

Bachelor-Thesis

Entwicklung eines Laborversuchs "schwebender Ball" auf Basis einer
Time of Flight Abstandsmessung

Eingereicht am: 27. Oktober 2016

von: Patrick Siemsen
geboren am 16. August 1991
in Kaltenkirchen

Bachelor-Thesis

für Herrn Patrick Siemsen

Entwicklung eines Laborversuchs „schwebender Ball“ auf Basis einer Time of Flight
Abstandsmessung

Development of a laboratory experiment „levitating ball“ on base of Time of Flight
measurement

Im Rahmen der Bachelorarbeit soll ein Laborversuch „schwebender Ball“ für die Module Embedded Control Systems und Regelungstechnik entwickelt werden. Ziel des Praktikumsversuches ist es, einen Ball mit Hilfe eines Lüfters auf eine definierte Höhe anzuheben. Für die Messung der Ballposition wird ein Abstandssensor nach dem Time of Flight Prinzip verwendet.

Die mechanischen Komponenten des Versuchsaufbaus müssen mittels CAD Software konstruiert und mit 3D-Drucktechnik gefertigt werden. Die Ansteuerung des Lüfters sowie des Sensors soll mit einem STM32 Microcontroller, unter Verwendung des mbed-Simulink Targets, realisiert werden. Aufgrund unterschiedlicher Anforderungen der Module sind analoge und digitale Schnittstellen zur Ansteuerung des Versuchs bereitzustellen. Für den Funktionsnachweis ist eine Positionsregelung zu implementieren.

Teilaufgaben:

- Literaturrecherche
- Konstruktion der mechanischen Komponenten des Versuchs
- Auswahl eines geeigneten Lüfters
- Einbindung der Sensorschnittstelle in die Programmierumgebung (mbed/Simulink)
- Bereitstellen von analogen und digitalen Schnittstellen zur Steuerung/Regelung
- Entwurf und Implementierung einer Positionsregelung
- Nachweis der Funktion
- Dokumentation

Tag der Ausgabe: 04.08.16
Tag der Abgabe : 27.10.16

i.v. M. Lochmann
Prof. Dr.-Ing. habil. Lochmann
Vorsitzender des
Prüfungsausschusses


Prof. Dr.-Ing. habil. Simanski
Betreuer HS Wismar

Zusammenfassung

Um den Studenten microcontrollergestützte Regelungen näher zu bringen, wurde in dieser Arbeit ein Versuchsaufbau für verschiedene Module des Studium entworfen. Dabei wurde die komplette Konstruktion, die Softwaregrundlage sowie die Erprobung im Rahmen dieser Arbeit durchgeführt. Die Position eines Tischtennisballs, der in einem Kunststoffrohr schwebt, wird geregelt. Ein auf dem Rohr aufgesteckter Lüfter saugt die Luft nach oben aus dem Rohr. Konstruktionsbedingt kann Luft von unten in das Rohr strömen. Ein im Aufbau untergebrachter Sensor misst die Position des Balls im Rohr.

Die Konstruktion wurde mit einem 3D-computer-aided design (CAD)-Programm entworfen und ein Prototyp im 3D-Druckverfahren erstellt. Da es sich um einen microcontrollerbasierten Versuchsaufbau handelt wurde für das Microcontrollerboard eine Basisplatine entworfen. Die Platine übernimmt verschiedene Funktionen im Versuchsaufbau (zum Beispiel Signalwandlung und Spannungserzeugung) und bietet die Möglichkeit das Board aufzustecken. Somit kann die gesamte Elektronik im Aufbau untergebracht werden. Zudem werden analoge sowie digitale Signale für die verschiedenen Anwendungen des Versuchs bereitgestellt. Für die Regelung wurde ein Tachosignal des verbauten PC-Lüfters benötigt. Daher wurde der Lüfter auf Basis ein Halleffektsensors um ein solches Signal erweitert. Um eine einfache Programmierung zu ermöglichen wurde ein Block für die grafische mbed Simulink-Programmierumgebung erstellt. Dieser Block liest den Messwert des verbauten Time of Flight (ToF)-Sensor zur Positionsbestimmung aus. Auf Grundlage des erstellten Blocks können Regelungen für den Modellversuch entwickelt werden. Um die verschiedenen Funktionen des Versuchs nachzuweisen wurde ein Positionsregler entworfen.

Abstract

In this thesis, an experiment setup was developed to teach microcontroller based control. For this the complete construction, the software basics and the testing was realised. The position of a ball, that levitate in a pipe, should be regulated. A fan placed on the pipe draw the air out of the pipe. Based on the contruction air can flow into it from below. A sensor placed in the construction measures the position of the ball.

The construction is made with a 3D-CAD-programm and a prototype is produced with 3d-printing. A base circuit board was designed for the used microcontroller-board. This circiut board takes several tasks at the experimental setup (for example signal transformation or power supply). The microcontrollerboard can be placed on the circuit board. So it is possible to place the complete electronic into the setup. In addition analogue and digital signals are needed for the different tasks. For the control a tacho signal of the used fan was needed. Therefore the fan was extended by a tacho signal on base of a Hall effect sensor. A block for the grafic mbed development enviroment Simulink is designed to offer an easy programming.

This block reads the measurement of the used ToF-sensor for positioning. On base of the created block it is possible to develope control for the experiment setup. The function is proofed by a position regulator.

Inhaltsverzeichnis

1 Einleitung	6
2 Theoretische Grundlagen	8
2.1 Physikalische Grundlagen	8
2.2 Positionsmessung	9
2.2.1 Möglichkeiten der Positionsmessung	10
2.2.2 Time of Flight Abstandsmessung	11
3 Hardware	13
3.1 Mechanische Konstruktion	13
3.1.1 Anforderungen an die Konstruktion	13
3.1.2 3D Konstruktion mit Solid Works	14
3.1.3 3D Druck	14
3.2 Elektrische Konstruktion	15
3.2.1 Lüfterauswahl und -anpassung	15
3.2.2 Auswahl eines geeigneten Lüfters	15
3.2.3 Drehzahlmessung mit Hallssensor TLE4905L	17
3.2.4 STM32L4-Nucleoboard	19
3.2.5 Basisplatine für STM32L4-Nucleoboard	20
4 Software	24
4.1 Programmierung	24
4.1.1 Einführung mbed Simulink-Target	24
4.1.2 Simulink-Block für ToF Abstandssensor VL6180X	26
4.2 Regelung	35
4.2.1 Einstellverfahren für Regler	35
4.2.2 Regelung für den Laborversuch	38
5 Schlussbemerkungen und Ausblick	43
Literaturverzeichnis	45
Abbildungsverzeichnis	47
Tabellenverzeichnis	48
A Abkürzungsverzeichnis	49
Selbstständigkeitserklärung	50

1 Einleitung

An der Hochschule Wismar ist neben dem theoretischen auch der praktische Aspekt des Studiums von großer Bedeutung. So besteht ein Bedarf an Versuchsaufbauten zur praktischen Ausbildung im Bereich der microcontrollergestützten Regelungen. Aufgrund dessen soll im Rahmen dieser Arbeit ein solcher Versuchsaufbau entworfen, konstruiert und programmiert werden. Bei dem Aufbau handelt es sich um den Laborversuch „schwebender Ball“, für welchen später microcontrollerbasierte Regelungen entworfen werden sollen. In Abbildung 1.1 wird der Versuchsaufbau schematisch dargestellt.

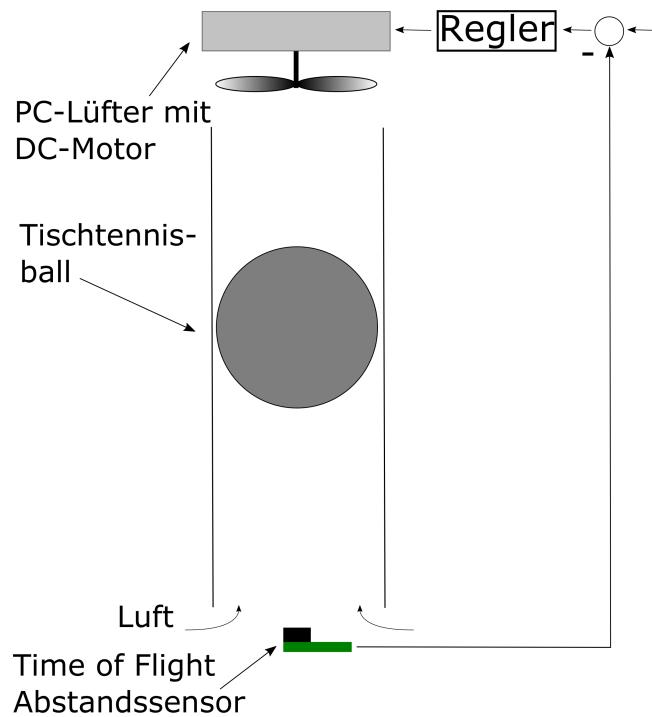


Bild 1.1: Versuchsaufbau

Der Tischtennisball befindet sich in einem durchsichtigen Kunststoffrohr. Ein PC-Lüfter ist mit einem im 3D-Druckverfahren erstellten Halter oben auf dem Rohr platziert und saugt die Luft aus dem Rohr. Unten steckt das Rohr in einem ebenfalls 3D-gedruckten Fuß. Durch diesen strömt Luft in das Rohr. Ein im Fuß untergebrachter ToF-Abstandssensor misst den Abstand zum Ball. Ebenfalls im Fuß untergebracht ist ein Microcontrollerboard auf ARM-Basis, auf dem eine Regelung läuft. Per Pulsweitenmodulation (PWM) steuert der Controller die Drehzahl des PC-Lüfters. Über die Inter-Integrated Circuit (I2C)-Schnittstelle wird der aktuelle Messwert des Sensors vom Controller ausgelesen.

Die Konstruktion soll komplett im 3D-Druckverfahren erstellt werden. Dieser Umstand muss bei der Konstruktion des Aufbau beachtet werden, da das Druckverfahren an bestimmte Bedingungen geknüpft ist. Zudem soll ein sicherer und anschaulicher Versuchsaufbau und -ablauf gewährleistet werden.

Der ausgewählte Lüfter muss stark genug ausgelegt werden um die Regelungsaufgabe zuverlässig zu bewältigen. Um weitere Hardwarekomponenten einzusparen, sollte der Lüfter zudem über eine Drehzahlsteuerung mittels PWM verfügen. Der zu entwerfende Simulinkblock, welcher zum Auslesen des Sensors dienen wird, soll über eine frei programmierbare Busadresse verfügen. Zudem soll der verwendete General Purpose Input/Output (GPIO) zum Initialisieren frei konfigurierbar sein.

Um den Aufbau in den Modulen Embedded Control Systems sowie Regelungstechnik nutzen zu können ist es notwendig analoge sowie digitale Schnittstellen bereitzustellen. Als digitale Schnittstelle dient die auf dem verwendeten Microcontrollerboard verbaute MicroUSB-Schnittstelle. Die analogen Schnittstellen sollen mithilfe des vom Controller zur Verfügung gestellten digital-to-analog converter (DAC) und analog-to-digital converter (ADC) realisiert werden. Mit den Schnittstellen soll ein analoges Tachosignal und der aktuelle Messwert des Sensors ausgegeben werden. Ein Sollwert für die Regelung soll ebenfalls vorgegeben werden können. Voraussetzung für die Nutzung der Schnittstellen ist eine entsprechende Programmierung des Controllers mit Simulink. Um die Funktion des Laborversuchs nachzuweisen wird ein Regler auf dem Controller implementiert.

In Kapitel 2 der Arbeit werden zunächst die theoretischen Grundlagen zum Verständnis der Arbeit erläutert. Anschließend werden in Kapitel 3 die mechanischen und elektrischen Komponenten des Versuchsaufbaus und in Kapitel 4 die Entwicklung des Simulink-Blocks beschrieben. Kapitel 5 gibt einen Überblick über die Ergebnisse und einen Ausblick über mögliche weitere Arbeiten.

2 Theoretische Grundlagen

2.1 Physikalische Grundlagen

Im folgenden Abschnitt werden die physikalischen Grundlagen zu dem in Abbildung 1.1 dargestellten Modellversuch näher beschrieben. Diese Formeln dienen dem Prozessverständnis. In Tabelle 2.1 werden die verwendeten Formelzeichen erläutert.

m	Ballmasse
g	Erdbeschleunigung
d_R	Röhrendurchmesser
d_B	Balldurchmesser
A_{Sp}	Luftspaltfläche
A_B	Ballfläche
K_L	Proportionalitätsfaktor Luftwiderstand
K_v	Proportionalitätsfaktor Gebläse

Tabelle 2.1: Formelzeichen

Die Summe aus der nach oben wirkenden Luftwiderstandskraft F_L und der nach unten wirkenden Gewichtskraft mg ist gleich dem Produkt aus Ballmasse und Beschleunigung $m\ddot{x}$.

$$F_L - mg = m\ddot{x} \quad (2.1)$$

Die Luftwiderstandskraft F_L ist proportional zum Quadrat der Strömungsgeschwindigkeit v_{Sp}^2 im Luftspalt zwischen Tischtennisball und Röhre.

$$F_L = K_L \cdot v_{Sp}^2 p \quad (2.2)$$

Die Strömungsgeschwindigkeit im Luftspalt berechnet sich aus dem Luftvolumenstrom \dot{V} und der Spaltfläche A_{Sp} , muss aber noch um die Bewegung des Balls korrigiert werden.

$$F_L = K_L \cdot \left(\frac{\dot{V} - A_B \cdot \dot{x}}{A_{Sp}} \right)^2 \quad (2.3)$$

Sinkt der Ball nach unten wird die Strömungsgeschwindigkeit im Luftspalt und damit der Luftwiderstand größer.

