



# High-End Audio-Streamer

Das Streamen von Musik in CD-Qualität bietet einen Ersatz für bestehende Radios und CD-Player in High-End Musikanlagen. Diese Anlagen stellen hohe Anforderungen an die Signalqualität, Langlebigkeit und Wertigkeit der Audio-Quellen. Ziel dieser Bachelor Thesis ist die Entwicklung eines High-End Audio-Streamers, der diesen Erwartungen gerecht wird.

## Bachelor Thesis

Autoren:	Rafael Klossner, Stefan Lüthi
Betreuer:	Roger Weber
Auftraggeber:	Institut für Intelligente Industrielle Systeme I3S, BFH
Experte:	Daniel Kühni
Datum:	27.06.2019, V1.0
Studiengang:	Elektrotechnik und Informationstechnologie



# Zusammenfassung

Ziel dieser Bachelor-Thesis und der vorangegangenen Projektstudie ist die Neuentwicklung eines Audio-Streamers als Ersatz für CD-Player und FM-Radio in High-End Systemen. Durch den Audio-Streamer sollen bestehende High-End Verstärker mit den modernen digitalen Audio-Quellen über das Internet oder von lokalen Speichermedien ergänzt werden.

Die Vorstudie dieser Arbeit beinhaltet den Vergleich und die Selektion der Kernkomponenten wie Digital/Analog Wandler (DAC) und Computermodul. Entsprechende Entwicklungs-Kits wurden für eine Evaluation beschafft. Mit diesem Testaufbau konnte das Linux System für das Computermodul (Toradex Apalis iMX6Quad) eingerichtet und der digitale Datenstrom zum DAC (AKM AK4493) sichergestellt werden. Zudem wurde für die Bedienoberfläche auf dem Touch-Display das GUI-Framework Qt eingerichtet. Nebst den Softwarekomponenten wurden die Schaltungen für die Spannungsversorgung, die Peripherie des Computermoduls und die Beschaltung des DAC erstellt. Daraus entstand das Design der entsprechenden Leiterplatten.

Zu Beginn der Thesis galt es, die Leiterplatten zu bestücken und die Hardware zu testen. Dabei wurden einige Fehler auf der komplexen Schaltung korrigiert und dokumentiert. Als Vorbereitung für die Bearbeitung des Gehäuses wurden die Ausschnitte und Taschen im Front- und Back-Panel in einem CAD-Programm geplant. Nach Abschluss der Gehäusebearbeitung konnten die Hardwarekomponenten verbaut werden. Parallel dazu wurde ein bestehender Audio-Treiber auf die verwendeten Komponenten angepasst. Dazu mussten auf dem Computermodul die Pfade für Takte und Signale konfiguriert werden. Nach erfolgreicher Inbetriebnahme des kompletten Audio-Streamers wurde die Qualität der Spannungsversorgungen, der digitalen Signale und des Audioausgangs messtechnisch überprüft. Zur Audiomeldung kommt auf dem Computermodul der Musikserver Mopidy zum Einsatz. Mit Plugins konnte der Server mit Streamingdiensten und lokalen Quellen ergänzt werden. Der Audioserver wird über das eingebaute Display gesteuert. Dazu wurde das Software-Design für ein Bedienprogramm erstellt. Mit Verwendung der Model-View-Controller Architektur wird die Erweiterbarkeit und Wartbarkeit der Software sichergestellt. Ein Teil des Designs wurde anschliessend implementiert.

Das Ergebnis der Thesis ist ein hochwertiges Gerät, welches Audiodateien in CD-Qualität verlustfrei wiedergeben kann. Mit der entwickelten Software kann der Audio-Streamer Informationen und das Cover zum aktuellen Titel auf dem Display anzeigen. Die Bedienoberfläche ist minimalistisch und mit einer einfachen Bedienstruktur gestaltet. Nebst verschiedenen Streaming-Diensten können Audiodateien direkt vom internen Datenträger des Audio-Streamers abgespielt werden.

Neben den positiven Resultaten bleiben noch einige Pendenzen offen. So gilt es in Zukunft die vorgeschlagenen Korrekturen der Schaltung zu übernehmen und die Leiterplatte anzupassen. Bei der Software soll die Bedienoberfläche mit weiteren Funktionen wie Suchen und Konfiguration der Hardware ausgestattet werden. Zur weiteren Aufwertung des digitalen Audiosignals soll der externe Oszillator anstatt des im Computer Modul verbauten Taktgenerators verwendet werden.



# Inhaltsverzeichnis

Zusammenfassung .....	iii
1 Einleitung .....	1
2 Grundlagen .....	3
2.1 Konzept .....	3
2.2 Software .....	3
2.3 Hardware .....	4
3 Layout & Gehäuse .....	5
3.1 Mainboard .....	5
3.2 Power Supply Unit .....	8
3.3 Rotary Encoder .....	9
3.4 Korrekturen Schema & Layout .....	9
3.5 Gehäuse .....	16
3.6 Verdrahtung .....	18
3.7 Endmontage .....	19
3.8 Kostenübersicht .....	20
4 Linux System & Treiber .....	21
4.1 Überblick .....	21
4.2 Begriffe und Zusammenhänge .....	21
4.3 Display Treiber .....	25
4.4 Backlight Treiber .....	26
4.5 Touchscreen Treiber .....	27
4.6 Audio-Treiber für den AK4493 DAC .....	28
4.7 Power-Management Skript .....	36
4.8 Treiber für den Drehgeber .....	38
5 Musikserver & Front-End .....	39
5.1 Software Architektur .....	39
5.2 Streamer-UI SW Design .....	42
5.3 CMake Build System .....	44
5.4 Graphical User Interface .....	45
6 Messungen & Tests .....	47
6.1 Material & Methoden .....	47
6.2 Ergebnisse & Diskussion .....	49
6.3 Qualitative Audiotests bei Dynavox .....	56
7 Fazit .....	59
7.1 Auswertung Ziele .....	59
7.2 Pendenzen .....	60
7.3 Ausblick .....	61
7.4 Persönliche Reflexion .....	61
8 Projektmanagement .....	63
8.1 Projektphasen .....	63
8.2 Zeitplan .....	63
8.3 Dokumente .....	67
Selbständigkeitserklärung .....	69
Abkürzungsverzeichnis .....	72

Literaturverzeichnis .....	73
Abbildungsverzeichnis .....	75
Tabellenverzeichnis .....	77
Versionenverzeichnis .....	79
A Bestückungspläne.....	81
B Stücklisten.....	87
C Vermassungen .....	99
D Verdrahtungsplan .....	105

# 1 Einleitung

Zum Erfolgreichen Abschluss des Studiums Bachelor of Science in Elektrotechnik und Informationstechnologie gehört die Bachelor Thesis. Im Rahmen dieser Arbeit sollen die erworbenen Kenntnisse aus dem Studium in der Praxis angewandt und vertieft werden. Dabei soll der Fokus auf der gewählten Vertiefung Embedded Systems liegen.

Die Bachelor Thesis *High-End Audio-Streaming* wird von Prof. Roger Weber betreut und wird von den Studenten Stefan Lüthi und Rafael Klossner bearbeitet. Daniel Kühni von der Firma Inetronic übernimmt die Funktion des Experten.

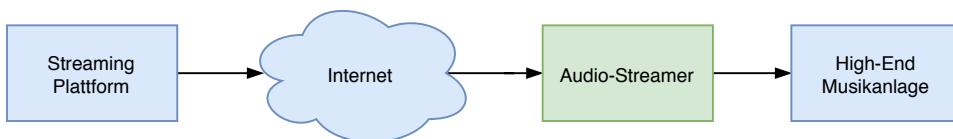


Abbildung 1.1: Audio-Streamer in seinem Umfeld

Im Rahmen der Thesis wird das ausgeschriebene Projekt *High-End Audio-Streaming* bearbeitet. In diesem Projekt geht es darum, ein Prototyp eines Streaming Gerätes zu entwickeln, welches bestehenden High-End Musikanlagen den Zugang zu modernen Musikquellen ermöglicht. Dieses Prinzip ist in Abbildung 1.1 dargestellt. Die CD als Datenträger war im Audio Bereich lange Zeit das Mass aller Dinge. Mit dem Aufkommen von Streaming Diensten, wurde der Zugang zu Musik einfacher. Mit einem Abonnement ist eine riesige Auswahl an Musik verfügbar. Allerdings war die Audio-Qualität, die viele Streaming Anbieter lieferten nicht mit der Audio-Qualität der CD vergleichbar. Da die Qualität aber immer besser wird und es einige Anbieter wie Tidal und Qobuz gibt, die ihre Musik sogar in CD-Qualität anbieten, hält das Streaming auch im High-End Bereich Einzug. Ein weiterer Aspekt ist der Empfang von Radio Signalen. Denn die UKW-Sender werden schon bald ausser Betrieb genommen. Der naheliegende Ersatz ist das Streaming von Radio-Daten über das Internet.

Es gibt heute Bereits Geräte welche das Streamen von Musik ermöglichen. Allerdings sind diese entweder teuer oder erfüllen die Qualitätsansprüche der Hörer nicht. Deshalb soll im Rahmen dieser Bachelor Thesis ein Prototyp eines Streaming Geräts entwickelt werden, welches die oben genannten Funktionen bietet.

Zur Marktanalyse und zur Definition des Konzepts wurde eine Vorstudie durchgeführt. Diese bildet die Basis für das Projekt High-End Audio-Streaming im Rahmen der Bachelor Thesis. Der Stand der Arbeit nach der Vorstudie wird im Kapitel 2 erklärt.

Der Inhalt dieses Dokuments, umfasst die Dokumentation der durchgeführten Arbeitsschritte im Entwicklungsprozess. Der Fokus bei der Dokumentation der Entwicklungsschritte liegt auf deren Wiederholbarkeit. Die Dokumentation soll als Grundlage zur Weiterentwicklung des Audio-Streamers verwendet werden können. Daher sollen auch Pendenzen präzise erfasst werden und Lösungsvorschläge für allfällige Probleme geliefert werden. Zusätzlich zur Dokumentation der einzelnen Schritte sollen die recherchierten Wissensgrundlagen zusammen mit ihren Referenzen beschrieben werden. Dies erleichtert das Verständnis der gesamten Thematik und dient der fundierten Belegung von getroffenen Entscheidungen.

In diesem Projekt soll ein Prototyp eines Audio-Streamers entwickelt werden. Das heißt das Ergebnis der Arbeit muss kein serienreifes Produkt sein. Da das Produkt voraussichtlich nur in einem kleinen Rahmen vertrieben wird, ist das Einhalten von Normen nicht notwendig. Trotzdem soll das fertige Produkt natürlich für den Benutzer sicher sein. Weiter wird auf Messungen rund um die Elektromagnetische Verträglichkeit (EMV) verzichtet. Es wird also nicht erfasst welche Störungen

das Gerät im Energienetz verursacht und in welcher Form das Gerät benachbarte Geräte in deren Funktion beeinträchtigt. Trotzdem wird diesem Aspekt im Konzept des Audio-Streamers Rechnung getragen.

Das Ziel dieses Projektes ist die Entwicklung eines funktionsfähigen Prototypen eines High-End Audio-Streamers. Das Gerät soll sich in erster Linie durch die gute Signalqualität auszeichnen. So soll der Audio-Streamer eine Audiodatei in CD-Qualität verlustfrei wiedergeben können. Diese Anforderung betrifft die Hardware. Die Grundlage für die Hardware wurde bereits bei der Auswahl der Komponenten und der Entwicklung der Leiterplatten gelegt. Dies geschah in der Vorstudie. In der Thesis geht es darum die Leiterplatten zu Bestücken und anschliessend eine umfassende Funktionsprüfung der einzelnen Leiterplatten und deren Komponenten durchzuführen. Damit das Gerät optisch einen guten Eindruck macht, sollen die Bestandteile in ein hochwertiges Gehäuse integriert werden. Die Anforderung an die Signalqualität wird messtechnisch überprüft. Die Resultate der Messungen werden in dieser Arbeit dokumentiert und interpretiert.

Ein weiterer zentraler Punkt ist das Bedienkonzept, welches bereits in der Vorstudie festgelegt wurde. Dazu ist die Bedienung per Touchscreen vorgesehen. Dies, weil ein Display mit Touchscreen Eingabe- und Ausgabegerät vereint und somit eine intuitive Bedienung ermöglicht. Als zusätzliches Feature soll in der Frontplatte des Geräts ein Drehgeber eingebaut werden, welcher das Scrollen durch Listen vereinfacht. Da der Drehgeber optisch ansprechend wirken soll, wird dieser mit einem Backlight hinterleuchtet.

Das Betreiben der einzelnen Bedienelemente und des DACs soll durch das Linux System erfolgen. Dazu soll das System so konfiguriert werden, dass die nötigen Treiber geladen werden. Um das Gerät bedienen zu können, muss eine Software für den Benutzer entwickelt werden. Diese soll dem Nutzer das Abspielen von Audiodateien aus unterschiedlichen Quellen ermöglichen. Zusätzlich soll die Software Informationen der Audiodatei auf dem Display anzeigen. Dies sind unter anderem der Titel des Songs, der Interpret und das Album. Um das Interface ansprechend zu gestalten soll zusätzlich das Cover angezeigt werden.

Damit das Projekt weitergeführt werden kann, sollen die Komponenten in Software und Hardware mit der nötigen Weitsicht bestimmt werden. Da das Projekt als Open Source Projekt weitergeführt wird, soll insbesondere im Bereich Software auf Open Source Lösungen gesetzt werden. Somit ist es für einen zukünftigen Besitzer eines solchen Streaming Geräts möglich, sich selbst an der Weiterentwicklung zu beteiligen.

# 2 Grundlagen

## 2.1 Konzept

Als Grundlage dieses Projekts dienen die Erkenntnisse, Entscheidungen und Entwicklungen aus der Vorstudie. In der Vorstudie wurde eine Marktanalyse durchgeführt, um eine Übersicht über vergleichbare existierende Geräte zu erhalten. Aufgrund dieser Analyse wurde das Konzept für den High-End Audio-Streamer erstellt. Das Konzept macht Angaben zur Bedienung, dem Gehäuse und dem Blockschaltbild des Gerätes. Das Blockschaltbild in Abbildung 2.1 zeigt den groben Aufbau der Hardware des Geräts. Anhand des Blockschaltbildes konnten die Komponenten ausgewählt werden. Die dazu infrage kommenden Komponenten wurden miteinander verglichen. Die wichtigsten Komponenten sind Digital/Analog-Converter (DAC), Operationsverstärker für die analoge Ausgangsstufe, das Computermodul (CoM) und das Display.

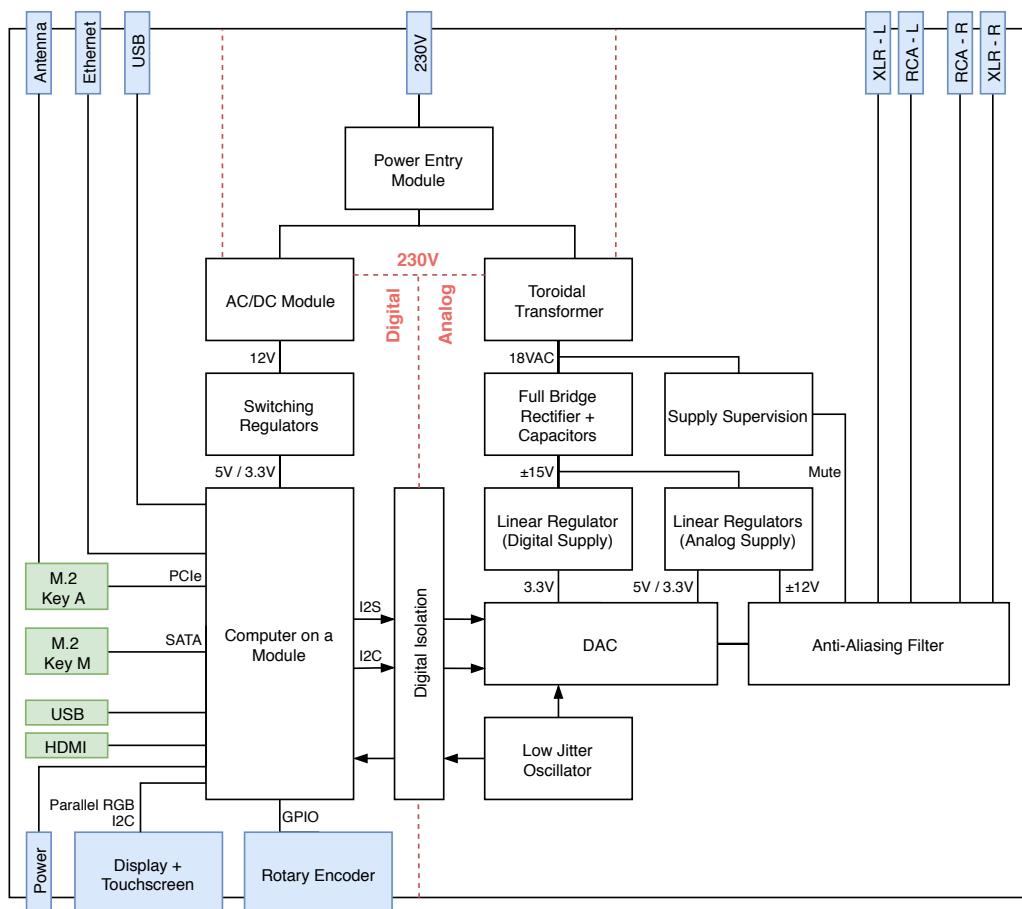


Abbildung 2.1: Schaltungskonzept des Audio-Streamers aus [1]

## 2.2 Software

Als Betriebssystem für den High-End Audio-Streamer wurde Linux gewählt. Dies wurde bereits vor dem Projektstart festgelegt und ist die einzige sinnvolle Lösung im Hinblick auf die Weiterführung des Projekts als Open Source Entwicklung. Damit das Linux System die Anforderungen erfüllt, welche durch das Konzept gestellt wurden, mussten die einzelnen Systemkomponenten

entsprechend ausgewählt und vorbereitet werden. Dies geschah bereits in der Vorstudie der Thesis. Während der Vorstudie wurden die wichtigsten Linux Systemkomponenten ausgewählt und ins Gesamtsystem integriert. Dabei handelt es sich um die Toolchain, den Linux Kernel, das U-Boot, die Flashing-Tools, den Device-Tree und das Filesystem. Wobei beim Kernel, beim Device-Tree, beim U-Boot und bei den Flashing-Tools auf die Lösung von Toradex gesetzt wurde. Dies weil als Computermodul das Apalis-iMX6Q von Toradex eingesetzt wird. Beim Filesystem wird auf Debian gesetzt, sodass einfach Pakete und Abhängigkeiten nachinstalliert werden können. Noch in der Vorstudie wurden die gewählten Komponenten zum Gesamtsystem zusammengesetzt. Das Ergebnis zum Ende der Projektstudie war ein bootbares, stabiles Betriebssystem. Dieses System diente während der Thesis als Grundlage für die Entwicklung des High-End Audio-Streamers.

Ein weiterer Aspekt ist das GUI-Framework. Es wird Qt zur Erstellung des Graphical User Interfaces (GUI) verwendet. In der Vorstudie wurden das GUI-Framework kompiliert, damit während der Thesis mit der Softwareentwicklung gestartet werden konnte. Allerdings bestanden nach Abschluss der Projektstudie noch Pendenzen im Bereich der Grafiktreiber und den Bibliotheken für die Verwendung der Vivante GPU des iMX6Q. Über den Framebuffer konnten aber bereits Qt Applikationen mit GUI gestartet werden.

Der letzte Aspekt hinsichtlich des Betriebssystems sind die Treiber für die Hardwarekomponenten. Die Hardwarekomponenten sind Display, Backlight des Displays, Touchscreen, Digital/Analog-Converter (Audio for System on Chip), Drehgeber und die GPIO's für die Ansteuerung der analogen und digitalen Speisungen. Während der Projektstudie konnte der Treiber für das Display erfolgreich angepasst werden. Damit war es mit einer Qt Applikation über den Framebuffer möglich, Informationen auf dem Display anzuzeigen. Die übrigen Treiber waren bereits im Linux Treiber vorhanden. Allerdings in der Vorstudie kein Test der Treiber mehr durchgeführt werden.

## 2.3 Hardware

Auf Basis der Marktanalyse und den gewählten Kernkomponenten wurde die Schaltung für den Audio-Streamer erstellt. Dieses umfasst die Gruppen: Speisung, DAC und CoM.

Die Speisung wurde aufgetrennt auf zwei Leiterplatten. Die Wandlung von AC-Netzspannung zu DC wird auf der Power Supply Unit (PSU) erledigt. Dazu wurde für den digitalen Teil ein Schaltnetzteil und für den analogen Teil eine Transformator-Gleichrichter-Schaltung verwendet. Durch diesen Aufbau sind der digitale und analoge Teil galvanisch getrennt.

Auf dem Mainboard wurden das CoM mit Peripherie und die DACs untergebracht. Die Beschaltung des CoM basierte auf dem Toradex Ixora Carrier Board Design, welches der Hersteller als Altium Projekt veröffentlichte. Das Design musste für die Verwendung auf dem Audio-Streamer angepasst und modernisiert werden. Die Beschaltung der DACs richtete sich nach dem Datenblatt und dem Evaluation Kits des Herstellers. Das Antialiasing Filter an den analogen Ausgängen der DACs wurde neu ausgelegt und simuliert. Um den analogen und den digitalen Teil galvanisch getrennt zu halten, wurden digitale Isolatoren für die Übermittlung des Audio-Streams vom CoM zum DAC verwendet.

Nach Abschluss des Schaltungsdesigns wurde das Layout für die verschiedenen Leiterplatten erstellt. Dabei mussten die Vorgaben für die High-Speed Signale eingehalten werden. Die Leiterplatten wurden am Ende der Projektstudie zur Herstellung in Auftrag gegeben. Die Stücklisten und Bestückungspläne wurden bis zu diesem Zeitpunkt noch nicht erstellt.

Der Audio-Streamer ist in einem Aluminium Gehäuse untergebracht, dieses wurde bereits während der Projektstudie beschafft und erste Skizzen einem CAD-Tool wurden erstellt. Die Ausarbeitung und Vermassung, sowie die Bearbeitung waren noch ausstehend.

# 3 Layout & Gehäuse

Dieses Kapitel beinhaltet die Erklärungen für das Routing der Leiterplatten, das Gehäusedesign, sowie Anweisungen für die Endmontage und eine Kostenübersicht.

Da das Layout zum Ende der Projektstudie fertiggestellt wurde, konnte die Platzierung der Komponenten bereits in der Dokumentation der Projektstudie [1] beschrieben werden.

## 3.1 Mainboard

### 3.1.1 Signal Layer

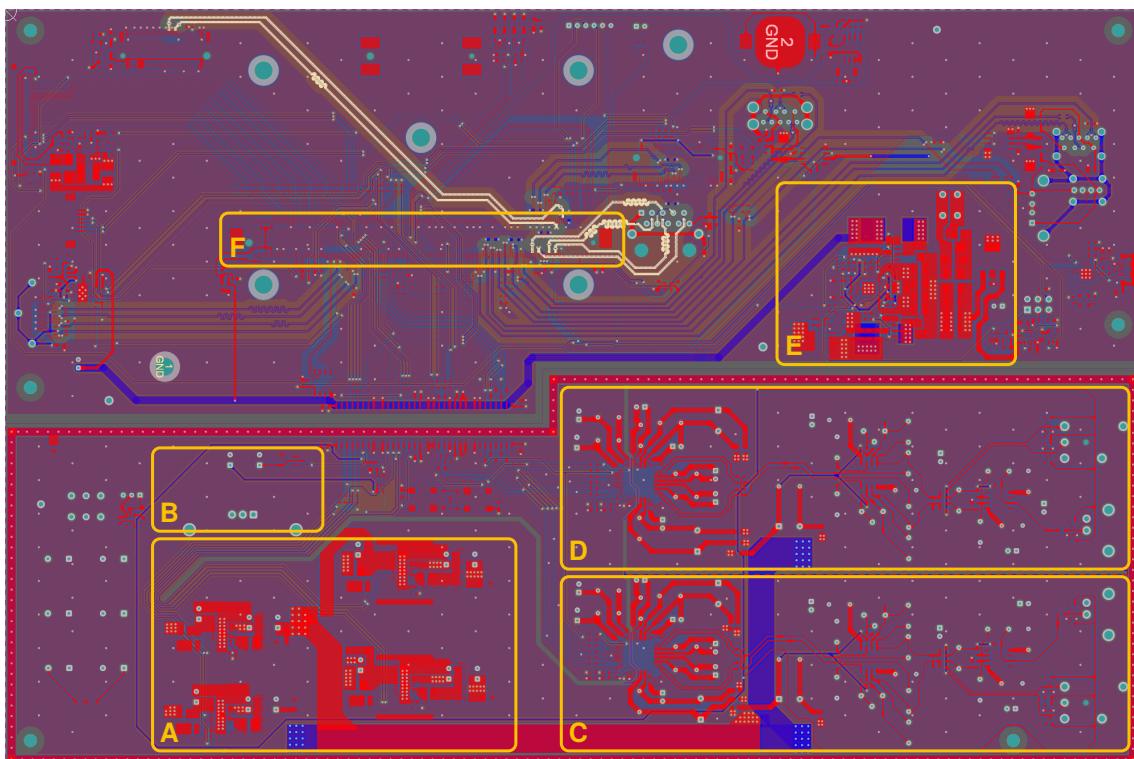


Abbildung 3.1: Layout Top (L1, rot) und Bottom (L4, blau) Layer

In Abbildung 3.1 sind beiden Signallayer des Mainboards dargestellt. Das CoM und dessen Be- schaltung befindet sich in der oberen, der DAC in der unteren Hälfte der Leiterplatte. Die beiden Zonen werden durch eine kupferfreie Fläche (grau) getrennt.

#### Speisungen

Zwischen der digitalen (Detail B) und den vier analogen Speisungen (Detail A) des DACs ist die Trennung zwischen beiden Speisungen als graue Linie ersichtlich. Die Signalleitungen der analogen Speisungen werden über den Sternpunkt geführt, um Rückweg des Stromes direkt auf der Fläche unterhalb der Signale sicherzustellen. Beim Layout der Spannungsregler wurde der Vor- schlag des jeweiligen Herstellers übernommen.

Die stromführenden Verbindungen der Speisungen wurden so breit wie möglich oder als Fläche geführt, um die Impedanz tief zu halten. Dies ist in den Details A und E anhand der roten Flächen

ersichtlich. Beim Wechsel auf einen anderen Layer werden etliche Vias verwendet. Somit wird die Induktivität verringert und die Strombelastbarkeit erhöht. Vias direkt neben den Spannungsreglern werden im Gegensatz zu den restlichen Vias nicht mit Stopplack überzogen, um ein Aufplatzen bei einem Temperaturanstieg zu vermeiden. Diese Vias dienen zudem zum Abführen der Hitze, welche aufgrund der Verlustleistung der Regler entsteht.

### DAC

Die beiden Kanäle des DAC in Detail C und D sind im Layout weitgehend identisch aufgebaut. Anschlüsse an die Speisung erfolgen jeweils über einen Blockkondensator und danach über ein Via auf die Speisungsfläche. Das GND-Polygon wurde um die DAC Chips herum entfernt, um den Strompfad über die Blockkondensatoren sicherzustellen. Die Audioleitungen werden nur auf dem Top Layer geführt, um Induktivitäten durch Vias zu verhindern. Die differentiellen Audiosignale sind anhand der parallelen Leiter zu erkennen.

### CoM

Der Sockel des CoM ist im Detail F abgebildet. Dieser wurde zentral platziert, um kurze Leitungen zu allen Peripheriebausteinen zu ermöglichen. Die high-speed Signalleitungen sind auf dem Bottom Layer geführt (blau) und anhand des grossen Freiraums zum GND Polygon zu erkennen. Die Anpassung der Leiterbahnlängen wurde durch Serpentinen realisiert. Alle high-speed Signale erfüllen die Anforderung der Toradex Layout Design Guidelines [2].

### GND Polygon

Ungenutzte Flächen sind mit einem GND-Polygon gefüllt. Dies soll die Signale vor Störungen abschirmen. Mit der Via-Stitching Funktion von Altium Designer konnte sichergestellt werden, dass die Impedanz zum GND Layer tief bleibt.

#### 3.1.2 VCC Layer

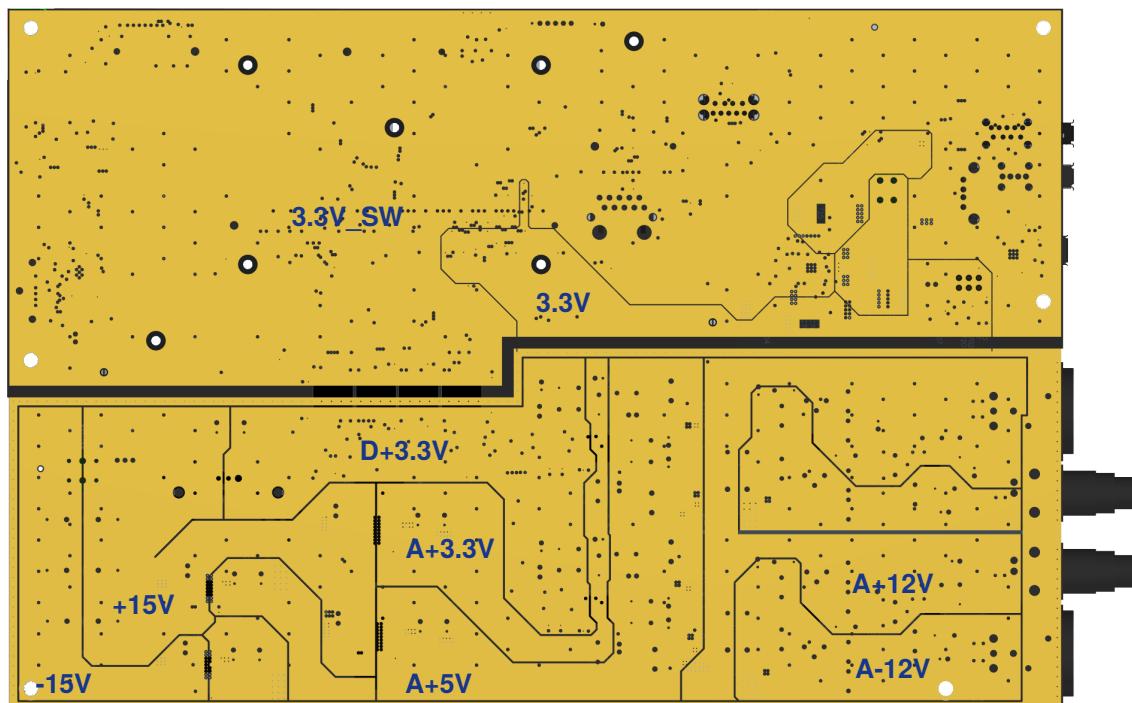


Abbildung 3.2: Layout VCC (L2) Layer

Aus Abbildung 3.2 werden die Auswirkungen der unterschiedlichen Anforderungen an die Stromversorgung im CoM und DAC Teil deutlich. Dies äussert sich durch die vielen Flächen auf DAC Teil (unten) im Vergleich zu den wenigen grossen Flächen im CoM Teil (oben).

## CoM

Die Speisung des CoM gestaltet sich sichtlich einfach. Es wird lediglich eine 3.3 V Speisung benötigt. Die 3.3V\_SW Speisung versorgt alle Peripheriebausteine und wird vom CoM beim Starten aktiviert. Somit kann der Stromverbrauch im Standby durch Deaktivierung der Peripherie minimiert werden. Damit das CoM im Standby mit Strom versorgt wird, ist die 3.3V Speisung immer eingeschaltet.

## DAC

Der analoge Teil benötigt insgesamt fünf verschiedene Speisungen. Zusätzlich bilden die DACs eine Brücke zwischen den digitalen und analogen Signalen. Es wurde systematisch auf deren Trennung geachtet. Die Speisungen sind möglichst sternförmig ausgelegt, um die Kanäle möglichst voneinander zu separieren. Dies ist jedoch aufgrund der Platzverhältnisse nicht überall möglich. So werden die analogen Speisungen von unten zu den beiden Kanälen kaskadiert. Dadurch könnten sich Störungen vom einem auf den anderen Kanal übertragen. Dies kann mit der Messung des Crosstalk überprüft werden.

### 3.1.3 GND Layer

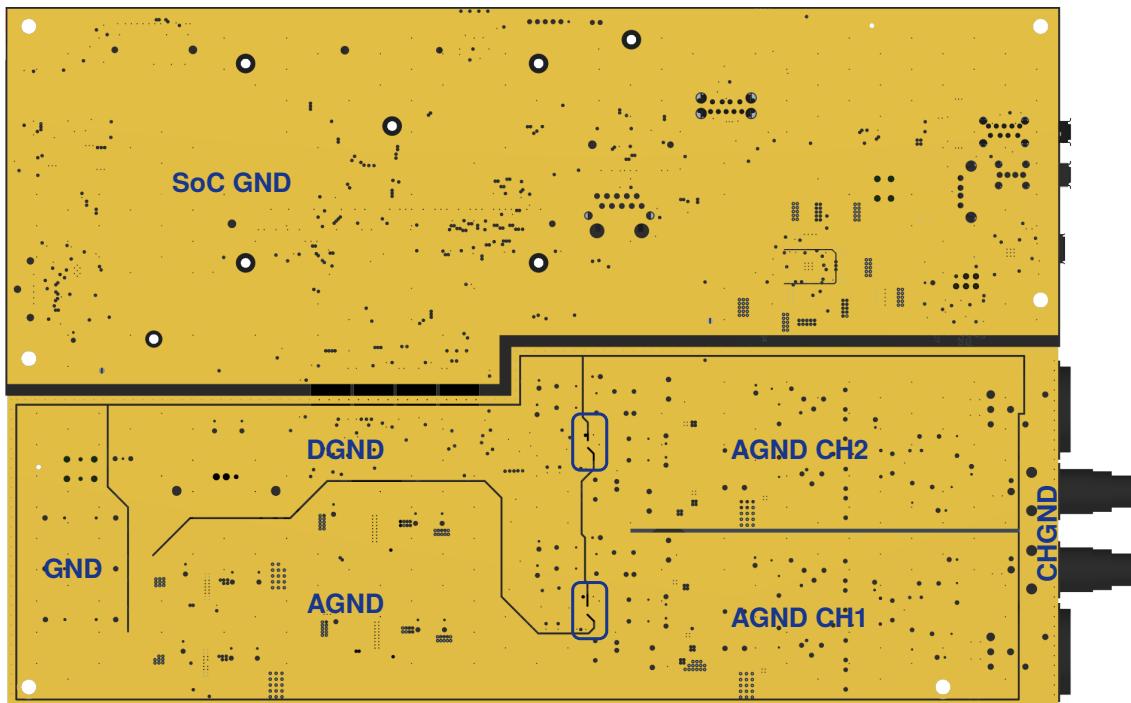


Abbildung 3.3: Layout GND (L3) Layer

Der GND Layer gestaltet sich auf der CoM Seite wiederum einfach. Auf der analogen Seite in Abbildung 3.3 wurde wiederum auf die sternförmige Trennung von analogem und digitalem GND geachtet. AGND und DGND werden beim Klemmverbinder zusammengeführt, wie auch jeweils unter dem DAC Chip (Markierung). Bei einem DAC Chip ist die Verbindung der beiden GND zwingend notwendig. Mit den Ausschnitten im GND Polygon auf den Signal-Layern werden parallele GND Brücken verhindert. Somit werden AGND und DGND unterhalb der DAC Chips jeweils nur auf einem Layer verbunden. Die beiden DAC Kanäle sind wiederum sternförmig getrennt.

Wie schon auf den anderen drei Layern wird der analoge Teil von einem Chassis Ground (CHGND) umringt. Dieses ist auf allen vier Lagen vorhanden und dicht mit Vias verbunden. Dieser Ring kann bei Bedarf auf der Ober- und Unterseite mit einer Abschirmung verlötet werden. Das Chassis Ground wird beim Klemmverbinder mit dem GND verbunden.

## 3.2 Power Supply Unit

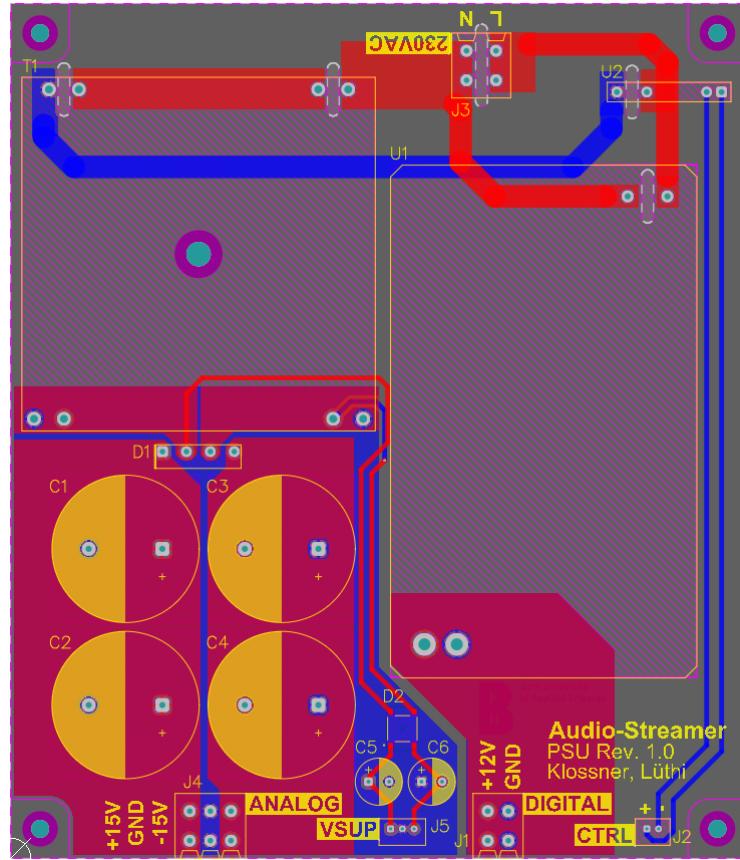


Abbildung 3.4: Layout PSU, Top (rot) und Bottom (blau)

Die Speisung für den DAC befindet sich in Abbildung 3.4 auf der linken Seite. Der Transformator ist zu oberst platziert. Darunter befinden sich der Gleichrichter und die Glättungskondensatoren. Auf der rechten Seite ist das AC/DC-Modul für das CoM platziert. Oben rechts liegt der elektronische Schalter, um den Transformator vom Netz zu trennen.

Die Leitungen im 230 V-Teil sind kurz gehalten, um den gemeinsamen Pfad der digitalen und analogen Speisung zu minimieren. Dadurch soll eine Antennenwirkung verhindert und EMV-Probleme vermieden werden. Zwischen den Anschlusspins auf der Netzseite sind Isolationsslots eingefügt. Diese sollen bei verschmutzten Leiterplatten (z.B. durch Flussmittelrückstände oder Staub) Funkenbrüche durch eine Verlängerung des Weges unterbinden.

Auf der DC-Seite sind die Verbindungen mit Polygonen realisiert. Dadurch wird der Leitungswiderstand zwischen den Komponenten minimiert. Zwischen der analogen Speisung und dem AC/DC-Modul befindet sich ein zusätzlicher Gleichrichter für die Spannungsüberwachung.

### 3.3 Rotary Encoder

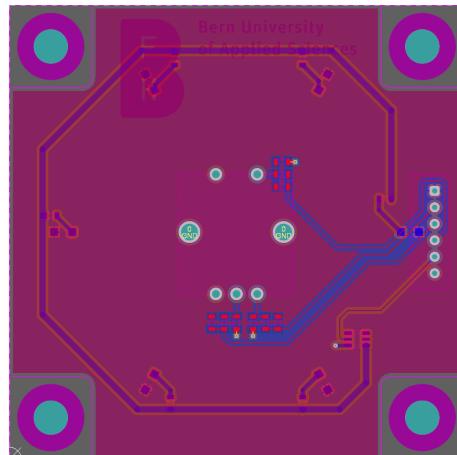


Abbildung 3.5: Layout Rotary Encoder, Top (rot) und Bottom (blau)

Die Leiterplatte für den Drehgeber ist, wie in Abbildung 3.5 abgebildet, simpel gestaltet. Die LEDs und Vorwiderstände befinden sich auf dem Bottom-Layer der Leiterplatte. Da die Anschlussleitung der LEDs einen Ring bildet und mit einem PWM-Signal versorgt wird, besteht die Gefahr Störungen über diese Antenne zu verbreiten. Um dies zu vermeiden ist der Top-Layer mit einer GND-Fläche gefüllt. Die Fläche ist somit nach der Endmontage zwischen der LED-Leitung und dem Innern des Audio-Streamers platziert und kann so die Störungen abschirmen.

