

Masterarbeit

**Entwurf und Test einer Regelung zur reinharmonischen Krafterregung
nicht-linearer mechanischer Systeme**

Felix Piela
310361

6. Mai 2016

TECHNISCHE UNIVERSITÄT BERLIN

Institut für Mechanik
FG Strukturmechanik und Strukturberechnung
Univ.-Prof. Dr.-Ing. habil. Manfred Zehn
Dipl.-Ing. Tobias Rademacher

Institut für Prozess- und Verfahrenstechnik
FG Mess- und Regelungstechnik
Prof. Dr.-Ing. habil. Rudibert King
M.Sc. Matthias Kiesner

Eidesstattliche Erklärung

Die selbständige und eigenhändige Anfertigung versichere ich an Eides statt.

Berlin, den 13. April 2016

Unterschrift

Inhaltsverzeichnis

1 Motivation	11
2 Aufgabenstellung	15
2.1 Hintergrund	15
2.2 Zielsetzung	16
2.3 Anforderungen & Schnittstellen	16
3 Grundlagen zur Regelungstechnik	19
3.1 Mathematische Grundlagen	19
3.2 Klassische und neuere Regelungsansätze	22
3.3 Iterativ lernende Regelung	24
3.4 Betrachtungen zur Stabilität	27
4 Planung und Auslegung der Regelung	29
4.1 Beschreibung des Versuchsaufbaus	29
4.2 Vorhandene Hardware	33
4.3 Regelungsansätze und Arbeitsweise des Reglers	34
4.4 Integration in den Versuchsaufbau	36
4.5 Hardware für die Regelung	38
4.6 Bewertung der Regelungsansätze und Entscheidung für ein Vorgehen	39
5 Umsetzung der Regelung	41
5.1 Implementierung	41
5.2 Elektronik und Integration in den Versuchsaufbau	46
5.3 Schnittstellen zum bestehenden Messaufbau	50
6 Test, Verifikation und Bewertung der Regelung	55
6.1 Durchführung der Messungen	55
6.2 Konvergenzverhalten der Lerngesetze	57
6.3 Bewertung der Regelgüte	58
6.4 Ergebnisse Messungen mit Glasfaserplatte	61
6.5 Ergebnisse Messungen Schiene	66
7 Fazit und Ausblick	75
7.1 Zusammenfassung	75
7.2 Verbesserungsideen	75
7.3 Ausblick	76
8 Anhang	93
8.1 Arduino Code	93
8.2 MATLAB® Code	102
8.3 Datenblätter, Begleitinformationen	105

Abstract

Complex mechanic structures with highly non-linear behaviour are increasingly examined in relation to their vibration and dynamic response behaviour. These types of structures include among others fibre-composites and absorbers. With the increased use of fibre-composites in industrial series production especially in the fields of automotive and aviation comes an increased demand for reliable non-destructive testing procedures for quality assurance during production and utilisation. Currently evolving testing procedures include methods evolved from modal testing to detect and locate defects. These methods demand high standards for the quality of the exciting signal.

Subject of the presented thesis is the excitation of non-linear structural components with a pure harmonic behaviour of the excitation force. The response movement of a non-linear structure may contain frequencies not included in the excitation signal. For these frequencies, the mounting point of the actuator acts as a fixation with the actuators momentum distorting the dynamics of the structure under examination. Therefore undesired frequencies in the excitation force can be observed. The goal of this thesis is to eliminate these undesired frequencies in the excitation force to ensure a controlled pure harmonic excitation.

Many approaches were considered when preparing this thesis. A solution was favoured that would easily integrate into the existing measurement setup while ensuring the best possible result regarding the accuracy of the signal. At the same time, it was tried to use low priced standard components. A closed loop control using iterative learning control (ILC) algorithm was selected and realised using an Arduino Due board as a hardware platform. This board features one ADC and one DAC with a resolution of 12 bit. The firmware was implemented complying with hard real-time requirements to achieve the best timing for both the signal period and the sample rate. Additionally, a surrounding electric circuit was designed to provide the necessary interfaces to the existing setup including an IEPE compliant interface for the piezo sensor used to measure the excitation force that has an integrated charge amplifier. Extensive tests were performed to confirm the mode of operation.

The results of the tests show that the iterative learning control proofs to be working correctly. The closed-loop control can completely eliminate undesired frequencies in the excitation force resulting from the structure's non-linearities. The overall quality of the control algorithm was assessed using the concept of total harmonic distortion (THD). The total harmonic distortion could be reduced to $\text{THD} < 4 \cdot 10^{-4}$ when only higher modes of the base frequency and to $\text{THD} < 5 \cdot 10^{-3}$ when all other noise frequencies were considered. This is close to the best possible results using a 12 bit analog input and output. A huge improvement of several orders of magnitude compared to the open loop control can be achieved. As outlook, a sweep experiment was successfully performed passing through a resonance frequency. The results are very promising and further testing and evaluation should be performed with different structures and update laws.

Zusammenfassung

Komplexe mechanische Strukturen mit starken nicht-linearen Eigenschaften wie zum Beispiel Faserverbundmaterialien oder Schwingungsdämpfer stehen immer öfter im Fokus schwingungsmess-technischer und strukturdynamischer Untersuchungen. Mit der steigenden Verbreitung von Faser-verbundwerkstoffen insbesondere in den Bereichen des Fahrzeugbaus und der Luftfahrt steigt der Bedarf an zuverlässigen zerstörungsfreien Prüfverfahren zur Qualitätssicherung bei der Produktion und während der Verwendung. Derzeit befinden sich unter anderem Verfahren in der Entwicklung, die aufbauend auf der modalen Analyse Fehler detektieren und lokalisieren sollen. Bei diesen Ver-fahren kommt der Qualität des anregenden Signals eine maßgebliche Bedeutung zu.

In der vorliegenden Arbeit wurden Möglichkeiten untersucht, für die Schwingungsmesstechnik an nicht-linearen Strukturen eine Anregung mit einem reinharmonischen Kraftverlauf zu erreichen. Das Antwortspektrum nicht-linearer Strukturen beinhaltet neben der Anregungsfrequenz auch zahlreiche andere Frequenzen, für die der Aktuator eine feste Einspannung darstellt. Da der Aktua-tor ebenfalls eine Eigendynamik aufweist, entstehen ungewollte Kraftverläufe zwischen Aktuator und Struktur. Ziel dieser Arbeit ist es, diese ungewollten Frequenzen im Kraftverlauf zu eliminieren.

In dieser Arbeit wurden verschiedene Ansätze für eine Regelung des Kraftverlaufs der Anregung un-tersucht, wobei die Integration in einen bestehenden Versuchsaufbau und die erreichbare Regelgüte die maßgeblichen Bewertungskriterien waren. Eine Realisierung mit günstiger, handelsüblicher Hardware wurde angestrebt. Als Ansatz für einen geschlossenen Regelkreis für den Kraftverlauf wurde eine iterativ lernende Regelung (ILR) ausgewählt. Als Plattform für die Realisierung der Regelung wurde ein Arduino Due mit integriertem ADC und DAC mit je 12 bit Auflösung aus gewählt. Die Implementierung des Regelalgoritmus wurde unter Berücksichtigung harter Echt-zeitanforderungen realisiert, um präzises Zeitverhalten im Hinblick auf die Frequenz des Signals und der Sample-Rate zu erreichen. Außerdem wurde ein elektrischer Schaltkreis entworfen und umgesetzt, der die Integration in den bestehenden Versuchsaufbau ermöglicht. Insbesondere wurde hier eine Schnittstelle zum integrierten Ladungsverstärker des für die Kraftmessung verwendeten Piezo-Sensors nach dem IEPE-Standard realisiert. Somit konnten Versuche zur Funktionsweise des Reglers erfolgreich durchgeführt werden.

Die Versuchsergebnisse zeigen, dass die höherharmonischen Störungen, die bei der herkömmli-chen Ansteuerung zwischen Aktuator und nicht-linearer Struktur auftreten, durch die Regelung vollständig eliminiert werden. Die Regelgüte wurde mittels der harmonischen Verzerrung (THD) bewertet; sie konnte auf $\text{THD} < 4 \cdot 10^{-4}$ reduziert werden, wenn nur höherharmonische Schwin-gungen und auf $\text{THD} < 5 \cdot 10^{-3}$, wenn alle anderen Signalanteile betrachtet werden. Als Aus-blick wurde ein Sweep-Experiment durch eine Resonanzfrequenz mit vielversprechendem Ergebnis durchgeführt. Für Sweep-Experimente sind weitere Untersuchungen im Hinblick auf unterschied-liche Versuchskonfigurationen und Lerngesetze der ILR nötig.

1 Motivation

Komplexe mechanische Systeme mit nicht-linearen Eigenschaften stehen zunehmend im Fokus schwingungsmesstechnischer Untersuchungen. Hohe Anforderungen an Leichtbau oder Schwingungsverhalten der Strukturen und ein hohes Maß an Funktionsintegration führen zu einer steigenden Komplexität der Strukturen. Lineare Eigenschaften der Strukturen können mittlerweile numerisch und experimentell untersucht und vorhergesagt werden. Häufig sind aber auch nicht-lineare Eigenschaften von nicht zu vernachlässigender Bedeutung, daher ist derzeit ihre Untersuchung Gegenstand zahlreicher wissenschaftlicher Arbeiten. Strukturen mit ausgeprägtem nicht-linearem Verhalten können unter anderem Bauteile aus Faserverbundwerkstoffen, komplizierte Blechkonstruktionen wie Fahrzeugkarosserien oder integrierte Dämpfungselemente sein.

Faserverbundwerkstoffe finden zunehmende Verbreitung vor allem in Bereichen wie dem Kraftfahrzeugbau oder der Luftfahrt [41, 54, 7, 48]. Sie ermöglichen viele Freiheitsgrade bei der Auslegung von Bauteilen unter Optimierung des Kraftverlaufs. Dadurch können Bauteile gewichtsoptimiert und gleichzeitig haltbar realisiert werden. Auch im allgemeinen Maschinenbau finden Faserverbundwerkstoffe mehr und mehr Verwendung, vor allem wenn Gewichtsoptimierung oder eine große Dynamik der Maschinen im Vordergrund steht [23].

Teilweise sind die technischen Möglichkeiten der Produktion den Prüfverfahren zur Qualitätssicherung weit voraus. So wäre es an vielen Stellen bereits möglich, Faserverbundwerkstoffe Gewinn bringend einzusetzen, allerdings existieren in einigen Bereichen bislang nur mangelhafte Zulassungsverfahren, Zertifizierungen für den Herstellungsprozess oder Richtlinien für die regelmäßige Kontrolle der Bauteile im Einsatz. Neben der Sicherstellung des Fertigungsprozesses gemäß der Vorgaben zur Faserrichtung und zum Füllungsgrad, Matrix, Aushärteprozess usw. sind ein kompliziertes Materialverhalten dabei die hauptsächliche Schwierigkeit. Faserverbundwerkstoffe weisen häufig im gesamten zulässigen Dehnungsbereich eine ausgeprägte Wölbung der Spannungs-Dehnungs-Kurve auf, sodass die Anwendung des Hookeschen Gesetzes nicht ausreichend exakt ist [48, S.203]. Durch die Richtungen der eingebrachten Fasern ergibt sich ein stark anisotropes Materialverhalten. Bei starken Verformungen verändern sich die Richtungen der Fasern, wodurch die Anisotropie selbst wiederum eine Abhängigkeit zur Verformung aufweist [48, S.227]. Zudem weist die Matrix die bei Kunststoffen bekannte geschwindigkeitsabhängigen Materialeigenschaften auf. Dies wird bei der Schwingungsanalyse von Faserverbundwerkstoffen meist in Form einer frequenzabhängigen Dämpfung berücksichtigt. Außerdem wurde beobachtet, dass die zeitliche Reihenfolge der unterschiedlichen Belastungsrichtungen einen Einfluss auf die Spannungszustände im Material haben [19] und somit bei Schwingungen auch der Phasenwinkel zwischen Schub- und Biegebelastung eine entscheidende Rolle spielt. Diese und weitere Einflüsse erschweren die analytische und numerische Auslegung von Bauteilen aus Faserverbundwerkstoffen. Des weiteren besteht noch keine ausreichende Kenntnis über die Alterungseigenschaften von Faserverbundwerkstoffen, insbesondere unter andauernder Wechselbelastung, schwankenden Temperaturen, Vibrationen etc. In diesem Zusammenhang sei insbesondere der Effekt der Delamination unter lokaler Beschädigung oder Dauerwechselbelastung erwähnt.

Um den genannten Schwierigkeiten bei der Herstellung und Verwendung von Faserverbundwerkstoffen zu begegnen, sind Prüfverfahren von Faserverbundwerkstoffen Gegenstand zahlreicher Forschungsprojekte [41, 17]. Dabei werden sowohl Bauteile im Auslieferungszustand untersucht, als auch eine regelmäßige Untersuchung der Bauteile im Betrieb zur Überprüfung angestrebt. Ein Verfahren zur zerstörungsfreien Prüfung von Strukturbau Teilen verwendet die Modalanalyse zur Detektion und Lokalisierung von Defekten [47]. Ausgangspunkt für die vorliegende Arbeit war

die Forschungsarbeit an einem Prüfverfahren, dass die Veränderung von Nicht–Linearitäten in der Schwingungsantwort bei Anregung in einer Resonanzfrequenz zur Lokalisierung von Schäden verwendet [45]. Bei diesem Prüfverfahren werden mechanische Bauteile mit einem möglichst reinharmonischen Kraftverlauf angeregt. Durch die Untersuchung bei unterschiedlichen Frequenzen und Auswertung der korrespondierenden Bauteilschwingungen und Nicht–Linearitäten können detaillierte Informationen über das Bauteil und einen möglichen Schadenszustand gewonnen werden.

Bei der Untersuchung der Charakteristik und Wirkungsweise von Dämpfungselementen können nicht–lineare Eigenschaften ebenfalls von entscheidender Bedeutung sein.

Nicht–Linearitäten können in drei Arten unterteilt werden: geometrische, materielle und zeitlich nicht–glatte. Geometrische Nicht–Linearitäten treten bei großen Auslenkungen auf, wenn der Spannungszustand maßgeblich durch die Auslenkung beeinflusst wird. Materielle nicht–Linearitäten beinhalten die Einflüsse der oben erwähnten nicht–linearen Spannungs–Dehnungs–Kurven oder zeitabhängiges Verhalten der Materialien wie zum Beispiel eine Hysterese. Zeitlich nicht glatte Probleme treten auf, wenn die Eigenschaften des Systems nur stückweise stetig sind. Dies kann zum Beispiel bei Aufliegen zweier Bauteile sein, zwischen denen unter Zugbelastung eine Lücke aufklaffen kann. Die Steifigkeit des Systems erlebt dann je nach Belastung und Verformung einen Sprung von einem weicheren Zustand wenn die Lücke aufklafft zu einem steiferen Zustand wenn die Lücke geschlossen ist. In der vorliegenden Arbeit wurde eine Glasfaserplatte untersucht, die beobachteten Nicht–Linearitäten sind materieller Art. Außerdem wurde ein Schienendämpfer untersucht, der neben materiellen Nicht–Linearitäten auch nicht–glatte Phänomene durch die Einspannung an der Schiene aufweist.

In der Regel wird in der Schwingungsmesstechnik ein Signalverlauf für die Anregung genau vorgegeben. Der Leistungsverstärker und Aktuator haben aber eine Eigendynamik, die wenig bis gar nicht bekannt ist oder nur teilweise im Signalverlauf berücksichtigt wird. Außerdem besteht zwischen dem Aktuator und der zu untersuchenden Struktur eine Einspannung, die insbesondere bei nicht–linearen Strukturen durch die Wechselwirkung zwischen Struktur und Aktuator an der Einspannung die Struktur erheblich beeinflusst. Dies führt dazu, dass der gewünschte Verlauf für die Anregung nicht exakt getroffen wird. Um dem Rechnung zu tragen wird häufig neben dem Signalverlauf an den Aktuator auch die physikalische Anregungsgröße (z.B. Kraftverlauf) während der Messung aufgezeichnet, um nicht nur die gewünschte, sondern auch die reale Anregung zu kennen. Bei der Auswertung der Versuchsergebnisse wird dann nicht die gewünschte, sondern die reale Anregung verwendet.

Unter der Annahme, dass linear elastisches Verhalten vorliegt genügt meist die genaue Kenntnis der Anregung. Im Falle von nicht–linearem Materialverhalten allerdings kommt dem genauen zeitlichen Verlauf der Anregung eine zentrale Bedeutung zu. Da das Verstärkungs– und Superpositionsprinzip nicht mehr anwendbar sind, sind die Messungen stets nur für einen bestimmten Arbeitspunkt d.h. eine bestimmte Frequenz und Amplitude und die Trajektorie zu diesem Arbeitspunkt gültig und es kann nicht zwischen Überlagerungen mit anderen Arbeitspunkten oder Frequenzen unterschieden werden. Daher ist für die Analyse nicht–linearer Bauteile eine Regelung wünschenswert, die ermöglicht, den vorgegebenen Verlauf der Anregung exakt zu treffen. Bei Schwingungs–Versuchen mit nicht–linearen Strukturen mit harmonischer Anregung durch einen Shaker treten zahlreiche höherharmonische Störungen auf, die eine exakte Auswertung der Messungen erheblich erschweren. Die grundlegende Idee hinter dieser Arbeit ist nun, einen geschlossenen Regelkreis aufzubauen, der diese Störungen kompensieren kann. Ansätze im Frequenzbereich wurden in [33] bereits verfolgt, allerdings ist ein gesamter Versuch eine Iterationen des Algorithmus, was eine lange Konvergenzzeit der Regelung verursacht. Außerdem wurde berichtet, dass dieses Verfahren in der Nähe von Resonanzfrequenzen keine Lösung für das Problem findet. Nach meinem Kenntnisstand ist eine geeignete Regelung mit hinreichend kurzer Konvergenzzeit und zuverlässiger Funktion auch in der Nähe der Resonanzfrequenzen derzeit nicht verfügbar.

Die vorliegende Arbeit beschäftigt sich mit dem Entwurf und der Implementierung eines geeigneten Regelansatzes, um einen bestimmten zeitlichen Anregungsverlauf exakt zu treffen, obwohl Leis-

tungsverstärker, Aktuator und Bauteil eine eigene Dynamik aufweisen, über die wenig Vorwissen bereit steht. Die Qualität integrierter Schaltungen ist soweit fortgeschritten, dass kostengünstige, handelsübliche ICs für die Realisierung in Frage kommen. Diese werden neben der kostenintensiven speziell entwickelten Messtechnik ebenfalls in Betracht gezogen.

2 Aufgabenstellung

2.1 Hintergrund

Bei Schwingungsuntersuchungen mechanischer Strukturen wird versucht, eine möglichst freie Lagerung zu realisieren. An einer Stelle wird eine zeitlich veränderliche Kraft aufgebracht, durch welche die Struktur in Schwingung versetzt wird. An der Antwort der Struktur auf die Anregung können wichtige Systemeigenschaften abgelesen werden.

Üblicherweise wird bei Schwingungsuntersuchungen linearer Systeme das Verstärkungs- und das Superpositionsprinzip angewendet. Aus einer Messung mit einer bestimmten Anregungs-Amplitude kann bei linearen Strukturen somit auch auf Systemantworten mit anderen Anregungs-Amplituden geschlossen werden. Dadurch ist die genaue Amplitude und der exakte zeitliche Verlauf einer Anregung nicht so entscheidend, so lange sie ausreichend genau vermessen werden können und bei der Auswertung zusammen mit der Systemantwort zur Verfügung stehen. Liegt ein Wert der Übertragungsfunktion für jede in der Anregung enthaltene Frequenz vor, so kann die Systemantwort aus dieser Übertragungsfunktion berechnet werden.

Die oben geschilderte Vorgehensweise findet vor allem in der sog. modalen Analyse Anwendung, mittels derer komplexe mechanische Strukturen auf ihre Eigenschaften untersucht werden können [15] [55, Kap. 6].

Bei nicht-linearen Strukturen kann sich das Verhalten bei Änderung der Amplitude oder bei Überlagerung mit anderen Frequenzen erheblich verändern. Es können Frequenzen in der Struktur beobachtet werden, die sich von der Anregungsfrequenz unterscheiden. Eine Messung ist somit nur für eine bestimmte Anregung gültig.

- Es kann aus einer Anregung mit einer Amplitude nicht auf Systemantworten mit einer anderen Anregungs-Amplitude geschlossen werden.
- Es kann aus zwei Messungen unterschiedlicher Anregungs-Frequenzen nicht auf die Systemantwort der überlagerten Anregungsfrequenzen geschlossen werden.

Aus dieser Tatsache folgen für den Messaufbau bei Schwingungsuntersuchungen nicht-linearer Strukturen einige praktische Schwierigkeiten:

- Als Anregung soll ein bestimmter Kraftverlauf vorgegeben werden. In der Nähe einer Resonanzfrequenz führt die Struktur auch bei relativ kleinen Anregungskräften große Bewegungen aus. Da der Aktuator aufgrund seiner Eigendynamik diese großen Bewegungen nicht nachvollzieht, wirkt die Befestigung mit dem Aktuator dann als eine zusätzliche Zwangsbeziehung, welche die Schwingung der Struktur einschränkt
- Wird durch Anregung mit einer bestimmten Frequenz eine Schwingung in einer anderen Frequenz hervor gerufen, so führt der Aktuator natürlich keine Schwingung mit dieser zweiten Frequenz aus. Für diese zweite Frequenz wirkt die Befestigung der Struktur am Aktuator ebenfalls als Zwangsbeziehung und schränkt die Bewegung der Struktur ein. Dadurch wird die Messung entscheidend beeinflusst.

2.2 Zielsetzung

Die Struktur soll mit einem bestimmten periodischen Kraftverlauf angeregt werden. Durch die Einspannung des Aktuators an der Struktur und die Eigendynamik des Aktuators kann jedoch eine unerwünschte Rückkopplung des Systems auf den Kraftverlauf entstehen, die für die Messung unerwünscht ist. Ein möglicher Ansatz zur Vermeidung dieser Zwangskräfte wäre die Vermeidung einer Einspannung zwischen Aktuator und Struktur, beispielsweise durch Ausnutzung magnetischer Effekte, pulsierender Druckluft o.ä. Dies würde auf die Entwicklung neuartiger Aktuatoren hinaus laufen, was mit erheblichem Aufwand verbunden wäre. Außerdem ist unklar, wie diese Aktuatoren in den Versuchsaufbau integriert werden könnten, ohne Schwingungen der Struktur zu beeinflussen. Deshalb wurde in dieser Arbeit der Ansatz verfolgt, den bestehenden Versuchsaufbau möglichst wenig zu verändern und die hochwertigen und erprobten Komponenten der bestehenden Messtechnik weiter zu verwenden. Vielmehr soll versucht werden, die negativen Effekte des bestehenden Systems durch einen geschlossenen Regelkreis zu unterdrücken.

Ziel der Regelung ist es, alle vom vorgegebenen Kraftverlauf abweichenden Kräfte zwischen Aktuator und Struktur möglichst stark zu unterdrücken.

Die Regelung soll möglichst einfach in einen bestehenden Messaufbau integrierbar sein.

Es muss davon ausgegangen werden, dass die Streckenübertragungsfunktion für die Auslegung oder Anpassung der Regelung nicht oder nur ungenügend zur Verfügung steht. Dies hat zwei Gründe:

- Die zu untersuchende Struktur ist vorerst unbekannt, da die Messungen ja gerade der Beschreibung der Struktur dienen.
- Die Regelung soll für verschiedene zu untersuchende mechanische Strukturen verwendbar sein. Idealerweise wäre die Regelung sogar für verschiedene Versuchsaufbauten anwendbar.

2.3 Anforderungen & Schnittstellen

Im folgenden werden Anforderungen gesammelt, um bei der Auslegung des Reglers und der Entwicklung der umgebenden Peripherie klar strukturiert vorgehen zu können. Eine klare und detaillierte Formulierung der Zielsetzung und der Nebenbedingungen ist maßgeblich für den Erfolg einer Entwicklung [22, 21].

Neben streng einzuhaltenden Anforderungen werden auch Wünsche gesammelt, die nicht zwingender Weise umgesetzt werden müssen. Anforderungen und Wünsche beinhalten konkrete messbare Funktionen. Die Reihenfolge der Anforderungen spiegelt grob ihre Priorisierung wider.

A | Anforderung: Integration in den bestehenden Messaufbau

Die Regelung muss in den bestehenden Messaufbau integrierbar sein

B | Anforderung: Frequenzbereich

Die Regelung muss in einem Frequenzbereich von 0Hz - 800Hz zuverlässig arbeiten.

C | Wunsch: erweiterter Frequenzbereich

Es wäre wünschenswert, wenn die Regelung bis zu einer Frequenz von 2000Hz oder mehr zuverlässig arbeitet.

D | Wunsch: günstige, handelsübliche Hardware

Es soll möglichst günstige, handelsübliche Hardware verwendet werden.

E | Anforderung: Obergrenze Kosten für Hardware

Die Kosten für die verwendete zusätzliche Hardware darf €300 nicht überschreiten

F | Anforderung: Sweep

Es muss möglich sein, die Frequenz zur Laufzeit in einem Bereich von min. $\pm 20\%$ ändern zu können.

G | Wunsch: Einstellung der Regelung

Nach Möglichkeit sollen die Parameter der Regelung zur Laufzeit änderbar sein.

- Änderung der Parameterwerte ohne Neustart bzw. Flashen
- Anpassung der Parameterwerte zur Laufzeit

Neben den Anforderungen und Wünschen gibt es Schnittstellendefinitionen, die wie die Anforderungen bindenden Charakter haben aber nicht-funktionaler Art sind. Schnittstellen lassen sich als Nebenbedingungen interpretieren.

H | Schnittstelle: Eingang

Zwei Varianten sind möglich, von denen nur eine oder beide realisiert werden können:

- a) Interne Erzeugung des Signalverlaufs: Informationen zum Signalverlauf werden auf abstrakter Ebene übergeben (z.B. Frequenz, Amplitude). Der genaue zeitliche Signalverlauf wird in der Komponente erzeugt. Die Informationen zum Signalverlauf müssen während der Laufzeit nicht verändert werden. Wie die Übergabe erfolgt ist nicht vorgegeben.
- b) Externe Erzeugung des Signalverlaufs: Eingang in den Regler ist eine analoge Spannung im Bereich von -10 V bis 10 V oder -1 V bis 1 V .

I | Schnittstelle: Ausgang

Ausgang des Reglers ist eine analoge Spannung im Bereich von -10V bis 10V . Die genaue Amplitude kann über die Verstärkung des Leistungsverstärkers für den Shaker angepasst werden und muss deshalb nicht einem genauen Pegel entsprechen.

J | Schnittstelle: Stromversorgung

Die Stromversorgung des Reglers soll idealerweise über eine bereits im übrigen Messaufbau verfügbare Stromquelle erfolgen. Dies könnte zum Beispiel sein:

- a) Netzspannung ($\sim 230\text{V}$)
- b) USB Stecker

Kapitel 2. Aufgabenstellung

Der Regler muss wahrscheinlich an einen bestimmten Messaufbau angepasst werden. Eine Anpassung des Reglers ist somit erforderlich, wenn er in anderer Umgebung eingesetzt werden soll.

K | Schnittstelle: Programmierbarkeit des Reglers

Der Regler soll über eine übliche Schnittstelle, z.B. USB oder RS232 wenn nötig auch unter Zuhilfenahme eines Programmieradapters programmierbar sein. Eine Programmierbarkeit des Reglers abgesehen von den Einstellmöglichkeiten aus Schnittstelle L und Wunsch G zur Laufzeit ist nicht vorgesehen.

L | Schnittstelle: Anpassung von Parametern

Die Parameter des Reglers sollen über eine übliche Schnittstelle, z.B. USB oder RS232 anpassbar sein. So können die Eigenschaften des Reglers aber auch sein Zustand, z.B. Start / Stop einfach eingestellt werden.

Siehe auch Anforderung F und Wunsch G

3 Grundlagen zur Regelungstechnik

Grundprinzip der Regelung ist die Rückkopplung einer gemessenen Größe eines dynamischen Systems auf einen geeigneten Eingang des Systems [58, 30]. Meist ist es das Ziel einer Regelung, auch unter Störungen eine oder mehrere Systemgrößen konstant zu halten oder einen bestimmten zeitlichen Verlauf abzubilden. Um eine mathematische Beschreibung des betrachteten Systems zu erreichen, wird es mit Hilfe eines Modells vereinfacht beschrieben und durch die Einführung einer Systemgrenze von der Umwelt abgetrennt.

Die Regelgröße $y(t)$ ist eine von einem Sensor gemessene Zustandsgröße der Regelstrecke. Diese wird über einen Soll–Istwert–Vergleich von der vorgegebenen Führungsgröße $w(t)$ abgezogen, wodurch als Differenz der Regelfehler $e(t)$ entsteht. Aus dem Regelfehler berechnet der Regler eine Stellgröße $u(t)$, welche als Eingangsgröße des zu regelnden Systems verstanden wird und so das System beeinflusst. Das zu regelnde System wird als Regelstrecke bezeichnet [26].

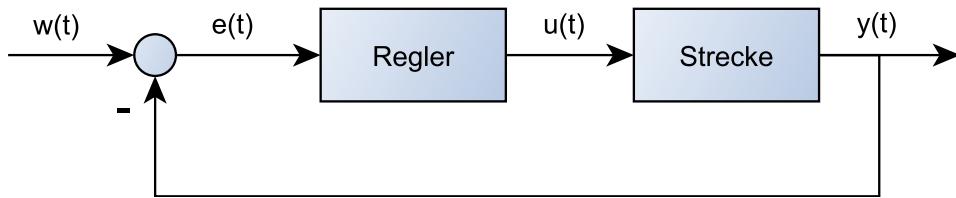


Abbildung 3.1: Blockschaltbild Standard–Regelkreis

In der Regelungstechnik werden Regelkreise meist durch Blockschaltbilder beschrieben. Blockschaltbilder folgen einer klar definierten Syntax (siehe z.B. [30, S. 41ff.]), wodurch sie äquivalent zu einer mathematischen Formulierung sind. In Abbildung 3.1 ist der oben beschriebene prinzipielle Aufbau einer Regelung in dieser Form dargestellt (vgl. Abbildung 4.3 in Kapitel 5 für die Übertragung auf den Versuchsaufbau).

3.1 Mathematische Grundlagen

Das zu regelnde dynamische System wird in der Regelungstechnik häufig in Form eines Modells approximiert, um anschließend analytische oder numerische Verfahren für die Auslegung und Analyse des Regelkreises anwenden zu können. Dabei führt die Modellbildung dynamischer Systeme zu Differentialgleichungen. Diese im Allgemeinen nicht-linearen Differentialgleichungen können durch Linearisierung um einen Arbeitspunkt lokal in lineare Differentialgleichungen überführt werden. Liegt das Modell der Regelstrecke in Form von linearen Differentialgleichungen vor, gibt es umfangreiche analytische und numerische Verfahren zur Reglerauslegung und Analyse des Regelkreises.

3.1.1 Zustandsraumdarstellung

Handelt es sich beim Modell der Regelstrecke um ein System aus Differentialgleichungen oder eine Differentialgleichung höherer Ordnung, bietet sich vor allem im linearen Fall die Darstellung im Zustandsraum an (siehe z.B. [30, S. 70ff.]). Eine Differentialgleichung höherer Ordnung wird

zunächst in einzelne Gleichungen erster Ordnung überführt. Das entstandene System aus Differentialgleichungen erster Ordnung wird dann in Vektorform umgestellt. Es entsteht ein Modell der Form

$$\dot{\vec{x}}(t) = A^* \vec{x}(t) + B^* \vec{u}(t) , \quad \vec{x}(0) = \vec{x}_0 \quad (3.1)$$

$$\vec{y}(t) = C^* \vec{x}(t) + D^* \vec{u}(t) . \quad (3.2)$$

Dabei beschreibt \vec{x} die Zustände des Systems. Gleichung 3.1 wird Systemgleichung genannt, Gleichung 3.2 heißt Messgleichung. Da die internen Zustände eines Systems nicht immer direkt gemessen werden können, beschreibt die Messgleichung die Abhängigkeit der Regelgröße $y(t)$ von den internen Zuständen $\vec{x}(t)$ des Systems und von der Stellgröße $u(t)$. Die Matrix beschreibt A^* die Dynamik der Regelstrecke, deshalb wird A^* auch Systemmatrix genannt. Der Einfluss der Stellgröße auf die Zustände wird durch die Steuermatrix B^* beschrieben. Mit der Beobachtungsmatrix C^* wird beschrieben, wie die Zustandsgrößen in den gemessenen Systemausgang \vec{y} eingehen und die Durchgangsmatrix D^* beschreibt einen möglichen direkten Einfluss der Steuergröße \vec{u} auf den Systemausgang \vec{y} . Je nach Dimension des Systems können die Matrizen B^* , C^* und D^* auch Vektoren bzw. Skalare sein. Um dies zu kennzeichnen, werden dann kleine Buchstaben, zum Beispiel b bzw. d verwendet. Um sie später besser von Parametern der zeitdiskreten Formulierung unterscheiden zu können, wurde hier in der zeitkontinuierlichen Formulierung der * als Kennzeichnung verwendet.

3.1.2 Laplace–Transformation

Mit Hilfe der Laplace–Transformation lassen sich lineare Differentialgleichungen in gebrochen–rationale Ausdrücke umwandeln. Wegen der Eindeutigkeit der Abbildung ist auch die Rücktransformation möglich. In der Regelungstechnik ist die Verwendung der Laplace–Transformation sehr verbreitet. Zum einen können die Differentialgleichungen im Laplace–Raum besonders einfach analytisch gelöst werden und durch anschließende Rücktransformation in den Zeitbereich kann die gewünschte Lösung im Zeitbereich erhalten werden. Zum anderen erleichtert die Laplace–Transformierte Form der Regelkreisglieder die Arbeit mit Blockschaltbildern, da eine Reihenschaltung von Regelkreisgliedern im Laplace–Bereich durch den Faltungssatz auf eine Multiplikation abgebildet wird. Dadurch können auch komplizierte Regler im Laplace–Bereich einfach berechnet werden und Einflüsse der einzelnen Regelkreisglieder sind viel leichter zu erkennen als in der Formulierung im Zeitbereich.

Der Formulierung von [30] folgend ist die Laplace–Transformierte einer Funktion $f(t)$ im Zeitbereich gegeben durch

$$F(s) = \int_0^\infty e^{-st} f(t) dt . \quad (3.3)$$

Meist werden große Buchstaben für die Laplace–Transformierten ihrer äquivalenten Funktionen im Zeitbereich verwendet. So ist $Y(s)$ die Transformierte von $y(t)$, $W(s)$ die Transformierte von $w(t)$ und so weiter.

Die Rücktransformation einer Funktion $F(s)$ im Laplace–Bereich zur korrespondierenden Funktion $f(t)$ im Zeitbereich ist gegeben durch

$$f(t) = \frac{1}{2\pi j} \int_{c-j\infty}^{c+j\infty} F(s) e^{st} ds \quad (3.4)$$

Meist wird die Transformation allerdings nicht analytisch ausgeführt, sondern die Funktion in einzelne Teile zerlegt, für die jeweils eine Lösung der Transformation oder Rücktransformation bekannt ist. Die Zerlegung erfolgt gemäß Rechenregeln, die aus den Eigenschaften der Laplace–Transformation resuliert [26, S. 73ff.].

Die Übertragungsfunktion der Strecke wird im Laplace–Bereich mit $G_S(s)$ und die Übertragungsfunktion des Reglers mit $G_R(s)$ bezeichnet. Übertragungsfunktion des offenen Regelkreises wird mit $G_0(s)$ bezeichnet, sie beschreibt die Eigenschaften von Regler und Strecke ohne Rückkopplungsweig.

$$G_0(s) = G_R(s) \cdot G_S(s) \quad (3.5)$$

Die Übertragungsfunktion bzgl. der Führungsgröße $w(t)$ des geschlossenen Regelkreises berechnet sich zu

$$G_w(s) = \frac{Y(s)}{W(s)} = \frac{G_0}{1 + G_0} = \frac{G_R(s) \cdot G_S(s)}{1 + G_R(s) \cdot G_S(s)} . \quad (3.6)$$

Ein wesentlicher Grund für die Verwendung eines Reglers ist oft nicht nur die Realisierung einer bestimmten Führungsgröße, sondern auch die Kompensation von äußereren Störungen. Diese Störungen können so modelliert werden, dass sie direkt auf den Systemausgang $y(t)$ einwirken. Die Übertragungsfunktion von Störungen ergibt sich dann zu

$$G_d(s) = \frac{Y(s)}{D(s)} = \frac{1}{1 + G_0(s)} = \frac{1}{1 + G_R(s) \cdot G_S(s)} . \quad (3.7)$$

Je nach Anwendung können beide Übertragungsfunktionen $G_w(s)$ und $G_d(s)$ Gegenstand der Reglerauslegung sein.

3.1.3 Fouriertransformation

Mittels der Fourieranalyse können Signalverläufe in ihre Frequenzanteile zerlegt werden. Häufig ist die Analyse von Signalen im Zeitbereich sehr schwierig, insbesondere wenn Schwingungen mit unterschiedlichen Amplituden und Frequenzen sowie Rauschen überlagert sind. Mittels der Fouriertransformation können Signale in die enthaltenen Frequenzen und deren Amplituden sowie Phasenwinkel zerlegt werden. Dies kann bei der Auswertung von Signalen sehr hilfreich sein.

Die Fouriertransformation ist zum Beispiel in [30, Kap. 6.2] oder [57, Kap. 3] ausführlich erklärt, deshalb wird sie hier nur kurz angerissen. Die Fouriertransformierte $F(j\omega)$ einer i.A. nicht periodischen Funktion $f(t)$ ist gegeben durch

$$F(j\omega) = \int_{-\infty}^{\infty} f(t) e^{-j\omega t} dt . \quad (3.8)$$

In dieser Arbeit wird zur Auswertung später das Amplitudendichtespektrum $|F(j\omega)|$ verwendet.

