Aufbau und Test eines Messsystems zur Detektion thermischer Strahlung während der Lasermaterialbearbeitung

Setup and test of a measuring system for the detection of thermal radiation during laser material processing

von
Marco Ferrari, Mat.# 01129062

Diplomarbeit

zur Erlangung des akademischen Grades
Diplomingenieur

eingereicht an der
Technischen Universität Wien

Betreuer
Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Gerhard Liedl

Projektass. Dipl.-Ing. Gerald Humenberger

E311 - Institut für Fertigungstechnik und Hochleistungslasertechnik

Fakultät für Maschinenwesen und Betriebswissenschaften

Wien, Oktober 2018

Ich habe zur Kenntnis genommen, dass ich zur Drucklegung meiner Arbeit unter der Bezeichnung

Diplomarbeit

nur mit Bewilligung der Prüfungskommission berechtigt bin.

Ich erkläre weiters Eides statt, dass ich meine Diplomarbeit nach den anerkannten Grundsätzen für wissenschaftliche Abhandlungen selbstständig ausgeführt habe und alle verwendeten Hilfsmittel, insbesondere die zugrunde gelegte Literatur, genannt habe.

Weiters erkläre ich, dass ich dieses Diplomarbeitsthema bisher weder im In- noch Ausland (einer Beurteilerin/einem Beurteiler zur Begutachtung) in irgendeiner Form als Prüfungsarbeit vorgelegt habe und dass diese Arbeit mit der vom Begutachter beurteilten Arbeit übereinstimmt.

Wien, am _____ _____

Marco Ferrari

Kurzzusammenfassung

Ein optisches Messsystem kann eine Prozessüberwachung und -regelung beim Laserstrahlschneiden schaffen. Die von der heißen Schneidfront emittierte Infrarot-Strahlung weist eine von deren Neigung abhängige Polarisation auf. Diese soll mit einem Messsystem erfasst und schnell digital ausgewertet werden.
Zentrales Element dieses Systems ist ein SCPEM (Single Crystal Photo Elastic Modulator), der beim Schwingen in einer seiner Resonanzfrequenzen durch seine Doppelbrechung eine Änderung der Polarisation bzw. mit einem Analysator der Strahlungsintensität bewirkt. Dieser Intensitätsverlauf wird von einer Photodiode erfasst und das verstärkte Signal digitalisiert. Ein für diese Aufgabe bestehender Messaufbau wird dazu optisch für Wellenlängen im Nahen-Infrarot-Bereich umgerüstet. Die Ansteuerung des Kristalls wird überarbeitet und als Platine ausgeführt.
Ein Gleichungssystem zur Berechnung der Intensität in Abhängigkeit von der Eingangs-Polarisation wird aufgestellt. Mit einigen Einschränkungen kann auf eine inverse Funktion geschlossen werden, die vom gemessenen Intensitätsverlauf auf die Polarisationsrichtung schließen lässt.
Zur Realisierung einer funktionsfähigen Auswerteeinheit wird auf einem Messboard ein ausreichend schnelles Programm für die Auswertung der digitalisierten Signale und die damit durchgeführte Berechnung der Polarisationswerte entwickelt. Die Darstellung und das Speichern der berechneten Ergebnisse wird auf einem Computer durchgeführt, für die dafür nötige Kommunikation zwischen den beiden Geräten werden verschiedene Möglichkeiten verglichen und ein eigenes Protokoll für eine effiziente und sichere Datenübertragung definiert.
Das dazu passend erstellte Messclient-Programm am Computer erfüllt mit seiner grafischen Oberfläche zahlreiche Aufgaben, unter anderem ermöglicht es die verschiedenen Einstellungen des Messprogramms, empfängt, visualisiert und archiviert die Polarisations-Messdaten und bietet Analysefunktionen für die gespeicherten Aufnahmen.

Abstract

An optical measurement device could give the possibility to monitor and regulate a laser cutting machine. The hot surface at the front of the cut emits infrared radiation which polarisation depends on the angle of this surface. A measurement unit should capture and evaluate the polarisation.
The main component is a SCPEM (Single Crystal Photo Elastic Modulator) that changes the polarisation when oscillating at one of its resonance frequencies because of its birefringence. When it's combined with an analyser, the resulting radiation has a changing intensity that can be detected with a photodiode, the amplified signal is then digitised. An existing test setup for this application is adapted for the use with infrared radiation. The control system of the crystal is revised and manufactured as a circuit board.
A system of equations for the radiation intensity in dependence of the polarisation can be found. This system can be reversed and a function to calculate the polarisation angle from the measured intensity is possible with some limitations.
To achieve a functioning evaluation unit for the measured signals, a fast program is developed on a measuring board. It evaluates the digital samples and calculates the current polarisation angle. This data is sent to a computer to store and display it. Different approaches for the communication are tested and a protocol for an efficient and secure data transfer is defined.
A compatible program is created for a computer. It features a graphical user interface that gives the ability to set different options for the measuring program, it receives polarisation data to save and visualize it, and it can be used to analyse recorded measurements.

Inhaltsverzeichnis

1 Einleitung und Zielsetzung

1.1 Qualitätssicherung beim Laserstrahlschneiden

1.1.1 Laserstrahlschneiden

Laserstrahlschneiden ist ein thermischer Schneidprozess, bei dem ein fokussierter Laserstrahl auf das Werkstück trifft und dieses im kleinen Bereich des Auftreffpunktes stark erwärmt. Die Erwärmung führt hier zu einem Phasenübergang in den flüssigen oder gasförmigen Zustand, dieses Material wird von zum Laserstrahl koaxial angeordneten Düse vom Grundmaterial entfernt. Dadurch entsteht ein Schnitt, der durch Bewegen des Werkstückes oder des Schneidkopfes in einer bestimmten Richtung fortgesetzt wird. Der Vorgang läuft berührungslos ab, die Wärmeeinbringung ins restliche Werkstück ist gering und es bildet sich ein schmaler Schneidspalt mit nahezu senkrechter seitlicher Schneidfläche.

Auf diese Weise können viele unterschiedliche Materialien und Materialstärken geschnitten werden. Der Laserstrahl wird meist von einem stationären Laser über Spiegel oder Fasern zum Schneidkopf geleitet. Häufig ist die Bearbeitung zweidimensional, zum Beispiel Ausschneiden aus Blechen, dreidimensionale Bearbeitung ist ebenfalls möglich. [1]

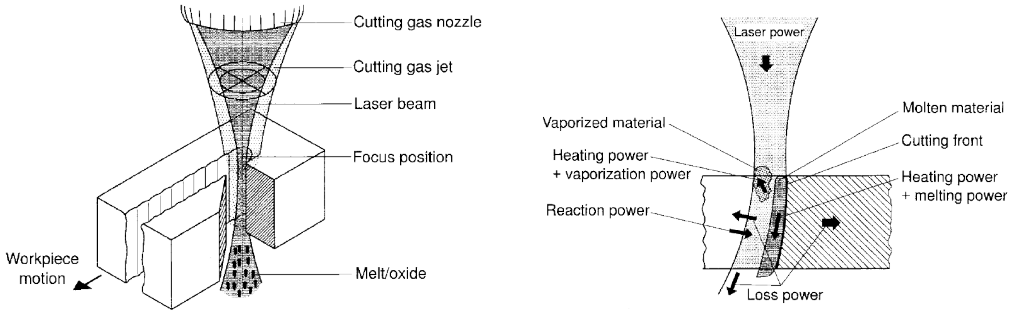


Abbildung 1.1: Schneidvorgang beim Laserschneiden, Schneidfront im Detail [1]

Abb. 1.1 zeigt den Schneidvorgang und die Vorgänge in der Schneidfuge. Der Laserstrahl wird vom Gas umströmt, das Material des Schneidspalts wird unten ausgeblasen. Die Schneidfront ist von der Senkrechten weggeneigt und von einem dünnen Film aus flüssigem Material überzogen.

1.1.2 Problemstellung

Laserstrahlschneiden hat durch den berührungslosen Schneidvorgang keine Möglichkeit, den aktuellen Zustand durch Reaktionskräfte oder Vibrationen zu überwachen und dar-

aus während des Schnittes Rückschlüsse auf die Qualität zu liefern. Erst nach dem Schnitt kann diese beurteilt werden und gegebenenfalls Prozessgrößen angepasst werden.

Hilfreich wäre eine Möglichkeit der Beurteilung und Überwachung des Schneidvorganges in Echtzeit, um somit eine Eingangsgröße für die Regelung der Laserleistung oder des Vorschubes zum Erhalt einer gleichbleibend guten Schnittqualität zu erhalten. Weiters können dadurch Daten für eine lückenlose Aufzeichnung der Qualität von jedem Bauteil ermittelt werden. [2]

1.1.3 Lösungsansatz der optischen Prozessüberwachung

Durch die Verwendung von optischen Komponenten im Schneidkopfes und den damit verbundenen Anforderungen an die Umgebung bietet sich ein optisches Messsystem an. Dieses ist, wie der Prozess, berührungslos und ermöglicht hohe Abtastraten. Schwierigkeiten bringen die starke Strahlung des Lasers im Schneidbereich, die andere Messgrößen überdecken könnte, und die sehr kleine Messfläche an der Schneidfront mit sich. [2]

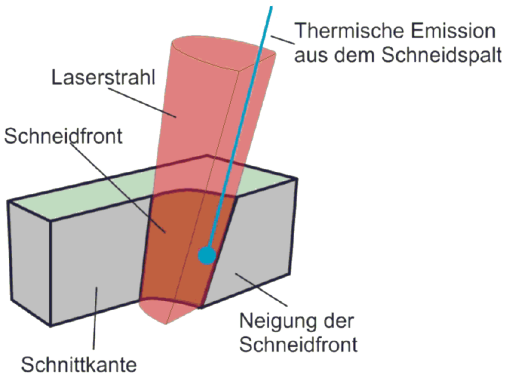


Abbildung 1.2: Neigung der Schneidfront [3]

Die Wärmezufuhr erfolgt nur sehr lokal, vom Werkstück emittierte Wärmestrahlung bietet daher automatisch eine Eingrenzung auf die nähere Umgebung der Schneidfront, deren Neigung δ ein wesentliches Merkmal eines korrekten Schnittes ist. Abhängig von der Neigung einer Oberfläche ändert sich die Polarisation der emittierten Strahlung. Diese Tatsachen lassen sich kombinieren, indem die Wärmestrahlung, die an der Schneidfront in Gegenrichtung des Laserstrahls emittiert wird, gesammelt und deren Polarisation untersucht wird. Damit lässt sich auf die Neigung der Front rückschließen und damit auf ein direktes Qualitätsmerkmal des Schnittes.

Es wird eine möglichst hohe Samplingrate angestrebt, die durch Verwendung eines SCPEM (Single Crystal Photo-Elastic Modulator) mit einer Eigenfrequenz im Bereich von 100 kHz und einem schnellen Auswertesystem erreicht wird. Dadurch können beim Schneiden in engen Abständen Messpunkte erfasst werden, Tab. 1.1. Die aktuell möglichen Abtastraten liegen beim verwendeten Kristall bei 80 kS/s bzw. bei Nutzung der doppelten Auswertung je Periode bei 160 kS/s.

Abtastrate in Samples/s	Prozessgeschwindigkeit	Messpunkte pro mm
100 S/s	100 m/min	0,06
10 kS/s	100 m/min	6
100 kS/s	100 m/min	60
80 kS/s	100 m/min	48
160 kS/s	100 m/min	96

Tabelle 1.1: Messpunkte je Schnittlänge abhängig von der Samplingrate [3]

1.1.3.1 Aufbau des optischen Prozessüberwachungssystems

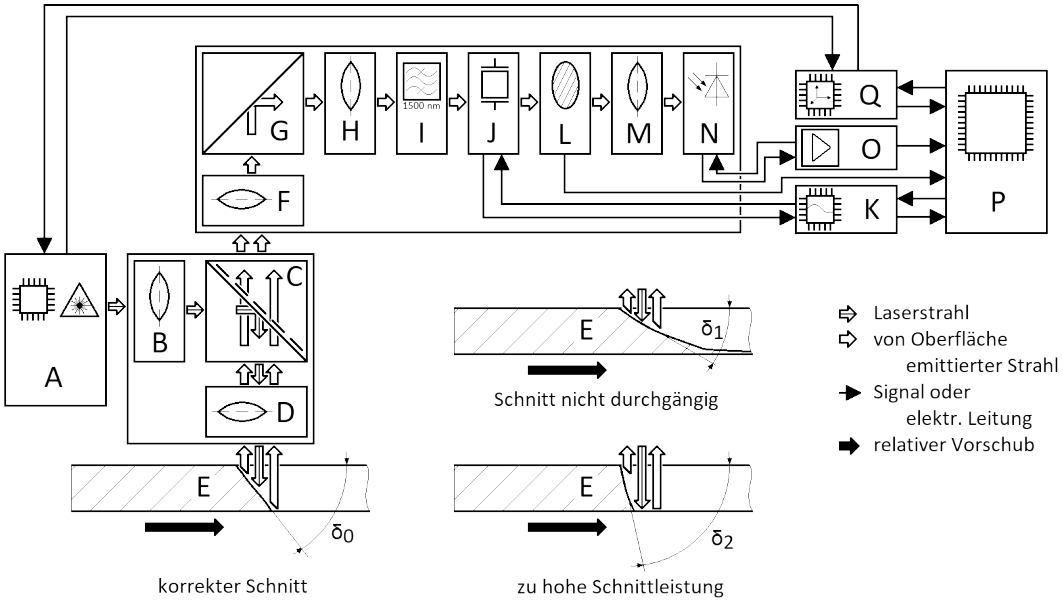


Abbildung 1.3: Gesamtsystem zur Überwachung und Regelung des Schneidprozesses

Abb. 1.3 zeigt einen möglichen Aufbau eines solchen Messsystems an einer Laserschneidbearbeitungsmaschine. Darin sind A-D die Schneidmaschine, F-M das optische Messsystem und N-Q elektrische und elektronische Komponenten des Messgerätes. Ebenfalls zu sehen ist der Winkel der Schneidfront δ, der ermittelt werden soll, und die beiden möglichen Fehler, wenn dieser zu klein oder zu groß ist. Die Steuergrößen sind Laserleistung und Schneidgeschwindigkeit und können in Abhängigkeit von diesem Winkel beeinflusst werden.

Wie in Tab. 1.2 zusammengefasst, kann ein zu kleiner Winkel (δ_1) als Anzeichen für einen teilweise nicht durchgängigen Schnitt und somit eines nach Fertigstellung der Schnittkontur unbrauchbaren Bauteils gewertet werden. Dieser Zustand soll von der Regelung zu einer Erhöhung der Laserleistung und/oder einer Verringerung der Schneidgeschwindigkeit führen, um wieder die richtige Neigung zu erreichen. Ist der Winkel

zu groß (δ_2), wird die mögliche Schneidgeschwindigkeit nicht genutzt oder mit unnötig hoher Laserleistung gearbeitet. Hier erfolgt die Anpassung dieser Größen in die andere Richtung, um eine höhere Effizient zu erreichen.

Winkel δ	Charakteristik	Ursache	Beeinflussung
δ klein (Fall δ_1)	flache Schneidfront, evtl. Material nicht vollständig getrennt	Laserleistung nicht ausreichend	Laserleistung erhöhen
		Schneidgeschwindigkeit zu hoch	Schneidgeschwindigkeit verringern
δ im richtigen Bereich (Fall δ_0)	vollständige Nutzung der Energie im Laserstrahl zur Materialerwärmung	richtig abgestimmte Prozessparameter, evtl. höhere Prozessgeschw. möglich	bei Bedarf Laserleistung und Schneidgeschwindigkeit kontinuierlich erhöhen
δ groß (Fall δ_2)	sehr steile Schneidfront, Laserstrahl	Laserleistung unnötig hoch	Laserleistung verringern
	trifft nicht vollständig auf Schneidfront	Schneidgeschwindigkeit nicht ausgereizt	Schneidgeschwindigkeit erhöhen

Tabelle 1.2: Charakteristik des Schneidfrontwinkels

Dem Weg des Lichtes folgend, sind die Komponenten des Gesamtsystems Abb. 1.3:

A: Laser und mechanischer Aufbau zur Bewegung des Schneidkopfes relativ zum Werkstück. Übertragung des Laserstrahls zum Schneidkopf über Spiegel oder Fasern.

B: Linse beim Eintritt in den Schneidkopf.

C: Halbdurchlässiger Spiegel, der den Laserstrahl Richtung Werkstück ablenkt.

D: Die Sammellinse fokussiert den Laserstrahl auf das Werkstück. Meist ist koaxial dazu eine Düse für ein Prozessgas angeordnet, die hier nicht näher betrachtet wird.

E: Am Werkstück erwärmt der Strahl das Material lokal, durch die Relativbewegung und das Entfernen des geschmolzenen/verdampften Materials entsteht die für die Messung relevante Neigung der Schneidfront. Diese gehört zu den heißesten Regionen am Werkstück und emittiert viel Strahlung. Ein Teil dieser Strahlung mit der für die Neigung typischen Polarisation (Kap. 2.1.4.1) geht durch die Sammellinse D und durch den halbdurchlässigen Spiegel C durch den Schneidkopf hindurch und gelangt schließlich in das Messsystem.

F: Linse beim Eintritt in das Messsystem.

G: Spiegel (optional, senkrechte Anordnung ebenfalls möglich).

H: Linse vor den polarisationsbeeinflussenden optischen Elementen.

I: Interferenzfilter für eine bestimmte Wellenlänge (1.500 nm). Für die korrekte Funktion des Messsystems ist eine möglichst einheitliche Frequenz erforderlich, siehe Kap. 2.1.3 und Kap. 2.2.1.

J: Ein SCPEM (Single Crystal Photo-Elastic Modulator) ermöglicht eine zeitliche Stückelung der Polarisationsanteile im Strahl. Durch die hohe Schwingfrequenz kann aus dem daraus austretenden Strahl bzw. dessen zeitl. Intensitätsverlauf schnell (im Bereich von 100 kHz) auf die Polarisationseigenschaften vor dem SCPEM geschlossen werden. Eine genauere Beschreibung der Funktion ist in Kap. 2.2.1 gegeben, die mathematische Beschreibung in Kap. 3 bzw. 3.2.3.

K: Damit der SCPEM mit einer bestimmten Amplitude stabil in der gewünschten Resonanzfrequenz (Längsschwinger) schwingt, ist eine elektrische Ansteuerung nötig. Diese wird in Kap. 5 beschrieben.

L: Analysator mit einer bestimmten relativen Winkelposition zur SCPEM-Achse. Diese Position muss der Auswerteeinheit P bekannt sein.

M: Sammellinse vor der Photodiode.

N: Photodiode mit Empfindlichkeit im richtigem Wellenlängenbereich.

O: Verstärkung des Signals der Photodiode auf ein für den Analog-Digital-Wandler passendes Niveau.

P: Auswerteeinheit, welche die Signale von SCPEM und Photodiode digitalisiert und mit weiteren Eingangsgrößen auf die Polarisation des vom Werkstück emittierten Strahls vor dem Messsystem und in weiterer Folge mit Richtungsdaten von Q auf den Winkel δ schließt. Es besteht die Möglichkeit, den SCPEM über dessen Ansteuerung K zu beeinflussen. Die Ergebnisse der Berechnungen werden angezeigt, gespeichert und/oder an das gerätespezifische Regelungssystem Q übergeben. Die Auswertung erfolgt im Idealfall mit der Frequenz des SCPEM, um ein Maximum an Informationen zu erhalten oder für stabilere Ergebnisse eine Mittelung über mehrere Perioden vornehmen zu können. Die Auswerteeinheit kann aus mehreren verbundenen Geräten für unterschiedliche Aufgaben bestehen.

Q: Das Regelungssystem ist spezifisch für das Bearbeitungsgerät ausgelegt und kann als Software in der Auswerteeinheit integriert oder als eigenes Gerät ausgeführt sein. Es hat zwei Aufgaben: Die erste ist die Ermittlung des Zusammenhanges zwischen Polarisation und Neigung der Schneidfront abhängig von der aktuellen Schneidrichtung. Diese Daten werden aus der Maschinensteuerung gewonnen und der Auswerteeinheit möglichst aktuell zur Verfügung gestellt. Die zweite Aufgabe ist die Regelung der Prozessgrößen Laserleistung und Schneidgeschwindigkeit und evtl. weiteren Größen (Gasvolumen etc.) anhand der Ergebnisse der Auswerteeinheit.

1.2 Umfang und Ziel dieser Arbeit

Diese Arbeit beschäftigt sich mit der Untersuchung und Umsetzung einiger Komponenten des in Kap. 1.1.3 vorgestellten Messgerätes. Der bereits zur Verfügung stehende Messaufbau ist für Licht im sichtbaren Bereich des Frequenzspektrums geeignet und soll für den Einsatz mit Infrarot-Strahlung adaptiert und die SCPEM-Ansteuerung angepasst werden. Kernaufgabe ist die Entwicklung einer funktionsfähigen Auswerteeinheit.

1.2.1 Betrachtete Komponenten des Prozessüberwachungssystems

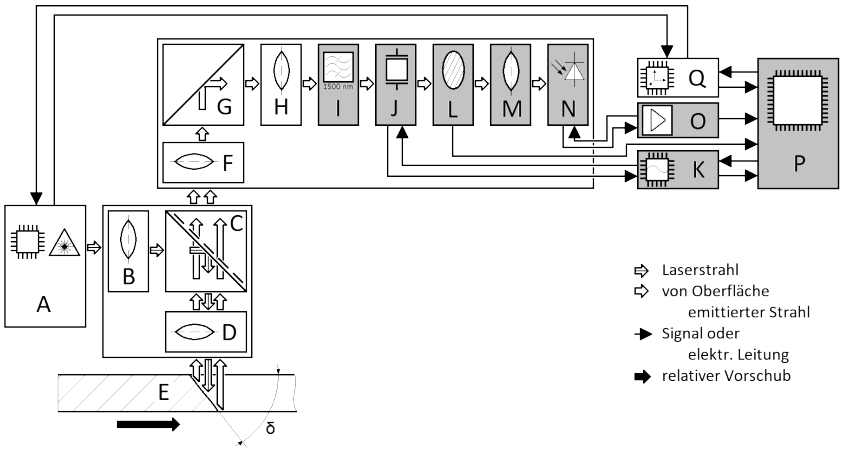


Abbildung 1.4: Betrachtete Komponenten aus dem Gesamtsystem zur Überwachung und Regelung des Schneidprozesses (grau hinterlegt)

Aus dem Gesamtmesssystem werden die Komponenten I-P betrachtet (Abb. 1.4). Der von der bearbeiteten Oberfläche kommende polarisierte Strahl wird durch einen Infrarotemitter und einen Polarisator ersetzt, die vor dem Interferenzfilter angeordnet werden. Damit wird eine Polarisation vorgegeben, die vom Messsystem bestimmt werden soll. Die Kommunikation mit der Bearbeitungsmaschine durch Q bleibt offen, die Möglichkeit einer solchen wird jedoch durch eine möglichst geringe Zeitverzögerung der Datenauswertung berücksichtigt.

Als Auswerteeinheit ist das Messboard STEMlab von RedPitaya (Kap. 2.3) zusammen mit einem Computer vorgesehen, es wird die Durchführbarkeit der Auswertung mit diesem System bzw. dessen Grenzen gezeigt.

1.2.2 Ziel

Am Ende der Arbeit soll ein Messaufbau vorhanden sein, an dem der Polarisationswinkel von polarisiertem Licht im NIR-Bereich ermittelt werden kann. Diese Ermittlung soll mit der Frequenz des SCPEM möglich sein, die Daten aufgezeichnet, analysiert und betrachtet werden können. Die Dokumentation erfolgt in den jeweiligen Sourcecode-Dateien sowie diesem Dokument selbst.

2 Grundlagen

2.1 Optik

Die physikalische Grundlage des Messsystems bilden die Gesetze der Optik. Besonders wichtig sind die Emission von Oberflächen, die Eigenschaften im Infrarotbereich und die Polarisationseigenschaften im Zusammenspiel mit unterschiedlichen optischen Elementen.

Die Eigenschaften werden in folgenden meist als ideal betrachtet und dargestellt, reale optische Bauteile können je nach Qualität mehr oder weniger von dieser Beschreibung abweichen.

2.1.1 Optische Strahlung, Wellenmodell

Optische Strahlung oder Licht ist eine Form von Energie, die auf unterschiedliche Weisen beschrieben werden kann. Das Wellenmodell wird für Erscheinungen verwendet, die Beugungs- und Überlagerungsverhalten aufweisen. Atomare oder opto-elektronische Themen werden mit einem Teilchen- oder Quantenmodell veranschaulicht.

Für viele Bereiche hat sich das Wellenmodell bewährt, welches beim Auftreten von sehr vielen Lichtquanten angewendet werden kann. Dies ist sowohl beim Laserstrahlschneiden als auch bei den durchzuführenden Messungen der Fall.

Die Strahlung wird als elektromagnetische Welle mit der Vakuumwellenlänge λ_0 betrachtet. Ein Teil davon wird als sichtbares Licht wahrgenommen, mit Wellenlängen von $\lambda_0 = 380$ nm bis 780 nm. Das Gebiet der ultravioletten Strahlung zeichnet sich mit kürzeren Wellenlängen aus ($100 \text{ nm} < \lambda_0 < 380$ nm), längere Wellenlängen im Infrarot-Bereich können in IR-A ($\lambda_0 = 780$ nm bis 1.400 nm), IR-B ($\lambda_0 = 1.400$ nm bis 3.000 nm) und IR-C ($\lambda_0 = 3.000$ nm bis 1 mm) unterteilt werden.

Der Zusammenhang zwischen Wellenlänge und Frequenz ist $\lambda = c/f$, mit der Frequenz f und der Lichtgeschwindigkeit c. Eine hohe Frequenz entspricht somit einer kleinen Wellenlänge und umgekehrt. [4]

2.1.2 Naher Infrarot-Bereich

Die Messung wird im nahen Infrarot-Bereich (NIR) durchgeführt, der die Spektralbereiche IR-A und IR-B umfasst und somit Wellenlängen von $\lambda_0 = 780$ nm bis 3.000 nm. Genauer soll sie bei 1.500 nm erfolgen. Die Beschränkung der Wellenlänge auf einen bestimmten Bereich ist durch Absorptionsfilter oder Interferenzfilter (Kap. 2.1.3) möglich. [4]

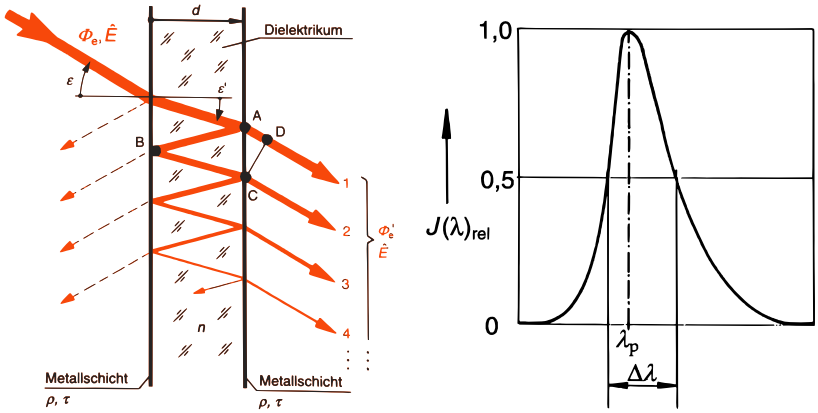


Abbildung 2.1: Vereinfachte Funktionsweise eines Interferenzfilters ($\varepsilon \approx 0$) [4]; Halbwertsbreite [4]

2.1.3 Interferenzfilter

Eine heiße Oberfläche emittiert Strahlung über einen breiten Frequenzbereich (Kap. 2.1.4). Der SCPEM (Kap. 2.2) benötigt ein möglichst schmales Frequenzband für die gewünschten Effekte.

Dadurch muss ein großer Teil der Strahlung ausgefiltert werden.

Die Funktion des Interferenzfilters ist in Abb. 2.1 vereinfacht dargestellt. Mehrere teildurchlässige Schichten bringen die Strahlung dazu, dass sie dazwischen mehrmals reflektiert wird. Bei Frequenzen, für die die Dicke d ein ganzzahliges Vielfaches der halben Wellenlänge darstellt, kommt es zu konstruktiver Interferenz, davon leicht abweichende Frequenzen verlieren an Intensität. Dadurch ist für die austretende Strahlung ein sehr schmalbandiges Spektrum möglich. Die grundlegenden Eigenschaften eines Bandpassfilters sind die Wellenlänge λ_p mit maximalem Transmissionsgrad τ_{max} und die Halbwertsbreite $\Delta\lambda$. [4]

2.1.4 Von Oberfläche emittierte Strahlung

Ein heißer Körper gibt Energie in Form von Strahlung ab. Dabei sind unterschiedliche spektrale Verteilungen möglich, wobei eine Erhöhung der Temperatur/Energie die Wellenlängen des Spektrums verringert.

Abb. 2.2 zeigt die spektrale Verteilung der Strahlung eines Schwarzen Strahlers bei unterschiedlichen Temperaturen.

Dieser Zusammenhang zwischen der Oberflächentemperatur eines Schwarzen Strahlers und dem Maximum der Abstrahlung wird vom Wienschen Verschiebungsgesetz beschrieben:

$$\lambda_{max} * T = 2{,}896\ mm * K\ [4]$$

$$\lambda_{max} * T = 2.897{,}8\ \mu m * K\ [5]$$

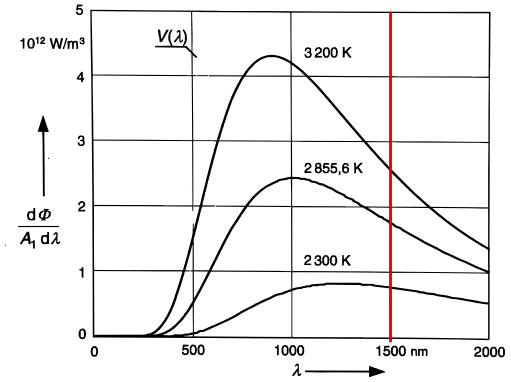


Abbildung 2.2: Spektrale Verteilung eines Schwarzen Strahlers [4]

Um mit $\lambda_{max} = 1.500$ nm die höchste Intensität im betrachteten Frequenzbereich zu erhalten, ist eine Temperatur im Bereich von 1.930 K nötig. Höhere Temperaturen erreichen durch die insgesamt viel stärkere Abstrahlung ebenfalls eine hohe Intensität bei dieser Wellenlänge. [4]

Zu beachten ist, dass es sich bei der Schneidoberfläche nicht um einen idealen Schwarzen Strahler handelt und die Temperatur in einem großen Bereich zwischen der Schmelztemperatur und der Siedetemperatur des geschnittenen Materials liegt. Eine für die Messung ausreichende Strahlungsmenge im IR-B-Bereich kann bei Verwendung einer geeigneten Optik dennoch erwartet werden, da bereits Verfahren mit Infrarot-Kameras eingesetzt werden. [2]

2.1.4.1 Einfluss des Abstrahlwinkels auf die Polarisation

Um aus dem emittiertem Strahl auf die Neigung der Oberfläche schließen zu können, wird die Abhängigkeit der Polarisation (Kap. 2.1.5) vom Abstrahlungswinkel genutzt.

Abb. 2.3 zeigt den Zusammenhang zwischen Abstrahlwinkel und Polarisationsanteilen (p parallel zur Fläche, s senkrecht dazu) für Stahl bei 300 K und 1.800 K.

Dabei ist der Temperatureinfluss auf das Polarisationsverhältnis gering und es lässt sich aus den Polarisationsanteilen auf den Neigungswinkel δ rückschließen bzw. dessen Änderung verfolgen. [3]

Dieser Zusammenhang zwischen dem Verhältnis der Emissionskoeffizienten und der Neigung wird in Kap. 3.4 auf das Vorliegen einer eindeutigen Lösung untersucht.

2.1.5 Polarisation und Müller-Formalismus

Der Polarisationszustand stellt eine wichtige Eigenschaft der Strahlung dar und bildet die zentrale Messgröße dieses Messsystems.

Im Wellenmodell ist die optische Strahlung durch Wellenlänge, Amplitude, Phase und Polarisation beschrieben. Polarisation beschreibt die Lage der Transversalwelle senkrecht zur Ausbreitungsrichtung. Abb. 2.4 zeigt verschiedene mögliche Polarisationszustände. [4]

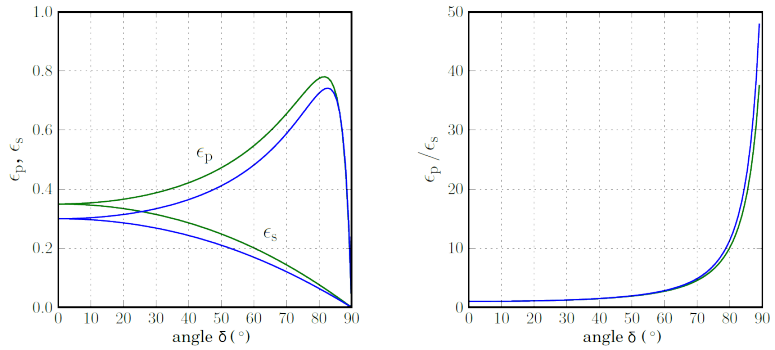


Abbildung 2.3: Polarisationsanteile der emittierten Strahlung einer heißen Stahloberfläche in Abhängigkeit vom Abstrahlwinkel [3]

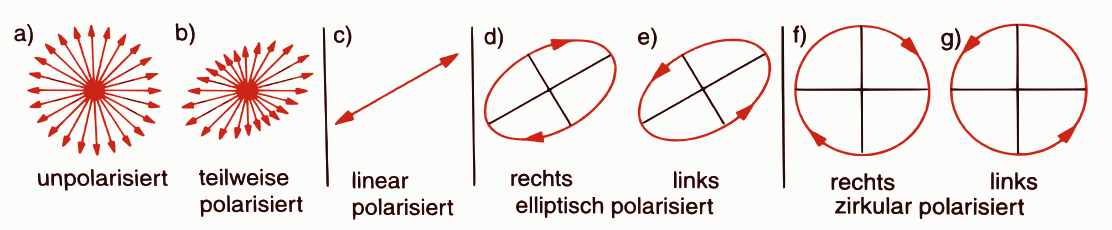


Abbildung 2.4: Polarisationszustände [4]

Es gibt einen fließenden Übergang zwischen unpolarisierter und vollständig polarisierter Strahlung, der durch den Polarisationsgrad DOP (Kap. 2.1.5.2) angegeben werden kann. [6]

2.1.5.1 Beeinflussung der Polarisation

Während die meisten Strahlungsquellen unpolarisiertes oder teilpolarisiertes Licht aussenden, gibt es eine Reihe von Elementen, die eine erwünschte oder unerwünschte Beeinflussung der Polarisation haben, zum Beispiel bei der Reflexion oder Brechung an Oberflächen oder beim Durchgang durch anisotrope Medien mit Doppelbrechung. Häufig verwendete Bauelemente sind Polarisatoren und auf dem Prinzip der Doppelbrechung beruhende Verzögerungsplatten.

Polarisatoren Polarisatoren ermöglichen die Erzeugung einer bestimmten Polarisationsrichtung aus unpolarisiertem Licht. Sie bestehen meist aus hochpolymeren Kunststofffolien mit eingelagerten dichroitischen Farbstoffen (unterschiedliche Absorption abhängig von Polarisationsrichtung). Durch Strecken der Folie werden die Farbstoffe parallel gerichtet. In der Durchlassrichtung polarisiertes Licht wird nur gering geschwächt, ist die Polarisation senkrecht dazu, wirkt der Polarisator absorbierend. Analysatoren unterscheiden sich von Polarisatoren nur in der Anwendung.

Wellenplatten Verzögerungsplatten haben unterschiedliche Brechungsindizes in verschiedenen Richtungen des Materials (doppelbrechend), wodurch sich Strahlung unterschiedlicher Polarisationsrichtungen unterschiedlich schnell fortbewegt. Dadurch kommt es zu einer Phasendifferenz der senkrecht zueinanderstehenden Wellen. Der Effekt auf die Phase ist abhängig von der Frequenz, die folgenden Wellenplatten funktionieren nur bei einer bestimmten Wellenlänge in dieser Weise. Die Verzögerung und damit die Phasendifferenz hängt vom Unterschied der Brechungsindizes und der Dicke der Platte ab.

Bei der Vollwellenplatte oder λ-Platte beträgt die Phasenverschiebung 2π. Bei einer Halbwellenplatte ($\lambda/2$-Platte) ist die Polarisation der austretenden Strahlung eine Spiegelung um eine Achse der Wellenplatte. Somit kann eine Drehung der Polarisation erfolgen, linear polarisiertes Licht wird zu linear polarisiertem Licht mit einer anderen Schwingungsrichtung. Mit der Viertelwellenplatte ($\lambda/4$-Platte) kann aus linear polarisiertem Licht durch die Phasenverschiebung von $\pi/2$ elliptisch oder zirkular polarisiertes Licht hergestellt werden.

Während Polarisatoren meist eine Änderung der Gesamtintensität zur Folge haben, bleibt diese bei Wellenplatten unverändert.

2.1.5.2 Beschreibung der Polarisation im Müller-Formalismus

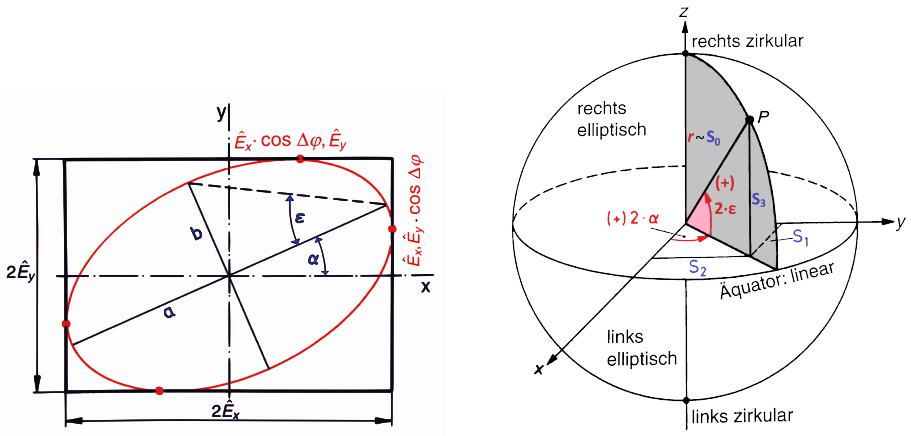


Abbildung 2.5: Ellipsenparameter α und ε [4]; Poincaré-Kugel [4]

Wird ein Koordinatensystem so festgelegt, dass die Strahlrichtung parallel zur z-Achse liegt, lässt sich die Polarisation in der x-y-Ebene veranschaulichen. E_x und E_y stellen dabei die Komponenten des elektrischen Feldes mit den Amplituden $\hat{E}_x$ und $\hat{E}_y$ dar.

Abb. 2.5 zeigt, wie vollständig, elliptisch polarisiertes Licht auf der Poincaré-Kugel dargestellt wird, wobei die Ellipsenparameter α und ε als Kugelkoordinaten (2α) und (2ε) verwendet werden. Damit sind automatisch auch die Spezialfälle der Ellipse (linear polarisiert: $b = 0$, $\varepsilon = 0$, zirkular polarisiert: $a = b$, $\varepsilon = \pi/4$) definiert. [4, 6]

Stokes-Vektor der polarisierten Strahlung Die kartesischen Koordinaten der Poincaré-Kugel S_1, S_2 und S_3 (Abb. 2.5) nennt man Stokes-Parameter. Diese können in einer einspaltigen Matrix $\begin{bmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \end{bmatrix}$ als Stokes-Vektor angegeben werden, der als Vektor keine direkte physikalische Bedeutung hat. S_0 ist die Gesamtintensität, bei vollständig polarisiertem Licht gilt $S_0 = \sqrt{S_1^2 + S_2^2 + S_3^2}$, ansonsten $S_0 > \sqrt{S_1^2 + S_2^2 + S_3^2}$. [4, 6]

Mit den Komponenten E_x und E_y und deren relative Phase ρ lautet die Definition des Stokes-Vektors:

$$\begin{bmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \end{bmatrix} = \begin{bmatrix} E_x^2 + E_y^2 \\ E_x^2 - E_y^2 \\ 2 * E_x * E_y * cos(\rho) \\ 2 * E_x * E_y * sin(\rho) \end{bmatrix}$$

Der Polarisationsgrad DOP (degree of polarisation) ist für teilweise polarisiertes Licht mit $S_0 > \sqrt{S_1^2 + S_2^2 + S_3^2}$ als das Verhältnis der Intensitäten der polarisierten und der gesamten Strahlung definiert: $DOP = \frac{I_{pol}}{I_{total}}$. Damit gilt

$$\begin{bmatrix} S_0 \\ S_1' \\ S_2' \\ S_3' \end{bmatrix} = (1 - DOP) \begin{bmatrix} S_0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + DOP \begin{bmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \end{bmatrix} \text{ mit } 0 \leq DOP \leq 1. \text{ [6]}$$

Müller-Matrizen für Polarisationsänderungen Die Polarisation eines Strahls und damit der ihn beschreibende Stokes-Vektor kann durch verschiedene polarisierende Materialien verändert werden, wobei ein linearer Zusammenhang zwischen den Stokes-Vektoren vor und nach der Änderung angenommen wird. Dann ist die Transformation durch eine 4x4-Matrix mit realen Einträgen m_{00} bis m_{33} möglich, welche als Müller-Matrix bezeichnet wird. [4]

$$\begin{bmatrix} S_{out\_0} \\ S_{out\_1} \\ S_{out\_2} \\ S_{out\_3} \end{bmatrix} = \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \\ m_{30} & m_{31} & m_{32} & m_{33} \end{bmatrix} \begin{bmatrix} S_{in\_0} \\ S_{in\_1} \\ S_{in\_2} \\ S_{in\_3} \end{bmatrix}$$

Die Müller-Matrix ist vom optischen Bauteil und dessen Winkellage abhängig. Für Elemente, die die Polarisation nicht beeinflussen (Linse, Filter) stellt sie eine 4x4 Einheitsmatrix dar, die den Stokes-Vektor unverändert belässt.

Müller-Matrix eines Polarisators/Analysators mit α als Polarisationswinkel zur x-Achse:

$$M_{Polarisator} = \frac{1}{2} \begin{bmatrix} 1 & cos(2\alpha) & sin(2\alpha) & 0 \\ cos(2\alpha) & cos^2(2\alpha) & sin(2\alpha) * cos(2\alpha) & 0 \\ sin(2\alpha) & sin(2\alpha) * cos(2\alpha) & sin^2(2\alpha) & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Müller-Matrix einer Wellenplatte mit relativer Phasenverschiebung/Retardation φ:

$$M_{Wellenplatte} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & cos(\varphi) & -sin(\varphi) \\ 0 & 0 & sin(\varphi) & cos(\varphi) \end{bmatrix}$$

2.2 Single Crystal Photo-Elastic Modulator (SCPEM)

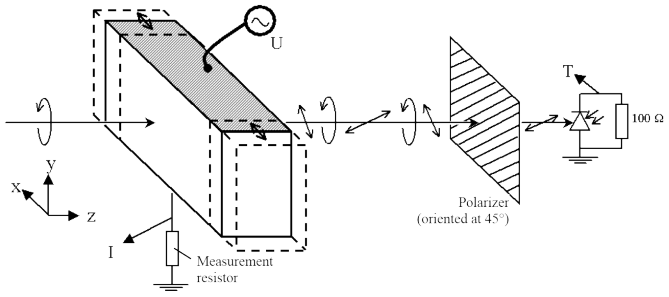


Abbildung 2.6: SCPEM in Standardkonfiguration [7]

Der Single Crystal Photo-Elastic Modulator (SCPEM) ist ein photoelastischer, transparenter Kristall, der elektrisch in einer seiner Resonanzfrequenzen angeregt wird. Durch die photoelastischen Eigenschaften werden bei der Schwingung eine veränderbare Doppelbrechung und damit eine Änderung der Polarisation der Strahlung erreicht. [8, 9, 10]

2.2.1 Funktion des SCPEM

Der SCPEM wird als Längsschwinger betrieben, je Periode erreicht er somit zweimal den spannungsfreien Ruhezustand, an dem die Strahlung nicht beeinflusst wird, und zweimal den Zustand höchster mechanischer Spannung, an dem die Phasenverschiebung der Polarisationsanteile maximal ist. Diese resultierende Phasenverschiebung wird auch als Retardation $\varphi(t)$ bezeichnet.

Mit der Erregerspannung $U(t)$ kann die maximale Retardation φ_{max} wie bei einer Viertel- oder Halbwellenplatte eingestellt werden, es sind aber auch alle Zustände dazwischen möglich. Über den Widerstand, der in Serie zum SCPEM angeschlossen ist (und weiterhin 100 Ω beträgt), kann der Strom durch den SCPEM bestimmt werden und damit seine Frequenz und aktuelle Phase. [8, 9]

2.2.2 Material

Es wird ein Kristall aus Lithiumtantalat ($LiTaO_3$) verwendet. Alternativ ist Lithiumniobat ($LiNiO_3$) als Kristallmaterial möglich, dieses hat jedoch eine viel größere Doppelbrechung als Lithiumtantalat und reagiert daher stärker auf Strahlwinkelabweichungen. Für sehr hochfrequente Anwendungen (im MHz-Bereich) ist Lithiumniobat aufgrund

der höheren Wellenausbreitungsgeschwindigkeit und damit höherer Schwingfrequenz bei gleichen Abmessungen besser geeignet. [7, 11]

2.2.3 Schwingung, Eigenfrequenz

Die Schwingung des Längsschwingers erfolgt analog zu Abb. 2.6 in Richtung der längsten Achse des Kristalls. Diese ist parallel zu den auftretenden Normalspannungen und damit zu der Richtung des zeitlich veränderten Brechungsindexes und liegt senkrecht zum Lichtstrahl. Die Frequenz ist abhängig von der Wellenausbreitungsgeschwindigkeit c und der Länge L des Kristalls:

$f = c/(2L)$

c hängt wiederum von der Steifigkeit E und der Dichte ρ des Materials ab:

$c = \sqrt{E/\rho}$

Das bedeutet, dass die Frequenz durch die Länge des eingesetzten Kristalls beeinflusst werden kann. [7, 11]

2.2.4 Elektrische Ansteuerung

Wie in Abb. 2.6 ersichtlich, benötigt der SCPEM eine Wechselspannungsansteuerung, wobei eine Rechteckspannung verwendet werden kann. Die Frequenz der Steuerspannung muss sehr genau mit der Eigenfrequenz übereinstimmen, damit eine ausreichende Schwingung und Retardation bei niedriger elektrischer Spannung erreicht wird, eine Rückkopplung des SCPEM-Stroms ist daher zur Stabilisierung der Frequenz sinnvoll. Konkrete Angaben zur Ansteuerung sind in Kap. 5 zu finden. [11]

2.2.5 Mögliche Einsatzmöglichkeiten eines SCPEM

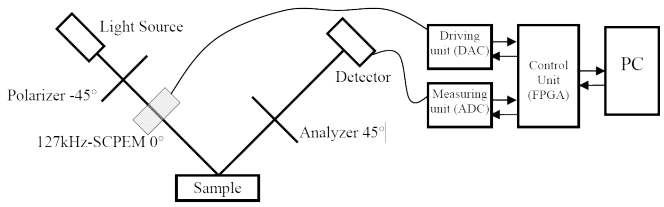


Abbildung 2.7: Ellipsometrie [12]

- Ellipsometrie (Abb. 2.7) wird für die Qualitätskontrolle von dünnen Schichten und die Beobachtung von physikalischen und chemischen Prozessen verwendet. Der SCPEM bildet dabei ein zentrales Element der Messung, indem er schnell aufeinanderfolgend mit hoher Wiederholgenauigkeit den Polarisationszustand des Strahls verändert, der dann von der Oberfläche reflektiert wird. Abb. 2.7 zeigt eine mögliche Anordnung, deren Auswerteeinheit sehr ähnlich zu der im Rahmen dieser Arbeit erstellten ist. Große Teile des entwickelten Messprogramms am STEMlab und am Computer sind für diese Aufgabe ebenso geeignet. [8, 12]

- Time Multiplexing [13]
- Q-Switching [7, 14, 15]
- Gepulster Laser [16]

2.2.6 Vorteil des SCPEM in dieser Anwendung

Mit dem SCPEM besteht die Möglichkeit, die Polarisation sehr schnell zu ändern und dadurch hochfrequente Messungen zu erreichen. Die Anwendung ist sowohl optisch als auch elektronisch vergleichsweise einfach zu realisieren, benötigt wenig Platz und arbeitet bei Raumtemperatur und geringen Spannungen. [12]

Alternativen wie die Pockels-Zelle benötigen hohe Spannungen, ein Flüssigkristall kann die gewünschte Geschwindigkeit nicht erreichen. [14, 15, 17]

2.2.7 Müller-Matrix des SCPEM

Um die Änderung der Polarisation mithilfe von Stokes-Vektoren berechnen zu können, wird die Müller-Matrix des SCPEM benötigt. Für die Gültigkeit wird die Ausrichtung der Längsachse des SCPEM parallel zur x-Achse vorausgesetzt.

In seiner Ausgangslage (Retardation $\varphi = 0$) wird die Polarisation nicht verändert:

$$M_{SCPEM}(\varphi = 0) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Der SCPEM verhält sich wie eine Wellenplatte mit zeitlich veränderlicher Phasenverschiebung $\varphi(t)$, Kap. 2.1.5.2. Damit ist die allgemeine Müller-Matrix in Abhängigkeit von $\varphi(t)$:

$$M_{SCPEM}(\varphi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & cos(\varphi(t)) & -sin(\varphi(t)) \\ 0 & 0 & sin(\varphi(t)) & cos(\varphi(t)) \end{bmatrix}$$

Darin ist der Fall $\varphi(t) = 0$ enthalten, weitere spezielle Fälle ergeben sich für $\varphi(t) = \pi/2$ und für $\varphi(t) = \pi$:

$$M_{SCPEM}(\varphi = \tfrac{\pi}{2}) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \; M_{SCPEM}(\varphi = \pi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

2.3 RedPitaya / STEMlab

Abbildung 2.8: Logo RedPitaya und Logo STEMlab [18]

RedPitaya ist ein 2013 gegründetes Unternehmen, welches die Messplatinen STEMlab 125-10 und STEMlab 125-14 inklusive zugehöriger Softwarelösungen anbietet.

Das verwendetet STEMlab 125-14 (Abb. 2.9) bietet einen Dual Core ARM Cortex A9 Prozessor und den FPGA (Field Programmable Gate Array) Zynq 7010 von Xilinx. Es stehen 512 MB RAM, USB 2.0, 1 Gbit Ethernet und diverse Ein- und Ausgänge für die Messgrößen zur Verfügung.

Die Messung erfolgt über die beiden RF-Inputs mit bis zu 125 MSamp/s und einer Auflösung von 14 bit bei einem Messbereich von ±1 V bzw. ±20 V.

Es steht ein Betriebssystem auf Linux-Basis als Download zur Verfügung, wodurch unter anderem das Ausführen von C- oder Python-Programmen sowie der Zugriff über SSH möglich ist. Die STEMlab-Software bietet eine Benutzeroberfläche und mehrere Programme (Oszilloskop, Spectrum Analyzer und einige weitere), die nach Verbindung mit einem Computer im Browser verwendet werden können.

Die Hauptanwendungsbereiche laut Webseite sind als multifunktionales Laborinstrument, als Datenerfassungsplattform und als SDR-Transceiver. [18, 19, 20, 21]

Genauere Informationen zu den spezifischeren Eigenschaften sind in den Kapiteln 6, 7.2 und 9.1.1 angegeben, ausführliche Informationen und Beispielprogramme sind in der STEMlab-Dokumentation [19] zu finden.

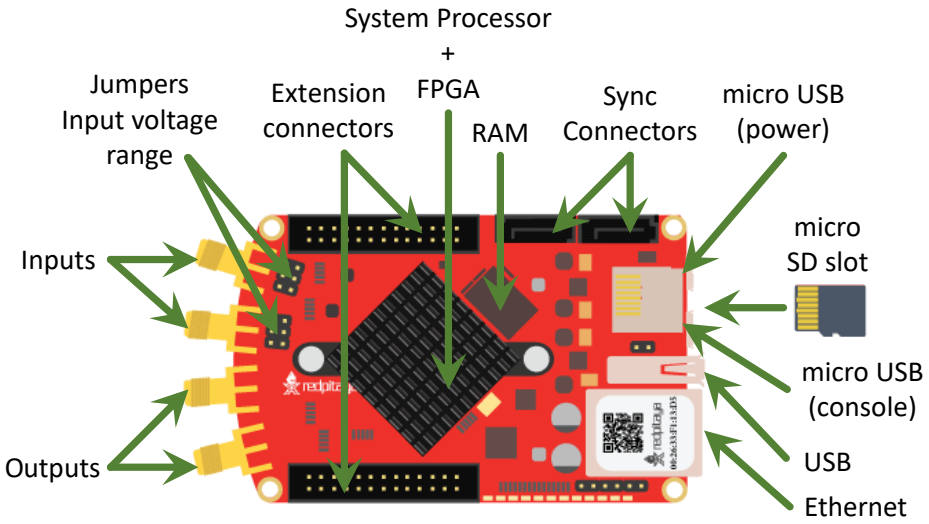


Abbildung 2.9: STEMlab Board mit den wichtigsten Komponenten [19]

3 Polarisationszustandsberechnung

Nach Bestimmung der Intensitäten aus dem Messsignal werden diese umgerechnet, um eine Kenngröße der Polarisationsrichtung des Infrarotanteils des Lichtes vor dem Messsystem zu erhalten. Für die Berechnung sind die Intensitäten bei den Retardationen $\varphi(t) = 0$ und $\varphi(t) = \varphi_{max}$ sowie der Analysatorwinkel α bekannt. Die Ermittlung dieser Intensitäten wird in Kap. 7.3 beschrieben.

3.1 Müllermatrix des Systems SCPEM und Analysator

Die Konstanten c und s in Abhängigkeit vom Analysatorwinkel α werden zur übersichtlicheren Darstellung verwendet:

$$c = cos(2\alpha), s = sin(2\alpha)$$

Der Eingangs-Stokesvektor ist über die Intensitätsanteile E_x und E_y definiert durch:

$$\begin{bmatrix} S_{In0} \\ S_{In1} \\ S_{In2} \\ S_{In3} \end{bmatrix} = \begin{bmatrix} E_x^2 + E_y^2 \\ E_x^2 - E_y^2 \\ 2 * E_x * E_y * cos(\rho) \\ 2 * E_x * E_y * sin(\rho) \end{bmatrix}$$

Dabei wird für alle folgenden Berechnungen zirkular polarisiertes Licht ausgeschlossen und damit gilt:

$$S_{In3} = 0;\ \rho = 0,\pi;\ \rho \text{ wird mit } \rho = 0 \text{ festgelegt}$$

Der Zusammenhang zwischen Eingangs- und Ausgangsstokesvektor ist:

$$S_{Out} = M_{Analysator} * M_{SCPEM} * S_{In}$$

Mit den Matrizen aus Kap. 2.1.5.2 und Kap. 2.2.7 ergibt sich:

$$\begin{bmatrix} S_{Out0} \\ S_{Out1} \\ S_{Out2} \\ S_{Out3} \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & c & s * cos(\varphi) & -s * sin(\varphi) \\ c & c^2 & c * s * cos(\varphi) & -c * s * sin(\varphi) \\ s & c * s & s^2 * cos(\varphi) & -s^2 * sin(\varphi) \\ 0 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} S_{In0} \\ S_{In1} \\ S_{In2} \\ S_{In3} \end{bmatrix}$$

3.2 Bestimmungsfunktionen für unterschiedliches SCPEM-Verhalten

Es sollen E_x und E_y ermittelt werden. Die Absolutwerte sind dabei nicht interessant, wichtig ist das Verhältnis $p = E_y/E_x$ der beiden, da dieses einer Polarisationsrichtung entspricht. Da das Verhältnis zwischen $-\infty$ und $+\infty$ liegen kann und damit beim effizienten Speichern und Übertragen als Integer-Zahl Probleme auftreten, wird stattdessen der Polarisationswinkel ermittelt, der sich durch den Arkustangens des Verhältnisses ergibt. Dadurch ist der Bereich auf -180° bis 180° bzw. $-\pi$ bis π beschränkt, wenn mit einem Polarisator getestet wird, stimmt das Ergebnis mit der Polarisatorskala überein.