## 3.4 Korrekturen Schema & Layout

Bei der Bestückung und Inbetriebnahme traten einige Probleme auf. Diese hatten je nach Situation die Wahl einer alternativen Komponente oder das Verbinden mit Drähten zur Folge. Die Fehlerquellen liegen in den Footprints, den Komponenten oder dem Schema. Nachfolgend werden die Fehler nach Auswirkungsgrad sortiert aufgelistet.

### 3.4.1 Verbesserungen

Mit den folgenden Punkten kann das Handbestücken vereinfacht und das Layout aufgewertet werden. Diese Vorschläge haben keine Auswirkung auf die Funktionalität der Schaltung. Es wurden deshalb auf dem Prototypen keine Anpassungen vorgenommen.

- Lochdurchmesser der JST PH Stecker vergrössern: Einige Stecker konnten nur mit Kraft in die Bohrungen gedrückt werden
- Bei Testpunkten den Paste Layer entfernen: Die Testpunkte werden fälschlicherweise verzinnnt, dadurch rutscht die Messsonde ab
- M.2 Key M Through Hole Plating entfernen: Die Bohrung für den Distanzhalter ist verzинnt
- Aktivieren von Thermal Relief auf dem Top und Bottom Layer des Mainboards: Zur Verbesserung der elektrischen Charakteristik wurden Wärmebrücken auf dem Mainboard deaktiviert, dies erschwert das Bestücken massiv; das Aktivieren von Wärmebrücken auf dem Top und Bottom Layer würde die Wärmeübertragung des Lötkolbens verbessern, der interne Ground-Layer würde weiterhin ohne Wärmebrücken angeschlossen werden
- Beim XLR Stecker verzinnte Bohrungen tauschen: Die Befestigungslöcher für die Plastikknöpfe sind verzinnt, die Bohrung für den elektrischen Chassis Anschluss ist nicht verzinnt
- Weitere Befestigungslöcher hinzufügen: Mit den sechs Befestigungslöchern wird das Mainboard beim Einsticken von Kabeln durchgebogen

- Restringe auf dem Mainboard vergrössern: Die kleinen Restringe erschweren das Löten von Pins, welche an die Ground-Flächen angeschlossen sind

### 3.4.2 Zwingende Korrekturen

Für die erfolgreiche Inbetriebnahme mussten einige Anpassungen auf den Leiterplatten vorgenommen werden. Deren Korrektur ist für die nächste Iteration zwingend. In der Tabelle 3.1 sind die umgesetzten Korrekturen in Bezug auf die Version 1.0 der Leiterplatten aufgeführt. Zudem wird jeweils eine Massnahme zur Korrektur in der nächsten Iteration angegeben.

Komponente	Fehler	Korrektur
Mainboard U400	<b>Verhalten</b> Bestücken nicht möglich	<b>Fix</b> Ersetzt durch Si8660BD-B-IS (SOIC Wide)
U402	<b>Fehler</b>	<b>Massnahme</b>
U402	Footprint SOIC-16 Wide anstatt QSOP-16	Footprint auf SOIC Narrow anpassen und Si8660BB-B-IS1 verwenden, da günstiger als Fix
Mainboard U401	<b>Verhalten</b> Bestücken nicht möglich	<b>Fix</b> Ersetzt durch MAX14937AWE+ (SOIC Wide)
	<b>Fehler</b>	<b>Massnahme</b>
	Footprint SOIC-16 Wide anstatt SOIC-16 Narrow	Footprint auf SOIC Narrow anpassen und MAX14933ASE+ verwenden, da günstiger als Fix
Mainboard U200	<b>Verhalten</b> MUTE immer auf Low	<b>Fix</b> Muting deaktiviert (R407 entfernt)
	<b>Fehler</b>	<b>Massnahme</b>
	$V_{IN} > V_{IN,MAX}$ , IC übersteuert	Muting testen und Schaltung anpassen (z.B. Spannungsteiler vor U200)
Mainboard VR203	<b>Verhalten</b> Regelt Spannung nicht	<b>Fix</b> Thermal Pad auf PCB mit Kapton Klebeband isoliert, Kupferband und Thermal Pad (IC) gelötet, IC eingelötet, Pin 1 mit Thermal Pad verdrahtet (siehe Abbildung 3.9)
	$V_{OUT} \approx V_{IN}$	
	<b>Fehler</b>	<b>Massnahme</b>
	Thermal Pad (Pin 13) auf GND statt VIN	Schemasymbol und Schaltung korrigieren, Pin 13 (Thermal Pad) → VIN
Mainboard VR203	<b>Verhalten</b> Power Good immer auf Low	<b>Fix</b> Power Good (Pin 4) und dessen Feedback (Pin 5) sind vertauscht, Leiterbahnen aufgetrennt und neu verdrahtet
	<b>Fehler</b>	<b>Massnahme</b>
	PG und PGFB vertauscht	Pin 4 und Pin 5 in Schema tauschen
Mainboard C213	<b>Verhalten</b> Kurzschluss nach ca. 1 h	<b>Fix</b> Kondensatoren umpolen
C217	Betrieb	<b>Massnahme</b>
	<b>Fehler</b>	→ Fix
	Kondensator verpolt	
Mainboard U302	<b>Verhalten</b> Udefinierte Zustände an den Eingängen des Operationsverstärkers	<b>Fix</b> Pin 7 (IC: VCC, PCB: NC) auf Pin 8 (IC: NC, PCB: VCC) verdrahtet
	<b>Fehler</b>	<b>Massnahme</b>
	Falscher Speisungspin, Pin 8 anstatt 7 (VCC)	Pin 8 auf 7 in Schemasymbol korrigieren

Tabelle 3.1: Korrekturen der Leiterplatten der Version 1.0 und empfohlene Anpassungen für die nächste Iteration

Komponente	Fehler	Korrektur
Mainboard R300 R301 C312 R341 C334	<b>Verhalten</b> Analog Filter schwingt <b>Fehler</b> Mit 5 V-Biasing beginnt der Filter im Leerlauf zu oszillieren	<b>Fix</b> R300, R301, C312, R341 und C334 entfernt, Biasing deaktiviert, resultiert in 2.5 V-Offset auf dem XLR Stecker, keine Auswirkung auf den single-ended Anschluss <b>Massnahme</b> siehe Unterabschnitt 3.4.3
Mainboard U402	<b>Verhalten</b> PS_EN funktionieren nicht <b>Fehler</b> Isolator Signal verschoben beschaltet, Seite A ist (soll): REF_CLK_EN_1 (PS_A12V_EN), REF_CLK_EN_2 (PS_A+3.3V_EN), PS_A12V_EN (PS_A+5V_EN), PS_A+3.3V_EN (MUTE), PS_A+5V_EN (REF_CLK_EN_1), MUTE (REF_CLK_EN_2)	<b>Fix</b> Anpassung der GPIO Pins in der Software <b>Massnahme</b> Pin 2 → 4, 3 → 5, 4 → 6, 5 → 7, 6 → 2 und 7 → 3 im Schema anpassen
Mainboard IC1400	<b>Verhalten</b> Bestücken nicht möglich <b>Fehler</b> Footprint DFN-16 anstatt QFN-18	<b>Fix</b> Nicht bestückt, da für Prototyp nicht notwendig <b>Massnahme</b> Footprint mit QFN18 ersetzen
Mainboard IC1000	<b>Verhalten</b> USB Buchsen nicht gespiesen <b>Fehler</b> Part A und B vertauscht	<b>Fix</b> Stromüberwachungs-IC entfernt, Verbindungen mit Drähten überbrückt <b>Massnahme</b> Part A und B in Schema tauschen
Mainboard RA900C	<b>Verhalten</b> USB 3.0 Buchse nicht gespiesen <b>Fehler</b> USB OTG ID high	<b>Fix</b> Pin auf GND verdrahtet, USBO1_ID muss im Host Modus auf Low sein <b>Massnahme</b> Pull-Down Widerstand auf USBO1_ID einfügen
Mainboard RA900C	<b>Verhalten</b> Toradex Easy Installer funktioniert nicht <b>Fehler</b> Keine USB OTG Device Buchse, USBO1_ID nicht umschaltbar	<b>Fix</b> CoM wird auf Ixora Board geflasht <b>Massnahme</b> USB Device Buchse und Jumper zum Umstellen der USBO1_ID einfügen

Tabelle 3.1: Korrekturen der Leiterplatten der Version 1.0 und empfohlene Anpassungen für die nächste Iteration

Komponente	Fehler	Korrektur
Mainboard IC1001	<b>Verhalten</b> USB 3.0 Strombegrenzung auf 500 mA begrenzt <b>Fehler</b> TPS 2052BD anstatt TPS2066CD	<b>Fix</b> Keine Anpassung, da keine Auswirkung auf Prototyp <b>Massnahme</b> Durch TPS2066CD (wie IC1000) ersetzen
Mainboard IC702	<b>Verhalten</b> Power Button schaltet Speisung aus, anstatt CoM zu benachrichtigen <b>Fehler</b> LTC294 falsch konfiguriert	<b>Fix</b> Keine Anpassung, da nur geringe Auswirkung auf Prototyp <b>Massnahme</b> Power-Downtime auf 3 s einstellen → C721 mit 470 nF bestücken; Pin 5 (INT) auf einen GPIO Pin des SoM führen, Pin als Power-Off Signal konfigurieren
Mainboard X1600	<b>Verhalten</b> CoM bootet nicht, wenn die SSD eingesteckt ist <b>Fehler</b> Pin 70 auf GND anstatt +3.3V	<b>Fix</b> Leiterbahn von Pin 70 zu GND getrennt <b>Massnahme</b> Schemasymbol korrigieren: Pin 70 → 3.3V, Pin 71 → CONFIG_1
Mainboard X1500 X1600	<b>Verhalten</b> SSD & WiFi Modul können nicht angeschraubt werden <b>Fehler</b> Distanzhalter falsch positioniert	<b>Fix</b> Module mit Klebeband befestigt <b>Massnahme</b> Bohrung im Footprint korrigieren
PSU U2	<b>Verhalten</b> Analog Speisung schaltet nicht ein <b>Fehler</b> Pin 1 und 2 vertauscht	<b>Fix</b> Kabel zu Mainboard kreuzen <b>Massnahme</b> Pin 1 und 2 in Schaltungssymbol tauschen
PSU T1	<b>Verhalten</b> Leerlaufspannung zu hoch <b>Fehler</b> Spannung höher als erwartet, über $V_{VIN,MAX} = 22\text{ V}$ der Linearregler	<b>Fix</b> Ersetzt durch L01-6361 <b>Massnahme</b> → Fix
PSU & Mainboard T1 & VR201	<b>Verhalten</b> Transformator blät sich auf (überhitzt) <b>Fehler</b> Analogspeisung des Mainboards hat keine Schutzschaltung	<b>Fix</b> Rückstellbare Sicherung in ±15 V-Leitungen eingebaut <b>Massnahme</b> Rückstellbare Sicherung (0ZCF0100AF2A, $I_{trip} = 1\text{ A}$ ) nach J200 auf Mainboard verbauen, zum Schutz vor Überspannung Zenerdioden (SMBJ5357B-TP, $V_Z = 20\text{ V}$ ) einbauen

Tabelle 3.1: Korrekturen der Leiterplatten der Version 1.0 und empfohlene Anpassungen für die nächste Iteration

Komponente	Fehler	Korrektur
Encoder R7 - R12	<b>Verhalten</b> LEDs leuchten schwach <b>Fehler</b> Vorwiderstände zu hoch	<b>Fix</b> Ersetzt mit $0\Omega$ -Widerständen <b>Massnahme</b> → Fix
Encoder D1 - D6	<b>Verhalten</b> Ausleuchtung ungleichmässig <b>Fehler</b> Zu wenige LEDs / Strahlungswinkel zu klein	<b>Fix</b> 6 weitere LEDs angebracht <b>Massnahme</b> Weitere LEDs verbauen oder LEDs mit grösserem Strahlungswinkel wählen

Tabelle 3.1: Korrekturen der Leiterplatten der Version 1.0 und empfohlene Anpassungen für die nächste Iteration

### 3.4.3 Audio Filter

#### Problem

Bereits bei der Inbetriebnahme fiel ein Fehlverhalten des Audio Filters durch den hohen Stromverbrauch der 12V Speisung auf. Es zeigte sich, dass das Filter bei deaktiviertem DAC ins Schwingen gerät. Dabei schafften auch unterschiedliche Einschaltsequenzen der Speisungen oder das Hinzufügen von Pull-Down Widerständen an den Eingängen keine Abhilfe. Durch weitere Tests und Trennen der einzelnen Filterstufen konnte der Fehler auf das Biasing des Sallen-Key Filters eingegrenzt werden.

Das Biasing wird verwendet, um die 2.5 V Offset des DAC zu entfernen, dies wird über die Rückkopplung R341 realisiert. Auf den single-ended Anschluss hat das Biasing keinen Einfluss. Beim Biasing handelt es sich um eine Eigenentwicklung, da der DAC Hersteller zum Entfernen des Offsets eine AC-Kopplung vorschlägt [3]. Das gewünschte Verhalten der Schaltung und deren Simulationen befinden sich in der Dokumentation der Projektstudie [1].

#### Lösungsvorschläge

Um das Problem zu lösen wurden empfohlene Audio Filter mit DC Kopplung dreier DAC Hersteller verglichen.

Das Filter von Analog Devices in Abbildung 3.6 zeigt einen Strom/Spannungswandler mit Tiefpass und einem nachfolgenden differenziell zu single-ended Wandler mit Tiefpass Filter. Beim AD1955 handelt es sich um einen DAC mit Strom-Ausgang. Da der DAC mit 5 V gespiesen wird, liegt auch bei diesem Signal (IOUT) ein Offset von 2.5 V an. Der Hersteller löst das Problem mit einem invertierenden Verstärker, bei dem der positive Eingang auf die Bias-Spannung angeschlossen ist (VBIAS). Das differenzielle Signal bei R31, R32 ist somit DC frei und könnte auf einen XLR Stecker geführt werden. Im Gegensatz zum Sallen-Key Filter, welches beim Audio-Streamer verwendet wird, ist die erste Filterstufe hier erster und nicht zweiter Ordnung.

Im Datenblatt von Burr Brown (Texas Instruments) wird für das differenzielle Signal zwei mal die Schaltung aus Abbildung 3.7 verwendet. Dadurch, dass für den Mono Modus jeweils beide Ausgänge zuerst zu einem single-ended Signal gewandelt und diese danach wieder zusammengeschaltet werden, wird der DC Anteil entfernt. Dazu müssen jedoch pro Audio Ausgang sieben Operationsverstärker verwendet werden.

Wolfson Mircodevices (Cirrus Logic) empfiehlt in Abbildung 3.8 ein Filter, das dem des Audio-Streamers sehr nahe kommt. Es werden drei Filter Stufen verwendet: ein RC-Tiefpass, ein Sallen-Key Tiefpass und ein differenziell zu single-ended Wandler mit Tiefpass. Dieser DAC hat einen Spannungsausgang, dessen Offset beträgt wiederum 2.5 V. Aus der Schaltung wird deutlich, dass dieser DC Anteil auf dem differenziellen Signal nicht entfernt wird.

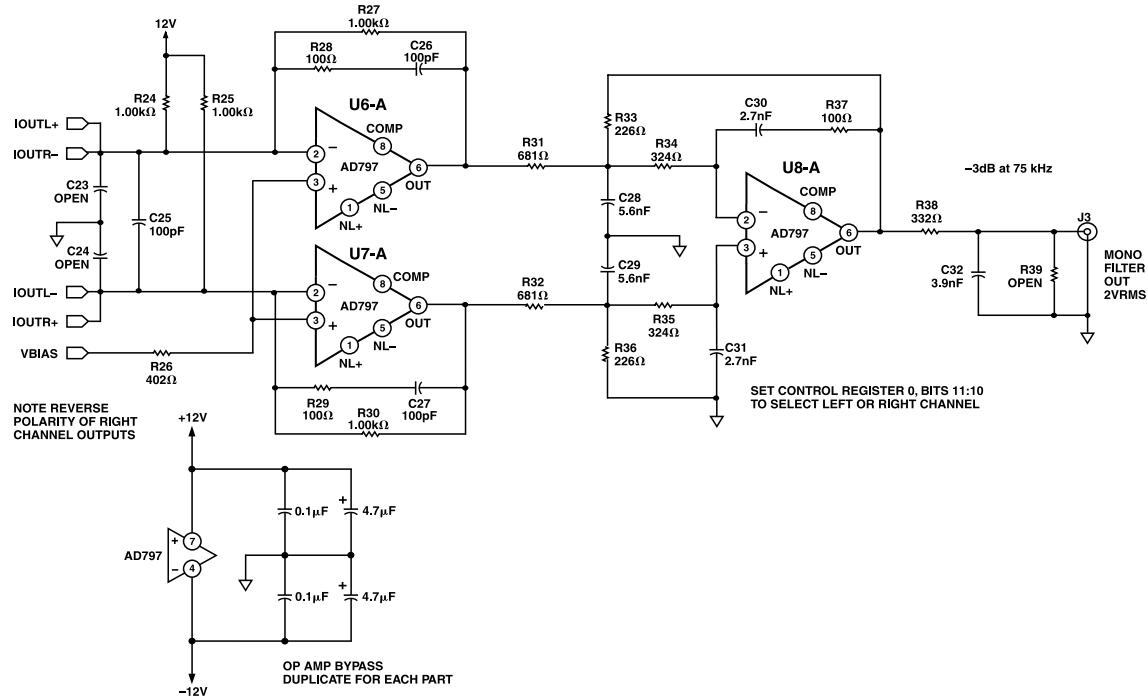


Abbildung 3.6: Audio-Filter Analog Devices AD1955 [4]

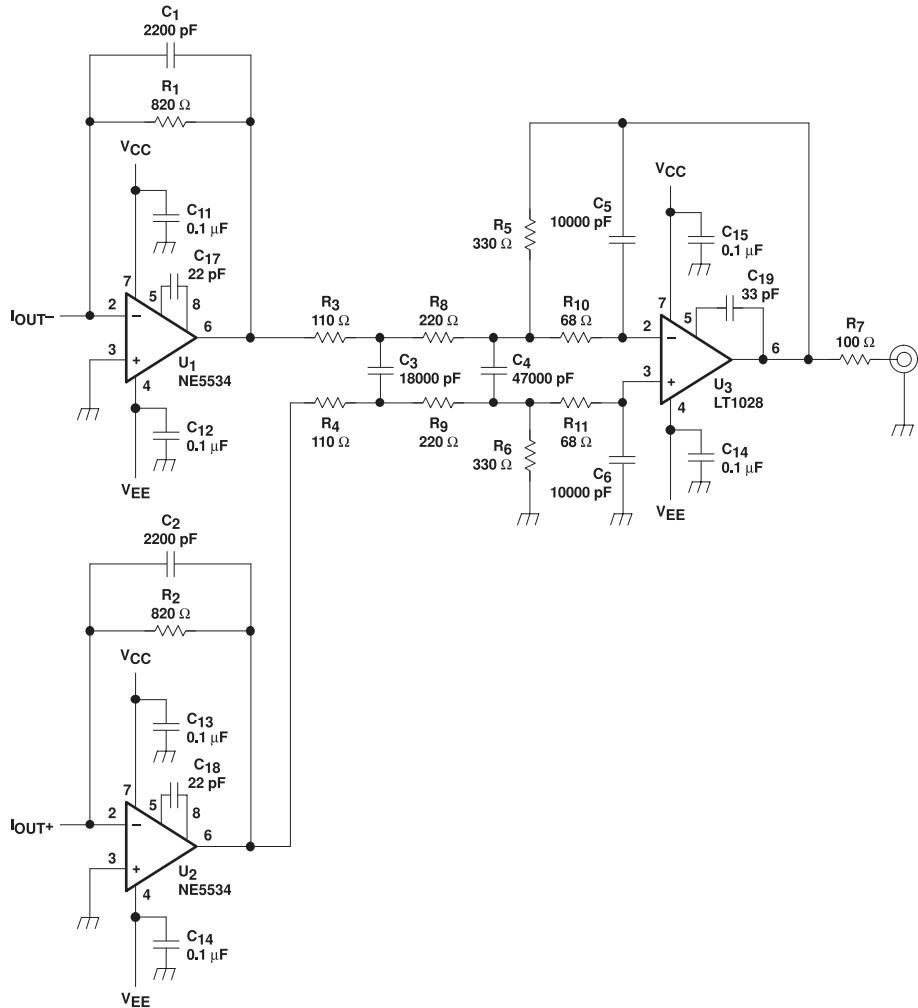


Abbildung 3.7: Audio-Filter Burr Brown (Texas Instruments) PCM1792A [6]

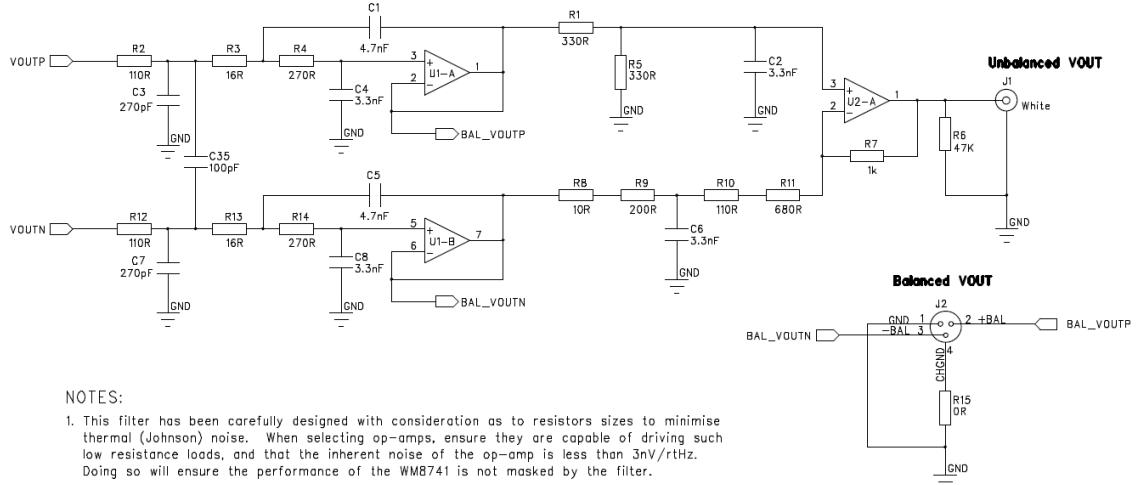


Abbildung 3.8: Audio-Filter Wolfson Microdevices (Cirrus Logic) WM8741 [5]

## Entscheidung

Die Schaltung auf dem Audio-Streamer wird an den dritten Vorschlag (WM8741) angepasst. Die vorzunehmenden Änderungen dazu sind minimal, es muss lediglich das Biasing entfernt werden. Mit dem Operationsverstärker vor dem single-ended Ausgang wird der DC Anteil entfernt. Der XLR Ausgang kann aber nicht mehr verwendet werden.

Langfristig ist die Schaltung von Analog Devices zu bevorzugen. Mit dem invertierenden Verstärker wird der Offset entfernt und da nur auf ein Eingang rückgekoppelt wird, ist die Stabilität höher als beim verwendeten Sallen-Key Filter.

### 3.4.4 Negative Spannungsversorgung

Da bei VR203 gleich drei Korrekturen notwendig waren, werden diese hier zusätzlich aufgeführt.

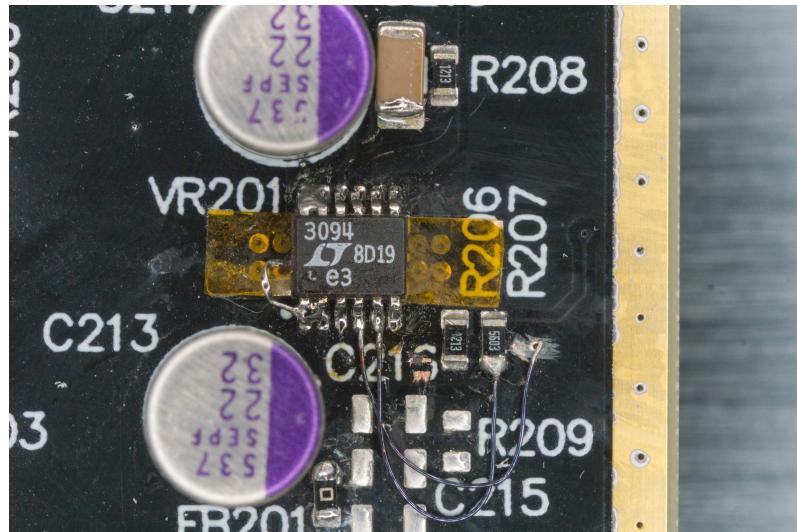


Abbildung 3.9: Korrekturen VR203

In Abbildung 3.9 ist zu erkennen, dass die PG und PG-FB Leitungen gekreuzt wurden (unten rechts). Dazu wurden die entsprechenden Leiterbahnen getrennt und anschliessend mit zwei Drähten neu verbunden.

Als weitere Korrektur wurden die Elektrolyt-Kondensatoren entgegen dem Bestückungsdruck verkehrt angelötet.

Das grösste Hindernis bildete das falsch angeschlossene Thermal-Pad des LT3094. Um diesen Fehler zu beheben wurde das IC entfernt. Das Thermal-Pad auf der Leiterplatte musste mit Kapton Klebeband isoliert werden. Um das Thermal-Pad des IC mit Pin 1 zu verbinden, muss ein Draht oder Kupferband angelötet werden. Das modifizierte IC konnte danach wieder eingelötet werden.

## 3.5 Gehäuse

In den folgenden Unterabschnitten werden die Details des Gehäuse Designs anhand Bilder aus dem CAD-Tool Autodesk Fusion 360 erläutert. Die Vermassungen befinden sich Anhang.

### 3.5.1 Front Panel



Abbildung 3.10: Renderansicht Front Panel (Ansicht: Audio-Streamer Rückseite)

Die Rückseite des Front-Panel ist in Abbildung 3.10 dargestellt. Darauf befinden sich von links nach rechts die Ausschnitte für den Drehgeber, das Display, den Schalter und die USB-Durchführung.

Da der Drehgeber auf einer separaten Leiterplatte montiert ist, werden vier Schrauben benötigt, um die Leiterplatte und die Plexiglas Platte zu befestigen.

Das Display kann über vier Schrauben befestigt werden, es erfordert zwei Besonderheiten:

1. Der Ausschnitt erlaubt nur sehr kleine Radien, da die Glasplatte des Display nahezu perfekt Rechteckig ist, d.h. die Ecken müssen nach dem Fräsen mit einer Feile nachgebessert werden
2. Der Befestigungsrahmen steht über das Display hinaus, es müssen dazu Stufen um den Ausschnitt gefräst werden

Der Spalt um das Display hat sich beim Zusammenbau als zu gross herausgestellt. Er ist deutlich zu erkennen und sollte bei einer Weiterentwicklung verkleinert werden.

Der Power-Button befindet sich oben rechts und wird mit einer Mutter befestigt. Für die USB-Durchführung muss eine Tasche gepräst werden. Dadurch wird nur der kreisrunde Ausschnitt der Durchführung sichtbar. Um diese befestigen zu können muss eine Adapter aus Blech und Distanzhülsen verwendet werden. Dieses Blech kann mit vier Schrauben am Front Panel befestigt werden.

Bei den Gewinde handelt es sich um Sacklöcher. Somit sind von der Vorderseite keine Schrauben sichtbar.

Das Front Panel wurde von der Maschinenbau Abteilung der BFH bearbeitet und danach zur Erhöhung Oberflächenhärte und der optischen Aufwertung schwarz anodisiert.

### 3.5.2 Back Panel

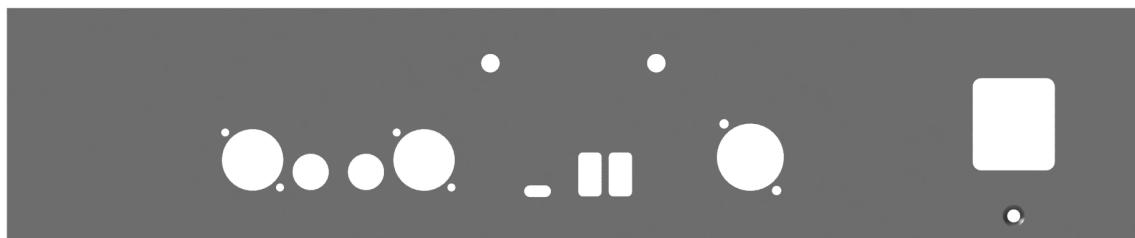


Abbildung 3.11: Renderansicht Back Panel (Ansicht: Audio-Streamer Rückseite)

Im Back Panel befinden sich die Ausschnitte für die Anschlüsse des Mainboards und der Speisung. Dies ist in Abbildung 3.11 dargestellt. Von links nach rechts befinden sich die Löcher für die Audioausgänge (jeweils XLR und RCA), sowie die Mirco USB 2.0 Buchse, die USB Buchsen, die Ethernet Durchführung und die Gerätebuchse. Die zwei Bohrungen oben in der Mitte sind für die Antennen des WiFi/Bluetooth Module vorgesehen. Die Ausschnitte richten sich nach den Angaben in den Datenblättern der jeweiligen Hersteller.

Unterhalb der Gerätebuchse ist eine Erdungsschraube vorgesehen. Beim Verbinden der Erdung ist darauf zu achten, dass anodisierte Schicht abgeschliffen und eine Fächerscheibe zwischen das Panel und den Kabelschuh angebracht wird. Dadurch wird der Kontakt zwischen dem Panel und dem Erdleiter sichergestellt.

### 3.5.3 Base Plate



Abbildung 3.12: Renderansicht Base Plate (Ansicht: Audio-Streamer Top)

Die Leiterplatten werden auf der Base Plate befestigt. Dazu wurden wie in Abbildung 3.12 dargestellt die Bohrungen der Leiterplatten auf die Base Plate übernommen. Unter dem Mainboard

befinden sich zwei zusätzliche Bohrungen: eine unter dem CoM Sockel und eine neben der Ethernet Buchse. Somit können zusätzliche Distanzhalter unter dem Mainboard befestigt werden, um die Stabilität auf Druck zu erhöhen. In den vier Ecken werden die Füsse an das Gehäuse befestigt.

Unten rechts befindet sich die Bohrung für die Erdungsschraube der Base Plate. Es sind nur Erdungsschrauben auf dem Back Panel und der Base Plate vorgesehen, da dies die einzigen Platten sind, die in Kontakt mit einem Netzleiter kommen können.

Es werden Senkkopfschrauben werden, um eine plane Unterseite beizubehalten.

### 3.6 Verdrahtung

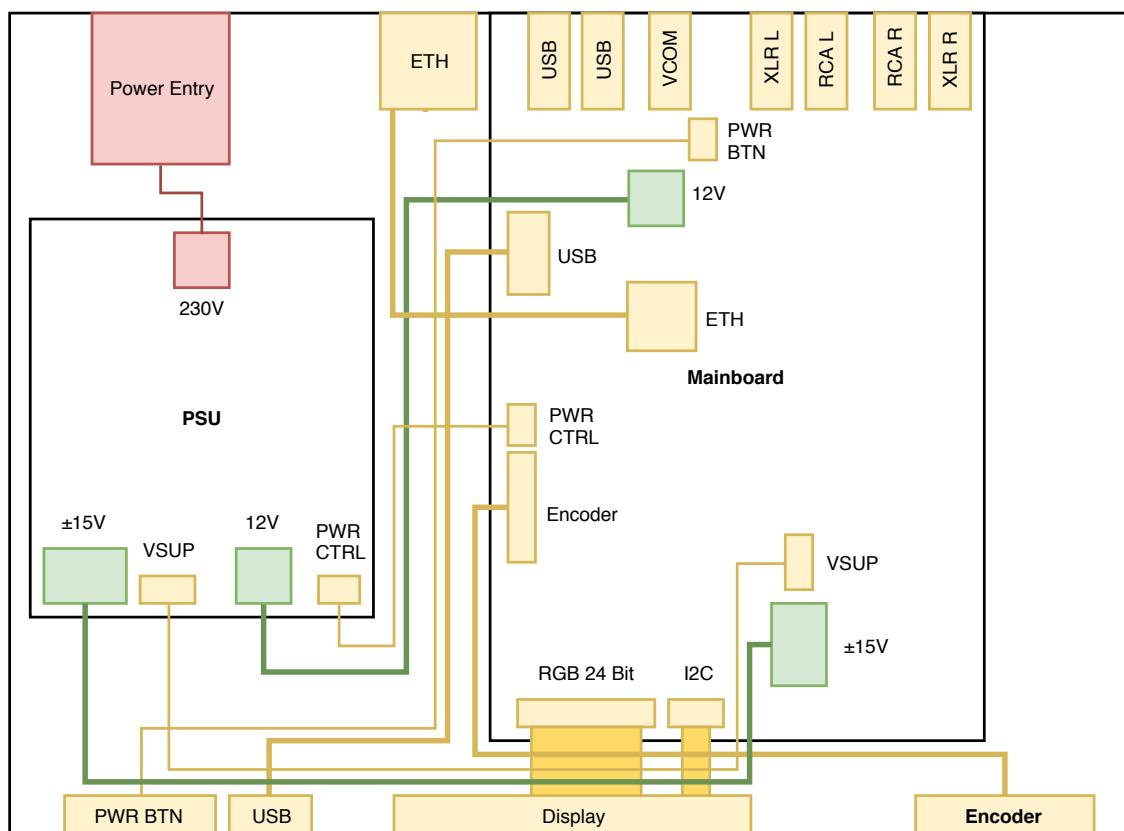


Abbildung 3.13: Verdrahtungsplan, rot: Netzspannung, grün: Spannungsversorgung, gelb: Signale

Der Verdrahtungsplan in Abbildung 3.13 zeigt die internen Leitungen und die externen Schnittstellen des Audio-Streamers. Die Positionen und Größen der Komponenten entspricht näherungsweise der physischen Platzierung. Kabel mit vielen oder dicken Leitern sind als dicke Linie dargestellt.

Die Trennung zwischen der Netzspannung (rot) und den anderen Leitungen ist deutlich zu erkennen. Durch die physische Trennung der Netzspannung von den anderen Leitungen wird ein Kurzschluss auf Signalleitungen verhindert.

Die Leitungen der ±15 V-Spannungsversorgung von der PSU zum Mainboard müssen möglichst getrennt von den digitalen Signalleitern verlegt werden. Zudem sollen die drei Leiter verdrillt werden, um das Einkoppeln von Störungen zu minimieren.

Der schematische Verdrahtungsplan mit den Bezeichnungen der verwendeten Kabel befindet sich Anhang.

### 3.7 Endmontage



Abbildung 3.14: Verdrahtung des Audio-Streamers

Abbildung 3.14 zeigt die physikalischen Verbindungen gemäss dem Verdrahtungsplan in Abbildung 3.13. Zudem sind oben links die Erdverbindungen ersichtlich. Der Erdleiter verbindet die Power Entry Modul mit der Rückplatte und der Base Plate. Somit sind alle metallischen Gegenstände geerdet, welche in der Nähe der Netzleiter sind, geerdet. Damit der Kontakt zur Alumini umplatte sichergestellt ist, wurde die Eloxalschicht entfernt und Fächerscheiben verwendet.

Weiter ist zu erkennen, dass die Leiter der analogen Speisung verdrillt und möglichst getrennt von den ungeschirmten digitalen Signalen des Displays geführt wurden. Die Platzierung wurde mit Kabelbinder und Kabelbindehaltern fixiert.



Abbildung 3.15: Montage der Bedienelemente

Um die Bedienelemente am Front Panel in Abbildung 3.15 zu befestigen, wurde eine Kombination aus Schrauben, Unterlegscheiben und Distanzhülsen verwendet. Als Referenz zur Herstellung weiterer Geräte dient der Audio-Streamer Prototyp. Um die USB-Durchführung zu montieren, wird ein Adapter aus Aluminium Blech benötigt. Die Durchführung wird zuerst mit Distanzhülsen an den Adapter befestigt, dieser wird anschliessend an das Front Panel geschraubt.

## 3.8 Kostenübersicht

Anhand der verwendeten Komponenten können die Gesamtkosten des Audio-Streamers berechnet werden.

Bezeichnung	Kosten / Fr.
Mainboard Komponenten (DAC)	ca. 170.-
Mainboard Komponenten (CoM)	ca. 110.-
Mainboard PCB	70.-
CoM	125.-
PSU Komponenten	64.-
PSU PCB	33.-
Display	65.-
Gehäuse Durchführungen	58.-
Gehäuse	65.-
<b>Basis</b>	<b>760.-</b>
Encoder Komponenten	14.-
Encoder PCB	15.-
<b>Encoder Option</b>	<b>29.-</b>
WiFi Module & Antennen	65.-
SSD (500GB)	65.-
<b>Upgrade Option</b>	<b>130.-</b>
<b>Total</b>	<b>919.-</b>

Tabelle 3.2: Kostenübersicht Prototyp (1 Stk.)

Aus Tabelle 3.2 geht hervor, dass das Mainboard mit CoM bereits über die Hälfte der Kosten des Basis Geräts beansprucht. Auffallend sind auch die hohen Preise für die Leiterplatten. Diese sind durch die geringe Bestellmenge bedingt. Es ist zudem zu beachten, dass es sich bei der Liste um Materialkosten handelt. Zeitintensive Arbeiten, wie das Bestücken der Leiterplatten oder die Bearbeitung des Gehäuse sind nicht eingerechnet.

Stückzahl	Kosten / Fr.
1	789.-
10	701.-
100	558.-

Tabelle 3.3: Materialkosten inkl. Encoder Option bei unterschiedlichen Produktionsmengen

Die Materialkosten sinken, wie in Tabelle 3.3 aufgelistet, mit den Skaleneffekten deutlich. Dies ist unter anderem auf die tieferen Herstellungskosten der Leiterplatten ab zehn Stück bedingt.

Kosteneinsparungen wären mit einer Reduktion der Funktionalität möglich. Das grösste Potential zeigt hier das CoM, welches mit Beschaltung 235.- kostet. Die Verwendung eines günstigeren Einplatinencomputers, wie einem Raspberry Pi, könnten zu starken Kostenreduktionen führen, jedoch auf Kosten etlicher Funktionen, wie WiFi über PCIe.

# 4 Linux System & Treiber

## 4.1 Überblick

In diesem Kapitel werden die Anforderungen, welche das Konzept des Audio-Streamers an das Linux-Betriebssystem stellt behandelt. Um die Anforderungen zu erfüllen, ist vor allem das Anpassen des Devices-Trees, der Kernelkonfiguration und teilweise auch der Treiber erforderlich.

Um den Einstieg in die Thematik zu erleichtern, werden im folgenden Abschnitt 4.2 wichtige Begriffe und Zusammenhänge erklärt. Da der Grossteil der Zeit dieser Thesis in die Anpassung von Treibern einfließt, wird der Fokus in diesem einführenden Kapitel auf die Linux Treiber und deren Zusammenhang mit dem Device-Tree gelegt. Weiter wird das Debuggen von Treibern und der indirekte Zugriff auf die Hardware behandelt.

In den darauf folgenden Kapiteln werden die ausgeführten Arbeitsschritte dokumentiert. Konkret wird für folgende Komponenten des Audio-Streamers Entwicklungsarbeit im Bereich Linux geleistet:

- Display
- Display Backlight
- Touchscreen
- Digital Analog Converter (DAC)
- GPIOs für das Power-Management
- Drehgeber

## 4.2 Begriffe und Zusammenhänge

### 4.2.1 Zweck und Funktion eines Linux Treibers

Ein sogenannter Treiber dient der Erweiterung eines Betriebssystem Kernels. Je nach Art des Kernels (monolithisch oder Microkernel) erfolgt das Laden eines Treibers anders. Beim monolithischen Kernel wird der Treiber direkt in den Kernel kompiliert. Wohingegen beim Microkernel der Treiber als Modul zur Laufzeit geladen wird. Unter Linux sind beide Varianten möglich. Dies erklärt auch gleich den Unterschied zwischen Treiber und Modul. Bei einem Modul handelt es sich um einen Treiber der zur Laufzeit geladen werden kann. Der Unterschied liegt also in der Konfiguration des Kernels vor dem Kompilieren und nicht im Code des Treibers, dieser ist in beiden Fällen identisch.

Für den Audio-Streamer soll die Möglichkeit genutzt werden, Treiber als Module zur Laufzeit zu laden. Dies erhöht die Flexibilität und die Transparenz des Systems. So können aus der Konsole zur Laufzeit zusätzliche Module geladen oder entladen werden. Beim Laden werden auch gleich allfällige Fehlermeldungen ausgegeben, was das System transparenter macht.

Der Aufbau eines Treibers unterscheidet sich von dem eines normalen C-Programms. Ein Linux-Treiber beinhaltet weder eine Main Funktion, noch wird dieser sequentiell ausgeführt. Dafür registriert sich ein Treiber über ein standardisiertes Interface beim Kernel. Weiter kann in einem Treiber nicht auf die Standard C Bibliotheken zugegriffen werden. Es kann ausschliesslich auf Bibliotheken im sogenannte Kernel Space zugegriffen werden. Die Standard C Bibliotheken befinden sich im sogenannten User Space.