Da $f(t)$ bei realen Messdaten immer endlich ist (Messung fängt an und hört irgendwann wieder auf), ist $f(t)$ bei realen Messdaten maximal stückweise periodisch. Zur mathematischen Modellierung wird das Signal dann zerlegt in einen periodischen Anteil und ein Rechteck–Funktion, sodass das endliche Signal mit Nullen aufgefüllt wird und damit $f(t)$ auch bis zu beiden Integralgrenzen aus Gleichung 3.8 definiert ist.

$$\hat{f}(t) = \Pi(t^*) \cdot f(t) \quad (3.9)$$

$$\text{mit } \Pi(t^*) = \begin{cases} 0 & \text{wenn } |t^*| > \frac{1}{2} \\ 1 & \text{wenn } |t^*| < \frac{1}{2} \end{cases} \quad (3.10)$$

Aus der Fouriertransformierten der Rechteckfunktion ergeben sich im Frequenzbereich Frequenzanteile, die in $f(t)$ nicht enthalten waren. Fouriertransformierte von endlichen Zeitsignalen müssen deshalb immer im Rahmen dieser Einschränkungen betrachtet werden. Um den Einfluss der Rechteckfunktion zu mildern, werden zusätzlich je nach Anwendung Fensterfunktionen verwendet (vgl. z.B. [20]).

In der digitalen Signalverarbeitung wird zur Fouriertransformation meist der *fast fourier transform* Algorithmus (FFT) verwendet. Dieser ist zum Beispiel in [57] näher erläutert. Er wurde von COOLEY und TUKEY entworfen und ist auch in MATLAB® implementiert.

3.2 Klassische und neuere Regelungsansätze

Bei der Vorbereitung dieser Arbeit wurden zahlreiche Ansätze zur Reglerauslegung in Betracht gezogen und für die Eignung der vorliegenden Aufgabenstellung geprüft. Im folgenden werden einige dieser Ansätze vorgestellt. Besondere Schwierigkeit der vorliegen Aufgabenstellung ist die geringe Kenntnis der Streckenübertragungsfunktion $G_S(s)$, da die zu untersuchenden mechanischen Bauteile bislang noch unbekannt sind. Da das System in dieser Arbeit einen Eingang und einen Ausgang hat (*single input single output* — SISO), beziehen sich die folgenden Ausführungen auch stets auf derartige SISO Systeme.

Unter den klassischen Ansätzen für die Reglersynthese versteht man analytische Verfahren, die unter Kenntnis der Regelstrecke im Bildbereich einen geeigneten Regler entwerfen. Die klassischen und die meisten neueren Ansätze bauen auf einer Modellierung der Streckenübertragungsfunktion $G_S(s)$ auf, ohne die keine Aussagen zur Regelgüte oder Stabilität getroffen werden kann. Im Folgenden werden einige Ansätze zur Reglersynthese etwas näher erläutert.

Direkte Vorgabe

Bei der direkten Vorgabe [26, S. 165ff.] wird versucht, aus der Vorgabe des gewünschten Verhaltens des geregelten Systems durch Umstellen und Auflösen der Gleichungen nach dem Regler-Term im Laplace-Bereich den gewünschten Regler zu erhalten. Diese Methode ist vergleichsweise einfach, allerdings ist eine genaue Kenntnis der Regelstrecke erforderlich. Außerdem existieren häufig nur weiche implizite Kriterien für die Eigenschaften des Reglers und keine exakte explizite Formulierung. Es besteht deshalb die Gefahr, das Verhalten des geregelten Systems unnötig stark einzuschränken (z.B. zu hohe Anforderungen an die Geschwindigkeit zu stellen, mit der die Fehler verschwinden sollen), was u.a. zu sehr großen Stellgrößen führen kann.

Die Methode der direkten Vorgabe ist für die vorliegende Aufgabenstellung ungeeignet, da sie auf der genauen Kenntnis der Streckenübertragungsfunktion aufbaut.

Wurzelortskurvenverfahren & Polvorgabe

Grundlegende Idee hinter dem Wurzelortskurvenverfahren [30, S. 475ff.] ist die gezielte Beeinflussung der Pole in der Wurzelortskurve eines Systems durch hinzufügen von Nullstellen und Polen des Reglers. Für einen geschlossenen Regelkreis berechnen sich die Pole aus der charakteristischen Gleichung

$$1 + G_R(s) \cdot G_S(s) = 0 \quad . \quad (3.11)$$

Nun können anhand der Lage der Pole verschiedene Eigenschaften des Regelkreises abgelesen werden. Meist wird ein Reglerparameter (z.B. der proportionale Anteil) variiert, wodurch die Wurzelortskurve entsteht. Die Eigenschaften des Reglers werden so variiert, dass für den geschlossenen Regelkreis bestimmte Anforderungen an Stabilität, Geschwindigkeit, Überschwingen etc. erfüllt werden.

Das Verfahren der Polvorgabe [26, S. 209ff.] baut auf dem Wurzelortskurvenverfahren auf. Es werden gewünschte Pole des geschlossenen Regelkreises in der charakteristischen Gleichung 3.11 vorgegeben und dann die Parameter des Reglers daraus berechnet.

Bei ungenügender Kenntnis der Regelstrecke ist die Übertragbarkeit der Ergebnisse der Reglersynthese auf die reale Regelstrecke u.U. problematisch. Es wird vermutet, dass insbesondere das Auftreten nicht-linearer höher-harmonischer Schwingungen zu Fehlern in der Auslegung führen kann.

Die klassischen Ansätze zeichnen sich durch die lineare Beschreibung der Regelstrecke aus und einen mehr oder weniger direkten Weg, analytisch eine geeignete Regelstruktur samt Parametern zu bestimmen.

Neuere Verfahren zur Reglersynthese versuchen weniger direkte Anforderungen an den Regler zu formulieren. Häufig ist eine Forderung der neueren Methoden zur Reglersynthese auch die Minimierung der Stellenergie des Reglers. Insbesondere durch intensiven Computer-Einsatz kann eine iterativere Herangehensweise unter Berücksichtigung höherer Komplexität verfolgt werden. Es wird außerdem möglich, mehr Anwendungs- oder Fehlerfälle zu betrachten. Bei einem der neueren Verfahren wird versucht, Betrag und Phase der Übertragungsfunktion des Regelkreises gezielt zu beeinflussen, deshalb heißt dieses Verfahren *loop-shaping*. Dieses Verfahren soll im Folgenden kurz erläutert werden.

loop-shaping

Die Beschreibung erfolgt nach [26, S. 270ff.] und [30, S. 527ff.]. Diese Methode verwendet die Übertragungsfunktion $G_0(s)$ meist in der Darstellung von logarithmischer Verstärkung bzw. Phasenwinkel über Frequenz (Bode-Diagramm). Es werden anhand von $G_0(s)$ bestimmte Kriterien formuliert, die durch hinzufügen von Reglerbausteinen (P-Anteil, I-Anteil, D-Anteil usw.) und Variation der zugehörigen Parameter zu erreichen versucht werden. Eine häufige Forderung ist zum Beispiel die Charakteristik eines I-Anteils für kleine Frequenzen, um einen verschwindenden Regelfehler bei konstanter Stellgröße zu erreichen. Dies resultiert in einer Steigung von -20 dB/dec am linken Rand des Bode-Diagramms.

Neben den funktionalen Anforderungen gibt es auch Bedingungen, um die Stabilität des geschlossenen Regelkreises zu gewährleisten. Für die Stabilitätsanalyse spielt die sog. Schnittfrequenz ω_s eine maßgebliche Rolle. Die Schnittfrequenz ist die Frequenz, bei der die Amplitudenkurve die 0 dB -Linie schneidet. Für höhere Frequenzen wirkt der offene Regelkreis demnach dämpfend. Im Folgenden sind die Bedingungen für Stabilität des geschlossenen Regelkreises aufgelistet.

- Phasenrand: Als Phasenrand wird die Differenz des Phasenwinkels bei Schnittfrequenz ω_s zu 180° bezeichnet.

$$\phi_R = 180^\circ - |\phi(\omega_s)| \quad (3.12)$$

Um theoretisch Stabilität zu erreichen, muss $\phi_R > 0$ sein, um Unsicherheiten bei der Modellbildung oder Änderung des Systems zu kompensieren wird aber meist ein wesentlich größerer Phasenrand von ca. 30° bis 60° gefordert.

- Amplitudenrand: Als Amplitudenrand wird der Kehrwert der Verstärkung verstanden, bei der der Phasenwinkel -180° beträgt.

$$k_R = \frac{1}{G_0(j\omega_{-180^\circ})} \quad (3.13)$$

Der Phasenrand muss theoretisch $k_R > 1$ sein, in der Praxis werden aber noch deutlich größere Werte gefordert.

- Steigung der Amplitudenkurve bei Schnittfrequenz ω_s : Häufig wird zusätzlich zum Phasen- und Amplitudenrand noch eine Steigung der Amplitudenkurve von mindestens -20 dB/dec gefordert.

Ebenfalls aus Mangel an ausreichender Kenntnis über die Streckenübertragungsfunktion ist das loop-shaping-Verfahren hier nicht anwendbar. Die Forderungen nach Phasen- und Amplitudenrand bieten aber ein besseres Verständnis über die Stabilitätsbedingungen, insbesondere für Totzeitglieder, die ausschließlich den Phasenwinkel beeinflussen und so die Stabilitätsgrenze verschieben. Die Totzeit eines Systems kann auch für nicht-lineare Systeme leicht bestimmt werden und kann so auch heran gezogen werden, wenn andere Komponenten der Übertragungsfunktion noch unbekannt sind.

3.3 Iterativ lernende Regelung

Die Idee der iterativ lernenden Regelung (ILR, engl. *iterative learning control*, ILC) soll nach [12] auf das US-Patent 3 555 252 aus dem Jahr 1967 [16] zurück gehen. Es scheint aber auch aus Japan davon unabhängige Ansätze gegeben zu haben [5, 25]. Die Grundidee hinter der ILR ist die Verbesserung der Regelgüte eines Systems für eine sich wiederholende Regelaufgabe durch Ausnutzung der bekannten Periodizität [1, 27].

Der Regelfehler wird zunächst für einen Zyklus gespeichert. Anschließend werden aus den Informationen eines ganzen Zyklus durch ein Lerngesetz (*update law*) Stellgrößen für den nächsten Zyklus berechnet. Der Regler arbeitet also nicht kontinuierlich oder für den jeweils nächsten Zeitschritt, sondern jeweils für einen Zyklus oder eine Periode eines sich wiederholenden Signals. Durch die Wiederholungen des Führungsgrößenverlaufs und das Speichern der Stellgrößen und des Fehlers für einen ganzen Zyklus können sehr effiziente Lerngesetze formuliert werden. Insbesondere die Forderung nach Kausalität innerhalb einer Periode, die für kontinuierliche Regler gilt, wird durch die quasi-offline *a posteriori* Verarbeitung der Regelfehler einer ganzen Periode aufgehoben.

Es wird unterschieden zwischen iterativ lernender Regelung (ILR) und repetitiver Regelung (RR). Bei der ILR werden einzelne Regelabläufe wiederholt, u.U. mit getrennten Anfangs- und Endpunkt und nicht in direkter Abfolge aufeinander. Bei der repetitiven Regelung gibt es einen durchgängigen Führungsgrößenverlauf, der sich allerdings periodisch wiederholt. Es kann jedoch gezeigt werden, dass beide Fälle mathematisch in das gleiche Problem überführt werden können und somit auch die Auslegung mit den gleichen Methoden erfolgen kann [27, 29].

Einige Vorteile der ILR bzw. RR gegenüber herkömmlichen Regelansätzen sind im Folgenden aufgeführt.

- Bei einigen Lerngesetzen der ILR ist keine oder nur sehr geringe Kenntnis der Regelstrecke notwendig. Dies ist insbesondere für nicht-lineare Regelstrecken von Vorteil, da eine lineare Modellierung der Strecke stets nur um einen bestimmten Arbeitspunkt gültig ist und die meisten analytischen Verfahren für nicht-lineare Übertragungsfunktionen nicht anwendbar sind.
- Bei der Implementierung auf einem Mikrocontroller oder mit einem Computer kann die Berechnung der neuen Stellgrößen getrennt erfolgen und erst nach einer oder mehr Perioden in den neuen Führungsgrößenverlauf einfließen. Die Echtzeitfähigkeit beschränkt sich damit auf das Ausgeben der Stellgrößen und Messen der Ausgangsgröße, nicht aber auf die Berechnung. Damit ist die ILR auch möglich, wenn zwischen jedem einzelnen Schritt nicht genügend Zeit zur Berechnung der nächsten Stellgröße vorhanden wäre.
- Da die Berechnung der neuen Stellgrößen unter Kenntnis eines gesamten Zyklus geschieht, können auch Fehlerwerte verwendet werden, die aus Sicht des aktuellen Zyklus „in der Zukunft“ liegen. Dies hebt die Forderung herkömmlicher Regelgesetze nach Kausalität auf. So können verschwindende Regelfehler erreicht werden, sofern überhaupt Steuerbarkeit vorliegt.

Da die Verarbeitung der Messdaten in einem Mikrocontroller oder Computer erfolgt, bietet sich die diskrete mathematische Formulierung der Regelstrecke (anstelle der kontinuierlichen aus Gleichung

3.1) an. Die Gleichungen für mehrere Zustände und eine Ein- bzw. Ausgangsgröße lauten dann:

$$\vec{x}(k+1) = A\vec{x}(k) + \vec{b} \cdot u(k) , \quad (3.14)$$

$$y(k) = \vec{c}^\top \vec{x}(k) + d \cdot u(k) . \quad (3.15)$$

In den obigen Gleichungen beschreibt $k \in \mathbb{N}$ den aktuellen Zeitschritt. Die Terme A , \vec{b} , \vec{c} und d haben gleiche Bedeutung wie in Gleichung 3.1, die Einträge in A und \vec{b} hängen aber explizit von der gewählten Schrittweite T bei der Diskretisierung ab (vgl. z.B. [31, S.444]):

$$A = e^{A^* \cdot T} , \quad (3.16)$$

$$\vec{b} = \int_0^T e^{A^* \cdot \tau} d\tau \cdot b^* , \quad (3.17)$$

$$\vec{c} = \vec{c}^* , \quad (3.18)$$

$$d = d^* . \quad (3.19)$$

Supervektordarstellung Da obige Gleichung 3.14 rekursiv ist, kann durch ineinander einsetzen der einzelnen Zeitschritte eine Gleichung zur Berechnung des Zustands aus den Anfangsbedingungen angegeben werden:

$$\vec{x}(k) = A^k \vec{x}(0) + A^{k-1} \vec{b} u(0) + \cdots + A \vec{b} u(k-2) + \vec{b} \cdot u(k-1) , \quad (3.20)$$

$$y(k) = \vec{c}^\top A^k \vec{x}(0) + \vec{c}^\top A^{k-1} \vec{b} u(0) + \cdots + \vec{c}^\top \vec{b} u(k-1) + du(k) . \quad (3.21)$$

Schreibt man alle Zeitpunkte eines Zyklus gemäß Gleichung 3.21 untereinander, erhält man die Supervektordarstellung, auf der weitere Formulierungen beruhen:

$$\underbrace{\begin{bmatrix} y(0) \\ y(1) \\ \vdots \\ y(N) \end{bmatrix}}_{\vec{y}} = \underbrace{\begin{bmatrix} \vec{c}^\top \\ \vec{c}^\top A \\ \vdots \\ \vec{c}^\top A^N \end{bmatrix}}_{\mathcal{G}_0} \vec{x}(0) + \underbrace{\begin{bmatrix} d & 0 & 0 & \dots & 0 \\ \vec{c}^\top \vec{b} & d & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \vec{c}^\top A^{N-1} \vec{b} & \dots & \dots & \vec{c}^\top \vec{b} & d \end{bmatrix}}_{\mathcal{G}} \underbrace{\begin{bmatrix} u(0) \\ u(1) \\ \vdots \\ u(N) \end{bmatrix}}_{\vec{u}} \quad (3.22)$$

$$\vec{y} = \mathcal{G}_0 \vec{x}(0) + \mathcal{G} \vec{u} \quad (3.23)$$

Lerngesetz Anhang der Supervektordarstellung lässt sich ein Lerngesetz (oder auch Update-Regel, englisch *update law*) für den Zyklus $m+1$ aus den Werten des Zyklus m formulieren

$$\vec{\mathcal{U}}_{m+1} = Q \cdot \vec{\mathcal{U}}_m + L \cdot \vec{\mathcal{E}}_m , \quad (3.24)$$

wobei $\vec{\mathcal{E}}_m$ der Regelfehler in Supervektordarstellung ist. Durch Q wird ein Stellgrößen-Filter eingeführt. Handelt es sich bei Q und L um Matrizen mit konstanten Einträgen spricht man von einer linearen ILR erster Ordnung. Normalerweise wird $Q = I$ gesetzt [42, S. 1116], wodurch der Einfluss verschwindet. Zur Erhöhung der Robustheit des Reglers kann aber hier z.B. auch ein digitaler Tiefpass durch Bandstruktur von Q eingeführt werden. Ein Lerngesetz höherer Ordnung entsteht, wenn in Q Stellgrößenwerte aus mehreren vergangenen Perioden einfließen: $Q = Q(\vec{\mathcal{U}}_m, \vec{\mathcal{U}}_{m-1}, \vec{\mathcal{U}}_{m-2}, \dots)$. In [42] wurde aber gezeigt, dass dies nicht zu einer Verbesserung des Lernverhaltens führt und deshalb wurde dieser Ansatz hier auch nicht verfolgt.

Je nach Ansatz kann die Lernmatrix L z.B. die Struktur einer Diagonalmatrix haben mit $L = \beta I$, eine obere Dreiecksmatrix sein oder Bandstruktur haben. Im Allgemeinen Fall könnte L natürlich auch voll besetzt sein. Dies entspricht sehr mächtigen Lerngesetzen, da in jeden einzelnen Wert der Stellgröße alle anderen Werte aus einer Periode einfließen. In der Literatur wird neben der Matrixnotation in der Supervektordarstellung auch die zeitdiskrete Darstellung verwendet, die näher an der konkreten Implementierung, beispielsweise auf einem Mikrocontroller ist.

Longman schlägt folgendes Lerngesetz für eine repetitive Regelung vor [29, S. 940, Formel 24], welches hier in zeitdiskreter Form dargestellt ist.

$$u(k) = u(k-p) + \sum_{i=i_0}^{i_f} \phi(i) e(k-p+\lambda+i) \quad (3.25)$$

Dabei ist k der aktuelle (diskrete) Zeitschritt, p die Anzahl Zeitschritte in einer Periode und λ eine positive Phasenverzögerung. Die $\phi(i)$ ergeben eine Regelverstärkung. In Supervektordarstellung entspräche Gleichung 3.25 der folgenden Lernmatrix:

$$L = \begin{bmatrix} & & & & & & & & & \\ & \leftarrow \lambda \rightarrow & & & & & & & & \\ 0 & \dots & \phi(i_0) & \phi(i_0+1) & \dots & \phi(i_f) & \dots & \dots & \dots & 0 \\ 0 & \dots & \dots & \phi(i_0) & \phi(i_0+1) & \dots & \phi(i_f) & \dots & \dots & 0 \\ 0 & \dots & \dots & \dots & \phi(i_0) & \phi(i_0+1) & \dots & \phi(i_f) & \dots & 0 \\ \vdots & \vdots \end{bmatrix} \quad (3.26)$$

Die Phasenverschiebung λ ist die Verschiebung des von null verschiedenen Bandes in der Lernmatrix L in Gleichung 3.26.

Durch den rekursiven Charakter des Lerngesetzes in Gleichung 3.25 lässt es sich auch als I-Regler interpretieren, der allerdings nicht von Zeitschritt zu Zeitschritt, sondern von Periode zu Periode für jeden einzelnen Zeitschritt der Periode arbeitet.

Die Summe im Lerngesetz in Gleichung 3.25 ähnelt einem gewichteten gleitenden Mittelwert und lässt sich als Tiefpassfilter interpretieren. Nach [29] ist die Grenzfrequenz dieses Tiefpasses zusammen mit der Phasenverschiebung λ maßgeblich für die Stabilität des Reglers.

Neben der Anwendung von Filtern bzw. Kompensatoren zur gezielten Beeinflussung bestimmter Frequenzanteile werden in der Literatur auch komplexere Lerngesetze vorgeschlagen. Stellvertretend sei hier auf den PID ähnlichen Aufbau aus [32] oder [44] verwiesen. Die genannte Formulierung wurde in die hier verwendete Nomenklatur überführt und lautet

$$u(k) = u(k-p) + K_p \cdot e(k-p) + K_i \sum_{i=i_0}^{i_f} e(k-p+i) + K_d (e(k-p+1) - e(k-p)) \quad . \quad (3.27)$$

Der in Gleichung 3.27 geschilderte Ansatz hat allerdings vollständig rekursiven Charakter, deshalb ist der Vergleich mit einem PID-Regler irreführend. Kritik soll hier nicht am Lerngesetz geäußert werden, sondern nur an der Benennung. Der Begriff PID führt eine Analogie zu den PID-Reglern im Zeitbereich ein, die aufgrund der Rekursion in Gleichung 3.27 nicht korrekt ist. Um dies zu verdeutlichen wurde in Gleichung 3.28 die „normale“ diskrete Formulierung eines PID-Reglers aufgeschrieben (nach [30, S.399]). Die Parameter sind nicht wie sonst üblich die Zeitkonstanten sondern wurden hier zur Betonung der Analogie zu Gleichung 3.27 umgeschrieben.

$$u(k) = K_p \cdot e(k) + K_i \sum_{j=0}^k e(j) + K_d (e(k) - e(k-1)) \quad (3.28)$$

Ein „normaler“ PID-Regler hat insgesamt keinen rekursiven Charakter. Stattdessen wird die neue Stellgröße direkt aus den Fehlern berechnet. Einzig der I-Anteil ließe sich in eine rekursive Formulierung umstellen. Somit handelt es sich bei dem P-Anteil und I-Anteil aus Gleichung 3.27 um eine Formulierung, die sehr ähnlich zu Gleichung 3.25 ist und um einen differentiellen Anteil erweitert wurde. Der vermeintliche P-Anteil entspricht eher einem I-Anteil und der vermeintliche I-Anteil realisiert den in [29] beschriebenen Filter als gleitenden Mittelwert.

Nichts desto trotz wurde die Idee eines PID ähnliches Ansatzes in dieser Arbeit aufgegriffen und das Lerngesetz aus Gleichung 3.25 entsprechend erweitert. Näheres dazu ist in Kapitel 5 beschrieben.

Die im folgenden geschilderten Verfahren scheiden für die hier vorliegende Anwendung aus, da sie auf einer bekannten Streckenübertragungsfunktion basieren:

Die Inversion der Streckenübertragungsfunktion verwendet als Lernmatrix in der Update–Regel die Inverse der Streckenübertragungsfunktion [18]. Diese Methode lässt sich als Analogie zur direkten Vorgabe interpretieren (siehe Abschnitt 3.2). Übertragen auf die hier verwendete Notation (vgl. Gleichung 3.24) lautet die Update Regel dann:

$$\vec{U}_{m+1} = \vec{U}_m + \beta \cdot G_0^{-1} \vec{\mathcal{E}}_m \quad (3.29)$$

$$\text{sodass} \quad L = \beta \cdot G_0^{-1} \quad (3.30)$$

Gradientenbasierende Verfahren gehen von der Annahme aus, dass der Regelfehler abgebaut wird, wenn schrittweise entlang des Gradienten gelernt wird und so Konvergenz erreicht werden kann. Der Weg der steilsten Steigung (schnellste Regelung) wird laut [43] durch Verwendung der transponierten Streckenübertragungsfunktion erreicht. Dies führt zur Formulierung folgenden Lerngesetzes:

$$\vec{U}_{m+1} = \vec{U}_m + \beta \cdot G_0^T \vec{\mathcal{E}}_m \quad (3.31)$$

$$\text{sodass} \quad L = \beta \cdot G_0^T \quad (3.32)$$

3.4 Betrachtungen zur Stabilität

Die Analyse der Stabilität des geschlossenen Regelkreises stellt ein Kernproblem dieser Arbeit dar. Wie bereits erwähnt existiert zum Zeitpunkt der durchzuführenden schwingungsmesstechnischen Untersuchungen noch kein analytisches oder numerisches Modell des zu untersuchenden Bauteils, da die durchzuführenden Versuche ja eben der Bestimmung dieses Modells dienen. Außerdem wird die Regelung gerade für Bauteile mit ausgeprägten nicht–Linearitäten benötigt, für die eine Modellbildung äußerst schwierig ist. Selbst wenn eine ausreichende Modellbildung vor den eigentlichen Identifikationsversuchen gelänge, würden anschließend die bekannten analytischen Verfahren scheitern, da diese auf einer linearen Formulierung des Modells aufbauen. Es könnte natürlich das Modell lokal linearisiert werden, um so für einen bestimmten Betriebspunkt Aussagen zur Stabilität treffen zu können. Insbesondere bei Sweep–Experimenten werden aber zahlreiche Betriebspunkte durchlaufen, für deren Stabilitätsuntersuchung dann jeweils ein lokales linearisiertes Modell vorliegen müsste. Nicht–Linearitäten können in einem linearen Modell allerdings nie abgebildet werden. Da diese aber gerade wesentlicher Grund für den Einsatz der Regelung sind, könnte durch ein lokal linearisiertes Modell nur eine ungenügende Bewertung des Reglers getroffen werden.

In [29] wird für eine Lernmatrix L mit ϕ auf allen Diagonaleinträgen eine formale Bedingung für einen verschwindenden Regelfehler aufgestellt (Notation angepasst):

$$0 < \vec{c}^\top \vec{b} \phi < 2 \quad . \quad (3.33)$$

Diese Bedingung enthält nicht (!) die Streckenübertragungsfunktion. Dies stellte unter anderem einen Grund dar, das hier vorliegende Problem mit einer iterativen Regelung zu lösen. Es wird jedoch durch Simulationen gezeigt, dass durch diese Bedingung alleine noch keine Stabilität im Sinne eines monoton fallenden Regelfehlers erreicht werden kann. Vielmehr kann bei der Wahl bestimmter Regelstrecken der Regelfehler zunächst dramatisch ansteigen, bevor er schließlich doch noch gegen Null konvergiert. Ein solches Verhalten einer Regelung ist natürlich weder wünschenswert noch akzeptabel. Die Forderung nach einem monoton fallenden Regelfehler führt für $Q = I$ zu einer Bedingung für den maximalen Singulärwert (Notation nach [27])

$$\bar{\sigma}(I - GL) < 1 \quad , \quad (3.34)$$

oder für den Fall $Q \neq I$ zu

$$\bar{\sigma}(G(Q - LG)G^{-1}) < 1 \quad (3.35)$$

Um eine Bedingung für einen monoton fallenden Regelfehler formulieren zu können, muss die Streckenübertragungsfunktion also doch bekannt sein.

Der Fokus dieser Arbeit liegt deshalb klar auf einem praktischen Ansatz, der ohne umfangreiche Messungen oder Simulationen die Anwendung der Regelung im Umfeld des Versuchsaufbaus ermöglichen soll. Bei der Analyse der unterschiedlichen Regler-Typen wurde deshalb insbesondere bewertet, welche Regler eine hohe Robustheit gegenüber unbekannten Streckenübertragungsfunktionen aufweisen. Die iterativ lernende Regelung verspricht hier mit einem Lerngesetz wie in Gleichung 3.25 Aussagen zur Stabilität auch bei geringer Kenntnis der Regelstrecke treffen zu können bzw. eine Regelung überhaupt realisierbar zu machen. Für einen empirischen Ansatz ist es wichtig, möglichst wenige Reglerparameter zu haben, die am Versuchsstand für jedes zu untersuchende Bauteil möglichst einfach eingestellt werden können.

Positiv zu erwähnen sei in diesem Zusammenhang, dass überhaupt nur Streckenübertragungsfunktionen möglich sind, die von sich aus Stabilität aufweisen. Da mechanische Bauteile in stabiler Ruhelage untersucht werden, kehren sie nach Auslenkung stets wieder in ihre Ruhelage zurück und weisen für alle Frequenzen eine positive Dämpfung auf. Die Versuche werden bislang durch eine Steuerung durchgeführt, was nur bei stabilen Streckenübertragungsfunktionen möglich ist.

4 Planung und Auslegung der Regelung

4.1 Beschreibung des Versuchsaufbaus

Kernstück des Versuchsaufbaus ist das zu untersuchenden Strukturauteil. In dieser Arbeit wurden beispielhaft eine Glasfaserplatte und eine Schiene mit Schienendämpfer als Testsystem verwendet. Das Strukturauteil wird mittels eines Aktuators (Shaker) an einer Stelle zu einer Schwingung angeregt. An der Stelle der Krafteinleitung ist ein Piezo-Kraftsensor angebracht. Für das Strukturauteil wird versucht, eine möglichst freie Lagerung zu realisieren. Dies kann z.B. durch eine Aufhängung an Schnüren erreicht werden. Die dominierende Bewegungsrichtung sollte dabei senkrecht zur Aufhängung sein.



Abbildung 4.1: Messaufbau mit Glasfaserplatte

Das Signal für den Aktuator wird mittels eines passenden Leistungsverstärkers verstärkt. Der Aufbau ist schematisch in Abbildung 4.3 dargestellt. Für die Regelung ist das Signal an den

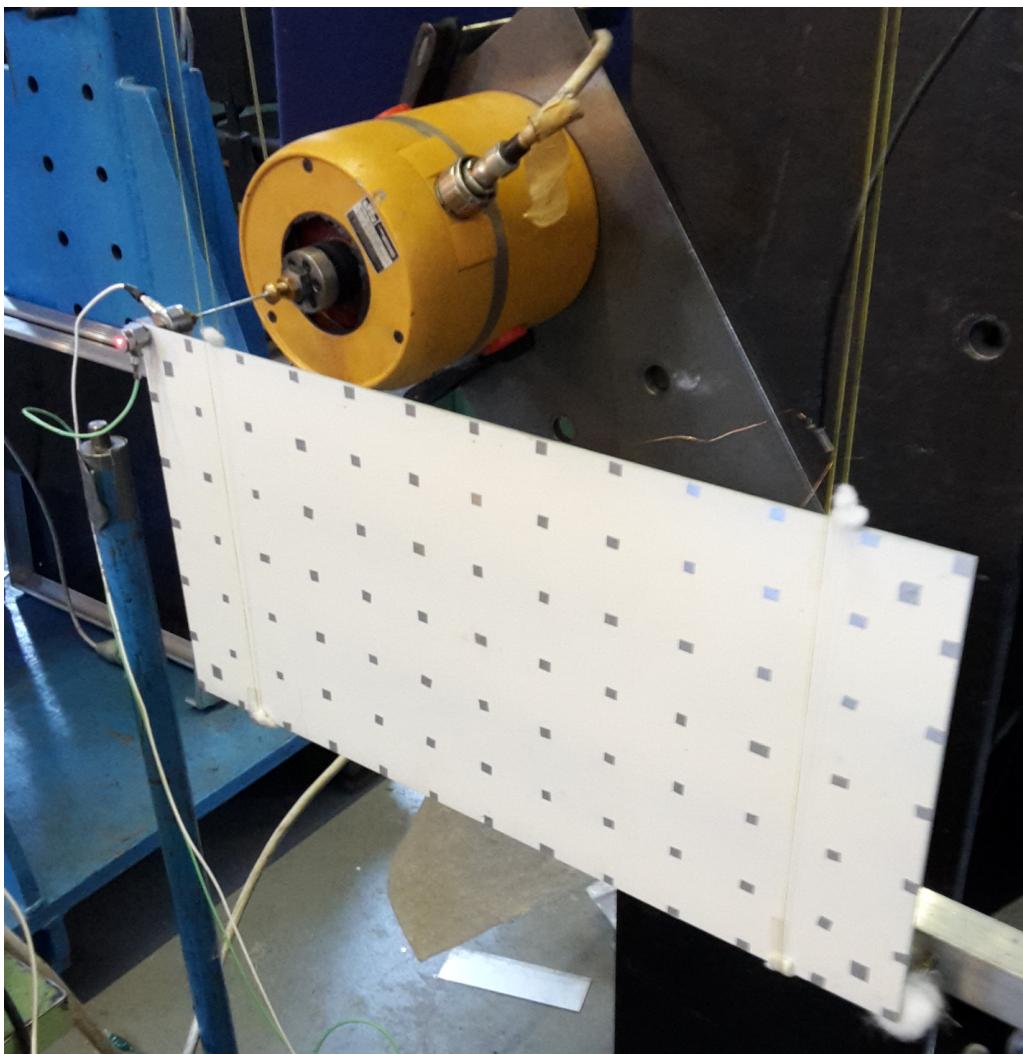


Abbildung 4.2: Glasfaserplatte und Shaker

Leistungsverstärker die Stellgröße $u(t)$ und das Signal aus Kraftsensor die Regelgröße $y(t)$.

Für die Durchführung einer Messung zur modalen Analyse werden mittels eines *laser scanning vibrometer* an unterschiedlichen Punkten auf dem Strukturauteil die Geschwindigkeiten gemessen. Diese Punkte sind auf den Bildern durch kleine reflektierende Klebchen zu erkennen. Aus den Geschwindigkeitsdaten kann das *laser scanning vibrometer* Verschiebungen und Beschleunigungen berechnen. Anhand der räumlich und zeitlich hoch aufgelösten Daten zur Verschiebung können die Schwingungsmoden des Bauteils bestimmt werden.

Abbildung 4.1 gibt einen Überblick über den Versuchsaufbau mit der Glasfaserplatte. Abbildung 4.2 zeigt den Shaker LDS V406 [10], die Verbindung zur Glasfaserplatte sowie den Piezo-Kraftsensor Dytran 1051V3 [14]. Die Glasfaserplatte wurde an einer Ecke normal zur Platte angeregt. Die Aufhängung war senkrecht, sodass die Platte in Normalen-Richtung möglichst frei schwingen konnte. Das *laser scanning vibrometer* nahm in den Versuchen zur modalen Analyse dann die Geschwindigkeiten an regelmäßig verteilten Punkten auf der Plattenoberfläche auf. Als Leistungsstärker für den Shaker wurde ein LDS PA 100 [9] verwendet.

Neben den Messungen an der Glasfaserplatte wurden auch Messungen an einer Eisenbahnschiene mit Schwingungsdämpfer durchgeführt. Der Aufbau stammt aus einem Projekt, in dem die

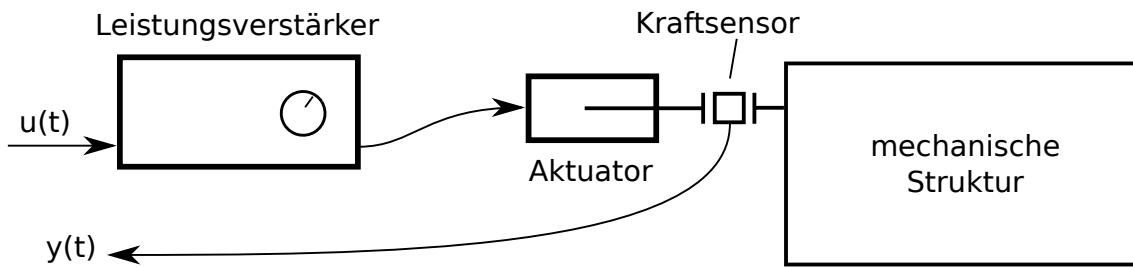


Abbildung 4.3: Schematische Darstellung des Versuchsaufbaus mit Kennzeichnung der Stell- und Regelgröße

Wirkungsweise und Dynamik der angebrachten Schienendämpfer untersucht wird. Mit dem *laser scanning vibrometer* werden dafür unterschiedliche Punkte auf dem Dämpfer vermessen. Die Schwingungsdämpfer dienen der Reduktion von Körperschall im Einsatzfeld. Durch den Schwingungsdämpfer reagiert die Struktur insbesondere in der Nähe der Resonanzfrequenzen mit ausgeprägten Nicht-Linearitäten in Form von höher-harmonischen Schwingungen. Der Aufbau des Versuchs war analog zum Aufbau mit der Glasfaserplatte. Einziger Unterschied war die Verwendung eines Aktuators LDS V455 [11] und passenden Leistungsverstärkers LDS PA 1000 [9] mit größerer Leistung, entsprechend der höheren Masse des Strukturauteils. Der Versuchsaufbau mit der Eisenbahnschiene ist in den Abbildungen 4.4 und 4.5 zu sehen.

Die Schiene war waagerecht aufgehängt. Die Anregung erfolgte an einem Ende in einem Winkel von 45° sowohl zur Schienenachse als auch zur Senkrechten. Dadurch konnten sowohl Längs- als auch beide Biegeschwingungen angeregt werden.

Im Rahmen dieser Arbeit wurden für die Versuche nur das Signal des Kraftsensors und das Signal an den Verstärker verwendet. Messungen mit dem *laser scanning vibrometer* wurden nicht durchgeführt, da dieses ausschließlich der Messung der Bauteilschwingungen dient und keine Rückkopplung auf die Bauteilschwingungen hat.

Während der Versuche wurden das Signal des Kraftsensors und das Signal zum Leistungsverstärker mit einem Oszilloskop überwacht. Dies war hilfreich, um das Verhalten der Regelung zu visualisieren und mit diesem Wissen die Reglerparameter richtig einzustellen. Die Aufzeichnung der Messwerte zur späteren Auswertung erfolgte aber stets mit den hochwertigen Messkarten.



