

HTBLuVA Salzburg

**Höhere Lehranstalt für
Elektronik und Technische Informatik**

DIPLOMARBEIT

Gesamtprojekt

Gitcon

Entwicklung einer MIDI-Schnittstelle für E-Gitarren

Daniel Bräumann	5AHEL	Betreuer:
Simon Grundner	5AHEL	Prof. Dipl.-Ing. Siegbert Schrempf
Laurenz Hödl	5AHEL	

ausgeführt im Schuljahr 2022/23

Abgabevermerk:

Datum: 31.03.2023

übernommen von:



 HTBLuVA Salzburg	HÖHERE TECHNISCHE BUNDESLEHR- UND VER- SUCHSANSTALT Salzburg
Elektronik und Technische Informatik	

DIPLOMARBEIT

DOKUMENTATION

Namen der Verfasserinnen / Verfasser	Daniel Bräumann Simon Grundner Laurenz Hözl
Jahrgang Schuljahr	5AHEL 2022/23
Thema der Diplomarbeit	Gitcon – Entwicklung einer MIDI-Schnittstelle für E-Gitarren ✓

Aufgabenstellung	Virtuelle Instrumente sind in der modernen Musikproduktion 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	Das Gitarrensignal muss zur weiteren Verarbeitung vorbereitet werden. Um die Funktion des FFT-Algorithmus zu prüfen, wurden diverse Testungen durchgeführt.
--------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------

Ergebnisse	Das Projekt erkennt die gespielten Noten und überträgt sie ohne spürbare Latenz an die DAW. ✓
------------	-----------------------------------------------------------------------------------------------

 HTBLuVA <i>Salzburg</i>	HÖHERE TECHNISCHE BUNDESLEHR- UND VER- SUCHSANSTALT Salzburg
Elektronik und Technische Informatik	

DIPLOMA THESIS

Documentation

Author(s)	Daniel Bräumann Simon Grundner Laurenz Hözl
Form Academic year	2022/23
Topic	Gitcon - Development of a MIDI gateway for electric guitars

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	The guitar signal needs to be prepared for further processing. Various tests were carried out to verify the functionality of the FFT algorithm. ✓
-------------	---------------------------------------------------------------------------------------------------------------------------------------------------

Results	The project detects the played notes and transfers them to the DAW without noticeable latency. ✓
---------	--------------------------------------------------------------------------------------------------

Vorwort

Die vorliegende Diplomarbeit ermöglicht die Verwendung einer E-Gitarre als MIDI-Controller. Das Gerät soll einzelne Noten 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 Hözl. Da sich beide in ihrer Freizeit viel mit Musik befassen. Durch ihr Interesse an digitaler Audio-Verarbeitung, war die Projektidee geboren. ✓

Moderne Musikstücke werden üblicherweise in einer digitalen Produktionsumgebungen aus mehreren Tonspuren zusammen gemischt. Um mit diesen Programmen zu interagieren, kommen MIDI-Controller zum Einsatz. Diese verwenden das MIDI-Protokoll, um Noten- und Parametereingaben an den Computer zu übertragen. Die Idee war es nun, eine elektrische Gitarre mit diesem Protokoll kompatibel zu machen, sämtliche gespielte Noten zu erkennen und in MIDI-Signale umzuwandeln.

Die individuellen Aufgabenstellungen wurden 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

Inhaltsverzeichnis

1 Überblick.....	13
1.1 Was ist ein MIDI-Controller?	13
1.2 Gitcon als MIDI-Controller	13
2 Systemspezifikationen	15
2.1 Zielbestimmungen	15
2.1.1 Musskriterien.....	15
2.1.2 Wunschkriterien.....	15
2.1.3 Abgrenzungskriterien	15
2.2 Produkteinsatz	15
2.2.1 Anwendungsbereiche.....	15
2.2.2 Zielgruppen.....	15
2.2.3 Betriebsbedingungen.....	15
2.3 Produktumgebung	16
2.3.1 Software.....	16
2.3.2 Hardware.....	16
2.4 Produktfunktionen	16
2.5 Produktleistungen.....	17
2.6 Benutzungsoberfläche	18
2.7 Entwicklungsumgebung	19
2.7.1 Software.....	19
2.7.2 Hardware.....	19
2.7.3 Orgware	19
2.8 Qualitätsziel Bestimmungen	20
2.9 Globale Testszenarien und Testfälle	20
3 Projektmanagement	24

4.4.1	Agile Softwareentwicklung mit C	72
4.5	ESP32	73
4.5.1	Bootloader Brennen mit dem ESP-Tool.....	73
4.5.2	Einrichten in PlatformIO (PIO)	76
4.5.3	ESP IoT Development Framework (ESP IDF).....	81
4.6	Realtime Operating-System (RTOS)	81
4.6.1	FreeRTOS	81
4.7	Universal Asynchronous Receive and Transmit (UART)	82
4.8	Musical Instrument Digital Interface (MIDI)	83
4.8.1	Status Bytes.....	84
5	Ergebnisse	87
5.1	Blockschaltbild	87
5.2	Hardware	89
5.2.1	Versorgung	89
5.2.2	Analog-Frontend (AFE)	93
5.2.3	Digital Frontend	99
5.2.4	Layout der Platine	102
5.3	Firmware	108
5.3.1	Treiber Firmware Diagramm	108
5.3.2	Doxxygen	109
5.4	Software	111
5.4.1	Virtueller MIDI-Port	111
5.4.2	MIDI Serial Bridge.....	112
5.4.3	Ableton Live Setup	113
5.5	CAD-Modelle	114
6	Fehlererfassung	115

1 Überblick

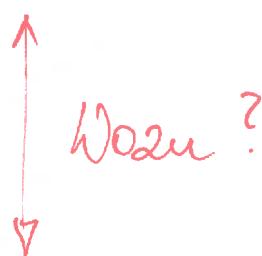
Das Projekt Gitcon beschäftigt sich rundum mit digitaler Musikproduktion und mit den Werkzeugen, Audio elektronisch zu manipulieren. Das Gitcon-Device ist ein MIDI-Controller, der ~~das~~ ^{signale} auf einer E-Gitarre Gespielte analysiert und in digitale Noten, sogenannte MIDI-Noten umwandelt.

1.1 Was ist ein MIDI-Controller?

Wie bereits erwähnt werden in der modernen Musikproduktion digitale Produktionsumgebungen, sogenannte Digital Audio Workstations (DAW), genutzt. Diese lassen sich über das MIDI-Protokoll mithilfe eines MIDI-Controllers bedienen. Die am weitesten verbreitete Form dieses Eingabegeräts ist das Keyboard. Dieses verfügt über Klaviertasten, welche für die Noteneingabe genutzt werden und einige Regler und Knöpfe mithilfe derer Parameter in der DAW eingestellt werden können.



Abbildung 2: MIDI-Klavier (Novation¹ Launchkey)



1.2 Gitcon als MIDI-Controller

Sowie ein MIDI-Klavier beim Tastenanschlag die Note vermittelt, soll unser Projekt eine MIDI-Note mit dem Saitenanschlag einer Gitarre senden. Wird

¹ <https://novationmusic.com/de/keys/launchkey>

2 Systemspezifikationen

2.1 Zielbestimmungen

2.1.1 Musskriterien

Einzelne Noten müssen zuverlässig erkannt und umgewandelt werden.

2.1.2 Wunschkriterien

Noten sollen mit möglichst geringer Latenz übertragen werden.

2.1.3 Abgrenzungskriterien

Projekt soll nicht auf verschiedene E-Gitarren getestet und optimiert werden.

Das Produkt soll nicht auf Basis anderer Musikinstrumente funktionieren, welche ähnliche elektrische Ausgänge haben.

2.2 Produkteinsatz

2.2.1 Anwendungsbereiche

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

2.2.2 Zielgruppen

Zielgruppen sind sowohl Musikproduktions-Neueinsteiger, welche Gitarre spielen als auch bereits erfahrene Produzenten, welche auf der Suche nach einzigartigen und inspirierenden Eingabemethoden sind.

2.2.3 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.

2.5 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.



2.7 Entwicklungsumgebung

2.7.1 Software

- PlatformIO (Core v6.1.6, Home v3.4.3)
- ESP IoT Development Framework (v5.3.0)
- Autodesk EAGLE (v9.6.2)
- Autodesk Fusion 360 (v2.0.15509)
- ESP Flash Download Tool (v3.9.4)
- LTSpice (XVII)
- Saturn PCB Toolkit (v8.23)
- Ableton Live Suite (v11.2.7) ✓
- Audacity

2.7.2 Hardware

- Prototypen
 - Firmware Test-board
 - Filter Prototyp ✓

2.7.3 Orgware

- GitHub Desktop (v3.2.0)
 - (<https://github.com/s-grundner/MTAP-MIDI-Guitar-Converter>)
- DrawIO/diagrams.net
- Obsidian (v1.1.9)
- Pro Create (v5.3.1) ✓

Pitch Bend	Aufwärtsrampe (8192 (Mitte) bis 16383 (Max))	
Pitch Bend	Abwärtsrampe ()	
Pitch Bend	Aufwärtsrampe (0 (Mitte) bis 8192 (Mitte))	
MIDI Note Off	Note: C4	Velocity: 0

Tabelle 1: MIDI-Unitest

Eine MIDI-Spur wird mit folgenden Einstellungen zur Aufnahme scharfge stellt:



Abbildung 3: Konfiguration einer MIDI-Spur beim MIDI-Unitest

Anschließend wird die Aufnahme gestartet:



Abbildung 4: Starten der Aufnahme des MIDI-Unitests Das in Abbildung 5 Ergebnis wird erwartet:

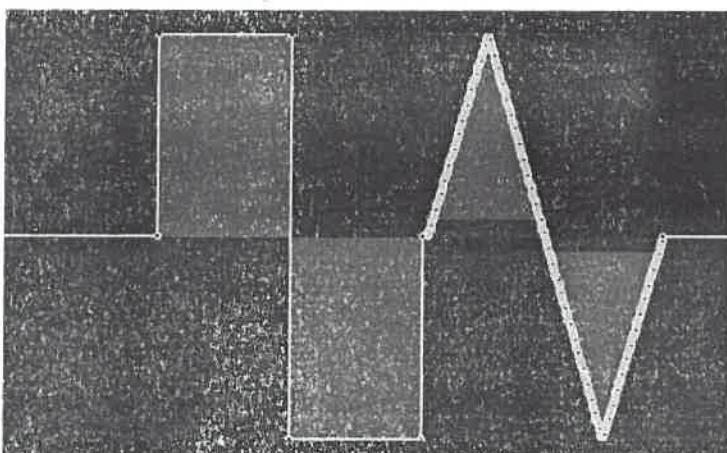


Abbildung 5: Ergebnis einer Aufnahme des MIDI-Unitests

/T0050/ Ausgabe: Live-Konversion

→ Wohin?

Beim Anschlagen der tiefen E-Saite wurde folgendes Ergebnis aufgezeichnet

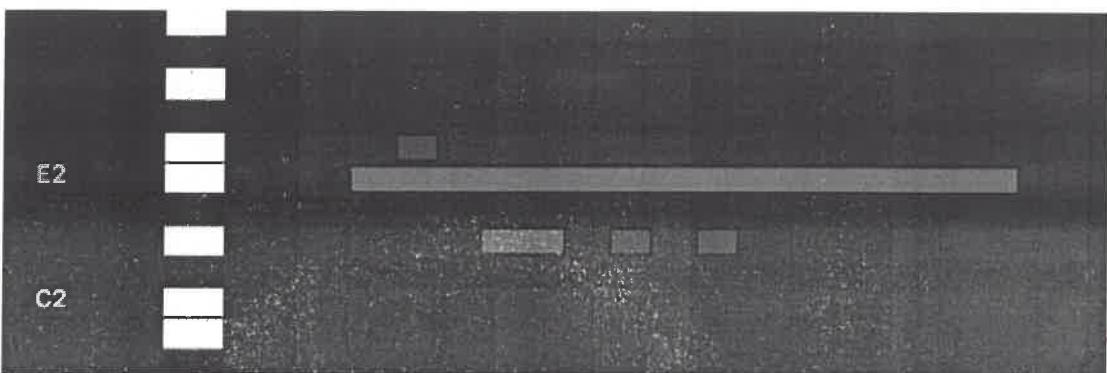
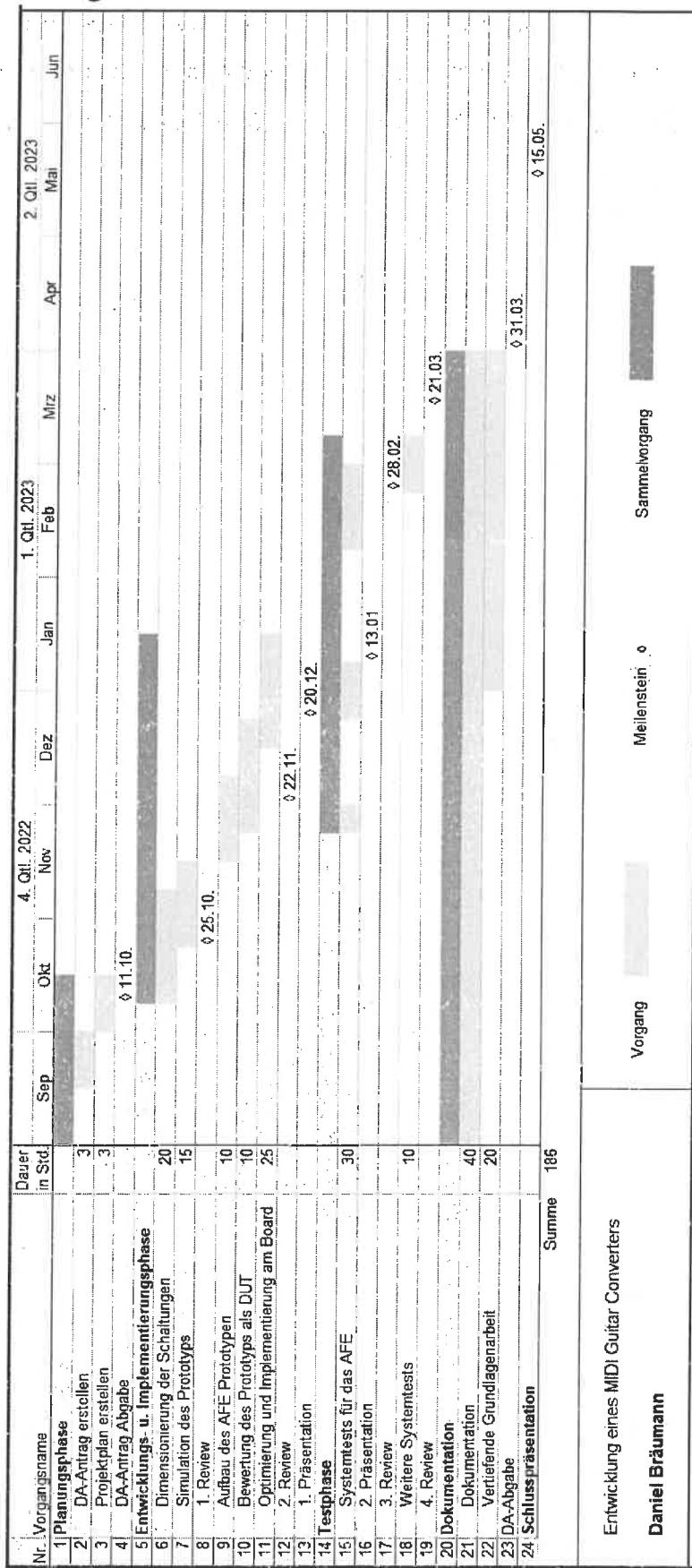


Abbildung 8: Ergebnis der Live-Konversion

→ auf S 22

3.2 GANTT-Diagramme



Nr.	Vorjahrsname	Dauer in Std	1. Qtr. 2023			4. Qtr. 2022			2. Qtr. 2023		
			Sep	Okt	Nov	Dez	Jan	Feb	Mrz	Apr	Mai
1.	Planungsphase										
2.	Recherche zur Umsetzbarkeit	3									
3.	DA-Antrag erstellen	3									
4.	Projektplan erstellen	3									
5.	DA-Antrag Abgabe										
6.	Entwicklungs- u. Implementierungsphase										
7.	Recherche zur FFT	5									
8.	Aufsetzen der Entwicklungsumgebung	10									
9.	1. Review										
10.	FFT Algorithmus	30									
11.	Testdatenerhebung	10									
12.	Überprüfung mit Matlab	3									
13.	2. Review										
14.	1. Präsentation										
15.	Noten Umwandlung & Übertragung	25									
16.	2. Präsentation										
17.	3. Review										
18.	Testphase										
19.	Test und Feinfertigung der Firmware	15									
20.	4. Review										
21.	Dokumentation										
22.	Dokumentation	60									
23.	Vertiefende Grundlagenarbeit	30									
24.	DA-Abgabe										
25.	Schlusspräsentation										
	Summe	197									
Entwicklung eines MIDI Guitar Converters			Vorgang			Meilenstein			Sammelvorgang		
Laurenz Hözl											

3.3.2 Integrierte Tools

3.3.2.1 Trello

Trello ist eine webbasierte Projektmanagement-Software, welche es dem Team ermöglicht hat, die einzelnen Aufgaben visuell zu organisieren. Mittels einer flexiblen Board-Struktur können Karten mit den Arbeitsaufgaben in Listen organisiert werden.

