

Ansteuerung einer Signalerzeugungseinheit zur automatisierten Platinenprüfung

Bachelorarbeit

Angefertigt von:
Andreas Fischer

am Bachelor-Studiengang Elektronik und Computer Engineering
der FH JOANNEUM – University of Applied Sciences, Austria

in Zusammenarbeit mit:
RSF Elektronik Ges.m.b.H.

unter der Anleitung von:
FH-Prof. DI Dr. Egon Teiniker

Tarsdorf, 11.06.2023

Eidesstattliche Erklärung

Ich versichere hiermit an Eides statt, dass ich die vorliegende Bachelorarbeit selbstständig und ohne fremde Hilfe angefertigt und die mit ihr verbundenen Tätigkeiten selbst erbracht habe und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Die hochgeladene Version und die allenfalls abgelieferte gedruckte Version sind identisch.

Ich erkläre zudem, dass ich die Arbeit im Sinne der Prinzipien der Richtlinie der FH JOANNEUM zur Sicherung Guter Wissenschaftlicher Praxis und zur Vermeidung von Fehlverhalten in der Wissenschaft erstellt habe. Insbesondere erkläre ich, dass ich Inhalte, die ich aus Werken Dritter oder auch aus eigenen Werken wörtlich oder inhaltlich übernommen habe, geeignet - und den Regeln für gutes wissenschaftliches Arbeiten entsprechend - gekennzeichnet und die Informationsquellen durch detaillierte Quellenangaben deutlich ersichtlich gemacht habe.

Die vorliegende Originalarbeit ist in dieser Form zur Erreichung eines akademischen Grades noch keiner anderen Hochschule im In- oder Ausland vorgelegt worden.

Ich bin mir bewusst, dass eine unwahre eidesstattliche Erklärung rechtliche Folgen haben kann.

Kurzfassung

In Zusammenarbeit mit dem Unternehmen RSF Elektronik Ges.m.b.H. wurde ein In-Circuit Tester entwickelt. Für diesen In-Circuit Tester wurde ein Arduino, ein Frequenzgenerator und eine selbst designte Platine verwendet, welche es ermöglicht, die im Haus bestückten Platinen zu testen.

Die selbst entwickelte Schaltung, kann das Ausgangssignal des Frequenzgenerators individuell verstärken, sowie dessen Offset verschieben. Darüber hinaus, wird das Eingangssignal bei einem weiteren Schaltungszweig invertiert. Das invertierte Signal kann ebenfalls individuell verstärkt und Offset verschoben werden. Durch die mehrfache Ausführung dieser Schaltungselemente, können aus vier Eingangssignalen, acht Ausgangssignale erzeugt werden.

Ebenfalls werden auf der Platine diverse Spannungs-Niveaus erzeugt, welche benötigt werden, um den Arduino und alle weiteren Bauteile betreiben zu können. Außerdem wurden einige weitere unternehmensrelevante (Mess-)Umschaltungen realisiert.

Die entwickelte Software ist in der Lage, alle Komponenten auf der Platine via einem SPI-Protokoll zu steuern. Der Arduino dient dabei als Datenübersetzer zwischen einem Modbus RTU Client und den SPI gesteuerten Bauelementen auf der Platine.

Die entwickelte Schaltung, in Kombination mit dem Arduino-Progamm, ist in der Lage, Sinus-Testsignale zu parametrieren. Dabei ist die entwickelte Schaltung in der Lage, Signale bis zu einer Grenzfrequenz von etwa 700 kHz zu modifizieren. Bis zu dieser Frequenz, wird eine Phasenverschiebung von 180° , mit einer maximalen Abweichung von $\pm 1\%$ realisiert.

Stichwörter: Arduino, SPI, Modbus RTU, Frequenzgenerator

Abstract

In cooperation with the company RSF Elektronik Ges.m.b.H., an in-circuit tester was developed. For this in-circuit tester, an Arduino, a frequency generator and a self-designed circuit board were used, which makes it possible to test the in-house assembled boards.

The self-developed circuit can amplify the output signal of the frequency generator individually and shift the offset. In addition, the signal is inverted in another circuit branch. The inverted signal can also be individually amplified and offset shifted. Due to the multiple design of these circuit elements, eight output signals can be generated from four input signals. Various voltage levels are also generated on the board, which are needed to operate the Arduino and all other components. In addition, some further company-relevant (measurement) switches were realised.

The developed software is able to control all components on the board via an SPI protocol. The Arduino serves as a data translator between a Modbus RTU client and the SPI-controlled components on the board.

The developed circuit, in combination with the Arduino programme, is able to parameterise sine test signals. The developed circuit is able to modify signals up to a cut-off frequency of about 700 kHz. Up to this frequency, a phase shift of 180° is realised with a maximum deviation of $\pm 1\%$.

Keywords: Arduino, SPI, Modbus RTU, frequency generator

Inhaltsverzeichnis

1 Einleitung	1
2 Kommunikationsprotokolle und Hardwarekomponenten	4
2.1 Interfaces	4
2.1.1 Serial Peripheral Interface: Internes Datenprotokoll	4
2.1.2 Modbus: Externe Datenkommunikation	9
2.2 Verwendete Bauteile	13
2.2.1 Arduino Mega 2560 Rev3	13
2.2.2 Frequenzgenerator: AD9106-ARDZ-EBZ (EVAL-BOARD)	14
2.2.3 Digital-Analog-Wandler: AD5672R	17
2.2.4 Nicht flüchtiges digitales Potentiometer: MAX5487	18
3 Ansteuerung einer Signalerzeugungseinheit.....	20
3.1 Dimensionierung der Schaltungsgruppen	20
3.2 Implementierung	35
3.2.1 Verwendete GPIOs	38
3.2.2 Funktionserklärung	40
3.2.3 Aufbau des Kommunikationsprotokoll	47
3.3 Projektumsetzungs Ergebnisse	48
4 Diskussion der Ergebnisse	54
5 Fazit und Ausblick	60
A Anhang	65

Abbildungsverzeichnis

2.1	Controller und Device Register Verbindung.	5
2.2	Kommunikation mit mehreren Busteilnehmern.	5
2.3	SPI Übertragung CPHA = 0.	7
2.4	SPI Übertragung CPHA = 1.	7
2.5	Arduin Mega 2560 Rev3.	13
2.6	EVAL-BOARD AD9106-ARDZ-EBZ.	14
2.7	Digital-Analog-Wandler: AD5672R, TSSOP-20.	17
2.8	Digitales Potentiometer: MAX5487, TQFN 16 mit EP.	18
2.9	In-Circuite Tester: Teradyn Teststation LH Ultrapin 2.	19
3.1	Blockschaltbild der entwickelten Schaltung.	21
3.2	Blockschaltbild des ersten Strangs der Schaltung.	22
3.3	Signalstrang 0 der entwickelten Schaltung.	23
3.4	Schaltungausschnitt des Eingangsdifferenzverstärkers.	25
3.5	Schaltungausschnitt der Signalinvertierung.	26
3.6	Schaltungausschnitt der variablen Verstärkung.	28
3.7	Schaltungausschnitt des Ausgangsdifferenzverstärker mit Impedanzwandlung. .	30
3.8	Schaltungausschnitt der Laufzeitanpassung.	31
3.9	Top- und Bottom-Layer der entwickelten PCB.	34
3.10	Aufbau der Software.	35
3.11	Visualisierung der @brief Kurzbeschreibung in Visual Studio Code 1.77.1.	37
3.12	Zeitsignal des vierten Kanals, bei einer Frequenz von $f = 100$ kHz, Messung der Tabelle A.7, Ch1 dunkelblau = Eingangssignal, Ch2 hellblau = Singal 6, Ch3 pink = Signal 7.	48
3.13	Vergleich des Bode Diagramm der Simulation sowie der Messung des Kanals 0. .	49
3.14	Vergleich der Phasenverschiebung der Simulation sowie der Messung des ersten Kanals.	50
3.15	Bode Diagramm aller gemessenen Signale.	51
3.16	Vergleich der Phasenverschiebung von allen Signalen.	52
3.17	Individuelle veränderbare Offset und Verstärkung.	53
4.1	Bode Diagramm des Kanal 0 und der Simulation vor der Optimierung. Messdaten in Tabelle A.8	54

4.2	Phasenunterschied des Kanal 0 und der Simulation vor der Optimierung. Messdaten in Tabelle A.8	55
4.3	Bode Diagramm der Teilschaltung des Kanal 0 und der Simulation. Messdaten in Tabelle A.9	56
4.4	Phasenunterschied der Teilschaltung des Kanal 0 und der Simulation. Messdaten in Tabelle A.9	57
A.1	PCB 3D view.	65
A.2	PCB top view.	66
A.3	PCB bottom view.	66
A.4	Vollbestückte Leiterplatte.	67

Tabellenverzeichnis

2.1	SPI-Modi.	6
2.2	Aufbau des Modbus Datenübertragungsprotokoll.	10
2.3	Modbus - Übersicht der wichtigsten Funktion-Codes.	10
2.4	AD9106-ARDZ-EBZ: Umbestückung der Jumper.	15
3.1	Überblick der auftretenden Phasenverschiebung.	32
3.2	Überblick der Transmission der Schaltung. Transmissionsparameter: $U_e = 1 \text{ V}$; $f = 500 \text{ kHz}$	33
3.3	Arduino GPIO Verwendung Teil 1/2.	38
3.4	Arduino GPIO Verwendung Teil 2/2.	39
4.1	Messdaten der digitalen Potentiometer (MAX5487).	58
A.1	Modbus Kommunikationsprotokoll Teil 1/3.	75
A.2	Modbus Kommunikationsprotokoll Teil 2/3.	76
A.3	Modbus Kommunikationsprotokoll Teil 3/3.	77
A.4	Messdaten des Kanal 0 für Bode Diagramm.	78
A.5	Messdaten des Kanal 1 für Bode Diagramm.	79
A.6	Messdaten des Kanal 2 für Bode Diagramm.	80
A.7	Messdaten des Kanal 3 für Bode Diagramm.	81
A.8	Messdaten des Kanal 0 für Bode Diagramm ohne Optimierung.	82
A.9	Messdaten des Kanal 0 für Bode Diagramm der Teilschaltung.	83

1. Einleitung

RSF Elektronik Ges.m.b.H. ist einer der Weltmarktführer im Bereich der hochpräzisen Längenmesssysteme [1, Unternehmen], welches seit 1994 teil des Heidenhain Konzerns ist. Neben den Längenmesssystemen zählen unter anderem auch Winkelmessgeräte, Präzisionsteilungen sowie kundenspezifische Kabelsysteme zum Produktportfolio [1, Produkte].

Bei der Längen- sowie Winkelmessung differenziert man zwischen inkrementellen und absoluten Messsystemen [1, Längenmesssysteme]. Außerdem kann zwischen offene und gekapselte Messsysteme unterschieden werden. In Bezug auf die Technologie kann in induktive, kapazitive sowie optische Systeme unterschieden werden, um einige der möglichen Messtechnologien zu nennen.

Im Unternehmen werden zur Gänze optische Längenmesssysteme entwickelt, sowie gefertigt. Die gesamte Produktion dieser Messsysteme erfolgt am Firmensitz in Tarsdorf (Oberösterreich). Um die im Unternehmen entwickelten Platinen zu Testen und zu Prüfen werden Platinenprüfsysteme verwendet.

Platinenprüfsysteme sind in der heutigen Zeit nicht mehr wegzudenken. Sie sind zuständig, um die richtige Bestückung sicherzustellen, darüber hinaus wird auch die Annahme getroffen, dass eine Leiterplatte funktioniert. Dieser Test kann auf Bauteilebene, sowie auf Baugruppen beziehungsweise der gesamten Platine durchgeführt werden, um die Funktion nachzuweisen.

Ein Großteil aller weltweit hergestellten Platinen werden mittels eines ICT überprüft. Dies ist nur einer, aus einer Vielzahl von möglichen Tests, welcher auch im Unternehmen RSF Elektronik angewendet wird. Bei dem sogenannten In-Circuit Test handelt es sich um ein Array, bestehend aus Federstiftkontakte, die es ermöglichen, Signale abzugreifen, aber auch deziidierte Prüfsignale in die Schaltung einzuspeisen. Um die richtige Positionierung sicherzustellen, wird jede Platine einzeln in eine Vorrichtung eingelegt, wodurch das Verrücken der Federstiftkontakte verhindert wird. [2]

Durch das Abgreifen und Einspeisen von Signalen können bestückte passive Bauteilwerte von Widerständen, Kapazitäten und Spulen, sowie nicht bestückte oder kurzgeschlossene Bauteile bestimmt werden. Ebenfalls kann in eingeschränkter Form, die Funktionalität der Schaltung bei bestimmten Eingangssignalen verifiziert werden. Der Hauptverwendungszweck ist jedoch das Prüfen der Bestückung. [2]

Die Genauigkeit der Messung der Bauteileigenschaften, kann jedoch bei Kapazitäten stark eingeschränkt sein, wenn die Störkapazität im Bereich der zu messenden Kapazität liegt. Ein ähnliches Problem besteht bei Induktivitäten. Jedenfalls kann der geringe Bauteilwert, beziehungsweise deren starke Frequenzabhängigkeit ermittelt werden. Ein weiteres Problem kann auftreten, wenn Kontakte von Bauteilen abgeschirmt sind, wodurch der Zugriff stark eingeschränkt ist. [2]

Das bisher verwendete Platinenprüfsystem, besteht aus einer Vorrichtung, welche die Platine mit Federkontakte stift kontaktiert. Die eigens entwickelte Schaltung, erzeugt die Prüfsignale in drei Kanälen. Das zentrale Problem besteht darin, dass die Hardware, die die Prüfsignale erzeugt, aufwendig sowie veraltet ist. Deshalb ist der Wartungsaufwand sehr groß und kostenintensiv. Ein weiteres Problem besteht darin, dass die Prüfsignalparameter nur sehr geringfügig geändert werden können und nicht universell einsetzbar, für eine Vielzahl an Messsystemen, sind. Die Messung der resultierenden Signale wird aktuell von einem Oszilloskop übernommen, welches in dieser Ausführung nicht mehr hergestellt wird, wodurch eine Anpassung von zukünftigen Test-Systemen ohnehin von Nöten wäre.

Um ein System zu entwickeln, womit die genannten Probleme behoben werden können, lassen sich folgende Ziele definieren: Das Nachfolgesystem soll generell kompakter und flexibler in der Prüfsignalerzeugung sein. Da für die Bauteileigenschaftenbestimmung ein Sinus, sowie dessen Konjunktion auf drei Kanälen benötigt wird, soll eine Schaltung entwickelt werden. Darüber hinaus ist es nötig, dass jedes der sechs resultierenden Signale, individuell Offset verschoben werden kann, sowie dessen Verstärkung individuell einstellbar ist. Die Amplitude des Signals soll 0,5 V betragen, wobei eine Phasenverschiebung von 90° für den zweiten Kanal und 135° für den dritten Kanal auftreten muss. Generell sollen diese Signalparameter während des Betriebs verändert werden können. Der Entwicklungsaufwand soll gering gehalten werden, indem möglichst viele Komponenten verwendet werden, welche frei am Markt verfügbar sind. Weiters sollen Prüfsignale für einen Frequenzbereich von bis zu 500 kHz möglich sein, wobei die Schaltung eine exakte Phasenverschiebung von 180° mit einer Toleranz von $\pm 1\%$ und eine maximale Dämpfung von -3 dB jedes individuellen Signals nicht überschreiten darf. Ebenfalls soll in Zukunft das Oszilloskop durch das Messsystem PWM 21 des Unternehmens

DR. JOHANNES HEIDENHAIN GmbH ersetzt werden. Da man für die Zukunft gerüstet sein will, soll jegliche Art von Erweiterung vorgesehen sein. Dazu zählen zusätzliche digitale/analoge Ein- und Ausgänge, Kommunikationsschnittstellen wie U(S)ART und I²C sowie ein vierter frei parametrierbarer Signalausgangskanal. Als Kommunikationsprotokoll soll ein Modbus RTU Protokoll implementiert werden, wodurch alle zuvor genannten Eigenschaften frei parametrierbar sind.

Aus diesen Systemanforderungen kann nun die Forschungsfrage abgeleitet werden, die lautet: Welche Schaltung sowie Hardware- und Softwareaufbau wird benötigt, um ein Signal zu Konjugieren und die zuvor genannten Parameter via einem Modbus RTU Protokoll zu verändern?

Um dieses Ziel zu erreichen, wurde ein Modbus RTU Server implementiert, welcher die Signalparameter empfängt. Diese Parameter werden dazu verwendet, um die Komponenten in der entwickelten Schaltung zu steuern und die gewünschten Prüfsignale zu erhalten.

2. Kommunikationsprotokolle und Hardwarekomponenten

Im folgenden Kapitel werden die beiden verwendeten Kommunikationsprotokolle, sowie die wichtigsten Schaltungskomponenten beschrieben.

2.1. Interfaces

Damit der Arduino MEGA 2560 mit der übergeordneten Steuereinheit (Personal Computer) sowie den untergeordneten Komponenten kommunizieren kann, werden Kommunikationsprotokolle verwendet.

Als Grundlage an Informationen wurden die folgenden Quellen als Referenz für dieses Kapitel verwendet: [3, S. 190-199] [4, SPI, Serial] [5] [6].

2.1.1. Serial Peripheral Interface: Internes Datenprotokoll

In der nachfolgenden Erläuterung wird anstelle des historisch gewachsenen Ausdrucks ‚Master‘ Controller und anstelle des Ausdrucks ‚Slave‘ Device verwendet. Bei der getakteten seriellen Schnittstelle SPI¹ handelt es sich anders als bei den meisten Schnittstellen um eine synchrone Schnittstelle. Durch den Einsatz eines separaten Takt-Signals erzielt diese Schnittstelle eine besonders hohe Datenübertragungsrate, jedoch muss bei zunehmender Länge der Leitungen auf das erhöhte EMV²-Risiko beachtet werden.

¹SPI: **S**erial **P**eripheral **I**nterface

²EMV: **E**lektro**M**agnetische **V**erträglichkeit

Aufbau und Funktionsweise

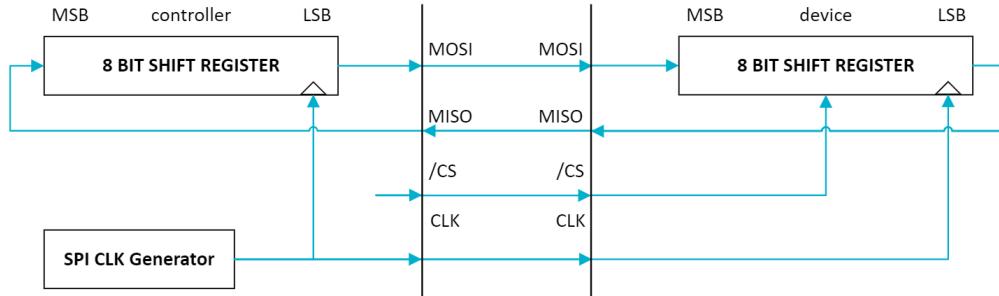


Abbildung 2.1.: Controller und Device Register Verbindung.

In jedem Aufbau muss mindestens ein Controller verfügbar sein, der den Datenaustausch anstößt. Der Vorteil von Controllern ist, dass diese meist auch als Device betrieben werden können, wenn es deren Konfiguration zulässt.

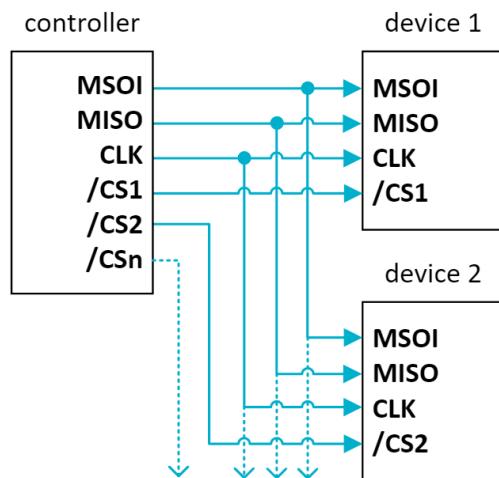


Abbildung 2.2.: Kommunikation mit mehreren Busteilnehmern.

Wie bereits in der vorangegangenen Abbildung assoziiert wurde, wird für jedes Device ein eigenes CS-Signal benötigt. Da es sein kann, dass die Empfangsdaten länger als die Sendedaten sind, muss die erweiterte Länge ebenfalls übertragen werden. Dies geschieht indem Dummy-Werte gesendet werden.

Konfiguration der Schnittstelle

Weil die meisten Devices eine voreingestellte Konfiguration besitzen, welche oft nicht geändert werden kann, muss der Controller so konfiguriert werden, damit das Device die übertragenen Daten richtig interpretiert.

Hierzu muss zu Beginn der Kommunikation festgelegt werden, ob das höchstwertigste Bit (MSB³) oder das niederwertigste Bit (LSB⁴) am Start übertragen wird. Darüber hinaus wird bei der Übertragung zwischen der Clock Polarität (CPOL⁵) sowie der Clock Phase (CPHA⁶) unterschieden. CPOL bestimmt, ob das Clock Signal active high oder active low ist, wohin CPHA angibt, bei welcher Flanke (fallenden/steigenden) die übertragenen Daten gültig sind. Aus den beiden genannten Parametern lassen sich nun die folgenden vier Modi ableiten.

Tabelle 2.1.: SPI-Modi.

Modus	CPOL	CPOH
0	0	0
1	0	1
2	1	0
3	1	1

³MSB: Most Significant Bit

⁴LSB: Least Significant Bit

⁵CPOL: Clock POLarity

⁶CPHA: Clock PHAse

In den nachfolgenden Grafiken werden die unterschiedlichen Modi grafisch veranschaulicht.

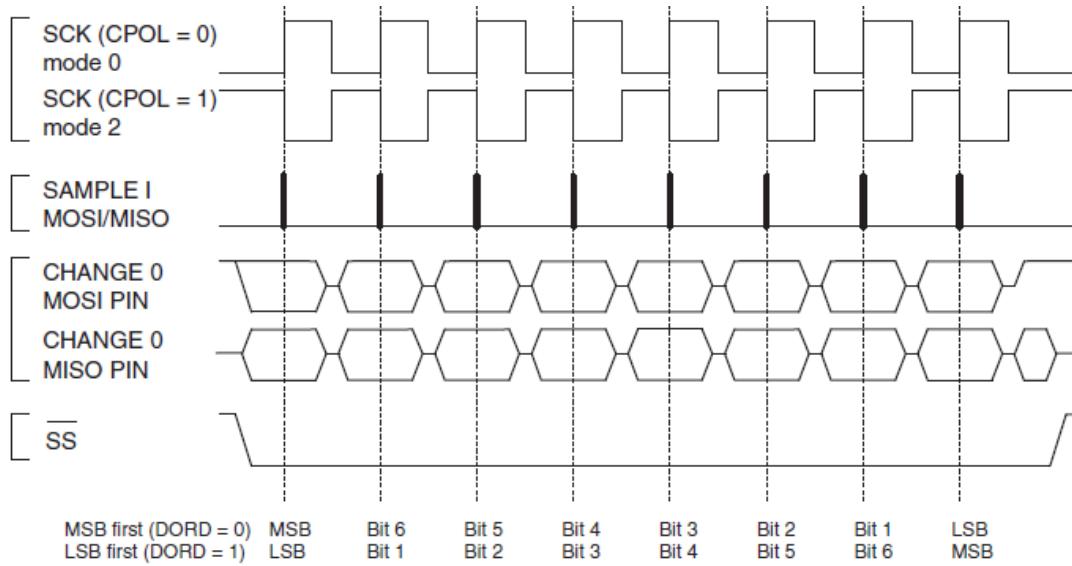


Abbildung 2.3.: SPI Übertragung CPHA = 0.[3, S. 196]

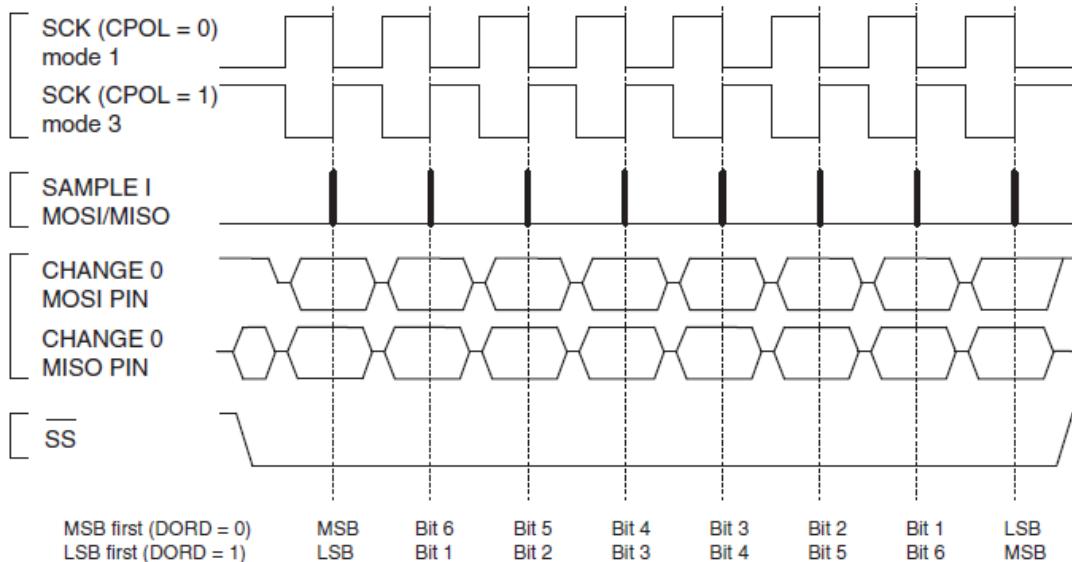


Abbildung 2.4.: SPI Übertragung CPHA = 0.[3, S. 196]

Des Weiteren kann neben den zuvor genannten Parametern der Clock-Divider eingestellt werden, welche die Wahl der Frequenz ermöglicht. Zumeist sind die Clock-Divider /2 /4 /8 /16 /32 /64 /128 möglich, woraus sich abhängig des System-Takts die SPI-Frequenz ableitet.