Abbildung 4.4: Versuchsaufbau für Messungen an der Eisenbahnschiene mit Schwingungsdämpfer — Gesamtansicht

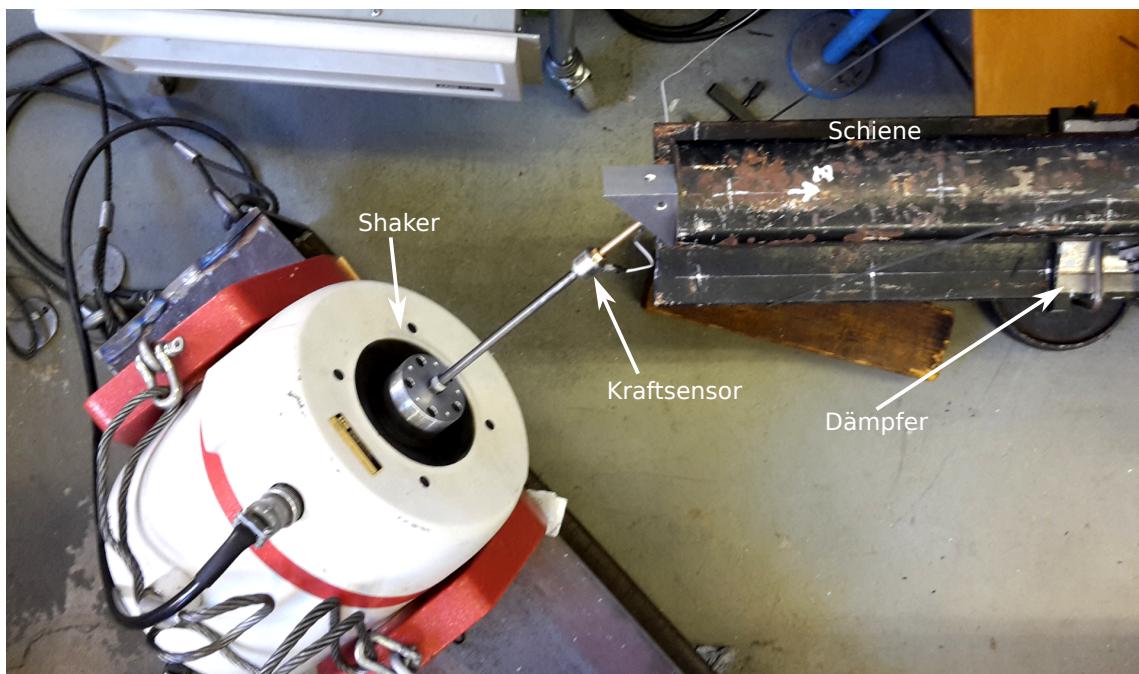


Abbildung 4.5: Versuchsaufbau für Messungen an der Eisenbahnschiene mit Schwingungsdämpfer — Ansicht von oben

4.2 Vorhandene Hardware

Der vorhandene Messaufbau besteht aus einem Steuerungrechner, auf dem das Programm LabView [40] der Firma National Instruments die Versuchsdurchführung steuert. LabView steuert die Ausgangskarte mit ihrem digital-analog-converter (DAC) an, welche das gewünschte Signal über den Leistungsverstärker an den Aktuator, einen Shaker weitergibt. Der Shaker ist mit der Struktur mechanisch verbunden.

Direkt an der Verbindungsstelle zwischen Struktur und Shaker befindet sich eine Kraftmessdose mit integriertem Ladungsverstärker, welche ein vorverarbeitetes, belastbares Signal proportional zur anliegenden Kraft ausgibt. Dieses Signal wird wiederum durch den analog-digital-converter (ADC) auf der Messkarte in den Computer eingelesen. Der oben geschilderte Messaufbau ist in der Abbildung 4.6 schematisch dargestellt.

Sowohl die Ausgangskarte, als auch die Eingangskarte werden mit einem Chassis NI 9178 [38, 39] betrieben (nicht in Abb. 4.6 eingezeichnet). Das Chassis kommuniziert mit dem Computer über USB. Da die USB-Schnittstelle des Computers nicht für harte Echtzeit-Kommunikation geeignet ist, werden zahlreiche zeitkritische Low-Level Aufgaben von dem Chassis übernommen. Die Ein-/Ausgangskarten übernehmen lediglich die Umwandlung von digitalen in analoge Signale und umgekehrt.

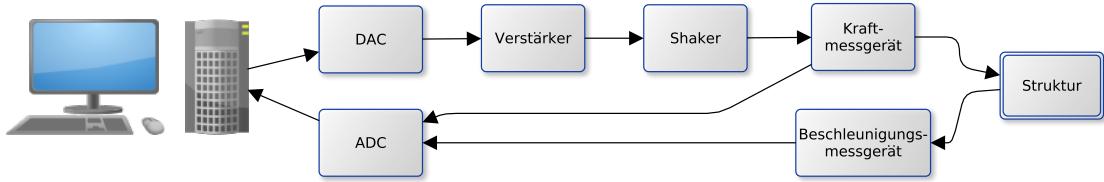


Abbildung 4.6: Schematische Darstellung des vorhandenen Messaufbaus

4.2.1 analoger Ausgang

Die Umwandlung von digitaler Information in ein analoges Signal erfolgt in der Ausgangskarte NI 9263 [37, 36]. Kenngrößen der Ausgangskarte sind in der Tabelle 4.1 dargestellt.

Tabelle 4.1: Kenngrößen der Ausgangskarte NI 9263

Eigenschaft	Wert	Einheit
Sample Rate	100	kS/s
Auflösung	16	bit
Spannungsbereich	± 10	V
Schrittweite	0,3	mV

4.2.2 analoger Eingang

Die Umwandlung des analogen Signals in digitale Information wird durch die Messkarte NI 9234 [34, 35] realisiert. Die Messkarte hat eine extrem hohe Auflösung bei hoher Taktrate und lässt sich durch softwareseitige Einstellungen für verschiedene Signalarten und Störunterdrückungen einstellen.

Tabelle 4.2: Kenngrößen der Eingangskarte NI 9234

Eigenschaft	Wert	Einheit
Sample Rate	52,2	kS/s
Auflösung	24	bit
Dynamikbereich	144	dB
Spannungsbereich	± 5	V
Schrittweite	0,3	μ V

4.2.3 Kraftsensor

Als Kraftsensor wird ein Piezo Sensor Dytran 1051V3 [14] nach IEPE Standard verwendet. Die technischen Daten des Kraftsensors sind in Tabelle 4.3 aufgeführt.

Tabelle 4.3: Technische Daten des Kraftsensors Dytran 1051V3

Eigenschaft	Wert	Einheit
Sensitivität	11,24	mV/N
Auflösung	6,23	mN
max. Kraft (Messbereich)	444	N
Steifigkeit	2,0	N/ μ m
Impedanz Ausgang	100	Ω
Linearität	< 1	% F.S.
Resonanzfrequenz	75	kHz

4.3 Regelungsansätze und Arbeitsweise des Reglers

Verschiedene Ansätze für die Regelung wurden im Rahmen dieser Arbeit entworfen. Zum einen muss zunächst unterschieden werden, an welcher Stelle der Regler in die bestehende Hardware eingebunden wird. Zum anderen kommen verschiedene Arbeitsweisen des Reglers in Frage. Auf die unterschiedlichen Ansätze wird im Folgenden näher eingegangen.

4.3.1 klassische Regelung im Zeitbereich

Eine Regelung im Zeitbereich wäre z.B. ein klassischer PID-Regler, der in das bestehende System integriert werden könnte, in dem er zwischen den analogen Ausgang des DACs und den Leistungsverstärker geschaltet wird (s.a. Abbildung 4.7).

Die bekannten Verfahren zur Reglerauslegung können angewendet werden: Polvorgabe, Loop-Shaping, Experimentelle Bestimmung der Regler Parameter und andere.

4.3.2 Lernende Regelung

Eine lernende Regelung arbeitet auch im Zeitbereich. Der Unterschied wurde hier aber eingeführt, um zwischen den klassischen Ansätzen im Zeitbereich zu unterscheiden und die lernende Regelung

mit ihren besonderen Vorteilen davon abzugrenzen.

Repetitive Regelung (Periode) Bei dieser Methode werden Stellwerte aus den Messwerten einer vergangenen Periode berechnet. Diese Methode erlaubt eine positive Phasenverschiebung („akausale Regelung“), da durch Ausnutzung der Periodizität des Signals quasi in die Zukunft geschaut werden kann.

Vorteile:

- Durch Ausnutzung der Periodizität des Signal sind acausalen Lerngesetze möglich
- Konvergenz innerhalb einer geringen Anzahl von Perioden (Größenordnung von Sekunden)
- überschaubare Datenmengen, da nur Daten von einer voran gegangenen Periode gespeichert und ausgewertet werden müssen

Nachteile:

- Regelung muss neue Stellgrößen in Echtzeit berechnen
- Änderung des Stellgrößenverlaufs muss zur Laufzeit realisiert werden (ohne „Knacken“ o.ä.)

Iterativ lernende Regelung (gesamtes Experiment) Erweitert man den Horizont der Regelung auf ein gesamtes Experiment (z.B. einen Sweep), so kann man den Algorithmus der lernenden Regelung natürlich auch auf den gesamten Datensatz anwenden und dann einen verbesserten Datensatz für die Stellgröße im nächsten Experiment verwenden usw.

Vorteile:

- bestehender Aufbau müsste physisch nicht verändert werden, sämtliche Hardware wird weiter benutzt
- Periodizität im Führungsgrößenverlauf wird nur im Hinblick auf ein sich wiederholendes Experiment gefordert. Jedes Experiment für sich genommen muss keinerlei Periodizität im Signalverlauf aufweisen. Dadurch ist dieser Ansatz maximal flexibel in Bezug auf den Signalverlauf eines Experiments.

Nachteile:

- Konvergenz über mehrere Experimente, dadurch erst nach langer Zeit (Größenordnung von Minuten)
- kein kontinuierlicher Betrieb möglich
- Verarbeitung großer Datenmengen erforderlich

4.3.3 Regelung im Frequenzbereich

Anpassung der Amplitude der Soll–Frequenz Ein einfacher, aber vielleicht schon sehr effektiver Ansatz ist, die Amplitude der Soll–Frequenz zur Laufzeit durch einen zwischengeschalteten Verstärker anzupassen. An der Struktur wird die Amplitude der Regelgröße (Kraft am Aktuator) gemessen und damit die Amplitude der Stellgröße (Spannung für den Aktuator) zur Laufzeit angepasst.

Vorteile:

- sehr einfach zu realisieren

- schnelle Konvergenz der Amplitude

Nachteile:

- Störungen in anderen Frequenzen können nicht kompensiert werden.

Anpassung von Amplituden und Phasengang sämtlicher Frequenzen Es wäre denkbar, den aktuellen Verlauf der Stellgröße mittels Fourier–Transformation in die einzelnen Frequenzanteile zu zerlegen und störende Frequenzanteile jeweils mit 180° Phasenverschiebung zurück zu koppeln. Zunächst muss ein Algorithmus störende Frequenzen ermitteln (oder störende Frequenzen werden *a priori* festgelegt, bspw. eine Anzahl höher-harmonischer Schwingungen). Anschließend wird anhand der Messdaten aus der letzten Versuchsdurchführung Amplitude und Phasengang dieser Störanteile bestimmt. Dies führt zu einem Optimierungsproblem mit je zwei Parametern für jede zu unterdrückende Frequenz. Bekannte Optimierungsverfahren können nun angewendet werden, um iterativ einen optimalen Signalverlauf der Stellgröße zu erreichen. Ein direktes Verfahren ist aufgrund der nicht–Linearität der Strecke nicht möglich.

Dieser Ansatz wird u.a. in [24] verfolgt. Die Umsetzung eines solchen Regelungsansatzes ist allerdings rechenintensiv und u.U. schwer in einen bestehenden Messaufbau zu integrieren. Dieser Ansatz wurde von MAGNEVALL bereits erfolgreich umgesetzt [33], allerdings erfolgte die Berechnung der neuen Stellgrößen offline, also zwischen einzelnen Experimenten. So ist ein dauerhafter Betrieb (z.B. unter Variation bestimmter Versuchs–Parameter) mit eingeschaltetem Regler nicht möglich. Die Konvergenzzeit ist außerdem relativ lang, da eine Iteration viele Sekunden dauert und zwischen den Messungen gewartet werden muss. Es wird außerdem von Problemen des Algorithmus in der Nähe der Resonanzfrequenzen berichtet.

Vorteile:

- geringe Echtzeitanforderungen an die Regelung für Analyse und Berechnung der Stellgrößen
- Kompensation anderer Frequenzen

Nachteile:

- falls Experiment nicht unterbrochen werden soll: Anpassung des Stell–Signals mit anderen Frequenzen ohne Unterbrechung (z.B. „Knacken“) nötig, komplexe Berechnungen während der Laufzeit
- falls Experiment wiederholt wird: lange Konvergenzzeit, da jede Iteration 30 Sekunden bis mehrere Minuten dauert
- Probleme des Algorithmus, in der Nähe der Resonanzfrequenzen eine Lösung zu finden
- gute Kenntnis des Zeitverhaltens der Regelung und der Komponenten notwendig, um Phasenverschiebung genau zu treffen

4.4 Integration in den Versuchsaufbau

Da insbesondere der analoge Eingang des bestehenden Aufbaus eine extrem gute Auflösung hat und die weitere Datenverarbeitung bereits für die Messkarte implementiert ist, ist es naheliegend, diesen wenn möglich weiter zu verwenden. Diese Überlegung hat den Entwurf der folgenden Ansätze inspiriert.

4.4.1 Analoge zwischengeschaltete Regelung

Die Regelung wird zwischen den analogen Ausgang und den Leistungsverstärker geschaltet (siehe Abbildung 4.7). Auch wenn die Regelung mit analogem Ein- und Ausgang arbeitet, kann sie intern

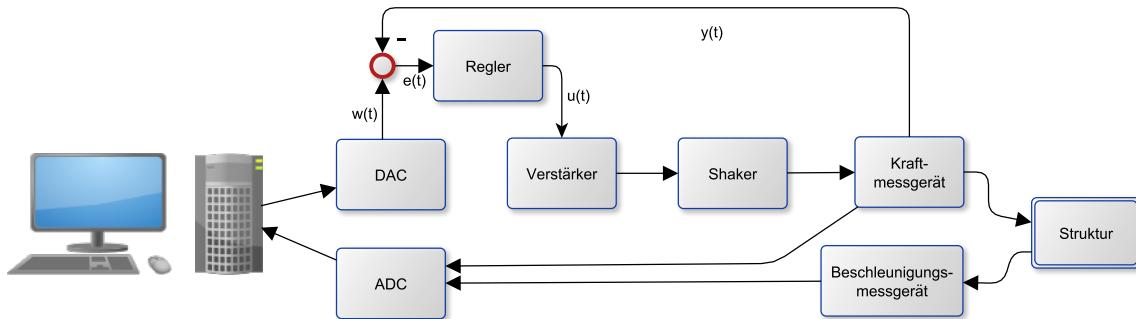


Abbildung 4.7: analoge zwischengeschaltete Regelung

natürlich auch digital umgesetzt werden.

Vorteile bei dieser Variante:

- sehr einfache, klare Schnittstelle. Keine Kommunikation auf höherer Abstraktionsebene erforderlich
- Nutzung der bestehenden Hardware für Ein- und Ausgänge
- gute Modularität: Regelung kann als eigene Komponente gut vom restlichen Aufbau abgetrennt oder zugeschaltet werden

Nachteile bei dieser Variante:

- Da keine abstrakten Informationen ausgetauscht werden ist nur eine Regelung im Zeitbereich möglich.
- analoges Signal muss eingelesen und wieder ausgegeben werden. Dadurch erhöhen sich evtl. Störungen durch Signalamwandlung oder -verarbeitung.

4.4.2 Regelung mit digitalem Eingang und analogem Ausgang

Abbildung 4.8 zeigt schematisch den Aufbau einer Regelung, die gleichzeitig auch die Erzeugung des analogen Signals übernimmt. Der Regler wird vom Computer aus digital gesteuert, insbesondere werden keine Daten in Echtzeit übertragen, sondern hauptsächlich Befehle zur Steuerung (gestrichelte Linie).

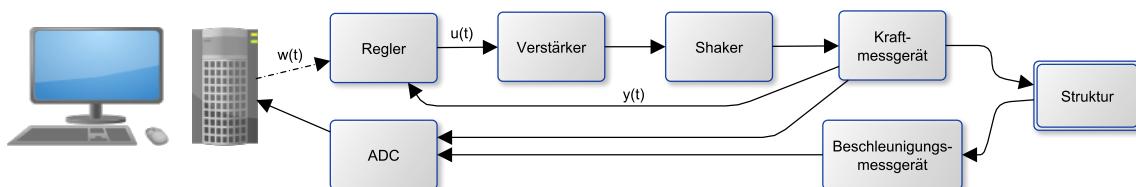


Abbildung 4.8: Struktur der Regelung mit Signalerzeugung im Regler

Vorteile:

- Soll–Istwert–Vergleich in Echtzeit möglich
- Verwendung bestehender analoger Leistungsverstärker und Messwertaufnehmer möglich

Nachteile:

- hohe Anforderungen an den DAC des Reglers, da dieser das Signal erzeugt

4.5 Hardware für die Regelung

Unterschiedliche Hardware wurde für die Realisierung der Regelung betrachtet: von diskreten analogen Bausteinen bis zu komplexen Controllern oder Einplatinen–Computern.

4.5.1 analoge Bausteine

Insbesondere die analoge zwischengeschaltete Regelung könnte gut mit einfachen analogen Schaltkreisen wie Operationsverstärkern realisiert werden. Bauteile von ausreichender Genauigkeit und zeitlicher Auflösung sind handelsüblich und ohne größere Kosten zu erwerben. Vorteilig zu erwähnen wären hohe Geschwindigkeiten und gutes Echtzeitverhalten, da natürlich keine Verarbeitungszeiten wie in einem Prozessor anfallen.

Da allerdings die Parameter der Regelung in diesem Fall durch die verwendeten Bauteile und die Struktur des Reglers durch die Verschaltung festgelegt werden, ist eine aus einzelnen analogen Bauteilen aufgebaute Regelung nur mit hohem Aufwand und viel Fachkenntnis zu ändern. Außerdem lassen sich mit einfachen Mitteln nur einfache Reglerstrukturen umsetzen (z.B. PID–Regler). Komplexere Reglerstrukturen, wie etwa Frequenzbereichsverfahren oder die lernende Regelung beruhen auf umfangreichen mathematischen Operationen, welche mit analogen Schaltungen nur sehr schwer abzubilden sind. Aus diesen Gründen wurde die vollständig analoge Umsetzung der Regelung verworfen.

4.5.2 Arduino

Arduino ist ein seit 2005 bestehendes OpenSource Projekt zur Entwicklung von Mikrocontroller–Programmen und Ansteuerung von Ausgängen und Einlesen von Eingängen. Das Arduino Projekt besteht aus käuflich erwerbbaren Boards. Diese Boards sind vorgefertigte Platten mit einem Controller, der nötigen Beschaltung, Stromversorgung, Anschlüssen an den PINs des Controllers und meist auch einer integrierten Programmierschnittstelle zum Computer. Außerdem hat das Arduino Projekt speziell für die Arduino Boards eine eigene Programmierumgebung (IDE) entwickelt. Die Programmierung erfolgt in C/C++, wobei die IDE bereits zahlreiche Befehle und Bibliotheken zur Verfügung stellt [4]. Der Begriff Arduino wird häufig für die Boards und die Programmierumgebung synonym verwendet.

Mittlerweile existieren für unterschiedliche Anwendungsbereiche eine Vielzahl von unterschiedlichen Arduino Boards. Für die Steuerung des Shakers muss ein analoges Signal erzeugt werden. Das ArduinoDue Board [2] ist das einzige Board mit nativen analogen Ausgängen. Prinzipiell wäre es natürlich möglich, mittels Pulsweitenmodulation und nachgeschaltetem RC–Glied auch aus den digitalen Ausgängen ein analoges Ausgangssignal zu erzeugen, oder über eine geeignete Schnittstelle einen DAC als externes Bauteil anzusteuern. Aus praktischen Gründen wurden diese Optionen allerdings vorerst nicht weiter untersucht, sondern die Möglichkeiten des in den ArduinoDue integrierten DACs evaluiert.

4.6. Bewertung der Regelungsansätze und Entscheidung für ein Vorgehen

Das ArduinoDue Board basiert auf dem SAM3X8E von Atmel [51]. Die für dieses Projekt wichtigsten technischen Daten sind in Tabelle 4.4 aufgelistet.

Tabelle 4.4: Technische Daten des SAM3X8E

Eigenschaft	Wert	Einheit
Architektur	ARM Cortex M3	
Taktrate	84	MHz
interner Speicher	512	kB
Auflösung DAC	12	bit
Aktualisierungsrate DAC (Implementierung der Arduino IDE, experimentell ermittelt, laut Datenblatt 1 MHz möglich [6])	ca. 170	kHz
Schrittweite DAC bei 0...2V Aussteuerung	0,5	mV
Auflösung ADC	12	bit
Aktualisierungsrate ADC	max 1	MHz

4.5.3 STM32F4

STMicroelectronics bietet zu ihrer Chipsatz-Familie STM32F4 mit ARM Cortex M4 Architektur unterschiedliche Evaluations-Boards zu sehr gutem Preis-Leistungsverhältnis an [51]. Die Ressourcen sind ähnlich wie zum ArduinoDue. Insbesondere die DACs haben gleiche Kennzahlen. Es gibt Varianten mit etwas mehr Speicher.

4.5.4 Raspberry Pi

Der Raspberry Pi ist ein OpenSource Einplatinencomputer [46]. Verglichen mit den Arduino-Boards ist die Rechenleistung und Speicherkapazität enorm. Allerdings ist die Architektur sehr komplex und das Echtzeitverhalten wie auch bei anderen modernen Computern mehr oder weniger unbestimmt. Der Raspberry Pi verfügt über zwei 16 bit analoge Ausgänge. Da diese aber über das Audio-Modul angesteuert werden, welches Daten wiederum zwischenpuffert sind Latenzen und Jitter unklar. Dadurch könnte die Ansteuerung für Mess- und Regelungsaufgaben kompliziert werden. Alternativ zum Audio-Modul könnte über die vorhandene SPI-Schnittstelle ein externer DAC angebunden werden.

Die Regelung mit dem Raspberry Pi zu entwerfen wäre sehr vielversprechend, da zwei hochauflösende DACs und Rechenkapazität für umfangreiche mathematische Operationen zur Verfügung stehen. Die Architektur ist allerdings sehr komplex weshalb die Echtzeit-Hardware-Ansteuerung als eigenes Thema vorher untersucht werden müsste.

4.6 Bewertung der Regelungsansätze und Entscheidung für ein Vorgehen

4.6.1 Entscheidung für einen Reglertyp

Die iterativ lernende Regelung ist vielversprechend in Bezug auf die erreichbare Regelgüte, insbesondere da durch Ausnutzung der Periodizität des Signals auch aklausale Lerngesetze möglich sind. Außerdem verspricht sie Stabilität und Robustheit mit wenigen Reglerparametern einhalten

zu können, auch wenn nur eine geringe Kenntnis der Streckenübertragungsfunktion im Vorfeld gegeben ist. Deshalb wurde entschieden, die Regelung mit einer ILR aufzubauen.

4.6.2 Entscheidung für analoge oder digitale Umsetzung des Reglers

Durch eine komplett analoge Bauweise der Regelung könnte nur eine kausale Regelung im Zeitbereich realisiert werden. Eine iterativ lernende Regelung basiert auf der Speicherung und digitalen Verarbeitung von Ein- und Ausgangsdaten, die analog nicht umzusetzen ist. Daher wurde die analoge Realisierung des Reglers für diese Arbeit verworfen.

Falls jedoch eine iterativ lernende Regelung scheitert oder mit einer Regelung im Zeitbereich kombiniert werden soll wäre eine analoge Regelung wegen ihrer schnellen Reaktionszeit und ihrem exakt bekannten Echtzeitverhalten auf jeden Fall erneut zu evaluieren.

Es wurde entschieden, die Regelung digital mit integrierter Signalerzeugung aufzubauen. Da es schwierig ist, Daten aus den bestehenden LabView Messkarten in Echtzeit zu exportieren, zu verarbeiten und ohne „Knacken“ neue Daten unterbrechungsfrei in der nächsten Periode zu verwenden, muss die Signalerzeugung in der Regler-Hardware erfolgen. Nur so ist ein Soll–Istwert–Vergleich mit anschließender Verarbeitung der Signale zu neuen Stellgrößen in Echtzeit möglich.

4.6.3 Entscheidung für eine Hardwareplattform

Es wurde entschieden, die Regelung auf einem ArduinoDue zu entwickeln, da dies ein hohes Maß an Kontrolle über das Echtzeitverhalten und gleichzeitig durch die Arduino IDE eine einfache Programmierung auch für spätere Anwendung bzw. Änderung ermöglicht.

Die Leistungsfähigkeit des ArduinoDue ist mittelmäßig verglichen mit anderen heute erhältlichen Prozessoren. Die Taktrate von 84 MHz und die 32 bit Architektur bieten eine gute Grundlage für Berechnungen zur Laufzeit. Die Auflösung der integrierten DAC und ADC ist hoch aber noch unter der Auflösung von professionellen Messkarten. Dafür ist der ArduinoDue vergleichsweise günstig in der Anschaffung und die OpenSource Lizenz sichert langjährige Verfügbarkeit und einen tiefen Einblick in die Wirkungsweise der internen Funktionalitäten. Aus diesen Gründen stellt der ArduinoDue eine gute Basis für einen Prototypen dar.

5 Umsetzung der Regelung

5.1 Implementierung

Herausforderung für die Implementierung der Regelung auf einem Mikrocontroller waren die Anforderungen an das Echtzeitverhalten der Regelung und eine hohe Geschwindigkeit der Datenverarbeitung, sodass möglichst hohe Sampleraten erreicht werden können. Es musste also ein Weg gefunden werden, das Einlesen der Regelgröße, das Lerngesetz und das Ausgeben der Stellgröße effektiv und robust auf einem Mikrocontroller abzubilden.

Zunächst wurde die Regelung mit einem doppelten Ringspeicher entworfen, um maximale Echtzeitfähigkeit zu erreichen. Grundidee war, dass der Inhalt eines Speichers immer wiederholt wird, bis die Berechnung neuer Werte abgeschlossen ist. Dann wird zwischen den Speichern umgeschaltet und der Ringspeicher mit den neuen Werten wird für die Ausgabe verwendet. Ein Flussdiagramm für diesen Ansatz ist in Abbildung 5.1 dargestellt.

Diese Vorgehensweise wurde implementiert und erprobt. Es konnten zwei Nachteile beobachtet werden: Zum einen ist die Regelung nicht so schnell wie theoretisch bei einer repetitiven Regelung möglich, da Änderungen frühestens in der übernächsten Periode von der Regelung ausgegeben werden. Die Ränder der Perioden stellen ebenfalls eine Schwierigkeit dar, da die Implementierung der Phasenverschiebung im Lerngesetz an den Enden der Speicher schwer zu realisieren ist. Bei längeren Berechnungen könnte es außerdem schwierig sein, dass noch während der Berechnung bereits neue Fehlerwerte eingelesen werden und dadurch das Lerngesetz auf unstetige Werte zurück greifen könnte.

Aus diesen Gründen wurde das Konzept grundlegend überarbeitet: Es wird nur ein Ringspeicher jeweils für die Fehlerwerte und das auszugebende Signal verwendet. Dieser Ringspeicher beinhaltet eine Periode des auszugebenden Signals bzw. der Fehlerwerte. Der Aufbau des Ringspeichers ist schematisch in Abbildung 5.2 dargestellt. Sowohl in der Abbildung als auch im folgenden Text werden in **Blockschrift** die Bezeichnungen aus dem Quelltext verwendet.

Die **Nval** Zeitschritte jeder Periode laufen im Uhrzeigersinn von 0 bis **Nval** – 1. Für jeden Zeitschritt **timeindex** liegt ein auf dem DAC auszugebender Wert im Array **outputSignal** vor. Die Inkrementierung des Zeitschritts erfolgt durch einen internen Interrupt. Dieser wird direkt aus dem Prozessortakt gespeist und ist damit quarzstabil. Für die Implementierung der internen Interrupts wurde die DueTimer Bibliothek [49] verwendet. Aus dem Soll–Istwert–Vergleich wird für jeden Zeitschritt ein Fehlerwert berechnet, der im Array **error** gespeichert wird. Der nächste auf dem DAC auszugebene Wert in **outputSignal** (gelb hinterlegt) wird mittels des Lerngesetzes **updateLaw** aus dem hellgrau hinterlegten Fenster mit der Breite $i_f - i_0$ (**Nsmooth**) berechnet (vgl. auch Gleichung 5.1). Das Fenster liegt genau eine Periode minus Phasenverschiebung λ (**PhaseLead**) zurück.

Vorteil dieser Implementierung ist die schnelle Reaktionszeit, da bereits in der nachfolgenden Periode aktualisierte Werte verwendet werden. Außerdem weist sie eine geringe Speichernutzung auf, da nur eine Periode der Fehler- und Ausgabewerte gespeichert werden müssen.

Einschränkung dieser Implementierung ist, dass die Fensterbreite **Nsmooth** maximal so groß sein darf wie die Phasenverschiebung **PhaseLead**. Außerdem bezieht sich die Phasenverschiebung nicht auf den eigentlichen Mittelwert aus dem **Nsmooth** breiten Fenster, sodass die reale Phasenverschiebung eigentlich **Phaselead** – **Nsmooth**/2 ist. Änderung der Implementierung zu einer expliziten

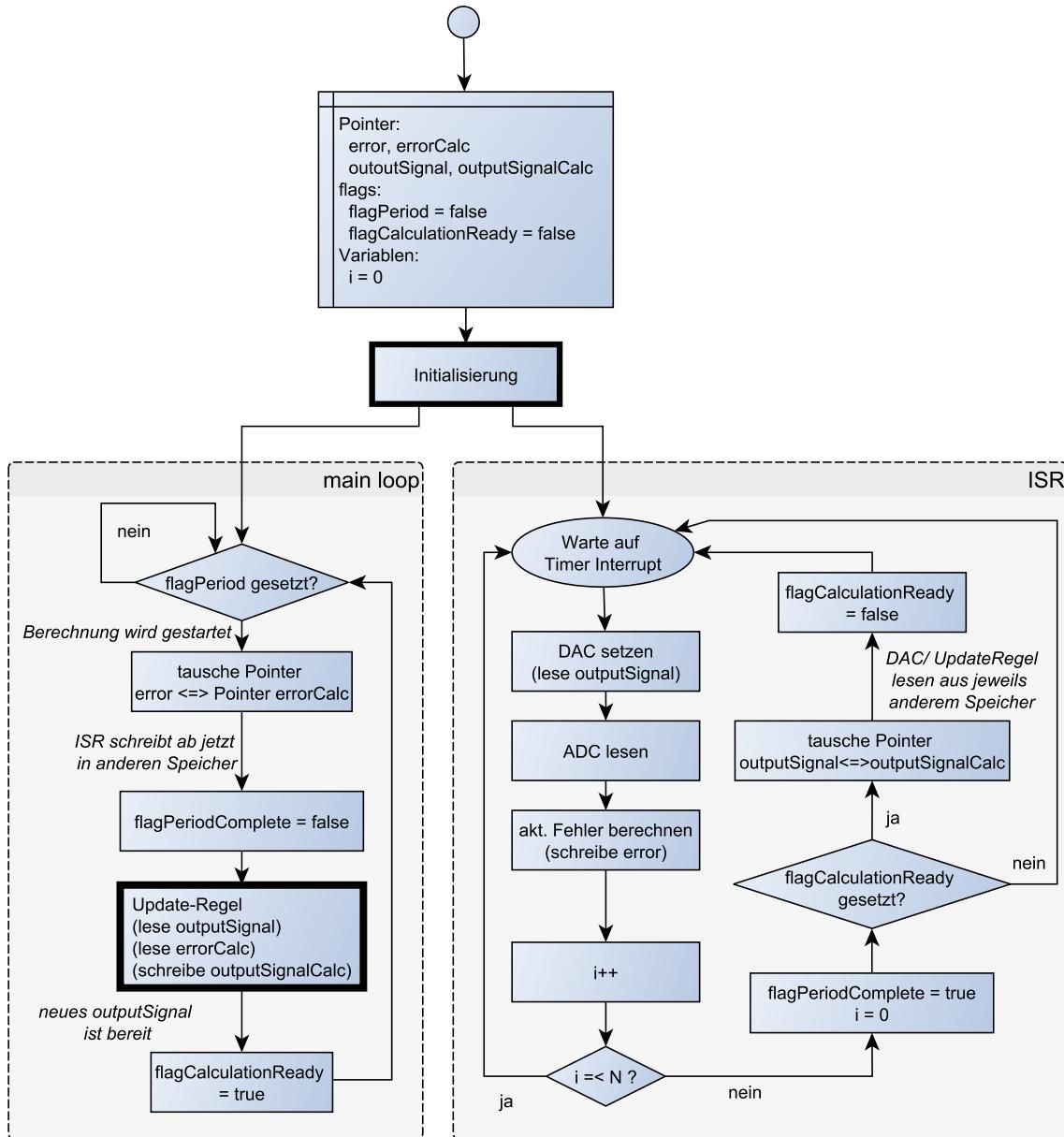


Abbildung 5.1: Flussdiagramm für den ersten Versuch der Implementierung einer repetitiven Regelung mittels zweier Ringspeicher

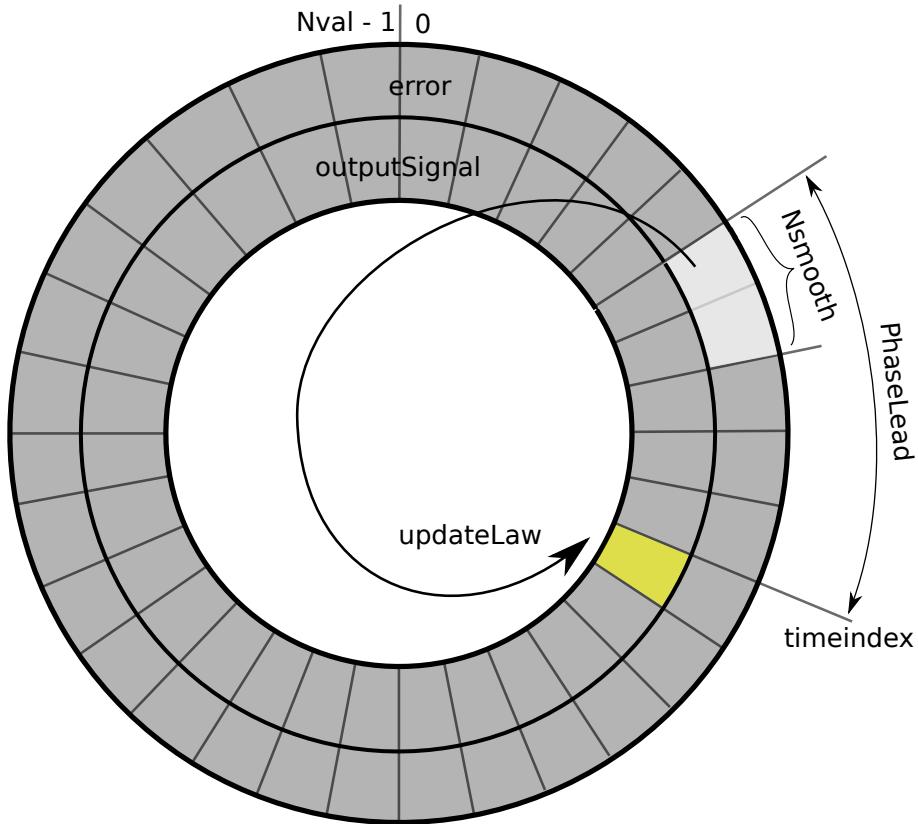


Abbildung 5.2: schematische Darstellung Ringspeicher für repetitive Regelung und Anwendung des Lerngesetzes

Angabe der „echten“ Phasenverschiebung, also bis zur Mitte des Fensters für den gleitenden Mittelwert wäre möglich, wurde allerdings aufgeschoben.

Es wurde zunächst ein Lerngesetz wie in Gleichung 3.25 implementiert. Allerdings wurde eine relativ lange Konvergenzzeit und teilweise Überschwingungen in der Amplitude beobachtet. Nach dem Einschalten der Regelung war die Amplitude des Signal zunächst viel zu klein, dadurch baute sich ein großer Wert in der Stellgröße auf, welcher nach Erreichen des Sollwerts erst langsam auf das eigentlich nötige Maß zurück fiel. Das Implementierte Lerngesetz lässt sich als reiner I-Regler interpretieren. Das beobachtete Verhalten ist von I-Reglern her bekannt und kann mit Methoden wie z.B. dem Anti-Windup (vgl. z.B. [53, Kap.3]) behandelt werden. Außerdem erschwerte die implizite Abhängigkeit des Tiefpassverhaltens aus der „Summenbreite“ $i_f - i_0$ und der Verstärkung $\phi(i)$ immer wieder die Einstellung.

Um die beiden oben genannten Probleme zu beheben wurde das Lerngesetz aus Gleichung 3.25 in zweierlei Hinsicht erweitert:

- Um bei geändertem Ausgangssignal schneller eine Änderung zu bewirken ohne das typische Überschwingen eines großen I-Anteils in Kauf nehmen zu müssen wird das Lerngesetz um einen P-Anteil erweitert.
- Die Summe wird auf die „Summenbreite“ $i_f - i_0$ normiert, um Tiefpassverhalten und Verstärkung getrennt voneinander einzustellen zu können. Der Einfluss unterschiedlicher $\phi(i)$ wurde noch nicht untersucht, sondern alle $\phi(i)$ gleich gesetzt und zu einem Reglerparameter K_p bzw. K_i zusammen gefasst, um die Analogie zu einem PI-Regler zu unterstreichen.