Abhängig davon, wie stark der SCPEM angesteuert wird, unterscheidet sich sein Verhalten. Für einige spezielle Fälle lassen sich einfachere Lösungen finden, das Messprogramm verwendet jedoch die allgemeine Lösung, die für eine beliebige bekannte Retardation φ_{max} gilt (Kap. 3.2.3).

Die gemessene Intensität an der Photodiode entspricht S_{Out0}, aus der ersten Zeile der Matrixmultiplikation erhält man:

$$S_{Out0} = S_{In0} + c * S_{In1} + s * cos(\varphi) * S_{in2} - s * sin(\varphi) * S_{in3}$$

Setzt man die Definition des Stokes-Vektors ein und eliminiert E_y durch $E_y = p * E_x$, erhält man:

$$S_{In0} = E_x^2 * (1 + p^2),\ S_{In1} = E_x^2 * (1 - p^2),\ S_{In2} = 2 * E_x^2 * p$$

$$S_{Out0} = \frac{E_x^2}{2} * ((1 + p^2) + c * (1 - p^2) + 2 * s * p * cos(\varphi(t)))$$

3.2.1 SCPEM als Halbwellenplatte

Der SCPEM hat bei $\varphi_{max} = \pi$ das Verhalten eines $\lambda/2$-Plättchens.

3.2.1.1 Herleitung der Bestimmungsfunktion

Setzt man für $\varphi(t) = 0, \pi/2, \pi$ ein, erhält man die Intensitätswerte S_{Out0}, die den gemessenen Intensitäten I entsprechen:

$$S_{Out0}|_{\varphi=0} = I_0 = \frac{E_x^2}{2} * ((1 + p^2) + c * (1 - p^2) + 2 * s * p)$$

$$S_{Out0}|_{\varphi=\pi/2} = I_{\pi/2} = \frac{E_x^2}{2} * ((1 + p^2) + c * (1 - p^2))$$

$$S_{Out0}|_{\varphi=\pi} = I_\pi = \frac{E_x^2}{2} * ((1 + p^2) + c * (1 - p^2) - 2 * s * p)$$

Mit den bekannten Intensitäten kann E_x und damit die Abhängigkeit von der absoluten Intensität eliminiert werden, es sollen zur Ermittlung von p nur Verhältnisse Q von $S_{Out1}(\varphi)$ bei verschiedenen Retardationswinkeln $\varphi = 0, \pi$ verwendet werden:

$$I_0 + I_\pi = E_x^2 * ((1 + p^2) + c * (1 - p^2))$$

$$I_0 - I_\pi = 2 * E_x^2 * s * p$$

$$Q = \frac{I_0+I_\pi}{I_0-I_\pi} = \frac{p^2*(1-c)+(1+c)}{2*s*p}$$

Durch Umformung ergibt sich für das Verhältnis $p = E_y/E_x$:

$$p = \frac{s*Q\pm\sqrt{c^2+s^2*Q^2-1}}{1-c}$$

Bestimmung des Polarisationswinkels:

$$\gamma_{In} = \arctan(\frac{E_y}{E_x}) = \arctan(p)$$

Es sind dabei einige Fälle zu unterscheiden: $I_0 + I_\pi = 0$ (führt zur Division durch 0), Q>0, Q<0, siehe Kap. 3.2.3

3.2.1.2 Prüfung auf Halbwellenplattenverhalten

Ein großer Vorteil der großen Retardation ist, dass die Regelung des SCPEM auf $\varphi_{max} = \pi$ (bzw. knapp darüber) leicht möglich ist. Bei einer Retardation etwas größer als 180° bildet sich ein weiteres Minimum/Maximum aus, wie Abb. 3.1 zeigt. Dieses Verhalten kann für die Regelung des SCPEM verwendet werden.

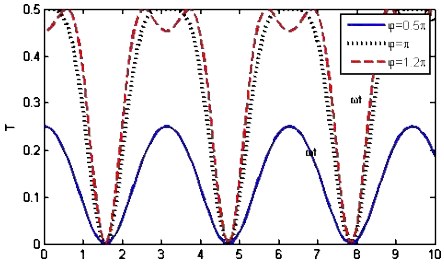


Abbildung 3.1: Intensitäts-Signal bei Retardation 90°, 180°, 216° [10]

3.2.2 SCPEM als Viertelwellenplatte

Der SCPEM hat bei $\varphi_{max} = \pi/2$ das Verhalten eines $\lambda/4$-Plättchens bzw. Zirkulationspolarisators.

3.2.2.1 Herleitung der Bestimmungsfunktion

Analog zu Kap. 3.2.1 lässt sich die Polarisation ermitteln, wenn die Retardation nur $\varphi_{max} = \pi/2$ erreicht. Der SCPEM wirkt dann wie ein Zirkulationspolarisator, vollständig polarisiertes Licht wird zu zirkular polarisiertem Licht:

$$S_{Out0}|_{\varphi=0} = I_0 = \frac{E_x^2}{2} * ((1+p^2) + c*(1-p^2) + 2*s*p)$$

$$S_{Out0}|_{\varphi=\pi/2} = I_{\pi/2} = \frac{E_x^2}{2} * ((1+p^2) + c*(1-p^2))$$

$$S_{Out0}|_{\varphi=-\pi/2} = I_{\pi/2} = \frac{E_x^2}{2} * ((1+p^2) + c*(1-p^2))$$

$$Q = \frac{I_{\pi/2}}{I_0 - I_{\pi/2}} = \frac{p^2 * (1-c) + (1+c)}{2 * s * p}$$

$$p = \frac{\pm s * Q + \sqrt{c^2 + s^2 * Q^2 - 1}}{(1-c)}$$

$$\gamma_{In} = \arctan(\frac{E_y}{E_x}) = \arctan(p)$$

3.2.2.2 Prüfung auf Viertelwellenplattenverhalten

Es ist schwieriger, den SCPEM auf diesen Arbeitsbereich zu stabilisieren. Bei einer Retardation von $\varphi = \pi/2$ ändert sich bei konstanter Gesamtintensität und Änderung der Polarisationsrichtung vor dem SCPEM die gemessene Intensität nicht, da die Strahlung nach dem SCPEM in jedem Fall zirkular polarisiert ist (bei vollständig polarisierter Eingangsstrahlung). Da die Intensität während des Versuchs schwanken kann, ist es schwierig festzustellen, woher eine Intensitätsänderung kommt. Unter einigen Bedingungen könnte die Retardation aber geregelt werden:

- es gibt keinen Zusammenhang zwischen Gesamtintensität und Polarisationsrichtung: das ist erst am Messobjekt zu prüfen
- die Polarisationsrichtung ändert sich schnell gegenüber der Spannungsabhängigkeit des SCPEM: der SCPEM sollte grundsätzlich nur sehr kleine Änderungen über die Zeit aufweisen, sodass die Polarisationsrichtung ohne Kompensation schon grob stimmt, es soll durch den Algorithmus nur eine zusätzliche Feinjustierung erfolgen (diese Bedingung ist nach bisherigen Erfahrungen erfüllt)

Es ergibt sich dann folgende Möglichkeit, die Retardation auf $\pi/2$ zu regeln:

- es wird eine sich ändernde Polarisation bei gleichbleibender Intensität gemessen
- über mehrere Perioden wird der Intensitätswert bei $\varphi = \pi/2$ beobachtet und mit jeweils zugehörigem berechnetem Polarisationswinkel gespeichert
- danach wird die Korrelation zwischen jeweiligem Polarisationswinkel und diesem Wert berechnet
- es sollte im Idealfall keine Korrelation geben, da die Werte bei $\varphi = \pi/2$ unabhängig vom Polarisationswinkel sind
- eine gefundene Korrelation (positiv oder negativ) kann ein Hinweis auf eine nach oben oder unten verschobene Retardation sein, die so auf $\varphi = \pi/2$ zurückgeregelt werden kann

Diese Vorgangsweise kann genutzt werden, um bei Betrachtung der Signale am Oszilloskop manuell eine Retardation nahe 90° zu erreichen. Dazu wird der Polarisator ständig gedreht, und die Spannung der SCPEM-Ansteuerung so angepasst, dass bestimmte Punkte im Intensitäts-Signal bei jeder Polarisatorstellung gleich bleiben. Dabei sind keine Berechnungen nötig und die Einstellung kann für Testzwecke schnell erfolgen. Eine automatisierte Feststellung ist nicht vorgesehen, da in Kap. 3.2.3.2 eine allgemeinere Retardationsbestimmungsmöglichkeit gezeigt wird.

3.2.3 SCPEM mit beliebiger, bekannter Retardation

Wird der SCPEM nicht mit $\varphi_{max} = \pi$ oder $\varphi_{max} = \pi/2$ betrieben, sondern mit einer beliebigen, bekannten Retardation $\varphi_{max} = \varphi_{max} > 0$, so unterscheidet sich sein Verhalten nicht grundlegend, es entfällt aber der Vorteil des Wegfallens der Sinus- und Kosinus-Terme beim Lösen der Bestimmungsgleichung. Es zeigt sich, dass das Gleichungssystems dennoch gelöst werden kann.

Das Messprogramm verwendet diesen Algorithmus zur Polarisationsbestimmung, φ_{max} wird vor jeder Messung bestimmt und festgelegt.

3.2.3.1 Herleitung der Bestimmungsfunktion

Die Vorgangsweise ist ähnlich wie beim Halb- oder Viertelwellenverhalten. Um die Schritte und das Ergebnis übersichtlich zu halten, werden einige neue Größen definiert. Diese sind so festgelegt, dass bei der späteren Ausführung des Algorithmus möglichst wenige Rechenschritte nötig sind.

$$S_{Out0}|_{\varphi=0} = I_0 = \frac{E_x^2}{2} * ((1+p^2) + c * (1-p^2) + 2 * s * p)$$

$$S_{Out0}|_{\varphi=\varphi_{max}} = I_{\varphi_{max}} = \frac{E_x^2}{2} * ((1+p^2) + c * (1-p^2) + 2 * s * p * cos(\varphi_{max}))$$

Aus den beiden Intensitäten I_0 und I_{φ_max} lässt sich ein Quotient zur Eliminierung von E_x bilden, mit $c' = cos(\varphi_{max})$, $C = 1 - c$ und $C' = 1 - c'$:

$$Q = \frac{I_0}{I_0 - I_{\varphi max}} = \frac{p^2*(1-c)+(1+c)+2*s*p}{2*s*p*(1-c')} = \frac{p^2*C+p*(2*s)+(2-C)}{p*(2*s*C')}$$

Das Gleichungssystem hat die folgende Form einer quadratischen Gleichung mit dem Ergebnis:

$$Q = \frac{p^2 Z+pY+X}{pW};\ p = \frac{QW-Y\pm\sqrt{Q^2W^2-2QWY-4XZ+Y^2}}{2Z},$$

durch Einsetzen und Umformen ergibt sich das Verhältnis $p = E_y/E_x$ zu:

$$p = \frac{s*(Q*C'-1)\pm\sqrt{C^2-2*C+s^2(Q*C'-1)^2}}{C}$$

bzw. mit $Q' = s * (Q * C' - 1)$:

$$p = \frac{Q'\pm\sqrt{C^2-2*C+Q'^2}}{C}$$

$I_0 + I_\pi = 0$ führt bei der Berechnung von Q zu einer Division durch 0 und ist zu verhindern. Konstante Intensität bedeutet entweder unpolarisiertes Licht oder eine Polarisation von $n * \pi/2$. Da der Messbereich um 0° sein soll, wird in diesem Fall $\gamma_{In} = 0°$ zurückgegeben.

Der gewünschte Messbereich bestimmt auch das Vorzeichen des Wurzelterms in der Bestimmungsgleichung:

$$Q' < 0 : \; p = \frac{Q' + \sqrt{C^2 - 2*C + Q'^2}}{C}$$

$$Q' \geq 0 : \; p = \frac{Q' - \sqrt{C^2 - 2*C + Q'^2}}{C}$$

Schließlich ergibt sich für den Polarisationswinkel γ_{In} der Strahlung vor dem SCPEM:

$$\gamma_{In} = \arctan(\frac{E_y}{E_x}) = \arctan(p)$$

3.2.3.2 Bestimmung der Retardation φ_{max}

Es gibt 2 Möglichkeiten zur Bestimmung der Retardation.

Kennlinie SCPEM-Strom-Retardation Im ersten Fall muss die Kennlinie des SCPEM bekannt sein, also die max. Retardation in Abhängigkeit vom SCPEM-Strom bei einer bestimmten Wellenlänge der Strahlung.

Da das Strom-Signal schon für die Trigger-Funktion benötigt und gemessen wird, lässt sich auch die Amplitude leicht ermitteln. Daraus kann dann mit der Kennlinie die aktuelle, maximale Retardation berechnet werden. Einflüsse auf die Kennlinie wie Temperatur oder zeitliche Änderungen werden nicht berücksichtigt.

Für die spätere Verwendung ist diese Variante gut geeignet, wenn am Messsystem keine Änderungen mehr vorgenommen werden und somit die Kennlinie gleich bleibt. Diese kann zum Beispiel nach der folgenden, zweiten Möglichkeit ermittelt werden.

Invertierung der Bestimmungsgleichung Um ohne Kennlinie die Retardation bestimmen zu können, wird eine andere Vorgehensweise benötigt. Ist die Polarisation vor dem SCPEM bekannt, kann aus dem Diodensignal bzw. den Intensitäten zu bestimmten Zeitpunkten die Retardation bestimmt werden.

Die Eingangspolarisation wird mit einem Polarisator festgelegt, bei Verwendung des Messprogramms wird eine Polarisation von 45° vorausgesetzt. Die Lösung ist für andere Polarisationen genauso möglich, es wird dann ein anderes $p = tan(\gamma_{In})$ verwendet.

Bei Polarisation 45° gilt $E_y = E_x$ bzw. $p = tan(\gamma_{In}) = 1$. Q kann aus den Intensitäten ermittelt werden, s ist bekannt, dadurch kann C' bzw. φ_{max} bestimmt werden:

$$Q = \frac{I_0}{I_0 - I_{\varphi max}} = \frac{p^2*C - C + 2 + 2*s*p}{2*s*p*C'} = \frac{C - C + 2 + 2*s}{2*s*C'} = 2 * \frac{1+s}{s*C'}$$

$$C' = \frac{2}{Q} * \frac{1+s}{s} = 1 - c' = 1 - cos(\varphi_{max})$$

Für das Programm reicht die Bestimmung von C' aus, da diese Variable in der Polarisationsberechnung verwendet wird. Die Retardation $\varphi_{max} = arccos(1 - C')$ wird nur für Informationszwecke direkt berechnet.

Wird nun mit dieser Methode bei bekannter Polarisation für unterschiedliche SCPEM-Ansteuerspannungen die Retardation ermittelt, lässt sich mit dem zugehörigem SCPEM-Spitzenstrom eine Kennlinie ermitteln, welche die Bestimmung der Retardation nach Variante 1 ermöglicht.

3.3 Messbereich und Fehlerabschätzung des Algorithmus

3.3.1 Theoretischer Messbereich

Da unterschiedliche Polarisationen zu gleichem Ausgangssignal führen können, unterliegt das Messsystem Grenzen, in denen es das richtige Ergebnis liefert. Ist die Polarisation außerhalb dieser Grenzen, wird ein falsches Ergebnis mitgeteilt. Der Messbereich ist so festgelegt, dass er sich symmetrisch um 0° befindet.

Die Grenzen dieses Messbereiches sind von der Stellung des Analysators (Winkel α) abhängig und betragen:

$\gamma_{In\_min} = -\pi/2 + \alpha$; $\gamma_{In\_max} = \pi/2 - \alpha$; $\gamma_{In\_max} - \gamma_{In\_min} = \pi - 2 * \alpha$;

Der Messbereich lässt sich also direkt über den Analysatorwinkel beeinflussen, bei der Standardstellung von +45° ist die Messung von -45° bis +45° möglich. In der Theorie ist bei $\alpha = \lim_{x \to 0} x$ ein Messbereich von -90° bis +90° möglich. Real setzen die Auflösung des Analog-Digitalwandlers, Signalstörungen und eine gewünschte Fehlertoleranz diese Grenze niedriger, da am Rand des Messbereiches die Auswertungsfehler größer werden, wie folgend gezeigt wird.

3.3.2 Realer Messbereich

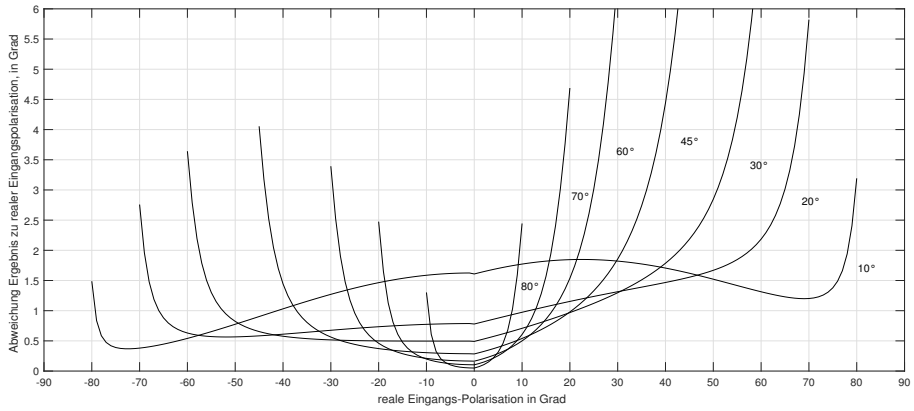


Abbildung 3.2: Messfehler bei 1% Einlesefehler für unterschiedliche Analysatorstellungen, Retardation 90° (Anh. B.2.1)

Um den tatsächlich sinnvoll nutzbaren Messbereich zu erhalten, wird der Algorithmus numerisch analysiert. Dabei wird jeweils der niedrigere der beiden Intensitätswerte um 1% des größeren Wertes variiert, wodurch die maximale Auswirkung auf das Ergebnis erreicht wird. Eine multiplikative Änderung würde in den Bereichen, wo die Intensität nahe 0 ist, keine Abweichungen erzielen. Die Fehler durch äußere Störeinflüsse sind als additiv zu betrachten, daher wird zu Vergleichszwecken die jeweils größere Intensität für die Fehlergröße verwendet.

Die Auswirkungen bei einem Intensitäts-Messfehler von 1% auf das Ergebnis bei einer Retardation von 90° und unterschiedlichen Analysatorstellungen sind in Abb. 3.2 veranschaulicht.

Die Daten in diesem Diagramm beziehen sich ausschließlich auf den Algorithmus, der mit rechnerisch ermittelten, idealen Eingangsgrößen Strahlungsintensität I_0 und $I_{\varphi max}$ arbeitet und als Ausgangsgröße einen Polarisationswinkel liefert.

Für Messergebnisse in guter Qualität sollten die Randbereiche gemieden werden. Ein Überschreiten des Messbereiches kann nicht erkannt werden, es wird ein Ergebnis geliefert, das jedoch nicht mit der tatsächlichen Polarisation übereinstimmt.

Neben dem Analysatorwinkel hat die verwendete Retardation Einfluss auf die Messgenauigkeit und die reale Messgrenze, wie Abb. 3.3 verdeutlicht.

Durch eine höhere Retardation kann generell eine bessere Genauigkeit über einen weiten Messbereich erreicht werden. Die Ecke in den Graphen bei 0° entsteht durch den Wechsel des um 1% veränderten Intensitätswerts, denn bei 0° sind beide gerade gleich groß. Eine Retardation deutlich unter 90° kann den nutzbaren Messbereich einschränken.

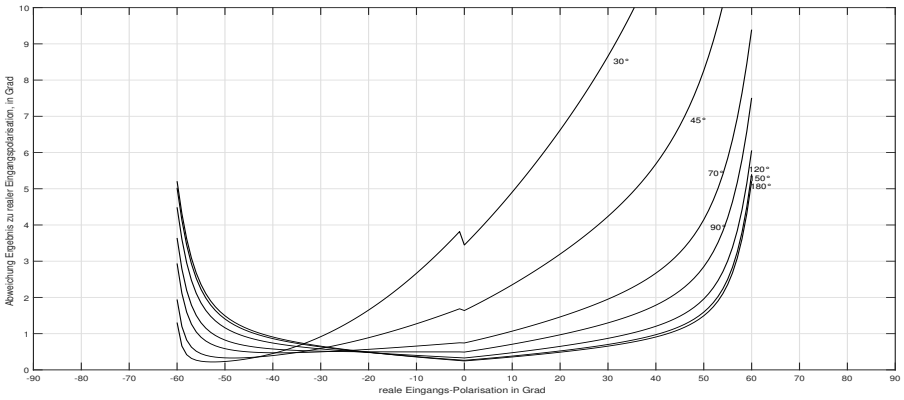


Abbildung 3.3: Messfehler bei 1% Einlesefehler für unterschiedliche max. Retardationen, Analysatorstellung 30° (Anh. B.2.2)

3.3.3 Empfohlene Einstellung des Messbereichs durch Analysatorwinkel

Die Analyse in Kap. 3.3.2 betrachtet nur eine Möglichkeit, wie sich ein Fehler in den gemessenen Daten auf das Ergebnis auswirkt.

Die Einstellungen sollen abhängig von den zu messenden Daten festgelegt werden. Eine höhere Messungenauigkeit in den gezeigten Diagrammen wirkt sich im Ergebnis durch einen größeren Rauschanteil in den Ergebnissen aus. Dieser kann durch Mittelung verringert werden, sodass der reale Messbereich für langsame Messungen vergrößert werden kann. Als Empfehlung für allgemeine Messungen kann ein Analysatorwinkel von 20° - 30° verwendet werden. Damit ist die Messung von -45° bis +45° Grad mit ausreichendem Abstand zur Messgrenze möglich. Ein Genauigkeitsverlust gegenüber der

45°-Analysatorstellung ist dabei im Bereich um 0° vorhanden, außerhalb ±30° ist die Messung mit 45° Analysator jedoch nicht mehr empfehlenswert.

Generell hat es sich bewährt, bei unbekannten Polarisationen mit einem großen Messbereich zu starten (Analysatorwinkel klein), um diesen dann bei Bedarf etwas zu verkleinern (Analysatorwinkel erhöhen).

3.3.4 Auswirkung von teilpolarisiertem Licht

In den bisherigen Berechnungen ist unter der Annahme von vollständig polarisiertem Licht vor dem SCPEM $S_0 = \sqrt{S_1^2 + S_2^2 + S_3^2}$, mit $S_3 = 0$ bleiben so zwei Unbekannte. Ist das Licht nur teilpolarisiert und damit $S_0 > \sqrt{S_1^2 + S_2^2 + S_3^2}$, kann S_0 nicht mehr aus den übrigen Stokes-Elementen gebildet werden und enthält einen unpolarisierten Anteil. Mit der Definition des DOP aus Kap. 2.1.5.2 gilt für den Stokes-Vektor $S = (1 - DOP) \begin{bmatrix} S_0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + DOP \begin{bmatrix} S_0 \\ S_1 \\ S_2 \\ S_3 \end{bmatrix}$. [6]

3.3.4.1 Bekannter, konstanter Polarisationsgrad

Bei bekanntem Polarisationsgrad DOP lässt sich einsetzen:

$$\begin{bmatrix} S_{In0} \\ S_{In1} \\ S_{In2} \\ S_{In3} \end{bmatrix} = \begin{bmatrix} DOP * \left(E_x^2 + E_y^2\right) + (1 - DOP) * \left(E_x^2 + E_y^2\right) \\ DOP * \left(E_x^2 - E_y^2\right) \\ DOP * 2 * E_x * E_y * cos(\rho) \\ DOP * 2 * E_x * E_y * sin(\rho) \end{bmatrix}$$

$$S_{Out0} = S_{In0} + c * S_{In1} + s * cos(\varphi) * S_{in2} - s * sin(\varphi) * S_{in3}$$

Wird zirkular polarisiertes Licht wieder ausgeschlossen ($S_{In3} = 0$), so ergeben sich damit wie in Kap. 3.2.3 folgende Gleichungen:

$$S_{In0} = DOP * E_x^2 * (1 + p^2) + (1 - DOP) * E_x^2 * (1 + p^2),\ S_{In1} = DOP * E_x^2 * (1 - p^2),$$
$$S_{In2} = DOP * 2 * E_x^2 * p$$

$$S_{Out0} = DOP * \tfrac{E_x^2}{2} * ((1 + p^2) + c * (1 - p^2) + 2 * s * p * cos(\varphi)) + (1 - DOP) * \tfrac{E_x^2}{2} * (1 + p^2)$$

$$S_{Out0}|_{\varphi=0} = I_0 = DOP * \tfrac{E_x^2}{2} * ((1 + p^2) + c * (1 - p^2) + 2 * s * p + \tfrac{(1-DOP)}{DOP} * (1 + p^2))$$

$$S_{Out0}|_{\varphi=\varphi_{max}} = DOP * \tfrac{E_x^2}{2} * ((1+p^2) + c * (1-p^2) + 2 * s * p * cos(\varphi_{max}) + \tfrac{(1-DOP)}{DOP} * (1+p^2))$$

Die Intensität des unpolarisierten Anteils passiert den SCPEM unverändert und wird beim Polarisieren im Analysator halbiert. S_{Out0} erhöht sich also jeweils um einen von DOP abhängigen Anteil, unabhängig von der Retardation φ.

Das Gleichungssystem lässt sich analog zu Kap. 3.2.3 auflösen. Zuerst wird zur Elimination von $DOP * E_x^2$ ein Quotient Q gebildet, mit $cos(\varphi_{max}) = c'$, $C = 1-c$, $C' = 1-c'$:

$$Q = \frac{I_0}{I_0 - I_{\varphi max}} = \frac{p^2*(1-c)+(1+c)+2*s*p+\frac{(1-DOP)}{DOP}*(1+p^2)}{2*s*p*(1-c')}$$

$$Q = \frac{p^2*(C+\frac{(1-DOP)}{DOP})+p*(2*s)+2-C+\frac{(1-DOP)}{DOP}}{p*(2*s*C')}$$

$$p = \frac{s*(Q*C'-1)\pm\sqrt{C^2-2*C+s^2(Q*C'-1)^2-(1-DOP)^2-2*\frac{(1-DOP)}{DOP}}}{C+\frac{(1-DOP)}{DOP}}$$

bzw. mit $Q' = s*(Q*C'-1)$ und $D = \frac{(1-DOP)}{DOP}$:

$$p = \frac{Q'\pm\sqrt{C^2-2*C+Q'^2-D^2-2*D}}{C+D}$$

Für die Bestimmung des Polarisationswinkels gilt weiterhin:

$$\gamma_{In} = \arctan(\frac{E_y}{E_x}) = \arctan(p)$$

Die Lösung für teilweise polarisiertes Licht bei bekanntem Polarisationsgrad DOP ist nur unwesentlich aufwendiger als bei DOP=1, der verwendbare Messbereich wird jedoch abhängig von DOP leicht eingeschränkt (Abb. 3.4).

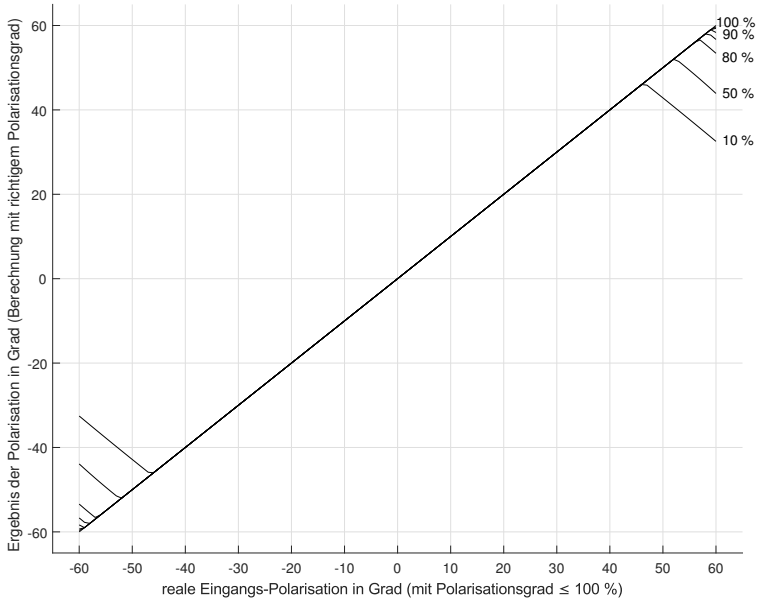


Abbildung 3.4: Eingeschränkter Messbereich bei bekanntem Polarisationsgrad ≤ 1, Analysatorstellung 30°, Retardation 90° (Anh. B.2.3)

3.3.4.2 Von Polarisation abhängiger Polarisationsgrad

Ist der Zusammenhang zwischen dem Polarisationswinkel, der damit verbundenen Neigung der Messfläche und dem zugehörigen Polarisationsanteil durch Messungen bekannt als $DOP(p)$ bzw. $D(p) = \frac{(1-DOP(p))}{DOP(p)}$, kann diese Funktion in die Bestimmungsgleichung von p eingesetzt werden:

$$p = \frac{Q' \pm \sqrt{C^2 - 2*C + Q'^2 - D(p)^2 - 2*D(p)}}{C + D(p)}$$

Da $DOP(p)$ oder $DOP(\gamma_{In})$ kein einfacher Zusammenhang ist (zum Beispiel als Interpolation aus mehreren zuvor bestimmten Punkten einer Kennlinie), ist eine analytische dieses Ausdrucks nach p nicht praktikabel.

Die Gleichung kann mit einem iterativen, numerischem Verfahren gelöst werden, als Startwert für p kann das p der letzten Berechnung verwendet werden, da aufgrund der hochfrequenten Abtastung die Polarisation sich nicht stark ändern sollte.

Unabhängig von den Schwierigkeiten der Bestimmung der Funktion $DOP(p)$ steigt die Berechnungsdauer mit gewünschter Genauigkeit durch mehrere Iterationen an, die numerische Stabilität des Systems ist unter anderem von der Struktur der Funktion $DOP(p)$ abhängig und kann daher nur begrenzt getestet werden. Alternativ dazu kann der Polarisationsgrad völlig aus der Betrachtung entfernt werden, wie in Kap. 3.5.

3.3.4.3 Fehler durch unpolarisierten Anteil

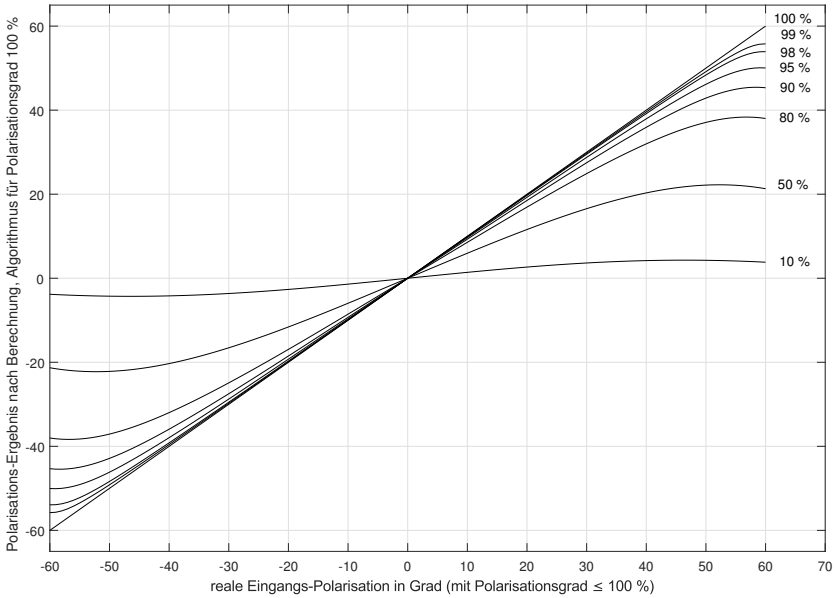


Abbildung 3.5: Messergebnis bei unberücksichtigtem Polarisationsgrad ≤ 1, Analysatorstellung 30°, Retardation 90° (Anh. B.2.4)

Mit den Erkenntnissen aus 3.3.4.1 wird numerisch ermittelt, wie sich ein von 1 abweichender Polarisationsgrad auf das Ergebnis auswirkt, wenn mit dem für DOP=1 definierten Algorithmus gerechnet wird (Kap. 3.2.3).

Abb. 3.5 zeigt diesen Zusammenhang. Für Polarisationsgrade nahe an 1 ist die Abweichung gering. Kleine Polarisationswinkel mit einem hohen Anteil an unpolarisierter Strahlung (zum Beispiel 50%) lassen eine Messung zu, die Abweichungen sind dabei jedoch erheblich. Da der Zusammenhang weitgehend streng monoton steigend ist, könnte die Korrektur für einen bestimmten Polarisationsgrad mit einer Kalibriertabelle vorgenommen werden.

3.4 Überlegungen zur Berechnung der Schneidflächenneigung

Ohne genauer auf die physikalischen Zusammenhänge einzugehen, wird der Zusammenhang der Emissionskoeffizienten mit dem Neigungswinkel darauf untersucht, ob eine eindeutige Lösungsfindung möglich ist. Dabei werden die Zusammenhänge vereinfacht, es wird kein unpolarisierter Anteil im Licht betrachtet und keine sonstigen Störeinflüsse.

Die in [3] auf Seite 4 angegebenen, auf den Fresnelschen Gleichungen beruhenden Funktionen und das Diagramm der Emissionskoeffizienten (Abb. 2.3) auf Seite 5 werden als Grundlage für die folgenden Überlegungen und Berechnungen verwendet.

Zuerst werden die Gleichungen für die Emissionskoeffizienten in Abhängigkeit von der Neigung δ (1.5) und (1.6) übernommen und angepasst. Es wird der materialspezifische Emissionskoeffizient ε in die Gleichungen aufgenommen und der Neigungswinkel von β auf δ umbenannt. Für ϵ_s werden die Brechungsindizes n_1 und n_2 richtiggestellt, diese sind in [3] vertauscht.

$$\epsilon_s(\delta) = \varepsilon \left(1 - \left| \frac{n_1 \sqrt{1 - \frac{n_2^2 sin(\delta)^2}{n_1^2}} - n_2 cos(\delta)^2}{n_1 \sqrt{1 - \frac{n_2^2 sin(\delta)^2}{n_1^2}} + n_2 cos(\delta)^2} \right|^2 \right)$$

$$\epsilon_p(\delta) = \varepsilon \left(1 - \left| \frac{n_2 \sqrt{1 - \frac{n_2^2 sin(\delta)^2}{n_1^2}} - n_1 cos(\delta)^2}{n_2 \sqrt{1 - \frac{n_2^2 sin(\delta)^2}{n_1^2}} + n_1 cos(\delta)^2} \right|^2 \right)$$

Darin sind $\epsilon_s(\delta)$ und $\epsilon_p(\delta)$ die Emissionskoeffizienten parallel und senkrecht zur Neigungsachse, n_1 der Brechungsindex des Materials (z.B. Stahl) und n_2 der Brechungsindex des umgebenden Mediums (Luft).

Da keine Daten für die verwendeten Kennwerte vorliegen, kann durch Einsetzen verschiedener Werte mit den Brechungsindizes $n_1 = 6.8$ und $n_2 = 1$ sowie dem Emissionskoeffizienten $\varepsilon = 0{,}78$ die Emissionskoeffizientskurve in Abhängigkeit von δ exakt nachgebildet werden, siehe Abb.3.6 mit der schwarz strichlierten, berechneten Kurve über der vorgegeben Kurve aus [3]. Damit liegen die Daten der Emissionskoeffizientskurve mit realistischen Werten numerisch vor und es können weitere Überlegungen vorgenommen werden.

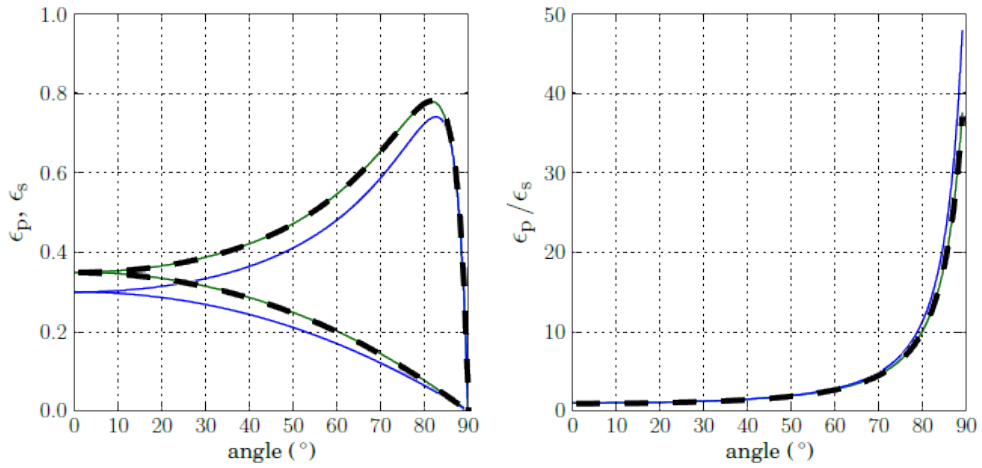


Abbildung 3.6: Emissionskoeffizienten, Anpassung der Kennwerte durch Überlagerung (Anh. B.2.5) [3]

Der Polarisationswinkel ist durch $\gamma_{In} = \arctan(\frac{E_y}{E_x}) = \arctan(p)$ festgelegt. Setzt man eine Abhängigkeit von E_y und ϵ_s sowie E_x und ϵ_p voraus, so erhält man mit den Emissionskoeffizienten $\gamma_{In}(\delta) = \arctan(\frac{\epsilon_s(\delta)}{\epsilon_p(\delta)})$ oder $\frac{\epsilon_s(\delta)}{\epsilon_p(\delta)} = p$. Dabei wird der materialspezifische Emissionskoeffizient ε eliminiert und damit die Abhängigkeit von diesem.

Eingesetzt ergibt sich mit p aus der Polarisationsbestimmung in Kap. 3.2.3:

$$f(\delta) = \frac{1 - \left| \frac{n_1 \sqrt{1 - \frac{n_2^2 sin(\delta)^2}{n_1^2}} - n_2 cos(\delta)^2}{n_1 \sqrt{1 - \frac{n_2^2 sin(\delta)^2}{n_1^2}} + n_2 cos(\delta)^2} \right|^2}{1 - \left| \frac{n_2 \sqrt{1 - \frac{n_2^2 sin(\delta)^2}{n_1^2}} - n_1 cos(\delta)^2}{n_2 \sqrt{1 - \frac{n_2^2 sin(\delta)^2}{n_1^2}} + n_1 cos(\delta)^2} \right|^2} = p$$

Es ist nicht direkt ersichtlich, ob daraus eine Umkehrfunktion $\delta(p) = f^{-1}(p)$ gebildet werden kann. Wie die Berechnung von $arctan(f(\delta))$ in Abb. 3.7 zeigt, ergibt sich ein umkehrbar eindeutiger Zusammenhang zwischen dem Neigungswinkel δ und der Polarisationsrichtung γ_{In} und damit p. Damit lässt sich die Funktion $\delta = f^{-1}(p)$ durch eine Reihe an vorberechneten Punkte ersetzten, womit die Lösung durch Interpolation schnell gefunden werden kann.

Die Ausrichtung der Neigungsachse ist in diesem Fall parallel zur SCPEM-Schwingrichtung (0°), eine Drehung um 90° wird durch Vertauschen von $\epsilon_s(\delta)$ und $\epsilon_p(\delta)$ oder durch Verwendung des Kehrwerts 1/p erreicht. Im Bereich kleiner Neigungswinkel ist mit einem großem Anteil unpolarisierter Strahlung zu rechnen, die eine starke Veränderung der gemessenen Polarisationsrichtung zur Folge haben kann, wie in Kap. 3.3.4 gezeigt wird. Der Effekt und dessen Auswirkung muss noch weiter untersucht werden.

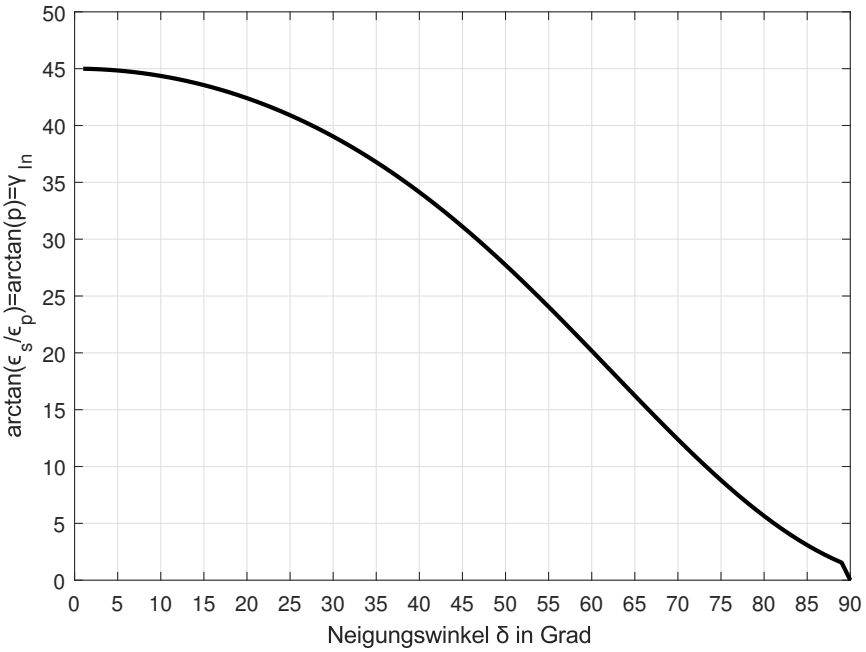


Abbildung 3.7: Theoretischer Zusammenhang zwischen Neigungswinkel und gemessenem Polarisationswinkel (Anh. B.2.5)

3.5 Polarisationsrichtung als Qualitätskriterium

Eine Möglichkeit der Eliminierung der unbekannten Funktion $DOP(p)$ oder $DOP(\gamma_{In})$ und der nur theoretisch bekannten Funktion $\delta(p) = f^{-1}(p)$ bzw. $\delta(\gamma_{In}) = f^{-1}(tan(\gamma_{In}))$ für die Neigung ist der direkte Vergleich des Schnittergebnisses mit den gemessenen Polarisationswerten. Bei mehreren unterschiedlichen Probeschnitten werden die Ergebnisse der Messwerterfassung mit der Qualität des Schnittes verglichen und daraus Grenzwerte für γ_{In} definiert. Dadurch werden die unbekannten Größen aus dem System entfernt, da der gemessene Polarisationsgrad selbst als Qualitätskriterium verwendet werden kann und nicht die Neigung der Schneidfläche, die erst über mehrere Schritte mit jeweils verschiedenen Unbekannten berechnet werden muss.

Nachteilig ist die Abhängigkeit des Messergebnisses von der Schneidrichtung, die eine gesonderte Betrachtung je Schnittrichtung nötig macht.

4 Messaufbau

4.1 Vorhandener Messaufbau

Abbildung 4.1: Messaufbau [11]

Es ist ein Messaufbau vorhanden (Abb. 4.1), welcher für sichtbares Licht konzipiert ist. Dieser verwendet eine Laserdiode als Strahlquelle, einen SCPEM mit Ansteuerung auf einem Steckboard und eine Photodiode mit Verstärker auf einer geätzten Platine.

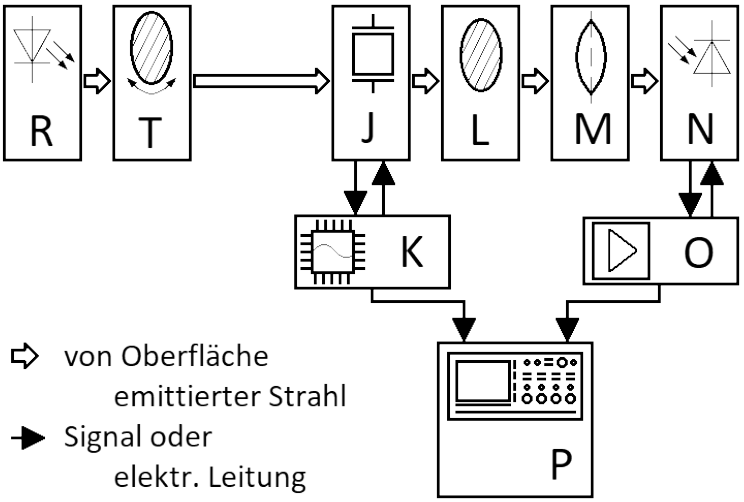


Abbildung 4.2: Komponenten des vorhandenen Messystems

Als Polarisator bzw. Analysator werden für sichtbares Licht geeignete Polarisatoren verwendet, die Auswertung erfolgt mit einem Oszilloskop, das die Signale von SCPEM und Photodiode darstellt. Eine automatische Berechnung des Polarisationswinkels ist nicht möglich. [11]

4.1.1 Strahlquelle

Laserdiode DB605-1-3-FA(22x105)-S-AP, 1 mW (Laserklasse 2 nach DIN EN 60825-1) [11]

4.1.2 Filter

In diesem Aufbau ist kein Filter für eine bestimmte Wellenlänge nötig, da die Laserdiode bereits ein sehr schmales Spektrum hat.

4.1.3 Polarisatoren

Es werden Linearpolarisatoren von Bernhard Halle verwendet. [11]

4.1.4 Photodiode und Verstärker

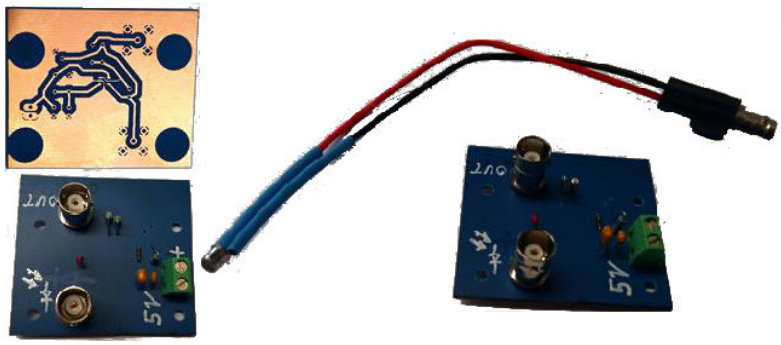


Abbildung 4.3: Photodiode mit Transimpedanzverstärker [11]

Das Signal der Photodiode wird mit einem bereits eigens angefertigten Transimpedanzverstärker verstärkt. Dieser wird mit 5 V versorgt und stellt das Signal für das Oszilloskop zur Verfügung. [11]

4.1.5 SCPEM und Ansteuerung

Im Messaufbau befindet sich ein SCPEM inkl. Halterung und zwei Kontakten für die Ansteuerung. Die Ansteuerung selbst ist auf einem Steckboard realisiert, siehe Abb. 4.4 und für Details zur Schaltung Kap. 5.2.

4.1.6 Auswerteeinheit

Zur Auswertung der beiden Signale (SCPEM und Diode) wird das Oszilloskop Tektronix 2014C verwendet. Damit besteht die Möglichkeit, Messdaten als Bildschirmaufnahme zu speichern oder mit der Software TekVISA V4.1.1 in MATLAB einzulesen. Die Übertragung der Daten ist dabei nicht kontinuierlich möglich, sondern nur aus dem Puffer des Oszilloskops, nachdem der Trigger ausgelöst wurde, vergleichbar mit Kap. 6.3. [22]

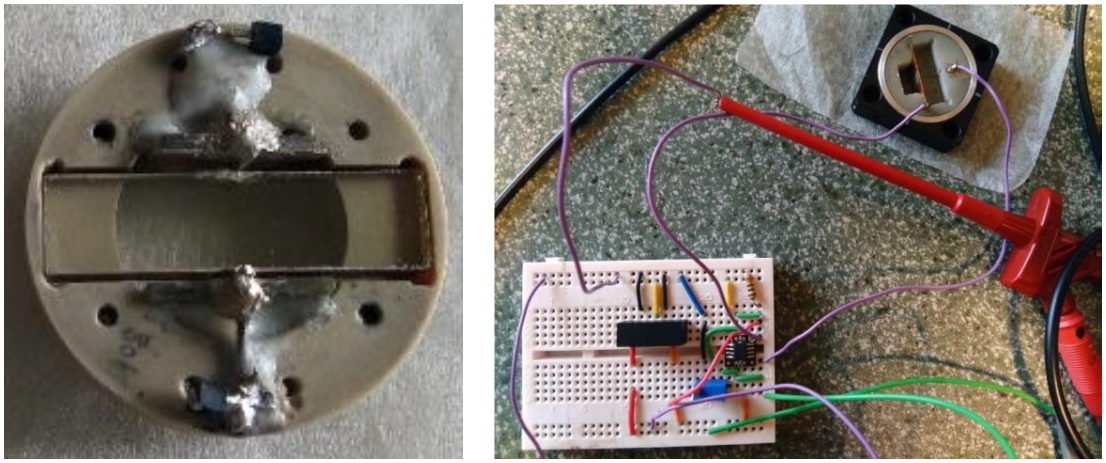


Abbildung 4.4: SCPEM im Kunststoffgehäuse, Ansteuerung des SCPEM [11]

4.1.7 Testmessungen bei $\lambda = 605\ nm$

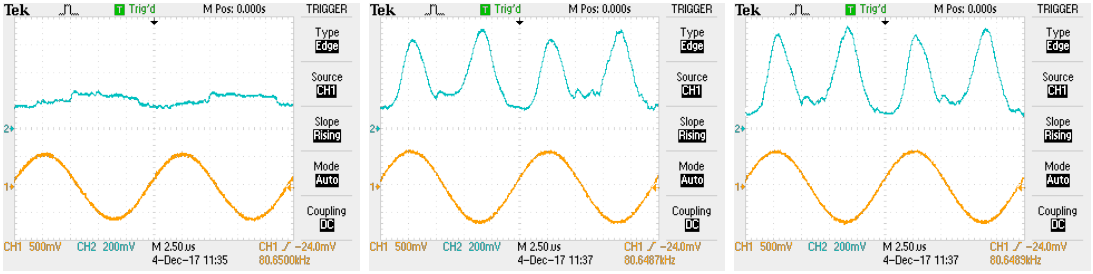


Abbildung 4.5: SCPEM-Signal (unten) und Dioden-Signal (oben) bei 0°, 22.5° und 45° Polarisatorstellung (Anh. B.1.1)

Der SCPEM wird für diese Aufnahmen mit einer maximalen Retardation von ca. 180° betrieben. Der Analysator ist auf 45° eingestellt. die Polarisatorstellung (T) beträgt 0°, 22,5° und 45°. Zu erkennen sind große Störeinflüsse auf das Dioden-Signal, das Abflachen bzw. die kleine Einbuchtung an den Minima-Stellen ist charakteristisch für diese Retardation (Kap. 3.2.1.2), im idealen, störungsfreien Fall wird die Intensität an dieser Stelle Null.

Die Phasenverschiebung zwischen SCPEM-Signal und Dioden-Signal ist hier 0°, die Nulldurchgänge des SCPEM-Stroms stimmen mit der Position der Dioden-Minima überein. Statt 0° ist dieses Ergebnis auch bei einem Vielfachen von 180° für die Phasenverschiebung möglich, wahrscheinlicher ist jedoch, dass es zu keiner Phasenverschiebung kommt.

4.2 Änderungen am Messaufbau

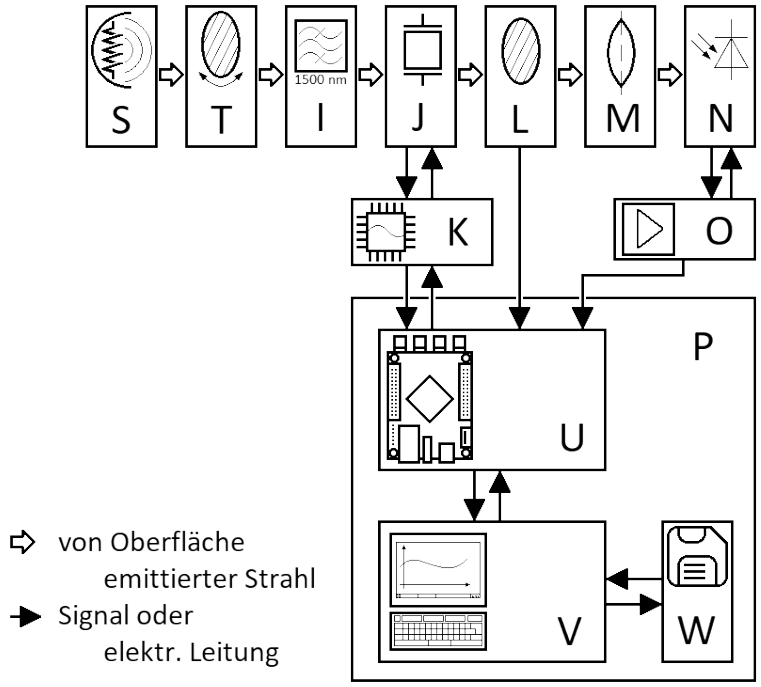


Abbildung 4.6: Komponenten des neuen Messaufbaus

Es sollen einige Änderungen am Aufbau (Abb. 4.6) vorgenommen werden, um die Anforderungen für die danach folgenden Versuche und die Entwicklung der Messsoftware zu erfüllen. Die Messung erfolgt im Nahen-Infrarot-Bereich, die optischen Elemente werden dafür angepasst. Es wird ein möglichst störungsfreies Signal gefordert und daher sollen für den Anschluss der Auswerteeinheit BNC-Buchsen zur Verfügung stehen, um die ungeschirmte Kabellänge möglichst zu minimieren.

S: Als Strahlquelle wird ein Infrarotemitter eingesetzt (Kap. 2.1.3).

T: Für die Einstellungen einer definierten Polarisation wird ein drehbarer Polarisator mit Winkelskala verwendet. Der eingestellte Winkel soll später möglichst genau und möglichst schnell von der Auswerteeinheit aus den Messsignalen ermittelt werden können.