Konfiguration der SPI-Schnittstelle des Arduino MEGA 2560

Zu Beginn der Konfiguration des SPI-Moduls muss die vom Framework bereitgestellte SPI-Bibliothek integriert werden. (#include <SPI.h>)

Der erste Befehl, ,SPI.begin()', aktiviert die SPI-Schnittstelle und stellt sicher, dass die richtigen Pins für die SPI-Kommunikation konfiguriert sind (MOSI &CLOCK = OUTPUT, MISO = INPUT). Im zweiten Befehl, ,SPISettings settings(SPI_SPEED, MSBFIRST, SPI_MODE3)', wird ein Objekt der Klasse SPI-Settings mit den gewünschten Parametern für die SPI-Kommunikation angelegt. Dabei werden die Datenübertragungsrate, die Reihenfolge der Bitübertragung sowie der Übertragungsmodus festgelegt. Im letzten Befehl, ,SPI.beginTransaction(settings)', startet eine neue SPI-Transaktion mit den Einstellungen, die zuvor im Objekt 'settings' definiert wurden.

Die konfigurierte SPI-Schnittstelle ist nun bereit Daten zwischen dem Arduino-Board und anderen Geräten auszutauschen, wobei die zuvor definierten Einstellungen für die Übertragung gelten.

Code 2.1: Beispiel Code für die Initialisierung und Parametrierung einer SPI-Schnittstelle.

```
1 #include <SPI.h>
2
3 void setup()
4 {
5     SPI.begin();
6     SPISettings settings(SPI_SPEED, MSBFIRST, SPI_MODE3);
7     SPI.beginTransaction(settings);
8 }
```

Nun kann die Schnittstelle mit dem Befehl ,SPI.transfer()' 8 Bits und mit dem Befehl ,SPI.transfer16()' 16 Bits an Daten übertragen. Es muss dabei berücksichtigt werden, dass das \overline{CS} -Signal des jeweiligen Busteilnehmers zusätzlich gesetzt werden muss, da dies nicht automatisch geschieht.

2.1.2. Modbus: Externe Datenkommunikation

Für die Verfassung des nachfolgenden Kapitel wurden die folgenden Quellen als Referenz herangezogen: [5] [7] [8]

Anders als bei der SPI-Kommunikation handelt es sich bei dem Modbus-Protokoll um ein Software-Protokoll. Prinzipiell kann zwischen drei Modbus-Arten unterschieden werden:

- Modbus TCP (Transmission Control Protocol)
- Modbus RTU (Remote Terminal Unit)
- Modbus ASCII (American Standard Code for Information Interchange)

Großteils wird zur Übertragung der Daten eine serielle Schnittstelle (RTU, ASCII) oder Ethernet verwendet. Aufgrund der Relevanz von Modbus RTU für dieses Projekt wird in dieser Betrachtung nur die Übertragungsart Modbus RTU behandelt.

Aufbau und Funktionsweise

In einem Modbus-Netzwerk muss es exakt einen Client, sowie mindestens einen Server geben. Die Kommunikation wird immer vom Client angestoßen, woraufhin der Server antwortet.

Bei dem Modbus RTU (nachfolgend kurz Modbus) Protokoll werden die Daten binär übertragen. Dabei beginnt jede Übertragung mit einer Mindestpause von 3,5 Z⁷, wodurch diese Mindestpause von der Datenübertragungsrate bestimmt wird. Gefolgt von den beiden Bytes bestehend aus der Adresse sowie des auszuführenden Funktion-Codes. Dabei liegen die Server-Adressen im Bereich von 1 bis 247, da die Adresse 0 reserviert ist für eine Sammelnachricht an alle Server (Broadcast) ist. Nach der Übertragung des Funktion-Codes erfolgen die eigentlichen Daten, wobei bis zu 252 Bytes an Daten während eines Übertragungszyklus gesendet werden können. Wird die Datenübertragung für länger als 1,5 Z unterbrochen, so werden die übertragenen Daten als unvollständig betrachtet und müssen neu übermittelt werden. Nach der Datenübertragung folgen zwei Bytes welche den CR-Check⁸ beinhalten.

⁷Z: Zeichen, 1 Z = 11 Bit

⁸CR-Check: Cyclic Redundancy-Check

Tabelle 2.2.: Aufbau des Modbus Datenübertragungsprotokoll.

Start	Adresse	Funktion-Code	Daten	CR-Check
Wartezeit (3,5 Z)	1 Byte	1 Byte	1 - 252 Byte(s)	2 Bytes

Funktion-Codes

Folgende Funktion-Codes werden als relevant gesehen, um mit einem Modbus-Server zu kommunizieren, welche ebenfalls in der verwendeten Modbus-Bibliothek 'ModbusRTUSlave' implementiert wurden.

Tabelle 2.3.: Modbus - Übersicht der wichtigsten Funktion-Codes.

Code	Beschreibung	
0x01	READ Coil	Zustand eines binären Ausgangs lesen
0x03	READ Holding	Inhalt eines Holding-Register lesen
0x04	READ Input	Inhalt eines Input-Register lesen
0x05	WRITE Coil	Zustand eines einzelnen binären Ausgangs schreiben
0x06	WRITE Holding	Inhalt eines einzelnen Holding-Register schreiben

Konfiguration der seriellen Schnittstelle des Arduinos

Eine beliebte Schnittstelle des Arduinos ist die serielle Schnittstelle, da jeder Arduino mindestens eine U(S)ART⁹ Kommunikationsschnittstelle besitzt. Mittels dieser Schnittstelle kann ein Arduino mit einer externen Hardware oder mit einem Personal Computer kommunizieren. Bei dieser Kommunikation ist es wichtig, dass sich die beiden Geräte auf eine gemeinsame Übertragungsrate (Zeichen pro Sekunde, Bit/s, 1 Zeichen = 1 Bit) sowie über die Parität (no-, even- oder odd-parity) und des Stoppbits einigen. Bei der Anbindung von externer Hardware muss ebenfalls auf den Logik-Pegel der Schnittstelle geachtet werden. Ausgeführt ist diese Schnittstelle an den beiden Pins RX (Receiver) und TX (Transmitter).

Mit dem Funktionsaufruf `Serial.begin()`, welcher als Parameter die gewünschte Baudrate benötigt, sowie optional der Wahl der Parity und des Stop-Bits (default: 8N1), wird die serielle Schnittstelle initialisiert.

Code 2.2: Initialisierung der seriellen Schnittstelle.

```
1 Serial.begin(BAUDRATE);
```

⁹U(S)ART: **U**niversal (**S**ynchronous/)A**synchronous** R**eceiver**/**T**ransmitter

Konfiguration des Modbus RTU Servers des Arduinos

Um die ausgewählte Modbus-Library zu wenden, muss diese eingebunden werden. Darüber hinaus wird eine Modbus Klasse angelegt, welcher das physische Protokoll, sowie ein Buffer und dessen Größe übergeben wird.

Bei der Initialisierung der Klasse wird die Server-Adresse und die physikalische Übertragungsrate festgelegt. In den nachfolgenden Zeilen werden die implementierten Schreibe- und Lesefunktionen der Klasse übergeben, welche von der Library aufgerufen werden, in Abhängigkeit des übermittelten Funktions-Codes.

Es ist jedoch wichtig zu verstehen, dass jegliches Antworten auf eine Anfrage in der Empfangsfunktion implementiert werden muss, da dies nicht von der ModbusRTUSlave-Library übernommen wird.

Code 2.3: Initialisierung der Modbus Schnittstelle.

```
1 #include <ModbusRTUSlave.h>
2
3 byte buffer[BUFFER_SIZE] = {0};
4
5 ModbusRTUSlave modbus(Serial, buffer, BUFFER_SIZE);
6
7 void setup()
8 {
9     modbus.begin(CLIENT_ID, BAUDRATE);
10    modbus.configureCoils(NUM_OF_COILS, coilRead, coilWrite);
11    modbus.configureInputRegisters(NUM_OF_INPUT_REGISTERS,
12                                    inputRegisterRead);
13    modbus.configureHoldingRegisters(NUM_OF_HOLDING_REGISTERS,
14                                    holdingRegisterRead, holdingRegisterWrite);
15 }
```

2.2. Verwendete Bauteile

Die nachfolgenden Hauptkomponenten des Projekts wurden bereits vor Beginn dieser Bachelorarbeit in einem Konzept des Unternehmens RSF Elektronik Ges.m.b.H. definiert.

2.2.1. Arduino Mega 2560 Rev3

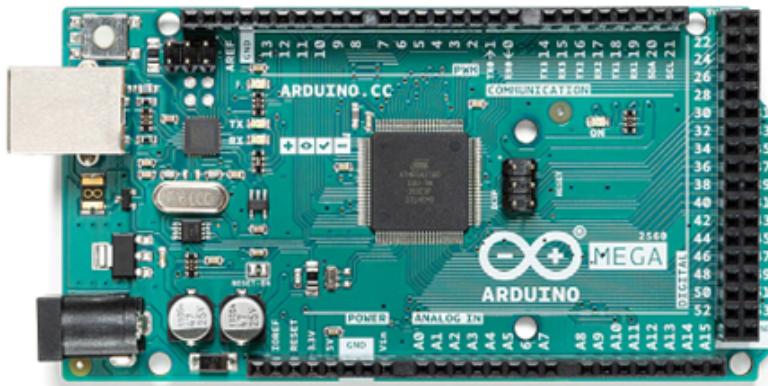


Abbildung 2.5.: Arduin Mega 2560 Rev3.[9]

Der Arduino Mega 2560 Rev3 (folgend kurz Arduino genannt) ist ein leistungsstarker Mikrokontroller basierend auf dem ATmega2560 Chip. Seine große Anzahl an digitalen Eingängen und Ausgängen, sowie analogen Eingängen in Kombination mit dessen Vielfalt an Kommunikationsschnittstellen macht ihn geeignet, für eine Vielzahl an Projekten.

Für dieses Projekt war nicht nur seine Vielfalt an GPIOs¹⁰ notwendig, sondern ebenso die SPI-Schnittstelle, sowie die I²C-Schnittstelle. Diese ermöglicht die einfache Erweiterung.

Der Arduino Mega verfügt über einen 16 MHz Takt mit 135 Anweisungen sowie 256 kB Flash-Speicher, 8 kB SRAM und 4 kB EEPROM. Mit 86 programmierbaren Eingängen/Ausgängen bietet er 16 ADCs (10 Bit) und 70 digitale Pins, von denen 16 PWM-fähig sind (4x 8 Bit, 12x 16 Bit). Zudem besitzt der Arduino Mega ein breites Spektrum an Schnittstellen wie SPI, USB, I²C und 4x U(S)ART sowie Timer/Counter (4x 16 Bit, 2x 8 Bit) und 6 Hardware-Interrupts als periphere Merkmale. [3]

¹⁰GPIO: General Purpose Input / Outputs

2.2.2. Frequenzgenerator: AD9106-ARDZ-EBZ (EVAL-BOARD)

Der verbaute Chip AD9106 ist ein leistungsstarker DAC, welcher ein DDS¹¹-Modul beinhaltet. Da dieser Chip sich auf dem AD9106-ARDZ-EBZ EVAL-BOARD befindet inklusive einer Vorverstärkerschaltung, eignet sich dieses Board für den Einsatz in dieser Signalerzeugungseinheit.



Abbildung 2.6.: EVAL-BOARD AD9106-ARDZ-EBZ.[10]

Der AD9106 ist ein hoch integrierter quad DAC mit einem On-Chip 4096×12 Bit Muster-Speicher (SRAM), On-Chip DDS für Sinus-Erzeugung und einer SPI-Schnittstelle (MODE3). Er bietet eine Abtastrate von 180 MSPS, einen Balun Umwandler (single-ended to differential), eine interne Clock mit 180 MHz sowie die Möglichkeit einer externen Clock. Das EVAL-BOARD wurde in einem kleinen Formfaktor gehalten und ist Stackable für Arduino Uno/Mega und SDP-K1 Boards. [11]

¹¹DDS: Direct Digital Synthesis

Einige Widerstände und Verbindungen wurden umgelötet, um die Sinus-Erzeugung und die SPI-Kommunikation über alternative Pins zu ermöglichen. Konkret wurden die folgenden Bauelemente umgelötet oder entfernt und kurzgeschlossen:

Tabelle 2.4.: AD9106-ARDZ-EBZ: Umbestückung der Jumper.[10, schematic]

AD9106-ARDZ-EBZ	Jumper	original	modifiziert
Kanal 0	JP3	A-COM	B-COM
	JP7	1-2	2-3
	JP8	1-2	2-3
Kanal 1	JP4	A-COM	B-COM
	JP9	1-2	2-3
	JP10	1-2	2-3
Kanal 2	JP5	A-COM	B-COM
	JP11	1-2	2-3
	JP12	1-2	2-3
Kanal 3	JP6	A-COM	B-COM
	JP13	1-2	2-3
	JP14	1-2	2-3

Des Weiteren wurden die Widerstände R_{19} , R_{23} , R_{29} und R_{38} entfernt und an Stelle der nicht bestückten Widerstände R_{20} , R_{24} , R_{30} und R_{39} bestückt, um die SPI-Pins des Arduinos verwenden zu können. Anschließend wurden die Kondensatoren C_{25} , C_{26} , C_{54} und C_{55} kurzgeschlossen, damit ein Gleichstrom am Ausgang fließen kann. [10, schematic]

Folgende 16 Bit Register des AD9106 werden benötigt, um ein Sinus-Signal am Ausgang des Frequenzgenerators zu erzeugen.

Einstellung der Signalart: Register 0x26 & 0x27

Das Register 0x26 ist für die Einstellung der Signalart der Kanäle 3 und 4, sowie das Register 0x27 für die Signalart der Kanäle 3 und 4 zuständig. Um einen Sinus auf allen 4 Kanälen zu erhalten, müssen beide Register mit dem Wert 0x3232 beschrieben werden. [11]

Einstellung der Amplitude: Register 0x32 – 0x35

Diese vier Register sind für die Amplitude des Frequenzgenerators verantwortlich. Die Einstellung der Amplituden der Kanäle 1-4 erfolgt über die Register 0x35 bis 0x32. Die maximale Amplitude beträgt 1300 mV, welches das 16 Bit Register abbilden kann. [11]

Einstellung der Frequenz: Register 0x3e & 0x3f

Die Frequenz der DAC-Kanäle kann nur kollektiv geändert werden. Das Register 0x3e beinhaltet dabei die beiden High-Bytes des 24 Bit Werts. Woraus gemeinsam mit dem höherwertigen Byte des Registers 0x3f die Frequenz resultiert. [11]

Einstellung der Phasenverschiebung: Register 0x40 – 0x43

Die Register 0x43 bis 0x40 sind für die Phasenverschiebung der Kanäle 1 bis 4 autorisiert. Dabei kann die Phase mittels des 16 Bit Register um bis zu 360° verschoben werden. [11]

Einstellung der Offsetverschiebung: Register 0x22 – 0x25

Diese vier Register sind für die Offsetverschiebung des Frequenzgenerators abgestellt. Dadurch wird eine maximale Offsetverschiebung von ± 360 mV erzielt. [11]

Wie im weiteren Verlauf erläutert, wird die Offsetverschiebung des Frequenzgenerators nicht verwendet, da die Offsetverschiebung mit der nachfolgenden Schaltung nicht verstärkt werden soll.

2.2.3. Digital-Analog-Wandler: AD5672R

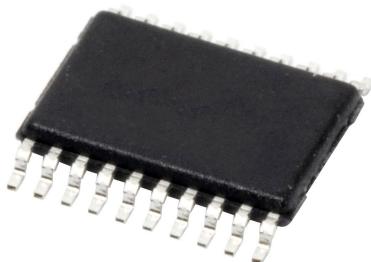


Abbildung 2.7.: Digital-Analog-Wandler: AD5672R, TSSOP-20.[12]

Dieser Digital-Analog-Wandler wurde aufgrund der 12 Bit einstellbaren Spannung in Abhängigkeit des Verstärkungsfaktors und der Referenzspannung ausgewählt. Des Weiteren besitzt dieser Baustein neben dem geringen Offsetfehler ebenso eine SPI-Schnittstelle.

Der AD5672 ist ein nanoDAC+ mit 8 Ausgängen und einer 12 Bit Auflösung. Er verfügt über eine SPI-Schnittstelle (MODE0) mit einer maximalen Geschwindigkeit von 50 MHz und bietet eine besonders kleine Bauform mit erweiterten Funktionen. Der AD5672 weist einen Offsetfehler von maximal $\pm 15 \text{ mV}$ auf und eine veränderbare Verstärkung mittels eines Pins (Gain = x1 oder x2) sowie eine INL¹² von $\pm 0,1 \text{ LSB}$ und eine DNL¹³ von $\pm 1 \text{ LSB}$. Ebenso kann der Power-Down-Modus gewählt werden sowie dessen Reset-Verhalten. Die acht Ausgangsregister sind doppelt gepuffert. [13]

¹²INL: Integrale NichtLinearität

¹³DNL: Differenzielle NichtLinearität

2.2.4. Nicht flüchtiges digitales Potentiometer: MAX5487

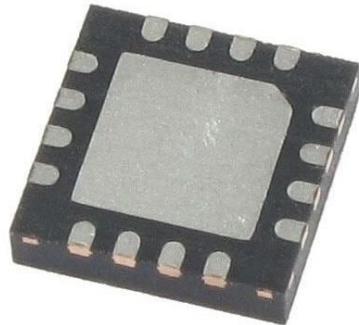


Abbildung 2.8.: Digitales Potentiometer: MAX5487, TQFN 16 mit EP.[14]

Dieses Bauelement wurde ausgewählt aufgrund der sehr guten Linearität sowie dessen 8 Bit Auflösung. Weiters besitzt es, wie alle anderen verwendeten Bauelemente, ein SPI-Interface mit nicht flüchtiger Speicherung der Schleifkontakteposition.

Der MAX5487 ist ein linearer digitaler Widerstand mit 256 Schritten und einem Längswiderstand von $10\text{ k}\Omega$, in dessen Gehäuse sich zwei unabhängig steuerbare Potentiometer befinden. Er verfügt über eine 3-Wire SPI-Schnittstelle (MODE3) und einen Schleifkontaktewiderstand von etwa $325\text{ }\Omega (\pm 25\%)$, welcher im integrierten EEPROM gespeichert werden kann. Der MAX5487 ist geeignet für den Single-Supply Betrieb und bietet eine INL von $\pm 1\text{ LSB}$ und eine DNL von $\pm 0,5\text{ LSB}$. [15]

In-Circuit Tester

Mit einem, wie in der nachfolgenden Abbildung dargestellten In-Circuit Tester, welcher auch im Unternehmen RSF Elektronik verwendet wird, werden die meisten Platinen in der Wirtschaft getestet. Jedoch ist ein solches System nicht für jegliche Prüfung geeignet.

Wie die Produktseite [16] verspricht, kann durch ein solches Gerät eine Großzahl an PCBs in einer kurzen Zeit getestet werden. Dabei ergibt sich eine hohe Fehlerabdeckung und Genauigkeit. Neben den geringen Erhaltungskosten wird damit geworben, dass kleinste Bauteile auf Platinen getestet werden können. Durch dessen Großzahl an Testpunkten (bis zu 4096) können viele Testfälle abgedeckt werden.



Abbildung 2.9.: In-Circuite Tester: Teradyn Teststation LH Ultrapin 2. [16]

Da ein In-Circuite Testsystem bestimmte Eigenschaften der Platine voraussetzt, kann ein solches System nicht für jede Platine verwendet werden. Eine der Vielzahl an Voraussetzungen ist, dass zu testende Bauteile nicht verdeckt oder abgeschirmt sein dürfen, um mit den Federstiftkontakten erreichbar zu sein. Aufgrund der Tatsache, dass der im Unternehmen vorhandene In-Circuit-Tester kein Licht emittieren kann und das zu prüfende Platinen-Layout nicht dafür ausgelegt ist, wird eine Eigenentwicklung für die Platinenprüfung bevorzugt.

Die entwickelte Testsignalerzeugung wird im nachfolgenden Punkt erläutert.

3. Ansteuerung einer Signalerzeugungseinheit

Für dieses Projekt wurden sowohl die Software für die Kommunikation zwischen den einzelnen SPI-Busteilnehmern als auch das Modbus-Kommunikationsprotokoll implementiert. Zusätzlich wurde eine Schaltung entwickelt, um die gewünschten Prüfsignale zu erzeugen.

3.1. Dimensionierung der Schaltungsgruppen

Als Grundlage für die Entwicklung der nachfolgenden Schaltungen wurden die beiden Skripten Halbleiterschaltungstechnik [17] und analoge Signalverarbeitung [18] herangezogen.

Nachfolgendes Blockschaltbild verkörpert den Signalfluss der Schaltung. Dabei steuert der Arduino alle SPI-Busteilnehmer, welche es ermöglichen, das Signal zu erzeugen, sowie dieses zu modifizieren.

Im ersten Schritt wird das Signal des Frequenzgenerators Offset verschoben, dass anschließend den beiden Signal-Sub Stränge, oben und unten, zugeführt wird.

Im oberen Signal-Sub Strang wird das Signal invertiert, um eine Phasenverschiebung von 180° zu erzielen. Weiters wird eine variable Verstärkung erzielt, indem das Signal erneut invertiert wird. Am Ende des Signal-Sub Strangs wird erneut das Signal Offset verschoben.

Im unteren Signal-Sub Strang wird nach der Eingangs-Offsetverschiebung die Signallaufzeit angepasst, damit am Ende der Schaltung die beiden Signale exakt um 180° Phasen verschoben sind. Analog zum oberen Sub Strang wird auch hier erneut eine variable Verstärkung bewirkt, sowie am Ende eine Offsetverschiebung.