So entstand auf Grundlage der Gleichung 3.25 mit den oben genannten Erweiterungen das implementierte Lerngesetz in Gleichung 5.1.

$$u(k) = \frac{K_p}{i_f - i_0} \cdot \sum_{i=i_0}^{i_f} e(k - p + \lambda + i) + \frac{K_i}{i_f - i_0} \cdot \sum_{n=1}^m \sum_{i=i_0}^{i_f} e(k - n \cdot p + \lambda + i) \quad (5.1)$$

Dabei ist k der aktuelle Zeitschritt, p die Anzahl Zeitschritte in einer Periode und m die Nummer der aktuellen Periode, also k/p abgerundet. Es ist zu beachten, dass im Gegensatz zu Gleichung 3.25 die obige Gleichung keine Rekursionsgleichung mehr ist. Der erste Term bildet den P-Anteil der zweite Term den I-Anteil. Wird $K_p = 0$ gesetzt, ergibt sich Gleichung 3.25, wobei die Rekursion in die Doppelsumme umgeformt wurde. Die beiden Terme wirken jeweils auf einen phasenverschobenen, Tiefpass-gefilterten Fehlerwert, der durch die innere Summe gebildet wird. Aus Gründen der einfacheren Implementierung auf dem Mikrocontroller wurde außerdem auf die Normierung der Faktoren K_p und K_i auf die Samplefrequenz verzichtet. Vergleichbare Werte entstehen bei unterschiedlicher Samplezeit Δt nur, wenn K_p und K_i in ihrer normierten Form verwendet werden:

$$K_p^* = \frac{K_p}{\Delta t} \quad (5.2)$$

$$K_i^* = \frac{K_i}{\Delta t} \quad (5.3)$$

Ein Ausschnitt aus dem Quelltext 3 ist unten dargestellt, um die Implementierung der obigen Gleichung 5.1 auf dem Mikrocontroller etwas näher zu erläutern. Zunächst wird der gemittelte Fehlerwert `smoothedError` berechnet. Die Phasenverschiebung λ wird durch eine Funktion `indexShift` berechnet (vgl. Quelltext 6). Es ist bei der Phasenverschiebung der in Abbildung 5.2 dargestellte Ringspeicher zu beachten; sollte $i + j$ größer sein als `Nval`, wird am Anfang des Arrays weiter gezählt, wodurch der Ringcharakter des Arrays zu Stande kommt. Anschließend wird der aktuelle gemittelte Fehler einer Fehlersumme `errorSum` hinzugefügt, dies umgeht die wiederholte Berechnung der langen Summe von der Startzeit bis zur aktuellen Zeit in jedem Zeitschritt. Die eigentliche Implementierung des Lerngesetzes ist dann relativ einfach.

```
for (unsigned int j=0; j<NsSmooth; j++)
{
    smoothedError = smoothedError + error [ indexShift ( i + j ) ];
}
smoothedError = smoothedError / NsSmooth;
errorSum [ i ] = errorSum [ i ] + smoothedError ;
newOutputSignal = 2048.0 + Kp * smoothedError + Ki * errorSum [ i ];
```

In der letzten Zeile des obigen Quelltextausschnitts ergibt sich der Startwert im Sinne einer Vorsteuerung aus der halben Betriebsspannung $2^{12}/2 = 2048$ und wurde eingeführt, um die Konvergenz zu beschleunigen.

Ein Flussdiagramm ist in Abbildung 5.3 dargestellt. Nach Start und Initialisierung wird der gesamte Regler nur durch interne Interrupts gesteuert. In jeder Interrupt Service Routine wird zunächst der ADC gelesen und damit der aktuelle Fehler berechnet und abgespeichert. Anschließend wird die Update Regel aufgerufen, die ein neues Ausgangssignal zur Verfügung stellt.

In der eigentlichen Hauptroutine werden lediglich Eingaben aus der seriellen Konsole verarbeitet, die zur Steuerung eines Versuchs und zur Einstellung der Reglerparameter dienen.

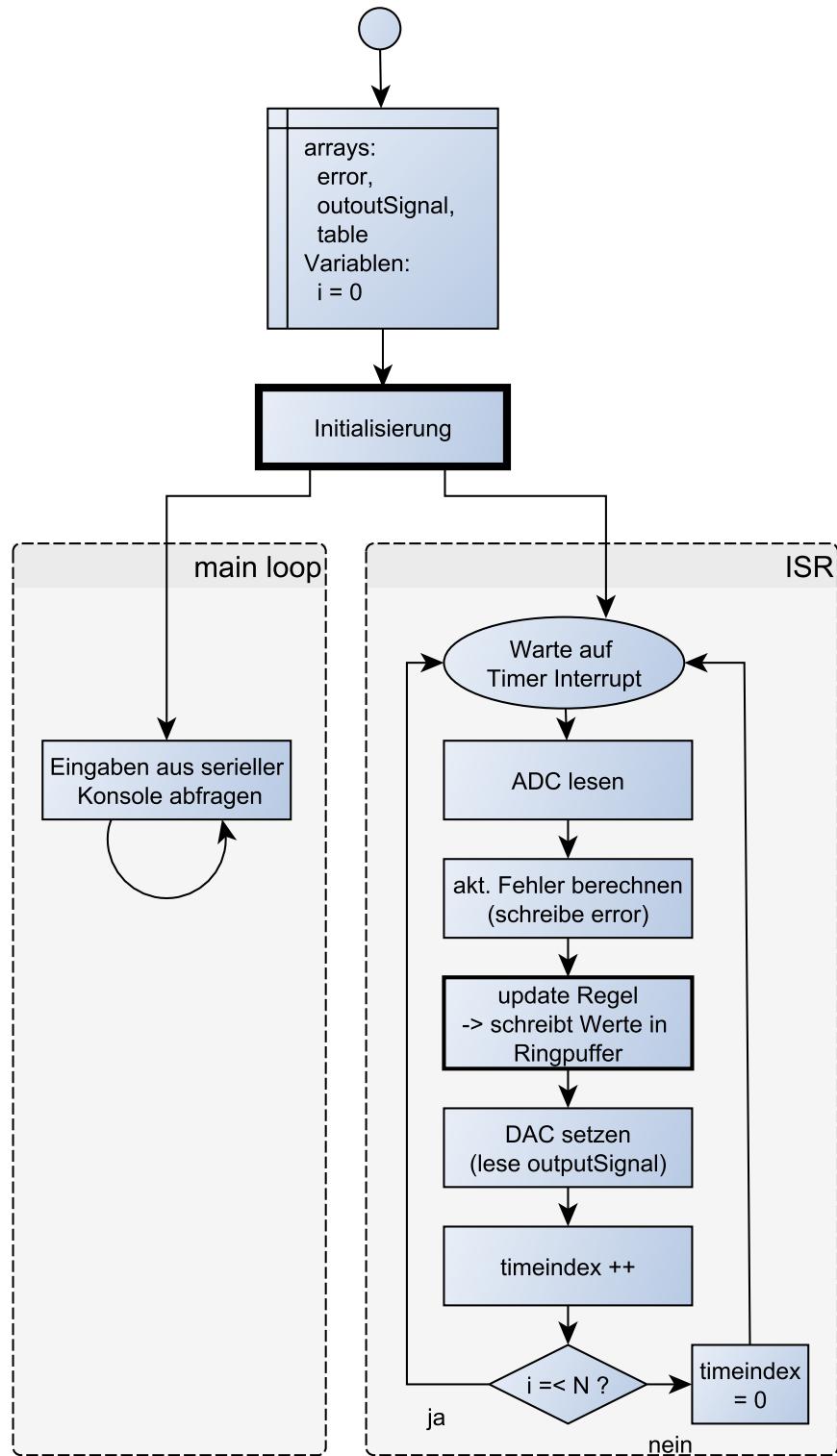


Abbildung 5.3: Finales Flussdiagramm: Implementierung mit nur einem Ringspeicher

5.2 Elektronik und Integration in den Versuchsaufbau

Die Elektronik in der Peripherie der Regelung hat mehrere Aufgaben:

- Schutz der Ein- und Ausgänge des Arduino
- Anpassung der Pegel, sodass der Aussteuerbereich der Ein- und Ausgänge gut ausgenutzt wird
- Kompatibilität des Eingangs der Regelung mit dem IEPE-Standard, der vom Piezo-Kraftsensor verwendet wird

Der IEPE Standard dient der Stromversorgung von integrierten Ladungsverstärkern für Piezo-Sensoren. Die Wirkungsweise ist in Abbildung 5.4 dargestellt. Die Messleitung wird kapazitiv entkoppelt und ein Versorgungsstrom für den Ladungsverstärker überlagert. Der Ladungsverstärker gibt das verstärkte Signal in Addition zur sich einstellenden Versorgungsspannung aus.

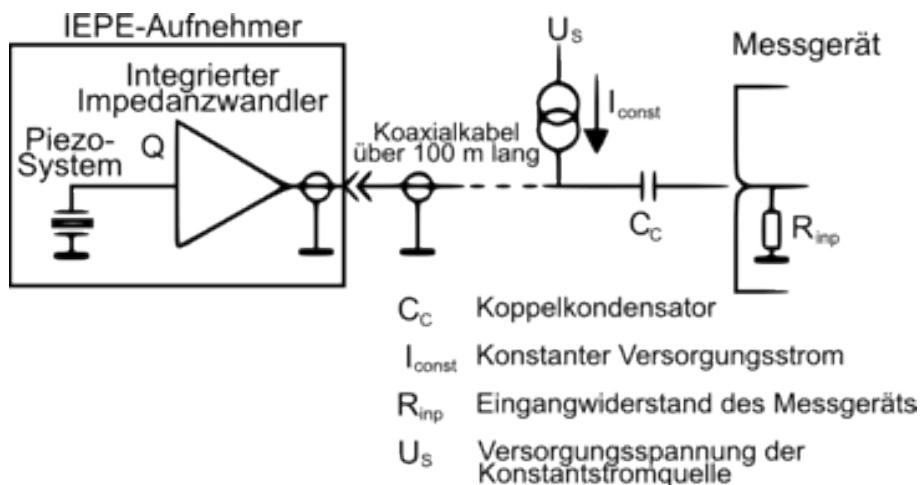


Abbildung 5.4: Schematische Darstellung IEPE (nach [56])

Da sich die Versorgungsspannung bei IEPE aus dem Versorgungsstrom einstellt, kann sie variieren. Daher kann diese Spannung aus dem Signal nicht subtraktiv entfernt werden, sondern muss mittels eines Hochpasses gefiltert werden. Für die Regelung ist das problematisch, da Gleichanteile aus der Regelung durch den Hochpass ausgefiltert werden und somit keinen Effekt haben. Die Regelung könnte also anfangen, langsam zu drifteten und die Drift würde durch den Hochpass nicht mehr erkannt werden. Es wurde im Rahmen dieser Arbeit eine Schaltung entworfen, die Kompatibilität mit dem IEPE-Standard herstellt, gleichzeitig aber durch getrennte Behandlung des Gleichanteils driften der Regelung kompensiert (siehe Abbildung 5.5). Außerdem können der Pegel des Eingangs und des Ausgangs angepasst werden, um auch unter sehr unterschiedlichen Versuchsbedingungen eine gute Aussteuerung des ADCs und des DACs zu erreichen.

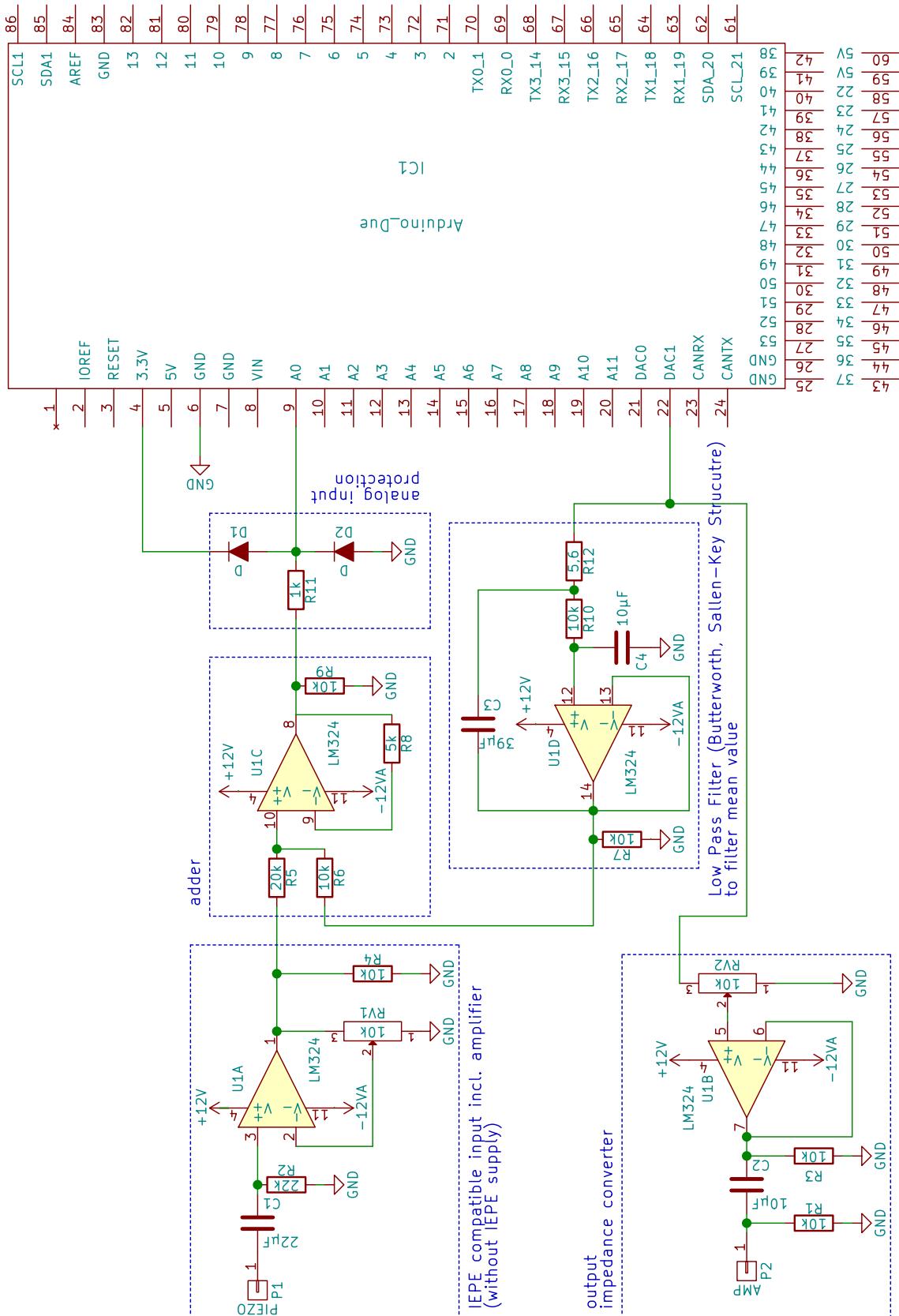


Abbildung 5.5: Schaltplan der Elektronik mit potentialfreiem Ein- und Ausgang

Grundlegende Wirkungsweise Aus dem auf dem DAC ausgegebenen Signal wird zum einen der Gleichanteil mittels eines Tiefpasses heraus gefiltert. Zum anderen wird das Signal mittels einer Verstärkerstufe gedämpft, um den Pegel an den Leistungverstärker des Aktuators anzupassen.

Das vom Piezo–Kraftsensor eintreffende Signal wird gemäß der IEPE Spezifikation durch einen Kondensator von Gleichanteilen entkoppelt. Durch eine nicht–invertierende Verstärkerstufe kann der Pegel des Signals angepasst werden. Im folgenden Schritt wird der heraus gefilterte Gleichanteil aus dem Ausgang zum Signal aus dem Eingang addiert, um so der Regelung einen Gleichanteil vorzutäuschen und Driften der Regelung zu verhindern. Es ist zu beachten, dass dementsprechend die Schaltung nicht verwendet werden kann, um Gleichanteile im Versuchsaufbau abzubilden. Da eine Messung mit Gleichanteil mit einem Piezo–Sensor aber ohnehin nicht möglich wäre, stellt dies für den hier betrachteten Versuchsaufbau keine Einschränkung dar.

Als Operationsverstärker wurde der LM324 von Texas Instrument verwendet [52]. Es handelt sich um einen günstigen handelsüblichen Operationsverstärker mit einer Bandbreite von 1 MHz (Verstärkung 1). Im Betrieb wurde experimentell ermittelt, ab welcher Frequenz die Verstärkung der Schaltung abnimmt. Dazu wurde ein 5 mV Eingangssignal ca. 400 mal verstärkt. Die Eigenschaften können bis ca. 10 kHz als konstant angenommen werden. Da die Verstärker–Schaltung im Rückkoppelzweig der Regelung sind, ist eine ganz genaue Kenntnis der Verstärkung bzw. des Phasengangs nicht essentiell, da sie durch die Regelung kompensiert werden. Falls höhere Güten oder Frequenzbereiche nötig werden, kann er einfach durch einen höherwertigen ersetzt werden.

Im folgenden werden die einzelnen Module der Elektronik einzeln näher beleuchtet. Alle Bezeichnungen beziehen sich auf den Schaltplan in Abbildung 5.5. In den Abbildungen 5.7 bis 5.9 sind die einzelnen Schritte vom Steckbrettaufbau bis zur gelösten Lochrasterplatte in einem Gehäuse dargestellt.

Auslegung des Tiefpasses Um den Gleichanteil aus dem auszugebenen Signal zu filtern wird ein Tiefpass 2. Ordnung mit Sallen–Key–Struktur (Widerstand R10 und R12, Kondensator C3 und C4) verwendet. Die Bauteile wurden so ausgelegt, dass sich ein Butterworth Filter mit Grenzfrequenz 1 Hz ergibt. Butterworth Filter zeichnen sich dadurch aus, dass ihre Verstärkung nie größer als 1 wird und sie oberhalb der Grenzfrequenz ω_0 möglichst rasch dämpfen.

Die Übertragungsfunktion eines Tiefpassfilters 2. Ordnung mit Sallen–Key–Struktur im Laplace–Bereich lautet

$$H(s) = \frac{\omega_0^2}{s^2 + 2\alpha s + \omega_0^2} \quad (5.4)$$

$$\text{mit } \omega_0 = \frac{1}{\sqrt{R_{10} \cdot R_{12} \cdot C_3 \cdot C_4}} \quad (5.5)$$

$$2 \cdot \alpha = \frac{1}{C_3} \left(\frac{1}{R_{10}} + \frac{1}{R_{12}} \right) \quad . \quad (5.6)$$

Für einen Butterworth Filter 2. Ordnung muss nun gelten:

$$\frac{2 \cdot \alpha}{\omega_0} = \sqrt{2} \quad (5.7)$$

Mit der Vorgabe der gewünschten Grenzfrequenz entsteht nach Gleichung 5.5 eine zweite Bedingung, sodass noch zwei Parameter des Filters frei gewählt werden können. Da für elektronische Bauteile nur diskrete Werte zur Verfügung stehen, gibt es bei realen Filtern leichte Abweichungen

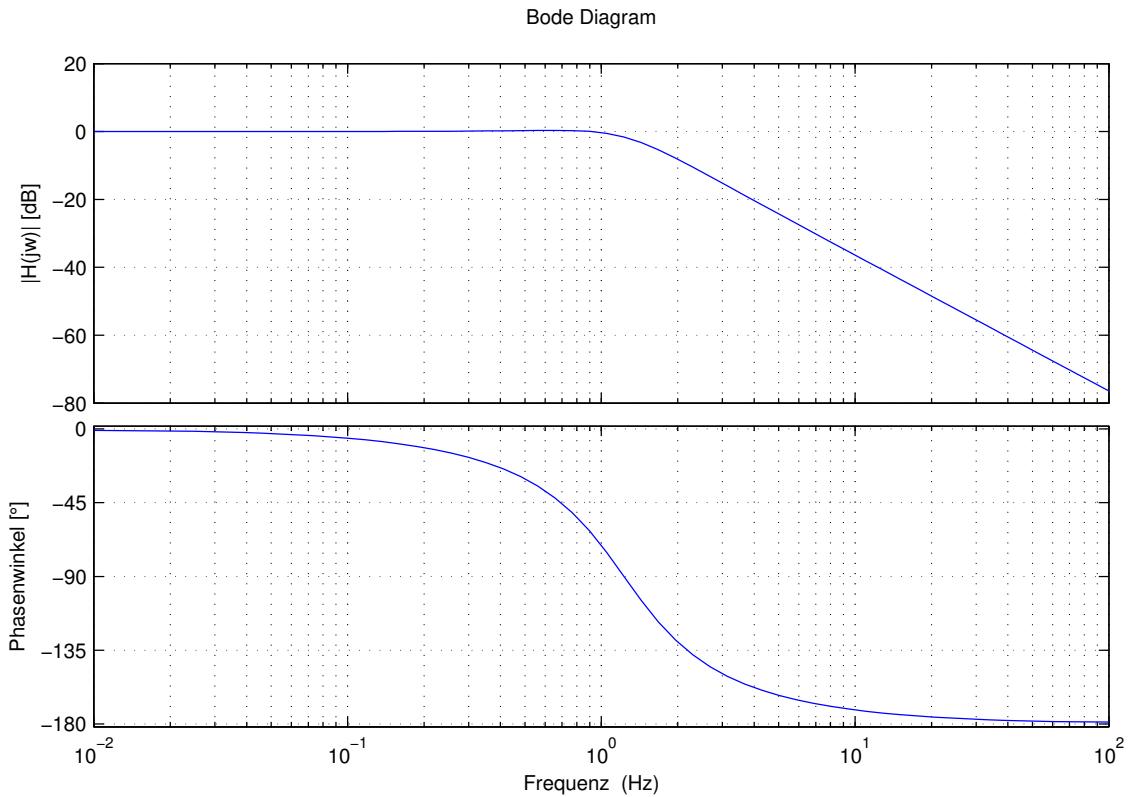


Abbildung 5.6: Übertragungsfunktion des Butterworth Tiefpassfilters 2. Ordnung

zu den idealen gewünschten Werten. Folgende Werte wurden für den Tiefpassfilter gewählt:

$$f_0 = 1 \text{ Hz} \Rightarrow \omega_0 = 2 \cdot \pi f_0 = 6.77 \text{ Hz} \quad (5.8)$$

$$R_{10} = 5.6 \text{ k}\Omega \quad (5.9)$$

$$R_{12} = 10 \text{ k}\Omega \quad (5.10)$$

$$C_3 = 30 \mu\text{F} \quad (5.11)$$

$$C_4 = 10 \mu\text{F} \quad (5.12)$$

Die Übertragungsfunktion des so umgesetzten Tiefpassfilters ist in Abbildung 5.6 dargestellt.

IEPE kompatibler Eingang Das Signal aus dem Piezo–Ladungsverstärker wird gemäß IEPE Standard kapazitiv entkoppelt (Kondensator C1). Durch die nicht–invertierende Verstärkerschaltung kann der Pegel des Signals über das Potentiometer RV1 angepasst werden, um eine gute Aussteuerung des ADC zu erreichen.

Anschließend wird mittels einer analogen Addierer–Schaltung der Gleichanteil und das Piezo–Signal addiert (Widerstände R5 und R6).

Schutz des analogen Eingangs des Arduinos Der Widerstand R11 und die Dioden D1 und D2 verhindern, dass das Signal an den analogen Eingang des Arduinos die Betriebsspannung über- und unterschreitet. Dadurch kann eine Beschädigung des ADCs auch in unzulässigen Betriebszuständen verhindert werden.

Ausgang der Schaltung Den Ausgang der Schaltung bildet ein nicht-invertierender Verstärker. Da der Leistungsverstärker für den Aktuator sehr kleine Spannungen aufnimmt wurde hier eine Schaltung realisiert, die das Signal dämpft und maximal eine Verstärkung 1 realisieren kann. Der Ausgang wurde außerdem aus Sicherheitsgründen kapazitiv entkoppelt (C2) und durch den Widerstand R1 auf Masse gelegt, um sowohl im ausgeschalteten als auch eingeschalteten Zustand ein gutes Null-Niveau zu erreichen.

5.3 Schnittstellen zum bestehenden Messaufbau

Es gibt drei Schnittstellen zum Messaufbau: einen logischen Eingang der Regelung, einen analogen Ausgang zum Shaker und einen analogen Eingang vom Piezo Kraftsensor (vgl. Abbildung 4.8).

Die Signalerzeugung geschieht im Arduino. Das gemessene Kraftsignal wird in die Regelung eingespeist, sodass die Regelung den Regelfehler berechnen kann. Die Messwerterfassung zur Auswertung des Versuchs geschieht durch die bestehende Messkarte.

logischer Eingang Der USB-Port des Arduinos dient zur Steuerung eines Versuchs und zur Einstellung der Parameter. Die Tabelle 5.1 zeigt eine Übersicht über die Steuerungs- und Einstellungsmöglichkeiten.

Befehle an die Regelung sind zweistufig aufgebaut: Zunächst wird durch die erste Ziffer der entsprechende Menüeintrag geöffnet. Anschließend wird der gewünschte Wert eingetragen.

Tabelle 5.1: Befehle an die Regelung über die serielle Konsole

Menü	Auswahl	Kommentar
1	1	Experiment starten
	2	Experiment stoppen (Werte in <code>error</code> , <code>errorSum</code> und <code>outputSignal</code> werden beibehalten)
	3	Experiment resetten (Werte in <code>error</code> , <code>errorSum</code> und <code>outputSignal</code> werden zurück gesetzt)
2	X	Sample Frequenz auf X Hz einstellen
3	X	K_i auf X/1000 einstellen (999 für 0 eingeben)
4	X	K_p auf X/1000 einstellen (999 für 0 eingeben)
5	X	Phasenverschiebung λ auf X Zeitschritte einstellen
6	X	Fensterbreite für Mittelwert auf X Zeitschritte einstellen

Es gibt zahlreiche Möglichkeiten, die serielle Konsole über den USB-Port eines Computers zu nutzen. Insbesondere gibt es keinerlei Restriktionen aus der Software, Systemarchitektur oder verwendeten Plattform. Die serielle Konsole kann u.a. aus der Arduino IDE oder aus MATLAB® (Befehl `serial()`, siehe auch MATLAB® Skript zur Steuerung eines Sweep-Experiments in Quellcode 10) angesprochen werden.

Das Interface über die serielle Konsole gibt Antworten zur Bestätigung und für Hilfestellungen zurück. Wird das Interface automatisch z.B. über ein MATLAB® Skript gesteuert, müssen diese Antworten abgeschaltet werden. Dies geschieht durch Änderung der `ILR.ino` (vgl. Quelltext 1) und erneutes Kompilieren der Firmware.

- `getParamValuesFromSerial(false);`
Serielle Konsole gibt keine Antworten, Steuerung z.B. über ein MATLAB® Skript
- `getParamValuesFromSerial(true);`
Serielle Konsole gibt Antworten, manuelle Steuerung

Ausgang zum Shaker (AMP) Am Ausgang der Regelung wird der Shaker–Verstärker angeschlossen. Über das Potentiometer kann der Pegel bei Bedarf angepasst werden.

Eingang Piezo Der Eingang der Regelung ist IEPE kompatibel. Der Versorgungsstrom für den integrierten Ladungsverstärker im Piezo Kraftsensor wird allerdings nicht durch die Schaltung der Regelung zur Verfügung gestellt, sondern durch die bestehende Messkarte, die auch zur Aufzeichnung der Messwerte verwendet wird.

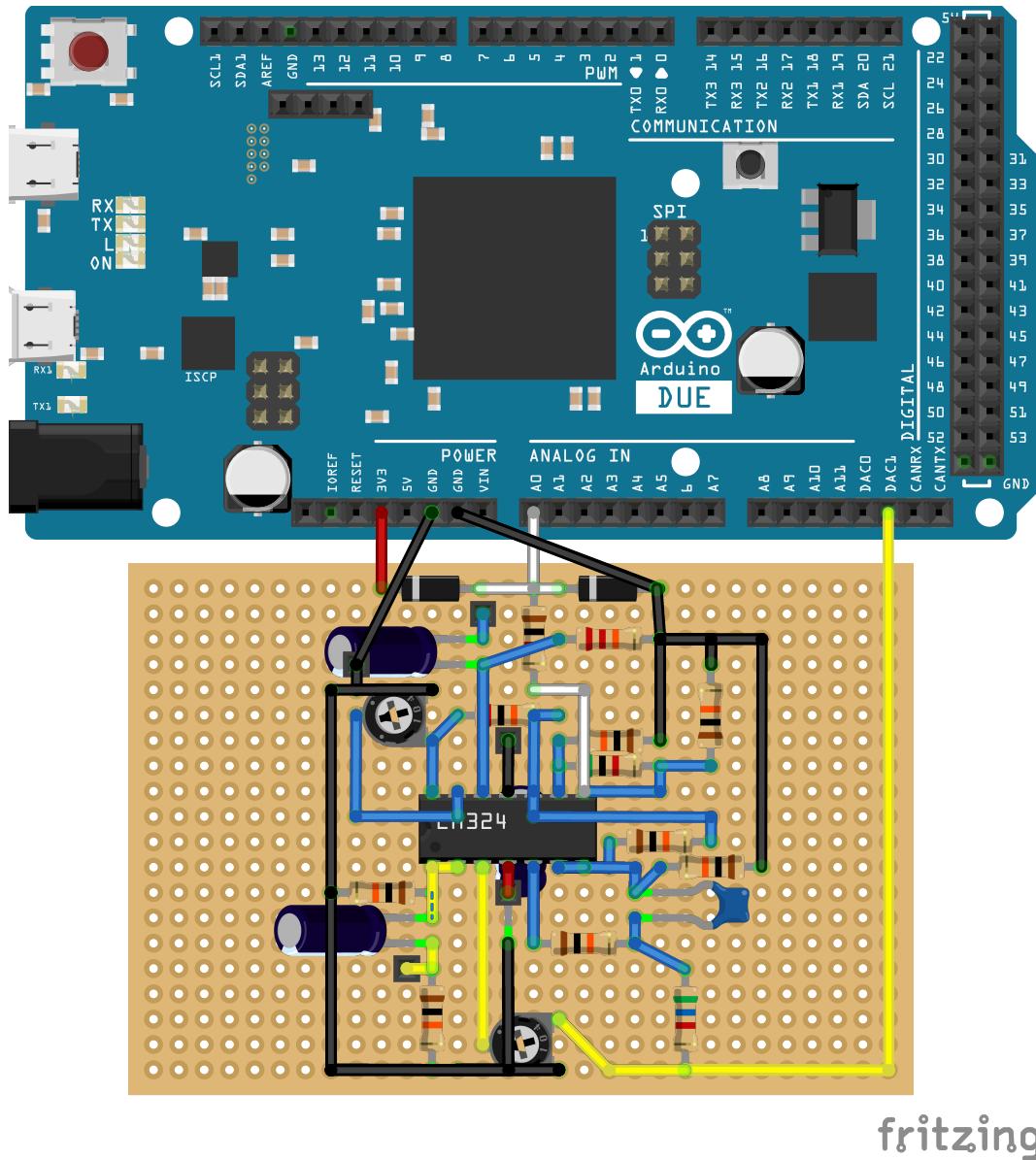


Abbildung 5.7: Lochraster Layout der Elektronik zur Pegelanpassung und Kompatibilität mit dem IEPE Standard des integrierten Piezo–Ladungsverstärkers

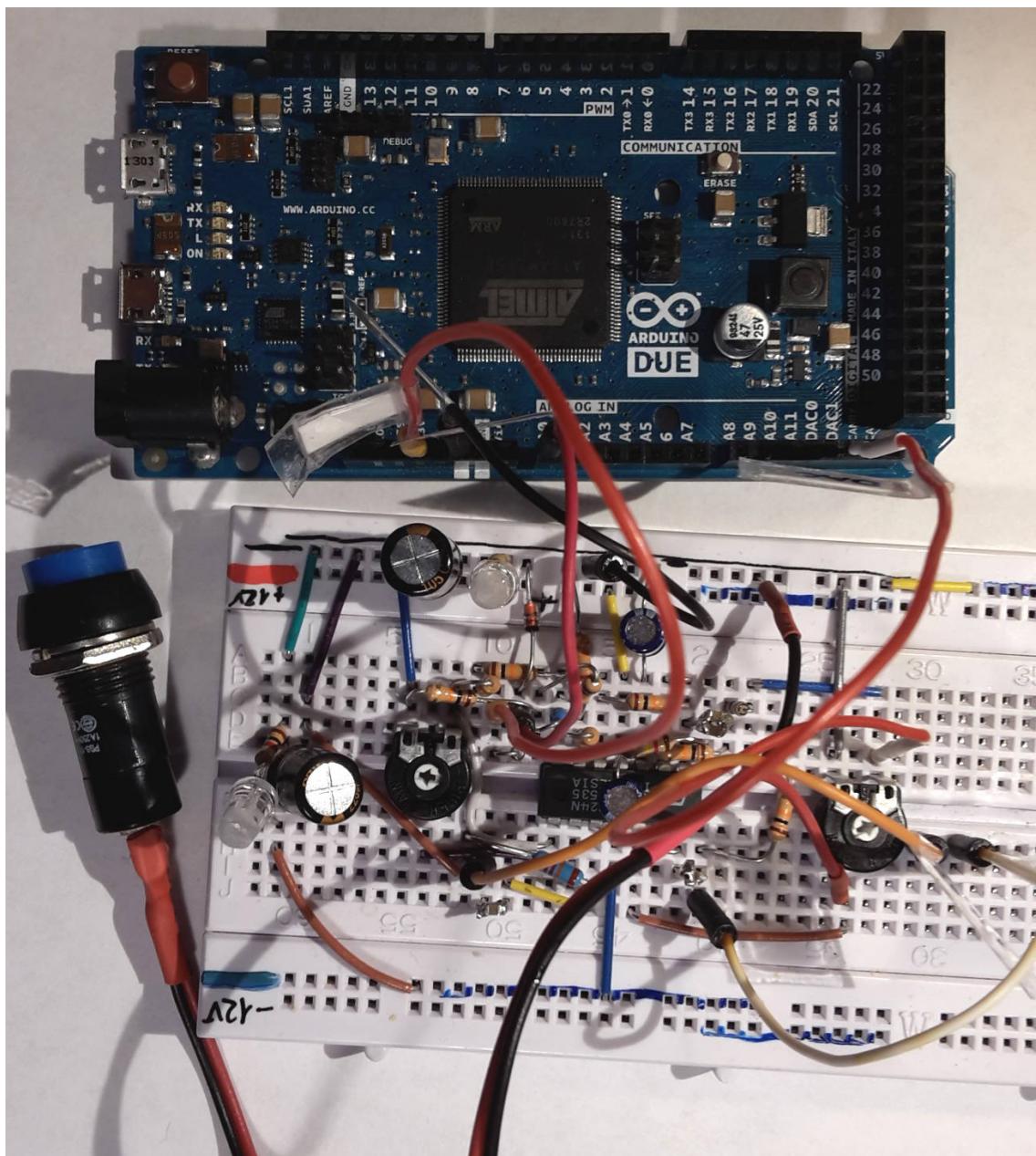


Abbildung 5.8: Steckbrettaufbau der Elektronik

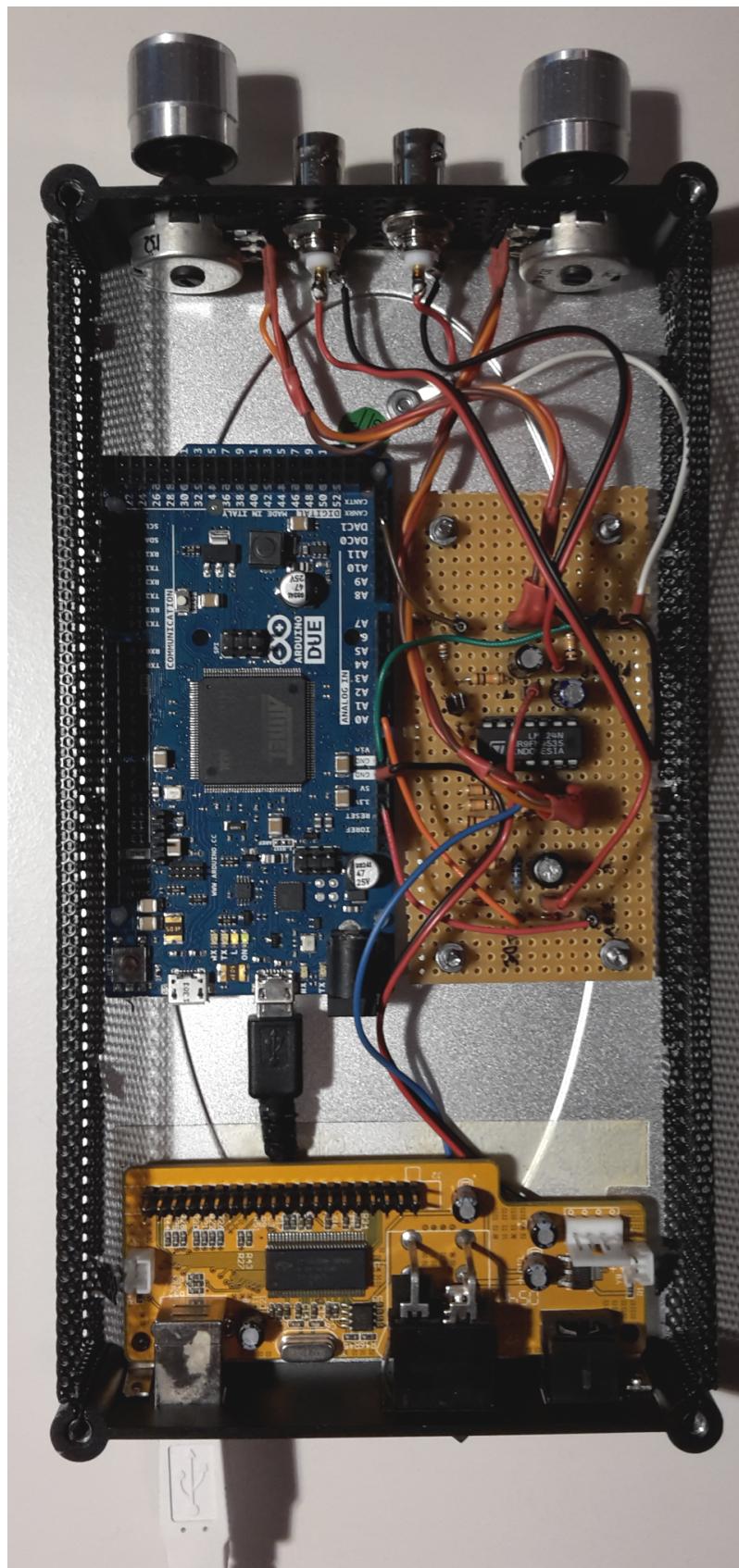


Abbildung 5.9: Integration der Elektronik und des Arduinos in ein Gehäuse (Platine ganz unten hat keine Funktion, sondern wurde nur für den USB–Stecker verwendet.)