U: STEMlab-Board, Kap. 4.2.6

V: Computer, Kap.4.2.6

W: Datenspeicher, Kap. 4.2.6

4.2.1 Strahlungsquelle

Es wird der Infrarotemitter EK-8620 von Helioworks eingesetzt. Dieser besteht aus einem aufgewickeltem Kanthal-Filament und erreicht bis zu 8,4 W Eingangsleistung bei 3,5 V und 2,4 A. Er weist im Gegensatz zur Laserdiode ein sehr breites Frequenzspektrum auf, wodurch für die gewünschte Funktion des SCPEM ein zusätzlicher Filter benötigt wird (Abb. 4.8 und 4.10). Die Abstrahlung erfolgt über einen großen Austrittswinkel relativ gleichmäßig (Abb. 4.7), dadurch ist die Einstellung der Strahlrichtung am Messaufbau einfach.

Die beiden letzten Eigenschaften erklären den großen Leistungsunterschied zwischen der eingesetzten Laserdiode mit 1 mW und des eingesetzten IR-Emitters, der bei Leistungen von einigen Watt betrieben wird (erfahrungsgemäß meist zwischen 3 W und 6 W im Testaufbau). Ein großer Teil wird seitlich am Messsystem vorbeigestrahlt, ein Teil trifft auf den Filter, welcher davon nur einen kleinen Teil der Strahlung mit der passenden Frequenz durchlässt. Eine grobe Abschätzung der Leistungsverluste der Strahlung am Weg zur Photodiode wird in Kap. 4.2.4.1 getroffen.

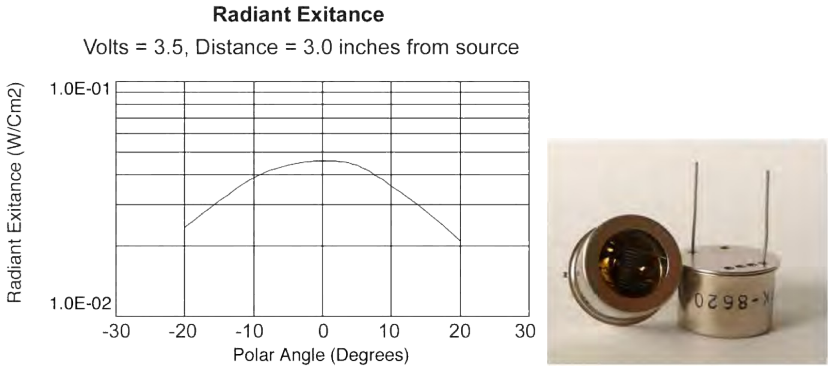


Abbildung 4.7: Winkelabhängige Abstrahlleistung des IR-Emitters bei 8,4 W Gesamtleistung (Anh. A.2.1)

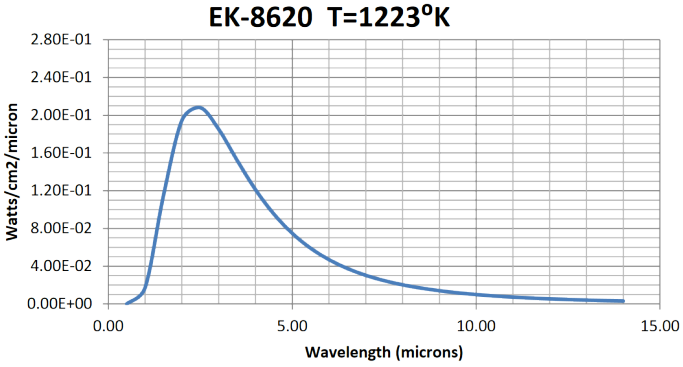


Abbildung 4.8: Strahlungsspektrum des Infrarotemitters (Anh. A.2.1)

4.2.2 Interferenzfilter

Für die nötige Beschränkung des Strahlungsspektrums wird ein Interferenzfilter verwendet (Kap. 2.1.3). Der eingesetzte Bandpassfilter der bk Interferenzoptik Elektronik GmbH bk-1500-090-B hat die Spektrallinie laut Prüfbericht bei 1.510,5 nm und eine Halbwertsbreite von 99 nm. Die maximale Transmission liegt bei 81%. Die Halbwertsbreite wird bei 40,5% ermittelt (Hälfte der maximalen Transmission) und reicht von 1.461 nm bis 1.560 nm.

4.2.2.1 Prüfbericht des Interferenzfilters

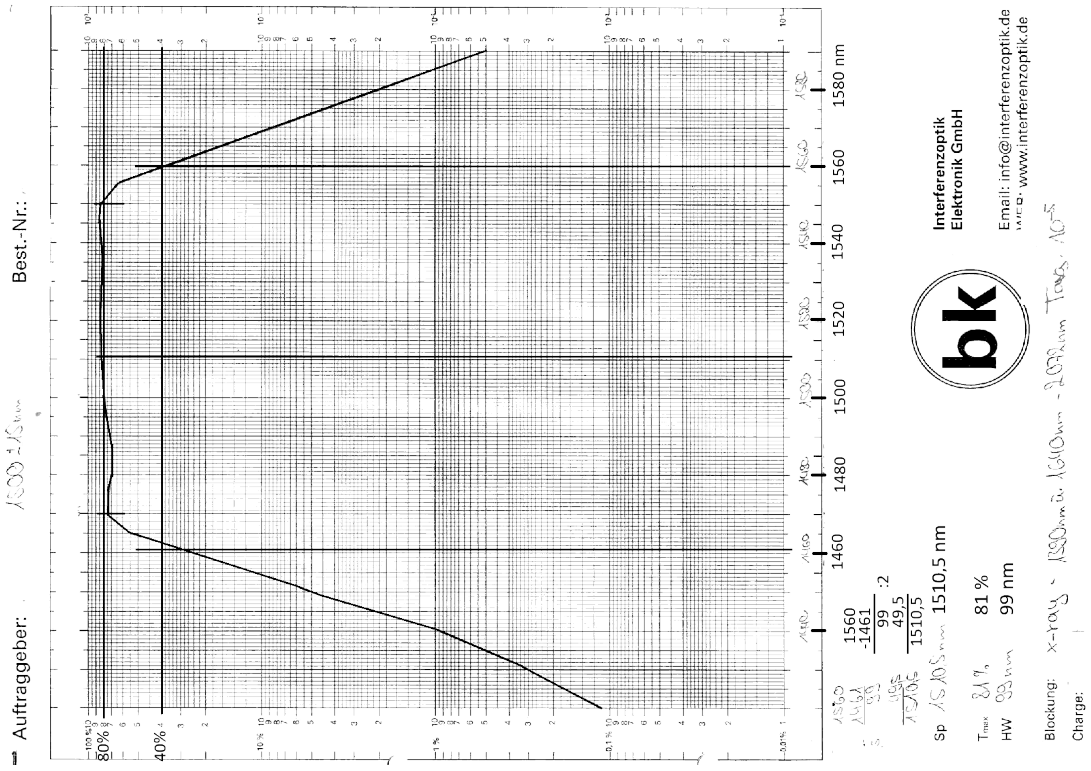


Abbildung 4.9: Prüfbericht des Interferenzfilters (Anh. 4.9, Kennwerte und Linien für bessere Lesbarkeit ergänzt)

Bei genauerer Betrachtung des Prüfberichtes (Abb. 4.9) ist zu erkennen, dass die maximale Transmission etwa dem Mittelwert der Transmission zwischen 1.470 nm und 1.550 nm entspricht. Die obere Grenze der Halbwertsbreite stimmt mit dem Schnittpunkt der Transmissionskurve bei 40,5% überein, die untere Grenze liegt 1 bis 2 nm zu weit links. Die Halbwertsbreite würde damit 97 bis 98 nm betragen, die Spektrallinie 1.511 bis 1.511,5 nm, womit sie weiterhin im Toleranzbereich der Herstellerangabe 1.500±15 nm liegt (Anh. A.2.2.1). Die Unterschiede sind minimal und es werden die am Prüfbericht angegeben Werte verwendet.

4.2.2.2 Einbau und Wirkung auf das Spektrum des Infrarotemitters

Der Filter ist jedenfalls vor dem SCPEM zu positionieren, die Einbaurichtung im Messsystem ist dabei zu beachten. Die kupferartige Seite des Filters muss Richtung Strahlquelle gerichtet sein und blockt den langwelligen Anteil, die goldene Seite in Richtung Photodiode blockt den kurzwelligen Anteil des Spektrums.

Die Wirkung des Filters auf die Strahlung des Infrarotemitters ist in Abb. 4.10 zu sehen. Nur ein kleiner Teil der abgestrahlten Leistung passiert den Filter.

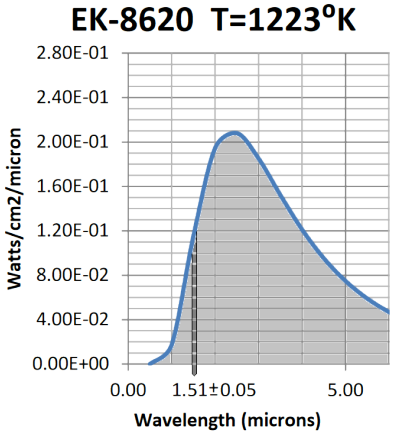


Abbildung 4.10: Bandpassfilter im Emitterspektrum

Durch grobe Messung der Fläche im Spektrum des Emitters (Abb. 4.8) lässt sich die durchgelassene Strahlung mit etwa 1-2% der auf den Filter ankommenden Strahlung angeben.

4.2.3 Polarisatoren

Es werden die Polarisatoren verwendet, die bereits für sichtbares Licht im Einsatz waren. Zu beachten ist, dass nicht alle Polarisatoren für Strahlung im Infrarotbereich geeignet sind. Diese haben dann nur wenig oder keine Polarisationswirkung und können nicht eingesetzt werden.

4.2.4 Photodiode und Verstärker

Abbildung 4.11: Photodiode im TO-18-Gehäuse (Anh. A.2.3)

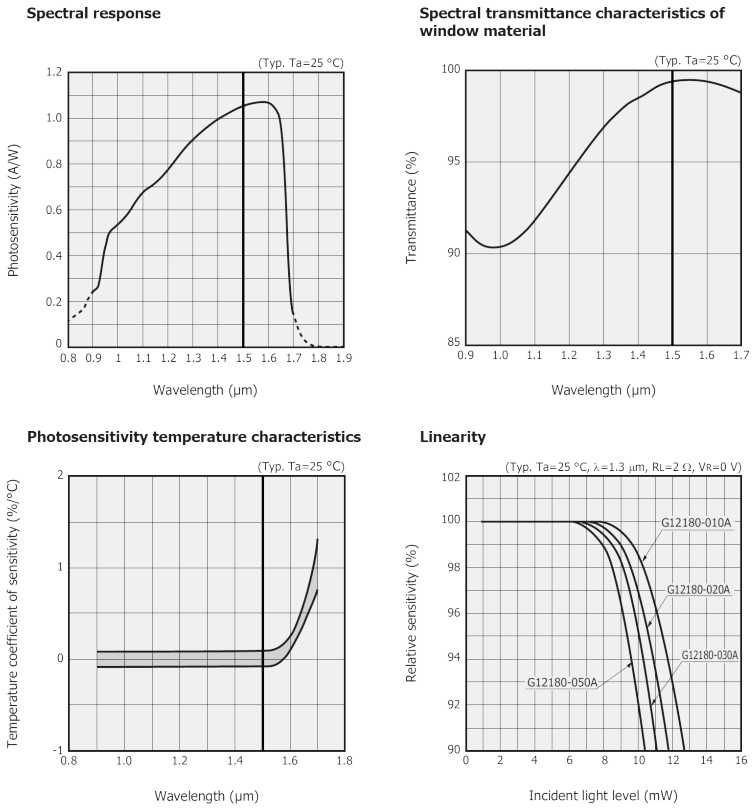


Abbildung 4.12: Wichtige Kennlinien der Photodiode (Anh. A.2.3)

Es wird eine Hamamatsu-Photodiode G12180-010A im TO-18-Gehäuse mit einer photosensitiven Fläche von 1 mm Durchmesser verwendet.

Die ersten drei Kennlinien in Abb. 4.12 zeigen die Eignung für die zu messende Wellenlänge von 1.500 nm. Die Empfindlichkeit ist nahezu maximal, die Temperaturabhängigkeit gering.

Die Änderung der relativen Sensitivität und damit eine nichtlineare Kennlinie wären störend für das Messergebnis, da jeweils mit dem Verhältnis von zwei Intensitäten gerechnet wird. Dabei ist die Linearität des Signals im Vergleich zur auftretenden Intensität vorauszusetzen. Um diese Linearität gewährleisten zu können, darf nach Abb. 4.12 rechts unten die Strahlungsleistung 8 mW nicht übersteigen. Daher muss die maximale Leistung der Strahlung, die auf die Photodiode trifft, abgeschätzt werden.

4.2.4.1 Abschätzung der maximalen Strahlungsleistung an der Photodiode

Es tragen primär zwei Effekte dazu bei, dass nur ein kleiner Anteil der vom Emitter abgestrahlten Leistung die Diode erreicht: Die Abstrahlung am optischen Pfad vorbei und die Filterung durch den Interferenzfilter.

Wird die transparente Fläche des SCPEM als engste Stelle im Pfad mit 1,12 cm^2

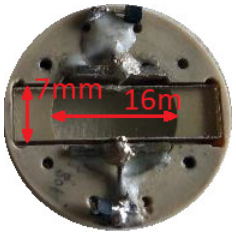


Abbildung 4.13: Transparente Fläche des SCPEM (Originalgröße)

angenommen (7 mm x 16 mm, Abb. 4.13) und der minimale Abstand vom Emitter mit 7,5 cm (3“), so ergibt sich mit Abb. 4.7 bei 0° Abstrahlwinkel 0,045 W/cm² bzw. 0,05 W, die durch den SCPEM strahlen. Der Interferenzfilter filtert nach Abb. 4.10 etwa 98% der Strahlung, sodass sich die 50 mW auf etwa 1 mW verringern und damit die zu erwartende Intensität an der Diode in jedem Fall unter 8 mW liegt. Verluste am Emitter, durch die Polarisatoren und bei der Sammellinse sind dabei noch nicht berücksichtigt.

Die an der Photodiode ankommende Leistung ist somit größenordnungsmäßig mit der 1 mW-Laserdiode des ursprünglichen Aufbaus vergleichbar.

4.2.4.2 Verstärker

Für die Verstärkung des Diodensignals wurde in der Diplomarbeit [3] in Zusammenarbeit mit Plasmo Industrietechnik GmbH bereits ein spezialisierter Messverstärker entwickelt, Abb. 4.14. Dieser beinhaltet jeweils einen Transimpedanzverstärker und einen Nachverstärker für vier unabhängige Signale. Bei jedem Kanal kann der Nachverstärker in drei unterschiedliche Verstärkungsstufen geschaltet werden. Dadurch ist eine Anpassung an den Spannungsbereich des ADC des STEMlab-Boards möglich, wo der Verstärkerkanal mit einem BNC-Kabel am RF-Eingang 1 angeschlossen wird. [3]

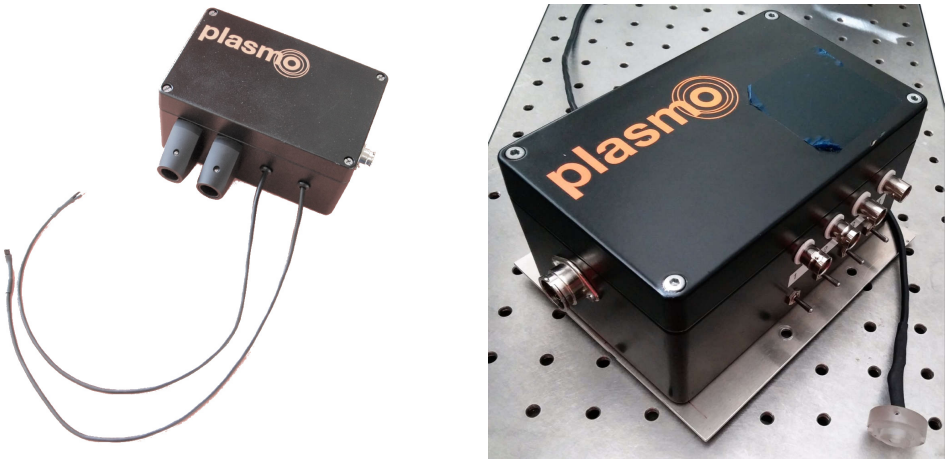


Abbildung 4.14: Messverstärker nach Fertigung [3] und aktuelle Aufnahme

4.2.5 SCPEM und Ansteuerung

Der SCPEM (Kap. 2.2) erreicht jetzt durch die größere Wellenlänge eine kleinere Retardation bei gleich großer Spannungsamplitude als bei Wellenlängen im sichtbaren Bereich. Wird mit einer bestimmten Spannung bei einer Wellenlänge von 605 nm die Wirkung einer Halbwellenplatte mit Phasendifferenz $\varphi_{max} = \pi$ oder 180° erreicht, wird bei 1.500 nm nur mehr $\varphi_{max} = \frac{605}{1.500} * \pi$ oder 72,6° erreicht. Da die Ansteuerspannung sowohl durch die mechanische Festigkeit des Kristalls als auch durch die verwendeten elektronischen Bauteile der Ansteuerung begrenzt ist, können über längere Zeit nur Retardationen bis etwas über 90° (Verhalten einer Viertelwellenplatte) erreicht werden.

Dieser Umstand wird in Kap. 3.2.3 berücksichtigt, indem eine Lösung des Gleichungssystems für beliebig große Retardationen φ_{max} sowie eine Möglichkeit zur Bestimmung dieser Retardation gefunden wird.

Am SCPEM selbst sind keine Änderungen nötig.

Die Ansteuerung des SCPEM wird auf eine geätzte Platine übertragen, um eine kompaktere Ausführung zu ermöglichen und Messfehler durch lose Drähte etc. zu verhindern. Die Schaltung und deren Anpassungen sind in Kap. 5.3 beschrieben.

4.2.6 Auswerteeinheit

Das STEMlab-Board (Kap. 2.3) stellt das Zwischenstück zwischen den analogen Signalen und der digitalen Datenverarbeitung dar. Ob es sich bei den an den Computer V übermittelten Daten um die digitalisierten Signale oder bereits um berechnete Polarisationen handelt, ist zu diesem Zeitpunkt noch unklar und hängt von der Leistungsfähigkeit der Hardware und der Datenübertragung ab.

Für die weitere Datenverarbeitung und Visualisierung wird ein Computer auf x86-64 Basis verwendet. Da bereits mehrere Geräte mit diesen Betriebssystemen im Laborbereich vorhanden sind, werden Microsoft Windows 7 und Windows 10 verwendet und die Funktion des Messprogramms damit getestet.

Die Daten werden auf einem nichtflüchtigen Speicher abgelegt, entweder im Computer integriert oder extern (Wechseldatenträger, Server).

4.2.7 Testmessungen bei $\lambda = 1.500\ nm$

Die folgenden Testmessungen werden mit dem Oszilloskop Tektronix 2014C erstellt, um eine Vergleichbarkeit zum ursprünglichen Aufbau zu erhalten. Messungen mit der neuen Auswerteeinheit sind in Kap. 9.3 zu finden.

4.2.7.1 Signalstörungen durch Komponenten ohne Masse-Verbindung

Die ersten Messungen (Abb. 4.15) im Infrarot-Bereich zeigen Strahlungsintensitäts-Signale, die bei genauerer Betrachtung nicht von der Strahlung stammen können (völliges Abdunkeln der Photodiode ergibt Signal ungleich Null). Diese Störsignale haben die gleiche Frequenz wie die SCPEM-Ansteuerung. Es zeigt sich, dass die Stangen des

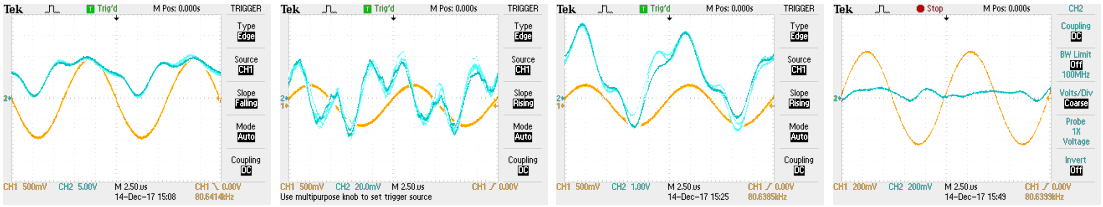


Abbildung 4.15: SCPEM und Dioden-Signal mit Störung: Aufnahme 1, 2, 3 und 4 (Anh. B.1.2)

Cage-Systems, in und an dem sich die optischen und elektronischen Komponenten befinden, durch die schwarz eloxierten Halterungen nicht elektrisch leitend verbunden sind. Die Verbindung der zur Photodiode führenden Stangen mit dem Masse-Kontakt verringert die Störungen auf ein vernünftiges Maß, bei dem das Signal der tatsächlichen Strahlungsintensität um Größenordnungen höher ist als die Störung (4.16).
Aufnahmen Abb. 4.15:

1. Polarisatorstellung 45°, Analysatorstellung 45°: Signal deckt sich nicht mit der in Kap. 3 erarbeiteten Theorie (Abb. 4.16 Aufnahme 2 zeigt als Vergleich das nahezu störungsfreie Signal).

2. Emitter deaktiviert, Diode abgedunkelt: Störsignal bei Verstärkerstufe 1

3. Emitter deaktiviert, Diode abgedunkelt: Störsignal bei Verstärkerstufe 3

4. Emitter deaktiviert, Diode abgedunkelt: Störsignal bei Verstärkerstufe 3 nach Verbindung einiger elektrisch leitender Teile des Messaufbaus

4.2.7.2 Messungen ohne Störeinfluss

Es sind alle Komponenten des Messaufbaus auf Masse gelegt (Bauteile des optischen Systems, Verstärker, SCPEM-Ansteuerung). Für die SCPEM-Ansteuerung wird zum Aufnahmezeitpunkt die Steckboard-Variante verwendet.

Die Signale stimmen mit Ausnahme der Phasenverschiebung gut mit der Theorie überein und die Messqualität lässt die Entwicklung der automatischen Auswertung der Polarisation zu.

Alle Aufnahmen in Abb. 4.16 sind mit Analysatorstellung 45° und einer Retardation von ca. 90° (eingestellt nach Kap. 3.2.2.2) erstellt:

1. Polarisatorstellung 0° (90°): Das Diodensignal ist konstant. Der SCPEM verändert die Polarisation nicht, wenn sie parallel zu einer seiner geometrischen Achsen ist.

2. Polarisatorstellung 45°: Das Diodensignal erhält ein Maximum an der Stelle von $\varphi(t) = 0$ und ein Minimum an der Stelle $\varphi(t) = \varphi_{max} \approx \pm\pi/2$. Beim Maximum wird die Polarisation nicht verändert und ist damit parallel zum Analysator (beide bei +45°, keine Auswirkung auf Intensität), beim Minimum wird die Polarisation zirkular polarisiert und damit die Intensität am Analysator verringert.

3. Polarisatorstellung -45°: Das Diodensignal erhält ein Minimum an der Stelle von $\varphi(t) = 0$ und ein Maximum an der Stelle $\varphi(t) = \varphi_{max} \approx \pm\pi/2$. Beim Minimum wird die Polarisation nicht verändert und ist damit normal zum Analysator (bei -45° und +45°, vollständige Absorption), beim Maximum wird die Polarisation zirkular polarisiert und damit die Intensität am Analysator verringert, jedoch nicht vollständig absorbiert. Die Intensität bei $\varphi(t) = \pm\pi/2$ stimmt für alle Polarisatorstellungen überein, da bei dieser Retardation nach dem SCPEM immer zirkular polarisiertes Licht vorliegt (bei vollständig linear polarisierter Eingangsstrahlung).

Es wird eine Phasenverschiebung des Intensitätssignals zum SCPEM-Signal festgestellt, diese ist der letzten Aufnahme in Abb. 4.16 visualisiert. Diese Verschiebung ist unabhängig von der Polarisatorstellung und der Retardation vorhanden. Die Phasenverschiebung muss bei der Bestimmung der Extremwerte vom Auswerteprogramm berücksichtigt werden. Sie kann später im Messprogramm gemessen oder vorgegeben werden.

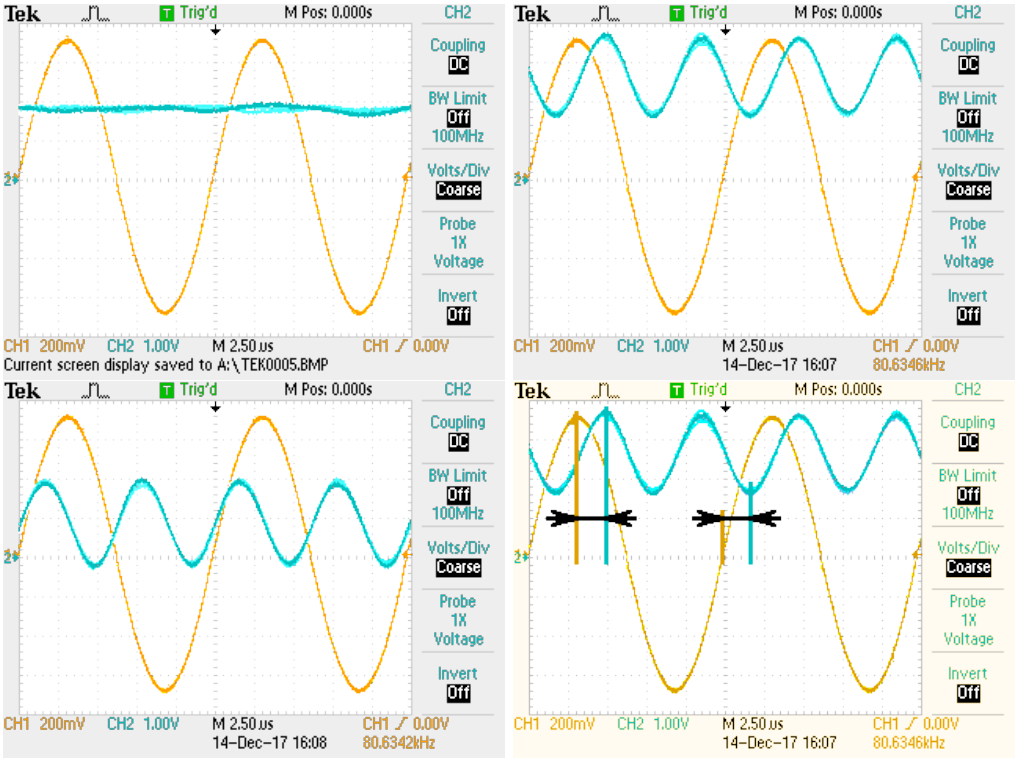


Abbildung 4.16: SCPEM-Signal und Dioden-Signal: Aufnahme 1, 2, 3; Visualisierung der vorhandenen Phasenverschiebung (Anh. B.1.3)

5 SCPEM-Ansteuerung

5.1 Ansteuerungsmöglichkeiten des SCPEM

Der SCPEM benötigt für die gewünschte Funktionalität eine angelegte alternierende Spannung. Diese muss auf die gewünschte Resonanzfrequenz abgestimmt sein, die Höhe der Spannung des Stromes bestimmt die Stärke der Retardierung.

5.1.1 Benötigte Steuersignalform

Die Entwicklung der bereits vorhandenen Ansteuerung (Kap. 5.2) hat gezeigt, dass ein Rechtecksignal ausreichend ist, um den Kristall mit seiner Resonanzfrequenz zu betreiben, es kann, aber muss dazu kein harmonisches Signal generiert werden. [11]

5.1.2 Signalgenerator

Ein Signalgenerator kann das benötigte Steuersignal generieren. Dazu muss die Eigenfrequenz des SCPEM sehr genau bekannt sein. Ein Abweichen dieser Eigenfrequenz durch Temperaturunterschiede oder andere Einflüsse wird nicht automatisch erkannt und korrigiert. Daher muss die Frequenz entweder manuell angepasst werden oder es kommt zu einer abgeschwächten Resonanz und damit abgeschwächter Retardation.

5.1.3 STEMlab-Board als Signalgenerator

Das STEMlab-Board besitzt zwei Signalgenerator-Ausgänge, die ebenfalls für die Ansteuerung genutzt werden können. Das gewünschte Signal wird dabei in einen Puffer geschrieben und dann kontinuierlich ausgegeben. Dadurch ergeben sich dieselben Probleme wie bei einem eigenständigen Signalgenerator. Da der Puffer während der Signalausgabe geändert werden kann, ist die Anpassung des Signals automatisierbar, jedoch mit erheblichem Aufwand, da das Signal bei Änderungen kontinuierlich bleiben muss, plötzliche Phasenverschiebungen dürfen nicht auftreten.

Da der Spannungsbereich der Ausgänge nur ± 1 V beträgt, ist ein zusätzlicher Verstärker nötig, um den SCPEM ausreichend anzuregen.

5.1.4 Vorhandene Ansteuerung

Die Ansteuerung auf Basis eines Operationsverstärkers hat einen Regelkreis, dadurch muss die Eigenfrequenz des SCPEM nicht exakt bestimmt werden, da sie sich selbst einstellt und über die Zeit nachjustiert. Bei Wechseln des SCPEM muss gegebenenfalls das Potentiometer der Rückkopplung neu justiert werden, dazu bietet es sich an, das SCPEM-Signal mit einem Oszilloskop zu betrachten oder die STEMlab-Web-Application „Oszilloscope“ zu verwenden.

Da diese Schaltung stabil läuft und zudem einfach und günstig ist, wird sie als Grundlage für etwaige Änderungen verwendet und die anderen Möglichkeiten zur Anregung des SCPEM nicht weiter ausgearbeitet.

5.2 Ursprünglicher Zustand

5.2.1 Steckplatine

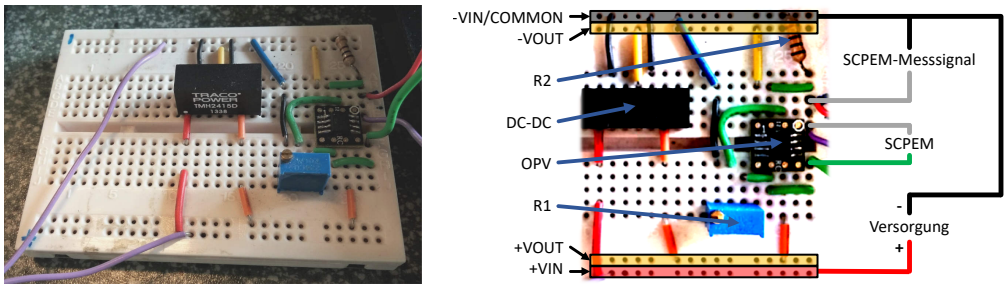


Abbildung 5.1: Aufbau der Ansteuerung als Steckplatine (Anh. B.3.1.2)

Die Schaltung ist auf einer Steckplatine realisiert. Diese hat neben der Spannungsversorgung Anschlüsse für den SCPEM sowie für die Signalmessung mit dem Oszilloskop bzw. später mit dem STEMlab-Board. Diese Leitungen sind als Drähte ausgeführt und werden mit Klemmen angeschlossen. Die Drähte zum SCPEM werden an den Kontakten auf dessen Halterung angeschlossen.

5.2.2 Schaltplan

Aus der aufgebauten Steckplatine lässt sich der Schaltplan Abb. 5.2 ableiten.

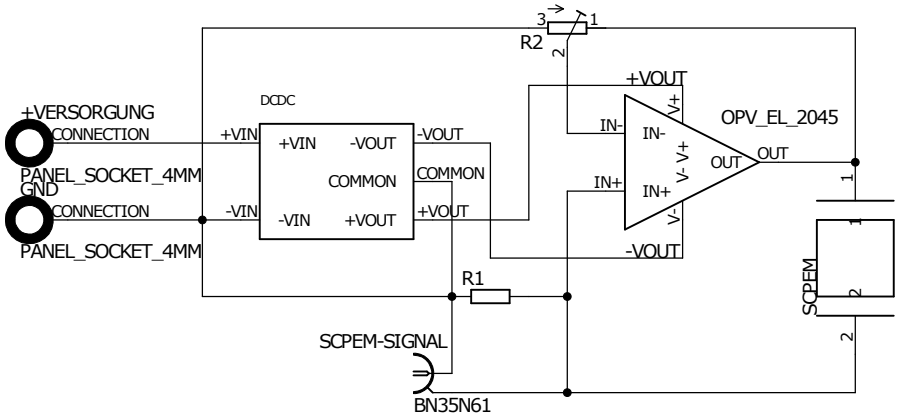


Abbildung 5.2: Schaltplan der SCPEM-Ansteuerung (Anh. B.3.1.1)

Die verwendeten Bauteile sind identisch mit den in Tabelle 5.1 aufgelisteten.

Das Messsignal (Spannungsabfall am Widerstand R_1) ist proportional zum Strom über den SCPEM. Der Operationsverstärker-Eingang IN+ ist hochohmig ($R_{IN} = 150\ k\Omega$, differential mode), daher fließt beinahe der gesamte SCPEM-Strom über den vergleichsweise kleinen Widerstand R_1 (100 Ω). Der Anteil des Stroms durch den Widerstand am gesamten SCPEM-Strom beträgt $\frac{150\ k\Omega}{150\ k\Omega + 100\ \Omega} = 99{,}9\ \%$ und kann als gleich angesehen werden. Im Gegensatz zur SCPEM-Spannung ist der SCPEM-Strom kein Rechtecksignal, sondern ein harmonisches.

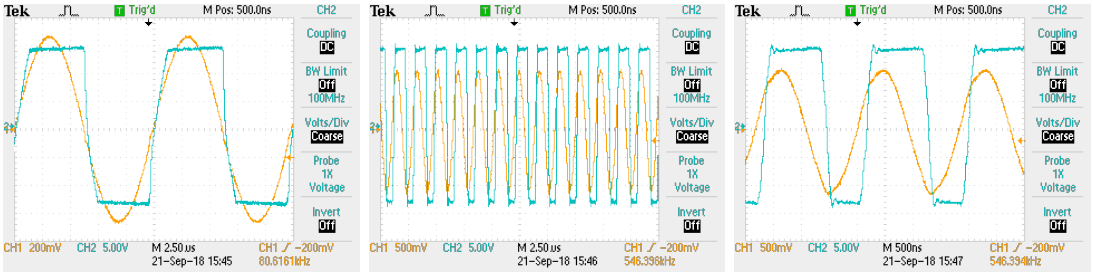


Abbildung 5.3: SCPEM-Spannung und SCPEM-Strom: links richtige Resonanzfrequenz, rechts falsche, zu hohe Resonanzfrequenz wegen falscher Potentiometereinstellung (links und mittig gleiche Zeitskala, rechts im Detail) (Anh. B.1.4)

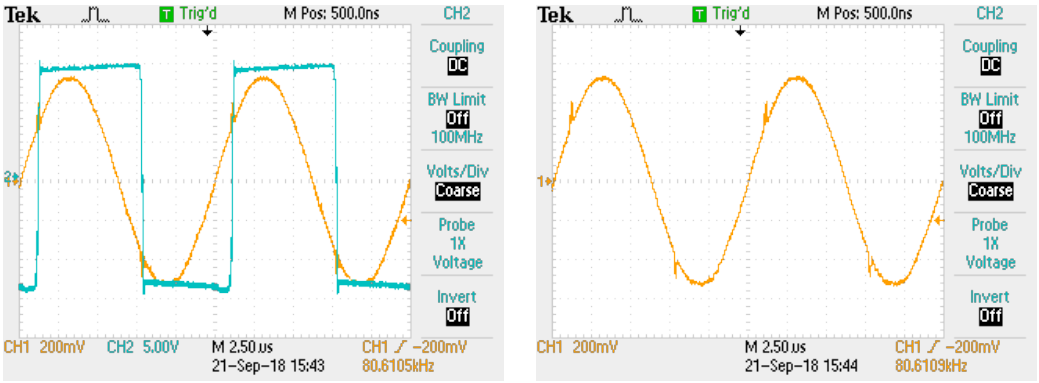


Abbildung 5.4: SCPEM-Spannung und SCPEM-Strom: Störungen im SCPEM-Strom-Signal bei falscher Potentiometereinstellung (Anh. B.1.4)

5.2.3 Einstellung der Rückkopplung

Am Potentiometer R_2 lässt sich die Rückkopplung beeinflussen. Ist der Widerstand zwischen Rückkopplung und GND zu hoch eingestellt, kommt es zum Schwingen des SCPEM in einer anderen Resonanzfrequenz (ca. 550 kHz, Abb. 5.3). Das passiert insbesondere beim Anschwingvorgang, ein Umschlagen der Frequenz ist jedoch später ebenso möglich. Bei zu kleinem Widerstand kommt es in einem kleinen Bereich der Periode zu

einem Oszillieren des SCPEM-Steuer- und des SCPEM-Messsignals (Abb. 5.4), wodurch die Bestimmung des Nulldurchgangs negativ beeinflusst werden kann. Außerdem sind durch diese vielen zusätzlichen, schnellen Spannungswechsel am Kristall größere Alterungseffekte möglich.

Das Potentiometer hat einen gemessenen Widerstand zwischen Rückkopplung und GND von 750 Ω. Schwingt der Kristall bereits, kann der Widerstand auf 3 bis 4 $k\Omega$ erhöht werden, um ein glatteres SCPEM-Signal zu erhalten, muss jedoch vor der nächsten Inbetriebnahme wieder verringert werden.

5.3 Überarbeitete Version

5.3.1 Änderungen

Die Schaltung wird als Platine umgesetzt, dadurch können lose Kontakte vermieden und ein sauberer und kompakter Aufbau gewährleistet werden.

Durch eine BNC-Buchse für das SCPEM-Signal wird der Anschluss des Messgerätes vereinfacht.

5.3.1.1 Angepasste Rückkopplung für Anschwingvorgang

Um die Problematik der richtigen Potentiometereinstellung (Kap. 5.2.3) zu lösen, wird ein weiteres Potentiometer verwendet, welches nur beim Startvorgang aktiv ist und nach dem Anschwingvorgang weggeschaltet wird. Dadurch kann zu Beginn ein höherer Widerstand sicherstellen, dass sich die richtige Eigenform einstellt, während im weiteren Betrieb das Oszillieren vermieden wird.

Bauteile Die zusätzlichen Bauteile sind das Potentiometer R_3, die Widerstände R_4 und R_5, ein Transistor T_1 zum Wegschalten des Zweiges und ein Kondensator, der durch seine Ladungskurve den Transistor nach einiger Zeit hochohmig schaltet.

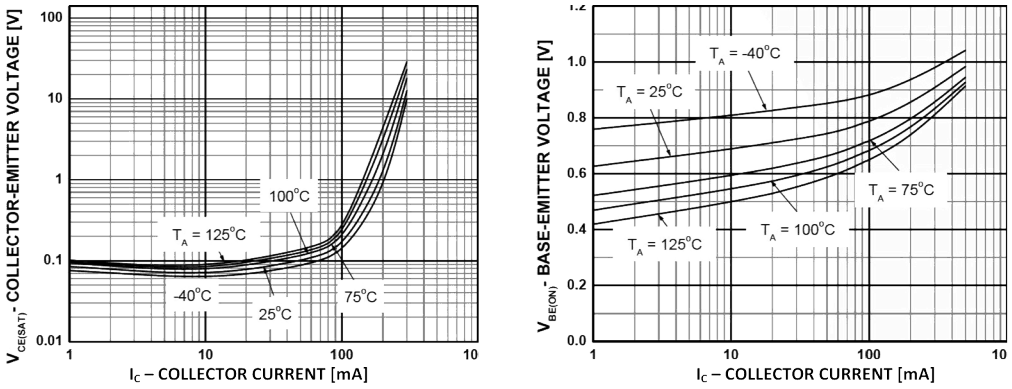


Abbildung 5.5: Transistor: Kollektor-Emitter-Spannung und Basis-Emitter-Spannung in Abhängigkeit vom Kollektorstrom (Anh. A.2.4)

Funktionsweise Wird die Versorgung eingeschaltet, ist der Kondensator ungeladen und es liegt am Widerstand R_4 und R_5 die Versorgungsspannung an. R_4 stellt in der Startphase einen ausreichenden Strom an der Basis des Transistors ein, um diesen leitend zu schalten. Am Transistor fallen bei Raumtemperatur auf der Strecke Kollektor-Emitter dann zwischen 0,08 V und 0,1 V ab (Abb. 5.5, der Kollektorstrom über R_3 ist größenordnungsmäßig im niedrigem mA-Bereich). Der Kondensator wird geladen, dadurch sinkt die Spannung an R_4 und R_5 ab, der Basistrom und damit der Kollektorstrom wird geringer. Sobald die Basis-Emitterspannung unter einen bauteilspezifischen Wert fällt (laut Datenblatt $V_{BE}(sat) = 0{,}6\ V$ bis $0{,}8\ V$ bei Raumtemperatur), fließt kein Basisstrom mehr, der Transistor wird sperrend und damit fließt durch R_3 ebenfalls kein Strom mehr. Der Kondensator lädt sich über R_5 weiter auf, wodurch die Basis-Emitter-Spannung nach einiger Zeit auf GND liegt und der Transistor weiter sperrend bleibt.

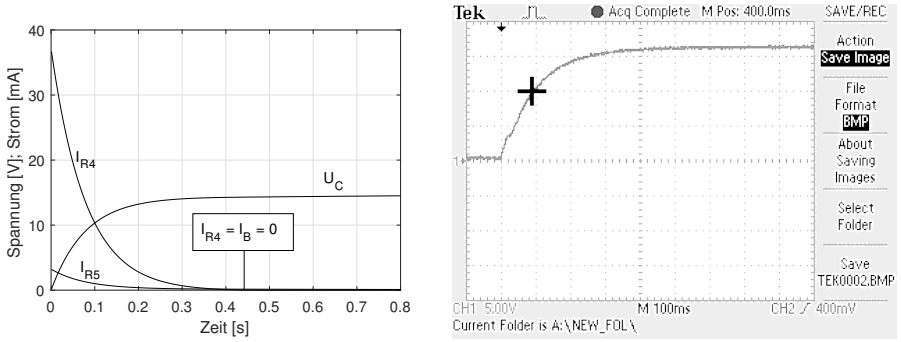


Abbildung 5.6: Sprungantwort der Startschaltung: Simulation und an der Platine gemessene Spannung U_C (Trigger durch Versorgung) (Anh. B.2.6)

Beeinflussungsdauer Die Dauer, in der der Transistor leitend ist, lässt sich schrittweise berechnen (MATLAB, Schrittweite 1 ms) und an der fertigen Platine messen. Die Zeitkonstante eines Kondensators mit Widerstand ist $\tau = R * C$. Lässt man den Transistor unberücksichtigt, gilt für $R = R4||R5 = 360\ \Omega$. Mit $C = 220\ \mu F$ ist die Zeitkonstante 0,08 s. Nach dieser Zeit sollten 63,2% der Endspannung erreicht sein. Das sind bei 15 V ca. 9,5 V. Die Grafiken in Abb. 5.6 zeigen, dass die Kondensatorspannung 10 V kurz vor 0,1 s überschreitet, und damit stimmen Simulation und Messung mit der vereinfachten theoretischen Betrachtung überein.

Nach knapp über 0,4 s wird der Basisstrom $I_B \leq 0\ A$, der Transistor wirkt auf der Strecke Kollektor-Emitter sperrend. Bereits nach 0,2 s ist der Basisstrom sehr gering. Diese Zeit entspricht bereits ca. 20.000 Kristallschwingungen (abhängig vom verwendeten Kristall), Tests mit angeschlossenem SCPEM zeigen, dass die Schaltung wie gewünscht funktioniert und den problemlosen Start der Schwingung ermöglicht.

Potentiometereinstellung Für die Widerstände zwischen Rückkopplung und GND sind $R_2 = 768\ \Omega$ und $R_3 = 2.390\ \Omega$ beim 80 kHz-Kristall gute Richtgrößen, die Funktion der Ansteuerung ist aber über einen weiten Widerstandsbereich gegeben.

5.3.2 Schaltplan und Bauteilliste

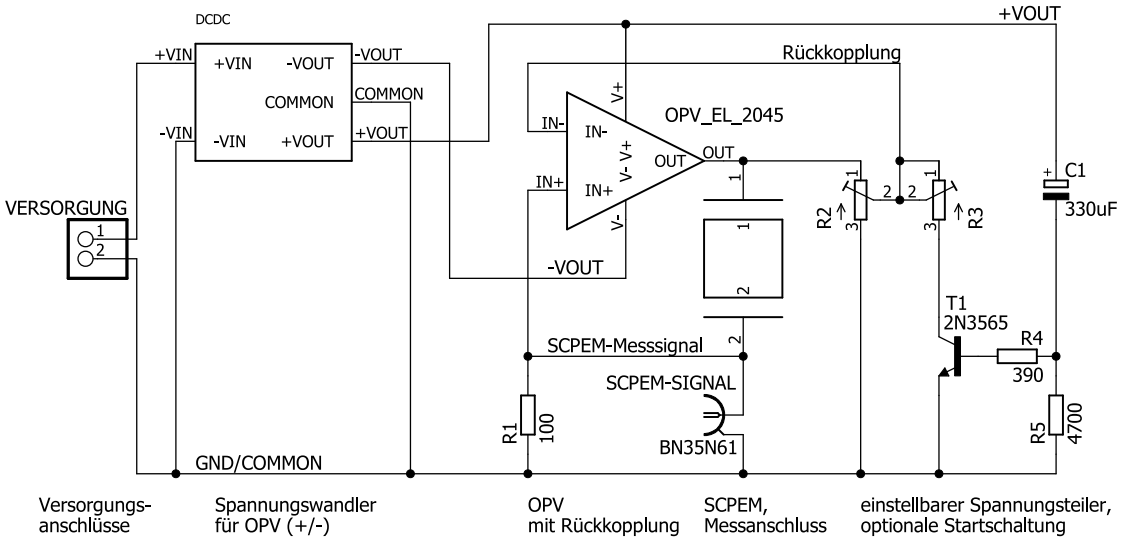


Abbildung 5.7: Neuer Schaltplan mit Startschaltung (Anh. B.3.2.1)

Name	Bauteil-Nr.	Bauteilwert	Beschreibung
DCDC	Traco TMH 2415D	$+24\ V \rightarrow \pm 15\ V$	symm. Versorgung des OPV
OPV	Intersil EL2045	-	auf SOIC zu DIP Adapter
-	-	-	DIP-Socket (8-Pin)
T1	Transistor MPSA42	Gain $h_{FE} = 25...40$	optional für Startvorgang
C1	Kondensator	$330\ \mu F$	optional für Startvorgang
R1	Widerstand	$100\ \Omega$	für SCPEM-Strommessung
R2	Potentiometer	$100\ k\Omega$	Einstellung Rückkopplung
R3	Potentiometer	$100\ k\Omega$	optional für Startvorgang
R4	Widerstand	$390\ \Omega$	optional für Startvorgang
R5	Widerstand	$4.700\ \Omega$	optional für Startvorgang
SCPEM-SIGNAL	BNC Through-Hole PCB Jack	-	Anschluss SCPEM-Signal
VERSOR-GUNG	Stiftleiste 1x2, Raster 0,1“	-	Anschluss Versorgung
SCPEM	Stiftleiste 1x2, Raster 0,1“	-	Anschluss SCPEM

Tabelle 5.1: Bauteilliste der SCPEM-Ansteuerung

5.3.3 Platine

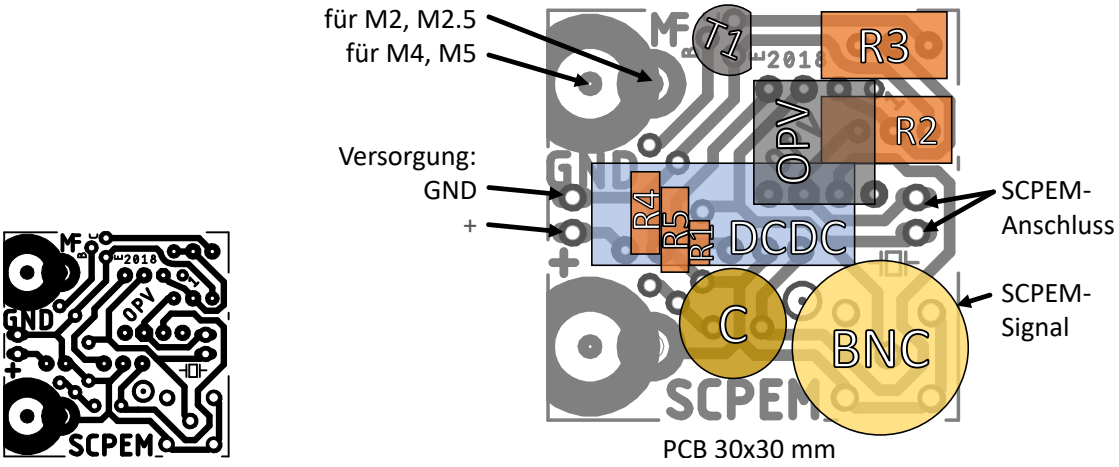


Abbildung 5.8: Layout in Originalgröße (links), mit Bauteilpositionen (rechts) (Anh. B.3.2.2)

Um den Testaufbau zu vereinfachen und Fehlerquellen durch schlechte Kontakte zu unterbinden, wird das Steckbrett durch eine geätzte Platine ersetzt. Der Schaltplan und das Layout werden mit Eagle 8.4.3 erstellt.

Die vier Verschraubungsbohrungen im Board sind so konzipiert, dass es mit unterschiedlichen Schraubengrößen (jeweils zwei Stück) befestigt werden kann. Der Abstand der Bohrungen ist an den optischen Aufbau angepasst und ermöglicht eine einfache Montage. Die Schraubenauflagen bleiben mit Kupfer beschichtet und sind mit GND verbunden. Die Platine ist einseitig geätzt, die Bauteile sind jedoch auf beide Seiten aufgeteilt, dadurch ist eine kompakte Größe von 30x30 mm mit kurzen Leiterbahnen möglich.

5.3.3.1 Herstellung

Die zur Verfügung stehenden Platinenstücke sind etwa 62x45 mm groß, daher ist es möglich, zwei gleiche Boards (je 30x30 mm) nebeneinander zu Ätzen. Um eine für die UV-Belichtung ausreichende Absorption der mit Laserdrucker bedruckten Folie zu erhalten, wird diese aus zwei bedruckten, übereinandergelegten und verklebten Folien hergestellt (beim mehrmaligen Drucken auf einer Folie ist die Ausrichtung nicht gewährleistet). Nach einem Testversuch liegt beim zweiten Ätzvorgang (Abb. 5.9) eine brauchbare Platine vor, bei der alle Bahnen durchgehend und gut voneinander isoliert sind.

Diese wird auf die richtige Größe geschnitten und die Löcher für die Befestigung und die Bauteile werden gebohrt. Auf der kupferbeschichteten Seite werden der OPV und die drei Widerstände angelötet, alle anderen Teile befinden sich auf der Rückseite und sollten bereits zuvor verlötet werden. Nach dem Bestücken aller Teile und der wiederholten Prüfung der Verbindung bzw. Isolation aller Pfade wird die Platine auf den Testaufbau

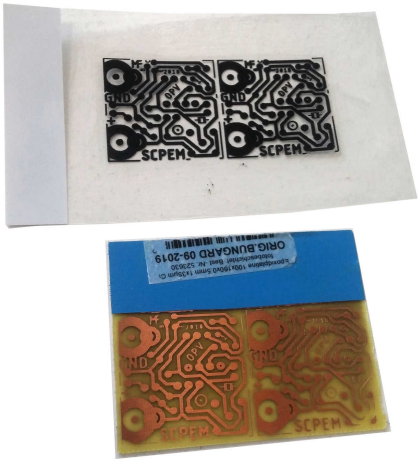


Abbildung 5.9: Transparente Folie mit aufgedrucktem Layout und halbfertige Platine nach Belichtungs- und Ätzvorgang

geschraubt und angeschlossen. Die Einstellung der Potentiometer kann nur mit angeschlossenem SCPEM erfolgen, dabei sind die Ziele einerseits ein möglichst störungsfreies SCPEM-Signal zu erhalten, bei dem keine hochfrequenten Oszillationen vorliegen (niedriger Widerstand R_2), andererseits einen sicheren Start in die richtige Resonanzfrequenz des SCPEM zu ermöglichen (hoher Widerstand R_3). Richtgrößen sind in Kap. 5.3.1.1 angeführt.

5.3.3.2 Ergebnis

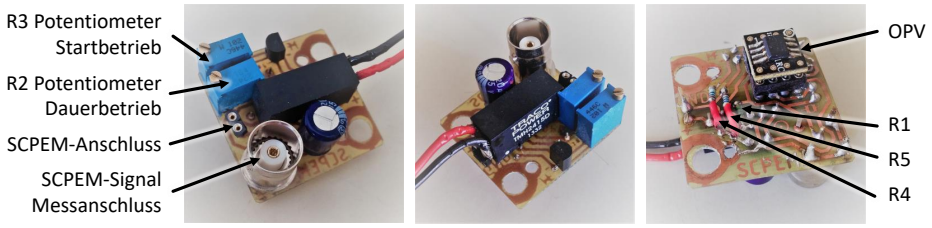


Abbildung 5.10: Ansteuerungsboard: Vorderseite, Vorderseite, Rückseite

Abb. 5.10 zeigt das fertig bestückte Board und die Position einiger Bauteile. Die beiden angelöteten Kabel dienen zur Spannungsversorgung, die Kabel des SCPEM werden in die vorgesehenen Anschlüsse gesteckt und das Messgerät mit einem BNC-Kabel verbunden (STEMlab-RF-Eingang 2). Das Potentiometer am Platinenrand (R_3) dient zur Einstellung des Startwiderstandes, das daneben befindliche Potentiometer (R_2) für die Rückkopplung im Dauerbetrieb. Der OPV sitzt für die einfachere Handhabung auf einem 8-Pin-DIP-Socket und kann vom Board abgenommen werden.

5.3.4 Zusammenhang Versorgungsspannung - Steuerspannung

Es wird die Versorgungsspannung der Ansteuerung variiert, um die gewünschte Retardation am SCPEM zu erreichen. Dabei besteht ein Zusammenhang zwischen der Versorgungsspannung und der Sättigungsspannung am Verstärkerausgang, welche am SCPEM und R_1 anliegt.

5.3.4.1 Spannungswandler (DCDC)

Um die Ausgangsspannungen von ±15 V zu erhalten, ist der Spannungswandler mit 24 V zu versorgen. Der Spannungswandler ist auf diesen Arbeitspunkt ausgelegt, daher ist im Datenblatt keine Kennlinie der Ausgangsspannungen in Abhängigkeit der Versorgung vorhanden.

Für die Aufnahme der Kennlinien werden das Netzteil ISO-TECH IPS601A und das Digital-Multimeter ISO-TECH IDM71 verwendet.

Es wird die Lehrlaufspannung ohne Last ermittelt, Abb. 5.11. Die Versorgung wird schrittweise gesteigert und gemessen, anschließend werden die Ausgangsspannungen gemessen. Der COMMON-Anschluss wird dabei, wie in der Anwendung auf der Ansteuerungsplatine, auf Masse gelegt.

Da sich der Spannungswandler während der Messung leicht erwärmt, wird nach auskühlen eine weitere Messreihe durchgeführt, wobei jetzt bei der höchsten Spannung begonnen wird. Dadurch wäre ein eventueller Temperatureinfluss in der Auswertung sichtbar, das ist jedoch nicht der Fall.