Die Karten können mit Checklisten, Fälligkeitsdaten und Benutzerzuweisungen versehen werden, um den Überblick über die Aufgaben und Fortschritt zu behalten.

Auf der Website des Repository sind unter dem Reiter „Projects“ die Trello-Boards für das Projekt vorhanden.

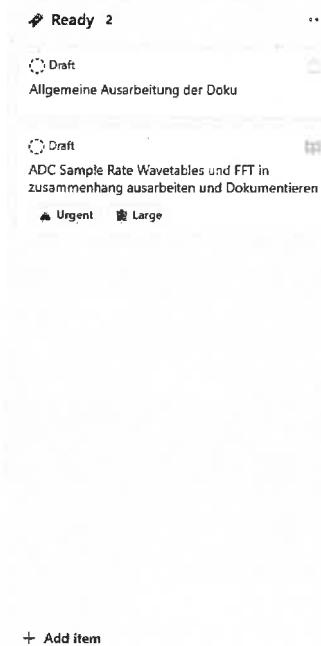


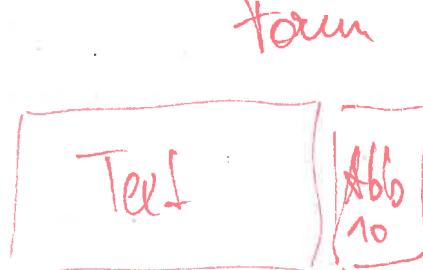
Abbildung 9: Trello Liste mit zwei Karten

3.3.2.2 GitHub Copilot

GitHub Copilot ist ein auf künstlicher Intelligenz (KI)-basiertes Tool von GitHub und kann als Erweiterung für den Visual Studio Code Editor installiert werden. Copilot ist darauf optimiert, Code anhand der bereits eingegebenen Informationen in der Datei zu generieren oder vervollständigen. Dieses Werkzeug war beim Entwickeln eines Code-Prototypen sehr hilfreich. Es ist jedoch nötig, die generierten Codeblöcke immer zu validieren.



Abbildung 10: Copilot Logo



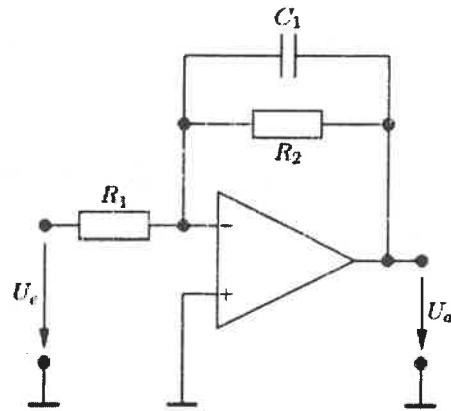
stellenden Betrieb bei sehr hohen Frequenzen. Allerdings tendiert der in der Schaltung verwendete ~~Induktor~~ ^{Worum} bei niedrigeren Frequenzen größer zu sein, wodurch die gesamte Schaltung ~~komplexer~~ wird. Außerdem steigen die Kosten, wenn eine höhere Qualität und eine kleine Größe erwünscht ist. 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.

Welche Differenzen

4.1.3 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 Elemente eine externe Stromquelle. [1]



→ *Verweis aufgen lassen*

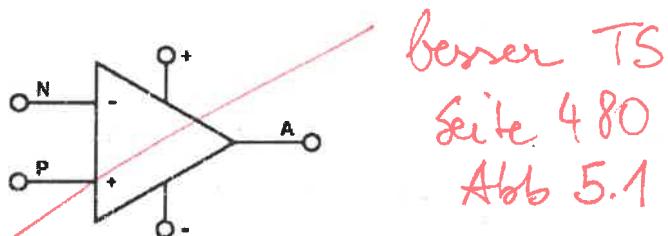
Da keine ~~Induktoren~~ zum Einsatz kommen, wird die Schaltung kompakter und weniger ~~schwer~~. Die Eingangsimpedanz ist hoch und die Ausgangsimpedanz ist niedrig, so können niedrige Lasten am Ausgang angesteuert werden. Weiters ist die Last von der internen Schaltung isoliert, daher hat die Veränderung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

Berücksichtigen!

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

4.1.5 Operationsverstärker

~~Der Operationsverstärker (kurz OPV)~~ ist eines der vielseitigsten Bauteile in der Elektronik. Dieses Thema ist sehr komplex und man könnte allein darüber eine ganze wissenschaftliche Arbeit verfassen, weshalb an dieser Stelle nur das Grundkonzept im Kontext der Filterthematik vorgestellt werden soll. OPVs sind elektronische Verstärker, deren Name sich vom mathematischen Begriff „Operator“ ableiten lässt. Sie werden vielfältig in der Audiotechnik verwendet und ersetzen häufig diskret aufgebaute Schaltungen mit nur einen integrierten Schaltkreis (IC). Operationsverstärker haben meist zwei Eingänge und einen Ausgang, worüber Signale verstärkt, addiert, subtrahiert, integriert, differenziert und geschaltet werden. Der OPV kommt oft als Differenzenverstärker zum Einsatz, sodass ein Signal an einem Eingang anliegt, welches über den Ausgang zum Eingang zwei rückgekoppelt wurde. Damit lassen sich schnell und unkompliziert mit Hilfe von Widerständen und Kondensatoren Verstärkerschaltungen realisieren. Weiters kommen zu diesen Ein- und Ausgängen noch Anschlüsse für die Spannungsversorgung hinzu. Man spricht hier von aktiven Bauteilen, welche eine Verstärkung bereitstellen können, also wird auch hier eine Versorgungsspannung benötigt.



besser TS
Seite 480
Abb 5.1

Abbildung 14: Operationsverstärker Symbol

Die obige Grafik zeigt das Schaltbild eines Operationsverstärkers. P ist der nicht-invertierende, N ist der invertierende Eingang. A ist der Ausgang, plus (+) und minus (-) sind die Eingänge für die Versorgungsspannung. [2]

Prüfen

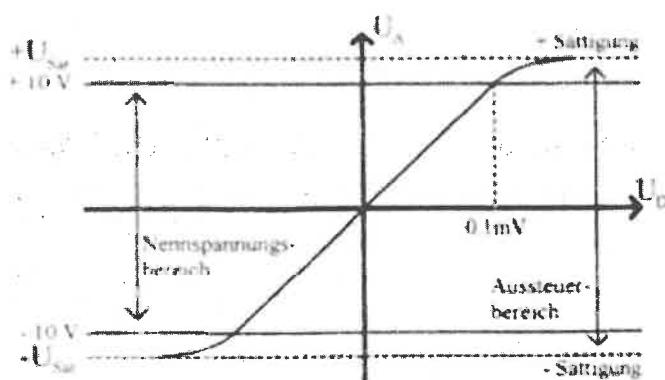


Abbildung 16: Differenzenverstärkung OPV

Abb 16

In dieser Grafik wird die Differenzverstärkung eines Operationsverstärkers, die auch als Leerlaufverstärkung bekannt ist, dargestellt. Diese Verstärkung hängt von der Versorgungsspannung des OPVs ab und hat ihre Grenzen im Sättigungsbereich, der sowohl im Negativen als auch im positiven Bereich auftritt. Im Idealfall ist die Leerlaufverstärkung linear und frequenz-unabhängig, aber bei einem realen Operationsverstärker gibt es aufgrund seiner Bandbreite eine obere Grenzfrequenz. Obwohl die Bandbreite bei 0 Hz beginnt, gibt es bestimmte Grenzen, die zu beachten sind. [2]

*Nachre***4.1.5.2 Invertierender und nicht-invertierender OPV**

Damit ein Operationsverstärker als Differenzverstärker verwendet werden kann, muss ein Teil des Ausgangssignals an einen der Eingänge zurückgeführt werden. [3] Dies wird als Gegenkopplung oder Feedback bezeichnet. Zunächst wird die einfachste und am häufigsten verwendete Operationsverstärkerschaltung betrachtet, der Impedanzwandler, der auch als Spannungsfolger bezeichnet wird.

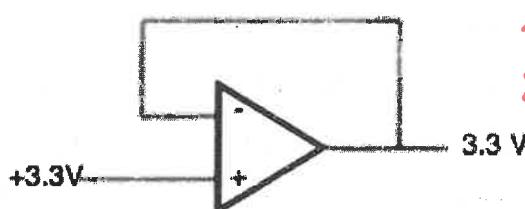


Abbildung 17: OPV als Impedanzwandler

- hierzu*
- 1) Invertierender
 - 2) Nichtinvertierender
- dannach Impedanzwandler von nichtinvert. V. ableiten*

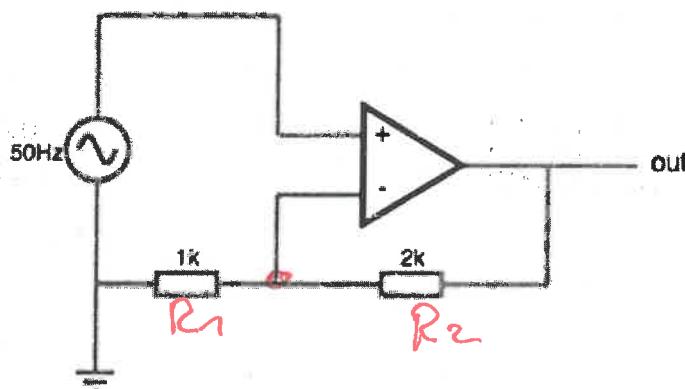


Abbildung 19: Nicht-invertierender OPV

Diese Abbildung zeigt jetzt die gegenteilige Schaltung. Der nicht-invertierende Verstärker arbeitet ebenso mit Gegenkopplung, die Widerstände R_1 und R_2 bilden einen Spannungsteiler, der als unbelastet angenommen wird, da der OPV eine hohe Eingangsimpedanz besitzt. [3]

nicht Eingang

$$\text{Für den Verstärkungsfaktor gilt: } U_a = U_e \left(1 + \frac{R_2}{R_1}\right) \quad (4)$$

Auch hier wird eine Verstärkung von drei erreicht. Es kommt bei Schaltungen dieser Art weniger auf die Widerstandswerte, sondern mehr auf deren Verhältnis an. Jedoch fließt bei zu niedrigem Wert ein hoher Strom, den der Operationsverstärker im Feedbackloop leisten muss, dies kann zu Verzerrungen oder Überhitzungen des OPVs führen. Andernfalls sind zu hohe Werte ebenfalls kritisch, da sie zu erhöhtem Rauschen oder Oszillation führen können. [3]

4.1.6 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** (*Abtastwerte*) 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

4.1.6.1 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.

$$f_{\text{Abtast}} > 2 * f_{\max} \quad (5)$$

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.

Was ist ein praktischer Wert?

4.2 Aktiver Filter

4.2.1 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.

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

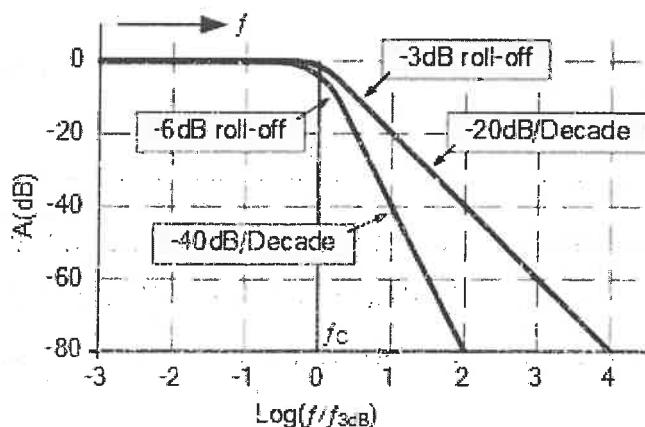


Abbildung 23: Flankensteilheit der Ordnungen [24]

Während der Sallen-Key erster Ordnung nur eine Resonanzfrequenz hat, hat der Sallen-Key zweiter Ordnung zwei Resonanzfrequenzen. Diese beeinflussen die Filtercharakteristik maßgeblich in deren Umgebung. Auch die Dämpfung verändert sich je nach Ordnung. Das bedeutet, dass ein Filter zweiter Ordnung eine höhere Dämpfungsfähigkeit aufweist und somit unerwünschte Frequenzen besser unterdrücken, kann als ein Filter erster Ordnung. [4] ✓

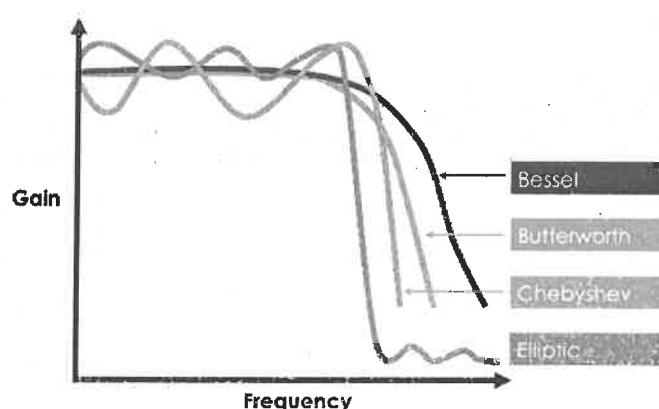


Abbildung 25: Frequenzverhalten der Filtercharakteristiken [32]

4.2.2.1 Toleranzschema

Um einen Filter erfolgreich zu dimensionieren, müssen zuallererst die Anforderungen des zu entwerfenden Filters festgelegt werden, das sogenannte Toleranzschema. Dies beschreibt die zulässigen Bereiche des Amplitudengangs, dazu wird der Frequenzgang in drei Arbeitsbereiche unterteilt.

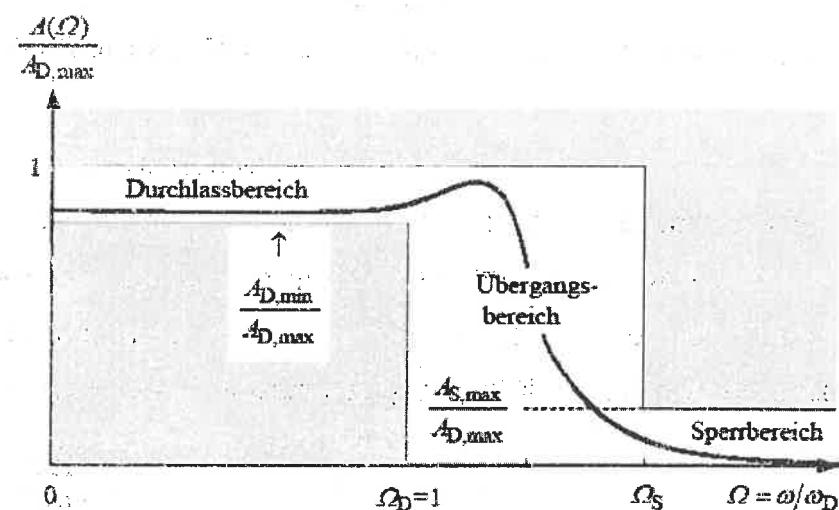


Abbildung 26: Toleranzschema [31]

- Im Durchlassbereich wird die maximal erlaubte Welligkeit des Amplitudengangs angegeben. Das Nutzsignal soll außerdem möglichst nicht beeinträchtigt werden. Der Toleranzbereich wird durch das Verhältnis $\frac{A_{D,\text{min}}}{A_{D,\text{max}}}$ festgelegt, wobei A_D die Grundverstärkung des Filters ist.

$$I_3(s) = \frac{U_r(s)}{Z_2(s) + Z_3(s)}$$

$$I_3(s) = \frac{U_a(s)}{\nu * Z_3(s)}$$

$$I_1 = I_2 + I_3$$

durch Einsetzen den Teilgleichungen:

$$\frac{U_e - U_r}{Z_1} = \frac{U_r - U_a}{Z_4} + \frac{U_a}{\nu * Z_3}$$

$$\text{mit: } U_r = I_3 * (Z_2 + Z_3)$$

$$\frac{U_e - I_3 * (Z_2 + Z_3)}{Z_1} = \frac{I_3 * (Z_2 + Z_3) - U_a}{Z_4} + \frac{U_a}{\nu * Z_3}$$

$$\frac{U_e}{Z_1} - \frac{U_a * (Z_2 + Z_3)}{\nu * Z_1 * Z_3} = \frac{U_a * Z_2 + U_a * Z_3 - U_a * \nu * Z_3 + U_a * Z_4}{\nu * Z_3 * Z_4}$$

$$\frac{U_e}{Z_1} - \frac{U_a * (Z_2 + Z_3)}{\nu * Z_1 * Z_3} = \frac{U_a[(Z_2 + Z_4) + Z_3(1 - \nu)]}{\nu * Z_3 * Z_4}$$

$$\frac{U_e}{Z_1} = \frac{U_a[(Z_2 + Z_4) + Z_3(1 - \nu)]}{\nu * Z_3 * Z_4} + \frac{U_a(Z_2 + Z_3)}{\nu * Z_1 * Z_3}$$

$$\frac{U_e}{Z_1} = \frac{U_a\{Z_1[(Z_2 + Z_4) + Z_3(1 - \nu)] + Z_4(Z_2 + Z_3)\}}{\nu * Z_1 * Z_3 * Z_4}$$