6 Test, Verifikation und Bewertung der Regelung

6.1 Durchführung der Messungen

Um die Funktionalität des Reglers und die Regelqualität zu überprüfen wurden Messungen an zwei Schwingungs–Versuchsständen durchgeführt. Für die Bewertung der Messungen diente die vorhandene Versuchssteuerung als Referenz, die für einen Vergleich zwischen dem Ergebnis aus der Anregung mit der reinen Steuerung und der Anregung im geschlossenen Regelkreis herangezogen werden konnte. Der Versuchsaufbau ist in Kapitel 4.1 beschrieben, vgl. auch Abbildungen 4.1 bis 4.5. Das Signal des Kraftsensors wird sowohl von der Versuchssteuerung zur Aufzeichnung eingelesen als auch von der Regelung, sofern diese verwendet wird. Für eine gute Vergleichbarkeit der Ergebnisse erfolgt mit und ohne Regelung die Aufzeichnung der Messdaten durch das gleiche Gerät und mit den gleichen Einstellungen (Samplerate, Auflösung usw.)

Während der Durchführung der Messungen wurden die Parameter der Regelung angepasst. Die Bedienung des Reglers ist in Kapitel 5.3 in Tabelle 5.1 aufgeführt. Auf die einzelnen Parameter wird im folgenden näher eingegangen:

Erzeugung der Wertetabelle Die Wertetabelle enthält für jeden Zeitschritt einer Periode einen Wert der Führungsgröße. Im Rahmen dieser Arbeit wurde eine Sinus–Funktion verwendet, die Werte wurden mit einem matlab–Skript erzeugt, vgl. Quelltext 8 im Anhang. Es wurde eine Periode des Sinus–Signals in die Wertetabelle geschrieben. Dadurch ist die Periode des Reglers auch genau eine Periode des Signals. Bei Bedarf könnten aber mehrere Perioden des Signals in die Wertetabelle geschrieben werden, sodass der Regler über mehrere Perioden des Signals iteriert.

Je größer das Fenster für den gleitenden Mittelwert gewählt wird, desto höher ist der Rechenaufwand für den Prozessor. Somit ergibt sich aus der Fensterbreite eine maximale Samplerate oder anders herum. Bei den hier verwendeten Werten von `Nsmooth` funktionierte die Regelung zuverlässig bis zu einer Samplerate von ca. 35 kHz. Die Prozessorlast wurde experimentell ermittelt, indem zum Start jedes Zeitschritts ein Signal ausgegeben wurde und dieses zur Fertigstellung der Berechnungen, Ein- und Ausgaben usw. zurückgesetzt wurde. Die so ermittelte Last auf den Arduino Due beträgt bei den u.g. Einstellungen ca. 80 %. Wird die Taktrate bzw. die Fensterweite zu hoch eingestellt, schafft der Prozessor es nicht mehr, zwischen zwei Zeitschritten die nötigen Aktionen auszuführen.

Aussteuerung Die Verstärkungen des Eingangs und des Ausgangs der Regelung wurden über die beiden Potentiometer so eingestellt, dass ein möglichst hoher Dynamikbereich entsteht, ohne dass das Signal jedoch über- oder untersteuert. Dadurch wird eine gute Ausnutzung der Auflösung des ADCs und des DACs erreicht, was Voraussetzung für eine hohe Regelgüte ist.

Das Verhältnis der beiden Potentiometer–Einstellungen legt außerdem fest, in welcher Proportion die Führungsgröße und die Stellgröße zueinander stehen. Durch die Einstellung der Aussteuerung wird also auch eine Kalibrierung der Regelung ermöglicht.

PhaseLead In den Versuchen hat sich die Empfehlung aus [29] bestätigt, die Phasenverschiebung der Regelung auf die Totzeit des System einzustellen. Deshalb wurde zunächst in einem anderen Versuch ermittelt, wie viele Zeitschritte die Totzeit des System beträgt und dann dieser Wert verwendet. Variation um diesen Wert hat ergeben, dass die Regelgüte in beiden Richtungen abnimmt.

Die Totzeit des Systems kann zum Beispiel mit einem Sprungexperiment ermittelt werden. Ein Sprungexperiment an der Schiene ist in Abbildung 6.1 dargestellt. Abbildung 6.2 zeigt einen Ausschnitt aus dem Sprungexperiment, aus dem die Totzeit abgelesen werden kann. Die gemessene Totzeit wird dann entsprechend der gewählten Samplerate in eine Anzahl Samples umgerechnet.

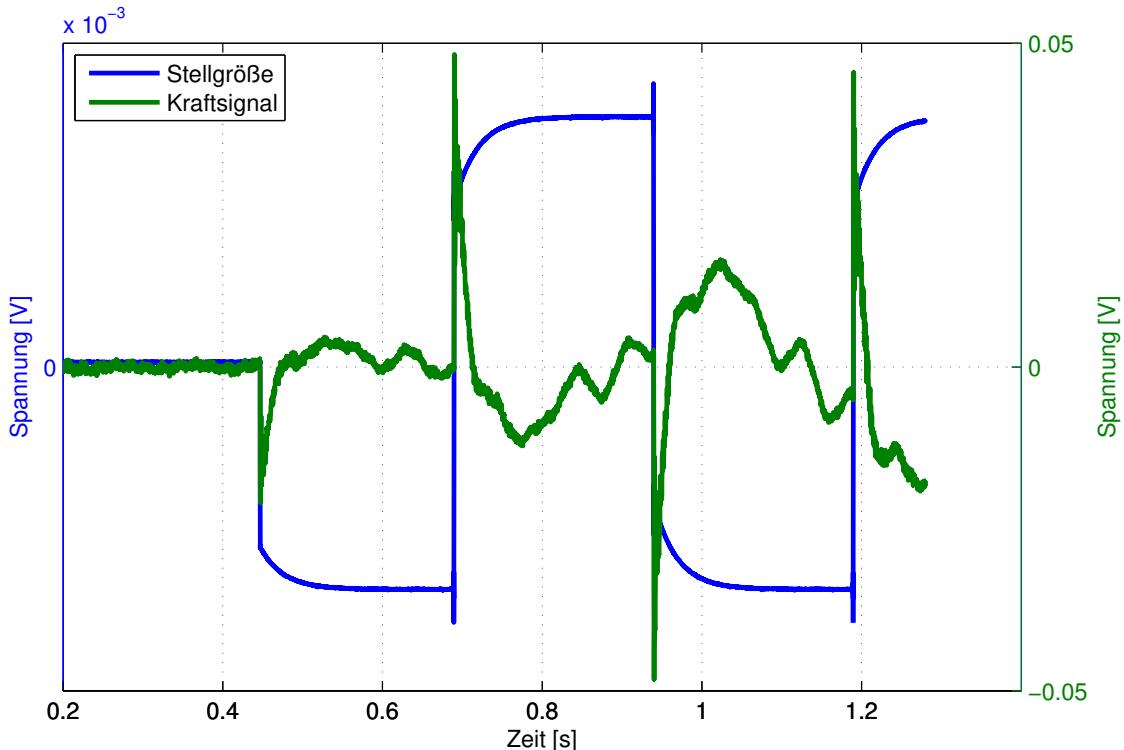


Abbildung 6.1: Sprungexperiment an der Schiene zur Bestimmung der Totzeit des Systems
— Übersicht

Einstellung Parameter K_p Faktor für proportionalen (P) Anteil der Regelung. Dieser wird zunächst möglichst groß eingestellt, da der P-Anteil die Fähigkeit der Regelung beeinflusst, schnell auf sprunghafte Störungen zu reagieren. Der P-Anteil beeinflusst die Fähigkeit der Regelung auf sprunghafte Änderungen der Stellgröße, z.B. An- oder Ausschalten oder eine Frequenzänderung zu reagieren. Für Experimente mit einer konstanten Frequenz und Amplitude ist der P-Anteil deshalb von geringerer Bedeutung, da lediglich das Einschwingverhalten beeinflusst wird (vgl. auch Abbildungen 6.3 und 6.4).

Einstellung Parameter K_i Faktor für integrierenden (I) Anteil der Regelung. Dieser wird sukzessive erhöht, da der I-Anteil die Geschwindigkeit beeinflusst, mit der der Regelfehler abfällt. Ein zu groß eingestellter I-Anteil kann den Regler allerdings instabil machen.

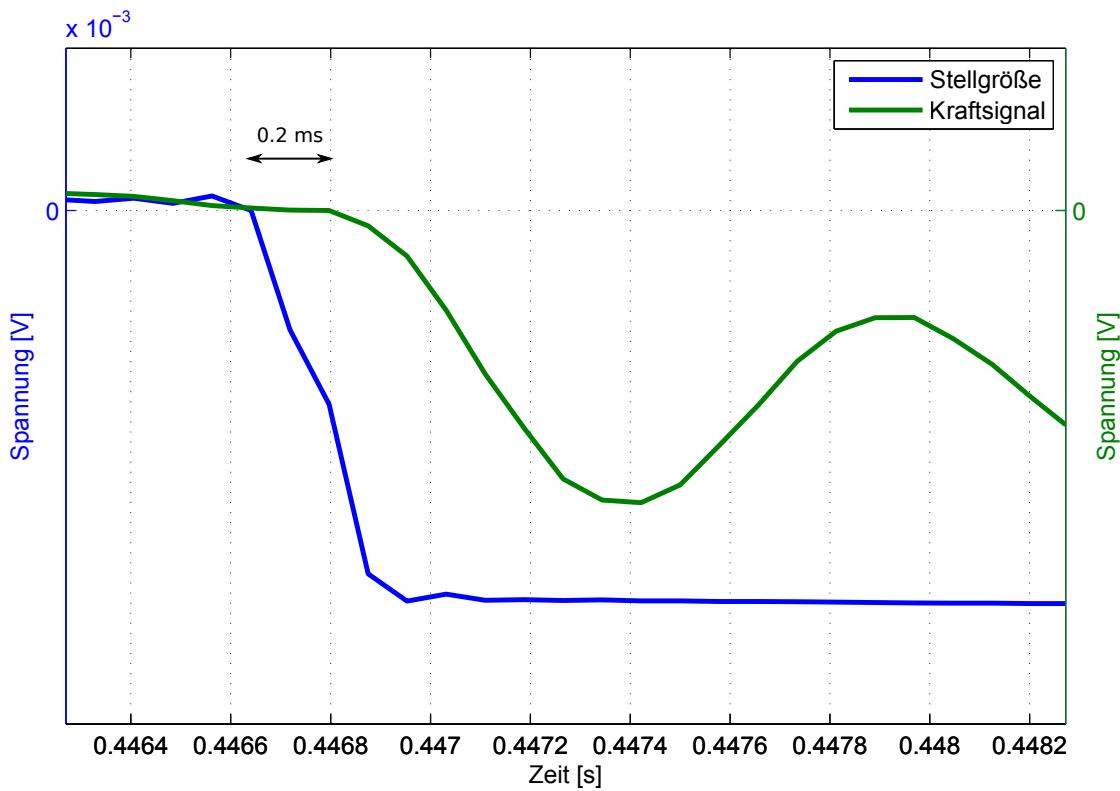


Abbildung 6.2: Sprungexperiment an der Schiene zur Bestimmung der Totzeit des Systems
— Ausschnitt

Einstellung Parameter N_{smooth} Es bietet sich an, die Fensterbreite des Tiefpassfilters als letztes einzustellen, da sie einen relativ geringen Einfluss auf die Regelgüte hat. Wird die Fensterbreite zu klein eingestellt, treten hochfrequente Schwingungen auf, die u.a. in [29] auch als Grund für die Filterung genannt werden. Wird die Fensterbreite zu groß eingestellt, nimmt die Regelgüte ab, da sich der Bereich, in dem überhaupt Störungen kompensiert werden können, verringert. Ziel ist also, den Wert für N_{smooth} möglichst klein zu wählen, ohne dass hochfrequente Störungen auftreten.

6.2 Konvergenzverhalten der Lerngesetze

Das Konvergenzverhalten des in dieser Arbeit entwickelten PI-ähnlichen Lerngesetzes (vgl. Gleichung 5.1) wurde mit dem Konvergenzverhalten von LONGMAN aus [29] (vgl. auch Gleichung 3.25) verglichen. Abbildung 6.3 zeigt den Verlauf der Summe der Fehlerquadrate aus einer Periode über die Perioden des Reglers. Der Vergleich wurde mit gleichem Wert von $K_i = 0.02$ und zwei unterschiedlichen Einstellungen von K_p durchgeführt. Zuerst wurde durch Einstellung $K_p = 0$ das Lerngesetz von LONGMAN verwendet, anschließend wurde das PI-Lerngesetz mit $K_p = 0.2$ eingestellt.

Durch die Einführung des P-Anteils kann der Fehler direkt nach Anschalten des Reglers schnell gesenkt werden, da der P-Anteil instantan wirkt. Anschließend ist das Lernverhalten ungefähr gleich. Größere Werte von K_p ergeben einen größeren Vorteil gegenüber dem Lerngesetz von Longman, allerdings wird die Regelung bei Wahl von zu großem K_p instabil. Die positive Wirkung von K_p wird auch in Abbildung 6.4 deutlich: nach Einschalten des Reglers gewinnt das Signal beim

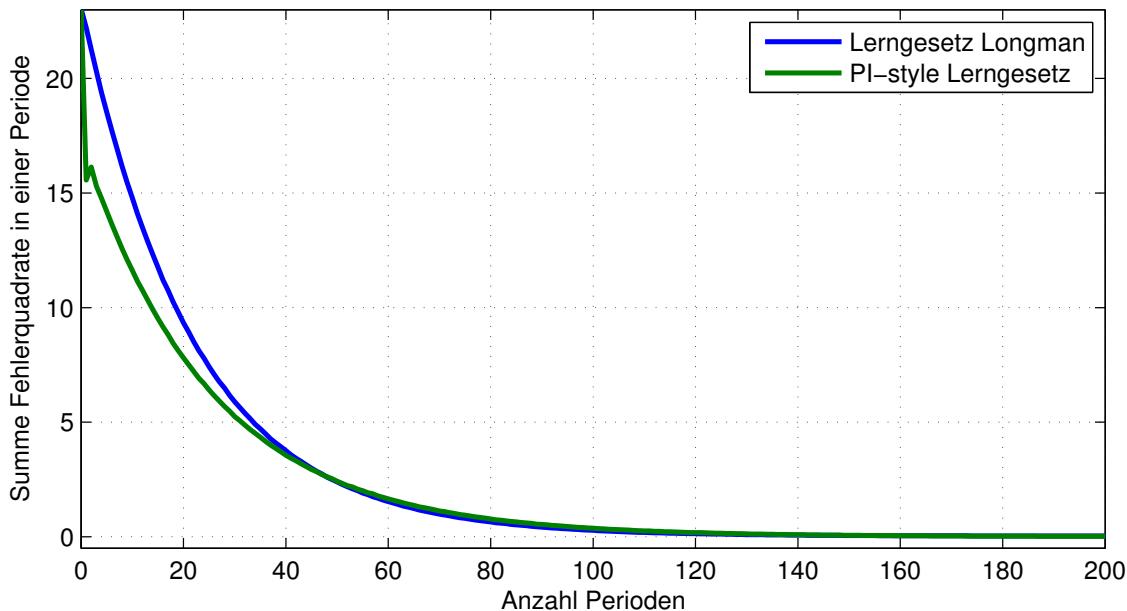


Abbildung 6.3: Vergleich des Lerngesetzes von LONGMAN nach [29] mit dem PI-Lerngesetz: Summe der quadratischen Fehler je Periode, Experiment an der Schiene mit 184 Hz

PI-Lerngesetz sofort an Amplitude, während sich das Signal mit dem Lerngesetz von LONGMAN erst langsam aufbauen muss in dem Maße, in dem sich die Fehlersummen akkumulieren.

6.3 Bewertung der Regelgüte

Die Versuche wurden zunächst jeweils für eine konstante Frequenz durchgeführt. Die Struktur wurde zunächst durch die Versuchssteuerung mit einem sinusförmigen Signal angeregt. Anschließend wurde die Regelung in den Versuchsaufbau integriert und der Versuch mit der gleichen Frequenz wiederholt. Durch Anpassung der Verstärkung am Leistungsverstärker sowie der Verstärkung am Eingang des Reglers wurde versucht, die Amplitude des ersten Versuchs zu reproduzieren.

Das Signal des Kraftsensors wurden in beiden Fällen durch die Messkarte aufgenommen. Abbildung 6.5 zeigt einen Ausschnitt des Signals aus dem Kraftsensor im eingeschwungenen Zustand bei einem Versuch mit der Glasfaserplatte. Aus dem Versuch mit und ohne Regelung wurde ein Stück Signalverlauf verglichen. In dieser Darstellung lassen sich Abweichungen vom reinen Sinus-Verlauf bei der Versuchssteuerung ohne ILR (rote Kurve) erahnen, eine quantitative Bewertung ist aber nicht möglich.

Um eine detaillierte Aussage über die Qualität der Regelung treffen zu können, wird das gemessene Signal mittels der Fouriertransformation (vgl. zum Beispiel [30, Kap. 6.2]) in seine Frequenzanteile zerlegt. Da es sich bei der Computer gestützten Messung und Auswertung um zeitdiskrete Daten handelt, müssen zusätzliche Effekte aus der Diskretisierung bzw. bei der diskreten Signalverarbeitung beachtet werden. Eine korrekte Darstellung im Frequenzbereich gelingt mit zeitdiskreten Daten nur, wenn die Abtastrate genau ein Vielfaches der gemessenen Frequenz ist. Dies ist im Allgemeinen für eine Messung nicht zu realisieren. Die diskrete Fouriertransformation nimmt implizit eine Periodizität des betrachteten Signalverlaufs an. So können Fehler entstehen, wenn Teile einer Periode des Signals am Anfang und Ende der Messung abgeschnitten werden. Der An- bzw. Ab-

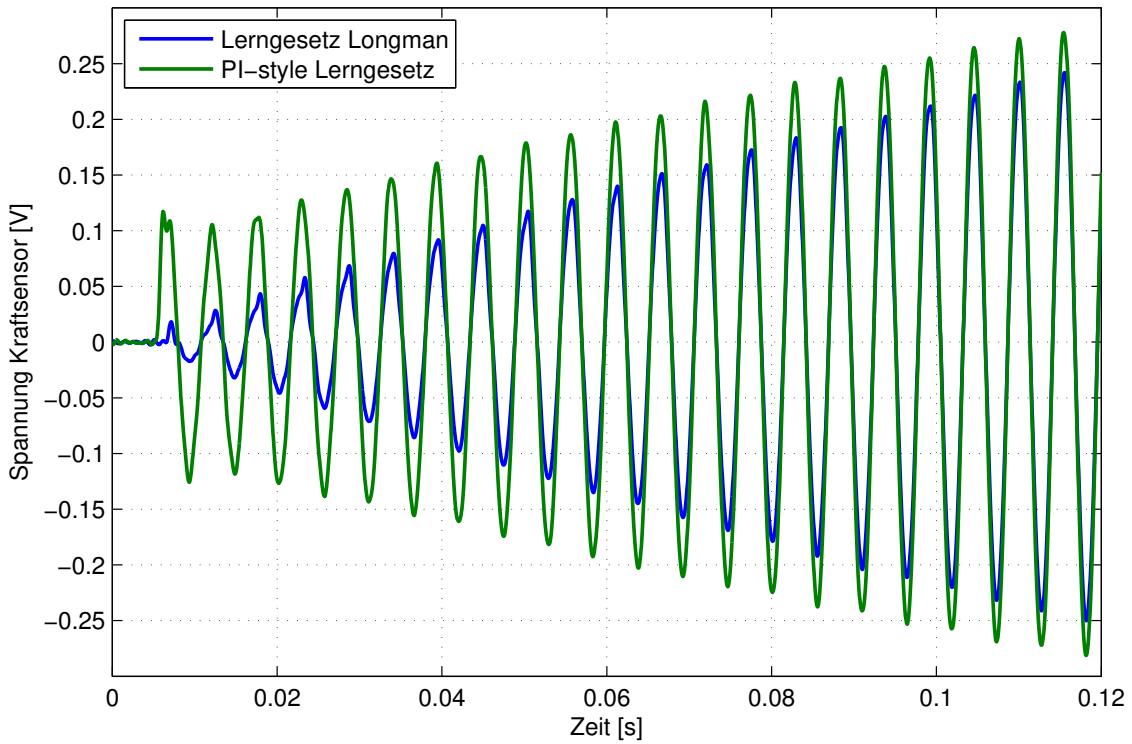


Abbildung 6.4: Vergleich des Lerngesetzes von LONGMAN nach [29] mit dem PI-Lerngesetz: zeitlicher Verlauf

schnitt der Messdaten führt im Frequenzbereich zu einer „Verschmierung“, was eine Interpretation der Ergebnisse erschwert.

Um dem oben genannten Problem zu begegnen werden nachträglich auf die Messdaten Fensterfunktionen angewendet, die die Messdaten an den Rändern des Datensatzes sukzessive ein- bzw. ausblenden und so negative Effekte vermindern. Die unterschiedlichen Fensterfunktionen haben jedoch alle Vor- und Nachteile, sodass je nach Fragestellung eine geeignete Fensterfunktion ausgewählt werden muss und die Ergebnisse auch nur im Zusammenhang dieser Fensterfunktion interpretiert werden können.

Für die Bewertung der Regelgüte in dieser Arbeit ist die Amplitude der Signalanteile maßgeblich. Die genaue Frequenz der Signalanteile ist hier nicht so relevant, da hauptsächlich untersucht werden soll, wie stark Fehler unterdrückt werden können und nicht, welche genauen Frequenzanteile diese Fehler haben. Aus dieser Überlegung heraus wurde als Fensterfunktion ein Flat-Top-Fenster SFT5M [20, S.38 ff.] ausgewählt, das eine sehr gute Amplitudengenauigkeit aufweist und dafür Abstriche bei der Frequenzgenauigkeit macht. Das verwendete Fenster ist in Abbildung 6.6 dargestellt.

Neben der graphischen Darstellung im Frequenzbereich wurde außerdem versucht, die Regelgüte in eine einzige Bewertungskennzahl zu komprimieren. Dafür wurde das Konzept der harmonischen Verzerrung (englisch *total harmonic distortion*) verwendet [50, 8]. Die harmonische Verzerrung wird definiert als

$$\text{THD} = \sqrt{\frac{\sum P_{\text{stör}}}{P_{\text{signal}}}} = \sqrt{\frac{\sum U_{\text{stör}}^2}{U_{\text{signal}}^2}} . \quad (6.1)$$

Es werden also die Leistungen $P_{\text{stör}}$ ungewollter Signalanteile mit der Leistung des gewünschten

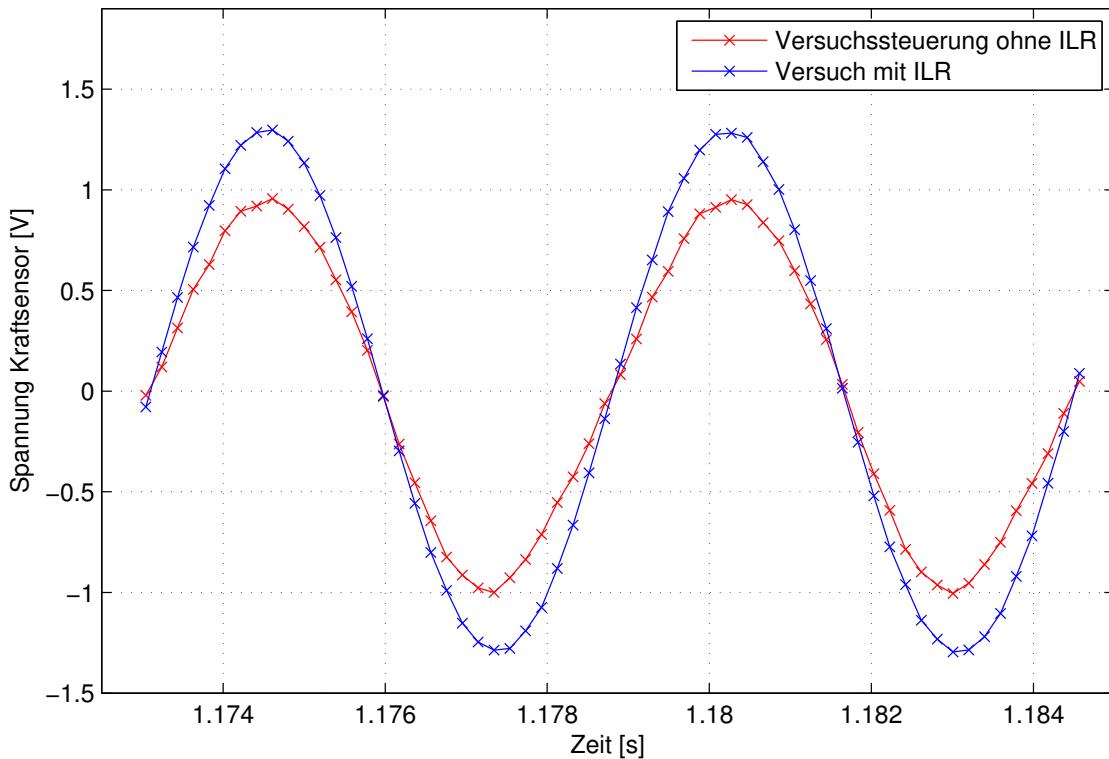


Abbildung 6.5: Vergleich der Messdaten im Zeitbereich aus den Messungen mit der Glasfaserplatte mit und ohne ILR

Versuch bei 175 Hz, eingeschwungener Zustand, die einzelnen Sample sind mit \times gekennzeichnet

Signals P_{signal} ins Verhältnis gesetzt. In dieser Arbeit sind sowohl die Stellgröße als auch die Regelgröße eine Spannung, weshalb anstelle der Leistung die quadratische Spannung verwendet wird gemäß der näherungsweisen Relation $P \sim U^2$, die eigentlich nur für rein ohmsche Lasten gilt. Für gewöhnlich werden nur bestimmte Frequenzen (z.B. von höher-harmonischen Störungen) für die Berechnung der harmonischen Verzerrung heran gezogen. Dieser Definition der harmonischen Verzerrung folgend wurden nur die Amplituden der 5 (bzw. 4 für den Versuch mit der Glasfaserplatte bei 454 Hz) höher-harmonischen Schwingungen als Störfrequenzen aufgefasst und so die *total harmonic distortion* in Relation zur Amplitude der gewünschten Frequenz berechnet.

Im Rahmen dieser Arbeit stellen aber nicht nur höher-harmonische Anteile des Signalverlaufs ein Problem für die spätere Anwendung des Verfahrens dar, sondern eigentlich alle anderen Frequenzen, z.B. auch Rauschanteile oder andere Störungen, die die Struktur, die Aktuatorik oder der Regler in das System einbringen könnten. Aus diesem Grund wurde das Konzept der *total harmonic distortion* leicht abgewandelt und auf alle anderen Frequenzen außer der gewünschten Frequenz ausgedehnt. Zur numerischen Realisierung wurde wegen der „Verschmierung“ der Frequenzanteile durch die Fensterfunktion ein schmales Band von etwa 5 Hz verwendet, in dem alle Leistungen als „richtig“ betrachtet werden. Alle anderen Frequenzanteile werden als Störung betrachtet und gehen in die Summe im Zähler ein.

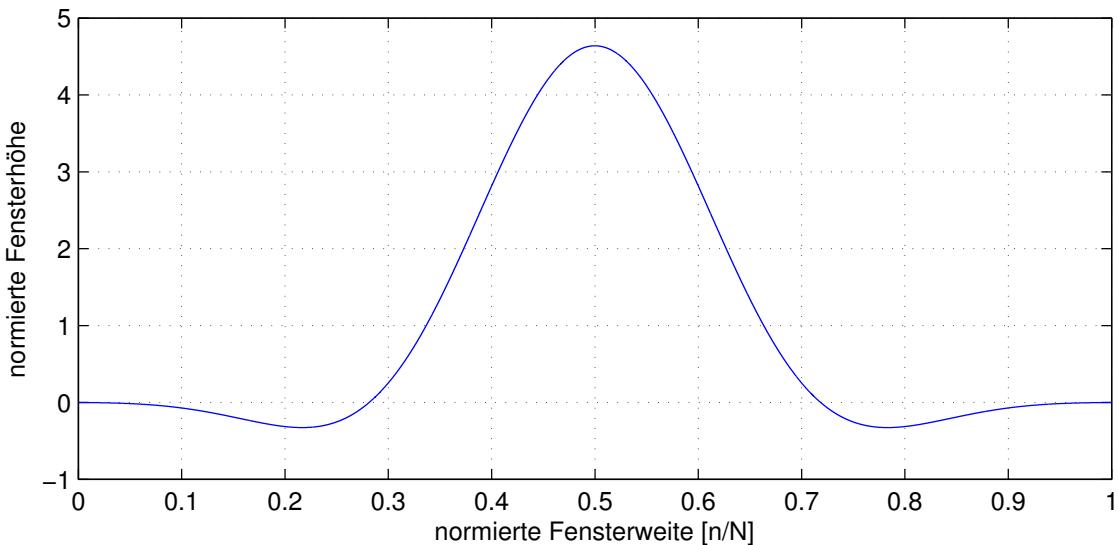


Abbildung 6.6: normierte Darstellung des verwendeten Flat–Top–Fensters SFT5M

6.4 Ergebnisse Messungen mit Glasfaserplatte

Für die Auswahl geeigneter Frequenzen für die Versuche wurde auf Ergebnisse von früheren Sweep–Experimenten zurückgegriffen. Abbildung 6.7 zeigt die Amplituden des Kraftsignals im Frequenzbereich aus einem solchen Sweep–Experiment und die Frequenzen, für die Versuche mit der ILR durchgeführt wurden. An einigen Stellen bricht die Kraft ein, dies kann insbesondere in der Nähe der Resonanzfrequenzen der Struktur sein. In der Nähe dieser Einbrüche gibt es besonders starke höherharmonische Anteile im Kraftsignal, deshalb wurden derartige Frequenzen für die Versuche verwendet.

Bei den Versuchen mit der Glasfaserplatte wurden die Parameter in Tabelle 6.1 verwendet. Problematisch bei den Versuchen waren die sehr kleinen Amplituden des Kraftsignals, wodurch eine sehr hohe Verstärkung am Eingang des Reglers verwendet werden musste.

Tabelle 6.1: Reglerparameter für die Versuche an der Schiene

Parameter	Wert	Einheit
Samplerate	20000	Hz
Ki	0,07	-
Kp	0,15	-
PhaseLead	10	Zeitschritte
Nsmooth	20	Zeitschritte

Das Kraftsignal der Versuchssteuerung weist aufgrund der hochwertigen Messtechnik ein allgemein niedrigeres Niveau der Rauschanteile auf. Dies wird auf den höheren Dynamikbereich zurück geführt. Die DAC–Karte NI9263 hat eine Auflösung von 16 bit, dies entspricht einem Dynamikbereich von ca. 96 dB. Dahingegen haben die integrierten DACs des Arduino Due nur eine Auflösung von 12 bit, was einem Dynamikbereich von ca. 72 dB entspricht. Diese Werte des theoretisch technisch möglichen Rausch–Niveaus stimmen sehr gut mit den Ergebnissen der Messungen überein (siehe Abbildung 6.8).

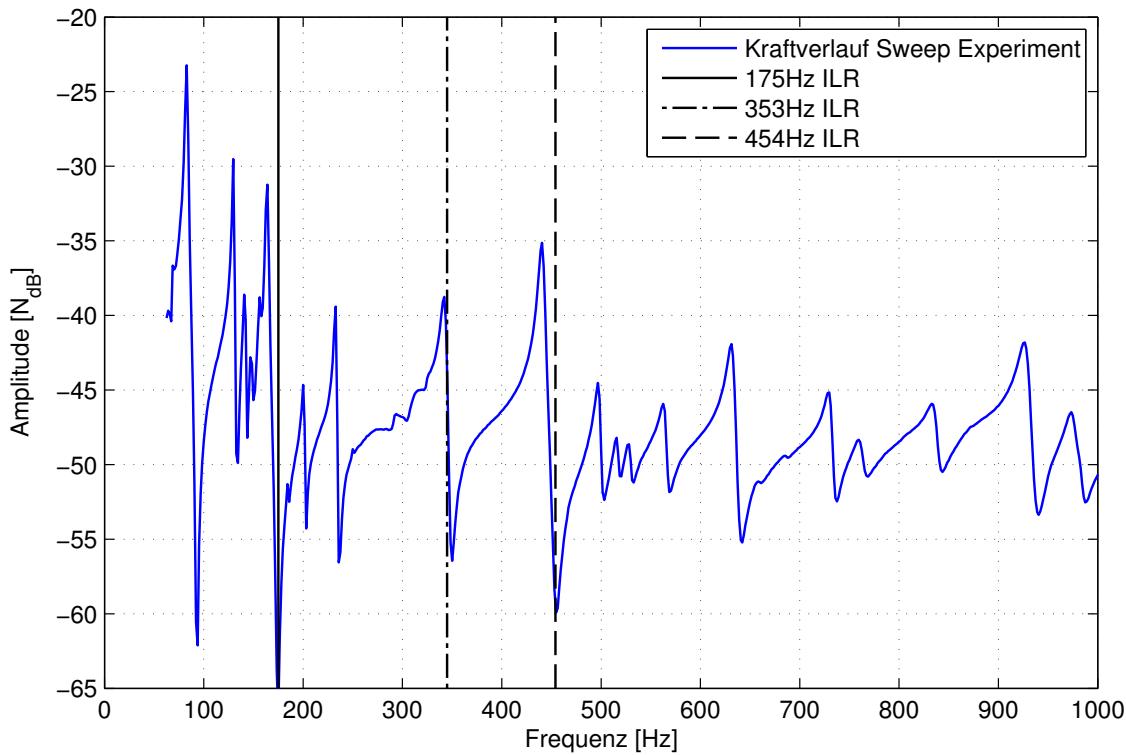


Abbildung 6.7: Amplituden der Erregerkraft aus einem Sweep-Experiment an der Glasfaserplatte, Kennzeichnung der Frequenzen, zu denen Versuche mit der ILR durchgeführt wurden

Die Abbildungen 6.8 bis 6.10 zeigen den Vergleich zwischen den Versuchen mit der herkömmlichen Versuchssteuerung und der Regelung mittels ILR. Zunächst fällt auf, dass das Kraftsignal der Versuchssteuerung starke höher-harmonische Anteile enthält, nämlich mit den Vielfachen der Anregungsfrequenz. Diese höher-harmonischen Anteile können durch die Regelung vollständig eliminiert werden.

Die berechneten Werte für die harmonische Verzerrung unter Berücksichtigung aller anderen Frequenzanteile sind in Tabelle 6.2 aufgeführt.

Tabelle 6.2: Berechnete Werte und Verhältnisse der harmonischen Verzerrung unter Berücksichtigung aller anderen Frequenzanteile, Messungen mit der Glasfaserplatte

Versuch	THD	THD _{ILR}	THD/THD _{ILR}
175 Hz	0,0605	0,0124	4,88
345 Hz	0,2094	0,0043	49,19
454 Hz	0,1296	0,0054	23,82

Die Ergebnisse der Analyse zur harmonischen Verzerrung unter ausschließlicher Berücksichtigung der höherharmonischen Störungen sind in Tabelle 6.3 dargestellt.

Die Auswertung der harmonischen Verzerrung zeigt eine starke Verbesserung der Signalqualität durch Einsatz der Regelung sowohl bei ausschließlicher Berücksichtigung der höher-harmonischen

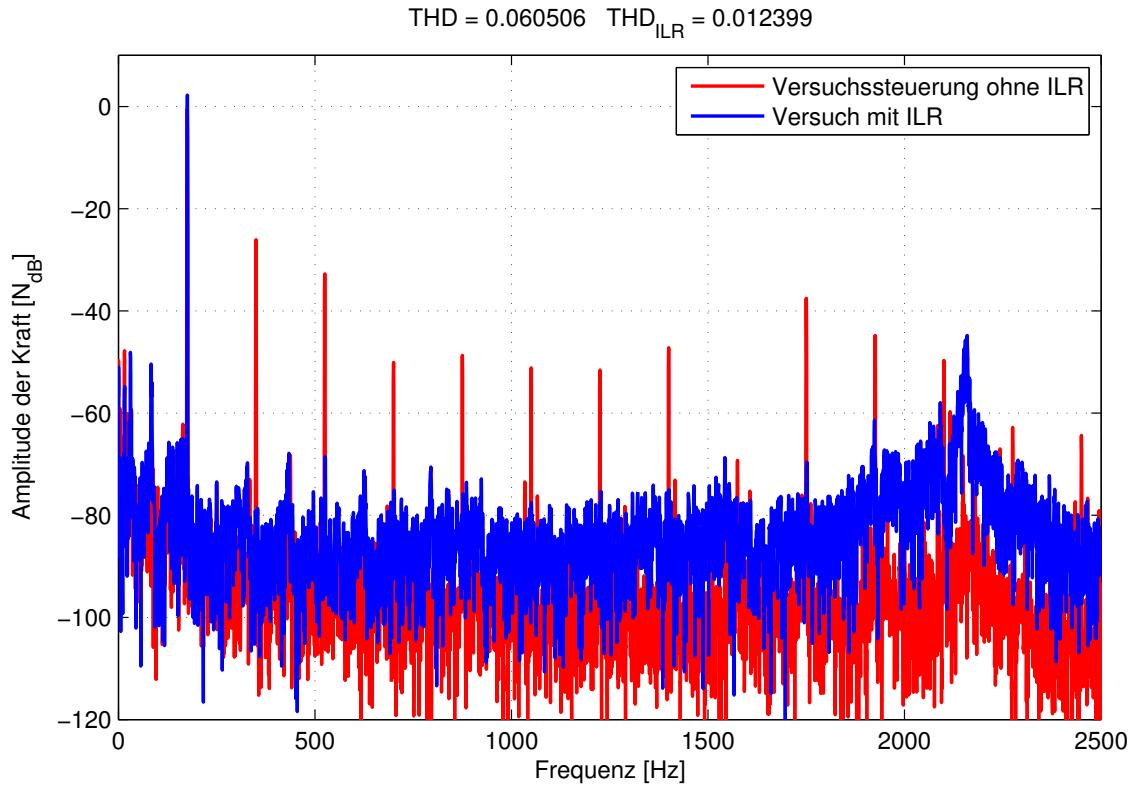


Abbildung 6.8: Vergleich Messung mit und ohne ILR im Frequenzbereich bei 175 Hz

Signalanteile als auch bei Berücksichtigung aller anderen Signalanteile. Die Regelung kann eine Verzerrung von $\text{THD} < 0,004$ erreichen, wenn nur höher-harmonische Anteile betrachtet werden und von $\text{THD} < 0,015$, wenn alle anderen Signalanteile in die Berechnung einfließen. Die Verbesserung der Signalqualität hängt dann maßgeblich davon ab, wie stark das Signal im ungeregelten Zustand gestört war.