Der Luftvolumenstrom ist proportional der Lüfterdrehzahl n .

$$\dot{V} = K_V \cdot n \quad (2.4)$$

Damit ergibt sich für die Luftwiderstandskraft

$$F_L = K_L \cdot \left(\frac{K_V \cdot n - A_B \cdot \dot{x}}{A_{Sp}} \right)^2 \quad (2.5)$$

Die Luftwiderstandskraft F_L ist dabei die Kraft, die der Ball beim Aufsteigen im Rohr überwinden muss. Wie Formel 2.5 zeigt, handelt es sich um einen nicht linearen Zusammenhang. [1, 2]

2.2 Positionsmessung

Im Folgenden wird die verwendete Möglichkeit zur Positionsmessung näher beschrieben sowie zwei alternative Methoden vorgestellt. Dabei werden die Vor - und Nachteile im Bezug auf die Anwendung bewertet.

2.2.1 Möglichkeiten der Positionsmessung

Lasertriangulation

Die Abstandsmessung mithilfe der Lasertriangulation beruht auf einfachen geometrischen Beziehungen. Eine Laserdiode emittiert einen Laserstrahl der auf das Messobjekt gerichtet ist. Ein daneben befindlicher Sensor beobachtet den Strahl. Ändert sich der Abstand ändert sich der Winkel des reflektierten Strahls. Somit kann über die geometrischen Beziehungen der Abstand bestimmt werden. Abbildung 2.1 verdeutlicht die Funktion.

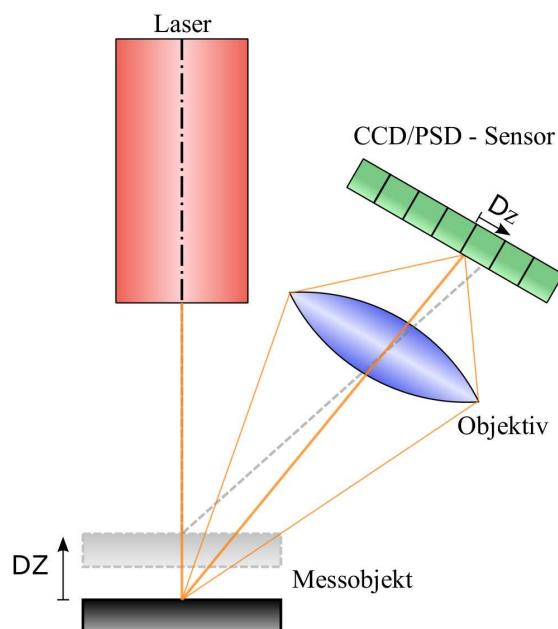


Bild 2.1: Lasertriangulation [3]

Je nach Bauart der Sensoren erlauben diese Messdistanzen von mehr als einem Meter. Durch die Verwendung eines Lasers als Messmittel bleibt der Messpunkt trotzdem klein genug. Somit wäre dieses Messverfahren eine Alternative zum eingesetzten ToF-Messverfahren.[4]

Ultraschall

Eine weitere Möglichkeit der Abstandsmessung ist die Messung per Ultraschall. Zur Messung wird ein Ultraschallimpuls ausgesendet, der vom zu messenden Objekt zurückgeworfen wird. Die Zeitdauer vom Aussenden des Impuls bis zum Eintreffen des Echoes ist direkt proportional zum Abstand.

Da die Abstandsmessung auf eine Zeitmessung zurückgeführt wird, handelt es sich um ein Laufzeitverfahren. Von Ultraschall wird in einem Frequenzbereich zwischen 20 Kilohertz (kHz) und einem Gigahertz (GHz) gesprochen. Ultraschallwellen unterliegen einer Streuung. Daher ist dieses Verfahren nur bedingt für den Einsatz in einem geschlossenen Rohr geeignet. Das Messverfahren erlaubt Messdistanzen größer einem Meter. [5, 6, 7]

2.2.2 Time of Flight Abstandsmessung

Bei dem in dieser Arbeit verwendeten ToF-Abstandsmessverfahren handelt es sich um ein optisches Laufzeitmessverfahren. Mithilfe dieses Verfahrens kann der Abstand zwischen einem entsprechendem Sensor und einem Messobjekt ermittelt werden. Um den Abstand zu einem Objekt zu ermitteln, wird ein zeitlicher Lichtpuls ausgesandt. Dieser Lichtpuls wird von dem Objekt reflektiert, und von einer im Sensor verbauten Photozelle wieder aufgefangen. Durch die Konstanz der Lichtgeschwindigkeit ergibt sich aus der für die Strecke benötigten Laufzeit direkt der Abstand zwischen Sensor und Objekt. Abbildung 2.2 stellt schematisch eine ToF-Messung dar.

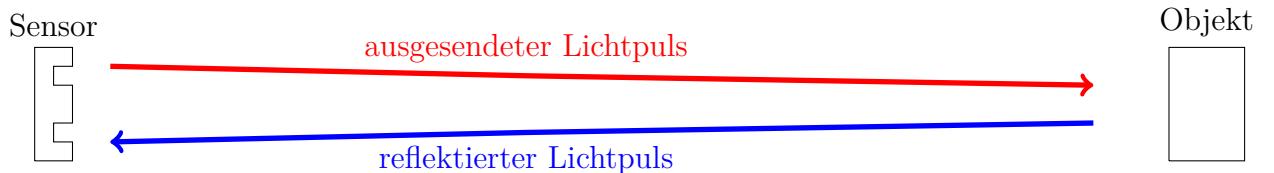


Bild 2.2: schematische Darstellung ToF-Messung

$$l = \frac{c\Delta t}{n} * 0,5 \quad (2.6)$$

Mithilfe der Formel 2.6 kann der Abstand aus dem Laufzeitmessverfahren ermittelt werden. Der Parameter l ist der ermittelte Abstand zwischen Sensor und Objekt. Die Konstante c entspricht der Lichtgeschwindigkeit. Diese beträgt $\sim 299,792 \frac{m}{s}$. Die Laufzeit, die der Lichtimpuls vom Sensor zu dem Objekt und zurück braucht, wird mit Δt beschrieben. Da der Lichtimpuls die Strecke doppelt zurücklegt (Hin- und Rückweg) wird mit dem Faktor 0,5 auf die einfache Strecke umgerechnet. Der Parameter n ist der Brechungsindex. An der Grenzfläche zwischen zwei Medien wird Licht gebrochen und reflektiert.

Um diesen Umstand zu Berücksichtigen wird der Brechungsindex genutzt. Tabelle 2.2 gibt Beispiele für typische Werte.

Medium	Brechungsindex
Vakuum	1
Luft (Bodennah)	1,000292
Wasser	1,33

Tabelle 2.2: Brechungsindex [8]

Der verwendete ToF-Sensor bietet eine maximale Messdistanz von 200 mm. Alternative Sensoren bieten jedoch auch die Möglichkeit Strecken von bis zu zwei Meter zu messen. Das ToF-Messverfahren ermöglicht mit einem geeigneten Sensor eine ausreichend große Messweite für die Anwendung in diesem Laborversuch. Zudem ist die Ansteuerung des Sensors leicht zu implementieren. Durch die Verwendung eines Lasers als Messmittel ist der Sensor empfindlich gegen seitlich einstrahlendes Licht. Dieser Umstand stellt in dieser Anwendung jedoch kein Problem dar.

Weitere Anwendungen finden ToF-Sensoren in ToF-Kameras. Diese stellen mithilfe des Messverfahrens dreidimensionale Bilder dar. [9, 10]

3 Hardware

Im folgenden Kapitel wird die mechanische und elektrische Konstruktion des Versuchsaufbaus erläutert. Zudem wird die verwendete Hardware, der ausgewählte Lüfter und dessen Anpassung an die Versuchsbedingungen beschrieben.

3.1 Mechanische Konstruktion

Die Konstruktion wurde mit dem 3D-CAD-Programm Solid Works 2015 durchgeführt. Die Fertigung des Modells erfolgte mithilfe eines 3D-Druckers, der an der Hochschule Wismar zur Verfügung steht.

3.1.1 Anforderungen an die Konstruktion

Hauptaugenmerk bei der Konstruktion der einzelnen Teile des Laborversuchs lag auf einem möglichst sicheren Betrieb und einer anschaulichen Versuchsdurchführung. Aufgrund dieser Anforderungen wurde ein Fuß entworfen, in dem sämtliche elektronischen Komponenten untergebracht werden können. So wird der verwendete ToF-Sensor in eine Halterung mittig im Fuß befestigt. Auf diese Weise kann der Sensor ohne Probleme von unten die Position des Balls messen. Um den Sensor leicht tauschen zu können, wurde dieser mit einer Klemmhalterung befestigt. Des Weiteren kann die in dieser Arbeit erstellte Platine, die zum Versuchsaufbau gehört, in einer passenden Halterung im Fuß festgeschraubt werden. Der Tischtennisball befindet sich in einem durchsichtigem Plexiglassrohr, welches fest mit dem Fuß verbunden ist. Auf der Oberseite des Rohrs ist ein Halter aufgesteckt, an dem der verwendete Lüfter angeschraubt werden kann. Da der Versuchsaufbau in der Lehre eingesetzt werden soll, ist es notwendig, alle zur erfolgreichen Programmierung des Modells benötigten Schnittstellen von außen zugänglich zu machen. Deshalb sind alle digitalen und analogen Schnittstellen am Fuß des Versuchs in Form von 4 mm Buchsen herausgeführt. Die Spannungsversorgung erfolgt ebenfalls über zwei Anschlüsse am Fuß.

3.1.2 3D Konstruktion mit Solid Works

Das 3D-CAD-Programm Solid Works 2015 bietet die Möglichkeit, rechnergestützt dreidimensionale Modelle in beliebiger Komplexität zu erstellen. Unter Zuhilfenahme verschiedener Features wird aus einer zuvor gezeichneten zweidimensionale Skizze ein dreidimensionaler Körper erstellt und entsprechend ausgeformt. Somit konnten alle Anforderungen an die einzelnen Bauteile des Versuchsaufbaus berücksichtigt werden. Auch die im folgenden Abschnitt beschriebenen Anforderungen an das eingesetzte 3D-Druckverfahren konnten berücksichtigt werden. Abbildung 3.1 zeigt ein Renderbild des erstellten Modells.



Bild 3.1: 3D-Modell

3.1.3 3D Druck

Um die Komponenten des Versuchsaufbaus herzustellen wurde das 3D-Druckverfahren angewendet. An der Hochschule Wismar stand zum Zeitpunkt der Erstellung dieser Arbeit ein solcher Drucker vom Typ Ultimaker 2 zur Verfügung. Ein 3D-Drucker bietet gegenüber der konventionellen Werkstoffbearbeitung diverse Vorteile. Prototypen können mit dem 3D-Druck innerhalb kürzester Zeit gefertigt werden, wohin gegen konventionelle Methoden oft bedeutend zeit- und kostenintensiver sind. So mit war diese Fertigungsmethode optimal geeignet um den Prototypen dieses Versuchsaufbaus zu fertigen.

Funktionsweise

Das 3D-Druckverfahren ermöglicht es, dreidimensionale Werkstücke schichtweise aufzubauen. Als Druckmaterial werden unter anderem Kunststoffe verwendet (hier ABS). Mithilfe eines Fördersystems wird das Material einer beheizten Düse zugeführt. Das geschmolzene Material wird durch die Düse schichtweise auf einem beheizten Druckbett aufgetragen. Dabei wird der Druckkopf durch drei Achsen bewegt. Diese Achsen werden anhand des G-Codes durch einen Controller gesteuert. Der G-Code wird vor dem Druck aus dem 3D-CAD-Modell erzeugt. Aufgrund des schichtweisen Ausbaus eines Druckmodells ist es nicht möglich, große Überhänge zu drucken. Daher musste das Modell in mehrere Teile zerschnitten werden, um einen fehlerfreien Druck zu ermöglichen.