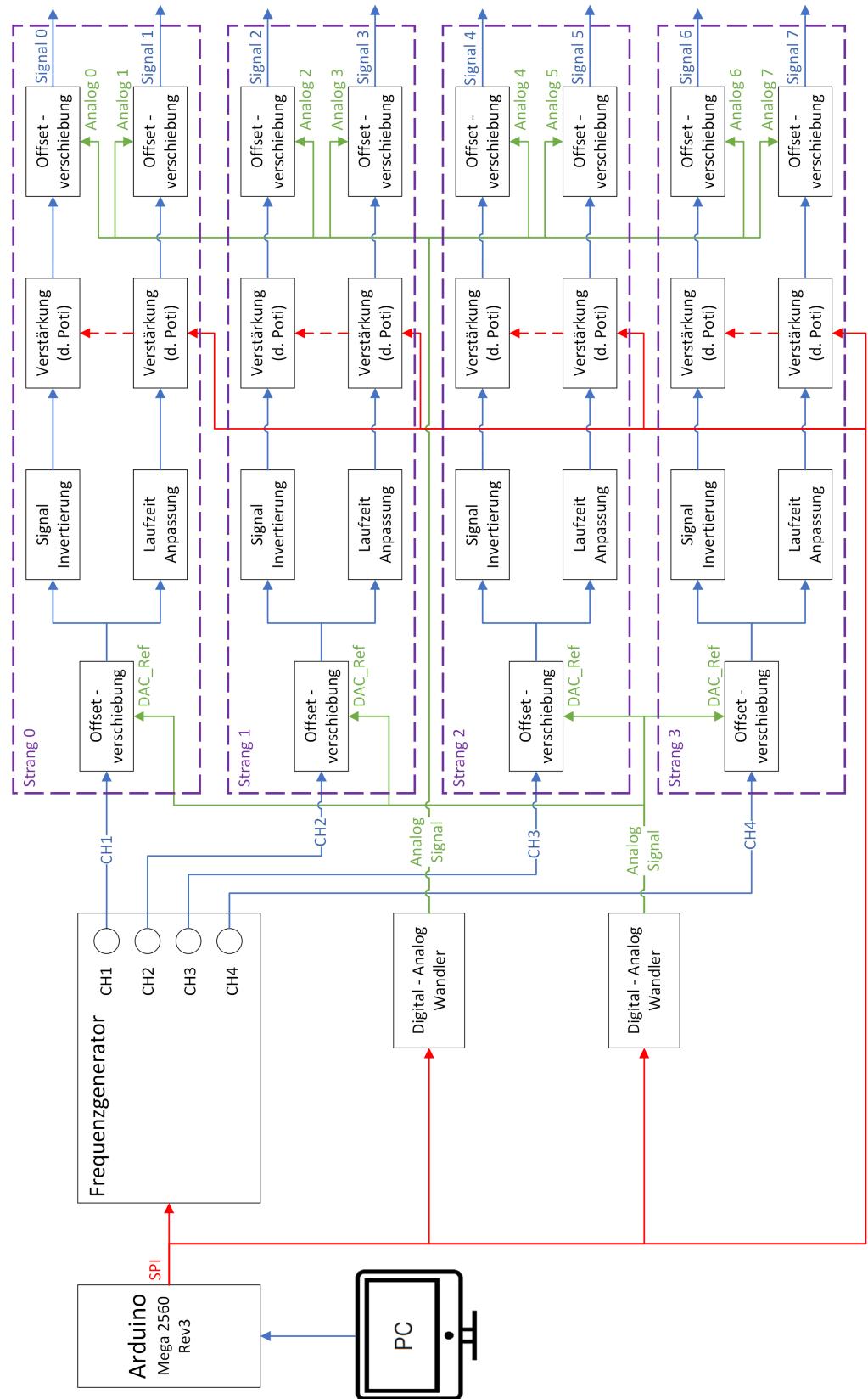


Abbildung 3.1.: Blockschaltbild der entwickelten Schaltung.

Steuerbereich der Prüfsignale

Für den Zweck der Platinenprüfung, werden lediglich Sinus-Signale mit einer maximalen Amplitude von 0,5 Volt benötigt. Dabei soll eine Frequenz bis zu 100 kHz ermöglicht sein. Die Offsetverschiebung des Signals sowie des komplementären Signals soll im Bereich von wenigen Volt in die positive Richtung möglich sein. Des Weiteren sollte es die Möglichkeit gegeben, jedes Signal individuell zu verstärken.

Berechnung der passiven Bauelemente

Für die Entwicklung dieser Schaltung wurde der OPV¹ LMH6644 verwendet. Der OPV MC33074 wurde speziell als Impedanzwandler am Signalausgang eingesetzt, da er im Vergleich zum LMH6644 in der Lage ist, große kapazitive Lasten bei einem geringeren Ausgangstrom zu bewältigen [19] [20].

In der folgenden Betrachtung werden alle Schaltungsgruppen eines Strangs beschrieben.

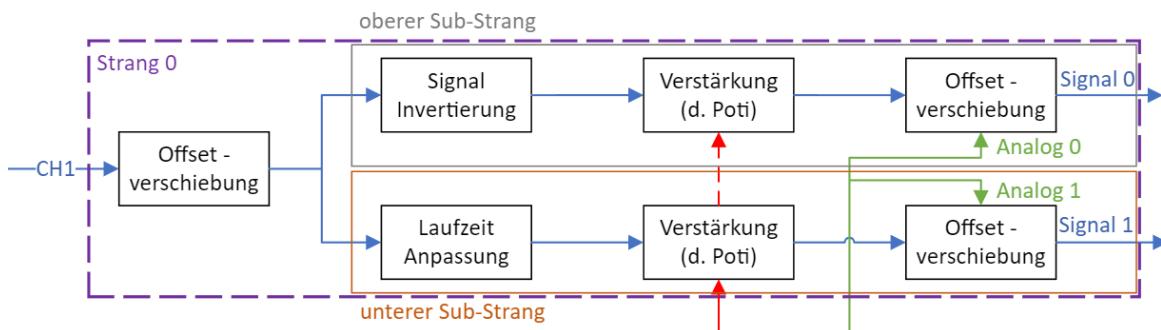


Abbildung 3.2.: Blockschaubild des ersten Strangs der Schaltung.

¹OPV: OPerationsVerstärker

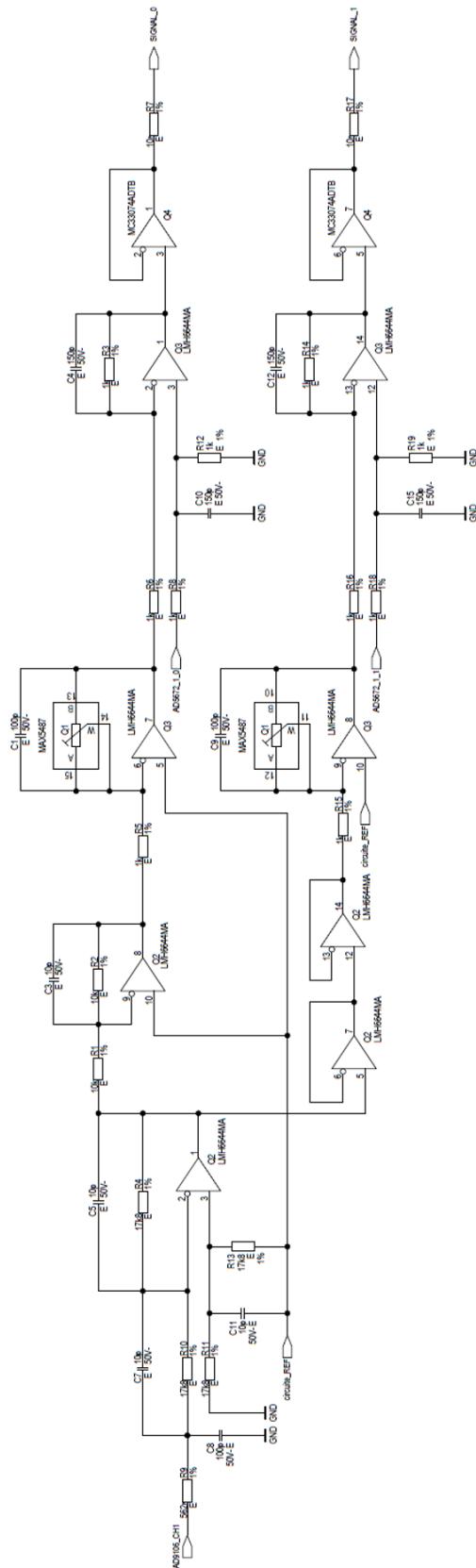


Abbildung 3.3.: Signalstrang 0 der entwickelten Schaltung.

Eingangsoffsetverschiebung

Damit am Beginn der Schaltung eine mögliche Frequenzbegrenzung durchgeführt werden kann, wurde ein passiver Tiefpass vorgesehen. Damit keine Frequenzbegrenzung auftritt wurde dieser mit den Bauteilwerten $R_9 = 562 \Omega$ und $C_8 = 100 \text{ pF}$ bestückt. Aus diesen Bauteilwerten ergibt sich folgende Grenzfrequenz:

$$f_g = \frac{1}{2 \cdot \pi \cdot R_9 \cdot C_8} = \frac{1}{2 \cdot \pi \cdot 562 \Omega \cdot 100 \text{ pF}} = 2,83 \text{ MHz} \quad (3.1)$$

Wie nachfolgend ersichtlich, wird dem passiven Tiefpass ein Differenzverstärker nachgeschalten. Da bei dieser Schaltung eine Offsetverschiebung erfolgt, wird als Referenzspannung eine analoge Spannung angelegt, welche durch einen DAC erzeugt wird.

Aufgrund dessen das keine Verstärkung durch den Differenzverstärker erzielt werden soll, wurden alle vier Widerstände $R = R_4 = R_{10} = R_{11} = R_{13} = 17,8 \text{ k}\Omega$ identisch gewählt. Aus diesen Bauteilwerten ergibt sich folgende Schlussfolgerung:

$$U_a = U_+ \cdot \frac{R_{13}}{R_{11} + R_{13}} \cdot \frac{R_{10} + R_4}{R_{10}} - U_- \cdot \frac{R_4}{R_{10}} \quad (3.2)$$

$$U_a = U_+ \cdot \frac{R}{R + R} \cdot \frac{R + R}{R} - U_- \cdot \frac{R}{R} \quad (3.3)$$

$$U_a = U_+ - U_- \quad (3.4)$$

Um die Stabilität dieser Schaltung zu gewährleisten, werden 10 pF Kondensatoren (C_5, C_{11}) parallel zu den Widerständen (R_4, R_{13}) geschaltet, um eine Tiefpasswirkung zu erzielen. Dies ist notwendig, da die gesamte Schaltung aufgrund des schnellen OPVs dazu neigt, zu schwingen.

Aus diesen Bauteilwerten ergibt sich folgende Grenzfrequenz:

$$f_g = \frac{1}{2 \cdot \pi \cdot R_4 \cdot C_5} = \frac{1}{2 \cdot \pi \cdot 17,8 \text{ k}\Omega \cdot 10 \text{ pF}} = 894,1 \text{ kHz} \quad (3.5)$$

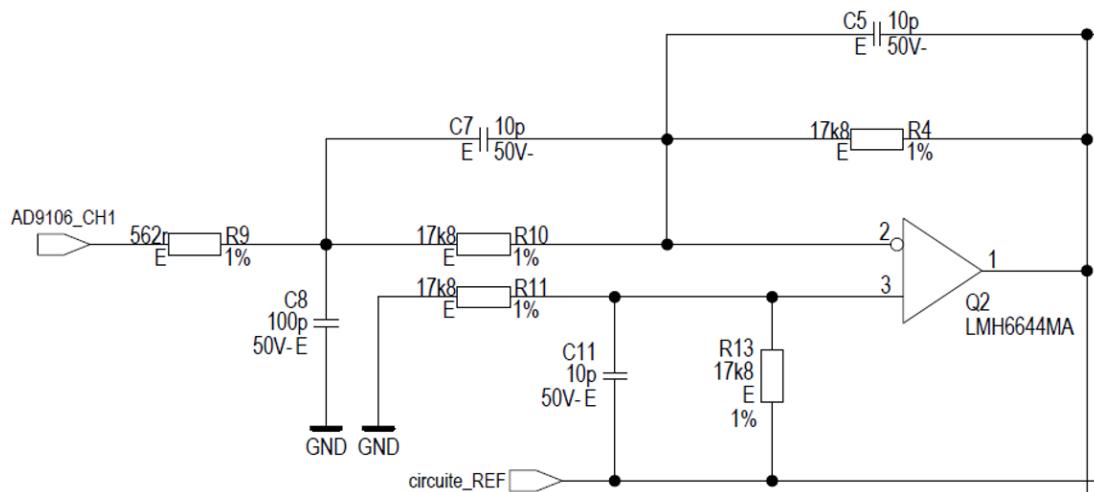


Abbildung 3.4.: Schaltungsausschnitt des Eingangsdifferenzverstärkers.

Auf die Begründung dieser Offsetverschiebung wird im Kapitel 4: Diskussion der Ergebnisse detaillierter eingegangen.

Signal Invertierung

Als erstes wurde das Signal des oberen Sub Strang invertiert. Dazu wurde eine invertierende OPV-Schaltung verwendet. Wie auch in der vorangegangenen Schaltung wird hier keine Verstärkung erzielt, wozu $R = R_1 = R_2 = 10 \text{ k}\Omega$ gewählt wurden. Weiters wurde dem Rückkoppelwiderstand ein Kondensator mit der Kapazität $C = 10 \text{ pF}$ parallel geschalten, um erneut eine Tiefpasswirkung zu erzielen.

$$U_a = V_u \cdot U_e \quad \text{mit:} \quad V_u = -\frac{R_2}{R_1} = -\frac{R}{R} = -\frac{10 \text{ k}\Omega}{10 \text{ k}\Omega} = -1 \quad (3.6)$$

$$U_a = -U_e \quad (3.7)$$

$$f_g = \frac{1}{2 \cdot \pi \cdot R_2 \cdot C_3} = \frac{1}{2 \cdot \pi \cdot 10 \text{ k}\Omega \cdot 10 \text{ pF}} = 1,59 \text{ MHz} \quad (3.8)$$

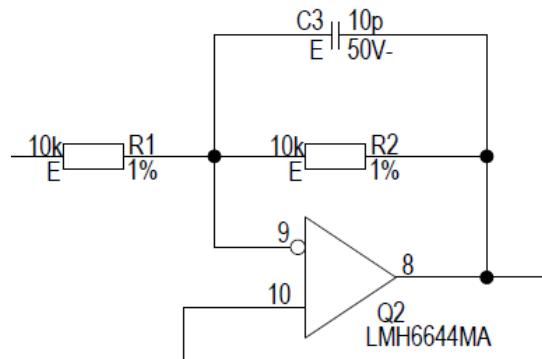


Abbildung 3.5.: Schaltungsausschnitt der Signalinvertierung.

Wie in dieser Schaltung ersichtlich, wurde als Referenz erneut die durch den DAC erzeugte Referenzspannung verwendet.

Veränderbare Verstärkung durch digitales Potentiometer

Um eine veränderbare Verstärkung zu erzielen, wird erneut eine invertierende Verstärkerschaltung verwendet. Da ein $10\text{ k}\Omega$ digitales Potentiometer verwendet wird und eine Verstärkung, welche größer sowie kleiner als minus eins werden soll, wird ein Eingangswiderstand von $1\text{ k}\Omega$ für diesen Schaltungsteil verwendet. Folgender Verstärkungsbereich kann dadurch erzielt werden:

$$V_{\max} = -\frac{R_{MAX5487}}{R_5} = -\frac{325\Omega}{1\text{ k}\Omega} = -0,325 \quad (3.9)$$

$$V_{\text{typ}} = -\frac{R_{MAX5487}}{R_5} = -\frac{1\text{ k}\Omega}{1\text{ k}\Omega} = -1 \quad (3.10)$$

$$V_{\min} = -\frac{R_{MAX5487}}{R_5} = -\frac{10325\Omega}{1\text{ k}\Omega} = -10,325 \quad (3.11)$$

Wie in allen Schaltungsteilen wird auch hier parallel zu dem Rückkoppelwiderstand ein Kondensator parallel geschalten, welcher ein Tiefpassverhalten erzeugt. Die Grenzfrequenz liegt dabei im Bereich von 154 kHz und $48,97\text{ MHz}$.

$$f_{g,\max} = \frac{1}{2 \cdot \pi \cdot R_{MAX5487} \cdot C_1} = \frac{1}{2 \cdot \pi \cdot 325\Omega \cdot 10\text{ pF}} = 48,97\text{ MHz} \quad (3.12)$$

$$f_{g,\text{typ}} = \frac{1}{2 \cdot \pi \cdot R_{MAX5487} \cdot C_1} = \frac{1}{2 \cdot \pi \cdot 1\text{ k}\Omega \cdot 10\text{ pF}} = 15,92\text{ MHz} \quad (3.13)$$

$$f_{g,\min} = \frac{1}{2 \cdot \pi \cdot R_{MAX5487} \cdot C_1} = \frac{1}{2 \cdot \pi \cdot 10325\Omega \cdot 10\text{ pF}} = 154\text{ kHz} \quad (3.14)$$

Da jedoch in den meisten Fällen eine Verstärkung von 1 gewünscht ist, welche bei einem Potentiometer-Widerstand von $1\text{ k}\Omega$ erzielt wird, beträgt die typische Grenzfrequenz 15,92 MHz.

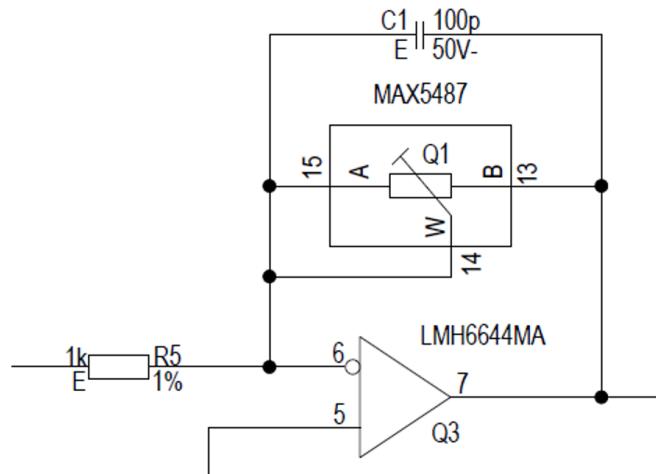


Abbildung 3.6.: Schaltungsausschnitt der variablen Verstärkung.

Wie auch in den vorangegangenen Schaltungen, wurde auch hier als Referenz erneut die durch den DAC erzeugte Referenzspannung verwendet.

Ausgangsoffsetverschiebung

Ähnlich wie bei der Eingangsoffsetverschiebung wird auch hier mit einem Differenzverstärker der Offset des Signals verschoben. Da hier keine Verstärkung des Signals realisiert werden soll, sind erneut alle Widerstandswerte gleich groß. Der Widerstandswert wurde mit $R = R_3 = R_6 = R_8 = R_{12} = 1 \text{ k}\Omega$ gewählt, wodurch sich analog zu der vorangegangenen Differenzverstärkungsberechnung eine Verstärkung von 1 ergibt.

Ebenso lässt sich die Grenzfrequenz des erzeugten Tiefpasses mit einem Kondensator von $C_4 = 150 \text{ pF}$ analog zu den vorangegangenen Berechnungen berechnen.

$$f_g = \frac{1}{2 \cdot \pi \cdot R_3 \cdot C_4} = \frac{1}{2 \cdot \pi \cdot 1 \text{ k}\Omega \cdot 150 \text{ pF}} = 1,06 \text{ MHz} \quad (3.15)$$

Um eine Neutralisierung der Eingangsoffsetverschiebung hervorzurufen, muss die angelegte analoge Spannung am nicht invertierenden Eingang, welche durch einen weiteren DAC-Kanal erzeugt wird, gleich groß der Referenz-Offsetspannung sein. Da eine negative Offsetverschiebung auftritt, wenn die analoge Spannung kleiner als die Referenz-Offsetspannung ist, darf diese Spannung niemals kleiner als die Referenzspannung sein, damit eine positive Offsetverschiebung auftritt.

$$U_{\text{off}} = U_{\text{DAC}} - U_{\text{Ref}} \quad (3.16)$$

Da die Referenzspannung mindestens den Wert des Betrags der minimalen Eingangsspannung betragen muss, bestimmt dieser Wert die maximale Offsetverschiebung in Kombination mit dem maximalen Aussteuerbereichs des DACs ($U_{\text{DAC,max}} = 5 \text{ V}$).

Maximale Offsetverschiebung anhand eines Beispiels:

Eingangssignal: Sinus mit 1 Volt Spitz-Spitze (= 0,5 V Amplitude), Offset = 0 V

Aus diesem Eingangssignal ergibt sich ein minimaler Eingangswert von -0,5 V. Daher muss die Referenzspannung auf mindestens 0,5 V ($U_{\text{Ref}} = 0,5 \text{ V}$) gesetzt werden. Aus diesen beiden Parametern kann nun der maximale Offset berechnet werden:

$$U_{\text{off,max}} = U_{\text{DAC,max}} - U_{\text{Ref}} = 5 \text{ V} - 0,5 \text{ V} = 4,5 \text{ V} \quad (3.17)$$

Da jedoch für alle vier Schaltungsstränge dieselbe Referenzspannung verwendet wird, muss darauf geachtet werden, dass U_{Ref} immer von jenem Signal abhängt, welches am negativsten wird.

Wird somit ein Signal mit einem örtlichen Minimum von -1 V eingespeist, somit muss $U_{\text{Ref}} = 1 \text{ V}$ erhöht werden, wodurch sich die maximale Offsetverschiebung auf $U_{\text{off,max}} = 4 \text{ V}$ verringert. Weiters muss darauf geachtet werden, dass alle OPVs mit einer Spannung von $\pm 5 \text{ V}$ versorgt werden und dadurch der minimale und maximale Aussteuerbereich weiter begrenzt wird.

Anschließend wird dem Differenzverstärker noch ein Impedanzwandler des Typs MC33074 nachgeschalten, welcher in der Lage ist, bei einer kapazitiven Last von bis zu 10 nF einen großen Ausgangsstrom zu treiben. Es ist dabei zu beachten, dass dieser OPV nur mit +5 V versorgt ist und dies erneut den Aussteuerbereich einschränkt. Zusätzlich wurde die Option eines seriellen Widerstands vorgesehen, welcher den Kurzschlussstrom in einem Fehlerfall begrenzen könnte.

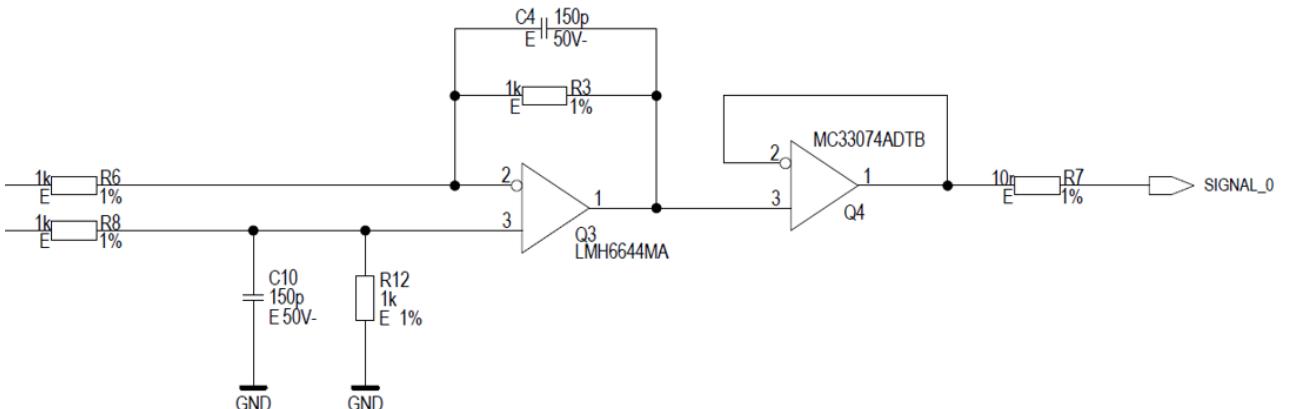


Abbildung 3.7.: Schaltungsausschnitt des Ausgangsdifferenzverstärker mit Impedanzwandlung.

Laufzeitanpassung

Da für diese Anwendung eine Phasenverschiebung von exakt 180° notwendig ist und die Relevanz der Gatterlaufzeit mit steigender Frequenz (= Periodendauer nimmt ab) zunimmt muss dieser Fall ebenfalls berücksichtigt werden.

Um dieser Tatsache entgegenzuwirken, wurden die beiden übrigen OPVs, die sich ohnehin in dessen Gehäuse eines Strangs befinden verwendet, um zwei in Serie geschaltete Impedanzwandler zu realisieren.

Diese beiden Impedanzwandler sollen jene Laufzeit kompensieren, welche durch die Invertierung des Signals im oberen Sub Strang geschieht.

Da es sich jedoch bei dem LMH6644 um einen sehr schnellen OPV mit einer großen Slew-Rate [20] handelt, ist diese Kompensation nur sehr eingeschränkt möglich.

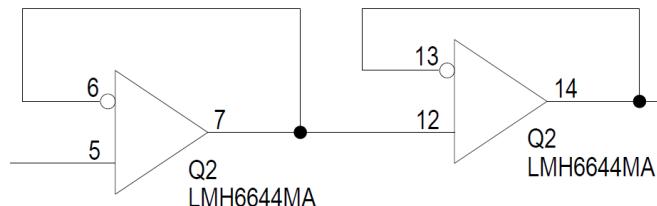


Abbildung 3.8.: Schaltungsausschnitt der Laufzeitanpassung.

Phasenverschiebungsüberblick

Wie in folgender Tabelle ersichtlich, wird im unteren Sub Strang die Phase bei der Laufzeitanpassung nicht wie in allen anderen Schaltungsgruppen um 180° gedreht, wodurch ein Phasenunterschied von 180° erzielt wird.

Tabelle 3.1.: Überblick der auftretenden Phasenverschiebung.