Mit der Glasfaserplatte wurden auch Sweep-Experimente versucht, allerdings war es nicht möglich, den Regler so einzustellen, dass eine Resonanzfrequenz im Sweep durchlaufen werden konnte. Es wird vermutet, dass der extrem hohe Dynamikbereich von bis zu zwei Größenordnungen aufgrund der schwachen Dämpfung der Glasfaserplatte das Hauptproblem darstellt. Nicht nur für den Regler ist der große Dynamikbereich schwierig, auch die Aktuatorik muss in der Lage sein, derart große Unterschiede in den Anregungen abzubilden.

Tabelle 6.3: Amplituden der Erreger-Frequenz und 5 bzw. 4 höher-harmonischer Störungen zur Berechnung der *total harmonic distortion*
aus den Versuchen mit der Glasfaserplatte
Angaben in Newton

Frequenz	175 Hz	350 Hz	525 Hz	700 Hz	817 Hz	1050 Hz	THD	THD/THD _{ILR}
Versuch ① bei 175 Hz	normal	0,949	0,050	0,023	0,0031	0,0037	0,0027	0,0583
	ILR	1,293	0,0001	0,0004	0,00018	0,00016	0,00013	0,00383
Versuch ② bei 345 Hz	normal	2,096	0,113	0,098	0,254	0,1574	0,284	0,2092
	ILR	2,759	0,00017	0,0001	0,00034	0,00036	0,0011	0,00044
Versuch ③ bei 454 Hz	normal	1,452	0,151	0,022	0,109	0,0072	0,129	472,2
	ILR	2,790	0,00032	0,00054	0,00067	0,00042	0,00242	53,38

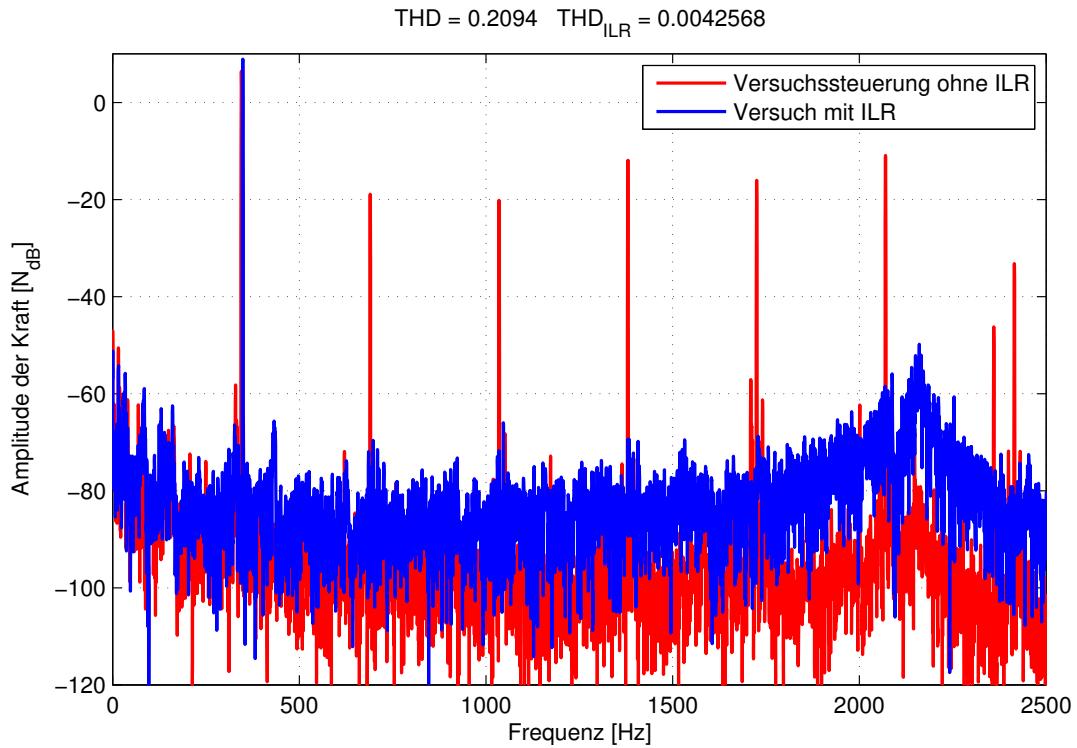


Abbildung 6.9: Vergleich Messung mit und ohne ILR im Frequenzbereich bei 345 Hz

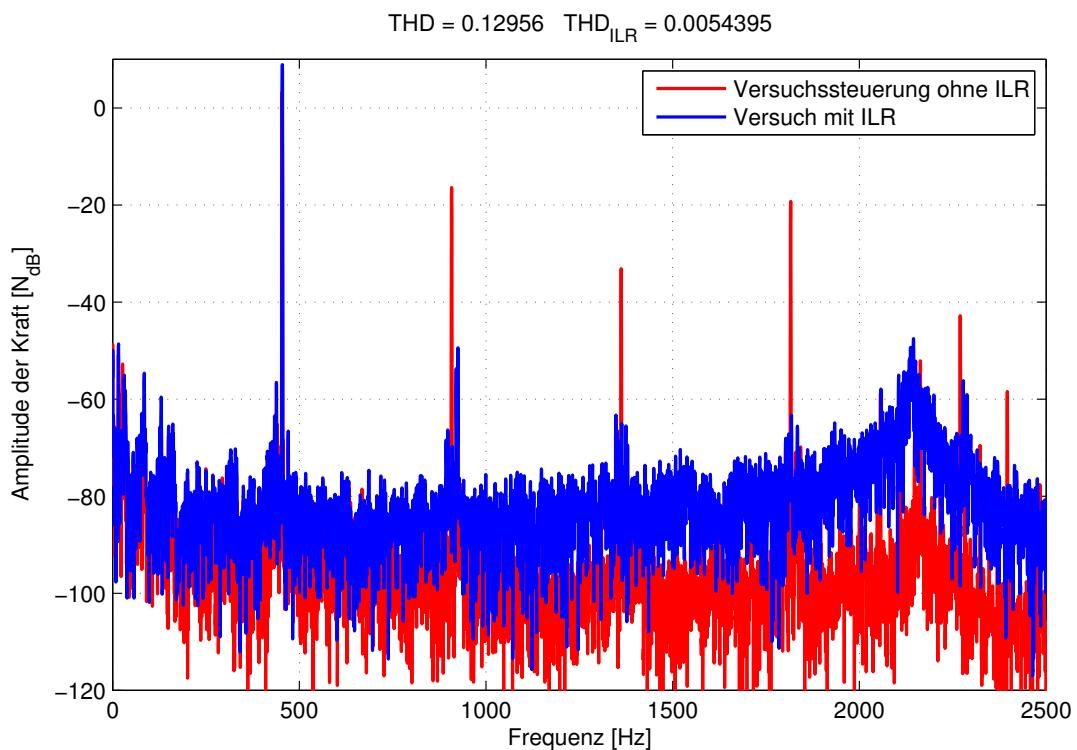


Abbildung 6.10: Vergleich Messung mit und ohne ILR im Frequenzbereich bei 454 Hz

6.5 Ergebnisse Messungen Schiene

6.5.1 Experimente mit konstanter Frequenz

Analog zu den Versuchen mit der Glasfaserplatte wurden zunächst Frequenzen identifiziert, bei denen starke höherharmonische Störungen im Kraftsignal auftreten. Zur Orientierung ist in Abbildung 6.11 die gemessene Übertragungsfunktion aufgetragen und die Frequenzen eingezeichnet, bei denen die Versuche durchgeführt wurden. Bei den Versuchen mit der Schiene wurden die Parameter in Tabelle 6.4 verwendet.

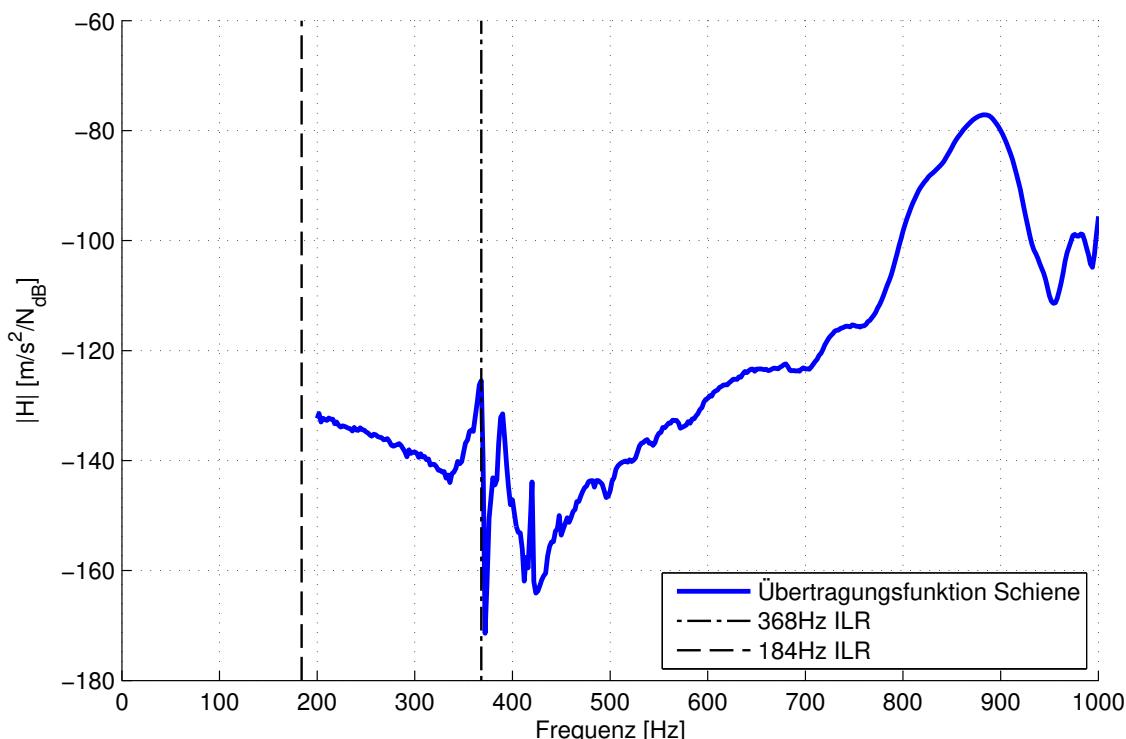


Abbildung 6.11: Übertragungsfunktion der Schiene von der Erregerkraft zur Beschleunigung am Erregerpunkt aus einem Sweep-Experiment, Kennzeichnungen der Frequenzen, zu denen Versuche mit der ILR durchgeführt wurden

Tabelle 6.4: Reglerparameter für die Versuche an der Schiene

Parameter	Wert 184 Hz	Wert 368 Hz	Einheit
Anzahl Werte Sinus	108	81	-
Samplerate	19867,55	29808,37	Hz
ermittelte Phasenverschiebung	0,6	0,2	ms
K_i	0,01	0,03	-
K_p	0,2	0,2	-
PhaseLead	5	6	Zeitschritte
N_{smooth}	8	10	Zeitschritte

Die Ergebnisse der Messungen mit der Versuchssteuerung und mit der ILR sind in den Ab-

bildungen 6.12 und 6.13 dargestellt. Wie auch schon bei der Glasfaserplatte ist zu sehen, dass höherharmonische Störungen innerhalb des Dynamikbereichs vollständig eliminiert werden.

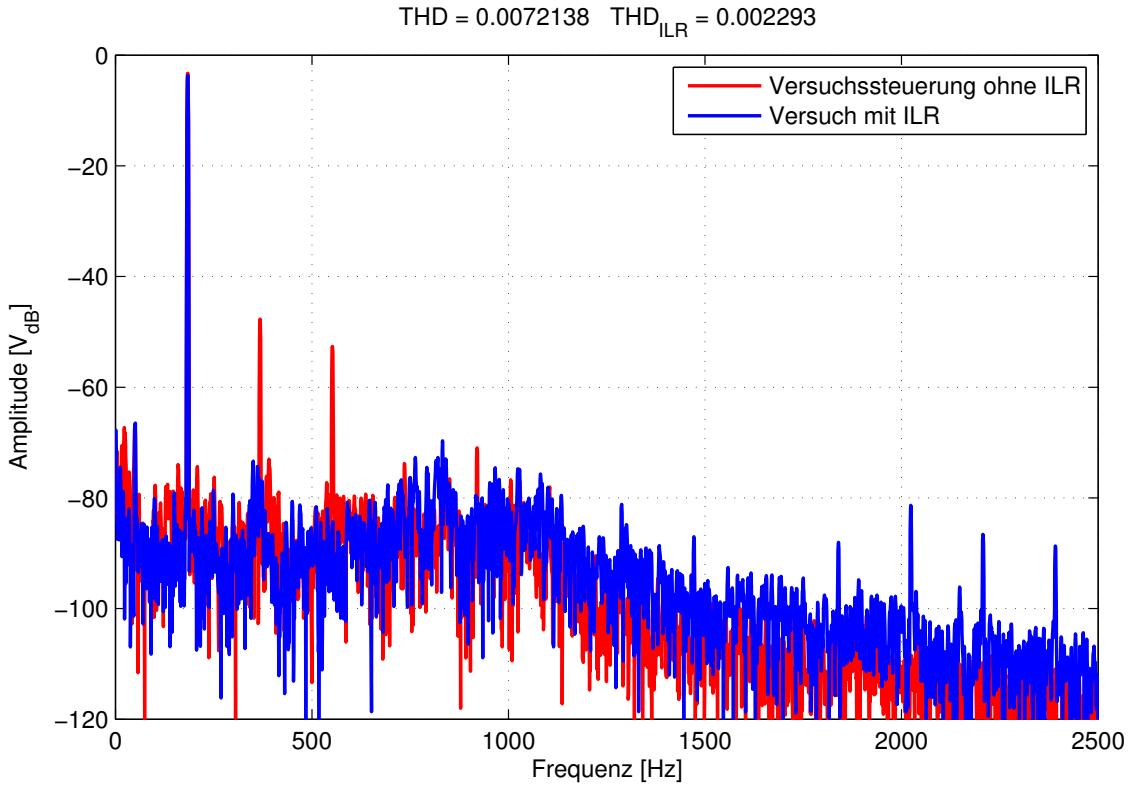


Abbildung 6.12: Vergleich Messung mit und ohne ILR im Frequenzbereich bei 184 Hz

Zum Vergleich ist in Abbildung 6.14 die Stellgröße (Signal an den Leistungsverstärker des Aktuators) im Frequenzbereich dargestellt. Dort ist zu erkennen, dass das Signal in der Versuchssteuerung (fast) nur einen Sinus enthält. Wird die Regelung verwendet, enthält das Stellgrößensignal zahlreiche höherharmonische Anteile — eben diese Anteile, mit denen die Störungen kompensiert werden.

Die harmonische Verzerrung der Experimente an der Schiene wurde analog zu den Experimenten an der Glasfaserplatte berechnet. Die Ergebnisse für die harmonische Verzerrung unter Berücksichtigung aller anderen gemessenen Frequenzen sind in Tabelle 6.5 zusammen gefasst. Die Ergebnisse für die harmonische Verzerrung unter Berücksichtigung nur der ersten 5 höher-harmonischen Frequenzanteile sind in Tabelle 6.6 dargestellt.

Tabelle 6.5: Berechnete Werte und Verhältnisse der harmonischen Verzerrung unter Berücksichtigung aller anderen Frequenzanteile, Messungen mit der Schiene

Versuch	THD	THD _{ILR}	THD/THD _{ILR}
184 Hz	0,00721	0,00229	3,15
368 Hz	0,4404	0,00477	92,3

Die Ergebnisse der harmonischen Verzerrung zeigen, dass die Regelung in der Lage ist, die harmonische Verzerrung auf ein $\text{THD} < 4 \cdot 10^{-4}$ zu reduzieren, wenn nur die höherharmonischen Anteile

Tabelle 6.6: Amplituden der Erreger-Frequenz und 5 höher-harmonischer Störungen zur Berechnung der *total harmonic distortion* aus den Versuchen mit der Schiene — Angaben in Volt

Frequenz	184 Hz	368 Hz	552 Hz	736 Hz	920 Hz	1104 Hz	THD	THD/THDLR
Versuch ① bei 184 Hz	normal ILR	0,685 0,652	0,00412 0,00012	0,00233 0,00006	0,00020 0,00005	0,00028 0,00010	0,00013 0,00013	0,00693 0,00033
<hr/>								
Frequenz	368 Hz	736 Hz	1104 Hz	1472 Hz	1840 Hz	2208 Hz	THD	THD/THDLR
Versuch ② bei 368 Hz	normal ILR	0,205 0,443	0,0886 0,00010	0,0176 0,00006	0,00100 0,00010	0,00058 0,00003	0,000008 0,00005	0,4407 0,00037
<hr/>								
							20,7	1191

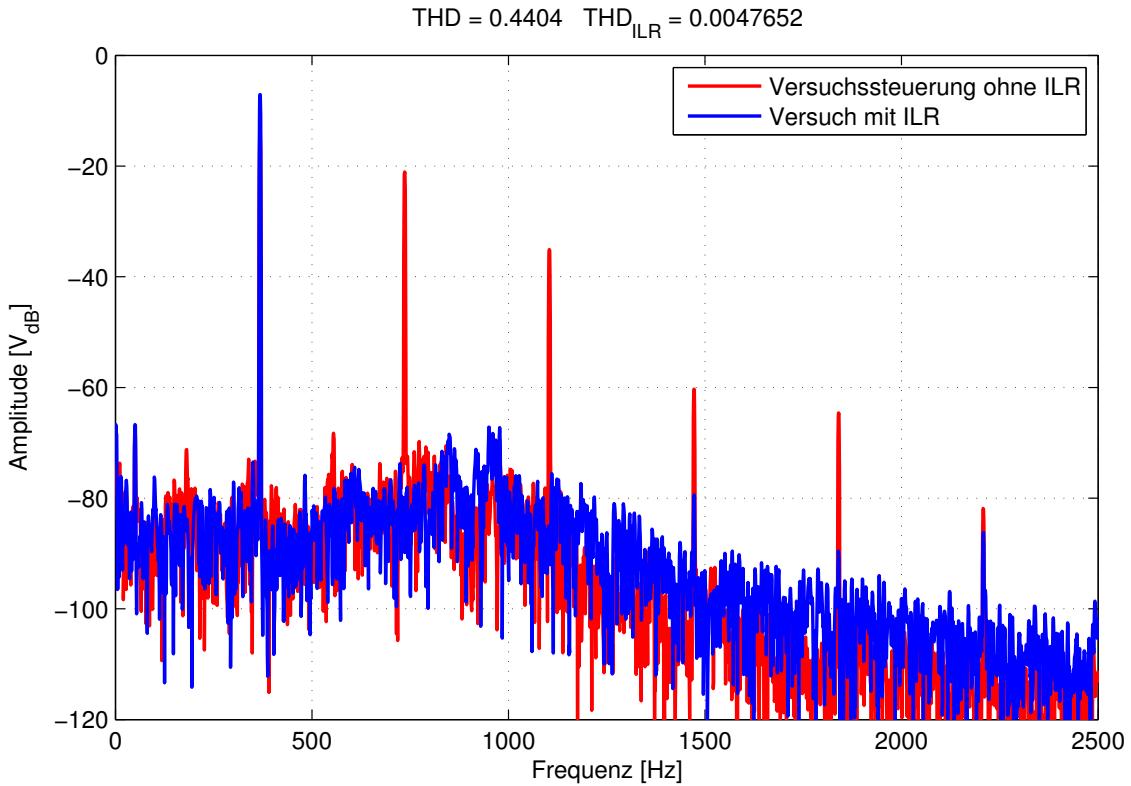


Abbildung 6.13: Vergleich Messung mit und ohne ILR im Frequenzbereich bei 368 Hz

berücksichtigt werden und auf $\text{THD} < 5 \cdot 10^{-3}$ wenn alle anderen Signalanteile berücksichtigt werden. Eine deutliche Verbesserung bei dem Experiment mit 184 Hz ist sichtbar, die in einer 3,15-fachen Verbesserung resultiert, wenn alle Frequenzen und in einer 20,7-fachen Verbesserung, wenn nur die ersten 5 höherharmonischen Frequenzen berücksichtigt werden. Da Störungen bei dem Experiment mit 368 Hz wesentlich stärker ausgeprägt waren, ist hier auch eine stärkere Verbesserung möglich. Unter Berücksichtigung aller Frequenzanteile ist eine 92,3-fache Verbesserung gelungen und unter Berücksichtigung der ersten 5 höherharmonischen Frequenzen eine 1191-fache Verbesserung.

6.5.2 Sweep–Experiment

An der Schiene konnte auch ein Sweep–Experiment im Bereich von ca. 360 Hz bis 382 Hz durchgeführt werden. Dies ist genau der Bereich um die Spitze in der Übertragungsfunktion, in dessen Nähe eine Resonanzfrequenz liegt. Zur Veranschaulichung wurde der entsprechende Ausschnitt aus Abbildung 6.11 vergrößert und ist in Abbildung 6.15 dargestellt. Die Steuerung des Sweep–Experiments erfolgte durch ein matlab–Skript, das über die serielle Konsole in regelmäßigen kurzen Abständen eine neue Samplerate für die ILR setzte, sodass die Frequenz sukzessive geändert wurde (siehe auch Tabelle 5.1 für die Befehle an den Regler). Die Samplerate wurde sukzessive von 29 kHz bis auf 31 kHz erhöht, wodurch ein *stepped frequency sweep* realisiert wurde. Die einzelnen Schritte der Frequenzänderung waren unter Ausnutzung der DueTimer Library [49] sehr klein, sodass das Frequenzband von 20 Hz in ca. 100 Schritten durchfahren wurde.

Das Signal zum Aktuator und das Signal vom Kraftsensor für das Sweep–Experiment sind im Zeitbereich in Abbildung 6.16 dargestellt. Es ist deutlich zu erkennen, dass die Amplitude des Kraftsignals trotz durchlaufen der Resonanzfrequenz näherungsweise konstant bleibt. Dafür wird

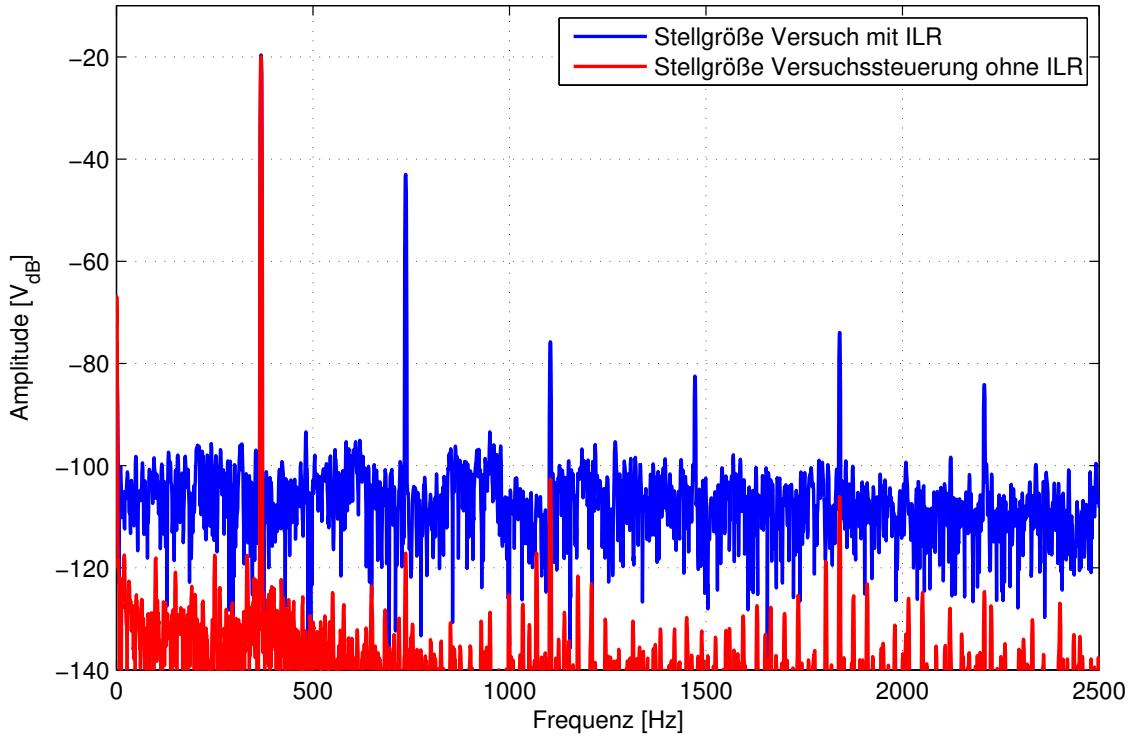


Abbildung 6.14: Vergleich der Stellgröße bei Messung mit und ohne ILR im Frequenzbereich bei 368 Hz

die Amplitude der Stellgröße im Verlauf des Sweeps durch die Regelung wesentlich verändert.

Zur Auswertung des Sweep–Experiments im Frequenzbereich konnte kein Flat–Top–Fenster verwendet werden, da dieses erstens eine starke Wichtung auf einen relativ kleinen Bereich innerhalb der Messung vornimmt und zweitens eine schlechte Frequenzauflösung hat. Stattdessen wurde für das Sweep–Experiment ein Tukey–Fenster mit einem Parameter $\alpha = 0,2$ verwendet. Das Tukey–Fenster ist in der Mitte flach und fällt an den Rändern ab. Mit dem Parameter α kann die Breite der fallenden Seiten relativ zur Breite des konstanten Bereichs eingestellt werden. Die verwendete und normierte Tukey–Fenster–Funktion ist in Abbildung 6.17 dargestellt. Die hier durchgeführte Auswertung des Sweep–Experiments hat dabei einen geringen messtechnischen Anspruch, vielmehr soll die generelle Wirkungsweise des vorstellten Regelungsansatzes demonstriert werden. Korrekte und umfangreiche Auswertung eines Sweep–Experiments im Frequenzbereich und detaillierte Bewertung der Regelgüte könnte zum Beispiel durch eine Kurzzeit–Spektralanalyse (vgl. z.B. [57, S.77 ff.]) erfolgen, geeignete Fensterfunktionen, Überlappung der Fenster und Vorgehen zur Mittelung der einzelnen Fenster werden u.a. in [20, Kap.10] vorgeschlagen.

In der Abbildung 6.18 ist das Sweep–Experiment im Frequenzbereich dargestellt. Neben dem starken Ausschlag der im Sweep enthaltenen Frequenzen sind andere Frequenzanteile kaum enthalten und haben eine Amplitude von -70 dB oder kleiner. Abbildung 6.19 zeigt den Ausschnitt zwischen 300 Hz und 440 Hz. Es ist zu erkennen, dass die Amplituden innerhalb des Sweeps ungefähr gleich stark sind. Das „Verschmieren“ an den Rändern des Sweeps wird auf die Tukey–Fensterfunktion zurück geführt. Die leichte Welligkeit in den Amplituden innerhalb des Sweeps könnte ebenfalls durch die Eigenschaften der Fouriertransformation in Kombination mit dem Sweep–Signal im Zeitbereich hervorgerufen werden. Zur genaueren Untersuchung müssen hier tiefer gehende Analysen durchgeführt werden.

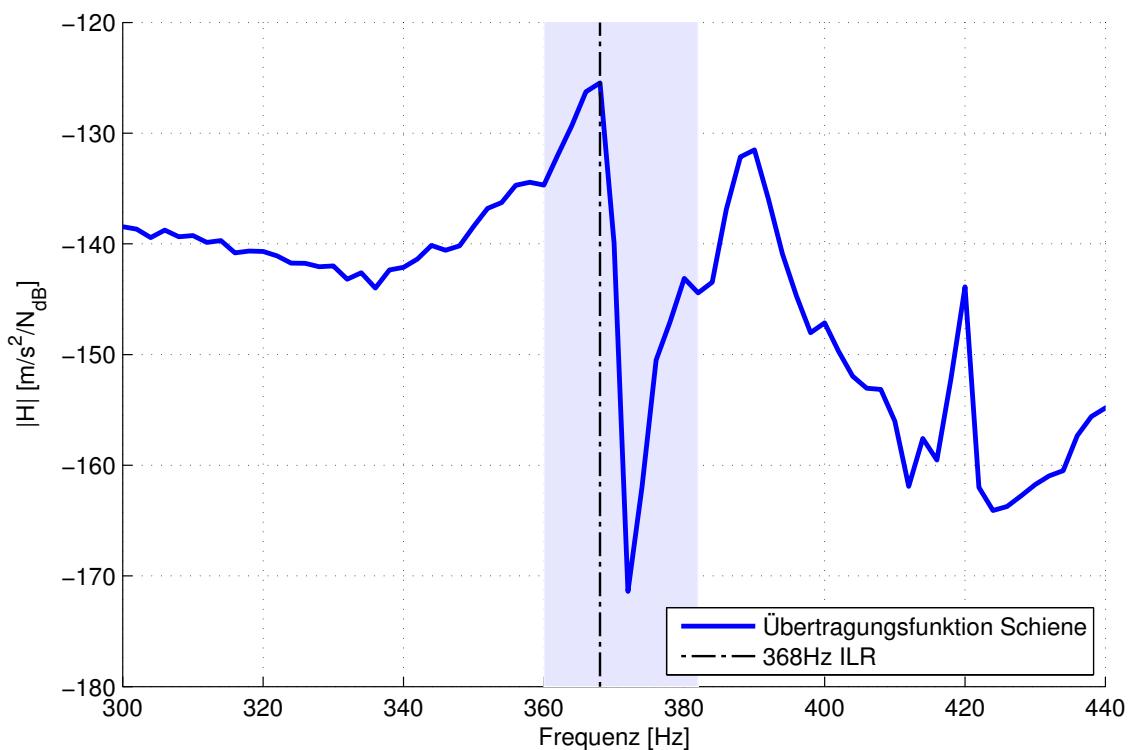


Abbildung 6.15: Ausschnitt aus der Übertragungsfunktion in Abbildung 6.11, der Bereich des Sweep-Experiments ist blau hinterlegt

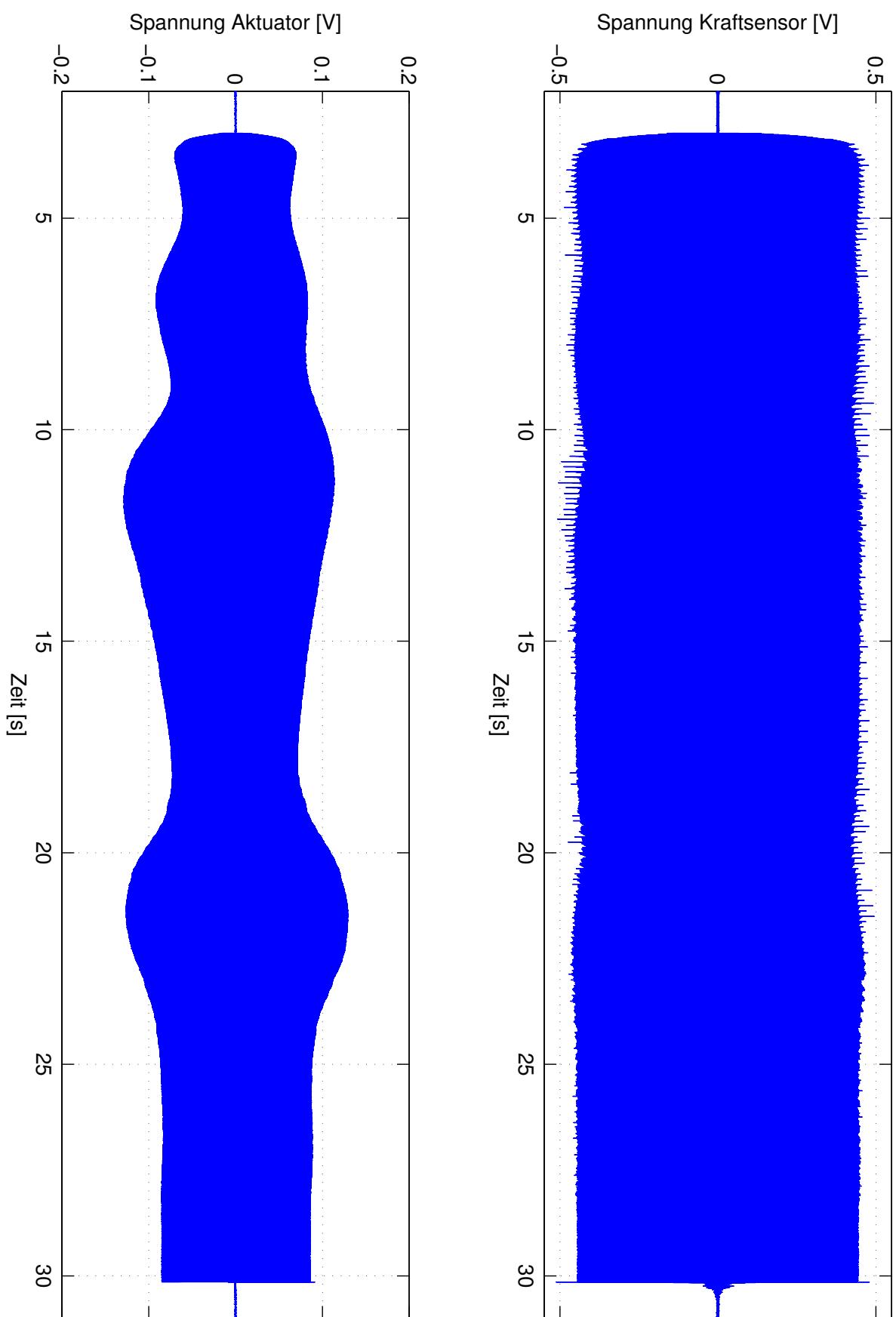


Abbildung 6.16: Sweep-Experiment mit ILR: Signal zum Aktuator und vom Kraftsensor

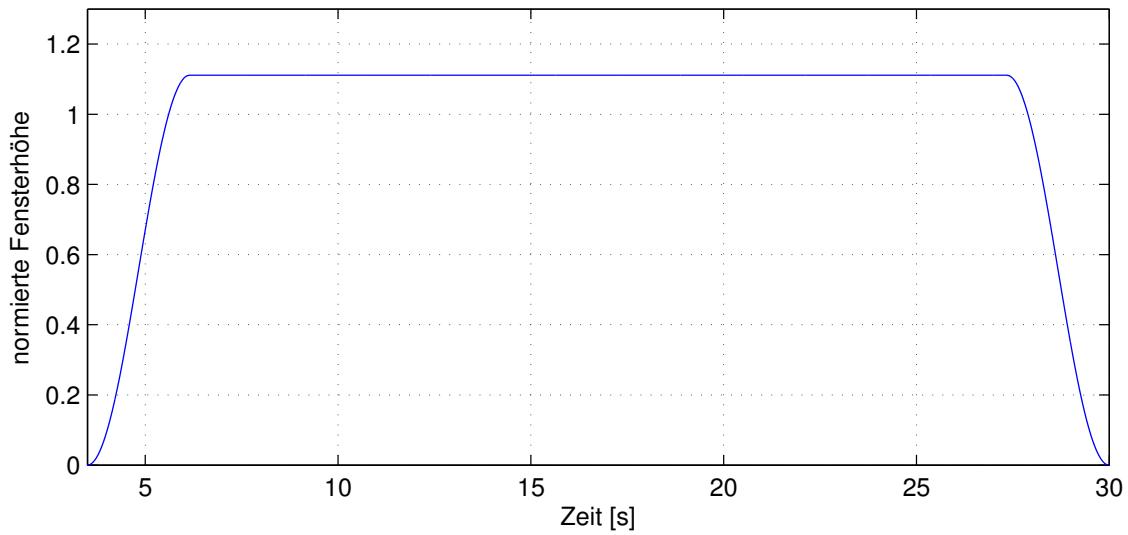


Abbildung 6.17: Tukey–Fensterfunktion mit $\alpha = 0,2$, die zur Auswertung des Sweep–Experiments verwendet wurde

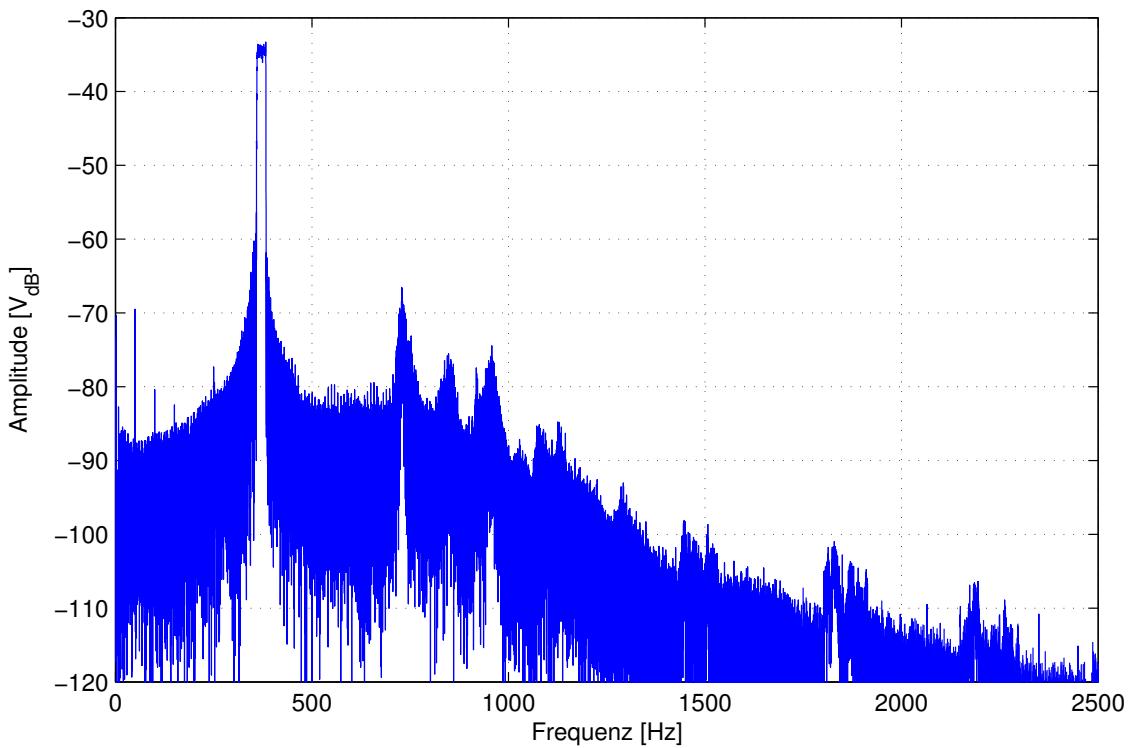


Abbildung 6.18: Kraftsignal des Sweep–Experiments im Frequenzbereich

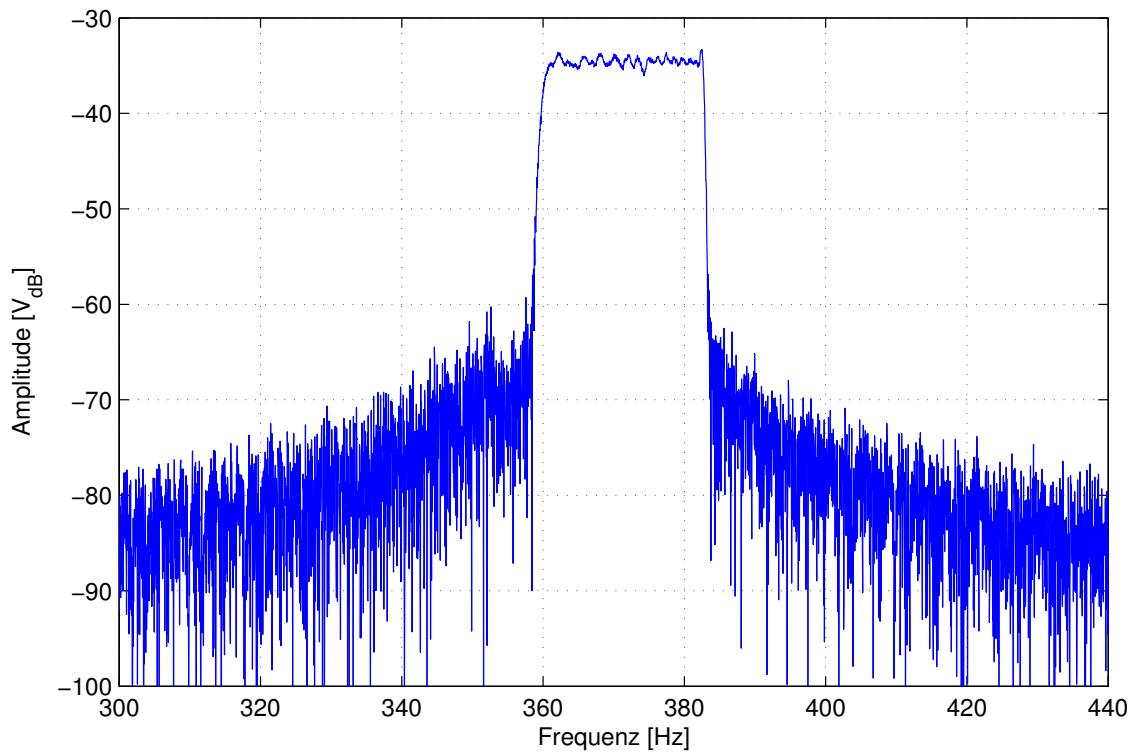


Abbildung 6.19: Kraftsignal des Sweep–Experiments im Frequenzbereich, Fokus auf Frequenzen nahe des Sweeps

7 Fazit und Ausblick

7.1 Zusammenfassung

Es wurde prototypisch ein Aufbau entwickelt, der mittels eines Arduino Due und dem integrierten ADC bzw. DAC sowie eines umgebenden Schaltkreises aus günstigen, handelsüblichen Bauteilen eine repetitive Regelung für die rein-harmonische Kraftanregung von Struktur-Bauteilen realisiert.