Im Linux Betriebssystem wird zwischen User Space und Kernel Space unterschieden. Die Unterscheidung wird aufgrund des unterschiedlichen Adressbereichs gemacht. Während im User Space mit virtuellen Adressen gearbeitet wird, verwendet der Kernel Space physikalische Adressen. Die

Memory Management Unit (MMU) ist dabei für die Umwandlung der Adressen zuständig. Anwendungen laufen dabei ausschließlich im User Space. Der direkte Zugriff auf die Hardware kann nur aus dem Kernel Space erfolgen.

Allerdings gibt es für einfache Hardwarezugriffe, wie das Setzen von einem GPIO oder das Konfigurieren einer PWM-Peripherie das Filesystem Sysfs. Das Sysfs wird in Unterabschnitt 4.2.5 beschrieben. Mit diesem ist es möglich indirekt aus dem User Space auf die Hardware zuzugreifen. Wird die Hardware aber komplexer ist ein direkter Zugriff aus dem Kernel Space durch einen Treiber unerlässlich. Wichtig zu beachten ist, dass aus dem User Space durch einen Fehler kein Systemabsturz herbeigeführt werden kann, da der irrtümliche Zugriff auf falsche Speicherbereiche durch die MMU verhindert wird. Aus dem Kernel Space wird der Zugriff nicht mehr geschützt. Das heißt das System kann durch einen falschen Speicherzugriff zum Absturz gebracht werden. Daher ist bei der Entwicklung von Treibern besondere Aufmerksamkeit geboten.

## 4.2.2 Linux Treiber Debuggen

Das Debuggen im Kernel Space ist ein komplexer Vorgang und ist sehr viel schwieriger als das Debuggen von User Space Applikationen. Zum Debuggen eines Kernel Moduls gibt es mehrere Möglichkeiten:

- Printing
- Querying
- Watching
- System Faults
- Debugging mit GDB

Hier werden die unterschiedlichen Methoden kurz beschrieben und gezeigt welche Variante für das Debuggen des DAC Treibers angewendet wird.

**Printing** ist die einfachste Methode zum Debuggen. Dabei werden Meldungen in den Source Code des Treiber eingefügt, die bei der Ausführung des Treibers ausgegeben werden. Die verwendete Funktion `printf()` ist analog zur Funktion `printf()` in C. Der einzige Unterschied ist die Angabe eines Loglevel Strings vor dem eigentlichen String. Damit ist es möglich zur Laufzeit, durch einstellen des Loglevels, zu bestimmen welche Meldungen ausgegeben werden sollen. Ein Beispiel sieht folgendermassen aus:

```
1 printk(KERN_DEBUG "Here I am: %s:%i\n", __FILE__, __LINE__);
```

Listing 4.1: Beispiel einer `printk` Meldung mit dem Loglevel Debug

Das Loglevel kann zur Laufzeit mit folgendem Shell Befehl angepasst werden:

```
1 echo 8 > /proc/sys/kernel/printk
```

Listing 4.2: Shell Befehl zur Anpassung des Loglevels

**Querying** ist das Abfragen von Systeminformationen bei Bedarf. Diese Methode hat den Vorteil, dass die Systemperformance nicht beeinflusst wird. Informationen können zum Beispiel aus dem Sysfs (Unterabschnitt 4.2.5) oder dem \proc Filesystem ausgelesen werden. Das \proc Filesystem hat den Zweck Informationen aus dem Kernel Space für den User Space sichtbar zu machen.

**Watching** ist eine Methode, die das Verhalten eines Treibers aus dem User Space umfasst. Dabei wird im User Space eine Applikation mithilfe eines Debuggers durchgesteckt, welche auf den Treiber zugreift. Aus dem Verhalten der User Space Applikation kann auf das Verhalten des Treibers geschlossen werden.

**System Faults** treten bei der Ausführung eines Treibers auf. Mögliche Ursachen sind falsche Speicherzugriffe oder fehlender Speicher beim Allozieren. Anhand der Informationen, die das System ausgibt, kann auf einen Fehler im Treiber geschlossen werden.

**Debugging mit GDB** ermöglicht das Durchstecken durch den Kernel Code. Diese Variante sollte als letzte aller Varianten angewendet werden. Denn sie ist sehr zeitaufwändig und gibt oft nur wenig Aufschluss über die Funktion des Gesamtsystems.

Zum Debuggen des DAC Treibers für den AK4493 werden die Methoden Printing und System Faults angewendet. Zur Anpassung der Master Clock wird Querying angewendet. Aus dem \proc Filesystem können Informationen über den Clock-Tree bezogen werden. [7]

#### 4.2.3 Plattform-Treiber

Sogenannte Plattform-Treiber sind eine spezielle Form von Treibern, die vor allem im Zusammenhang mit Embedded Systemen verwendet werden. Der Grund dafür ist, dass die Hardware bei Embedded Systemen oft nicht detektiert werden kann. Im Desktop PC Bereich sind alle Komponenten über einen Bus detektierbar. Aus diesem Grund können die richtigen Treiber beim Starten des Systems automatisch geladen werden. Bei Embedded Systemen unter Linux wird zur Detektion der Hardware der Device-Tree verwendet. Die detektierten Devices aus dem Device-Tree werden im Linux System an den Plattform Bus angeschlossen. Alle Devices sind nach dem Booten im Filesystem unter /sys/bus/platform/devices zu finden.

Beim Plattform Bus handelt es sich um einen virtuellen Bus unter Linux, welcher alle nicht detektierbaren Geräte verbindet. Oft handelt es sich bei diesen Geräten um Peripherie Bausteine eines SoC, welche direkt an den CPU Bus angeschlossen sind. Dies ist allerdings nicht immer der Fall. So werden auch Geräte am I2C Bus an diesen virtuellen Bus angeschlossen. Der Vorteil dieses Systems ist, dass nach der Registrierung im Device-Tree unter Verwendung des passenden Treibers direkt auf die Register der Geräte zugegriffen werden kann, egal um welchen Typ Gerät es sich handelt. [8]

Die meisten für den Audio-Streamer eingesetzten Treiber sind Plattform-Treiber. So auch die Treiber für das Display und den DAC, die im Rahmen dieser Arbeit angepasst werden.

#### 4.2.4 Zusammenhang zwischen Treiber und Device-Tree

Die Funktion und der Grundaufbau des Device-Trees wurde bereits in der Dokumentation der Vorstudie beschrieben. Der komplexe Teil liegt allerdings im Zusammenspiel zwischen Device-Tree und Treiber. Dazu wird im Device-Tree das Property compatible verwendet. Mit diesem wird durch einen String der Form "Hersteller,Modell" spezifiziert, um was für ein Device es sich handelt. Über den Kernel Namespace wird beim Booten der entsprechende Treiber für das Device gesucht und geladen. Wobei in jedem Treiber eine sogenannte Match Table registriert wird, welche alle unterstützten Devices enthält. Es muss insbesondere beachtet werden, dass der String im Device-Tree exakt mit dem String im Treiber übereinstimmt. Natürlich muss der entsprechende Treiber vor dem Kompilieren des Kernels in der Kernelkonfiguration aktiviert werden.

In einem Node des Device-Trees werden zusätzlich noch weitere Informationen angegeben. Diese Informationen werden beim Laden des Treibers durch den Treiber gelesen. Die Idee dahinter ist, dass keine fertigen Treiber angepasst werden müssen. Die Konfiguration eines Treibers soll vollständig über den Device-Tree erfolgen. Da das Konzept des Device-Trees unter Linux aber noch relativ neu ist, kann noch nicht jeder Treiber vollständig durch den Device-Tree konfiguriert werden. Daher müssen teilweise Anpassungen im Treiber selbst vorgenommen werden. Dies gilt auch für den Audio-Streamer. [9]

Als Grundlage zur Erstellung eines Device-Tree Nodes dient die Dokumentation des Treibers. Diese ist für jeden Treiber unter dem Pfad Documentation/devicetree/bindings im Kernel verfügbar. Dort werden für jeden Treiber die entsprechenden Properties kurz beschrieben. Dabei wird oft zwischen notwendigen und zusätzlichen Properties unterschieden.

#### 4.2.5 Sysfs

Das Sysfs ist ein RAM basiertes virtuelles Filesystem unter Linux. Mit seiner Struktur repräsentiert es die Device Architektur des Kernels. Darin abgelegt sind auch die aktuellen Informationen der Hardware. Der grosse Vorteil dieses System ist der Zugriff auf Hardware aus dem User Space. Denn der Zugriff auf Hardware ist normalerweise nur aus dem Kernel Space möglich. [10]

Da mit dem Sysfs auf die Hardware zugegriffen werden kann, ist bei einem falschen Zugriff ein Systemabsturz möglich. Aus diesem Grund wird der Zugriff auf die Devices im Sysfs geschützt.

Soll als gewöhnlicher User auf ein solches Device zugegriffen werden, muss der User zur Gruppe hinzugefügt werden, zu der das Device gehört. Zudem muss der Modus des Files im Sysfs durch eine sogenannte udev Rule angepasst werden. Eine solche udev Rule sieht folgendermassen aus:

```
SUBSYSTEM=="gpio", KERNEL=="gpiochip*", ACTION=="add", PROGRAM="/bin/sh -c 'chown root:
    ↵ dialout /sys/class/gpio/export
    ↵ /sys/class/gpio/unexport ; chmod 220 /sys/class/gpio/export /sys/class/gpio/unexport'"
SUBSYSTEM=="gpio", KERNEL=="gpio*", ACTION=="add", PROGRAM="/bin/sh -c 'chown root:
    ↵ dialout /sys%p/active_low /sys%p/direction /sys
    ↵ %p/edge /sys%p/value ; chmod 660 /sys%p/active_low /sys%p/direction /sys%p/edge /sys%p/
    ↵ value'"
```

Listing 4.3: udev Rule für den Zugriff auf die GPIOs

Diese udev Rule erlaubt nun den Zugriff auf die GPIOs aus dem User Space. Folgende Befehlsfolge zeigt das Exportieren, das Setzen der Richtung als Output und das Setzen des Wertes auf High für das GPIO 124:

```
echo 124 > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio124/direction
echo 1 > /sys/class/gpio/gpio124/value
```

Listing 4.4: Zugriff auf das GPIO 124 aus dem User Space über das Sysfs

Die GPIO Nummer im Sysfs kann über die Bezeichnung des GPIOs im Datenblatt berechnet werden. Ein Beispiel einer Bezeichnung ist: GPIO4\_I028. Mit folgender Formel kann die Nummer berechnet werden.

$$\text{number}_{\text{Sysfs}} = 32 \cdot (\text{controller}_{\text{GPIO}} - 1) + \text{number}_{\text{IO}} \quad (4.1)$$

Für das Beispiel ergibt sich:

$$\text{number}_{\text{Sysfs}} = 32 \cdot (4 - 1) + 28 = 124 \quad (4.2)$$

Damit handelt es sich um das GPIO das mit den Befehlen in Listing 4.4 gesetzt wurde.

#### 4.2.6 Die IOMUX Peripherie des iMX6

Die IOMUX Peripherie wird für alle nachfolgenden Komponenten benötigt, ohne direkt etwas mit deren Funktion zu tun zu haben. Aus diesem Grund wird die Hardware des IOMUX und deren Konfiguration in diesem separaten Kapitel dokumentiert. Die IOMUX Peripherie ist insbesondere in der Entwicklungsphase wichtig, in welcher das System vom Ixora Carrier Board auf die Hardware des Audio-Streamers portiert wird. Die Portierung kann durch alleinige Umkonfiguration der IOMUX Peripherie vorgenommen werden.

Der iMX6 hat viel mehr interne Signale als Pins. Darum werden Multiplexer benötigt, welche die gewünschten Signale auf die richtigen Pins leiteten. Diese Multiplexer werden im iMX6 durch die IOMUX Peripherie zusammengefasst. Jeder Pin besitzt seinen eigenen Multiplexer. Durch dessen Konfiguration kann gewählt werden, mit welchem Signal der Pin verbunden werden soll. Beim iMX6 stehen in der Regel pro Pin fünf Signale zur Auswahl. Zusätzlich kann für jeden Pin der Ausgang ähnlich wie bei einem GPIO konfiguriert werden.

Um die Nutzung der IOMUX Peripherie unter Linux zu erleichtern, wird von Freescale ein Treiber zur Verfügung gestellt. Dieser ermöglicht es die gesamte Pin Konfiguration im Device-Tree vorzunehmen. Dazu wird der Node `iomux` im Device-Tree erstellt.

```
iomuxc: iomuxc@020e0000 {
    patable = "fsl,imx6dl-iomuxc", "fsl,imx6q-iomuxc";
    reg = <0x020e0000 0x4000>;
};
```

Listing 4.5: Device-Tree Eintrag IOMUX Peripherie

Nun kann in jeder Datei im Device-Tree mit &iomux auf diesen Node referenziert werden. Soll nun eine Pinkonfiguration für eine Peripherie erstellt werden, wird ein Node mit dem Namen der Peripherie erzeugt. Darin kann nun durch Makros der Form PIN\_Funktion die Konfiguration der Multiplexer vorgenommen werden. Die Makros sind im File `imx6q-pinfuc.h` definiert. Hinter dem Makro wird der Wert angegeben, welcher in das Register für die Konfiguration des Ausgangs geschrieben werden soll. Hier kann zum Beispiel das Zu- und Wegschalten von Pull-Up oder Pull-Down Widerständen realisiert werden. Hier ein Beispiel für die Peripherie I2C-1. [11]

```
pinctrl_i2c1: i2c1grp {
    fsl,pins = <
        MX6QDL_PAD_CSI0_DAT8_I2C1_SDA 0x4001b8b1
        MX6QDL_PAD_CSI0_DAT9_I2C1_SCL 0x4001b8b1
    >;
};
```

Listing 4.6: Device-Tree Eintrag für die I2C-1 Pins

Wichtig ist, dass der Device-Tree selbst keine Registerkonfiguration vornehmen kann. Dies wird ausschliesslich durch den Treiber übernommen. Dieser parst die Angaben aus dem Device-Tree und konfiguriert den IOMUX entsprechend.

## 4.3 Display Treiber

Das Display wurde bereits in der Vorstudie in Betrieb genommen und die nötigen Arbeitsschritte dokumentiert. Allerdings ist die Dokumentation der Vorstudie bezüglich der Beziehung zwischen Device-Tree und Treiber nicht ganz komplett. Deshalb wird dieses Unterkapitel der Vollständigkeit halber auch in die Dokumentation der Thesis eingefügt und ergänzt.

Der Display Treiber wird zur Verwendung der Display Peripherie des iMX6 benötigt. Der Treiber sorgt dafür, dass die Daten aus dem Framebuffer der GPU mithilfe der Display Peripherie über die parallele LCD Schnittstelle ausgegeben werden. Um den Treiber zu verwenden muss ein Eintrag im Device-Tree erstellt werden.

```
lcd: lcd@0 {
    compatible = "fsl,lcd";
    ipu_id = <0>;
    disp_id = <1>;
    default_ifmt = "RGB24";
    pinctrl-names = "default";
    pinctrl-0 = <&pinctrl_ipu1_lcdif>;
    status = "disabled";
};
```

Listing 4.7: Device-Tree Eintrag Display Treiber

Hier ist vor allem das Property `pinctrl-0` wichtig. Darin werden alle nötigen Pins für die parallele LCD Schnittstelle angegeben. Da der Node `pinctrl_ipu1_lcdif` teil des Nodes `iomux` ist, werden die Pins auch gleich auf die Funktion für die LCD Schnittstelle konfiguriert. Dies erfolgt wie in Unterabschnitt 4.2.6 beschrieben.

Das eingesetzte parallele Display Interface bietet keine Device Erkennung über I2C, wie dies bei HDMI der Fall ist. Aus diesem Grund müssen die Display Timings manuell konfiguriert werden. Die Timings sind für die Korrekte Darstellung des Bildes sehr wichtig. Die Pixel Clock taktet dabei die Übertragung der einzelnen Bits, daher liegt sie auch im Megahertz Bereich. Die Bildwiederholrate bestimmt mit welcher Frequenz das gesamte Display einmal neu beschrieben wird. Zusätzlich zu den Frequenzen werden auch Angeben zur Grösse der nicht sichtbaren Randbereiche benötigt. Dies werden in Anzahl Pixel angegeben. In Abbildung 4.1 sind die verschiedenen Randbereiche dargestellt.

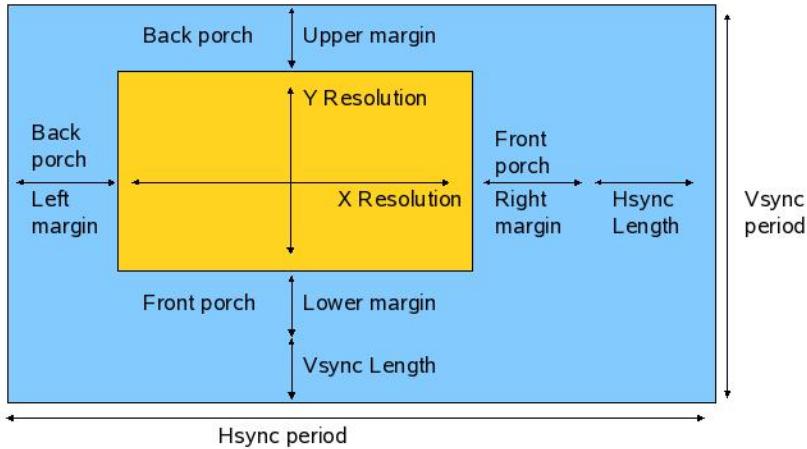


Abbildung 4.1: Übersicht über die LCD Timings [12]

Die Timings können dem Datenblatt des Displays entnommen werden [13]. Der Treiber für die parallele Display Schnittstelle bietet keine Möglichkeit die Timing Konfiguration über den Device-Tree vorzunehmen. Die Timings werden daher direkt im Treiber eingetragen. Anschliessend kann durch setzen einer Umgebungsvariable in U-Boot die richtige Display Konfiguration ausgewählt werden. Dies ist auch die empfohlene Vorgehensweise von Toradex [14]. Der erstellte Eintrag im Treiber `mxc_lcdif.c` unter `video/fbdev/mxc` im Linux Kernel sieht folgendermassen aus:

```
static struct fb_videomode lcdif_modedb[] = {
    /* 800x480 @ 65 Hz , pixel clk @ 32 MHz */ /* 31250 ps*/
    "Newhaven", 65, 800, 480, 31250,
    .left_margin = 40,
    .right_margin = 128,
    .hsync_len = 48,
    .upper_margin = 29,
    .lower_margin = 45,
    .vsync_len = 3,
    .sync = 0,
    .vmode = FB_VMODE_NONINTERLACED,
    .flag = 0,},
}
```

Listing 4.8: Display Treiber Eintrag für das Newhaven Display

Das File `uEnv.txt` enthält die Umgebungsvariablen für das U-Boot. Diese werden beim Booten aus dem File gelesen und gesetzt. Das File befindet sich im Image für den Audio-Streamer. Darin wird nun der Eintrag für das Display angepasst:

```
vidargs=mxc_hdmi.only_cea=0 video=mxcfb0:dev= lcd,Newhaven,if=RGB24 video=mxcfb1:off
    ↳ video=mxcfb2:off video=mxcfb3:off fbmem=32M
```

Listing 4.9: U-Boot Umgebungsvariable zur Auswahl des Displays

Der Name des Displays ("Newhaven") muss dabei mit dem Namen im Treiber übereinstimmen.

## 4.4 Backlight Treiber

Der Backlight Treiber wird zur Ansteuerung der Hintergrundbeleuchtung des Displays verwendet. Damit kann das Backlight bei Bedarf an die Umgebungshelligkeit angepasst werden. Die Ansteuerung des Backlights erfolgt mittels PWM. Für den Audio-Streamer wird der generische Backlight Treiber im Linux Kernel eingesetzt. Dieser Treiber erweitert den generischen PWM Treiber um ein Array an vorkonfigurierbaren Helligkeitswerten und um die Möglichkeit einen initialen Wert beim Systemstart vorzugeben.

Das Array mit den Helligkeitswerten hat den Vorteil, dass Werte vorgegeben werden können, die für das Helligkeitsempfinden des menschlichen Auges ungefähr linear erscheinen. Mithilfe der logistischen Funktion aus der Mathematik werden 25 Werte berechnet die der Form einer S-Kurve folgen. Die Werte sind im untenstehenden Listing unter dem Property `brightness-level` aufgeführt. Zu beachten ist, dass hier keine präzisen Berechnungen unter der Berücksichtigung der Helligkeitsempfindung des menschlichen Auges durchgeführt wurden. Es handelt sich lediglich um eine Schätzung.

Für die Verwendung wird der Backlight Treiber in der Kernelkonfiguration aktiviert und anschließend ein Node im Device-Tree erzeugt. In diesem Fall wurde bereits eine Konfiguration durch Toradex vorgenommen [15]. Nur das Array mit den Helligkeitswerten und der initiale Wert muss für den Audio-Streamer angepasst werden. Der fertige Node sieht folgendermassen aus:

```
&backlight {
#ifndef /* PWM polarity: if 1 is brightest */
pwms = <& pwm4 0 5000000 0>;
#endif /* Fusion 7 needs 10kHz PWM frequency */
pwms = <& pwm4 0 100000 0>;
#endif
#ifndef /* PWM polarity: if 0 is brightest */
pwms = <& pwm4 0 5000000 1>;
#endif
/* 26 values of Logistic function */
brightness-levels = <0 1 2 3 4 7 11 17 26 39 57 81 109 140 168 193 213 227 237 243 248
    ↳ 250 252 253 254 255>;
default-brightness-level = <20>;
status = "okay";
};
```

Listing 4.10: Device-Tree Eintrag Backlight

Über das Sysfs im Filesystem kann der Helligkeitswert zur Laufzeit aus dem User Space verändert werden. Dies geschieht mit folgendem Befehl:

```
sudo su
echo 25 > /sys/class/backlight/pwm-backlight/brightness
exit
```

Listing 4.11: Shell Befehl für die Anpassung der Backlight Helligkeit

Der Befehl `sudo su` versetzt den User in die Rolle des Users Root. Dies ist notwendig um die nötige Berechtigung für die Anpassung des Helligkeitswertes zu haben.

## 4.5 Touchscreen Treiber

Die Interaktion des Audio-Streamers mit dem Benutzer erfolgt in erster Linie über den Touchscreen. Um dies zu ermöglichen wird für das verwendete Display der passende Treiber benötigt. Der Treiber ist bereits in der verwendeten Kernel Version vorhanden. Daher kann er bei der Konfiguration des Kernels ausgewählt werden. Um den Treiber auch verwenden zu können, muss ein Eintrag im Device-Tree erstellt werden. Damit werden dem Treiber, alle nötigen Informationen zum Display übergeben. Da es sich beim Touchscreen um ein I2C-Device handelt, wird der Eintrag unter dem Node `i2c1` angelegt. Da das Label `i2c1` bereits an höherer Stelle im Device-Tree definiert wurde, wird mit & darauf referenziert. Der erstellte Device-Tree Node ist in folgendem Listing dargestellt:

```
&i2c1 {
/* Focaltech Touch Controller Newhaven Display */
focaltech@38{
    compatible = "focaltech,fts";
    reg = <0x38>;
    interrupt-parent = <& gpio6>;
    interrupts = <10 IRQ_TYPE_EDGE_FALLING>;
    focaltech,reset-gpio = <& gpio6 9 GPIO_ACTIVE_HIGH>;
    focaltech,irq-gpio = <& gpio6 10 GPIO_ACTIVE_LOW>;
```

```

    focaltech,max-touch-number = <5>;
    focaltech,display-coords =  <0 0 800 480>;
};


```

Listing 4.12: Device-Tree Eintrag Touchscreen

Der Name des Treibers ist `focaltech,fsl`, dieser wird mit dem Property `compatible` angegeben. Darauf folgend die Informationen zu I2C Slave-Adresse, Interrupt- und Reset-Pin. Zum Schluss werden dem Treiber Informationen zur Konfiguration übergeben. Dies sind: Zulässige Anzahl gleichzeitiger Berührungen und die Auflösung des Displays.

Das Interrupt Signal ist beim Touchscreen besonders wichtig. Es signalisiert, dass eine Berührung des Displays stattgefunden hat. Damit kann auf ein regelmässiges Pollen verzichtet werden. Im Listing 4.12 wird mit dem Property `focaltech,irq-gpio` das GPIO angegeben, an welches die Interrupt Leitung angeschlossen ist. Mit dem Property `interrupts` wird angegeben zu welcher IO-Gruppe das GPIO gehört und auf welche Flanke der Interrupt getriggert werden soll. Da das GPIO mit `GPI06_I010` bezeichnet ist, gehört es zur IO-Gruppe 10. Der Interrupt wird auf die fallende Flanke getriggert. Mit dem Property `interrupt-parent` wird der Interrupt-Controller spezifiziert.

Mithilfe dieser Angaben im Device-Tree registriert der Touchscreen Treiber, welcher mit dem Device "focaltech,fts" (Touchscreen Controller) kompatibel ist, eine Interrupt Service Routine beim Kernel. Damit wird bei jedem Berühren des Displays die Funktion im Treiber aufgerufen und die Position kann über I2C eingelesen werden. Anschliessend werden die Daten an das Linux Input Subsystem weitergeleitet [33].

Um die Funktion des Interrupts zu testen, kann die Querying Methode aus Unterabschnitt 4.2.2 angewandt werden. Mit folgendem Shell Befehl wird eine Tabelle mit allen Interrupts zusammen mit der Häufigkeit des Auftretens angezeigt.

```
sudo cat /proc/interrupts
```

Listing 4.13: Interrupt Querrying

Beim Testen des Touchscreen Treibers wird überprüft, ob ein Interrupt im System ausgelöst wird. Dazu wird der obige Befehl ausgeführt, das Display berührt, danach wird der Befehl noch einmal ausgeführt. Unterscheidet sich der Zählerstands des Interrupts in der Tabelle wurde das Interrupt Signal korrekt registriert.

## 4.6 Audio-Treiber für den AK4493 DAC

### 4.6.1 Linux ALSA

Die Advanced Linux Sound Architecture oder kurz ALSA ist die Basis des gesamten Soundsystems unter Linux. ALSA stellt die nötigen Treiber zum Ansprechen von Soundkarten zur Verfügung. Zusätzlich zu den Soundkarten von Desktop PC's werden auch Codecs unterstützt. Dies ist vor allem im Zusammenhang mit Embedded Systemen wichtig. Die Hauptaufgabe der Treiber ist es die Soundkarte oder den Codec dem System zur Verfügung zu stellen, so dass Programme aus dem User Space auf die Karte zugreifen können. Damit kann aus dem System einfach eine Audio-Datei abgespielt werden, ohne dass die Hardware der Karte bekannt sein muss. Alle hardwarespezifischen Zugriffe sind im Treiber implementiert. [16]

### 4.6.2 Linux ASoC Layer

Der Audio for System on Chip Layer (ASoC Layer) hat zum Ziel ALSA für Embedded SoC zu unterstützen. Der Fokus liegt dabei auf der Unabhängigkeit der Treiber untereinander. Damit wird erreicht, dass jeder Codec durch jedes SoC betrieben werden kann, ohne dass die Treiber neu geschrieben werden müssen. Das Wort Codec wird dabei stellvertretend für sämtliche externe Audio-Hardware verwendet. Es muss sich also nicht zwingend um einen Codec handeln. Im Fall des Audio-Streamers handelt es sich um den DAC AK4493. Als Schnittstelle zwischen SoC und

Codec wird I2S, PCM und AC97 unterstützt. Zusätzlich bietet der ASoC Layer das Dynamic Audio Power Management (DAPM). Dieses sorgt dafür, dass sich das Audio-System stets im Zustand des kleinsten Energieverbrauchs befindet. Dies ist insbesondere für portable Geräte interessant. Weiter bietet ASoC die Möglichkeit das Audio-System zu konfigurieren. [17]

Um die Austauschbarkeit von SoC oder Codec zu gewährleisten, wird das Embedded-Audio-System in 3 Treiber unterteilt. Diese sind:

- Codec Class Driver
- Platform Class Driver
- Machine Class Driver

**Der Codec Class Driver** ist plattformunabhängig, das bedeutet er kann auf jedem SoC laufen.  
Der Treiber implementiert die Funktionalitäten rund um einen spezifischen Codec.

**Der Platform Class Driver** ist SoC spezifisch. Im wesentlichen implementiert dieser Treiber das Digital Audio Interface (DAI), über welches die Audiodaten mittels Direct Memory Access (DMA) zum Codec gelangen. Die ganze Implementation ist Codec unabhängig.

**Der Machine Class Driver** ist sowohl SoC wie auch Codec spezifisch. Er dient zur Verbindung von Codec Class Driver und Platform Class Driver. Daher ist dies der einzige Treiber, der nicht generisch über mehrere Plattformen oder Codecs hinweg eingesetzt werden kann. Allerdings ist die Implementierung des Machine Class Drivers auch wesentlich weniger aufwändig, als die der anderen Treiber.

Im Idealfall muss bei der Entwicklung eines Embedded-Audio-Systems kein Treiber selbst geschrieben werden, vorausgesetzt es handelt sich um eine gängige Konfiguration von SoC und Codec. Ist dies nicht der Fall muss ausschließlich der Machine Class Driver implementiert werden, sofern die Treiber für SoC und Codec bereits existieren.

#### 4.6.3 Ausgangslage für den iMX6 und den DAC AK4497

Im Fall des Audio-Streamers wird als Platform der iMX6 und als Codec (DAC) der AK4493 eingesetzt. Im Linux Kernel, sind bereits alle 3 oben erwähnten Treiber für die Kombination aus iMX6 und AK4497 implementiert. Da sich die DACs AK4493 und AK4497 gleich verhalten, kann der Treiber für den AK4497 auch für den AK4493 eingesetzt werden. Allerdings hat sich gezeigt, das noch zusätzliche Anpassungen vorgenommen werden müssen. Diese sind:

- Konfiguration der AUDMUX Peripherie des iMX6
- Konfiguration der IOMUX Peripherie des iMX6
- Konfiguration des Clock Trees des iMX6
- Ergänzung der Nodes für Codec und Platform Class Driver im Device-Tree
- Anpassung der ASoC Treiber

##### Konfiguration der AUDMUX Peripherie des iMX6

Die erste Hürde um den AK4493 DAC für den Audio-Streamer in Betrieb zu nehmen ist die Konfiguration der AUDMUX Peripherie. Diese dient dem Multiplexen der digitalen Audiosignale von und zu den Audio Peripherien. Bei den Audio Peripherien handelt es sich um das Synchronous Serial Interface (SSI), dieses wird im Unterabschnitt Audioperipherie des iMX6 beschrieben. Auf dem Apalis iMX6 selbst, ist bereits der Codec SGTL5000 von NXP verbaut. Daher wird standardmäßig dieser zur Audiowiedergabe verwendet. Um nun die Signale auf den DAC des Audio-Streamers umleiten zu können, muss der AUDMUX konfiguriert werden. Die Signale sollen von AUD1 anstatt nach AUD4 (SGTL5000) zu AUD5 geleitet werden. Damit werden die Signale bei richtiger Konfiguration des IOMUX direkt zu den beiden DACs des Audio-Streamers geleitet. In der Abbildung 4.2 ist der Aufbau der AUDMUX Peripherie abstrahiert dargestellt.

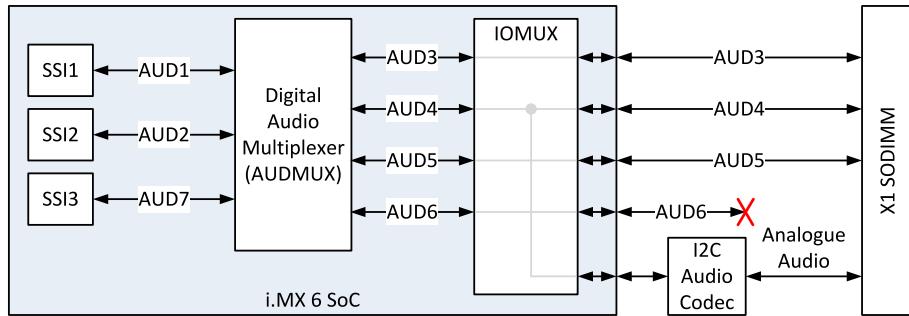


Abbildung 4.2: Übersicht über die IOMUX Peripherie [18]

Bei der Standardkonfiguration mit dem Codec SGTL5000 wurde die Konfiguration der AUDMUX Peripherie direkt durch den Treiber des Codecs übernommen. Dies liegt vor allem daran, dass der Codec wie auch der iMX6 von NXP hergestellt werden. Da für den Audio-Streamer der AK4493 und damit ein anderer Treiber eingesetzt wird, ist die Konfiguration über den Treiber nicht mehr möglich. Allerdings hat sich gezeigt, dass die Konfiguration auch direkt über den AUDMUX Treiber von NXP erfolgen kann. Die Parameter werden dabei vom AUDMUX Treiber aus dem Device-Tree geparsst. In folgendem Listing ist der erstellte Device-Tree Eintrag für die Konfiguration der AUDMUX Peripherie zu sehen.

```
<audmux >
  pinctrl-names = "default";
  pinctrl-0 = <&pinctrl_audmux>;
  status = "okay";

  /* configuration for the Audio-Streamer */
  /* internal */
  audmux_port1 {
    fsl,audmux-port = <0>;
    fsl,port-config = <
      0x00000000
      IMX_AUDMUX_V2_PDCR_RXDSEL(4)
    >;
  };

  /* external */
  audmux_port5 {
    fsl,audmux-port = <4>;
    fsl,port-config = <
      (IMX_AUDMUX_V2_PTCR_TFSDIR |
       IMX_AUDMUX_V2_PTCR_TFSEL(0) |
       IMX_AUDMUX_V2_PTCR_TCLKDIR |
       IMX_AUDMUX_V2_PTCR_TCSEL(0))
      IMX_AUDMUX_V2_PDCR_RXDSEL(0)
    >;
  };
};
```

Listing 4.14: Device-Tree Eintrag AUDMUX Peripherie

Zu beachten ist, dass die Nummerierung im Device-Tree um eins von der Grafik abweicht. Dies weil die Nummerierung der Ein- und Ausgänge bei 0 startet. Mit den Makros DIR kann nun jeweils die Richtung als Ausgang gesetzt werden. Mit den Makros SEL kann das Mapping vorgenommen werden. Mit dieser Konfiguration werden die Signale TCLK (Bit Clock) und TFS (Word Clock) von AUD1 zu AUD5 gemappt und bei AUD5 als Ausgang deklariert. Das Signal RXD wird von AUD5 nach AUD1 gemappt, wobei dieses Signal für den Audio-Streamer nicht verwendet wird. Das TXD Signal, also die eigentlichen Daten, sind standardmässig richtig gemappt.

### Konfiguration der IOMUX Peripherie des iMX6

Damit die Digitalen Audiosignale und Clocks auf die richtigen Pins gelenkt werden, muss die IOMUX Peripherie konfiguriert werden. Diese Konfiguration geschieht durch die Erstellung eines No-

des mit dem Namen pinctrl\_audmux unter dem Node &iomux. Dieser sieht folgendermassen aus:

```
pinctrl_audmuaudmuxgrp {
    /* using AUD3 (using SPI1 interface) */
    fsl,pins = <
        /* ak4497 Audio-Streamer */
        MX6QDL_PAD_DISP0_DAT16_AUD5_TXC PAD_CTRL_HYS_PD // MXM3 200
        MX6QDL_PAD_DISP0_DAT18_AUD5_TXFS PAD_CTRL_HYS_PD // MXM3 204
        MX6QDL_PAD_DISP0_DAT17_AUD5_TXD PAD_CTRL_HYS_PD // MXM3 196
        MX6QDL_PAD_DISP0_DAT19_AUD5_RXD PAD_CTRL_HYS_PD // MXM3 202
        /* masterclock */
        MX6QDL_PAD_GPIO_19_CCM_CLK01 PAD_CTRL_HYS_PD // MXM3 194
    >;
};
```

Listing 4.15: Device-Tree Eintrag für die Konfiguration der Audio Pins

### Konfiguration des Clock Trees des iMX6

Damit der DAC AK4493 das Prinzip der Überabtastung nutzen kann muss zusätzlich zum Bit Clock ein Master Clock zur Verfügung gestellt werden. Anders als die Clocks die zum synchronisierten Lesen und Schreiben der Audiodaten dienen, wird die Master Clock nicht von der Audioperipherie (SSI) erzeugt. Stattdessen wird sie direkt von einem PLL auf ein Pin des iMX6 geführt. Gemäss Datenblatt des AK4493 besteht eine Beziehung zwischen der Sample Rate der Audiodatei und der Master Clock. Dieser Zusammenhang ist in der Tabelle 4.1 ersichtlich. Bei einer fest eingesetzten Master Clock kann nur das vielfache einer bestimmten Sample Rate abgespielt werden. Mit den Frequenzen 24.576 MHz und 22.579 MHz können alle gängigen Sample Raten abgespielt werden.

Sample Rate / kHz	Faktor	MCLK / MHz	Unterstützt
44.1	512	22.5792	x
48	512	24.576	(x)
88.2	256	22.5792	x
96	256	24.576	(x)
176.4	128	22.5792	x
192	128	24.576	(x)
384	64	24.576	-
768	32	24.576	-

Tabelle 4.1: Mögliche Abspielraten für den Audio-Streamer [3]; x: Unterstützt, (x): Bei Änderung der Master Clock Frequenz unterstützt, -: Nicht unterstützt

Leider ist im Treiber für den AK4497 keine Konfiguration des Clock-Trees vorgesehen. Die Konfiguration des Clock-Trees wird direkt durch den Clock Treiber vorgenommen und ist fix im Treiber programmiert. Standardmäßig ist die Master Clock auf 24 MHz konfiguriert. Dies liegt daran, dass die meisten Codecs damit arbeiten können. Für den Audio-Streamer muss der Clock Tree für die Master Clock umkonfiguriert werden. Dies geschieht durch Anpassen des Audio-PLL (PLL 4) und durch Anpassen der nachgeschalteten Teiler. Der PLL4 kann gemäss Reference Manual [19] auf eine Frequenz zwischen 650 MHz und 1300 MHz eingestellt werden. Die Genauigkeit ist dabei besser als 1 MHz. Nachgeschaltet ist ein Teiler der die Frequenz durch 1, 2, oder 4 teilen kann. Für die Generierung der Master Clock für den AK4493 wird der Ausgang des PLL4 (nach dem Teiler) auf 180.633 600 MHz eingestellt. Der Clock Treiber konfiguriert den PLL und den nachgeschalteten Teiler automatisch, so dass im Treiber mit `imx_clk_set_rate()` nur die gewünschte Frequenz am Ausgang eingestellt werden muss.

Die Konfiguration im Clock Treiber `clk-imx6q.c` unter `drivers/clk/imx` im Kernel sieht folgendermassen aus:

```

/* Mux for ssi1 source selection */
imx_clk_set_parent(clk[IMX6QDL_CLK_SSI1_SEL], clk[IMX6QDL_CLK_PLL4_AUDIO_DIV]);

/* Signal mapping */
imx_clk_set_parent(clk[IMX6QDL_CLK_CK02_SEL], clk[IMX6QDL_CLK_SSI1]);
imx_clk_set_parent(clk[IMX6QDL_CLK_CK0], clk[IMX6QDL_CLK_CK02]);

/* Divider configuration for 22.5792 MHz */
imx_clk_set_rate(clk[IMX6QDL_CLK_PLL4_AUDIO_DIV], 180633600);
imx_clk_set_rate(clk[IMX6QDL_CLK_SSI1], 180633600/8);

/* Enable master clock */
imx_clk_prepare_enable(clk[IMX6QDL_CLK_CK0]);
imx_clk_prepare_enable(clk[IMX6QDL_CLK_CK02]);

```

Listing 4.16: Clock Treiber Eintrag zur Konfiguration der Master Clock

Ganz oben im Listing 4.16 wird der Multiplexer aus Abbildung 4.3 konfiguriert. So, dass die Clock vom PLL4 als Quelle für die SSI1 Peripherie gewählt wird. Dem PLL4 sind weitere zwei Teiler nachgeschaltet. Mit diesen kann die Frequenz für die SSI1 Peripherie angepasst werden. Die Frequenz am Ausgang dieser Teiler wird auf  $180.633\,600\text{MHz}/8 = 22.5792\text{MHz}$  eingestellt. Die Konfiguration der Teiler wird hier wiederum dem Treiber überlassen. Die Teilung von 8 entspricht dabei genau der Standardkonfiguration.