Allgemeine Übertragungsfunktion:

$$\frac{U_a}{U_e} = \frac{\nu * Z_3 * Z_4}{Z_1 * Z_2 + Z_1 * Z_4 + Z_2 * Z_4 + Z_3 * Z_4 + Z_1 * Z_3(1 - \nu)} \quad (6)$$

Mit $s = j\omega$ folgt:

$$\underline{G}_{TP} = \frac{\nu}{1 - R_1 R_2 C_1 C_2 \omega^2 + j\omega [C_1(R_1 + R_2) + R_1 C_2(1 - \nu)]} \quad (8)$$

Durch einen Vergleich sich entsprechender Komponenten im normierten allgemeinen TP-Filter zweiter Ordnung können Grenzfrequenzen und Dämpfungswerte (a) 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.

$$\left(\frac{\omega}{\omega_g}\right)^2 = R_1 R_2 C_1 C_2 \omega^2$$

$$\omega_g^2 = \frac{1}{R_1 R_2 C_1 C_2} ; f_g = \frac{1}{2\pi\sqrt{R_1 R_2 C_1 C_2}}$$

$$a = \frac{\omega}{\omega_g} = \omega [C_1(R_1 + R_2) + R_1 C_2(1 - \nu)]$$

$$a = \omega_g [C_1(R_1 + R_2) + R_1 C_2(1 - \nu)] \quad (9)$$

Nach Einsetzen von ω_g :

$$a = \frac{C_1(R_1 + R_2) + R_1 C_2(1 - \nu)}{\sqrt{R_1 R_2 C_1 C_2}}$$

Mit $R_1 = R_2 = R$ und $C_1 = C_2 = C$

$$f_g = \frac{1}{2\pi R C} ; a = 3 - \nu$$

Wenn Widerstände und Kondensatoren unterschiedliche Werte haben:

$$f_g = \frac{1}{2\pi\sqrt{R_1 R_2 C_1 C_2}} \quad (10)$$

4.3 Grundlagen der digitalen Signal-Verarbeitung (DSV)

4.3.1 Allgemein

Schlägt man auf einer Gitarre eine Saite an, so schwingt diese mit einer gewissen Frequenz. Am ESP32 soll nun mithilfe eines Algorithmus die Frequenz und daraus folgend die gespielte Note erkannt werden. Hierfür wird der Fast Fourier Transform (FFT) Algorithmus verwendet. Mithilfe der FFT kann das Frequenzspektrum des Signals ermittelt 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 (Tonabnehmer) der Gitarre auftreten können. Diese „Zusatzschwingungen“ treten bei allen Instrumenten in einer individuellen Kombination auf und sind der Grund, warum sich die Gleiche Note auf zum Beispiel Klavier und Gitarre unterschiedlich anhört. H

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 ein möglichst obertonfreies Signal zu gewährleisten, ist es ratsam, den Pickup, welcher am Hals sitzt, zu wählen. Die Brücke reflektiert einige Schwingungen und beeinflusst dadurch das Signal. ✓

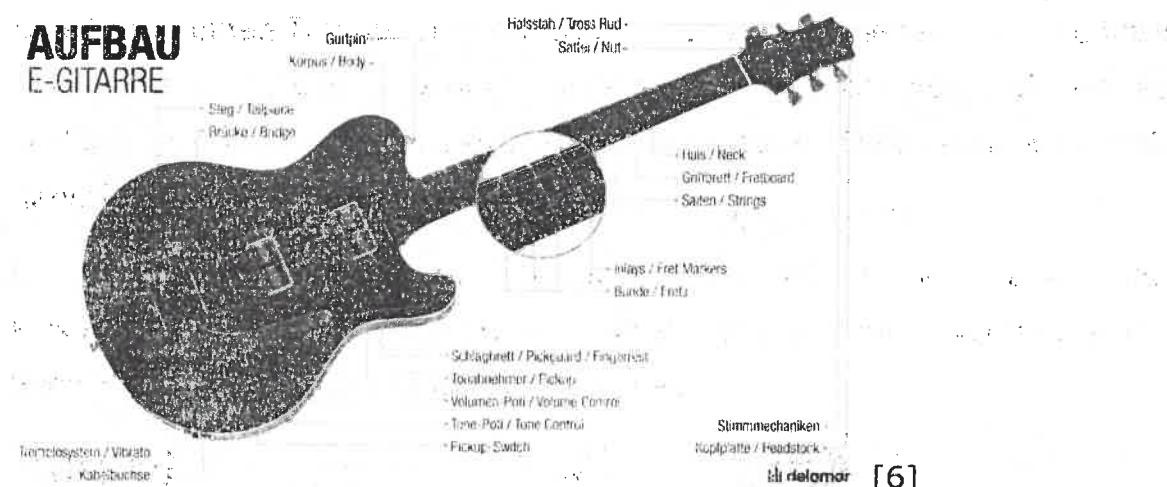


Abbildung 30: Aufbau einer E-Gitarre ✓

4.3.3 Ursprung der Fourier Transformation

4.3.3.1 Fourier Reihe

1807 fand Jean Baptiste Fourier heraus, dass sich eine periodische Funktion als eine Linearkombination von Sinus- und Cosinus-Schwingungen, eine sogenannte Fourier-Reihe, ausdrücken lässt:

$$x_p(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty} (a_k * \cos(2\pi k f_0 t) + b_k * \sin(2\pi k f_0 t)) \quad (11)$$

Hierbei sind a_k und b_k sogenannte Fourier-Koeffizienten, welche den Amplituden der entsprechenden (also kten) Schwingungsanteile gleichkommen. Für den Fall „ $k = 0$ “ existiert das, den arithmetischen Mittelwert darstellende, zeitunabhängige Glied $\frac{a_0}{2}$. Die Grundfrequenz der Fourier-Reihe ist über f_0 dargestellt.

Wird nun die Cosinus-Funktion durch $\cos(j\omega t) = \frac{1}{2} * (e^{-j\omega t} + e^{j\omega t})$ und Sinus-Funktion durch $\sin(\omega t) = \frac{1}{2j} * (e^{j\omega t} - e^{-j\omega t})$ ersetzt so erhält man die komplexe Fourier-Reihe.

$$x_p(t) = \sum_{k=-\infty}^{\infty} c_k * e^{j\omega t} \quad (12)$$

Der Koeffizient c_k ist der komplexe Fourier Koeffizient, aus welchen sich Amplituden und Phasen der Harmonischen berechnen lassen. Multipliziert man hier beide Seiten mit $e^{-j\omega t}$ und integriert diese anschließend von $-\frac{T_0}{2}$ bis $\frac{T_0}{2}$ erhält man die als Analysegleichung bezeichnete Bestimmungsgleichung:

$$c_k = \frac{1}{T_0} * \int_{-\frac{T_0}{2}}^{\frac{T_0}{2}} x_p(t) * e^{-j\omega t} dt \quad (13)$$

Das Bestimmen von c_k über eine Integration gestaltet sich in der Praxis sehr mühsam. Mithilfe der FFT lassen sich diese Koeffizienten viel einfacher und effizienter berechnen. [8, p. 25 ff.]

4.3.4 Herleitung der Diskreten Fourier Transformation

Tatsächlich handelt es sich bei der Diskreten Fourier Transformation (DFT) um eine Annäherung der Fourier Transformation, welche es ermöglicht sie effizient von einem digitalen Rechner berechnen zu lassen.

Hierbei wird die Formel für die Fourier Transformierte (16) als Ausgangspunkt genutzt. Das zeitkontinuierliche Signal wird durch seinen Abtastwert $x(nT)$ und das Differential durch das Abtastintervall T ersetzt. Zur Annäherung des Integrals wird die Summe verwendet:

$$X_s(f) = \sum_{n=-\infty}^{+\infty} x(nT) * e^{-j\omega nT} * T \quad (18)$$

Da eine unendliche Anzahl an Abtastwerten unmöglich zu berechnen ist, werden eine endliche Anzahl N dieser herausgeschnitten/„gefenstert“ (engl: windowing). Außerdem kann der Faktor T aus „Bequemlichkeit“ weggelassen werden:

$$X_{sw}(f) = \sum_{n=0}^{N-1} x(nT) * e^{-j\omega n f_s} \quad (19)$$

Dies Funktion ist f_s -periodisch und hat nur an N -Stellen linear unabhängige Funktionswerte. Ausgewertet wird sie an N gleichentfernten Frequenzstellen $f = 0, \frac{f_s}{N}, 2 * \frac{f_s}{N}, \dots, (N-1) * \frac{f_s}{N}$. Werden der Einfachheit halber wieder einige Faktoren $(\frac{f_s}{N}, T)$ und die Kennzeichnung sw weggelassen ergibt sich die Definition (Analysegleichung) der DFT:

$$X[k] = \sum_{n=0}^{N-1} x[n] * e^{-jkn \frac{2\pi}{N}} \quad (20)$$

Die inverse DFT (IDFT) (Synthesegleichung) ist definiert als:

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] * e^{jkn \frac{2\pi}{N}} \quad (21)$$

[8, p. 163 ff.]

4.3.4.2 Eigenschaften der DFT

- „Die DFT einer Linearkombination von Signalen ist gleich der Linearkombination ihrer DFTs.“ [8, p. 169]
- „Die DFT und die IDFT sind N -periodisch.“ [8, p. 169]
- „Die Energie des Signals im Zeitbereich ist gleich der Energie des Signals im Frequenzbereich geteilt durch N .“ (Parseval-Theorem) [8, p. 170]
- „Die DFT eines reellen Signals ist bezüglich dem Punkt $k = N/2$ symmetrisch.“ [8, p. 170] ✓

4.3.4.3 DFT als Annäherung der Fourier Transformation

Vorausgesetzt ein zeitkontinuierliches Signal $x(t)$ ist beschränkt auf ein Intervall der Dauer T_0 , dann lässt sich für dieses an den diskreten Frequenzpunkten $f_k = \frac{f_s}{N}$ durch die DFT folgendermaßen annähern:

$$X(f)|_{f=k\frac{f_s}{N}} \approx TX[k], \quad k = \begin{cases} -\frac{N}{2}, \dots, -1, 0, 1, \dots, \frac{N}{2}-1 & : N \text{ gerade} \\ -\frac{N-1}{2}, \dots, -1, 0, 1, \dots, \frac{N-1}{2} & : N \text{ ungerade} \end{cases} \quad (29)$$

Die Messdauer NT muss hierbei größer oder gleich der Signaldauer T_0 sein. Sollte das Messfenster länger sein als die Signaldauer, so wird der Signalvektor mit Nullen ergänzt, bis er die Länge N hat. Dadurch wird ebenfalls eine bessere graphische Auflösung erzielt. Durch Verkleinerung des Abtastintervalls $T = \frac{1}{f_s}$ wird die physikalische Auflösung des Spektrums verbessert. Proportional zur Verkleinerung der Abtastintervalls wird auch der Approximationfehler, welcher durch die Bandüberlappung entsteht.

[8, p. 171]

4.3.5 Fast Fourier Transform

Gegensätzlich zur populären Meinung ist die FFT selbst keine Transformedierte, sondern eine Methode zur effizienten Berechnung der DFT. Betrachtet man die Definition der DFT:

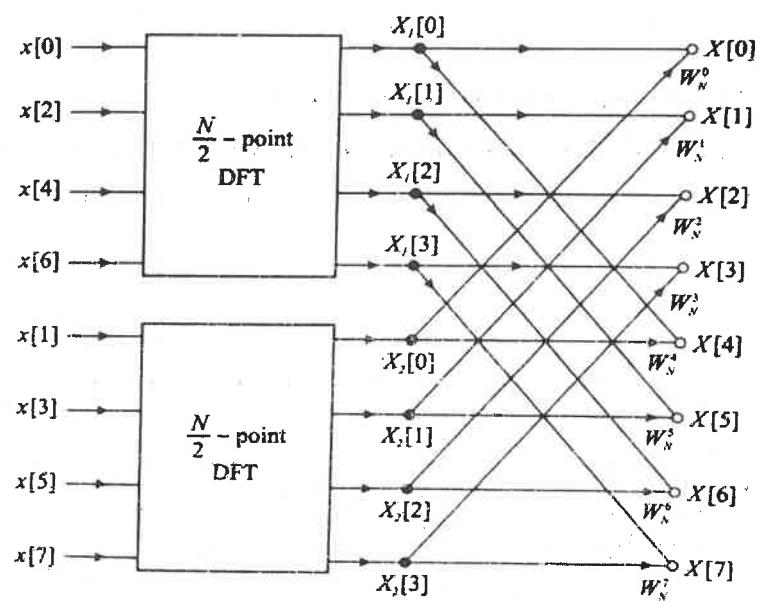


Abbildung 32: Aufspaltung 8-Punkte-DFT

Die Vorgehensweise der FFT ist somit eklatant: Die beiden $N/2$ -Punkte-DFTs werden nun fortlaufend immer weiter zerlegt (in diesem Fall in je zwei $N/4$ -Punkte-DFTs usw.) bis nur noch 2-Punkte-DFTs übrig sind. Das setzt jedoch voraus, dass es sich bei N um eine Zweierpotenz handelt. Die Anzahl der benötigten Zerlegungsschritte beträgt somit $q = \log_2(N)$. Im folgenden Bild wird die weitere Zerlegung der 8-Punkte-DFT dargestellt:



Zur Berechnung eines solchen Graphen werden je zwei komplexe Multiplikationen und Additionen benötigt. Dies resultiert in einem Rechenaufwand von N Operationen pro Zerlegungsebene.

Mit q Zerlegungsebenen ergibt sich ein Rechenaufwand von

$$N_q = N * \log_2(N). \quad (34)$$

Bedenkt man hierbei, dass zur Berechnung der regulären DFT N^2 Operationen benötigt werden, so stellt dies eine massive Aufwandsersenkung dar.

Berücksichtigt man noch die Beziehung

$$W_N^{r+N/2} = W_N^{N/2} * W_N^r = -W_N^r, \quad (35)$$

lässt sich der Graph noch weiter vereinfachen, so, dass nur noch **eine** Multiplikation pro Butterfly erforderlich ist:

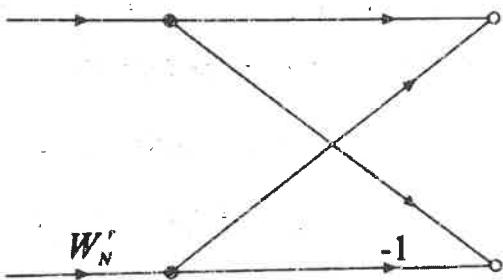


Abbildung 35: vereinfachter Butterfly

Alle oben beschriebenen Maßnahmen machen die FFT dermaßen effizient, dass bei ihrer Verwendung, bei beispielsweise einer 1024-Punkte-DFT, über 99% der Rechenoperationen eingespart werden können. [8, p. 174 ff.]

4.3.6.1 Erklärung der FFT anhand eines Testprogramms

```
#define NFFT 8192
#define F_ABТ 44100 ✓
```

Codesegment 1: Macros

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

```
#include <stdio.h>
#include <stdlib.h>
#include "esp_log.h"
#include "driver/adc.h"
#include "esp_adc_cal.h"
#include <math.h>
#include "fft.h"
#include "processed-data.h" //enthält test_buffer
```

Codesegment 2: Includes

Codesegment 2 importiert sämtliche benötigte Bibliotheken.

Der Header „processed-Data.h“ enthält die vorbereiteten Testdaten. (siehe Datenauszug 2) ✓ *Weiterer Hinweis auf Seite*

```
float fft_buffer[NFFT];
float magnitude[NFFT / 2];
float frequency[NFFT / 2];
float keyNR[NFFT / 2];
float ratio = (float)F_ABТ / (float)NFFT;
```

Codesegment 3: Array Initialisierung

Alle benötigten Buffer und Variablen werden in Codesegment 3 initialisiert.

Das Array „fft_buffer“ enthält nach dem Ausführen der FFT die Ergebnisse.