Versorgung	V+	V-
2,7	1,808	1,807
4,04	2,755	2,754
5,514	3,683	3,679
7	4,764	4,759
8,53	5,462	5,458
10,08	6,36	6,36
12,04	7,74	7,73
13,97	9,15	9,14
16,13	10,7	10,69
17,97	12,01	12,01
20	13,44	13,45
22,02	14,87	14,88
23,07	15,59	15,6
24	16,22	16,25
25,04	16,92	16,95
25,95	17,51	17,56
28,01	18,84	18,92
30	20,13	20,28
Testdaten (2. Messung)		
2,218	1,766	1,759
5,199	3,449	3,446
10,64	6,76	6,75
14,87	9,79	9,78
19,75	13,26	13,27
24	16,22	16,24
28,04	18,84	18,93
Näherungsfunktion		
0	0	0
30	20	20

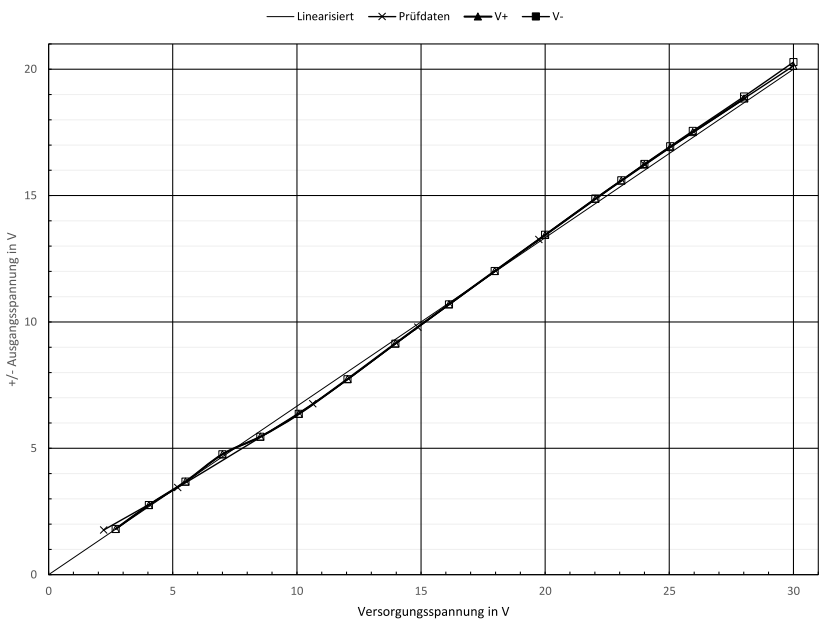


Abbildung 5.11: Ermittelter Zusammenhang der Ausgangsspannung mit der Versorgungsspannung des DC-DC-Konverters Traco TMH 2415D (Anh. B.1.5)

Die Kennlinie lässt sich durch eine Gerade mit der Steigung 2/3 gut annähern (Abb. 5.11). Unbelastet gilt für die Ausgangsspannungen des Spannungswandlers $V_{OUT} = \pm\frac{2}{3} * V_{IN}$.

5.3.4.2 Operationsverstärker (OPV)

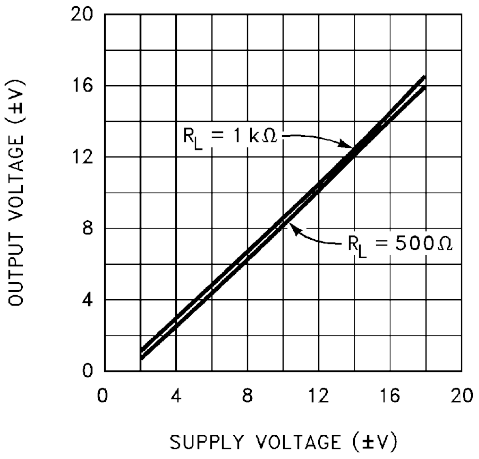


Abbildung 5.12: Ausgangsspannung des OPV in Abhängigkeit von der Versorgung (Anh. A.2.5)

Für den verwendeten Operationsverstärker EL2045 liegt die Kennlinie von Versorgungs- zu Ausgangsspannung vor. Für das Erreichen der Sättigungsspannung muss an den Eingängen eine ausreichende Differenzspannung vorliegen und die Frequenz darf eine bestimmte Grenze nicht überschreiten

Kap. 5.3.5 zeigt, dass diese Bedingungen erfüllt sind und sich somit näherungsweise folgender Zusammenhang ergibt (aus der Kennlinie Abb. 5.12):

$$V_{OUT} = \pm\frac{16}{18} * V_S = \pm\frac{8}{9} * V_S$$

5.3.4.3 Gesamteinfluss

Da sich bei beiden Bauteilen, die die Ausgangsspannung des OPV direkt beeinflussen, eine lineare Funktion ohne Offset gezeigt hat, kann der Zusammenhang einfach durch

$$V_{SCPEM} = \pm\frac{2}{3} * \frac{8}{9} * V_{Versorgung} = \pm\frac{16}{27} * V_{Versorgung} = \pm 0{,}59 * V_{Versorgung}$$

ausgedrückt werden. Die am SCPEM und R_1 anliegende Wechselspannung ist somit proportional zur Versorgungsspannung der Schaltung.

5.3.5 OPV-Verhalten als astabiler Multivibrator

Der Operationsverstärker EL2045 ist für geringe Leistung und hohe Frequenzen geeignet. Während diese Anwendung abhängig vom verwendeten Kristall bei etwa 0,1 MHz liegt, ist der OPV problemlos für Frequenzen über 1 MHz geeignet. Selbst beim Betrieb als Kippstufe mit Ausgangsspannung im Sättigungsbereich tritt bis $f = 3\ MHz$ keine Änderung der Peak-to-Peak-Spannung auf, wie Abb. 5.13 zeigt.

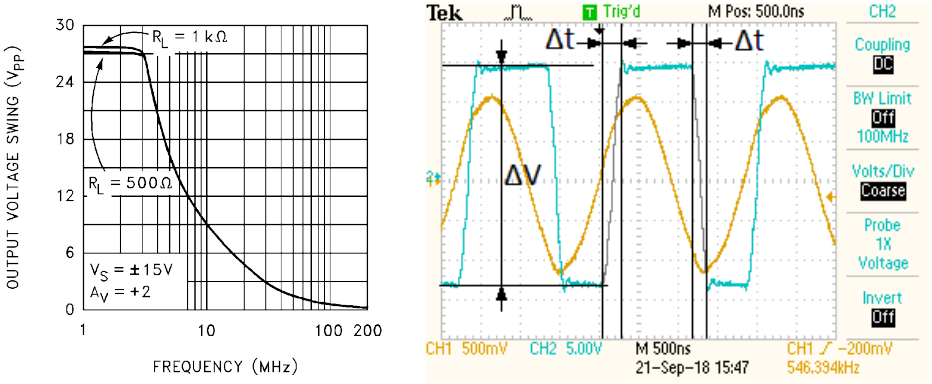


Abbildung 5.13: Ausgangsspannung des OPV in Abhängigkeit von der Frequenz (Anh. A.2.5); Analyse des tatsächlichen Spannungsanstiegs am SCPEM (Anh. B.1.4)

Die Slew Rate für eine rein ohmsche Last liegt laut Datenblatt bei min. 200 $V/\mu s$. Beim maximalen V_{PP} von 32 V beträgt die Anstiegsdauer 0,16 µs.

Der SCPEM ist keine ohmsche Last, er besitzt zusätzlich einen kapazitiven und induktiven Anteil. Die tatsächliche Slew Rate kann zum Beispiel aus der Aufnahme in Abb. 5.3 gewonnen werden, wie in Abb. 5.13 rechts dargestellt. Diese hat aufgrund der höheren SCPEM-Frequenz eine höhere zeitl. Auflösungen als andere Messungen.

Der Spannungsanstieg/Abfall beträgt etwa 27,5 V und die Anstiegs-/Abfallsdauer zwischen 0,18 und 0,24 μs. Damit ergibt sich eine Spannungsänderungsrate zwischen 115 und 153 $V/\mu s$. Die halbe Periodendauer bei 100 kHz beträgt zum Vergleich 5 μs und damit ein Vielfaches der Anstiegsdauer. Das Spannungssignal am Verstärkerausgang ist somit ein Trapezsignal mit steilen Flanken, das näherungsweise als Rechtecksignal betrachtet werden kann.

6 Kommunikation Messboard-Computer

6.1 Zuständigkeiten der Geräte

Das Messsystem sieht zwei Geräte für die Messung und Auswertung der Signale von SCPEM und Infrarot-Photodiode vor. Einerseits das STEMlab-Messboard, andererseits einen Computer, der eine einfache Ein- und Ausgabe von Einstellungen und Informationen ermöglicht und weitere Aufgaben übernehmen kann. Je nach Aufteilung der Zuständigkeiten sind die zu übermittelnden Daten unterschiedlich. Die wichtigsten Varianten sind in Tab. 6.1 zusammengefasst.

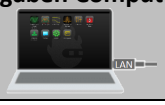
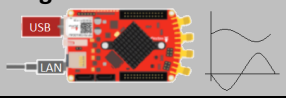
Variante	**Aufgaben Computer** 	**Datenübertragung** Ethernet 1 Gbit	**Aufgaben STEMlab-Board** 
Variante 1 Verarbeitung PC	Benutzeroberfläche Einstellungen erfassen Einstellungen verarbeiten Daten erhalten Auswertung der Daten Datenmanagement	grundlegende API-Befehle (Start, Stopp) ---------->	Sample-Daten verfügbar machen
		<---------- Messsamples aus Puffer	
Variante 2 Verarbeitung STEMlab	Benutzeroberfläche Einstellungen erfassen Einstellungen übermitteln Benachrichtigungen anzeigen	alle Einstellungen Befehle ---------->	Einstellungen erhalten Einstellungen verarbeiten Sample-Daten einlesen Auswertung der Daten Datenmanagement Benachrichtigungen übermitteln
		<---------- Informationen Benachrichtigungen	
Variante 3 Kombination	Benutzeroberfläche Einstellungen erfassen Einstellungen übermitteln Auswertung der Daten Datenmanagement Benachrichtigungen anzeigen	alle Einstellungen Befehle ---------->	Einstellungen erhalten Einstellungen verarbeiten Sample-Daten einlesen Auswertung der Daten Benachrichtigungen übermitteln
		<---------- vorbereitete Daten Informationen Benachrichtigungen	
Variante 4 STEMlab eigenständig	-	keine	Einstellungen aus Datei Einstellungen verarbeiten Sample-Daten einlesen Auswertung der Daten Datenmanagement

Tabelle 6.1: Zuständigkeitsaufteilung zwischen Computer und STEMlab-Board (Grafiken aus [19])

Variante 1 ist am schnellsten testbar. Die Aufgaben des STEMlab-Boards sind bereits durch die API abgedeckt, die Übertragung erfolgt ebenfalls durch fertige API-Befehle. Die Samples können dann in der gewünschten Programmierumgebung weiterverarbeitet werden, Änderungen sind sehr einfach möglich. Die Hardware ist leistungsstark und spezielle Optimierungen des Codes nachrangig. Die kontinuierliche Übertragung der Daten von zwei Eingängen mit der benötigten Samplerate ist nach Kap.6.3 nicht möglich, daher kann diese Methode nicht verwendet werden.

Die anderen Varianten zeichnen sich durch ein eigenes Programm am STEMlab-Board aus, das die Aufgabe des Einlesens vom Puffer übernimmt und diese Daten sofort auswertet. Das Protokoll zur Übertragung kann frei festgelegt und die zu übertragende Datenmenge stark reduziert werden.

Bei Variante 2 werden alle Vorgänge am STEMlab-Board durchgeführt, der Computer dient zur Eingabe der Einstellungen und zur Anzeige von Informationen zur aktuellen Aufnahme. Die Daten werden lokal am STEMlab-Board gespeichert. Variante 3 verarbeitet die Samples aus dem Puffer und sendet die Ergebnisse an den Computer, wo diese gespeichert oder weiter ausgewertet werden können.

Variante 4 stellt ein eigenständiges Programm am STEMlab-Board dar, die Einstellungen sind vorgegeben, die Ergebnisse werden am STEMlab-Board aufgezeichnet. Es wird kein zusätzlicher Computer benötigt. Da während der Entwicklung die Möglichkeit von schnellen Änderungen der Einstellungen und Zugriff auf Informationen zur Messung erwünscht ist, wird diese Methode nicht umgesetzt. In der fertigen Version kann durch wenige Änderungen statt der Übermittlung der Einstellungsbefehle vom Computer ein Lesevorgang aus einer Datei mit den gewünschten Einstellungen erfolgen und vollständige Eigenständigkeit des Messboards erreicht werden.

Es werden sowohl Variante 2 als auch Variante 3 umgesetzt. Es besteht die Möglichkeit, die Messdaten am STEMlab-Board zu speichern und/oder diese an den Computer zu senden und dort zu verarbeiten. Zur Betrachtung und Analyse der Daten ist die Übertragung auf den Computer jedenfalls nötig, der Vorgang kann jedoch nach Abschluss der Messung erfolgen.

6.2 Schnittstellen am STEMlab-Board

Abb. 6.1 zeigt verschiedene Möglichkeiten zur Datenverbindung zwischen dem Computer und dem STEMlab-Board. [19]

6.2.1 Universal Serial Bus (USB)

Das STEMlab-Board besitzt mehrere Anschlüsse nach dem USB-Standard:

- 1x Micro USB Type B: Spannungsversorgung
- 1x Micro USB Type B: Console (USB)-Verbindung, Datenübertragung
- 1x USB Type A 2.0: WLAN-Dongle, USB-Flashspeicher

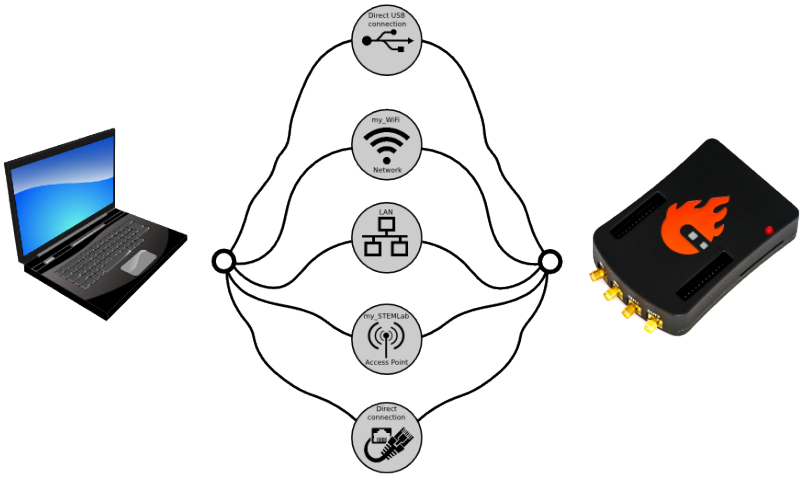


Abbildung 6.1: Schnittstellen zur Kommunikation mit dem STEMlab-Board [19]

USB 2.0 ermöglicht eine maximal mögliche Datenübertragungsrate von 480 Mbit/s. In der STEMlab-Dokumentation [19] wird die Verwendung als serielle Schnittstelle erklärt, die Verwendung eines WLAN-Dongle ist ebenso möglich.

Diese Schnittstellen werden vom Messprogramm nicht zur Kommunikation verwendet, es besteht jedoch die Möglichkeit, die Messdaten direkt auf einen Wechseldatenträger am USB Type A - Port zu speichern. [19, 23]

6.2.2 Ethernet

Es steht eine RJ45-Buchse am STEMlab-Board und am Computer zur Verfügung. Die Verbindung ist direkt oder indirekt über einen Router/ein Netzwerk mit Datenraten bis 1.000 Mbit/s realisierbar, der verwendete Standard ist 1000BASE-T.

Für die Kommunikation und Datenübertragung wird eine direkte Ethernet-Verbindung genutzt, die dafür nötigen Einstellungen sind in Kap. 9.1.1 angeführt. [19, 24]

6.3 SCPI-Server und RedPitaya-API

Das STEMlab-Board kann über LAN mit SCPI (Standard Commands for Programmable Instrumentation) gesteuert werden bzw. wird damit das Einlesen der Daten am Computer ermöglicht. Es stehen eine Vielzahl an SCPI-Befehlen zur Verfügung, die in unterschiedlichen Programmierumgebungen am Computer genutzt werden können, in denen dann die Auswertung der Daten erfolgt. Dadurch stehen einerseits die vielseitigen Funktionen dieser Programmierumgebung zur Verfügung, andererseits ist die Rechenleistung dieses Gerätes anpassbar und nicht auf die Hardware des Messboards beschränkt. [19]

Nachteilig ist, dass in diesem Fall nicht nur die Berechnungsergebnisse, sondern alle Samples über LAN übermittelt werden müssen. Es zeigt sich schnell, dass auf diesem

Weg keine kontinuierliche Aufzeichnung möglich ist. Hier sollen nur einige kurze Beispiele gezeigt werden, wie die Kommunikation aussehen kann und auf welche Grenzen sie stößt.

Vor jeder Verwendung ist in den STEMlab-Einstellungen (im Browser) unter Development der SCPI-Server zu aktivieren.

Die Messungen werden jeweils mehrmals wiederholt, die Ergebnisse unterschieden sich nur um einzelne Millisekunden. Die gemessene Lesedauer (Kap. 6.3.1 und Kap. 6.3.2) beträgt jeweils über einige 100 ms je Kanal, bei Dezimation 64 sind jedoch max. 8,389 ms für eine kontinuierliche Messung zulässig, da dann bereits der gesamte Puffer gefüllt ist. In dieser Zeit müssen beide Kanäle eingelesen und ausgewertet sein, daher ist dies selbst mit möglichen Optimierungen kein Weg, der weiterverfolgt werden sollte (die Übertragung müsste für die gewünschte Anwendung mindestens 60-mal so schnell erfolgen).

6.3.1 MATLAB mit SCPI-Server

Anzahl einzulesende Samples	Dauer Einlesen	Dauer inkl. Konvertierung
1 Sample	280 ms	300 ms
gesamter Puffer (16.384 Samples)	289 ms	990 ms

Tabelle 6.2: Einlesedauer von Samples aus dem STEMlab-Puffer mittels SCPI in MATLAB (Anh. B.1.6.1)

Das Testprogramm wird ähnlich dem MATLAB-Beispiel in 2.4.4.4.1.4. in der STEMlab-Dokumentation realisiert. [19]

Nach Änderung der Dezimation auf 64 und des Triggers auf „NOW“ werden die Samples wie im Beispielprogramm eingelesen und konvertiert. Da die eingelesenen Werte zunächst als char-String vorliegen, ist die zeitintensive Konvertierung nötig. Es wird dazu der Code aus dem STEMlab-Beispiel verwendet, eine Optimierung würde die Konvertierungszeit evtl. reduzieren. In Tabelle 6.2 sind die gemessenen Zeiten für das Übertragen der Daten und die Konvertierung angegeben.

6.3.2 Python mit SCPI-Server

Anzahl einzulesende Samples	Dauer Einlesen	Dauer inkl. Konvertierung
1 Sample	Fehler	Fehler
gesamter Puffer (16.384 Samples)	240 ms	250 ms

Tabelle 6.3: Einlesedauer von Samples aus dem STEMlab-Puffer mittels SCPI in Python (Anh. B.1.6.2)

Das Testprogramm wird ähnlich dem Python-Beispiel in 2.4.4.4.1.6. in der STEMlab-Dokumentation realisiert. [19]

Es ist die Installation von pyvisa-py nötig (`pip install pyvisa pyvisa-py`), die redpitaya_scpi.py-Datei muss im selben Ordner liegen. Zeile 88 dieser Datei wurde geringfügig geändert, um die Ausführung zu ermöglichen.

In der Beispieldatei wird in Zeile 7 die IP-Adresse des STEMlab-Boards eingegeben und anschließend werden die gleichen Messungen wie im MATLAB-Script durchgeführt.

Während die Abfrage eines einzelnen Samples stattdessen eine Liste mit 16.384 Samples inklusive einer Error-Meldung liefert und damit keine aussagekräftige Dauer bestimmt werden kann, ist die Erfassung des gesamten Puffers etwas schneller als mit MATLAB, die Konvertierung erfolgt mit 10 ms sehr viel schneller als in MATLAB mit 700 ms, siehe Tabelle 6.3.

6.4 Anwendungsspezifisches STEMlab-Programm

Da die RedPitaya-API nicht für die benötigte Datenübertragung geeignet ist, muss eine andere Lösung gefunden werden. Ziel ist es, die Daten möglichst schnell für die Auswertung zur Verfügung zu haben, um in einem weiteren Schritt damit eine schnelle Regelung einer Prozessgröße verwirklichen zu können.

Ein Programm am STEMlab-Board kann durch die Berechnung der gewünschten Daten aus allen vorhandenen Informationen bereits vor dem Senden einerseits die Datenmenge reduzieren und andererseits die Art der Übertragung selbst bestimmen.

Eine einfache, jedoch unsaubere und langsame Lösung ist das Abspeichern der Daten am Messboard und das anschließende Übertragen dieser Dateien an den Computer, während die Messung aktiv ist, Kap. 6.4.1.

Aus der Internetprotokollfamilie stehen die Netzwerkprotokolle UDP (User Datagramm Protocol) und TCP (Transmission Control Protocol) zur Verfügung, Kap. 6.4.2.

6.4.1 Übertragung von Dateien per Secure Copy (SCP)

SCP ist ein Protokoll zur Übertragung von Daten, das auf SSH basiert und den Transport über TCP abwickelt. Dadurch können ganze Dateien einfach zwischen verschiedenen Geräten kopiert/verschoben werden.

Das Programm am STEMlab-Board schreibt die Daten in eine lokale Datei. Dabei werden jeweils nur einige Tausend Messwerte in eine Datei geschrieben, sodass mehrere Dateien pro Sekunde erstellt werden. Diese haben eine fortlaufende Nummer als Dateinamen.

Das Computer-Programm fragt per SSH in regelmäßigen Abständen die Dateiliste am Board ab, ist eine neue Datei vorhanden, kann diese mit SCP kopiert und gelesen werden.

Die Übertragungsgeschwindigkeit ist ausreichend und die Daten werden entweder sicher übertragen oder liegen noch am STEMlab-Board vor, dadurch ist ein Datenverlust ausgeschlossen. Es ergibt sich eine große Verzögerung, da die Datei abgeschlossen sein muss, bevor sie übertragen werden kann.

Nachdem diese Variante anfangs erfolgreich eingesetzt wird und für langsame Testmessungen ausreichend ist, wird sie später nicht weiterverfolgt.

6.4.2 Datenübertragung mit Transportprotokoll

6.4.2.1 Transportprotokolle TCP/IP und UDP

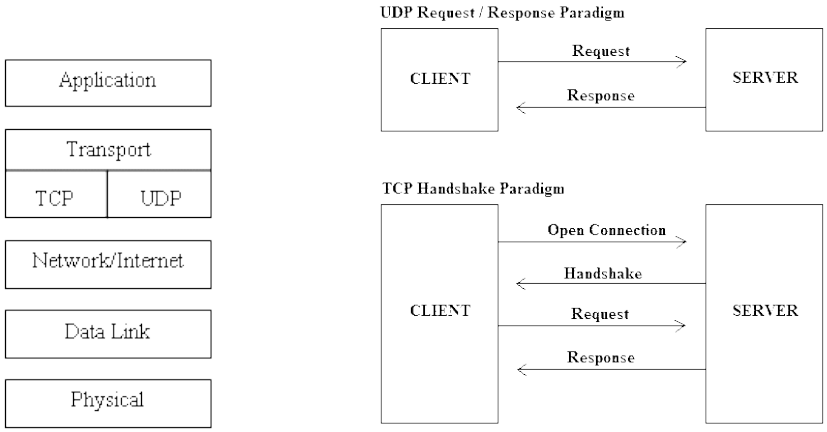


Abbildung 6.2: Fünfschichtiges TCP/IP-Architekturmodell [25]; Request/Response-Modell von UDP und TCP [25]

In der Internet-Protocol-Familie (Architekturmodell Abb. 6.2) stehen zwei Transport-Protokolle zur Verfügung, die sehr häufig verwendet werden. TCP macht etwa 75% des weltweiten Internetverkehrs aus, UDP rund 20%. Der Grund für die Verwendung verschiedener Protokolle ist die unterschiedliche Zielsetzung: TCP garantiert die sichere Übertragung der Daten, während UDP eine hohe Übertragungsgeschwindigkeit sicherstellt. Beide Protokolle wurden bereits zwischen 1970 und 1980 entwickelt. [26, 27, 25]

Transmission Control Protocol (TCP/IP) Transmission Control Protocol/Internet Protocol (TCP/IP) ist ein verbindungsorientiertes Protokoll und stellt sicher, dass die Daten beim Empfänger ankommen, indem der Empfänger dies bestätigen muss. Das Abwarten auf die Antwort benötigt in jedem Fall Zeit, bei Datenverlust werden die Daten erneut gesendet und die Antwort abgewartet. Die richtige Reihenfolge der Nachrichten beim Empfänger wird vom Protokoll sichergestellt. Je Paket ist ein Header mit 20 bis 60 Byte nötig.

Die Empfangsprüfung und das erneute Senden kann im Programm zu Verzögerungen führen, die in diesem Fall das rechtzeitiges Einlesen des Sample-Puffers verhindern und somit zu Datenverlust führen können. [25, 26]

User Datagram Protocol (UDP) User Datagram Protocol (UDP) ist verbindungslos und nachrichtenorientiert. Die Daten werden sofort gesendet und das Programm fortgesetzt, ohne abzuwarten, ob diese beim Empfänger angekommen sind. Der UDP-Header ist nur 8 Byte lang. Nachteilig an UDP ist der mögliche Verlust von Paketen und die nicht garantierte Reihenfolge beim Empfangen, wodurch eigene Funktionen im Programm diese Kontrolle übernehmen müssen. [25, 27]

Gegenüberstellung TCP/UDP TCP wird für Datenübertragungen genutzt, bei denen die Daten kritisch sind und der Verlust sehr problematisch ist. UDP dagegen ist sehr schnell und wird für echtzeitfähige Anwendungen oder Streaming verwendet. Der Ressourcenverbrauch von UDP ist wesentlich geringer.

Da für das Senden der Daten nur begrenzte Zeit zur Verfügung steht, wird als Datenprotokoll auf UDP gesetzt.

Beim Verwenden des Protokolls zeigt sich, dass bei geeigneten Einstellungen und Maßnahmen wie mehrmaligem, redundanten Senden die Verlustrate vernachlässigbar klein wird (Verluste treten nur bei Einstellungen für besonders schnellen Sendevorgänge auf), während die für die Verarbeitung und das Senden benötigte Zeit am STEMlab-Board minimal gehalten werden kann. [25, 26, 27]

6.4.2.2 UDP-Performance mit MATLAB und Python

Es werden einige charakteristische Zeiten in MATLAB und Python gemessen und in Tabelle 6.4 dargestellt. Das C-Programm am STEMlab-Board importiert und verwendet das Socket-Modul für die UDP-Kommunikation.

In MATLAB wird dazu die MATLAB-Funktion udp() verwendet, welche die Kommunikation in JAVA realisiert, in Python wird ein Socket-Objekt aus dem Socket-Modul erstellt.

Mit 64-Byte-Paketen wird getestet, wie lange es dauert, bis der Computer ein Paket auf das STEMlab-Board gesendet und dieses wieder ein Paket an den Computer zurückgeschickt hat, wie lange das Empfangen am Computer für mehrere Pakete dauert, und wie lange das Senden am STEMlab-Board benötigt.

Da in Python keine Möglichkeit gefunden wurde, die Zeit in Mikrosekunden zu messen, muss die Dauer in Millisekunden angegeben werden und kann damit bei kurzen Zeitabständen nur geschätzt werden. Die jeweils besten Ergebnisse mehrerer Versuche sind in Tabelle 6.4 aufgelistet.

Kriterium	MATLAB	Python
Computer-STEMlab-Computer: Rückmeldung 1 Paket	19 ms	0-1 ms
Computer: Empfangen 100 Pakete	226 ms	0-3 ms
Computer: Empfangen 1.000 Pakete	2.260 ms	26 ms
STEMlab: Empfangen 1 Paket	0,015 ms	0,015 ms
STEMlab: Senden 100 Pakete	1,729 ms	1,706 ms
STEMlab: Senden 1.000 Pakete	16,801 ms	16,842 ms

Tabelle 6.4: Vergleich der UDP-Kommunikation: MATLAB und Python (Anh. B.1.7)

Die Geschwindigkeit beim Einlesen ist in Python um rund zwei Größenordnungen höher als in MATLAB. Die Sendezeiten am STEMlab-Board sind nicht vom Empfängerprogramm abhängig, was aufgrund der Eigenschaften von UDP vorhersehbar war und so erwünscht ist. Würde der Test länger laufen, würde sich der UDP-Puffer am Computer füllen und die Pakete verloren gehen, weil das STEMlab-Board schneller sendet, als die

Pakete gelesen werden. Das Senden eines Paketes dauert ca. 17 µs, das Empfangen/Einlesen eines kleinen Paketes (2 Byte) am STEMlab-Board 15 µs. Das Empfangen/Einlesen am Computer dauert je Paket etwa 2.260 µs in MATLAB oder 26 µs in Python.

6.4.2.3 UDP-Performance bei unterschiedlichen Paketgrößen

Da die Dauer des Sendevorganges am STEMlab-Board von der Paketgröße abhängig ist, wird diese ermittelt und in Tab. 6.5 gezeigt. Diese Daten lassen den Schluss zu, dass jeder Aufruf der Sendefunktion rund 16 µs und zusätzlich für jedes Byte (ohne Header) 9 ns anfallen. Mit der Paketgröße pkgsize in Byte gilt für die Sendedauer näherungsweise:

$T_{senden}(pkgsize) = T_{fix} + pkgsize * T_{variabel} = 16.000\ ns + pkgsize * 9\ ns$

Die gemessenen Werte und diese Näherung sind in Abb. 6.3 dargestellt und stimmen näherungsweise überein.

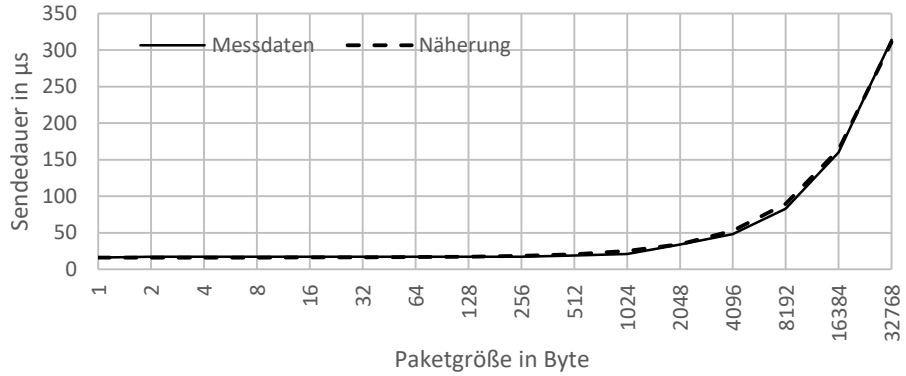


Abbildung 6.3: Sendedauer eines Paketes in Abhängigkeit von seiner Größe (Anh. B.1.8)

Die Dauer des Funktionsaufrufs zum Senden entspricht ungefähr der Sendedauer von 1.777 zusätzlichen Bytes. Aufgrund der kleinen Paketgrößen, die bei der Messung zu erwarten sind (einige bis einige Dutzend Messergebnisse zu je 2 Byte), besteht die Möglichkeit, die selben Daten in mehreren Paketen nacheinander zu senden und so den Ver-

log. Paketgröße in Byte (Basis 2)	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Paketgröße in Byte	1	2	4	8	16	32	64	128	256	512	1.024	2.048	4.096	8.192	16.384	32.768
Sendedauer in μs	16	17	17	17	17	17	17	17	17	19	21	34	48	83	160	314

Tabelle 6.5: STEMlab-C-Programm: gemittelte Sendedauer eines Pakets per UDP bei unterschiedlichen Paketgrößen (Anh. B.1.8)

lust eines Paketes durch diese Redundanz ausgleichen zu können, ohne die Sendedauer signifikant zu erhöhen.

Bei 9 ns/Byte beträgt die Übertragungsrate 111,111 $MByte/s$ oder 0,888 $Gbit/s$. Es wird das theoretische Limit der 1 Gbit-LAN-Verbindung zum größten Teil ausgeschöpft.

6.4.2.4 Protokolleigenschaften und -aufbau

Kap. 6.4.2.3 zeigt, dass es bei Verwendung von UDP signifikant schneller ist, verschiedene Informationen und Daten zusammenzufassen und gesammelt in einem Paket zu senden als jede Information einzeln zu senden. Daher wird ein Protokoll definiert, dass diese Daten nach einem bestimmten Schema aufnimmt und vom Empfänger wieder in seine Bestandteile zerlegt werden kann.

Möglichkeiten eines eigens angepassten Protokolls Während für eine Regelungsaufgabe einzelne verlorene Daten gegenüber der Ermöglichung einer möglichst hohen Paketfrequenz toleriert werden können, ist dies bei kontinuierlicher Aufzeichnung zu vermeiden. Um beide Situationen erfüllen zu können, wird ein eigenes Protokoll als UDP-Paket gesendet. Dadurch bieten sich folgende Möglichkeiten:

- Senden der Messdaten und Status-Informationen in einem Paket
- Senden von kleinen Paketen bei maximaler Sendefrequenz mit der Möglichkeit von Datenverlust für Regelungsaufgaben
- Senden von größeren Paketen bei geringerer Sendefrequenz mit der Möglichkeit von nachträglicher Datenkorrektur durch Redundanz der Daten für die Aufzeichnung und spätere Analyse

Beeinflussung der Eigenschaften im Messprogramm Auf der Benutzeroberfläche des Messprogramms stehen zwei Einstellungen zur Verfügung, um Geschwindigkeit und Datensicherheit der Übertragung anzupassen:

- Minimaler zeitlicher Sendeabstand: Das Programm schickt vorhandene Daten so schnell wie möglich an den Computer. Da diese Frequenz (abhängig von den Einstellungen) im zweistelligem kHz-Bereich liegen kann, kommt es teilweise zu Paketverlusten. Daher kann eine Mindestzeit vorgegeben werden, in der die Daten gesammelt und erst bei Erreichen dieser Dauer seit dem letzten Sendevorgang gemeinsam gesendet werden. Es haben sich Zeiten $\geq$ 500 μs (2.000 Pakete/s) für schnelle und $\geq$ 2.000 μs (500 Pakete/s) für langsamere Messungen bewährt, diese Einstellungen hängen aber von der verwendeten Hardware ab und können ggf. auch geringer gewählt werden. Wird keine Mindestzeit vorgegeben, werden die Pakete gesendet, sobald neu berechnete Daten vorliegen, wodurch die mittlere Zeitdifferenz unter 100 μs (über 10.000 Pakete/s) fallen kann und es häufig zu Paketverlusten kommt.

- Redundanz (mehrfaches Senden der gleichen Daten): Das Gesamtpaket (Abb. 6.6) ist so aufgebaut, dass es zusätzlich zu den aktuellen Daten eine bestimmte Anzahl der zuletzt gesendeten Datenpakete enthalten kann. Gehen Gesamtpakete verloren, kann der Empfänger aus dem nächsten erhaltenen Paket die vorhergehenden rekonstruieren. Gehen mehr Pakete hintereinander verloren, als die Redundanz beinhaltet, kommt es dennoch zum Datenverlust. Es wird in dem Fall durch die Paketstruktur bemerkt, dass Daten verloren gegangen sind und wie viele es exakt waren. Dadurch bleibt die zeitliche Zuordnung der nachfolgenden Daten erhalten.

Umsetzung und Aufbau des Protokolls Es ist darauf zu achten, dass die Anzahl von Messwerten je Paket unterschiedlich ist. Dadurch müssen die Indizes verschiedener Daten im Paket zusätzlich übermittelt werden, da diese nicht festgelegt werden können. Der genaue Aufbau der gesendeten Pakete inklusive der Header der Transportschicht wird in Tab. 6.6 gezeigt. Darin bedeutet n die Redundanz (0... nur aktuelle Daten werden gesendet, 1... einfache Redundanz, usw.), N ist ein fortlaufender Index, der mit jedem Sendevorgang erhöht wird.

Während das IP-Datagramm und das UDP-Datagramm vom Socket generiert werden, werden die UDP-Daten vom STEMlab-Programm selbst erstellt. Das Programm am Computer zerlegt diese Daten wieder in seine Bestandteile und verarbeitet sie weiter.

Der Gesamtheader des Gesamtpaketes (8Byte) beinhaltet die Gesamtlänge des Paketes (ohne Netzwerk-Header), die Redundanz und eine fortlaufende Nummer, die mit jedem gesendetem Gesamtpaket erhöht wird.

Danach folgen die Startindizes der einzelnen Pakete im Gesamtpaket. Damit weiß das Empfangsprogramm, wo diese Pakete im Gesamtpaket zu finden sind. Da der Index des ersten Pakets immer gleich ist (direkt nach dem Header, Startindex 8), wird dieser nicht mitgeschickt. Ist die Redundanz Null, werden jeweils nur die aktuellen Daten gesendet, und dieser Teil des Paketes entfällt.

Danach folgen je nach Redundanz ein oder mehrere Pakete, wobei das letzte Paket (Paket_N) die aktuellsten Daten beinhaltet. Diese enthalten den Messindex des letzten Messwertes dieses Paketes, gezählt vom aktuellen Aufzeichnungsbeginn, und dessen Zeitstempel in Mikrosekunden, ebenfalls vom Aufzeichnungsbeginn gemessen. Durch diese Implementierung entsteht die zeitliche Beschränkung einer einzelnen, durchgehenden Aufzeichnung. Mit der 32 bit-unsigned-Integer können $2^{32} = 4{,}295 * 10^9$ Werte dargestellt werden können. Das entspricht somit knapp 4.295 s oder 71:35 min maximaler Aufzeichnungsdauer je Messung. An den Zeitstempel anschließend folgen die Anzahl der Messwerte und Informationen sowie diese selbst. Im Normalfall (kein Paketverlust) wird nur das Paket_N verwendet, da die Daten der anderen Pakete bereits aus den vorhergehenden Gesamtpaketen am Computer vorhanden sind.

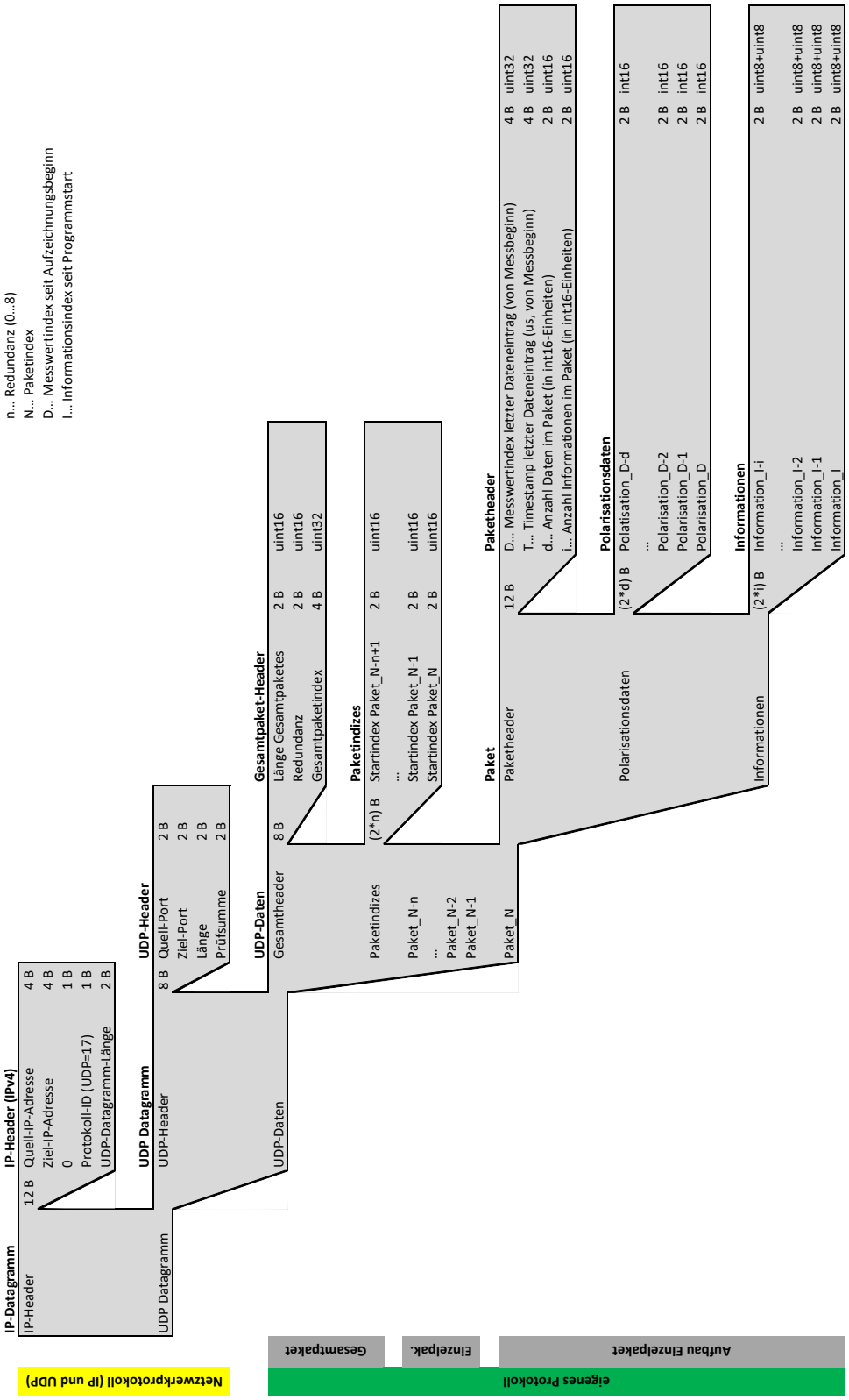


Tabelle 6.6: Aufbau des verwendeten Protokolls (Anh. B.3.3)

7 Messprogramm für das STEMlab-Board

Aus Kap. 6.3 folgt, dass für eine kontinuierliche Aufnahme mit der gewünschten Geschwindigkeit ein eigens dafür angepasstes Programm am STEMlab-Board benötigt wird. Zunächst soll die Nutzung des FPGA auf die Bereitstellung der Messdaten beschränkt sein (Umsetzung von RedPitaya verfügbar) und die Verarbeitung dieser Daten am ARM-Prozessor im Messprogramm erfolgen. Sollte die CPU nicht ausreichend sein, können Teile der Auswertung am FPGA vorgenommen werden. Im Laufe der Entwicklung zeigt sich, dass der Prozessor alleine für das Ausführen dieser Anwendung reicht.

7.1 Python oder C

7.1.1 Python

Am STEMlab-Board können Python-Programme ausgeführt werden. Vorteile gegenüber C liegen darin, dass bei Änderungen kein Kompilieren nötig ist und Variablendeklarationen/Speicherverwaltung nicht im selben Umfang bedacht werden müssen. Die Programmierung kann dabei zum Beispiel über Jupyter Notebook im Browser erfolgen, wodurch ein sofortiges Ausführen des Codes möglich ist und Ergebnisse direkt ausgegeben werden können. [19]

Um die RedPitaya-API-Funktionen in Python nutzen zu können, muss das „PyRedPitaya“-Package installiert werden. Das Package beinhaltet ein C-Library, welches den Zugriff auf die FPGA-Register ermöglicht. [28]

7.1.2 C

Das Kopieren des RedPitaya-Repository auf das STEMlab-Board (siehe Kap. 9.1.1.5), ermöglicht die Verwendung zahlreicher Funktionen der RedPitaya-Programmierschnittstelle (API) für den Zugriff auf die Hardware. Die inkludierten Beispielprogramme und Funktionen können als Grundlage für eigene Entwicklungen verwendet werden. [19]

Da das Programm vor dem Ausführen kompiliert werden muss, ergeben sich zum Testen einer Änderung einige zusätzliche Zwischenschritte im Vergleich zu Python.

7.1.3 Performancevergleich Python und C am STEMlab-Board

Um den Geschwindigkeitsunterschied bei Nutzung unterschiedlicher Programmiersprachen besser abschätzen zu können, wird ein einfaches Testprogramm in Python und in C erstellt und am STEMlab-Board ausgeführt. Dabei soll die Partialsumme der harmonischen Reihe mit 10^6 Summanden berechnet werden, Tab. 7.1 zeigt die berechneten Ergebnisse und die dafür benötigte Zeit.

Programmiersprache (Datentyp)	Ergebnis $sum = \sum_{n=1}^{10^6} \frac{1}{n}$	Ermittlungsdauer
Python	$sum = 14{,}392726$	$\approx 3.375\ ms$
C (double: 64 bit)	$sum = 14{,}392726$	$\approx 54\ ms$
C (float: 32 bit)	$sum = 14.357357$	$\approx 39\ ms$

Tabelle 7.1: Performancevergleich Python und C am STEMlab-Board (Anh. B.1.9)

Die Summen sind bei Verwendung von Double-Precision-Gleitkommavariablen in C zumindest auf 8 signifikante Stellen gleich wie bei Python. Das Python-Skript benötigt für diesen Programmteil über 60 mal so lange wie das kompilierte C-Programm (die Messung umfasst ausschließlich die Schleife zur Berechnung, Programmstart und Ergebnisausgabe sind nicht inkludiert). Das Messprogramm wird daher in C umgesetzt.

Die Verwendung des kleineren Variablentyps „float“ spart nochmals knapp 30% Rechenzeit gegenüber „double“, dabei tritt jedoch durch die Aufsummierung kleiner Fehler eine erkennbare Abweichung des Ergebnisses auf (hier etwa 0,25%).

7.2 Einlesen der Messwerte aus dem Puffer

Die analogen Signale des SCPEM-Stroms sowie des verstärkten Photodiodensignals liegen an den beiden Fast-Analog-Eingängen des Messboards an und werden vom ADC (Analog-Digital-Converter) digitalisiert. Im FPGA werden diese Werte abhängig von den Einstellungen in den Sample-Puffer geschrieben, woraus sie bei Bedarf ausgelesen werden können.

7.2.1 Eigenschaften der Fast-Analog-Eingänge

Für das Messprogramm sind einige Eigenschaften der beiden Signaleingänge des STEMlab-Boards von Bedeutung:

- Messbereich: ±1 V (LV) oder ±20 V (HV) [19]
  Der Messbereich kann durch Jumper auf Low Voltage oder High Voltage festgelegt werden, die Auswahl des Spannungsbereiches ist in Kap. 9.1.1.7 beschrieben.

- DAC-Auflösung: 14 bit [19]
  Der Messbereich wird mit 14 bit in $2^{14} = 16.384$ Schritten aufgelöst.
  LV: Messbereich 2 V, Auflösung 0,122 mV
  HV: Messbereich 40 V, Auflösung 2.441 mV

- Bandbreite: 50 MHz: [19]
  Das Intensitätssignal besitzt die höchste Eingangsfrequenz (doppelte Frequenz des SCPEM), diese liegt im aktuellen Anwendungsfall bei etwa 160 kHz. Die Bandbreite ist selbst bei schneller schwingendem SCPEM weiterhin mit großer Reserve

ausreichend, solange die Frequenz nicht in den zweistelligen MHz-Bereich gesteigert wird (Abb. 7.1).

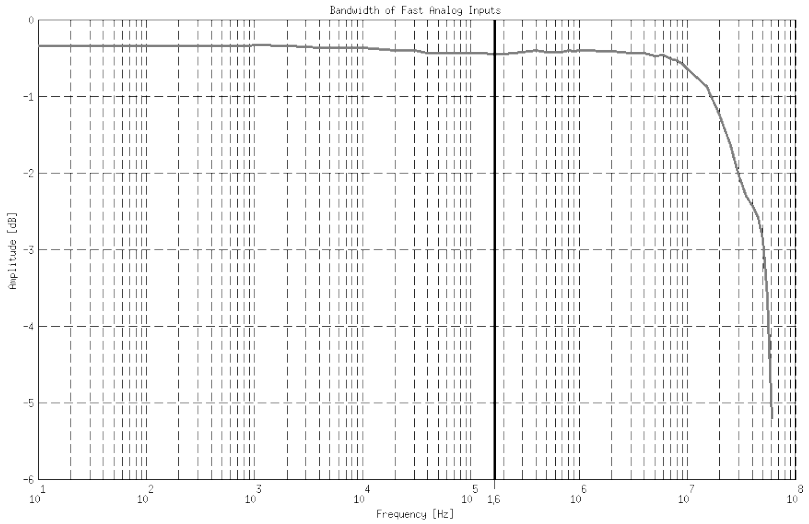


Abbildung 7.1: Bandbreite der Fast-Analog-Eingänge [19]

- Sample Rate: 125 MS/s [19]
  Die maximale Samplerate beträgt je Eingang 125 MS/s, diese kann durch Dezimations-Einstellungen direkt im FPGA verringert werden.

- Dezimation: 1, 8, 64, 1.024, 8.192 oder 65.536 [19]
  Die Dezimation gibt an, wie viele der vom ADC ermittelten Samples vom FPGA beim Schreiben in den Puffer übersprungen werden.
  Während eine höhere zeitliche Auflösung bei niedriger Dezimation das Signal genauer darstellt, wird die zur Verfügung stehende Zeit der Auswertung durch den 16.364 Samples fassenden Puffer beschränkt.
  Die gewählte Dezimation ist 64, das entspricht einer Sample-Rate von $\frac{125}{64}\ MS/s = 1.953.125\ S/s$. Es werden etwa 12 Samples pro Dioden-Signal-Periode (bei 80 kHz-SCPEM) ermittelt, wodurch die Rekonstruktion der Extremwerte möglich ist. Der Puffer reicht für 8,3 ms, das entspricht damit der maximalen Zeit, die für einen vollständigen Durchlauf (Einlesen des SCPEM-Signals, Nulldurchgangssuche, Einlesen des Dioden-Signals, Extremwertsuche, Berechnung der Polarisation, Senden bzw. Speichern der Ergebnisse, Vorbereiten des nächsten Durchlaufes) zur Verfügung steht. Dabei müssen in jedem Durchlauf etwa 670 SCPEM-Perioden abgearbeitet werden können. Tab. 7.2 stellt diese Daten für alle Dezimationseinstellungen dar.

- Rauschen des Eingangs (ohne angeschlossener Messgröße):
  Die Standardabweichung des Rauschens der Eingänge beträgt laut Dokumentation 3,5 ADC-Schritte bei 125 MS/s (berechnet durch FFT aus 16.384 Samples, Abb. 7.2). [19]

Es kann eine Mittelwertbildung der übersprungenen Samples am FPGA aktiviert werden, wodurch das Rauschen des Signals reduziert wird. Die exakte Definition dieser Mittelwertbildung/Filterung ist nicht dokumentiert.
Aus jeweils 163.840 Samples (10 gefüllte Puffer) des Einganges 1 wird die Standardabweichung mit der MATLAB-Funktion std() berechnet (Tab. 7.3). Am Eingang ist dabei nichts angeschlossen. Das so ermittelte Rauschen liegt im Bereich des in der Dokumentation angegebenen Wertes und es zeigt sich wie erwartet eine Verringerung bei höherer Dezimation und aktivierter Mittelwertbildung, während die Standardabweichung bei deaktivierter Mittelwertbildung unabhängig von der Dezimation ist.
Durch Dezimation bei aktivierter Mittelwertbildung wird die Standardabweichung der vorliegenden Samples auf knapp unter einem ADC-Schritt verringert. Diese Information ist für den Extremwert-Algorithmus von Bedeutung.

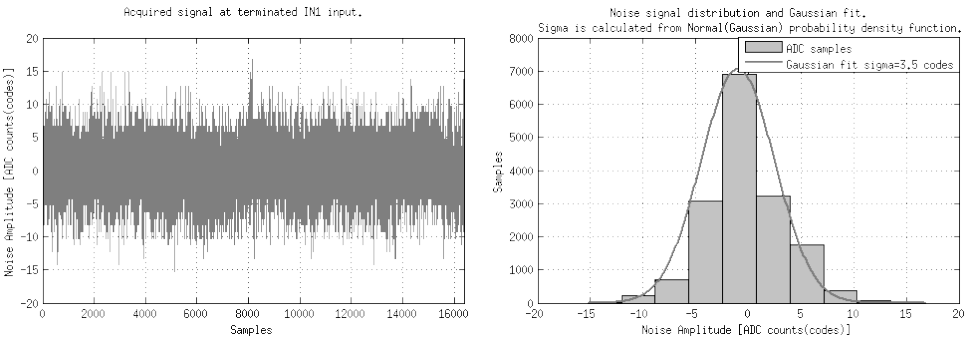


Abbildung 7.2: Rauschen des Eingangssignals [19]

Dezimation	Sample-Rate kS/s	Puffer-Speicher-kapazität μs	vollst. Perioden im Puffer SCPEM 80 kHz	Samples/Periode bei SCPEM 80 kHz SCPEM-Signal	SCPEM 80 kHz Diodensignal (160 kHz)	SCPEM 120 kHz Diodensignal (240 kHz)
1	125.000.000	131	10	1.563	781	521
8	15.625.000	1.049	83	195	98	65
64	1.953.125	8.389	671	24,41	12,21	8,14
1.024	122.070	134.218	10.737	1,53	0,76	0,51
8.192	15.259	1.073.742	85.899	0,191	0,095	0,064
65.536	1.907	8.589.935	687.194	0,024	0,012	0,008

Tabelle 7.2: Dezimation und Messwerte je Periode

Dezimation	Standardabweichung in ADC-Schritten	
	Mittelwertbildung Aus	Mittelwertbildung Ein
1	3,824	3,822
8	3,829	1,657
64	3,811	0,844
1.024	3,802	0,537
8.192	3,827	0,331
65.536	3,827	0,241

Tabelle 7.3: Ermitteltes Signalrauschen mit und ohne Mittelwertbildung (Anh. B.1.10)

7.2.2 Samplepuffer

Bevor die Daten eingelesen werden können, muss der Puffer und seine Arbeitsweise genauer betrachtet werden, um die Samples in der richtigen Reihenfolge und ohne Datenverlust ausreichend schnell zu erhalten.

7.2.2.1 Funktion des Ringpuffers

Für jeden Eingang ist ein Ringpuffer mit 16.384 Elementen vorgesehen. Der Index des zuletzt beschriebenen Elements (des aktuellsten Samples) kann als Write-Pointer abgefragt werden. Der FPGA schreibt die Samples der beiden Eingänge der Reihe nach in die Puffer, wobei der Schreibindex für beide Puffer gleich ist (die Dezimation kann ebenso nicht unterschiedlich vorgegeben werden). Am Ende angelangt, wird wieder vom Pufferbeginn aufgefüllt und alte Samples überschrieben.