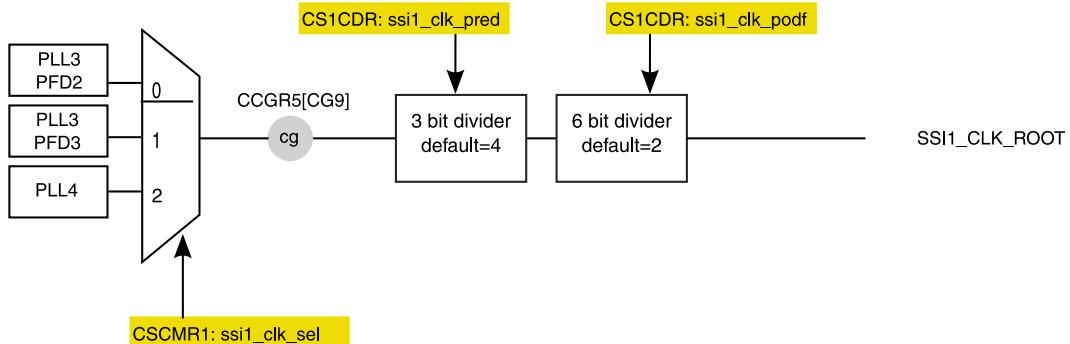


Abbildung 4.3: Übersicht über den Clock Tree von PLL4 nach SSI1 [19]

Der Nachteil von diesem Vorgehen ist, dass die Master Clock beim Booten auf eine feste Frequenz eingestellt wird. Für eine Umkonfiguration muss der Clock Treiber angepasst und der Kernel neu kompiliert werden. Um die Konfiguration aus dem Treiber für den AK4497 zu ermöglichen, wäre eine starke Anpassung notwendig. Deshalb wurde im Hinblick auf die Verwendbarkeit des Treibers mit anderen Kernel Versionen auf die Anpassung verzichtet.

Da die meisten Audiodateien mit dem Vielfachen von 44.1 kHz abgetastet werden (CD-Standard), wird die Master Clock Frequenz fest auf 22.5792 MHz eingestellt.

### Ergänzung der Nodes für Codec und Platform Class Driver im Device-Tree

Damit der Codec Class Driver, der Platform Class Driver und der Machine Class Driver beim Booten geladen werden, müssen zwei Nodes im Device-Tree ergänzt werden. Die Nodes wurden Anhand der Vorgabe in der Kernel Dokumentation unter Documentation/devicetree/ → bindings/sound erstellt. Beim Ersten Eintrag handelt es sich um den Eintrag zur Erstellung der Soundkarte. Neben dem Property compatible wird zusätzlich eine Referenz auf die Audio Peripherie SSI1 und auf den Codec (I2C Device) übergeben. Das Audio-Routing ist für die Anwendung im Zusammenhang mit dem Audio-Streamer nicht von Bedeutung.

```

sound_ak4497: sound-ak4497{
    compatible = "fsl,imx-audio-ak4497";
    model = "ak4497-audio";
    audio-cpu = <&ssi1>;
    audio-codec = <&codec_ak4497>;
    audio-routing =

```

```

    "AOUTLN", "Playback",
    "AOUTLP", "Playback",
    "AOUTRN", "Playback",
    "AOUTRP", "Playback";
};


```

Listing 4.17: Device-Tree Eintrag für das Sound Device

Der zweite Eintrag ist in folgendem Listing zu sehen. Dieser erstellt einen Codec Node als I2C1 Device. Dieser Node wird für das Laden des Codec Class Drivers benötigt. Als wichtigster Parameter wird hier die Slave Adresse des Codec angegeben. Wichtig ist, dass die I2C Adresse nicht direkt vom Codec Class Driver geparsst wird. Dies wird hier durch den I2C Treiber übernommen. Im Codec Treiber selbst kann dann mithilfe von Kernel API Funktionen auf das I2C Device zugegriffen werden. Die beiden GPIO's Power Down und Mute werden nicht benötigt. Diese Signale werden für den Audio-Streamer durch das Power Management geliefert 4.7.

```

&i2c1 {
    codec_ak4497: codec-ak4497@0x10 {
        compatible = "asahi-kasei,ak4497";
        reg = <0x10>;
        asahi-kasei,pdn-gpios = <&gpio2 5 GPIO_ACTIVE_HIGH>;
        asahi-kasei,mute-gpios = <&gpio2 4 GPIO_ACTIVE_HIGH>;
    };
};


```

Listing 4.18: Device-Tree Eintrag für den Codec

### Audioperipherie des iMX6

Der iMX6 bietet mehrere Möglichkeiten um digitale Audiosignale zu generieren und auszugeben. Da für die Übertragung der Audiosignale zum AK4493 eine I2S Schnittstelle verwendet wird, ist das Synchronous Serial Interface (SSI) des iMX6 das Mittel der Wahl. Das SSI ist im Prinzip eine I2S Peripherie mit zusätzlichen Konfigurationsmöglichkeiten. In Abbildung 4.4 ist eine Übersicht über die gesamte Audioperipherie des iMX6 dargestellt.

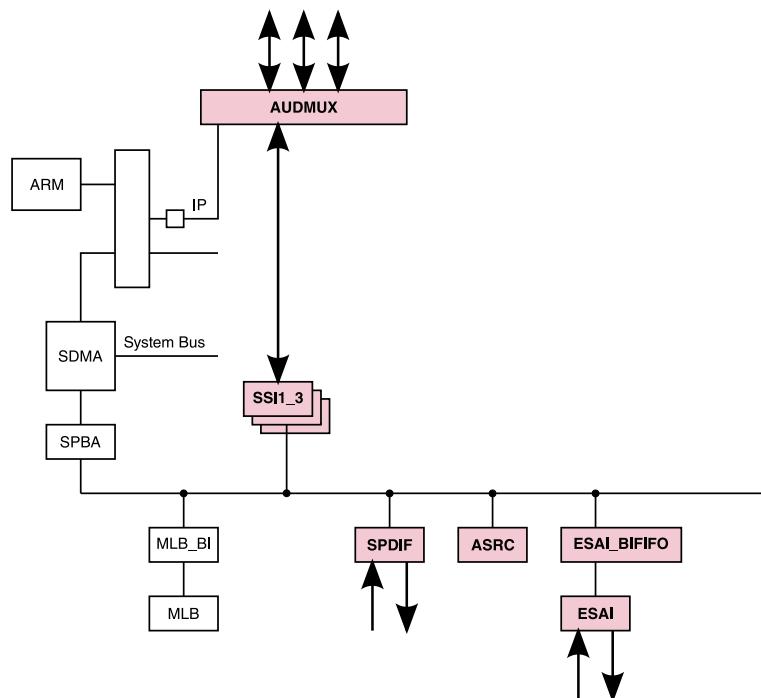


Abbildung 4.4: Übersicht über die Audio Peripherie des iMX6 [19]

Sämtlich Audio Peripherien sind an den Shared Peripheral Bus des iMX6 angeschlossen. Daher kann über den Treiber mittels DMA auf die Peripherie zugegriffen werden. So gelangen auch die

Audio Daten zur entsprechenden Peripherie. Hinter den 3 SSI befindet sich der AUDMUX, dieser ist im Unterabschnitt 4.6.3 dokumentiert.

Einer der wichtigsten Punkte für eine korrekt funktionierende Audio Peripherie ist das Clocking. Eine Übersicht über das Clocking der SSI Peripherie ist in Abbildung 4.5 dargestellt. Es gibt insgesamt 3 Clocks die benötigt werden. Dies ist die Master Clock, die Bit Clock und die Word Clock. Die Master Clock wird ausserhalb der Audio Peripherie generiert. Mehr dazu ist in 4.6.3 zu finden. Die Bit Clock sowie die Word Clock wird in der Audio Peripherie generiert. Wichtig ist, dass die Bit Clock und die Word Clock absolut synchron zur Master Clock sind [3]. Die Phase spielt dabei keine Rolle. Um die Synchronität zu gewährleisten, werden die Bit Clock und die Word Clock direkt von der Master Clock abgeleitet. Dies setzt voraus, dass die Master Clock bis zum SSI Sysclock (Rot im Bild) durchgeschaltet wird. Ist dies der Fall kann nun durch das Einstellen der Teiler DIV2, PSR und PM die Bit Clock eingestellt werden. Mit dem zusätzlichen Teiler WL wird die Word Clock von der Bit Clock abgeleitet. Die Konfiguration der Teiler wird durch den Platform Class Driver von Freescale übernommen. Der Treiber wird automatisch geladen, wenn im Machine Class Driver das PCM Format ausgewählt wird.

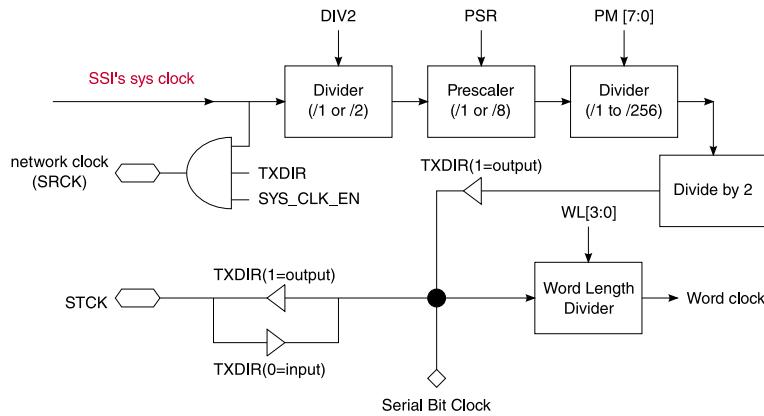


Abbildung 4.5: Übersicht über den SSI Clock Tree [19]

Die Formel zur Berechnung der Teiler für eine vorgegebene Sysclk (Master Clock) und Bit Clock, sieht folgendermassen aus:

$$f_{\text{Bit}} = \frac{f_{\text{Sys}}}{(\text{DIV2} + 1) \cdot (7 \cdot \text{PSR} + 1) \cdot (\text{PM} + 1) \cdot 2} \quad (4.3)$$

In Gleichung 4.4 kann leicht erkannt werden, dass  $f_{\text{Bit}}$  maximal die Hälfte von  $f_{\text{Sys}}$  sein kann. Da  $f_{\text{Sys}}$  die Master Clock ist und diese auf 22.5792 MHz festgelegt wurde, ergibt sich eine maximale Bit Clock von 11.2896 MHz.

Die nötige Bit Clock kann wie folgt aus der Sample Rate berechnet werden:

$$f_{\text{Bit}} = 32 \cdot N_{\text{Channels}} \cdot f_{\text{Sample}} \quad (4.4)$$

Die Bitbreite von 32 Bit wird im Mehrkanalbetrieb von der Hardware fix vorgegeben. Die Anzahl der Kanäle ist im Fall des Audio-Streamers immer 2. In Tabelle 4.2 sind die berechneten Bit Clock Werte in Abhängigkeit der Sample Rate darstellt.

Sample Rate / kHz	Bit Clock / MHz
44.1	2.8224
48	3.072
88.2	5.6448
96	6.144
176.4	11.2896
192	12.288
384	24.576
768	49.152

Tabelle 4.2: Zusammenhang zwischen Sample Rate und Bit Clock

Damit ist klar, dass die maximale vom iMX6 unterstützte Sample Rate für den AK4493 176.4 kHz bei einer 22.5792 MHz Master Clock und 192 kHz bei einer 24.576 MHz Master Clock ist.

### Anpassung der ASoC Treiber

Damit der AK4493 Treiber für den Audio-Streamer eingesetzt werden kann, müssen einige Anpassungen vorgenommen werden. Damit die Vorgänge im Treiber nach dem Laden besser verfolgt werden können, wurde ein einfaches Debug Level in die 3 Audio-Treiber Dateien eingefügt. Mit dem Makro #DEBUG kann die Kompilierung der Debug Meldungen ein- und ausgeschaltet werden. Diese Meldungen sind vor alle zur Identifizierung von falsch berechneten Clock Frequenzen nützlich. In folgenden Abschnitten werden nun die vorgenommenen Änderungen in den ASoC Treibern dokumentiert.

Im Codec Class Driver in der Datei AK4497.c muss zusätzlich das Mono-Bit gesetzt werden. Dies ist nötig, da für den Audio-Streamer zwei DAC's im Mono Mode eingesetzt werden. Zudem wird das Power-Bit, das Reset-Bit und das AutoSync-Bit gesetzt. Für den DAC der durch den Codec Class Driver angesteuert wird, wird zusätzlich das SELLR-Bit gesetzt. Dies bewirkt, dass nur die rechte Tonspur berücksichtigt wird. Damit das System auch in Stereo betrieben werden kann, muss auch der zweite DAC konfiguriert werden. Allerdings ist der Treiber für den AK4497 nur für einen DAC ausgelegt. Dazu kommt, dass der Platform Class Driver vom Codec Class Triber abhängig ist. Somit müsste eine zweite Soundkarte erstellt werden, was das Abspielen in Stereo aus dem User-Space verunmöglicht. Deshalb wurde entschieden den zweiten DAC aus dem User-Space zu konfigurieren. Damit muss der Treiber nur geringfügig angepasst werden. So steigt die Wahrscheinlichkeit, dass das Vorgehen auch mit einer neuen Kernel Version noch funktioniert. Die Konfiguration des zweiten DAC's wird mit der Ausführbaren Datei AK4497\_stereo\_dac\_config vorgenommen. Sie befindet sich im Filesystem des Audio-Streamers unter /home/audiostreamer/ ↴ applications.

Im Platform Class Driver fsl\_ssi.c wurde durch die eingefügten Debug Meldungen ein Problem bei der Berechnung der Bit Clock identifiziert. Dies weil die Clock, welche die SSI Peripherie betreibt falsch angenommen wurde. Dies wurde durch ein eingefügtes Switch Case Statement behoben, welches den richtigen Prescaler direkt einstellt. Der erstellte Code Abschnitt ist in folgendem Listing ersichtlich.

```
/* added by klosr1, in assumption that MCLK and peripheral clock of SSI is 22'579'200Hz or
   ↵ 24'576'000Hz */
/* chose right bitclk prescaler for given bitclk */
switch (freq) {
case 2822400: /* rate = 44100Hz */
pm = 3;
break;
case 5644800: /* rate = 88200Hz */
pm = 1;
break;
case 11289600: /* rate = 176000Hz */
pm = 0;
```

```

        break;
    case 3072000: /* rate = 48000Hz */
    pm = 3;
    break;
    case 6144000: /* rate = 96000Hz */
    pm = 1;
    break;
    case 12288000: /* rate = 192000Hz */
    pm = 0;
    break;
default:
    dev_err(cpu_dai->dev, "failed to handle the required sysclk\n");
}

```

Listing 4.19: Platform Class Driver Eintrag zur Konfiguration der Bit Clock

Im Machine Class Driver `imx-ak4497.c` wurde die Tabelle für die Beziehung zwischen Master Clock und Word Clock angepasst. Dies weil der Treiber für den AK4497 ausgelegt ist und das Datenblatt für den im Audio-Streamer eingesetzten AK4493 leicht andere Werte angibt. Die geänderte Tabelle ist in folgendem Listing ersichtlich.

```

/* modified by Klosr1 */
/* relation between rate (LRCLK) and MCLK */
} fs_mul[] = {
/**
 * Table 7      - mapping multiplier and speed mode
 * Tables 8 & 9 - mapping speed mode and LRCK fs
 */
{ .min = 8000,   .max = 32000,   .mul = 768 }, /* Normal, <= 32kHz */
{ .min = 44100,  .max = 48000,  .mul = 512 }, /* Normal */
{ .min = 88200,  .max = 96000,  .mul = 256 }, /* Double */
{ .min = 176400, .max = 192000, .mul = 128 }, /* Quad */
{ .min = 352800, .max = 384000, .mul = 64 }, /* Oct */
{ .min = 705600, .max = 768000, .mul = 64 }, /* Hex */
};

```

Listing 4.20: Machine Class Driver Eintrag für die Bestimmung des Faktors zwischen Master Clock und Word Clock

## 4.7 Power-Management Skript

Beim Booten des Audio-Streamers müssen die Speisungen für den analogen und digitalen Schaltungsteil in einer bestimmten Reihenfolge eingeschaltet oder ausgeschaltet werden. Damit wird sichergestellt, dass der DAC beim laden des Audio-Treibers von diesem auch erkannt wird. Bei fehlender Speisung werden die DACs über I2C nicht erkannt und damit wird auch keine Soundkarte erstellt. Das kontrollierte Einschalten dient auch dem Verhindern von Klick-Geräuschen am Ausgang des Audio-Streamers. Dazu müssen auch die Signale Mute und Reset der DACs kontrolliert werden. Der Ablauf beim Starten des Audio-Streamers ist in Abbildung 4.6 zu sehen.

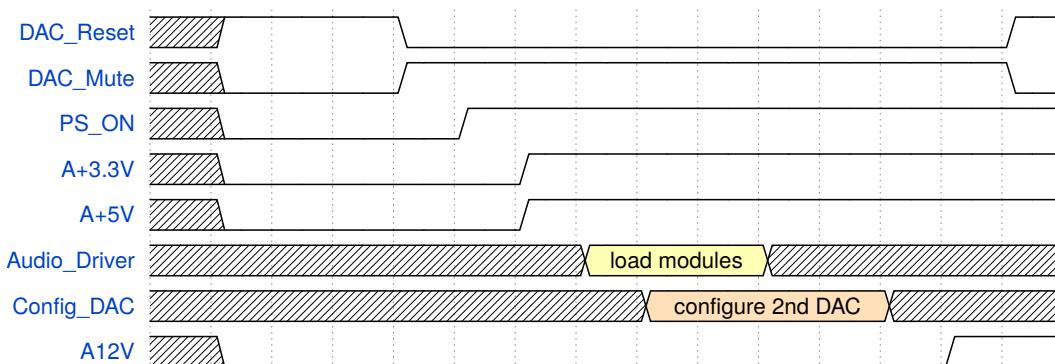


Abbildung 4.6: Timing Diagramm des Startup Prozesses [19]

Unter Linux gibt es mehrere Möglichkeiten beim Booten nach einem definierten Muster die nötigen GPIO's zu setzen und zu löschen. Eine Variante ist das Schreiben eines Treibers. Dies ist allerdings sehr aufwändig und sollte deshalb nur wenn nötig gemacht werden. Aus diesem Grund wurde entschieden ein Shell-Skript zu schreiben, welches beim Booten automatisch ausgeführt wird. Das automatische Ausführen wird durch das Systemd unter Linux übernommen. Dazu wird ein Service-File mit folgendem Inhalt unter /etc/systemd/system angelegt:

```
[Unit]
Description=/etc/rc.local
ConditionPathExists=/etc/rc.local

[Service]
Type=forking
ExecStart=/etc/rc.local start
TimeoutSec=0
StandardOutput=tty
RemainAfterExit=yes
SysVStartPriority=99

[Install]
WantedBy=multi-user.target
```

Listing 4.21: Eintrag zur Aktivierung des rc.local Services

Dieser Service führt das Skript /etc/rc.local beim Booten automatisch aus. Unter Verwendung von Sysfs können in diesem Skript GPIO's leicht gesetzt und gelöscht werden. Dies bedingt natürlich, dass die Pins in der IOMUX Peripherie richtig gesetzt sind. Dies geschieht im Device-Tree. Die einzelnen Pins und deren Konfiguration sind in folgender Tabelle 4.3 ersichtlich.

<b>MXM3 Nummer</b>	Pin Funktion nach Revision	Pin Funktion aktuell	Pin Bezeichnung im Datenblatt	Sysfs GPIO Nummer
<b>225</b>	PS_EN_A+3.3V	REF_CLK_EN_2	CSI0_DAT5	151
<b>227</b>	PS_EN_A+5V	PS_EN_A12V	CSI0_DAT7	153
<b>223</b>	PS_EN_A12V	REF_CLK_EN_1	CSI0_DAT6	152
<b>198</b>	DAC_RST	DAC_RST	EIM_LBA	59
<b>1</b>	DAC_MUTE	PS_EN_A+3.3V	NANDF_D4	36
<b>3</b>	REF_CLK_EN_1	PS_EN_A+5V	NANDF_D5	37
<b>5</b>	REF_CLK_EN_2	DAC_MUTE	NANDF_D6	38
<b>233</b>	PS_PG_A+3.3V	PS_PG_A+3.3V	EIM_RW	58
<b>235</b>	PS_PG_D+3.3V	PS_PG_D+3.3V	EIM_CS0	55
<b>231</b>	PS_PG_A+5V	PS_PG_A+5V	EIM_CS1	56
<b>229</b>	PS_PG_A-12V	PS_PG_A-12V	EIM_OE	57
<b>221</b>	PS_PG_A+12V	PS_PG_A+12V	CSI0_DAT4	150
<b>4</b>	PS_CTRL	PS_CTRL	GPIO_1	1

Tabelle 4.3: Übersicht über die Power-Management Pins

In der Spalte ganz links sind die Nummern des MXM3 Verbinders vom Apalis-iMX6Q angegeben. Der MXM3 Verbinde ist die Schnittstelle zwischen CoM und Hardware des Audio-Streamers. Gleich daneben ist die Funktion des Pins nach der Revision aufgeführt. Da in der aktuellen Version im Schema ein Verbindungsfehler vorliegt, gilt momentan die Funktion in der mittleren Spalte. Im Datenblatt des iMX6 und damit auch im Device-Tree wird die Bezeichnung in der zweiten Spalte von rechts verwendet. Ganz rechts ist die GPIO Nummer im Sysfs angegeben. Damit kann das GPIO aus dem User Space angesprochen werden.

## 4.8 Treiber für den Drehgeber

Der Drehgeber oder auch Rotary Encoder genannt, dient als Bedienelement des Audio-Streamers dem einfachen Scrollen durch Listen. Zusätzlich soll der Drehgeber durch seine Hintergrundbeleuchtung den Audio-Streamer auch optisch aufwerten. Um den Drehgeber ans System anzubinden, wird ein Linux Treiber benötigt. Da der iMX6 keine Peripherie für die direkte Verarbeitung der Encoder-Signale bietet, wird ein generischer Treiber verwendet. Für diesen Treiber sind lediglich GPIO's nötig. Mithilfe von Interrupts, welche auf die gewünschte Flanke der beiden Encoder-Signale getriggert werden, kann die Richtung und die Anzahl der Pulse detektiert werden.

Die Dokumentation des verwendeten Treibers ist unter Documentation/input/rotary-encoder  
↪ .txt im Kernel zu finden. Der Device-Tree Eintrag dazu ist unter Documentation/devicetree  
↪ /bindings/input/rotary-encoder.txt dokumentiert.

# 5 Musikserver & Front-End

In diesem Kapitel wird das Design und Zusammenspiel von Musikserver und dem Front-End erläutert. Das Front-End bildet die Schnittstelle zwischen dem Musikserver und dem Nutzer des Audio-Streamers. Das Design des eigens entwickelten Streamer-UI als Front-End wird in diesem Kapitel vorgestellt.

## 5.1 Software Architektur

### 5.1.1 Übersicht System

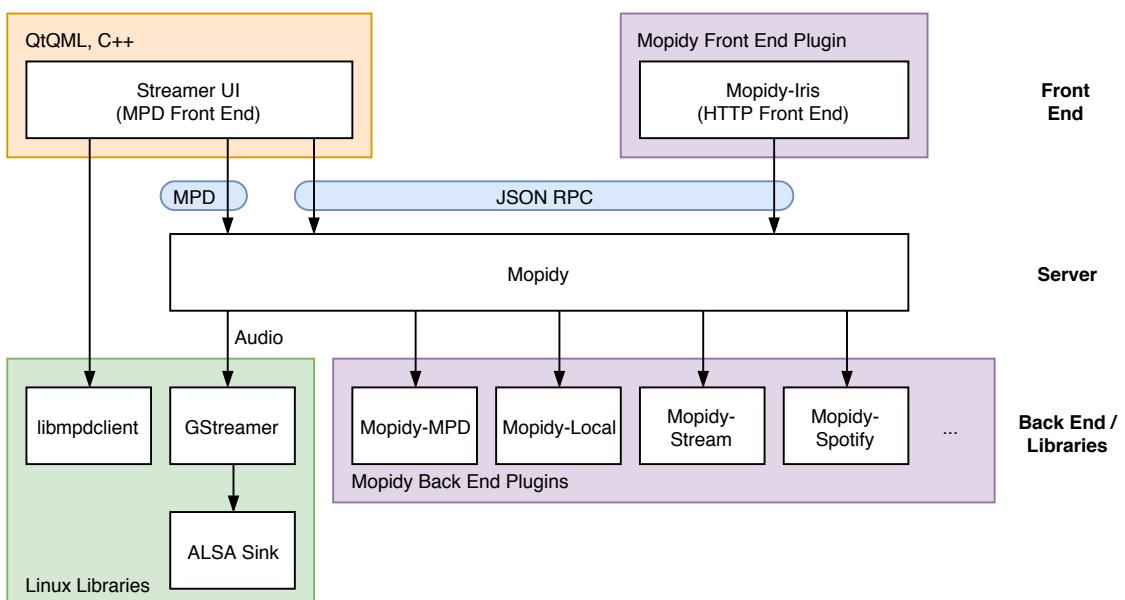


Abbildung 5.1: Architektur des Musikservers und Front-Ends

Die Architektur des Systems ist in Abbildung 5.1 dargestellt. Sie kann in die Schichten Front-End, Musikserver, Backend und Libraries unterteilt werden.

Das Front-End bedient dabei den Benutzer. Hier werden Informationen zum aktuellen Titel oder Suchresultate dargestellt. Zudem werden Benutzereingaben verarbeitet und an den Musikserver weitergeleitet.

Der Musikserver ist das Bindeglied zwischen dem Front-End und den Back-Ends zu den verschiedenen Streaming Diensten. Der Server hat dafür zu sorgen, dass auf der Audio Schnittstelle das gewünschte Stück abgespielt wird. Er kommuniziert dem Front-End, welche Streaming Dienste zur Verfügung stehen und leitet Anfragen an das verantwortliche Plugin weiter.

Die Back-Ends sind bei Mopidy modular aufgebaut und haben jeweils nur eine Aufgabe. Jede Quelle des Servers wird als solches Plugin realisiert. Die Back-Ends melden ihre Funktion beim Starten dem Musikserver.

Über die Linux Libraries können die einzelnen Komponenten auf Systemfunktionen zugreifen. Hier befindet sich die Senke des Audio Pfads.

## 5.1.2 Front-End

### Mopidy-Iris

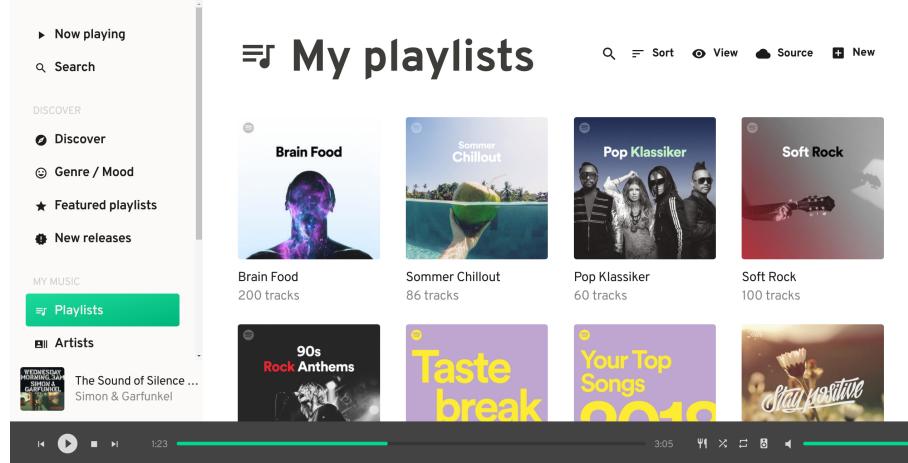


Abbildung 5.2: Iris Web Front-End für Mopidy

Nebst dem Front-End, welches auf dem Display des Audio-Streamers angezeigt wird, kann ein Web Front-End verwendet werden. Dazu wird das Mopidy-Iris Plugin installiert. Iris ist, wie in Abbildung 5.2 ersichtlich, eine Webseite, mit der Mopidy über einen Browser im lokalen Netzwerk bedient werden kann. Sie wird auf dem Audio-Streamer mit dem Mopidy Webserver gehostet. Über Iris können Musikstücke gesucht und abgespielt werden. Sie wird parallel zum Streamer-UI verwendet, so kann einerseits während der Entwicklung des Streamer-UI die fehlenden Funktionen über die Webseite bedient werden, anderseits kann der Benutzer mit einem Computer oder Smartphone den Audio-Streamer über das Netzwerk steuern. [20]

### Streamer-UI

Das Front-End für das Display des Audio-Streamers wird mit QtQML und C++ realisiert. Aufgrund durch QtQML vorgegebener Aufteilung der Verantwortlichkeiten in der Software drängt sich die Model-View-Controller (MVC) Architektur auf.

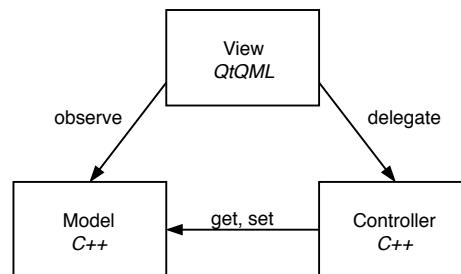


Abbildung 5.3: Architektur des Streamer-UI

Das View hat die Aufgabe dem Benutzer Informationen anzugeben und Steuerbefehle weiterzuleiten. Das View wird mit Qt Modeling Language (QML) beschrieben. QML ist eine deklarative Sprache für Benutzeroberflächen von Qt. Die Navigation innerhalb der Applikation und einfache Logik werden bereits in QML abgebildet. [21]

Die Verbreitung der Benutzereingaben und die Schnittstelle zum Musikserver wird im Controller implementiert. Dazu wird die Programmiersprache C++ verwendet. Dieser Teil umfasst die grösste Logik und somit den meisten Code. Um Methoden des Controllers aus dem View aufrufen zu können, müssen die Methoden beim Qt Framework bekannt gemacht werden. Der Controller schreibt seine Daten nicht direkt auf das View, da er die Elemente im View nicht kennt, sondern speichert diese im Model ab.

Das Model umfasst die Daten der Applikation. Der Zugriff erfolgt über getter und setter Methoden. Die Daten des Model werden wiederum beim Qt Framework registriert. So kann sich das View Updates von Daten abonnieren.

Dieser Aufbau wird von Qt empfohlen [22], da so die Elemente der Benutzeroberflächen vom Controller getrennt werden. Dieses Verhalten setzt das Observer Design Pattern um, dieser erlaubt mehreren Views auf einheitliche Daten zuzugreifen. [23]

Die MVC Architektur stellt eine einfache Wartung sicher. So kann die Implementation des Controllers ausgetauscht werden, ohne das View anpassen zu müssen, da das Model unverändert bleibt. Somit kann die Software in Zukunft angepasst werden, ohne eine komplette Umstrukturierung vorzunehmen.

### 5.1.3 Musikserver: Mopidy

Im Zentrum steht der Musikserver, dazu wird Mopidy verwendet. Mopidy bietet nebst dem Kernserver auch etliche Plugins zur Erweiterung der Funktionalität an. Auf Mopidy kann über TCP/IP zugegriffen werden. Als Applikationsprotokolle können das Mopidy JSON RPC oder das Music Player Deamon (MPD) Protokoll verwendet werden.

### 5.1.4 Mopidy Backend-Plugins

Der modulare Aufbau von Mopidy erlaubt, die gewünschte Funktionalitäten einzeln hinzuzufügen.

#### Lokalen Dateien

Mit dem Mopidy-Local können Musikstücke auf dem Audio-Streamer verwaltet werden. Es erstellt dazu mit einem Scan eine Datenbank mit den Informationen zu einzelnen Titeln. Diese Daten können durch das Mopidy-Local-Images Plugin automatisch mit dem Album Cover ergänzt werden.

Somit können bestehende CD-Sammlung mit einer verlustfreien Komprimierung wie flac komprimiert und auf dem Audio-Streamer geladen werden. Die einzelnen Stücke sind danach über den Mopidy Server in Originalqualität verfügbar.

#### Internet Stream

Nebst den lokalen Dateien und Streaming-Diensten können direkte Audio-Streams über das Internet genutzt werden. Dies ist bei Internet Radios der Fall. Radio Stationen stellen dazu einen Link zum Audio-Stream bereit. Über das Mopidy-Stream Plugin können diese live abgespielt werden.

Dadurch wird die Funktionalität bestehender Radios ersetzt.

#### Streaming Dienste

Moderne on-demand Streaming-Angebote werden wiederum mit Plugins zu Mopidy hinzugefügt. Nachfolgend sind einige unterstützte Streaming Dienste aufgelistet:

- Qobuz [24]
- Tidal [25]
- Spotify [26]
- SoundCloud [27]

#### Music Player Daemon

Der Music Player Daemon (MPD) ist ein freier Musikserver, welcher seit 2003 existiert (sieben Jahre vor Mopidy). MPD verwendet für zum Steuern des Servers ein eigenes TCP/IP Protokoll. Diese Schnittstelle kann bei Mopidy zur Kompatibilität mit bestehenden MPD Clients aktiviert werden.

### 5.1.5 Linux Libraries

Die Linux Bibliotheken werden zur Ansteuerung von System Komponenten oder für erweiterte Funktionalität verwendet.

#### libmpdclient

Die libmpdclient Bibliothek stellt C-Funktionen zur Ansteuerung des MPD Servers zur Verfügung. Sie wird vom Streamer-UI verwendet, um Informationen zum aktuellen Titel abzufragen, die Musik Datenbank zu durchsuchen oder die Wiedergabe zu steuern.

#### GStreamer & ALSA Sink

Mopidy greift über die GStreamer Bibliothek auf die ALSA Sink zu. Dadurch wird ein Audio-Stream direkt an ALSA geleitet. ALSA gibt den Audio-Stream über den AK4493 Treiber auf die I<sup>2</sup>S-Schnittstelle des Audio-Streamers aus.

## 5.2 Streamer-UI SW Design

### Use Cases

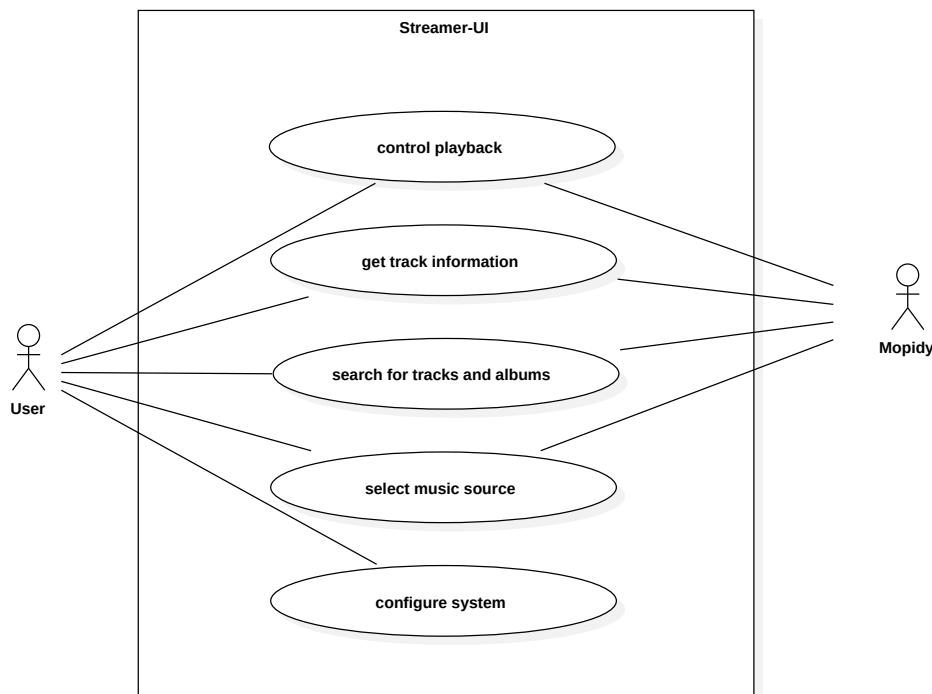


Abbildung 5.4: Use-Case Diagramm des Streamer-UI

Das Use Case Diagramm zeigt die unterschiedlichen Anwendungsfälle, welche bei der Benutzung des Audio-Streamers auftreten können. Im wesentlichen gibt es bei allen Anwendungsfällen zwei Akteure. Dies sind der Nutzer des Audio-Streamers (User) und der Server worüber die Daten bezogen werden. Beide Akteure können dabei einen Vorgang aktiv anstoßen.

Hier werden nun die einzelnen Anwendungsfälle im Zusammenhang mit den Akteuren beschrieben:

**Control Playback** ist die Steuerung des Abspielvorgangs. Der User kann: den Song abspielen, pausieren, vor und zurück wechseln. Diese Befehle werden direkt an den Mopidy Server weitergeleitet, wo die Aktion ausgeführt wird.

**Get Track Information** ist die Anzeige der Informationen auf dem Display des Geräts. Der User kann Informationen zum aktuellen Song anzeigen lassen. Die Daten werden von Mopidy geliefert.

**Search for tracks and albums** ist die Suchfunktion, um nach einem bestimmten Titel oder Album zu suchen. Der User kann durch Texteingabe auf dem Display dienst- sowie quellenübergreifend nach Titel oder Album suchen. Die Daten werden Dabei von Mopidy bezogen.

**Select music source** ist die Auswahl der Musikquelle. Der User kann im Hauptmenü über die aktuelle Quelle (lokal oder online) entscheiden. Mopidy liefert dabei die Daten.

**Configure System** ist die Konfiguration des Audio-Streamers. Der User kann Konfigurationen des Audiosystems vornehmen. Dabei sollen insbesondere die Filter im DAC intern konfiguriert werden können.

## Klassendiagramm

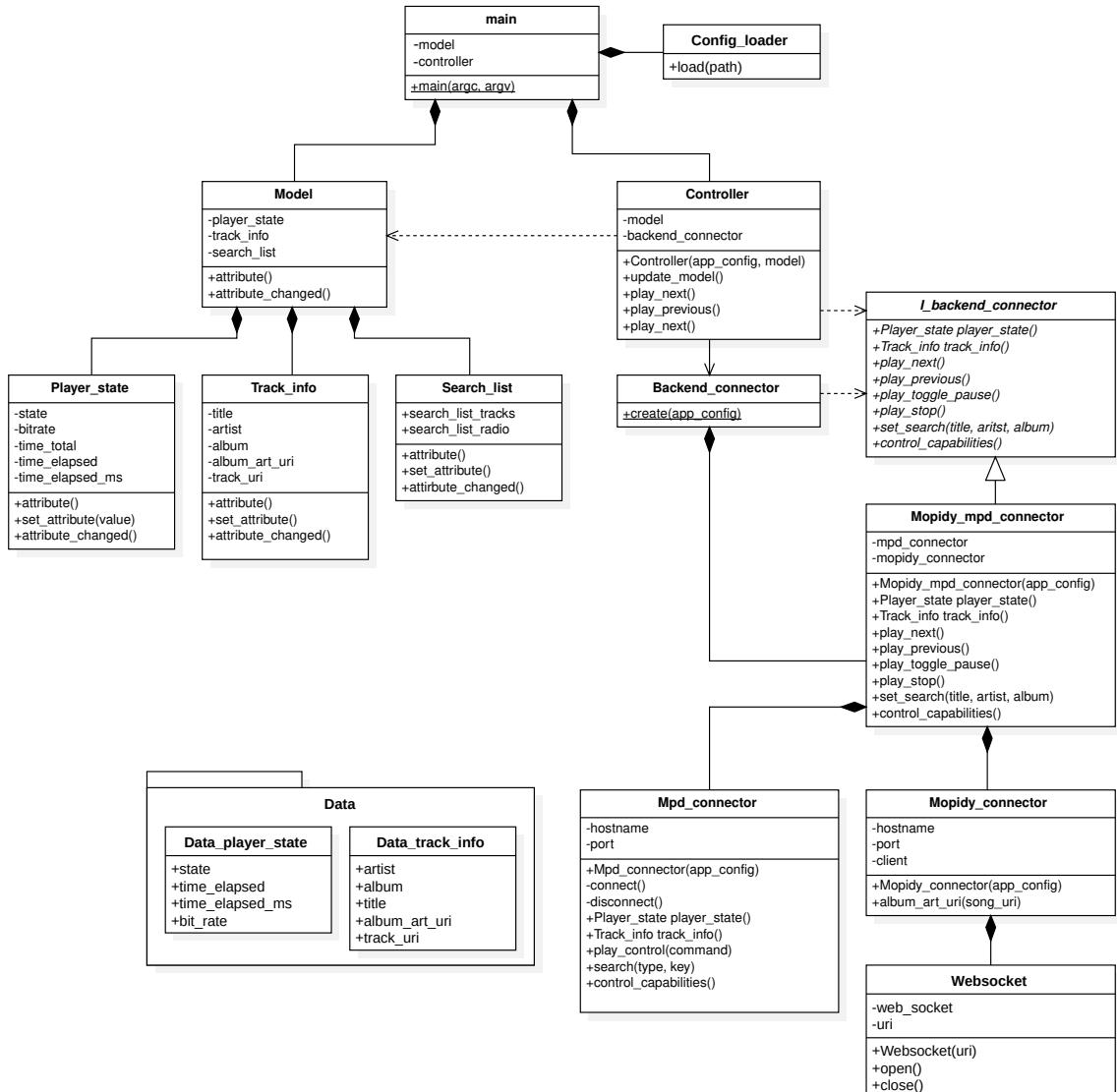


Abbildung 5.5: Klassendiagramm des Streamer-UI

Im Klassendiagramm in Abbildung 5.5 ist die Trennung zwischen Model (links) und Controller (rechts) zu erkennen. Das View ist im Diagramm nicht dargestellt, da dieses mit QML beschrieben wird. Die QML Dateien werden von der QQmlApplicationEngine verarbeitet, diese wird in im main gestartet.

