Channel 2 auf ADC Channel 1 überbrücken

Bild einfügen

## Fehlende Features

* Die Implementation einer Debug-LED wäre praktisch gewesen, um beliebige Parameter anzeigen zulassen, zum Beispiel:
  + wann eine Transiente erkannt wurde;
  + eine Note übertragen wurde;
* Eine RX/TX-Package-LED an der UART-Bridge

# Glossar

AFE Analogue-Frontend

CAD Computer Aided Design

DAW Digital Audio Workstation

DMA Direct Memory Access

DSV Digitale Signalverarbeitung

ESD Electrostatic Discharge

ESP Kürzel für Espressif Produkte

FFT Fast Fourier Transformation

GPIO General Purpose Input/Output

IDF IoT Development Framework

IoT Internet of Things

LDO Low Dropout Regulator

LED Light Emitting Diode

LVR Linear Voltage Regulator

MIDI Musical Instrument Device Interface

PIO PlatformIO

RTOS Realtime Operating-System

SPI Serial Peripheral Interface

UART Universal Asynchronous Recieve and Transmit

USB Universal Serial Bus

Via Durchkontaktierung

# Abbildungsverzeichnis

[Abbildung 1: Filter 1. Ordnung 21](https://htlsalzburg-my.sharepoint.com/personal/laurenz_hoelzl03_htl-salzburg_ac_at/Documents/DA_2022-23_5AHEL_SreS_BraD_GruS_HoeL_Gitcon.docx#_Toc129272970)

[Abbildung 2: Aufbau einer E-Gitarre 27](#_Toc129272971)

[Abbildung 3: Funktionsblockdiagramm des Gitcon 38](#_Toc129272972)

[Abbildung 4: Spannungsregler und USB-Connector im Schaltplan 39](#_Toc129272973)

[Abbildung 5: Warnung zu ESD-Empfindlichen Objekten 39](https://htlsalzburg-my.sharepoint.com/personal/laurenz_hoelzl03_htl-salzburg_ac_at/Documents/DA_2022-23_5AHEL_SreS_BraD_GruS_HoeL_Gitcon.docx#_Toc129272974)

[Abbildung 6: Symbol der Bipolaren TVS-Diode 40](https://htlsalzburg-my.sharepoint.com/personal/laurenz_hoelzl03_htl-salzburg_ac_at/Documents/DA_2022-23_5AHEL_SreS_BraD_GruS_HoeL_Gitcon.docx#_Toc129272975)

[Abbildung 7: SC7538-08UTG Block Diagramm 40](https://htlsalzburg-my.sharepoint.com/personal/laurenz_hoelzl03_htl-salzburg_ac_at/Documents/DA_2022-23_5AHEL_SreS_BraD_GruS_HoeL_Gitcon.docx#_Toc129272976)

[Abbildung 8: Beschaltung der USB-Bridge 45](#_Toc129272977)

[Abbildung 9:Versorgungsleitungen 46](#_Toc129272978)

[Abbildung 10: Komponenten des Pi-Filters 46](https://htlsalzburg-my.sharepoint.com/personal/laurenz_hoelzl03_htl-salzburg_ac_at/Documents/DA_2022-23_5AHEL_SreS_BraD_GruS_HoeL_Gitcon.docx#_Toc129272979)

[Abbildung 11: Visualisierung einer Sternpunktmasse 47](#_Toc129272980)

[Abbildung 12: SMD-Jumper 47](https://htlsalzburg-my.sharepoint.com/personal/laurenz_hoelzl03_htl-salzburg_ac_at/Documents/DA_2022-23_5AHEL_SreS_BraD_GruS_HoeL_Gitcon.docx#_Toc129272981)

[Abbildung 13: Bottom (links) und Top (rechts) Masse-Layer 47](#_Toc129272982)

[Abbildung 14: Ground-Stitches auf der Platine 48](#_Toc129272983)

[Abbildung 15: Einkerbung zur Befestigung 49](https://htlsalzburg-my.sharepoint.com/personal/laurenz_hoelzl03_htl-salzburg_ac_at/Documents/DA_2022-23_5AHEL_SreS_BraD_GruS_HoeL_Gitcon.docx#_Toc129272984)

[Abbildung 16: MIDI-Port eines klassischen Controllers 51](https://htlsalzburg-my.sharepoint.com/personal/laurenz_hoelzl03_htl-salzburg_ac_at/Documents/DA_2022-23_5AHEL_SreS_BraD_GruS_HoeL_Gitcon.docx#_Toc129272985)

[Abbildung 17: hairless-midiserial Benutzeroberfläche 52](#_Toc129272986)

# Quellenverzeichnis

# Anhang

## PCB-Fertigungsunterlagen

1. Verzögerung [↑](#footnote-ref-2)
2. Transient Voltage Suppression [↑](#footnote-ref-3)