3.2 Elektrische Konstruktion

In diesem Abschnitt wird die in dieser Arbeit verwendete Hardware und die erstellten Schaltungen näher erläutert. Zu Beginn wird der verwendete Lüfter vorgestellt sowie die Nachrüstung eines Tachosignals erläutert. Anschließend wird das verwendete Microcontrollerboard sowie die Basisplatine für das Board vorgestellt.

3.2.1 Lüfterauswahl und -anpassung

Im Folgenden wird die Lüfterauswahl und die Nachrüstung eines geeigneten Tachosignals für die Regelung erläutert.

3.2.2 Auswahl eines geeigneten Lüfters

Für den Laborversuch wurde ein geeigneter, ausreichend starker PC-Lüfter benötigt. Nach ersten Versuchen mit verschiedenen Lüftern in der Größe zwischen 60mm und 120mm zeigte sich, dass diese ungeeignet sind. Die Fördermenge dieser Lüfter ist nicht ausreichend um den Tischtennisball in dem Rohr anzuheben.

Aus diesem Grund fiel die Wahl auf einen Lüfter mit einem Durchmesser von 140 mm. Zunächst wurde ein Industrielüfter vom Typ NF-A14 industrialPPC-3000 PWM des Herstellers NOCTUA. In Tabelle 3.1 sind die Daten des Lüfters dargestellt.

Größe(mm)	140x140x25
Drehzahl(1/min)	3000
Fördermenge (m^3/h)	269,3

Tabelle 3.1: Daten NOCTUA NF-A14 industrialPPC-3000 PWM [11]

Erste Tests haben ergeben, dass sich der Lüfter per PWM in einem Drehzahlbereich von etwa 25% bis 100% steuern lässt. Jedoch ist die Förderleistung dieses Lüfters zu hoch. Selbst bei 25% der maximalen Drehzahl ist der Tischtennisball nicht in einer Position zu halten. Auch eine Reduzierung des Querschnitts des Rohrs bewirkt keine Besserung. Aufgrund dessen wurde entschieden einen schwächeren Lüfter einzusetzen. Die Wahl fiel auf einen NF-A14 PWM des Herstellers NOCTUA. Tabelle 3.2 stellt die Betriebsdaten des Lüfters dar.

Größe(mm)	140x140x25
Drehzahl(1/min)	1500
Fördermenge (m^3/h)	140,2

Tabelle 3.2: Daten NOCTUA NF-A14 PWM [11]

Dieser Lüfter ist mit halbierter Drehzahl und geringerer Fördermenge gegenüber dem NOCTUA NF-A14 industrialPPC-3000 PWM deutlich besser für den Versuchsaufbau geeignet. Der Lüfter lässt sich in einem Drehzahlbereich von 15% bis 100% steuern. So ist es möglich, den Tischtennisball auf einer bestimmten Position zu halten, ohne den Querschnitt des Rohr zu reduzieren oder andere Maßnahmen zu treffen.

3.2.3 Drehzahlmessung mit Hallsensor TLE4905L

Der verwendete Lüfter ist mit einem vierpoligen Anschluss ausgestattet. Der Anschluss ist wie in Abbildung 3.2 dargestellt belegt.

GND ist der Masseanschluss und an V+ werden 12 V als Versorgungsspannung für den Lüfter angelegt. Der Tachoanschluss stellt ein Tachosignal bereit, mit dem die aktuelle Drehzahl des Lüfters ermittelt werden kann. An dem PWM-Anschluss kann ein PWM-Signal eingespeist werden. Mit diesem Signal lässt sich die Drehzahl des Lüfters steuern. Dabei wird nicht die Betriebsspannung des Lüfters variiert. Diese bleibt konstant bei 12 V. Mithilfe des PWM-Signals ist somit eine leistungslose Steuerung der Lüfterdrehzahl möglich. Eine Untersuchung des Tachosignals mithilfe eines Oszilloskops ergab, dass dieses stark verrauscht und so nicht nutzbar ist. Daher wurde der Lüfter um ein neues Tachosignal erweitert. Zur Erzeugung des Tachosignals wurde der unipolare Hallsensor TL4905L und ein entsprechender Magnet verwendet.

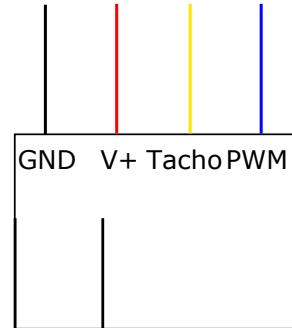


Bild 3.2: Stecker Lüfter

Hallsensor TLE4905L

Abbildung 3.3 zeigt die schematische Darstellung des verwendeten Hallsensors TLE4905L.

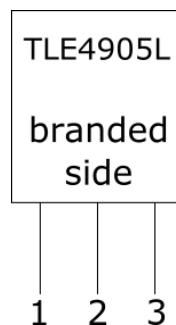


Bild 3.3: Hallsensor TLE4905L

Pin No.	Symbol	Function
1	V_s	Supply Voltage
2	GND	Ground
3	Q	Signal

Tabelle 3.3: Pinbelegung TLE4905L [12]

Der unipolare Hallsensor TLE4905L erzeugt an seinem Output Pin Q (siehe Abbildung 3.3) einen digitalen High-Pegel, wenn ein Magnetfeld mit der vorgegebenen Feldrichtung auftritt. Dabei muss die magnetische Turn-ON Induktion B_{OP} überschritten werden (vergleiche Tabelle 3.4). Dies ist immer der Fall, wenn der am Rotor angebrachte Magnet an dem, am Gehäuse des Lüfters angebrachten Sensor, vorbeikommt. Somit erzeugt der Sensor pro Umdrehung einen Rechteckimpuls. Mithilfe eines Timers des Microcontrollers können die Pulse des erzeugten Rechtecksignals gemessen werden. So kann zu jeder Zeit die aktuelle Lüfterdrehzahl ermittelt werden.

	Symbol	min	max	Einheit
Supply voltage	V_s	3,8	24	V
Turn-ON induction	B_{OP}	7	18	mT
Turn-OFF induction	B_{RP}	5	16	mT
Hysteresis	$B_{OP} - B_{RP}$	2	6,5	mT

Tabelle 3.4: Daten TLE4905L [12]

Um den Sensor verwenden zu können, ist eine Schaltung nach Abbildung 3.4 notwendig. Der Pullupwiderstand R_1 zieht den Pegel des Signals Q auf die Spannung V_s . Für den Widerstand wurde der interne Pullup-Widerstand des Controllers verwendet. Dadurch wird das Signal Q nicht auf die Versorgungsspannung V_s (hier 12 V) sondern auf die Controllerspannung (hier 3,3 V) gezogen. [12]

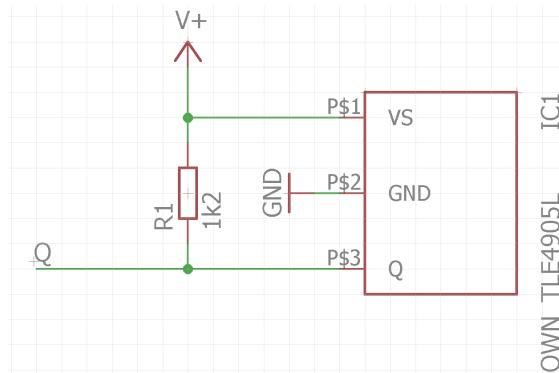
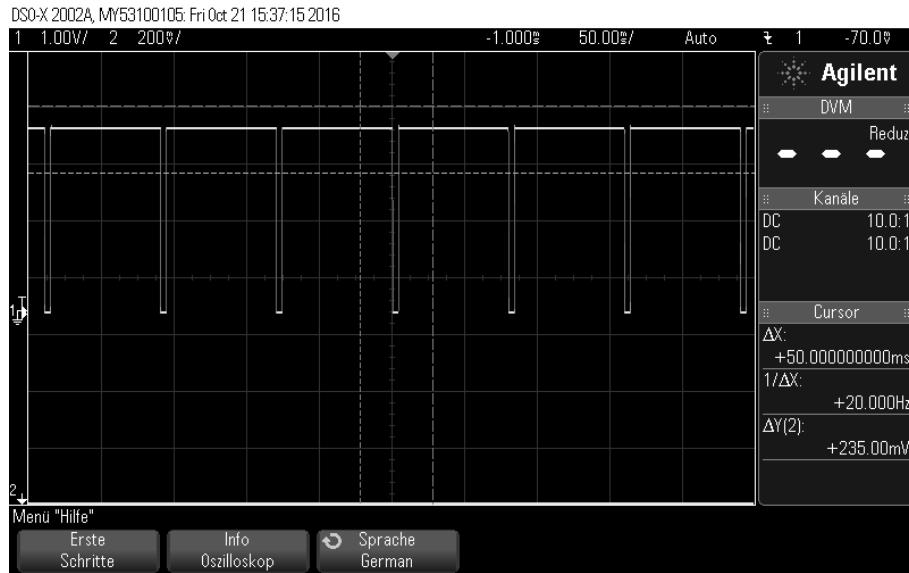


Bild 3.4: Schaltung TLE4905L

Abbildung 3.5 zeigt das Rechtecksignal, das durch die Schaltung bei etwa 750 Umdrehungen pro Minute (U/min) erzeugt wird.

**Bild 3.5:** erzeugtes Tachosignal

3.2.4 STM32L4-Nucleoboard

Zur Steuerung des Laborversuchs wird ein STMicroelectronics (STM)32L432 Nucleo-32 Board eingesetzt. Auf diesem Board ist ein STM32L432KC ARM Cortex-M4 Microcontroller verbaut. Er zeichnet sich durch seine kompakte Bauform aus, ist preiswert und ist bereits mit einem ST-Link Debugger-Programmieradapter mit einer MicroUSB-Schnittstelle ausgestattet. Alle für den Laborversuch benötigten Schnittstellen sind in Form von Pinleisten nach außen geführt, sodass diese leicht verwendet werden können. Der Controller läuft mit einem Takt von 80 Megahertz (MHz), verfügt über 256 Kilobyte (kB) Flash und 64 kB Static random-access memory (SRAM). Er ist unter anderem mit folgender Peripherie ausgestattet:

- 3x Serial Peripheral Interface (SPI)
- 2x I2C
- 3x Universal Synchronous/Asynchronous Receiver Transmitter (USART)
- 1x Controller Area Network (CAN)
- 1x 12-bit ADC mit 9 Channels
- 2x 12-bit DAC mit 3 Channels
- real-time clock (RTC)
- 26 GPIO's

Davon wurden für diese Arbeit eine I2C-Schnittstelle, ein ADC, ein DAC, mehrere GPIO's sowie die MicroUSB-Schnittstelle genutzt. [13]

Zunächst wurde ein STM32F303 Nucleo-32 Microcontrollerboard eingesetzt. Der auf diesem Board eingesetzter Controller läuft mit einem Takt von 72 MHz, besitzt 64 kB Flash und 16 kB SRAM. Da der zur Verfügung stehende Speicher von 64 kB nicht für den von Simulink erzeugten Code ausreichte, wurde auf das STM32LF432 Nucleo-32 Board gewechselt. [14] Da zum Zeitpunkt dieser Arbeit jedoch noch kein Simulink-Target für das STM32L432 Nucleo-32 Board zur Verfügung stand, wurde als Übergangslösung ein STM32F446 Nucleo-32 Board verwendet. Dieses wurde mit Steckkabeln mit dem Versuchsaufbau verbunden. Alle folgenden Messergebnisse wurden mit diesem Board aufgenommen. Das STM32L432 Board und das STM32F303 Board sind pinkompatibel sodass das neue Board ohne Probleme in den Praktikumsversuch passt. Sobald das Simulink-Target für STM32L432 Nucleo-32 Board zur Verfügung steht kann neuer Code für das Board durch die automatische Codegenerierung von Simulink erzeugt werden. Dafür müssen lediglich die verwendeten Pins des Controllers in dem Simulink-Modell angepasst werden.

3.2.5 Basisplatine für STM32L4-Nucleoboard

Da der Microcontroller dauerhaft im Fuß des Versuchsaufbaus untergebracht werden soll und die Verbindungen zu Sensor, Lüfter und Schnittstellen mit Jumperkabeln hergestellt werden sollen, wurde eine Basisplatine für das Microcontrollerboard erstellt. Auf diese Platine kann das Nucleoboard mit den Stifteleisten aufgesteckt werden. Die MicroUSB-Schnittstelle ist weiterhin zugänglich. In Abbildung 3.6 ist der mit Eagle 7.6.0 erstellte Schaltplan für die Basisplatine dargestellt. Die Platine erfüllt mehrere Funktionen, die im Folgenden erläutert werden.

Transistor T1

Der NPN Transistor T1 vom Typ BC846B wird in dieser Schaltung als Schalttransistor genutzt. Er wandelt das vom Microcontroller erzeugt PWM-Signal mit 3,3 V Pegel in eine PWM-Signal mit 5 V Pegel um. Diese Wandlung wird benötigt, damit der Lüfter sicher mit dem PWM-Signal drehzahlgesteuert werden kann. Allerdings ist zu beachten, dass durch diese Schaltung das Signal invertiert wird. Dies gilt es bei der späteren Regelung zu beachten. Abbildung 3.7 zeigt das gewandelte PWM-Signal bei 50% Aussteuerung.