Die drei darauffolgenden Arrays beinhalten jeweils die Magnitude, die Frequenz und die am Piano korrespondierende Tastennummer. Die Variable „ratio“ beinhaltet die Auflösung des Amplitudenspektrums in Hertz.

```
float getMaxMag();
float max;
```

Codesegment 4: Deklaration von max

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

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

$$keyNR = \log\left(\frac{f}{440Hz}\right) * 12 + 49 \quad (36)$$

verwendet. Die 440Hz sind der Kammerton A. Hierbei ist zu beachten, dass, um die Midi-Notennummer zu erhalten noch 20 dazu addiert werden müssen, das Midi-Format besitzt 128 mögliche Notennummern, von denen die ersten 20 nicht belegt sind. [9]

```
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]

Codesegment 9: Ausgabe
```

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 Matlab zu Überprüfen. Im endgültigen Programm sind diese nicht mehr vorhanden da sie rein zum Debuggen dienen.

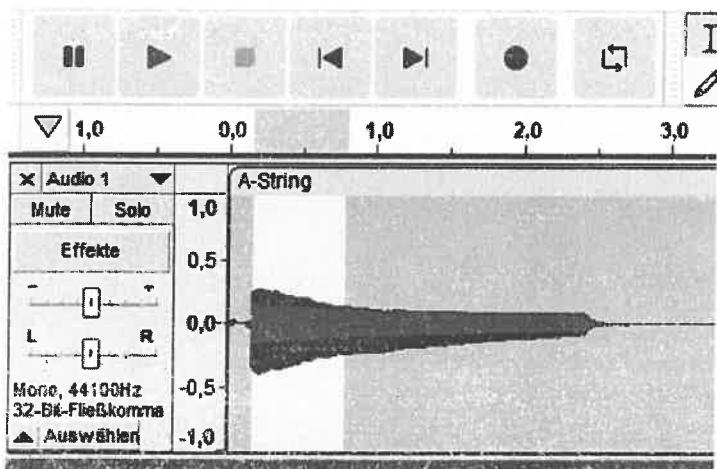


Abbildung 38: Auswahl eines Signalabschnitts

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

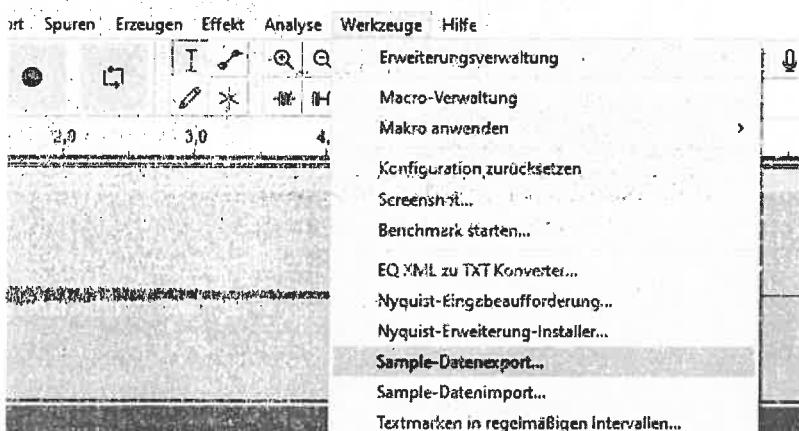


Abbildung 39: Audacity Tools

Daraufhin öffnet sich das folgende Fenster.

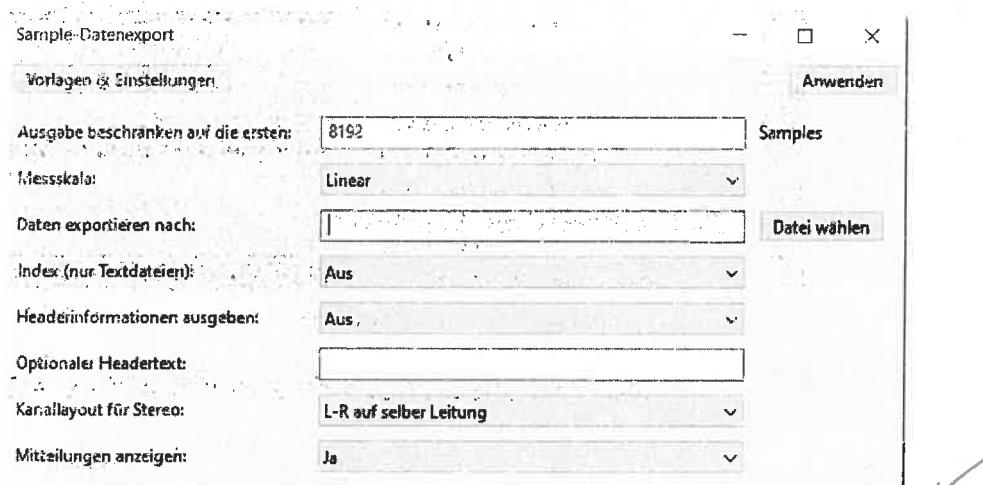


Abbildung 40: Sample-Datenexport Fenster

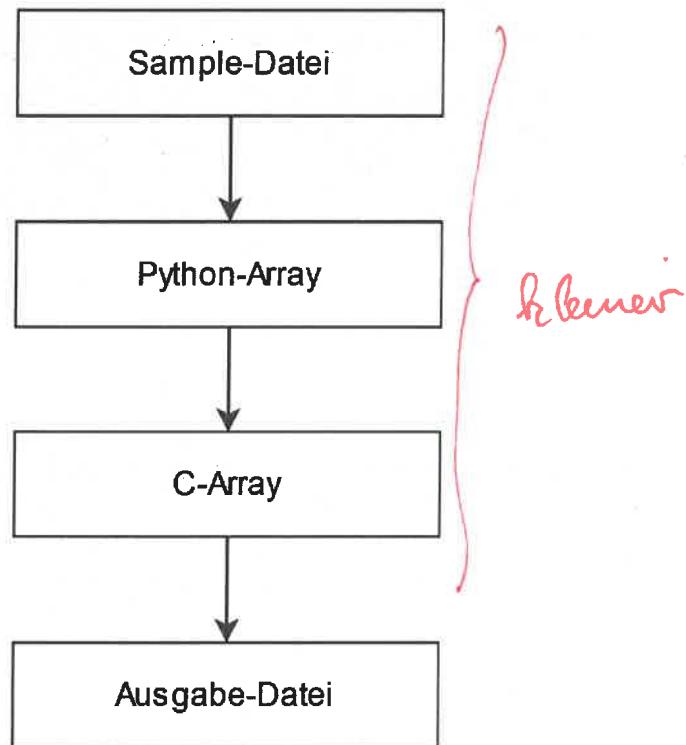


Abbildung 41: Dataflow Python Tool

Die Funktionsweise des Tools ist im Folgenden beschrieben.

```
import shutil  
Codesegment 11: Import des "shutil" Moduls
```

In Codesegment 11 wird Pythons „shutil“-Modul importiert, welches benötigt wird, um mit externen Dateien zu arbeiten.

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

Codesegment 12: Variablen Initialisierung

Ein Array mit der Größe 8192 wird in Codesegment 12 erstellt. In dieses werden die Werte aus der „.txt“ Datei gespeichert. Die Variable i dient hier als Zähler.

```
shutil.copyfile('processed-data.h',
'/home/laurenz/Dokumente/GitHub/MTAP-MIDI-Guitar-
Converter/firmware/ESP_DSP/src/processed-data.h')
Codesegment 16: Kopieren der Datei
```

In Codesegment 16 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

4.3.7 Ausgabe der Testdaten

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

Datenauszug 2: Output des Testprogramms

Die Richtigkeit des Outputs wird mit Audacity und Matlab überprüft.

4.3.7.1 Audacity - *Bessere Üb. finden*

In Audacity muss in der Menüleiste unter Analyse, Spektrum zeichnen gewählt werden.

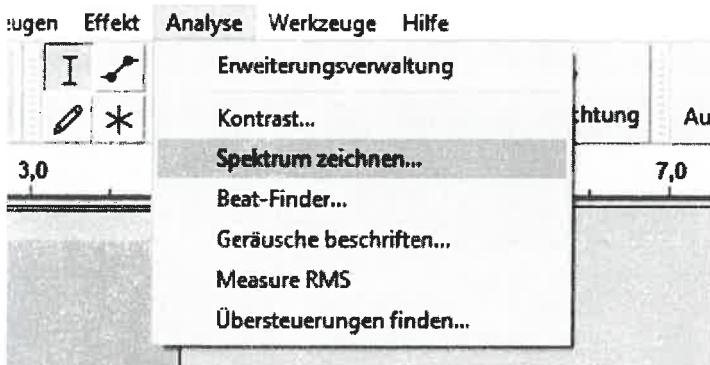


Abbildung 42: Audacity Analyse Tools

Im dadurch geöffneten Fenster kann nun das Frequenzspektrum untersucht werden.

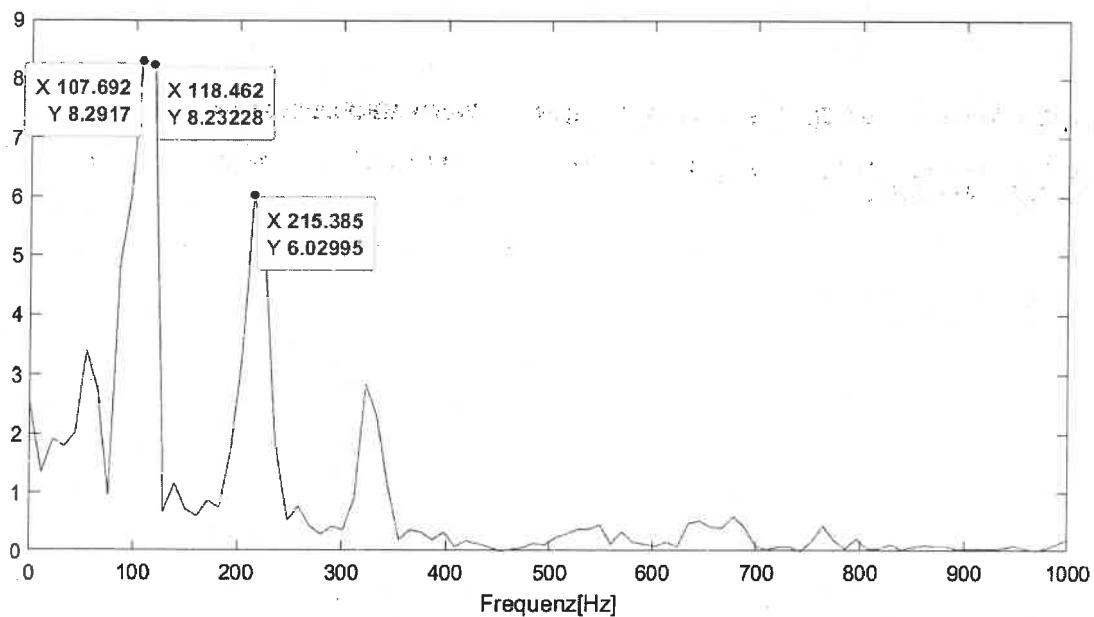


Abbildung 45: Matlab FFT

Abbildung 34 zeigt den Plot der Matlab FFT. In diesem wurden die 3 Frequenzen, welche das Testprogramm ausgibt, markiert. Es fällt auf, dass bei den zwei rechteren Punkten eine Abweichung von circa 5 Hz zum Testprogramm besteht. Das liegt daran, dass das Testprogramm nur einen kleinen Teil (8192 Datenpunkte) analysiert, Matlab aber die komplette „.wav“ Datei. Außerdem treten in C durch die verschiedenen Datentypen immer wieder Rundungsfehler auf. Da die Frequenzen trotz der Abweichung zu den gleichen Noten korrespondieren, kann dieser Fehler vernachlässigt werden. ✓

4.5 ESP32

Der ESP32 ist ein Leistungsstarkes System on a Chip (SoC) mit einem Dual-Core 32-bit Xtensa LX6 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. [11] [12]

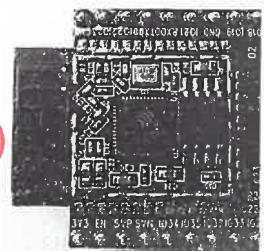


Abbildung 46: ESP32 Prozessor (Mitte), PIF WLAN-Antenne (links)



Abbildung 47: ESP-WROOM mit IPX-Connector

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-Antenne³ oder IPX⁴ Connector, bestimmt ist. Eine vom Footprint unabhängige Kenngröße, ist die Größe des Flash-Memory.

4.5.1 Bootloader Brennen mit dem ESP-Tool

Um den ESP32 programmieren zu können, muss zuerst die Firmware (ESP-AT) auf den Chip heruntergeladen werden. Dafür kommt das ESP Flash Download Tool⁵ von Espressif⁶ zum Einsatz. Dieses Tool wird dafür benötigt, die Binary-Files der Firmware über USB auf den SPI-Flash zu spielen. Im Nächsten muss zuerst das Tool heruntergeladen, entpackt und die .exe ausgeführt werden. [13]

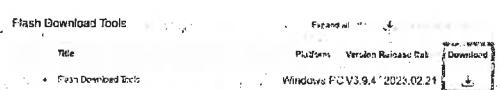


Abbildung 49: File Folder des heruntergeladenen Ordner

³ Planar Inverted F-Shaped Antenna

⁴ Koaxial-Steckverbinder

⁵ <https://www.espressif.com/en/support/download/other-tools>

⁶ <https://www.espressif.com/en>

SPIDownload		
✓	ESP32-WROOM-32-V2.4.0.0\partition_table\partition-table.bin	... @ 0x8000
✓	ESP32-WROOM-32-V2.4.0.0\ota_data_initial.bin	... @ 0x10000
✓	ESP32-WROOM-32-V2.4.0.0\phy_multiple_init_data.bin	... @ 0xf000
✓	ESP32-WROOM-32-V2.4.0.0\bootloader\bootloader.bin	... @ 0x1000
✓	ESP32-WROOM-32-V2.4.0.0\esp-at.bin	... @ 0x100000
✓	ESP32-WROOM-32-V2.4.0.0\at_customize.bin	... @ 0x20000
✓	ESP32-WROOM-32-V2.4.0.0\customized_partitions\server_cert.bin	... @ 0x24000
✓	ESP32-WROOM-32-V2.4.0.0\customized_partitions\mqtt_key.bin	... @ 0x39000

Abbildung 52: Tool Interface mit eingetragenen Binaries

Auch die SPI Flash Konfigurationen werden aus den Flasher Arguments entsprechend übernommen: [13]

```
"flash_settings" : {
    "flash_mode": "dio",
    "flash_size": "detect",
    "flash_freq": "40m"
},
```

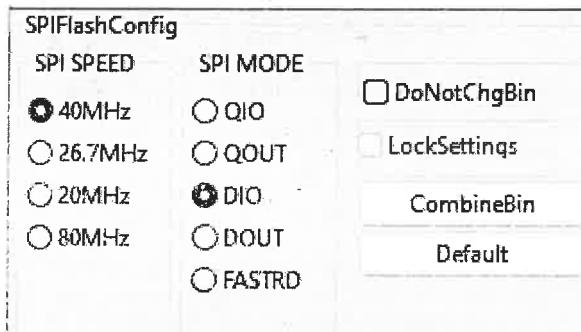
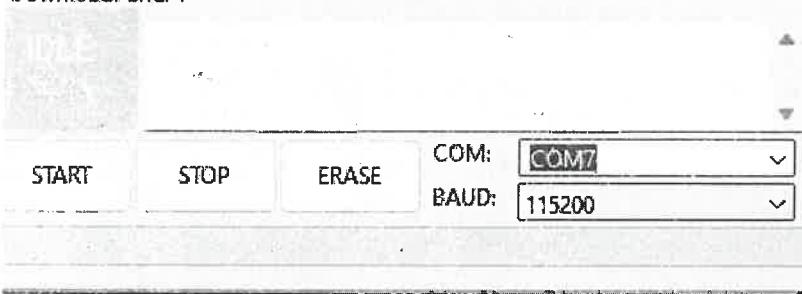
Abbildung 53: *flasher_args.json - flash_settings*

Abbildung 54: SPI Flash Config in dem GUI

Nun muss nur noch der richtige COM-Port selektiert werden und „Start“ ausgeführt werden.

DownloadPanel 1



Der Bootloader sollte nun erfolgreich auf den ESP32 gebrannt worden sein. Der Erfolg kann in PlatformIO, welches im folgenden Kapitel installiert wird validiert werden.