Die einzelnen Models werden mit der Model Klasse zusammengefasst. Die Attribute der Models sind jeweils privat und können nur über Methoden zugegriffen werden. Zu jedem Attribut werden die Methoden zum Setzen und Lesen bereitgestellt. Signale bei Änderungen werden mit `changed()` bekanntgegeben. Im Diagramm sind diese Methoden jeweils verallgemeinert aufgeführt. Die Zugriffsmethoden sind zudem im Qt Framework registriert, somit können sie von QML gelesen werden. Beim Lesen eines Attributs aus QML wird automatisch das Observer Pattern umgesetzt, um ein manuelles Aktualisieren zu verhindern.

Die Controller Klasse fasst alle Controller Funktionen zusammen. Hierbei wurde beachtet, dass nur der Controller das Model kennen muss und nur dieser Code verwendet, der von Qt abhängig ist. Die Klassen, welche den Backend Connector implementieren sollen möglichst unabhängig von Qt sein.

Der Backend Connector ist für die Kommunikation mit den Musikserver verantwortlich. Da verschiedene Protokolle und Musikserver existieren, soll die Schnittstelle möglichst austauschbar bleiben. Dazu bietet sich das Factory-Pattern gemäss [23] an. Die abstrakte Klasse `I_backend_connector` definiert die Schnittstelle, welche der Controller verwenden kann. Wird ein Connector zu einem Musikserver implementiert, müssen diese Schnittstellen umgesetzt werden. Hier ist die `Mopidy_mpd_connector` Klasse die Implementation. Will der Controller einen neuen Connector erzeugen tut er dies über die Factory (`Backend_connector`). Die Factory liest aus der `app_config`, welche Implementation der Schnittstelle es zu instanziieren hat. Das instanzierte Objekt erhält der Controller über die abstrakte Schnittstelle zurück.

Für den Zugriff auf den Mopidy Server werden zwei Protokolle verwendet. In erster Linie erfolgt die Kommunikation über die `libmpdclient` Bibliothek, diese ist in der `Mpd_connector` Klasse verpackt. Da MPD keine Cover Bilder unterstützt, werden diese über den `Mopidy_connector` geladen. Das Mopidy Protokoll läuft über Websockets, dieser Kommunikationsschnittstelle ist mit einer eignen Klasse abgekapselt.

Informationen, die gruppiert zwischen den Klassen verschoben werden, sind in den `Data_` Klassen abgebildet. Diese sind im in der Abbildung im Data Package gruppiert. Die enthaltenen Klassen dürfen aus allen Controllerklassen verwendet werden. Die Abhängigkeiten wurden zur erhöhten Übersichtlichkeit nicht eingezzeichnet.

Die Informationen zum Musikserver (Hostname, Port) können für das Streamer-UI in einer Konfigurationsdatei abgespeichert werden. Diese wird zu Beginn der Applikation geladen und den Klassen zur Verfügung gestellt. Jede Klasse ist selbst dafür verantwortlich ihre Konfigurationsdaten auszulesen. Dadurch sind die Konfigurationsdaten eng an die Klassen gekoppelt und leicht erweiterbar. Für die Datei wird das JSON Format verwendet.

## 5.3 CMake Build System

### 5.3.1 Übersicht

CMake ist ein Open Source System, mit dem Build Prozesse gemanagt werden. Das System ist vom Compiler und von der Entwicklungsumgebung unabhängig [28]. Beim Ausführen von CMake werden die Make-Dateien erzeugt und der gewählte Compiler aufgerufen. Der Vorteil von CMake besteht darin, dass ein Projekt in QtCreator, CLion oder auf der Kommandozeile compiliert werden kann. Da CMake heute als de-facto Standard bei Open Source Bibliotheken genutzt wird, können diese mit minimalen Aufwand in ein Projekt integriert werden.

Ein typischer Buildvorgang in der Kommandozeile ist nachfolgend abgebildet.

```
mkdir build && cd build  
cmake ..  
make
```

Listing 5.1: Buildvorgang mit CMake in einem bestehenden Projektordner (`CMakeLists.txt` bereits existierend)

CMake generiert dabei zuerst die Make-Dateien, mit make kann der Code dann kompiliert werden.

Des Weiteren kann in CMake mit Toolchain-Dateien einfach zwischen verschiedenen Prozessorarchitekturen (x86, ARM) gewechselt werden. Aufgrund der hohen Verbreitung bei Open Source Projekten und dem simplen Toolchain-Management wird CMake in diese Arbeit verwendet.

### 5.3.2 Toolchain

In einer Toolchain-Datei werden die Pfade zum Debian-Dateisystem, sowie den Compiler aus der Linaro Toolchain spezifiziert. Die Toolchain-Datei für den Audio-Streamer ist in gekürzter Form in Listing 5.2 aufgeführt.

```
#####
# User Paths
#####
set(BASE_PATH /opt/audio-streamer/tools)
set(TOOLCHAIN_PATH ${BASE_PATH}/linaro/gcc-linaro-6.2.1-2016.11-x86_64_arm-linux-
    ↳ gnueabihf)
set(CMAKE_SYSROOT ${BASE_PATH}/rootfs/debian)

# Target
set(CMAKE_SYSTEM_NAME Linux)
set(CMAKE_SYSTEM_PROCESSOR arm)

# Toolchain
set(PLATFORM arm-linux-gnueabihf)
set(CMAKE_C_COMPILER ${TOOLCHAIN_PATH}/bin/${PLATFORM}-gcc)
set(CMAKE_CXX_COMPILER ${TOOLCHAIN_PATH}/bin/${PLATFORM}-g++)

...
# Sysroot library paths
link_directories(
    ${CMAKE_SYSROOT}/usr/lib/${PLATFORM}
    ${CMAKE_SYSROOT}/lib/${PLATFORM}
    ${CMAKE_SYSROOT}/local/qt5/lib
)
```

Listing 5.2: bfh-audio-streamer.cmake

Der Buildvorgang mit der Toolchain unterscheidet sich nur in einem Parameter vom normalen Build:

```
mkdir build && cd build
cmake -DCMAKE_TOOLCHAIN_FILE=bfh-audio-streamer.cmake ..
make
```

Listing 5.3: Buildvorgang mit CMake mit einer Toolchain

Die Anwendung kann danach mit scp oder rsync auf den Audio-Streamer geladen werden.

## 5.4 Graphical User Interface

Bis zum Abschluss der Arbeit wurde im Streamer-UI das Hauptmenü und die *Now Playing* Anzeige implementiert. Bei der QML Implementation handelt es sich um eine Neuentwicklung ohne Vorlage. Die Applikation kann durch Hinzufügen neuer views ergänzt werden.

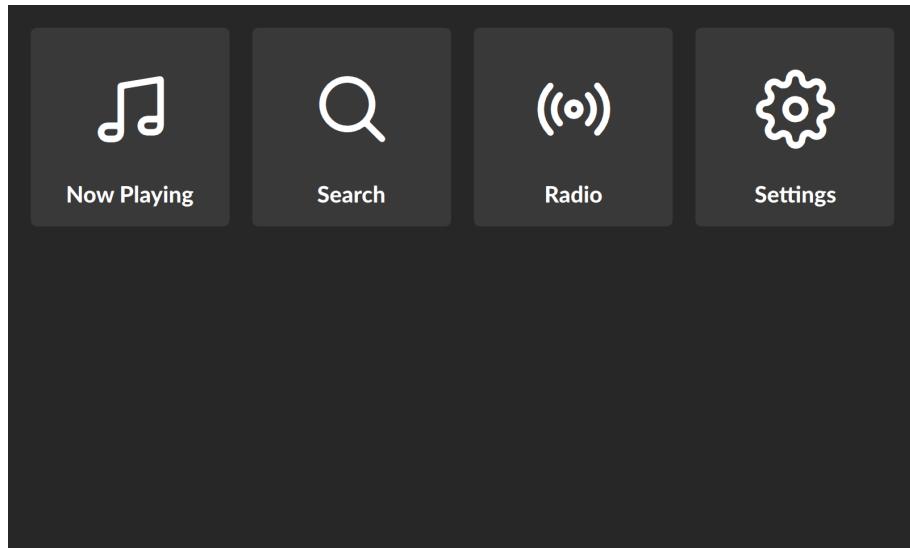


Abbildung 5.6: Streamer-UI Hauptmenü

Die Elemente des Hauptmenüs des Streamer-UI sind in Abbildung 5.6 zu sehen. Die Optionen *Search*, *Radio* und *Settings* dienen lediglich zur Visualisierung des Menüs und haben keine Funktionalität implementiert. Durch Tippen auf den Now Playing Menüpunkt wechselt die Anwendung auf die neue Anzeige.

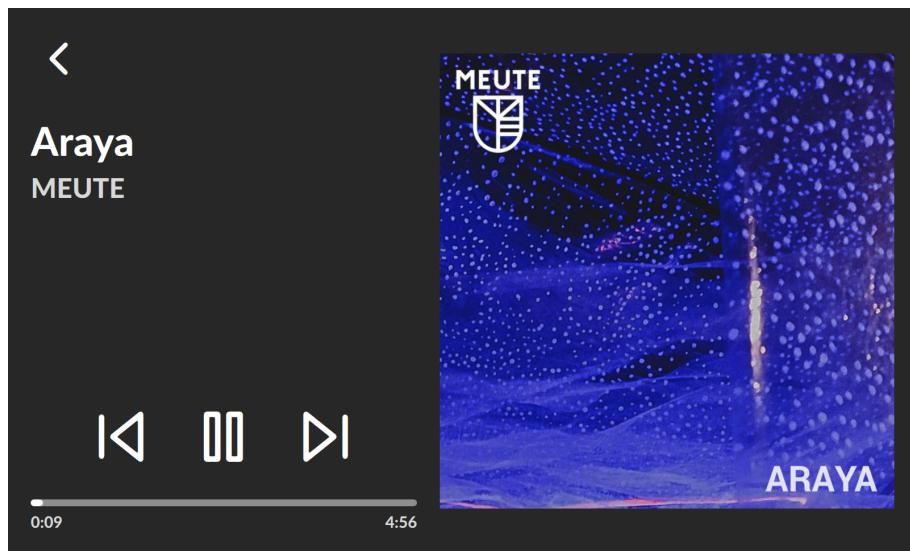


Abbildung 5.7: Streamer-UI Informationen zum aktuellen Titel

Auf Now Playing Anzeige in Abbildung 5.7 wird dem Nutzer der Title, Interpret, das Cover und ein Fortschrittsbalken des Musikstückes angezeigt. Der Nutzer kann die Wiedergabe starten und pausieren, sowie zum nächsten oder vorherigen Stück springen. Über den zurück-Pfeil oben links gelangt man ins Hauptmenü.

# 6 Messungen & Tests

Die Funktionsweise des Audio-Streamers muss qualitativ und quantitativ überprüft werden. Nur so kann das korrekte Verhalten der Hardware und Software nachgewiesen werden. Das Vorgehen und diese Resultate dieser Messungen und Test sind in diesem Kapitel aufgeführt.

## 6.1 Material & Methoden

Nachfolgend werden die einzelnen Messaufbauten und das Vorgehen bei den Messungen und Tests beschrieben. Dadurch wird die Reproduzierbarkeit des Resultate sichergestellt.

### 6.1.1 Material

	Bezeichnung	Hersteller	Typ	Inventarnummer
P <sub>1</sub>	Oszilloskop	Rohde & Schwarz	RTO1044	-
zu P <sub>1</sub>	Differenzielle Sonde	Rohde & Schwarz	RT-ZS60	-
P <sub>2</sub>	Multimeter	Fluke	87-V	T201-Nr4
P <sub>3</sub>	Audio Analyzer	NTI Audio	FX100	-
P <sub>4</sub>	Audio Analyzer	NTI Audio	XL2	PM0027-T211-4H
-	Mainboard V1.0 (mod. V1.1)	BFH	-	-
-	PSU V1.0 (mod. V1.1)	BFH	-	-

Tabelle 6.1: Verwendentes Material für die Messungen und Tests

Für die Messungen wurde die Geräte aus Tabelle 6.1 verwendet. Bei Wiederholung der Messungen können äquivalente Geräte verwendet werden.

Die Leiterplatten des Audio-Streamer entstammen der PCB Version 1.0 jedoch mit den Modifizierungen, welche in die Schema Version 1.1 einfließen.

### 6.1.2 Spannungsversorgungen

Für die Überprüfung der Funktionalität der Spannungsversorgungen müssen die DC Spannungen, die Ströme und die Spannungsripple gemessen werden.

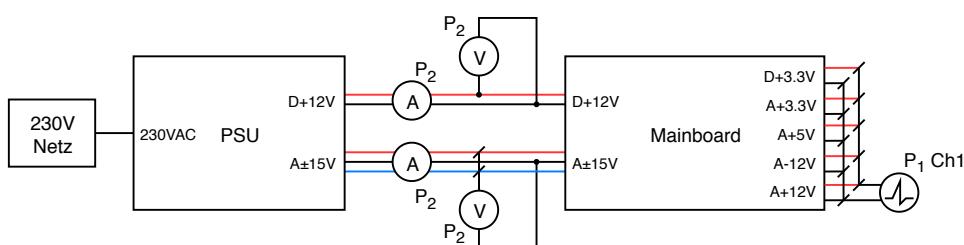


Abbildung 6.1: Messaufbau Spannungsversorgungen

Wie in Abbildung 6.1 ersichtlich wird dazu mit einem Multimeter nacheinander die einzelnen Ströme und Spannungen zwischen der PSU und dem Mainboard gemessen.

Für die Messung der Spannungsripple wird mit einer differenziellen Sonde verwendet. Da der analog Teil des Mainboards vom 230 V-Netz galvanisch getrennt ist, kann keine normale Sonde verwendet werden. Eine normale Sonde könnte durch die Erdung des Außenleiters zu einer Erdschleife und somit zu Messfehler führen. Die Ripple werden direkt nach den Spannungsregler mit den vorgesehenen Testpunkten gemessen. Die Messungen mit dem Oszilloskop wurden den Einstellungen aus Tabelle 6.2.

Einstellung	Wert
Bandwidth	20 MHz
Acquisition	Real-Time
Interpolation	Sample Hold
Decimation	Sample

Tabelle 6.2: Einstellungen des Oszilloskops zur Messungen der Spannungsripple

Die Messungen sind im *aktiven* Zustand durchzuführen, d.h. der Musikserver ist aktiv und gibt eine Audio-Stream auf den analog Ausgang.

### 6.1.3 Signalqualität

Die Signalqualität der I<sup>2</sup>S Leitungen hat einen Einfluss auf die Audioqualität. Insbesondere soll die Qualität des Master Clock (MCLK) überprüft werden, da dieser Takt vom DAC als Systemtakt verwendet wird.

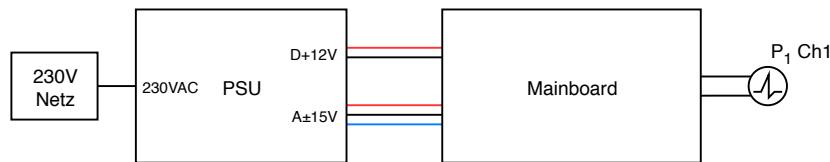


Abbildung 6.2: Messaufbau Signalqualität

Der Aufbau für die Messungen gestaltet sich, wie Abbildung 6.2 dargestellt, einfach. Die PSU wird mit dem Mainboard verbunden, die Messungen werden wiederum mit einer differenziellen Sonde direkt auf der Leiterplatte durchgeführt. Für die Taktquellen sind dafür Testpunkte vorgesehen.

### 6.1.4 Audio Messungen

Das Ziel des Audio-Streamers ist die verlustfreie Wiedergabe von Musikstücken in CD-Qualität. Dies soll messtechnisch überprüft werden.

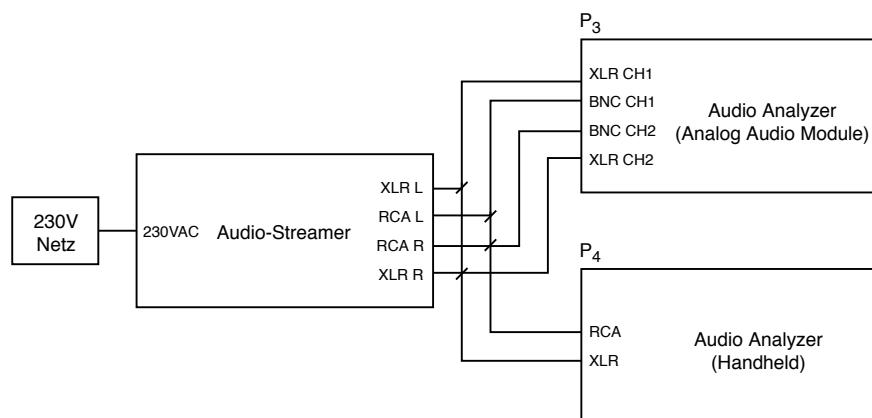


Abbildung 6.3: Messaufbau Audio

Die Abbildung 6.3 zeigt, dass dazu zwei verschiedene Messgeräte verwendet werden können. Einerseits können mit einem Audio Analyzer Gerät (hier wird der NTI FX100 verwendet) oder ein Handheld Audio Analyzer (NTI XL2) genutzt werden. Das Handheld Gerät ist einfacher zu bedienen und 50 Hz Probleme durch Masseschleifen sind ausgeschlossen. Der FX100 ist jedoch präziser, deshalb sind dessen Messwerte zu bevorzugen. Zudem können alle Ausgänge simultan an den FX100 angeschlossen werden, was die Zeit zwischen den Messungen verringert.

Mit den Audiomessungen sollen den SNR, THD+N, Frequenzgang und Crosstalk gemessen werden. Dazu wird auf dem FX100 ein GlideSweep als Testsignal verwendet. Dabei wird am Audioausgang die Frequenz monoton gesteigert. Somit kann das Frequenzspektrum innerhalb von 3 s durchgemessen werden. Dieses Signal kann direkt mit der Software des FX100 generiert werden. Die Testsignale und die Konfigurationsdatei für den FX100 sind im digitalen Anhang zu finden.

Bei der Verwendung der single-ended Anschlüsse des Analog Audio Module des FX100 muss zwingend der Erdungsanschluss des Moduls mit dem Außenleiter der BNC Anschlüsse verbunden werden. Der single-ended Anschluss ist ansonsten nicht geerdet, was das Empfangen 50 Hz-Signale zur Folge hat.

### 6.1.5 CoM Peripherie Funktionstest

Die einzelnen Komponenten auf dem digitalen Teil des Mainboards werden mit Funktionstests überprüft. Diese Tests werden aus der Kommandozeile des Linux-Systems durchgeführt.

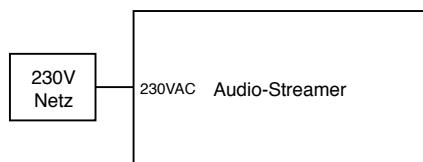


Abbildung 6.4: Aufbau Funktionstests

Für die Test wird lediglich der Netzanschluss und eine Ethernet-Verbindung benötigt. Der Aufbau ist in Abbildung 6.4 ersichtlich.

## 6.2 Ergebnisse & Diskussion

Nachfolgend werden die Ergebnisse der einzelnen Messungen erläutert und kommentiert.

### 6.2.1 Spannungsversorgungen

Versorgung	Spannung	Strom	Leistung
A+15V	13.21 V	156.2 mA	2.06 W
A-15V	-13.67 V	-30.4 mA	0.41 W
D+12V	12.18 V	410.0 mA	4.99 W

Tabelle 6.3: Spannungen und Ströme der PSU

Aus den Messungen aus Tabelle 6.3 wird deutlich, dass die Spannungen der  $\pm 15$  V Speisungen mit ca.  $\pm 13$  V deutlich unter der angegebenen Spannung liegt. Dies liegt daran, dass der Transformator bei der Inbetriebnahme durch einen anderen Typ ersetzt wurde. Eine zu hohe Leerlaufspannung war der Grund für den Wechsel. Weiter fällt auf, dass die positive analoge Speisung mehr Strom liefern muss als die negative. Dies wird auch bei der Leistungsaufnahme deutlich.

Die negative Spannungsversorgung wird weniger belastet, weil nur die Operationsverstärker diese Versorgung nutzen. Der DAC belastet ausschließlich die positive Versorgung. Die tiefere Stromaufnahme führt zudem zu einer höheren Spannung, da der Transformator weniger belastet wird.

Die digitale Speisung (D+12V) versorgt das CoM. Dieses bezieht ca. 5 W im Betrieb. Dieser Wert ist für einen Einplatinencomputer zu erwarten.

<b>Speisung</b>	<b>Spannung</b>	<b>Power Good</b>	<b>Resultat</b>
<b>A+12V</b>	12.07 V	High	PASS
<b>A-12V</b>	-12.13 V	High	PASS
<b>A+5V</b>	4.988 V	High	PASS
<b>A+3.3V</b>	3.335 V	High	PASS
<b>D+3.3V</b>	3.278 V	Low	PASS

Tabelle 6.4: Analoge Speisungen (Mainboard)

In Tabelle 6.4 sind die Spannungen der Speisungen und die Power Good Signale aufgeführt. Letzteres signalisiert dem CoM, ob die Spannungsversorgung läuft. Die maximale Abweichung der Spannungen beträgt 35 mV. Bei den Power Good Signalen ist darauf zu achten, dass der Ausgang der Spannungsüberwachung der D+3.3V Versorgung Low-aktiv ist.

<b>Signal</b>	<b>RMS</b>	<b>Peak-Peak</b>	<b>Scale</b>
<b>+15V</b>	13.205 V	208.52 mV	400 mV/div
<b>-15V</b>	-13.656 V	126.48 mV	400 mV/div
<b>A+12V</b>	12.052 V	110.67 mV	400 mV/div
<b>A-12V</b>	-12.135 V	110.67 mV	400 mV/div
<b>A+5V</b>	5.1496 V	7.1146 mV	10 mV/div
<b>A+3.3V</b>	3.3308 V	4.7431 mV	10 mV/div
<b>D+3.3V</b>	3.275 V	4.7431 mV	10 mV/div
<b>GND</b>	4.7431 mV	2.0062 mV	10 mV/div

Tabelle 6.5: Spannungsripple der Spannungsversorgungen

Die Tabelle 6.5 zeigt noch einmal die DC Spannungen, jedoch diesmal mit der Differenzialsonde gemessen. Da die Sonde nur DC-Kopplung unterstützt, musste die Skalierung der Spannungs-Achse angepasst werden. Zudem können nur Peak-Peak Ripple gemessen werden.

Die Ripple auf den  $\pm 15$  V Leitungen liegen im erwarteten Bereich. Bei den  $\pm 12$  V Speisungen ist der Wert deutlich höher als erwartet. Denn die verwendeten Linearregler haben bei 100 Hz eine Unterdrückung von bis zu 120 dB. Die Ripple auf den restlichen Speisungen sind mit einigen Milliampere tief, aber deutlich höher als erwartet. Es zeigt sich zudem ein grosser Unterschied zwischen Messungen in der 400 mV/div Skala und der 10 mV/div Skala. Zuletzt ist eine Messung mit einer kurzgeschlossenen Sonde aufgeführt. Diese hat einen Ripple von 2 mV.

Aus der Messung der kurzgeschlossenen Sonde und den Unterschieden bei verschiedenen Skalen lässt sich schliessen, dass die Messungen nicht genau sind. Messungen mit Spannungen unter einem 1 mV sind schwierig ohne Messfehler durchzuführen, insbesondere, wenn die minimale Skalierung 10 mV/div beträgt. Daher sind aus der Tabelle nur die Ripple der  $\pm 15$  V Leitungen relevant.

## 6.2.2 Signalqualität

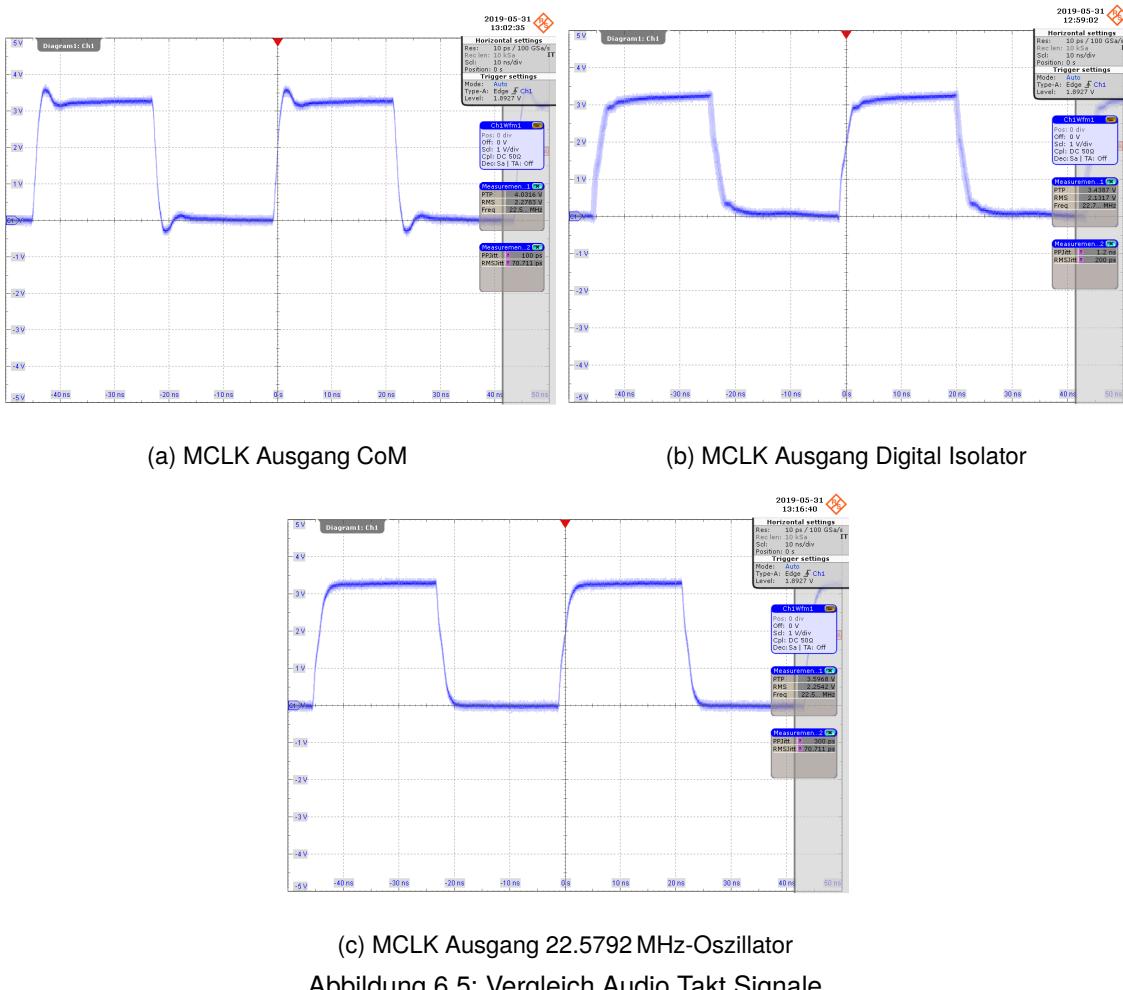


Abbildung 6.5: Vergleich Audio Takt Signale

Die Abbildung 6.5 zeigt den Master Clock (MCLK) der I<sup>2</sup>S-Schnittstelle an verschiedenen Punkten in der Schaltung. In (a) ist der Ausgang des CoM, welches den Takt generiert, dargestellt. Dieser Takt wird danach über den digitalen Isolator zu den DACs gebracht. Das Signal auf der DAC Seite ist in (b) dargestellt. Es durch die Unschärfe deutlich ein Jitter im Signal zu erkennen.

Als Vergleich dazu ist in (c) das Signal einer der Oszillatoren auf dem DAC Teil abgebildet. Dieser wird momentan nicht verwendet, da das MCLK Signal vom CoM generiert wird.

Messpunkt	RMS	Peak-Peak
CoM	70.7 ps	100 ps
DAC	200 ps	1.2 ns
Oszillator	70.7 ps	300 ps

Tabelle 6.6: Zusammenfassung MCLK Jitter

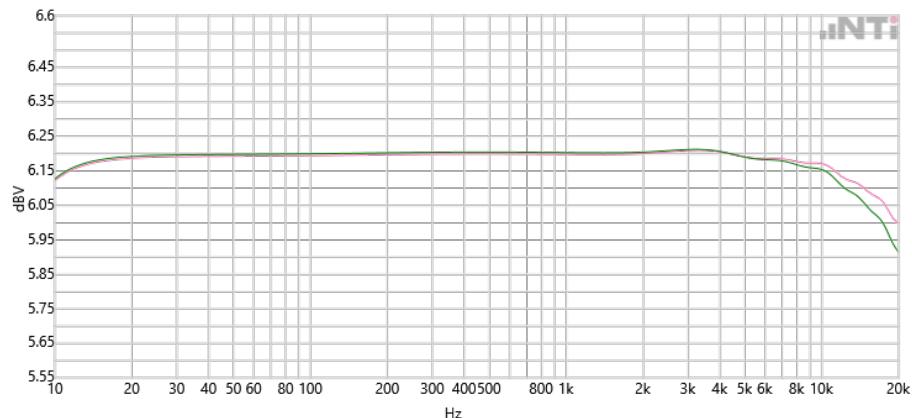
Eine Zusammenfassung der Jitter-Zeiten ist in Tabelle 6.6 aufgeführt. Es wird wiederum verdeutlicht, dass das MCLK Signal nach dem digital Isolator den höchsten Jitter aufweist.

Der RMS Jitter des Oszillators liegt deutlich über den angegebenen 1 ps des Herstellers [29]. Ein Grund für die Abweichung könnte in der unterschiedlichen Konfiguration der Messgeräte liegen.

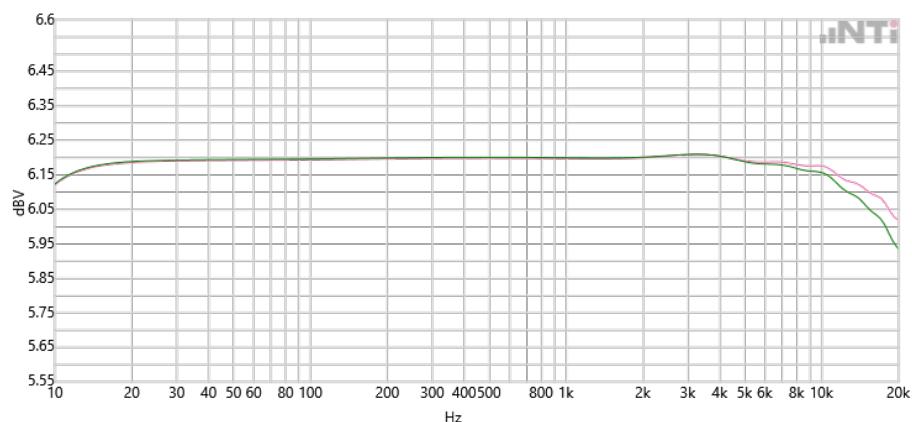
Der MCLK in (c) hat sichtlich weniger Störungen als derjenige in (b) und wäre somit für den DAC besser geeignet. Der Einfluss auf den analogen Audioausgang muss aber noch geprüft werden.

### 6.2.3 Audio Messungen

#### Level (FX100)



(a) single-ended (RCA)



(b) Differenziell (XLR)

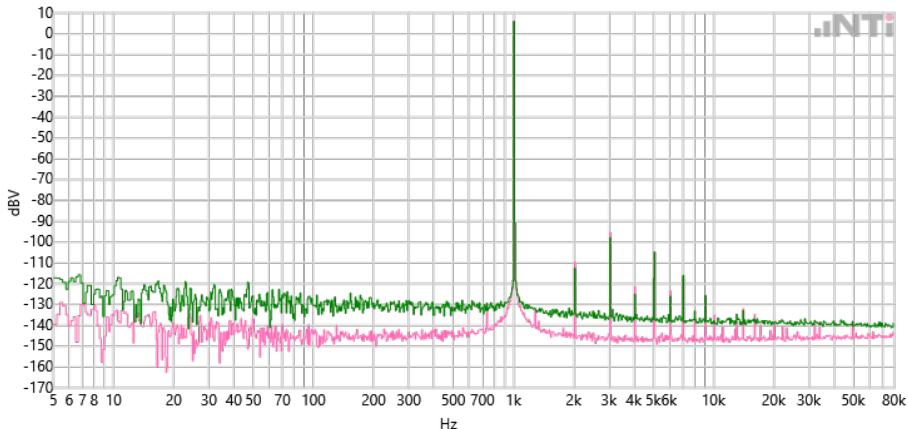
Abbildung 6.6: Frequenzgang der Audioausgänge; grün: rechter Kanal, rosa: linker Kanal

Aus den beiden Frequenzgängen in Abbildung 6.6 ist ersichtlich, dass die Abweichungen im Bereich von 20 Hz bis 4 kHz minimal sind. Von 4 kHz bis 20 kHz nimmt der Level um 0.25 dB ab. Weiter wird deutlich, dass die differenziellen und single-ended Ausgänge praktisch denselben Frequenzgang haben.

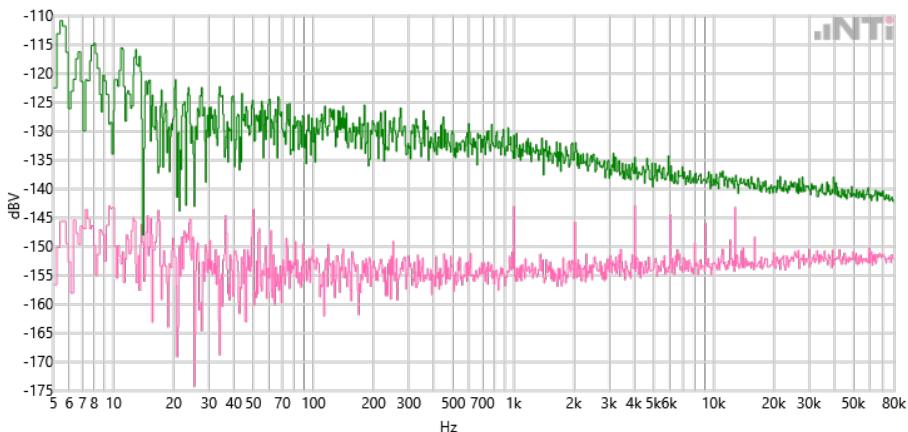
Die Abweichungen unterhalb 20 Hz sind wahrscheinlich auf Messgenauigkeiten mit dem GliedSweep Verfahren zurückzuführen.

Ein Vergleich der Messresultate mit den Simulationen aus der Projektstudie [1] zeigt, dass die Abweichung maximal 0.1 dB beträgt. Das Filter verhält sich somit wie erwartet und erfüllt seine Aufgabe zuverlässig.

## SNR & THD+N (FX100)



(a) Frequenzspektrum bei 1 kHz Sinussignal



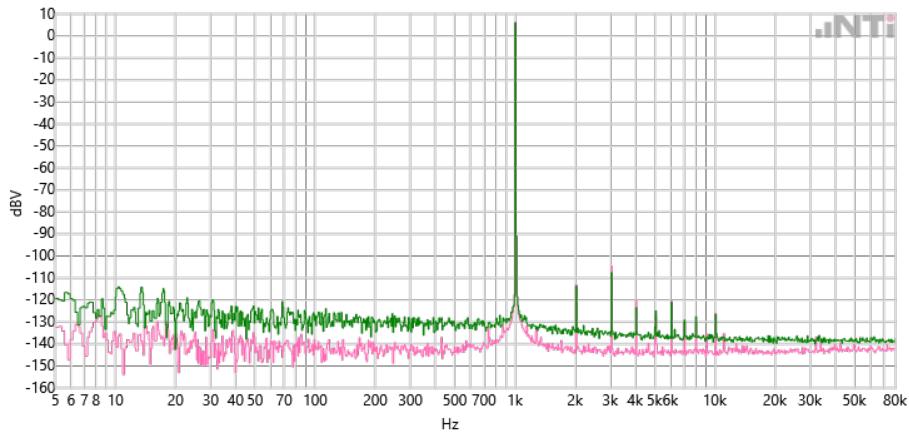
(b) Frequenzspektrum bei 0 Hz Signal

Abbildung 6.7: Frequenzspektren der single-ended Ausgänge (RCA); grün: rechter Kanal, rosa: linker Kanal

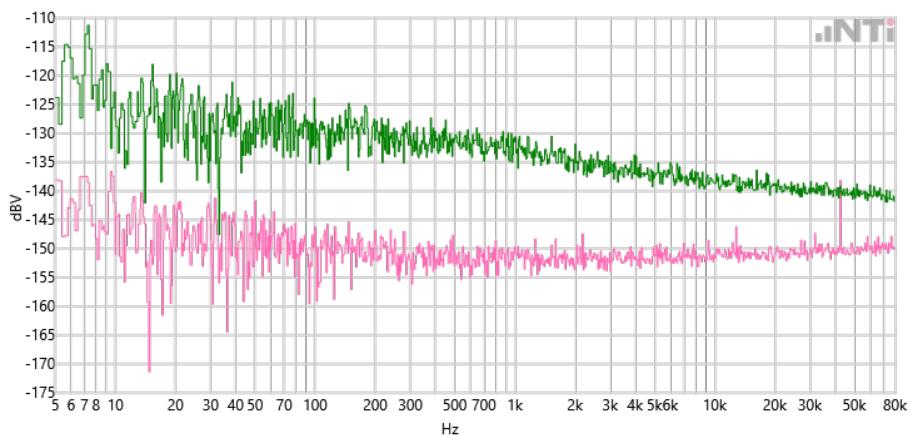
In Abbildung 6.7 (a) ist die FFT eines 1 kHz-Signals bei maximaler Amplitude dargestellt auf den single-ended Ausgängen. Die Harmonischen des Signals sind deutlich bei 2 kHz, 3 kHz, 4 kHz, usw. zu erkennen. Aus der Amplitude des 1 kHz und dessen Harmonischen lässt sich der Klirrfaktor (THD+N) berechnen. Der Wert ist in Tabelle 6.7 aufgelistet.

Das Grundrauschen des Audioausgangs ist in (b) ersichtlich. Dazu wurde eine 0-Folge als Signal ausgegeben. Aus der Differenz des Signalpegels in (a) und (b) lässt sich der Störabstand (SNR) berechnen. Dieser ist wiederum in Tabelle 6.7 aufgeführt.

Bei beiden Messungen ist ein Unterschied zwischen dem linken und rechten Kanal zu erkennen. Ein Messfehler wurde durch vertauschen der Anschlüsse ausgeschlossen. Der Grund dafür könnte im Layout liegen. So ist die Speisung der beiden DACs kaskadiert. Dadurch könnten Störungen vom linken Kanal auf den rechten übersprechen. Interessant ist jedoch, dass sich die Amplituden der Harmonischen des linken Kanals zum Teil höher sind, als diejenigen des rechten. Ein anderer Grund könnte die Streuung der Qualität der Operationsverstärker sein. Die konkrete Lokalisierung des Problems bedarf weitere Abklärung.



(a) Frequenzspektrum bei 1 kHz Sinussignal



(b) Frequenzspektrum bei 0 Hz Signal

Abbildung 6.8: Frequenzspektren der differenziellen Ausgänge (XLR); grün: rechter Kanal, rosa: linker Kanal

Die Spektren der differenziellen Ausgängen decken sich mit denen der single-ended Ausgängen. Dies ist nicht weiter erstaunlich, da sich bei den Messungen nur das Kabel unterscheidet.

RCA/XLR	SNR		THD+N		
	Signal	Right	Left	Right	Left
RCA		99.59 dB	113.3 dB	-98.16 dB	-99.41 dB
XLR		101.2 dB	114.4 dB	-98.23 dB	-102.4 dB

Tabelle 6.7: Zusammenfassung SNR und THD+N (alle Harmonischen bis 80 kHz)

Der Unterschied im Rauschen zwischen dem linken und rechten Kanal wird in Tabelle 6.7 noch einmal verdeutlicht. Dies hat jedoch keinen Einfluss auf den Klirrfaktor da die Harmonischen über das Grundrauschen ragen.

Mit einem SNR von grösser 98 dB wurde das in der Projektstudie gesetzte Ziel erreicht. Das Quantisierungsrauschen einer CD (16 Bit) ist somit höher als der SNR des Audio-Streamers, dadurch bleiben die Informationen in der digitalen Datei auch im analogen Audiosignal erhalten.