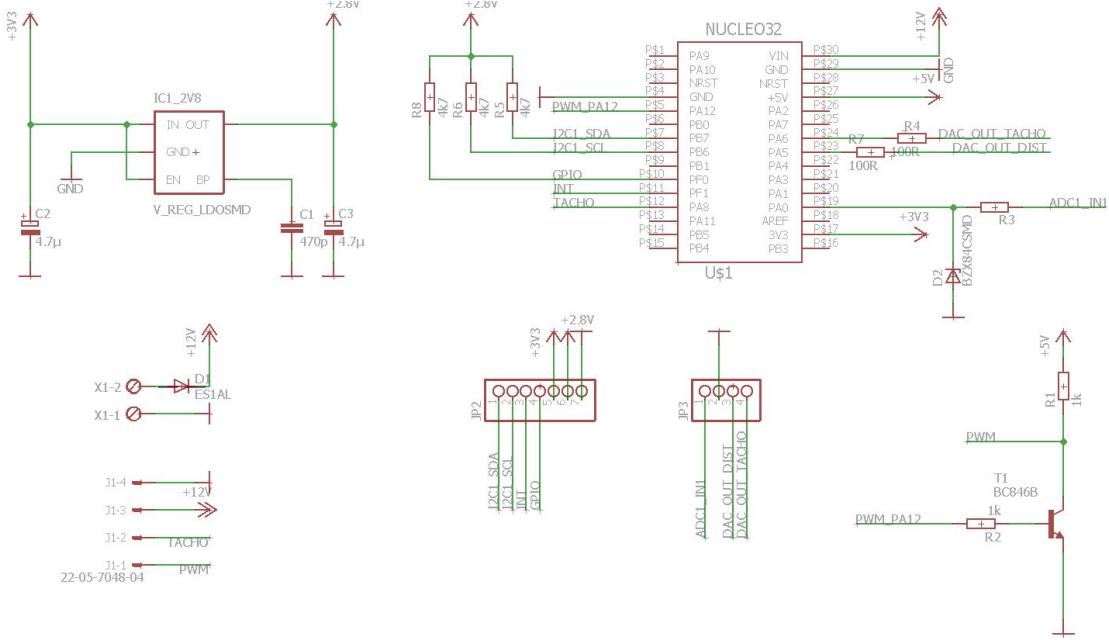


Bild 3.6: Schaltplan Basisplatine

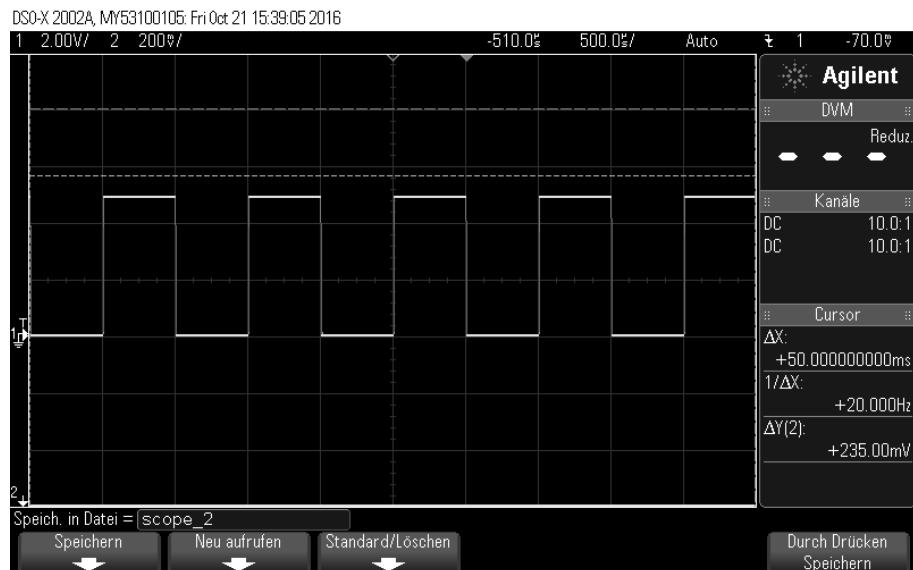


Bild 3.7: PWM Signal

Schutzbeschaltung

Um eine Zerstörung der Schaltung oder des Microcontrollerboards durch falsches Beschalten der äußeren Anschlüsse oder durch Überspannung zu verhindern, wurden einige Schutzmaßnahmen vorgesehen.

Verpolschutzdiode D1 Typ ES1AL

Die Verpolschutzdiode D1 (siehe Abbildung 3.6) ist auf der Platine in der 12 V Zuleitung, unmittelbar hinter der Einspeisung der Platine platziert. Dioden haben eine Durchlassrichtung und eine Sperrrichtung. Liegt an einer Diode eine Spannung größer als ihre Schwellspannung (hier 0,95 V) an, wird sie leitend. Wird eine Spannung in Sperrrichtung angelegt, sperrt die Diode bis zu ihrer Durchbruchspannung (hier 50 V). Ist die angelegte Spannung größer, wird die Diode zerstört. Wird an den äußeren Anschlüssen des Laborversuchs die 12 V Versorgungsspannung falsch herum angeschlossen, wird die Platine durch die Diode vor einer Zersetzung geschützt.

[15]

Überspannungsschutzdiode Typ BZX84C

Die Zener-Diode D2 (siehe Abbildung 3.6) vom Typ BZX84C dient in dieser Schaltung als Überspannungsschutzdiode. Der ADC muss vor Spannungen höher als 3,6V geschützt werden.

Strombegrenzungswiderstände

Die Widerstände R3,R4 und R7 (siehe Abbildung 3.6) dienen zur Strombegrenzung für die DAC und den ADC.

2,8 V Spannung

Der Integrierte Schaltkreis (IC) IC1_2V8 (siehe Abbildung 3.6) ist ein Step-Down Wandler. Dieser IC erzeugt aus der vom Controllerboard bereitgestellten Spannung (3,3 V) eine Spannung von 2,8 V. Diese wird für den I2C-Bus sowie den GPIO des ToF-Sensors benötigt.

Pullup-Widerstände

Der eingesetzte I2C-Bus und der benötigte GPIO sind als Open Drain Outputs konfiguriert. Mit einem Open Drain Output ist es möglich, Bauteile mit verschiedenen Spannungspegeln aber einer gemeinsamen Masseverbindung zu betreiben.

Dafür werden Pull-Up Wiederstände (R1 bis R3, siehe Abbildung 3.6) benötigt, die das Signal auf die gewünschte Spannung (hier 2,8 V) ziehen.

Spannungsversorgung

Das STM32L432 Nucleo-32 Board wird über die Basisplatine mit den angelegten 12 V an dem VIN-Pin versorgt. Somit ist es möglich den gesamten Laborversuch auch ohne angeschlossenes USB-Kabel zu nutzen. Um den ST-Link Debugger-Programmieradapter trotzdem zum Programmieren des Boards nutzen zu können ist es wichtig zunächst 12 V an die Basisplatine anzulegen und erst danach das MicroUSB-Kabel mit dem ST-Link zu verbinden.

Abbildung 3.8 zeigt den Prototypen des Versuchs.



Bild 3.8: Prototyp des Versuchsaufbaus

4 Software

In dem folgenden Kapitel wird die für diese Arbeit erstellte Software beschrieben. Dabei wird näher auf die Erstellung eines Simulink-Blocks sowie auf einige Möglichkeiten der Positionsregelung eingegangen.

4.1 Programmierung

Im folgendem Abschnitt wird die für den Laborversuch notwendige Programmierung näher erläutert. Zunächst wird die Erstellung des Simulinkblocks und seine Funktionen erläutert. Danach werden verschiedene Möglichkeiten der Reglereinstellung sowie der erstellte Regler für den Laborversuch vorgestellt.

4.1.1 Einführung mbed Simulink-Target

Ziel des Laborversuch ist es, den Studenten controllerbasierte Regelungen näher zu bringen. Daher soll auf eine Programmierung der Regelung in C oder einer vergleichbaren Programmiersprache verzichtet werden. Alternativ kommt Simulink, ein Zusatzprodukt zu Matlab, als grafische Programmierumgebung zum Einsatz. Matlab bietet die Möglichkeit, aus einem Simulinkmodell ANSI/ISO-konformen C-Code zu erzeugen. Diese Funktion wird für den Laborversuch genutzt. Um den erzeugten Quellcode auf den verwendeten Microcontrollern nutzen zu können, wurde an der Hochschule Wismar ein eigenes mbed Simulink-Target entwickelt. Mit diesem Target ist es möglich, nicht nur die allgemeinen Simulink-Blöcke, sondern auch speziell auf die Hardware abgestimmte Blöcke zu nutzen. So werden z. B. Blöcke für die verschiedene Peripherie der Controller (zum Beispiel ADC, DAC, GPIO, SPI, I2C, etc.), aber auch für externe Hardware (zum Beispiel diverse Sensoren) bereitgestellt. Matlab bietet mit den S-functions und dem Target Language Compiler (TLC) Werkzeuge an, um eigene Blöcke zu erstellen, aus denen Code für Microcontroller erzeugt werden soll. Um eigenen Code in Form von Blöcken in Simulink nutzen zu können werden die drei im Folgenden beschriebenen Komponenten benötigt.

Mask

In der Mask wird das Aussehen des Konfigurationsfensters für den Simulinkblock festgelegt. Sie wird mithilfe des Mask Editors, welcher in Abbildung 4.1 dargestellt wird, entworfen. Der Editor bietet die Möglichkeit Berechnungen durchzuführen (zum Beispiel um Adressen umzurechnen) oder mithilfe der MATLAB Programmiersprache andere Funktionen umzusetzen. Es müssen zudem die verwendeten Parameter festgelegt werden.

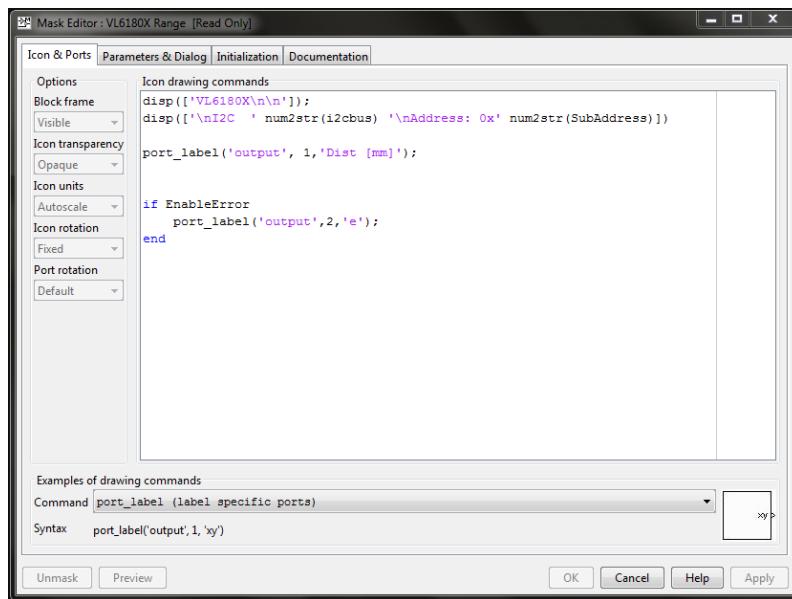


Bild 4.1: Mask Editor

S-function

Die S-function dient zur Konfiguration des Blocks. In ihr müssen unter anderem Informationen wie die Anzahl der Ein- und Ausgänge, die jeweiligen Datentypen, sowie die Namen der Parameter, die in der Mask festgelegt worden sind, eingetragen werden. Zu beachten ist, dass der Name der S-funtion genau so definiert wird, wie die Datei heißt. Damit die S-function genutzt werden kann muss sie nach dem Edtieren compiliert werden. Dies geschieht mithilfe des MEX-Befehls in MATLAB. Die compilierte Datei hat die Dateiendung .mexw64. Voraussetzung ist ein installierter Compiler. Dafür wurde in dieser Arbeit Visual Studio 2010 genutzt.

TLC

Das TLC-File wird genutzt, um den durch die Codegenerierung zu erzeugenden Code zu beschreiben. Dafür müssen die Source-Files des C/C++ - Quellcodes eingebunden. Es können Objekte, die für die Funktion benötigt werden erzeugt werden (z. B. DigitalIn etc.) und Funktionen, die im eingebundenen Source-Code programmiert sind, aufgerufen werden.