7.2.2.2 Vorgangsweise beim kontinuierlichen Lesen der Messwerte

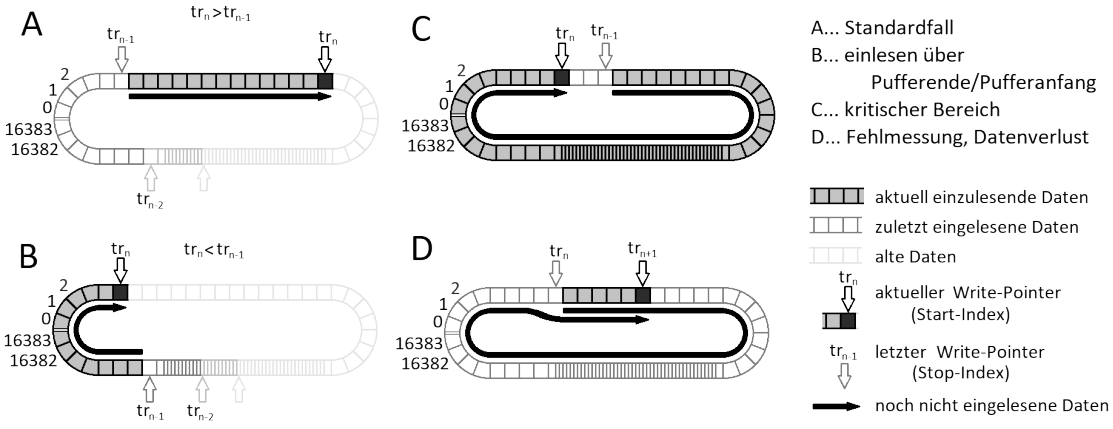


Abbildung 7.3: Kontinuierliches Lesen aus dem Ringpuffer

Um alle Samples der Reihe nach zu erfassen, wird folgende Vorgehensweise angewendet: Vom letzten Durchlauf ist der Index des letzten bereits eingelesenen, verarbeiteten Elements bekannt (Start-Index), mit dem aktuellen Write-Pointer (Stopp-Index) kann bestimmt werden, welche Elemente neu hinzugekommen sind und verarbeitet werden müssen (Start-Index bis Stopp-Index, Abb. 7.3 A).

Wenn der aktuelle Write-Pointer (Stopp-Index) kleiner als der Start-Index ist, bedeutet das, dass der Ringpuffer in der Zwischenzeit gefüllt worden ist und wieder beim Index 0 zu schreiben begonnen wurde. In diesem Fall muss vom Start-Index bis zum Index 16.383 und vom Index 0 bis zum Stopp-Index eingelesen werden (Abb. 7.3 B, C).

Im Fall C wurde vom Zeitpunkt der letzten Auswertung bis zum neuerlichen Einlesezeitpunkt bereits nahezu die gesamte Pufferzeit aufgebraucht (Tab. 7.2: 8,389 ms), verlustfreies Einlesen ist jedoch noch möglich. Zur Erhöhung der Dauer eines Durchlaufes kann es durch außergewöhnliche Vorkommnisse wie das Senden großer Datenmengen, Verzögerungen beim Schreiben in eine Datei oder Verarbeitung von neuen Einstellungen in diesem Durchlauf kommen. Problematisch daran ist, dass im darauffolgenden Durchlauf sehr viele Daten verarbeitet werden müssen und daher wieder viel Zeit vergeht, bis der übernächste Durchlauf gestartet werden kann. Die Verarbeitung der Daten erfolgt dabei aber schneller, als diese in den Puffer geschrieben werden, daher sollte diese Durchlaufdauer nach einigen Durchläufen und ohne weitere Zwischenfälle wieder kürzer werden. Beachtet werden muss, dass während des Abarbeitens der vielen Samples die ersten Daten im auszuwertenden Bereich bereits wieder überschrieben werden können und diese daher nur für begrenzte Zeit und nicht zwingend bis zum Abschluss des Durchlaufs verfügbar sind.

Sollte die Verzögerung so groß sein, dass ein Durchlauf die max. zulässige Dauer überschreitet, tritt Abb. 7.3 Fall D ein. Es wurden mehr als 16.384 Samples in den Puffer geschrieben. Dabei tritt eine Fehlinterpretation der Pointer ein, die aktuell einzulesenden Daten decken nicht alle neuen Daten ab. Es gehen in jedem Fall Daten verloren, die Aufnahme wird jedoch fortgesetzt. Aus den Pointer-Positionen selbst ist dieser Fall nicht von Fall A oder B zu unterscheiden, durch die verstrichene Zeit oder durch Vergleich einiger Samples im Puffer auf Veränderung seit dem letzten Durchlauf kann der Datenverlust jedoch erkannt werden.

7.2.2.3 Möglichkeit von Dezimation 8 statt 64

Nach vielen Optimierungen im gesamten Datenverarbeitungsvorgang (Einlesen, Berechnen, Ausgeben) beträgt die Dauer je Durchlauf in der aktuellen Version des Messprogramms etwa 0,1 bis 0,2 ms. Der Puffer wird in dieser Zeit nur im einstelligem Prozentbereich gefüllt und es stehen danach knapp 8 ms für außergewöhnliche Ereignisse zur Verfügung. Dadurch tritt der Fehlerfall D praktisch nicht auf.

Bei Dezimation 8 steht der Puffer rund 1 ms zur Verfügung, die Auswertung wäre bei der aktuellen Geschwindigkeit somit zumindest theoretisch möglich. Mit der höheren Samplerate könnten weitaus höhere SCPEM-Frequenzen ausgewertet werden. Für den verwendeten 80 kHz-SCPEM (und noch etwas höhere Frequenzen) ist die Samplerate bei Dezimation 64 ausreichend, daher wird diese Möglichkeit nicht weiter untersucht.

7.2.3 Umsetzung des Einlesens der Messwerte

Von jeweils 64 Samples wird vom FPGA der Mittelwert gebildet und in den Puffer geschrieben. Die Vorgangsweise für kontinuierliches Einlesen aller Daten aus dem Ringpuffer ist in Kap. 7.2.2.2 definiert, in diesem Kapitel werden verschiedene Möglichkeiten des eigentlichen Lesens aus dem Puffer selbst verglichen.

7.2.3.1 Zugriff über SCPI-Server und RedPitaya-API

Es besteht die Möglichkeit, die Samples mit der RedPitaya-API über einen SCPI-Server direkt an den Computer zu übermitteln, siehe hierzu Kap. 6.3. Für die kontinuierliche Aufzeichnung beider Signale ist diese Variante um Größenordnungen zu langsam.

7.2.3.2 Zugriff am STEMlab-Board über RedPitaya-API

Einige relevante Funktionen der RedPitaya-API zum Einlesen der Daten sind: [19]

- `int rp_AcqSetDecimation(rp_acq_decimation_t decimation);`

  Legt die Dezimation, mit der aufgenommen wird, fest. Diese ist für beide Eingänge gleich.

- `int rp_AcqStart();`

  Startet die Aufnahme, Signale der Eingänge werden in die Register geschrieben.

- `int rp_AcqGetWritePointer(uint32_t* pos);`

  Gibt die Position des aktuellen Write-Pointer (zuletzt beschriebenen Index im Puffer) zurück. Dieser ist für beide Eingänge gleich.

- `int rp_AcqGetDataV(rp_channel_t channel, uint32_t pos, uint32_t* size, float* buffer);`

  Gibt die Samples vom Eingang “channel” (Eingang 1 oder Eingang 2) im Array „buffer“ als float in Volt zurück, beginnend beim Index pos und Anzahl size.

- `int rp_AcqGetDataRaw(rp_channel_t channel, uint32_t pos, uint32_t* size, int16_t* buffer);`

  Gibt die Samples vom Eingang “channel” (Eingang 1 oder Eingang 2) im Array „buffer“ als Integer zurück, beginnend beim Index pos mit Sample-Anzahl size.

- `int rp_AcqGetDataPosRaw(rp_channel_t channel, uint32_t start_pos, uint32_t end_pos, int16_t* buffer, uint32_t *buffer_size);`

  Gibt die Samples vom Eingang “channel” (Eingang 1 oder Eingang 2) im Array „buffer“ als Integer zurück, vom Index start_pos und bis zum Index end_pos.

- `int rp_AcqGetLatestDataRaw(rp_channel_t channel, uint32_t* size, int16_t* buffer);`

  Gibt die aktuellsten (zuletzt aufgenommenen) Samples zurück, also vom Index (Write-Pointer - size) bis Write-Pointer. Der Write-Pointer wird dabei von der Funktion selbstständig ermittelt.

Diese und viele weitere API-Funktionen sind inkl. kurzer Beschreibungen im RedPitaya-Repository in der Datei rp.h (zu finden am STEMlab-Board: `/root/RedPitaya/api/include/redpitaya/rp.h`) dokumentiert.

Für die Vorgangsweise des Einlesens nach Kap. 7.2.2.2 kann nach Ermittlung des Write-Pointers (Stopp-Index) die Funktion rp_AcqGetDataPosRaw() mit Start-Index und Stopp-Index genutzt werden.

Tabelle 7.4 zeigt, ob es bei Verwendung der Funktion rp_AcqGetDataPosRaw() schneller ist, alle aufeinanderfolgenden Samples einzulesen (inkl. nicht benötigter) oder nur die benötigte Samples mit mehrmaligem Funktionsaufruf für je ein Sample. Die einzelnen Lesezugriffe dauern dabei rund doppelt so lange je Sample. Werden weniger als die Hälfte der Samples benötigt, ist das einzelne Auslesen schneller. Für die Nullpunktsuche am SCPEM-Signal wird nur rund jedes 10. Sample benötigt (min. 2 je Periode), für die Ermittlung der Extremwerte etwa ein Viertel bis zur Hälfte der Samples. Im Messprogramm ist das einzelne Lesen der benötigten Samples demnach schneller.

Ermittelte Daten	Dauer
100 Samples, sequentiell	17,892 μs (0,18 μs/Samp)
100 Samples, einzeln	32,108 μs (0,32 μs/Samp)
100 Sample, jedes 2. einzeln (50 aus 100)	16,152 μs (0,32 μs/Samp)
100 Samples, jedes 10. einzeln (10 aus 100)	3,279 μs (0,33 μs/Samp)

Tabelle 7.4: Pufferzugriffszeiten (Mittelwert aus 1.000 Aufrufen) (Anh. B.1.11.1)

7.2.3.3 Zugriff direkt

Aus der STEMlab-Dokumentation und dem Sourcecode der RedPitaya-API kann ermittelt werden, wie der Zugriff auf das FPGA-Register erfolgt. Damit wird das Einlesen ohne Verwendung der API möglich, wodurch einige für diese Anwendung unnötige Funktionsschritte wegfallen. Ziel ist es, einzelne Samples noch schneller einlesen zu können.

Speicheradresse RedPitaya-User Manual Release 1.0 zeigt unter Developers Guide - Software - FPGA die Adressbelegung im FPGA-Speicher. Dieser besteht aus acht Teilen zu je 4 MB, die für unterschiedliche Daten reserviert sind (Tab. 7.5).

Die benötigten Daten befinden sich im Teil CS[1] (Oszilloscope). Zur Startadresse 0x40100000 kommt ein Offset abhängig vom gewünschtem Dateneintrag (Tab. 7.6). Für die korrekte Verarbeitung der Daten ist zu beachten, dass diese „little-endian“-organisiert sind und jeder Eintrag 32 bit belegt. [29]

Part	Start	End	Module Name
CS[0]	0x40000000	0x400FFFFF	Housekeeping
CS[1]	0x40100000	0x401FFFFF	Oscilloscope
CS[2]	0x40200000	0x402FFFFF	Arbitrary signal generator (ASG)
CS[3]	0x40300000	0x403FFFFF	PID controller
CS[4]	0x40400000	0x404FFFFF	Analog mixed signals (AMS)
CS[5]	0x40500000	0x405FFFFF	Daisy chain
CS[6]	0x40600000	0x406FFFFF	FREE
CS[7]	0x40700000	0x407FFFFF	Power test

Tabelle 7.5: Speicherbereiche im FPGA-Speicher [29]

Offset	Description	Bits	R/W
...			
0x18	Write pointer - current		
	Reserved	31:14	R
	Current write pointer	13:0	R
...			
0x10000 to 0x1FFFC	Memory data (16k samples)		
	Reserved	31:16	R
	Captured data for ch A	15:0	R
0x20000 to 0x2FFFC	Memory data (16k samples)		
	Reserved	31:16	R
	Captured data for ch B	15:0	R

Tabelle 7.6: Relevante Daten im Bereich CS[1] (Oscilloscope) [29]

Am Offset 0x18 befindet sich der aktuelle Write-Pointer (Index des zuletzt geschriebenen Samples). Dieser gilt für beide Channel und ist zwischen 0 und 16.383.

Die Messsamples von Channel A und Channel B befinden sich bei Offset 0x10000 bzw. 0x20000. Beispielsweise ist die Adresse für Sample i aus Channel B (0x40100000 + 0x20000 + i). Die erhaltenen 32 Bits je Sample müssen dann umgerechnet werden, sodass sie eine zum Spannungswert am Eingang proportionale Größe inklusive Vorzeichen ergeben. Ein weiterer Rechenschritt, um daraus die Spannung in Volt zu erhalten, ist in diesem Anwendungsfall nicht nötig.

Mapping Das Mapping der Speicherbereiche (Alg. 7.1) ist an die Vorlagen aus der RedPitaya-API oscilloscope.c -> osc_Init() und common.c -> cmn_Map angelehnt. Die Ermittlung des Write-Pointers kann durch direkten Zugriff auf die zugewiesene Adresse erfolgen. Er befindet sich am Offset 0x14 in Byte, somit in 32-Bit-Einheiten am Index 6, und kann daher mit pointer_osc_reg[6] aufgerufen werden (Alg. 7.2).

Die Pointer ch1 und ch2 können nach dem Mapping wie Arrays verwendet werden.

Algorithmus 7.1 Definition der Speicherbereiche ch1 und ch2 (Anh. C.1.1)

```
// ...

static int fd;
fd = open("/dev/uio/api", O_RDWR | O_SYNC);

uint32_t * pointer_osc_reg = (uint32_t *)mmap(NULL, 0x30000,
    PROT_READ, MAP_SHARED, fd, (0x00100000 >> 20) * sysconf(
    _SC_PAGESIZE));

static uint32_t * ch1 = NULL;
static uint32_t * ch2 = NULL;

// OSC_CHA_OFFSET=0x4000=16.384 in uint32 (0x10000 in char):
ch1 = (uint32_t*)((uint32_t*)pointer_osc_reg + 0x4000);
// OSC_CHB_OFFSET=0x8000=32.763 in uint32 (0x20000 in char):
ch2 = (uint32_t*)((uint32_t*)pointer_osc_reg + 0x8000);

// ...

ch1=NULL;
ch2=NULL;
close(fd);
```

Algorithmus 7.2 Makro zur Berechnung der Messwerte mit Offset (Anh. C.1.1)

```
#define GET_POINTER pointer_osc_reg[6]

#define GET_SAMP_OFFSET1(index,sample) {                \
     index_real=(index)%0x4000; sample=ch1[index_real]; \
     if (sample&0x2000){sample=-((sample^0x3FFF)+1);}   \
     sample-=CH1_offset;                                \
     }

#define GET_SAMP_OFFSET2(index,sample) {                \
     index_real=(index)%0x4000; sample=ch2[index_real]; \
     if (sample&0x2000){sample=-((sample^0x3FFF)+1);}   \
     sample-=CH2_offset;                                \
     }
```

Umrechnung Die Umrechnung der Werte der beiden Kanäle sowie die Ermittlung des Write-Pointers sind als Makros umgesetzt (Alg. 7.2). Zuerst wird mit einer Modulo-Funktion sichergestellt, dass der Index zwischen 0 und 16.383 ist. Ein größerer Index würde zum Lesen außerhalb des zugewiesenen Speicherbereiches führen. Das Sample wird aus dem Array gelesen, welches auf die Adresse der Puffer im FPGA-Register zeigt.

Das Bit 15 zeigt das Vorzeichen an. Ist dieses 1 ($sample 0x2000 == 1$), muss mittels einer XOR-Operation $sample \hat{} 0x3FFF$ umgerechnet werden. Die Vorlage dafür findet sich in common.c in cmn_CalibCnts. Dabei ist (1 << (field_len - 1))=0x2000 und ((1 << field_len) - 1)=0x3FFF.

Verwendung im Programm Die Ermittlung der gewünschten Werte erfolgt nach Definition der Makros (Alg. 7.2) folgenderweise:

```
uint16_t index1, index2, index_real, pointer;
int16_t CH1_offset, CH2_offset, sample1, sample2;
// ...
pointer=GET_POINTER;
GET_SAMP_OFFSET1(index1, sample1);
GET_SAMP_OFFSET2(index2, sample2);
```

7.2.3.4 Performancevergleich Zugriff API - Zugriff Direkt

Die verschiedenen Möglichkeiten des Lesens vom Puffer werden anhand ihrer benötigten Ausführungsdauer verglichen. Die folgenden Werte werden jeweils als Mittelwert von je 1.000 Messungen erfasst.

Bei der Ermittlung des Write-Pointers ist der Direktzugriff auf das FPGA-Register etwa 10% bis 15% schneller als der API-Aufruf, Tab. 7.7.

Ermittelte Daten	API	Direkt
Write-Pointer	0,162 μs	0,141 μs

Tabelle 7.7: Zugriffszeiten Write-Pointer (Anh. B.1.11.2)

Tabelle 7.8 zeigt die Dauer des Einlesevorganges für unterschiedliche Szenarien. Die Verwendung der Samplewerte in Volt bringt keinen Informationsgewinn für den Algorithmus, die eingesparte Umrechnung und kleinere Variablengröße (float: 32 bit, int16: 16 bit) bringt bereits eine erhebliche Zeitersparnis von 35-50% mit sich, es wird daher mit den Integer-Werten (Raw-Daten) gerechnet. Beim Einlesen einer größeren Menge von direkt aufeinanderfolgenden Samples ist der API-Aufruf etwas schneller als die eigene Umsetzung mit jeweils einzeln eingelesenen Samples.

Ermittelte Daten	API: Voltage	API: Raw	Direkt
10.000 S.	3.739 μs (0,37 μs/S)	1.786 μs (0,18 μs/S)	2.236 μs (0,22 μs/S)
100 Samp.	37,2 μs (0,37 μs/S)	17,9 μs (0,18 μs/S)	22,5 μs (0,23 μs/S)
3 Samples	1,285 μs (0,43 μs/S)	0,764 μs (0,25 μs/S)	0,624 μs (0,21 μs/S)
1 Sample	0,625 μs	0,385 μs	0,231 μs

Tabelle 7.8: Zugriffszeiten sequentielle Messsamples (Anh. B.1.11.3)

Für die vom Messprogramm benötigten einzelnen Samples kann die Dauer im Vergleich zum Einlesen in Volt um 50-60% und im Vergleich zum Raw-API-Aufruf zusätzlich um 20-30% verringert werden.

Der Vergleich zeigt, dass offenbar das Ermitteln eines Messwertsamples länger dauert als das Ermitteln des Write-Pointers, obwohl diese im selben Speicher an unterschiedlichen Adressen zu finden sind. Das liegt an der Umrechnung, die für die Samples nötig ist, während der Pointer bereits richtig formatiert vorliegt (Alg. 7.2).

7.2.4 Alle Samples kontinuierlich einlesen und übertragen

7.2.4.1 Theoretische Betrachtung

Da die Dauer für das Einlesen und das Senden der Daten bekannt sind, kann geprüft werden, ob die dauerhafte, kontinuierliche Übermittlung aller Daten der Eingänge an den Computer möglich ist (bei Dezimation 64).

Die benötigte Zeit je Sample beträgt im Idealfall rund 180 ns (siehe Tabelle 7.8 bei großen Gruppen von Samples), die Dauer für das Senden bei 2 Byte je Sample mindestens 2*9 ns (siehe Kapitel 6.4.2.3), wodurch sich etwa 0,198 μs je Sample ergeben. Nicht berücksichtigt ist die nötige Konvertierung beim Vorbereiten zum Senden.

Bei zwei Eingängen mit je 1,953 $MSamp/s$ ($\frac{125}{64}$ $MSamp/s$) müssen 3,906 Millionen Samples je Sekunde eingelesen und gesendet werden. Es stehen somit pro Sample im Schnitt 0,256 μs zur Verfügung (0,198 μs werden mind. benötigt), weshalb die Umsetzung knapp möglich scheint.

7.2.4.2 Umsetzung (Anh. B.2.7)

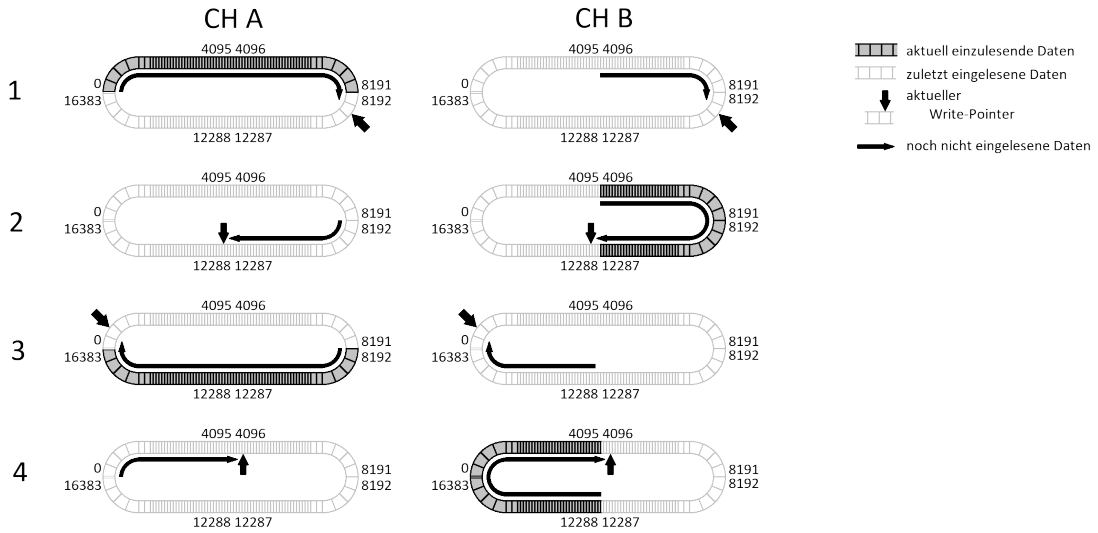


Abbildung 7.4: 4 Schritte beim kontinuierlichen Einlesen aller Samples

Die Übertragung wird mit zwei einfachen Programmen am STEMlab-Board und am Computer getestet. Eines liest die Samples ein und schickt sie per UDP, das andere empfängt diese Pakete und speichert die Daten ab.

Es wird wie in Abb. 7.4 dargestellt eingelesen und gesendet, um beide Puffer zu übertragen und dabei das Überschreiben während des Lesevorganges möglichst zu verhindern. Dabei wird jeweils gewartet, bis der Write-Pointer ein Viertel des Puffers verlässt, dann wird eine Hälfte gelesen und gesendet. Anschließend wird beim anderen Puffer das gleiche durchgeführt, jedoch um ein Viertel des Puffers versetzt. Nach vier Schritten oder ca. 8 ms sind beide Puffer vollständig gelesen und gesendet und der Vorgang beginnt von vorne.

Im Test funktioniert die Übertragung und Aufzeichnung, es kommt jedoch gelegentlich zu Datenverlusten bei der Übermittlung (gelegentlich bedeutet in der Größenordnung von einem Paket aus einigen Tausend Paketen). Dadurch sind fehlerfreie Aufnahmen im Sekundenbereich möglich, die Anzahl der übermittelten und gespeicherten Samples liegt dann im zweistelligen Millionenbereich, pro Sekunde fallen knapp 8 MB an Daten an.

7.3 Ermittlung der Extremwerte aus dem Intensitäts-Signal

Zur Ermittlung der Polarisation muss die Intensität der Strahlung zu bestimmten Zeitpunkten ermittelt werden. Diese Zeitpunkte zeichnen sich dadurch aus, dass der SCPEM dann keine Retardation oder die maximale Retardation hat, und somit die Intensität minimal oder maximal ist.

Nach Ermittlung des Nulldurchgangs des SCPEM-Signals befinden sich diese Extremwerte eine bestimmte Zeit (oder eine bestimmte Anzahl von Samples) davon versetzt. Es treten je SCPEM-Periode zwei Maxima und zwei Minima auf, wobei jeweils eine Viertelperiode dazwischenliegt. Die Ermittlung soll durch kleine Fehler beim Zeitpunkt und durch Störungen im Signal möglichst wenig beeinträchtigt werden.

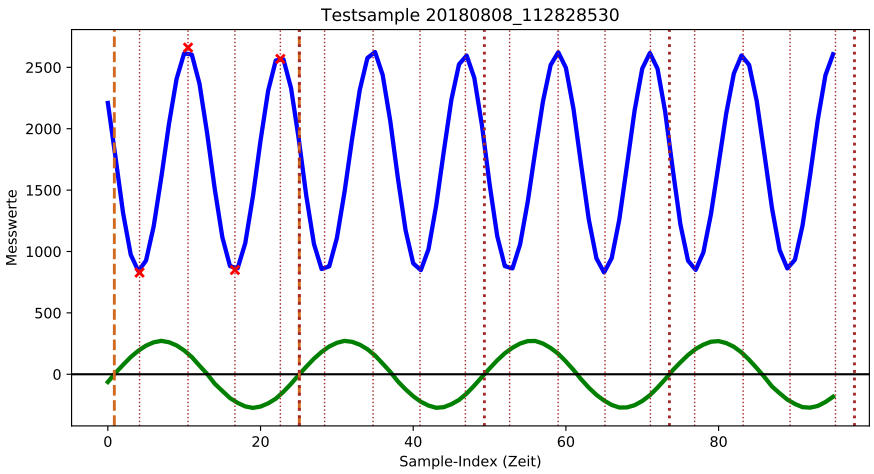


Abbildung 7.5: Testsample mit SCPEM- und Intensitätssignal (Anh. B.1.12)

Abb. 7.5 zeigt eine Testaufnahme der beiden Signale (SCPEM unten, Photodiode oben). Darin sind die ermittelten Nulldurchgänge sowie die Positionen der Extremwerte eingezeichnet, wie sie vom Messprogramm ermittelt werden. Häufig ist das Photodioden-Signal flacher und die Extremwerte weniger stark ausgeprägt als in diesem Beispiel.

7.3.1 Schwierigkeiten und Herausforderungen

Folgende Randbedingungen müssen betrachtet werden:

- Extremwerte können abhängig von der Polarisation jeweils sowohl Minimal- als auch Maximalwerte sein
- Sehr flache Kurven weisen keine ausgeprägten Extremwerte auf (Position des Extremwertes schwer zu finden)
- Beeinflussung durch Signalstörungen und Digitalisierung sind zu minimieren
- Möglichst wenige Datenpunkte für schnelle Auswertung verwenden (Einlesedauer)
- Zeitliche Position der Extremwerte kann leicht abweichen und soll korrigiert werden

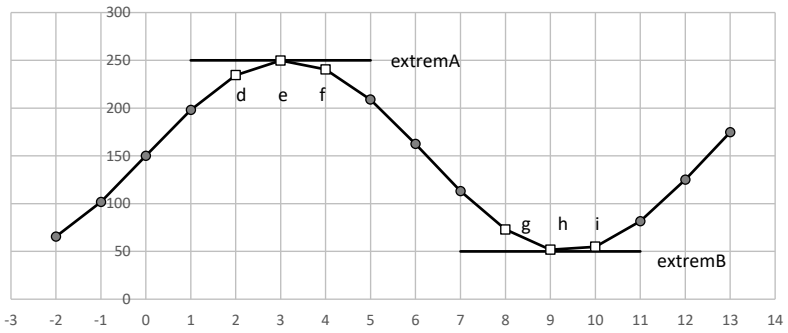


Abbildung 7.6: Einfache Ermittlung der Extremwerte auf störungsfreiem Signal (Anh. B.2.8.1)

In Abb. 7.6 sind die Extremwerte weit auseinander und dadurch sind die Störungen im Vergleich zum Signal klein (ähnlich den gemessenen Werten in Abb. 7.5). Die Bestimmung der Werte extremA und extremB ist einfach möglich, für ein brauchbares Ergebnis reicht es schon, die Werte e und h zu verwenden. Bessere Ergebnisse bei Messwerten mit sehr geringer Störung erzielen jedoch in Kap. 7.3.2 beschriebene Methoden.

In Abb. 7.7 sind die gemessenen Werte kleiner und die Störungen größer. Die Differenz zwischen extremA und extremB ist bei Messungen beliebig, oft sind die Extremwerte etwa gleich groß. Der Algorithmus soll hier keine Fehlinterpretation liefern. Durch Verwenden von drei Punkten anstatt eines Punkts kann die Standardabweichung der Ergebnisse reduziert werden. Die Parabel-Methode würde hier zu unbrauchbaren Ergebnissen führen.

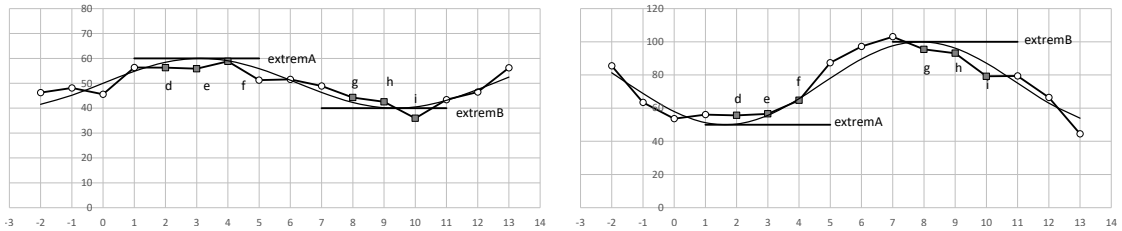


Abbildung 7.7: Schwierigere Ermittlung der Extremwerte bei Signalstörungen (Anh. B.2.8.1)

Zusammengefasst muss der Algorithmus für Minima und Maxima gleichermaßen funktionieren und soll auch bei Positions-Verschiebungen und Signalstörungen das ungestörte Extremum möglichst genau rekonstruieren.

7.3.2 Unterschiedliche Methoden der Berechnung

7.3.2.1 Nächstgelegenes Sample als Extremwert

Die einfachste und schnellste Möglichkeit ist das zum errechnetem Zeitpunkt des Extremwertes nächstgelegene Sample zu verwenden. Es wird nur ein Sample eingelesen und in jedem Fall ein verwendbares Ergebnis geliefert.

Einfluss Position Die Position auf der Signalkurve hat großen Einfluss auf das Ergebnis, da keine Korrektur erfolgt, wenn nicht das Sample am Extremwert verwendet wird. Nachteilig ist, dass durch die Abweichung der Position Minima im Schnitt leicht überschätzt, Maxima leicht unterschätzt werden.

Einfluss Rauschen Das Rauschen wirkt direkt ins Messergebnis.

Zusammenfassung

\+ Sehr einfach und schnell (nur ein Sample nötig)

\- Abweichung stark von Position abhängig

\- Neigt zum Unterschätzen von Maxima und Überschätzen von Minima

\- Keine Reduzierung von Rauschen

7.3.2.2 Mittelwert über nächstgelegene Samples

Gegenüber der Verwendung nur eines Samples wird beim Mittelwert über drei Samples der Fehler durch Rauschen reduziert. Eine Mittelung führt jedoch zum stärkeren Unterschätzen von Maximalwerten sowie zum stärkeren Überschätzen von Minimalwerten, da sich jeweils zwei Punkte weiter weg vom Extremwert befinden.

Die Wahl von drei Samples für die Mittelwertberechnung anstatt von zwei oder vier liegt daran, dass sowohl für die Parabelberechnung (Kap. 7.3.2.3) als auch für die Extremwertsuche (Kap. 7.3.3.2) drei Samples benötigt werden.

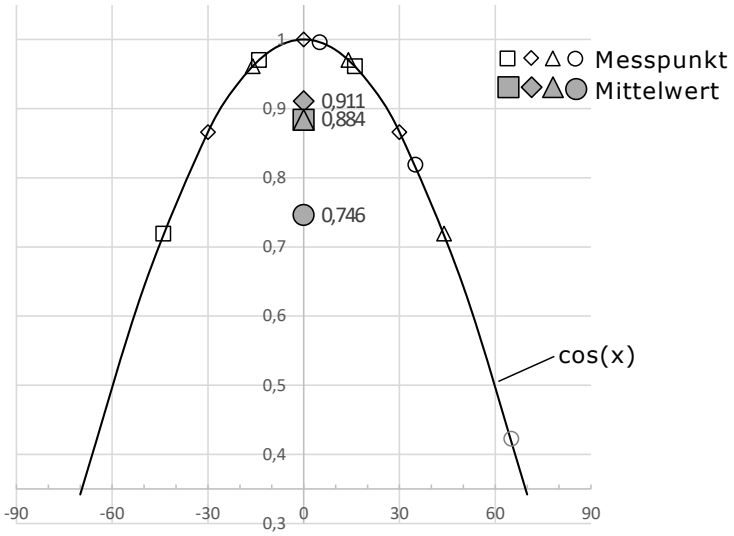


Abbildung 7.8: Ergebnisse der Mittelwertberechnung bei unterschiedlichen Positionen der jeweils drei Messwerte (Anh. B.2.8.1)

Einfluss Position Der Maximalwert wird in jedem Fall unterschätzt, siehe Abb. 7.8. Wenn die Positionsabweichung nicht groß ist, sind die Ergebnisse der Mittelung nahe beieinander. Eine falsche Position führt zu großen Abweichungen, es ist daher wichtig, die Positionsabweichung durch einen Algorithmus, der vor der Mittelwertberechnung ausgeführt wird, zu minimieren.

Die Mittelwerte unterscheiden sich dann relativ wenig voneinander, daher ist es möglich, sie mit einem Korrekturfaktor Richtung Soll-Wert zu korrigieren und so der relativ konstanten Abweichung entgegenzuwirken.

Einfluss Rauschen Das Rauschen wird durch die Mittelwertbildung reduziert. Fehlinterpretationen wie bei der Parabelberechnung sind nicht möglich.

Zusammenfassung

\+ Einfache und stabile Berechnung

\+ Reduzierung der Einwirkung von Messfehlern

\+ Verbesserungen durch weitere Maßnahmen möglich

\- Abweichung ohne weitere Maßnahmen relativ groß

\- Neigt zum Unterschätzen von Maxima und Überschätzen von Minima

7.3.2.3 Annäherung des harmonischen Signals mit Parabel durch drei Punkte

Ein harmonisches Signal lässt sich in der Nähe der Extremwerte durch eine Parabel annähern. Diese ist durch drei Punkte vollständig bestimmt und ermöglicht eine sehr schnelle Berechnung des Extremwertes sowie der Position des Extremwertes anhand weniger Rechenschritte für Minima und für Maxima.

Selbst wenn der ermittelte Zeitpunkt nicht genau mit dem Extremum übereinstimmt, stimmt das Ergebnis gut mit dem theoretischen Signal überein. Hier liegt der große Vorteil gegenüber den anderen Methoden.

Problematisch ist jedoch, dass das Rauschen des Signals zu groben Fehlern in der Parabel führen kann. Bei ungünstiger Anordnung der gemessenen Punkte können unbrauchbare Ergebnisse auftreten.

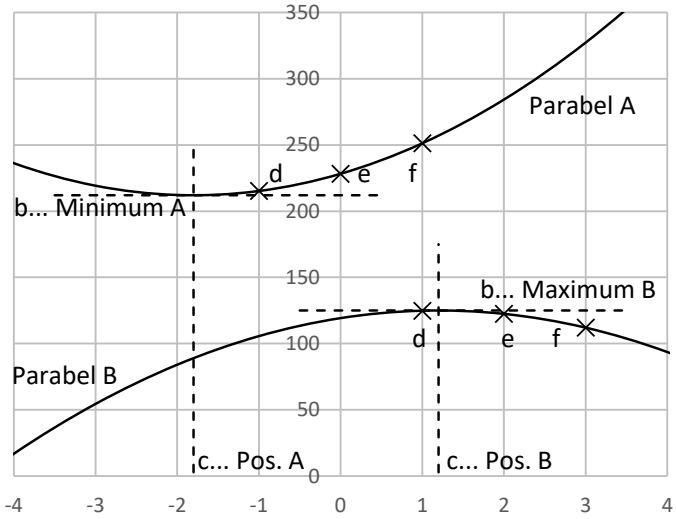


Abbildung 7.9: Parabeln mit Minimum und Maximum (Anh. B.2.8.1)

Ermittlung In Abb. 7.9 sind zwei unabhängige Parabeln zu sehen, mit jeweils drei Punkten darauf lassen sich die Parameter Krümmung a, Extremwert b und Verschiebung c bestimmen. Die Punkte müssen dabei nicht in der Nähe des Scheitelpunktes liegen und die Bestimmung für Minima und Maxima ist gleich. Parabel-Funktion $y(x)$:

$$y(x) = a * (x - c)^2 + b$$

Da der zeitliche Abstand zwischen den Punkten jeweils eine Einheit beträgt, erhält man durch Einsetzen der drei Punkte $[-1,d]$, $[0,e]$ und $[1,f]$:

$$d = y(-1) = a * (-1 - c)^2 + b$$

$$e = y(0) = a * (-c)^2 + b$$

$$f = y(1) = a * (1 - c)^2 + b$$

Daraus ergeben sich die Bestimmungsgleichungen für die Kennwerte der Parabel:

$$a = \frac{d+f}{2} - e$$

$$b = e - \frac{(d-f)^2}{8*(d+f)-16*e}$$

$$c = \frac{d-f}{2*(d+f)-4*e}$$

Der Extremwert b der Parabel durch die drei Punkte ist damit nur unwesentlich aufwendiger zu berechnen als deren Mittelwert. Liegen die Punkte auf einer Gerade, wird der Nenner Null (bei $d - e = e - f$), das ist vor der Berechnung zu auszuschließen.

Die Verschiebung c in x-Richtung ist vom Index des mittleren Punktes e ausgehend in Sampleschritt-Einheiten definiert.

Einfluss Position Da die Parabel die harmonische Funktion in der Nähe des Extremwertes gut annähert, kommt es bei einer Positionsverschiebung und ungestörten Messwerten zu bedeutend kleineren Abweichungen gegenüber dem Sollwert (Amplitude=1) als bei der Mittelwertbildung (MW) (Abb. 7.8) und dem nach Kap. 7.3.5.2 korrigierten Mittelwert (korr. MW), siehe Tab. 7.9 (Spalte b) und die grafische Darstellung in Abb. 7.10.

rel. Verschiebung		rel. Verschiebung	Winkel x	Wert cos(x)	a	b	c	MW	korr. MW
0,0 (□)	d	-1,0	-30°	0,866	-0,134	1,000	0,000	0,911	1,008
	e	0,0	0°	1,000					
	f	1,0	30°	0,866					
0,6 (△)	d	-0,4	-12°	0,978	-0,127	0,998	-0,606	0,866	0,959
	e	0,6	18°	0,951					
	f	1,6	48°	0,669					
1,2 (×)	d	0,2	6°	0,995	-0,108	1,008	-1,356	0,737	0,819
	e	1,2	36°	0,809					
	f	2,2	66°	0,407					
1,7 (+)	d	0,7	21°	0,934	-0,084	1,077	-2,304	0,573	0,634
	e	1,7	51°	0,629					
	f	2,7	81°	0,156					

Tabelle 7.9: Parabelkennwerte bei unterschiedlichen Positionen und Vergleich zu den Mittelwertmethoden (Anh. B.2.8.1)

Dabei werden auf einer Kosinus-Funktion die Punkte für die Ermittlung der Parabel ermittelt, die Punkte liegen darauf mit 30° Abstand. Die Verschiebung ist zwischen dem Extremwert (0°) und dem mittleren Punkt e zu verstehen.

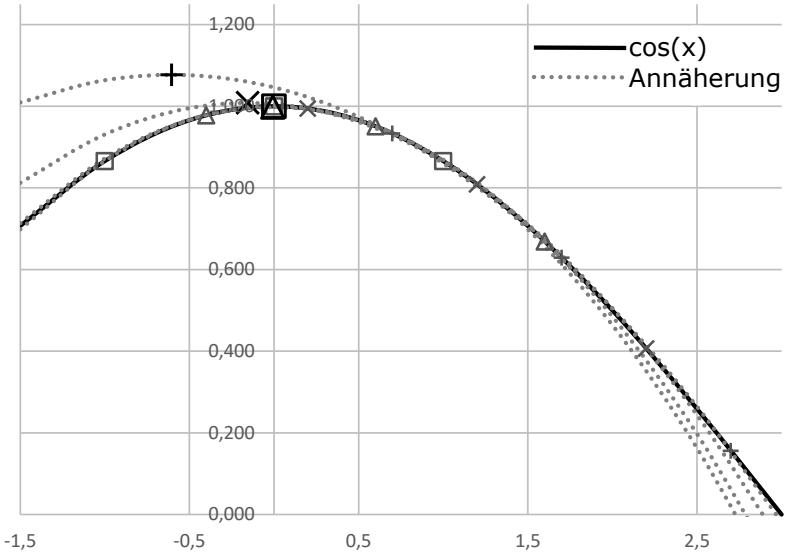


Abbildung 7.10: Parabeln und deren Extremwerte durch 3 Punkt bei unterschiedlichen Positionen (Anh. B.2.8.1)

Einfluss Rauschen Die vorhergehenden Ergebnisse wurden mit idealen Messwerten ermittelt. Fehlerhafte Messpunkte können große Auswirkungen auf die Form und Position der Parabel haben, wodurch Ergebnisse entstehen, die nicht mit der ursprünglichen Kurve vergleichbar sind. Besonders empfindlich auf Änderungen reagiert die Parabel, wenn die drei Punkte annähernd auf einer Geraden liegen. In vielen Fällen kann diese Vorgehensweise daher nicht genutzt werden.

Zusammenfassung

\+ Minimale Abweichung bei guten Messdaten selbst bei schlechter Position

\- Anfälligkeit für völlig falsche Ergebnisse bei etwas schlechteren Messdaten

\- Prüfung der Daten nötig, ansonsten Division durch 0 möglich

7.3.3 Maßnahmen zur Verbesserung der Extremwertermittlung

Da die Varianten aus Kap. 7.3.2 unterschiedliche Stärken und Schwächen haben, können gezielte Maßnahmen eine Verbesserung des letztendlich ermittelten Ergebnisses bringen.

7.3.3.1 Kombination der Verfahren

Das Parabel-Verfahren ermöglicht eine sehr geringe Abweichung bei guten Messdaten, während der korrigierte Mittelwert ebenfalls eine gute Annäherung bietet, aber bei Signalstörungen überlegen ist. Es ist möglich, die Verfahren zu kombinieren und dadurch ein im Schnitt besseres Ergebnis zu erzielen (Kap. 7.3.5.3).

Durch max. Abweichung vom gemittelten Ergebnis Durch eine maximale Abweichung des Parabel-Ergebnisses vom korrigiertem Mittelwert in Abhängigkeit der Differenz der Extremwerte können größere Abweichungen der Parabelnäherung bei ungünstigen Konstellationen vermieden werden, im Zweifel wird dann der Mittelwert verwendet. Stimmt die Parabel gut mit dem Mittelwert überein, ist sie oft genauer als dieser. Es wird durch diese Kombination ein geringfügig besseres Ergebnis erreicht.

Durch Plausibilitätsprüfung der Parabelkrümmung Um die Vorteile der Parabel-Berechnung nutzen zu können, die Fehlinterpretationen jedoch zu vermeiden, kann als Prüfparameter die Krümmung der Kurve (Parameter a) verwendet werden. Diese Krümmung wird mit der Krümmung eines harmonischen Signals verglichen, das näherungsweise gleiche Extremwerte besitzt. Es zeigt sich, dass dadurch keine weiteren nennenswerten Vorteile im Vergleich zur zuvor beschriebenen Möglichkeit erreichbar sind (Tab. 7.10 G und H), die Komplexität der Auswertung jedoch zunimmt.

7.3.3.2 Suche des Extremwertes in nächster Umgebung der Positionsabschätzung

Die Bestimmung des Extremwertes funktioniert am besten, wenn dessen Position möglichst genau bekannt ist.

Wenn die anhand des Nulldurchgangs des SCPEM-Signals vorbestimmte Position des Extremwerts nicht stimmt, kann sie anhand der Intensitäts-Messwerte korrigiert werden, bevor der Extremwert selbst ermittelt wird. Dazu wird zuerst überprüft, ob es sich bei den drei Punkten um einen Extremwert handelt. Da dort eine Vorzeichenänderung der ersten Ableitung eintritt, müssen die Vorzeichen der Differenzen d-e und e-f unterschiedlich sein. Ist dies nicht der Fall, wird in der nächsten Umgebung gesucht, bis ein Extremwert gefunden ist. Die genaue Umsetzung dieser Suche ist in Kap. 7.3.5.1 beschrieben.

7.3.3.3 Verringerung der Abweichung bei Mittelwertermittlung

Die zu Beginn verwendete Variante $(d + 2 * e + f)/4$ gewichtet den mittleren Punkt stärker und verringert dadurch die Abweichung vom Extremwert.

Ein eigener Korrekturwert bietet eine andere Möglichkeit, die Abweichung zu verringern. Die Ermittlung erfolgt dann durch die Mittelwertbildung $(d + e + f)/3$, anschließend wird das Ergebnis korrigiert. Der Korrekturfaktor hängt davon ab, wie weit die drei Messpunkte auf der Messkurve voneinander entfernt sind und kann nach Kap. 7.3.5.2 ermittelt werden.

7.3.4 Vergleich der Methoden

Die unterschiedlichen Algorithmen werden an einem simulierten Signal, von dem alle Kenngrößen bekannt sind, verglichen. Dem Algorithmus werden jeweils 3 Punkte von dieser Kurve mit überlagerten normalverteilten zufälligen Fehlern zur Verfügung gestellt. Wie bei der Messung am Messaufbau sind die Messpunkte zueinander in einem ähnlichen Abstand wie beim 80 kHz-SCPEM mit unterschiedlichen, zufälligen zeitlichen Verschiebungen zum tatsächlichem Extremwert angeordnet.

Aus vielen Durchläufen dieser Simulation lassen sich Zielgrößen ermitteln, diese sind die mittlere Abweichung vom Soll-Extremwert sowie die Variation der Abweichungen, beide sind zu minimieren.

Für folgende Algorithmusvariationen wird in der Simulation jeweils 10.000 mal je Kriterium der Extremwert mit zufällig überlagerten Fehlern gesucht und mit dem Soll-Extremwert (der Amplitude) verglichen (A... einfachste Methode bis H... aufwändigste Methode):

A: Einzelwert, ohne Extremwertsuche

B: Mittelwert, ohne Extremwertsuche

C: Parabel, ohne Extremwertsuche

D: Mittelwert, mit Extremwertsuche

E: Parabel, mit Extremwertsuche

F: Korrigierter Mittelwert, mit Extremwertsuche (im Messprogramm umgesetzt)

G: Kombination korrigierter Mittelwert/Parabel mit Überprüfung durch max. Abweichung, mit Extremwertsuche (im Messprogramm umgesetzt)

H: Kombination korrigierter Mittelwert/Parabel mit Überprüfung durch Krümmungsanalyse, mit Extremwertsuche

Tab. 7.10 zeigt die Ergebnisse für die mittlere Abweichung und die Verteilung der Ergebnisse, jeweils in Promille auf die Signalamplitude bezogen. Für kleine Amplituden ist der relative Fehler bei gleichem Signalrauschen und Digitalisierungsfehler daher höher.

Die ersten vier Kriterien entsprechen einer Messung bei guten Bedingungen und unterschiedlichen Amplituden, die letzten drei zeigen die Auswirkung eines höheren Rauschanteils und der Verschiebung der Auswertungsposition weh vom Extremwert.

F, G und H schneiden am besten ab (E hat eine zu große Standardabweichung der Ergebnisse), damit können die eingesetzten Verbesserungsmethoden als effektiv bezeichnet werden. Die Methoden, die eine Parabelberechnung verwenden, liefern bei großen Amplituden sehr exakte Ergebnisse. Die Abweichung des Algorithmus F liegt in diesem Bereich im Schnitt ebenfalls unter 1%.

Die Gesamtbewertung entspricht der gemittelten Bewertung über die sieben Kriterien.

Amplitude Intensitäts-Signal			**1**		**10**		**100**		**1000**		100		100		100	
Signalrauschen (normalvert.)			1		1		1		1		**5**		1		**5**	
Zusätzl. Positionsabweichung			0		0		0		0		0		**2**		**2**	
Variante	Gesamt: Abweichung ‰	Gesamt: Verteilung ‰	Fehler ‰	Verteilung ‰	Fehler ‰	Verteilung ‰	Fehler ‰	Verteilung ‰	Fehler ‰	Verteilung ‰	Fehler ‰	Verteilung ‰	Fehler ‰	Verteilung ‰	Fehler ‰	Verteilung ‰
A	272	212	748	1031	76	103	14	15	11	10	41	51	507	131	507	141
B	283	137	478	607	102	60	100	11	100	9	100	30	551	119	552	123
C	379	646	979	1175	166	309	8	10	1	1	44	89	358	584	1098	2357
D	156	110	478	607	102	60	100	11	100	9	107	36	100	12	108	38
E	177	228	979	1175	166	309	8	10	1	1	37	45	8	10	37	45
F	93	117	515	642	50	64	10	12	8	10	30	37	10	12	31	40
G	93	117	515	642	51	68	8	10	1	1	35	44	8	10	35	45
H	93	116	515	642	51	66	8	10	1	1	32	41	8	10	33	42

Tabelle 7.10: Simulierter Vergleich verschiedener Methoden bei unterschiedlichen Messpunktkriterien (Anh. B.2.8.2)

Aufgrund des höheren Aufwands von H und der vernachlässigbaren Verbesserung von H gegenüber G werden die Algorithmen F und G implementiert, es kann bei Bedarf zwischen diesen beiden ausgewählt werden (durch eine Änderung im Sourcecode, Kap. 7.3.5.3).

7.3.5 Implementierung im Messprogramm

Im Messprogramm können verschiedenen Algorithmen aus Kap. 7.3.4 verwendet werden, die letztendlich eingesetzte Methode nutzt eine Extremwertsuche und die korrigierte Mittelwertberechnung mit je drei Samples (Variante C), Variante A ist ebenfalls möglich.

7.3.5.1 Suche der zeitlichen Position des Extremwerts

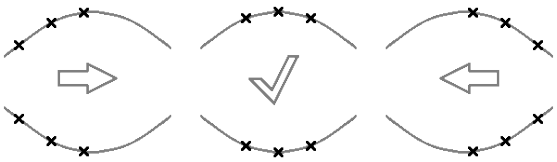


Abbildung 7.11: Vorgangsweise beim Suchen der Extremwertposition

Vorgangsweise Zunächst wird auf einen ausreichend großen Unterschied zwischen den ursprünglich ermittelten Mittelwerten geprüft, erst bei ausreichend ausgeprägten Extremwerten funktioniert die Suche gut und wird nicht zu sehr durch überlagertes Rauschen gestört.

Dann wird überprüft, ob ein Extremum in den drei Punkten vorhanden ist (Abb. 7.11). Ist dies nicht der Fall, wir abhängig davon, ob es ein Maximum oder Minimum sein soll, der Index in eine bestimmte Richtung verschoben und die Überprüfung wiederholt. Ist ein Extremum gefunden oder eine maximale Anzahl an Verschiebungen erreicht, wird die Suche abgebrochen. Abschließend wird ein neuer Mittelwert an der letzten Position berechnet.

Implementierung Es sind die neun in Tab. 7.11 dargestellten Formationen von den drei aufeinanderfolgenden Samples möglich (es wird jeweils betrachtet, ob diese gleich, größer oder kleiner als die anderen in der Dreiergruppe sind).

	1			2			3			4			5			6			7		
	d	e	f	d	e	f	d	e	f	d	e	f	d	e	f	d	e	f	d	e	f
+				X				X				X									
0	X	X	X		X	X	X		X	X	X			X	X	X		X	X	X	
-													X				X				X

	8			9		
	d	e	f	d	e	f
+			X	X		
0		X			X	
-	X					X

d, e, f	Messpunkte
X	relative Position
+	relativ größer
0	relativ gleich
-	relativ kleiner

Tabelle 7.11: Mögliche Kombinationen von drei aufeinanderfolgenden Messpunkten

Wird eine ungestörte Kurve mit ausreichend großer Periodendauer vorausgesetzt, weisen alle Formationen der ersten Zeile (1-7) auf einen Extremwert zwischen den Messwerten hin, da dieser entweder direkt ersichtlich ist (3 und 6) oder durch zwei gleichbleibende Werte angedeutet wird (2, 4, 5, 7), wodurch sich ein Extremwert dazwischen befinden muss.

Die Formationen der zweiten Zeile (8 und 9) befinden sich eindeutig nicht auf einem Extremum und lassen die Suchrichtung direkt erkennen, wenn bekannt ist, ob in Richtung eines Maximums oder Minimums gesucht wird. Diese Unterscheidung wird durch den Größenvergleich der ersten Näherung der beiden Extremwerte festgestellt. Handelt es sich um ein Maximum, ist bei Formation 8 der Extremwert nach rechts und bei Formation 9 nach links zu suchen. Bei einem Minimum ist es umgekehrt. Wird der Extremwert nicht in der Nähe der berechneten Position gefunden, wird die Suche bei einem Grenzwert abgebrochen.

Algorithmus 7.3 Suche der Position des Extremwertes (Anh. C.1.2)

```
1  // Buffer-Index der Extremwerte abschaetzen:
2  index0= (uint16_t)(roundf(trig+(*tp0 )));
3  index90=(uint16_t)(roundf(trig+(*tp90)));
4
5  // benoetigte Samples einlesen (Extremwert 1) -> Verschiebung 0deg SCPEM-Periode
6  GET_SAMP_OFFSET1(index0-1 ,d0 );
7  GET_SAMP_OFFSET1(index0   ,e0 );
8  GET_SAMP_OFFSET1(index0+1 ,f0 );
9  // benoetigte Samples einlesen (Extremwert 2) -> Verschiebung 90deg SCPEM-Periode
10 GET_SAMP_OFFSET1(index90-1,d90);
11 GET_SAMP_OFFSET1(index90  ,e90);
12 GET_SAMP_OFFSET1(index90+1,f90);
13
14 // Mittelwerte bilden (fuer erste Abschaetzung Minima oder Maxima):
15 mid0 =( d0+ e0+ f0)/3;
16 mid90=(d90+e90+f90)/3;
17
18 // Extremwert-Position suchen (falls diese nicht bereits vorhanden ist...):
19 // Ausreichend ausgepraegte Kurve noetig -> bei zu flacher Kurve kann durch Rauschen der
       Extremwert nicht zuverlaessig erkannt werden -> (Amplitude > 16) als Grenze gewaehlt
20 if (abs(mid0-mid90)>32){
21     // Suchen des ersten Extremwertes:
22
23     // index_corr ist der korrigierte Index:
24     index_corr=index0;
25     // Pruefen, ob Samp. steigend/fallend sind und Versch. < Grenzwert ist (max. 3
           Verschiebungen gewaehlt):
26     while ( ( ( d0>e0 && e0>f0 ) || ( d0<e0 && e0<f0 ) ) && ( abs(index_corr-index0)<4 ) ){
27         //        \>/       \>/           \</      \</                          \____<____/
28         //         \___and___/             \___and___/                              /
29         //             \_____________or_____________/                              /
30         //                           \                                            /
31         //           (Samples steigend oder fallend)   (Verschiebung noch nicht am Grenzwert)
32         //                            \_____________________and___________________/
33         //                                          /
34
35         // Samples sind steigend oder fallend, Verschiebung der Position:
36         // Verschieberichtung in Abhaengigkeit von Minima/Maxima feststellen:
37         if ((d0>f0) == (mid0>mid90)) {
38             // TRUE == TRUE: Samples fallend, Maximum
39             // FALSE==FALSE: Samples steigend, Minimum
40             // -> Extremwert liegt weiter links (Verschieben 1 Sampleschritt, 1x neu einlesen)
41             index_corr-=1;                          // <--
42             f0=e0;                                  // f bei index_corr+1
43             e0=d0;                                  // e bei index_corr
44             GET_SAMP_OFFSET1(index_corr-1,d0);      // d bei index_corr-1
45         } else {
46             // TRUE !=FALSE: Samples fallend, Minimum
47             // FALSE!= TRUE: Samples steigend, Maximum
48             // -> Extremwert liegt weiter rechts (Verschieben 1 Sampleschritt, 1x neu einlesen)
49             index_corr+=1;                          // -->
50             d0=e0;                                  // d bei index_corr-1
51             e0=f0;                                  // e bei index_corr
52             GET_SAMP_OFFSET1(index_corr+1,f0);      // f bei index_corr+1
53         }
54     }
55
56     // nach Suche: -> Samples entsprechen jetzt Extremwert, lassen keinen eindeutigen Schluss
           ueber die Richtung zu oder es wurde bereits 3-mal verschoben
57     // bei Aenderung des Index Mittelwert aktualisieren:
58     if (index_corr!=index0){
59         mid0=(d0+e0+f0)/3;
60         // wenn stark ausgepraegte Kurve (eindeutige Extremwertposition): Phasenverschiebung
               leicht in richtige Richtung korrigieren (bei Problemen einer instabilen
               Extremwertposition im Messverlauf diesen Teil entfernen oder Grenzwert erhoehen):
61         if (abs(mid0-mid90)>256){
62             if (index_corr>index0) {(*tp0)+=0.001;} else {(*tp0)-=0.001;}
63         }
64     }
65
66     // Suchen des zweiten Extremwertes:
67     // ...
68     // ... (analog Extremwert 1)
69 }
```

Alg. 7.3 zeigt die optimierte Umsetzung dieser Suche für einen der beiden benötigten Extremwerte.