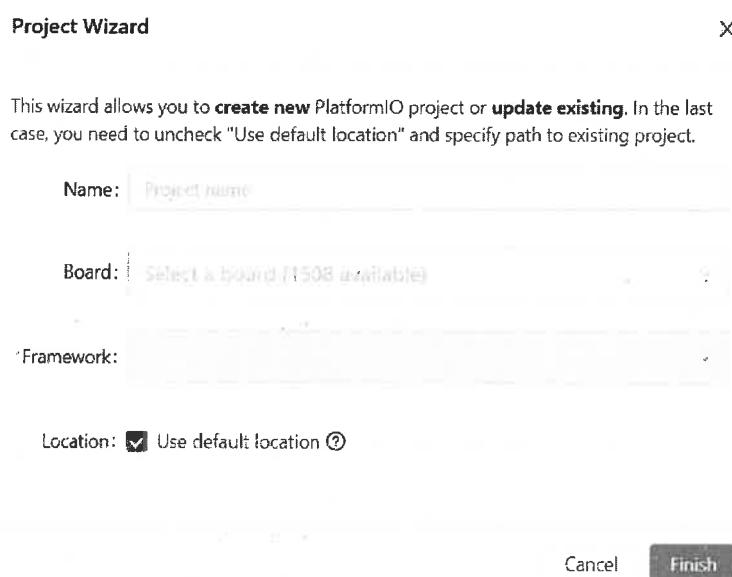


Abbildung 57: PlatformIO Project Wizard

Das Projekt wurde folgendermaßen konfiguriert:

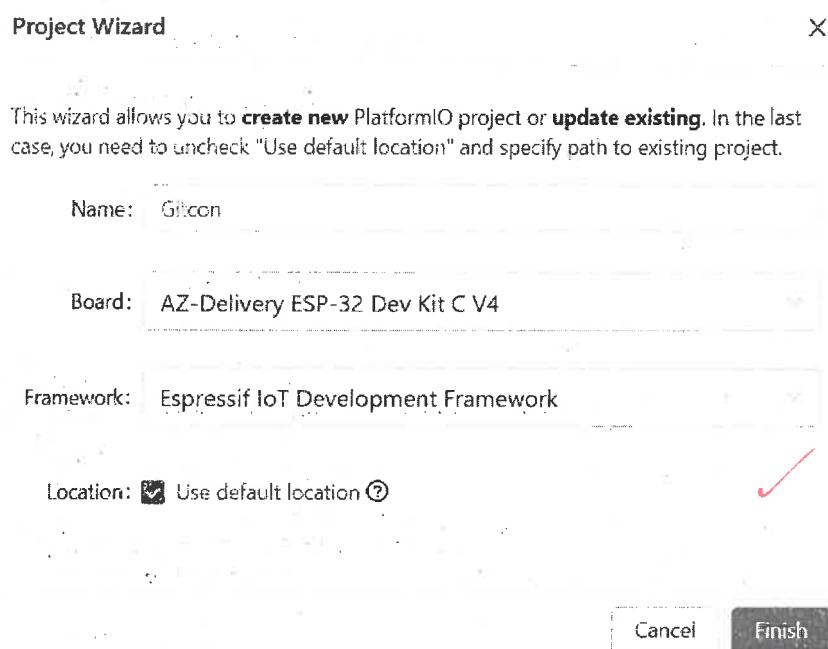


Abbildung 58: Konfiguriertes Projekt

Wird die Option „Use default location“ abgewählt, so kann ein individueller Dateipfad für das Projekt festgelegt werden. ✓

Nach einem Klick auf „Finish“ wird das Projekt erstellt. Ist der Vorgang abgeschlossen, erscheint im Filebrowser das von PlatformIO strukturierte Projekt. ✓

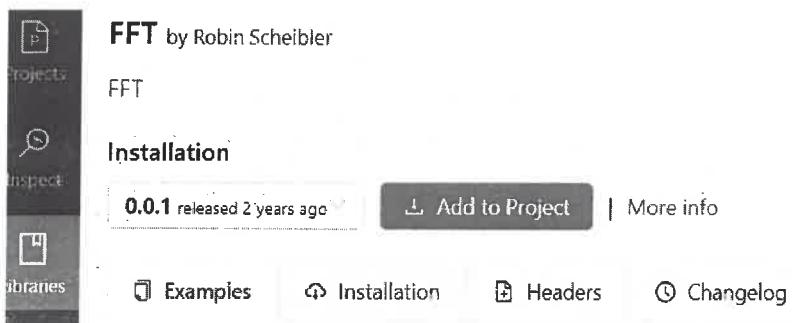


Abbildung 61: Bibliothek Untermenü

Der Knopf „Add to Project“ öffnet den Importdialog. ✓

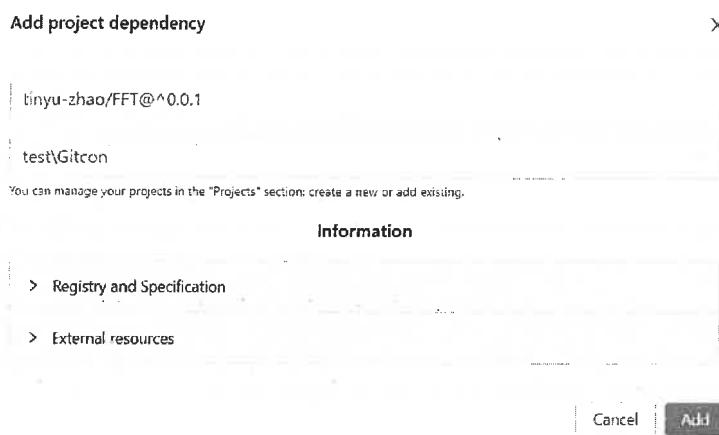


Abbildung 62: Einbinden von Dependencies

Nach dem die Schaltfläche „Add“ betätigt wurde, findet sich in der Datei „platformio.ini“ ein neuer Eintrag. ✓

```
10
11 [env:az-delivery-devkit-v4]
12 platform = espressif32
13 board = az-delivery-devkit-v4
14 framework = espidf
15 lib_deps = tinyu-zhao/FFT@^0.0.1 ✓
16
```

Abbildung 63: platformio.ini

Eine Änderung der Konfigurationsdatei wird erst mit dem nächsten „Build“ übernommen, bei welchem die Library in den „Include“ Ordner geladen wird.

Befehle wie „Build“ oder „Upload“ können im Project-Tasks Reiter ausgeführt werden. ✓

4.5.3 ESP IoT Development Framework (ESP IDF)

Das Internet of Things Development Framework (IDF) ist ein von Espressif Systems bereitgestellte Firmware, zur Entwicklung von Software auf ESP-SoCs. Das Framework ist in der Programmiersprache C verfasst und Open Source auf der ESP-IDF Projektseite dokumentiert. Mit dem IDF lassen sich mittels bereitgestellter Funktionen die Peripherien des Systems ansteuern.

4.6 Realtime Operating-System (RTOS)

Oft haben Programme mehrere dedizierte Prozesse mit striktem Ablauf. Wenn nun Echtzeitdaten vermessen werden, wie zum Beispiel Audio, müssen die Laufzeiten dieser Prozesse geregelt werden, um sich gegenseitig nicht zu blockieren, da sie sich meistens einen Kern teilen. Dafür kommt ein Echtzeitbetriebssystem zum Einsatz. Der im Gitcon implementierte FreeRTOS Kernel übernimmt die besagte Aufgabe, Tasks zu priorisieren und ermöglicht dadurch einen zeitlich reibungslosen Ablauf zu ermöglichen.

4.6.1 FreeRTOS

FreeRTOS ist ein marktführendes Echtzeitbetriebssystem, das für eingebettete Systeme und IoT-Geräte entwickelt wurde. Es bietet einen Multitasking-Kernel, der die gleichzeitige Ausführung mehrerer Tasks mit jeweils eigenen Prioritätsstufen und Laufzeitbeschränkungen ermöglicht. FreeRTOS ist Open Source unter der MIT Lizenz und kann privat, als auch kommerziell genutzt werden. [14]

FreeRTOS ist hochgradig konfigurierbar und seine Struktur betont geringen Overhead und hohe Leistung, wodurch es für Systeme mit begrenzten Ressourcen geeignet ist. Es enthält Funktionen wie Task-Scheduling, Kommunikation zwischen Tasks und Speicherverwaltung. FreeRTOS ist in C geschrieben und enthält Ports für eine Vielzahl von Mikrocontrollern und Prozessoren unter anderem für ESP32 basierte Systeme.

Schre *informativ, gut lesbar, passendes Layout.*

4.8 Musical Instrument Digital Interface (MIDI)

Das MIDI-Protokoll wurde in den frühen 80ern entwickelt und standardisiert die Kommunikation zwischen Computern und Musik-Hardware, sogenannten MIDI-Controllern. Jedes Mal, wenn eine Taste auf einem Controller gedrückt wird, erstellt dieser eine MIDI-Nachricht und sendet sie an den Computer. Diese Tasten sind nicht nur auf die Klaviatur eines MIDI-Controllers beschränkt, es können ebenfalls MIDI-Wörter gesendet werden, welche andere Parameter in einer digitalen Musikproduktionsumgebung steuern, wie zum Beispiel die Intensität eines Audioeffekts.

Beispielsweise kann die Grenzfrequenz des in Abbildung 68 gezeigten digitalen Filters mittels eines encodierten Potentiometers am MIDI-Controller gesteuert werden. ✓

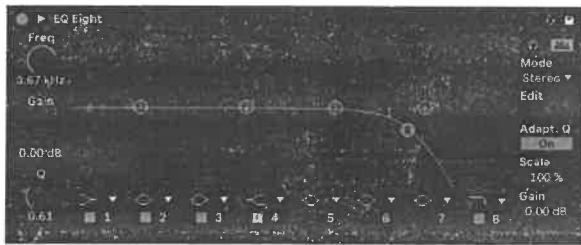


Abbildung 68: Ableton Live Effekt EQ Eight (Filter)

Es ist wichtig zu wissen, dass MIDI-Signale nichts mit niederfrequenten Audiosignalen zu tun haben. Weder analoges- noch digital aufgefasstes Audio kommt in einer MIDI-Kommunikation vor. [17]

Das MIDI-Protokoll beschreibt unter anderem die zu übertragenden Noten mit digitalen Wörtern, welche in einer genormten Tabelle festgehalten sind. Jede Note hat daher eine Adresse, durch welche dann ein anderes digitales Instrument weiß, mit welcher Tonhöhe es diese Note spielen muss. ✓

Pitch Bend	0xE	Ändert die Tonhöhe des Gesamten Kanals in feinen Schritten.
System Messages	0xF	MIDI Clock, Start, Stop, System Reset, Active Sensing

Tabelle 2: MIDI-Status

4.8.1.2 MIDI-Kanal

Mit den Vier übrigen Bit wird der Kanal eingestellt, an den die Nachricht geschickt werden soll. Ein einzelner Controller kann also bis zu 16 verschiedenen MIDI-Kanäle in einer DAW ansprechen. Auf einem MIDI-Keyboard würden zum Beispiel die Klaviertasten auf einen anderen Kanal geschickt werden als die Drumpads.

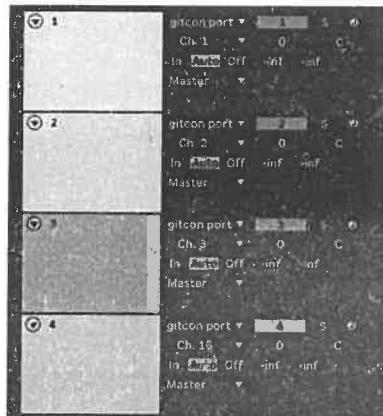
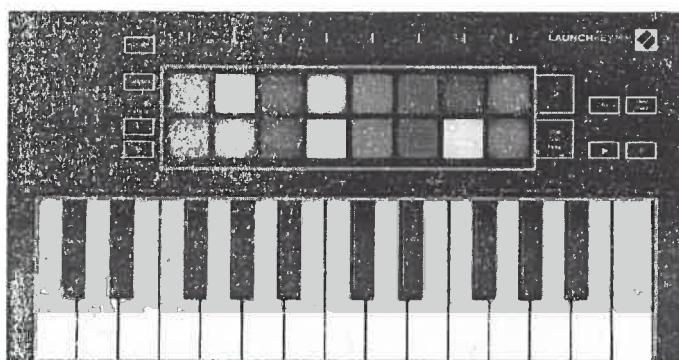


Abbildung 70: Vier Verschieden MIDI-Spuren mit dem gleichen Controller als Input

Abbildung 71: Novation⁹ Launchkey Mini

4.8.1.3 Datenbytes

Für das Projekt sind nur die Funktionen Note On und Note Off relevant, weshalb die Dokumentation der Datenbytes auf diese beiden Status beschränkt ist.

Im Datenbyte eins ist bei beiden Status die Adresse der Note gespeichert, während in Byte zwei die Anschlagstärke übertragen wird. Ungewöhnlicherweise wird auch beim Ausschalten einer Note die Anschlagstärke (Loslass-

Klaviertasten (Channel 1)

*Klingt Bend
ist aber Rauscht
Nibbles
Werte
oder so*

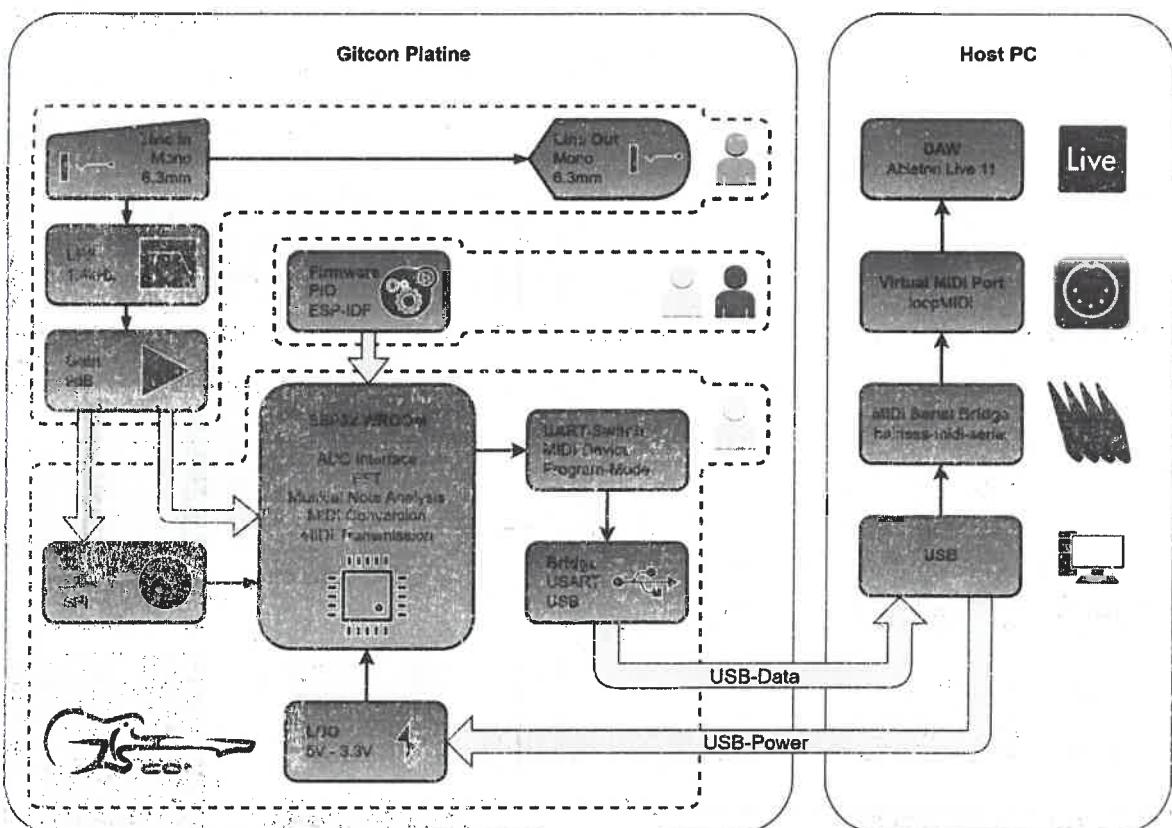
⁹ <https://novationmusic.com/de/keys/launchkey-mini>

5 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.

5.1 Blockschaltbild

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



- 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.

Sektion 3: Firmware

Sobald die Daten digitalisiert sind, analysiert die Firmware das aufbereitete Signal der E-Gitarre und erkennt die Frequenzanteile, aus welchen sich die gespielte Note ergibt. Erkannte Noten werden auf den USB-Port geschickt.

5.2 Hardware

Im folgenden Teil wird zunächst der Schaltplan des funktionalen Blockschaltbildes (Abbildung 72) dokumentiert, sowie die Wahl der Komponenten begründet. Darauffolgend werden die Layout Kriterien und Guidelines für das Leiterplatten-Design beschreiben.

5.2.1 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 heruntergeregt werden. Hierzu wird ein Low Drop-Out Regulator verwendet.

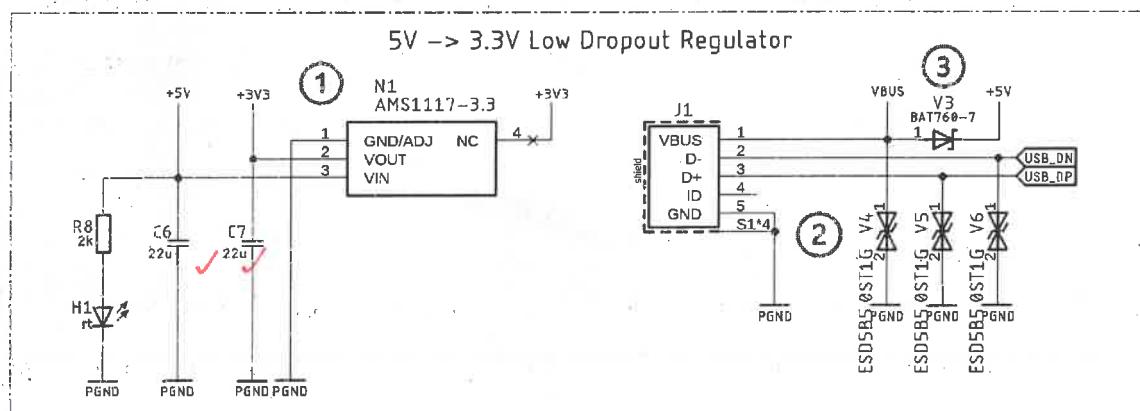


Abbildung 74: Spannungsregler und USB-Connector im Schaltplan

5.2.1.1 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 heruntergeregt.