4.1.2 Simulink-Block für ToF Abstandssensor VL6180X

Um den ToF-Abstandssensor VL6180X für den Praktikumsversuch nutzen zu können, wurde im Rahmen dieser Arbeit ein Simulinkblock erstellt. Da in dem Simulinkblock C/C++ Quellcode eingebunden werden muss, wurde der Sensor erst mit einem klassischen mbed-Projekt für die Entwicklungsumgebung Keil uVision V5.21.1.0 getestet. Auf Basis des Codes wurde der Simulinkblock erstellt. Ein vollständiges Beispielprojekt für die Entwicklungsumgebung Keil uVision V5.21.1.0 befindet sich auf der beigefügten CD. Der für den Simulinkblock verwendete Code basiert auf angepasstem Quellcode, der auf mbed.org für ein Nucleo Expansion Board für den ToF-Sensor zur Verfügung stand. [16] Im Folgenden sollen die einzelnen Komponenten des Blocks (Mask, S-function und TLC-File) genauer erläutert werden. Abbildung 4.2 zeigt den erstellten Block.

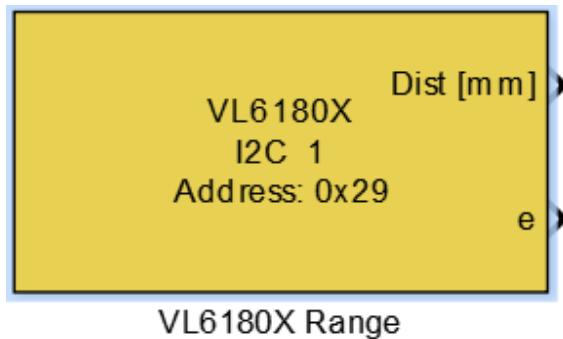


Bild 4.2: Simulinkblock VL6180X

Mask VL6180X

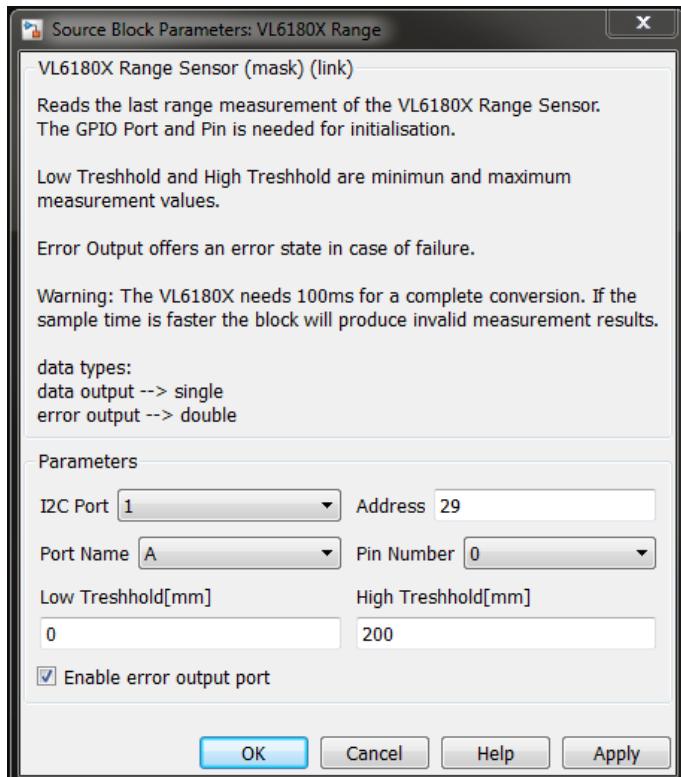


Bild 4.3: Parameterdialog VL6180X

Abbildung 4.3 zeigt den Parameterdialog für den Simulinkblock. In dem Popupfeld „I2C Port“ kann der verwendete I2C Port eingestellt werden. Port 1 - 3 sind auswählbar. Dabei ist der ausgewählte Port nicht zwingend identisch mit dem I2C-Port des Controllers. Um den I2C-Bus in Simulink nutzen zu können wird zusätzlich ein I2C Config Block benötigt. In dem Config Block werden die verwendeten Pins, die Busgeschwindigkeit sowie ein Port für den I2C Bus eingestellt. Der eingestellte I2C Port im Sensorblock muss lediglich mit den Einstellungen im I2C Config Block übereinstimmen.

Im Adress-Feld wird die Adresse des Sensors im I2C-Bus festgelegt welche direkt als hexadezimaler Wert eingetragen wird. Im Block wird die Adresse ebenfalls als hexadezimaler Wert angezeigt.

Mit den beiden Popupfeldern „Port Name“ und „Pin Number“ wird ein zusätzlicher GPIO eingestellt, mit dem der Sensor verbunden ist. Dieser GPIO wird benötigt, um den Sensor richtig zu initialisieren.

Die Felder Low Treshold und High Treshold bieten die Möglichkeit eine untere und eine obere Messwertgrenze einzustellen.

Standardmäßig ist die untere Grenze auf 0 mm eingestellt, die obere Grenze auf die maximale Messdistanz des Sensors von 200 mm.

Mit der Checkbox „Enable error output port“ wird der Error Output „e“ am Block ein- und ausgeschaltet. Dieser Output gibt Fehlermeldungen aus, sollte der Sensor nicht ordnungsgemäß funktionieren. Tabelle 4.1 zeigt die Fehlermeldungen und ihre Bedeutungen.

Error Code	Error	Bedeutung
0	API NO ERROR	API keine Fehler
-1	API ERROR	API undefinierter Fehler
-2	INVALID PARAMETER	ungültige Parameter
-3	NOT SUPPORTED	Funktion im gewählten Modus nicht verfügbar
-4	RANGE ERROR	Abstandsfehler
-5	TIME OUT	Abbruch durch Time Out

Tabelle 4.1: Error Codes VL6180X

Der Mask Editor bietet verschiedene Reiter um alle benötigten Einstellungen vorzunehmen.

Im Reiter „Parameters & Dialog“ des Editors werden die Parameter des Blocks eingetragen. Der Block beinhaltet folgende Parameter:

Type	Prompt	Name	Optionen
popup	I2C Port	i2cbus	1, 2, 3
edit	Adress	SubAdress	-
popup	Port Name	PortName	A, B, C, D, E, F
popup	Pin Number	PinNumber	0 - 15
edit	Low Treshold	lowtreshold	-
edit	High Treshold	hightreshold	-
checkbox	Enable error output port	EnableError	-

Tabelle 4.2: Parameter VL6180X

Der „Type“ legt fest, um was für einen Typ von Parameter es sich handelt. Für diesen Block wurden die Typen popup (Drop-Down Menü), edit (Eingabefeld) und checkbox verwendet. Bei einigen Typen stehen nur feste Optionen zur Auswahl (hier nur popup). Für diese Typen können in dem Feld „Type options“ auswählbare Optionen erstellt werden (siehe Tabelle 4.2). Der „Prompt“ ist der Anzeigenname der in dem Parameterdialog verwendet wird. Der „Name“ ist die Bezeichnung des Parameters. Mit dieser Bezeichnung wird in dem Mask Editor, der S-function und dem TLC-file auf den Parameter zugegriffen.

Im Reiter „Icon & Ports“ werden die Ports und Anzeigen des Blocks konfiguriert.

Listing 4.1 zeigt die Anzeigekonfiguration des Blocks.

```

1 disp(['VL6180X\n\n']);
2 disp(['\nI2C  num2str(i2cbus) '\nAddress: 0x' num2str(SubAddress)]);
3
4 port_label('output', 1, 'Dist [mm]');
5
6 if EnableError
7     port_label('output', 2, 'e');
8 end

```

Listing 4.1: Icon & Ports

Der Befehl „disp“ stellt eine Zeichenfolge dar. Dabei kann auf die aus Programmiersprachen wie C bekannten Formatierungsoptionen zurückgegriffen werden. In Zeile 1 des Listings wird der Name des Blocks dargestellt. In der zweiten Zeile werden mehrere Informationen angezeigt. Zum einen die Nummer des verwendeten I2C-Bus. Diese Information wird aus dem Parameter i2cbus bezogen. Um eine korrekte Anzeige zu gewährleisten, wird der Parameter mit dem Befehl „num2str(i2cbus)“ von einem numeric type in einen character type gewandelt. In dem daneben liegenden edit-Feld kann die Adresse des Sensors eingegeben werden. Auch dieser Parameter wird von einem numeric type in einen character type gewandelt. Der Befehl „port_label('port_type', port_number, 'label')“ in Zeile 4 erzeugt ein Output-Symbol an dem Block. Er wird als output konfiguriert, erhält die Nummer 1 und ist mit Dist [mm] beschriftet. Mit den Zeilen 6-8 wird ein weiteres Output-Symbol angezeigt, wenn in dem Block der Haken bei „Enable error output port“ gesetzt wird.

S-function VL6180X

Im Folgenden werden die wichtigsten Einstellungen in der S-function erläutert. Die gesamte S-function ist im Anhang dieser Arbeit zu finden.

Zu Beginn der S-function muss der Name definiert werden. Dabei ist wichtig, dass dieser genau so eingetragen wird wie die Datei heißt.

```
1 #define S_FUNCTION_NAME          sfunar_VL6180X
```

Mithilfe dieser Funktionen wird die Anzahl der Ausgangsports in Abhängigkeit der „Enable error output port checkbox“ gesetzt. Ist der Error-Port abgewählt, wird die Anzahl der Ausgangsports auf eins gesetzt. Ist der Error-Port angewählt, wird die Anzahl auf zwei gesetzt.

```

1   if(errorOutputEnable)
2   {
3       if (!ssSetNumOutputPorts(S, 2))
4           return;
5   }
6   else
7   {
8       if (!ssSetNumOutputPorts(S, 1))
9           return;
10 }

```

Der Funktion ssSetNumSFcnParams() wird die Anzahl der Parameter übergeben. In dieser S-function wurden 7 Parameter verwendet.

```

1  /* Number of expected parameters */
2  ssSetNumSFcnParams(S, 7);

```

Mit der Funktion ssRegDlgParamAsRunTimeParam() wird ein Dialogparameter als Runtimeparameter sowie der Name und der Datentyp übergeben. Dabei sollten die Parameter in der gleichen Reihenfolge angegeben werden, in der sie auch im Mask Editor eingetragen wurden. Wurden die S-function den Anforderungen entsprechend erstellt muss diese noch kompiliert werden.

```

1  ssRegDlgParamAsRunTimeParam(S, 0, 0, "i2cbus", ssGetDataTypeId(S, "uint8"))
2  ;
3  ssRegDlgParamAsRunTimeParam(S, 1, 1, "SubAddress", ssGetDataTypeId(S, "uint8"));
4  ssRegDlgParamAsRunTimeParam(S, 2, 2, "PortName", ssGetDataTypeId(S, "uint8"));
5  ssRegDlgParamAsRunTimeParam(S, 3, 3, "PinNumber", ssGetDataTypeId(S, "uint8"));
6  ssRegDlgParamAsRunTimeParam(S, 4, 4, "lowthreshold", ssGetDataTypeId(S, "uint8"));
7  ssRegDlgParamAsRunTimeParam(S, 5, 5, "hightreshold", ssGetDataTypeId(S, "uint8"));
8  ssRegDlgParamAsRunTimeParam(S, 6, 6, "EnableError", ssGetDataTypeId(S, "uint8"));

```

TLC-File VL6180X

In dem TLC-File wird gesteuert, welcher Code erzeugt werden soll.

In der ersten Zeile des TLC-Files wird der Name der zugehörigen S-function (sfunar_VL6180X) sowie die verwendete Sprache (C) angegeben.

```
1 %implements sfunar_VL6180X "C"
```

In der Funktion „BlockTypeSetup“ werden die benötigten Source- und Headerfiles eingebunden. Für diesen Block werden die Sensorklasse „vl6180x_class.cpp“, das dazugehörige Headerfile „vl6180x_class.h“ sowie das Headerfile „vl6180x_platform.h“ eingebunden.

```

1 %function BlockTypeSetup(block, system) void
2
3   %% Ensure required header files are included
4   %<MbedCommonBlockTypeSetup(block, system)>
5
6   %<LibAddToCommonIncludes("vl6180x_class.h")>
7   %<LibAddToModelSources("vl6180x_class")>
8   %%<LibAddToCommonIncludes("vl6180x_platform.h")>
9
10 %endfunction

```

In der ersten Zeile des folgenden Codeausschnitts wird ein DigitalInOut Objekt erzeugt. Dadurch wird der angegebene Pin konfiguriert. %<name>_pin bildet den Namen des Pins im erzeugten Quellcode. Die Variable %<name> wird aus dem Projektnamen und dem Blockname gebildet. In Zeile 3 des Ausschnitts wird ein Sensorobjekt erzeugt. Es wird der I2C-Bus sowie der im Block eingestellt GPIO übergeben. Der Name des I2C-Objekts wird aus dem I2C-Config Block bezogen. Dort wird der Bus initialisiert.

```

1     DigitalInOut %<name>_pin(%<pname>, PIN_OUTPUT, OpenDrain, 0);
2
3     VL6180X %<name>(%<i2c_name>, %<name>_pin, NC);

```

Der daraus von Simulink generierte Code sieht wie folgt aus:

```

1 DigitalInOut SensorsTest_VL6180X_Range__pin(PF_0, PIN_OUTPUT, OpenDrain, 0);
2 VL6180X SensorsTest_VL6180X_Range_(i2c1, SensorsTest_VL6180X_Range__pin, NC);

```

Direkt in mbed programmiert wäre eine mögliche Lösung folgender Code:

```

1 DigitalInOut pin(PF_0, PIN_OUTPUT, OpenDrain, 0);
2
3 VL6180X sensor(i2c, pin, NC, 0x29);

```

Wie zu erkennen ist wird durch die automatische Codegenerierung äquivalenter Code erzeugt. Lediglich die Namensgebung ist, gesteuert durch das TLC-File, unterschiedlich.