	Offsetverschiebung	Invertierung / Laufzeitanpassung	Verstärkung	Offsetverschiebung	Summe
Oberer Sub Strang		180°	180°	180°	$720^\circ = 0^\circ$
Unterer Sub Strang	180°	0°	180°	180°	$540^\circ = 180^\circ$

Transmissionsüberblick

Die Transmission von Tiefpassfiltern gibt Aufschluss über die Fähigkeit verschiedene Frequenzen durchzulassen. Dabei werden höhere Frequenzanteile eines Signals gedämpft, wohin gegen tieferen Frequenzen nahezu vollständig durchgelassen werden. [21]

Dabei wird das Verhältnis der Ausgangsspannung zur Eingangsspannung, in Logarithmischer Form (Dezibel) angegeben. Je höher der Transmissionswert eines Filters ist, desto besser ist seine Eigenschaft, diese bestimmte Frequenz nicht zu Dämpfen. [21]

Die Transmission geht daher einher mit den Eigenschaften der Grenzfrequenz, sowie der Steilheit und der Art des Filters.[21]

Die Transmission kann dabei wie folgt berechnet werden:

$$U_a = \frac{U_e}{\sqrt{1 + (\omega \cdot C \cdot R)^2}} [21] \quad (3.18)$$

Als Eingangsspannung für folgende Transmissions-Tabelle wurde $U_e = 1 \text{ V}$ angenommen, sowie eine Frequenz $f = 500 \text{ kHz}$.

Tabelle 3.2.: Überblick der Transmission der Schaltung.

Transmissionsparameter: $U_e = 1 \text{ V}$; $f = 500 \text{ kHz}$

	Offsetverschiebung	Invertierung / Laufzeitanpassung	Verstärkung	Offsetverschiebung	Summe
Oberer Sub Strang	0,860	0,954	0,954	0,905	0,708
Unterer Sub Strang		1	0,954	0,905	0,742

Aus dieser Berechnung geht hervor, dass die gesamte Transmission des oberen Strangs 70,8% und des unteren 74,2% beträgt bei einer Frequenz von $f = 500 \text{ kHz}$. Dabei entspricht die Transmission von 70,8% einer Dämpfung von etwa -3 dB.

Printed Circuit Board

Aus dieser Schaltung wurde nun im Unternehmen ein PCB-Layout angefertigt, welches den konzerninternen Regelungen entspricht und in dieser Erläuterung nicht weiter eingegangen wird.

Es handelt sich dabei um eine sechslagige Platine mit den Abmessungen 120 mm x 115 mm, auf der sich 316 Bauteile befinden.

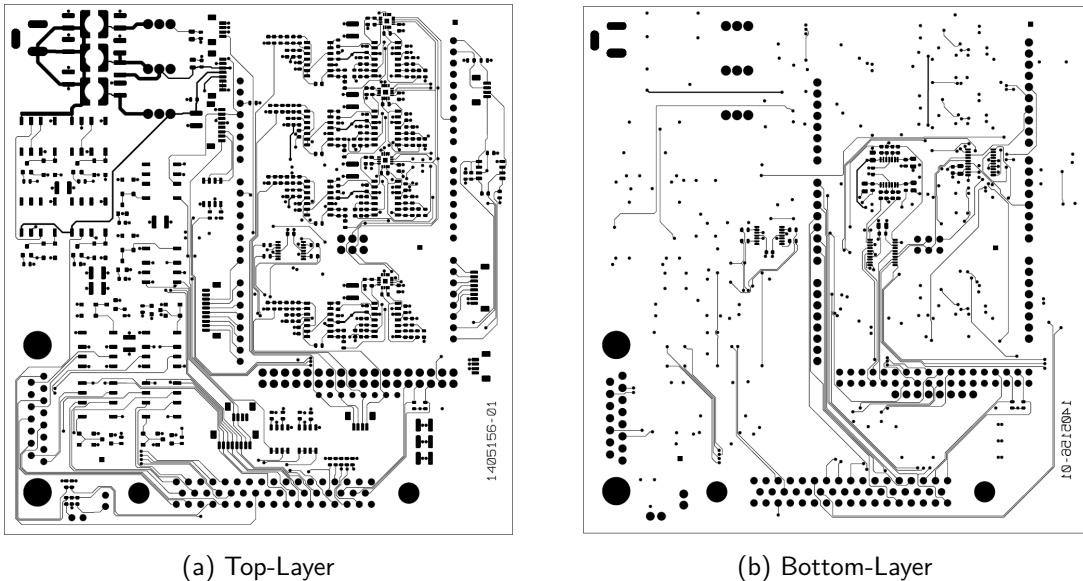


Abbildung 3.9.: Top- und Bottom-Layer der entwickelten PCB.

Die 3D-Ansicht der entwickelten Leitplatte kann in der Abbildung A.1 im Anhang betrachtet werden. Des Weiteren findet man in Abbildung A.4 die bestückte Leiterplatte.

3.2. Implementierung

Grundlagen

Als Framework wird das Arduino-Framework verwendet. Bei diesem Framework handelt es sich um eine Sammlung von Bibliotheken und Werkzeugen, die speziell für die Implementierung von Software für Arduino-Mikrokontroller-Boards entwickelt wurde.

Dieses Framework stellt eine einfache Entwicklungsumgebung [22] zur Verfügung, sowie Funktionen, die einen einfachen Zugriff auf die Hardware der Boards ermöglichen und weiters die Kommunikation mit einer externer Hardware vereinfacht [23].

Die Implementierung der Software wurde in einer Ordnerstruktur mit drei Unterordner aufgebaut: Hardware Abstraction Layer (HAL), Driver Layer (DL) und Application Layer (AL).

Dabei werden alle hardwarenahen Funktionen in separaten Dokumenten, je nach Aufgabengebiet beziehungsweise Verwendungszweck im HAL-Ordner implementiert. Alle Funktionalitäten der Software werden im Driver Layer, sowie die Algorithmen in dem Application Layer implementiert.

Dieser Aufbau der Software stellt sicher, dass die Übersicht des gesamten Codes gewahrt wird, sowie ein einfaches Austauschen von einzelnen Hardware-Modulen beziehungsweise der gesamten Hardware ermöglicht.

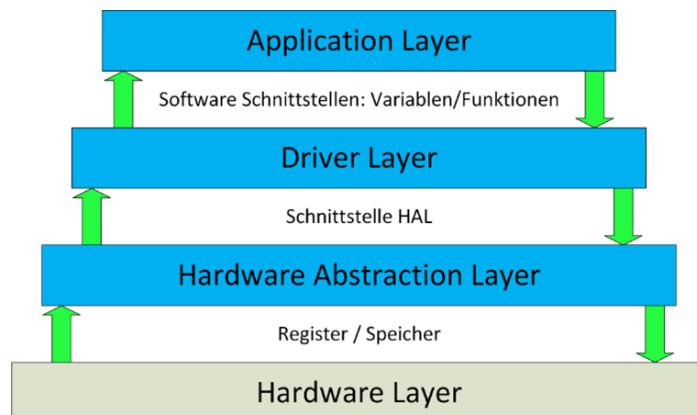


Abbildung 3.10.: Aufbau der Software.

Um den Überblick über die Vielzahl an implementierten Funktionen zu wahren, wurde für jede Funktion eine Kurzbeschreibung sowie eine minimale Beschreibung der Eingangsparameter und des Rückgabewerts hinzugefügt.

Im nachfolgenden Beispiel, wird die @brief Beschreibung an der Boundary-Check Funktion eines uint8_t Werts dargestellt. [24]

Code 3.1: Beispiel der @brief Beschreibung anhand eines boundaryChecks.[24]

```
1 /*
2  * @brief Check if value is within boundarys (uint8_t)
3  * @param val the value to check
4  * @param min lower boundary
5  * @param max high boundary
6  * @return value which is within the boundary
7 */
8 uint8_t boundaryCheck8(uint8_t val, uint8_t min, uint8_t max)
9 {
10     if (val >= max)
11         return max;
12     else if (val <= min)
13         return min;
14     else
15         return val;
16 }
```

Diese Kurzbeschreibung wird bei dem Positionieren des Mauszeigers über dem Funktionstext wie folgt dargestellt [24]:

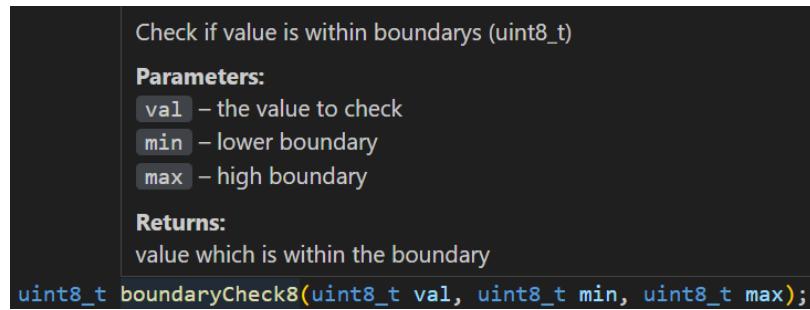


Abbildung 3.11.: Visualisierung der @brief Kurzbeschreibung in Visual Studio Code 1.77.1.[24]

Wie bereits erwähnt, wurde eine Kurzbeschreibung für alle Funktionen hinzugefügt. In den nachfolgenden Code-Beschreibungen wird jedoch auf dessen Ausführung verzichtet.

3.2.1. Verwendete GPIOs

In der nachfolgenden Tabelle sind die verwendeten GPIOs angeführt, welche für die Signalerzeugung relevant sind. Dabei sind in der zweiten Spalte die benötigten Pins für den Frequenzgenerator, in der dritten Spalte die Pins für die digitalen Potentiometer und in der letzten Spalte die GPIOs für die Digital-Analog-Wandler angeführt.

Tabelle 3.3.: Arduino GPIO Verwendung Teil 1/2.

Arduino ATmega2560	Frequenzgenerator AD9106	digitale Potentiometer MAX5487	Digitale-Analog-Wandler AD5672
GPIO 2	EN_CVDDX	-	-
GPIO 3	-	-	CS0_AD5672
GPIO 4	SDHN_N_LT3472	-	-
GPIO 5	-	-	LDAC0
GPIO 6	-	-	RESET0
GPIO 7	TRIGGER	-	-
GPIO 8	RESETB	-	-
GPIO 9	CS_AD9106	-	-
GPIO 10	-	CS0_MAX5487	-
GPIO 11	-	-	GAIN0
GPIO 12	-	CS1_MAX5487	-
GPIO 13	-	CS2_MAX5487	-
GPIO 22	-	CS3_MAX5487	-

Tabelle 3.4.: Arduino GPIO Verwendung Teil 2/2.

Arduino ATmega2560	Frequenzgenerator AD9106	digitale Potentiometer MAX5487	Digitale-Analog-Wandler AD5672
GPIO 46	-	-	GAIN1
GPIO 47	-	-	RESET1
GPIO 48	-	-	LDAC1
GPIO 49	-	-	CS1_AD5672
ICSP 1		MISO	
ICSP 3		SCK	
ICSP 4		MOSI	

Darüber hinaus wurde eine Vielzahl an General Purpose Inputs/Outputs als Reserve auf der Platine ausgeführt, welche für zusätzliche Funktionalitäten sorgen, sowie für mögliche Erweiterungen verwendet werden können. Diese Erweiterungen sowie zusätzlichen Funktion sind im Schaltplan ersichtlich.

Auf jegliche zusätzlichen Funktionen wird in dieser Arbeit jedoch nicht eingegangen.

3.2.2. Funktionserklärung

Alle nachfolgenden SPI-Übertragungsstrukturen sowie Registerfunktionen können dem jeweiligen Datenblatt des Herstellers entnommen werden. Dies gilt ebenfalls für allfällige SPI-Übertragungskommandos zum Steuern der integrierten Schaltungen. [11] [13] [15]

Implementierung des Frequenzgenerators

Der AD9106 besitzt 66 frei parametrierbare SPI-Register, welche in Abhängigkeit ihrer 16 Bit Werte, das Ausgangssignal bestimmen. Da nur ein Bruchteil der Register, welche bereits in dem Kapitel 2.2.2 verwendete Bauteile beschrieben wurden, verwendet werden, werden alle Parameter zu Beginn des Programms auf einen vordefinierten Wert gesetzt. Dazu wurden zwei Arrays angelegt, die zum einen die Registerwerte und zum anderen die Register-Adresse beinhalten.

Um Daten lesen und schreiben zu können, wurden die beiden Funktionen „AD9106readRegister“ und „AD9106writeRegister“ implementiert. Am Beginn der Datenübertragung muss der Funktionscode für das Schreiben (0x0000) oder für das Lesen (0x8000) von Werten vereinigt mit der Register-Adresse, kommuniziert werden. Bei dem Lesen von Werten müssen nachfolgend Dummy-Werte (zum Beispiel: 0x0000) kommuniziert werden, damit die gültigen Werte transferieren. Anstelle der Dummy-Werte müssen bei dem Schreiben von Registerwerten die Daten übermittelt werden. Selbstverständlich muss der Chip-Select-Pin während der Datenübertragung auf LOW gezogen werden.

Code 3.2: Implementierung der Lese- und Schreibfunktionen des AD9106.

```
1 void AD9106writeRegister(uint8_t reg, uint16_t value)
2 {
3     SELECT_AD9106
4     SPI.transfer16(WRITE + reg);
5     SPI.transfer16(value);
6     UNSELECT_AD9106
7 }
8
9 uint16_t AD9106readRegister(uint8_t reg)
10 {
11     SELECT_AD9106
12     SPI.transfer16(READ + reg);
13     uint16_t value = SPI.transfer16(DUMMY0);
14     UNSELECT_AD9106
15     return value;
16 }
```

Neben den beiden Funktionen zum Lesen und Schreiben von SPI-Registern wurden weitere Funktionen im Hardware Abstraction Layer implementiert, welche die aktuellen Registerwerte über eine serielle Schnittstelle ausgeben können, den AD9106 Hardware-Resetten kann, sowie den Hardware-Enable und -Disable des AD9106 steuern kann.

Alle weiteren Funktionen, die in Zusammenhang mit der Parametrierung des Frequenzgenerators stehen, wurden in dem Driver Layer implementiert. Bei diesen Funktionen werden die folgenden Register-Manipulationen durchgeführt:

- void AD9106enableChannel(uint8_t ch)
AD9106_regval[1] &= (0x1 « (4 - ch));

- void AD9106disableChannel(uint8_t ch)
AD9106_regval[1] |= (0x1 « (4 - ch));

- void AD9106setAmplitude(uint8_t ch, int16_t amp)
AD9106_regval[37 - ch] = (float)amp * (float)12.46513;

- ```
void AD9106setFrequency(uint32_t freq)
 uint32_t i32FrequencyTmp = freq / fDivisor;
 AD9106_regval[39] = (i32FrequencyTmp & 0xFFFF00) » 8; // frequency high Bytes
 AD9106_regval[40] = (i32FrequencyTmp & 0x0000FF) « 8; // frequency low Byte
```
  
- ```
void AD9106setPhase(uint8_t ch, uint16_t phase)
    float phaTmp = phase / phase_factor;
    AD9106_regval[45 - ch] = (uint16_t)phaTmp;
```

Zusätzlich zu diesen Manipulationen, kann der Gain-Faktor verändert werden, um einen Abgleich des Frequenzgenerators zu ermöglichen. Ebenso kann der Offset des Frequenzgenerator eingestellt werden, die jedoch für diese Implementierung nicht verwendet wurden. Damit die zuvor erläuterten Schaltung ebenfalls die Signalamplitude verändern kann.

Mit der nachfolgenden Funktion werden alle manipulierten Register übertragen. Dazu wird ein lokales Array angelegt, dass die zuvor übertragenen Daten an den AD9106 beinhaltet. Da darauf geachtet werden muss, dass keine SPI-Daten während einer Messung übertragen werden, wurde die folgende Methode ausgewählt. Dazu werden alle veränderten Daten des AD9106 auf einmal Übertragen und nicht nach jeder einzelnen Registerveränderung, wodurch die Parametrierung des Frequenzgenerators gesteuert wird.

Bei dem erstmaligen Aufrufen der Funktion (Programm start) werden alle 66 Werte übertragen, damit die zuvor festgelegte Standard-Werte einmalig zur Gänze gültig sind.

Wie in dem Code-Ausschnitt ersichtlich ist, wird zum Übertragen der Daten, die im HAL implementierte, „AD9106writeRegister“ Funktion verwendet.

Code 3.3: Implementierung der Übertragungsfunktion des AD9106.

```
1 void AD9106updateRegister(void)
2 {
3     static uint16_t AD9106_regval_local[66];
4     static bool init_finished = false;
5
6     for (int i = 0; i < 66; i++)
7     {
8         if (AD9106_regval_local[i] != AD9106_regval[i]
9             || init_finished == false)
10        {
11            AD9106_regval_local[i] = AD9106_regval[i];
12            AD9106writeRegister(reg_add[i], AD9106_regval[i]);
13        }
14    }
15    AD9106updatePatternRegister();
16
17    init_finished = true;
18 }
```

Der AD9106 verfügt ebenso über ein ERROR-Register, welches ausgelesen werden kann, sowie in einen Textfehler durch den Arduino umgewandelt werden kann. Eine weitere Funktion „AD9106clearError“ ermöglicht das Fehler-Register rückzusetzen, sobald auf den aufgetretenen Fehler reagiert wurde.

Implementierung der Digital-Analog-Wandler

Der AD5672 besitzt acht individuell steuerbare DAC-Kanäle, wobei für jeden Kanal individuell der Power-Down-Modus eingestellt werden kann. Dabei wird zwischen drei unterschiedlichen Modi unterschieden: normal, 1 kOhm und tristate. Die Betriebsart ist für diese Anwendung unwesentlich, da es sich nicht um ein energieeffizientes System handelt. Daher wird der Power-Down-Modus während der Initialisierung auf 1 kOhm eingestellt. Weiters werden alle Ausgangsspannungen auf 0 mV eingestellt, bis auf jene DAC-Spannung, welche als Referenz für die Schaltung verwendet wird.

Als Standard und nicht veränderbare Wert, wurde ein Gain von x2 eingestellt, um analoge Spannungen bis 5 V realisieren zu können. Ebenso werden bei einem Hardware-Reset alle Ausgänge auf 0 V rückgesetzt.

Vergleichbar mit der SPI-Kommunikation des AD9106 wurde hier ebenfalls eine Funktion implementiert, jene es ermöglicht die Daten in verständlicher Weise an den AD5672 zu übermitteln. Dazu wird zu Beginn der Funktion „AD5672writeDAC“ der SPI-Modus gewechselt. Nachfolgend wird das Kommando um vier Bits nach links verschoben und mit dem zu verändernden Kanal vereinigt. Analog dazu wird der in Millivolt (maximal 5000 mV) erhaltene Wert in einen 12 Bit-Wert umgerechnet und ebenfalls um vier Bits nach links geshiftet. Selbstverständlich muss der Chip-Select-Pin während der Datenübertragung auf LOW gezogen werden, wozu die CS-Pins in einem 8 Bit Array hinterlegt wurden.

Code 3.4: Implementierung der Schreibfunktionen des AD5672.

```

1 void AD5672writeDAC(bool device, uint8_t cmd, uint8_t ch, uint16_t
                      mvoltage)
2 {
3     SPI.beginTransaction(SPISettings(SPI_SPEED, MSBFIRST, SPI_MODE0));
4
5     uint8_t command = cmd << 4 | ch;
6     uint16_t data = (uint16_t)((float)mvoltage / (float)ADC12Bit);
7
8     digitalWrite(CS_AD5672[device], LOW);
9     SPI.transfer(command);
10    SPI.transfer16(data << 4);
11    digitalWrite(CS_AD5672[device], HIGH);
12
13    SPI.beginTransaction(SPISettings(SPI_SPEED, MSBFIRST, SPI_MODE3));
14 }
```

In Abhängigkeit von dem angewendeten Kommando ändert sich die Ausgangsspannung des Kanals sofort, nachdem nächsten Software-Set (Kanal individuell / gesamter Chip) oder durch einen Hardware-Set, der durch das Kippen eines Pins ausgelöst wird.

Der Vollständigkeit halber wird nachfolgend die Implementierung des Power-Down-Modus erklärt:

Die drei zuvor erwähnten möglichen Modi wurden in einem enum abgespeichert. Der Power-Down-Modus eines Kanals besteht aus zwei Bits, wodurch der Modus des gesamten Chips (acht Kanäle) in einem 16 Bit-Wert gespeichert werden kann. Zu diesem Zweck wurde ein zwei-dimensionales statisches lokales Array angelegt, indem die Power-Down-Modi einzeln abgespeichert werden, um dann zu einem einzelnen Wert vereint zu werden. Dieser Wert kann dann anschließend mit der Funktion „AD5672writePower-Down“ mit dem entsprechenden Kommando übertragen werden.

Code 3.5: Implementierung des Power-Down-Modus des AD5672.

```
1 void AD5672setPower-Down(bool device, uint8_t ch, powerMode mode)
2 {
3     static uint8_t modes[2][DAC7 + 1] = {0};
4     modes[device][ch] = mode;
5
6     for (uint8_t i = 0; i < 8; i++)
7     {
8         powerModes[device] |= (modes[device][i] << (i * 2));
9     }
10 }
11
12 void AD5672writePower-Down(bool device)
13 {
14     SPI.beginTransaction(SPISettings(SPI_SPEED, MSBFIRST, SPI_MODE0));
15
16     digitalWrite(CS_AD5672[device], LOW);
17     SPI.transfer(POWER_UP_DOWN); // command
18     SPI.transfer16(powerModes[device]);
19     digitalWrite(CS_AD5672[device], HIGH);
20
21     SPI.beginTransaction(SPISettings(SPI_SPEED, MSBFIRST, SPI_MODE3));
22 }
```

Implementierung der digitalen Potentiometer

Die verwendeten digitalen Potentiometer MAX5487 besitzen jeweils zwei individuell via SPI steuerbare Potentiometer. Während der Initialisierung der Potentiometer, werden diese auf einen vordefinierten Wert von 1 kOhm gesetzt, um eine Standardverstärkung von 1 zu realisieren.

Mit der Funktion „MAX5487writeData“ können die digitalen Daten mittels SPI übermittelt werden. Dazu wird das Kommando um vier Bits nach links geshiftet und mit dem ausgewählten Kanal, welcher mit der Kanal-Nummer 1 startet, vereinigt. Nachfolgend werden die umgerechneten Daten übermittelt. Wichtig ist dabei, dass der zugehörige Chip-Select-Pin erneut während der gesamten Übertragung auf Low gezogen wird.

Code 3.6: Implementierung der Übertragungsfunktion des MAX5487.

```

1 void MAX5487writeData(uint8_t device, uint8_t ch,
2                         uint8_t cmd, uint8_t data)
3 {
4     digitalWrite(CS_MAX5487[device], LOW);
5
6     SPI.transfer((cmd << 4) | (ch + 1));
7     SPI.transfer(data);
8
9     digitalWrite(CS_MAX5487[device], HIGH);
10}
```

Die übermittelten Daten werden in der zuvor ausgeführten Funktion „MAX5487writeResistance“ umgerechnet. Diese Funktion muss erneut das ausgewählte Device, der Kanal sowie der gewünschte Widerstandswert übermittelt werden. Dazu wird dem Widerstandswert der typische Schleifkontaktwiderstand [15] abgezogen und nachfolgend durch den Widerstandswert pro Abgriff geteilt. Der anschließend noch gerundet und der zuvor beschriebenen Funktion „MAX5487writeData“ weitergegeben wird.

Code 3.7: Implementierung der Umrechnungsfunktion des MAX5487.

```

1 void MAX5487writeResistance(uint8_t device, uint8_t ch,
2                             uint16_t resistance)
3 {
4     uint8_t data = (uint8_t)round(((float)
5                               (resistance - R_MIN) / (float)RperWiper));
6
7     MAX5487writeData(device, ch, WRITE_REGISTER, data);
8 }
```

3.2.3. Aufbau des Kommunikationsprotokoll

Wie die vorangegangenen Erläuterungen verdeutlichen, wird der Arduino Mega 2560 als Datenübersetzer zwischen einer konzerninternen Software und der SPI gesteuerten Hardware verwendet. Dabei kann das erzeugte Signal nicht nur parametriert werden, sondern ebenfalls durch das Produkt PWM 21 des Mutterunternehmens DR. JOHANNES HEIDENHAIN GmbH überprüft werden. Dieser geschlossene Regelkreis erlaubt es schnell das gewünschte, abgeglichene Signal zu erzeugen, um die Funktionalität einiger Produkte zu prüfen.

Ebenfalls können die getätigten Messungen exportiert und abgespeichert werden, was eine spätere Validierung eines durchgeföhrten Tests ermöglicht.