5.2.1.2.1 Kennwerte der TVS-Diode

Uni- oder Bidirektional

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.

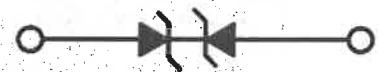


Abbildung 77: Symbol der Bipolaren TVS-Diode

Anzahl von Kanälen

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

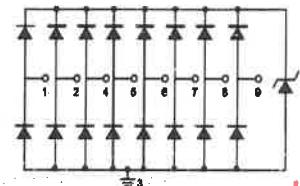


Abbildung 78: SC7538-08UTG Block Diagramm

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

das Innere
groß

5.2.2 Analog-Frontend (AFE)

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

5.2.2.1 Schaltung

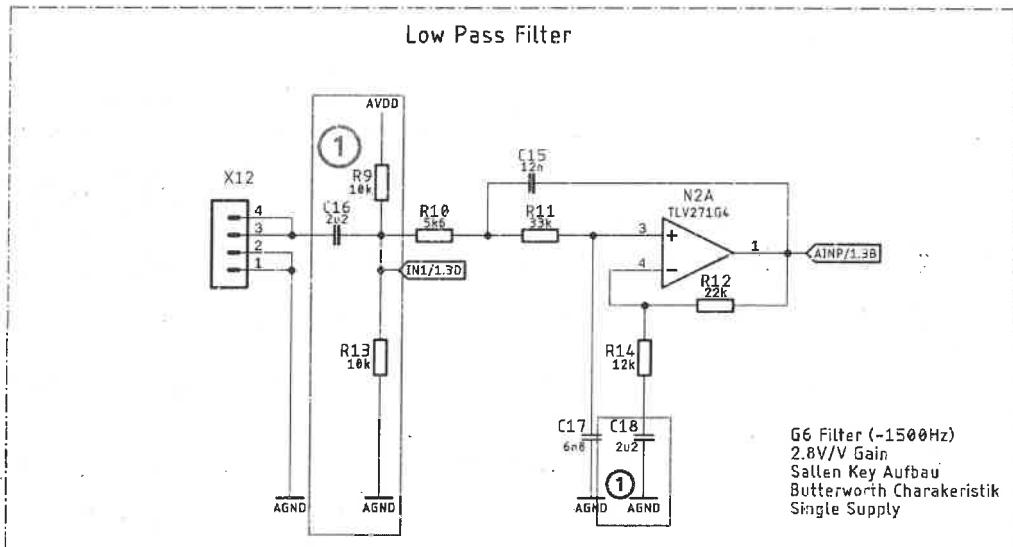


Abbildung 79: LPF-Schaltung

Als Operationsverstärker wird der OPV TLV271G4 von Texas Instrumentes verwendet, dieser ist Teil der TLV27x- Serie und wurde für Anwendungen mit niedrigem Stromverbrauch und dem Einsatz in batteriebetriebenen Geräten entwickelt. Der OPV hat einen breiten Versorgungsspannungsbereich von 2,7V bis 12V und kann einen Ausgangsstrom von bis zu 60 mA liefern. Weiters verfügt dieser über einen hohen Eingangswiderstand, was zu einer erhöhten Empfindlichkeit gegenüber Eingangssignalen macht.

5.2.2.1.1 Einkopplung des Audios

Da der Gitcon über keine negative Versorgung verfügt, muss das Audio zwischen Masse und +3.3 Volt eingekoppelt werden. Dafür sorgen die Widerstände R_9 , R_{13} und der Kondensator C_{16} . Wobei ein Kondensator nur Wechselspannungen durchlässt und Gleichspannungen herausgefiltert. Eine erfolgreiche Einkopplung wird erreicht, indem der Kondensator die Spannung auf das Potential zwischen R_9 und R_{13} anhebt bzw. senkt.



(\therefore deshalb)

$$\therefore C_{1n} = \sqrt{2} = 1,414 F$$

$$C_{2n} = \frac{1}{C_{1n}} = 0,707 F$$

Da die Widerstandswerte sehr klein und die Kapazitäten unrealistisch sind müssen diese skaliert werden. Wenn man m als Skalierungsfaktor bezeichnet, werden die Widerstände m-mal erhöht und die Kapazitäten um 1/m verringert, um dieselbe Grenzfrequenz von 1 zu behalten.

$$m = 10000$$

$$R_1 = R_{1n} * m = 10k\Omega$$

$$R_2 = R_{2n} * m = 10k\Omega \quad \checkmark$$

$$C_1 = \frac{C_{1n}}{m * \omega_0} = \frac{1,141}{10k * 2 * \pi * 10kHz} = 2,2nF$$

$$C_2 = \frac{C_{2n}}{m * \omega_0} = \frac{0,707}{10k * 2 * \pi * 10kHz} = 1,1nF$$

Dies ist eine ideale Dimensionierung, wobei beide Widerstände gleich sind. ✓

In der obigen Grafik ist die Verstärkung erkennbar, da U_a um 9dB höher als U_e ist. Weiters ist der Roll-off von -6dB gut erkennbar (siehe Abbildung 82), was einen Filter zweiter Ordnung charakterisiert.

5.2.2.5 Messtechnische Bestimmung des Frequenzgangs

Die Bestimmung des Frequenzgangs wurde im Frequenzbereich von 207Hz bis 3140Hz durchgeführt, um den Roll-off Point und die abfallende Steigung gut zu erkennen.

*→ neue Abb 83
Es sind aber... und die mit x dB/Dezade*

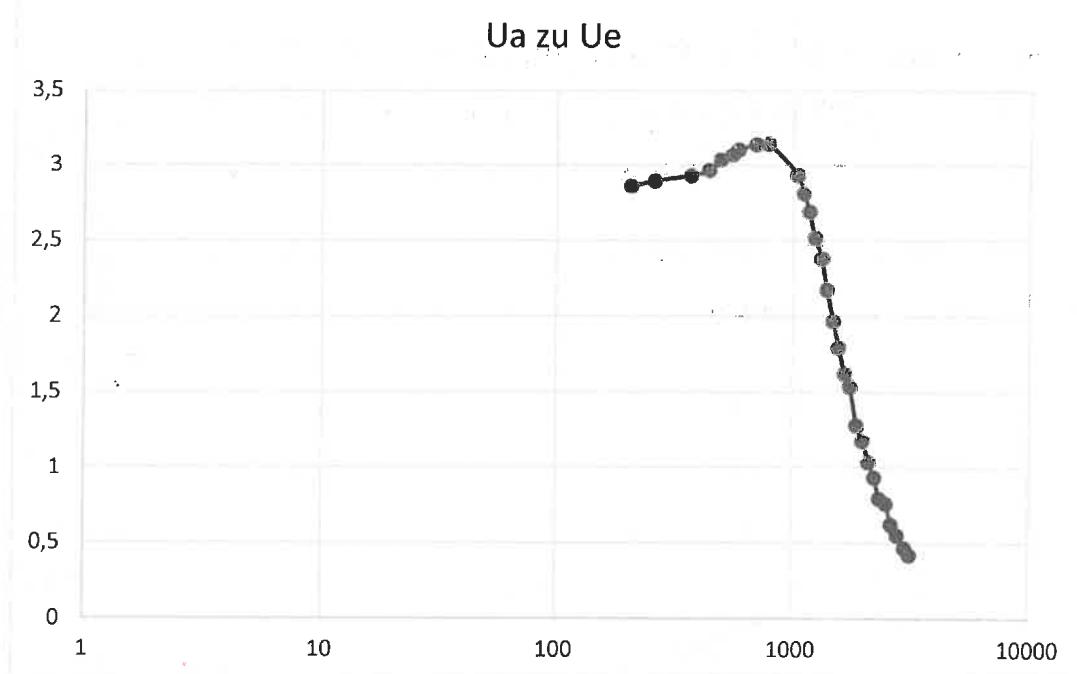


Abbildung 83: Messung des Filters

Abbildung 83 zeigt die Messung des Verhältnisses U_a/U_e , hierbei ist der Roll-off bei der Grenzfrequenz, die etwas über 1000Hz liegt, ersichtlich. Auch zu erkennen ist die kontinuierlich fallende Steigung. Die Messwerte befinden sich im Toleranzbereich der Simulation und sind daher gültig.

ggf Abweichung benennen.

In der obigen Grafik ist die Verstärkung erkennbar, da U_a um 9dB höher als U_e ist. Weiters ist der Roll-off von -6dB gut erkennbar (siehe Abbildung 82), was einen Filter zweiter Ordnung charakterisiert.

5.2.2.5 Messtechnische Bestimmung des Frequenzgangs

Die Bestimmung des Frequenzgangs wurde im Frequenzbereich von 207Hz bis 3140Hz durchgeführt, um den Roll-off Point und die abfallende Steigung gut zu erkennen.

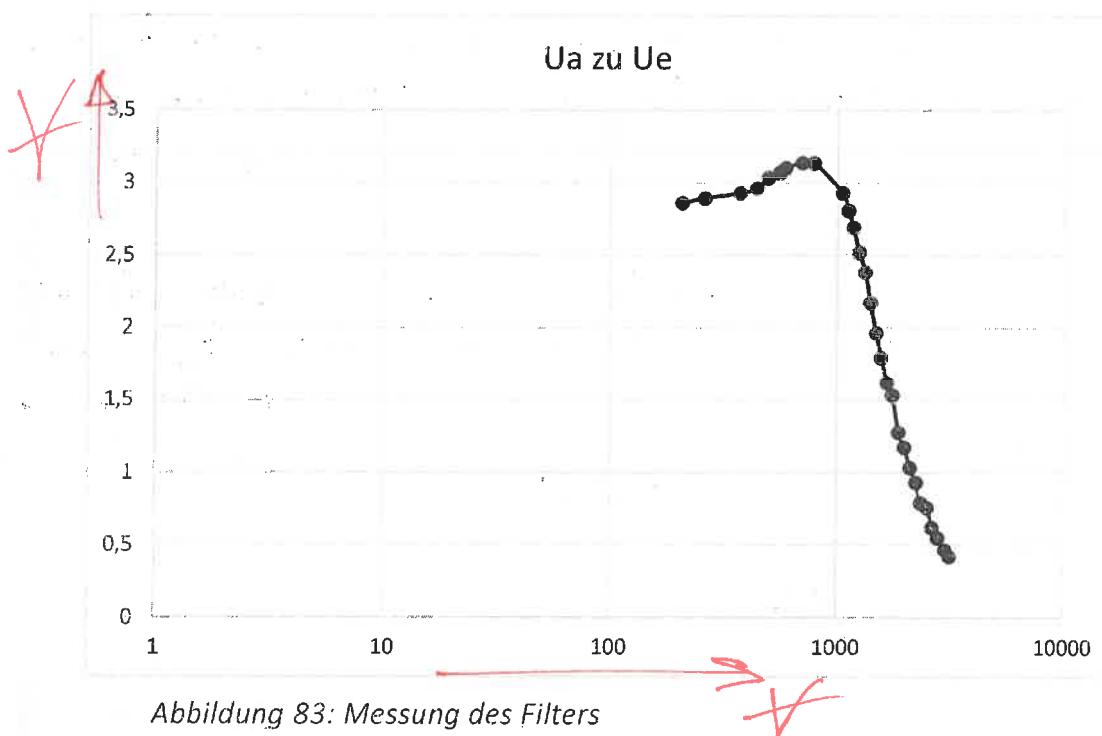


Abbildung 83 ~~Fehler! Verweisquelle konnte nicht gefunden werden.~~ zeigt die Messung des Verhältnisses U_a/U_e , hierbei ist der Roll-off bei der Grenzfrequenz, die etwas über 1000Hz liegt, ersichtlich. Auch zu erkennen ist die kontinuierlich fallende Steigung. Die Messwerte befinden sich im Toleranzbereich der Simulation und sind daher gültig.

✓ Meßschaltung

5.2.3 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 Pseudostatisches RAM (PSRAM) ✓

Tabelle 5: ESP-WROOM Spezifikationen

Da keine kabellosen Kommunikationssysteme wie WLAN oder Bluetooth benötigt werden, ist die Wahl auf ein kompaktes Modul mit IPX-Steckverbinde, anstelle einer großflächigen PIF-Antenne, gefallen. ✓

5.2.3.1 Beschaltung des *ESP-WROOM-32UE* Chips

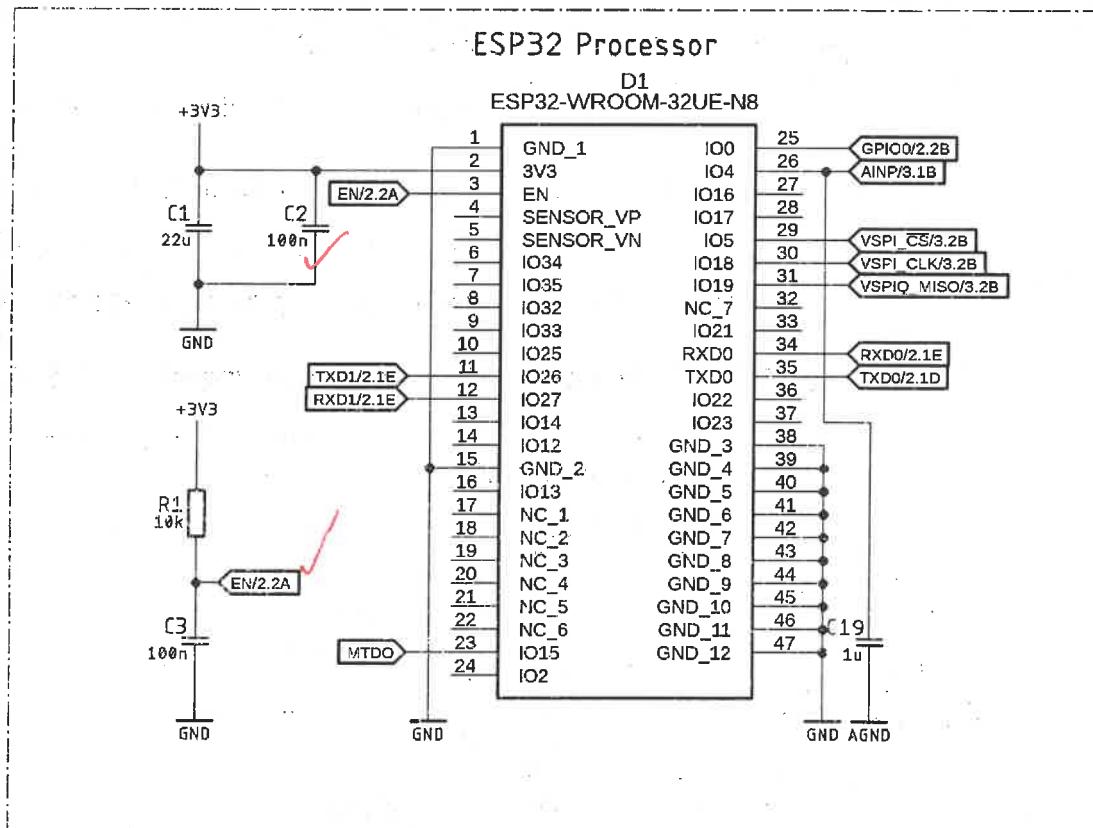


Abbildung 84: ESP32 Beschaltung



Tabelle 6: ESP32 Pin-Out

↑
Form

Um einen neuen kompilierten Code auf den ESP32 herunterzuladen, muss er sich während eines Resets im „Boot“ Modus befinden. Dieser Modus wird durch einen Taster, der, während dem Uploaden des Codes gedrückt werden muss, initialisiert.