Durch Integration der Regelung in den Versuchsaufbau und Untersuchung von zwei unterschiedlichen Strukturbauarten konnten störende höher-harmonische Anteile der Anregung vollständig eliminiert werden. Damit wurde das primäre Ziel der Arbeit erreicht. Anhand der Bewertungskennzahl *total harmonic distortion* (THD) wurde die Güte des geregelten Signals bei Versuchen mit konstanter Frequenz im eingeschwungenen Zustand mit den Ergebnissen aus der Versuchssteuerung verglichen. Die harmonische Verzerrung konnte drastisch reduziert werden auf ein Niveau von $\text{THD} < 5 \cdot 10^{-3}$. Es konnte außerdem ein Sweep-Experiment durch eine Resonanzfrequenz mit der Regelung durchgeführt werden. Die Ergebnisse zeigen, dass die Amplitude der Anregung konstant gehalten wird und des weiteren andere störende Frequenzen weitestgehend unterdrückt werden.

Die Materialkosten sind 40 € für den Arduino Due und wenige Euro für die Bauteile des umgebenden Schaltkreises.

Die integrierten ADC bzw. DAC zeigen eine gute Performance, sowohl in der Geschwindigkeit und Echtzeitverhalten als auch in der Genauigkeit. Die Ergebnisse aus den Versuchen zeigen, dass der theoretische Dynamikbereich von 74 dB bei einer guten Einstellung auch erreicht werden kann und für Versuche mit konstanter Frequenz ausreichend ist. Insbesondere beim Durchlaufen von Resonanzfrequenzen bei Sweep-Experimenten mit nur schwach gedämpften Strukturen wäre ein höherer Dynamikbereich allerdings hilfreich, da alleine der Dynamikbereich der Übertragungsfunktion schon bis zu 70 dB betragen kann.

Die Arduino Firmware, die Scripte und der Schaltplan wurden auf github unter <https://github.com/piflixe/ILR> unter der GNU GENERAL PUBLIC LICENSE veröffentlicht damit sie für die Verwendung und Weiterentwicklung zur Verfügung stehen.

7.2 Verbesserungsideen

Verbesserung Protokoll Serielle Konsole

- Bestätigung für empfangene Nachrichten vom Arduino senden, damit erkannt werden kann, ob Nachrichten richtig übertragen wurden
- Menü verbessern, um Parameter durch einen einzigen Befehl Parameter setzen zu können in der Form `parameter = value`

Verbesserungen der Implementierung des Reglers Eine ganze Reihe von Verbesserungsideen sind bei der Durchführung der Versuche bereits aufgekommen. Leider fehlte für die Umsetzung bislang die Zeit. Für diese Arbeit wurden nur die wesentlich Kernfunktionen der Regelung bearbeitet.

- Implementierung der Phasenverschiebung **PhaseLead** ändern, sodass die reale Phasenverschiebung angegeben werden kann. Bei der jetzigen Implementierung ergibt sie sich implizit aus **PhaseLead** + **Nsmooth**/2.
- Implementierung eines Reglers mit automatischer Anpassung der Phasenverschiebung. Es wird vermutet, dass dies die Fähigkeit des Reglers, Sweep–Experimente zu regeln, deutlich verbessert.
- Überprüfung, ob eingegeben Parameterkombinationen zulässig sind, insbesondere die Kombination aus Phasenverschiebung **PhaseLead** und Fensterbreite **Nsmooth**
- Erzeugung der Funktionswerte auf dem Regler ohne ein zusätzliches Script und erneutem flashen der Firmware. Nach Start des Reglers könnten die Funktionswerte unter Angabe von gewünschte Frequenz und Abtastrate über die serielle Console berechnet werden.
- Speichern der Funktionswerte und Reglerparameter zur Versuchsdokumentation, z.B. als Textdatei auf dem Computer, das später wieder geladen werden kann
- Implementierung, sodass höhere Sampling–Raten möglich werden. DAC–Implementierung im Arduino IDE bleibt hinter den theoretischen Möglichkeiten zurück. Code der Regelung kann u.U. optimiert werden, um Rechenschritte, Speichernutzung usw. zu verringern und Hardware–Unterstützung zu verbessern.
- Vereinfachung des Lerngesetzes, um die Anzahl der Parameter im Regler zu reduzieren und so die Einstellung zu erleichtern bzw. die Abhängigkeit bspw. vom Phasenwinkel der Strecke zu reduzieren.
- dynamische Anpassung der Reglerparameter, z.B. je nach aktueller Größe des Fehlers (z.B. quadratisches Mittel), um schnelle Konvergenz und hohes Maß an Stabilität zu erreichen.

7.3 Ausblick

Für die weitere Verifikation und Verbesserung des hier vorgestellten Verfahrens müssen weitere Versuche mit unterschiedlichen Strukturauteilen durchgeführt werden. Da bislang nur ein empirischer Ansatz entwickelt wurde, die Reglerparameter richtig einzustellen ist es essentiell, diesen Ansatz weiter zu belegen oder einen analytischen bzw. numerischen Weg zu finden, der am Versuchsstand ohne exakte vorherige Kenntnis der Struktur und des Versuchsaufbaus ohne großen Aufwand durchgeführt werden kann.

Insbesondere wäre die Untersuchung von Strukturen mit einer Hysterese interessant, die Frequenzen unterhalb der Erreger–Frequenz aufweisen können. Es wird vermutet, dass Signalanteile mit der doppelten Regler–Periode (evtl. auch mit der vierfachen, sechsfachen, achtfachen usw.) nicht durch den Regler kompensiert werden können. Hier wird vorgeschlagen, die Periode des Reglers zu verdoppeln (bzw. zu verdreifachen oder zu vervierfachen usw.), um auch niederfrequente Störungen noch zu erfassen und eliminieren zu können.

Die Anpassung der Pegel für den Ein- und Ausgang der Regelung hat sich in der Praxis als unpraktisch erwiesen. Abhilfe könnte hier eine einfache Maßnahme wie eine Referenztabelle für die nötige Einstellung der Potentiometer je nach gewünschter Amplitude schaffen. Schöner wäre jedoch, die Amplituden entweder über die Regler–Steuerung (Serielle Konsole) anpassen zu können oder sogar als automatische Einstellung in den Algorithmus zu integrieren. Dies könnte z.B. durch Verwendung digitaler Potentiometer wie dem AD5242 geschehen, welche anstelle der jetzigen manuellen Potentiometer verwendet werden können. Weiterer Vorteil wäre die exakte Reproduzierbarkeit und Dokumentierbarkeit der Reglereinstellungen. Für den Eingang könnte die Pegelanpassung auch über Einstellung der Referenzspannung des ADC–Moduls oder über die Anpassung des ADC–gains erfolgen (hier sind intern 3 Stufen in den Faktoren 0,5 1 und 2 möglich) [6, Kap.44]. Eine

weitere, darauf aufbauende Verbesserungsidee ist die automatische Anpassung der Verstärkung während eines Sweep-Experiments, um den hohen Dynamikbereich der Übertragungsfunktion im Bereich von Resonanzfrequenzen besser abbilden zu können.

Zur Verbesserung der Konvergenzgeschwindigkeit, Stabilität und Regelgüte könnten außerdem andere Lerngesetze implementiert und erprobt werden. Insbesondere wären hier Möglichkeiten von Interesse, die Reglerparameter automatisch anzupassen, wofür [28] Ansätze geben könnte. Der Einfluss eines geeigneten Stellgrößenfilters Q könnte ebenfalls untersucht werden. In [42] oder [13] werden durch einen Q -Operator auch Stellgrößen der vergangenen Regler-Iteration verwendet, wodurch eine ILR höherer Ordnung entsteht.

Danksagung

Besonderer Dank gilt meinen Gutachtern Herrn Prof. Zehn und Herrn Prof. King sowie Herrn Dipl.-Ing. Tobias Rademacher und Herrn Dipl.-Ing. Matthias Kiesner für die Ermöglichung und Betreuung dieser Arbeit. Meine Betreuer unterstützten mich bei dieser interdisziplinären Arbeit sowohl inhaltlich als auch methodisch. Inspiration für diese Arbeit stammt sicherlich aus den spannenden und anregenden Vorlesungen von Herrn Prof. King. Dem Fachgebiet von Herr Prof. Zehn bin ich besonders dankbar für die anwendungsnahe Entwicklung des Themas und die umfangreiche Unterstützung bei der praxisnahen Verifikation an den Versuchseinrichtungen. In diesem Zusammenhang sei auch Herrn Dipl.-Ing. Johannes Thaten für Hilfe mit der Messtechnik gedankt. Außerdem gilt mein Dank meinem Kommilitonen und Kollegen Herrn Robert Felten, der mich bei Fragen zur Implementierung und zum Debugging der Firmware beraten hat. Ich danke auch meiner Partnerin Dr.-Ing. Birthe Grzembra für die Beratung in Fragen der Form, Wissenschaftlichkeit und Konsistenz.

Nicht zu vergessen sind meine Freunde und meine Mutter, die während des Masterstudiums an meiner Seite standen und mich moralisch gestärkt und unterstützt haben. Schlussendlich geht ein herzlicher Dank an meinen Kompagnon Tristan August und seine Frau Julia August, die mir mit ihrem Einsatz in der Firma ausreichend Raum und Zeit geschaffen haben, diese Arbeit zu vollenden.

Abbildungsverzeichnis

3.1	Blockschaltbild Standard–Regelkreis	19
4.1	Messaufbau mit Glasfaserplatte	29
4.2	Glasfaserplatte und Shaker	30
4.3	Schematische Darstellung des Versuchsaufbaus	31
4.4	Versuchsaufbau Eisenbahnschiene — Gesamtansicht	32
4.5	Versuchsaufbau Eisenbahnschiene — Ansicht von oben	32
4.6	Schematische Darstellung des vorhandenen Messaufbaus	33
4.7	analoge zwischengeschaltete Regelung	37
4.8	Struktur der Regelung mit Signalerzeugung im Regler	37
5.1	erster Versuch RR mit zwei Ringspeichern	42
5.2	Ringspeicher für RR	43
5.3	Flussdiagramm: Implementierung RR mit einem Ringspeicher	45
5.4	Schematische Darstellung IEPE	46
5.5	Schaltplan der Elektronik mit potentialfreiem Ein- und Ausgang	47
5.6	Übertragungsfunktion des Butterworth Tiefpassfilters 2. Ordnung	49
5.7	Layout Elektronik	52
5.8	Steckbrettaufbau der Elektronik	53
5.9	Integration der Elektronik in ein Gehäuse	54
6.1	Sprungexperiment Schiene — Übersicht	56
6.2	Sprungexperiment Schiene — Ausschnitt	57
6.3	Vergleich Lerngesetz von LONGMAN mit PI-Lerngesetz — Fehlerquadrate	58
6.4	Vergleich Lerngesetz von LONGMAN mit PI-Lerngesetz — Zeitverlauf	59
6.5	Vergleich der Messdaten im Zeitbereich mit und ohne ILR	60
6.6	normierte Darstellung des verwendeten Flat–Top–Fensters SFT5M	61
6.7	Amplitudenverlauf der Erregerkraft Sweep–Experiment mit Glasfaserplatte	62
6.8	Vergleich Messung mit und ohne ILR im Frequenzbereich bei 175 Hz	63
6.9	Vergleich Messung mit und ohne ILR im Frequenzbereich bei 345 Hz	65
6.10	Vergleich Messung mit und ohne ILR im Frequenzbereich bei 454 Hz	65
6.11	Übertragungsfunktion Schiene aus Sweep–Experiment	66
6.12	Vergleich Messung mit und ohne ILR im Frequenzbereich bei 184 Hz	67
6.13	Vergleich Messung mit und ohne ILR im Frequenzbereich bei 368 Hz	69
6.14	Vergleich Stellgröße mit und ohne ILR im Frequenzbereich	70
6.15	Ausschnitt Übertragungsfunktion, Kennzeichnung Sweep–Experiment	71
6.16	Sweep–Experiment mit ILR: Signal zum Aktuator und vom Kraftsensor	72
6.17	Tukey–Fensterfunktion für Sweep Experiment	73
6.18	Kraftsignal des Sweep–Experiments im Frequenzbereich	73
6.19	Kraftsignal des Sweep–Experiments im Frequenzbereich — Ausschnitt	74

Tabellenverzeichnis

4.1	Kenngrößen der Ausgangskarte NI 9263	33
4.2	Kenngrößen der Eingangskarte NI 9234	34
4.3	Technische Daten des Kraftsensors Dytran 1051V3	34
4.4	Technische Daten des SAM3X8E	39
5.1	Befehle an die Regelung über die serielle Konsole	50
6.1	Reglerparameter für die Versuche an der Schiene	61
6.2	Harmonische Verzerrung unter Berücksichtigung aller Frequenzanteile — Glasfaserplatte	62
6.3	Amplituden der Grundfrequenz und höher-harmonischer Störungen für THD — Glasfaserplatte	64
6.4	Reglerparameter für die Versuche an der Schiene	66
6.5	Harmonische Verzerrung unter Berücksichtigung aller Frequenzanteile — Schiene	67
6.6	Amplituden der Grundfrequenz und höher-harmonischer Störungen für THD — Schiene	68

Liste der Quelltexte

1	ILR.ino — Hauptfunktion auf dem Arduino	93
2	Funktionswerte.h — Wertetabelle der Führungsgröße	94
3	updateLaw.ino — ILR update Regel	95
4	changeIndex.ino — interrupt service routine (Echtzeit-Komponente)	95
5	getParamValuesFromSerial.ino — Schnittstelle zur manuellen und automatischen Kontrolle der ILR	96
6	indexShift.ino — Funktion zur Realisierung des Ring-Speichers	100
7	setup.ino — Arduino Funktion zur Initialisierung	100
8	Funktionswerte.m — MATLAB® Skript zur Generierung der Wertetabelle der Führungs- größe	102
9	saveValuesToFile.m — Abspeichern der Funktionswerte für Arduino	102
10	SweepExperimentControl.m — Steuerung eines Sweeps aus MATLAB®	103

Anforderungsliste

A	Anforderung: (Integration in den bestehenden Messaufbau)	16
B	Anforderung: (Frequenzbereich)	16
C	Wunsch: (erweiterter Frequenzbereich)	16
D	Wunsch: (günstige, handelsübliche Hardware)	17
E	Anforderung: (Obergrenze Kosten für Hardware)	17
F	Anforderung: (Sweep)	17
G	Wunsch: (Einstellung der Regelung)	17
H	Schnittstelle: (Eingang)	17
I	Schnittstelle: (Ausgang)	17
J	Schnittstelle: (Stromversorgung)	17
K	Schnittstelle: (Programmierbarkeit des Reglers)	18
L	Schnittstelle: (Anpassung von Parametern)	18

Literaturverzeichnis

- [1] AHN, HYO-SUNG, Y. CHEN und K. L. MOORE (2007). *Iterative Learning Control: Brief Survey and Categorization*. IEEE Transaction on Systems, Man, and Cybernetics, 37(6):1099ff.
- [2] ARDUINO CC (2015a). *Arduino Due*. website: <http://arduino.cc/en/Main/ArduinoBoardDue>. letzter Zugriff: 26. Februar 2015.
- [3] ARDUINO CC (2015b). *Arduino Due schematic*. PDF. <http://www.arduino.cc/en/uploads/Main/arduino-Due-schematic.pdf> letzter Zugriff: 10. April 2016.
- [4] ARDUINO SA (2015). *Arduino Reference Page*. website <https://www.arduino.cc/en/Reference/HomePage>. Zuletzt besucht: 10. April 2016.
- [5] ARIMOTO, SUGURU, S. KAWAMURA und F. MIYAZAKI (1984). *Bettering operation of Robots by learning*.
- [6] ATMEL CORPORATION (2012). *AT91SAM ARM-based Flash MCU — SAM3X, SAM3A Series*. datasheet.
- [7] AVK — INDUSTRIEVEREINIGUNG VERSTÄRKTE KUNSTSTOFFE E. V. (2014). *Handbuch Faserverbundkunststoffe / Composites — Grundlagen, Verarbeitung, Anwendungen*. Springer Vieweg, 4 Aufl. Editor: Elmar Witten, ISBN: 978-3-658-02755-1.
- [8] BLAGOUCHINE, IAROSLAV V. (2011). *Analytic Method for the Computation of the Total Harmonic Distortion by the Cauchy Method of Residues*. IEEE Transaction on Communivations, 59(9):2478–2491.
- [9] BRÜEL & KJÆR (2016a). *LDS Power Amplifiers*. Brüel & Kjær Sound & Vibration Measurement A/S, DK-2850 Nærum, Denmark. <http://www.bksv.com/doc/bu3077.pdf> letzter Zugriff: 9. April 2016.
- [10] BRÜEL & KJÆR (2016b). *LDS V406 and V408 Shakers*. Brüel & Kjær Sound & Vibration Measurement A/S, DK-2850 Nærum, Denmark. <http://www.bksv.com/Products/shakers-excitors/lds-vibration-test/shakers/permanent-magnet/V406> letzter Zugriff: 9. April 2016.
- [11] BRÜEL & KJÆR (2016c). *LDS V450, V451, V455 and V456 Shakers*. Brüel & Kjær Sound & Vibration Measurement A/S, DK-2850 Nærum, Denmark. <http://www.bksv.com/doc/bp2401.pdf> letzter Zugriff: 9. April 2016.
- [12] CHEN, YANGQUAN und K. L. MOORE (2000). *Comments on United States Patent 3,555,252*. researchgate.
- [13] CHEN, YANQUAN und J.-X. XU (1998). *A Higher-order Terminal Iterative Learning Control Scheme*. Technischer Bericht, Dept. of Electrical Engineering, National University of Singapore; School of EEE, Nanyang Technological University Singapore.
- [14] DYTRAN INSTRUMENTS INC. (2016). *IEPE Force Sensor Model 1051V3*. <http://www.dytran.com/Model-1051V3-IEPE-Force-Sensor-P2504.aspx> letzter Zugriff: 2. April 2016.
- [15] EWINS, D. J. (2000). *Modal Testing — Theory, Practice and Application*. John Wiley & Sons, 2 Aufl. ISBN: 978-0863802188.

- [16] GARDEN, MURRAY (1971). *Learning control of actuators in control systems.* Patent US3555252.
- [17] GRASSE, FABIAN, M. ZUR und V. TRAPPE (2014). *Schubversuch mittels Schubrahmen.* Messen und Prüfen, 59(10):10–13.
- [18] HARTE, T. J., J. HÄTÖNEN und D. H. OWENS (2005). *Discrete-time inverse model-based iterative learning control: stability, monotonicity and robustness.* International Journal of Control, 78(8):577–586. DOI: 10.1080/00207170500111606.
- [19] HASHIN, ZVI, D. BAGCHI und B. W. ROSEN (1974). *Non-linear Behaviour of Fiber Composite Laminates by.* NASA contractor report NASA CR-2313, Materials Science Corporation, Blue Bell, Pennsylvania.
- [20] HEINZEL, G., A. RÜDIGER und R. SCHILLING (2002). *Spectrum and spectral density estimation by the Discrete Fourier transform (DFT), including a comprehensive list of window functions and some new flat-top windows.* Report, Max-Planck-Institut für Gravitationsphysik (Albert-Einstein-Institut) Teilinstitut Hannover.
- [21] HERRMANN, ANDREA, R. WEISSBACH und E. KNAUS (2013). *Requirements Engineering und Projektmanagement.* Springer Vieweg. ISBN 978-3-642-29431-0.
- [22] HULL, ELIZABETH, K. JACKSON und J. DICK (2011). *Requirements Engineering.* Springer-Verlag. ISBN 978-1-84996-404-3.
- [23] JACKISCH, UTZ-VOLKER und T. HIPKE (2014). *Möglichkeiten und Anwendungen moderner Werkstoffe für den hochdynamischen Textilmaschinenbau.* In: *Ressourceneffiziente Maschinen und Verfahren*, 14. Chemnitzer Textiltechnik Tagung. https://www.tu-chemnitz.de/mb/FoerdTech/ctt/pdf/1_1.pdf letzter Zugriff 18. März 2016.
- [24] JOSEFSSON, ANDREAS, M. MAGNEVALL und K. AHLIN (2006). *Control Algorithm For Sine Excitation On Nonlinear Systems.* Technischer Bericht, Department of Mechanical Engineering, Blekinge Institute of Technology.
- [25] KAWAMURA, PROF. SADAO und D. M. SVININ, Hrsg. (2006). *Advances in Robot Control.* Springer Verlag Berlin Heidelberg.
- [26] KING, PROF. DR.-ING. RUDIBERT (2012). *Grundlagen der Mess- und Regelungstechnik.* Technische Universität Berlin, Institut für Prozess- und Verfahrenstechnik, FG Mess- und Regelungstechnik. Vorlesungsskript.
- [27] KING, PROF. DR.-ING. RUDIBERT (2014). *Lernende Regelung.* Technische Universität Berlin, Institut für Prozess- und Verfahrenstechnik, FG Mess- und Regelungstechnik. Vorlesungsskript.
- [28] LONGMAN, R. W. und S. WIRKANDER (1998). *Automated tuning concepts for iterative learning and repetitive control laws.* In: *Proceeding of the 37th IEEE Conference on Decision and Control*, Bd. 1, S. 192–1198.
- [29] LONGMAN, RICHARD W. (2010). *Iterative learning control and repetitive control for engineering practice.* International Journal of Control, 73(10):930–954.
- [30] LUNZE, PROF. DR.-ING. JAN (2014a). *Regelungstechnik 1.* Springer Vieweg, Berlin Heidelberg, 10 Aufl. ISBN 978-3-642-53908-4.
- [31] LUNZE, PROF. DR.-ING. JAN (2014b). *Regelungstechnik 2.* Springer Vieweg, 8 Aufl. ISBN 978-3-642-53943-5.
- [32] MADADY, ALI (2008). *PID Type Iterative Learning Control with Optimal Gains.* International Journal of Adhesion and Adhesives, 6(2):192–208.

- [33] MAGNEVALL, MARTIN, A. JOSEFSSON und K. AHLIN (2006). *Experimental Verification of a Control Algorithm for Nonlinear Systems*. Technischer Bericht, Department of Mechanical Engineering, Blekinge Institute of Technology.
- [34] NATIONAL INSTRUMENTS GERMANY GMBH (2014a). *NI 9234 — 4 analoge Eingänge*. website. <http://sine.ni.com/nips/cds/view/p/lang/de/nid/208802> letzter Zugriff: 26. Februar 2015.
- [35] NATIONAL INSTRUMENTS GERMANY GMBH (2014b). *NI 9234 Produktdatenblatt*. Ganghoferstr. 70 b 80339 München.
- [36] NATIONAL INSTRUMENTS GERMANY GMBH (2014c). *NI 9263 Analogausgangsmodul mit 4 Kanälen*. website. <http://sine.ni.com/nips/cds/view/p/lang/de/nid/208806> letzter Zugriff: 26. Februar 2015.
- [37] NATIONAL INSTRUMENTS GERMANY GMBH (2014d). *NI 9263 Produktdatenblatt*. Ganghoferstr. 70 b 80339 München. <http://sine.ni.com/ds/app/doc/p/id/ds-59/lang/de> letzter Zugriff: 26. Februar 2015.
- [38] NATIONAL INSTRUMENTS GERMANY GMBH (2015a). *NI-CompactDAQ-USB-Chassis mit 8 Steckplätzen*. website. <http://sine.ni.com/nips/cds/view/p/lang/de/nid/207534> letzter Zugriff: 26. Februar 2015.
- [39] NATIONAL INSTRUMENTS GERMANY GMBH (2015b). *NI9178 Produktdatenblatt*. Ganghoferstr. 70 b 80339 München. <http://www.ni.com/pdf/manuals/374046a.pdf> letzter Zugriff: 13. April 2016.
- [40] NATIONAL INSTRUMENTS GERMANY GMBH (2015c). *Systemdesignsoftware NI LabVIEW*. website. <http://www.ni.com/labview/d/> letzter Zugriff: 26. Februar 2015.
- [41] NATIONALE PLATTFORM ELEKTROMOBILITÄT (2014). *Fortschrittsbericht 2014 — Bilanz der Markt vorbereitung*. Technischer Bericht, Gemeinsame Geschäftsstelle Elektromobilität der Bundesregierung, Scharnhorststraße 34–37, 10115 Berlin.
- [42] NORRLÖF, MIKAEL und S. GUNNARSSON (2002). *Time and frequency domain convergence properties in iterative learning control*. International Journal of Control, 75(14):1114–1126.
- [43] OWENS, D.H., J. J. HATONEN und S. DALEY (2009). *Robust monotone gradient-based discrete-time iterative learning control*. International Journal of Robust and non-linear control, 19:634–661. DOI: 10.1002/rnc.1338.
- [44] PARK, KWAN-HYUN, Z. BIEN und D.-H. HWANG (1999). *PID-type Iterative Learning Controller against Initial State Error*. Technischer Bericht, Department of Electrical Engineering, Korea Advanced Institute of Science and Technology; Department of Electronics Engineering Chungnam National University.
- [45] RADEMACHER, TOBIAS und M. ZEHN (2016). *Modal triggered Non-Linearities for Damage Localization in thin walled FRC Structures — a numerical study*. Facta Universitatis: Series Mechanical Engineering, 14(1):21 – 36. unveröffentlicht, angenommenes Paper.
- [46] RASPBERRY PI FOUNDATION (2015). *Raspberry Pi 2 Model B*. Website, <https://www.raspberrypi.org/products/raspberry-pi-2-model-b/>. letzter Zugriff: 12. März 2016.
- [47] SANDERS, D.R., Y. KIM und N. STUBBS (1992). *Nondestructive Evaluation of Damage in Composite Structures Using Modal Parameters*. Experimental Mechanics, 32(3):240 – 251.
- [48] SCHÜRMANN, HELMUT (2007). *Konstruieren mit Faser-Kunststoff-Verbunden*. Springer-Verlag, 2 Aufl.

- [49] SEIDEL, IVAN (2016). *DueTimer Library*. online <https://github.com/ivanseidel/DueTimer>. letzter Zugriff: 7. April 2016.
- [50] SHMILOVITZ, DORON (2005). *On the Definition of Total Harmonic Distortion and Its Effect on Measurement Interpretation*. IEEE Transaction on Power Delivery, 20(1):526–528.
- [51] STMICROELECTRONICS (2014). *STM32 32-bit MCU family*. http://www.st.com/web/en/resource/technical/document/reference_manual/DM00031020.pdf letzter Zugriff: 10. April 2016.
- [52] TEXAS INSTRUMENTS (2015). *LM324 Datasheet*. PDF, <http://www.ti.com/lit/ds/symlink/lm124-n.pdf>. letzter Zugriff: 23.1.2016.
- [53] VISOOLI, ANTONIO (2006). *Practical PID Control*. Springer Verlag London.
- [54] VOIT-NITSCHMANN, R., T. KEILIG und M. BERESINSKI (1999). *Technologieverbesserung an Faserverbund-Strukturen von Flugzeugen der Allgemeinen Luftfahrt*. Technischer Bericht, Universität Stuttgart, Fakultät Luft- und Raumfahrttechnik, BFE Bereich Flugzeugentwurf. BMBF/BMWi-Förderkennzeichen 20L 9705A.
- [55] VÖTH, STEFAN (2006). *Dynamik schwingungsfähiger Systeme*. Friedr. Wieweg & Sohn Verlag (Studium und Technik), 1 Aufl. Editor: Thomas Zipsner, ISBN: 3-8348-0111-9.
- [56] WEBER, MANFRED (2016). *IEPE-Standard*. website. <http://www.mmf.de/iepe-standard.htm> letzter Zugriff: 8. April 2016.
- [57] WERNER, MARTIN (2012). *Digitale Signalverarbeitung mit MATLAB®*. Vieweg+Teubner, 5 Aufl. ISBN 978-3-8348-1473-9.
- [58] ZACHER, SERGE und M. REUTER (2014). *Regelungstechnik für Ingenieure*. Springer Vieweg, 14 Aufl. ISBN 978-3-8348-1786-0.