In Zeile 1 des folgenden Codeausschnitts wird in die Integervariable „newaddress“ der Inhalt des Adressfelds des Konfigurationsdialogs geschrieben. Zeile 3 ruft die Initialisierungsfunktion des Sensors auf.

Dabei wird auch die neue Adresse für den Sensor übergeben. Um den Sensor zu initialisieren, wird der im Block eingestellte GPIO auf einen High-Pegel gezogen, und nach 400 us die Funktion VL6180x_WaitDeviceBooted() aus der von STM für den Sensor bereitgestellten API aufgerufen. [17] In Zeile 5 des Ausschnitts wird die Abstandsmessung mit der Funktion „StartMeasurement“ gestartet. Der Funktion wird der Betriebsmodus des Sensors, eine Call Back function für den Interrupt-Modus sowie eine untere und eine obere Messwertbegrenzung übergeben. Die Messwertbegrenzungen werden aus dem Parameterdialog bezogen.

```

1     uint8_t newaddress = %<subaddress_val>;
2
3     %<name>.Init(&newaddress);
4
5     %<name>.StartMeasurement(range_continuous_polling, NULL, 0, 200);

```

Der von Simulink automatisch generierte Code:

```

1 // Start for S-Function (sfunar_VL6180X): '<Root>/VL6180X Range'
2 uint8_t newaddress = 0x29;
3 Sensortest_VL6180X_Range_.Init(&newaddress);
4 Sensortest_VL6180X_Range_.StartMeasurement(range_continuous_polling, NULL,
5 0, 200);

```

Mögliche Lösung in mbed:

```

1     uint8_t newaddress = 0x29;
2
3     ret=sensor.Init(&newaddress);
4
5     ret=sensor.StartMeasurement(range_continuous_polling, NULL, 0, 200);

```

Die automatische Codegenerierung von Simulink erzeugt äquivalenten Code zu mbed Quellcode.

Um die Messdaten des Sensors auszulesen wird folgende Funktion im TLC-File verwendet:

```
1     %<name>.GetMeasurement(range_continuous_polling, &VL6180X_data);
```

Der Funktion wird nochmal der Betriebsmodus des Sensors, sowie ein Pointer auf eine Struktur vom Typ MeasureData_t in der die Messwerte gespeichert werden, übergeben. Jeder Aufruf der Funktion liest den aktuellen Messwert des Sensors aus.

Der automatisch von Simulink erzeugte Code:

```

1     Sensortest_VL6180X_Range_.GetMeasurement(range_continuous_polling,
2     &VL6180X_data);

```

Eine mögliche Lösung in mbed:

```
1     sensor.GetMeasurement(range_continuous_polling, &data);
```

Erprobung Simulinkblock

Um den Simulinkblock für den ToF-Sensor zu testen, wurde ein einfacher Versuch durchgeführt. Zunächst wurde der PC-Lüfter entfernt und der Tischtennisball wurde an einer Schnur befestigt, die aus dem oberen Ende des Rohrs hing. Der Ball wurde mit der Schnur auf feste Messwerte angehoben und die Messdaten wurden aufgezeichnet. Diese wurden anschließend mithilfe von MATLAB ausgewertet. Um zu prüfen, ob die Daten plausibel sind, wurde an dem Kunststoffrohr eine Skala angebracht. Die Sensordaten wurden mit dem in Abbildung 4.4 dargestellten Simulinkmodell ausgelesen.

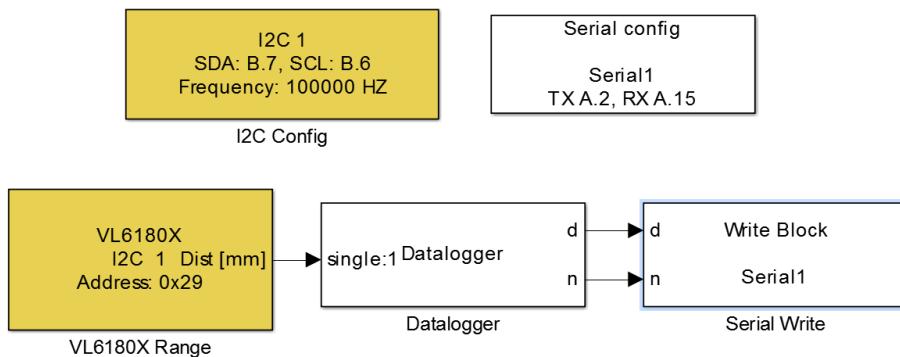


Bild 4.4: Simulinkmodell Sensor test

Der Block „I2C Config“ konfiguriert die Pins sowie die Bus-Frequenz für den I2C-Bus. Im „Serial Config Block“ werden die Pins sowie die Baudrate für die serielle Schnittstelle eingestellt. Der „VL6180X Range Block“ liest die Messwerte des ToF-Sensors aus. Der Block „Datalogger“ wandelt die Zeichen in ASCII-Zeichen. Mit dem „Serial Write Block“ werden die Zeichen über die serielle Schnittstelle an den PC gesendet. Die Auswertung der Daten mit Matlab ergab den in Abbildung 4.5 dargestellten Plot.

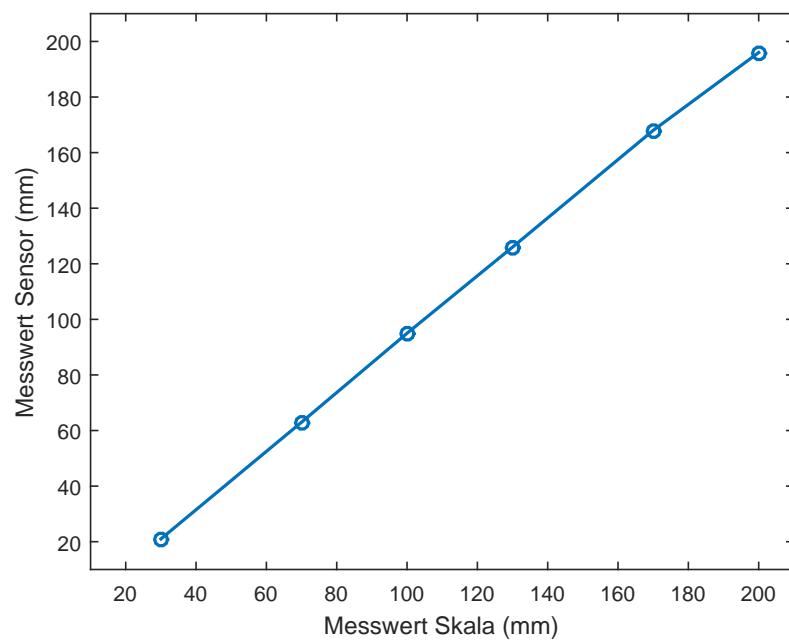


Bild 4.5: Messdaten VL6180X

Die Abbildung zeigt die Messwerte des Sensors dargestellt über die abgelesenen Messwerte an der Skala. Somit konnte bestätigt werden das die Messwerte des Sensors etwa mit den Messwerten der Skala übereinstimmen. Die aufgenommenen Daten sind in Tabelle 4.3 dargestellt.

Messwert Skala (mm)	Messwert Sensor (mm)
30	21
70	63
100	95
130	126
170	168
200	196

Tabelle 4.3: Messdaten

4.2 Regelung

Im folgenden Abschnitt werden verschiedene Reglereinstellverfahren vorgestellt, sowie der Beispielregler für den Laborversuch näher erläutert.

4.2.1 Einstellverfahren für Regler

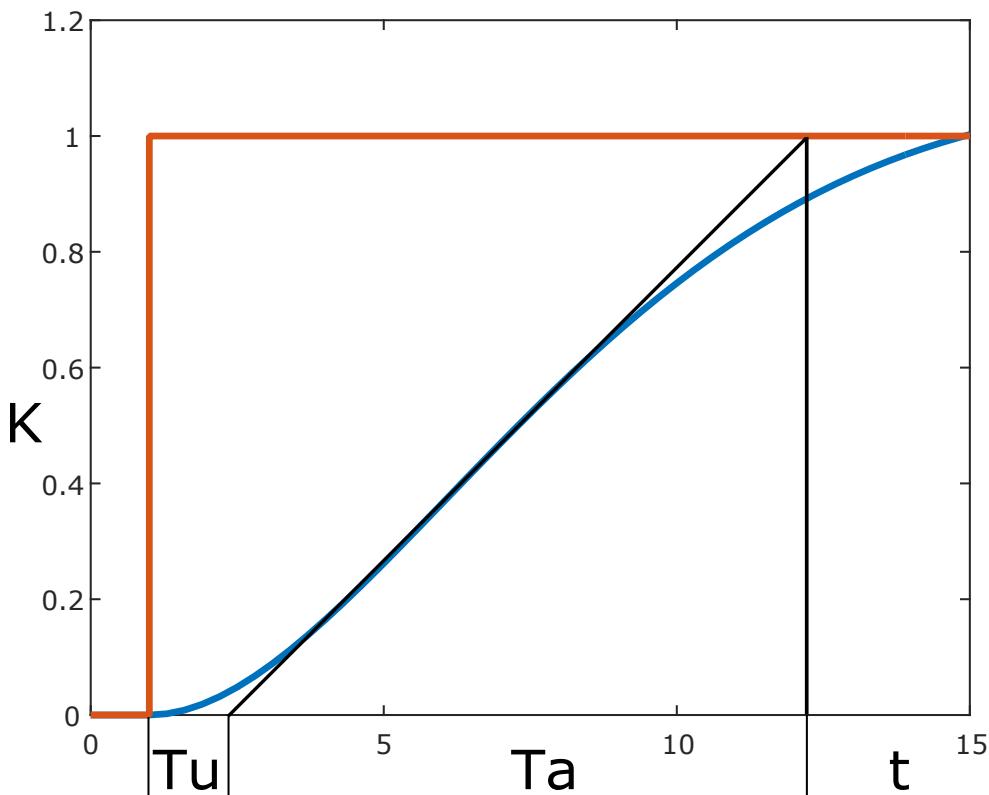
Um lineare Regler wie P-, PD- oder PID-Regler in realen Prozessen einzustellen stehen verschiedene Einstellverfahren zur Verfügung. Die Reglerparameter können unter anderem empirisch ermittelt werden oder mit Einstellverfahren anhand von Experimenten ermittelt werden.

Empirische Dimensionierung

In der Praxis ist es üblich Reglerparameter empirisch zu ermitteln. Dabei wird der verwendete Regler anhand von Erfahrungswerten grob eingestellt, und variiert bis das gewünschte Verhalten erreicht wird.

Reglereinstellung durch Einstellregeln

Es stehen verschiedene Einstellregeln zur Verfügung, um die Reglerparameter linearer Regler zu ermitteln. Dabei werden die Reglerparameter mit Hilfe gemessener Parameter gewonnen. Als Grundlage dient meist die Sprungantwort des Systems oder die Ermittlung der Prozessparameter an der Stabilitätsgrenze. Es gibt drei bekannte Vertreter der Reglereinstellverfahren (Ziegler/Nichols, Chien/Hrones/Rewick und T-Summenregel.) Mithilfe des Wendetangentenverfahrens können aus der Sprungantwort des Systems die Verzugszeit T_u und die Ausgleichszeit T_a bestimmt werden. Abbildung 4.6 zeigt die Ermittlung der Parameter anhand eines Beispiels.

**Bild 4.6:** Sprungantwort

In rot ist der auf das System gegebene Sprung dargestellt, blau ist die Sprungantwort des Systems. Unter Verwendung von Tabellen können aus den angelesenen Parametern mithilfe der Einstellverfahren die Reglerparameter für verschiedene Regler ermittelt werden.

Voraussetzung für die Anwendung dieses Verfahrens ist es, dass es ohne Gefährdung möglich ist, einen Sprung auf das System zu geben. Abbildung 4.7 zeigt eine aufgenommene Sprungantwort des Systems. Diese Sprungantwort wurde mit dem Ball aus Ruheposition aufgenommen. Abbildung 4.8 zeigt eine weitere aufgenommene Sprungantwort des Systems. Diese Sprungantwort wurde mit einem vorher schwebenden Ball aufgenommen.