Modbus-Register

Wie bereits erwähnt wurde, wird für den Datenaustausch ein Modbus RTU Protokoll verwendet. Das aufgebaute Datenprotokoll besteht aus:

- 149 Coils zum Steuern und Lesen von digitalen GPIOs
- 16 Input Register, welche es ermöglichen die analogen Eingänge zu lesen
- 105 Holding Register, um das Signal zu spezifizieren und Registerwerte des AD9106 zu lesen

Die 149 Coils wurden als zusätzliche Funktionalität implementiert. Diese ermöglichen das Umschalten von potentialfreien Kontakten (Relais), Leistungsschaltern (MOSFETs) sowie zusätzlich, nicht potentialfrei, Ein- und Ausgänge zu schalten oder lesen.

Bei den 16 Input Registern handelt es sich um lesbare Register, welche den aktuellen ADC-Wert der analogen Eingänge beinhalten. Dabei beträgt die Referenzspannung der 10 Bit ADCs 5 V. Bei diesen 16 Registern handelt es sich ebenfalls um eine zusätzliche Funktionalität.

Bei den Holding Registern 1 bis 39 handelt es sich um Register, welche es ermöglichen die Schaltung zu Parametrieren und zu Steuern. Die wichtigsten 39 Holding Register des Modbus RTU Kommunikationsprotokolls sind im Anhang in den Tabellen Tabelle A.1 bis Tabelle A.3 abgebildet. Die weiteren 66 Register beinhalten den Wert des AD9106-Registers.

3.3. Projektumsetzungs Ergebnisse

Um die Schaltungsergebnisse nicht nur quantitativ, sondern ebenso qualitativ zu ermitteln, wurde ein Bode Diagramm, von jedem einzelnen Signal aufgenommen. Dazu wurde ein Kanal nach dem anderen aktiviert. Die Referenzspannung der Schaltung wurde auf 1,0 V gesetzt und jeder individuelle DAC-Offset auf 2,5 V, wodurch laut Formel 3.16 ein Signaloffset von 1,5 V resultiert. Des Weiteren wurde das digitale Potentiometer auf 1 k Ω gesetzt, wodurch sich eine Verstärkung von 1 ergibt. Wie in der nachfolgenden Abbildung ersichtlich ist, wurde dazu am ersten Kanal des Oszilloskops der direkte Ausgang des Frequenzgenerators angeschlossen. Dieses Eingangssignal wird dabei als Referenz für die Messung der Phasenverschiebung verwendet. Auf dem zweiten beziehungsweise dritten Kanal des Oszilloskops wurden jeweils die beiden Signale eines Strangs angeschlossen. Mit diesen Parametern wurde nun die Frequenz verändert, wodurch sich ein Bode Diagramm ergibt.

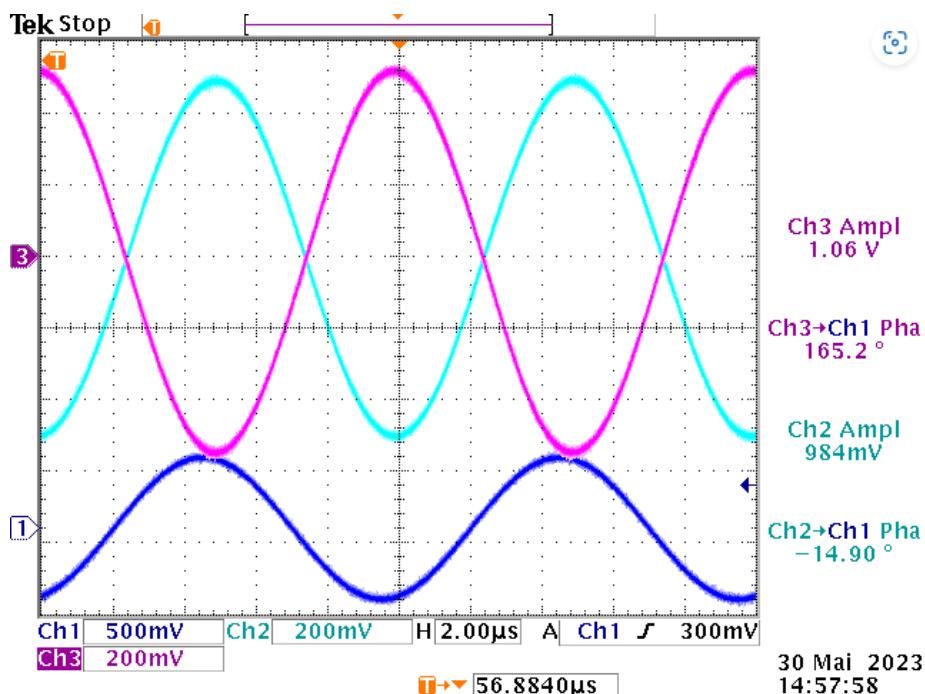


Abbildung 3.12.: Zeitsignal des vierten Kanals, bei einer Frequenz von $f = 100$ kHz,
Messung der Tabelle A.7, Ch1 dunkelblau = Eingangssignal,
Ch2 hellblau = Singal 6, Ch3 pink = Signal 7.

Bode Diagramm: Kanal 0 und Simulation

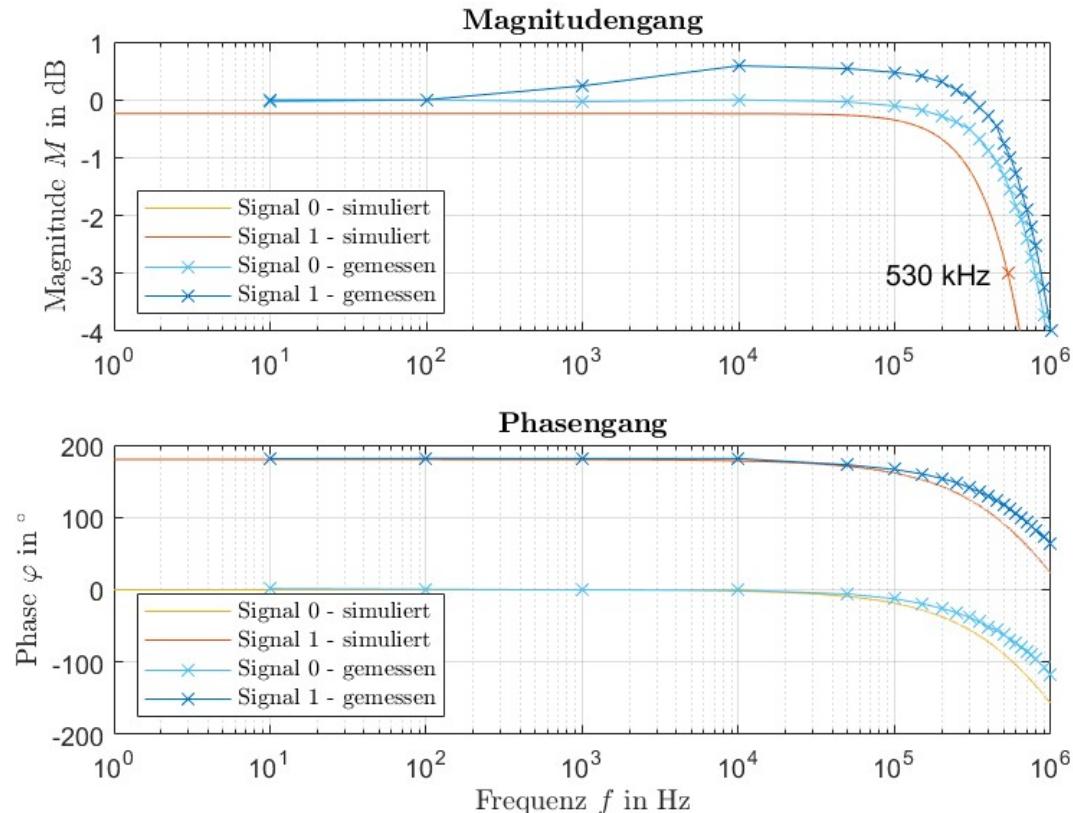


Abbildung 3.13.: Vergleich des Bode Diagramm der Simulation sowie der Messung des Kanals 0.

Wie der Legende dieses Bode Diagramms entnommen werden kann, wurde hier die simulierten Schaltungsergebnisse mit den beiden gemessenen Signalen Null und Eins verglichen.

In diesem Bode Diagramm wird ersichtlich, dass sich in der Simulation mit Orcard Capture CIS 17.4 die beiden simulierten Signale in der Magnitude nicht unterscheiden. Des Weiteren wird in der Phase ersichtlich, dass die beiden simulierten Signale eine ungefähre Phasenverschiebung von 180° aufweisen. Bei den beiden simulierten Signalen ist eine geringe konstante Dämpfung zu sehen. Die beiden gemessenen Kurven weisen dagegen keine Dämpfung auf. Ebenso kann beobachtet werden, dass jenes Signal, welches eine um 180° höhere Gesamt-Phasenverschiebung erfährt, etwas verstärkt wird. Darüber hinaus tritt die Grenzfrequenz bei den gemessenen Signalen, deutlich später als bei den simulierten Signalen auf. In Bezug auf die Phase sind

die simulierten also auch die gemessenen Signale nahezu identisch bis zu einer Frequenz von 100 kHz. Über diese Frequenz hinaus tritt auch hier die Phasendämpfung der gemessenen Signale deutlich später ein.

Da jedoch in einem Bode Diagramm der Phasenunterschied der beiden Kanäle nicht beobachtet werden kann, wurde die nachfolgende Grafik angefertigt.

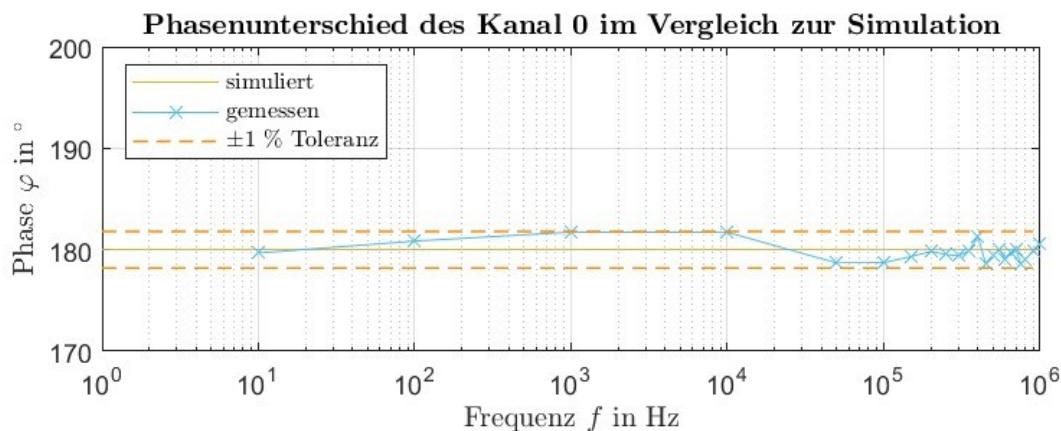


Abbildung 3.14.: Vergleich der Phasenverschiebung der Simulation sowie der Messung des ersten Kanals.

In dieser Grafik kann der Phasenunterschied der beiden simulierten Signale sowie jene Phasenverschiebung der gemessenen Signale beobachtet werden. Dabei ist zu beobachten, dass nur eine geringfügige Abweichung der Phase auftritt.

Um nicht nur den ersten Kanal in einem Bode Diagramm mit der Simulation zu vergleichen, wurde ebenso ein Bode Diagramm mit den genannten Parametern aller verbleibenden Signalen aufgenommen.

Bode Diagramm der gemessenen Signale

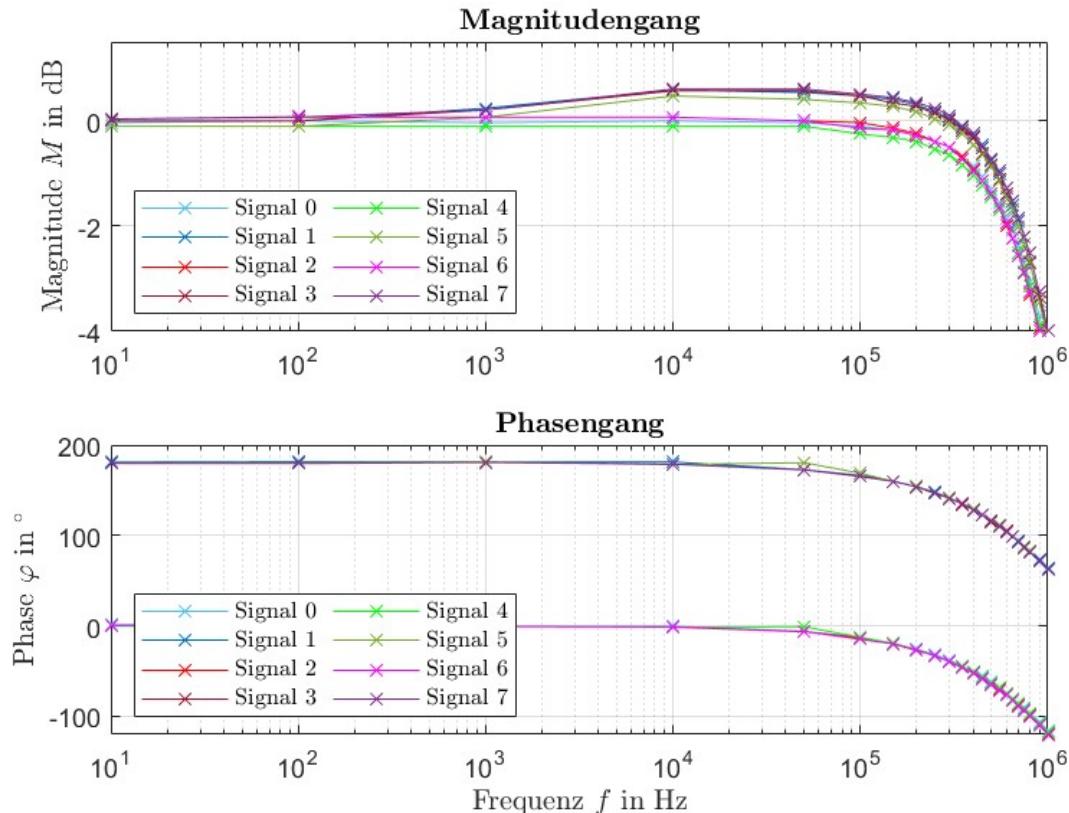


Abbildung 3.15.: Bode Diagramm aller gemessenen Signale.

In diesem Bild wird deutlich, dass sich alle vier Kanäle identisch verhalten. Dabei tritt eine -3 dB Dämpfung bei einer Grenzfrequenz von 750 kHz für alle Signale, welche durch den unteren Sub Strang der Schaltung erzeugt werden, auf. Jene Signale, die durch den oberen Sub Strang erzeugt werden, erfahren jedoch erst bei einer erhöhten Grenzfrequenz von 850 kHz eine Dämpfung von -3 dB. Der Phasengang des Diagramms zeigt ebenfalls, dass sich alle Sub Stränge identisch verhalten.

Erneut wurde die Phasendifferenz zwischen jedem Signal eines Strangs berechnet und visualisiert.

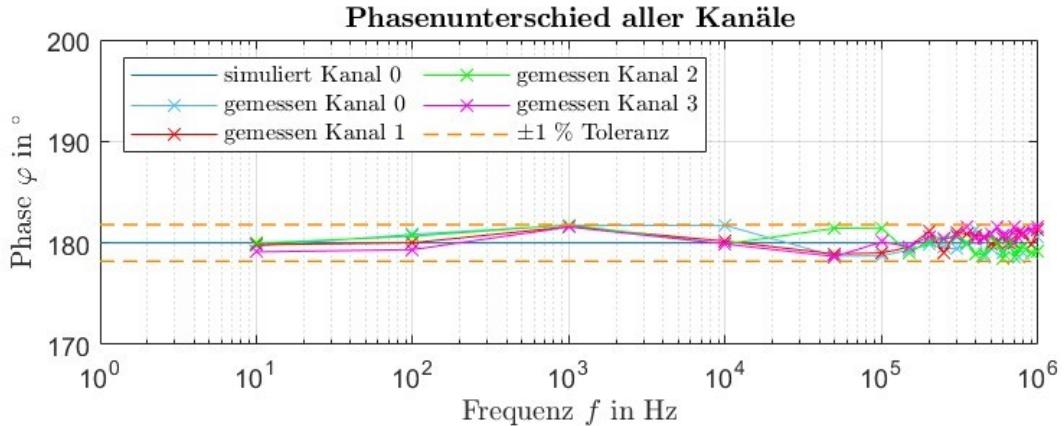


Abbildung 3.16.: Vergleich der Phasenverschiebung von allen Signalen.

Wie dieses Bild zeigt, erfahren alle Kanäle eine annähernd konstante Phasenverschiebung von 180° .

Wie die letzten beiden Abbildungen belegen, wurden die Schaltungsziele der Forschungsfrage nicht nur erreicht, sondern übertroffen. Die gewünschte Grenzfrequenz von 500 kHz wurde um 250 kHz (+50%) übertroffen und es wird eine Phasenverschiebung mit einer maximalen Abweichung von $\pm 1\%$ bis zu einer Frequenz von 1 MHz gewährleistet.

Zur einfacheren Visualisierung wurden ebenso die beiden Linien eingezeichnet welche die $\pm 1\%$ Marke darstellen.

Wie in der nachfolgenden Abbildung ersichtlich ist, kann auch die Verstärkung sowie der Offset jedes einzelnen Signals individuell und während der Laufzeit verändert werden.

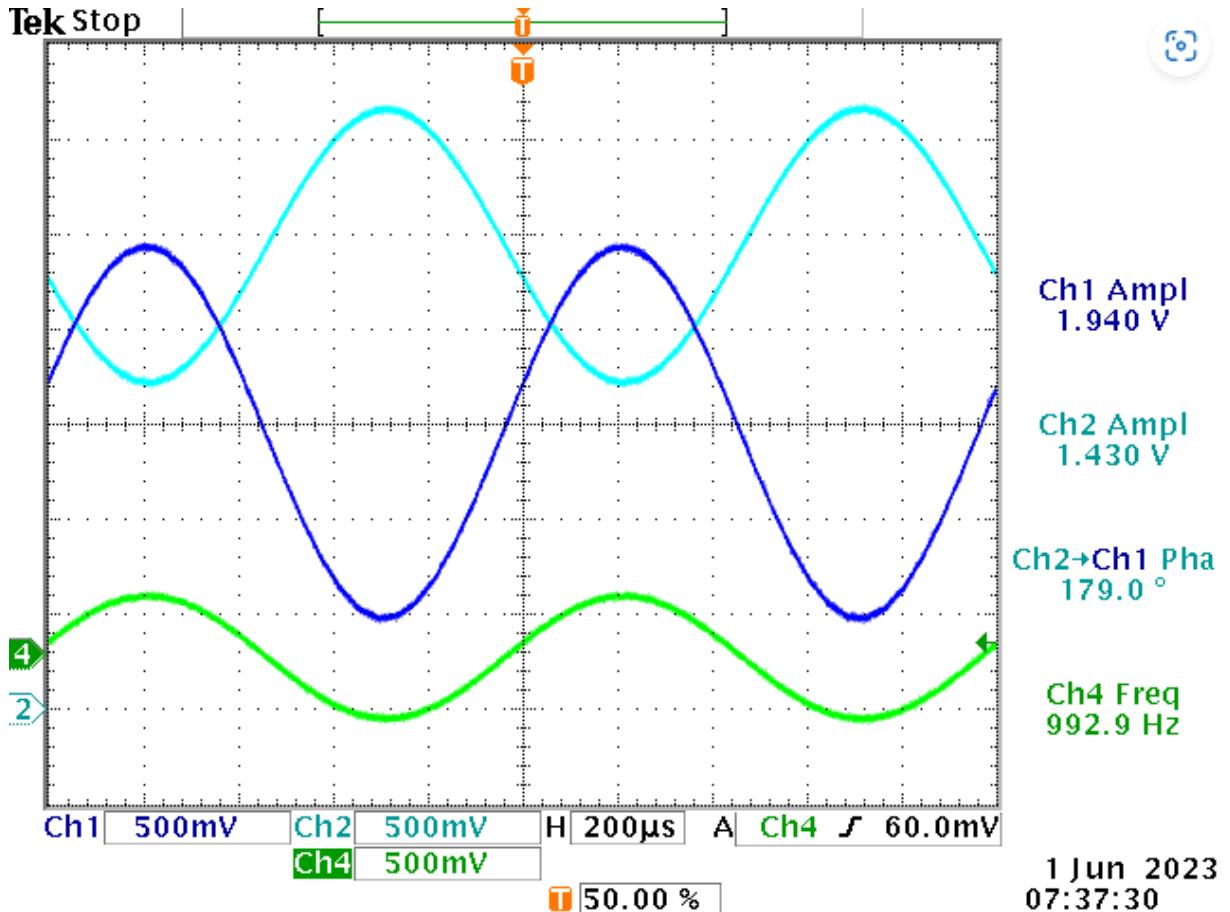


Abbildung 3.17.: Individuelle veränderbare Offset und Verstärkung.

Im Hinblick auf die Software-Ziele kann ebenso eine positive Bilanz gezogen werden. Jegliche relevanten einstellbaren Parameter des Frequenzgenerators können mittels dem Modbus RTU Protokoll übertragen werden. Darüber hinaus kann auch die DACs sowie die digitalen Potentiometer parametrisiert werden, um die gewünschten Modifizierungen des Signals hervorzurufen. Die eingestellten Parameter können gleichermaßen gelesen werden.

4. Diskussion der Ergebnisse

Um die im Kapitel 3.3 Umsetzungsergebnisse beschriebenen Resultate zu erzielen, wurde eine kleine Änderung in der Bestückung vorgenommen, da das gemessene Bode Diagramm nicht den Erwartungen entsprach.

Für die nachfolgenden Beobachtungen wurden die im Kapitel 3.3 Umsetzungsergebnisse beschriebenen Testparameter verwendet, damit die unterschiedlichen Diagramme einfacher miteinander verglichen werden können.

Das ursprünglich aufgenommene Bode Diagramm ist in der nachfolgenden Abbildung ersichtlich.

Bode Diagramm: Kanal 0 & Simulation - vor Optim.

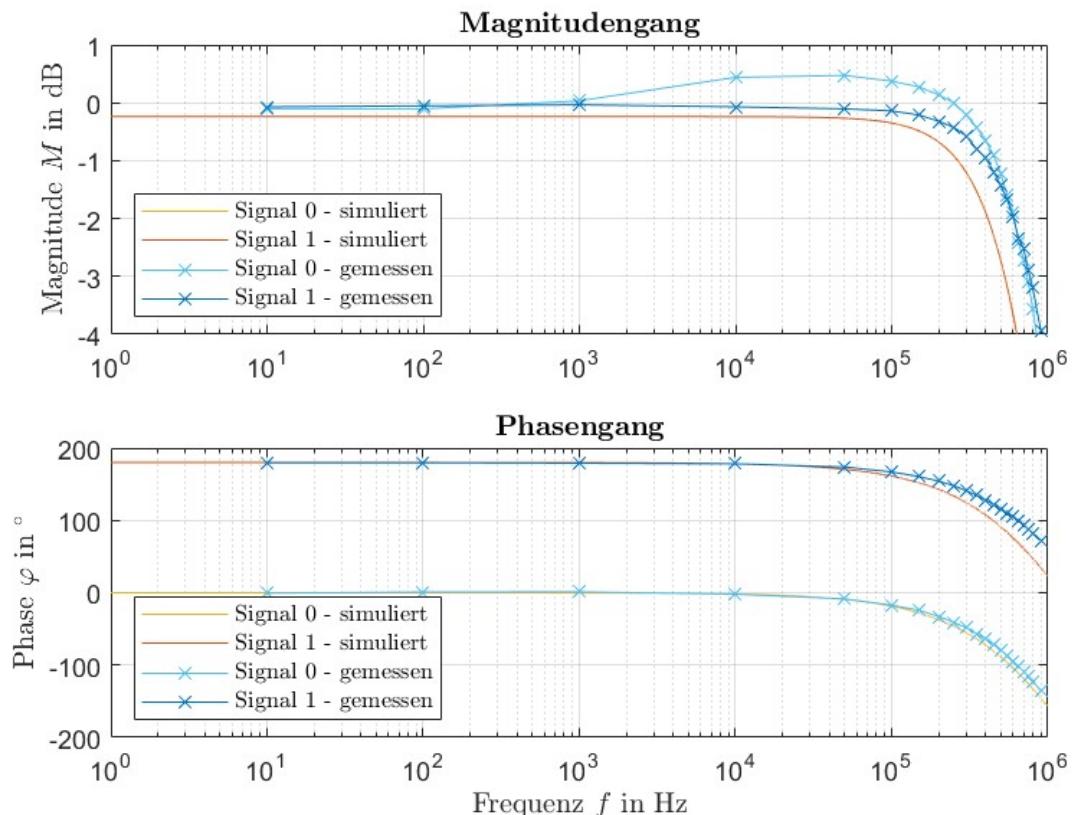


Abbildung 4.1.: Bode Diagramm des Kanal 0 und der Simulation vor der Optimierung.
Messdaten in Tabelle A.8

In diesem Bode Diagramm sind erneut die beiden Signale des ersten Stangs der Messung sowie der Simulation dargestellt.