7.3.5.2 Korrekturfaktor für verbesserte Mittelwertberechnung

Bestimmung Korrekturfaktor f_{corr} anhand eines Beispiels Trotz richtiger Bestimmung der Extremwertposition kommt es durch die verbleibende Verschiebung und die Verteilung der drei Punkte auf der Signalkurve zu einer Abweichung des Mittelwertes (siehe Diagramm 7.8: je Position 0,884 bis 0,911 statt 1,000). Da diese relativ nahe beieinander liegen, kann ein Korrekturfaktor das Ergebnis näher an den theoretischen Wert bringen.

Die Zielgröße ist eine möglichst kleine mittlere Abweichung des Ergebnisses. Durch Probieren ist beispielsweise der Korrekturfaktor $f_{corr} = 1{,}110$ möglich, damit ergeben sich die Ergebnisse $0{,}884 * 1{,}110 = 0{,}981$ und $0{,}911 * 1{,}110 = 1{,}011$, welche nahe an 1,000 liegen. Dieser Wert ist nur für eine bestimmte SCPEM-Frequenz gültig.

Allgemeine Herleitung und Berechnung des Korrekturfaktors Es wird ein Faktor gesucht, der sich schnell aus der aktuellen Frequenz errechnen lässt und nicht durch Probieren gesucht werden muss.

Der Phasenabstand zwischen den Samples auf der Intensitätskurve (doppelte SCPEM-Frequenz) wird als ψ bezeichnet. Bei Verwendung des 80 kHz-SCPEM und Dezimation 64 ergibt sich ungefähr $\psi = 29{,}5°$.

$$\psi = 2\pi * \frac{2 * freq_{SCPEM}}{Samplerate}\ rad$$

Dann liegt der mittlere Messpunkt im Bereich $-\alpha/2$ und $+\alpha/2$ um den idealen Extremwert, wobei jede Position im Schnitt gleich oft auftritt (es besteht im Allgemeinen kein ganzzahliger Zusammenhang zwischen SCPEM-Frequenz und Sampleaufzeichnungsfrequenz). Die relative Mittelwert M sehr vieler Mittelwerte m bei einem harmonischen Signal der Amplitude amp lässt sich daher als

$$M = \frac{1}{\psi} * \int_{-\psi/2}^{\psi/2} m * d\beta$$

mit dem Mittelwert der jeweiligen drei Punkte

$$m = \frac{d+e+f}{3} = amp * \frac{cos(\beta-\psi)+cos(\beta)+cos(\beta+\psi)}{3}$$

ausdrücken. Die Extremwertposition ist bei der verwendeten Kosinus-Funktion bei 0.

$$M = \frac{amp}{3\psi} * \int_{-\psi/2}^{\psi/2} (cos(\beta - \psi) + cos(\beta) + cos(\beta + \psi)) * d\beta$$

$$M = \frac{amp}{3\psi} * \left(\left(sin\left(\frac{3\psi}{2}\right) - sin\left(\frac{\psi}{2}\right)\right) + \left(2 * sin\left(\frac{\psi}{2}\right)\right) + \left(sin\left(\frac{3\psi}{2}\right) - sin\left(\frac{\psi}{2}\right)\right)\right)$$

$$M = \frac{2 * amp}{3\psi} * sin\left(\frac{3\psi}{2}\right)$$

Um daraus den Korrekturfaktor f_{corr_ideal} zu erhalten, muss $m_{corr} = m * f_{corr\_ideal} = amp$ sein. Im Mittel gilt $m = M$ und damit $f_{corr\_ideal} = amp/M = (3\psi)/(2 * sin(3\psi/2))$. Die Amplitude kann damit eliminiert werden und der Korrekturfaktor lässt sich durch ψ ausdrücken. Im Anwendungsfall ergibt sich etwa $f_{corr\_ideal} = 1{,}107$.

Optimierte Anwendung der Korrektur im Messprogramm Für die Anwendung im Messsystem muss der Korrekturfaktor für eine möglichst schnelle Ausführung ohne Zwischenschritte angepasst werden. Bisher wurde nur die Abweichung der Amplitude berücksichtigt, das Signal hat zusätzlich einen Gleichanteil, daher ist eine einfache Multiplikation nicht zielführend. Es werden zwei Extremwerte a und b betrachtet (ein Maximum und ein Minimum), m_a und m_b sind die aus jeweils drei Punkten berechneten Mittelwerte.

Der Gleichanteil ist als $mean = (m_{a\_corr} + m_{b\_corr})/2 = (m_a + m_b)/2$ und die Amplitude als $amp_{corr} = (m_{a\_corr} - m_{b\_corr})/2$ berechenbar. Der unkorrigierte Gleichanteil ist gleich dem korrigiertem, die unkorrigierte mittlere Amplitude ist $M = amp_{not\_corr} = amp_{corr}/f_{corr\_ideal} = \pm(m_a - m_b)/2$. Damit sind die beiden korrigierten Extremwerte:

$$m_{a\_corr} = mean + amp = mean + amp_{not\_corr} * f_{corr\_ideal}$$

$$m_{a\_corr} = (m_a + m_b)/2 + (m_a - m_b)/2 * f_{corr\_ideal}$$

$$m_{b\_corr} = mean - amp = mean - amp_{not\_corr} * f_{corr\_ideal}$$

$$m_{b\_corr} = (m_a + m_b)/2 + (m_b - m_a)/2 * f_{corr\_ideal}$$

Bezieht man die Abweichung der Amplitude $(amp_{corr} - amp_{not\_corr})$ auf die doppelte, unkorrigierte Amplitude $2 * amp_{not\_corr} = \pm(m_a - m_b)$, erhält man den Faktor

$$f_{corr} = \frac{f_{corr\_ideal} * amp_{not\_corr} - amp_{not\_corr}}{2 * amp_{not\_corr}} = \frac{f_{corr\_ideal} - 1}{2} = \frac{\frac{3\psi}{2 * sin(\frac{3\psi}{2})} - 1}{2} = \frac{3\psi}{4 * sin(\frac{3\psi}{2})} - \frac{1}{2}$$

Damit lässt sich mit $f_{corr\_ideal} = 2 * f_{corr} + 1$ in die Extremwertgleichung einsetzen und umformen (analog für m_{b_corr}):

$$m_{a\_corr} = (m_a + m_b)/2 + (m_a - m_b)/2 * f_{corr\_ideal} = (m_a + m_b)/2 + (m_a - m_b)/2 * (2 * f_{corr} + 1)$$

$$m_{a\_corr} = (m_a + m_b)/2 + (m_a - m_b)/2 + (m_a - m_b) * f_{corr} = m_a + (m_a - m_b) * f_{corr}$$

Als Ziel dieser Umformungen ergeben sich für die korrigierten Extremwerte die einfachen Ausdrücke:

$$m_{a\_corr} = m_a + (m_a - m_b) * f_{corr} \quad und \quad m_{b\_corr} = m_b + (m_b - m_a) * f_{corr}$$

Der Korrekturfaktor f_{corr} wird vor Messbeginn abhängig von der SCPEM-Frequenz wie folgend berechnet (Anh. C.1.1, Anh. C.1.3 und Anh. C.1.4) und dann bei jeder Extremwertermittlung (Anh. C.1.2) verwendet:

```
// p_akt...          Samples je SCPEM-Periode (aktuelle Periodendauer):
        p_akt          = (125000000/64.)/f_SCPEM;
// psi...            Phasenabstand zwischen 2 Samples:
        psi            = 4.*pi/p_akt;
// factor_corr...    Korrekturfaktor:
        factor_corr = 0.75*psi/sin(1.5*psi) - 0.5;
```

Im Anwendungsfall ergibt sich etwa $f_{corr} = 0{,}053$. Eine kleine Änderung (1-2%) des Faktors in der Simulation (Kap. 7.3.4) zeigt bereits eine Verschlechterung des Ergebnisses, sowohl beim Erhöhen wie beim Verringern, wodurch die Argumentation beim Herleiten bestätigt wird.

7.3.5.3 Ermittlung der Extremwerte und Kombination der Verfahren

Die detaillierte Umsetzung der Berechnung befindet sich in Anh. C.1.2, hier sind nur kurze Ausschnitte davon erwähnt.

Ermittlung des korrigierten Extremwertes Die Ermittlung des korrigierten Extremwertes erfolgt nach der Suche der Extremwertposition (Alg. 7.3) mit dem Korrekturfaktor aus Kap. 7.3.5.2:

```
*I0 = mid0+(mid0-mid90)*factor_corr;
*I90=mid90+(mid90-mid0)*factor_corr;
```

Ermittlung nach der Parabel-Methode Nachdem sichergestellt ist, dass die Samples nicht auf einer Gerade liegen ($d0 - e0! = e0 - f0$), gilt für die Bestimmung des Parabelextremwertes:

```
*I0 = e0-(pow(d0 - f0,2))/(8*(d0 + f0)-16*e0 );
*I90=e90-(pow(d90-f90,2))/(8*(d90+f90)-16*e90);
```

Kombination Die Kombination der beiden Verfahren kann das Ergebnis im Schnitt verbessern, wie Tab. 7.10 zeigt. Die Auswahl des Verfahrens ist dabei im Messprogramm (und in der Simulation, Kap. 7.3.4) folgenderweise definiert:

- die Extremwerte werden nach beiden Verfahren bestimmt
- wenn die Abweichung zwischen den Verfahren größer als 1/16 der (näherungsweise bekannten) Amplitude ist, wird das Ergebnis der korrigierten Mittelwertberechnung verwendet (damit werden grobe Abweichungen der Parabelmethode ausgeschlossen)
- ansonsten wird das Ergebnis der Parabelberechnung verwendet

Im Sourcecode (Anh. C.1.2) sind beide Varianten umgesetzt, die Auswahl (Mittelwert oder Kombination) kann durch Auskommentieren geändert werden. Standardmäßig wird aufgrund des guten Verhaltens und der einfacheren Nachvollziehbarkeit der korrigierte Mittelwert verwendet.

7.3.6 Auswirkung der Verbesserungen auf das Messergebnis

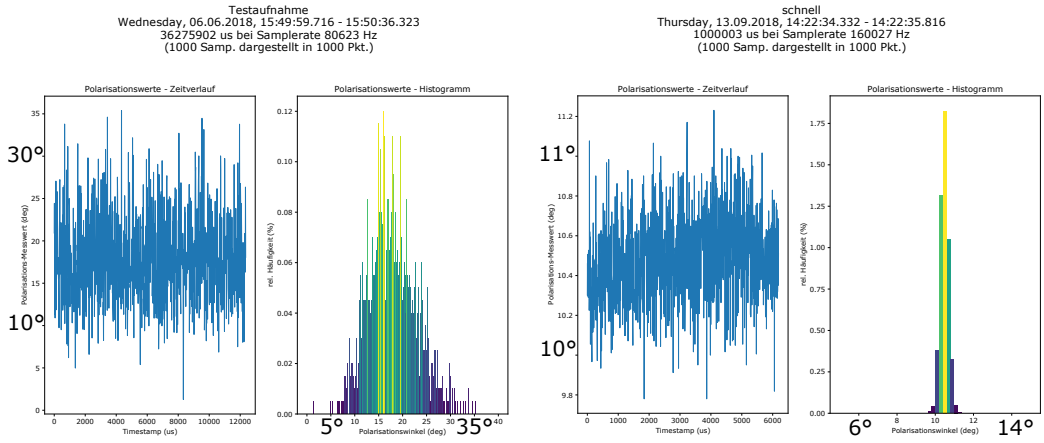


Abbildung 7.12: Verbesserung der Extremwertermittlung - Auswirkung auf das Messergebnis (Anh. B.1.13)

Der betriebene Aufwand der Optimierung des Algorithmus scheint groß, jedoch kann nur dadurch ein brauchbares Messergebnis bei hoher Messfrequenz erreicht werden. In Abb. 7.12 sind zwei Aufnahmen mit dem Messprogramm zum Vergleich angeführt:

- Links: kurze Aufnahme mit einfacher Extremwertsuche ohne Optimierungen (Parabelmethode) bei 80 kHz Samplerate
- Rechts: kurze Aufnahme mit verbessertem, aktuellen Algorithmus (Kombination korrigierter Mittelwert/Parabel mit Extremwertsuche) bei doppelter Samplerate (160 kHz)

Zu beachten ist dabei die verringerte Streuung der aufeinanderfolgenden Messwerte im Histogramm. Die Standardabweichung der Messwerte konnte von ca. 10 Grad oder mehr (abhängig vom Messbereich) stark reduziert werden und beträgt trotz doppelter Samplerate weniger als ein Grad Polarisationswinkel. Bei niedrigerer Samplerate und Mittelwertbildung kann diese noch weiter verringert werden.

Verantwortlich für die Verbesserung ist die weiterentwickelte Extremwertermittlung, da der Berechnungsalgorithmus der Polarisation (Kap. 7.5) und der Messaufbau bei diesen Messungen unverändert ist.

7.4 Vorbereitung der benötigten Kennwerte zur Messung

Bevor die Messung gestartet werden kann, müssen einige Kennwerte festgestellt oder festgelegt werden. Die Werte können dabei vom Nutzer vorgegeben oder vom Programm gemessen werden, die gewünschten Einstellungen werden vom Computer-Clientprogramm über UDP an das Messprogramm übermittelt und dort ausgeführt.

7.4.1 Ermittlung von SCPEM-Trigger und -Frequenz

Das SCPEM-Signal ist harmonisch, der Gleichanteil beträgt Null. Zur Ermittlung der SCPEM-Frequenz muss in jeder Periode eine charakteristische Position am Signal ermittelt werden. Dafür wird der Nulldurchgang des Signals an der ansteigenden Flanke genutzt. Diese ist einfach festzustellen, ein Sample muss negativ sein, das darauffolgende positiv. Aus diesen beiden kann dann durch Interpolation die Position des Nulldurchgangs genauer bestimmt werden (ein harmonisches Signal kann in der Nähe des Nulldurchganges gut durch eine Gerade angenähert werden).

Dieser Nulldurchgang wird im gesamten Messprogramm als Trigger verwendet. Damit wird die Frequenz bestimmt und der Trigger dient als Bezug für die Position der Extremwerte des Intensitätssignals relativ dazu.

Die Umsetzungen sind dem Sourcecode (Anh. C.1.4 und Anh. C.1.2) zu entnehmen und in den Programmablaufplänen Abb. 7.19 und Abb. 7.6.3 angedeutet.

7.4.2 Ermittlung der Phasenverschiebung

Aufgrund der Phasenverschiebung zwischen Intensitätssignal und SCPEM-Signal muss die relative Position der Extremwerte ermittelt werden und während der Aufnahme bekannt sein.

Bei der Bestimmung wird eine definierte Polarisation vorausgesetzt (ungefähr parallel zum Analysator). Dann können die stark ausgeprägten Maxima und Minima am Intensitätssignal bestimmt und deren relative Position zum Triggerzeitpunkt festgelegt werden. Diese Positionen werden gespeichert und bei der Ermittlung der Extremwerte während der Messung verwendet.

Die Umsetzung ist dem Sourcecode (Anh. C.1.4) zu entnehmen, die vier ermittelten Positionen je Periode sind in in Abb. 7.5 erkennbar.

7.4.3 Ermittlung der maximalen Retardation

Die Retardation kann bei einer definierten Polarisation automatisch ermittelt werden, dazu wird die in Kap. 3.2.3.2 gezeigte Methode verwendet, die Umsetzung ist Anh. C.1.4 zu entnehmen.

7.4.4 Startinitialisierung

Direkt vor Aufnahmebeginn müssen einige Aktionen erledigt werden (Kap. 7.6.3). Es besteht die Option, vor Start der Aufnahme eine bestimmte Signalgröße abzuwarten.

Sind alle Bedingungen erfüllt, wird der erste Trigger bestimmt, dann wird die Aufnahme gestartet.

Genauere Informationen sind im Programmablaufplan (Kap. 7.6.3, Abb. 7.17) und im Sourcecode (Anh. C.1.2) zu finden.

7.5 Polarisationsberechnung

Algorithmus 7.4 Polarisationsberechnung im Messprogramm (Anh. C.1.2)

```
1  // Berechnung des Polarisatioswinkels pol
2  // aus den ermittelten Intensiaeten I0 und I90:
3
4  if (I0==I90) {                    // Verhinderung div0
5      pol=0;
6  } else {
7      Q=s*(C_*I0/(I0-I90)-1.);      // Quotient
8      root=C*C-2.*C+Q*Q;            // Radikand (Wurzelausdruck)
9      if (root<0) {                 // fuehrt zu nichtreellem Ergebnis
10         root=0;
11     }
12     if (Q<0) {                    // Fallunterscheidung fuer Vorzeichen
13         pol=(int16_t)((atan((Q+sqrt(root))/C))*10428);
14     } else {
15         pol=(int16_t)((atan((Q-sqrt(root))/C))*10428);
16     }
17 }
```

Analog zu Kap. 3.2.3 wird die Polarisation aus den beiden zuvor ermittelten Intensitätswerten berechnet (Alg. 7.4).

Dabei wird die Polarisation in einer int16-Variable (pol) gespeichert, die ursprünglich zwischen $\pm\pi$ und nach der Multiplikation mit 10.428 zwischen $\pm$32.761 liegt (32.768 stellt die darstellbare Grenze von int16 dar). Dadurch kann bei möglichst hoher Auflösung (ca. 20 Winkelsekunden oder 0,0055°) das Ergebnis mit nur je 2 Byte für das Senden und Speichern dargestellt werden.

Anstatt des theoretisch maximal möglichem Multiplikators 10.430 wurde 10.428 gewählt, da damit die Rückrechnung sowohl in Radiant als auch in Grad durch Integerdivisionen umgesetzt werden kann. Um aus dem gespeicherten int16-Werten wieder auf den Winkel in Radiant zu schließen, ist die Integerdivision mit 10.428 nötig, für den Winkel in Grad kann durch 182 dividiert werden. Die Abweichung von der exakten Umrechnung $10.428 * \pi/180 = 182{,}003$ beträgt dabei nur 0,0016% und liegt damit unter der Auflösung).

7.6 Programmablaufpläne wichtiger Funktionen des Messprogramms

Die Programmablaufpläne auf den folgenden Seiten geben einen Überblick über einige Funktionen und ihren Ablauf. Zur genaueren Betrachtung kann der Sourcecode herangezogen werden. Abb. 7.13 zeigt eine Übersicht aller eigenen Funktionen des Messprogramms und wie diese zusammenhängen, die farbig gekennzeichneten sind dabei während der Aufnahme ständig in Verwendung und daher stärker optimiert, die restlichen Funktionen werden nur selten während der Aufnahme aufgerufen, sondern häufig davor und danach.

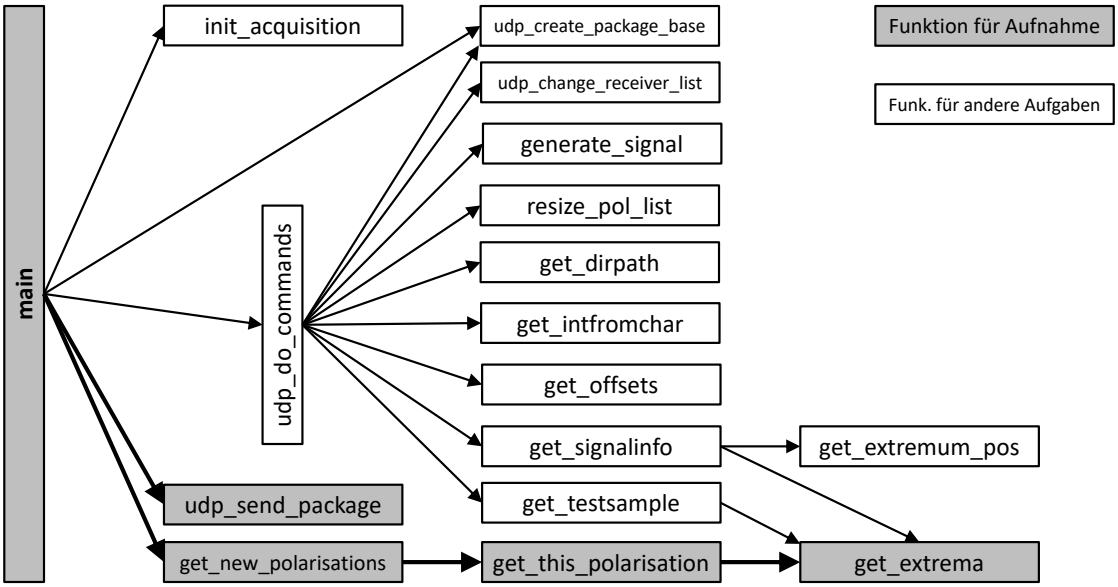


Abbildung 7.13: Funktionen des STEMlab-Messprogramms

7.6.1 Hauptfunktion (main)

Die Hauptfunktion führt nach dem Programmstart alle nötigen Vorbereitungen aus und läuft anschließend bis zum Beenden in der Hauptschleife. Abhängig von erhaltenen Befehlen vom Computer oder auftretenden Ereignissen werden verschiedene Funktionen ausgeführt und die Aufnahme gestartet. Die Auswahl der Tätigkeit in der Hauptschleife erfolgt dabei durch die Variable prog_status. Abb. 7.14 und Abb. 7.15 zeigen den grundlegenden Aufbau des Hauptprogramms und der darin verwendeten Funktionen.

7.6.2 UDP-Befehle ausführen (udp_do_commands)

Wenn neue Befehle oder Informationen vom Messclient empfangen werden, werden diese an die Funktion udp_do_commands übergeben. Sie trennt diese in einer Schleife auf und übernimmt die Einstellungen oder ruft die Funktionen zum Ermitteln der gewünschten Informationen auf. Abb. 7.16 stellt den Ablauf nur kurz dar und geht nicht auf die detaillierte Umsetzung einzelner Befehle ein.

7.6.3 Messung starten (init_acquisition)

Damit für die Aufnahmefunktionen alle benötigten Informationen verfügbar sind, müssen einige Variablen richtig festgelegt werden (Abb. 7.17). Im Wesentlichen sind die aktuellen Indizes im Puffer zu bestimmen, ab denen die Daten verarbeitet werden sollen, der erste Trigger ist festzustellen und der Status bei Erreichen der Startbedingung von prog_status=3 auf prog_status=1 zu ändern, sodass ab dann die Aufnahmefunktion in der Programm-Hauptschleife aufgerufen wird.

Der Aufnahmestart kann von der Funktion noch verzögert werden, wenn vom Computer-Clientprogramm die Einstellung zum Warten auf einen bestimmten Mindestwert des Intensitätssignals übermittelt wurde. In diesem Fall wird der Mittelwert des Signals in jedem Funktionsaufruf berechnet und entweder gegenüber einem vorgegeben Mindestwert verglichen oder auf einen genügend schnellen Anstieg überprüft. Abhängig davon wird die Aufzeichnung gestartet oder die Prüfung im nächsten Durchlauf wiederholt.

7.6.4 Neue Samples auswerten (get_new_polarisations)

Während der Aufnahme wird diese Funktion regelmäßig vom Hauptprogramm aufgerufen, sie verarbeitet die neuen Samples in den Puffern und gibt eine Liste mit neuen Polarisationswerten zurück.

Es werden die Nulldurchgänge im SCPEM-Signal (Trigger) bestimmt und die Messwerte ermittelt, dieser Vorgang wird bis zum Erreichen des Endes der neuen Daten im Puffer in einer Schleife wiederholt. Abhängig davon, ob Messwerte übersprungen werden sollen, ob aus den übersprungenen Polarisationswerten der Mittelwert gebildet werden soll und ob je SCPEM-Periode ein oder zwei Polarisationen berechnet werden sollen, wird die Polarisationsberechnungsfunktion aufgerufen und die davon erhaltenen Werte zu einer Liste verarbeitet. Abb. 7.18 und Abb. 7.19 zeigen diese Schritte.

7.6.5 Polarisation berechnen (get_this_polarisation)

In Abb. 7.20 wird eine Polarisationsberechnung, wie in Kap. 7.5 gezeigt und theoretisch in Kap. 3.2.3 behandelt, durchgeführt. Die Intensitätswerte werden über die Funktion get_extrema ermittelt und daraus mit bekannter max. Retardation und Analysatorwinkel die Polarisation berechnet.

Direkt im Anschluss ist die Möglichkeit der Kalibrierung gegeben, wenn dafür vor Aufnahmebeginn eine Kalibriertabelle vom Computer-Clientprogramm übermittelt wurde.

7.6.6 Intensitäten ermitteln (get_extrema)

Kap. 7.3 beschreibt die Ermittlung der gesuchten Intensitäten aus dem Messsignal. Der Triggerindex und die ungefähren relativen Positionen der Extremwerte dazu werden übergeben und damit mit Samples aus dem Signalpuffer zwei Extremwerte rekonstruiert.

Die in Kap. 7.3.5 beschriebene Implementierung ist in der Funktion get_extrema umgesetzt und in Abb. 7.21 dargestellt.

7.6.7 UDP-Paket senden (udp_send_package)

Diese Funktion (Abb. 7.22) erstellt aus den Messdaten und Informationen ein Paket, wie es das dafür entwickelte Protokoll (Tab. 6.6) vorgibt, und sendet dieses dann über den UDP-Socket an die gewünschten Adressen.

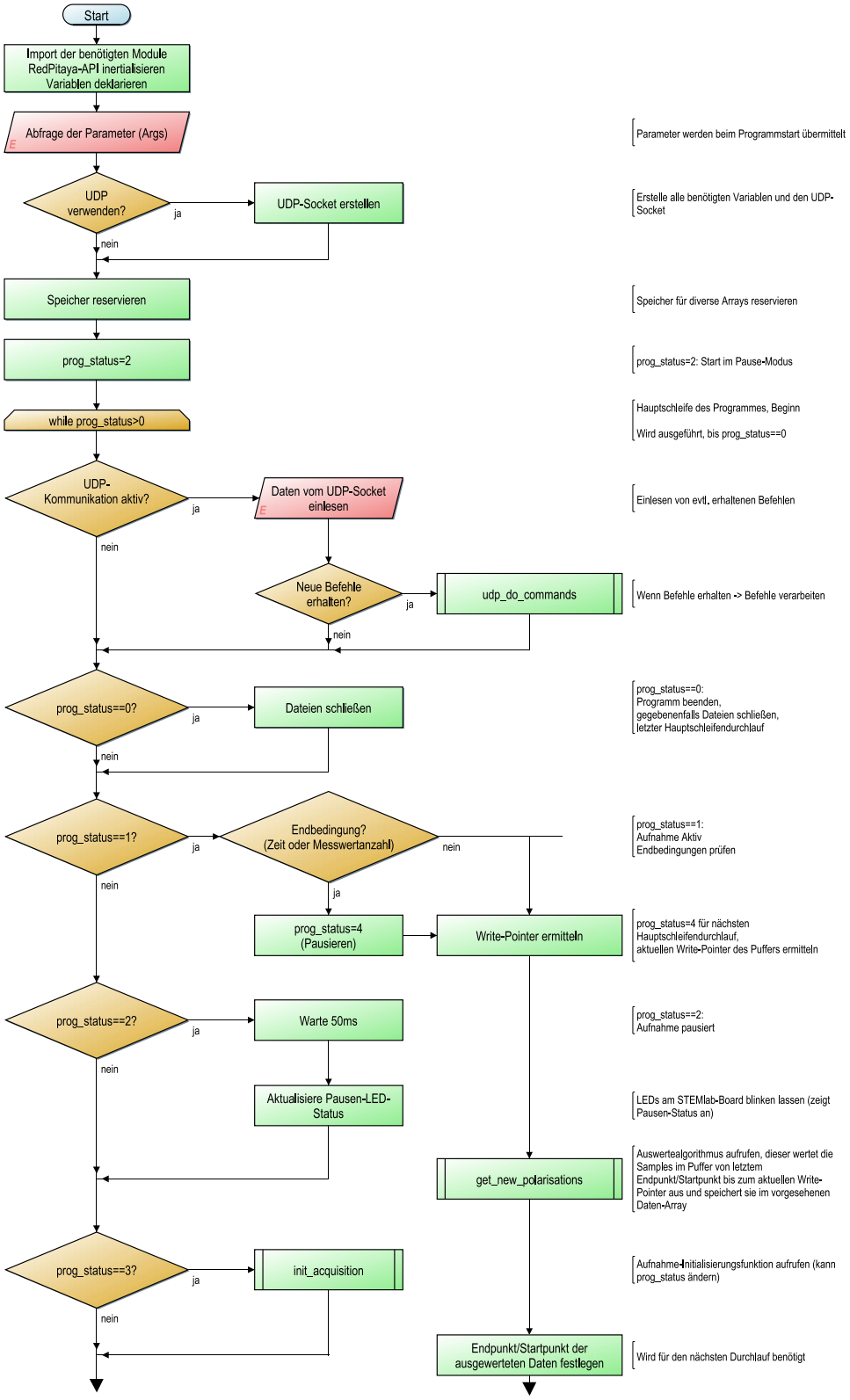


Abbildung 7.14: SCPEM-Hauptprogramm (main) Teil 1/2 (Anh. B.3.4, Anh. C.1.1)

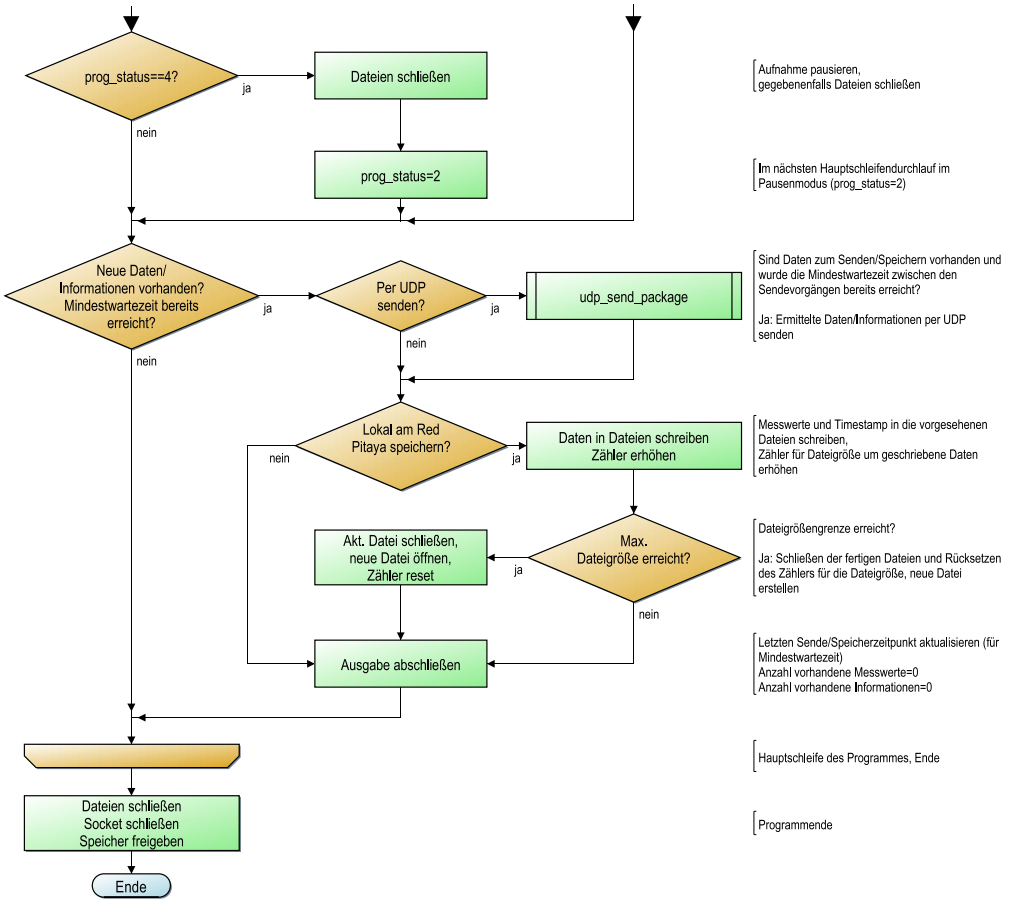


Abbildung 7.15: SCPEM-Hauptprogramm (main) Teil 2/2 (Anh. B.3.4, Anh. C.1.1)

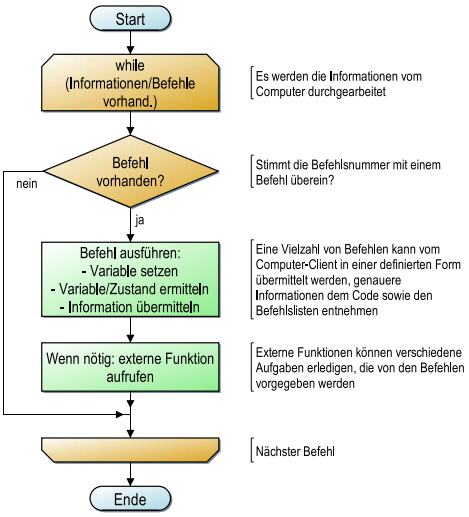


Abbildung 7.16: UDP-Befehle ausführen (udp_do_commands) (Anh. B.3.4, Anh. C.1.3)

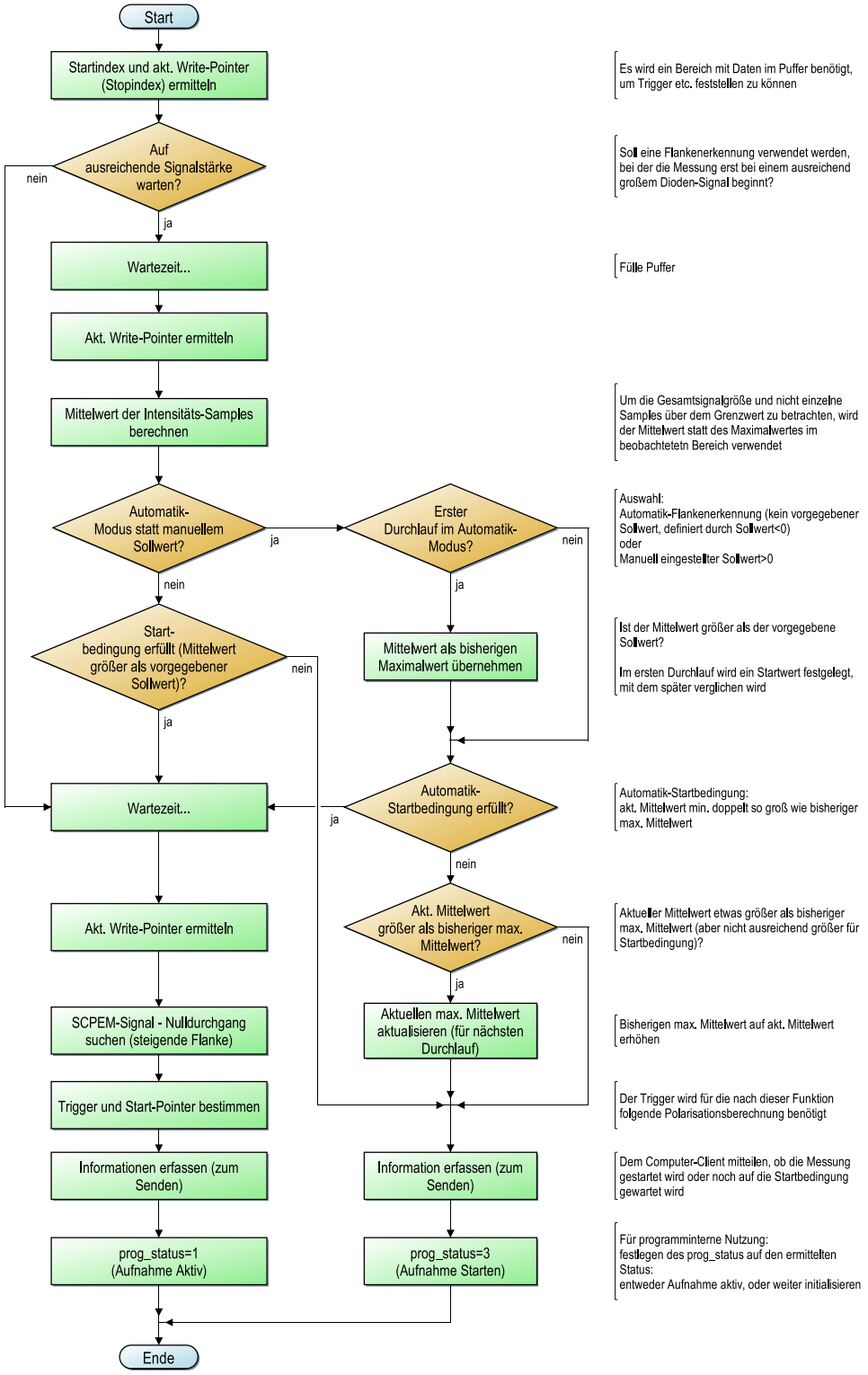


Abbildung 7.17: Messung starten (init_acquisition) (Anh. B.3.4, Anh. C.1.2)

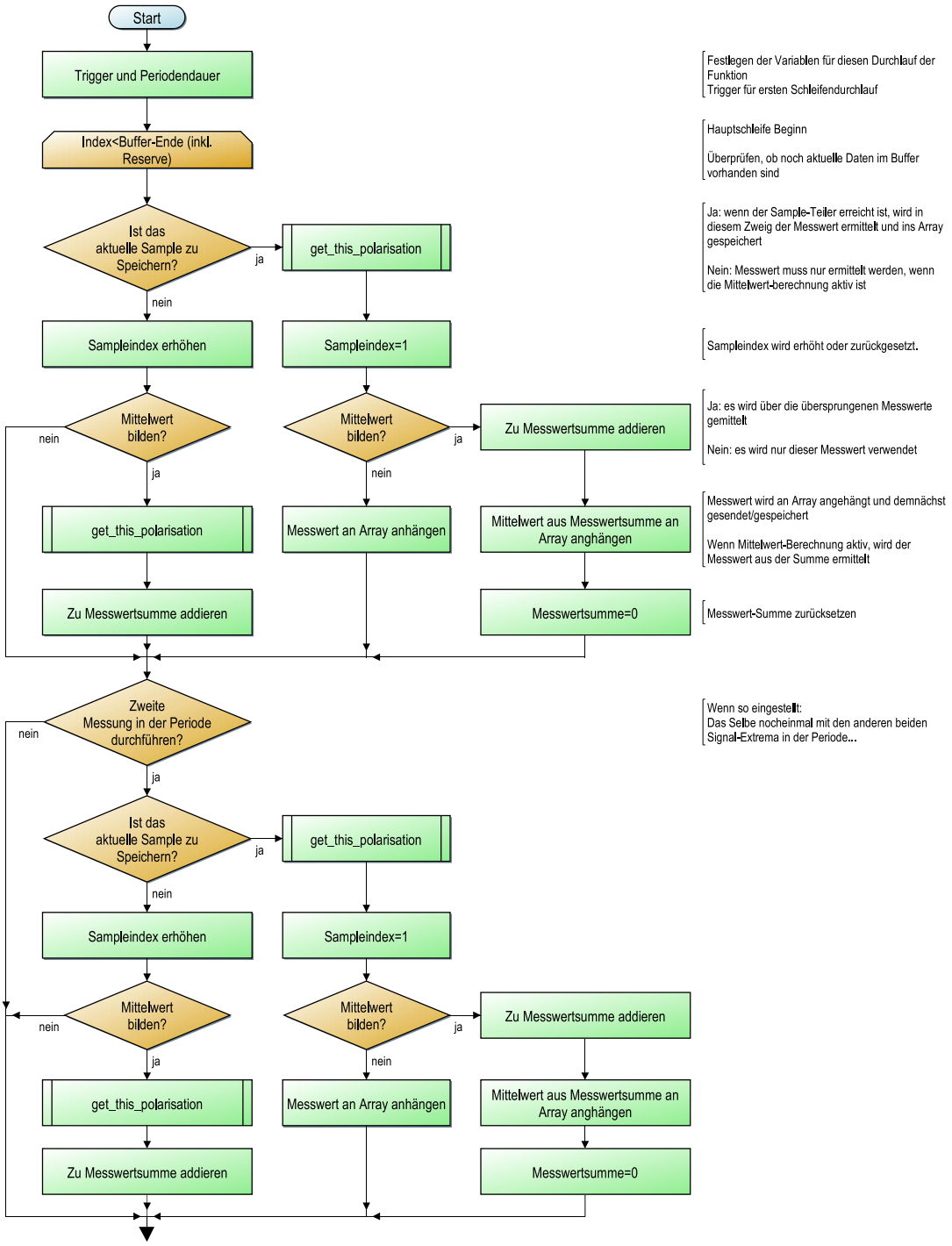


Abbildung 7.18: Neue Samples auswerten (get_new_polarisations) Teil 1/2 (Anh. B.3.4, Anh. C.1.2)

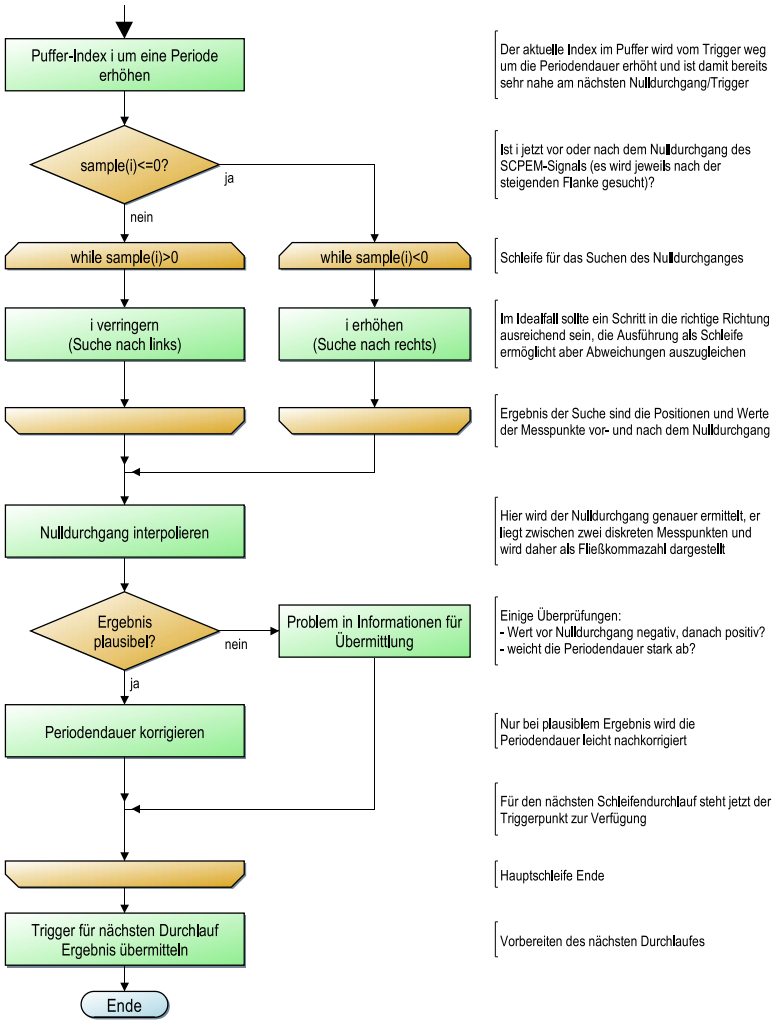


Abbildung 7.19: Neue Samples auswerten (get_new_polarisations) Teil 2/2 (Anh. B.3.4, Anh. C.1.2)

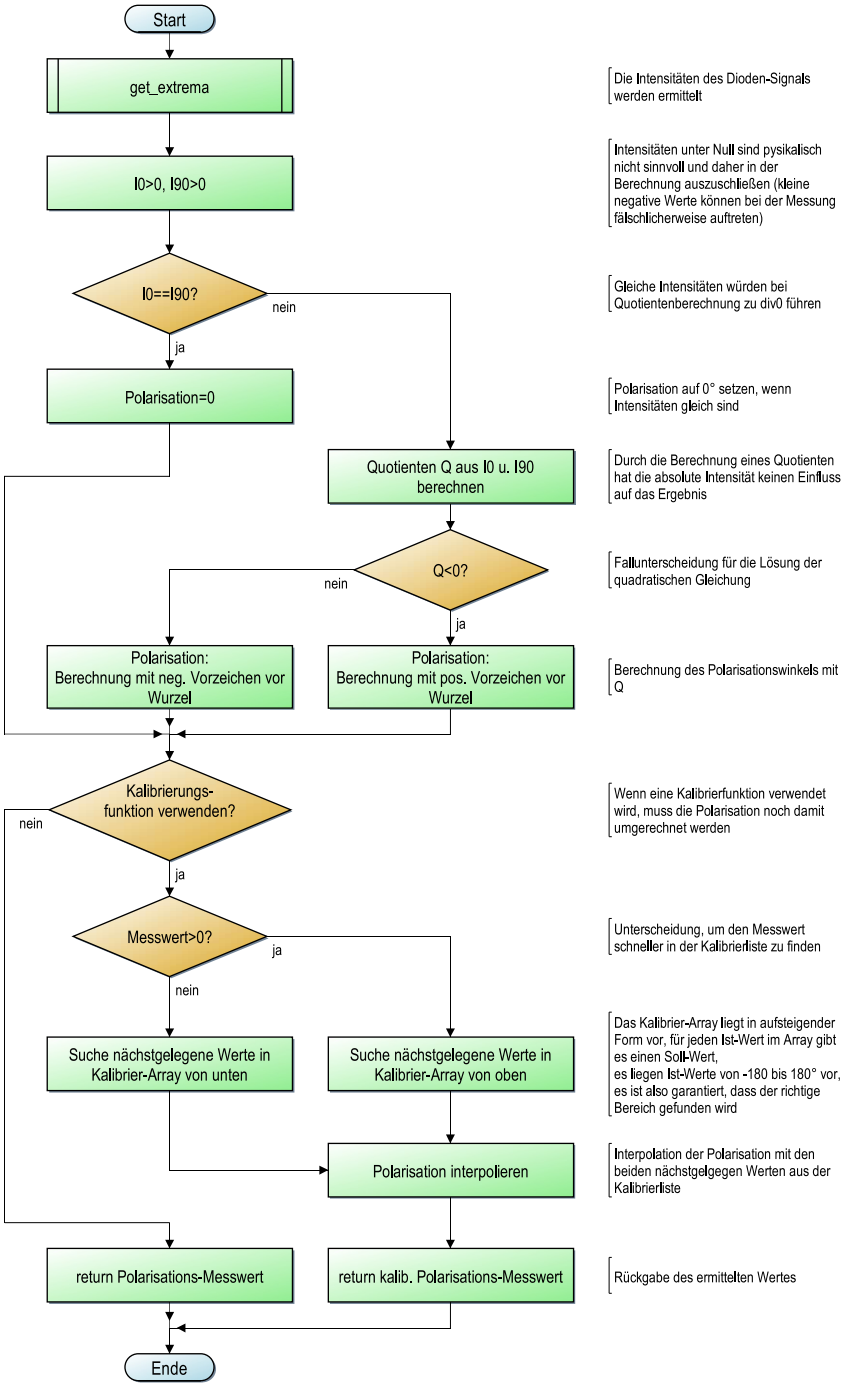


Abbildung 7.20: Polarisation berechnen (get_this_polarisation) (Anh. B.3.4, Anh. C.1.2)

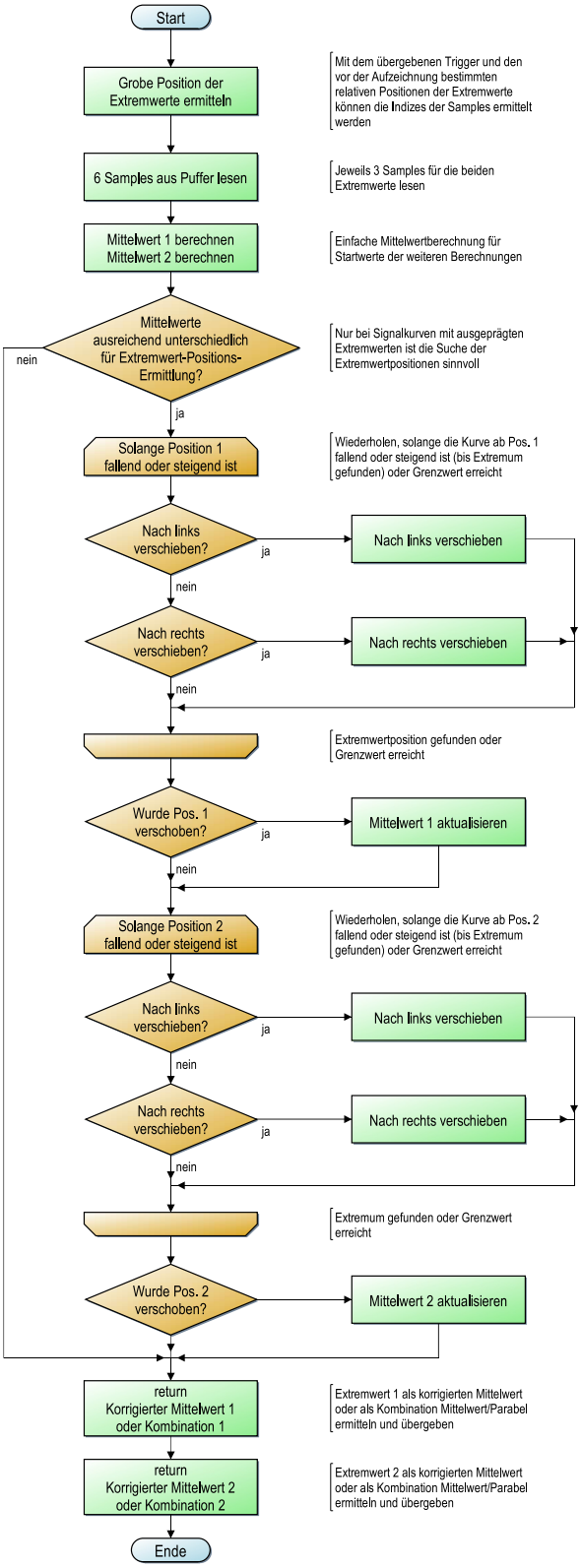


Abbildung 7.21: Intensitäten ermitteln (get_extrema) (Anh. B.3.4, Anh. C.1.2)

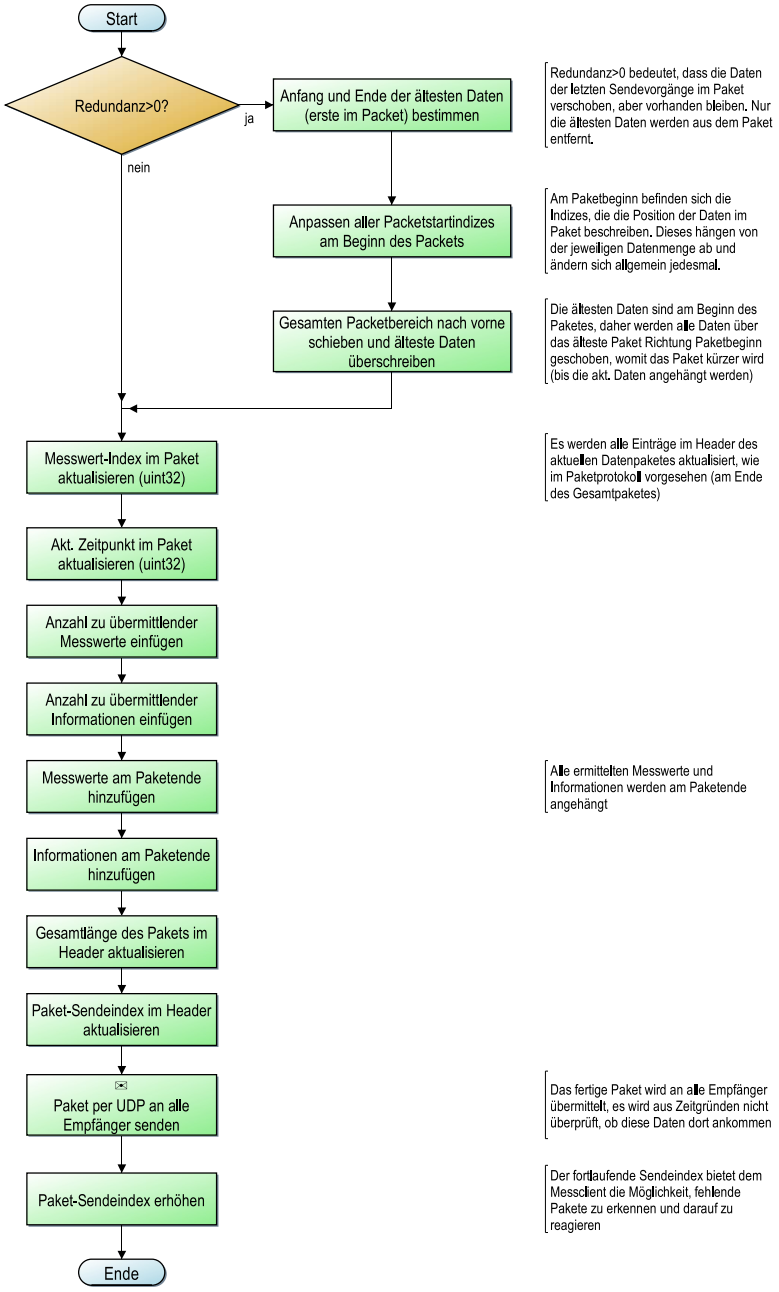


Abbildung 7.22: UDP-Paket erstellen und senden (udp_send_package) (Anh. B.3.4, Anh. C.1.3)

8 Messclient-Programm für den Computer

Das Messclient-Programm soll folgende Aufgaben erfüllen:

- Ausreichend schnelle Schnittstelle zum Messprogramm zur Verfügung stellen
- Verarbeitung der ankommenden Daten
- Grafische Oberfläche, die einfachen Zugang zu zahlreichen Einstellungsmöglichkeiten bietet
- Möglichkeit zur Betrachtung und Analyse der erhaltenen Daten

8.1 Versuche: Messclient mit MATLAB

Am Weg zum Python-Messclient sind einige Entwicklungen bis zu einem teilweise funktionsfähigen Stand gebracht worden, jedoch an einer oder mehreren Hürden gescheitert, die keine sinnvolle Weiterführung ermöglichten. Viele Ideen und Erfahrungen können in der aktuellen Version verwendet werden.