### SNR & THD+N (XL2)

Mit dem Handheld Analyzer wurden die Messwerte des FX100 verifiziert. Zudem wurden die Messungen mit Gewichtungen durchgeführt, um die Werte mit dem Datenblatt des Evaluation

Kit AKD4493 [30] vergleichen zu können.

RCA Signal	A-Weighting		Z-Weighting	
	Right	Left	Right	Left
<b>1 kHz</b>	6.2 dBV	6.2 dBV	6.2 dBV	6.2 dBV
<b>0 Hz</b>	-98.4 dBV	-114.6 dBV	-96.1 dBV	-112.0 dBV
<b>SNR</b>	104.6 dB	120.8 dB	102.3 dB	118.2 dB

Tabelle 6.8: SNR des single-ended Ausgangs (RCA)

XLR Signal	A-Weighting		Z-Weighting	
	Right	Left	Right	Left
<b>1 kHz</b>	6.1 dBV	6.1 dBV	6.1 dBV	6.1 dBV
<b>0 Hz</b>	-98.4 dBV	-115.4 dBV	-96.3 dBV	-113.0 dBV
<b>SNR</b>	104.5 dB	121.5 dB	102.4 dB	119.1 dB

Tabelle 6.9: SNR des differenziellen Ausgangs (XLR)

Die Messwerte für den SNR der aller Ausgänge in Tabelle 6.8 und Tabelle 6.9 weichen von den vorherigen Messungen um bis zu 7 dB ab. Dies liegt an den Gewichtungen, welche für die Messung der Level durchgeführt wurde.

Im Vergleich zum AKD4493 ist der SNR mit A Gewichtung beim rechten Kanal 20 dB beim linken Kanal 4 dB tiefer.

RCA/XLR Signal	A-Weighting		Z-Weighting	
	Right	Left	Right	Left
<b>RCA</b>	-97.2 dB	-98.2 dB	-96.1 dB	-97.2 dB
<b>XLR</b>	-98.4 dB	-98.5 dB	-96.2 dB	-97.4 dB

Tabelle 6.10: THD+N bei 1 kHz aller Ausgänge

Der Klirrfaktor in Tabelle 6.10 deckt sich mit den vorherigen Messungen.

### Crosstalk (FX100)

Für die Messung des Übersprechens wurde jeweils ein 1 kHz auf den einen Kanal, eine Nullfolge auf den anderen gegeben. Mit einem Marker wurde Differenz bei 1 kHz berechnet.

Signal Right	Left	L <sub>Peak</sub> @1 kHz		
		Right	Left	Δ
1 kHz	0 Hz	5.2 dBV	-121.4 dBV	126.6 dB
0 Hz	1 kHz	-140.7 dBV	3.9 dBV	144.6 dB

Tabelle 6.11: Crosstalk bei 1 kHz auf dem XLR Ausgang

Die Werte in Tabelle 6.11 verdeutlichen wiederum den Unterschied zwischen den beiden Kanälen. Es gilt jedoch zu beachten, dass der -126.4 dBV bei der ersten Messung dem Grundrauschen entspricht. Es kann daher nicht auf direktes Übersprechen des linken Kanals geschlossen werden.

#### 6.2.4 CoM Peripherie Funktionstest

Die Funktionstest der Peripherie dienen in erster Linie der Verifizierung der Hardware. Daher wurden etliche Tests über die Linux Kommandozeile durchgeführt. Oft reichen rudimentäre Test wie `lspci` um zu überprüfen, ob ein Peripheriebaustein vom System erkannt wird.

Komponente	Schnittstelle	Test	Resultat
X1100	Ethernet	Verbindung (SSH)	PASS
X1003	USB 2.0 (intern)	Memory Stick beschreiben	PASS
X1002	USB 2.0 (extern)	Memory Stick beschreiben	PASS
X1000	USB 2.0 (extern)	Memory Stick beschreiben	PASS
J1500	Virtual COM Port	Serial Port Login	PASS
X1300	Parallel Display	Bildausgabe	PASS
X1301	Display I <sup>2</sup> C	i2cdetect	PASS
X1400	HDMI	Bildausgabe	FAIL
X1600	SATA (SSD)	SSD beschreiben	PASS
X1700	PCIe (WiFi)	Intel AC 7265 <code>lspci</code>	PASS
		Intel AC 7265 <code>lsusb</code>	PASS
		Intel AC 7265 WiFi Verbindung	PASS
		Intel AC 7265 BLE Verbindung	TBD

Tabelle 6.12: Zusammenfassung der Funktionstests der digitalen Schnittstellen

Die Resultate der Tests sind Tabelle 6.12 zusammengefasst. Eine Schnittstelle ist jedoch noch nicht komplett funktionstüchtig: Die HDMI Schnittstelle benötigt eine Anpassung der Leiterplatte (Tabelle 3.1).

### 6.3 Qualitative Audiotests bei Dynavox

Die Hörttest wurden bei Dynavox Electronics SA mit Unterstützung der Inhaber Pascal und Ivo Aebischer durchgeführt.

Dies Testanlage ist in Abbildung 6.9 ersichtlich. Es wurden Lautsprecher, Vorverstärker und Endverstärker der Marke Rowen verwendet. Der Audio-Streamer wurde im direkten Vergleich zum Bluesound Node i2 getestet. Der Bluesound ist ein Streamer, welcher ca. 600.- Fr. kostet. Für den Hörttest wurden Stücke aus verschiedenen Genre auf beiden Streamer gleichzeitig abgespielt. Durch Umschalten des Eingangs des Vorverstärkers konnten die Streamer direkt verglichen werden.

Der Audio-Streamer vermochte die tiefen Töne präziser nachzubilden als der Bluesound. Die Räumlichkeit wurde mit dem Bluesound jedoch besser erhalten. So konnte die Stimme des Sängers beim Bluesound besser auf einen Punkt geortet werden. Beim Audio-Streamer war die Position etwas versetzt und leicht verfärbt. Insgesamt vermochte der Audio-Streamer zu überzeugen und zeigt im Vergleich zu etablierten Streamern nur geringe Abweichungen.



Abbildung 6.9: Testanlage der Dynavox Electronics SA [31]



# 7 Fazit

## 7.1 Auswertung Ziele

### 7.1.1 Hardware

Die Hardware ist funktionsfähig. Das CoM mit Peripherie, die DACs und die Operationsverstärker werden mit den korrekten Spannungen versorgt. Die Kommunikation zwischen dem CoM und den DACs ist mit den digitalen Isolatoren sichergestellt und wurde messtechnisch überprüft. Bei den Schnittstellen des CoM wurden USB, Ethernet, Virtual COM-Port, PCIe mittel WiFi-Modul, SATA mittels SSD und der parallel Display Port erfolgreich überprüft. Einzig die HDMI Schnittstelle konnte aufgrund eines falschen Footprints nicht getestet werden. Nebst dem falschen Footprint wurden weitere Schaltungsfehler entdeckt und wenn möglich korrigiert. Die Fehler und vorzunehmenden Korrekturen sind aufgelistet.

Die Messungen der Signalqualität zeigen, dass der digitale Isolator durchaus zu einer Erhöhung des Signaljitters führt. Jedoch konnte der Einfluss auf die Audio-Qualität nicht mehr überprüft werden.

Mit der messtechnischen Überprüfung des Audio-Ausgangs konnte nachgewiesen werden, dass der Audio-Streamer mit einem SNR von >99 dB eine Audiodatei in CD-Format verlustfrei wiedergeben kann. Der Klirrfaktor beträgt dabei -98 dB. Nebst den Messungen im statischen Zustand, wurde das dynamische Verhalten mit Hörtests überprüft. Auch hier vermochte der Audio-Streamer zu überzeugen. Verbesserungspotential besteht jedoch beim stereo imaging, welches zeigt wie präzise Komponenten in einem Lied örtlich wiedergegeben werden.

Nebst Elektronik wurde das Gehäuse mit einem CAD-Tool vermasst und bearbeitet. Die Bedienelemente konnten alle befestigt werden, jedoch könnte der Spalt um das Display herum verringert werden.

### 7.1.2 Linux System

Bezüglich des Linux Betriebssystems wurden die Ziele weitgehendst erfüllt. Über das System können die Hardwarekomponenten des Audio-Streamers betrieben werden. Durch die Treiber ist das Ansprechen aus dem User Space und damit durch die Software Streamer-UI möglich.

Das Display kann für die Anzeige von Informationen der Applikation Streamer-UI verwendet werden. Das Backlight funktioniert und kann aus dem User Space konfiguriert werden. Über den Touchscreen können vom Benutzer Eingaben entgegengenommen werden.

Mit dem GUI-Framework Qt konnte erfolgreich die Basis für die Entwicklung einer Applikation gelegt werden, welche Daten über den Framebuffer auf das Display ausgeben kann. Das EGLFS System von Qt, welches über das OpenGL Interface auf die GPU des iMX6 zugreift, konnte noch nicht in Betrieb genommen werden. Dies, weil ein Problem mit den nötigen Abhängigkeiten vorliegt. Die erforderlichen Bibliotheken sind nur schwer zugänglich. Durch einen Yocto Build mit dem Layer für Qt könnten die nötigen Abhängigkeiten erzeugt werden.

Audiodateien können über die Soundkarte im Linux System abgespielt werden. Wobei Sample Raten, die ein Vielfaches von 44.1kHz betragen unterstützen werden. Die maximale Sample Rate beträgt bei der aktuellen Konfiguration 176.4kHz. Die Vielfachen von 48kHz können nur durch Konfiguration des Clock Treibers verwendet werden. Dazu muss der Kernel neu kompiliert und das CoM neu geflasht werden.

Das Power Management Skript ermöglicht das kontrollierte Ein- und Ausschalten der GPIOs für die Steuerung der Spannungsversorgungen. So können die Komponenten des Audio-Streamers kontrolliert aufgestartet werden. Zusätzlich in diesen Prozess integriert sind die Audio Treiber,

welche zum richtigen Zeitpunkt geladen werden. Das Skript steuert auch die Mute und Reset Signale für die DACs. Allerdings hat das Ansteuern des Reset Signals noch keine Wirkung, da ein Problem mit dem GPIO vorliegt. Weiter können die Power Good Signale nicht gelesen werden. Auch hier wird ein Problem mit der Konfiguration der GPIOs vermutet. Eine Anpassung im Device-Tree würde das Problem vermutlich beheben.

### 7.1.3 Software

Auf der Softwareseite wurde der Mopidy Musikserver installiert und konfiguriert. Als Streaming-Quelle wurde Qobuz, Spotify, Internet Radio und lokale flac-Dateien verwendet. Zur Steuerung des Audio-Streamers über das Display wurde das auf Qt basierende Streamer-UI entwickelt. Das Programm kann Titel, Interpret, Fortschrittsbalken und Cover des aktuellen Musikstücks anzeigen. Die Wiedergabe kann pausiert und gestartet werden. Ein Hauptmenü erlaubt die Navigation zwischen den verschiedenen Funktionen des Streamer-UI.

Bis zum Abschluss der Arbeit wurde die Anzeige der Informationen zum aktuellen Musikstück implementiert. Das Suchen und Wählen von Stücken und Playlists ist noch nicht implementiert. Das erstellte Softwaredesign sieht die Funktionen bereits vor. Über den Mopidy internen Webserver kann parallel zum Streamer-UI die Wiedergabe aus dem lokalen Netzwerk gesteuert werden. Dadurch konnten die im Streamer-UI fehlenden Funktionen ersetzt werden.

Das Softwaredesign ist stark auf die Verwendung der libmpdclient gestützt. Diese wird pollend verwendet. Es hat sich herausgestellt, dass die Event basierte Verbindung zu Mopidy mit JSON-Objekte über einen WebSocket einfacher zu implementieren ist. Unter anderem weil die Mopidy API besser dokumentiert ist als die libmpdclient. Das Umstellen von einer pollenden zu einer Event gesteuerten Applikation würde jedoch eine Anpassung des Software-Designs nach sich ziehen.

## 7.2 Pendenzen

### 7.2.1 Hardware

Bei den Leiterplatten sind die Massnahmen aus der Tabelle 3.1 in der nächste Schema Revision umzusetzen.

Zur Kosteneinsparung wird das Weglassen des Encoder Moduls empfohlen, da dessen Funktion bereits mit dem Touchscreen abgedeckt wird. Als weitere Einsparung könnte der Einsatz eines Raspberry Pis abgeklärt werden. Dies würde aber aufgrund der wenigen Schnittstellen zu einer deutlich Reduktion des Funktionsumfangs führen.

### 7.2.2 Linux System

Im Bereich Linux ergeben sich noch folgende Pendenzen, welche im Rahmen dieser Arbeit nicht erledigt werden konnten:

- Qt mit EGLFS ergänzen (Verwendung der GPU)
- Power Good Signale im Power Management Skript erfassen
- Ansteuerung von Mute GPIO der DACs
- Drehgeber testen

Aufgrund von Problemen mit Abhängigkeiten konnte die Verwendung von Qt EGLFS nicht realisiert werden. Die Anzeige erfolgt im Moment über den Framebuffer. Damit bleibt die Grafikbeschleunigung ungenutzt. Die Probleme kommen vor allem daher, dass der iMX6 Hersteller NXP das Yocto Projekt verwendet um das Filesystem zu generieren. Bei Yocto sind die Libraries für Qt in einem Metalayer enthalten. Eine mögliche Lösung ist die Durchführung eines Yocto Builds mit den entsprechenden Layern für Qt. Anschliessend kann versucht werden die fehlenden Abhängigkeiten ins Debian Filesystem zu integrieren. Auf diesen Vorgang wurde aus zeitlichen Gründen verzichtet.

Mit dem Power Management Skript können die Power Good Signale noch nicht gelesen werden. Die Pegel der Eingänge werden im System immer als Low angezeigt. Dadurch kann nicht überprüft werden, ob eine Speisung stabil läuft. Damit sich die Speisungen stabilisieren können, sind Verzögerungen zwischen den Einschaltvorgängen eingebaut. Es wird ein Fehler in der GPIO-Konfiguration im Device-Tree vermutet, welcher bisher nicht lokalisiert werden konnte.

Die Ansteuerung des Reset Pins des DAC vom CoM aus funktioniert bisher noch nicht. Merkwürdig ist die Tatsache, dass nur einer der Outputs nicht angesteuert werden kann. Es wird vermutet das dieses GPIO durch einen Treiber verwendet wird, ohne im Device-Tree zu erscheinen.

Der Treiber für den Drehgeber konnte erfolgreich geladen werden. Allerdings ist ein Test noch ausstehend, da noch keine Unterstützung für den Drehgeber im Streamer-UI implementiert ist. Es wurde kein Kommandozeilen Programm gefunden, welches die Funktion des Drehgebers verifizieren kann.

### 7.2.3 Software Streamer-UI

Bei der Software Streamer-UI sind folgende Pendenzen offen:

- Suchfunktion
- Quellenauswahl
- Konfiguration der DAC internen oversampling Filter

Das Streamer-UI soll mit weiteren Funktionen ausgestattet werden. Das Ziel ist, dass der Audio-Streamer ohne Smartphone oder PC über das Display gesteuert werden kann. Das Streamer-UI soll zusätzlich in der Lage sein, die internen Filter des DACs zu konfigurieren. Dadurch kann der Einfluss auf Frequenzgang und Phase der digitalen oversampling Filter überprüft werden.

Es wird eine Vereinfachung und Umstrukturierung des Software-Designs empfohlen. Die libmpd-client soll mit dem Mopidy API ersetzt und das Factory Pattern entfernt werden. Dadurch kann der Backend Connector nicht mehr zu Laufzeit gewechselt werden, jedoch wird die Struktur einfacher zu verstehen.

## 7.3 Ausblick

Das Projekt wird nach Abschluss der Thesis in ein Open Source Projekt überführt. D.h. es werden alle Hardware Designs, das Linux System und der Source Code des Streamer-UI veröffentlicht. Die Erkenntnisse des Projekts können so auch von weiteren High-End Audio und Elektronik Interessierten genutzt werden.

Eine weitere Revision der Leiterplatten ist für den Herbst 2019 geplant. Bei der Anpassung ist zudem eine Modularisierung des Mainboards vorgesehen. So wird der CoM und DAC Teil auf zwei Leiterplatten aufgeteilt und über Stecker verbunden. Dies erlaubt später das Austauschen des CoM Teils.

## 7.4 Persönliche Reflexion

### Rafael Klossner

Das von uns geplante Vorgehen und die Arbeitsaufteilung hat sich meiner Meinung nach bewährt. Die Aufteilung zwischen Hardware und Software hat sich bis auf die letzten zwei Wochen und bis auf wenige Ausnahmen durchgezogen. Dadurch war es möglich sich in einem Bereich zu vertiefen. Im Austausch konnten so die Synergien optimal genutzt werden.

Das Vorgehen bei den Entwicklungen rund um das Betriebssystem Linux hat sich bewährt. Allerdings war es oft schwierig die erforderliche Zeit für einen Entwicklungsschritt abzuschätzen. Dies weil einerseits Wissen in spezifischen Gebieten fehlte und andererseits nicht von Beginn weg klar war, welche Anpassungen am bestehenden System nötig waren. Als nützliche Strategie hat sich das Überlappen von Arbeiten erwiesen. Damit ist es möglich einen Einblick in den nächsten Arbeitsschritt zu gewinnen. Oft zeigte sich dann, dass geplante Abhängigkeiten in der

Realität gar nicht existieren. Zusätzlich wird es möglich beim Zugrückwechseln zur vorherigen Arbeit das Problem aus einer anderen Perspektive zu betrachten. Was oft zu einer Lösung führt. Zur Betrachtung eines Problems aus einer anderen Perspektive war auch oft das Gespräch im Team sehr hilfreich.

Ich erhielt während meiner Arbeit einen tiefen Einblick in das Betriebssystem Linux. Dabei hat mich die vielseitige Einsetzbarkeit des Systems beeindruckt. Es gab aber auch immer wieder Rückschläge. Da es sich bei Linux um ein Open Source Projekt handelt, ist der Code Stil oft sehr gewöhnungsbedürftig. Insbesondere, dass es Treiber gibt mit über 1000 Zeilen Code ohne einen einzigen Kommentar, hat mich doch sehr erstaunt. Oft war auch die Dokumentation zu einem ganzen System nur sehr dürftig. Das macht die Einarbeitung schwierig bis beinahe unmöglich. Daher habe ich während der ganzen Arbeit viele Stunden mit der Erarbeitung von neuem Wissen verbracht. So ist es mir insbesondere bei der Anpassung des Audio Treibers ergangen. Am Ende war die Anpassung im Vergleich zum getriebenen Aufwand sehr klein. Wenn man aber darüber nachdenkt, was nötig wäre um die Ausgabe von digitalen Audiosignalen mit DMA ohne ein Betriebssystem zu realisieren, ist das Endresultat wiederum sehr beeindruckend.

Obwohl das Projekt erfolgreich verlaufen ist, gibt es doch einige Punkte, die ich in einem nächsten Projekt anders angehen würde. Insbesondere bei der Anpassung des Audio-Treibers, wäre das realisieren einzelner kleiner Funktionalitäten einfacher gewesen, als von Anfang an das grosse ganze zu betrachten. Zwar ist ein grober Überblick hilfreich, allerdings ist es oft nicht möglich einen Überblick über ein ganzes System zu haben. Daher würde ich mich in einem zukünftigen Projekt eher auf einzelne Minimalfunktionalitäten fokussieren und diese Stück für Stück zusammensetzen.

Trotz allen Rückschlägen und Schwierigkeiten war es eine Bereicherung an diesem Projekt zu Arbeiten. Ich bin der Meinung das Ergebnis ist gut gelungen, dies hat auch der Besuch bei Dynavox gezeigt. Das Ergebnis ist eine solide Basis, auf der in Zukunft aufgebaut werden kann. Hinsichtlich der Weiterentwicklung als Open Source Projekt vermute ich ein grosses Potential in diesem Gerät.

### **Stefan Lüthi**

Die Inbetriebnahme der Elektronik war wie erwartet eine Herausforderung. In der komplexen Schaltung waren etliche kleine, wie auch einige wenige beeinträchtigende Fehler aufgetreten. Mit einem ausführlicheren Schema-Review in der Design-Phase, hätte man viele dieser Fehler bereits finden können. Leider verhielt sich der Analog Filter nicht wie in der Simulation und begann zu Schwingen. Mit einer Anpassung der Simulation, welche ein instabiles Verhalten provoziert, hätte man dieses Verhalten vielleicht schon früher finden können. Insgesamt war ich jedoch erfreut, dass bis zum Abschluss der Arbeit fast die komplette Elektronik (insbesondere die High-Speed Peripherie über PCIe und SATA) funktionierte.

Beim Gehäuse konnte ich glücklicherweise das Front Panel bei der Maschinenbau Abteilung fräsen lassen. Dies war nur möglich, weil die Bearbeitung frühzeitig abgeklärt wurde. Die Ausschnitte im Back Panel, welche ich selbst ausgefräst habe, können nicht mit der makellosen Qualität des Front Panels mithalten.

Die Entwicklung der Software setzte die Einarbeitung in die QtQML Sprache voraus. Der Aufwand hatte sich jedoch ausgezahlt, da mit QML schöne Benutzeroberflächen gestalten lassen. Durch die Anbindung an C++ Code kann nicht nur eine schöne, sondern auch eine effiziente Applikation implementiert werden. Beim Erstellen des Software Designs wurde zu stark auf die libmpdclient Bibliothek eingegangen. Die Anbindung an den Musikserver hätte sich aber mit dem Mopidy API eleganter lösen lassen.

Trotz der guten Messwerte bei den Audio-Messungen war ich vor dem Hörtest bei Dynavox durchaus angespannt. Was wenn die Klangqualität nicht zu überzeugen vermag? Die positive Rückmeldung und den Einblick die Welt des High-End Audio hat mich aber sehr erfreut. Die Vorschläge für die Anpassung der Schaltung werden in der nächsten Revision evaluiert und umgesetzt.

Insgesamt bin ich von der Elektronik, der Software und der Klangqualität des Audio-Streamers begeistert. Dies war ein sehr vielfältiges Projekt, welches viele verschiedene Arbeiten mit sich brachte. Ich freue mich auf die Weiterentwicklung des Audio-Streamers.

# 8 Projektmanagement

Diese Kapitel beinhaltet die Projektplanung, sowie deren Auswertung. Abweichungen werden begründet und Verbesserungspunkte werden aufgelistet.

## 8.1 Projektphasen

### Konzept & Evaluation

Mit dem Konzept und der Evaluation wird der Grundstein für das Projekt gelegt. Diese Phase wurde bereits in der Projektstudie erledigt. So wurden die Konzepte für die Schaltung und die Benutzeroberfläche definiert und Komponenten evaluiert.

### Realisierung

In der Realisierung werden die Design der einzelnen Komponenten des Projekts erstellt und anschliessend umgesetzt. Ein Teil dieser Phase wurde in der Projektstudie erledigt. Es wurde das Schaltungs- und Leiterplattendesign abgeschlossen. Das Gehäusedesign wurde begonnen. Das Linux System wurde vorbereitet und Treiber evaluiert.

Während der Thesis wird im Bereich Hardware die Hardware gefertigt, d.h. die Leiterplatten werden bestückt und das Gehäuse wird bearbeitet. Im Bereich Linux werden Treiber auf den Audio-Streamer angepasst.

Für die Software muss ein Design erstellt und die Software anschliessend implementiert werden.

### Verifikation & Tests

In der Testphase wird die Funktionalität der einzelnen Komponenten und des kompletten Systems überprüft. Diese Tests sind vom Linux System und der Hardware abhängig.

### Projektabchluss

Für den Abschluss des Projekts muss das weitere Vorgehen, sowie die Übergabe aller Daten sichergestellt werden.

## 8.2 Zeitplan

### 8.2.1 Übersicht

Die Arbeiten können in die Gruppen Hardware, Linux System, Messungen, Software und Dokumentation aufgeteilt werden. Dabei ist jeweils eine Person für die Umsetzung einer Gruppe verantwortlich. Die Arbeiten der Gruppen Hardware und Linux sind voneinander unabhängig und wurden daher auf zwei Personen aufgeteilt, um möglichst parallel arbeiten zu können. Die Entwicklung der Bedienersoftware wird danach zu zweit durchgeführt.

Meilensteine wurden dort platziert, wo ein definiertes Ziel erreicht werden muss, um die weiteren Arbeiten auszuführen zu können. Arbeiten, wie Messungen, die keine Auswirkung auf andere Punkte haben, wurden flexibel geplant um keine künstlichen Abhängigkeiten zu schaffen.

Der Soll und Ist Zeitplan sind auf der folgenden Doppelseite abgebildet.

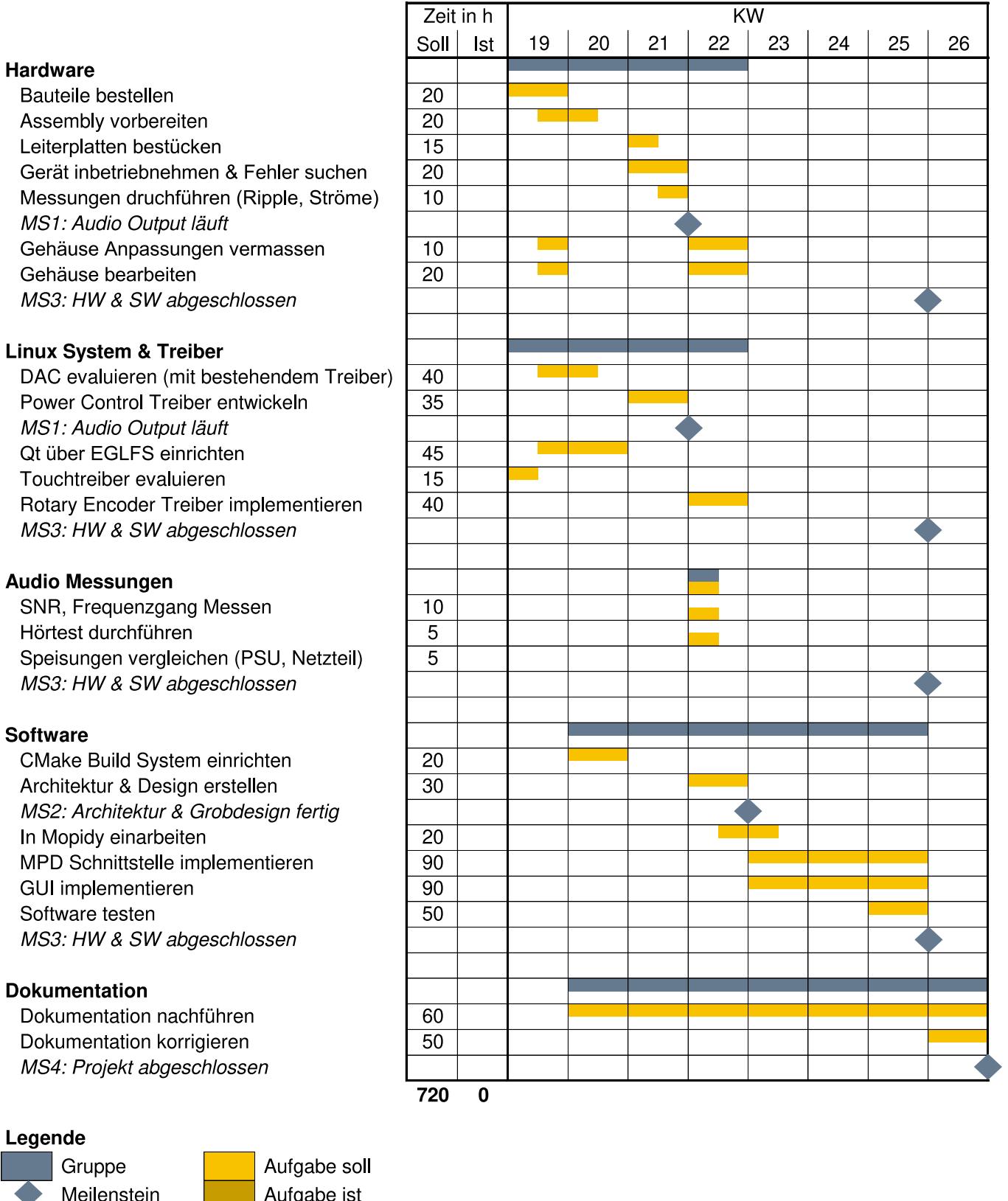
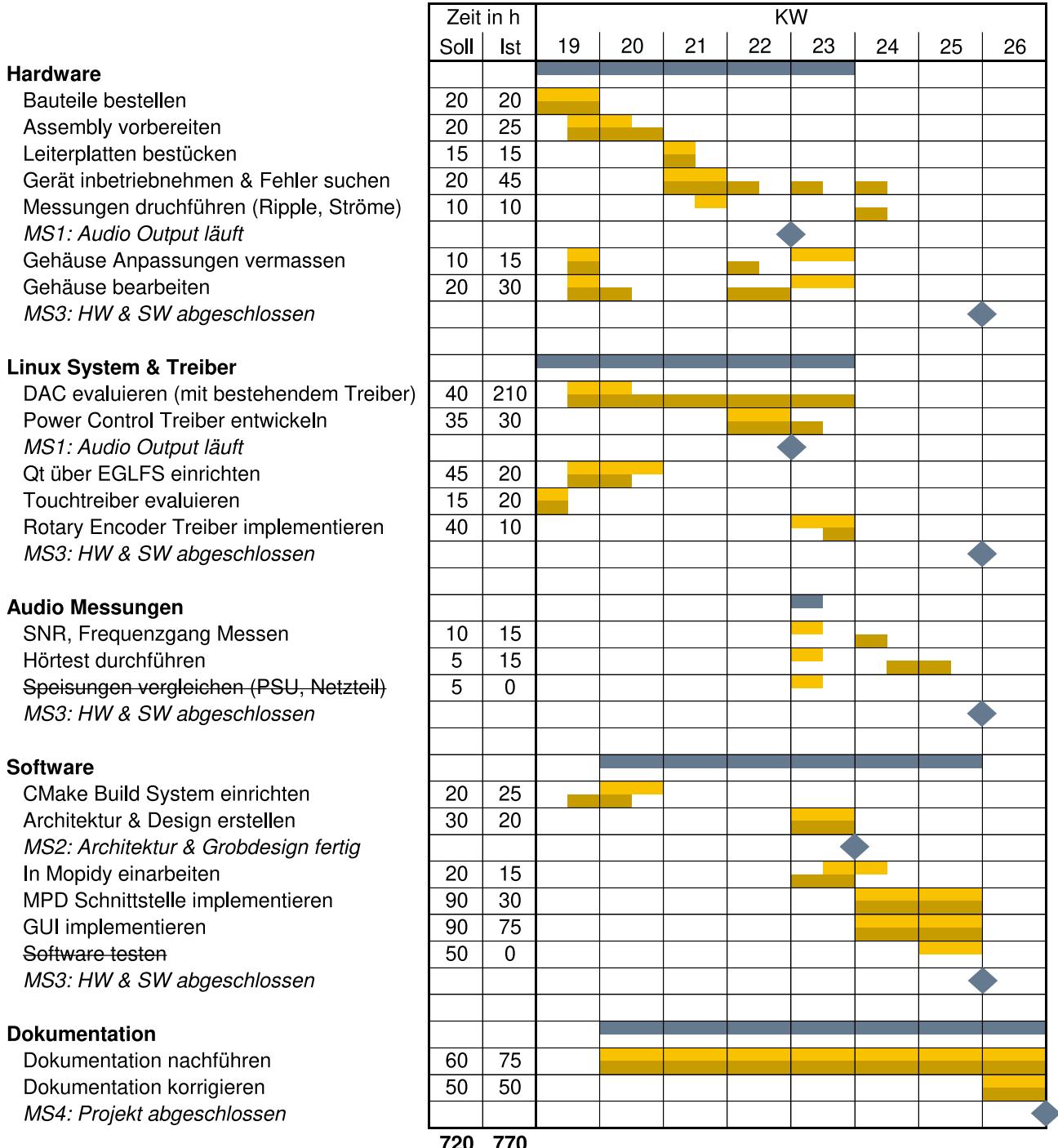


Abbildung 8.1: Projektplan V1.0 Soll



#### Legende

- Gruppe
- Aufgabe soll
- Aufgabe ist
- abe Aufgabe gestrichen
- Meilenstein

Abbildung 8.2: Projektplan V1.1 Ist

## 8.2.2 Meilensteine

Die Ziele der Meilensteine und deren Auswertung auf dem Zeitplan sind nachfolgend aufgelistet.<

---

### MS1: Audio Output läuft

---

**Soll 24.05.2019 Ziele:**

1. DAC wird initialisiert
  2. CoM gibt Audio Stream auf I2S-Schnittstelle
  3. Funktion qualitativ überprüft
- 

**Ist 31.05.2019 Erledigt:**

- Konfiguration der AK4493 im Mono Mode läuft
- MCLK von 22.5760 MHz vom CoM läuft
- I2S Stream läuft
- Audio Signale auf Hardware validiert

**Begründung Abweichung:**

Der Termin konnte nicht gehalten werden, da die Implementation des Audio-Treibers und die Konfiguration des korrekten MCLK deutlich länger als geplant dauerten. Aufgrund einiger Fehler auf der Hardware dauerte die Inbetriebnahme länger als geplant.

**Auswirkungen:**

Aufgrund des Rückstands wurden für das Erreichen des Milestones eine zusätzliche Woche investiert. In der Planung wurde dazu die KW24 entfernt und die KW22 und KW24 um eine Woche nach hinten geschoben. Diese Zeit wird bei der Entwicklung der Bedienersoftware eingespart. Einerseits da die Implementation der MPD Schnittstellen nach ersten Versuchen einfacher wird als geplant, andererseits wird der Funktionsumfang der Software eingeschränkt.

---

### MS2: Architektur & Grobdesign fertig

---

**Soll 07.06.2019 Ziele:**

1. Architektur und Schnittstellen definiert
  2. Grobdesign erstellt
  3. Mopidy Interface bekannt
- 

**Ist 07.06.2019 Erledigt:**

- 1. Architektur mit Mopidy definiert
- 2. Wartbarkeit durch MVC sichergestellt
- 3. Erstes Klassendiagramm und Use Cases definiert

**Abweichung:**

keine

**Auswirkungen:**

keine, weiter nach Plan

---

Tabelle 8.1: Meilensteine

---

### **MS3: HW & SW abgeschlossen**

---

**Soll 21.06.2019 Ziele:**

1. Hardware kann Audio abspielen
  2. Software kann Hardware ansteuern
  3. Audio-Streamer kann auf Display bedient werden
- 

**Ist 21.06.2019 Erledigt:**

1. Hard- & Software läuft grösstenteils
2. Klangqualität wurde mit Messungen und Hörtests verifiziert
3. Pendenzen sind in Dokumentation festgehalten

**Abweichung:**

keine

**Auswirkungen:**

keine, weiter nach Plan

---

Tabelle 8.2: Meilensteine (Fortsetzung)

### **8.2.3 Auswertung**

Mit Ausnahme der Entwicklung des Audio Treibers konnten die geplanten Arbeiten zeitgerecht erledigt werden. Eine grosse Abweichung von der Soll-Planung gab es bei der Implementation des Audio Treibers und der Konfiguration des Master Clocks, da beim bestehenden Treiber mehr Anpassungen gemacht werden mussten als erwartet und die Einarbeitungszeit in das Linux Treiber System gross ist. Bei der Hardware nahm die Inbetriebnahme aufgrund der komplexen Schaltung mehr Zeit in Anspruch als geplant.

Dies hatte eine Einschränkung des Funktionsumfangs der Bedienersoftware zur Folge. Dies ist nicht weiter tragisch, da das Hauptziel des Projekts auf hochqualitativen Audioausgabe liegt. Durch das ausführliche Software-Design ist die einfache Weiterentwicklung sichergestellt.

Bei den Audio Messungen wurde auf einen Vergleich mit verschiedener Speisungen verzichtet, da die gewünschten Werte bereits mit der PSU erreicht wurden.

Bei den Softwaretests waren nebstd den qualitativen Tests durch den Bediener auch automatisierte Unit-Tests vorgesehen. Da QML und C++ in Kombination verwendet werden, kann nicht gtest von Google verwendet werden. Dies hätte das Einarbeiten in ein anderes Test-Framework zur Folge. Aus zeitlichen Gründen wurde die Unit-Tests daher ausgelassen.

## **8.3 Dokumente**

Anhand der Dokumente sollen Entscheidungen nachvollziehbar, Ergebnisse reproduzierbar werden und eine leichte Einarbeitung in das Projekt erlauben. Dazu sind folgende Dokumente zu erstellen:

- Dokumentation
- Code Dokumentation
- Arbeitsjournal
- Sitzungsprotokolle

Die Dokumentation enthält die Erläuterungen der umgesetzten Hard- und Software. Dies erleichtert den Einstieg in das Projekt und stellt die Reproduzierbarkeit der Ergebnisse sicher.

Die Code Dokumentation wird mit Doxygen umgesetzt. Den Gebrauch, die Abhängigkeiten, den Buildvorgang und die verwendeten Code Richtlinien sind in der Readme-Datei zu beschreiben.

Im Arbeitsjournal sollen Überlegungen und Lösungen auf Probleme festgehalten werden. Es wird dazu ein handschriftliches Journal für Skizzen und weitere Notizen verwendet. Kommandozeilenbefehle und Konfigurationen werden digital in Boostnote notiert. Dadurch kann die copy-paste Funktion genutzt werden.

Sitzungsprotokolle werden zwingend bei Beschlüssen und Meilensteinen erstellt. Ein Meilenstein-protokoll muss die Auswertung der Ziele und bei Abweichungen eine Begründung, sowie einen Korrekturvorschlag enthalten. Protokolle werden als PDF per E-Mail an alle Teilnehmenden ver-sendet. Einsprüche können bis eine Woche nach dem Erhalt des Protokolls erhoben werden.

# **Selbständigkeitserklärung**

Ich bestätige, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der im Literaturverzeichnis angegebenen Quellen und Hilfsmittel angefertigt habe. Sämtliche Textstellen, die nicht von mir stammen, sind als Zitate gekennzeichnet und mit dem genauen Hinweis auf ihre Herkunft versehen.

Burgdorf, 27.06.2019

Burgdorf, 27.06.2019

Rafael Klossner

Stefan Lüthi



# Abkürzungsverzeichnis

- AC** Alternating Current.
- AGND** Analog Ground.
- ALSA** Advanced Linux Sound Architecture.
- API** Application Programming Interface.
- ASoC** Audio for System on Chip.
- AUDMUX** Audio Multiplexer (iMX6).
- CAD** Computer Aided Design.
- CD** Compact Disc.
- CHGND** Chassis Ground.
- CoM** Computer on a Module.
- CPU** Central Processing Unit.
- DAC** Digital Analog Converter.
- DAI** Digital Audio Interface.
- DAPM** Dynamic Audio Power Management.
- DC** Direct Current.
- DGND** Digital Ground.
- DMA** Direct Memory Access.
- EMV** Elektromagnetische Verträglichkeit.
- FFT** Fast Fourier Transform.
- GCD** Greatest Common Divisor.
- GDB** GNU Debugger.
- GND** Ground.
- GPIO** General Purpose Input/Output.
- GPU** Graphics Processing Unit.
- GUI** Graphical User Interface.
- HW** Hardware.
- I2C** Inter Integrated Circuit.
- I2S** Inter-IC Sound.
- IC** Integrated Circuit.
- IOMUX** Input/Output Multiplexer (iMX6).
- LCD** Liquid Crystal Display.
- LCM** Least Common Multiple.
- LED** Light Emitting Diode.
- MCLK** Master Clock.