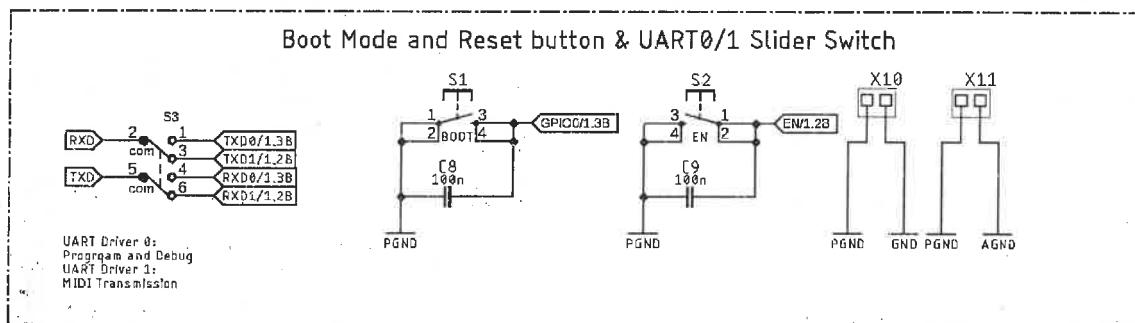


Abbildung 86: Taster und Schiebeschalter

5.2.3.2 Beschaltung der USB-Bridge

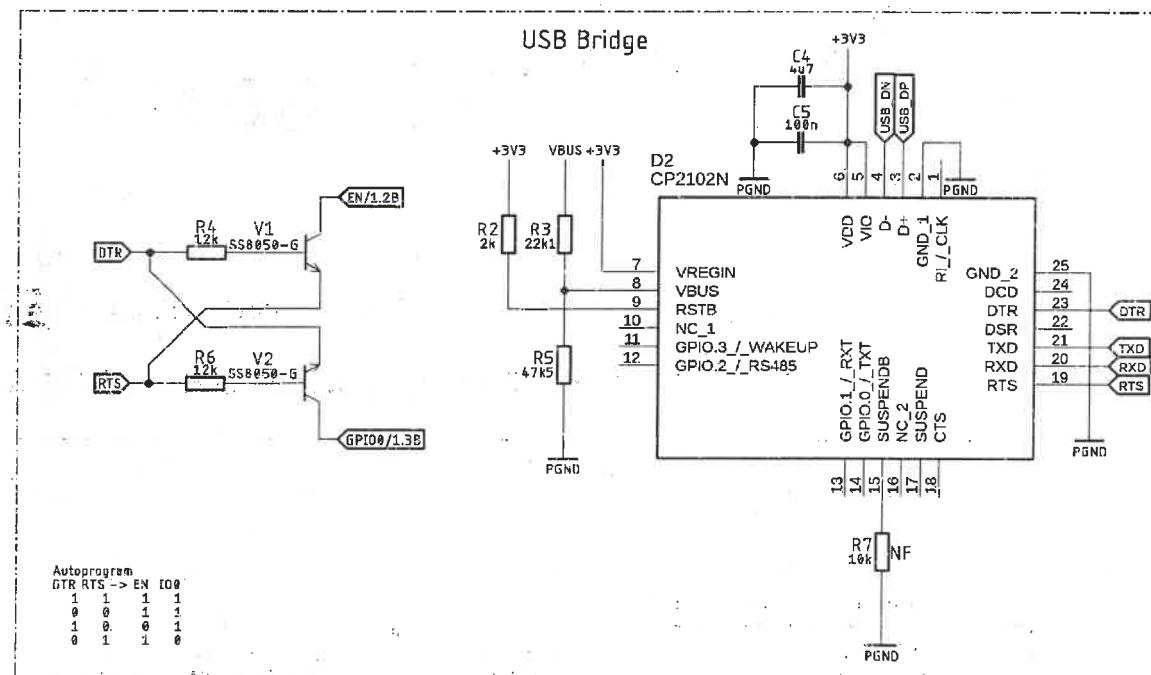


Abbildung 87: Beschaltung der USB-Bridge

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 ✓

eine solche Leiterbahn bis zu 3,8 Ampere (mit einem Temperaturanstieg um ca. 30°C) standhalten.

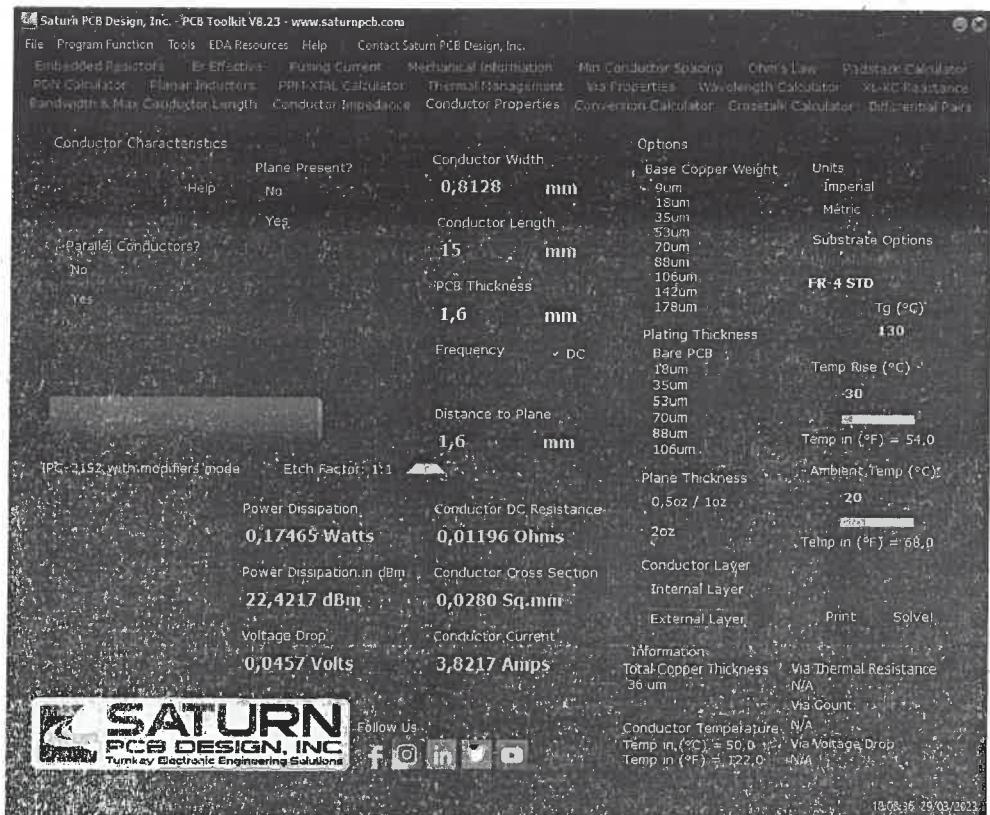


Abbildung 88: Berechnung des maximalen Stromes im PCB-Toolkit Rechner

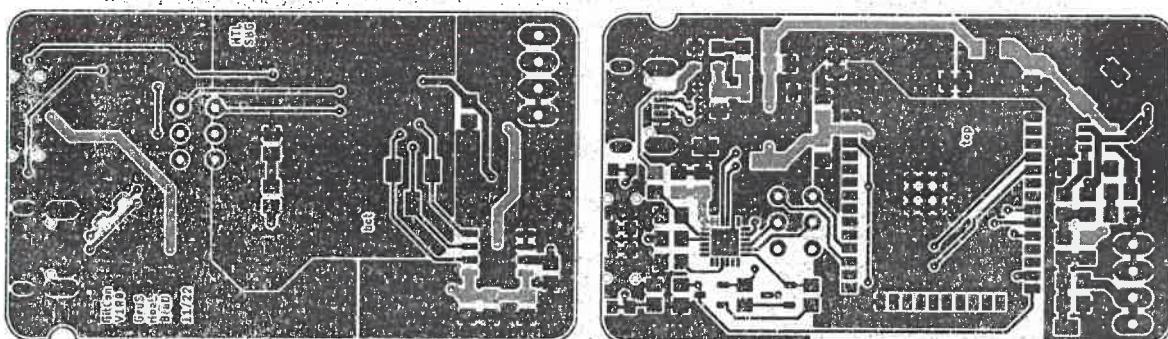


Abbildung 89: Versorgungsleitungen (5V rot, 3V3 orange)

Die Versorgung der USB-Bridge, des ESP32 und der Analog-Sektion sind Sternpunkt-formig 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.

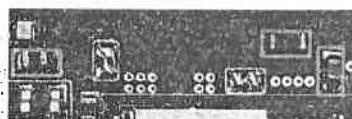


Abbildung 90: Komponenten des Pi-Filters

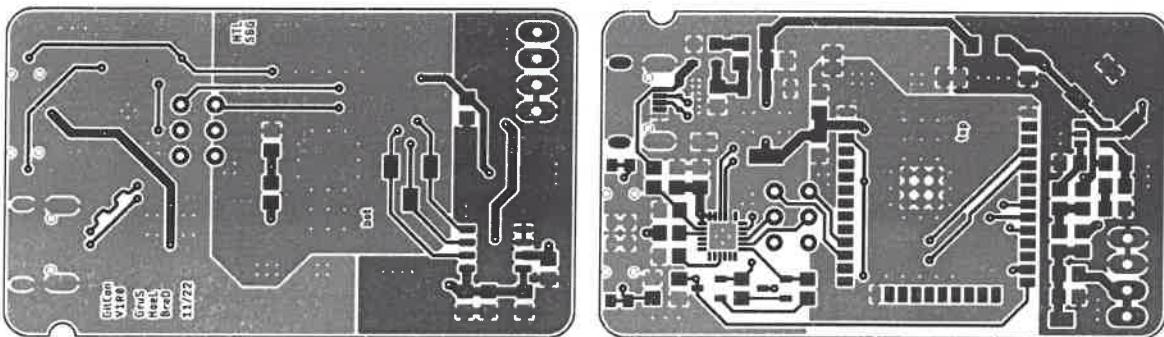


Abbildung 93: Bottom (links) und Top (rechts) Masse-Layer

PGND: Leistungsmasse (Orange)

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

GND: Digitalmasse (Blau)

Digitale 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 elektromagnetische Feld, Störungen verursachen (Crosstalk¹¹). Diesen Effekt bezeichnet man als kapazitive Kopplung.

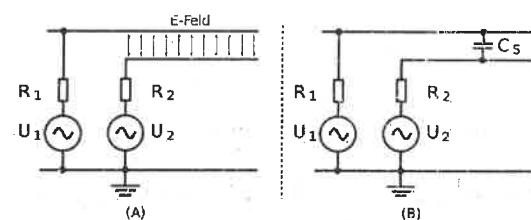


Abbildung 94: kapazitive Kopplung

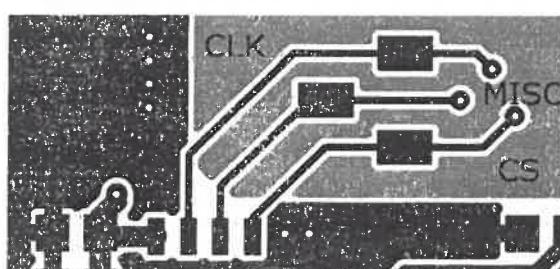


Abbildung 95: Massefläche zwischen HF-Leitungen

Um dieser Kopplung zwischen den Leiterbahnen entgegenzuwirken, muss zwischen besagten Leitungen eine Massefläche ausgeprägt sein. Da diese Störungen nun von der Masse absorbiert werden, muss die Fläche vom Rest isoliert werden. Dies wird

¹¹ Ungewollte Übertragung von Signalen auf eine andere Leitung durch Kopplungsarten

5.2.4.4 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 97:
Masse unter Au-
dio-Sektion

5.2.4.5 Differentielles Paar

Die Datenleitungen eines USB sind ein differentielles Paar und müssen auf der Platine auch als solches geroutet werden. Datenübertragungen mit einem Differentiellen Paar unterscheiden sich von normalen Leitungen, da sie kein Bezugspotential (Masse) haben, sondern zwei Leitungen mit demselben Datensignal, wobei eines der beiden invertiert ist. Die Impedanz des Differentiellen Paars muss angepasst werden, da sonst die maximale Übertragungsrate eingeschränkt werden könnte. Die Leitungen sind angepasst, wenn beide Leitungen gleich lang geroutet sind und dieselbe Impedanz aufweisen. [21] ✓

5.2.4.6 Abmessung und Kompakter Footprint U

Die Platine verfügt über eine Einkerbung, mit welcher sich die Platine mit einer M3 Schraube an ein Gehäuse befestigen lässt.

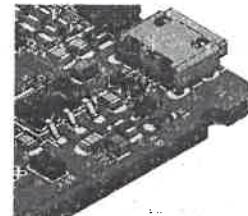


Abbildung 98: Einker-
bung zur Befestigung

5.2.4.7 SMD-Testterminals

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

X hier fehlt noch Text

Drei vom RTOS geregelte Tasks formen die Prozesskette. Mittels Queues (First In First Out, FIFO) werden die Daten in den nächsten Task übergeben. Von der zentralen Datenstruktur „Driver Parameter Handler“ kann von allen Tasks aus auf die Peripherien und FIFOs referenziert werden.

5.3.2 Doxygen

Der Code wurde mittels sogenannter ungarischer Notation versehen. Ungarische Notation kommt in der Firmware in Form von dokumentativen Kommentaren vor, welche anschließend von dem Programm **Doxygen** erkannt werden und daraus Referenzfiles generiert, welche hilfreich für die Dokumentation des Codes sind. Alle im Quellverzeichnis vorkommenden Dateien werden indiziert und es wird ein Inhaltsverzeichnis daraus erstellt. Weiters werden die verwendeten Datentypen aufgelistet und mit einer Beschreibung versehen, mit welchen der Code inline dokumentiert ist. In Summe ergibt sich daraus das Referenzhandbuch für die Firmware mit folgender Struktur:

1. Data Structure Index
 - Inhaltsverzeichnis für Typedefs und Datenstrukturen
2. File Index
 - Inhaltsverzeichnis für Header- und Quelldateien
3. Data Structure Documentation
 - Beschreibung der für Typedefs und Datenstrukturen
4. File Documentation
 - Beschreibung der Methoden und Algorithmen

Das Handbuch der Firmware dieses Projekts lässt sich im Anhang finden.

ist nicht in der DA enthalten und wird separat zur Laf. geliefert.

5.4 Software

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

5.4.1 Virtueller MIDI-Port

Originale MIDI-Controller verfügen über zwei DIN 5-Pin 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.



Abbildung 100: MIDI-Port eines klassischen Controllers

Hierfür kommt die Software **loopMIDI** 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. Um einen Port zu erzeugen, gibt man den Namen in das Textfeld ein und klickt auf die Plus (+) Schaltfläche.

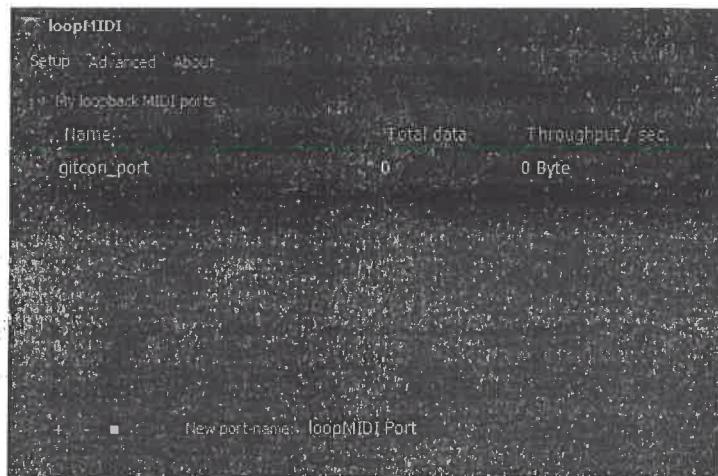


Abbildung 101: Loop MIDI GUI

5.4.3 Ableton Live Setup

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 kann im Reiter **MIDI** selektiert werden.

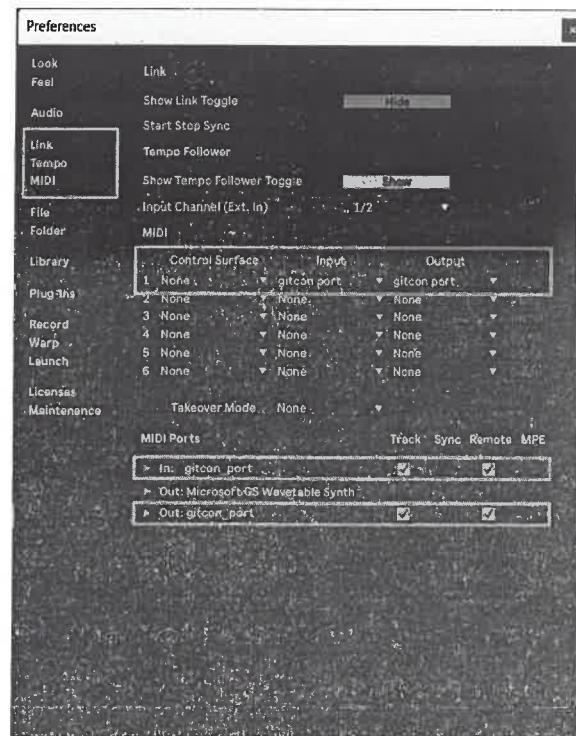


Abbildung 104: Live Preferences

Eingehende MIDI-Signale können nun von Ableton Live in einen MIDI-Kanal geroutet werden, in welchem dann die Signale zur Verfügung stehen. Das Gitcon Device schickt die MIDI-Noten auf den Kanal 1. In der MIDI-Spur muss also der entsprechende Kanal ausgewählt werden. Um nun die eintreffenden MIDI-Noten aufzunehmen, muss die MIDI-Spur aufnahmen bereit geschalten werden (Arming). Eine Spur ist für die Aufnahme scharfgestellt, wenn man die „Arm“-Schaltfläche anklickt, sodass sie rot aufleuchtet.