Im ersten Blick erscheint kein Unterschied zwischen diesem sowie jenem Bode Diagramm, welches im Abschnitt 3.3 Umsetzungsergebnisse beschrieben wurde. Die Magnitude der beiden Diagramme ist nahezu identisch. Beobachten wir nun den Phasengang bei einer erhöhten Frequenz von über 100 kHz, so ist ersichtlich, dass die dunkelblaue Linie (Signal 1) immer weiter von dem simulierten Verhalten abweicht, wodurch ebenfalls die Phasenabweichung von 180° größer wird. Dieses Verhalten kann verdeutlicht werden, wenn man das nachfolgende Diagramm des Phasenunterschieds betrachtet.

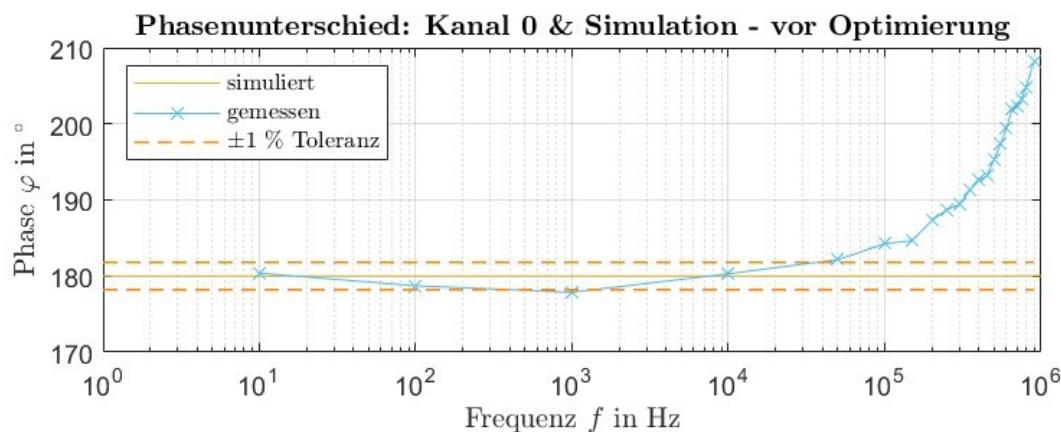


Abbildung 4.2.: Phasenunterschied des Kanal 0 und der Simulation vor der Optimierung.
Messdaten in Tabelle A.8

In diesem Diagramm wird nun deutlich sichtbar, dass bereits bei einer Frequenz von 40 kHz die $\pm 1\%$ Toleranzmarke überschritten wird, wodurch das Signal unbrauchbar für derartige Tests wird.

Um diesem Verhalten auf den Grund zu gehen, wurde ein Bode Diagramm an den beiden Pins 8 und 14 des OPVs Q2 der Schaltung, welche in der Abbildung 3.3 abgebildet ist, aufgenommen.

Bode Diagramm: Kanal 0 & Simulation - Teilschaltung

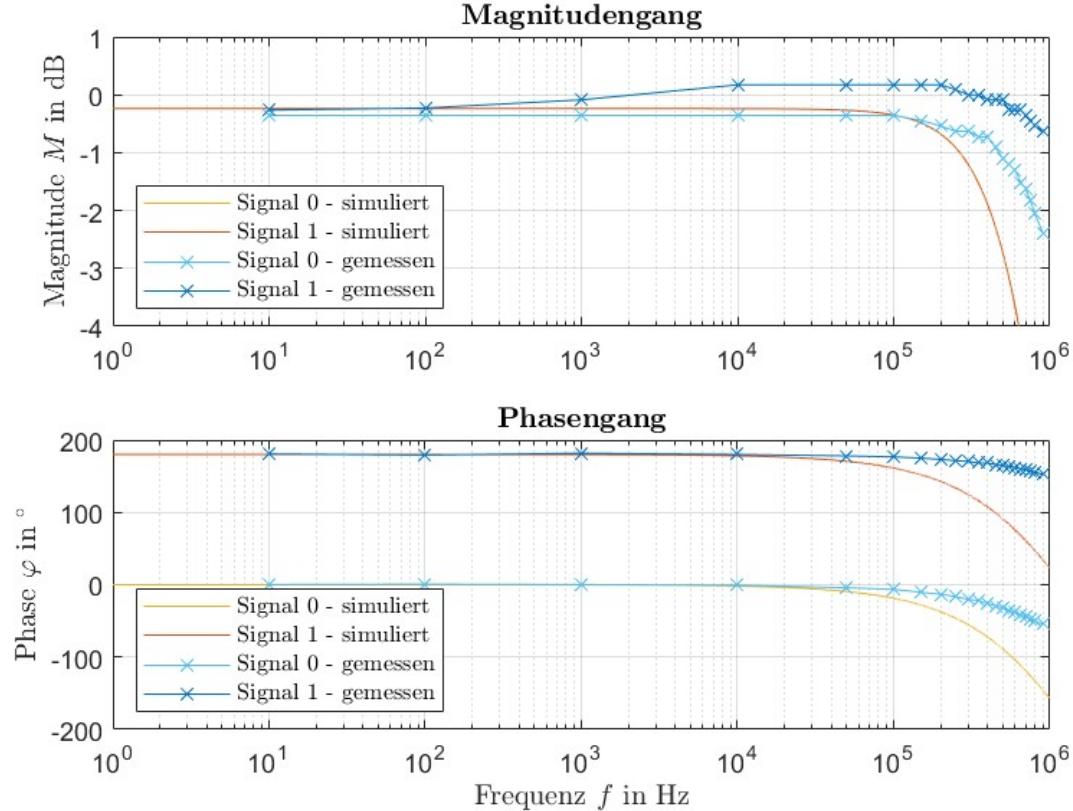


Abbildung 4.3.: Bode Diagramm der Teilschaltung des Kanal 0 und der Simulation.
Messdaten in Tabelle A.9

Dieses Bode Diagramm zeigt nun eine deutlich erhöhte Grenzfrequenz, sowie einen bemerkenswerten späteren Einbruch der Phasengänge. Dies kann dadurch begründet werden, da einige Tiefpässe der nachfolgenden OPV-Schaltungen nicht durchlaufen werden. Das Phasenunterschieds Diagramm gibt auch hier detaillierten Aufschluss über den Phasenunterschied der beiden Signale.

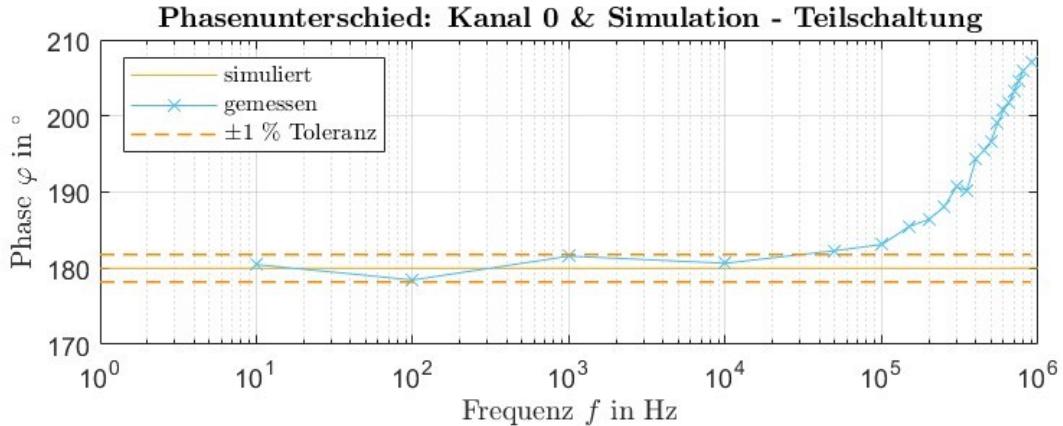


Abbildung 4.4.: Phasenunterschied der Teilschaltung des Kanal 0 und der Simulation.
Messdaten in Tabelle A.9

Wie in diesem, sowie dem vorangegangenen Phasenunterschieds Diagramm beobachtet werden kann, tritt in beiden Diagrammen ein ähnliches Verhalten auf. Dies lässt nun Schlussfolgern, dass diese unterschiedliche Phasenbeeinträchtigung durch den invertierenden Verstärker des oberen Sub Strangs auftritt. Um diesen Vorwurf zu beweisen, wurde die in Abbildung 3.3 abgebildete Schaltung der Kondensator C3 entfernt, wodurch kein Tiefpassverhalten in dieser Operationsverstärkerstufe hervorgerufen wird. Nach dieser Änderung wurde erneut ein Bode Diagramm dieser Schaltung erstellt, welches keine Phasenbeeinträchtigung aufweist. Die oben genannte Veränderung wurde anschließend bei den verbleibenden drei Kanälen durchgeführt. Diese Bode Diagramme sowie dessen Erläuterungen wurde bereits im vorherigen Kapitel 3.3 Umsetzungsergebnisse behandelt.

Um der Verstärkung des Komplementären des ursprünglichen Signals auf den Grund zu gehen, wurden die beiden OPVs Q2 und Q3 durch einen MC34074 (selber Baustein wie MC33074, jedoch andere Bauform) ausgetauscht. Jene Veränderung erzielte jedoch nicht das gewünschte Ergebnis, wodurch keine weitere Optimierung erzielt werden konnte.

Bei der Erörterung dieser unbekannten Verstärkung des Signals, muss darauf geachtet werden, dass die im Rückkoppelpfad der variablen Verstärkerstufe befindenden digitalen Potentiometer eine große Toleranz aufweisen. Diese beträchtlichen Abweichungen könnten ihren Ursprung

in der äußerst ungenauen Einstellung eines genauen Widerstands von $1\text{ k}\Omega$ liegen. Wodurch jene Verstärkung stark schwanken kann. Um die Widerstandstoleranzen zu betrachten, wurde während der Inbetriebnahme die Widerstandsgrenzen der einzelnen Potentiometer aufgenommen.

Tabelle 4.1.: Messdaten der digitalen Potentiometer (MAX5487).

		Kanal 0		Kanal 1	
		min	max	min	max
Q1	ePoti 0 in Ω	284	10350	282	10350
Q5	ePoti 1 in Ω	288	10500	283	10460
Q8	ePoti 2 in Ω	274	9850	273	9850
Q12	ePoti 3 in Ω	287	10510	286	10430

Diese Tabelle zeigt, dass sich jene Potentiometer, welche sich in einem gemeinsamen Gehäuse befinden, kaum eine Toleranz aufweisen. Betrachtet man nun das ePoti 2, so kann man feststellen, dass diese deutlich von den anderen drei Potentiometern abweicht.

Jene Unstimmigkeiten werden dabei nicht im implementierten Code beachtet, wodurch es zu Verstärkungstoleranzen in den verschiedenen Strängen kommen kann. Diese Toleranz treten jedoch nicht innerhalb eines Strangs auf.

Zur Begründung der Eingangsoffsetverschiebung kann gesagt werden, dass diese aus dem Grund der Lieferbarkeit des digitalen Potentiometers entstanden ist. Am Anfang dieses Projektes war geplant ein anderes digitales Potentiometer zu verwenden. Das ursprüngliche Potentiometer wäre in der Lage gewesen, negative Spannungen an dessen Widerstand auszuhalten. Da dieses eine sehr lange Lieferzeit mit sich brachte, wurde kurzerhand eine Alternative gesucht: MAX5487. Das Problem ist, dass die Alternative nur mit einer Versorgungsspannung von +5 V versorgt wird und somit keine negativen Spannungen verträgt.

Für den Einsatz in einer derartigen Verstärkerstufe, ist es jedoch notwendig negative Spannungen an den invertierenden Eingang des OPV zurückzuspeisen. Um den MAX5487 für dieses Projekt verwenden zu können, wurde die Eingangsoffsetverschiebung um eine konstante Spannung eingeführt, wodurch es möglich wurde, alle Spannungen so anzuheben, dass das digitale Potentiometer keine negativen Spannungen sieht. Dazu wurde ebenfalls die Bezugsspannung aller OPV-Schaltungen um die Eingangsoffsetspannung angehoben.

Im Bezug auf die Implementierungsergebnisse ist es wichtig zu verstehen, dass die verwendete ModbusRTUSlave [8] Library lediglich den Datenempfang des Arduinos übernimmt. Wie im Kapitel 2.1.2 Modbus erläutert wird, besteht jegliche Art von Datenaustausch aus einer Anfrage und einer Antwort. Da die Library nicht in der Lage ist, Antworten auf eine Frage zu übermitteln, muss diese zusätzlich implementiert werden. Die erwartete Antwort auf unterschiedliche Anfragen hängt dabei nicht nur von der Menge an zu übertragenden Daten ab, sondern ebenfalls von dem Funktions-Code des empfangenen Datenpackets. Die zu erwartende Antwort wurde dem Onlinebeitrag 'Modbus RTU made simple with detailed descriptions and examples' [7] entnommen und dementsprechend implementiert. Ein Teil dieser Übertragungsdaten ist eine CRC Summe. Für diesen Zweck wurde eine Funktion erstellt, welche in der Lage ist, die Checksumme der zu sendenden Daten zu berechnen und dem Datenpacket anzuhängen, um diese Daten dann gesammelt zu übertragen.

Der Arduino Mega 2560 Rev.3 muss mittels einem USB-Kabel mit einem Personal Computer zum Steuern der Modbus-Register verbunden und die entwickelte Schaltung mit einer externen Spannungsversorgung versorgt werden (erhöhte Stromaufnahme der Schaltung möglich). Daher ist es von Nöten, dass die Sicherung F1 des Arduino Entwicklungsboards entfernt wird, welche den Stromfluss zwischen Arduino und der USB-Schnittstelle verhindert. Der Arduino wird dadurch zur Gänze durch die externe Spannungsversorgung versorgt.

An jedem Signalstrang-Ausgang befindet sich ein Impedanzwandler realisiert mit dem OPV MC33074, welcher dafür Sorgt, dass größere kapazitive Lasten betrieben werden können. Dieser OPV ist lediglich mit +5 V versorgt. Der MC33074 besitzt jedoch keine Rail-to-Rail Unterstützung, wodurch sowohl die positive als auch die negative Aussteuergrenzen stark ausgeprägt ist. Das zugehörige Datenblatt gibt an, dass der Ausgangssteuerbereich bei dieser Versorgung typischerweise zwischen 0,1 V und 4,0 V [19] liegt.

Um dieses Problem zu lösen, könnte ein alternativer OPV verwendet werden oder durch Kurzschließen der Pins 2 & 3, 5 & 6, 9 & 10 sowie 12 & 13 diese Operationsverstärkerstufe übersprungen werden.

5. Fazit und Ausblick

Wie im Kapitel 3.3 Umsetzungsergebnisse bereits erläutert und diskutiert, ist das Projekt ein großer Erfolg.

Es wurde nicht nur eine Schaltung entwickelt, welche in der Lage ist, ein Signal digital zu verstärken sowie Offset zu verschieben, sondern ebenfalls die selben Operationen mit dem erzeugten komplementären Signal durchzuführen. Diese Testsignalerzeugung ist auf vier individuellen Kanälen möglich.

Das implementierte Modbus RTU Kommunikationsprotokoll ermöglicht es, die folgenden Sinus-Signalparameter jedes einzelnen Kanals individuell zu verändern. Dabei kann jedes der vier, durch den Frequenzgenerator erzeugte Signal in Amplitude und Phasenlage verändert werden und jedes der acht Signale individuell verstärkt, sowie Offset verschoben werden. Die eingestellte Frequenz des Frequenzgenerators ist dabei für alle Kanäle gültig.

Darüber hinaus befinden sich auf der Platine weitere Funktionsbausteine, welche vom Unternehmen RSF Elektronik Ges.m.b.H. geplant wurden. Diese Teilschaltungen können der Schaltung im Anhang A entnommen werden.

Nachdem Prototypen für den Test beziehungsweise dem Beweisen der Funktionalität von Schaltungsstrukturen gedacht sind, müssen diese zu meist in einer weiteren Version optimiert werden. Für eine Weiterführung dieses Projekts sollten folgende Verbesserungsforschläge beachtet werden:

Bei den SPI-Kommunikationsleitungen soll darauf geachtet werden, dass diese nicht neben analogen Signalen auf der Platine geführt werden, da es durch die steilen Flanken der MISO, MOSI sowie Chip-Select Leitungen zu Störungen durch Singalübersprechen kommen kann.

Zumal die Aussteuergrenze des MC33074 Operationsverstärker bei +5 V Spannungsversorgung sehr eingeschränkt ist, wäre es vorteilhaft, wenn dieser in Zukunft mit $V_{EE} = -5$ V und $V_{CC} = +12$ V versorgt wird. Alternativ könnte auch ein anderer Operationsverstärker verwendet werden, welcher in der Lage ist kapazitive Lasten zu treiben.

Im Bezug auf das Layout sollte darauf geachtet werden, dass VIAs nicht zu nahe an anderen Bauteilen plaziert werden, da dies das Bestücken per Hand deutlich vereinfachen würde. Des Weiteren ist bei dem Erstellen des Layouts der Platine ein kleiner Layoutfehler geschehen, welcher jedoch durch die Software behoben werden konnte. Dabei wurden die beiden Reihen des 2x18 poligen Pin headers vertauscht.

Da sich die konzipierte analoge Schaltung, unter dem Frequenzgenerator befindet, welche die erzeugten Signale manipuliert, können Messungen nur sehr schwer durchgeführt werden. Daher wäre es sinnvoll, einige Testpunkte einzuführen, welche einen leichteren Zugang ermöglichen.

Hinsichtlich der Implementierung des Modbus RTU Kommunikationsprotokoll zu vereinfachen, könnte eine andere Modbus Library verwendet werden, die das Senden von Antworten unterstützt. Alternativ könnte auch die verwendete ModbusRTUSlave Library erweitert werden, damit diese ebenfalls das Senden von Daten unterstützt.

Bei der aktuellen Implementierung können nur einzelne Register übertragen oder abgefragt werden. Daher wäre es Sinnvoll, die im Modbus vorgesehenen Funktions-Codes zu verwenden, welche es erlauben würden mehrere Coils/Register während einer Übertragung zu übertragen. Durch diese Erweiterung würde der Datendurchsatz erheblich erhöht werden.

Weiterführende Dokumente

Alle weiterführenden Dokumente können in diesem [GitHub repository](#) gefunden werden.

In diesem [GitHub repository](#) können die folgenden Inhalte gefunden werden:



- Implementierter Arduino Code
- Kommunikationsprotokoll
- Schaltplan
- Bestückungsplan
- PCB Gerber files
- Verwendetes Bildmaterial
- Datenblätter

Link: https://github.com/F1schi/bachelor_thesis_at_RSFElektronik

Literaturverzeichnis

- [1] RSF Elektronik Ges.m.b.H., „RSF Elektronik Ges.m.b.H.” 2023. [Online]. Verfügbar: <https://www.rsf.at/> [Zugriff am 10.04.2023].
- [2] Electronics Notes, „ICT, In Circuit Test Tutorial,” 2023. [Online]. Verfügbar: <https://www.electronics-notes.com/articles/test-methods/automatic-automated-test-ate/ict-in-circuit-test-what-is-primer.php> [Zugriff am 24.05.2023].
- [3] Microchip Technology Inc., „ATmega640/V-1280/V-1281/V-2560/V-2561/V, 8-bit Microcontroller with 16/32/64KB In-System Programmable Flash,” 2014. [Online]. Verfügbar: <https://ww1.microchip.com/downloads/aemDocuments/documents/OTH/ProductDocuments/DataSheets/ATmega640-1280-1281-2560-2561-Datasheet-DS40002211A.pdf>
- [4] Arduino SA, „Arduino Reference,” 2023. [Online]. Verfügbar: <https://www.arduino.cc/reference/en/> [Zugriff am 24.05.2023].
- [5] A. Merothund und P. Sora, *Sensornetzwerke in Theorie und Praxis: Embedded Systems-Projekte erfolgreich realisieren*, 2. Aufl. Wiesbaden: Springer Vieweg, 2021.
- [6] H. Siebeneicher, „Arduino® & Modbus Protocol,” 2023. [Online]. Verfügbar: <https://docs.arduino.cc/learn/communication/modbus> [Zugriff am 13.04.2023].
- [7] IPC2U, „Modbus RTU made simple with detailed descriptions and examples,” 2023. [Online]. Verfügbar: <https://ipc2u.com/articles/knowledge-base/modbus-rtu-made-simple-with-detailed-descriptions-and-examples> [Zugriff am 20.04.2023].
- [8] CMB27, „ModbusRTUSlave: A library for Arduino enabling Modbus RTU communication,” 2019. [Online]. Verfügbar: <https://github.com/CMB27/ModbusRTUSlave> [Zugriff am 15.04.2023].
- [9] Arduino SA, „Arduino Mega 2560 Rev3,” 2021. [Online]. Verfügbar: <https://store.arduino.cc/products/arduino-mega-2560-rev3> [Zugriff am 24.05.2023].

- [10] Analog Devices, „EVAL-AD9106 Evaluation Board,” 2023. [Online]. Verfügbar: <https://www.analog.com/en/design-center/evaluation-hardware-and-software/evaluation-boards-kits/EVAL-AD9106.html> [Zugriff am 22.02.2023].
- [11] Analog Devices, Inc., *AD9106, Quad, Low Power, 12-Bit, 180 MSPS, Digital-to-Analog Converter and Waveform Generator Rev. B*, 2021. [Online]. Verfügbar: <https://www.analog.com/media/en/technical-documentation/data-sheets/ad9106.pdf>
- [12] Mouser Electronics, „AD5672RBRUZ Analog Devices,” 2023. [Online]. Verfügbar: <https://www.mouser.at/ProductDetail/Analog-Devices/AD5672RBRUZ?qs=wFnXgzJ2EGOH3mpx6kyZwg%3D%3D> [Zugriff am 17.05.2023].
- [13] I. Analog Devices, *AD5672R/AD5676R, Octal, 12-/16-Bit nanoDAC+ with 2 ppm/°C Reference, SPI Interface Rev. E*, 2021. [Online]. Verfügbar: https://www.analog.com/media/en/technical-documentation/data-sheets/ad5672r_5676r.pdf
- [14] Mouser Electronics, „MAX5487ETE+T Analog Devices / Maxim Integrated,” 2023. [Online]. Verfügbar: <https://www.mouser.at/ProductDetail/Analog-Devices-Maxim-Integrated/MAX5487ETE%2bT?qs=CDqwynd4ZNo8zcGwMBTiMA%3D%3D> [Zugriff am 17.05.2023].
- [15] I. Maxim Integrated Products, *MAX5487-MAX5489, Dual, 256-Tap, Nonvolatile, SPI-Interface, Linear-Taper Digital Potentiometers Rev. 4*, 2010. [Online]. Verfügbar: <https://www.analog.com/media/en/technical-documentation/data-sheets/MAX5487-MAX5489.pdf>
- [16] Teradyne Inc., „TestStation LH,” 2023. [Online]. Verfügbar: <https://www.teradyne.com/teststation-lh/#1588270837137-3401cb90-06b4> [Zugriff am 23.05.2023].
- [17] R. Okorn, „Halbleiterschaltungstechnik (V3.7),” Graz: Vorlesungsskript FH JOANNEUM, 2021.
- [18] R. Okorn, „Analoge Signalverarbeitung (Rev. 2.10),” Graz: Vorlesungsskript FH JOANNEUM, 2021.
- [19] L. Semiconductor Components Industries, *MC34071,2,4,A; MC33071,2,4,A; NCV33072,4,A, Single Supply 3.0 V to 44 V Operational Amplifiers Rev. 24*, 2014. [Online]. Verfügbar: <https://www.onsemi.com/download/data-sheet/pdf/mc34071-d.pdf>

- [20] T. I. Incorporated, *LMH6642, LMH6643, LMH6644, LMH664x Low Power, 130 MHz, 75 mA Rail-to-Rail Output Amplifiers Rev. P*, 2013. [Online]. Verfügbar: https://www.ti.com/lit/ds/symlink/lmh6644.pdf?ts=1686060133338&ref_url=https%253A%252F%252Fwww.ti.com%252Fproduct%252Fde-de%252FLMH6644
- [21] Wikimedia Foundation Inc., „Tieypass,” 2023. [Online]. Verfügbar: <https://de.wikipedia.org/wiki/Tieypass> [Zugriff am 01.06.2023].
- [22] Arduino SA, „Software Arduino IDE,” 2023. [Online]. Verfügbar: <https://www.arduino.cc/en/software> [Zugriff am 23.03.2023].
- [23] Arduino SA, „Arduino CLI,” 2020. [Online]. Verfügbar: <https://arduino.github.io/arduino-cli/0.32/> [Zugriff am 21.03.2023].
- [24] Doxygen, „Documenting the code,” 2023. [Online]. Verfügbar: <https://www.doxygen.nl/manual/docblocks.html> [Zugriff am 14.04.2023].