Aufgrund der Trägheit des System ist dies der korrekte Weg um die Sprungantwort aufzunehmen. Allerdings ergaben die Messungen wie in der Abbildung zu sehen keine verwertbaren Ergebnisse. Aus diesem Grund wurde ein Zweipunktregler entworfen um die Funktion des Versuchsaufbaus zu verdeutlichen.

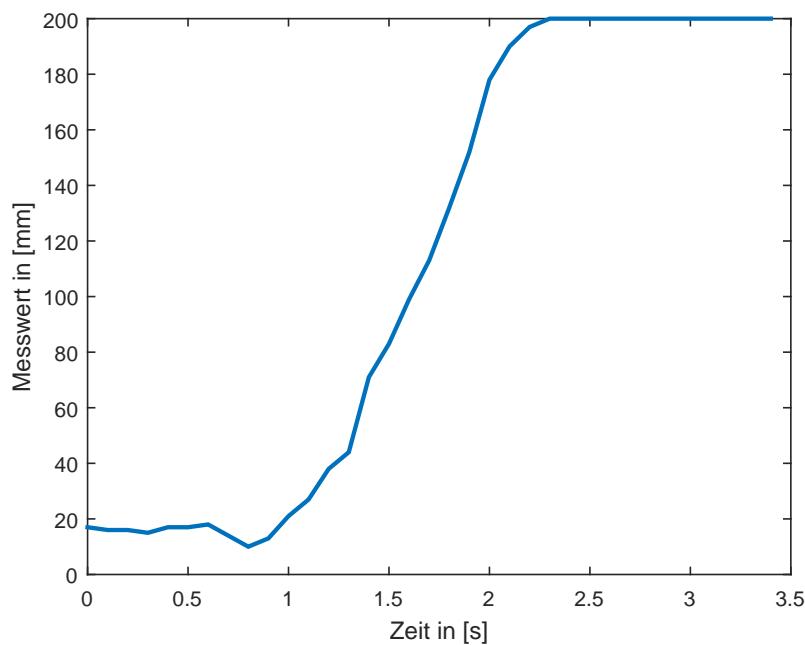


Bild 4.7: Sprungantwort ruhender Ball

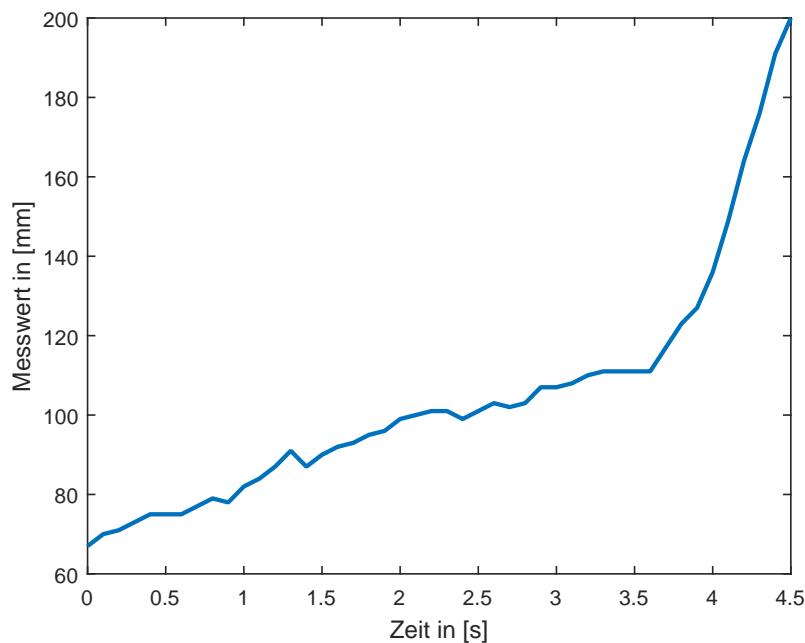


Bild 4.8: Sprungantwort schwebender Ball

T-Summenregel

Die T-Summenregel ist eine Einstellregel für PID-Regler. Die Regel gilt besonders für Strecken mit S-förmiger Sprungantwort.

Als Kenngröße für die Schnelligkeit der Strecke wird die Summenzeitkonstante T_s verwendet. Abbildung 4.9 zeigt die Anwendung der T-Summenregel.

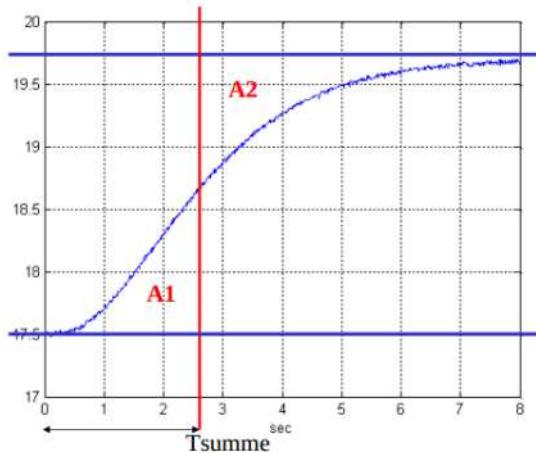


Bild 4.9: Beispiel T-Summenregel [18]

Sind die beiden Flächen A_1 und A_2 gleich groß kann T_s abgelesen werden. Anhand der Streckenverstärkung und der Summenzeitkonstante können die Regelparameter bestimmt werden.

4.2.2 Regelung für den Laborversuch

Um die Funktion des Laborversuchs nachzuweisen wurde ein einfacher Zweipunktregler entworfen. Im Folgenden wird dieser Regler kurz vorgestellt.

Zweipunktregler

Ein Zweipunktregler ist ein Regler mit zwei Ausgangszuständen. Es werden ein oberer und ein unterer Ausgangszustand festgelegt. (zum Beispiel Ein/Aus) Liegt der Istwert über oder unter dem Sollwert wird der entsprechende Ausgangszustand angenommen. Um zu verhindern, dass der Ausgangszustand ständig hin und her schaltet wenn Soll- und Istwert nahezu gleich sind kann eine Schalthysterese eingesetzt werden. Für den Laborversuch wurde mithilfe der grafischen Programmierumgebung Simulink ein solcher Zweipunktregler entworfen. Abbildung 4.10 zeigt die Regelung.

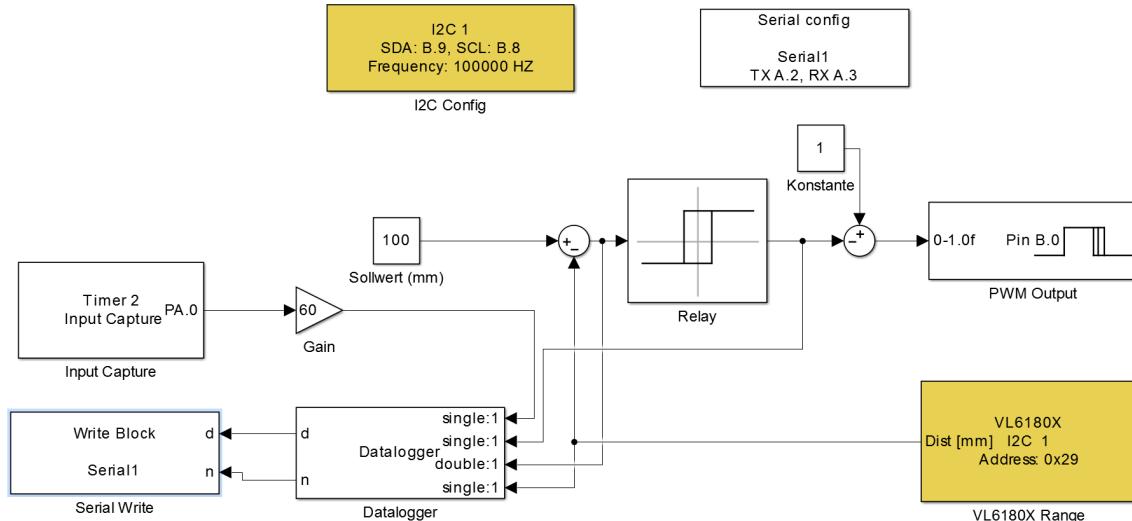


Bild 4.10: Zweipunktregler

Mit der Konstanten „Sollwert“ wird ein Sollwert von 100 mm eingestellt. Der ToF-Sensor wird mithilfe der I2C-Schnittstelle ausgelesen. Zur Konfiguration der Schnittstelle wird der „I2C Config“ benötigt. Um den Sensor auslesen zu können wird zudem der „VL6180X Range“-Block gebraucht. Dieser liest den aktuellen Messwert des Sensors in Millimetern aus. Der „sum“-Block zieht den Istwert (Messwert des Sensors) vom Sollwert ab. Aus diesen Werten wird die Regelabweichung gebildet. Der „Relay“-Block sorgt für die benötigte Zweipunktreglerfunktion. Abbildung 4.11 zeigt die Einstellungen des Blocks.

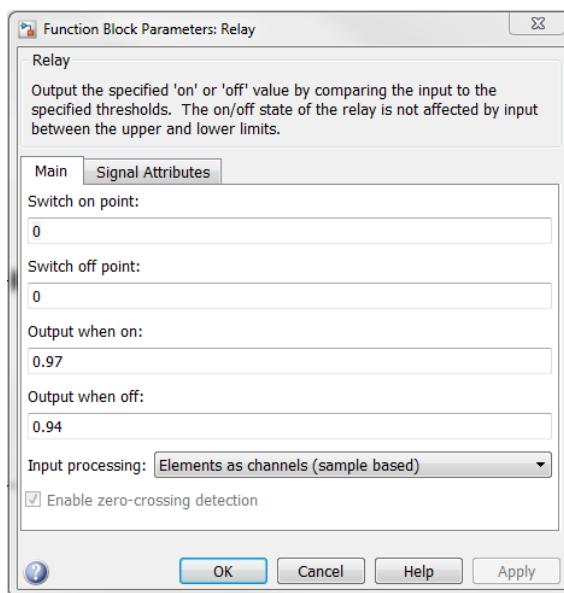
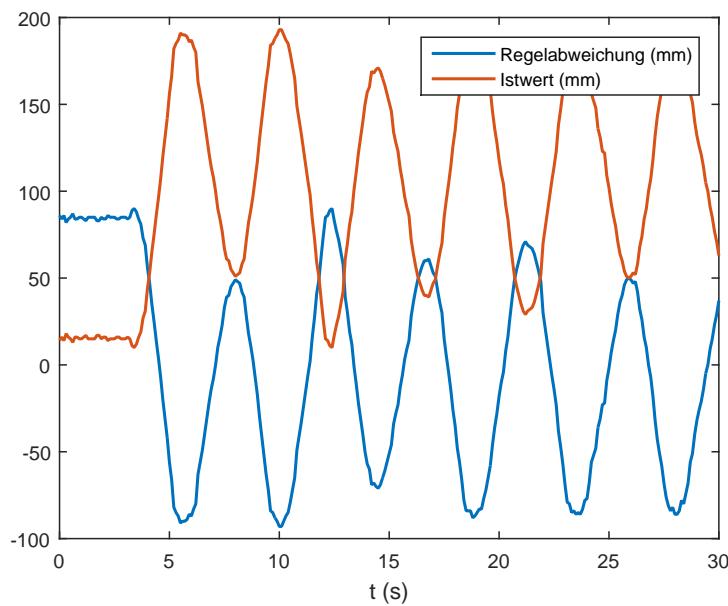
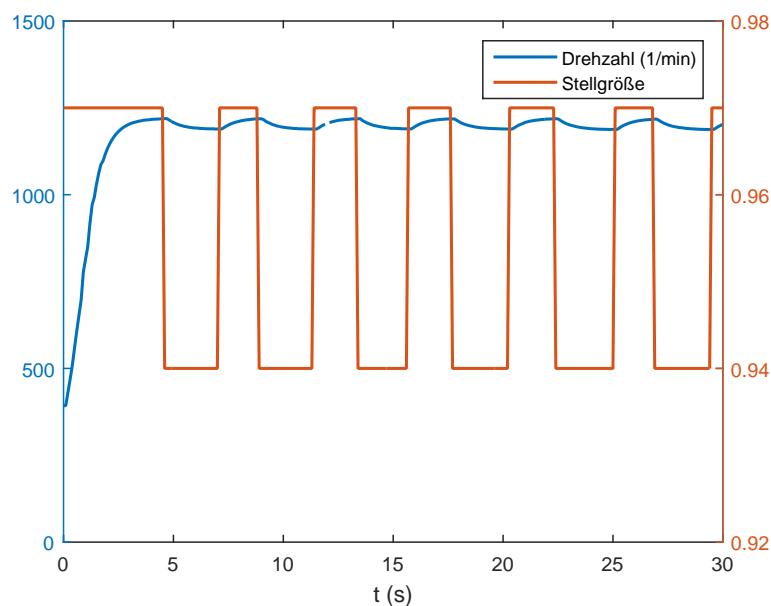