Abbildung 105: MIDI-Spur Konfiguration

aufnahmebereit?

6 Fehlererfassung

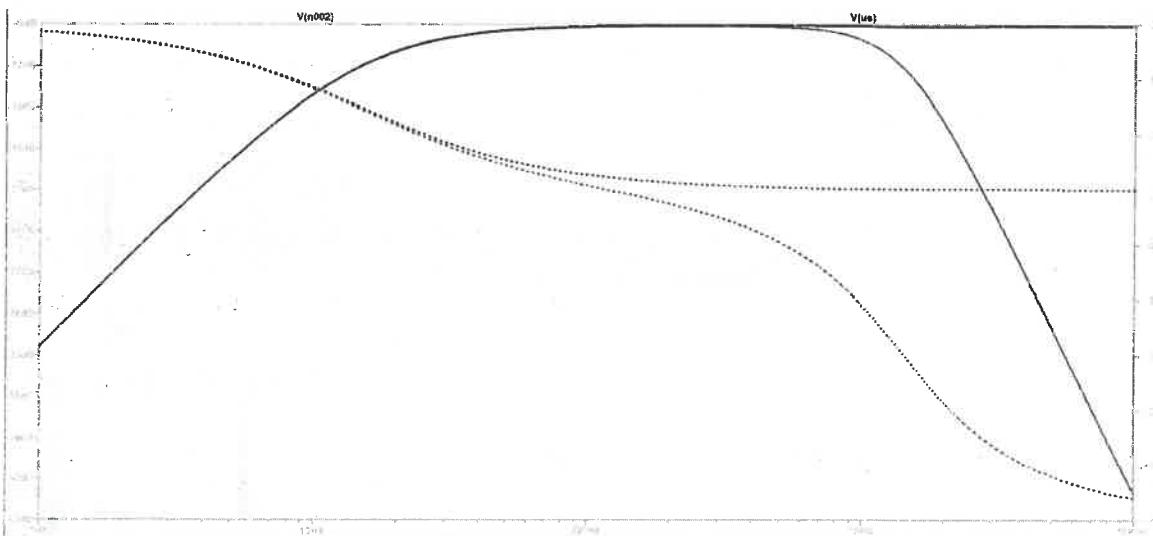
6.1 Platine

- Taster Footprint ist Falsch
- Teil des Doku-Layers wurde nicht auf den Silkscreen gedruckt.

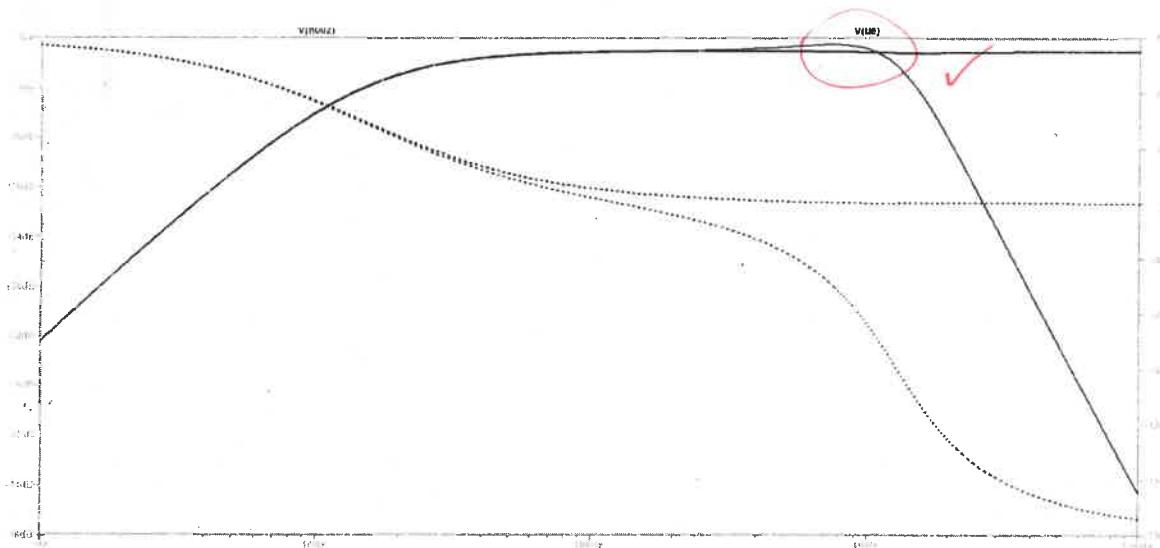
6.2 Bestückung

- 10nF statt 12nF bei Tiefpass verfügbar

Simulation mit 10nF (Ist)



Simulation mit 12nF (Soll)



7 Glossar

ADC	Analog-Digital-Converter
AFE.....	Analogue-Frontend
CAD	Computer Aided Design
CPU	Central Processing Unit
DAW	Digital Audio Workstation
DFT.....	Diskrete Fourier Transformation
DSP	Digital Signal Processing
DSV	Digitale Signalverarbeitung
ÉKG	Elektrokardiogramm
ESD	Electrostatic Discharge
ESP	Kürzel für Espressif Produkte
FFT	Fast Fourier Transform
FIFOs	First In First Out (Queue)
Gitcon	Guitar Converter
GND	Ground (Masse)
GPIO	General Purpose Input/Output
HF	Hoch Frequent
I ₂ C	Inter Integrated Circuit
I ₂ S	Inter Integrated Sound
IC	Integrated Circuit
IDF	IoT Development Framework
IoT	Internet of Things
LDO	Low Dropout Regulator
LED	Light Emitting Diode
MIDI	Musical Instrument Device Interface
MQTT	Message Queuing Telemetry Transport
OPV	Operationsverstärker
PC	Personal Computer
PCB	Printed Circuit Board
Pickup	Tonabnehmer
PIF	Planar Inverted F-Shaped
PIO	PlatformIO
PSRAM	Pseudostatisches RAM
RAM	Random Access Memory (Volatile Speicher)
RTOS	Realtime Operating-System
SDK	Software Development Kit
SMD	Surface Mount Device
SNR	Signal to Noise Ratio
SoC	System on a Chip
TI	Texas Instruments (Firma)
TVS	Transient Voltage Supresion
UART	Universal Asynchronous Recieve and Transmit
USB	Universal Serial Bus

8 Abbildungsverzeichnis

Abbildung 1: Gitcon Logo	4
Abbildung 2: MIDI-Klavier (Novation Launchkey)	13
Abbildung 3: Konfiguration einer MIDI-Spur beim MIDI-Unitest	21
Abbildung 4: Starten der Aufnahme des MIDI-Unitests	21
Abbildung 5: Ergebnis einer Aufnahme des MIDI-Unitests	21
Abbildung 6: Konfiguration einer MIDI-Spur beim Live-Konversion Test.	22
Abbildung 7: Starten der Aufnahme des Live-Konversionstest	22
Abbildung 8: Ergebnis der Live-Konversion	23
Abbildung 9: Trello Liste mit zwei Karten	29
Abbildung 10: Copilot Logo	29
Abbildung 11: passiver LCR-Tiefpass [35]	30
Abbildung 12: aktiver Tiefpass [36]	31
Abbildung 13: OPV-Schalsymbol	33
Abbildung 14: Operationsverstärker	33
Abbildung 15: Schaltbild OPV	34
Abbildung 16: Differenzenverstärkung OPV	35
Abbildung 17: OPV als Impedanzwandler	35
Abbildung 18: Invertierender OPV	36
Abbildung 19: Nicht-invertierender OPV	37
Abbildung 20: Sallen-Key Tiefpass (links) & Hochpass (rechts)	40
Abbildung 21: Filter erster Ordnung	40
Abbildung 22: Filter zweiter Ordnung	40
Abbildung 23: Flankensteilheit der Ordnungen [24]	41
Abbildung 24: Butterworth Frequenzverhalten [33]	42
Abbildung 25: Frequenzverhalten der Filtercharakteristiken [32]	43
Abbildung 26: Toleranzschema [31]	43
Abbildung 27: Schaltbild Filter zweiter Ordnung	44
Abbildung 28: Schaltbild Sallen-Key zweiter Ordnung	46
Abbildung 29: Dämpfungskurven	48
Abbildung 30: Aufbau einer E-Gitarre	49
Abbildung 31: Veranschaulichung Fourier Transformation	50

Abbildung 64: Project-Tasks Panel	80
Abbildung 65: Statusleiste	80
Abbildung 66: UART-Verbindung.....	82
Abbildung 67: UART-Datenpaket.....	82
Abbildung 68: Ableton Live Effekt EQ Eight (Filter).....	83
Abbildung 69: MIDI-Word	84
Abbildung 70: Vier Verschieden MIDI-Spuren mit dem gleichen Controller als Input.....	85
Abbildung 71: Novation Launchkey Mini.....	85
Abbildung 72: Funktionsblockdiagramm des Gitcon	87
Abbildung 73: Legende der Arbeitsunterteilung.....	88
Abbildung 74: Spannungsregler und USB-Connector im Schaltplan	89
Abbildung 75: ESD-Warnhinweis.....	90
Abbildung 76: ESD-Schutz in Schaltungen	90
Abbildung 77: Symbol der Bipolaren TVS-Diode.....	91
Abbildung 78: SC7538-08UTG Block Diagramm	91
Abbildung 79: LPF-Schaltung.....	93
Abbildung 80: Filter Prototyp mit TL074.....	94
Abbildung 81: Spannungsverhältnis Ua/Ue.....	96
Abbildung 82: Simulation des Verhältnis Ua/Ue	96
Abbildung 83: Messung des Filters	97
Abbildung 84: ESP32 Beschaltung.....	99
Abbildung 85: Reset bei Spannungseinbruch	100
Abbildung 86: Taster und Schiebeschalter.....	101
Abbildung 87: Beschaltung der USB-Bridge	101
Abbildung 88: Berechnung des maximalen Stromes im PCB-Toolkit Rechner	103
Abbildung 89: Versorgungsleitungen (5V rot, 3V3 orange)	103
Abbildung 90: Komponenten des Pi-Filters	103
Abbildung 91: Visualisierung einer Sternpunktmasse.....	104
Abbildung 92: SMD-Jumper.....	104
Abbildung 93: Bottom (links) und Top (rechts) Masse-Layer.....	105

Datenauszug 1: Ausgabe Audacity	66
Datenauszug 2: Output des Testprogramms.....	69

- [11] Espressif Systems (Shanghai) Co., Ltd, „<https://www.espressif.com/en/support/documents/technical-documents>,“ 13 02 2023. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32e_esp32-wroom-32ue_datasheet_en.pdf. ✓
- [12] Wdwd, „ESP32,“ Wikimedia Foundation Inc, 12 03 2023. [Online]. Available: <https://de.wikipedia.org/wiki/ESP32>. ✓
- [13] Espressif Systems (Shanghai) Co., Ltd., „ESP-AT User Guide,“ Espressif Systems (Shanghai) Co., Ltd., 2023. [Online]. Available: https://docs.espressif.com/projects/esp-at/en/latest/esp32/Get Started/Downloading_guide.html.
- [14] „FreeRTOS,“ Amazon Web Services, Inc., [Online]. Available: <https://www.freertos.org/>.
- [15] „UART verstehen,“ Rohde & Schwarz Österreich GesmbH, 2023. [Online]. Available: https://www.rohde-schwarz.com/at/produkte/messtechnik/essentials-test-equipment/digital-oscilloscopes/uart-verstehen_254524.html#:~:text=UART%20steht%20f%C3%BCr%20Universal%20Asynchronous,zu%20senden%20und%20zu%20empfangen..
- [16] E. Peña, „A Hardware Communication Protocol Understanding Universal Asynchronous Receiver/Transmitter,“ Analog Devices, Inc., 04 12 2020. [Online]. Available: <https://www.analog.com/en/analog-dialogue/articles/uart-a-hardware-communication-protocol.html>.
- [17] ZeM College GbR, „ZeM College MIDI Kompendium,“ ZeM College GbR, 2008. [Online]. Available: <https://www.zem-college.de/indexf.html>.

- [25] Espressif Systems (Shanghai) Co., Ltd, „API Reference - ESP32,” Espressif Systems (Shanghai) Co., Ltd, 2016. [Online]. Available: <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/index.html>.
- [26] „ElectronicsTutorials,” [Online]. Available: <https://www.electronics-tutorials.ws/filter/sallen-key-filter.html>.
- [27] „ScienceDirect,” [Online]. Available: <https://www.sciencedirect.com/topics/engineering/sallen-key>.
- [28] M. Hughes, „Ableton-Referenzhandbuch Version 11,” Ableton AG, 2021. [Online]. Available: <https://www.ableton.com/de/manual/credits/>.
- [29] M. Hughes, „Neue Clips aufnehmen,” Ableton AG, 2021. [Online]. Available: <https://www.ableton.com/de/manual/recording-new-clips/>.
- [30] Espressif Systems (Shanghai) Co., Ltd., „Get Started,” Espressif Systems (Shanghai) Co., Ltd., 2023. [Online]. Available: <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/index.html>.
- [31] „spsc.tugraz.at,” [Online]. Available: <data:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAASQAAACtCAMAAAAU7/J6AAAB8IBMVEX///d3d3g4ODm5ubV1dUAAADb29v19fvqamrt7e20tLS6urrAwMDS0tJvb2/KysqZmZmurq4QEBCKioo8PDzw8PD6//9GRkb//r6+vorKyumpqaenp7///fp6emDg4N2dnZOTk4gICD//+xYWFj/+9+fm8Pt///38OmRkZH88N3iw5sAV>.
- [32] „blog.bliley.com,” [Online]. Available: https://blog.bliley.com/fs/hubfs/filter_post/filter-response-comparison.png?width=640&name=filter-response-comparison.png.

10 Anhang

10.1 Begleitprotokolle

Name: Hr. Daniel Bräumann
Diplomarbeitstitel: Gitcon – Entwicklung einer MIDI-Schnittstelle für E-Gitarren

KW	Beschreibung	Zeitaufwand
38	Diplomarbeitsantrag erstellen	3
39	Diplomarbeitsantrag einreichen	3
40	Projektplanung, Aufgabenverfeinerung, Ganttdiagramm	3
41	Schaltungsentwurf	5
42	Erstdimensionierung der Schaltung	10
43	Prototypenbau	6
44	Prototypenbau & Simulation	6
45	Simulation	4
46	Bewertung des Prototyps als DUT	10
47	Bewertung des Prototyps als DUT	7
48	Optimierung der Schaltung	6
49	Optimierung der Schaltung	4
50	Simulation	6
51	Prototypenbau	5
52	Bewertung des Prototyps als DUT	7
01	Diplomarbeit schreiben	4
02	Diplomarbeit schreiben	6
03	Systemtests für das AFE	8
04	Systemtests für das AFE	6
05	Vertiefende Grundlagenarbeit	6
06	Vertiefende Grundlagenarbeit	8
07	Simulation des gesamten AFEs	7
08	Diplomarbeit schreiben	3
09	Vertiefende Grundlagenarbeit	4
10	Diplomarbeit schreiben	5
11	Diplomarbeit schreiben	7
12	Diplomarbeit schreiben	6
13	Diplomarbeit schreiben	3

KW ...Kalenderwoche

Name: Hr. Laurenz Hözl
Diplomarbeitstitel: Gitcon – Entwicklung einer MIDI-Schnittstelle für E-Gitarren

KW	Beschreibung	Zeitaufwand
38	Diplomarbeitsantrag erstellen	3
39	Diplomarbeitsantrag einreichen	3
40	Projektplanung, Aufgabenverfeinerung, Ganttdiagramm	3
41	Recherche zur FFT	8
42	Recherche zur FFT	12
43	Einrichten der Entwicklungsumgebung	10
44	Entwicklung Testprogramm	10
45	Entwicklung Testprogramm	10
46	Erstellen der Presentation	10
47	Entwicklung Testprogramm	10
48	Testdatenfertigung	9
49	Testdatenfertigung	4
50	Matlab programmierung	7
51	Tests	5
52	Anpassung des Testprogramms	6
01	Anpassung des Testprogramms	5
02	Entwicklung der Notenerkennung und -Übertragung	10
03	Entwicklung der Notenerkennung und -Übertragung	10
04	Entwicklung der Notenerkennung und -Übertragung	5
05	Diplomarbeit schreiben	6
06	Vertiefende Grundlagenarbeit	8
07	Diplomarbeit schreiben	7
08	Test und Feintuning der Firmware	8
09	Test und Feintuning der Firmware	7
10	Test und Feintuning der Firmware	5
11	Diplomarbeit schreiben	7
12	Diplomarbeit schreiben	6
13	Diplomarbeit schreiben	3

KW ...Kalenderwoche