A. Anhang

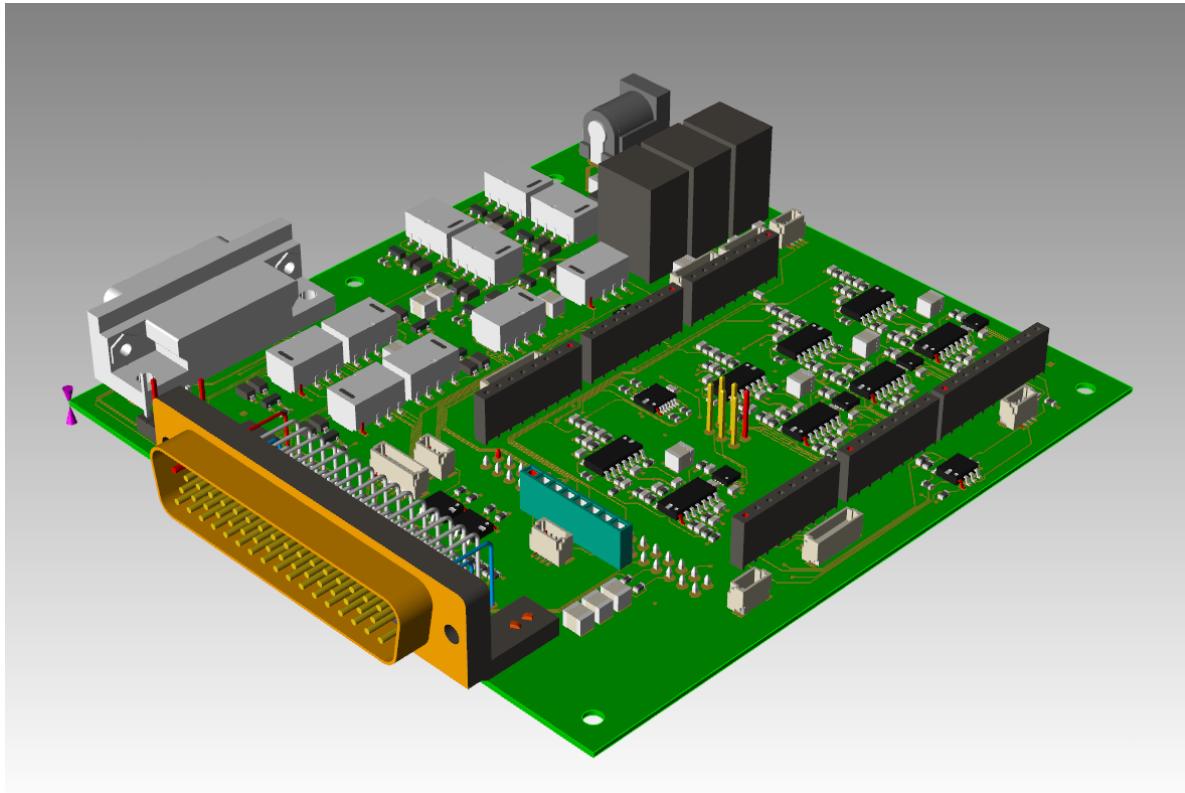


Abbildung A.1.: PCB 3D view.

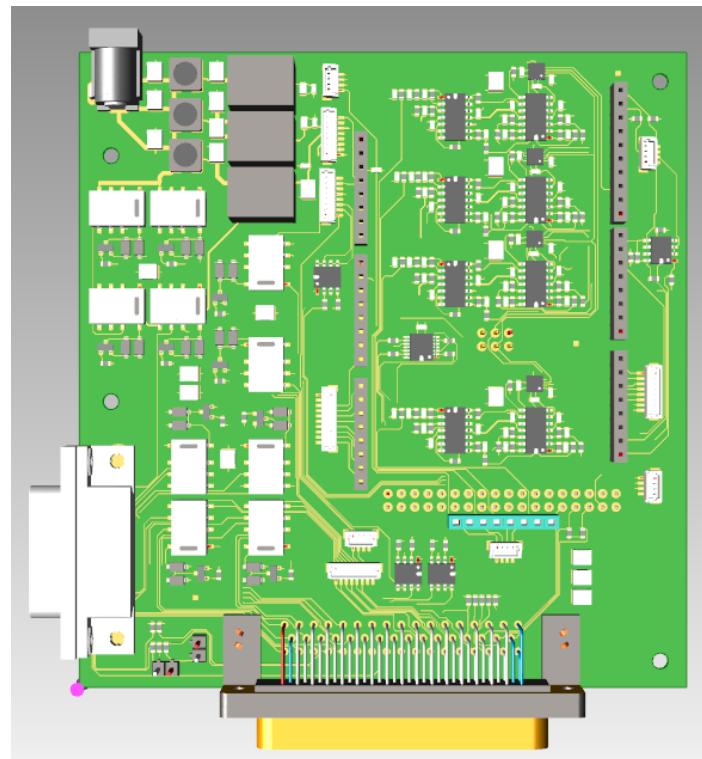


Abbildung A.2.: PCB top view.

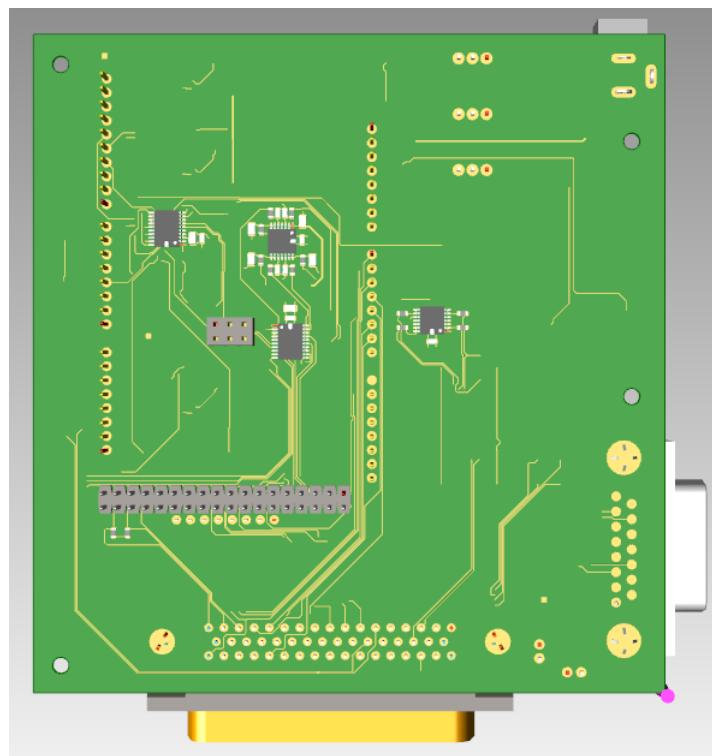


Abbildung A.3.: PCB bottom view.

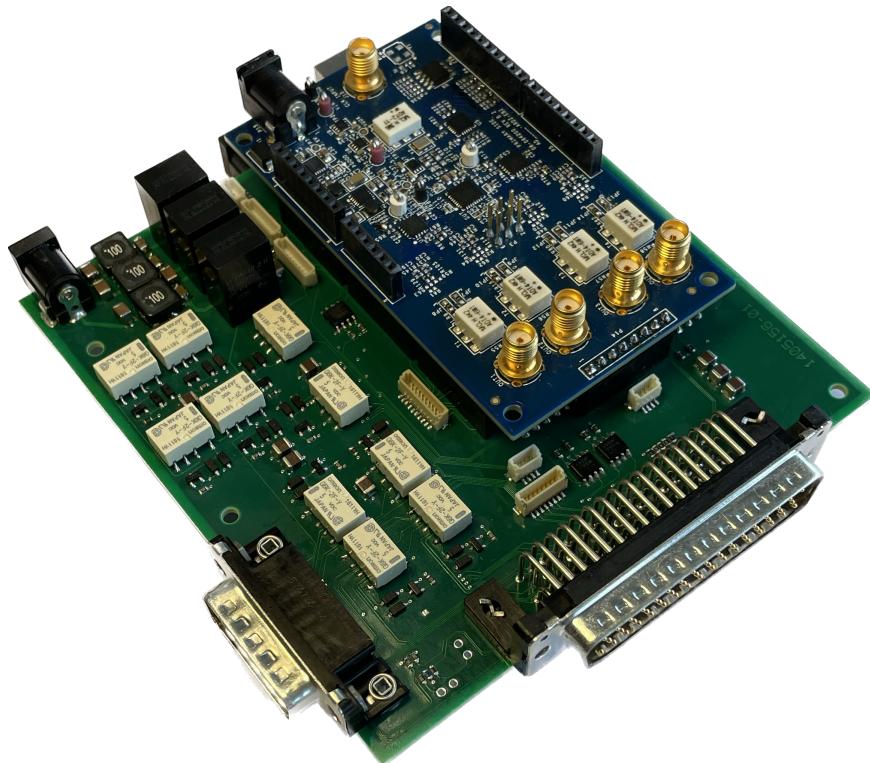
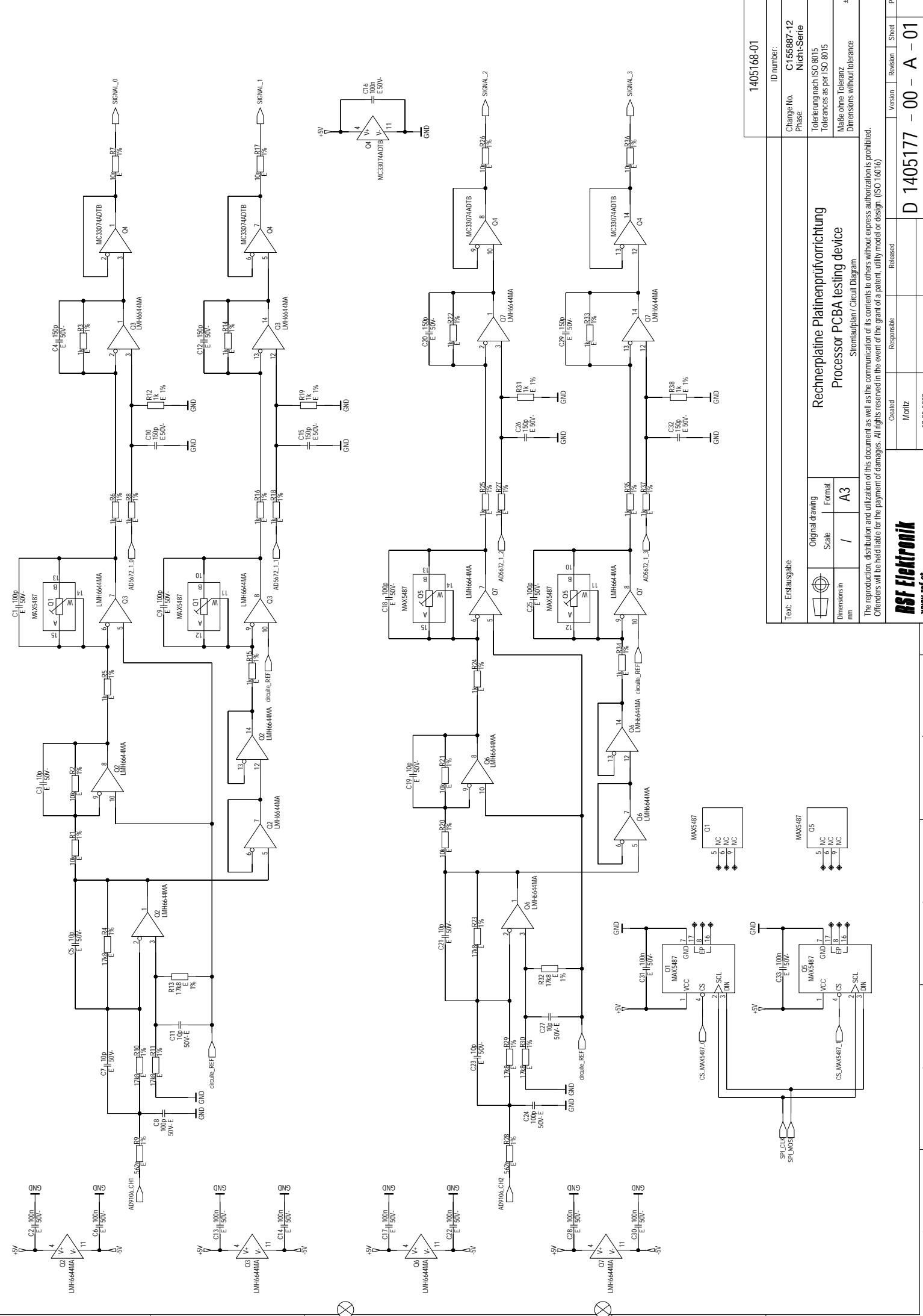
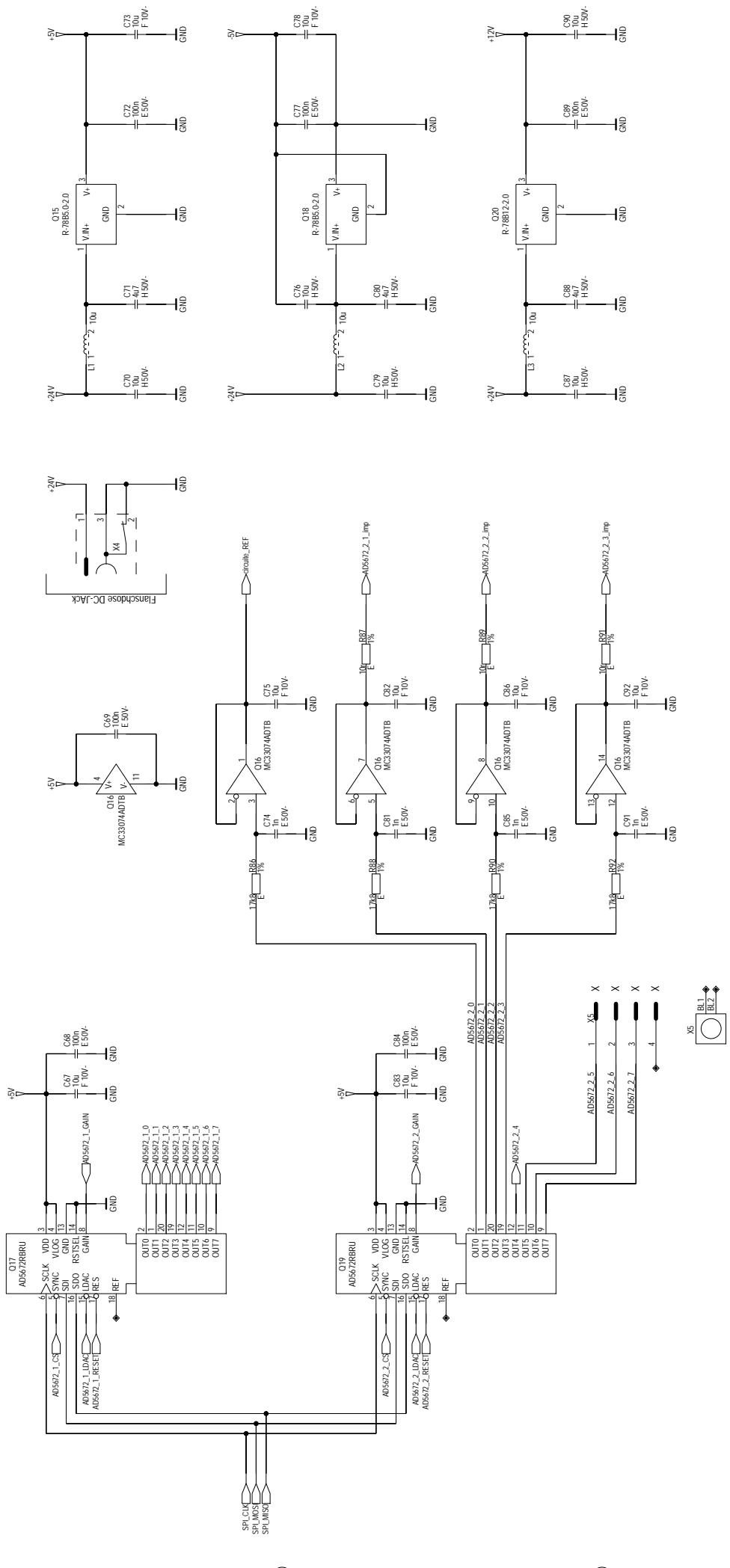
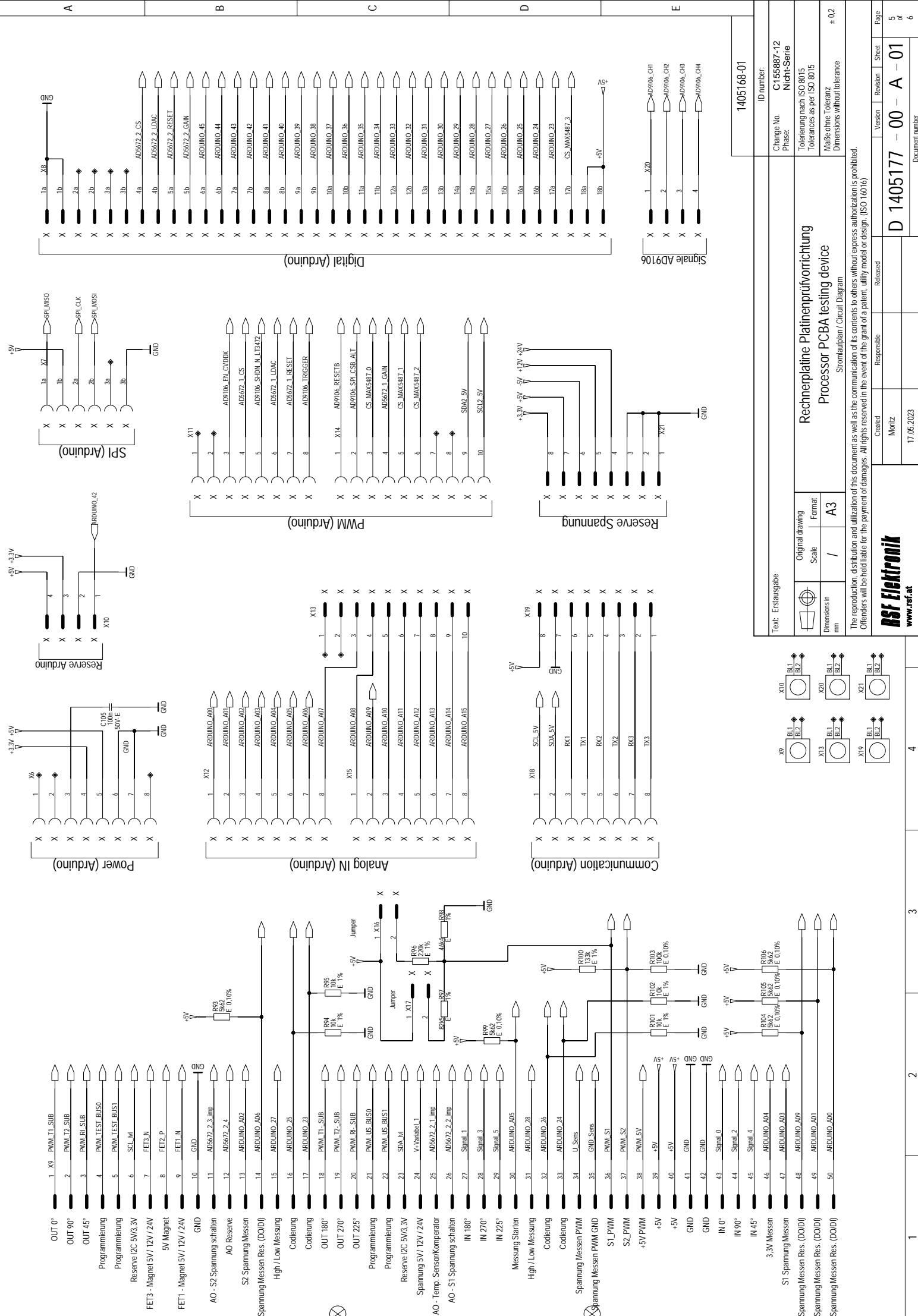
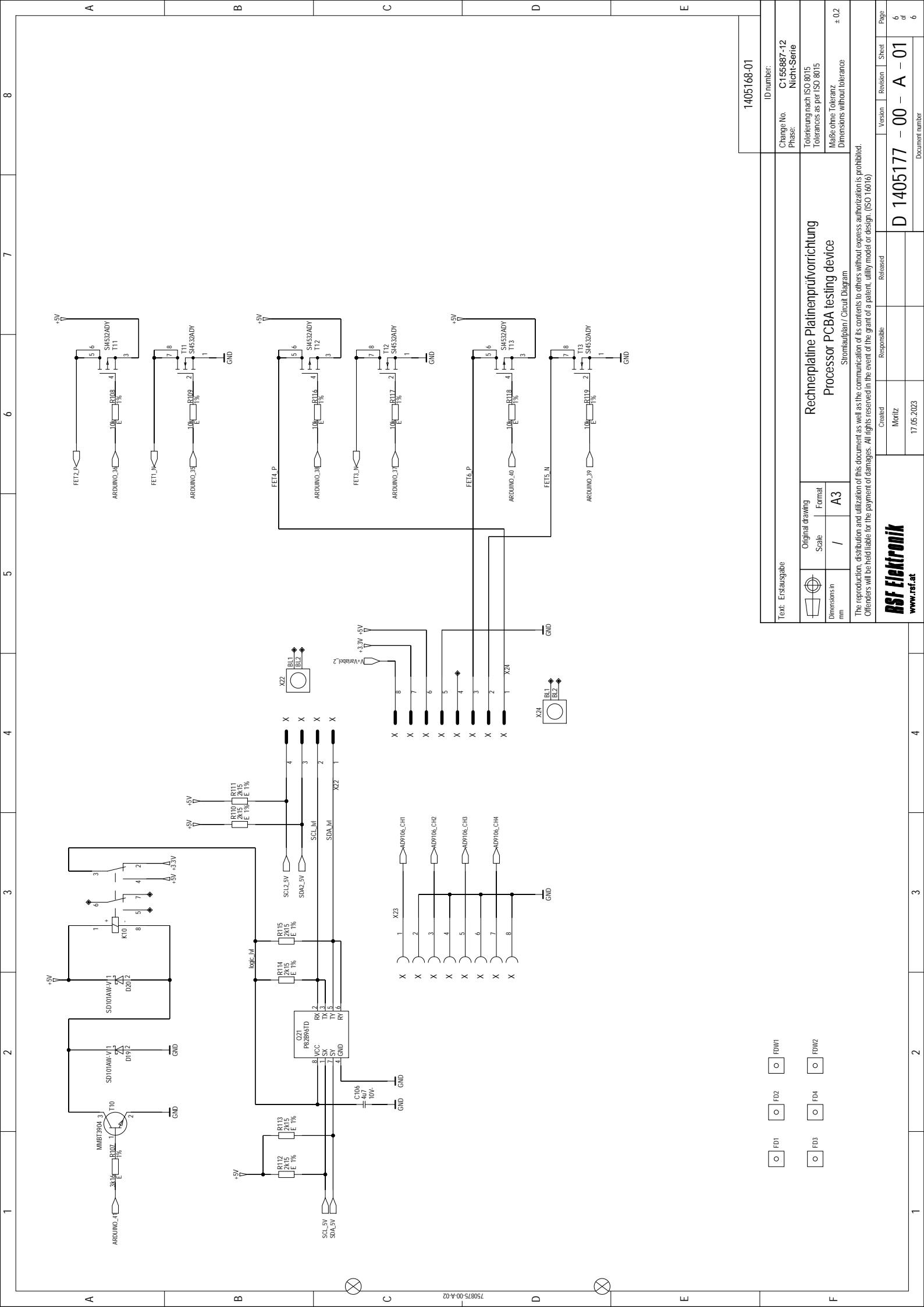


Abbildung A.4.: Vollbestückte Leiterplatte.