Bild 4.11: Einstellungen Relay-Block

Mithilfe der edit-Felder „Switch on point“ und „Switch off point“ wird eingestellt, bei welcher Regelabweichung der Ausgang des Blocks jeweils den unteren oder den oberen Ausgangszustand annehmen soll. Mit diesen Einstellungen kann also eine Schalthysterese eingestellt werden. Bei diesem Regler wurde auf eine solche Hysterese verzichtet. Mit den beiden edit-Felder „Output when on“ sowie „Output when off“ wird konfiguriert welchen Wert der Ausgangszustand jeweils annehmen soll. In diesem Fall nimmt der Ausgang den Zustand 0,97 ($\hat{=} 97\%$ Lüfterdrehzahl) an wenn der Istwert kleiner dem Sollwert ist. In diesem Fall ist die Position des Balls im Rohr zu tief. Der Ausgang nimmt den Zustand 0,94 ($\hat{=} 94\%$ Lüfterdrehzahl) wenn der Istwert größer dem Sollwert ist. Die Position des Balls im Rohr ist zu hoch. Der nachfolgende „sum“-Block wird benötigt um die Stellgröße zu invertieren. Durch den auf der Basisplatine verbauten Schalttransistor T1 (vergleiche Abbildung 3.6) wird das erzeugte PWM-Signal wiederum invertiert. Der „PWM Output“-Block erzeugt ein PWM-Signal am eingestellten Pin des Microcontrollers (in diesem Fall PB0). Dem Block muss ein floatwert zwischen 0 und 1 vorgegeben werden. Daraus wird ein entsprechendes PWM-Signal gebildet. Mit diesem PWM-Signal wird die Drehzahl des Lüfters gesteuert. Mithilfe des „Input Capter“-Blocks wird das an dem Lüfter nachgerüstete Tachosignal ausgelesen. Der Block ist so konfiguriert, das bei jeder steigenden Flanke ein Input Capture ausgelöst wird. Da pro Umdrehung des Lüfters durch den verbauten Hallsensor in Kombination mit dem vorbeikommenden Magneten eine steigende Flanke erzeugt wird, kann mit dem Input Capture die Zeit zwischen zwei Flanken (also die Zeit für eine Umdrehung) gemessen werden. Der Block kann diesen Wert als Frequenz ausgeben. Der verwendete „Gain“-Block rechnet den Wert in U/min um. Mit dem „Datalogger“-Block und dem „Serial Write“-Block können die Daten über die Serielle Schnittstelle des Microcontrollerboards zum PC gesendet werden. Unter Zuhilfenahme des Terminalprogramms „HTerm Version 0.8.1“ können die Daten am PC dargestellt werden. So kann der aktuelle Messwert, die Regelabweichung, die Drehzahl sowie die Stellgröße während des Versuchsablaufs überwacht werden. Die Stellgröße ist die vom Regler gebildete Größe die dem „PWM Output“-Block zugeführt wird. Abbildung 4.12 stellt den Istwert (Messwert des ToF-Sensor) und die gebildete Regelabweichung dar.

**Bild 4.12:** Istwert und Regelabweichung

Wie in der Abbildung zu erkennen schwankt der Messwert um den eingestellten Sollwert von 100 mm. Dies ist in der Charakteristik des Reglers begründet. Mit einer anderen Regelung sollte eine geringere Regelabweichung erreicht werden können. Abbildung 4.13 zeigt die vom Regler gebildete Stellgröße sowie die aktuelle Drehzahl des Lüfters.

**Bild 4.13:** Stellgröße und Drehzahl

Bei dem verwendeten Zweipunktregler kann die Stellgröße nur die oberen oder die unteren Ausgangsgröße annehmen (hier 0,97 und 0,94). Ist die Ballposition zu tief oder zu hoch, und somit die Regelabweichung größer 0 (vergleiche Abbildung 4.12 schaltet der Zweipunktregler. Ist die Position des Balls tiefer als der vorgegebene Sollwert schaltet der Regler auf 0,97. Somit steigt die Drehzahl des Lüfters und der Ball wird in dem Rohr angehoben. In diesem Fall liegt eine positive Regelabweichung vor. Wenn die Ballposition größer als der Sollwert ist liegt eine negative Regelabweichung vor. Der Zweipunktregler schaltet auf einen Ausgangszustand von 0,94. So sinkt die Drehzahl des Lüfters und der Ball sinkt im Rohr. Im Simulinkmodell wurde für diesen Regler eine Abtastzeit von 100 ms eingestellt um den ToF-Sensor korrekt auslesen zu können.

5 Schlussbemerkungen und Ausblick

Im Rahmen dieser Arbeit wurde dargestellt wie der Laborversuch „schwebende Kugel“ entworfen, konstruiert und programmiert wurde. Zudem wurde die Funktion mit einem Zweipunktregler nachgewiesen.

Die Konstruktion für den Laborversuch wurde mit dem 3D-CAD-Programm Solid Works 2015 erstellt. Somit konnten die Besonderheiten des Fertigungsverfahrens beachtet werden. Die Fertigung des Modells erfolgte im 3D-Druckverfahren. Um eine einfache Durchführung des Laborversuchs sicherzustellen, stehen alle benötigten Schnittstellen zur Verfügung (analoge sowie digitale Schnittstelle). Ebenso nimmt die Konstruktion alle elektrischen Komponenten des Versuchs auf. Für die Versuchsdurchführung muss sich daher nur auf die zu erstellende Regelung konzentriert werden.

Für das verwendete STM32L432 Nucleo-32 Board wurde eine Basisplatine designed und gefertigt. Diese Platine erfüllt verschiedene Aufgaben, die für den Betrieb des Versuchs benötigt werden. Da zum Zeitpunkt dieser Arbeit noch kein Simulink-Target für das STM32L432 Board zur Verfügung stand wurde alternativ auf ein STM32F446 Board ausgewichen. Da dieses Board größer ist, konnte es nicht in den Versuchsaufbau eingebaut werden. Stattdessen wurde das Board mithilfe von Jumperkabeln mit der Basisplatine verbunden.

Zur Programmierung des verbauten ToF-Sensors wurde ein Simulink-Block erstellt. Mithilfe der automatischen Codegenerierung von Simulink kann somit lauffähiger Quellcode für den Microcontroller erstellt werden. Das Augenmerk bei diesem Versuch liegt nicht auf der Programmierung einer Regelung in C oder einer ähnlichen Hochsprache sondern im Entwerfen und Verstehen der benötigten Regelung. Daher ist Simulink als grafisches Programmierwerkzeug einer klassischen Programmierung vorzuziehen. Um die Funktion des Versuchs nachzuweisen wurde ein Zweipunktregler entworfen. Dieser kann den Tischtennisball auf einer eingestellten Höhe halten. Zukünftig sollte die Halterung der Basisplatine im Fuß des Versuchsaufbaus dahingehend angepasst werden, dass die Platine angeschraubt werden kann. So könnte ein sicherer Transport gewährleistet werden.

Eine weitere Verbesserungsmöglichkeit ist der verwendete ToF-Sensor VL6180X von STM. Mit seiner maximalen Messdistanz von 200 mm ist der Sensor nur bedingt für den Versuch geeignet. Durch die Limitierung ist das Kunststoffrohr länger als der Messbereich wodurch der Ball diesen verlassen kann. Während der Bearbeitungszeit dieser Arbeit wurde von STM ein neuer ToF-Sensor veröffentlicht. Dieser Sensor mit der Bezeichnung VL53L0X verfügt über eine Messdistanz von bis zu 2 m. Somit könnte die gesamte Länge des Kunststoffrohrs für den Versuch verwendet werden. Der Sensor verfügt ebenfalls über eine I2C-Schnittstelle. So kann der Sensor in den bestehenden Versuch integriert werden. Es müsste ein angepasster Simulinkblock für die Anwendung erstellt werden. Der Halter im Fuß, der den Sensor aufnimmt müsste ebenfalls leicht angepasst werden, da der neue Sensor in einem kleineren Gehäuse untergebracht ist.

Ebenso steht ein Test des STM32L432-Nucleoboard aus. Dieser sollte erfolgen sobald ein Simulink-Target für das Board zur Verfügung steht um sicherzustellen das auch mit diesem Controller alle Anforderungen erfüllt werden.

Literaturverzeichnis

- [1] *Regelungstechnische Experimente Skript*
- [2] STROPPE, Heribert: *PHYSIK für Studierende der Natur- und Ingenieurwissenschaften*. Hanser, 2011
- [3] https://de.wikipedia.org/wiki/Laserprofilometrie#/media/File:Laserprofilometer_DE.svg
- [4] <http://blog.micro-epsilon.de/methoden-der-wegmessung/laser-triangulation/>
- [5] https://tu-dresden.de/ing/elektrotechnik/ressourcen/dateien/iee/old2_pmp/studium/lehre/MST/dl/Pr/us?lang=de
- [6] http://www.gris.informatik.tu-darmstadt.de/lehre/seminarpraktikum/pm+medicomp/ws0809/slides/01_Ultraschall_Handout.pdf
- [7] <http://www.baumer.com/de-de/services/anwenderwissen/ultraschall-sensoren/funktionsweise/>
- [8] <http://www.chemie.de/lexikon/Brechzahl.html>
- [9] <https://de.wikipedia.org/wiki/TOF-Kamera>
- [10] [https://de.wikipedia.org/wiki/Abstandsmessung_\(optisch\)](https://de.wikipedia.org/wiki/Abstandsmessung_(optisch))
- [11] <http://noctua.at/de/products/fan>
- [12] http://www.infineon.com/dgdl/Infineon-TLE49X5L-DS-v01_05-en%5B2%.5D.pdf?fileId=db3a30431f848401011fbc64a85f6363
- [13] http://www.st.com/content/st_com/en/products/microcontrollers/stm32-32-bit-arm-cortex-mcus/stm3214-series/stm3214x2/stm321432kc.html

- [14] http://www.st.com/content/st_com/en/products/microcontrollers/stm32-32-bit-arm-cortex-mcus/stm32f3-series/stm32f303/stm32f303k8.html
- [15] http://www.mouser.com/ds/2/395/ES1AL%20SERIES_K15-884039.pdf
- [16] <http://bit.ly/2eHbyHR>
- [17] <http://www.st.com/content/ccc/resource/technical/document/datasheet/c4/11/28/86/e6/26/44/b3/DM00112632.pdf/files/DM00112632.pdf/jcr:content/translations/en.DM00112632.pdf>
- [18] DÜNOW, Prof. Dr. P.: *Grundlagen der Regelungstechnik 1; Vorlesungsskript*

Abbildungsverzeichnis

1.1	Versuchsaufbau	6
2.1	Lasertriangulation [3]	10
2.2	schematische Darstellung ToF-Messung	11
3.1	3D-Modell	14
3.2	Stecker Lüfter	17
3.3	Hallsensor TLE4905L	17
3.4	Beschaltung TLE4905L	18
3.5	erzeugtes Tachosignal	19
3.6	Schaltplan Basisplatine	21
3.7	PWM Signal	21
3.8	Prototyp des Versuchsaufbaus	23
4.1	Mask Editor	25
4.2	Simulinkblock VL6180X	26
4.3	Parameterdialog VL6180X	27
4.4	Simulinkmodell Sensortest	33
4.5	Messdaten VL6180X	34
4.6	Sprungantwort	36
4.7	Sprungantwort ruhender Ball	37
4.8	Sprungantwort schwebender Ball	37
4.9	Beispiel T-Summenregel [18]	38
4.10	Zweipunktregler	39
4.11	Einstellungen Relay-Block	39
4.12	Istwert und Regelabweichung	41
4.13	Stellgröße und Drehzahl	41

Tabellenverzeichnis

2.1	Formelzeichen	8
2.2	Brechungsindex [8]	12
3.1	Daten NOCTUA NF-A14 industrialPPC-3000 PWM [11]	16
3.2	Daten NOCTUA NF-A14 PWM [11]	16
3.3	Pinbelegung TLE4905L [12]	17
3.4	Daten TLE4905L [12]	18
4.1	Error Codes VL6180X	28
4.2	Parameter VL6180X	28
4.3	Messdaten	34

A Abkürzungsverzeichnis

ADC analog-to-digital converter

CAD computer-aided design

CAN Controller Area Network

DAC digital-to-analog converter

GHz Gigahertz

GPIO General Purpose Input/Output

I2C Inter-Integrated Circuit

IC Integrierter Schaltkreis

kB Kilobyte

kHz Kilohertz

MHz Megahertz

PWM Pulsweitenmodulation

RTC real-time clock

SPI Serial Peripheral Interface

SRAM Static random-access memory

STM STMicroelectronics

TLC Target Language Compiler

ToF Time of Flight

U/min Umdrehungen pro Minute

USART Universal Synchronous/Asynchronous Receiver Transmitter

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die hier vorliegende Arbeit selbstständig, ohne unerlaubte fremde Hilfe und nur unter Verwendung der aufgeführten Hilfsmittel angefertigt habe.

Ort, Datum

Unterschrift