8 Anhang

8.1 Arduino Code

Quelltext 1: ILR.ino — Hauptfunktion auf dem Arduino

```
1  /*
2   * Arduino Due Sketch for ILC
3
4   * Author: Felix Piela
5   * Feb 2016
6  */
7
8  /* DEFINES */
9 // short call to debug messages
10 #define DEBUGPRINT(x) if(debug==true) Serial.println( x );
11
12 /* INCLUDING LIBRARIES */
13 // #include <avr/pgmspace.h> // accessing program memory
14 #include "DueTimer.h"
15 #include <stdio.h>
16 #include <stdlib.h>
17
18 // loading data
19 #include "Funktionswerte.h" // loading file with values of desired
   function
20
21 // DECLARING CONSTANTS -----
22 // for PINS
23 const unsigned int PIN_ADC = A0;
24 const unsigned int PIN_DAC = DAC1;
25 const unsigned int PIN_HARDWAREDEBUG = 22;
26 const unsigned int PIN_ADC_OVERRANGE = 52;
27 const unsigned int PIN_DAC_OVERRANGE = 53;
28
29 // DECLARING VARIABLES -----
30 // core ILC
31 int Nsmooth = 6;           // number of values used as smoothing in update
   law (must be smaller than NsmoothMax)
32 float Ki = 0.001;          // I gain of ILC
33 long errorSum[Nval];       // error sum for I part of control
34 float Kp = 0.000;          // P gain of ILC
35 int PhaseLead = 4;         // discrete Phase Lead for digital smoothing in
   update law (must be in the range of [1:Nval-Nsmooth])
36 boolean Stop = true;        // start and stop the experiment
37
38 float outputSignal[Nval];    // array of values to be written on DAC
39 int error[Nval];           // array of values with errors (desired
```

```

        value - real value)
40 volatile unsigned int timeIndex = 0; // index to be incremented by ISR
41
42 // Variables for Serial Communication
43 String inputString1 = ""; // a string to hold incoming data for
   menu
44 String inputString2 = ""; // a string to hold incoming data for
   menu
45 boolean stringComplete = false; // whether the string is complete
46 int menuState = 0;
47
48
49 // debugging variables
50 const boolean debug = false; // debugging with Serial Console (
   works only for very low sample frequ.)
51 const boolean hardwareDebug = false; // debugging with measuring
   certain timings via digital i/o PINs
52 unsigned long slowEventTime = 0;
53
54 void loop() {
55     getParamValuesFromSerial(true); // use with true for verbose output
56
57     if ((millis() - slowEventTime) > 1000)
58     {
59         slowEventTime = millis(); // reset overrange indicator PINs after
           one second
60         digitalWrite(PIN_DAC_OVERRANGE, LOW);
61         digitalWrite(PIN_ADC_OVERRANGE, LOW);
62     }
63     delay(10);
64 }
```

Quelltext 2: Funktionswerte.h — Wertetabelle der Führungsgröße

```

1 // values for ILR control
2 // generated by matlab
3 // target frequency: 368 Hz
4
5 #ifndef _Funktionswerte_h_
6 #define _Funktionswerte_h_
7
8 const unsigned int Nval = 81; // number of entries in outputSignal
9 float sampleFreq = 30000; // sampling frequency [Hz]
10 const unsigned int table[] = {
11     2048, 2086, 2125, 2162, 2200, 2236, 2271, 2304, 2337, 2367, 2396, 2422,
12     2446, 2468, 2487, 2504, 2518, 2529, 2537, 2542, 2544, 2544, 2540, 2533,
13     2523, 2511, 2496, 2478, 2457, 2434, 2409, 2381, 2352, 2321, 2288, 2253,
14     2218, 2181, 2144, 2106, 2067, 2029, 1990, 1952, 1915, 1878, 1842, 1808,
15     1775, 1744, 1714, 1687, 1662, 1639, 1618, 1600, 1585, 1572, 1563, 1556,
16     1552, 1552, 1554, 1559, 1567, 1578, 1592, 1609, 1628, 1650, 1674, 1701,
17     1729, 1760, 1792, 1825, 1860, 1897, 1934, 1972, 2010
18 };
19
20 #endif
```

Quelltext 3: updateLaw.ino — ILR update Regel

```

1 float updateLaw(unsigned int i)
2 /*
3  * ILC update law
4  * derived from Longman:
5  * Iterative learning control and repetitive control for engineering
6  * practice
7  *
8  * input: current time index
9  * returns: computed output signal
10 */
11 {
12     float newOutputSignal;
13     int smoothedError = 0;
14
15     // smoothing error values
16     for (unsigned int j=0; j<Nssmooth; j++)
17     {
18         smoothedError = smoothedError + error [indexShift(i + j)];
19     }
20     smoothedError = smoothedError / Nsmooth;
21     errorSum [i] = errorSum [i] + smoothedError;
22
23     // applying PI style update law
24     newOutputSignal = 2048.0 + Kp * (float)smoothedError + Ki * (float)
25             errorSum [i];
26
27     if (newOutputSignal > 4095) // Signal runs into upper limit of DAC
28     {
29         newOutputSignal = 4095;
30         // DEBUGPRINT("Signal on upper limit of DAC");
31         digitalWrite(PIN_DAC_OVERRANGE, HIGH);
32     }
33     if (newOutputSignal < 0)
34     {
35         newOutputSignal = 0;           // Signal runs into lower limit of DAC
36         // DEBUGPRINT("Signal on lower limit of DAC");
37         digitalWrite(PIN_DAC_OVERRANGE, HIGH);
38     }
39
40     return newOutputSignal;
41 }
```

Quelltext 4: changeindex.ino — interrupt service routine (Echtzeit-Komponente)

```

1 void changeIndex() {
2     int ADCvalue;
3     // if(hardwareDebug==true){ digitalWrite(PIN_HARDWAREDEBUG,HIGH); }
4     ADCvalue = analogRead(PIN_ADC);
5     if(ADCvalue < 100) digitalWrite(PIN_ADC_OVERRANGE,HIGH);
6     if(ADCvalue > 4000) digitalWrite(PIN_ADC_OVERRANGE,HIGH);
7
8     error [timeIndex] = table [timeIndex]- ADCvalue;
9 }
```

```

10    outputSignal[timeIndex] = updateLaw(timeIndex); // applying update
11    law
12
13    analogWrite(PIN_DAC, (int)outputSignal[timeIndex]); // setting DAC
14    value
15
16    timeIndex = timeIndex + 1;
17    if (timeIndex >= Nval) // checking if Period is complete
18    {
19        timeIndex = 0;
20    }
21    //if(hardwareDebug==true){ digitalWrite(PIN_HARDWAREDEBUG,LOW); }
22 }
```

Quelltext 5: getParamValuesFromSerial.ino — Schnittstelle zur manuellen und automatischen Kontrolle der ILR

```

1 void getParamValuesFromSerial(boolean rich)
2 {
3     int userInput1;
4     int userInput2;
5     if (stringComplete) {
6         stringComplete = false;
7         userInput1 = atoi(inputString1.c_str()); // convert String from
8             Serial to int
9         userInput2 = atoi(inputString2.c_str());
10
11     switch(userInput1)
12     {
13         // START / STOP MENU
14         case 1: // set value for stop variable
15             menuState = 1;
16             if (rich==true) Serial.println("start [1], pause [2] or reset [3] experiment");
17             switch (userInput2)
18             {
19                 case 0:
20                     break; // no second input yet
21                 case 1:
22                     Timer3.start(); // starting ISR on internal interrupt
23                     if (rich==true) Serial.println("starting Experiment... ");
24                     returnToMenu();
25                     break;
26                 case 2:
27                     Timer3.stop(); // stopping ISR
28                     if (rich==true) Serial.println("pausing Experiment... ");
29                     returnToMenu();
30                     break;
31                 case 3:
32                     Timer3.stop();
33                     if (rich==true) Serial.println("resetting Experiment... ");
34                     initOutput();
35                     if (rich==true) Serial.println("all values have been
36                         resetted");
37                     returnToMenu();
38 }
```

```

36         break;
37     default:
38         userInput2 = 0;
39         returnToMenu();
40         if (rich==true) Serial.println("invalid input in stop parameter field");
41     }
42     break;
43
44 // SETTING SAMPLE FREQUENCY
45 case 2: // set value for sampleFreq
46     menuState = 2;
47     if (rich==true) {Serial.println("set value for sample frequency");}
48     switch (userInput2)
49     {
50     case 0:
51         break; // no second input yet
52     default:
53         if (userInput2 > 100000)
54         {
55             if (rich==true) Serial.println("value is too high (System would freeze)");
56         }
57         else
58         {
59             sampleFreq = (unsigned int)userInput2;
60             Timer3.setFrequency(sampleFreq);
61             Timer3.start(); // restart Timer since it seems
62                         // to stop after setPeriod
63             if (rich==true)
64             {
65                 Serial.print("sample frequency changed to ");
66                 Serial.print(sampleFreq);
67                 Serial.print(" hertz\n");
68             }
69             returnToMenu();
70             userInput2 = 0;
71             break;
72         }
73     break;
74
75 // SETTING ILC GAIN
76 case 3:
77     menuState = 3;
78     if (rich==true) Serial.println("set value for I control gain (times 1000, type 1 for 0.001)");
79     switch(userInput2)
80     {
81     case 0:
82         break; // no second input yet
83     default:
84         Ki = (float)userInput2;

```

```
85          if (Ki == 999)
86          {
87              Ki = 0;
88          }
89          {
90              Ki = Ki / 1000;
91          }
92          if (rich==true)
93          {
94              Serial.print("Ki\u201cuchanged\u201cto\u201d");
95              Serial.print(Ki,3);
96              Serial.print("\n");
97          }
98          userInput2 = 0;
99          returnToMenu();
100         break;
101     }
102     break;
103
104 case 4:
105     menuState = 4;
106     if (rich==true) Serial.println("set\u201cvalue\u201ffor\u201cP\u201ccontrol\u201cgain\u201c
107         times\u201cu1000,\u201ctype\u201cu1\u201ffor\u201c0.001\u201d)");
108     switch(userInput2)
109     {
110         case 0:
111             break; // no second input yet
112         default:
113             Kp = (float)userInput2;
114             if (Kp == 999)
115             {
116                 Kp = 0;
117             }
118             Kp = Kp / 1000;
119         }
120         if (rich==true)
121         {
122             Serial.print("Kp\u201cuchanged\u201cto\u201d");
123             Serial.print(Kp,3);
124             Serial.print("\n");
125         }
126         userInput2 = 0;
127         returnToMenu();
128         break;
129     }
130     break;
131 // SETTING PHASE LEAD
132 case 5:
133     menuState = 5;
134     if (rich==true) Serial.println("set\u201cvalue\u201ffor\u201cPhaseLead\u201cof\u201cILC"
135         );
136     switch(userInput2)
137     {
```

```

137     case 0:
138         break; // no second input yet
139     default:
140         PhaseLead = (int)userInput2;
141         if (rich==true)
142         {
143             Serial.print("ILC_Phase_Lead_changed_to_");
144             Serial.print(PhaseLead);
145             Serial.print("_time_steps\n");
146         }
147         returnToMenu();
148         break;
149     }
150     break;
151 // SETTING MOVING AVERAGE FILTER WIDTH
152 case 6:
153     menuState = 6;
154     if (rich==true) Serial.println("set_width_of_moving_average_low"
155                                     "pass_filter");
156     switch(userInput2)
157     {
158         case 0:
159             break; // no second input yet
160         default:
161             Nsmooth = (float)userInput2;
162             if (rich==true)
163             {
164                 Serial.print("Nsmooth_changed_to_");
165                 Serial.println(Nsmooth);
166             }
167             userInput2 = 0;
168             returnToMenu();
169             break;
170     }
171     break;
172 // PRINT MENU IF SOMETHING WEIRD WAS ENTERED
173 default:
174     returnToMenu();
175     if (rich==true)
176     {
177         Serial.println("invalid_input_in_menu");
178         Serial.print("\nusage:");
179         Serial.print("\n1: Start/Stop/Reset");
180         Serial.print("\n2: set_sample_rate-sampleFreq=");
181         Serial.print(sampleFreq);
182         Serial.print("\n3: set_ILC_I_gain-Ki=");
183         Serial.print(Ki,3);
184         Serial.print("\n4: set_ILC_p_gain-Kp=");
185         Serial.print(Kp,3);
186         Serial.print("\n5: set_PhaseLead-PhaseLead=");
187         Serial.print(PhaseLead);
188         Serial.print("\n6: set_moving_average_filter_width(Lowpass"
189                     "-Nsmooth=");
190         Serial.print(Nsmooth);
191         Serial.print("\nactual_sample_rateis:");
192         Serial.print(

```

```

185     Timer3.getFrequency()); Serial.print("actual frequency
186     is:"); Serial.print(Timer3.getFrequency()/Nval);
187     Serial.print("\n");
188 }
189 }
190 }
191
192 void serialEvent() {
193     while (Serial.available()) {
194         // get the new byte:
195         char inChar = (char)Serial.read();
196         if (menuState==0) // base menu
197         {
198             inputString1 += inChar; // add it to the inputString1:
199         }
200         else // parameter setting
201         {
202             inputString2 +=inChar;
203         }
204
205         // if the incoming character is a newline, set a flag
206         // so the main loop can do something about it:
207         if (inChar == '\n') {
208             stringComplete = true;
209         }
210     }
211 }
212
213 void returnToMenu() {
214     menuState = 0; // return to main menu
215     inputString1 = ""; // clear the Serial communication string
216     inputString2 = "";
217 }
```

Quelltext 6: indexShift.ino — Funktion zur Realisierung des Ring-Speichers

```

1 unsigned int indexShift(unsigned int i)
2 /*
3     Index Shift in a FIFO ring buffer manner
4     input is current index
5     returns shifted index
6 */
7 {
8     unsigned int shiftedIndex;
9     if (i < (Nval - PhaseLead))           // rollover
10     { shiftedIndex = i + PhaseLead; }
11     else                                // no rollover needed
12     { shiftedIndex = i - (Nval - PhaseLead); }
13
14     return shiftedIndex;
15 }
```

Quelltext 7: setup.ino — Arduino Funktion zur Initialisierung

```

1 void setup() {
2   // setting resolution of ADC and DAC to 12 bit
3   analogWriteResolution(12);
4   analogReadResolution(12);
5
6   // Using internal Timer interrupt (DueTimer library)
7   Timer3.setFrequency(sampleFreq);
8   Timer3.attachInterrupt(changeIndex).stop(); // init Timer in stop
      condition
9
10  // using Serial Interface for debugging
11  Serial.begin(115200);
12  // Serial.println("Arduino is up and running");
13  inputString1.reserve(3); // reserve some bytes for the inputString
14  inputString2.reserve(3); //
15
16  // configuring PINS
17  pinMode(PIN_ADC, INPUT);
18  pinMode(PIN_DAC, OUTPUT);
19  pinMode(PIN_HARDWAREDEBUG, OUTPUT);
20  digitalWrite(PIN_HARDWAREDEBUG, LOW);
21  pinMode(PIN_ADC_OVERRANGE, OUTPUT);
22  digitalWrite(PIN_ADC_OVERRANGE, LOW);
23  pinMode(PIN_DAC_OVERRANGE, OUTPUT);
24  digitalWrite(PIN_DAC_OVERRANGE, LOW);
25
26  initOutput();
27
28  if(debug==true) delay(1000);
29 }
30
31 void initOutput()
32 {
33   DEBUGPRINT("gelesene Werte:")
34   for (int j = 0; j<Nval; j++)
35   {
36     error[j] = 0;           // initialising error values
37     errorSum[j] = 0;        // resetting I sum
38     outputSignal[j] = table[j]; // setting DAC values to data table
39                           // use pgm_read_word(&table[j]) if
                           // table is stored in PROGMEM
40     analogWrite(PIN_DAC, 2048); // Write mean value to DAC so that
                                // physics can settle
41
42     // printing outputSignal on Serial Console for debugging
43     DEBUGPRINT(outputSignal[j]);
44   }
45 }
```

8.2 Matlab® Code

Quelltext 8: Funktionswerte.m — MATLAB® Skript zur Generierung der Wertetabelle der Führungsgröße

```
1 %% script to automatically create values for ILC
2 % in sine waveform
3 %
4 % Felix Piela
5 % Feb 2016
6
7 % Frequenz
8 f = 368;           % desired frequency [Hz]
9 f_s = 30e3;        % desire sample rate [Hz]
10 A = 0.4;          % desired amplitude [V]
11 DCfactor = 0.5;   % factor for DC offset (0.5 for midrange)
12
13 % DAC properties
14 V_ref = 3.3;      % reference voltage [V]
15 bit_DAC = 12;     % DAC resolution
16 offset = 2^bit_DAC * DCfactor; % calculate integer offset value for DAC
17
18 % filename for output
19 filename = './ILR/Funktionswerte.h';
20
21 % calculate periods
22 T = 1/f;
23 T_s = 1/f_s;
24
25 t = linspace(0, T-T_s, T/T_s);
26
27 % calculate values
28 y = ((2^bit_DAC)-1) / V_ref * (A*sin(2*pi*f*t)) + offset;
29 y=round(y); % round to integer values
30
31 %% save values
32 saveValuesToFile(filename, f, f_s, T, T_s, y, t);
```

Quelltext 9: saveValuesToFile.m — Abspeichern der Funktionswerte für Arduino

```
1 function error = saveValuesToFile(filename, f, f_s, T, T_s, y, t)
2 % Function for saving data to a arduino readable header File
3 % PROGMEN functionality to be added later
4 %
5 % author: Felix Piela
6 % 2016
7 %
8 % part of master thesis in iterative learning control
9 %
10 % USAGE
11 % saveValuesToFile(filename, f, f_s, T, T_s, y, t)
12 %
13 % returns 0 if everything worked fine (to be improved)
14
15 disp(['speichere Daten in Datei ',filename,' ...']);
```

```

16 disp(['Frequenz:',int2str(f), 'Hz']);
17 disp(['Abtastfrequenz:',int2str(f_s), 'Hz']);
18 disp(['... das entspricht', int2str(f_s/f), ' SampleproPeriode']);
19
20 fid = fopen(filename, 'w+');
21
22 % write header
23 fprintf(fid, '//values for ILR control\n');
24 fprintf(fid, '//generated by matlab\n');
25 fprintf(fid, '//target frequency: %g Hz\n\n', f);
26 fprintf(fid, '#ifndef _Funktionswerte_h_\n');
27 fprintf(fid, '#define _Funktionswerte_h_\n\n');
28
29 % write parameter
30 fprintf(fid, 'const unsigned int Nval=%g; //number of entries in
   outputSignal\n', length(y));
31 fprintf(fid, 'float sampleFreq=%g; //sampling frequency [Hz]\n', f_s
   );
32 % fprintf(fid, 'const unsigned int Tmic = %g; // length of one period
   in outputSignal [micro s]\n\n', T * 1e6);
33
34 fprintf(fid, 'const unsigned int table[]={\n');
35 for i=1:(length(y)-1)
36     fprintf(fid, '%4.0f, ', y(i));
37     if(mod(i,12)==0)
38         fprintf(fid, '\n'); % newline for better readability
39    end
40 end
41 fprintf(fid, '%4.0f\n', y(end)); % no comma after last value
42 fprintf(fid, '};\n\n');
43 fprintf(fid, '#endif\n'); % write footer
44
45 fclose(fid);
46
47 error = 0;
48 end

```

Quelltext 10: SweepExperimentControl.m — Steuerung eines Sweeps aus MATLAB®

```

1 % script to control sweep experiments
2 %
3 % author: Felix Piela
4 % dec 2015
5 %
6
7 % Arduino needs to be configured so that Serial Console doesn't return
8 % anything. (Proper Messages over Serial still in progress
9 % see https://github.com/pifliffe/ILR/issues/5
10
11 %% HEADER
12 sampleFreq_start = 29000;
13 sampleFreq_end = 31000;
14
15 %% SET UP
16 ILR_Serial = serial('COM8'); % Create Serial Port

```

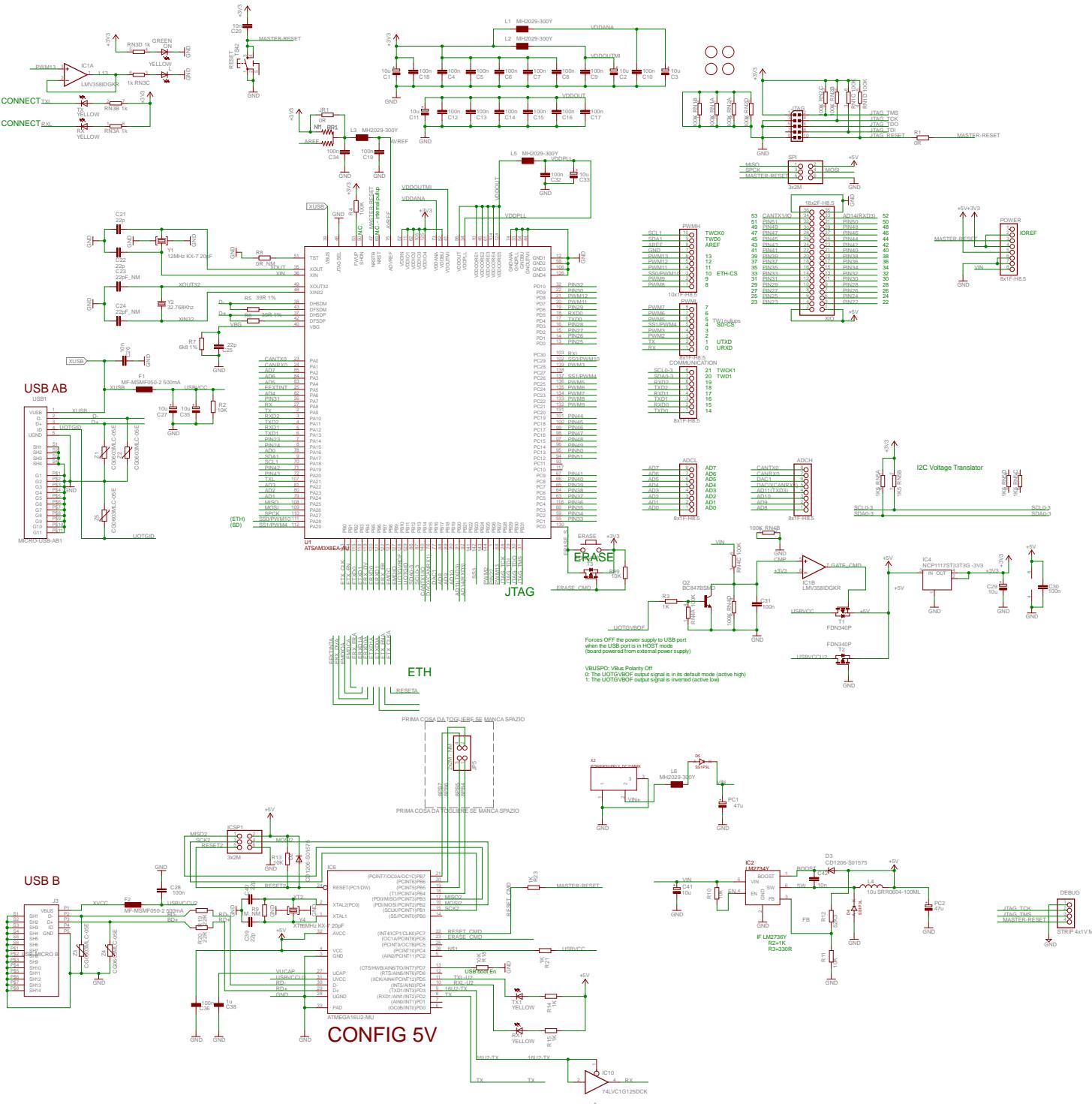
```
17 set ( ILR_Serial , ... % Set Serial Port Properties
18     'BaudRate',115200, ... % set BAUD rate
19     'terminator','LF' ... % set end of message to Line Feed (Arduino)
20 );
21 fopen(ILR_Serial); % Open Serial Port
22 pause(0.3); % seems to need some time to settle
23 pause(2); % wait for system to settle on possible distortion after
               activation
24
25 %% INIT ARDUINO
26 fprintf(ILR_Serial , '2'); % go to sampleFreq menu
27 pause(0.1);
28 fprintf(ILR_Serial ,num2str(sampleFreq_start)); % set to start frequency
29 pause(0.1);
30 fprintf(ILR_Serial , '1');
31 pause(0.1);
32 fprintf(ILR_Serial , '3'); % stop experiment since it is started right
               after setting frequency
33 pause(0.1);
34
35 fprintf(ILR_Serial , '3'); % go to Ki menu % function not implemented
36 pause(0.1);
37 fprintf(ILR_Serial , '30'); % set Ki value
38 pause(0.1);
39
40 fprintf(ILR_Serial , '4'); % go to Kp menu % function not implemented
41 pause(0.1);
42 fprintf(ILR_Serial , '99'); % set Kp value
43 pause(0.1);
44
45 fprintf(ILR_Serial , '5'); % go to PhaseLead menu % function not
               implemented
46 pause(0.1);
47 fprintf(ILR_Serial , '6'); % set PhaseLead
48 pause(0.1);
49
50 fprintf(ILR_Serial , '6'); % go to Nsmooth menu % function not
               implemented
51 pause(0.1);
52 fprintf(ILR_Serial , '10'); % set smoothing width
53 pause(0.1);
54
55
56 %%
57 disp('starting experiment... ');
58 fprintf(ILR_Serial , '1'); % go to start menu
59 pause(0.01);
60 fprintf(ILR_Serial , '1'); % start experiment
61 disp('waiting for start frequency to settle... ');
62 pause(1);
63
64 %% SWEEP
65 disp('start sweep')
66 for i = 0:10:(sampleFreq_end-sampleFreq_start)
```

```
67   fprintf(ILR_Serial , '2'); % go to sample rate menu
68   pause(0.05);
69   fprintf(ILR_Serial ,num2str(sampleFreq_start + i)); % set new sample
    rate
70   %disp ('current sample frequency: ');
71   disp(sampleFreq_start + i);
72   pause(0.05);
73 end
74
75 disp('end_sweep');
76 pause(2);
77
78 %%%
79 disp('ending_experiment_and_closing_connection');
80 fprintf(ILR_Serial , '1'); % go to start menu
81 pause(0.01);
82 fprintf(ILR_Serial , '3'); % stop experiment
83 pause(1);
84
85
86 %% clear session
87 fclose(ILR_Serial); % close Serial Port
88 delete(ILR_Serial); % deleting Serial Port Object
89 clear ILR_Serial;
```

8.3 Datenblätter, Begleitinformationen

Von den Datenblättern sind hier nur die wichtigsten Seiten abgebildet. Für die vollständigen Datenblätter sei auf die Quellen verwiesen:

- Arduino Due: [3]
- LM 324: [52]
- SAM3XE: [6]



LMx24-N, LM2902-N Low-Power, Quad-Operational Amplifiers

1 Features

- Internally Frequency Compensated for Unity Gain
- Large DC Voltage Gain 100 dB
- Wide Bandwidth (Unity Gain) 1 MHz (Temperature Compensated)
- Wide Power Supply Range:
 - Single Supply 3 V to 32 V
 - or Dual Supplies ± 1.5 V to ± 16 V
- Very Low Supply Current Drain (700 μ A) —Essentially Independent of Supply Voltage
- Low Input Biasing Current 45 nA (Temperature Compensated)
- Low Input Offset Voltage 2 mV and Offset Current: 5 nA
- Input Common-Mode Voltage Range Includes Ground
- Differential Input Voltage Range Equal to the Power Supply Voltage
- Large Output Voltage Swing 0 V to $V^+ - 1.5$ V

Advantages:

- Eliminates Need for Dual Supplies
- Four Internally Compensated Op Amps in a Single Package
- Allows Direct Sensing Near GND and V_{OUT} also Goes to GND
- Compatible With All Forms of Logic
- Power Drain Suitable for Battery Operation
- In the Linear Mode the Input Common-Mode, Voltage Range Includes Ground and the Output Voltage
- Can Swing to Ground, Even Though Operated from Only a Single Power Supply Voltage
- Unity Gain Cross Frequency is Temperature Compensated
- Input Bias Current is Also Temperature Compensated

2 Applications

- Transducer Amplifiers
- DC Gain Blocks
- Conventional Op Amp Circuits

3 Description

The LM124-N series consists of four independent, high-gain, internally frequency compensated operational amplifiers designed to operate from a single power supply over a wide range of voltages. Operation from split-power supplies is also possible and the low-power supply current drain is independent of the magnitude of the power supply voltage.

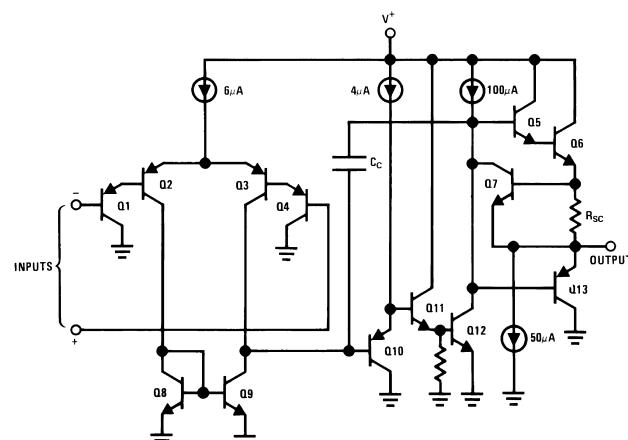
Application areas include transducer amplifiers, DC gain blocks and all the conventional op amp circuits which now can be more easily implemented in single power supply systems. For example, the LM124-N series can directly operate off of the standard 5-V power supply voltage which is used in digital systems and easily provides the required interface electronics without requiring the additional ± 15 V power supplies.

Device Information⁽¹⁾

PART NUMBER	PACKAGE	BODY SIZE (NOM)
LM124-N	CDIP (14)	19.56 mm \times 6.67 mm
LM224-N		
LM324-N	CDIP (14)	19.56 mm \times 6.67 mm
	PDIP (14)	19.177 mm \times 6.35 mm
	SOIC (14)	8.65 mm \times 3.91 mm
	TSSOP (14)	5.00 mm \times 4.40 mm
LM2902-N	PDIP (14)	19.177 mm \times 6.35 mm
	SOIC (14)	8.65 mm \times 3.91 mm
	TSSOP (14)	5.00 mm \times 4.40 mm

(1) For all available packages, see the orderable addendum at the end of the datasheet.

Schematic Diagram



An IMPORTANT NOTICE at the end of this data sheet addresses availability, warranty, changes, use in safety-critical applications, intellectual property matters and other important disclaimers. PRODUCTION DATA.

Table of Contents

1	Features	1
2	Applications	1
3	Description	1
4	Revision History.....	2
5	Pin Configuration and Functions	3
6	Specifications.....	4
6.1	Absolute Maximum Ratings	4
6.2	ESD Ratings.....	4
6.3	Recommended Operating Conditions	4
6.4	Thermal Information.....	5
6.5	Electrical Characteristics: LM124A/224A/324A	5
6.6	Electrical Characteristics: LM124-N/224-N/324-N/2902-N	6
6.7	Typical Characteristics	8
7	Detailed Description	11
7.1	Overview	11
7.2	Functional Block Diagram	11
7.3	Feature Description.....	11
7.4	Device Functional Modes.....	11
8	Application and Implementation	13
8.1	Application Information.....	13
8.2	Typical Applications	13
9	Power Supply Recommendations	23
10	Layout.....	23
10.1	Layout Guidelines	23
10.2	Layout Example	23
11	Device and Documentation Support	24
11.1	Related Links	24
11.2	Trademarks	24
11.3	Electrostatic Discharge Caution	24
11.4	Glossary	24
12	Mechanical, Packaging, and Orderable Information	24

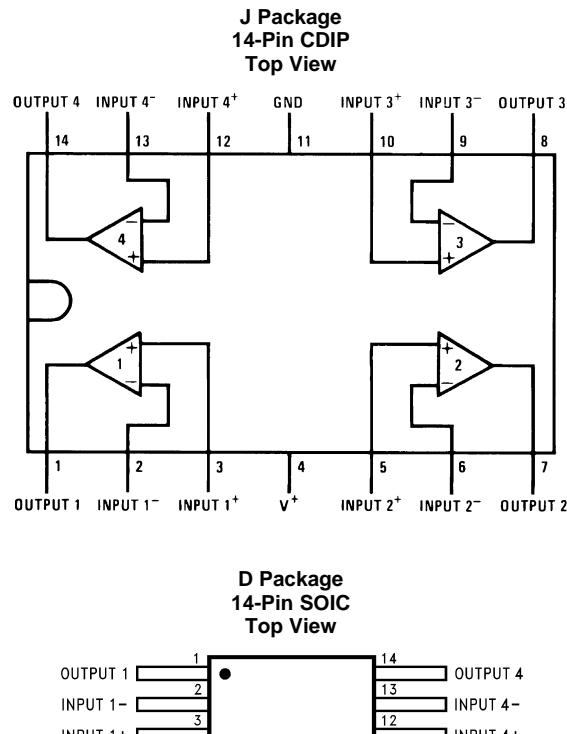
4 Revision History

Changes from Revision C (November 2012) to Revision D

Page

- Added *Pin Configuration and Functions* section, *ESD Ratings* table, *Feature Description* section, *Device Functional Modes*, *Application and Implementation* section, *Power Supply Recommendations* section, *Layout* section, *Device and Documentation Support* section, and *Mechanical, Packaging, and Orderable Information* section 1

5 Pin Configuration and Functions

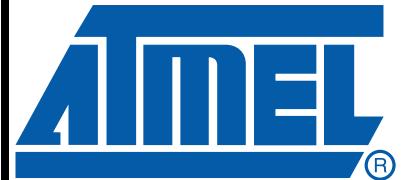


Pin Functions

PIN		TYPE	DESCRIPTION
NAME	NO.		
OUTPUT1	1	O	Output, Channel 1
INPUT1-	2	I	Inverting Input, Channel 1
INPUT1+	3	I	Noninverting Input, Channel 1
V+	4	P	Positive Supply Voltage
INPUT2+	5	I	Noninverting Input, Channel 2
INPUT2-	6	I	Inverting Input, Channel 2
OUTPUT2	7	O	Output, Channel 2
OUTPUT3	8	O	Output, Channel 3
INPUT3-	9	I	Inverting Input, Channel 3
INPUT3+	10	I	Noninverting Input, Channel 3
GND	11	P	Ground or Negative Supply Voltage
INPUT4+	12	I	Noninverting Input, Channel 4
INPUT4-	13	I	Inverting Input, Channel 4
OUTPUT4	14	O	Output, Channel 4

Features

- Core
 - ARM® Cortex®-M3 revision 2.0 running at up to 84 MHz
 - Memory Protection Unit (MPU)
 - Thumb®-2 instruction set
 - 24-bit SysTick Counter
 - Nested Vector Interrupt Controller
- Memories
 - From 256 to 512 Kbytes embedded Flash, 128-bit wide access, memory accelerator, dual bank
 - From 32 to 100 Kbytes embedded SRAM with dual banks
 - 16 Kbytes ROM with embedded bootloader routines (UART, USB) and IAP routines
 - Static Memory Controller (SMC): SRAM, NOR, NAND support. NAND Flash controller with 4-kbyte RAM buffer and ECC
- System
 - Embedded voltage regulator for single supply operation
 - POR, BOD and Watchdog for safe reset
 - Quartz or ceramic resonator oscillators: 3 to 20 MHz main and optional low power 32.768 kHz for RTC or device clock.
 - High precision 8/12 MHz factory trimmed internal RC oscillator with 4 MHz Default Frequency for fast device startup
 - Slow Clock Internal RC oscillator as permanent clock for device clock in low power mode
 - One PLL for device clock and one dedicated PLL for USB 2.0 High Speed Mini Host/Device
 - Temperature Sensor
 - Up to 17 peripheral DMA (PDC) channels and 6-channel central DMA plus dedicated DMA for High-Speed USB Mini Host/Device and Ethernet MAC
- Low Power Modes
 - Sleep and Backup modes, down to 2.5 µA in Backup mode.
 - Backup domain: VDDBU pin, RTC, eight 32-bit backup registers
 - Ultra Low-power RTC
- Peripherals
 - USB 2.0 Device/Mini Host: 480 Mbps, 4-kbyte FIFO, up to 10 bidirectional Endpoints, dedicated DMA
 - Up to 4 USARTs (ISO7816, IrDA®, Flow Control, SPI, Manchester and LIN support) and one UART
 - 2 TWI (I2C compatible), up to 6 SPIs, 1 SSC (I2S), 1 HSMCI (SDIO/SD/MMC) with up to 2 slots
 - 9-Channel 32-bit Timer/Counter (TC) for capture, compare and PWM mode, Quadrature Decoder Logic and 2-bit Gray Up/Down Counter for Stepper Motor
 - Up to 8-channel 16-bit PWM (PWMC) with Complementary Output, Fault Input, 12-bit Dead Time Generator Counter for Motor Control
 - 32-bit Real Time Timer (RTT) and RTC with calendar and alarm features
 - 16-channel 12-bit 1Msps ADC with differential input mode and programmable gain stage
 - One 2-channel 12-bit 1 MSPS DAC
 - One Ethernet MAC 10/100 (EMAC) with dedicated DMA
 - Two CAN Controller with eight Mailboxes
 - One True Random Number Generator (TRNG)
 - Write Protected Registers
- I/O
 - Up to 103 I/O lines with external interrupt capability (edge or level sensitivity), debouncing, glitch filtering and on-die Series Resistor Termination
 - Up to Six 32-bit Parallel Input/Outputs (PIO)
- Packages
 - 100-lead LQFP, 14 x 14 mm, pitch 0.5 mm
 - 100-ball LFBGA, 9 x 9 mm, pitch 0.8 mm
 - 144-lead LQFP, 20 x 20 mm, pitch 0.5 mm
 - 144-ball LFBGA, 10 x 10 mm, pitch 0.8 mm



AT91SAM ARM-based Flash MCU

SAM3X SAM3A Series



1. SAM3X/A Description

Atmel's SAM3X/A series is a member of a family of Flash microcontrollers based on the high performance 32-bit ARM Cortex-M3 RISC processor. It operates at a maximum speed of 84 MHz and features up to 512 Kbytes of Flash and up to 100 Kbytes of SRAM. The peripheral set includes a High Speed USB Host and Device port with embedded transceiver, an Ethernet MAC, 2x CANs, a High Speed MCI for SDIO/SD/MMC, an External Bus Interface with NAND Flash controller, 5x UARTs, 2x TWIs, 4x SPIs, as well as 1 PWM timer, 9x general-purpose 32-bit timers, an RTC, a 12-bit ADC and a 12-bit DAC.

The SAM3X/A series is ready for capacitive touch thanks to the QTouch library, offering an easy way to implement buttons, wheels and sliders.

The SAM3X/A architecture is specifically designed to sustain high speed data transfers. It includes a multi-layer bus matrix as well as multiple SRAM banks, PDC and DMA channels that enable it to run tasks in parallel and maximize data throughput.

It operates from 1.62V to 3.6V and is available in 100- and 144-pin QFP and LFBGA packages.

The SAM3X/A devices are particularly well suited for networking applications: industrial and home/building automation, gateways.

1.1 Configuration Summary

The SAM3X/A series devices differ in memory sizes, package and features list. [Table 1-1](#) below summarizes the configurations.

Table 1-1. Configuration Summary

Feature	SAM3X8E	SAM3X8C	SAM3X4E	SAM3X4C	SAM3A8C	SAM3A4C
Flash	2 x 256 Kbytes	2 x 256 Kbytes	2 x 128 Kbytes	2 x 128 Kbytes	2 x 256 Kbytes	2 x 128 Kbytes
SRAM	64 + 32 Kbytes	64 + 32 Kbytes	32 + 32 Kbytes	32 + 32 Kbytes	64 + 32 Kbytes	32 + 32 Kbytes
Nand Flash Controller (NFC)	Yes	-	Yes	-	-	-
NFC SRAM⁽¹⁾	4K bytes	-	4K bytes	-	-	-
Package	LQFP144 LFBGA144	LQFP100 LFBGA100	LQFP144 LFBGA144	LQFP100 LFBGA100	LQFP100 LFBGA100	LQFP100 LFBGA100
Number of PIOs	103	63	103	63	63	63
SHDN Pin	Yes	No	Yes	No	No	No
EMAC	MII/RMII	RMII	MII/RMII	RMII	-	-
External Bus Interface	16-bit data, 8 chip selects, 23-bit address	-	16-bit data, 8 chip selects, 23-bit address	-	-	-
SDRAM Controller	-	-	-	-	-	-
Central DMA	6	4	6	4	4	4
12-bit ADC	16 ch. ⁽²⁾	16 ch. ⁽²⁾	16 ch. ⁽²⁾	16 ch. ⁽²⁾	16 ch. ⁽²⁾	16 ch. ⁽²⁾
12-bit DAC	2 ch.	2 ch.	2 ch.	2 ch.	2 ch.	2 ch.
32-bit Timer	9 ⁽⁴⁾	9 ⁽⁵⁾	9 ⁽⁴⁾	9 ⁽⁵⁾	9 ⁽⁴⁾	9 ⁽⁴⁾
PDC Channels	17	15	17	15	15	15

Table 1-1. Configuration Summary (Continued)

Feature	SAM3X8E	SAM3X8C	SAM3X4E	SAM3X4C	SAM3A8C	SAM3A4C
USART/ UART	3/2 ⁽⁶⁾	3/1	3/2 ⁽⁶⁾	3/1	3/1	3/1
SPI⁽³⁾	1/4 + 3	1/4 + 3	1/4 + 3	1/4 + 3	1/4 + 3	1/4 + 3
HSMCI	1 slot 8 bits	1 slot 4 bits	1 slot 8 bits	1 slot 4 bits	1 slot 4 bits	1 slot 4 bits

- Notes:
1. 4 Kbytes RAM buffer of the NAND Flash Controller (NFC) which can be used by the core if not used by the NFC
 2. One channel is reserved for internal temperature sensor
 3. $2 / 8 + 4 = \text{Number of SPI Controllers} / \text{Number of Chip Selects} + \text{Number of USART with SPI Mode}$
 4. 6 TC channels are accessible through PIO
 5. 3 TC channels are accessible through PIO
 6. USART3 in UART mode (RXD3 and TXD3 available)

Note: The SAM3X-EK evaluation kit for the SAM3X and SAM3A series is mounted with a SAM3X8H in an LFBGA217 package. This device is not commercially available.

Figure 2-3. SAM3X4/8E (144 pins) Block Diagram