**MMU** Memory Management Unit.

**MPD** Music Player Deamon.

**MVC** Model View Controller.

**PCB** Printed Circuit Board.

**PCI** Peripheral Component Interconnect Express.

**PCM** Puls Code Modulation.

**PG** Power Good.

**PLL** Phase Lock Loop.

**PSU** Power Supply Unit.

**PWM** Pulsweitenmodulation.

**QML** Qt Modeling Language.

**RAM** Random Access Memory.

**RMS** Root Mean Square.

**SNR** Signal to Noise Ratio.

**SoC** System-on-a-Chip.

**SSI** Synchronous Serial Interface.

**SW** Software.

**TCP/IP** Transmission Control Protocol/Internet Protocol.

**THD** Total Harmonic Distortion.

**UI** User Interface.

**UKW** Ultrakurzwellen.

**USB** Universal Serial Bus.

# Literaturverzeichnis

- [1] R. Klossner und S. Lüthi, „High-End Audio-Streamer - Komponenten-Evaluation und Hardware-Design eines Audio-Streamers als Ersatz für CD-Player und FM-Radio in HiFi Systemen“, Projektstudie, Berner Fachhochschule, 3. Mai 2019.
- [2] Toradex, *Layout Design Guide*, Rev. 1.0, 14. Apr. 2015.
- [3] Asahi Kasei Microdevices, *Quality Oriented 32-Bit 2ch DAC*, AK4493 Datenblatt, Rev. 00, Dez. 2017.
- [4] Analog Devices, *High Performance Multibit Sigma-DAC with SACD Playback*, AD1955 Datenblatt, Rev. 0.
- [5] Cirrus Logic, *24-bit 192kHz DAC with Advanced Digital Filtering*, WM8741 Datenblatt, Rev. 4.3, Feb. 2013.
- [6] Texas Instruments, *24-Bit, 192-kHz Sampling, Advanced Segment, Adio Stereo Digital-to-Analog Converter*, PCM1792 Datenblatt, Nov. 2006.
- [7] J. C. Greg Kroah-Hartman Alessandro Rubini, *Linux Device Drivers, 3rd Edition*. O'Reilly Media, 2005, Chapter 4. Debugging Techniques.
- [8] T. kernel development community, *Platform Devices and Drivers*, <https://www.kernel.org/doc/Documentation/driver-model/platform.txt>, 2019. (besucht am 10.06.2019).
- [9] G. Likely, *Platform Devices and Drivers*, <https://www.kernel.org/doc/Documentation/devicetree/usage-model.txt>, 2019. (besucht am 19.06.2019).
- [10] P. Mochel, *sysfs - The filesystem for exporting kernel objects*, <https://www.kernel.org/doc/Documentation/filesystems/sysfs.txt>, 2019. (besucht am 21.06.2019).
- [11] Toradex, *Device Tree Customization*, <https://developer.toradex.com/device-tree-customization>, Rev. 29, 2018. (besucht am 19.06.2019).
- [12] mini-box.com, *Linux LCD timing handling and interfacing*, [http://arm.mini-box.com/index.php?title=Linux\\_LCD\\_timing\\_handling\\_and\\_interfacing](http://arm.mini-box.com/index.php?title=Linux_LCD_timing_handling_and_interfacing), 30. Nov. 2011. (besucht am 19.06.2019).
- [13] Newhaven Display, *NHD-5.0-800480TF-ATXL-CTP*, NHD-5.0 Datenblatt, Rev. 15, Okt. 2018.
- [14] Toradex, *Display Output, Resolution and Timings (Linux)*, <https://developer.toradex.com/knowledge-base/display-output-resolution-and-timings-linux>, Rev. 34, 2018. (besucht am 10.06.2019).
- [15] ——, *Backlight (Linux)*, [https://developer.toradex.com/knowledge-base/backlight-\(linux\)](https://developer.toradex.com/knowledge-base/backlight-(linux)), Rev. 20, 2017. (besucht am 10.06.2019).
- [16] T. kernel development community, *Advanced Linux Sound Architecture (ALSA) project homepage*, [https://www.alsa-project.org/wiki/Main\\_Page](https://www.alsa-project.org/wiki/Main_Page), 2019. (besucht am 07.06.2019).
- [17] ——, *ALSA SoC Layer*, <https://www.kernel.org/doc/html/v4.14/sound/soc/index.html>, 2019. (besucht am 31.05.2019).
- [18] Toradex, *Apalis iMX6*, Apalis iMX6 Datenblatt, Rev. 1.4, Okt. 2018.
- [19] NXP, *i.MX 6Dual/6Quad Applications Processor Reference Manual*, Apalis iMX6 Reference Manual, Rev. 5, Juni 2018.
- [20] J. Barnsley, *Iris*, <https://github.com/jaedb/Iris>, Commit b314fd8. (besucht am 15.06.2019).
- [21] JRyannel und JThelin, *Qt5 Cadaques*. GitHub, 14. Jan. 2019.
- [22] Qt, *Models and Views in Qt Quick*, <https://doc.qt.io/qt-5/qtquick-modelviewsdata-modelview.html>, 2019. (besucht am 07.06.2019).
- [23] E. Gamma, R. Helm, R. Johnson und J. Vlissides, *Design Patterns, Elements of Resusable Object-Oriented Software*. Addison-Wesley, 1995.

- [24] taschenb, *Mopidy-Qobuz*, <https://github.com/taschenb/mopidy-qobuz>, Commit 26abcf4. (besucht am 18. 06. 2019).
- [25] mones88, *Mopidy-Tidal*, <https://github.com/mones88/mopidy-tidal>, Commit 4bd1a6f. (besucht am 07. 06. 2019).
- [26] Mopidy, *Mopidy-Spotify*, <https://github.com/mopidy/mopidy-spotify>, Commit 77a2930. (besucht am 07. 06. 2019).
- [27] ——, *Mopidy-SoundCloud*, <https://github.com/mopidy/mopidy-soundcloud>, Commit 8776297. (besucht am 07. 06. 2019).
- [28] CMake, *About CMake*, <https://cmake.org/overview/>. (besucht am 21. 06. 2019).
- [29] Crystek Crystals, *C33xx Model*, C3391 Datenblatt, Rev. P, 16. Feb. 2016.
- [30] Asahi Kasei Microdevices, *AK4493 Evaluation Board*, AKD4493 Manual, Rev. 1, Juli 2017.
- [31] Dynavox Electronics SA, *Dynavox in Givisiez*, <http://www.swisshd.ch/dealers/dynavox-givisiez.aspx>. (besucht am 21. 06. 2019).
- [32] T. Reenskaug, „Mvc xerox parc 1978-79“, *Trygve/MVC*, 1979.
- [33] T. kernel development community, *The Linux Input Documentation*, <https://www.kernel.org/doc/html/v4.12/input/index.html>, 2018. (besucht am 16. 06. 2019).
- [34] Wikipedia, *Platform Devices and Drivers*, [https://de.wikipedia.org/wiki/Logistische\\_Funktion](https://de.wikipedia.org/wiki/Logistische_Funktion), 2019. (besucht am 20. 06. 2019).

# Abbildungsverzeichnis

1.1	Audio-Streamer in seinem Umfeld .....	1
2.1	Schaltungskonzept des Audio-Streamers .....	3
3.1	Layout Top und Bottom Layer .....	5
3.2	Layout VCC Layer .....	6
3.3	Layout GND Layer .....	7
3.4	Layout PSU, Top und Bottom .....	8
3.5	Layout Rotary Encoder, Top und Bottom .....	9
3.6	Audio-Filter Analog Devices AD1955 .....	14
3.7	Audio-Filter Burr Brown (Texas Instruments) PCM1792A .....	14
3.8	Audio-Filter Wolfson Microdevices (Cirrus Logic) WM8741 .....	15
3.9	Korrekturen VR203 .....	15
3.10	Renderansicht Front Panel .....	16
3.11	Renderansicht Back Panel .....	17
3.12	Renderansicht Base Plate .....	17
3.13	Verdrahtungsplan .....	18
3.14	Verdrahtung des Audio-Streamers .....	19
3.15	Montage der Bedienelemente .....	19
4.1	Übersicht über die LCD Timings .....	26
4.2	Übersicht über die IOMUX Peripherie .....	30
4.3	Übersicht über den Clock Tree von PLL4 nach SSI1 .....	32
4.4	Übersicht über die Audio Peripherie des iMX6 .....	33
4.5	Übersicht über den SSI Clock Tree .....	34
4.6	Timing Diagramm des Startup Prozesses .....	36
5.1	Architektur des Musikservers und Front-Ends .....	39
5.2	Iris Web Front-End für Mopidy .....	40
5.3	Architektur des Streamer-UI .....	40
5.4	Use-Case Diagramm des Streamer-UI .....	42
5.5	Klassendiagramm des Streamer-UI .....	43
5.6	Streamer-UI Hauptmenu .....	46
5.7	Streamer-UI Informationen zum aktuellen Titel .....	46
6.1	Messaufbau Spannungsversorgungen .....	47
6.2	Messaufbau Signalqualität .....	48
6.3	Messaufbau Audio .....	48
6.4	Aufbau Funktionstests .....	49
6.5	Vergleich Audio Takt Signale .....	51
6.6	Frequenzgang der Audioausgänge .....	52
6.7	Frequenzspektren der single-ended Ausgänge (RCA) .....	53
6.8	Frequenzspektren der differenziellen Ausgänge (XLR) .....	54
6.9	Testanlage der Dynavox Electronics SA .....	57
8.1	Projektplan V1.0 Soll .....	64
8.2	Projektplan V1.1 Ist .....	65



# Tabellenverzeichnis

3.1	Korrekturen der Leiterplatten der Version 1.0 und empfohlene Anpassungen für die nächste Iteration .....	13
3.2	Kostenübersicht Prototyp (1 Stk.).....	20
3.3	Materialkosten inkl. Encoder Option bei unterschiedlichen Produktionsmengen .....	20
4.1	Mögliche Abspielraten für den Audio-Streamer.....	31
4.2	Zusammenhang zwischen Sample Rate und Bit Clock.....	35
4.3	Übersicht über die Power-Management Pins .....	37
6.1	Verwendetes Material für die Messungen und Tests .....	47
6.2	Einstellungen des Oszilloskops zur Messungen der Spannungsripple .....	48
6.3	Spannungen und Ströme der PSU .....	49
6.4	Analoge Speisungen (Mainboard) .....	50
6.5	Spannungsripple der Spannungsversorgungen .....	50
6.6	Zusammenfassung MCLK Jitter .....	51
6.7	Zusammenfassung SNR und THD+N (alle Harmonischen bis 80 kHz).....	54
6.8	SNR des single-ended Ausgangs (RCA) .....	55
6.9	SNR des differenziellen Ausgangs (XLR) .....	55
6.10	THD+N bei 1 kHz aller Ausgänge.....	55
6.11	Crosstalk bei 1 kHz auf dem XLR Ausgang.....	55
6.12	Zusammenfassung der Funktionstests der digitalen Schnittstellen.....	56
8.1	Meilensteine .....	66
8.2	Meilensteine (Fortsetzung) .....	67



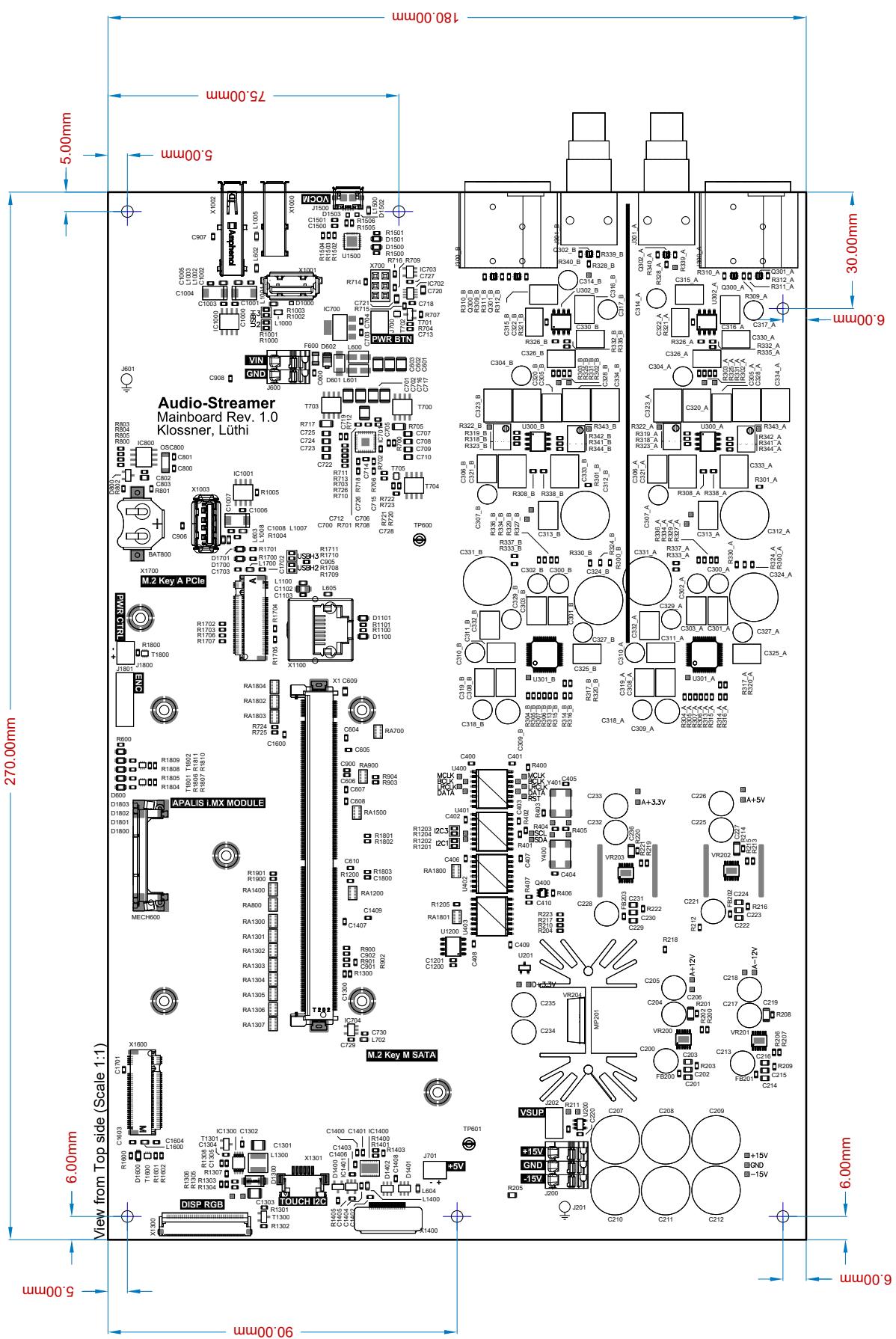
# **Versionenverzeichnis**

<b>Version</b>	<b>Datum</b>	<b>Status</b>	<b>Bemerkungen</b>
1.0	27.06.2019	Freigegeben	Erste komplette Version



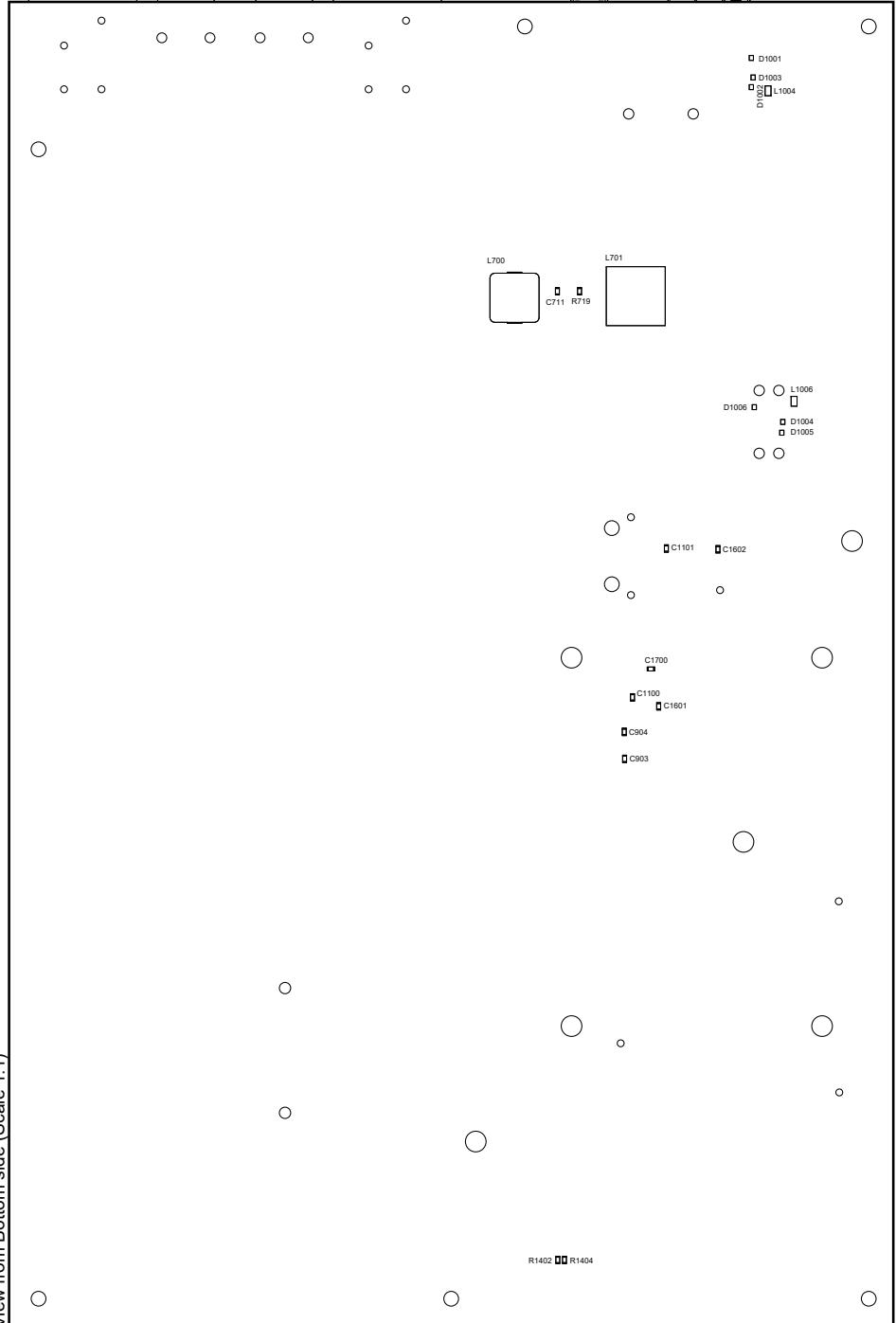
## **Anhang A**

# **Bestückungspläne**



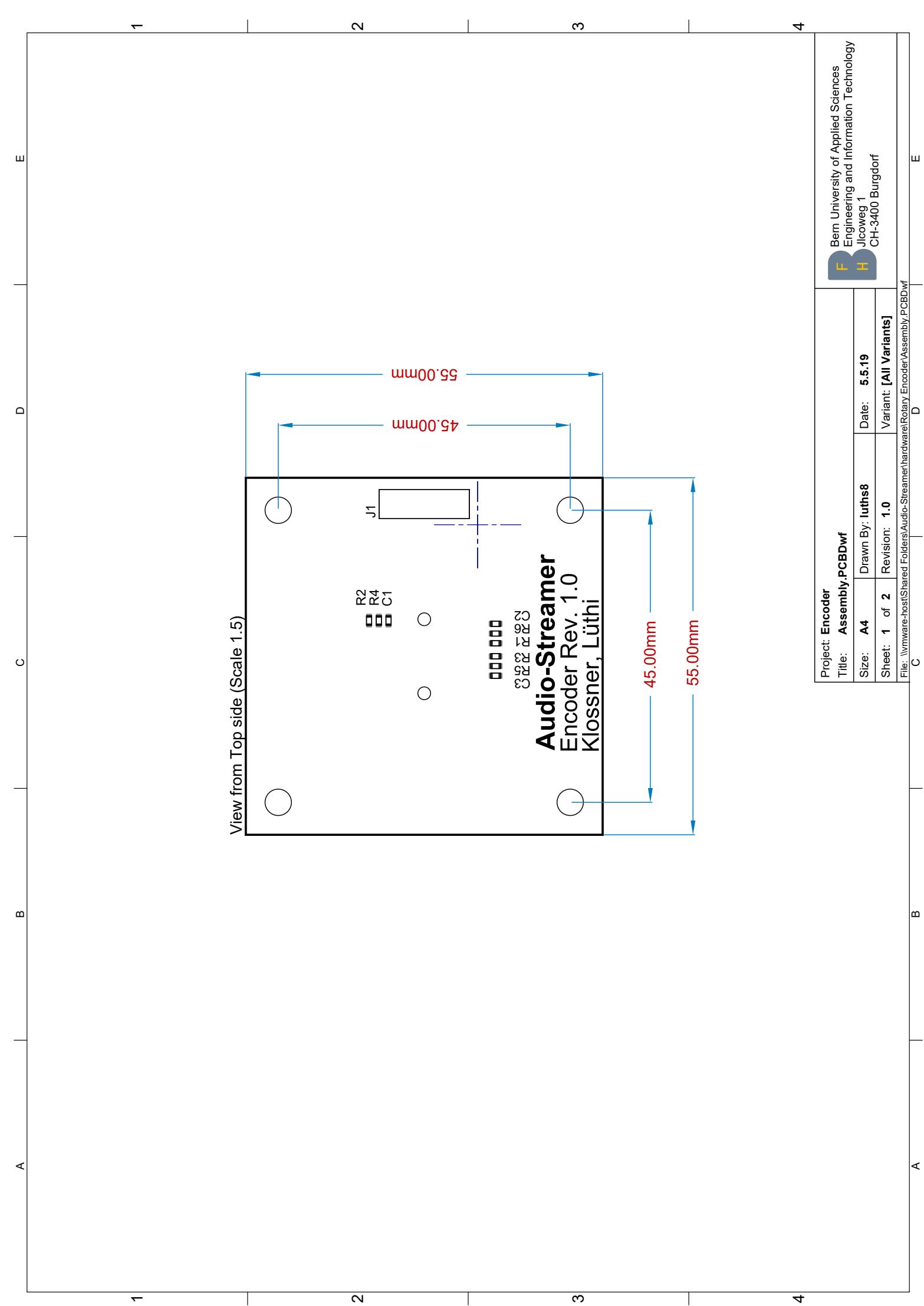
<b>Project:</b> Mainboard	<b>Title:</b> AssemblyPCB.Dwf	<b>Date:</b> 9.5.19
<b>F</b>	<b>G</b>	<b>H</b>
Bern University of Applied Sciences	Engineering and Information Technology	Jägweg 1 CH-3400 Burgdorf
<b>I</b>	<b>J</b>	<b>K</b>
File: \vmware-host\Shared Folders\Audio-Streamer\hardware\Mainboard\Draftman\Assembly\Mainboard.Dwf		

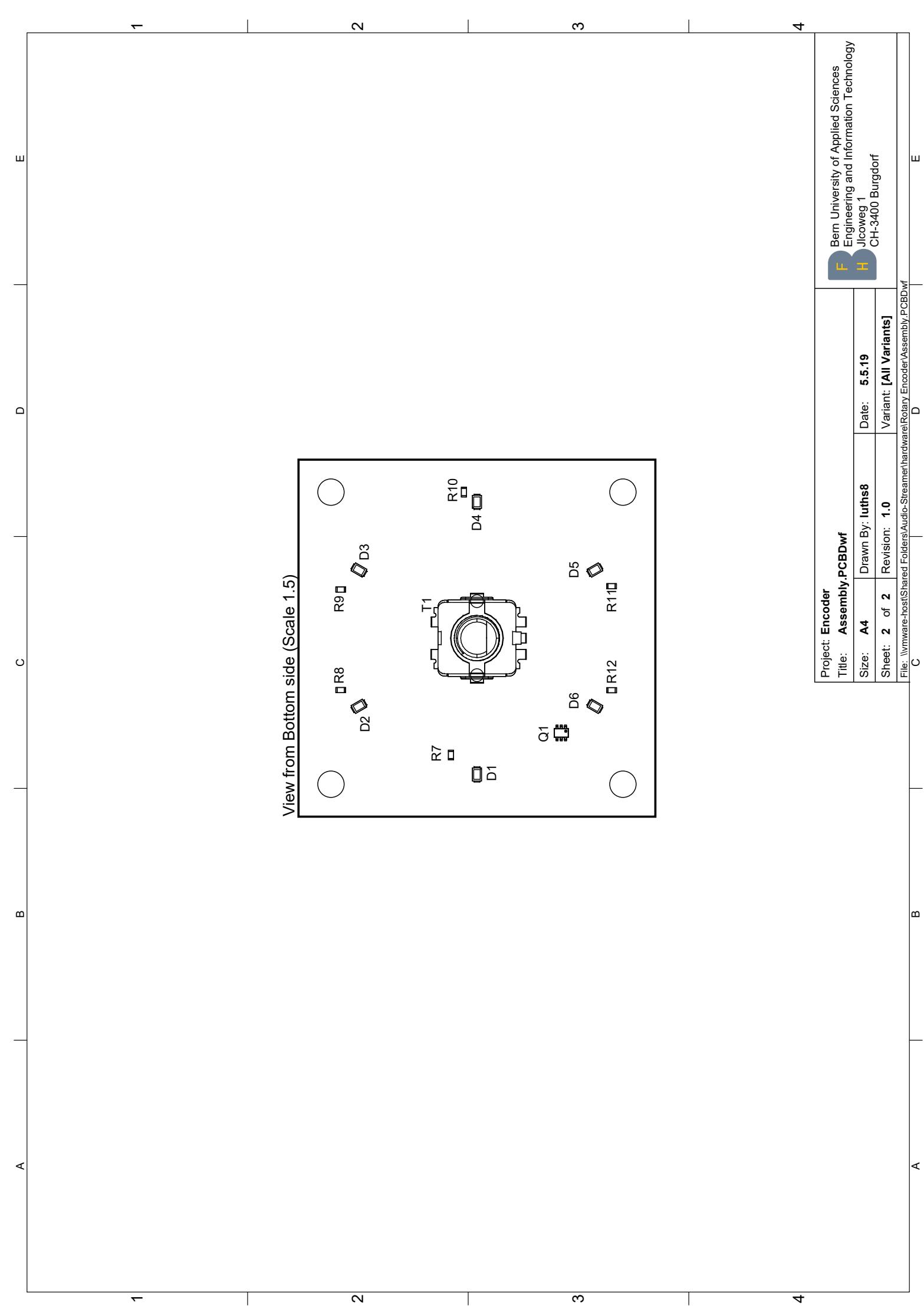
View from Bottom side (Scale 1:1)

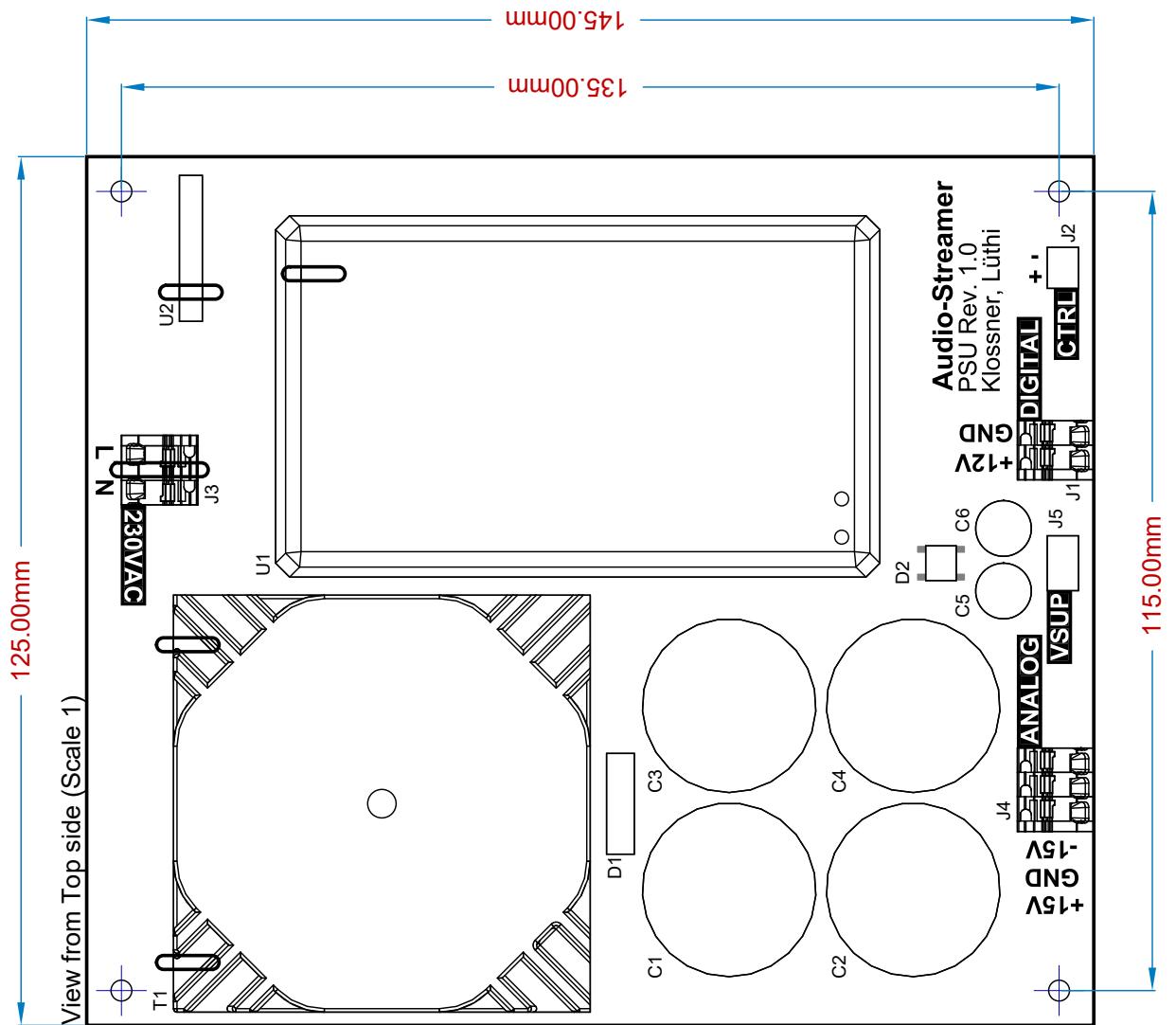


Project:	Mainboard
Title:	Assembly.PCDBwf
Size:	A3
Sheet:	2 of 2
Date:	9.5.19
Revision:	1.0
Variant:	Complete
File:	\vmware-host\Shared Folders\Audio-Streamer\hardware\Mainboard\Draftman\Assembly.PCDBwf

Bern University of Applied Sciences  
Engineering and Information Technology  
JUeweg 1  
CH-3400 Burgdorf







Project: PSU	PCB Dwg		
Title: Assembly	PCB Dwg		
Size: A4	Drawn By: luths8	Date: 5.5.19	
Sheet: 1 of 1	Revision: 1.0	Variant: Default	
File: \vmware-host\Shared Folders\Audio-Streamer\hardware\Power-Supply\Assembly.PCBBdwf			

Bern University of Applied Sciences  
Engineering and Information Technology  
Jlicoweg 1  
CH-3400 Burgdorf

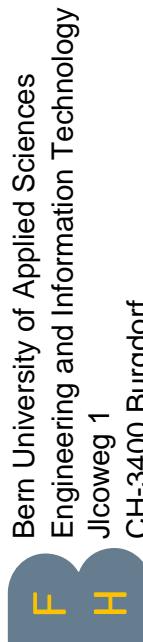
# **Anhang B**

# **Stücklisten**



# Bill of Materials

Project: Mainboard  
 Author: luths8  
 Variant: Complete  
 Revision: 1.0  
 Date: 11.05.2019



Quantity	Designator	Comment	Description	Footprint	Manufacturer Part No.	Distributor	Distributor Part No.	Price	Line Total
1	BAT800	Keystone-3000	Battery Holder	T_Keystone_3000	3000	Mouser	534-3000	CHF 0.228	CHF 0.23
C200, C204, C213, C217, C221, C225, C228, C232, 10 C234, C235	22µF	Elektrolytic Capacitor	CAP-6.3-5.5-2.5	32SEPFP22M	Mooser	667-32SEPFP22M+TSS	CHF 0.765	CHF 7.65	
4 C206, C219, C227, C236	4.7µF	Ceramic Capacitor	SMD1206CAP	GCJ31CR71E475KA12L	Digi-Key	490-5808-1-ND	CHF 0.558	CHF 2.23	
C207, C208, C210, C211, C312_A, C312_B, C324_A, 10 C324_B, C331_A, C331_B	470µF	Elektrolytic Capacitor	CAP-12.5-20-5	EEU-FR1H471	Distrelec	16723924	CHF 0.749	CHF 7.49	
C220, C404, C405, C728, C903, C904, C905, C906, C907, C908, C1100, C1101, C1408, C1409, C1601, 18 C1602, C1700, C1701	10nF	Ceramic Capacitor, Capacitor Ceramic	SMD0603CAP	GRM188R72A103JA01D	Mouser	81-GRM188R72A103JA1	CHF 0.040	CHF 0.72	
C300_A, C300_B, C302_A, C302_B, C304_A, C304_B, C307_A, C307_B, C310_A, C310_B, C314_A, C314_B, C317_A, C317_B, C318_A, C318_B, C327_A, C327_B, 20 C329_A, C329_B	10µF	Elektrolytic Capacitor	CAP-5-11.5-2	EEU-FR1H100	Mouser	667-EEU-FR1H100	CHF 0.097	CHF 1.94	
C301_A, C301_B, C303_A, C303_B, C305_A, C305_B, C306_A, C306_B, C308_A, C308_B, C311_A, C311_B, C313_A, C313_B, C315_A, C315_B, C316_A, C316_B, C319_A, C319_B, C325_A, 24 C325_B, C332_A, C332_B	100nF	Film Capacitor	CAP_WIMA_MKP	MKP2D031001F00KF00	Mouser	505-MKP2D031001F00K0	CHF 0.279	CHF 6.70	
C309_A, C309_B	1µF	Elektrolytic Capacitor	CAP-6.3-11.2-2.5	UPW1H010MDD1TD	Digi-Key	493-11354-1-ND	CHF 0.189	CHF 0.38	
C320_A, C320_B, C328_A, 4 C328_B	33nF	Film Capacitor	CAP_WIMA_FKP	FKP2C023301L00HSSD	Distrelec	16543045	CHF 0.636	CHF 2.54	
C321_A, C321_B, C333_A, 4 C333_B	22nF	Film Capacitor	CAP_WIMA_FKP	FKP2D022201L00HF00	Mouser	505-FKP2D022201L00HF	CHF 0.442	CHF 1.77	
C322_A, C322_B, C330_A, 4 C330_B	1.5nF	Film Capacitor	CAP_WIMA_FKP	FKP2D011501D00HSSD	Distrelec	16542880	CHF 0.235	CHF 0.94	

Quantity	Designator	Description	Comment	Footprint	Manufacturer Part No.	Distributor Part No.	Price	Line Total
4	C323_A, C323_B, C334_A, C334_B	15nF	Film Capacitor	CAP_WIMA_FKP2	FKP2D021501J00HSSD	Mouse	505-FKP20.015/100/2	CHF 0.608 CHF 2.43
2	C326_A, C326_B	2.2nF	Film Capacitor	CAP_WIMA_FKP2	FKP2D012201D00HSSD	Distrelec	16542906	CHF 0.282 CHF 0.56
30	C400, C401, C402, C403, C406, C407, C408, C409, C600, C610, C703, C705, C718, C719, C727, C802, C1000, C1003, C1006, C1103, C1201, C1302, C1304, C1305, C1401, C1402, C1403, C1404, C1501, C1800	100nF	Ceramic Capacitor, Capacitor Ceramic SMD0603CAP	GCJ188R72A104KA01D	Mouse	81-GCJ188R72A104KA11	CHF 0.098	CHF 2.94
10	C601, C602, C603, C701, C702, C712, C716, C717, C1301, C1303	10uF	Capacitor Ceramic X5R 10uF 50V 20 SMD1210CAP	GCM32EC71H106MA03L	Mouse	81-GCM32EC71H106MA	CHF 0.834	CHF 8.34
5	C604, C605, C606, C607, C608	4.7uF	Capacitor Ceramic X5R 4.7uF 10V 1 SMD0603CAP	GRM188R61C475KAAJD	Digi-Key	490-10481-1-ND	CHF 0.248	CHF 1.24
2	C609, C803	22uF	Capacitor Ceramic X5R 22uF 10V 20 SMD0805CAP	GRM219R61C226ME15K	Mouse	81-GRM219R61C226ME	CHF 0.339	CHF 0.68
8	C700, C704, C711, C1400, C1603, C1604, C1702, C1703	10uF	Capacitor Ceramic X5R 10uF 10V 20 SMD0603CAP	GRM188R61A106KE68J	Digi-Key	490-14372-1-ND	CHF 0.072	CHF 0.58
1	C706	1uF	Capacitor Ceramic X5R 1uF 16V 20% SMD0603CAP	GCJ188R71E105KA01J	Mouse	81-GCJ188R71E105KA11	CHF 0.223	CHF 0.22
8	C707, C708, C709, C710, C722, C723, C724, C725	47uF	Capacitor Ceramic X5R 47uF 10V 20 SMD0805CAP	GRM21BR61A476ME15L	Digi-Key	490-9961-1-ND	CHF 0.556	CHF 4.45
9	C714, C900, C1002, C1005, C1008, C1200, C1300, C1407, C1600	1nF	Capacitor Ceramic X7R 1nF 50V 10% SMD0603CAP	GRM1885C1H102JA01J	Digi-Key	490-6379-1-ND	CHF 0.033	CHF 0.30
1	C715	470pF	Capacitor Ceramic X7R 470pF 50V 1 SMD0603CAP	GCM1885G2A471JA16D	Mouse	81-GCM1885G2A471JA6	CHF 0.047	CHF 0.05
1	C726	680pF	Capacitor Ceramic X7R 680pF 50V 1 SMD0603CAP	GCM1885C2A681GA16D	Mouse	81-GCM1885C2A681GA6	CHF 0.087	CHF 0.09
5	C729, C730, C1405, C1406, C1500	2.2uF	Capacitor Ceramic X5R 2.2uF 6.3V 2 SMD0603CAP	GRM188C81C225KA12D	Digi-Key	490-11993-1-ND	CHF 0.073	CHF 0.36
2	C901, C902	100pF	Capacitor Ceramic X7R 100pF 50V 1 SMD0603CAP	GCM1885C2A101JA16D	Digi-Key	490-4771-1-ND	CHF 0.050	CHF 0.10
3	C1001, C1004, C1007	100uF	Capacitor TA 100uF 10V 10% 6032 CAPMP6032X280	TR3C107K010C0100	Mouse	74-TR3C107K010C0100	CHF 0.746	CHF 2.24
1	C1102	47uF	Capacitor TA 47uF 6.3V 20% 3216 T_CAPMP3216X1	T55A476M6R3C0070	Mouse	74-T55A476M6R3C0070	CHF 0.418	CHF 0.42
10	D600, D1100, D1101, D1500, D1501, D1600, D1700, D1701, D1800, D1802	green	LED green	LGM67KG1J224Z	Mouse	720-LGM67KG1J224Z	CHF 0.249	CHF 2.49
1	D601		D-Littelfuse-SMAj Bidirectional ESD protection	T_LITTLEFUSE_S SMAj30CA	Mouse	576-SMAj30CA	CHF 0.332	CHF 0.33
1	D602		SS3P3 SS3P3-E3/84A Schottky Diode	T_VISHAY_SS3P3-E3/84A	Mouse	625-SS3P3-M3	CHF 0.319	CHF 0.32
2	D800, D1400		BAT54C BAT54 Schottky Diode	SOT95P230X110-BAT54C	Digi-Key	BAT54CFSCST-ND	CHF 0.218	CHF 0.44