8

7

6

5

4

3

2

1

A

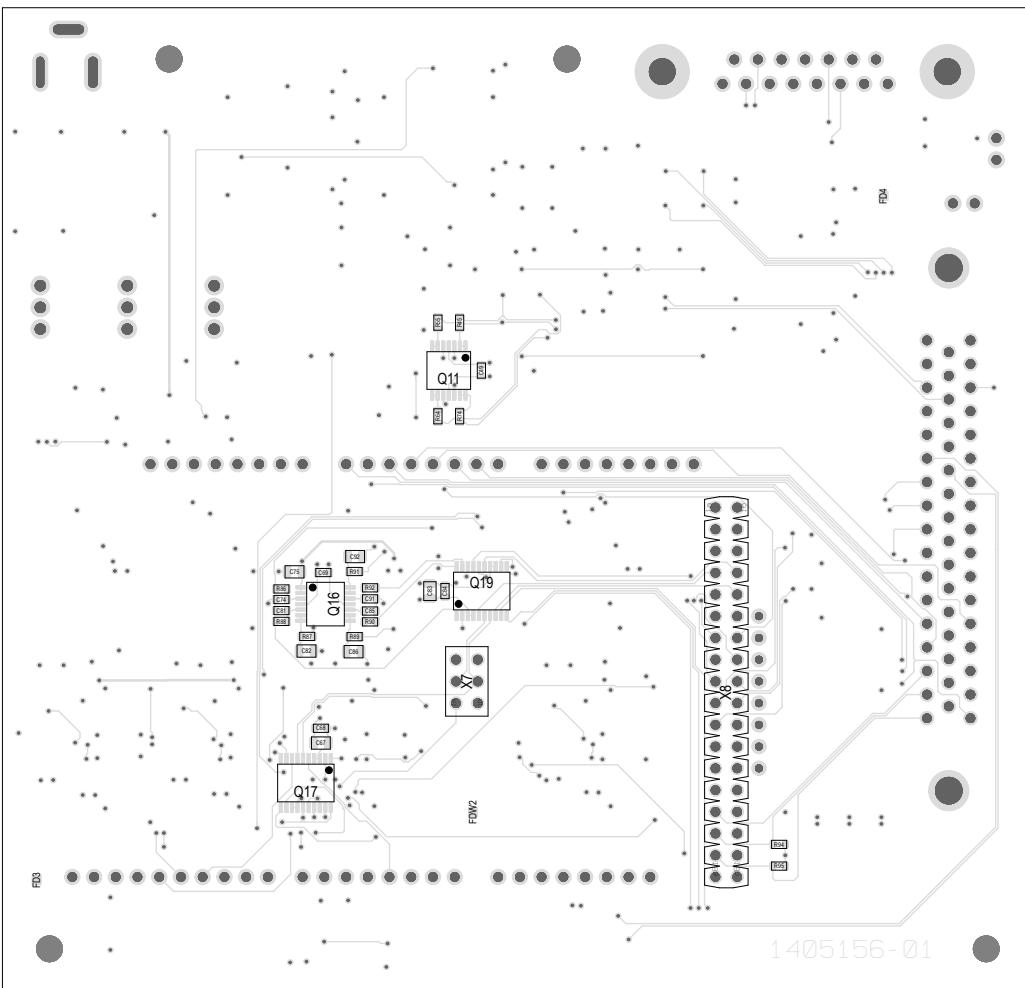
B

C

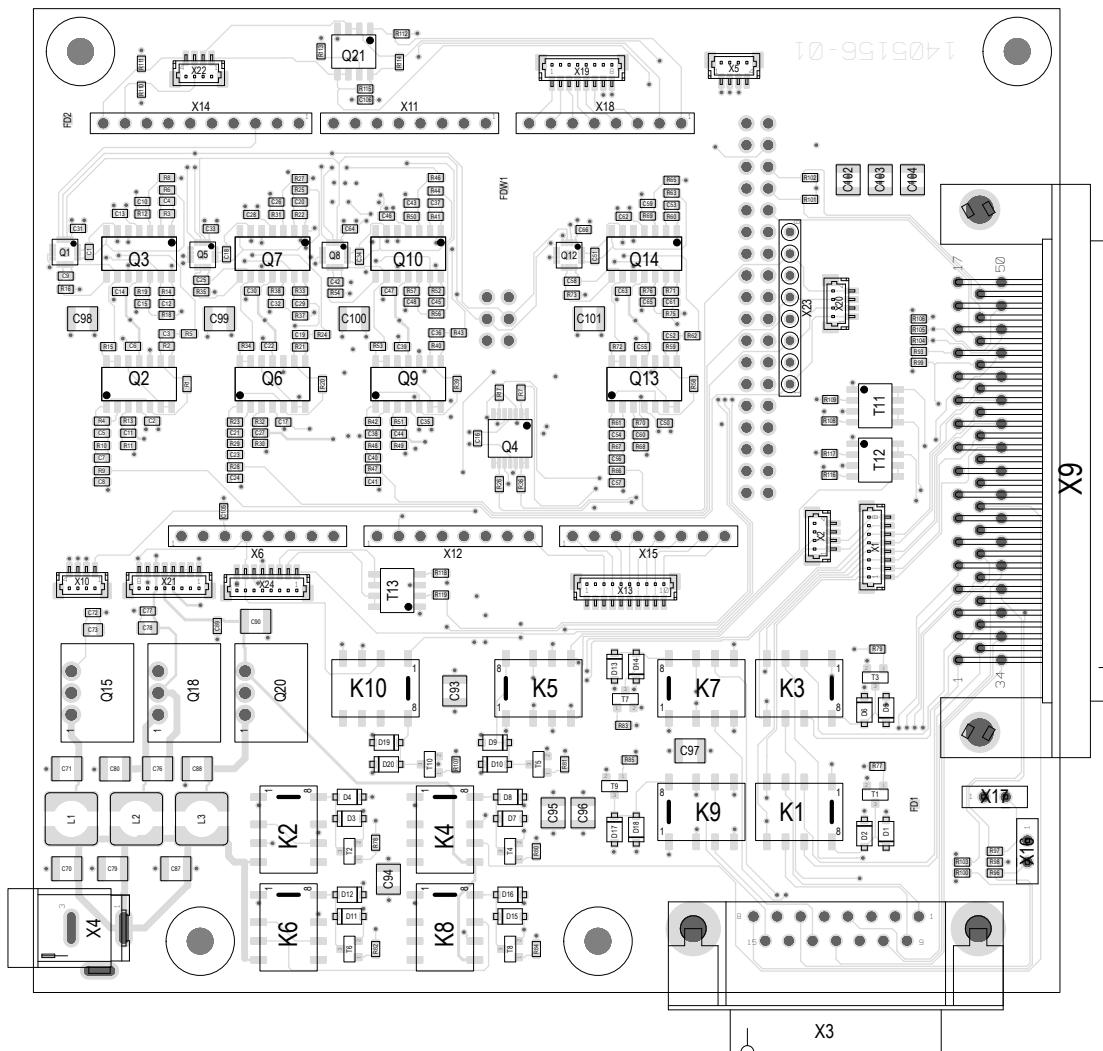
D

E

F



				Change No. Phase
				Teleferrung nach ISO 8015 Tolerances as per ISO 8015
				Masse ohne Toleranz Dimensions without tolerance ±0.2 mm
				The reproduction, distribution and utilization of this document as well as the communication of its contents to others without express authorization is prohibited. Offenders will be held liable for the payment of damages. All rights reserved in the event of the grant of a patent, utility model or design. (ISO 16016)
RSF Elektronik	Created	Responsible	Released	D 1405173 - 00 - -01 1
www.rsf.at	28.03.2023	Moritz		Page



750 875 A3

Tabelle A.1.: Modbus Kommunikationsprotokoll Teil 1/3.

Tabelle A.2.: Modbus Kommunikationsprotokoll Teil 2/3.

Addr	Register Name	Bits	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	R/W
14	Potentiometer 2	[15-8] [7-0]	RES.		Potentiometer Ch2 [13:8]						R/W
15	Potentiometer 3	[15-8] [7-0]	RES.		Potentiometer Ch3 [13:8]						R/W
16	Potentiometer 4	[15-8] [7-0]	RES.		Potentiometer Ch3 [7:0]						R/W
17	Potentiometer 5	[15-8] [7-0]	RES.		Potentiometer Ch4 [13:8]						R/W
18	Potentiometer 6	[15-8] [7-0]	RES.		Potentiometer Ch5 [7:0]						R/W
19	Potentiometer 7	[15-8] [7-0]	RES.		Potentiometer Ch6 [13:8]						R/W
20	Offset 0	[15-8] [7-0]	RES.		Potentiometer Ch6 [7:0]						R/W
21	Offset 1	[15-8] [7-0]	RES.		Offset Ch0 [7:0]						R/W
22	Offset 2	[15-8] [7-0]	RES.		Offset Ch1 [12:8]						R/W
23	Offset 3	[15-8] [7-0]	RES.		Offset Ch1 [7:0]						R/W
24	Offset 4	[15-8] [7-0]	RES.		Offset Ch2 [12:8]						R/W
25	Offset 5	[15-8] [7-0]	RES.		Offset Ch2 [7:0]						R/W
26	Offset 6	[15-8] [7-0]	RES.		Offset Ch3 [12:8]						R/W
					Offset Ch3 [7:0]						
					Offset Ch4 [12:8]						
					Offset Ch4 [7:0]						
					Offset Ch5 [12:8]						
					Offset Ch5 [7:0]						
					Offset Ch6 [12:8]						
					Offset Ch6 [7:0]						

Tabelle A.3.: Modbus Kommunikationsprotokoll Teil 3/3.

Addr	Register Name	Bits	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	R/W
27	Offset 7	[15-8]	RES.					Offset Ch7 [12:8]			R/W
		[7-0]					Offset Ch7 [7:0]				
28	Circuit Ref.	[15-8]	RES.					Offset Circuit Ref. [12:8]			R/W
		[7-0]					Offset Circuit Ref. [7:0]				
29	DAC 1	[15-8]	RES.					Offset DAC 1 [12:8]			R/W
		[7-0]				Offset DAC 1 [7:0]					
30	DAC 2	[15-8]	RES.					Offset DAC 2 [12:8]			R/W
		[7-0]				Offset DAC 2 [7:0]					
31	DAC 3	[15-8]	RES.					Offset DAC 3 [12:8]			R/W
		[7-0]				Offset DAC 3 [7:0]					
32	DAC 4	[15-8]	RES.					Offset DAC 4 [12:8]			R/W
		[7-0]				Offset DAC 4 [7:0]					
33	DAC 5	[15-8]	RES.					Offset DAC 5 [12:8]			R/W
		[7-0]				Offset DAC 5 [7:0]					
34	DAC 6	[15-8]	RES.					Offset DAC 6 [12:8]			R/W
		[7-0]				Offset DAC 6 [7:0]					
35	DAC 7	[15-8]	RES.					Offset DAC 7 [12:8]			R/W
		[7-0]				Offset DAC 7 [7:0]					
36	PowerDown Offset	[15-8]	Power Offset Ch7	Power Offset Ch6	Power Offset Ch5	Power Offset Ch4					R/W
		[7-0]	Power Offset Ch3	Power Offset Ch2	Power Offset Ch1	Power Offset Ch0					
37	PowerDown DAC Offset	[15-8]	Power DAC Ch7	Power DAC Ch6	Power DAC Ch5	Power DAC Ch4					R/W
		[7-0]	Power DAC Ch3	Power DAC Ch2	Power DAC Ch1	Power Circuit Ref.					
38	Reset AD9106	[15-8]	RES.								R/W
		[7-0]			RES.				Reset AD		
39	Fehler AD9106	[15-8]	AD9106 Error [15:8]								R/W
		[7-0]	AD9106 Error [7:0]								

Tabelle A.4.: Messdaten des Kanal 0 für Bode Diagramm.

Frequenz Hz	gemessen				berechnet			
	Signal 0		Signal 1		Phasendifferenz °	Signal 0 Magnitude dB	Signal 1 Magnitude dB	
	Amplitude mV	Phase °	Amplitude mV	Phase °	%			
10	1,00	181,10	1,00	1,42	-0,18	-0,03	0,00	
100	1,00	181,50	1,00	0,65	0,47	0,00	0,00	
1000	1,03	181,40	1,00	-0,34	0,97	-0,03	0,24	
10000	1,07	181,30	1,00	-0,42	0,95	0,00	0,59	
50000	1,06	172,70	1,00	-6,00	-0,73	-0,03	0,54	
100000	1,06	166,30	0,99	-12,43	-0,71	-0,10	0,47	
150000	1,05	160,00	0,98	-19,33	-0,37	-0,18	0,41	
200000	1,04	154,00	0,97	-25,84	-0,09	-0,28	0,31	
250000	1,02	147,90	0,96	-31,63	-0,26	-0,39	0,17	
300000	1,00	141,20	0,94	-38,20	-0,33	-0,50	0,03	
350000	0,98	135,30	0,92	-44,58	-0,07	-0,69	-0,14	
400000	0,97	129,60	0,90	-51,67	0,71	-0,88	-0,28	
450000	0,95	122,50	0,88	-56,05	-0,81	-1,07	-0,46	
500000	0,92	116,80	0,86	-62,65	-0,31	-1,31	-0,76	
550000	0,89	111,30	0,84	-68,69	-0,01	-1,56	-0,99	
600000	0,86	104,90	0,81	-74,14	-0,53	-1,85	-1,27	
650000	0,83	99,27	0,79	-80,41	-0,18	-2,07	-1,60	
700000	0,80	94,12	0,76	-85,97	0,05	-2,38	-1,89	
750000	0,78	88,05	0,73	-90,50	-0,81	-2,71	-2,20	
800000	0,75	82,58	0,70	-96,42	-0,56	-3,05	-2,52	
900000	0,69	73,29	0,65	-106,60	-0,06	-3,72	-3,25	
1000000	0,63	63,99	0,60	-116,60	0,33	-4,44	-3,99	

Tabelle A.5.: Messdaten des Kanal 1 für Bode Diagramm.

Frequenz Hz	gemessen				berechnet			
	Signal 2		Signal 3		Phasendifferenz °	Signal 2 Magnitude dB	Signal 3 Magnitude dB	
	Amplitude mV	Phase °	Amplitude mV	Phase °	%			
10	1,00	0,18	1,00	180,00	179,82	-0,10	0,03	0,00
100	1,01	-0,01	1,00	180,00	180,01	0,01	0,07	0,00
1000	1,01	-0,51	1,02	181,10	181,61	0,89	0,07	0,21
10000	1,01	-1,37	1,07	178,80	180,17	0,09	0,07	0,57
50000	1,00	-6,41	1,07	172,50	178,91	-0,60	0,00	0,57
100000	1,00	-13,31	1,06	165,70	179,01	-0,55	-0,03	0,47
150000	0,98	-20,00	1,04	159,60	179,60	-0,22	-0,14	0,34
200000	0,97	-27,46	1,03	153,70	181,16	0,64	-0,25	0,27
250000	0,96	-32,77	1,02	146,30	179,07	-0,52	-0,39	0,14
300000	0,94	-40,08	1,00	141,00	181,08	0,60	-0,50	0,02
350000	0,92	-46,25	0,98	134,60	180,85	0,47	-0,69	-0,18
400000	0,90	-52,37	0,96	128,40	180,77	0,43	-0,92	-0,32
450000	0,88	-58,21	0,93	122,40	180,61	0,34	-1,15	-0,61
500000	0,85	-64,22	0,91	115,60	179,82	-0,10	-1,39	-0,84
550000	0,83	-70,25	0,88	109,80	180,05	0,03	-1,64	-1,11
600000	0,80	-76,00	0,85	104,50	180,50	0,28	-1,98	-1,39
650000	0,77	-81,82	0,82	98,20	180,02	0,01	-2,25	-1,68
700000	0,74	-87,25	0,79	92,83	180,08	0,04	-2,57	-2,03
750000	0,72	-93,04	0,76	87,98	181,02	0,57	-2,90	-2,34
800000	0,68	-98,34	0,73	82,57	180,91	0,51	-3,30	-2,71
900000	0,64	-108,30	0,68	71,53	179,83	-0,09	-3,93	-3,35
1000000	0,58	-119,10	0,62	62,24	181,34	0,74	-4,67	-4,10

Tabelle A.6.: Messdaten des Kanal 2 für Bode Diagramm.

Frequenz Hz	gemessen				berechnet			
	Signal 4		Signal 5		Phasendifferenz °	Signal 4 Magnitude dB	Signal 5 Magnitude dB	
	Amplitude mV	Phase °	Amplitude mV	Phase °	%			
10	0,99	-0,16	0,99	179,80	179,96	-0,02	-0,10	-0,10
100	0,99	-0,27	0,99	180,40	180,67	0,37	-0,10	-0,10
1000	0,99	-0,28	1,01	181,50	181,78	0,99	-0,10	0,07
10000	0,99	-1,49	1,06	178,40	179,89	-0,06	-0,10	0,47
50000	0,99	-0,95	1,05	180,50	181,45	0,81	-0,10	0,41
100000	0,97	-12,61	1,04	168,80	181,41	0,78	-0,25	0,34
150000	0,96	-19,22	1,03	159,80	179,02	-0,54	-0,32	0,27
200000	0,96	-26,32	1,02	153,90	180,22	0,12	-0,39	0,17
250000	0,94	-32,97	1,00	147,40	180,37	0,21	-0,54	0,03
300000	0,93	-39,19	0,99	141,40	180,59	0,33	-0,65	-0,07
350000	0,91	-45,08	0,97	135,00	180,08	0,04	-0,84	-0,25
400000	0,89	-50,43	0,95	128,50	178,93	-0,59	-1,03	-0,46
450000	0,87	-56,45	0,93	122,40	178,85	-0,64	-1,23	-0,65
500000	0,85	-63,76	0,90	117,10	180,86	0,48	-1,43	-0,88
550000	0,82	-68,84	0,88	111,00	179,84	-0,09	-1,68	-1,15
600000	0,80	-74,34	0,85	104,10	178,44	-0,87	-1,94	-1,43
650000	0,77	-80,89	0,82	99,62	180,51	0,28	-2,25	-1,68
700000	0,75	-85,54	0,79	93,06	178,60	-0,78	-2,51	-2,03
750000	0,72	-91,72	0,76	87,75	179,47	-0,29	-2,85	-2,34
800000	0,70	-97,27	0,74	82,29	179,56	-0,24	-3,15	-2,66
900000	0,64	-107,30	0,67	71,71	179,01	-0,55	-3,88	-3,45
1000000	0,59	-117,20	0,62	61,94	179,14	-0,48	-4,55	-4,10

Tabelle A.7.: Messdaten des Kanal 3 für Bode Diagramm.

Frequenz Hz	gemessen				berechnet			
	Signal 6		Signal 7		Phasendifferenz °	Signal 6 Magnitude dB	Signal 7 Magnitude dB	
	Amplitude mV	Phase °	Amplitude mV	Phase °	%			
10	1,00	0,59	1,00	179,70	179,11	-0,49	0,03	0,03
100	1,01	0,26	1,01	179,60	179,34	-0,36	0,08	0,07
1000	1,01	-0,78	1,02	180,80	181,58	0,88	0,07	0,21
10000	1,01	-1,09	1,07	178,80	179,89	-0,06	0,07	0,60
50000	1,00	-6,16	1,07	172,50	178,66	-0,74	0,00	0,60
100000	0,98	-14,90	1,06	165,20	180,10	0,06	-0,14	0,51
150000	0,98	-19,93	1,05	159,70	179,63	-0,21	-0,18	0,44
200000	0,97	-27,04	1,04	153,60	180,64	0,36	-0,28	0,34
250000	0,96	-33,22	1,03	147,20	180,42	0,23	-0,39	0,24
300000	0,94	-39,84	1,01	140,90	180,74	0,41	-0,50	0,10
350000	0,92	-46,72	0,99	134,90	181,62	0,90	-0,72	-0,10
400000	0,90	-52,23	0,97	128,20	180,43	0,24	-0,95	-0,25
450000	0,88	-58,42	0,94	122,00	180,42	0,23	-1,15	-0,50
500000	0,85	-65,04	0,92	115,80	180,84	0,47	-1,39	-0,72
550000	0,83	-71,82	0,90	109,70	181,52	0,84	-1,64	-0,95
600000	0,80	-76,67	0,86	104,00	180,67	0,37	-1,94	-1,27
650000	0,77	-82,44	0,84	98,12	180,56	0,31	-2,25	-1,51
700000	0,74	-88,59	0,81	92,97	181,56	0,87	-2,57	-1,85
750000	0,72	-93,56	0,78	86,86	180,42	0,23	-2,90	-2,20
800000	0,69	-99,48	0,75	81,54	181,02	0,57	-3,25	-2,52
900000	0,63	-109,90	0,69	71,47	181,37	0,76	-3,99	-3,25
1000000	0,58	-120,00	0,63	61,54	181,54	0,86	-4,73	-3,99

Tabelle A.8.: Messdaten des Kanal 0 für Bode Diagramm ohne Optimierung.

Frequenz Hz	gemessen				berechnet			
	Signal 4		Signal 5		Phasendifferenz °	Magnitude dB	Signal 4	Signal 5
	Amplitude mV	Phase °	Amplitude mV	Phase °			Magnitude dB	Magnitude dB
10	0,988	-0,68	0,992	179,70	180,38	0,21	-0,105	-0,070
100	0,988	0,98	0,994	179,70	178,72	-0,71	-0,105	-0,052
1000	1,004	1,23	0,996	179,10	177,87	-1,18	0,035	-0,035
10000	1,052	-2,11	0,992	178,20	180,31	0,17	0,440	-0,070
50000	1,056	-8,45	0,988	173,70	182,15	1,19	0,473	-0,105
100000	1,044	-17,26	0,984	167,00	184,26	2,37	0,374	-0,140
150000	1,032	-24,26	0,976	160,40	184,66	2,59	0,274	-0,211
200000	1,016	-32,99	0,964	154,30	187,29	4,05	0,138	-0,318
250000	1,000	-40,94	0,952	147,80	188,74	4,86	0,000	-0,427
300000	0,976	-48,51	0,936	140,90	189,41	5,23	-0,211	-0,574
350000	0,952	-56,76	0,912	134,50	191,26	6,26	-0,427	-0,800
400000	0,928	-64,30	0,896	128,30	192,60	7,00	-0,649	-0,954
450000	0,900	-71,67	0,872	121,50	193,17	7,32	-0,915	-1,190
500000	0,868	-79,53	0,848	115,80	195,33	8,52	-1,230	-1,432
550000	0,832	-87,21	0,824	110,10	197,31	9,62	-1,598	-1,681
600000	0,804	-94,59	0,796	104,80	199,39	10,77	-1,895	-1,982
650000	0,756	-102,20	0,764	99,70	201,90	12,17	-2,430	-2,338
700000	0,732	-108,70	0,748	93,70	202,40	12,44	-2,710	-2,522
750000	0,700	-115,80	0,716	87,55	203,35	12,97	-3,098	-2,902
800000	0,664	-122,50	0,692	82,32	204,82	13,79	-3,557	-3,198
900000	0,596	-135,50	0,636	72,61	208,11	15,62	-4,495	-3,931

Tabelle A.9.: Messdaten des Kanal 0 für Bode Diagramm der Teilschaltung.

Frequenz Hz	gemessen				berechnet			
	Signal 0		Signal 1		Phasendifferenz		Signal 0	Signal 1
	Amplitude mV	Phase °	Amplitude mV	Phase °	°	%	Magnitude dB	Magnitude dB
10	0,97	180,80	0,96	0,32	180,48	0,27	-0,35	-0,26
100	0,97	179,20	0,96	0,74	178,46	-0,85	-0,35	-0,23
1000	0,99	181,80	0,96	0,21	181,59	0,89	-0,35	-0,09
10000	1,02	180,10	0,96	-0,56	180,66	0,37	-0,35	0,17
50000	1,02	178,10	0,96	-4,20	182,30	1,28	-0,35	0,17
100000	1,02	176,70	0,96	-6,42	183,12	1,73	-0,35	0,17
150000	1,02	175,00	0,95	-10,50	185,50	3,06	-0,45	0,17
200000	1,02	173,20	0,94	-13,23	186,43	3,57	-0,54	0,17
250000	1,01	171,50	0,93	-16,54	188,04	4,47	-0,63	0,09
300000	1,00	170,60	0,93	-20,07	190,67	5,93	-0,63	0,00
350000	1,00	168,20	0,92	-22,07	190,27	5,71	-0,72	0,00
400000	0,99	168,30	0,92	-26,04	194,34	7,97	-0,72	-0,09
450000	0,99	166,10	0,90	-29,41	195,51	8,62	-0,92	-0,09
500000	0,99	164,60	0,88	-32,10	196,70	9,28	-1,11	-0,09
550000	0,97	163,40	0,87	-35,63	199,03	10,57	-1,21	-0,26
600000	0,97	162,10	0,86	-38,67	200,77	11,54	-1,31	-0,26
650000	0,97	160,10	0,84	-41,57	201,67	12,04	-1,51	-0,26
700000	0,96	158,90	0,83	-44,35	203,25	12,92	-1,62	-0,35
750000	0,95	156,90	0,81	-47,68	204,58	13,66	-1,83	-0,45
800000	0,94	156,00	0,79	-49,89	205,89	14,38	-2,05	-0,54
900000	0,93	152,40	0,76	-54,62	207,02	15,01	-2,38	-0,63