8.1.1 MATLAB mit Figures

Nach ersten erfolgreichen Tests zum Programmstart und weiterer Kommunikation mit dem Messprogramm mit MATLAB wird darauf aufbauend eine einfache Oberfläche auf Basis eines MATLAB-Figure-Fensters geschaffen. Die Möglichkeit der Platzierung grafischer Objekte ermöglicht in erster Linie die Ausgabe von Informationen und Messwerten, erlaubt aber zusätzlich bereits einfache Interaktionen (einige Einstellungen, Aufnahmestart/-stopp, ...).

Es ist bereits möglich, Messergebnisse während der Aufnahme anzuzeigen und zu speichern, die Datenübertragung erfolgt erst noch mit dem SSH File Transfer Protocol (SFTP), später bereits testweise mit UDP.

8.1.2 MATLAB mit App-Designer

Mit dem App-Designer (seit MATLAB-Version R2016a) können komplexere grafische Benutzeroberflächen erstellt werden. Nach längerer Entwicklung und Test verschiedener UDP-Module muss aufgrund der nicht ausreichenden Geschwindigkeit der Kommunikation auf eine andere Programmierumgebung gesetzt werden. Beim Vergleich zur Alternative Python (Tab. 6.4) erkennt man die große Differenz, die nicht einfach durch Optimierungen überwunden werden kann.

Die folgenden Gründe führen zur Entscheidung, die Programmierung des Messclients völlig neu mit Python zu beginnen:

- Unzureichende Geschwindigkeit des UDP-Sockets
- Lange Startzeiten des Messclients (im Bereich 30 s), deutlich größerer Ressourcenverbrauch beim Programmieren und Ausführen (die Programmierumgebung des App-Designers reagiert bei vielen Codezeilen auf der verwendeten Hardware unbrauchbar langsam)
- MATLAB-Installation und -Lizenz am Client-Rechner für die spätere Verwendung des Messprogramms nötig

Einige Teile dieses Programms kommen im Laufe der Entwicklung weiterhin zum Einsatz, werden jedoch am Ende nicht weiter benötigt.

Die in Abb. 8.1 und Abb. 8.2 dargestellten Tabs des MATLAB-Messclients sind im Anschluss kurz beschrieben.

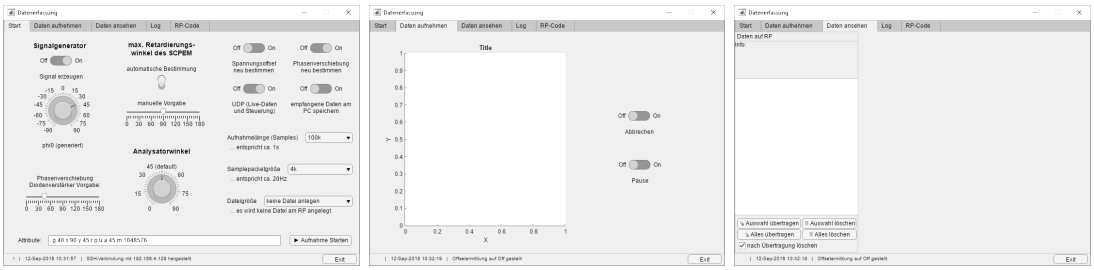


Abbildung 8.1: MATLAB-GUI-Tabs: Start, Aufnahme, Analyse

8.1.2.1 Start-Tab

Der Tab „Start“ beinhaltet Einstellungen für die Messung, diese werden beim Programmstart am STEMlab-Board (über SSH2) als Argumente übermittelt. Die Funktionen sind nicht mehr mit der aktuellen Version des Messprogramms kompatibel, da die Einstellungen aktuell per UDP jederzeit geändert werden können.

Viele dieser Einstellungen finden sich im Python-Programm wieder (Kap. 8.2, Kap. 9.2.2).

8.1.2.2 Aufnahme- und Datenanalyse-Tab

Die Anzeige von aktuellen Informationen während der Aufnahme sowie die Analysemöglichkeit der Daten ist nur ansatzweise umgesetzt und nicht fertiggestellt, da die nötige Übertragungsgeschwindigkeit nicht erreicht werden konnte.

8.1.2.3 Log-Tab

Eine Zusammenfassung aller Ereignisse bietet der Log-Tab. Während hier eine Sortierung nach der Herkunft (entspricht den verschiedenen Tabs) und Priorität („Alle“, „Wichtig“ oder „Fehler“) möglich ist, wird diese Übersicht im Python-Programm nur vereinfacht umgesetzt.

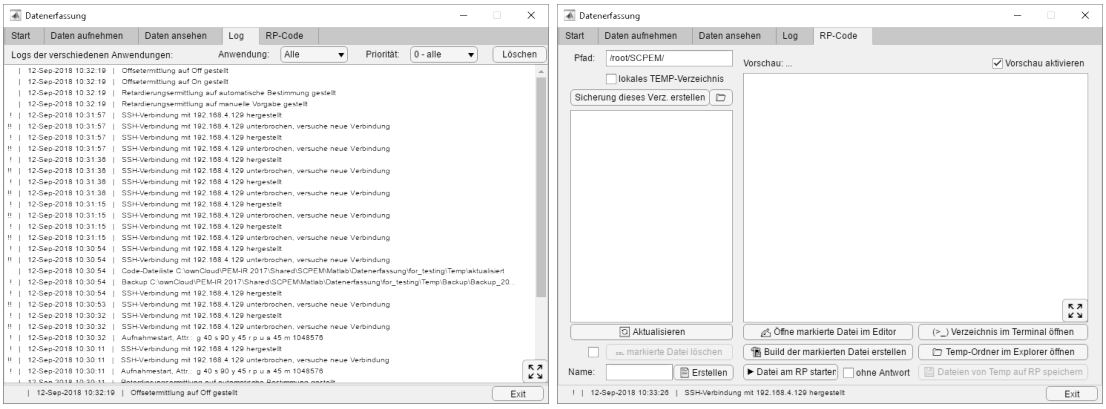


Abbildung 8.2: MATLAB-GUI-Tabs: Start, Aufnahme, Analyse

8.1.2.4 Code-Tab

Der Tab „RP-Code“ ist vollständig verwendbar und weiterhin kompatibel. Er ermöglicht die Erstellung von Backups der Sourcecode-Daten des STEMlab-Boards am Computer, die automatische Synchronisation der Daten bei Änderungen am Code und das direkte Öffnen, Kompilieren und Ausführen einer ausgewählten Datei am Messboard.

Im aktuellen Python-Messprogramm ist die Möglichkeit gegeben, das Programm neu zu kompilieren. Für weitere Funktionen werden PuTTY bzw. WinSCP verwendet.

8.2 Messclient mit Python

Zu Beginn wird die Grundfunktion des Empfangens und Speicherns der Daten umgesetzt, dafür stehen bereits viele Informationen aus den Versuchen mit MATLAB zur Verfügung. Nachdem diese Funktionen stabil und schnell genug ausgeführt werden und damit funktionsfähig sind, wird der Fokus auf die Implementierung der weiteren benötigten Funktionen und die Erstellung der Nutzeroberfläche gelegt.

Von besonderem Interesse ist die Entwicklung der Nutzeroberfläche mit sinnvollen Einstellungsmöglichkeiten, Informationen und Funktionen sowie die Aufteilung des Programms in mehrere Threads, um während dem Empfangens und Verarbeiten der Daten die Bedienbarkeit zu gewährleisten.

8.2.1 Entwicklungsumgebung für das Graphical User Interface (GUI)

Python unterstützt eine große Menge verschiedener GUI-Umgebungen. [30, 31]

Es werden keine besonderen Anforderungen an die Benutzeroberfläche gesetzt, aus den unterschiedlichen Möglichkeiten wird TkInter gewählt.

8.2.2 TkInter

TkInter ist eine Benutzeroberflächen-Bibliothek von Python und ist standardmäßig in der Python-Installation inkludiert. Das vereinfacht die Verwendung des Programms auf unterschiedlichen Geräten und verhindert die Abhängigkeit von kostenpflichtigen Lizenzen.

Zu TkInter existieren eine Vielzahl an Editoren, die die Erstellung eines GUI erleichtern. Getestet werden die Editoren Page [32] von Don Rozenberg, Pygubu [33] von Alejandro Autalan und Visual Python/Visual TkInter [34].

Von diesen wird Pygubu gewählt.

8.2.2.1 Pygubu

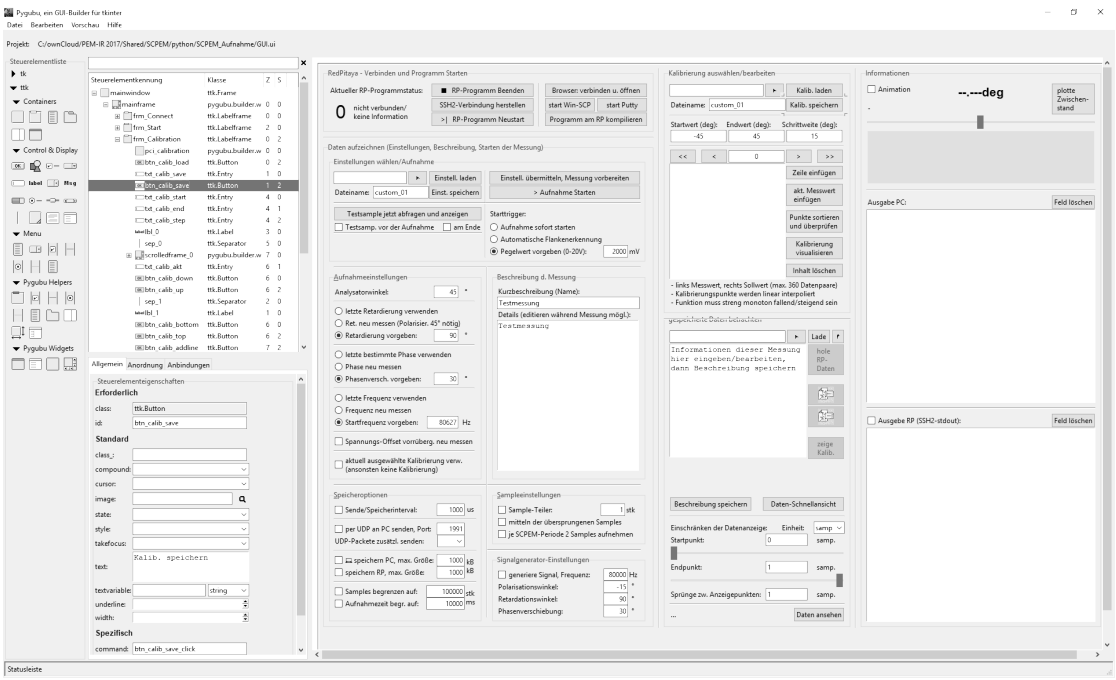


Abbildung 8.3: Erstellung der Nutzeroberfläche mit Pygubu

Pygubu ist ab Python 2.7 verfügbar, es handelt sich um einen Editor mit grafischer Oberfläche, der die generierte Benutzeroberfläche als XML-Datei (Extensible-Markup-Language) speichert. Beim Ausführen des Python-Programms wird diese Datei vom Pygubu-Builder interpretiert und die TkInter-Objekte selbstständig wie gewünscht erzeugt.

Das Programm bietet eine Reihe von Elementen an, die nach deren Platzierung und Anpassung der Eigenschaften in einer Vorschau der Oberfläche dargestellt werden. Die Kommunikation mit dem Python-Programm erfolgt einerseits über Command-Funktionen, die von Steuerelementen bei verschiedenen Aktionen aufgerufen werden können. Viele

Elemente bieten die Möglichkeit der Verbindung mit einer Variable, die den schnellen Zugriff auf den angezeigten Text oder einen spezifischen Wert des Elements zulassen. [33]

Abb. 8.3 zeigt Pygubu mit der geöffneten Benutzeroberfläche des Client-Programms. Links die verfügbaren Steuerelemente, daneben die verschachtelte Liste aller aktuell verwendeten Elemente und darunter die Einstellungen des aktuell ausgewählten Elements. Rechts ist die aktuelle Vorschau der erstellten Oberfläche (Anh. C.2.3) zu sehen.

8.2.3 Multi-Threading/Multi-Processing

Das Programm in einem einzelnen Thread auszuführen ist nicht zielführend, da die Benutzeroberfläche durch das ununterbrochene Ausführen der UDP-Empfangs-Funktion während der Aufnahme unbenutzbar ist. Es werden entweder mehrere Threads oder Subprozesse erstellt, die gleichzeitig und unabhängig voneinander ausgeführt werden und dabei miteinander kommunizieren können.

8.2.3.1 Subprozess oder Thread

Zuerst wird die Umsetzung mit Subprozessen versucht, wobei die Kommunikation dabei über Pipes läuft. Hier besteht das Problem, dass diese „Blocking“ sind, das bedeutet, wenn davon gelesen wird, bleibt das Programm an diesem Punkt stehen und wartet auf Daten. Das kann zu Datenverlust oder zum Absturz des TkInter-GUI führen.

Threads dagegen verwenden Queues, welche eine „nowait“-Abfrage ermöglichen. Sind keine Daten in der Queue vorhanden, wird sofort ein leeres Array zurückgegeben und das Programm fortgesetzt. Damit sind sie besser geeignet als Subprozesse.

8.2.3.2 Threads und TkInter

Da die Threads in einem Memory Space arbeiten, ist aus allen ein Zugriff auf TkInter-Variablen möglich. Das stellt sich jedoch als problematisch heraus.

Nachdem das Programm in verschiedene Threads für unterschiedliche Funktionen aufgeteilt und die Kommunikation untereinander definiert ist, funktioniert die gleichzeitige Aufzeichnung und Darstellung der Messdaten. Es kommt es jeweils nach wenigen Sekunden zum Einfrieren der Benutzeroberfläche, die dann keine Informationen mehr anzeigt nicht bedient werden kann. Als Grund dafür stellt sich heraus, dass verschiedene Threads auf TkInter-Variablen zugreifen und Änderungen vornehmen, TkInter jedoch nicht threading-fähig ist.

Das bedeutet, es kann nur im Hauptthread damit gearbeitet werden, und um in der Queue ankommende Daten ständig darstellen zu können, muss eine Funktion regelmäßig ausgeführt werden. Das ermöglicht die Timer-Funktion „after“ von TkInter, welche nach einer bestimmten Zeit eine Funktion aufrufen kann. Dieser Aufruf erfolgt im selben Thread, in der Zwischenzeit wird das Programm jedoch, anders als beispielsweise bei der Funktion „sleep“, nicht angehalten.

8.2.3.3 Aufteilung und Aufgaben der Threads

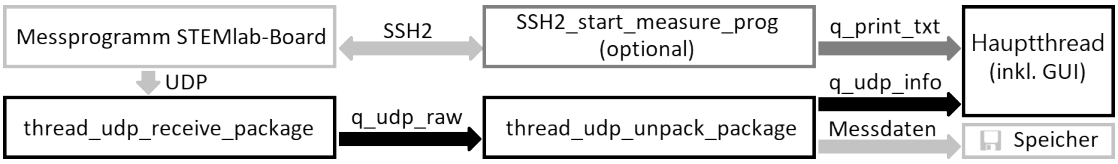


Abbildung 8.4: Threads und Queues im Messclient

Insgesamt werden 4 Threads eingesetzt, die über Queues kommunizieren. Diese sind in Abb. 8.4 dargestellt.

- main: Im Hauptthread laufen alle Funktionen, die keine längere Laufzeit haben und/oder Zugriff auf die Benutzeroberfläche benötigen.
- SSH2_start_measure_prog: Diese Funktion wird nur als Thread gestartet, wenn die SSH2-Verbindung für die Übermittlung des StdOut des Messprogramms genutzt werden soll. In diesem Fall werden die erhaltenen Informationen beim Ausführen des Messprogramms per Queue an den Hauptthread übergeben und dort angezeigt. Ansonsten ist die Funktion bereits direkt nach dem Messprogrammstart abgeschlossen und muss daher nicht als eigener Thread ausgeführt werden.
- thread_udp_receive_package: Dieser Thread wird beim Start des Messprogramms gestartet und schreibt die Rohdaten ständig die vom Messprogramm am UDP-Socket erhaltenen Daten in eine Queue q_udp_raw ein. Diese arbeitet als zusätzlicher Puffer, falls es beim Entpacken und Speichern der Daten zu Verzögerungen kommt.
- thread_udp_unpack_package: Der Thread liest die Daten aus der Queue und verarbeitet sie dem Protokoll (Tab. 6.6) entsprechend. Die Daten werden gespeichert und ausgewählte Informationen über die Queue q_udp_info an den Hauptthread zur Anzeige übermittelt. Er wird ebenso mit dem Messprogramm gestartet.

8.2.4 Funktionen des Messclients

Um eine Übersicht über die vielen Funktionen des Messclient-Programms zu erhalten, sind diese hier nach Aufgabenbereich getrennt aufgelistet und kurz beschrieben. Die Funktionen selbst sind in Anh. C.2 zu finden.

[GUI] bedeutet dabei, dass die Funktion von einem TkInter-Element und damit vom Nutzer jederzeit aufgerufen werden kann. Eine Beschreibung der Bedienung dieser Funktionen aus Sicht des Anwenders ist in Kap. 9.2 gegeben.

8.2.4.1 Externe Programme und Informationen

Einige Funktionen dienen zum Öffnen von externen Programmen, zum Anzeigen und Ändern von verschiedenen Informationen sowie zum Abruf eines Testsamples (Abb. 7.5).

```
# Externe Programme oeffnen:

def btn_connect_openbrowser_click(self):
#   [GUI] Browser-Link zu STEMlab-Startseite oeffnen
def btn_connect_openwinscp_click(self):
#   [GUI] WinSCP extern oeffnen
def btn_connect_openputty(self):
#   [GUI] PuTTY extern oeffnen
def btn_connect_compile(self):
#   [GUI] Programm am STEMlab-Board neu kompilieren


# Informationsanzeigen:

def btn_del_info_PC_click(self):
#   [GUI] info_PC-Feld leeren
def btn_del_info_RP_click(self):
#   [GUI] info_RP-Feld leeren
def btn_plot_current_samples_click(self):
#   [GUI] Uebersicht ermittelte Samples plotten (waehrend Aufnahme)
def gui_writer_task(self, run_again=False):
#   Task zum Hinzufuegen/Aendern von Text (noetig wegen Nicht-Threading-Faehigkeit
     von TKinter)
def enable_disable_input(self):
#   Abhaengig vom Status werden verschiedene GUI-Elemente inaktiv geschalten
def set_acquireinformation(self):
#   Information ueber Aufnahme anpassen


# Testsample erstellen:

def btn_get_testsample_click(self):
#   Testsample per UDP anfordern
def testsample_show(self, s_testsample, save_show):
#   erhaltenes Testsample auswerten (Diagramm erstellen, anzeigen, speichern)
```

8.2.4.2 Programmstatus und Verbindungen

Der aktuelle Status des Programms wird mit der Variable prog_status festgelegt. Diese hat für die Werte 0-8 folgende Bedeutung:

0: Keine Verbindung vorhanden/Status unbekannt

1: SSH2-Verbindung wird hergestellt

2: SSH2-Verbindung zum STEMlab-Board aktiv

3: Starte Messprogramm am STEMlab-Board per SSH2

4: Messprogramm gestartet; Noch keine Einstellungen übermittelt

5: Alle aktuellen Einstellungen werden per UDP übermittelt

6: Alle Einstellungen und Informationen vorhanden; Bereit für Aufnahme

7: Aufnahme wird gestartet/abgeschlossen

8: Aufnahme läuft

Der Status wird nach erfolgreichem Abschluss einer Änderung (Status ungerade: Änderung wird durchgeführt) erhöht und bleibt dann in diesem Zustand (Status gerade: Zustand erreicht/aktiv), ein Fehler oder Timeout während der Ausführung führt zum Reset auf den Programm-Status 0.

```
# Commands des GUI zur Statusaenderung (Verbinden, Programmstart, ...):

def btn_connect_SSH2_click(self):
#   [GUI] SSH-Verbindung herstellen, Optionen updaten -> change_prog_status(1)
def btn_connect_startRP_click(self):
#   [GUI] Programm am STEMlab-Board starten           -> change_prog_status(3)
def btn_connect_stopRP_click(self):
#   [GUI] Programm am STEMlab-Board beenden           -> change_prog_status(0)
def btn_sendoptionsRP_click(self):
#   [GUI] Einstellungen an Messprogramm senden        -> change_prog_status(5)
def btn_Start_Pause_click(self):
#   [GUI] Aufnahme Starten/Pausieren                  -> change_prog_status(7)


# Haupt-Statusänderungsfunktion (prueft geforderte Aenderung des Status und fuehr
    dementsprechend Funktionen aus):

def change_prog_status(self,new_RP_status):
#   Aenderung Programmstatus durchfuehren


# Verbindungsfunktionen (SSH2 und UDP-Senden):

def SSH2_connect_RP(self):
#   SSH2-Verbindung herstellen
def SSH2_start_measure_prog(self, read_stdout):
#   Programm am STEMlab-Board starten, read_stdout gibt an, ob Programmausgabe
    übertragen und angezeigt wird  (wird bei read_stdout==True als Thread
    gestartet)
def sendtoRP(self,msg):
#   Daten aus msg per UDP an Messprogramm senden
```

8.2.4.3 Messeinstellungen und Aufnahmestart/-stopp

Beim Programmstart werden die Einstellungen, die von der GUI dargestellt werden sollen, vom Speicher geladen. Zusätzlich werden die Einstellungen bei jeder Aufnahme gespeichert und sind somit für spätere Analysen verfügbar.

Für jede Einstellung steht eine Funktion zur Verfügung, die eingegebene Werte überprüft und weiter Aktionen ausführt. Werden Einstellungen geändert, werden diese bei laufendem Messprogramm direkt per UDP gesendet. Dabei sind die ersten beiden Bytes jeweils ein Code für die Einstellung, danach folgen abhängig von der Einstellung die zu übermittelnden Werte, die vom Messprogramm abhängig vom Einstellungscode interpretiert werden. Es können mehrere Einstellungen direkt hintereinander in einem Paket übermittelt werden, das Messprogramm kann dieses dann wieder zerlegen.

Bei einigen Einstellungen ist der UDP-Paketaufbau erläutert, 251 bedeutet dabei eine Einstellungsänderung, die folgende Zahl die zugewiesene Einstellungsnummer. Diese sind auch im Messprogramm in der UDP-Empfangsfunktion Kap. 7.6.2 zu finden und können mit weiteren Nummern erweitert werden.

```
# Einstellungen laden und speichern:

def btn_options_save_click(self):
#   [GUI] Einstellungen an gewuenschtem Pfad speichern
def options_update_from_json(self):
#   Einstellungen an geladene json-Einstellungen anpassen
def options_update_from_GUI(self):
#   Einstellungen auf Gueltigkeit pruefen und in json-Einstellungen aktualisieren
def udp_send_alloptions(self):
#   alle Einstellungen auf Gueltigkeit pruefen und per UDP an Messprogramm senden


# Einzelne Optionen: von GUI abfragen, auf Gueltigkeit pruefen, json-Einstellung
    anpassen; mode=True: Einstellung direkt an RP senden, mode=False: Einstellung
    return (fuer zusammenhaengende Einstellungskette):

def changed_option_startvalue(self):
#   [GUI] Startoption (wird beim Aufnahmestart gesendet)
def changed_option_testsample(self):
#   [GUI] Testsampleoption (wird beim Aufnahmestart und Ende gesendet)
def changed_option_ret_phase_freq(self, retardation, phase, freq):
#   [GUI] UDP: [251 24 uint16(mode)]
def changed_option_retardation(self, mode=True):
#   [GUI] UDP: [251 21 int16(retvalue)]
def changed_option_phase(self, mode=True):
#   [GUI] UDP: [251 22 int16(phasevalue)]
def changed_option_freq(self, mode=True):
#   [GUI] UDP: [251 23 int32(freqvalue)]
def changed_option_analysator(self, mode=True):
#   [GUI] UDP: [251 20 int16(analysator)]
def changed_option_offset(self, mode=True):
#   [GUI] UDP: [251 25 0/1 0]
def changed_option_calibration(self, mode=True):
#   [GUI] UDP: [251 26 uint16(calib)]
def changed_option_send_save_interval(self, mode=True):
#   [GUI] UDP: [251 30 uint16(0)/int16(RPintervalvalue)]
def changed_option_udp(self, mode=True):
#   [GUI] UDP: [251 31/32 uint16(udpport)]
def changed_option_savePC(self, mode=True):
#   [GUI] UDP: [251 34 uint32(0)/uint32(savePCsize)]
def changed_option_saveRP(self, mode=True):
#   [GUI] UDP: [251 35 uint32(0)/uint32(saveRPsize)]
def changed_option_duration(self, mode=True):
#   [GUI] UDP: [251 36 uint32(0)/uint32(maxsample) 251 37 int32(0)/int32(maxtime)]
def changed_option_sample(self, mode=True):
#   [GUI] UDP: [251 40 uint32()/uint32(samplestepvalue) 0/1 0/1]
def changed_option_generator(self, mode=True):
#   [GUI] UDP:
#         [251 50/51 (int32(genfreq)+int16(genpol)+int16(genret)+int16(genphase))]


# Aufzeichnung starten, pausieren und Informationen vom Messprogramm auswerten:

def aquire_prepare_and_start(self):
#   startet alle Tasks, uebermittelt Informationen an RP-Programm
def aquire_request_pause(self):
#   uebermittelt Pause-Information an RP-Programm
def aquire_pause_and_finish(self, sample_num, duration):
#   verarbeitet erhaltene Daten, schliesst Messung ab
def aquire_read_information(self, run_again=False):
#   uebermittelte Informationenen auswerten, Messungsdaten aufbereiten und
    visualisieren
```

8.2.4.4 Kalibrierungsfunktion

Um eine Kalibrierung des Polarisationsergebnisses zu ermöglichen, kann eine Liste mit den gewünschten Werten an das Messprogramm gesendet werden, dass dann die Berechnung vornimmt. Es stehen einige Funktionen zur einfachen Erstellung und Prüfung dieser Liste zur Verfügung.

```
# Kalibrierung laden und speichern:

def btn_calib_load_click(self):
#   [GUI] Kalibrierungsdatei laden
def btn_calib_save_click(self):
#   [GUI] Kalibrierungsdatei speichern


# Navigiere Soll-Wert:

def btn_calib_bottom_click(self):
#   [GUI] << niedrigster Wert
def btn_calib_down_click(self):
#   [GUI] <  Schritt hoch
def btn_calib_up_click(self):
#   [GUI] >  Schritt hinunter
def btn_calib_top_click(self):
#   [GUI] >> hoechster Wert


#   Daten hinzufuegen und bearbeiten:

def btn_calib_addline_click(self):
#   [GUI] aktuelle Wertauswahl -> Zeile hinzufuegen
def btn_calib_addsample_click(self):
#   [GUI] aktueller Messwert -> Zeile hinzufuegen
def btn_calib_check_click(self):
#   [GUI] ueberpruefen, sortieren und vervollstaendigen der Eintraege
def btn_calib_show_click(self, save_show='show', path=''):
#   [GUI] Kalibrierung visualisieren (Diagramm) oder Diagramm speichern
def btn_calib_clear_click(self):
#   [GUI] Kalibrierungs-Textfeld leeren
```

8.2.4.5 Datenanalysefunktion

Alle erfassten Daten und Informationen zu Aufnahme liegen nach dem Erfassen als Binary-Dateien und in einigen weiteren Dateien vor und können ausgewertet werden.

Eine einfache Möglichkeit dazu bieten die folgenden Datenanalysefunktionen, die den zeitlichen Verlauf der Polarisation, die Häufigkeit der Messwerte und eine Frequenzanalyse darstellen. Dabei ist für eine rasche Erstellung der Diagramme bei vielen Messpunkten die Einschränkung der Datenmenge möglich, indem nur ein bestimmter Bereich der Aufnahme verwendet wird oder Messwerte übersprungen werden.

```
# Aufnahme wählen, öffnen und Änderungen speichern:

def btn_data_load_info_click(self,var=None):
#   [GUI] ausgewaehlte Daten importieren (nur Informationen)
def btn_data_save_info_click(self):
#   [GUI] Beschreibung speichern
def btn_data_open_folder_click(self):
#   [GUI] Datenordner extern oeffnen (Explorer)
def btn_data_get_rp_click(self):
#   [GUI] Daten von STEMlab-Speicher holen (falls vorhanden)


# Anzeigemenge einstellen:

def btn_data_change_amout_to_preview_click(self):
#   [GUI] Datenpunkteinstellungen auf Preview anpassen (<=1.000 Datenpunkte) und
    anzeigen
def data_amount_change(self,var=None):
#   [GUI] Mengenaenderung
def data_unit_change(self,var=None):
#   [GUI] Einheitsaenderung


# Diagramme anzeigen:

def btn_data_show_diagramm_click(self):
#   [GUI] Diagramme mit aktuellen Einstellungen erstellen
def btn_data_show_testsample1_click(self):
#   [GUI] Start-Testsample anzeigen (falls vorhanden)
def btn_data_show_testsample2_click(self):
#   [GUI] End-Testsample anzeigen (falls vorhanden)
def btn_data_show_calib_click(self):
#   [GUI] Kalibrierung anzeigen (falls vorhanden)
```

8.2.4.6 UDP-Empfangsthreads

Es stehen zwei unabhängige Threads zur Verfügung, die eine möglichst verlustfreie Aufzeichnung der Daten garantieren. Ein Thread liest dabei ständig die Daten vom UDP-Socket ein und schreibt diese in eine Queue, der zweite liest aus dieser Queue und entpackt die Pakete vom Messprogramm in ihre Einzelkomponenten. Die Messwerte und Zeitstempel werden gespeichert, Informationen per Queue an den Hauptthread übergeben und dort von der Funktion aquire_read_information verarbeitet und von dieser entweder grafisch dargestellt oder eine entsprechende Aktion gestartet.

```
# Empfangen und Verarbeiten der UDP-Daten

def thread_udp_receive_package(q_udp_raw, target_port, sock_udp_timeout,
    sock_udp_maxsize):
#   Einlesen vom UDP-Socket und schreiben der Daten in die Queue q_udp_raw
def thread_udp_unpack_package(q_udp_raw, q_udp_info, path):
#   Einlesen von der Queue q_udp_raw, entpacken und verarbeiten der Daten und
    Informationen, speichern und weiterleiten an GUI-Thread (Hauptthread) per
    q_udp_info
```

9 Installation und Benutzung des Messprogramms

9.1 Installation

Um das Messprogramm nutzen zu können, sind eine Reihe von Schritten am STEMlab-Board und am Computer nötig. Diese stellen die Ausführbarkeit der beiden Programme und deren Kommunikationsmöglichkeit untereinander sicher.

9.1.1 STEMlab-Board für Messungen vorbereiten

Das Messboard selbst muss für die Verwendung vorbereitet werden, danach sind noch einige Schritte speziell für das Messprogramm durchzuführen.

9.1.1.1 Installation des Betriebssystems

- Herunterladen und Entpacken des gezippten Images (Link in der STEMlab-Dokumentation (Anh. A.3.2.1) in „Quick start - Prepare SD card“; verwendete Version: 0.98-617)
- Inhalt wie in der Dokumentation auf die SD-Karte schreiben (unter Windows mit dem Programm „Win32DiskImager“; es werden 2 Partitionen erstellt)
- SD-Karte einlegen und STEMlab-Board starten

9.1.1.2 Einrichten der Ethernet-Direktverbindung

Abbildung 9.1: STEMlab-Startseite und RedPitaya-Netzwerkmanager

Für die erste Verbindung mit einem unkonfigurierten System wird ein Router benötigt. Computer und STEMlab-Board im selben Netzwerk anschließen (Computer-Verbindung zum Netzwerk per WLAN möglich) und im Browser am Computer rp-xxxxxx.local/ eingeben (wobei xxxxxx die letzten 6 Stellen der Seriennummer darstellen, diese ist auf der LAN-Buchse zu finden, z.B. rp-f01a23.local/). [19]

Es erscheint die STEMlab-Startseite, Abb. 9.1. Unter System/Networkmanager kann eine statische IP eingestellt werden (192.168.4.129 ist die derzeitige Einstellung zur Kommunikation mit dem Client-Programm, die Änderung ist im Sourcecode möglich).

Das Messboard ist nach der Anwendung dieser Einstellung evtl. nicht mehr über den Router erreichbar, es sollte jetzt allerdings eine LAN-Direktverbindung zwischen Computer und Messboard möglich sein. Nach Herstellen der Kabelverbindung kann im Browser wieder rp-xxxxxx.local/ oder die statische IP-Adresse des verwendeten STEMlab-Boards (z.B. 192.168.4.129) eingegeben werden, um auf die STEMlab Startseite zu gelangen.

9.1.1.3 Internetzugriff für STEMlab-Board freigeben

Damit dem Messboard bei einer direkten Ethernet-Verbindung mit dem Computer ein Internetzugang zur Verfügung steht, müssen einige Einstellungen vorgenommen werden.

Der Zugang ermöglicht die einfache Installation der RedPitaya-API, die Installation von System-Updates, die Synchronisation der Uhrzeit (das Messboard verliert die Uhrzeit ohne Spannungsversorgung) und den Zugang zum Application Marketplace. Der Betrieb ohne Internetzugang ist möglich, die RedPitaya-API-Ordner müssen dann allerdings wie der SCPEM-Ordner manuell kopiert werden (Kap. 9.1.1.5).

Die Vorgangsweise wird anhand von Windows 10 in Abb. 9.2 beschrieben. In den Netzwerkeinstellungen muss zuerst die vom Computer genutzte Internet-Verbindung für andere Nutzer im Netzwerk gestattet werden (in dem Fall „Eigenschaften von WLAN“). Danach müssen die Einstellungen der Verbindung zum STEMlab-Board bearbeitet werden, genauer deren IPv4-Eigenschaften. Als IP-Adresse wird eine Adresse festgelegt, die dann im RedPitaya-Netzwerkmanager als Gateway angegeben wird (zum Beispiel 192.168.4.130). Zusätzlich ist ein gültiger DNS-Server anzugeben (z.B. 1.1.1.1 und 1.0.0.1 oder 8.8.8.8 und 8.8.4.4). [35, 36]

In den RedPitaya-Netzwerkeinstellungen im Browser ist zur IP-Adresse 192.168.4.129 die Gateway-Adresse anzugeben, diese muss mit der IP-Adresse des Computers übereinstimmen.

Zum Testen des Internetzuganges am Messboard kann der Application Marketplace aufgerufen oder auf ein Softwareupdate geprüft werden.

9.1.1.4 Verbindung mit PuTTY und WinSCP herstellen

Die Programme PuTTY und WinSCP sind sehr hilfreich bei der Nutzung des STEMlab-Boards. Für die Verbindung müssen nur je einige Einstellungen gesetzt werden, bei Verwendung einer anderen IP-Adresse des STEMlab-Boards ist dies zu berücksichtigen.

Genauere Informationen zu diesen Programmen können aus den jeweiligen Dokumentationen entnommen werden.

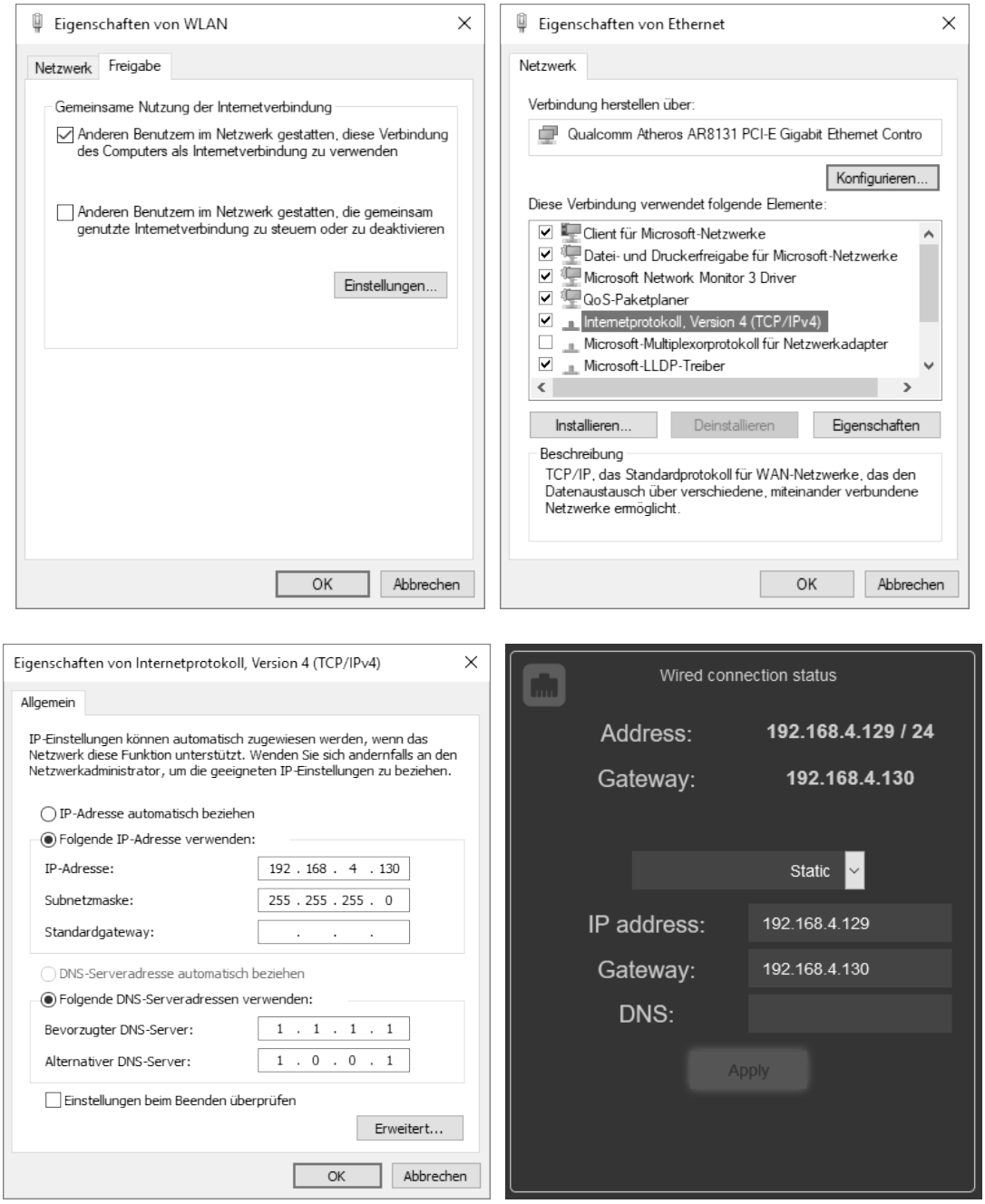


Abbildung 9.2: Netzwerkeinstellungen: WLAN, Ethernet, IPv4 und RedPitaya-Netzwerkmanager

PuTTY [37] PuTTY ermöglicht den SSH2-Zugriff und bietet damit viele Möglichkeiten, unter anderem Programme kompilieren und starten. [37]

Host Name/IP Adresse	192.168.4.129
Port	22
Connection Type	SSH
Login as	root

WinSCP [38] WinSCP zeigt eine einfache und übersichtliche Darstellung des Dateisystems am Messboard und lässt Daten übertragen oder direkt bearbeiten. [38]

Übertragungsprotokoll	SCP
Rechnername	192.168.4.129
Portnummer	22
Benutzername	root
Kennwort	root

9.1.1.5 Installation der RedPitaya-API [19]

Für diesen Vorgang muss dem STEMlab-Board eine Verbindung zum Internet zur Verfügung stehen. Nach dem Herstellen des SSH-Zugriffs mit PuTTY und Einloggen in das System (Passwort: „root“) kann das RedPitaya-Git-Repository auf das STEMlab-Board bzw. dessen SD-Karte kopiert werden:

```
git clone https://github.com/RedPitaya/RedPitaya.git
```

Während des Kopiervorganges sollte die Ausgabe etwa folgendermaßen aussehen:

```
Cloning into 'RedPitaya'...
remote: Counting objects: 76973, done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 76973 (delta 0), reused 0 (delta 0), pack-reused 76966
Receiving objects: 100% (76973/76973), 158.48 MiB | 3.09 MiB/s, done.
Resolving deltas: 100% (42817/42817), done.
Checking connectivity... done.
```

Zur Überprüfung kann eine SCP-Verbindung mit WinSCP hergestellt werden, es sollte wie in Abb. 9.3 ein Verzeichnis „root/root/RedPitaya“ vorhanden sein. Dieses beinhaltet die API und zahlreiche hilfreiche Beispielprogramme in unterschiedlichen Programmiersprachen.

9.1.1.6 Installation des SCPEM-Messprogramms

Um das Messprogramm am Messboard zu verwenden, muss dieses an einem bestimmten Ort vorliegen, damit es vom Clientprogramm gestartet werden kann. Der geforderte Pfad ist „root/root/SCPEM“ und in Abb. 9.3 ersichtlich.

Der SCPEM-Ordner muss zumindest die SCPEM-Datei (das kompilierte Programm) enthalten. Mithilfe der fünf C-Sourcefiles und des Makefiles ist das Ändern des Codes

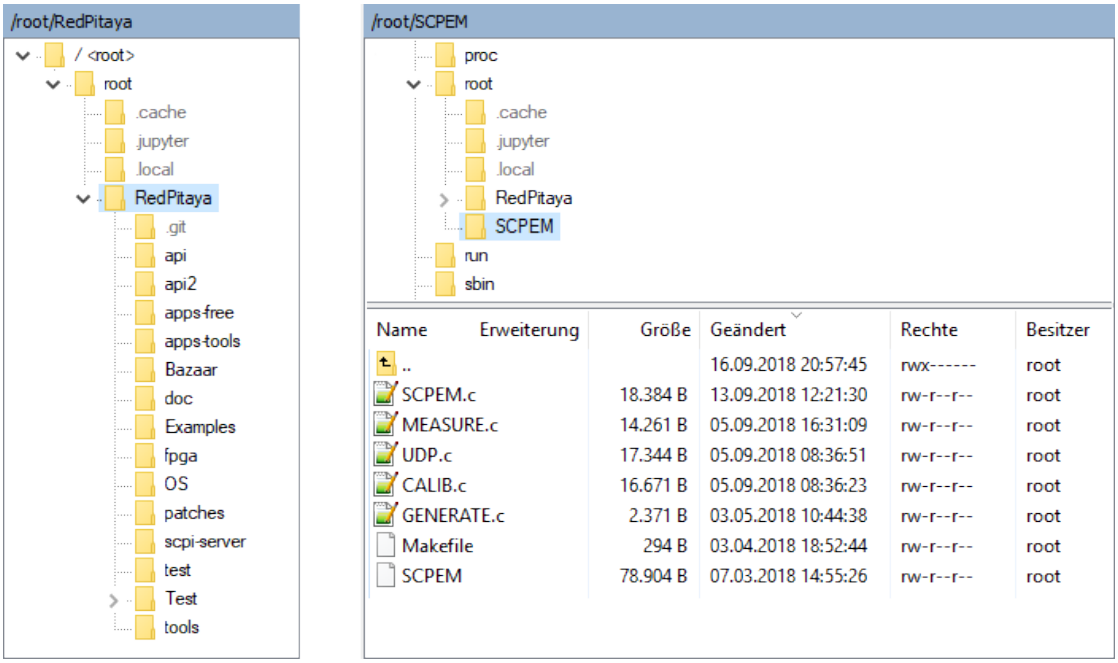


Abbildung 9.3: Erstellte Verzeichnisse der RedPitaya-API und Standardverzeichnis des Messprogramms

und das erneute Kompilieren möglich (mit PuTTY oder über die SCPEM-Messclient-Benutzeroberfläche). Beim Ausführen werden weitere Dateien und Ordner in diesem Verzeichnis erstellt.

9.1.1.7 Anschluss an den Messaufbau

Beim Anschließen des STEMlab-Boards an den Messaufbau sind die richtige Zuordnung der Eingänge und die Jumpereinstellungen für den richtigen Spannungsbereich zu beachten. Ein Vertauschen der Eingangskanäle ist im Programm nicht vorgesehen und ist bei Bedarf nur durch Änderungen im Sourcecode möglich. Vertauscht angeschlossene Signale führen dazu, dass die Messung nicht initialisiert werden kann (Nullpunktsuche des vermeintlichen SCPEM-Signals nicht möglich, da das Intensitätssignal stets positiv ist), unerkannte Fehlmessungen aufgrund vertauschter Eingänge sind daher beinahe ausgeschlossen.

Jumpereinstellung Das STEMlab-Board bietet die Möglichkeit, den Eingangsspannungsbereich mit Jumpern zwischen ±1 V (LV, Low Voltage) und ±20 V (HV, High Voltage) zu setzen. [19]

Das Signal des Dioden-Verstärkers liegt mit bis zu 10 V meist außerhalb des LV-Bereichs, das SCPEM-Signal liegt je nach Einstellung oft im LV-Bereich. Empfohlen wird, beide Jumper auf ±20 V (HV) zu setzen. Wird bei der Einstellung ±1 V (LV) der

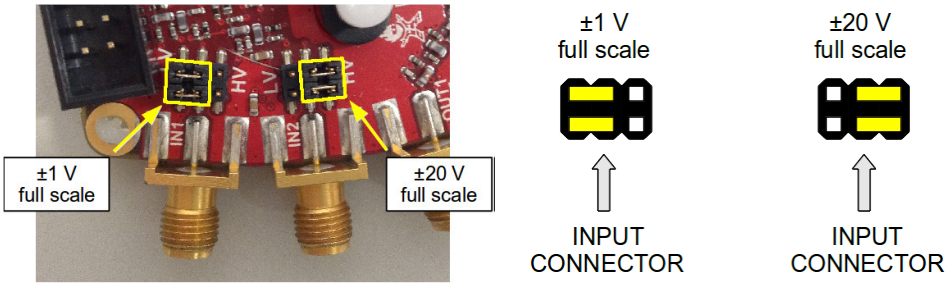


Abbildung 9.4: Festlegung des Spannungsbereiches am STEMlab-Board [19]

Messbereich überschritten, kommt es zu falsch interpretierten Messwerten. Eine Kontrolle auf das Überschreiten ist im Programm nicht integriert und lässt sich anschließend aus den erhaltenen Messdaten nicht feststellen, es können die richtigen Daten nicht rekonstruiert werden.

Nach Kontrolle der zu erwartenden Spannung kann eventuell für den Kanal 2 (SCPEM-Signal) ±1 V verwendet werden, dadurch sind jedoch praktisch keine Vorteile in der Messgenauigkeit zu erwarten, da dieser nur für die Nulldurchgangsbestimmung genutzt wird. Sollte das Dioden-Signal ständig kleiner als 1 V sein (bei Verwendung eines anderen Verstärkers oder aus anderen Gründen), kann für Kanal 1 der Jumper auf ±1 V gesetzt werden, in diesem Fall ist eine Verbesserung des Ergebnisses möglich.

Anschluss Kanal 1 Kanal 1, Channel1 oder IN1 befindet sich am Board rechts außen und ist für das verstärkte Diodensignal vorgesehen.

Anschluss Kanal 2 Kanal 2, Channel2 oder IN2 ist am Board rechts in der Mitte zwischen IN1 und OUT1 und ist für das SCPEM-Signal vorgesehen.

9.1.2 Installation von Python und Messclient am Computer

Für die Verwendung des Messclients werden neben Python 2.7 einige zusätzliche Module benötigt. Die Installationsschritte sind am Ende zusammengefasst (Kap. 9.1.2.2), genauere Informationen sind aus den jeweiligen Dokumentationen zu beziehen.

Das zum Entwickeln und Testen verwendete System hat unter anderem folgende relevante Eigenschaften:

- Betriebssystem: Microsoft Windows 10 Pro
- Prozessor: Intel® Core™ i3 CPU M 350, 2.27 GHz, 2 Kerne, 4 logische Prozessoren
- Installierter RAM: 4,00 GB
- Speicher: Samsung SSD 830 Series 128 GB
- Grafikkarte: NVIDIA GeForce 310M
- Netzwerkadapter: Qualcomm Atheros AR8131 PCI-E Gigabit Ethernet Controller

9.1.2.1 Python und Module

Die verwendete Python-Version ist 2.7.13.
Download des Installers für 32 bit/64 bit:
`https://www.python.org/ftp/python/`
Lizenz: `https://www.python.org/download/releases/2.7/license/`

NumPy NumPy wird für komplexere Berechnungen und für zusätzliche Datentypen benötigt. [39]
Installieren via pip im Verzeichnis Python27\Scripts:

```
pip install numpy
```

Alternativ Installation mit Windows Installer (inoffiziell):
`https://www.lfd.uci.edu/~gohlke/pythonlibs/#numpy`
Lizenz: `http://www.numpy.org/license.html`

Matplotlib/Pyplot Für die Darstellung der Diagramme wird die Matplotlib-Bibliothek verwendet. [40]
Installieren via pip im Verzeichnis Python27\Scripts:

```
pip install matplotlib
```

Lizenz: `https://matplotlib.org/devel/license.html`

TkInter TkInter ist eine Standard-Benutzeroberfläche von Python. [41, 42]
Es sollte in der Installation von Python bereits enthalten sein.
Lizenz: siehe Python

Pygubu Pygubu ist ein Werkzeug zum einfacheren Erstellen einer Benutzeroberfläche mit TkInter. [33, 43]
Installieren via pip im Verzeichnis Python27\Scripts:

```
pip install pygubu
```

Lizenz: GNU General Public License v3 (GPLv3/GPL-3)

Paramiko Für die Verwendung von SSH2 zum Starten des Messprogramms sowie für die Übermittlung des StdOut wird Paramiko verwendet. [44]
Installieren via pip im Verzeichnis Python27\Scripts:

```
pip install paramiko
```

Lizenz: `https://github.com/paramiko/paramiko/blob/master/LICENSE`

9.1.2.2 Gesamter Installationsablauf für Python und Module

- Python 2.7.13 (oder aktuellere Version von 2.7, wenn vorhanden) installieren von:
  `https://www.python.org/ftp/python/`
  `https://www.python.org/ftp/python/2.7.13/`
- Das Python-Package-Installationsprogramm pip ist standardmäßig in der Python-Installation enthalten. Sollte das nicht der Fall sein, ist die nachträgliche Installation in `https://pip.pypa.io/en/stable/installing/` beschrieben.
- Testen, ob TkInter vorhanden ist, ansonsten installieren (standardmäßig bereits in Python-Installation enthalten):
  `https://tkdocs.com/tutorial/install.html`
- Installieren der benötigten Module via pip:
  Im Verzeichnis „Python27\Scripts“ die Windows Power Shell öffnen und jeweils nach Vervollständigung der Installation des Moduls den nächsten Befehl eingeben:

  ```
  pip install numpy
  pip install matplotlib
  pip install pygubu
  pip install paramiko
  ```

  Es sollten alle benötigten Pakete erfolgreich installiert sein, bei Problemen sind die jeweiligen Dokumentationen heranzuziehen.

9.1.2.3 Ausführen des Messclients

Zum Ausführen des Messclients und des Messprogramms muss die gesamte Ordnerstruktur aus Anh. C.2 übernommen werden.

Die Programm-Hauptdatei „SCPEM-Aufnahme.py“ enthält die meisten Funktionen des Messclients.

In „Datenerfassung.py“ befinden sich die Programmteile, die für den Empfang und die Verarbeitung der Daten vom Messprogramm verantwortlich sind und als eigene Threads ausgeführt werden.

GUI.ui enthält die XML-Daten der Oberfläche, die vom Pygubu-Builder im Hauptprogramm nach diesen Informationen aufgebaut wird.

Im Unterordner „data“ werden die Aufnahmeordner angelegt, „options“ enthält die gespeicherten Einstellungen („recent.json“ darf nicht entfernt werden), die erstellten Kalibrierungen werden im Ordner „calibration“ abgelegt.

Zum Starten des Messclients muss das Python-Programm „SCPEM-Aufnahme.py“ ausgeführt werden.

9.2 Nutzungshinweise zum Messprogramm

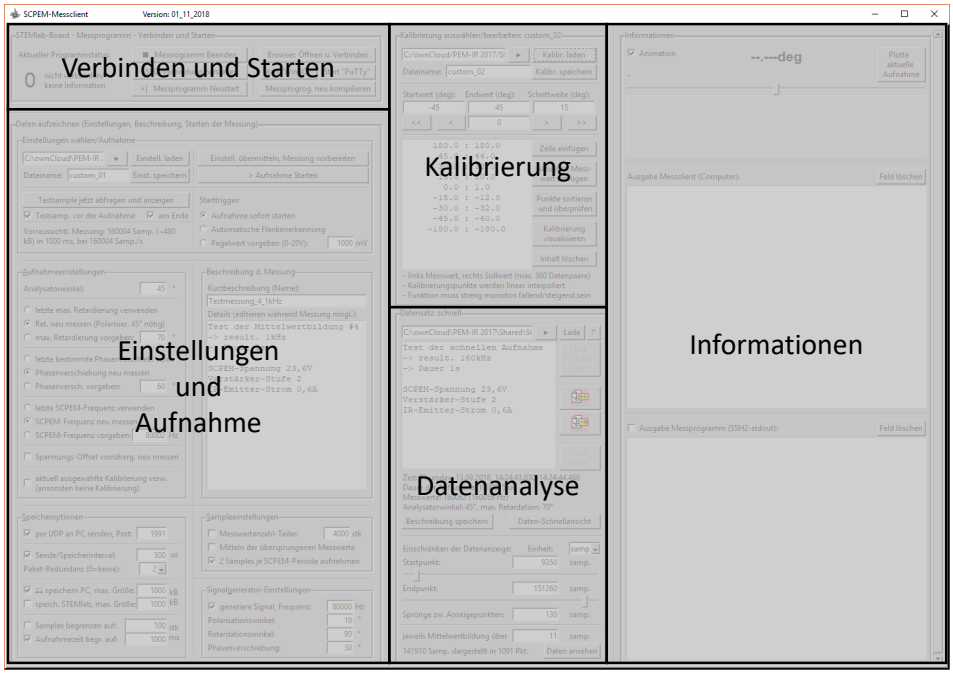


Abbildung 9.5: Aufteilung der Benutzeroberfläche in unterschiedliche Aufgaben

Die Oberfläche des Messprogramms (Abb. 9.5) befindet sich vollständig in einem Fenster und ist in mehrere Bereiche für unterschiedliche Aufgaben unterteilt. Um die Lesbarkeit zu gewährleisten, sind die Rahmen auf den folgenden Seiten einzeln abgebildet. In diesem Kapitel wird die Nutzung der Funktionen durch den Nutzer und die Auswirkung der Einstellungen auf die Messung beschrieben.