Quantity	Designator	Description	Comment	Footprint	Manufacturer Part No.	Distributor Part No.	Price	Line Total
9	D1000, D1001, D1002, D1003, D1004, D1005, D1006, D1502, D1503	TPD2EUSB30DR12 Channel Protection for Super Speed	T_TL_TPD2EUSB30DRTR	Distrelec	30023305	CHF 0.458	CHF 4.12	
1	D1300	SS14 Schottky Diode 40V 1A	T_DO-214-AC	SS14	Mouse	512-SS14	CHF 0.293	CHF 0.29
2	D1401, D1402	RCLAMP0504S TVS Array low cap	T_SOT95P285X90	RCLAMP0504S	Mouse	947-RCLAMP0504S-TCT	CHF 0.760	CHF 1.52
2	D1801, D1803	LED red	LED	LS_M67K-J2L1-1-Z	Mouse	720-L-SM67K-J2L1-1-Z	CHF 0.288	CHF 0.58
1	F600	Fuse 7A Fast	T_Littelfuse_0458	0458007	Mouse	576-0458007.DR	CHF 1.080	CHF 1.08
1	IC700	L5150BNTR LDO Regulator	T_ON_NCV4264-2	L5150BNTR	Mouse	511-L-5150BNTR	CHF 1.050	CHF 1.05
1	IC701	TPS51120 Power Supply Controller	QFN50P500X500	TPS51120	Mouse	595-TPS51120RHBR	CHF 5.360	CHF 5.36
1	IC702	LTC2954CTS8-2# Push Button On/Off Controller	T_LINEAR_LTC2954CTS8-2#TRMPBF	Mouse	584-C2954CTS8-2TMRPF	CHF 4.540	CHF 4.54	
1	IC703	NC7SZ126M5X Buffer Three-State	SOT95P279X142	NC7SZ126M5X	Mouse	512-NC7SZ126M5X	CHF 0.312	CHF 0.31
2	IC704, IC1401	NCP511SN33T1G LDO Regulator	SOIC127P600X17	NCP511SN33T1G	Mouse	863-NCP511SN33T1G	CHF 0.459	CHF 0.92
1	IC800	RTC M41T0M6	SOIC127P600X17	M41T0M6	Mouse	511-M41T0M6F	CHF 1.920	CHF 1.92
1	IC1000	TPS2066CD 1A Dual High Side Switch (Current Limit)	SOIC127P600X17	TPS2066CD	Mouse	595-TPS2066CD	CHF 1.860	CHF 1.86
1	IC1001	TPS2052BD Dual High Side Switch With OC DET	SOIC127P600X17	TPS2052BD	Distrelec	30023325	CHF 1.950	CHF 1.95
1	IC1300	PAM2841SR Diodes Inc, PAM2841GR, DC/DC step-down	T_TSOP65P490X	PAM2841GR	Mouse	621-PAM2841SR	CHF 0.617	CHF 0.62
1	IC1400	HDMI12C1-6C1 HDMI ESD protection, level shifter and DFN16	HDMI12C1-6C1		Mouse	511-HDMI12C1-6C1	CHF 1.090	CHF 1.09
1	J200	1861946 Screw Terminal, 3 Position	1861946		Digi-Key	277-12125-ND	CHF 1.256	CHF 1.26
1	J202	B3B-PHK-S Connector 3x1P	B3B-PHK-S		Digi-Key	455-1705-ND	CHF 0.184	CHF 0.18
2	J300_A, J300_B	NC3MBH XLR Connector Male, Panel Mount	NC3MBH		Mouse	568-NC3MBH	CHF 3.800	CHF 7.60
1	J301_A	43-029 RCA Connector, red	RCA-Plug	43-029	Canford	43-029	CHF 6.560	CHF 6.56
1	J301_B	43-028 RCA Connector, black	RCA-Plug	43-028	Canford	43-028	CHF 6.560	CHF 6.56
1	J600	1861933 Screw Terminal, 2 Position, 5mm Spacing	1861933		Mouse	651-1861933	CHF 0.825	CHF 0.83
3	J700, J701, J1800	B2B-PHK-S Connector 2x1P	B2B-PHK-S		Digi-Key	455-1704-ND	CHF 0.164	CHF 0.49
1	J1500	1981568-1 USB Micro Connector	HEADER_USB	1981568-1	Mouse	571-1981568-1	CHF 1.620	CHF 1.62
1	J1801	B6B-PHK-S Connector 6x1P	B6B-PHK-S		Digi-Key	455-1703-ND	CHF 0.316	CHF 0.32
2	L600, L601	36R@100MHz Power Supply FB 10A	T_Laird_35F0121-35F0121-0SR-10		Digi-Key	240-2520-1-ND	CHF 0.190	CHF 0.38
6	L602, L603, L604, L605, L1600, L1700	33R@100MHz 3A EMI FILTER	INDC1608X95N	BLM18PG330SN1D	Digi-Key	490-5220-1-ND	CHF 0.044	CHF 0.26
1	L700	4.7uH Inductor 4.7uH 10A 20% 15.5 mOhm	T_INDMD110100	SRP1038A-4R7M	Mouse	652-SRP1038A-4R7M	CHF 1.100	CHF 1.10
1	L701	8.2uH Inductance 8.2uH 6.25A 20%	T_INDMD1120120X8	744771008	Digi-Key	732-1207-1-ND	CHF 2.140	CHF 2.14
9	L702, L1001, L1002, L1003, L1005, L1007, L1008, L1100, L1500	220R@100MHz 2A EMI FILTER	INDC1608X95N	BLM18KG221SN1D	Digi-Key	490-5255-1-ND	CHF 0.046	CHF 0.41
3	L1000, L1004, L1006	90R@100MHz Common Mode Choke High Speed	T_Murata_DLW21	DLW21SN900SQ2	Distrelec	15865733	CHF 0.570	CHF 1.71
1	L1300	22uH Inductor shielded 22uH 1.6A	T_INDMD5050	SRN5040-220M	Mouse	652-SRN5040-220M	CHF 0.279	CHF 0.28
1	L1400	120R@100MHz 2A EMI FILTER	INDC1608X95N	BLM18PG121SN1D	Mouse	81-BLM18PG121SN1D	CHF 0.035	CHF 0.04

Quantity	Designator	Description	Comment	Footprint	Manufacturer Part No.	Distributor Part No.	Price	Line Total
1	MECH600	Mini-PCIe Latch	Mini-PCIe Latch	T_Molex_48099-5	48099-5701	Mouser	538-48099-5701	CHF 1.580
1	MP201	FA-T220-38E	Heatsink	FA-T220-38E		Distrelec	30118244	CHF 1.450
3	MP600, MP601, MP602	9774030360R	Spacer M3 SMD		9774030360R	Mouser	710-9774030360R	CHF 1.45
2	MP603, MP604	9774080360R	Spacer M3 SMD		9774080360R	Mouser	710-9774080360R	CHF 4.65
2	MP1600, MP1700	9774015360R	Spacer M3 SMD		9774015360R	Distrelec	30070963	CHF 3.10
1	OSC800	CC4V-T1A	Quartz 32.768kHz	T_MC_CC4V-T1A		Mouser	428-203148-MG02	CHF 2.86
6	Q300_A, Q300_B, Q301_A, Q301_B, Q302_A, Q302_B	PMBT3904VS,114	Transistor Bipolar Dual	SOT-666	PMBT3904VS115	Mouser	771-PMBT3904VS115	CHF 0.849
1	Q400	SQ1539EH-T1_G	Transistor Dual, NMOS & PMOS	SOT-363-6	SQ1539EH-T1_GE3	Mouser	78-SQ1539EH-T1_GE3	CHF 0.185
2	R200, R207	560kΩ	Resistor	SMD0603RES	CRCW0603560KFKEA	Digi-Key	541-5601KHCT-ND	CHF 0.341
2	R201, R208	121kΩ	Resistor	SMD0603RES	CRCW0603121KFKEA	Digi-Key	541-1211KHCT-ND	CHF 0.016
2	R202, R206	15kΩ	Resistor	SMD0603RES	CRCW060315K0FKEA	Digi-Key	541-15.0KHCT-ND	CHF 0.016
19	R204, R210, R215, R217, R221, R223, R314_B, R316_A, R320_A, R320_B, R406, R704, R714, R715, R1301, R1308, R1400, R1401, R1504	10kΩ	Resistor	SMD0603RES	CRCW060310K0FKEA	Digi-Key	541-10.0KHCT-ND	CHF 0.34
1	R213	150kΩ	Resistor	SMD0603RES	CRCW0603150KFKEA	Digi-Key	541-1501KHCT-ND	CHF 0.016
1	R214	49.9kΩ	Resistor	SMD0603RES	CRCW060349K9FKEA	Digi-Key	541-49.9KHCT-ND	CHF 0.016
1	R219	91kΩ	Resistor	SMD0603RES	CRCW060391K0FKEA	Digi-Key	541-91.0KHCT-ND	CHF 0.016
1	R220	33.2kΩ	Resistor	SMD0603RES	CRCW060333K2FKEA	Digi-Key	541-33.2KHCT-ND	CHF 0.016
6	R302_A, R302_B, R303_A, R303_B, R328_A, R328_B	47Ω	Resistor Thin Film	SMD0603RES	TNPW060347R0BEEA	Mouser	71-TNPW060347R0BEEA	CHF 0.518
21	R304_A, R304_B, R305_A, R305_B, R306_A, R306_B, R307_A, R307_B, R313_A, R313_B, R315_A, R315_B, R404, R405, R725, R900, R901, R1200, R1300, R1505, R1506, R1900, R1901	22Ω	Resistor	SMD0603RES	CRCW060322R0FKEA	Digi-Key	541-22.0HCT-ND	CHF 0.34
4	R308_A, R308_B, R338_A, R338_B	22Ω	Resistor Thin Film	SMD0603RES	TNPW060322R0BEEA	Mouser	71-TNPW060322R0BEEA	CHF 0.518
12	R309_A, R309_B, R310_A, R311_A, R312_A, R312_B, R339_A, R339_B, R340_A, R340_B	2.2kΩ	Resistor	SMD0603RES	CRCW06032K20FKEA	Mouser	71-CRCW06032K20FKEA	CHF 0.016
8	R318_A, R318_B, R319_A, R319_B, R341_A, R341_B, R342_A, R342_B	270Ω	Resistor Thin Film	SMD0603RES	TNPW0603270R0BEEA	Mouser	71-TNPW0603270R0BEEA	CHF 0.463

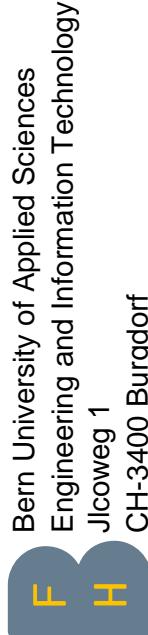
Quantity	Designator	Comment	Description	Footprint	Manufacturer Part No.	Distributor Part No.	Price	Line Total
8	R321_A, R321_B, R325_A, R325_B, R331_A, R331_B, R335_A, R335_B	340Ω	Resistor Thin Film	SMD0603RES	TNPW0603340RBEA	Mouser	71-TNPW0603340RBEA	CHF 0.353 CHF 2.82
9	R324_A, R324_B, R330_A, R330_B, R333_A, R333_B, R337_A, R337_B, R802	10Ω	Resistor	SMD0603RES	CRCW060310R0FKEA	Mouser	71-CRCW0603-10-E3	CHF 0.027 CHF 0.24
4	R326_A, R326_B, R332_A, R332_B	200Ω	Resistor Thin Film	SMD0603RES	TNPW0603200RBEA	Digi-Key	541-3049-1-ND	CHF 0.660 CHF 2.64
8	R327_A, R327_B, R329_A, R329_B, R334_A, R334_B, R336_A, R336_B	82Ω	Resistor Thin Film	SMD0603RES	ERA-3AEB822N	Digi-Key	P82DBCT-ND	CHF 0.300 CHF 2.40
2	R401, R402	1.5kΩ	Resistor	SMD0603RES	CRCW06031K50FKEA	Mouser	71-CRCW0603-1.5K-E3	CHF 0.027 CHF 0.05
12	R600, R1100, R1101, R1500, R1501, R1600, R1700, R1701, R1804, R1805, R1808, R1809	560Ω	Resistor 560 ohm 63mW 5% 0603	SMD0603RES	CRCW0603560R0FKEA	Digi-Key	541-560HCT-ND	CHF 0.016 CHF 0.20
3	R700, R701, R712	4.7R	Resistor 4.7 ohm 63mW 5% 0603	SMD0603RES	CRCW06034R70FKEA	Digi-Key	541-4.70HHCT-ND	CHF 0.018 CHF 0.05
20	R702, R707, R709, R713, R716, R720, R721, R724, R726, R805, R902, R903, R904, R1005, R1602, R1801, R1806, R1807, R1810, R1811	100K	Resistor 100 Kohm 63mW 5% 0603	SMD0603RES	CRCW0603100KFKEA	Mouser	71-CRCW0603-100K-E3	CHF 0.016 CHF 0.32
26	FB200, FB201, FB202, FB203, R300_A, R300_B, R301_A, R301_B, R400, R407, R1201, R1202, R703, R711, R722, R1002, R1003, R1004, R1703, R1704, R1705, R1706, R1707, R1708, R1709, R1803	0R	Resistor 0 ohm 63mW 1% 0603	SMD0603RES	CRCW0603000Z0EAC	Mouser	71-CRCW0603000Z0EA	CHF 0.027 CHF 0.70
1	R705	12mR	Resistor 12 mohm 250mW 1% 1206	SMD1206RES	WSK1206R0120FEA18	Mouser	71-WSK1206R0120FEA1	CHF 0.776 CHF 0.78
3	R706, R718, R1405	24K	Resistor 24 Kohm 63mW 1% 0603	SMD0603RES	CRCW060312K0FKEA	Distrelec	16059731	CHF 0.027 CHF 0.08
1	R708	3.3K	Resistor 3.3 Kohm 63mW 1% 0603	SMD0603RES	CRCW06033K30FKEA	Digi-Key	541-3.30KHCT-ND	CHF 0.016 CHF 0.02
1	R717	15mR	Resistor 15 mohm 250mW 1% 1206	SMD1206RES	WSK1206R0150FEA18	Mouser	71-WSK1206R0150FEA1	CHF 0.776 CHF 0.78
1	R719	3.6K	Resistor 3.6 Kohm 63mW 1% 0603	SMD0603RES	CRCW06033K60FKEA	Mouser	71-CRCW06033K60FKEA	CHF 0.038 CHF 0.04
1	R800	470R	Resistor 470 ohm 63mW 5% 0603	SMD0603RES	CRCW060347RFKEA	Digi-Key	541-470HCT-ND	CHF 0.016 CHF 0.02
4	R803, R804, R1402, R1404	1.8K	Resistor 1.8 Kohm 63mW 5% 0603	SMD0603RES	CRCW06031K80FKEA	Digi-Key	541-1.80KHCT-ND	CHF 0.016 CHF 0.07
1	R1303	910K	Resistor 910 Kohm 63mW 1% 0603	SMD0603RES	CRCW0603910KFKEA	Mouser	71-CRCW0603910KFKEA	CHF 0.036 CHF 0.04
1	R1304	59K	Resistor 59 Kohm 63mW 1% 0603	SMD0603RES	CRCW060359K0FKEA	Digi-Key	541-59.0KHCT-ND	CHF 0.016 CHF 0.02
1	R1305	5K	Resistor 5 Kohm 63mW 1% 0603	SMD0603RES	CRCW06034K99FKEA	Digi-Key	541-4.99KHCT-ND	CHF 0.016 CHF 0.02
1	R1306	3.3R	Resistor 3.3 ohm 63mW 1% 0603	SMD0603RES	CRCW06033R30JNEB	Mouser	71-CRCW06033R30JNEB	CHF 0.008 CHF 0.01

Quantity	Designator	Comment	Description	Footprint	Manufacturer Part No.	Distributor Part No.	Price	Line Total
1	R1307	90K	Resistor 90 Kohm 63mW 1% 0603	SMD0603RES	CRCW060391K0FKEAC	Mouse	71-CRCW060391K0FKEA	CHF 0.04
2	R1403, R1601	470K	Resistor 470 Kohm 63mW 5% 0603	SMD0603RES	CRCW060391K0FKEA	Digi-Key	541-470KHCT-ND	CHF 0.03
1	R1502	24kΩ	Resistor	SMD0603RES	CRCW060324K0FKEAC	Mouse	71-CRCW060324K0FKEA	CHF 0.03
2	R1503, R1702	47kΩ	Resistor	SMD0603RES	CRCW060347K0FKEAC	Mouse	71-CRCW060347K0FKEA	CHF 0.02
1	R1800	220Ω	Resistor	SMD0603RES	CRCW0603220RFKEA	Digi-Key	541-220HCT-ND	CHF 0.02
19	RA700, RA800, RA900, RA1200, RA1300, RA1301, RA1302, RA1303, RA1304, RA1305, RA1306, RA1307, RA1400, RA1500, RA1800, RA1801, RA1802, RA1803, RA1804	22R	Resistor Array 22 R 63mW 5% 1206	T_Resistor_Array	CAY16-22R0F4LF	Mouse	652-CAY16-22R0F4LF	CHF 0.65
2	T700, T703	FDS6910DKR	N-Channel PowerTrench SyncFET	SOIC127P600X17	FDS6910DKR	Digi-Key	FDS6910CT-ND	CHF 1.450
1	T701	IRLML6401PbF	p-channel MOSFET, IRLML6401PbF	SOT95P237X112-IRLML6401PbF		Mouse	942-IRLML6401TRPB	CHF 0.326
6	T702, T705, T1600, T1800, T1801, T1802	T-Vishay-SI-1024	Dual N-Channel 20V (D-S) MOSFET	SOTFL50P160X6(SI-1024-X)		Mouse	781-SI1024X-T1-GE3	CHF 0.432
1	T704	DMP2022LSS-13	p-channel MOSFET, DMP2022LSS-	SOIC127P600X17	DMP2022LSS-13	Mouse	621-DMP2022LSS-13	CHF 0.555
1	T1301	SI2312CDS	N-Channel 20V (D-S) MOSFET	T_SOT95P230X11S12312CDS		Mouse	781-SI2312CDS-T1-GE3	CHF 0.342
2	TP600, TP601	5011	Testpin Eye TH	T_TP-THT - EYE	5011	Mouse	534-5011	CHF 0.268
1	U200	BD5250G-TR	Voltage Supervisor, Threshold 5V	SOT-23-5	BD5250G-TR	Mouse	755-BD5250G-TR	CHF 0.367
1	U201	TLV810SDBZ	Voltage Supervisor, Threshold 2.94V	SOT-23	TLV810SDBZR	Mouse	595-TLV810SDBZR	CHF 0.488
2	U300_A, U300_B	OPA1612	SoundPlus Operational Amplifier	SOIC8	OPA1612AIDR	Mouse	595-OPA1612AIDR	CHF 5.730
2	U301_A, U301_B	AK4493	32-Bit 2ch Audio DAC	LQFP48	AK4493EQ	Digi-Key	974-1162-ND	CHF 7.954
2	U302_A, U302_B	OPA1611	SoundPlus Operational Amplifier	SOIC8	OPA1611AIDR	Mouse	595-OPA1611AIDR	CHF 3.660
3	U400, U402, U403	Si8660BB-B-IU	Unidirectional digital isolator	SOIC16WIDE	Si8660BB-B-IU	Digi-Key	336-4298-ND	CHF 3.340
1	U401	MAX14933	Bidirectional digital isolator	SOIC16WIDE	MAX14933	Mouse	700-MAX14933ASE+	CHF 2.420
1	U1200	MAX9110	LVDS Transmitter	SOIC8	MAX9110ESA+	Mouse	700-MAX9110ESA+	CHF 3.660
1	U1500	CP2105	USB to UART Bridge	QFN24	CP2105-F01-GMR	Digi-Key	CP2105-F01-GMRC-T-ND	CHF 1.580
3	VR200, VR202, VR203	LT3045	Linear Regulator, positive Rail, Low NSOP12	LT3045EMSE#PBF	Digi-Key	LT3045EMSE#PBF-ND	CHF 6.850	
1	VR201	LT3094	Linear Regulator, negative Rail Low MSOP12	LT3094EMSE#PBF	Mouse	584-LT3094EMSE#PBF	CHF 6.740	
1	VR204	LD1086V/33-DG	Linear Voltage Regulator, positive Ra TO220-3	LD1086V/33-DG	Mouse	511-LD1086V/33-DG	CHF 1.530	
1	X1	CON-JAE-MM70-1MXM3 Connector	T_JAE_MM70-314(MM70-314-310B1	Digi-Key	670-2431-1-ND	CHF 7.250		
1	X700	TSW-103-01-G-D	Receptacle Female Socket, 2x3 Pins	T_Samtec_SSW-SSW-103-01-G-D	Digi-Key	SAM1208-03-ND	CHF 0.940	
1	X1000	CON-Fci-73725-0	USB Type A Connector	T_FCI_73725-01173725-0110BLF	Mouse	649-73725-0110BLF	CHF 0.886	
1	X1002	GSB4116341HR	USB 3.1, Type A, sideways	GSB4116341HR	Mouse	523-GSB4116341HR	CHF 1.850	
1	X1003	GSB412137CHR	USB 3.1, Type A, vertical	GSB412137CHR	Mouse	523-GSB412137CHR	CHF 1.440	
1	X1100	5-2301995-1	Vertical 10/100/G Mag Jack	5-2301995-1	Digi-Key	A126265-ND	CHF 8.520	
1	X1300	XF2M-4015-1A	RGB Connector	T_Omron_XF2M-4XF2M-4015-1A	Mouse	653-XF2M-4015-1A	CHF 2.290	

Quantity	Designator	Description	Comment	Footprint	Manufacturer Part No.	Distributor Part No.	Price	Line Total
1	X1301	52207-0633	1mm Pitch FFC/FPC Connector	T_MOLEX_52207	52207-0633	Mouser	538-52207-0633	CHF 0.64
1	X1400	0476591001	HDMI Connector Vertical	0476591001	0476591001	Mouser	538-47659-1001	CHF 2.08
1	X1600	1-2199119-5	Connector M.2 Key M	1-2199119-5-Key-	1-2199119-5	Mouser	571-1-2199119-5	CHF 1.27
1	X1700	2199119-8	Connector M.2 Key A	2199119-8-KEY-A	2199119-8	Mouser	571-2199119-8	CHF 1.16
1	Y400	C3391-22.5792	HCMOS Oscillator	OSC_CRYSTEK	C3391-22.5792	Mouser	549-C3391-22.5792	CHF 2.01
1	Y401	C3391-24.576	HCMOS Oscillator	OSC_CRYSTEK	C3391-24.576	Mouser	549-C3391-24.576	CHF 2.01
<b>545</b>								<b>CHF 279.57</b>

## Bill of Materials

Project: PSU  
 Author: luths8  
 Variant: Default  
 Revision: 1.0  
 Date: 11.5.19



Quantity	Designator	Description	Comment	Footprint	Manufacturer Part No.	Distributor	Distributor Part No.	Price	Line Total
2	C1, C3	10mF	Elektrolytic Capacitor	CAP-25-50-12.5 -	UKW1H103MRD	Digi-Key	493-10688-ND	CHF	5.93 CHF 11.86
2	C5, C6	100µF	Elektrolytic Capacitor	CAP-8-11.5-3.5	EEUJFR1H101	Digi-Key	P14454-ND	CHF	0.31 CHF 0.62
1	D1	KBP308G	Full Bridge Rectifier	KBP308G	Mouser	621-KBP308G	CHF	0.78 CHF 0.78	
1	D2	SLDB107S	Full Bridge Rectifier	SLDB107S	Mouser	583-SLDB107S	CHF	0.30 CHF 0.30	
1	J1	1861933	Screw Terminal, 2 Position	1861933	Mouser	651-1861933	CHF	0.83 CHF 0.83	
1	J2	B2B-PH-K-S	Connector 2x1P	B2B-PH-K-S	Digi-Key	455-1704-ND	CHF	0.16 CHF 0.16	
1	J3	1862275	Screw Terminal, 2 Position, 5mm Spacing	1862275	Digi-Key	277-121239-ND	CHF	0.88 CHF 0.88	
1	J4	1861946	Screw Terminal, 3 Position	1861946	Digi-Key	277-12125-ND	CHF	1.26 CHF 1.26	
1	J5	B3B-PH-K-S	Connector 3x1P	B3B-PH-K-S	Digi-Key	455-1705-ND	CHF	0.18 CHF 0.18	
1	T1	L01-6363	Transformer	L01-6364	Digi-Key	TE2259-ND	CHF	22.95 CHF 22.95	
1	U1	IRM-60-12	AC/DC Converter (100VAC - 240VAC)	IRM-60-12	Mouser	709-IRM60-12	CHF	18.34 CHF 18.34	
1	U2	CPC1926Y	OptoMOS Solid State Relay	CPC1926Y	Digi-Key	CLA210-ND	CHF	5.66 CHF 5.66	
<b>14</b>							<b>CHF 63.81</b>		

## Bill of Materials

Project: Encoder		Bern University of Applied Sciences Engineering and Information Technology Jicoweg 1 CH-3400 Burgdorf	
Author: luths8	Revision: 1.0	Date	11.5.19
Variant: None			

Quantity	Designator	Description	Comment	Footprint	Manufacturer Part No.	Distributor	Distributor Part No.	Price	Line Total
3	C1, C2, C3	10nF	Ceramic Capacitor	SMD0603CAP	GRM188R72A103JA01D	Mouser	81-GRM188R72A103JA1	CHF 0.04	CHF 0.12
6	D1, D2, D3, D4, D5, D6	LED blue	LED blue	LED	LB_M673-M1N2-35-Z	Digi-Key	475-2681-1-ND	CHF 0.69	CHF 4.14
1	J1	B6B-PHK-S	Connector 6x1P	B6B-PHK-S	B6B-PHK-S	Digi-Key	455-1703-ND	CHF 0.32	CHF 0.32
1	Q1	SQ1539EH-T1_G	Transistor Dual, NMOS & PMOS	SOT-363-6	SQ1539EH-T1_GE3	Mouser	78-SQ1539EH-T1_GE3	CHF 0.34	CHF 0.34
6	R1, R2, R3, R4, R5, R6	10kΩ	Resistor	SMD0603RES	CRCW060310K0FKEA	Digi-Key	541-10-0KHCT-ND	CHF 0.02	CHF 0.10
6	R7, R8, R9, R10, R11, R12	270Ω	Resistor	SMD0603RES	CRCW0603270KFKEAC	Mouser	71-CRCW0603270KFKEA	CHF 0.04	CHF 0.23
1	T1	PEC11L-4220F-S	Rotary Encoder with button	PEC11L-4220F-S	PEC11L-4220F-S0015	Mouser	652-PEC11L4220FS0015	CHF 2.00	CHF 2.00
								<b>CHF 7.24</b>	

## Bill of Materials

Project: **Audio-Streamer Assembly (1 pcs)**

Author: **luths8** Revision: **1.0**  
Date **08.06.2019**



Bern University of Applied Sciences  
Engineering and Information Technology  
Jlcoweg 1  
CH-3400 Burgdorf

Quantity	Designator	Comment	Description	Manufacturer Part No.	Distributor	Distributor Part No.	Price	Line Total
1	A1	V1.1 (Complete)	Mainboard Assembly	-	BFH	-	CHF 279.57	CHF 279.57
1	A1	V1.0	Mainboard PCB	-	Multi-CB	-	CHF 70.00	CHF 70.00
1	A2	V1.1 (Default)	PSU Assembly	-	BFH	-	CHF 63.81	CHF 63.81
1	A2	V1.0	PSU PCB	-	Multi-CB	-	CHF 33.00	CHF 33.00
1	A3	V1.1 (No Variant)	Encoder Assembly	-	BFH	-	CHF 7.24	CHF 7.24
1	A3	V1.0	Encoder PCB	-	Multi-CB	-	CHF 15.00	CHF 15.00
1	A4	1GB	Apalis i.MX6 Quad	00271101	Toradex	00271101	CHF 125.30	CHF 125.30
1	HS	-	Heat Sink Apalis Type 3	23071001	Toradex	23071001	CHF 12.30	CHF 12.30
1	CASE	-	Aluminium Case	BZ4309-L	Aliexpress	BZ4309-L	CHF 65.00	CHF 65.00
1	CBL1	2P	Jumper Cable, 102mm	A02KR02KR26E102B	Digi-Key	455-3445-ND	CHF 0.94	CHF 0.94
1	CBL10 G	mm, black	Stranded Wire 1mm2	-	BFH	-	CHF 0.00	CHF 0.00
1	CBL10 N	mm, blue	Stranded Wire 1mm2	-	BFH	-	CHF 0.00	CHF 0.00
1	CBL10 P	mm, red	Stranded Wire 1mm2	-	BFH	-	CHF 0.00	CHF 0.00
1	CBL2	2P	Jumper Cable, 305mm	A02KR02KR26E305B	Digi-Key	455-3154-ND	CHF 1.08	CHF 1.08
1	CBL3	6P	Jumper Cable, 305mm	A06KR06KR26E305B	Digi-Key	455-3156-ND	CHF 1.06	CHF 1.06
1	CBL5	CAT5e	Ethernet Patch Cable	1403926	Digi-Key	277-12036-ND	CHF 10.31	CHF 10.31
1	CBL6	-	USB 3.0 Cable Male A - Male A	3023001-01M	Digi-Key	Q542-ND	CHF 6.69	CHF 6.69
1	CBL7	3P	Jumper Cable, 254mm	A03KR03KR26E254B	Digi-Key	455-3455-ND	CHF 1.10	CHF 1.10
1	CBL8 L	50mm, brown	Stranded Wire 1mm2	-	BFH	-	CHF 0.00	CHF 0.00
1	CBL8 N	50mm, blue	Stranded Wire 1mm2	-	BFH	-	CHF 0.00	CHF 0.00
1	CBL9 G	mm, black	Stranded Wire 1mm2	-	BFH	-	CHF 0.00	CHF 0.00
1	CBL9 P	mm, red	Stranded Wire 1mm2	-	BFH	-	CHF 0.00	CHF 0.00
1	DISP1	-	Display with Capacitive Touch Panel	NHD-5.0-800480MB-ATXL	Digi-Key	NHD-5.0-800480MB-ATXL	CHF 64.50	CHF 64.50
4	FT1	M6	Rubber Feet	-	BFH	-	CHF 0.00	CHF 0.00
1	PAN1	-	USB A Shell Panel	AC-USB3-AAB	Mouse	523-AC-USB3-AAB	CHF 10.89	CHF 10.89
1	PAN2	-	Power Entry Module	FN261S-6-06-10	Mouse	631-FN261S-6-06-10	CHF 12.22	CHF 12.22
1	PAN3	CAT5e	Ethernet Shell Panel	CP30220SM	Distrelec	300-30-442	CHF 8.55	CHF 8.55
1	PAN4	-	Pushbutton	PV2F240NNM01	Mouse	612-PV2F240NNM01	CHF 5.65	CHF 5.65
12	SCR2	M3 x 8mm	Screw, lens head	-	BFH	-	CHF 0.00	CHF 0.00
12	SCRW1	M3 x 8mm	Screw, flat head	-	BFH	-	CHF 0.00	CHF 0.00
12	SPCR1	M3 x 15mm	Spacer Plastic	-	BFH	-	CHF 0.00	CHF 0.00
67								CHF 794.21

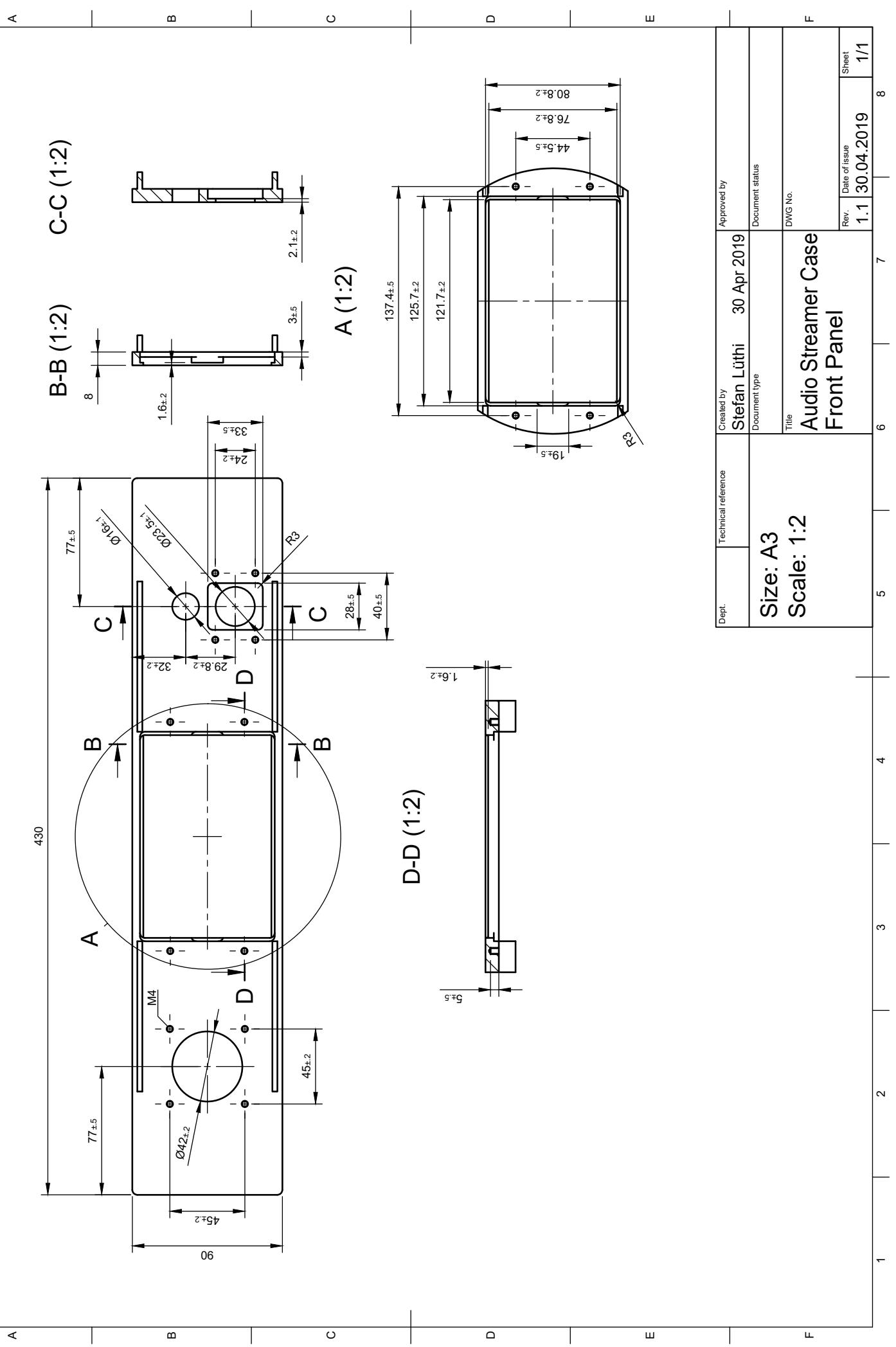
### Extension Pack

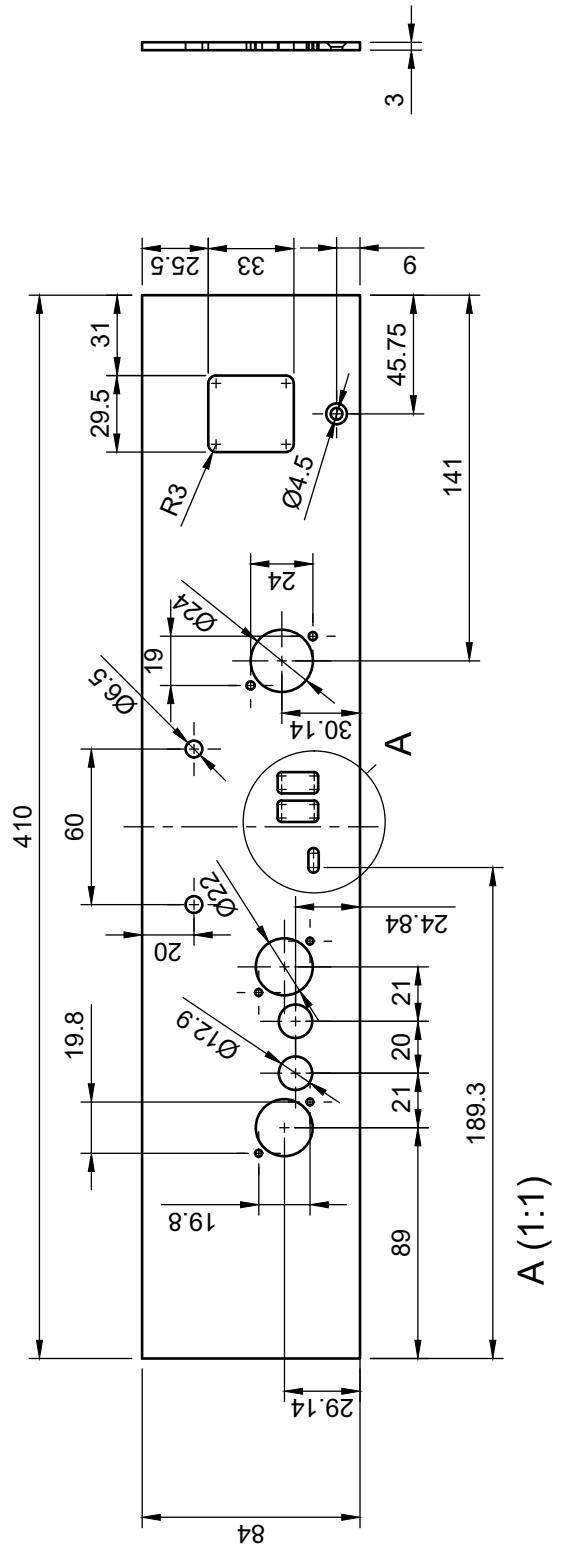
2	ANT1	RPSMA	Antenna 2.4GHz, 5GHz	WTSB2450- RPSMA	Digi-Key	994-1031-ND	CHF 7.78	CHF 15.56
1	A4	-	WiFi & Bluetooth Module M.2 Key A&E 2230	Wireless-AC 7265	Digitec	5336643	CHF 32.90	CHF 32.90
1	A5	500GB	SSD M.2 Key B&M 2280 SATA	WDS500G2B0B	Digitec	6408467	CHF 65.60	CHF 65.60
2	CBL4	-	RF Cable Assembly U.FL - RP-SMA	336314-14-0200	Digi-Key	ARF2393-ND	CHF 8.19	CHF 16.38
6								CHF 130.44

## **Anhang C**

# **Vermassungen**







Dept.	Technical reference	Created by	Approved by
		Stefan Lüthi	28.5.19
	Document type		
Size: A3	Scale: 1:2	Title	DWG No.
View: Back		Audio Streamer Case Back Panel	
Rev.	Date of issue		
1.0	28.05.19		
		Sheet	1/1

A

B

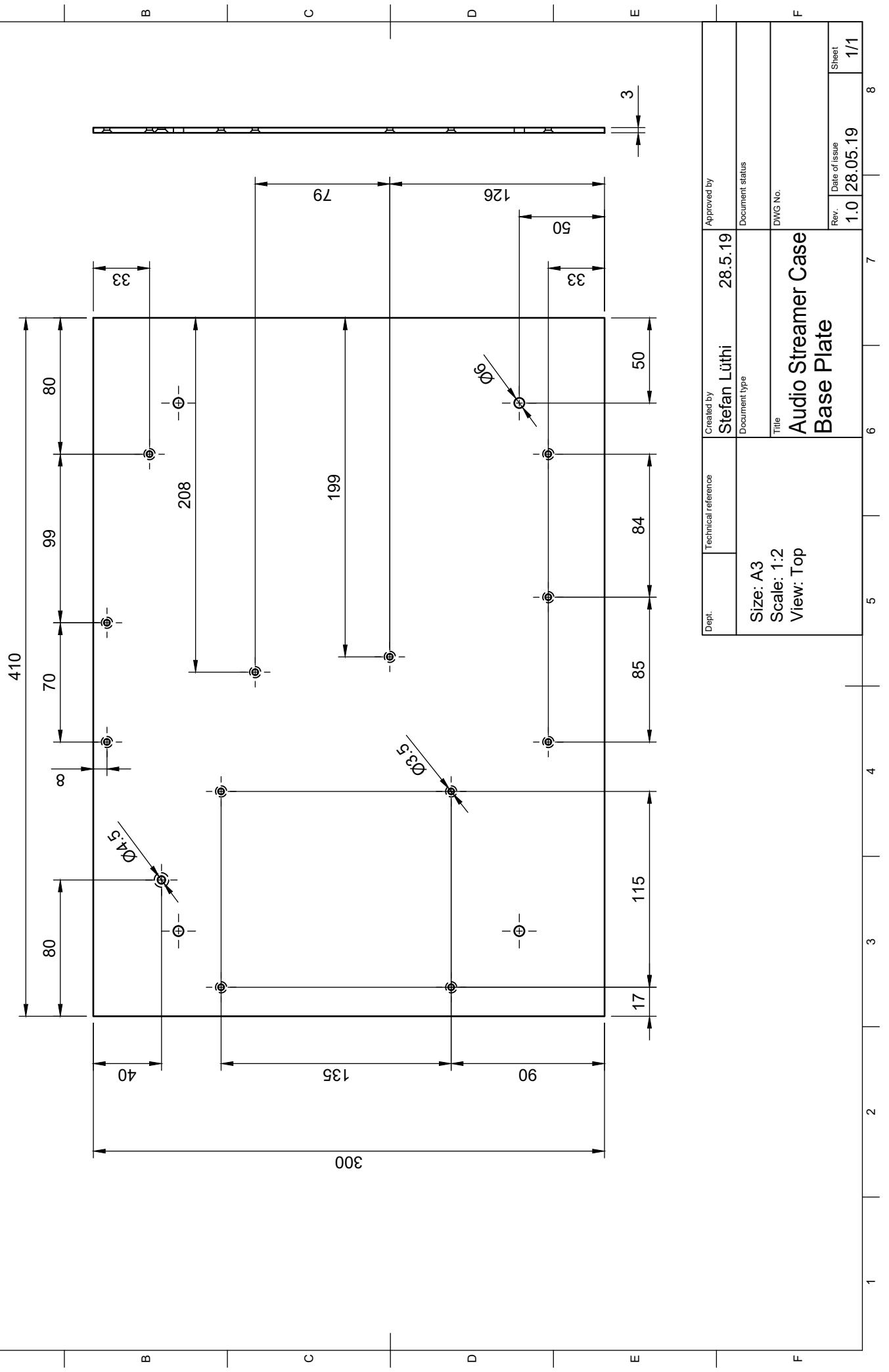
C

D

E

F

8  
7  
6  
5  
4  
3  
2  
18  
7  
6  
5  
4  
3  
2  
1



Dept.	Technical reference	Created by	Approved by
		Stefan Lüthi	28.5.19
	Document type		Document status
Size: A3	Scale: 1:2	DWG No.	
View: Top	Title: Audio Streamer Case Base Plate		
Rev. 1.0	Date of issue 28.05.19	Sheet 1/1	



## **Anhang D**

# **Verdrahtungsplan**



# Verdrahtungsplan