9.2.1 Verbinden und Starten

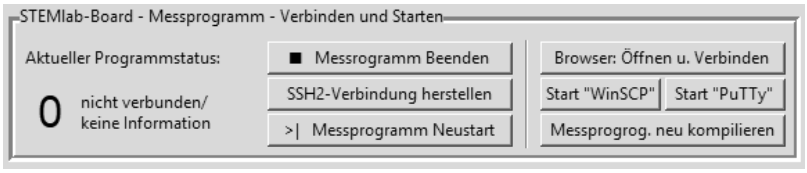


Abbildung 9.6: Oberfläche „Verbinden und Starten“

In Abb. 9.6 werden Funktionen zum Verbinden des Messclients mit dem STEMlab-Board sowie zum Starten vom Messprogramm und anderen externen Anwendungen angezeigt.

Links ist der aktuelle Programmstatus sichtbar, der in Kap. 8.2.4.2 definiert ist. Es ist nicht nötig, alle Zwischenschritte des Status manuell durchzuführen, beim Programm-

start oder dem Übermitteln der Einstellungen ohne bereits bestehender Verbindung werden automatisch alle davor benötigten Funktionen ausgeführt.

Die Kompilierungsfunktion ermöglicht das Erstellen der ausführbaren Datei aus dem Sourcecode am STEMlab-Board, dadurch sind Änderungen schnell durchführbar. Eventuelle Kompilierungsfehler werden angezeigt.

9.2.2 Aufnahmeeinstellungen und -start

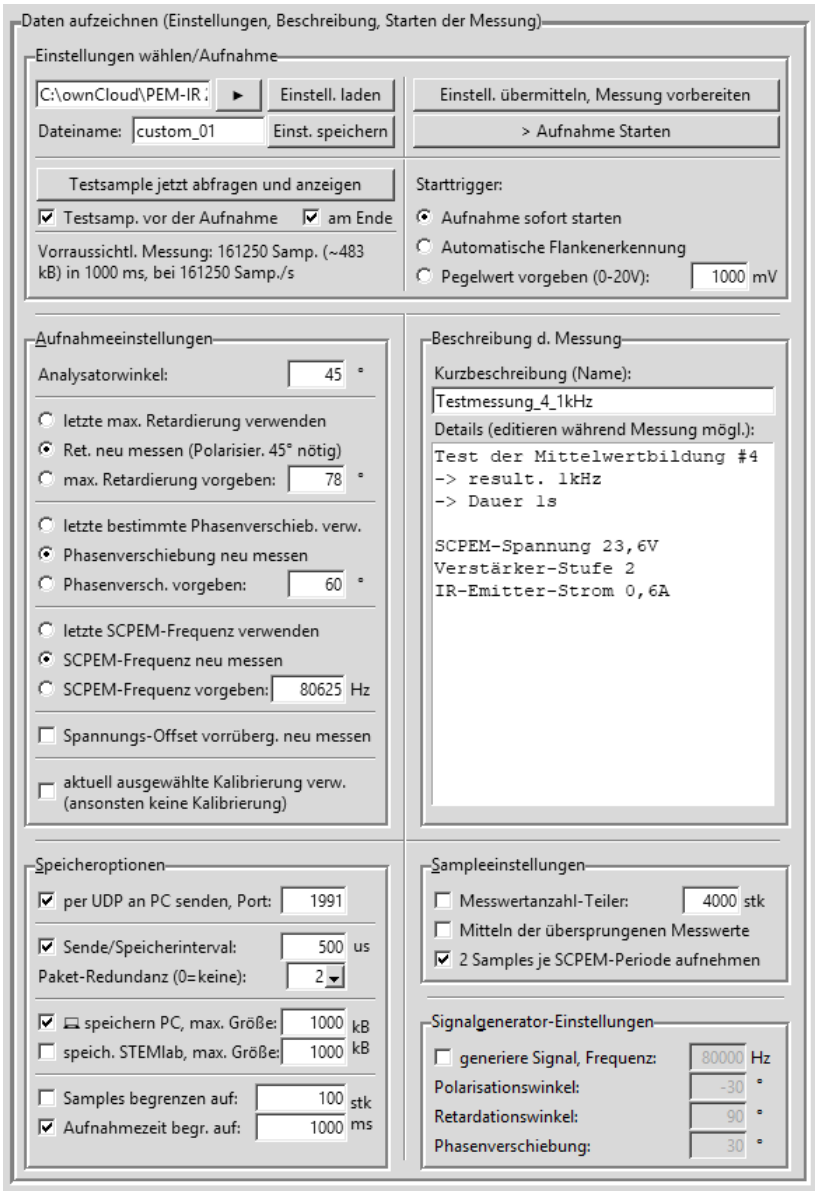


Abbildung 9.7: Oberfläche „Einstellungen und Aufnahme“

Die Aufnahmeeinstellungen sind in Abb. 9.7 gezeigt. In diesem Programmbereich wird die Aufnahme definiert, gestartet und beendet.

Einstellungs-Vorlage wählen, Start der Aufnahme: Die Kommunikation mit dem Messprogramm wird definiert und die Aufnahme mit gewünschten Einstellungen gestartet.

- Einstellungen laden: Die gesamten Einstellungen können aus einer Vorlage geladen oder für spätere Aufnahmen gespeichert werden. Standardmäßig werden bei jedem Programmstart die letzten Einstellungen wiederhergestellt.
- Testsample abfragen: Ein Testsample zeigt die Aufnahme einiger Perioden der beiden Signale (vergleiche Abb. 7.5). Die ermittelten Werte (Nulldurchgang, Extremwerte) sind ebenfalls im Diagramm eingetragen. Bei geänderten Einstellungen kann ein Testsample zu deren kurzen Überprüfung sinnvoll sein. Es kann automatisch zu Beginn und am Ende der Aufnahme ein Testsample erstellt und im Datenordner abgelegt werden, wodurch Fehler im Nachhinein besser erkennbar sind.
- Voraussichtliche Messung: Hier werden einige Eigenschaften der Aufnahme bei den aktuellen Einstellungen abgeschätzt.
- Einstellungen übermitteln: Es werden alle gewählten Einstellungen an das Messprogramm übermittelt. Ist noch keine Verbindung hergestellt, wird die Verbindung automatisch hergestellt und das Messprogramm gestartet, anschließend werden die Einstellungen übermittelt.
- Aufnahme starten: Abhängig von der Starttrigger-Einstellung wird die Aufnahme mit den aktuellen Einstellungen gestartet. Dieser Button erhält nach dem Aufnahmestart die Funktion zum Pausieren/Beenden der Aufnahme.
- Starttrigger: Die Aufnahme kann sofort oder erst bei Erreichen einer bestimmten Strahlungsintensität gestartet werden. Dadurch lässt sich eine unnötige Aufnahmezeit ohne Messdaten verhindern. Die automatische Erkennung startet bei einem schnellen Anstieg des Intensitätssignals, alternativ kann ein bestimmter Mindest-Spannungspegel des Signals vom Photodioden-Verstärker für den Aufnahmestart vorgegeben werden (berechnet mit dem HV-Messbereich, Kap. 9.1.1.7). Für die Auswertung des Signals im Messprogramm ist die in Kap. 7.6.3 beschriebene Funktion zuständig.

Aufnahmeeinstellungen: Für die korrekte Auswertung der Polarisation müssen einige Variablen richtig ermittelt oder vorgegeben werden. Falsche Einstellungen führen zu schlechten/falschen Ergebnissen oder behindern den Aufnahmestart.

- Analysatorwinkel: Der am Messaufbau eingestellte Analysatorwinkel (bezogen auf die Längsachse des SCPEM) ist manuell vorzugeben.
- Retardierung: Die maximale Retardierung, die der SCPEM aktuell erreicht, muss für die Berechnung der Polarisation bekannt sein. Für deren Messung muss das

Messprogramm gestartet sein (Aufnahme pausiert) und die Strahlung vor dem SCPEM mit einem Polarisator auf möglichst genau 45° vollständig polarisiert werden. Die Bestimmung erfolgt analog zu Kap. 3.2.3.2. Ist die Retardation durch die Versorgungsspannung etc. bekannt, kann diese manuell vorgegeben werden.

- Phase: Die Phasenverschiebung zwischen SCPEM-Signal und Intensitätssignal wird vom Messprogramm für die Bestimmung der Intensitäts-Extremwerte benötigt. Für die Messung ist eine Polarisation von etwa 15°-75° nötig, bei dem die Minima und Maxima deutlich ausgebildet sind. Die Phasenverschiebung wird während der Messung selbstständig an kleine Änderungen angepasst. Der angezeigte Phasenwert entspricht nur der Position eines Extremwerts, die restlichen drei werden im Messprogramm jeweils etwa 90° der SCPEM-Phase versetzt ermittelt.

- Frequenz: Für die Messung der Frequenz des SCPEM muss die Ansteuerung eingeschaltet und angeschlossen sein. Die Frequenz wird im Messprogramm während der Aufnahme ständig aktuell gehalten.

- Spannungs-Offset: Die Offset-Spannung der Eingänge kann im STEMlab-Programm „Oszilloscope“ (aufrufbar über den Browser) ermittelt werden. Dieser Wert wird vom Messprogramm beim Programmstart automatisch eingelesen und berücksichtigt. Alternativ kann der Offset vom Programm ermittelt werden (an den Eingängen darf dabei kein Signal liegen). Dadurch kann die vordefinierte Offset-Spannung überprüft oder vorübergehend (bis zum Neustart des Messprogramms) überschrieben werden. Die STEMlab-Offset-Einstellung dabei wird nicht beeinflusst.

- Kalibrierung verwenden: Die aktuell im Kalibrierungs-Frame ausgewählte Kalibrierung wird an das Messprogramm geschickt und in der Funktion aus Kap. 7.6.5 bei der Ermittlung der Polarisation berücksichtigt.

Speicheroptionen: Diese legen fest, wie und wo die erhaltenen Messdaten behandelt und abgelegt werden.

- Port: Wenn der Port durch einen Programmabsturz nicht richtig geschlossen wurde, kann ein anderer Port zugewiesen werden. Das längere Deaktivieren der UDP-Kommunikation ist nicht empfohlen und kann zu undefiniertem Verhalten führen.

- Intervall: Dieses gibt die Zeit an, die vom Messprogramm mindestens zwischen zwei Sende- oder Speichervorgängen gewartet wird. Ist der Wert 0, werden neue Daten sofort gesendet, wenn sie bereitstehen. Wenn es häufig zu UDP-Paketverlusten kommt, kann eine Erhöhung des Intervalls die Stabilität verbessern, bewährt haben sich Intervalle von 0,5 ms bis 2 ms.

- Paket-Redundanz: Einzelne verlorene Pakete können durch eine erhöhte Redundanz der Daten in den Paketen wiederhergestellt werden. Dadurch wird die zu übertragende Datenmenge erhöht. Die Redundanz gibt an, wie viele direkt aufeinanderfolgende verlorenen Pakete ohne Datenverlust toleriert werden können.

- Speichergröße am Computer: Die Messwerte werden nach dem Empfangen in mehreren Dateien mit fortlaufender Nummer gespeichert, deren maximale Größe hier festgelegt wird (0 bedeutet eine einzelne Datei unbegrenzter Größe).
- Speichergröße am STEMlab-Board: Es besteht alternativ die Möglichkeit, die Messdaten direkt vom Messprogramm am STEMlab-Board speichern zu lassen. Dabei sollte ein Wechseldatenträger am USB-Anschluss des Messboards verwendet werden. In diesem Fall werden die Daten automatisch auf diesen geschrieben, ansonsten direkt auf die SD-Karte. Die Dateien können nach Abschluss der Aufnahme manuell oder mit der dafür vorgesehenen Funktion im Datenanalyse-Frame (Kap. 9.2.4) zur weiteren Verwendung auf den Computer kopiert werden.
- Begrenzung der Aufnahmedauer: Wenn die Dauer der Aufnahme oder die gewünschte Anzahl an Messwerten bekannt ist, kann diese vor Start der Aufnahme festgelegt werden. Die Aufnahme wird beim Überschreiten der zuerst erreichten Grenze automatisch pausiert. Dadurch sind sehr kurze Messungen möglich, die manuell nicht realisierbar wären.

Beschreibung der Aufnahme: Jeder Messung soll ein Name zugeordnet werden, der den Ordnernamen der Messdaten bestimmt und daher nach Messbeginn nicht mehr geändert werden kann. Die Beschreibung kann beliebige Informationen zur Messung enthalten (Temperatur, Material, etc.), Aufnahmezeit, -datum und -einstellungen werden vom Programm automatisch gespeichert und müssen nicht manuell eingegeben werden. Die Beschreibung wird am Ende der Messung erfasst und kann anschließend bei Bedarf im Datenanalyseframe (Kap. 9.2.4) geändert werden.

Sampleeinstellungen: Diese Einstellungen ermöglichen die Beeinflussung der Datenmenge bei vorgegebener SCPEM-Frequenz. Sie werden im Messprogramm von der in Kap. 7.6.4 gezeigten Funktion berücksichtigt.

- Messwertanzahl-Teiler: Wenn langsame Änderungen der Polarisation beobachtet werden, kann es sinnvoll sein, die Anzahl der Messwerte zu reduzieren. Dadurch wird einerseits die Übertragung aufgrund geringerer Datenmengen stabiler, die spätere Auswertung ist schneller möglich und durch Mittelwertbildung kann eine Erhöhung der Genauigkeit erreicht werden.
- Mittelwertbildung: Es wird der arithmetische Mittelwert der übersprungenen Polarisations-Messwerte gebildet. Abhängig vom Messwertanzahl-Teiler kann dadurch die Messgenauigkeit erhöht werden. Diese Einstellung ist nicht zu verwechseln mit der Mittelwertbildung der Samples im FPGA, diese ist standardmäßig aktiviert (Kap. 7.2.1).
- Zwei Messwerte je SCPEM-Periode: Da das Intensitätssignal im Vergleich zum SCPEM-Signal die doppelte Frequenz aufweist, ist es möglich, zweimal je Periode einen Polarisationswert zu ermitteln und somit die doppelte Messfrequenz zu erreichen.

Signalgenerator: Für Testzwecke besteht die Möglichkeit, Signale an den beiden Ausgängen des STEMlab-Boards zu erzeugen, die etwa den gemessenen Größen am Messaufbau bei einer bestimmten Frequenz, Polarisation, max. Retardation und Phasenverschiebung entsprechen. Dazu werden von den 4 BNC-Anschlüssen die beiden äußeren (Intensitätssignal) und die beiden inneren (SCPEM-Signal) Stecker verbunden. Beim Neustart des Client-Programms wird diese Option standardmäßig deaktiviert.

9.2.3 Kalibrierungserstellung

Im linken Rahmen in Abb. 9.8 kann eine fertige Kalibriertabelle geladen und geändert werden oder eine neue Kalibrierung erstellt und gespeichert werden.

Die Navigationstasten ermöglichen auf einfache Weise, die Zeilen mit aufsteigenden Werten zu füllen. Ist zum aktuellen Zeitpunkt eine Messung aktiv (für diese Aufgabe idealerweise mit hohem Messwertanzahl-Teiler und Mittelwertbildung), kann der aktuelle Messwert eingefügt werden.

Um die Kalibrierung abzuschließen, müssen die Werte sortiert und überprüft werden, die Visualisierung bietet zusätzlich eine grafische Darstellung der Kalibrier-Funktion. Beim Speichern wird jeweils die Punkteliste und die Visualisierung abgelegt.

Die Anzahl der Kalibrierungspunkte ist auf 360 beschränkt und die Kalibrierungskurve muss streng monoton fallend oder steigend sein.

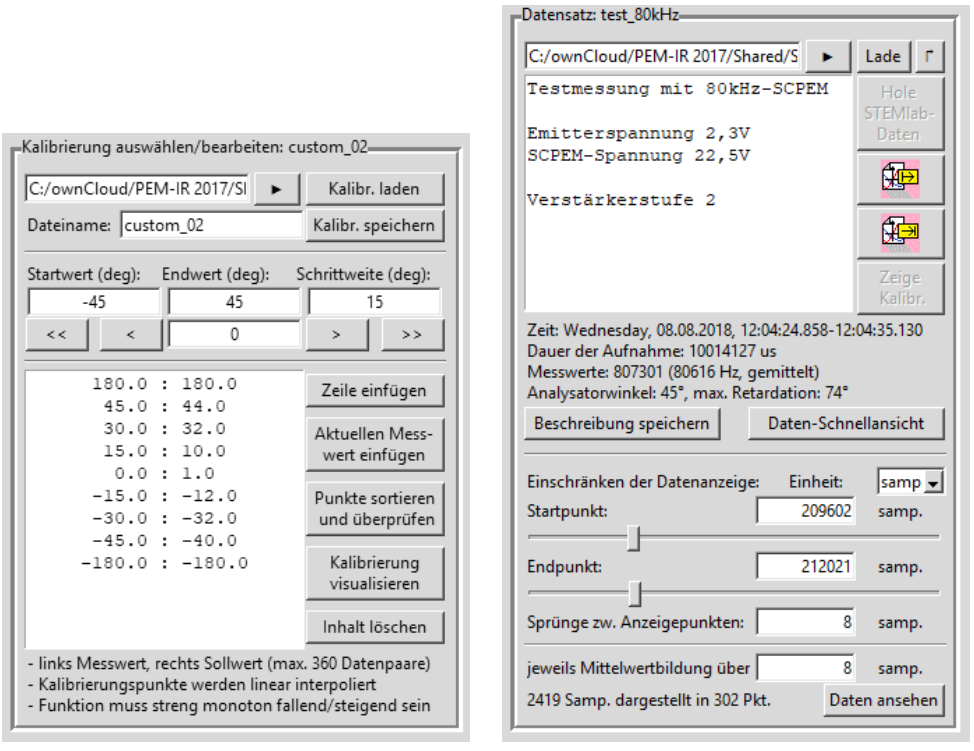


Abbildung 9.8: Oberfläche „Kalibrierung“, Oberfläche „Datenanalyse“

9.2.4 Betrachten und Analysieren einer Aufnahme

Der rechte Rahmen in Abb. 9.8 zeigt den Zustand nach Auswahl und Laden eines Datensatzes. Es werden Informationen der Aufnahme angezeigt, die Beschreibung kann geändert und gespeichert werden.

Wurden die Daten lokal am STEMlab-Board gespeichert, können sie mit dem Button „Hole STEMlab-Daten“ in das Computer-Verzeichnis kopiert werden. Darunter sind die beiden Testsamples vom Beginn und Ende der Aufnahme und die verwendete Kalibrierung abrufbar.

Die Datenmenge kann für eine schnellere Darstellung in verschiedenen Einheiten eingeschränkt werden. Die Daten-Schnellansicht nimmt die Einstellungen automatisch so vor, dass etwa 1.000 Punkte ausgewertet werden und schnell ein Ergebnis vorliegt. Es kann bei Bedarf für jeden Anzeigewert über mehrere Messwerte gemittelt werden.

Nach der Erstellung der Diagramme stehen im neu erstellten Fenster (Kap. 9.3) einige Funktionen der pyplot-Library zur Verfügung, mit denen die Plots bearbeitet werden können. Die angezeigten Diagramme werden automatisch im Aufnahmeordner gespeichert, dieser kann über den kleinen Pfeil rechts oben geöffnet werden.

9.2.5 Informationen

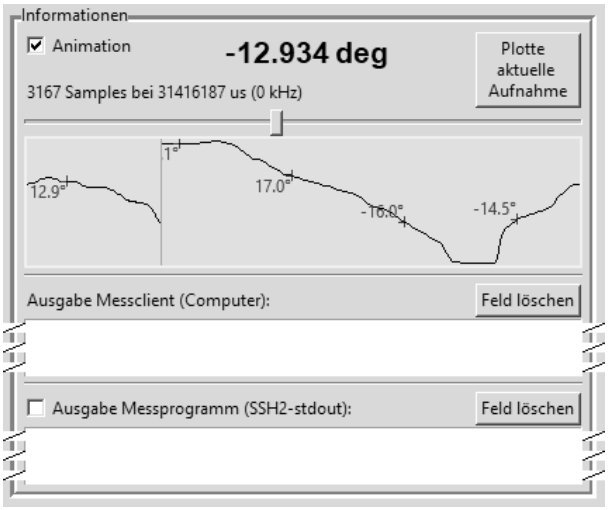


Abbildung 9.9: Oberfläche „Informationen“ während aktiver Aufnahme

Der rechte Teil des Messclient-Fensters (Abb. 9.9) dient zur Informations-Ausgabe.

Oben werden Daten zur aktuellen Aufnahme angezeigt, der gemessene Winkel und die Anzahl der bereits gespeicherten Messwerte wird mehrmals pro Sekunde aktualisiert und bei aktivierter Animation direkt darunter grafisch dargestellt. Eine Übersicht über die gesamte aktuelle Messung kann während der Aufzeichnung als Plot ausgegeben werden.

Darunter befindet sich die Text-Ausgabe des Messclients, wo Vorgänge und vom Messprogramm übermittelte Daten ausgegeben werden.

Im untersten Feld kann die StdOut- und StdErr-Ausgabe des Messprogramms angezeigt werden. Dafür muss diese Option vor dem Start des Messprogramms gewählt werden, die Daten werden dann über SSH2 an den Messclient übermittelt und angezeigt. Die Kommunikation wird dadurch instabiler, daher sollte diese Übertragung nur wenn nötig aktiviert werden. Die Ausgabe des Messprogramms wird ansonsten in den Dateien stdout.txt und stderr.txt im SCPEM-Verzeichnis am STEMlab-Board abgelegt.

9.3 Messergebnis

9.3.1 Daten im Aufnahmeordner

Im Ordner der Aufnahmen befinden sich mehrere Dateien, einige davon sind direkt lesbar, andere liegen als Binary-Dateien vor.

9.3.1.1 Sample-Dateien

Die Dateien mit den Messwerten haben als Namen eine fortlaufende Nummer (beginnend bei samp_00000000, begrenzt auf 100 Millionen Dateien je Aufnahme), die Größe ist abhängig von der Einstellung bei der jeweiligen Aufnahme.

16 bit	16 bit	16 bit	16 bit	16 bit	16 bit		16 bit
Samp. 0	Samp. 1	Samp. 2	Samp. 3	Samp. 4	Samp. 5	...	Samp. D

Die D Polarisationswerte (vergleiche Tab. 6.6) werden als int16-Werte in diesen Binary-Files abgespeichert. Um die Polarisation in Radiant zu erhalten, ist die Division durch 10.428 nötig, für Grad durch 182 (siehe Kap. 7.5). Die Dateigröße beträgt jeweils ein Vielfaches von 2 Byte (D*2 Byte).

9.3.1.2 Timestamp-Dateien

Diese haben ebenfalls eine fortlaufende Nummer (beginnend bei time_00000000) als Namen, die Größe ist abhängig von der Dateigrößen-Einstellung und dem Sende-/Speicher-Intervall bei der jeweiligen Aufnahme. Beim Beginn jeder neuen Sample-Datei wird auch eine neue Timestamp-Datei erstellt, dadurch stimmen die Daten mit gleicher Dateinummer überein. Die Datei besteht aus Informationspaaren, es wird jeweils einem Sampleindex ein Zeitstempel (Aufnahmezeitpunkt des Samples, gemessen vom Aufnahmebeginn in μs) zugeordnet.

32 bit	32 bit	32 bit	32 bit	32 bit	32 bit	...	32 bit	32 bit
Index 0	Zeit 0	Index 1	Zeit 1	Index 2	Zeit 2	...	Index N	Zeit N

Die Werte liegen dabei jeweils im uint32-Format vor. Der Abstand der Sampleindizes hängt von den Sende-/Speichereinstellungen ab, da bei jedem Paket nur ein Timestamp mitgesendet wird (Tab. 6.6). Die Aufnahmezeitpunkte der restlichen Samples lassen sich linear interpolieren, da die SCPEM-Frequenz in diesen kurzen Zeitabschnitten praktisch konstant ist. Es gilt in jedem Fall $N \leq D$, wobei N allgemein im Bereich einer bis zwei Größenordnungen kleiner als D ist. Die Dateigröße beträgt jeweils ein Vielfaches von 8 Byte (N*8 Byte).

9.3.1.3 Weitere Dateien für allgemeine Informationen zur Aufnahme

Im Aufnahmeordner befinden sich zwei Dateien mit allen Einstellungen zu Beginn und zum Abschluss der Messung (start.json und finished.json). Wenn die Testsampleoption bei der Aufnahme aktiviert war, sind diese sowohl als Vektorgrafik wie auch als Textdatei im CSV-Format verfügbar (Trennzeichen Semikolon, erste Zeile Spaltentitel) und können in Tabellenkalkulationsprogrammen importiert werden. Wurde bei der Messung eine Kalibrierung verwendet, ist diese ebenfalls als Grafik und als Text-Liste abgelegt.

9.3.2 Import der Binary-Daten mit MATLAB

Algorithmus 9.1 Einlesen der abgelegten Daten mit MATLAB (Anh. B.1.15)

```
% Importieren der Daten aus den folgenden Dateien in Arrays:
file_index=0;   % Datei-Index (beginnend bei 0 fuer erste Datei)
path='C:\...\SCPEM_Aufnahme\data';              % Abs. Pfad zum data-Ordner
folder='2018_08_08\12_02_28_895_test_80kHz';  % Ordner mit Aufnahmedaten
samp_array=[];  % Array mit Messwerten
time_array=[];  % Array mit Messwerte-Indizes und zugehoerigem Timestamp

while 1
    % Messwerte importieren:
    file_path = [path, '\', folder, sprintf('\\samp_%08d', file_index)]
    if exist(file_path, 'file')~=2    % Pruefung, ob Datei vorhanden
        break;
    end
    file_ID = fopen(file_path);
    samp_array = [samp_array; fread(file_ID, Inf, 'int16')];
    fclose(file_ID);

    % Timestamps importieren:
    file_path = [path, '\', folder, sprintf('\\time_%08d', file_index)]
    if exist(file_path, 'file')~=2    % Pruefung, ob Datei vorhanden
        break;
    end
    file_ID = fopen(file_path);
    time_array = [time_array, fread(file_ID, [2, Inf], 'uint32')];
    fclose(file_ID);

    file_index=file_index+1; % naechster Datei-Index
end

%samp_array = samp_array/10428.;   % Umrechnung fuer Radiant
%samp_array = samp_array/182.;     % Umrechnung fuer Grad

samp_array    % Messamples:           [s0; s1; s2; ...]
time_array    % Index und Timestamp: [i0, t0; i1, t1; i2, t2; ...]
```

Alg. 9.1 aus Anh. B.1.15 zeigt eine Möglichkeit zum Einlesen der Dateien. Die Daten stehen dann in Arrays zur weiteren Verarbeitung zur Verfügung.

9.3.3 Import der Binary-Daten mit Python

An dieser Stelle sei nur auf die Funktion btn_data_show_diagramm_click des Messclients (Anh. C.2.1) verwiesen. Dort wird das Einlesen und Verarbeiten der Daten für die Analyse durchgeführt, die Vorgehensweise ist dabei ähnlich zu Kap. 9.3.2.

9.3.4 Beispiel einer Aufnahme und deren Analyse (Anh. B.1.14)

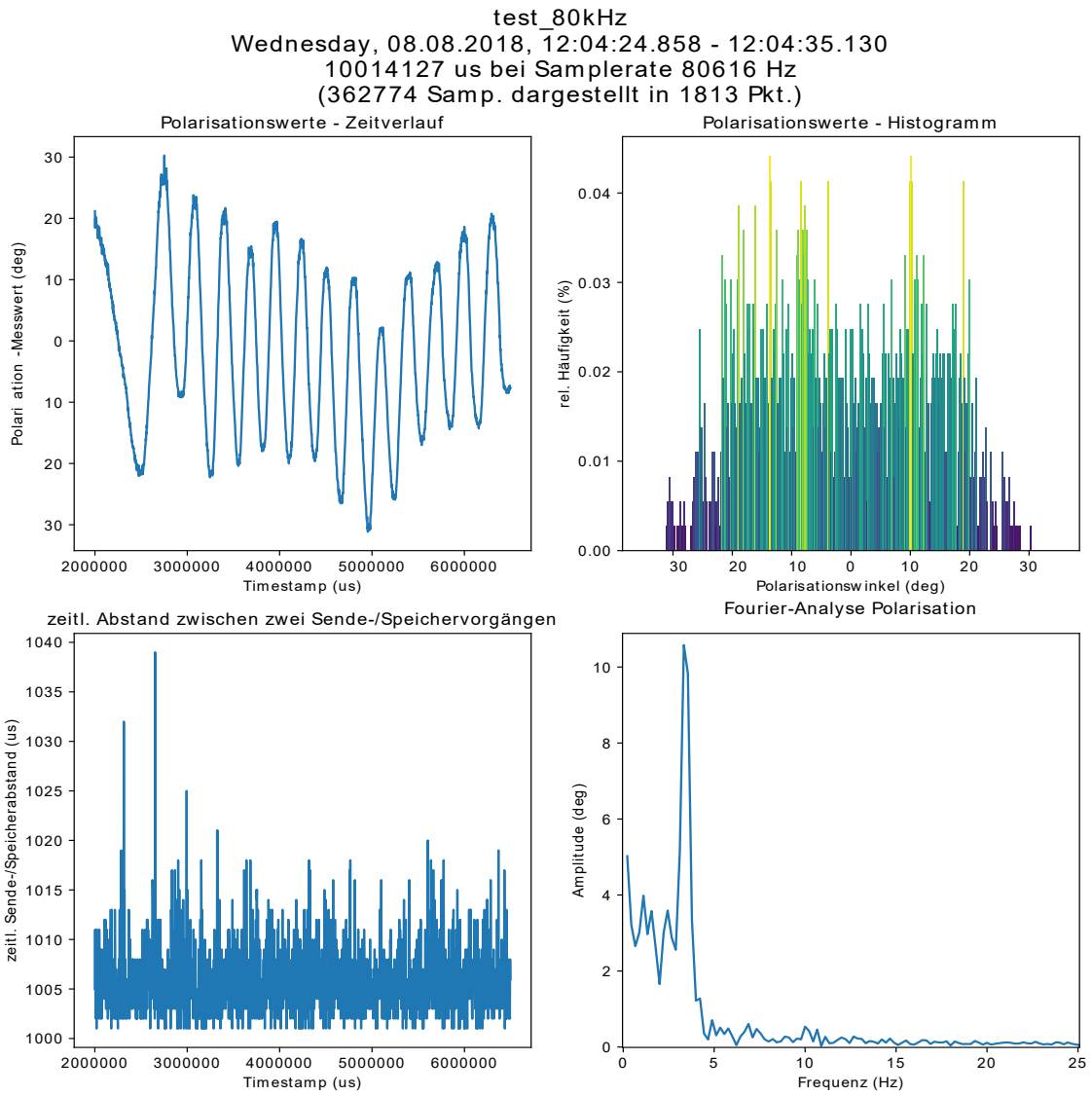


Abbildung 9.10: Analyse einer Testmessung mit dem Messclient-Programm

Abb. 9.10 zeigt die Analyse einer Messung, wie sie vom Messclient ausgegeben wird. Nicht relevante Daten am Anfang und am Ende der Aufnahme sind bereits entfernt, für

die übrigen ca. 4 s wird nur jeder 200. Messwert dargestellt (ohne Mittelwertbildung).

Bei dieser Aufnahme wurde der Polarisator manuell so schnell wie möglich hin- und herbewegt, dadurch ist jedoch im Vergleich zur Aufnahmegeschwindigkeit von über 80 kHz nur eine langsame Polarisationsänderung erreichbar. In der Analyse ist der Zeitverlauf und das Histogramm der Messwerte enthalten, der Plot der Timestamp-Differenzen deutet darauf hin, dass das Sende-/Speicherintervall auf 1 ms eingestellt war. Im Frequenzspektrum der Aufnahme ist in diesem Fall die Bewegung des Polarisators mit etwa 3 Hz zu erkennen.

Für diese Anwendung ist die hohe Messgeschwindigkeit nicht nötig und nicht sinnvoll, wie im kürzeren Ausschnitt der selben Aufnahme im Detail Abb. 9.11 zu erkennen ist. Dort wird für eine Zeit von 0,03 s jeder ermittelte Polarisationswert dargestellt. Die Messfehler sind ist im Vergleich zur gleichmäßigen Veränderung des Messwertes groß, weniger Messwerte mit Mittelwertbildung sind für die Abbildung dieser Messgröße sinnvoller. Ändert sich die Polarisation sehr schnell, können viele Messpunkte trotz des bleibenden Messfehlers Zusatzinformationen gegenüber deren Mittelwert bringen.

Abb. 9.12 zeigt den gleichen Ausschnitt der Aufzeichnung wie Abb. 9.11, allerdings mit reduzierter Samplerate von 10 kHz bzw. 1 kHz bei aktivierter Mittelwertbildung (Mittelwertbildung nachträglich bei der Datenanalyse).

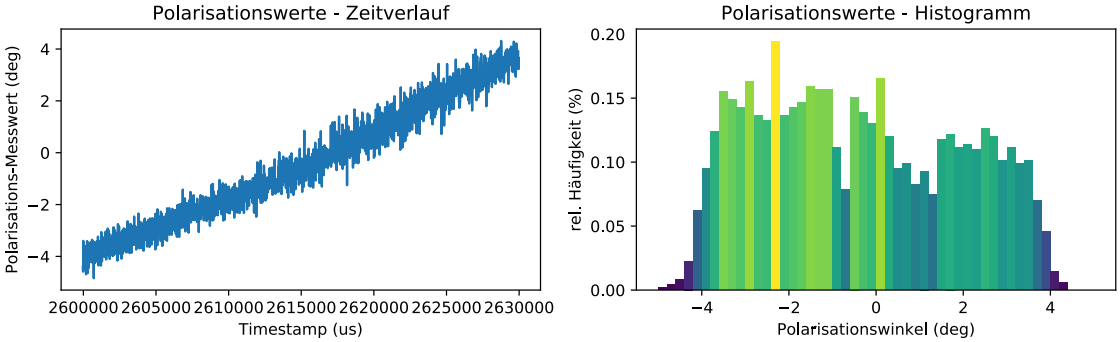


Abbildung 9.11: Messdaten im Detail, Darstellungs-Schrittweite 1, 80 kHz

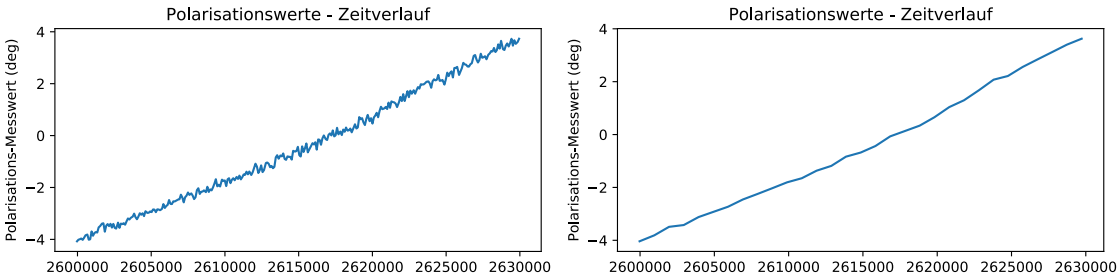


Abbildung 9.12: Messdaten im Detail, Schrittweite 8, 10 kHz; Schrittweite 80, 1 kHz

10 Zusammenfassung und Ausblick

10.1 Ergebnis der durchgeführten Arbeiten

- Durch die theoretische Betrachtung der polarisierten Strahlung am Weg durch das Messsystem und insbesondere den SCPEM steht ein gelöstes Gleichungssystem zur Verfügung, das die Berechnung des Polarisationswinkels aus dem Intensitätssignal der Photodiode ermöglicht. (→ Kap. 3)
- Der Messaufbau ist für Infrarotstrahlung mit einer Wellenlänge von 1.500 nm geeignet. Die Ansteuerungs-Schaltung des SCPEM befindet sich auf einer eigenen Platine mit Anschlüssen für SCPEM, Versorgung und Signal. (→ Kap. 4, Kap. 5)
- Die Signalmessung und -auswertung mit dem STEMlab-Board von RedPitaya ist umgesetzt und dokumentiert. Das gleiche gilt für die schnelle Übermittlung der Polarisationsdaten an andere Geräte zur weiteren Verarbeitung. Mit den beiden Programmen für das Messboard und den damit verbundenen Computer steht eine einfache Möglichkeit der Messung, Auswertung, Aufzeichnung und Analyse zur Verfügung. Sie können als Ausgangspunkt für weitere Entwicklungen herangezogen werden. (→ Kap. 6, Kap. 7, Kap. 8)
- Die Vorbereitung der Geräte und die Verwendung des Messprogramms ist detailliert beschrieben. (→ Kap. 9)

10.2 Mögliche Verbesserungen und Erweiterungen

Es sind einige Änderungen am Messaufbau und beim Messprogramm möglich, mit denen eventuell teilweise auftretende Probleme minimiert werden können oder weitere Performancesteigerungen möglich sind.

10.2.1 Entladewiderstand für Kondensator in SCPEM-Ansteuerung

Der Kondensator zur Steuerung der Verringerung des Startwiderstandes in der Rückkopplung ist beim Starten des SCPEM entladen und lädt sich dann in wenigen Sekunden auf die Ausgangsspannung des Spannungswandlers auf (Kap. 5.3.1.1). Wird die Ansteuerung abgeschaltet und kurze Zeit darauf wieder gestartet, ist der Kondensator noch geladen und die Widerstandsreduktion tritt nur sehr abgeschwächt auf und kann zum Misslingen des Anschwingvorganges führen. Die Entladedauer beträgt einige wenige Minuten. Gelöst werden kann dies durch einen Widerstand, der parallel zum Kondensator zwischen V_{Out} und GND liegt.

Beispielsweise wird ein zusätzlicher 22 $k\Omega$-Widerstand R_6 verwendet (Abb. 5.7 und Abb. 10.1), damit kann sich der Kondensator relativ schnell über R_5 und R_6 entla-

den, wenn keine Versorgungsspannung anliegt (vereinfachte Betrachtung, Zweig über den Transistor T_1 nicht berücksichtigt). Die Zeitkonstante beim Entladen beträgt dann knapp 9 s und die Verlustleistung am Widerstand R_6 im Betrieb liegt bei Nennspannung etwa bei 10 mW (etwa 1% der zulässigen Leistung des Spannungswandlers). Der Widerstand kann einfach auf der Kupferseite der bestehenden Platine angelötet werden.

$$\tau = (R_5 + R_6) * C_1 = (22\ k\Omega + 4{,}7\ k\Omega) * 330\ \mu F = 8{,}8\ s$$

$$P_{V\_R} = \frac{V_{Out}^2}{R_6} = \frac{(15\ V)^2}{22\ k\Omega} = 10{,}2\ mW$$

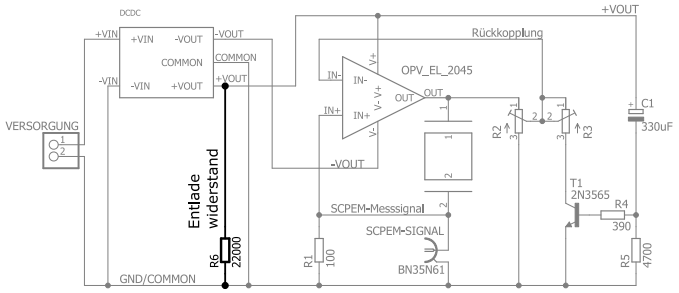


Abbildung 10.1: Schaltplan mit Kondensator-Entladewiderstand R_6 (Anh. B.3.2.3)

10.2.2 Kennlinie zwischen SCPEM-Stromsignal und Retardation verwenden

Die Retardation muss im Messprogramm bei einer definierten, eingestellten Polarisation berechnet werden (Kap. 3.2.3.2).

Es herrscht ein Zusammenhang zwischen SCPEM-Strom und Retardation, daher kann eine Kennlinie erstellt werden, aus der das Messprogramm sehr einfach mit der Amplitude des SCPEM-Stroms die max. Retardierung interpolieren kann.

Solange keine Änderungen am Aufbau vorgenommen werden, ist dann keine Retardationsmessung vor der Aufnahme nötig und eine Änderung der Retardation während der Aufnahme wird automatisch erfasst und berücksichtigt.

10.2.3 Zeitlicher Verlauf des Spannungs-Offsets berücksichtigen

Wenn die Photodiode abgedunkelt wird oder sich nicht im Strahlungsbereich befindet, bleibt ein kleines Intensitätssignals mit der Frequenz des SCPEM übrig. Dieses wird durch den Messaufbau induziert und lässt sich möglicherweise durch bessere Abschirmung aller Komponenten weiter reduzieren.

Zusätzlich besteht die Möglichkeit, diese Offsetkurve bei der Ermittlung der Intensitäts-Extremwerte zu berücksichtigen. Es wird für jede der vier Extremwertpositionen ein eigener Offset bestimmt, anstatt jeweils den gleichen zu verwenden.

10.2.4 Breiterer Kristall zur Erhöhung der möglichen Retardation

Die Erhöhung der Wellenlänge der betrachteten Strahlung hat zu einer Verringerung der möglichen Retardation mit dem verwendetem SCPEM geführt (Kap. 4.2.5).

Die Strahllänge im Kristall ist proportional zur möglichen Retardation, bei einem dickeren Kristall kann sie dadurch erhöht werden. Sind die restlichen Abmessungen gleich, bleiben Ansteuerungsspannung und Resonanzfrequenz gleich, der SCPEM-Strom erhöht sich ebenfalls proportional dazu.

10.2.5 Dezimation 8 statt Dezimation 64

Für den verwendeten SCPEM mit 80 kHz Resonanzfrequenz ist die Samplerate bei Dezimation 64 ausreichend. Kap. 7.2.2.3 zeigt, dass es zeitlich theoretisch möglich sein sollte, im Messprogramm die FPGA-Einstellung Dezimation 8 statt Dezimation 64 zu verwenden. Wenn kleinere SCPEM mit deutlich höherer Resonanzfrequenz verwendet werden sollen, kann diese Änderung in Betracht gezogen werden. Nach Tab. 7.2 ist dabei mit höherem Signalrauschen zu rechnen.

10.2.6 Flüssigkristall-Element als Analysator

Bestimmte Polarisationsrichtungen lassen sich abhängig vom Analysatorwinkel nicht oder nur sehr schlecht ermitteln (Kap. 3.3). Bei Verwendung eines Flüssigkristall-Elements kann durch Verwendung mehrerer überlappender Messbereiche möglicherweise eine Verbesserung erzielt werden. Nähert sich die Polarisation einem ungünstigen Wert, wird der Messbereich auf einen besser geeigneten umgeschaltet.

10.3 Ausblick

Durch die Änderungen am Messaufbau und der Schaffung einer funktionsfähigen Auswertemöglichkeit sind einige Bausteine (Abb. 1.4) des in Abb. 1.3 gezeigten Gesamtsystems zur Qualitätsverbesserung beim Laserschneiden vorangebracht worden. Für die beschriebene Anwendung fehlen noch einige Schritte und Entwicklungen.

Die nächste Aufgabe ist die erfolgreiche Messung der Polarisation, wenn für die Strahlung kein Polarisator verwendet wird, sondern eine heiße Oberfläche mit einem schrägen Abstrahlwinkel. Zusätzlich ist die Untersuchung des Polarisationsgrads dieser Strahlung für den Berechnungsalgorithmus von Interesse (Kap. 3.3.4, Kap. 3.3.4).

Ein weiterer großer Schritt ist die Anbringung des Systems am Laserschneidkopf, um die Strahlung direkt vom Schneidspalt über ein optisches System in das Messsystem zu leiten. Dafür werden konstruktive Maßnahmen nötig sein, um eine ausreichende Genauigkeit der Positionierung der optischen Komponenten zu ermöglichen, zusätzlich ist eine bessere elektromagnetische Schirmung im Bereich Photodiode-Verstärker sinnvoll.

Wenn damit die Neigung der Fläche an der Schneidfront für gerade Schnitte abgeschätzt werden kann, sollte dies als nächstes für beliebige Schneidrichtungen ermöglicht werden. Zur Ermittlung der aktuellen Richtung ist zusätzlich die Kommunikation zwischen Bearbeitungsmaschine und Messgerät nötig.

Der letzte Schritt ist die Auswertung der Polarisation und daraus folgend die Regelung verschiedener Parameter der Laserbearbeitungsmaschine während des Schnitts zur Optimierung des Schneidprozesses und des Bearbeitungsergebnisses.

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

ADC	Analog-Digital-Converter
API	Application Programming Interface, Programmierschnittstelle
ARM	Advanced RISC Machines
BNC	Bayonet Neill Concelman
CPU	Central Processing Unit, Prozessor
CSV	Comma-separated Values
DC	Direct Current, Gleichstrom
DNS	Domain Name System
DOP	Degree Of Polarisation, Polarisationsgrad
FPGA	Field Programmable Gate Array
GND	Ground, Masse
GUI	Graphical User Interface, grafische Benutzeroberfläche
IP	Internet Protocol
IR	Infrarot
LAN	Local Area Network, lokales Netzwerk
NIR	Nahes Infrarot
OPV	Operationsverstärker
PCI-E	Peripheral Component Interconnect Express
RAM	Random-Access Memory
RF	Radio Frequency, Hochfrequenz
SCP	Secure Copy
SCPEM	Single Crystal Photo Elastic Modulator
SCPI	Standard Commands for Programmable Instruments
SD	Secure Digital
SDR	Software Defined Radio (RF-Sender/-Empfänger auf Softwarebasis)
SSD	Solid-State-Drive, Halbleiterlaufwerk
SSH	Secure Shell
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
USB	Universal Serial Bus
WLAN	Wireless Local Area Network, drahtloses lokales Netzwerk
XML	Extensible Markup Language, erweiterbare Auszeichnungssprache

Literaturverzeichnis

[1] A. Gropp, J. Hutfless, and S. et al. Schuberth. *Opt Quant Electron (1995) 27: 1257*. 1995.

[2] Christoph Deininger. Prozessüberwachung in der lasermaterialbearbeitung – qualifizierung von verfahren und systemtechnik. *Laser Technik Journal*, 3(4):29–32, sep 2006.

[3] Thomas Pauger. Diodenbasierte prozessüberwachung eines remote-schneidprozesses basierend auf einer polarisationsmessung. Master's thesis, TU Wien, 2013.

[4] Schröder/Treiber. *Technische Optik*. Vogel, 10 edition, 2007.

[5] Robert Siegel, John R. Howell, and Joachim Lohrengel. *Wärmeübertragung durch Strahlung: Teil 1 Grundlagen und Materialeigenschaften (Wärme- und Stoffübertragung)*. Springer, 1988.

[6] Edward Collett. *Field Guide to Polarization (SPIE Vol. FG05)*. SPIE Publications, 2005.

[7] F. Bammer, T. Schumi, and R. Petkovsek. A new material for single crystal modulators: BBO. In Joachim Hein, Luis O. Silva, Georg Korn, Leonida A. Gizzi, and Chris Edwards, editors, *Diode-Pumped High Energy and High Power Lasers*. SPIE, may 2011.

[8] F. Bammer and R. Petkovsek. Scpem-based polarization modulation ellipsometry in the nir. volume 8169, pages The Society of Photo–Optical Instrumentation Engineers (SPIE); Communaute Urbaine Marseille Provence Metropole; Ville de Marseille –, Marseille, France, 2011. Analog to digital converters;Cost-effective solutions;Ellipsometric measurements;LiTaO3;Lock-in amplifier;On-board processing;Photoelastic modulator;Polarization modulation;Sampling rates;Signal measurement;Upper limits;.

[9] R. Petkovšek, Jaka Petelin, J. Možina, and F. Bammer. Fast ellipsometric measurements based on a single crystal photo-elastic modulator. *Optics Express*, 18(20):21410, sep 2010.

[10] Dong e Zhao, You hua Chen, Zhi bin Wang, Yuan yuan Chen, Li fu Wang, and Rui Zhang. Study on signal crystal photo-elastic modulator based on lithium niobate piezoelectric and photo-elastic effect. In *2012 Symposium on Piezoelectricity, Acoustic Waves, and Device Applications (SPAWDA)*. IEEE, nov 2012.

[11] ANDRÉS SIO SEVER. Development of a system to measure the polarization state of thermal radiation with the use of single crystal photo elastic modulators. Master's thesis, TU Wien, 2018.

[12] F. Bammer, R. Petkovšek, J. Možina, and J. Petelin. A small and fast SCPEM-based ellipsometer. In Shibin Jiang, Michel J. F. Digonnet, John W. Glesener, and J. Christopher Dries, editors, *Optical Components and Materials VII*. SPIE, feb 2010.

[13] Ferdinand Bammer, Bernhard Holzinger, and Thomas Schumi. Time multiplexing of high power laser diodes with single crystal photo-elastic modulators. *Optics Express*, 14(8):3324, apr 2006.

[14] F. Bammer and R. Petkovsek. Q-switching with single crystal photo-elastic modulators. In Vladislav Panchenko, Gérard Mourou, and Aleksei M. Zheltikov, editors, *LAT 2010: International Conference on Lasers, Applications, and Technologies*. SPIE, sep 2010.

[15] F. Bammer, R. Petkovšek, H. Dominguez, and G. Liedl. Nd:YAG-laser-q-switching with a photo-elastic modulator and applications. In Thomas Graf, Jacob I. Mackenzie, Helena Jelinková, Gerhard G. Paulus, Vincent Bagnoud, and Catherine Le Blanc, editors, *Solid State Lasers and Amplifiers IV, and High-Power Lasers*. SPIE, apr 2010.

[16] J. Bachmair, R. Gómez Vázquez, and F. Bammer. A simple pulsed laser design for high harmonics generation. In Kerim R. Allakhverdiev, editor, *XIX International Symposium on High-Power Laser Systems and Applications 2012*. SPIE, jan 2013.

[17] Rok Petkovšek, Vid Novak, Ferdinand Bammer, Janez Možina, and Boštjan Podobnik. Power scaling of AOM-switched lasers with SCPEM-based time-multiplexing. *Optics Express*, 19(21):19855, sep 2011.

[18] RedPitaya. Redpitaya, October 2018. https://www.redpitaya.com/ (03.10.2018).

[19] RedPitaya. *Red Pitaya STEMlab Documentation Release 0.97*. RedPitaya, June 2018. https://redpitaya.readthedocs.io/en/latest/ (23.07.2018).

[20] Moritz Kütt, Malte Göttsche, and Alexander Glaser. Information barrier experimental: Toward a trusted and open-source computing platform for nuclear warhead verification. *Measurement*, 114:185–190, jan 2018.

[21] A. C. Cardenas Olaya, S. Micalizio, M. Ortolano, C. E. Calosso, E. Rubiola, and J-M. Friedt. Digital electronics based on red pitaya platform for coherent fiber links. In *2016 European Frequency and Time Forum (EFTF)*. IEEE, apr 2016.

[22] Tektronix. *TekVISA Reference Card v3.0*. Tektronix, September 2006. https://www.tek.com/oscilloscope/tds7054-software/tekvisa-connectivity-software-v411 (18.01.2018).

[23] John Garney and Jon Lueker. Usb 2.0 specification released on april 27, 2000, April 2000. https://www.usb.org/document-library/usb-20-specification-released-april-27-2000 (14.09.2018).

[24] Mattias Nilsson. Gigabit ethernet over category 5 unshielded twisted pair cable evaluating electrical characteristics of 1000base-t physical layer devices. Master's thesis, Chalmers University of Technology, Gothenburg, Sweden, 200.

[25] Erstellen von netzwerkanwendungen mit der udp support library in labwindows™/cvi. *National Instruments Whitepaper*, January 2009.

[26] Jon Postel. Transmission control protocol darpa internet program protocol specification, April 1981. https://tools.ietf.org/html/rfc793 (30.08.2018).

[27] J. Postel. User datagram protocol, April 1980. https://www.ietf.org/rfc/rfc768.txt (30.08.2018).

[28] Pyredpitaya, January 2016. https://github.com/clade/RedPitaya/tree/master/python (10.04.2018).

[29] RedPitaya. *Red Pitaya User Manual Release 1.0*. RedPitaya, July 2017.

[30] Malcolm Smith. Gui programming, September 2018. https://wiki.python.org/moin/GuiProgramming (10.04.2018).

[31] Gui programming in python, September 2018. https://wiki.python.org/moin/GUI Programming in Python (10.04.2018).

[32] Don Rozenberg. Page - python automatic gui generator - version 4.17. http://page.sourceforge.net/ (16.04.2018).

[33] Adrian Autalan. A simple gui designer for the python tkinter module. https://github.com/alejandroautalan/pygubu (16.04.2018).

[34] Visual tkinter python ide, April 2013. https://sourceforge.net/projects/visualtkinter/ (16.04.2018).

[35] Introducing 1.1.1.1. https://1.1.1.1/ (04.11.2018).

[36] Google public dns. https://developers.google.com/speed/public-dns/ (04.11.2018).

[37] Putty. https://www.putty.org/ (12.01.2018).

[38] Winscp. https://winscp.net/ (12.01.18).

[39] Numpy. http://www.numpy.org/ (03.18.2018).

[40] matplotlib. https://matplotlib.org/index.html (12.07.2018).

[41] Tkinter documentation, November 2017. https://wiki.python.org/moin/TkInter (17.05.18).

[42] Mark Roseman. Tkdocs. https://tkdocs.com/index.html (21.05.18).

[43] Pygubu 0.9.8.2 project description. https://pypi.org/project/pygubu/ (16.05.2018).

[44] Jeff Forcier. Welcome to paramiko! http://www.paramiko.org/ (09.04.2018).

Anhang

Alle Anhänge sind dieser Arbeit mit der folgenden Ordnerstruktur auf einem digitalen Speichermedium beigelegt.

A. Datenblätter und Dokumente

A.1. Geräte
- A.1.1. Oszilloskop Tektronix 2014C
- A.1.2. Multimeter ISO-TECH IDM71

A.2. Komponenten
- A.2.1. Infrarotemitter Helioworks EK-8620 Datenblatt
- A.2.2. Interferenzfilter bk-1500-090-B
  - A.2.2.1. Interferenzfilter bk-1500-090-B Datenblatt
  - A.2.2.2. Interferenzfilter bk-1500-090-B Prüfbericht
- A.2.3. Photodiode Hamamatsu G12180-010A Datenblatt
- A.2.4. Transistor ON Semiconductor MPSA42 Datenblatt
- A.2.5. Operationsverstärker Renesas EL2045 Datenblatt
- A.2.6. DCDC-Converter Traco Power TMH 2415D Datenblatt

A.3. Dokumente
- A.3.1. Erstellen von Netzwerkanwendungen mit der UDP Support Library
- A.3.2. RedPitaya STEMlab-Dokumentation
  - A.3.2.1. RedPitaya STEMlab Documentation, Release 0.97
  - A.3.2.2. RedPitaya User Manual, Release 1.0

B. Daten (Messungen, Simulationen)

B.1. Messungen, Aufnahmen, Testprogramme
- B.1.1. Testmessungen Oszilloskop 605 nm
- B.1.2. Testmessungen Oszilloskop 1500 nm (Störung)
- B.1.3. Testmessungen Oszilloskop 1500 nm
- B.1.4. Testmessungen Oszilloskop 1500 nm (Fehler)
- B.1.5. Kennlinie Spannungswandler
- B.1.6. Dateneinlesedauer SCPI
  - B.1.6.1. MATLAB
  - B.1.6.2. Python
- B.1.7. UDP-Performance für unterschiedliche Programmierumgebungen
  - B.1.7.1. MATLAB